TÉCNICO LISBOA

Multi-speaker TTS with Deep Learning

Ivan Volossovitch Carapinha

Thesis to obtain the Master of Science Degree in

Electrical and Computer Engineering

Supervisor(s): Prof. Isabel Maria Martins Trancoso

Examination Committee

Chairperson: Prof. João Fernando Cardoso Silva Sequeira
Supervisor: Prof. Isabel Maria Martins Trancoso
Member of the Committee: Dr. Xavier Anguera

December 2020

Declaration

I declare that this document is an original work of my own authorship and that it fulfills all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa.

Acknowledgments

First of all, I would first like to thank my supervisor, Professor Isabel Trancoso, for sharing her knowledge, for the tireless guidance and willingness to help solving various challenges that arose during the course of this dissertation. Furthermore, I would like to extend my gratitude to Sérgio Paulo for the patience and readiness to answer all my questions, and for providing me with tools that had a key role in this work. Also, I would like to thank Wang Ling for all the suggestions and feedback on a report preliminary to this thesis.

I would like to thank all Professors and students from the speechMEET research group at INESC-ID for sharing their insights, for giving me the opportunity to learn about state-of-the-art methodologies within the field of Speech Processing, and for always providing me with valuable feedback when I needed.

I cannot fail to express a profound gratitude to my mother, for playing two roles simultaneously: as a Professor, for giving me helpful advice and perspectives for my academic work; and as a mom, for supporting me and showing me the right directions when the path was unclear. I thank my father for sharing his immense knowledge concerning the most variate topics, and for inciting me to be curious and eager to learn.

I would also like to thank my friends Francisco, Edgar, André and Gonçalo for the interesting and entertaining conversations, and for always being available to share a good laugh.

Finally, I would like to thank my entire family for all the joyful moments throughout this journey, and for always reassuring me of the true priorities in life.

Resumo

A recente evolução tecnológica contribuiu para um desenvolvimento considerável da área de Síntese da Fala. Os sistemas de síntese atuais produzem fala em tempo cada vez mais reduzido e para diversas vozes. O presente estudo desenvolveu um sistema de texto para fala (em inglês, TTS) para português europeu, que permite incorporar novas vozes sem necessitar de um conjunto de dados extenso e um processo de treino exaustivo. A estrutura do modelo proposto contempla dois sistemas: um sistema regressivo *sequence-to-sequence* (Seq2Seq) que produz representações acústicas a partir de texto, seguido de um *vocoder* neuronal, destinado à geração de áudio a partir de representações acústicas. O modelo proposto emprega um *vocoder* universal que não carece de *fine-tuning* perante a adição de novas vozes.

O modelo regressivo Seq2Seq gera representações acústicas na forma de Mel-espetrogramas. Este processo decorre da descodificação da combinação de representações linguísticas (*linguistic embeddings*), extraídas a partir de texto, e representações da identidade de voz (*speaker embeddings*). O modelo regressivo opera para várias vozes e permite *fine-tuning* para múltiplas vozes novas simultaneamente.

Os testes subjetivos demonstraram que o modelo proposto registou um desempenho comparável ao de outro sistema TTS estado-da-arte, empregando menos de metade dos dados para treino. Além disso, o sistema proposto gerou resultados relevantes quando treinado com um conjunto de dados reduzido — menos de 3 minutos de fala. Por último, o *vocoder* universal teve um desempenho, em média, 11 vezes mais rápido que o *vocoder* neuronal empregue no sistema TTS estado-da-arte utilizado para comparação.

Palavras-chave: Síntese de Fala, Multi-Speaker TTS, Conversão de Voz, Clonagem de Voz.

Abstract

Recent advancements in technology have allowed for great development in the field of Speech Synthesis. As such, present-day speech synthesis applications are expected to function for multiple voices, and ensure a fast generation of natural–sounding synthetic speech for enhanced feasibility. This study suggests a multi-speaker text-to-speech (TTS) system for European Portuguese that enables the addition of new speakers without requiring extensive training and data. The proposed model framework comprises two systems: a sequence-to-sequence (Seq2Seq) regressive stage for acoustic feature prediction, followed by a neural vocoder for waveform generation. The model employs a universal vocoder which does not require fine-tuning for new voices.

The Seq2Seq regressive model predicts acoustic features in the form of Mel-spectrograms by decoding the combination of linguistic embeddings — extracted from the text input —, and speaker embeddings conveying the target speaker identity. The model operates in a multi-speaker setting and can be fine-tuned simultaneously to multiple unseen speakers.

Subjective tests have shown that the proposed model registered comparable performance to another state-of-the-art TTS system, while employing less than half of training data. Furthermore, the proposed model was capable of producing meaningful results when trained with reduced data — under three minutes of speech. At last, the universal vocoder performed, on average, 11 times faster than the speaker–dependent neural vocoder of the state-of-the-art TTS approach used for comparison.

Keywords: Speech Synthesis, Multi-Speaker TTS, Voice Conversion, Voice Cloning.

Contents

Acknowledgments v
Resumo vii
Abstract ix
List of Tables xv
List of Figures xvii
Glossary xix

1 Introduction **1**
1.1 Objectives 2
1.2 Dissertation Outline 2

2 Background **5**
2.1 Neural networks 5
2.2 Speech Synthesis 6
2.2.1 Voice Conversion 8
2.2.1.1 Speaker embeddings 9
2.2.2 Voice Cloning 10
2.2.3 Evaluation metrics 11
2.2.3.1 Mean Opinion Score 11
2.2.3.2 Same/Different paradigm 11
2.2.3.3 AB/ABX preference test 12
2.2.3.4 MUSHRA 12
2.3 Neural vocoding 12
2.3.1 WaveNet 12
2.3.1.1 Architecture 14
2.3.1.2 Evaluation 14
2.3.2 Parallel WaveNet 15
2.3.2.1 Probability Density Distillation 15
2.3.2.2 Evaluation 15
2.3.3 WaveRNN 16
2.3.3.1 Evaluation 17

2.3.4 Universal vocoding . . . 17
2.3.4.1 Experiments and evaluation . . . 17
2.4 Sequence-to-Sequence TTS systems . . . 19
2.4.1 Tacotron . . . 19
2.4.1.1 Architecture . . . 20
2.4.1.2 Evaluation . . . 21
2.4.2 Tacotron 2 . . . 21
2.4.2.1 Architecture . . . 21
2.4.2.2 Evaluation . . . 22
2.4.3 Multi-speaker generative model . . . 22
2.5 TTS for European Portuguese . . . 24
2.6 Final remarks . . . 24

3 Proposed model **27**
3.1 Model overview . . . 27
3.2 Spoken language corpora . . . 29
3.3 Text preprocessing . . . 30
3.3.1 Punctuation . . . 30
3.3.2 Phonemizer with eSpeak backend . . . 31
3.3.2.1 Phoneme list reduction . . . 32
3.3.3 Sequence-to-Sequence GtoP toolkit . . . 34
3.3.4 Festival-based GtoP . . . 34
3.3.5 Non-standard words and homographs . . . 35
3.4 Audio preprocessing . . . 35
3.4.1 Leading and trailing silence removal . . . 36
3.5 Seq2Seq regressive model — pre-train implementation . . . 37
3.5.1 Data selection process . . . 38
3.5.2 Hyperparameters . . . 38
3.5.2.1 Experiment parameters . . . 38
3.5.2.2 Data parameters . . . 39
3.5.2.3 Training parameters . . . 40
3.5.3 Pre-training process . . . 40
3.5.3.1 Encoder-decoder alignment plot . . . 41
3.6 Neural vocoder — implementation . . . 42
3.6.1 Model training . . . 42
3.7 Final remarks . . . 43

4 Experiments **45**
4.1 Experimental setup . . . 45
4.1.1 Differences in training data: mean pitch and prosody . . . 46

4.1.2 Seq2Seq regressive model — fine-tune implementation 48
4.1.2.1 Adult–speaker configurations 49
4.1.2.2 OOD configurations 50
4.1.3 Speaker identity discrimination 53
4.1.3.1 Adult speakers: *sp_01* and *sp_02* 53
4.1.3.2 Adolescent speakers: *sp_03* and *sp_04* 53
4.1.3.3 Child speakers: *sp_11* and *sp_36* 55
4.1.4 Synthetic speech for OOD configurations 55
4.2 Evaluation 56
4.2.1 Naturalness and similarity 56
4.2.1.1 Results 57
4.2.2 Intelligibility 59
4.2.2.1 Results 59
4.2.3 Synthesis speed 59
4.3 Summary 60

5 Conclusions **61**
5.1 Future work 62

References **65**

A Fine-tune hyperparameters **A.1**

B Semantically unpredictable sentences **B.1**

List of Tables

2.1 5-scale MOS evaluation for previous state-of-the-art systems (LSTM-RNN parametric and HMM-driven concatenative) . . . 15
2.2 Parallel WaveNet – Evaluation of "teacher" model and "student" model . . . 16
2.3 WaveRNN MOS results . . . 17
2.4 5-scale MOS evaluation - Tacotron . . . 21
2.5 5-scale MOS evaluation of Tacotron 2 . . . 22

3.1 Open-source repositories . . . 28
3.2 Voice corpora used as reference for pre-train and fine-tune stages of the regressive Seq2Seq model . . . 30
3.3 List of all the phonemes that are involved in the phoneme list reduction . . . 33
3.4 Occurrences of suprasegmentals . . . 33
3.5 Phoneme replacement/removal . . . 33
3.6 DSP parameters . . . 36
3.7 Multi-speaker Seq2Seq pre-train stage hyperparameters . . . 39
3.8 Neural vocoder hyperparameters . . . 43

4.1 Amount of fine-tuning data per configuration . . . 46
4.2 Pitch values for EP speakers in different age groups . . . 46
4.3 Type of sentences for EP corpora . . . 48
4.4 Intelligibility test results . . . 59
4.5 Frequently mistaken words . . . 59

A.1 Multi-speaker Seq2Seq fine-tune stage hyperparameters . . . A.1

B.1 Intelligibility test sentences – speaker *sp_01* . . . B.1
B.2 Intelligibility test sentences – speaker *sp_02* . . . B.1

List of Figures

2.1 Stages in a TTS pipeline 6
2.2 A speech synthesis framework based on a DNN 8
2.3 Example of a speaker encoder architecture 9
2.4 Speaker adaptation and speaker encoding techniques for voice cloning 11
2.5 Stack of causal convolutional layers 13
2.6 Stack of dilated causal convolutional layers 13
2.7 WaveNet model architecture 14
2.8 WaveRNN architecture 17
2.9 WaveRNN-based Universal Vocoder architecture 18
2.10 Tacotron model architecture 20
2.11 Architecture of a Seq2Seq multi-speaker generative model 23

3.1 Inference-time pipeline of the proposed model 28
3.2 Predicted and original Mel-spectrograms for the same test sentence 41
3.3 Encoder-decoder alignment plots at different checkpoints 42

4.1 Histograms of pitch mean values of speakers in the BD-PUBLICO corpus 47
4.2 Histograms of pitch SD values of speakers in the BD-PUBLICO corpus 47
4.3 Encoder-decoder alignments for fine-tuning test sentences — adult speakers 49
4.4 Mel-spectrograms for fine-tuning test sentences — adult speakers 50
4.5 Encoder-decoder alignments for fine-tuning test sentences — adolescent speakers 51
4.6 Mel-spectrograms for fine-tuning test sentences — adolescent speakers 51
4.7 Encoder-decoder alignments for fine-tuning test sentences — child speakers 52
4.8 Mel-spectrograms for fine-tuning test sentences — child speakers 52
4.9 Visualization of speaker embeddings 54
4.10 Original speech signals and synthesized signals from the Mel-spectrograms of original speech 56
4.11 Naturalness AB preference test results 57
4.12 Similarity ABX preference test results 58

Glossary

CNN Convolutional Neural Network

DL Deep Learning

DNN Deep Neural Network

EN English (language)

EP European Portuguese (language)

GAN Generative Adversarial Network

GRU Gated Recurrent Unit

HMM Hidden Markov Model

IAF Inverse Autoregressive Flow

KLD Kullback-Leibler divergence

LSTM Long short-term memory

MOS Mean Opinion Score

MUSHRA MUltiple Stimuli with Hidden Reference and Anchor

OOD Out-of-domain

RNN Recurrent Neural Network

SD Standard deviation

Seq2Seq Sequence-to-Sequence

SPSS Statistical parametric speech synthesis

TTS Text-to-Speech

VC Voice Conversion

WER Word error rate

Chapter 1

Introduction

Speech is the most natural and immediate form of communication. Nowadays, Speech Synthesis is broadly used in many applications, ranging from voice assistants to speaking aid systems for vocally handicapped people. The demand for speech synthesis in a wide variety of applications propels the development of new approaches that revolutionize the way we use and perceive technology.

The dynamic nature of human speech due to characteristics like language, intonation, vocabulary or accent, poses a demanding challenge for Speech Synthesis systems. Retaining all this information and transforming it such that it can be interpreted by a machine, motivated the conception of many methods that seek to address this subject. The new ability to gather a vast amount of data enabled great progress in speech synthesis applications.

The latest research on Speech Synthesis has developed systems that can generate high-quality natural–sounding speech, and address several voice conversion tasks. However, for the sake of synthetic speech quality, these systems are often trained for very particular configurations, posing several limitations. Most of these applications only operate for a restricted variety of voices and speaking styles. Moreover, these systems are usually configured for data that must attend to very specific requirements in terms of recording conditions and audio quality. Naturally, the more restrictions are raised, the harder is to find data meeting such conditions. For the Portuguese language this problem is even more accentuated as the diversity of speech corpora is considerably smaller than for the English language.

Present synthesis systems are set apart from its predecessors for two essential reasons: 1) they generate synthetic speech of distinctively superior quality; 2) they comprise a great amount of parameters, and employ very large datasets for training. Besides the drawbacks identified in the previous paragraphs, other downsides arise from the complexity of these models: training can be an intricate and exhaustive process, involving the tuning of multiple parameters; synthesis speed at inference time can be slow, often requiring high computational power.

Speech Synthesis is a broad scientific field in constant evolution. Thus, it would not be reasonable to address all the challenges regarding this subject, namely the problems identified in the previous two paragraphs. Within Speech Synthesis, this study focuses on solving the limitations that are pivotal to the performance of text-to-speech (TTS) techniques — that is, systems that generate a synthetic utterance

from a given text — in a real-world context: the limited variety of voices for synthesis, and the often slow inference speed of present-day systems.

1.1 Objectives

The fundamental objective of this study consists in developing a TTS system for European Portuguese (EP), based on existing state-of-the-art implementations of Speech Synthesis systems. To attain this goal, it is first necessary to address the characteristics inherent to EP, in order to ensure correct pronunciation, specially in exceptional cases such as homographs. Moreover, one must identify the main components within the TTS framework and arrange them so the system can adapt to new voices, and ensure reasonable inference speed.

More specifically, this study aims to:

- Adapt a TTS framework to EP, particularly regarding the pronunciation in exceptional cases, namely homographs;
- Incorporate new speaker identities without requiring extensive training;
- Ensure faster synthesis of speech than previously proposed models for EP.

1.2 Dissertation Outline

The thesis comprises five chapters. The first chapter introduces the topic of this study. It starts by pointing out the pertinence of Speech Synthesis and its domains of applicability. Then, it identifies the limitations of present synthesis systems, in particular of TTS. Based on these, it presents the objectives proposed to achieve with the conclusion of this thesis.

The second chapter presents the fundamental concepts for the realization of this study, as well as the state-of-the-art methodologies that take part in the traditional TTS pipeline. It first defines neural networks — the underlying structures of current methods, and characterizes the present-day Speech Synthesis systems in context of previous generation techniques. It also presents specific state-of-the-art implementations that are applied within TTS systems, including those employed for the proposed model. Additionally, this chapter provides an overview of existing TTS systems for EP.

The third chapter describes the proposed model. It starts by defining the structure of the system and its components. Then, it focuses on important aspects regarding the training procedure of the system, namely the nature of data that is used, and the necessary data–conditioning stages upon training. The last section in this chapter details the key aspects of the training procedures for each component in the system, as well the assessment methods to evaluate the training processes' convergence.

The fourth chapter specifies the experiments that were performed regarding the addition of new speakers to the model, as well as the evaluation procedures that were carried out to assess the performance of the system in terms of naturalness, voice similarity, intelligibility and synthesis speed.

The fifth and final chapter of this dissertation presents the conclusions of this study in context of the proposed objectives, as well as suggestions for future work within the topic.

Chapter 2

Background

The main goal of this chapter is to review the state-of-the-art of Speech Synthesis. It is structured as follows: Section 2.1 revises the basic concepts of neural networks; Section 2.2 provides an overview of Speech Synthesis and introduces other related topics, namely voice conversion, voice cloning and the evaluation metrics for synthetic speech (see subsections 2.2.1, 2.2.2 and 2.2.3, respectively). Sections 2.3 and 2.4 present the essential elements of a state-of-the-art TTS framework: a neural vocoder and a sequence-to-sequence model for acoustic feature prediction. Furthermore, these sections exemplify some of the prominent and more recent implementations of these systems, describing their rationale and structure. Finally, Section 2.5 introduces some of the approaches used in the past for TTS in European Portuguese.

2.1 Neural networks

Neural networks are systems capable of performing classification and regression tasks when given input data. This concept, inspired by the structure and function of the brain, is composed of neurons, connections, weights, and activation functions.

Neurons are nodes that generate an output from a combination of received inputs. Each neuron has an activation function. The output of a neuron corresponds to the value of its activation function $f(s)$, where s denotes the combination of its inputs. Connections are weighted links between the output of one neuron and the input of another neuron. Neurons may have multiple input and output connections.

The activation function computes the output of a neuron from its inputs, which are combined as a weighted sum of each input. The weight assigned to each input is the weight of the corresponding connection. A bias term can be added to the sum. As such, for n inputs, the combination of inputs s of a neuron is given by equation (2.1),

$$s = \sum_{i=1}^{n} (x_i w_i) + b \tag{2.1}$$

where x_i, w_i and b denote input i, weight of connection i and the bias term, respectively. Depending on the nature of the problem to be solved, adequate activation functions must be chosen. For nonlinear

problems, nonlinear activation functions are appropriate. The sigmoid function, hyperbolic tangent and rectifier linear unit are some examples of popular activation functions.

Neural networks group neurons in three types of layers: the input layer, hidden layers, and the output layer. The input layer, as the name states, receives the input data of the neural network. Hidden layers perform intermediate processing. The output layer produces the output of the neural network according to the desired format.

A network should be adapted to the characteristics of input data and the type of problem to be solved. This implies adjusting parameters such as connection weights and learning rate. Connection weights can be adjusted through back-propagation [1], which intends to minimize the loss function by using gradient descent to adjust the weights of the network.

Over the years, the increase of computational power motivated the emergence of deep neural networks (DNNs) and other architectures, namely recurrent neural networks (RNNs) and convolutional neural networks (CNNs), that are able to detect more meaningful dependencies in the input data than the traditional multilayer perceptron. When referring to DNNs, the concept of depth is related to the number of hidden layers in the network, so the more layers are added, the deeper the network is.

2.2 Speech Synthesis

Speech synthesis aims to generate synthetic speech acceptable to human listeners. It can take in either textual or conceptual input to reproduce the characteristics of the typical human speaking process [2]. Synthesis from text, also known as text-to-speech (TTS), converts written text to a speech signal. TTS essentially consists of three stages, illustrated in figure 2.1: 1) text analysis; 2) regression; and 3) waveform generation [3]. Text analysis, also known as “frontend”, is responsible for processing text inputs, and extracting the corresponding linguistic representations. The regression stage performs linguistic to acoustic feature mapping. Finally, the waveform generation stage produces a speech signal from the acoustic features previously generated. This stage defines the synthesis technique employed by the system (TTS techniques are described in the following paragraphs). From all the types of acoustic features that exist, Mel-spectrograms are the most popular choice for current TTS systems. A Mel-frequency spectrogram is related to the linear-frequency spectrogram, i.e., the short-time Fourier transform (STFT) magnitude. It is obtained by applying a nonlinear transform to the frequency axis of the STFT, inspired by measured responses from the human auditory system, and summarizes the frequency content with fewer dimensions.

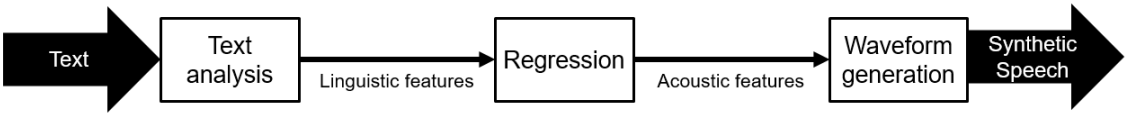


Figure 2.1: The stages in a TTS pipeline. Adapted from [3].

The main TTS synthesis techniques developed in the past are the following: articulatory synthesis, formant synthesis, concatenative synthesis, and statistical parametric synthesis. More recently, Deep Learning (DL) has also made a profound impact on speech synthesis, being the current state-of-the-art approach.

Articulatory synthesis is based on the physical characteristics of the human speech production system, and its purpose is to simulate the acoustic functions and dynamic motion of the vocal tract. The model of the vocal and nasal tracts is determined by the position of phonatory organs. Based on the obtained model, the synthesizer simulates the airflow through the vocal tract. This approach, however, has not led to good quality speech synthesis due to inaccurate vocal tract model representations [4].

Formant synthesis is a rule-based method which focuses on representing the resonant frequencies of the vocal (and nasal) tract. This strategy aims to generate source signals, and feed these through a vocal tract model to produce synthetic speech. Similarly to articulatory synthesis, this technique produces unnatural speech given the limitations of the source and vocal tract models [4]. Concatenative synthesis consists in synthesizing speech by concatenating short samples (units) of recorded speech. In most cases, as long as a large dataset of speech units is provided, the synthetic speech preserves naturalness and intelligibility [5]. Nevertheless, deviations between natural sound transitions of speech and inaccuracies in the waveform segmenting process may originate audible glitches in synthesized speech [4].

In a very simplistic approach, statistical parametric synthesis can be described as generating the average of some sets of similarly sounding speech segments [6]. Typically, a statistical parametric speech synthesis (SPSS) system comprises the extraction of speech parameters followed by their statistical parametric model representation. The speech parameters, namely the spectrum, fundamental frequency (F_0), and phoneme duration, are estimated by the maximum likelihood criterion and represented by statistics, such as means and variances of probability density functions [7]. HMM-based synthesis, one of the most widely used approaches for SPSS, performs the maximum likelihood estimation of the model parameters by using the Expectation-Maximization (EM) algorithm. Then, a speech waveform is synthesized from the estimated parameters of speech [5, 6]. Speech produced by an SPSS system is usually smooth and resilient to voice changes. Nevertheless, it has some drawbacks because it usually sounds less natural than concatenative synthesized speech, and may sound muffled in case over-smoothing occurs during the synthesis process.

Contrarily to previous approaches, Deep Learning (DL) synthesis techniques process large amounts of data, allowing to extract more intricate features from raw inputs. This is particularly useful to tackle the limitations of the previous models [9], such as the lack of naturalness in speech produced by conventional SPSS systems. DL synthesis is mostly based on DNNs, CNNs, and sequence-to-sequence (Seq2Seq) neural networks, as presented in the upcoming sections 2.3 and 2.4.

Figure 2.2 depicts a possible framework for DNN-based speech synthesis. Input features $\{x_n^t\}$, where x_n^t denotes the n-th input feature at frame t, are extracted after preprocessing of the text to be synthesized. Output features $\{y_n^t\}$ are computed by a trained DNN from the input features. Speech parameters are obtained from the output features. Finally, the waveform synthesis module generates

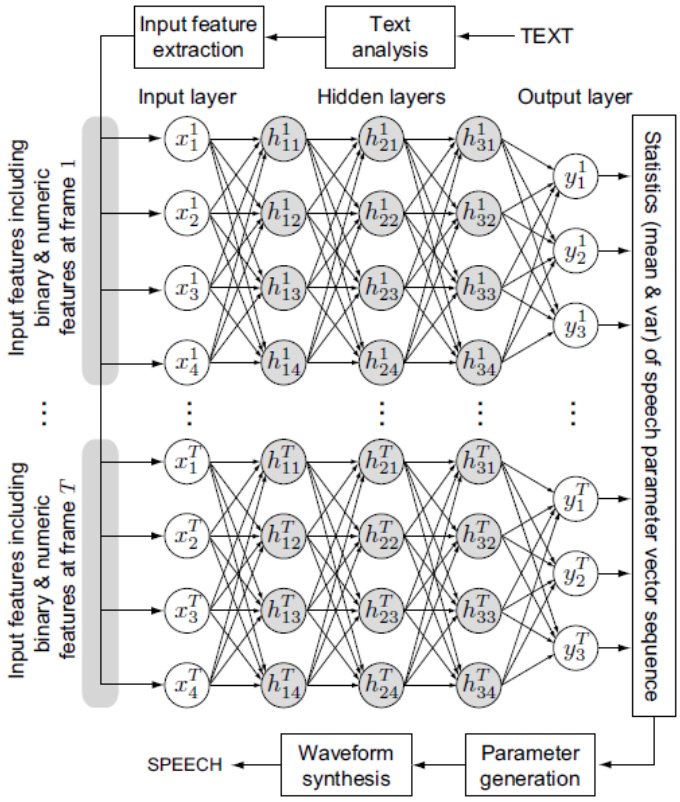


Figure 2.2: A speech synthesis framework based on a DNN. Extracted from [8].

the intended speech waveform [8].

In the scope of TTS, Seq2Seq neural networks (also known as encoder-decoder neural networks) are currently one the most effective approaches for linguistic to acoustic feature sequence mapping. Based on recurrent mechanisms, these networks suit well the sequential nature of speech signals, by converting variable-length inputs into fixed-length outputs, while retaining the meaningful temporal dependencies [10]. Besides TTS, Seq2Seq networks have been used for other tasks involving sequential data, such as machine translation and speech recognition [9]. In DL synthesis, the TTS pipeline stages comprise two essential blocks: 1) a Seq2Seq system, which implements the text analysis[1] and regression stages; and 2) a neural vocoder for waveform generation. CNN-based architectures can be an alternative to models that heavily rely on recurrent-based mechanisms, demanding high computational power. In these cases, the use of CNN-based networks may enable a faster training process, while capturing long-term dependencies successfully [9].

2.2.1 Voice Conversion

Voice conversion (VC) consists in the conversion of the perceived speaker identity. It is a technique for modifying the speech of a source speaker to sound like a target speaker while preserving the source speaker utterance [11, 12].

[1] Text preprocessing is excluded from this stage, as it is performed beforehand. Only the extraction of linguistic features from raw text is considered.

VC can be divided into two fields: parallel VC and non-parallel VC, depending on the nature of the training data. Parallel VC techniques are applied when datasets store identical linguistic content across multiple speakers. Therefore, acoustic features can be directly mapped from the source speaker to the target speaker through frame alignment [13].

Despite being a more intricate task, non-parallel VC is more useful in real applications than parallel VC, because it is applicable to a wider variety of data, not parallel datasets exclusively. Non-parallel VC is usually implemented in one of two ways: 1) converting non-parallel data to parallel data (e.g. synthesizing parallel utterances using TTS) and performing feature mapping by frame alignment; 2) retaining the source speaker linguistic content and transforming the speaker representation from source to target speaker. Unlike in parallel VC approaches, linguistic and speaker–related features are processed separately [13].

A typical voice conversion framework can be divided in two stages: the training stage and the conversion stage. In the training stage, it is first necessary to extract the target speaker and source speaker identity features. For this, a common approach is to use a speaker encoder, which generates a high-dimension representation of the speaker identity, also known as speaker embedding. Following this step, there are various ways to train the model and obtain a conversion function. A conventional technique is dynamic frequency warping, which essentially aligns the spectra of different speakers [14]. Neural methods [13, 15–17] have also been widely used to model spectral conversion.

2.2.1.1 Speaker embeddings

Speaker embeddings are representations that encode speaker–related features, such as identity, gender and speaking rate [18]. Speaker embeddings should be selected according to the purpose of the system, since some features may be better encoded in a specific type of embedding. For example, i-vectors perform much better in speaker identification than d-vectors and s-vectors [18] (i,d and s-vectors are kinds of speaker embeddings). In recent past, i-vectors were the most popular among speaker recognition systems.

Figure 2.3 illustrates a speaker encoder architecture proposed by [14]. According to this scheme, the system obtains the Mel-spectrograms of speaker audio samples and computes the corresponding speaker embeddings. The prenet performs spectral processing feature transformation from the Mel-spectrograms. Temporal processing is done by convolutional layers with gated linear unit and residual connections. After this, average pooling is applied to summarize the whole utterance [14]. The self attention block is used to assign weights to different audio samples in order to get combined embeddings.

Figure 2.3: Example of a speaker encoder architecture. Extracted from [14].

The quality of synthesized speech depends on the ability to properly encode necessary features to

generate speech with the intended characteristics. Since one of the main challenges nowadays, both for VC and other speech synthesis applications, is to extract the best data as possible from speaker embeddings, the research in these two fields is converging.

2.2.2 Voice Cloning

Voice cloning differs from voice conversion in two factors: 1) it can reproduce new speaker identities from reduced data — seconds or minutes of speech —, while voice conversion requires larger datasets — hours of speech — for the same task; and 2) it can generalize to unseen text, while voice conversion is bound to the linguistic content of the source utterance.

Voice cloning is capable of reproducing an unseen target voice from few samples of data by combining a model trained on data from many speakers with a small amount of target speaker data. This enables the resultant voice model to synthesize an unseen voice better than a model trained form scratch, exclusively on target speaker data [19]. The addition of a multi-speaker synthesis setting introduces a prominent improvement in speech generative systems. By enabling speech synthesis in multiple voices, these approaches effectively reduce the gap between single-speaker TTS and voice cloning.

According to Arik and colleagues (2018), there are two approaches for voice cloning: speaker adaptation and speaker encoding. Speaker adaptation consists in fine-tuning a trained multi-speaker model to a new speaker, using a small amount of data. This stage can be applied to the model and the target speaker embedding, or solely to the embedding. Contrary to speaker adaptation, speaker encoding directly predicts a speaker embedding of an unseen speaker and does not require a fine-tuning stage. As such, the multi-speaker model is combined with the new embedding, hence generalizing to unseen voices [14]. Both approaches are illustrated in figure 2.4. On the one hand, speaker encoding consumes less time and memory than speaker adaptation because it does not require and additional fine-tuning stage, which is more adequate when resources are limited. On the other hand, for speaker encoding to generalize to new voices, it is necessary to train the multi-speaker generative model and speaker encoder on significantly more speakers than for speaker adaptation. Moreover, speaker adaptation usually ensures better synthetic speech naturalness due to the fine-tuning stage. As such, speaker adaptation remains as the more practical voice cloning approach.

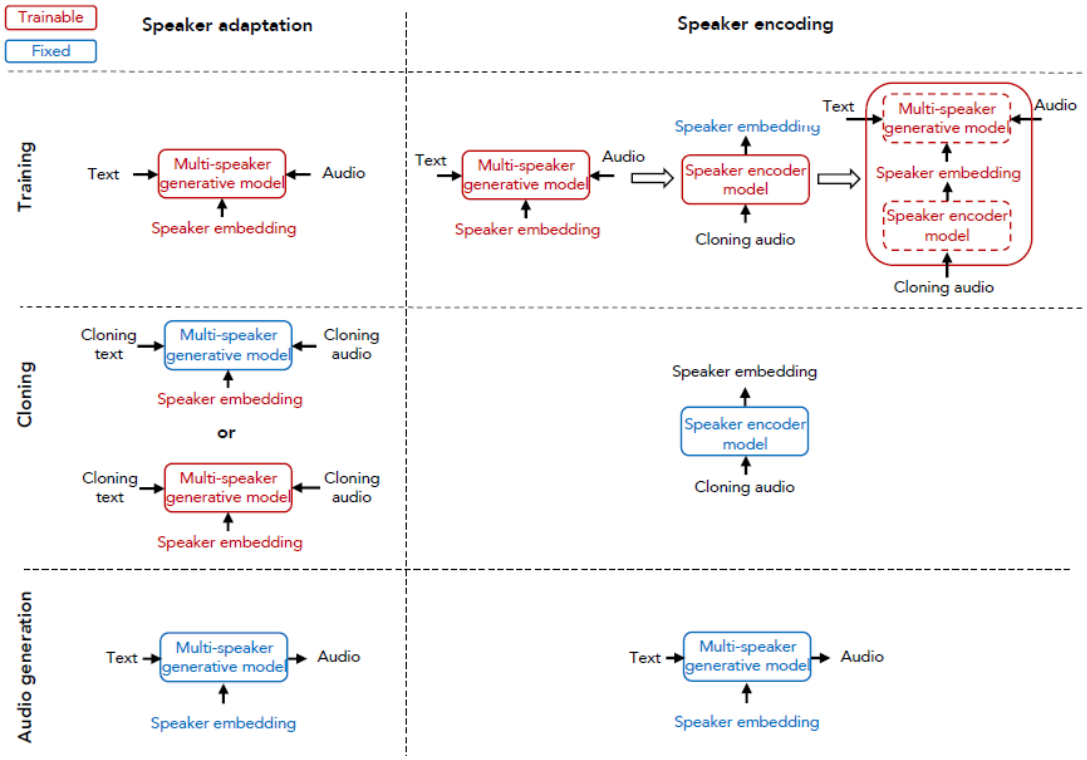


Figure 2.4: Speaker adaptation and speaker encoding techniques for voice cloning. Extracted from [14].

2.2.3 Evaluation metrics

This section presents the evaluation metrics used in the Voice Conversion Challenge 2018 [20]. Besides these subjective metrics, the frequently used preference test (A/B or ABX), as well as the more recent MUltiple Stimuli with Hidden Reference and Anchor (MUSHRA) are described.

2.2.3.1 Mean Opinion Score

In the field of Speech Synthesis, mean opinion score (MOS) tests are often used as an evaluation measure to assess the degree of naturalness of synthesized speech. Listeners are asked to rate the quality of synthesized speech according to a 5-point scale (1: bad, 2: poor, 3: fair, 4: good, 5: excellent). The final score of a system is given by the arithmetic mean over all the individual scores given by each listener.

2.2.3.2 Same/Different paradigm

The Same/Different paradigm is a subjective evaluation metric that measures the similarity of VC samples. From two samples, subjects are asked if the samples could have been produced by the same speaker, disregarding distortion and focusing only on identifying the voice. Based on their degree of certainty, subjects may choose one of the following answers: “Same speaker, absolutely sure”, “Same speaker, not sure”, “Different speaker, not sure” and “Different speaker, absolutely sure”.

2.2.3.3 AB/ABX preference test

Preference tests are frequently used to assess speech synthesis systems. In an AB preference test, as the name states, listeners are presented with two speech samples and are asked to select their preferred one according to a specific property, such as naturalness or similarity. A “no-preference” answer slot may be included. ABX tests differ from the traditional AB test with the inclusion of an “X” speech sample to be used as reference. In this case, samples “A” and “B” are evaluated using “X” as reference.

2.2.3.4 MUSHRA

In the scope of Speech Synthesis, the MUlti Stimuli with Hidden Reference and Anchor (MUSHRA) enables the subjective assessment of synthesized utterances and is mentioned in several studies, such as [21] and [22]. According to this method, listeners rate audio samples, together with a low-quality anchor, and a hidden reference sample, in comparison to a high-quality reference sample. The low-quality anchor corresponds to a low-pass filtered sample, and its purpose is to ensure minor artifacts are not improperly penalized. Samples are rated regarding similarity or perceived quality on a scale of 0 to 100, where 0 and 100 are the worst and best scores, respectively [23].

2.3 Neural vocoding

Vocoders perform speech waveform reconstruction from acoustic features. In traditional approaches, vocoders essentially comprise two blocks: acoustic feature extraction, and waveform generation. These techniques usually require different types of features (such as fundamental frequency, voiced / unvoiced binary value, or spectrum and band aperiodicities) which can make acoustic feature extraction an intricate task. Furthermore, conventional vocoders have difficulty in recovering the phase information of the waveform signal [24, 25].

Presently, state-of-the-art neural vocoders can directly generate speech waveforms from acoustic features, replacing speech analysis and waveform generation modules by neural networks. Despite requiring large amounts of data and computational power, neural vocoders preserve speech naturalness and recover phase information much more accurately than its predecessors [24, 26].

This section describes the reasoning, architecture, evaluation results of the following neural vocoders: WaveNet, Parallel WaveNet, and WaveRNN. Additionally, a Universal Vocoding architecture based on WaveRNN is also described.

2.3.1 WaveNet

According to Oord and co-authors (2016), WaveNet is a generative model for raw audio. As such, it is most frequently used in speech synthesis and can be efficiently trained on a large number (tens of thousands) of samples of audio data. WaveNet is fully probabilistic and autoregressive [27].

The joint probability of a waveform x with T timesteps, is given by equation (2.2),

$$p(\mathrm{x}) = \prod_{t=1}^{T} p\left(x_t | x_1, \ldots, x_{t-1}\right) \tag{2.2}$$

which means the predictive distribution for each audio sample is conditioned on all previous samples.

An important element in WaveNet is the causal convolution. Causal convolutions are used to ensure that the ordering of input data is not changed while it is being manipulated. Using causal convolutions is usually faster than training data with RNNs, nevertheless, causal convolutions require many layers to enlarge the receptive field, which results in increased computational costs, and consequently in slower synthesis time. This is one of the drawbacks of WaveNet that motivated the development of a faster solution, without sacrificing synthesized speech quality (see Parallel WaveNet, section 2.3.2).

In figure 2.5 a stack of convolutional layers is depicted. Neurons represent predictions at a specific timestep, where timesteps grow from left to right. Knowing this, we observe that every output of a neuron only depends on neurons from past and present timesteps, which goes along with equation (2.2). To reduce the high complexity of the network, which is caused by a large number of convolutional

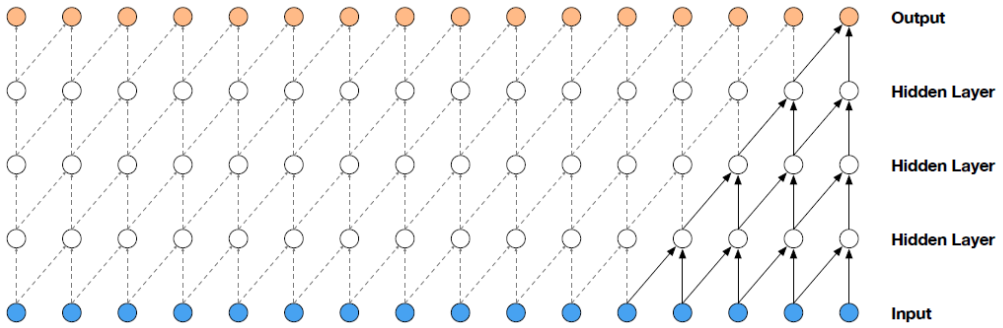


Figure 2.5: Stack of causal convolutional layers. Extracted from [27].

layers, WaveNet uses dilated causal convolutional layers. For these layers, the filter is applied in such fashion that inputs of neurons are skipped with a certain step [27]. This step is called dilation factor. The use of dilated convolutions ensures an increase in the receptive field without needing too many layers. Dilated convolutional layers and the dilation factor can be observed in figure 2.6. Since audio

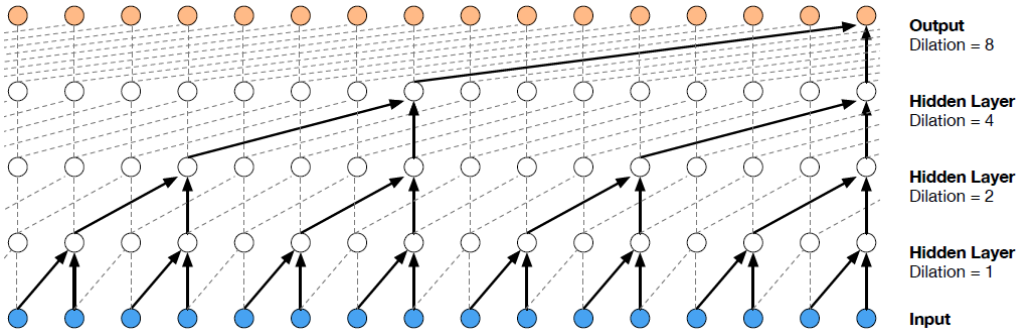


Figure 2.6: Stack of dilated causal convolutional layers. Extracted from [27].

is stored as 16-bit integers per timestep, a large number of outputs (65 536) is required to represent all

values per timestep, which is very costly. To avoid this, input data is compressed according to an 8-bit μ-law companding transformation. Therefore, data is quantized to 256 values per timestep according to equation (2.3),

$$f(x_t) = \text{sign}(x_t) \frac{\ln(1 + \mu|x_t|)}{\ln(1 + \mu)} \tag{2.3}$$

where $-1 < x_t < 1$ and $\mu = 255$. This kind of quantization proved to be significantly more efficient in terms of synthesis speed than linear quantization.

2.3.1.1 Architecture

The neurons in WaveNet's hidden layers are represented by a residual block. Instead of using the rectified linear activation function, a gated activation unit is applied instead:

$$\mathbf{z} = \tanh(W_{f,k} * \mathbf{x}) \odot \sigma(W_{g,k} * \mathbf{x}) \tag{2.4}$$

where $*$ represents a convolution operator, $\odot$ designates an element-wise multiplication operator, $\sigma(\cdot)$ is a sigmoid function, k is the layer index, f and g denote filter and gate, respectively, and W is a learnable convolution filter [27]. Figure 2.7 illustrates the architecture of the WaveNet model. The residual units

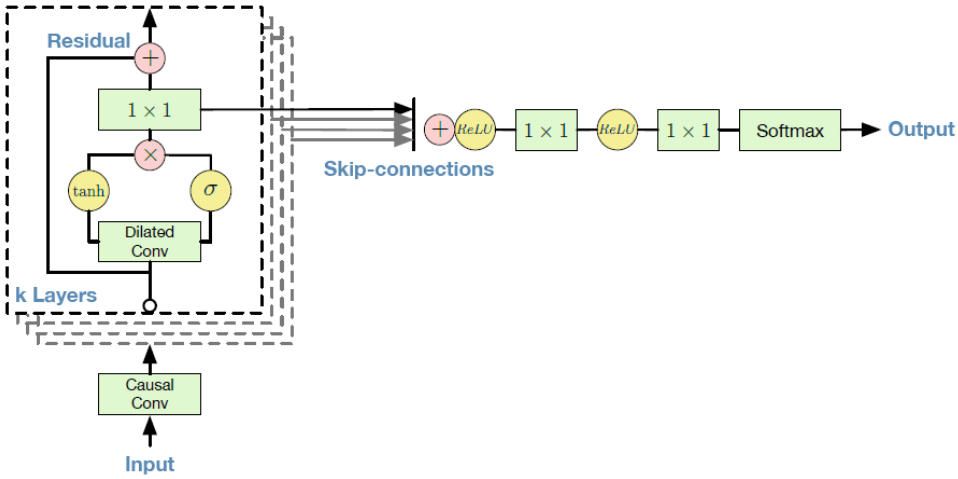


Figure 2.7: WaveNet model architecture. Extracted from [27]

are used to prevent the increasing training loss due to the network's significant depth. Together with these, skip connections enable a faster convergence.

2.3.1.2 Evaluation

In the scope of TTS, WaveNet models were conditioned on linguistic features extracted from input text. Besides linguistic features, the models were also given $\log F_0$ as input. Models were evaluated according to two metrics: 5-scale MOS and subjective paired comparison. Two other synthesizers were also tested, to compare WaveNet to other TTS implementations. Table 2.1 shows the results of MOS tests for North American English.

Table 2.1: 5-scale MOS evaluation for previous state-of-the-art systems (LSTM-RNN parametric and HMM-driven concatenative), for WaveNet conditioned on linguistic features and $\log F_0$, for 8-bit μ-law encoded natural speech and for 16-bit linear PCM natural speech. Extracted from [27].

Model	MOS
LSTM-RNN parametric	3.67 ± 0.098
HMM-driven concatenative	3.86 ± 0.137
WaveNet (L+F)	4.21 ± 0.081
Natural (8-bit μ-law)	4.46 ± 0.067
Natural (16-bit linear PCM)	4.55 ± 0.075

From table 2.1, it is clear that WaveNet achieved a remarkable score, very close to natural speech. Also, we notice that not even natural speech reached a perfect score.

2.3.2 Parallel WaveNet

Parallel WaveNet is based on the original WaveNet. It generates high-fidelity speech samples significantly faster than the original version. Parallel WaveNet relies on a method called Probability Density Distillation to train a parallel feed-forward network from a trained WaveNet model.

WaveNet ensures a fast training process of the network, as it is possible to achieve high temporal resolution because input samples are already available — the predictions for all timesteps can be done in parallel since all timesteps of the ground truth signal are known [27]. Nevertheless, the generation of samples from the output distribution is sequential, and therefore, a slow process. To tackle this issue, Parallel WaveNet uses inverse autoregressive flows (IAFs) to ensure a fast, parallel generation. IAFs [28] are stochastic generative models that allow to generate elements of a high dimensional sample in a parallel fashion [29].

2.3.2.1 Probability Density Distillation

Probability Density Distillation joins the best features of the aforementioned methods: for training, a classical trained WaveNet as “teacher”, and for sample generation, a parallel WaveNet as “student”, which learns from the trained WaveNet [29]. The goal is to match the “student’s” probability distributions of output samples to the “teacher’s” distributions. In practice, the input can be a noise signal that is gradually modeled to the desired output speech waveform by matching the output samples’ probability distributions. It is important to stress that the objective is not to match the output samples directly, as this would be a very complex task, but rather approximate them by matching their distributions.

2.3.2.2 Evaluation

In comparison with the original implementation of WaveNet [27], two changes were made that allowed for a higher audio fidelity. 16-bit audio modeling was used instead of 8-bit. Also, sampling rate was increased from 16 kHz to 24 kHz.

5-scale MOS tests were carried out to compare the "teacher" WaveNet and the "student" WaveNet performances. A concatenative model was also tested under the same conditions. Table 2.2 illustrates the obtained results. It is clear that the "student" WaveNet learns successfully and is able to produce speech according to the auto-regressive "teacher" WaveNet, since it yields practically the same result.

Table 2.2: Parallel WaveNet – Evaluation of "teacher" model and "student" WaveNet. Audio synthesized at 24kHz, 16-bit linear PCM. Extracted from [29].

Model	**MOS**
HMM-driven concatenative	4.19 ± 0.097
"Teacher" WaveNet	4.41 ± 0.069
"Student" WaveNet	4.41 ± 0.078

According to the authors, Parallel WaveNet can generate high-fidelity synthetic speech over 20 times faster than real-time on an NVIDIA P100 GPU, thus covering the low inference speed issues raised by the original WaveNet.

2.3.3 WaveRNN

In comparison to previous sequential models, such as WaveNet [27], WaveRNN aims to reduce sampling time, while preserving output quality. Combined with a dual softmax layer, this model can generate 24 kHz 16-bit audio $4\times$ faster than real-time on a GPU.

The sampling process duration $T(\mathbf{u})$, specified in equation (2.5), corresponds to the product of the number of samples in the target, $|\mathbf{u}|$, and the necessary time to generate each sample. To obtain the amount of time to produce a sample, the number of operations N, the computation time $c\,(op_i)$ and the overhead $d\,(op_i)$ are taken into account. Therefore, sampling time efficiency improves by reducing the factors N, $c\,(op_i)$, $d\,(op_i)$ and $|\mathbf{u}|$.

$$T(\mathbf{u}) = |\mathbf{u}| \sum_{i=1}^{N} \left(c\,(op_i) + d\,(op_i)\right) \tag{2.5}$$

Opposed to convolutional models, WaveRNN does not require deep architectures to process each sample of audio. Its recurrent nature allows for retaining features in the input sequence and establishing dependency among them. Hence, WaveRNN computes the combination of context with the input at each training step, significantly reducing the network's depth and consequently the number of performed operations [30].

WaveRNN predicts 16-bit audio samples, divided in two 8-bit parts: 8 coarse bits $\mathbf{c}_t$, and 8 fine bits $\mathbf{f}_t$. The model, illustrated in figure 2.8, is composed by a single-layer RNN with a dual softmax layer. It takes as input the previous audio sample $\mathbf{c}_{t-1}, \mathbf{f}_{t-1}$, the coarse input $\mathbf{c}_t$, and outputs two discrete probability distributions, $P(\mathbf{c}_t)$ and $P(\mathbf{f}_t)$, for the sample at time t [30, 31].

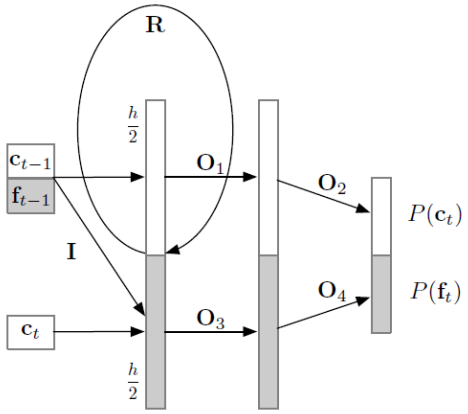


Figure 2.8: Architecture of WaveRNN: single-layer RNN combined with a dual softmax layer. $\mathbf{R}$ and $\mathbf{I}$ are matrices denoting the hidden state weights of the RNN. Extracted from [30].

2.3.3.1 Evaluation

Although WaveRNN does not surpass WaveNet in terms of naturalness, it still obtains a high MOS, close to WaveNet's. WaveRNN proves to be more suitable than WaveNet (2016) for real-world applications because it ensures faster synthesis, up to 96 000 samples per second.

Table 2.3: WaveRNN MOS results in comparison to WaveNet. For this table, a WaveRNN with a state of 896 units was considered. Extracted from [30].

Model	MOS
WaveNet	4.51 ± 0.08
WaveRNN	4.37 ± 0.07

2.3.4 Universal vocoding

Universal vocoders aim to improve the generalization capabilities of neural vocoders. The dependency on extensive datasets and computational power motivated the search for new solutions that could incorporate new speaker-styles, without further training [26].

Lorenzo-Trueba and colleagues (2019) proposed a Speaker Independent WaveRNN-based universal vocoder, depicted in figure 2.9, capable of generalizing to unseen speakers. The model is based on WaveRNN's architecture because it provides more stable outputs than CNN-based approaches. The RNNs hidden state continuance retains context across the spectrogram and therefore generates a more stable output [26]. Similarly to WaveRNN, the autoregressive side of the model includes one GRU and a softmax output layer. Additionally, two affine layers link the GRU to the softmax. The conditioning network comprises a two-layer bidirectional GRU that processes the input Mel-spectrograms.

2.3.4.1 Experiments and evaluation

The authors considered four different training settings: a single-speaker configuration, two multi-speaker configurations, consisting of three and seven speakers respectively, and a universal vocoding

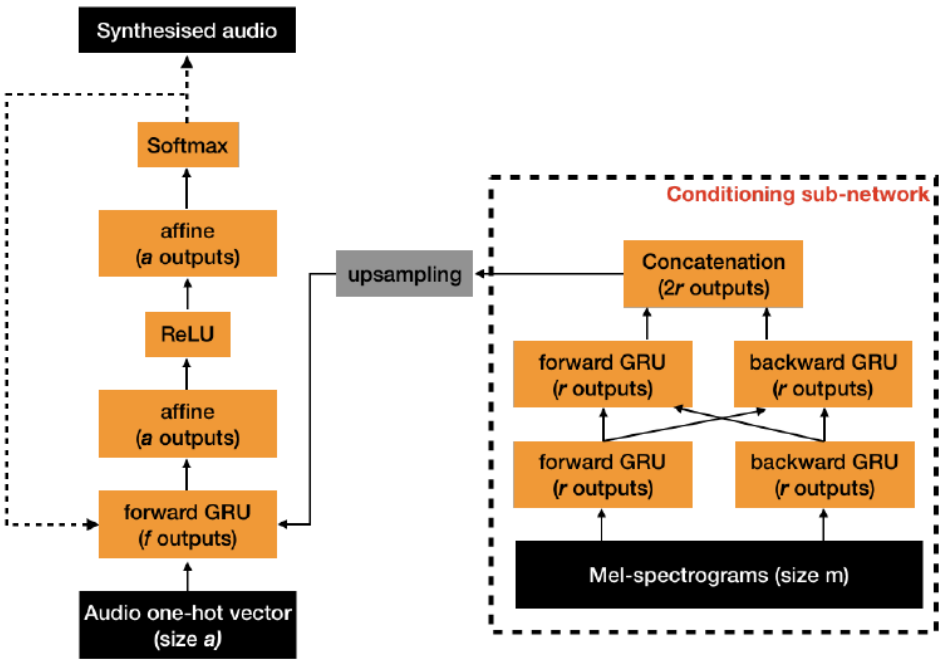


Figure 2.9: Block diagram of the WaveRNN-based Universal Vocoder. Extracted from [26].

configuration, comprising 74 speakers and 17 languages. The multi-speaker configurations aimed to assess how reducing the dataset size (number of utterances) and increasing the number of speakers would influence the output quality.

Evaluation and experiments included several scenarios: 1) in-domain speakers and speaking style; 2) out-of-domain speakers but similar speaking style; 3) out-of-domain speakers and speaking style. Additionally, various unseen scenarios were tested. From these, we will focus on the following: different voice qualities and non-optimal recording conditions. MUSHRA tests were used to evaluate each case [26].

No significant difference was registered for scenario 1) since all training settings produced similar results at inference time. The universal vocoder setting registered a 98.5% relative MUSHRA score[2]. For scenario 2), results have shown that the more speakers are included during training, the better is the output quality. Although the 3-speaker setting comprised more data than the 7-speaker setting, the output quality was better for the latter, which indicates that speaker variability is more important than quantity of data regarding the universal vocoding task [26]. For scenario 3), the universal vocoder setting still provided a stable output, reaching a 98% relative MUSHRA score. Single-speaker and 7-speaker settings revealed poor results, unlike the remaining (3-speaker and universal vocoder). This contrast was mainly due to speaker dissimilarity among train and test speakers. The authors used Kullback-Leibler divergence (KLD) to measure speaker similarity. KLD was measured between the Gaussian Mixture Models of the training data of each vocoding approach and the speaker. The KLD between the test speaker and each one of the settings was 2.64 for the universal vocoder, 5.42 for 3-speaker, 14.45 for 7-speaker, and 14.62 for single-speaker, proving that dissimilarity between train and test speakers

[2]The relative MUSHRA score is the ratio between mean MUSHRA scores of a system, and of natural speech.

severely degraded the output quality for unseen speaking style scenarios.

Regarding robustness to voice quality degradation, the universal vocoder still generalized well, achieving 91.6% and 89.5% relative MUSHRA scores for breathy and pressed voices, respectively. In terms of signal quality, the model's performance further worsened, achieving 79.4% and 76.4% relative MUSHRA scores for noisy, and reverberating signals, respectively. The most significant drop occurs for simultaneously noisy and reverberating signals, for which the relative MUSHRA score drops to 57.8%.

2.4 Sequence-to-Sequence TTS systems

Over the past few years, TTS systems have been implemented according to two frameworks: previous generation statistical parametric synthesis systems (SPSS), and the more recent Seq2Seq approach with better performance, overall. These two TTS approaches mostly differ in three aspects: 1) In SPSS, linguistic processing is rule-based, and therefore an extensively scripted procedure, whereas, in Seq2Seq, text analysis is performed by a neural text encoder; 2) While in SPSS text analysis and speech are processed separately, in Seq2Seq these are processed jointly, using $\{$text, audio$\}$ pairs as input; 3) SPSS predicts adjacent acoustic frames independently, while Seq2Seq performs the same task autoregressively [32].

Although Seq2Seq TTS performs better, there are still advantages and disadvantages in both approaches, given their core differences. The traditional SPSS framework does not require large text datasets, because linguistic representations are obtained by rule – like in the Festival pipeline [33], for example. Seq2Seq TTS models are usually "data-hungry" instead, and require large amounts of text data because linguistic representations are learned naturally along with speech representations. On the one hand, the SPSS approach may ensure more accurate linguistic content than Seq2Seq TTS — for example, the correct pronunciation of homograph words in the same sentence — for smaller amounts of data. On the other hand, the seamless learning process of Seq2Seq TTS generates much more natural synthetic speech as opposed to the rule-based and detached processing pipeline performed in SPSS.

This section presents several Seq2Seq systems, namely Tacotron and Tacotron 2. Ultimately, these systems constitute a significant step towards fully end-to-end architectures. Tacotron somewhat achieves this, but only because it uses a Griffin-Lim vocoder, which substantially degrades the quality of synthetic speech. Tacotron 2 however, produces synthetic speech of much better quality because it uses a separate trainable system (WaveNet) as a neural vocoder, hence not being truly end-to-end.

2.4.1 Tacotron

According to Wang and colleagues (2017), Tacotron is an end-to-end TTS model that generates synthetic speech directly from text. A Tacotron model can be trained from scratch with random initialization, given $\{$text, audio$\}$ pairs. Tacotron is frame-based, which is significantly faster than sample-level auto-regressive methods and it does not require conditioning on linguistic features [34].

The model operates as follows: it receives characters as input, generates a linear-scale raw spectro-

gram from the input, and outputs synthetic speech by applying the Griffin-Lim reconstruction algorithm (GLA) to the spectrogram.

An alternative to Tacotron, regarding audio generative models, is WaveNet. Nevertheless, WaveNet is slower than Tacotron because of its sample-level auto-regressive nature, and because it requires conditioning on linguistic features (it is not an end-to-end model).

2.4.1.1 Architecture

Tacotron is composed of three main blocks: an encoder, a decoder and a post-processing net. Figure 2.10 illustrates the model's architecture, including all three main blocks. A CBHG module is included both in the encoder and in the post-processing net. The purpose of the CBHG (1-D convolution

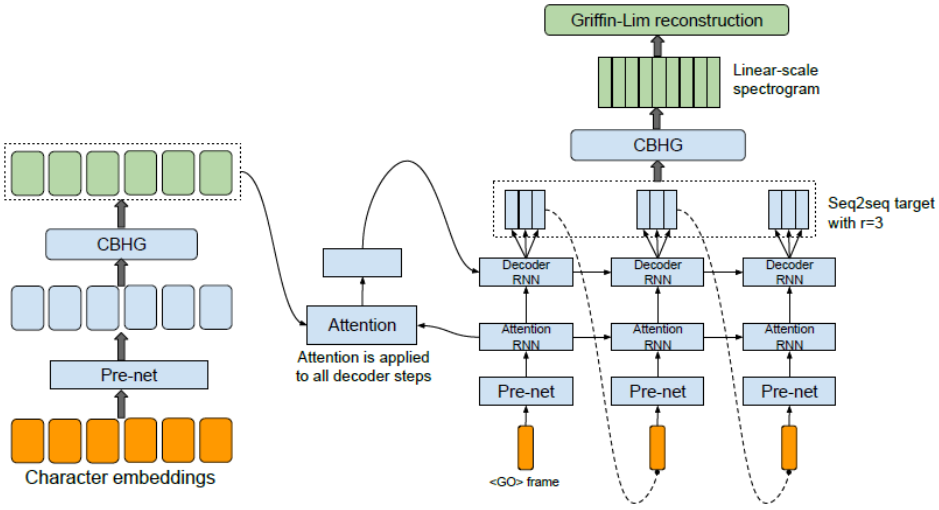


Figure 2.10: Tacotron model architecture. Extracted from [34].

bank + highway network + bidirectional GRU) module is to extract features from sequences. Essentially, it extracts higher-level features such as phonetic, prosodic, and lexical information [35] from the input sequence.

The encoder extracts robust sequences of text. It receives a continuous vector of embeddings, where each embedding corresponds to a character represented by a one-hot vector. The pre-net module corresponds to a set of non-linear transformations applied to each embedding. On top of this, a bottleneck layer with dropout is applied, in order to improve generalization. Finally, a CBHG module is used obtain the final output of the encoder. The presence of this module is advantageous because it reduces overfitting and mispronunciations [34].

Tacotron uses a content-based hyperbolic tangent attention decoder, where the attention query for each decoder time step is generated by a recurrent layer. Multiple non-overlapping output frames are predicted at each decoder time step, which is less time and space consuming.

The post-processing net converts the Seq2Seq output of the decoder to a linear scale spectrogram. This spectrogram is transformed into a speech waveform by the Griffin-Lim algorithm [34]. The choice

of this algorithm is justified by its parsimonious nature. Although it does not achieve the best results in terms of convergence and waveform quality, given its simplicity, it fits the demands of the model adequately.

2.4.1.2 Evaluation

To evaluate the system, MOS tests were conducted. The tests consisted of 100 unseen phrases, rated eight times each. Besides Tacotron, concatenative and parametric synthesizers were tested for comparison purposes. The achieved results are presented in table 2.4.

Table 2.4: 5-scale MOS evaluation - Tacotron. Extracted from [34].

Model	**MOS**
Tacotron	3.82 ± 0.085
Parametric	3.69 ± 0.109
Concatenative	4.09 ± 0.119

Tacotron achieved better results than the parametric model. Synthetic speech generated by Tacotron comprises artifacts due to the Griffin-Lim algorithm waveform generation step — the Griffin-Lim algorithm does not perform an accurate estimation of the phase of an audio signal, thus generating unnatural speech. The concatenative model accomplished better results than Tacotron, because Tacotron does not employ a neural vocoder that would ensure better phase estimation, and consequently, better synthetic speech quality.

2.4.2 Tacotron 2

Tacotron 2 is a Seq2Seq TTS system. It uses a recurrent sequence-to-sequence feature prediction network, adapted from Tacotron, to convert input text to Mel-scale spectrograms. A modified version of WaveNet performs waveform synthesis from the Mel-spectrograms. The use of Mel-scale spectrograms as input to WaveNet instead of linguistic, duration and F_0 features proved to reduce greatly the complexity of the original WaveNet architecture. As such, Tacotron 2 gathers the best properties from its predecessor models, Tacotron and WaveNet [36].

A spectrogram represents the evolution of frequencies of a signal over time. Therefore, in the case of a speech waveform, a linear spectrogram corresponds to the short-time Fourier transform (STFT) magnitude of the waveform. A Mel-scale spectrogram is obtained by applying a non-linear transformation to the linear-scale spectrogram. This transformation is particularly useful when dealing with speech signals because it focuses on low frequency variations, which are very important for speech intelligibility, and de-emphasizes high frequency variations, which do not require high-fidelity modeling [36].

2.4.2.1 Architecture

The encoder and decoder blocks in Tacotron 2 are based on Tacotron's architecture but with some simplifications. CBHG and GRU-RNNs are replaced by vanilla LSTMs and convolutional layers.

To generate a 16-bit audio samples from the Mel-spectrograms, WaveNet uses a 10-component mixture of logistics distribution.

Because Mel-spectrograms provide more information about the waveform than linguistic features, WaveNet can be modified to a shallower architecture, with less hidden layers, and still capture long-term dependencies across frames. Therefore, it is clear that conditioning WaveNet on Mel-spectrograms does not require as large receptive fields as in the original implementation of WaveNet.

2.4.2.2 Evaluation

To evaluate Tacotron 2, MOS tests were used as evaluation method. Other models were evaluated as well for comparison purposes. Table 2.5 displays the results of the MOS tests. The results prove that Tacotron 2 is able to synthesize speech with quality very close to natural speech.

Table 2.5: 5-scale MOS evaluation of Tacotron 2. Extracted from [36].

Model	**MOS**
Parametric	3.492 ± 0.096
Tacotron (Griffin-Lim)	4.001 ± 0.087
Concatenative	4.166 ± 0.091
WaveNet[3]	4.341 ± 0.051
Ground truth	4.582 ± 0.053
Tacotron 2	4.526 ± 0.066

2.4.3 Multi-speaker generative model

Zhang and co-authors (2019) proposed a non-parallel Seq2Seq voice conversion model, illustrated in figure 2.11, that operates in one of two configurations depending on the type of input: voice conversion (VC), or TTS. For the VC configuration, the acoustic feature sequence (in the form of a Mel-spectrogram) is extracted from a source utterance and is fed to the Seq2Seq regressive model, as depicted in figure 2.11. The VC configuration follows the traditional framework, which takes a source utterance as input. This setting preserves the input's linguistic content and embeds the target speaker's identity into the output. For the TTS configuration, text inputs are converted to phonetic transcriptions before being fed to the model. The TTS process unfolds similarly to the VC procedure since the output is also a combination of linguistic content and speaker identity. The most notable difference is the textual input, as opposed to a source utterance. Since TTS is the main focus of this dissertation, we will focus on the system's TTS configuration.

The model incorporates five components: a text encoder $\mathbf{E}^t$, a speaker encoder $\mathbf{E}^s$, a Seq2Seq decoder $\mathbf{D}^a$, a recognition encoder $\mathbf{E}^r$, and an auxiliary classifier $\mathbf{C}^s$. The text encoder extracts linguistic embeddings $\boldsymbol{H}^t$ from phoneme transcriptions $\boldsymbol{T}$ of input text sequences. It is formed by three convolutional layers, followed by a single-layer BLSTM and a fully connected layer. The speaker encoder

[3]WaveNet conditioned on linguistic features

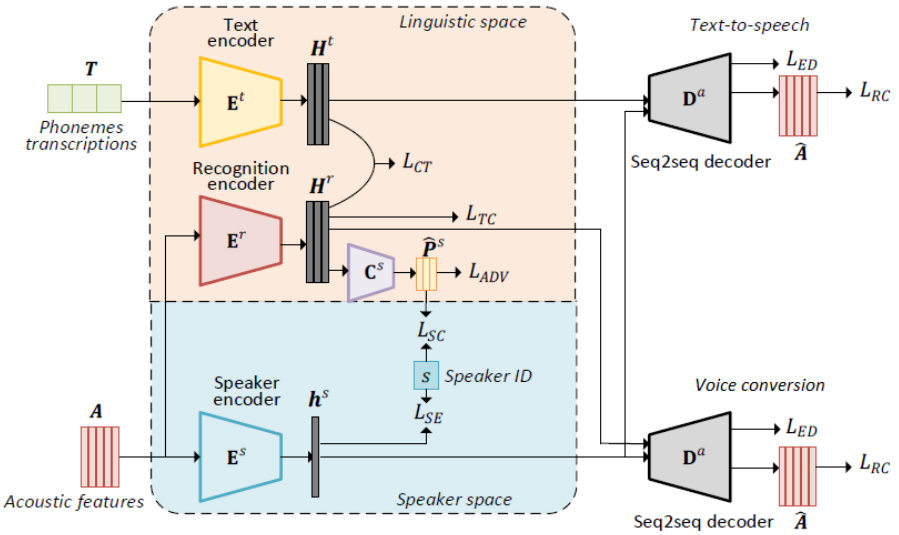


Figure 2.11: Architecture of a Seq2Seq VC model comprising the voice conversion and TTS configurations. Adapted from [13].

takes spectrograms as input and generates speaker embeddings $\boldsymbol{h}^s$, capable of identifying a speaker. It consists of a 2-layer BLSTM, followed by average pooling and a fully connected layer. The decoder generates an acoustic feature sequence $\hat{\boldsymbol{A}}$ by combining previously extracted linguistic and speaker embeddings. Its structure is analogous to Tacotron [34, 36]. Authors used a WaveNet vocoder [37] to recover speech waveforms from acoustic features [13].

The recognition encoder $\mathbf{E}^r$ and the auxiliary classifier $\mathbf{C}^s$ are only taken into account during the training process, thus are not used by the TTS configuration at inference time. The recognition encoder extracts linguistic representations from audio signals. For that, it takes acoustic feature sequences $\boldsymbol{A}$ as input, and outputs a linguistic embedding $\boldsymbol{H}^r$. Since $\boldsymbol{H}^t$ and $\boldsymbol{H}^r$ are expected to be similar, a contrastive loss is introduced to increase similarity between both linguistic representations. The auxiliary classifier predicts the speaker identity from a linguistic $\boldsymbol{H}^r$. It is used for adversarial training to remove the remaining speaker-related content within the linguistic embedding [13].

Training the model consists of two stages: pre-training, and fine-tuning. In the pre-training phase, the system is trained with a large multi-speaker dataset, comprising utterances, text transcriptions, and the corresponding speaker identity tag. Fine-tuning was performed in a 2-speaker setting, nevertheless, it is possible to fine-tune the model to more than two speakers. The fine-tuning stage introduces unseen speakers during pre-training and converges faster than the first stage.

The authors evaluated the effect of training data reduction on the model's performance at the fine-tuning stage. Results have shown that the model has similar performance regardless of the amount of training data, suggesting that this implementation is suitable for scenarios where the amount of data is scarce [13].

2.5 TTS for European Portuguese

This section provides a brief overview of TTS systems for European Portuguese. ResponsiveVoice[4], the Nuance Vocalizer[5] and ReadSpeaker[6] are systems available online that yield a good performance, although they lack exclamatory and interrogative intonation, and sometimes introduce artifacts.

INESC ID has a long history in what concerns TTS systems for different varieties of Portuguese. This history started with the development of DIXI [38], a rule-based synthesizer using a formant model. This TTS system was followed by DIXI+, the concatenative version based on the Festival framework.

More recently, two Master Theses, [39] and [40], were developed in this field. Gonçalves (2018) [39] proposed a model based on Merlin, which is a toolkit for neural network-based speech synthesis [41]. Merlin's framework, based on the Festival synthesis platform, was originally designed to synthesize a single English voice, and therefore had to be adapted to the Portuguese phonology. The model was trained with an EP dataset with total duration close to 16 hours.

Quintas (2019) [40] developed a Portuguese speech synthesizer, based on Tacotron 2, targeting applications to Amyotrophic Lateral Sclerosis (ALS) patients. Two datasets (one of a female speaker and the other of a male speaker) were used for speaker adaptation as those yielded high quality utterances. To train the model from scratch, only a female speaker dataset was used, since it was the only sufficiently large voice bank available. The model's hyperparameters were adjusted to ensure a proper training from the voice banks. The system was evaluated in terms of naturalness and intelligibility and more specifically, interrogative and exclamatory intonation.

The implementation developed by Quintas (2019) [40] yields good results in terms of speech quality, nevertheless it utilizes the original WaveNet [27] as a vocoder, which poses a drawback in terms of synthesis time. This characteristic makes the system inoperable for real time applications. Another downside was the fact that the datasets used for speaker adaptation were small, and did not provide data diverse enough. The scarcity of interrogative and exclamatory intonations in these datasets limited the performance of the system. As such, further development in TTS for European Portuguese should aim to ensure speaker adaptation based on fewer input samples, with faster synthesis time.

2.6 Final remarks

Other methods greatly contributed to the state-of-the-art of Speech Synthesis besides those presented in this chapter: Deep Voice 3 [42], VoiceLoop [43], and more recently the ESPnet-TTS toolkit [44] regarding TTS, StarGAN-VC2 [45] in the field of voice conversion, and technologies outside the scope of this study, namely multilingual TTS [46] and speech-to-speech translation [47], systematized by Luís Bernardo (2019) [48].

The concepts of TTS and voice conversion are originally distinct right from the starting point, as TTS takes raw text, and VC takes speech waveforms as input. Nevertheless, from a practical standpoint,

[4] https://responsivevoice.org/
[5] https://www.nuance.com/omni-channel-customer-engagement/voice-and-ivr/text-to-speech.html
[6] https://www.readspeaker.com/pt/demonstracoes/

both fields are conjoining, essentially for two reasons: 1) both ultimately aim to reach the same goal — to generate synthetic speech as natural as possible, while preserving the given linguistic content; and 2) both concepts are structurally very similar when using an encoder-decoder framework.

Recalling the system described in item 2.4.3, it becomes clear that most of the underlying structure of multi-speaker TTS and non-parallel voice conversion is common to both, only diverging in terms of input encoding. Likewise, the waveform generation module can be identical in both cases, since the output type is the same. For this reason, choosing a Seq2Seq framework coupled with neural waveform generation for a multi-speaker TTS system is adequate, as it can be easily adjusted to similar confluent tasks, like VC. The next chapter describes a model that follows this premise.

Chapter 3

Proposed model

This chapter aims to minutely describe the structure of a proposed model for multi-speaker TTS. Other aspects related to the model's operation are also detailed, namely the speech corpora that were employed, and the training process involving the main modules of the system.

Section 3.1 identifies these components, and explains how they interconnect. Section 3.2 enumerates the spoken language corpora that were employed to train the model, and details their main characteristics that influence the performance of the model. Sections 3.3 and 3.4 describe, respectively, the text and audio preprocessing stages that precede the training of the model's main components. Additionally, sections 3.5 and 3.6 focus on the hyperparameter selection and training processes of the system's main modules.

3.1 Model overview

The present section describes the proposed multi-speaker TTS system. This approach (depicted in figure 3.1) follows the prevailing state-of-the-art premise, combining a Seq2Seq system for acoustic feature prediction, with a neural vocoder for speech waveform recovery. Currently, multiple architectures for neural vocoders are available, given their fundamental purpose within Speech Synthesis. To select a suitable neural vocoder, we focused on open-source implementations that enabled faster synthesis than the original WaveNet. Early experiments involved an implementation[1] inspired by the original WaveRNN. However, this model demanded extensive training data (hours of speech), and required fine-tuning to generate speech for new voices. Fine-tuning for new speakers did not produce reasonable results, suggesting that the model only performed well in a single-speaker setting. Since this system clearly did not fit the demands of speaker adaptation, a different neural vocoding approach was selected.

A WaveRNN-based universal vocoding approach was chosen for the waveform generation module. This approach is an independently developed implementation of the original universal vocoding system [26]. The WaveRNN-based universal vocoding architecture allows for faster synthesis (as opposed to WaveNet), and more importantly, for better generalization to new speaker identities. This aspect

[1] `https://github.com/fatchord/WaveRNN`

was crucial to prevent training the neural vocoder at the speaker adaptation stage, thereby simplifying that process. The Seq2Seq system was adapted from the non-parallel Seq2Seq VC implementation [13], given its similarities with multi-speaker TTS. Sections 2.3.4 and 2.4.3 of chapter 2 explain the implementations of the modules selected for the TTS framework in detail.

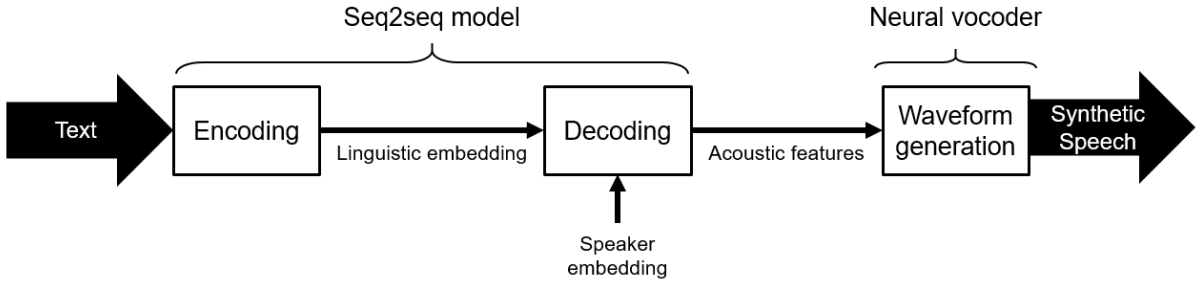


Figure 3.1: Inference-time pipeline of the proposed model. The system receives a textual input, together with a speaker embedding in order to select the speaker identity for synthesis. At inference time, the encoding stage extracts a linguistic embedding from the textual input. The decoding stage combines linguistic and speaker embeddings to predict a Mel-spectrogram, conveying the adequate linguistic content and speaker identity. Finally, the neural vocoder produces a synthetic speech waveform from the predicted Mel-spectrogram.

The Seq2Seq VC model was originally designed for the English language (EN), therefore, several adjustments were introduced to adapt this system to European Portuguese (EP). These adaptations consisted in text and audio preprocessing stages (described in sections 3.3 and 3.4, respectively). As stated in section 2.4.3, the text encoder of this model relies on phonetic transcriptions, which require changing the grapheme-to-phoneme (GtoP) conversion stage from EN to EP.

When choosing a speech corpus, one must account for several factors, as these might pose a complex challenge for the Seq2Seq model. The speech corpus should include a broad variety of speakers as well as enough data per speaker for the model to generalize properly in a multi-speaker setting. Moreover, the duration of its utterances should be limited. Very long utterances may be difficult to decode and therefore prevent the model from converging. At last, recording conditions may also undermine the performance of the Seq2Seq model (e.g. samples containing silence, background noise or breathing sounds). Audio preprocessing was used to mitigate these, whenever possible. Audio data underwent the same digital signal processing (DSP) stages both in the Seq2Seq model and neural vocoder, otherwise, it would not have been possible to link the models together.

The proposed model was adapted from open-source repositories 1 and 2, depicted in table 3.1. We will refer to this as default implementations. Repositories 3 and 4 were used at preprocessing stages.

Table 3.1: Open-source repositories used for the proposed model.

#	URL	Task
1	`https://github.com/bshall/UniversalVocoding`	Neural vocoder
2	`https://github.com/jxzhanggg/nonparaSeq2seqVC_code/`	Non-parallel Seq2Seq VC
3	`https://github.com/cmusphinx/g2p-seq2seq`	Text preprocessing (GtoP)
4	`https://github.com/pyannote/pyannote-audio`	Audio preprocessing (silence removal)

3.2 Spoken language corpora

Selecting suitable voice banks to train the two main blocks of the proposed model (Seq2Seq model + neural vocoder) was an important step during implementation because it greatly influenced synthetic speech quality. Therefore, EP voice banks with similar specifications to English ones were chosen, as the latter already produced solid results.

As previously described in section 2.4.3, the Seq2Seq model comprises two training stages: pre-train, followed by fine-tune. Different corpora were used at each stage, since the purpose of fine-tuning is for the model to perform speaker adaptation, requiring new speakers. The default implementation was pre-trained with data of 99 speakers from the VCTK dataset [49], and fine-tuned with the data of two speakers from the CMU Arctic database [50].

Table 3.2 indicates the specifications for the corpora used as reference for pre-train and fine-tune stages of the Seq2Seq regressive model. BD-PUBLICO is a corpus of newspaper text, and its subset used for pre-training comprises a vocabulary of 20 000 words uttered by 100 speakers aged between 19 and 28. The subset of BDFALA used for fine-tuning contains 600 phonetically rich sentences [51] uttered by two adult speakers. When fine-tuning the model, however, one could have selected an arbitrary number of speakers. Two speakers were chosen merely to include both a male and a female voice during this stage. Two additional corpora were employed for speaker adaptation experiments: 1) a different subset of BDFALA, which will be referred to as BDFALA02; and 2) a subset of the EUROM.1 corpus [51]. These will be further detailed in chapter 4.

In terms of speaker diversity, both EP datasets are adequate, nevertheless, in terms of utterance duration, and number of utterances per speaker, BD-PUBLICO falls behind VCTK. It is clear that BD-PUBLICO not only includes less utterances per speaker, but also comprises substantially longer utterances than VCTK. In fact, to ease the convergence task during training, a larger number of shorter utterances per speaker is preferable. To test the impact of utterance length in the convergence of the training process, an experiment was carried out, in which training data comprised isolated words exclusively. This training setting comprised two voices, and over 4500 uttered words for each voice. Even though only two speakers were employed, the model successfully reproduced both voices, generating natural and intelligible synthetic speech for unseen isolated words significantly better than when trained with the same two speakers, but longer utterances. In case one wants to train a model capable of decoding very long utterances, a possible approach is to use curriculum learning [52], in which the utterance length is gradually increased as the model trains, ensuring smoother convergence. This, however, poses another drawback because it is hard to find corpora with such specifications. When training speech generative models using curriculum learning, a frequent but laborious approach is to split utterances into shorter ones. Although this eases the convergence process of the model, it must be done carefully to preserve the utterances' original intonation.

*Original sampling frequencies are higher, but all waveforms were downsampled to 22.05 kHz in DSP stage.
**Corresponds to the number of speakers used for training the model. Originally, these speech corpora contain more speakers.

Table 3.2: Voice corpora used as reference for pre-train and fine-tune stages of the regressive Seq2Seq model.

Specification		Pre-train		Fine-tune	
		VCTK	**BD-PUBLICO**	**CMU-ARCTIC**	**BDFALA**
Language		EN	EP	EN	EP
Sampling frequency [kHz]		22.05*	16	22.05*	16
Number of speakers		99**	100	2**	2**
Utterances per speaker		231-503	79-83	539	300
Total duration [hh:mm:ss]		40:25:40	21:48:39	02:02:45	00:51:36
Utterance duration [s]	Mean	3.6	9.7	3.3	5.2
	Median	3.3	9.5	3.2	5.1

The neural vocoder's default implementation (repository 1 in table 3.1) relied on a 102-speaker English dataset [53] different from VCTK, comprising 9474 utterances and a duration of 20h 20min. For EP the vocoder was also trained with the BD-PUBLICO corpus, firstly to ensure that speaking style and recording quality were preserved, and secondly because speaker diversity and total corpus duration are similar to the EN corpus employed in the original implementation.

3.3 Text preprocessing

Regarding text preprocessing, three different GtoP approaches were employed in the following order: 1) the Phonemizer [54] using the eSpeak backend; 2) a Sequence-to-Sequence GtoP toolkit, developed by CMUSphinx (repository 3 in table 3.1); and 3) a Festival-based approach, which is employed in DIXI+ [55].

Besides assessing the ability to generate correct phonetic transcriptions, other factors were considered in order to choose which approach is best, in particular, text normalization, and the ability to correctly transcribe homographs and acronyms. Text normalization is an important step in GtoP methods because it allows to obtain correct phonetic transcriptions for non-standard text, such as abbreviations or digits.

3.3.1 Punctuation

Although punctuation marks are not phonetic symbols, the main reason for including punctuation in phonetic transcriptions is for the model to learn pauses correctly, which are most frequently represented by commas or full-stops. Ideally, punctuation would contribute to a large extent to learning how to convey exclamatory and interrogative sentences in synthetic speech. Nevertheless, this was not possible due to the scarcity of these sentences in training data, as detailed further in this section.

From all punctuation marks, only commas, full stops, and question marks were included in phonetic transcriptions. Including all possible punctuation would contribute to unnecessarily large phoneme lists, that is, an excessively large number of different phonemes in the dataset. A phoneme list with many instances would not only hinder the training task, but could potentially cause out-of-memory (OOM)

errors, since the number of instances in the phoneme list corresponds to the output size of one of the layers employed in the decoding process.

In total, BD-PUBLICO comprises 13 002 commas, 273 colons and 38 semicolons. Given that commas are substantially more frequent, colons and semicolons were replaced with commas. Likewise, ellipses were replaced with full stops. Although BD-PUBLICO does not include any ellipses, the remaining corpora employ these in some transcriptions, hence the replacement with full stops.

Given that the sentences of BD-PUBLICO were all selected from newspaper text, the number of exclamatory and interrogative sentences is very scarce in the whole corpus. Moreover, several of these exclamatory sentences were uttered in declarative fashion (in chapter 4, section 4.1.1, the type of sentences and prosody of each corpus is thoroughly analyzed). As such, exclamatory sentences were assumed as declarative ones, meaning that exclamation marks were replaced with full stops at the end of sentences.

3.3.2 Phonemizer with eSpeak backend

The open-source repository used for the Seq2seq model already employs a GtoP toolkit, the Phonemizer [54], which supports different languages and frameworks, namely Festival and eSpeak. Given that this toolkit was readily available, this was the first GtoP procedure to be experimented. The rule-based eSpeak backend was used since it was the only one available for EP in this toolkit.

Unlike the Festival backend, eSpeak employs the International Phonetic Alphabet (IPA) to represent phonemes. In comparison to other phonetic notations, such as SAMPA, IPA is more complex, comprising over 100 different symbols representing vowels and consonants only. Hence, using the eSpeak backend generated very large phoneme lists. To put it in perspective, the EN Festival backend generated a 41-symbol phoneme list, while the EP eSpeak backend generated a list comprising 73 symbols, excluding punctuation. Given that each backend refers to a different language, it is expected that each phoneme list has a different number of symbols, nevertheless, eSpeak generated an abnormally large number of phonemes, almost as twice as EN Festival.

Preliminary experiments regarding the training of the Seq2Seq regressive model did not include punctuation in phoneme lists, as these were already extensive — adding more symbols to the phoneme list would increase memory usage, and thus cause OOM errors. However, excluding punctuation from phoneme lists posed a significant drawback in the performance of the regressive model, as it was incapable of conveying pauses in predicted acoustic representations. To overcome the problem of extensive phoneme lists, 22 symbols were merged, and one symbol was excluded. The details of the phoneme list reduction procedure are presented in the following section 3.3.2.1.

Besides employing a far intricate phonetic alphabet for the intended task, the present GtoP procedure presented other limitations: 1) it relied on a set of rules, which is outdated and often impractical; and 2) it incorrectly transcribe certain types of words. Regarding limitation 1), rule-based methods are impractical because they involve extensive hand-engineered procedures that are laborious to update. A concrete example of this would be adapting the system to the new EP orthography (following the 1990

Orthographic Agreement). Regarding limitation 2), this approach incorrectly transcribed some mute consonants (e.g. *exce**p**ção*), and acronyms (e.g. *SIC*, *ONU*) as voiced consonants and spelled acronyms (siglæ), respectively.

3.3.2.1 Phoneme list reduction

Table 3.3 specifies all the symbols that either were removed from the phoneme list, or that replace preexisting symbols in the list. SAMPA was employed to decide which symbols to replace/remove from the phoneme list as follows: 1) IPA symbols were grouped based on their SAMPA equivalent; 2) The most frequent/common symbol from each group served as replacement for the remaining.

Some of the IPA phonemes displayed on table 3.3 only occurred in rare or specific circumstances. This is the case for: 1) symbols ʌ, ɪ, ɒ, ð, θ, which only occurred in foreign words — the nativized pronunciation was considered for these phonemes when replacing them in foreign words; 2) all phonemes that include the palatalized diacritic, (ʲ), only found in acronyms and spelled acronyms (siglæ); and 3) all symbols including the length suprasegmental, (ː). The latter case is detailed in table 3.4. Finally, table 3.5 summarizes all substitutions or deletions of phonemes in the list.

Table 3.3: List of all the phonemes that are involved in the phoneme list reduction.

Phonemes			Examples	
Type	**IPA**	**SAMPA**	**Text transcription**	**IPA (eSpeak)**
Vowels	a	a	“coloc**a**r”	kuluk**aɹ**
	ɑ		“vit**a**l”	vit**ɑ**l
	ɐ	6	“escol**a**s”	ʃkɔl**ɐ**ʃ
	ʌ		“s**u**bholding”	s**ʌ**bhəʊldɪŋ
	ə	–	“Portug**a**l”	pur**ə**tugɑl
	i	i	“d**i**sposto”	d**i**ʃpoʃtʊ
	ɪ		“**E**l**i**zabeth”	**ɪ**l**ɪ**zəbəθ
	ɔ	O	“n**o**va”	n**ɔ**vɐ
	ɒ		“sh**o**pping”	ʃ**ɔ**pɪŋ
	u	u	“**u**ma”	**u**mɐ
	ʊ	u/w	“privad**o**”	pɹivad**ʊ**
Palatalized Diacritic (ʲ)	fʲ	f	“**FF**S”	ɛf**ʲ**ɛf**ʲ** ɛs
	lʲ	l	“K**L**M”	kɐpɐ ɛl**ʲ** ɛm
	mʲ	m	“ST**M**L”	ɛs te ɛm**ʲ** ɛl
	nʲ	n	“G**N**R”	ʒe ɛn**ʲ** ɛr
	rʲ	r	“I**R**S”	i ɛr**ʲ** ɛs
	sʲ	s	“T**S**F”	te ɛs**ʲ** ɛf
Length Suprasegmental (ː)	ɑː	a	“ap**a**rtheid”	ɐp**ɑː**θeɪd
	aː		“livre **à** lei”	livɹɨ **aː** leɪ
	ɐː		“**à**s eleições”	**ɐː**z eleɪsõjʃ
	iː	i	“l**ea**sing”	l**iː**sɪŋ
	ɔː	O	“th**ou**ght”	θ**ɔː**t
	uː	u	“vie**w**”	vj**uː**
	tː	t	“etc.” (etcé**t**era)	etsɛ**tː**ɾɐ
Simple consonants	ɾ	r	“trajectó**r**ia”	tɹɐʒɛtɔ**ɾ**jɐ
	ɹ		“p**r**ivado”	p**ɹ**ivadʊ
	r		“Spo**r**ting”	spɔ**r**ting
	ð	d	“**th**e”	**ð**ə
	θ	s	“Commonweal**th**”	kɒmənwɛl**θ**

Table 3.4: List of cases in which suprasegmentals occur.

Phoneme	Particularity
ɑː, iː, ɔː, uː	Only found in phonetic transcriptions of foreign words
aː	Only found in the phonetic transcription of the word *à*
ɐː	Only found in the phonetic transcription of the word *às*
tː	Only found in the phonetic transcription of the abbreviation *etc.*

Table 3.5: Phoneme replacement/removal. Phonemes on top (first row) were replaced with the phonemes on the bottom (second row).

ɑ	ɑː	aː	ɐː	ʌ	ɪ	iː	ɒ	ɔː	ʊ	uː	fʲ	lʲ	mʲ	nʲ	sʲ	ɹ	r	ɾʲ	tː	ð	θ	ə
a				ɐ	i		ɔ		u		f	l	m	n	s	ɾ			t	d	s	removed

3.3.3 Sequence-to-Sequence GtoP toolkit

This GtoP approach relies on the Transformer architecture [56], thus only using attention mechanisms to establish the mapping between text inputs and phonetic sequence outputs. The training process of the model relies on a dictionary of words and their phonetic transcriptions. The main goal of using this approach was to solve the limitations of the rule-based system detailed in section 3.3.2. Since this toolkit employs a dictionary of words for training, it does not depend on any specific language or phonetic alphabet — it follows the language, orthography, and phonetic notation specified in the dictionary. Hence, it can be trained for EP, while employing a more practical and concise notation, like SAMPA.

The model for EP was trained with a dictionary comprising 60 700 entries (the word orthography and the corresponding phonetic transcription per line). Phonetic transcriptions followed the SAMPA notation, including primary stress. With this GtoP approach, the phoneme list comprised 50 different symbols, excluding punctuation. The phoneme list included more symbols than the SAMPA alphabet because each stressed vowel was counted as a different symbol.

Despite being easily trainable and efficient with individual words, this toolkit had several drawbacks: 1) it did not perform any text normalization; 2) it processed words separately, therefore, sentences were incorrectly converted at word boundaries; and 3) it did not incorporate any mechanism that allowed to distinguish homographs, such as part-of-speech (PoS) tagging. The lack of a text normalization stage made the toolkit inoperable for TTS — the system could only process lower case letters, as the word dictionary only contained these, and could not translate several one-letter words, such as definite articles *o* (pronounced /u/), and *a* (pronounced /6/), which were incorrectly transcribed as /O/, and /a/, respectively. To generate reasonable phonetic transcriptions additional processing stages were added to include punctuation signs in phonetic transcriptions, and detect acronyms (composed of upper case words) in input sentences. Furthermore, the word dictionary had to be adapted: instances of alternative pronunciations, that were only present in very specific contexts (e.g. the word *vi* pronounced /vi/, or alternatively, /s"6jStu/, the roman numeral *sexto*), were removed from the dictionary to avoid ambiguity. In what concerns word–boundary errors, the most common sandhi rules were manually implemented on top of the output phonetic sequences. Although including additional processing stages improved the quality of phonetic transcriptions, it posed a laborious and yet incomplete task — it did not contemplate several important cases, namely homographs, abbreviations, digits and symbols. As such, a third GtoP approach (detailed in section 3.3.4) was brought in to answer the problems raised by the prior two procedures.

3.3.4 Festival-based GtoP

Unlike the neural-based approach previously described in section 3.3.3, this technique follows the Festival framework, relying on a set of classification trees for GtoP conversion [55]. Phonetic transcriptions followed the SAMPA notation, and additional symbols were employed to represent foreign phones – the symbol "H", for example, was used to represent the sound of letter H in foreign words like "**H**ollywood", "Man**h**attan" or "**H**ezbollah"). In total, the phoneme list obtained with this approach was

composed of 44 symbols excluding punctuation.

Regarding punctuation, this approach did not discern between different signs, hence converting every sign to the symbol "#" in phonetic transcriptions. To include punctuation in the phonetic transcriptions of each sentence, the signs and their order of appearance were saved to a list. Then, each sign in the list replaced each occurrence of the symbol "#" in the phonetic transcription in the same order.

3.3.5 Non-standard words and homographs

From all three GtoP approaches, the Festival-based is the one which deals best with non-standard words. The existence of an addenda allows to specify the pronunciation of such words and override the output the system would normally produce [55]. Therefore, expressions such as *D. João I* or *S. António* are correctly converted to /do~ Zw6~w~ prim6jru/ and /s6~tw 6~tOnju/, respectively[2]. Likewise, this approach can correctly transcribe digits and date formats.

Although pronunciations of non-standard words could also be added to eSpeak, its current implementation for EP only converts a limited number of abbreviations and acronyms correctly. It can also transcribe digits, but it does not ensure a text normalization as extensive as the Festival-based GtoP.

Regarding homographs, upon phonetic symbol prediction, PoS tagging is employed both in the eSpeak backend as well as in the Festival-based approach. The usage of PoS tagging allows to correctly predict most of homograph cases.

The only type of homographs the GtoP modules fail to transcribe are homographs with the same PoS category (e.g. *Tenho s**e**de. Hoje estive na s**e**de.*). When the PoS categories differ (e.g. verb/noun as in *Eu almo**ç**o depois. Está na hora do almo**ç**o.*), both modules produce the correct transcription.

Given that the Festival-based approach was the most complete from the three methods detailed in this section, it was the preferred one for the multi-speaker TTS model. Besides being superior in terms of non-standard words processing, it employed SAMPA, a more concise phonetic alphabet than IPA, and a more practical notation to incorporate in a TTS system.

3.4 Audio preprocessing

According to section 3.1, the proposed model operates by linking two different systems: the Seq2Seq model and the neural vocoder. Although these systems operate jointly to perform TTS, they are trained separately. Given that acoustic features, in the form of Mel-spectrograms, are what bridges these systems together, it is essential to ensure that both utilize acoustic features of the same type. To do so, it is necessary that audio data undergoes the same digital signal processing (DSP) stages in the two systems.

Before Mel-spectrogram extraction, audio signals were normalized and underwent a pre-emphasis filter, a common technique in the field of speech analysis. Speech signals register predominantly low

[2]The Festival-based approach did not include stress marks in phonetic transcriptions

values of frequency, usually below 4 kHz [57]. Pre-emphasis is employed to enhance high-frequency components of an audio signal and to reduce its high spectral dynamic range [58].

Mel-spectrogram extraction was performed with the python package *librosa* [59]. For each audio signal, this process unfolded in two steps: 1) computing the magnitude of the short-time Fourier transform (STFT) of the audio signal, which corresponds to a linear spectrogram; 2) converting the linear spectrogram to a Mel-spectrogram.

Table 3.6 specifies the DSP parameters employed at the audio preprocessing stage. For computing the STFT, the parameters `n_fft`, `win_length`, and `hop_length` were used. The STFT works by taking discrete Fourier transforms (DFT) over several, overlapping windowed sections of the signal. These sections were obtained by applying a Hanning window of size `win_length` to the audio frame, and then padding the windowed signal with zeros to match the frame size defined by `n_fft`. A window shift defined by `hop_length` was utilized. The zero-padding added to the windowed signal does not increase the information in the input signal, but results in higher frequency resolution. The values selected for the aforementioned DSP parameters correspond to a 50 ms window length, 12.5 ms window shift, and 2048-point Fourier transform, as in the Tacotron original implementation [34].

Table 3.6: DSP parameters used in audio preprocessing.

Parameter name	**Value**	**Description**
`sample_rate` [Hz]	16000	Sampling rate
`n_fft, num_fft`	2048	Number of samples in a frame
`win_length`	800	Window size (in samples)
`hop_length`	200	Window shift (in samples)
`n_mels, num_mels`	80	Number of Mel bands
`fmin` [Hz]	50	Minimum frequency
`fmax` [Hz]	8000	Maximum frequency
`preemph`	0.97	Pre-emphasis filter coefficient

Mel-spectrograms comprised 80 Mel-bands, a minimum frequency of 50 Hz, and a default maximum frequency of 8 kHz, that is, half of the sampling frequency. Energy values were converted to dB as defined in expression 3.1, where x stands for the energy value to be converted.

$$x_{\text{dB}} = 20 \cdot \log_{10}(x) \tag{3.1}$$

3.4.1 Leading and trailing silence removal

The existence of long sections of leading/trailing silence in audio signals may prevent the model from generating meaningful encoder-decoder alignments (encoder-decoder alignments are analyzed in detail in section 3.5.3). In fact, this was witnessed in the first attempts of training of the multi-speaker Seq2Seq model with the BD-PUBLICO corpus. Several BD-PUBLICO audio recordings had large sections of leading and trailing silence, in many occasions longer than two seconds. During training, the only type of silence the model is capable of recognizing are pauses, which are represented by punctuation marks in the text input and its corresponding phonetic transcriptions. As such, if the model stumbles upon

large sections of leading silence, which are unidentified in the phonetic transcriptions, it may incorrectly misinterpret silence for the first phonemes that follow. Likewise, too long sections of trailing silence may hamper the task of predicting the end of the signal, or even mapping the adequate amount of pause to punctuation signs.

As mentioned in section 3.1, very long utterances hinder the decoding task. Therefore, removing leading/trailing silence not only contributes for better linguistic-acoustic mapping, but also allows for more data to be used, which is convenient. That way, several utterances that were originally too long to be utilized, can be included for training after leading/trailing silence removal.

Two approaches were used with the aim of removing leading/trailing silence sections from audio recordings. First, a fixed–threshold approach was employed with *librosa* [59]. In this approach, a threshold was defined in dB, and values under this threshold were considered as silence. However, this method was primitive and had several limitations, since the silence threshold was not the same for different utterances. Some examples of why using a fixed threshold is not a good solution could be the following: 1) distance of the speaker to the microphone could vary, therefore, the perceived sound intensity would vary also; 2) the loudness of speech from one speaker to another could also vary; 3) different sounds other than voice could potentially register values above the threshold, and thus be incorrectly considered as speech. As such, when the threshold was too high for a given recording, the resultant audio signal skipped speech sections, and when the threshold was too low, the leading/trailing silence remained present.

In order to tackle the limitations of the fixed–threshold approach, a more robust method was used, which consisted in a toolkit for speech diarisation [60] (repository 4 in table 3.1). One of the modules of this toolkit is voice activity detection (VAD). This module was employed solely to capture the audio samples at which speech started and ended. Although the model occasionally cut off small sections of speech, it removed leading and trailing silence much more accurately, since the reference was not merely sound intensity, but voice activity instead. To solve the minor issue of cutting off small speech sections, a margin was added. That way, when the VAD model predicted that speech started/ended at a given sample, the audio file was cutoff a number of samples before the predicted start-of-speech sample, and a number of samples after the predicted end-of-speech sample.

3.5 Seq2Seq regressive model — pre-train implementation

This section details the training process of the multi-speaker Seq2Seq model. It starts by explaining how data was selected and why some of the data had to be excluded. Then, it specifies the most relevant hyperparameters during training and how their values were selected. At last, it describes how the training process was assessed, and what factors led to choosing an appropriate training checkpoint.

3.5.1 Data selection process

In the first attempts of training the model, OOM errors occurred as a consequence of two factors: too long phoneme lists, as mentioned in section 3.3, and/or too long utterances. A threshold was employed to establish a safe utterance length that prevented OOM errors from occurring. Naturally, the shorter the phoneme list is, the larger the utterance length threshold can be.

The utterance-length threshold is established as a number of frames. In this case, a frame is considered as a column of a Mel-spectrogram. Following the spectrogam extraction process detailed in section 3.4, and the respective DSP parameters, each column/frame of a Mel-spectrogram comprises a number of samples corresponding to the window shift, that is, 200 samples (12.5 ms). An utterance-length threshold of 800 frames (10 s) was established, thus, utterances longer than 10 seconds were excluded from training.

Depending on utterance length, the amount of data used for training, validation, and testing varied from one speaker to another. Before excluding long utterances, data was split as follows: 5 utterances for testing (for comparison purposes), 10 utterances for validation, and the remaining for training. After this stage, utterances longer than 10 seconds were excluded from training and validation sets in order to preserve the proportion of data in each set, as much as possible. Otherwise, the distribution of data among training and validation sets could occasionally be unbalanced, since for some speakers a large number of utterances was excluded. In the case of speaker *o041*, only 30 utterances were used for training, while for speaker *o015* 60 utterances were included in the training set. Using the same number of utterances for validation in both cases would not be appropriate, as the ratio of training utterances to validation utterances would substantially differ. Hence, the utterance-length threshold was applied both to training and validation sets. Since OOM errors did not occur at inference time, even for the longest test utterances, no utterances were excluded from the test set.

3.5.2 Hyperparameters

Table 3.7 details the most important hyperparameters upon training the Seq2Seq model. Hyperparameters regarding the model structures follow the original implementation [13].

3.5.2.1 Experiment parameters

Experiment hyperparameter `epochs` was set to 120 as this number was more than enough for the model to converge. The number of necessary epochs for the model to converge varied greatly depending on several factors, namely the data, but more importantly, the text preprocessing stage. Different GtoP approaches generated different phonetic transcriptions, either because of distinct phonetic alphabets, or errors in phonetic transcriptions, mostly in non-standard words or at word boundaries. Furthermore, the inclusion of punctuation in phonetic transcriptions greatly decreased the number of epochs necessary for the model to converge. Using the same GtoP backend, in this case eSpeak, the model converged in 13 200 iterations (approximately 93 epochs) without punctuation, and in 8250 iterations (approximately 58 epochs) with punctuation.

Hyperparameter `iters_per_checkpoint` was set to 150, so that between each checkpoint the model was trained for at least one epoch.

Table 3.7: Multi-speaker Seq2Seq pre-train stage hyperparameters.

Type	Parameter name	Value	Description
Experiment parameters	`epochs`	120	Number of training epochs
	`iters_per_checkpoint`	150	Number of iterations per checkpoint
Data parameters	`training_list`	–	Path to file listing data for training
	`validation_list`	–	Path to file listing data for validation
	`test_list`	–	Path to file listing data for testing
	`n_mel_channels`	80	Number of Mel bands
	`n_symbols`	47	Number of symbols in the phoneme list, 47 using Festival, 54 using Seq2Seq toolkit, and 53 using eSpeak
	`n_speakers`	100	Number of speakers used for training
	`predict_spectrogram`	False	Set as false to use Mel spectrograms instead of linear spectrograms
Training parameters	`learning_rate`	1e-3	Learning rate
	`weight_decay`	1e-6	Weight decay coefficient
	`grad_clip_thresh`	5.0	Gradient norms above this value are clipped
	`batch_size`	32	Batch size
	`warmup`	70	Number of epochs with constant LR
	`decay_rate`	0.5	LR penalizing factor
	`decay_every`	7	LR decays every `decay_every` epochs
	`contrastive_loss_w`	30.0	Contrastive loss weighting factor
	`speaker_encoder_loss_w`	1.0	Speaker encoder loss weighting factor
	`text_classifier_loss_w`	1.0	Text classifier loss weighting factor
	`speaker_adversarial_loss_w`	20.0	Adversarial loss weighting factor
	`speaker_classifier_loss_w`	0.1	Auxiliary classifier loss weighting factor
	`ce_loss`	False	Set the adversarial loss as the symmetric of the auxiliary classifier loss

3.5.2.2 Data parameters

Regarding data parameters, three paths had to be specified, each one corresponding to a file listing training, validation, and testing data, respectively. Each one of this files contained in each line the path of the specific data file (Mel-spectrogram/phonetic transcription).

Depending on the GtoP backend, the parameter `n_symbols` varied and had to be set accordingly. For large and phonetically extensive corpora, the number of symbols in the phoneme list most likely contemplates all phonemes of the language that is used. Pre-training the model with phonetically poor data may result in overfitting. In this scenario, training data would only comprise a narrow variety of phonetic sequences, thus the model would perform poorly for new/unseen sequences of phonemes.

3.5.2.3 Training parameters

The Adam optimizer [61] was employed during training. L2 regularization with weight 10^{-6} (given by the parameter `weight_decay`) was used. The learning rate (LR) was fixed at 10^{-3} during the first 70 epochs, after which it decayed by 50% (`decay_rate` parameter) every seven epochs (`decay_every` parameter). This LR decay strategy is the same as in the fine-tuning stage of the default implementation. Freezing the LR for at least 70 epochs proved to be useful in the convergence of the model. Likewise, decreasing it towards the end of training allowed for slight improvement.

Upon the occurrence of OOM errors, before adopting the procedure of phoneme list reduction (described in section 3.3), the first approach was to reduce the batch size from 32 to 16, as an attempt to decrease the memory usage during training. Although memory usage was effectively reduced, decreasing the batch size prevented the model from generating meaningful encoder-decoder alignments, which led to poor results. Thus, the original batch size value was kept.

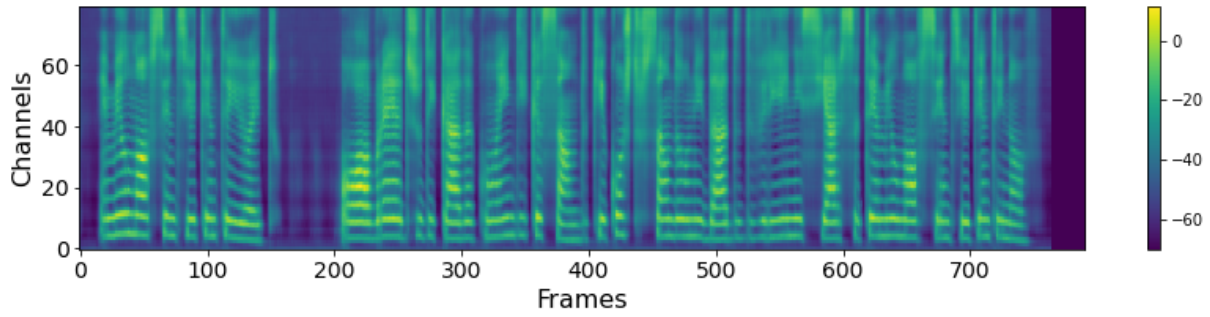
At last, the different loss weighting factors were kept as in the original implementation of the model.

3.5.3 Pre-training process

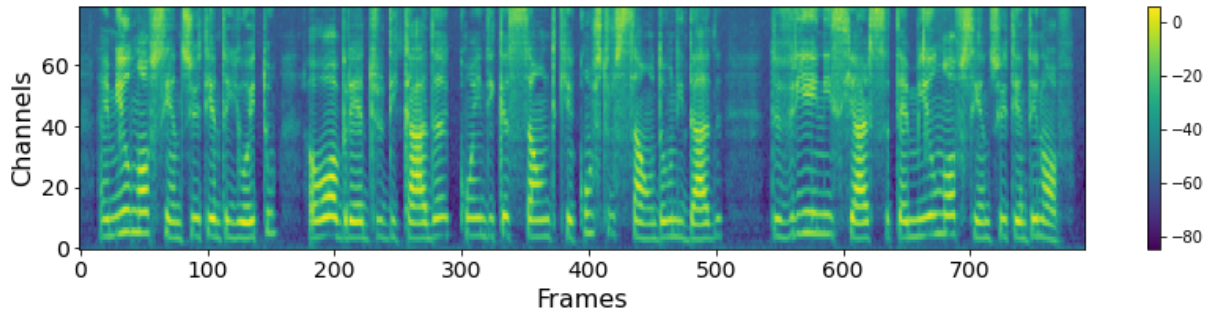
The version of the model used as reference employed the Festival-based GtoP, as it provided the best results regarding text preprocessing. Punctuation was included in phonetic transcriptions. The model was trained for 8250 iterations (approximately 58 epochs), at a constant learning rate (10^{-3}).

The present model aims to solve the regression problem of predicting acoustic representations, which is a complex task. Although training and validation losses are important factors to be considered when evaluating the training process of DL models, one should not entirely rely on loss to infer if a given model is well trained or not. There is no reference loss value to serve as target, because different data originates distinct optimal loss values. Therefore, it is more meaningful to analyze the evolution of training and validation losses throughout the training process, rather than just comparing their values at a given iteration. In the present model, the training loss had an initial value of 9.994 and decreased steadily for 5000 iterations, after which it stabilized between 0.9217 and 1.969. The validation loss had an initial value of 11.18, decreased constantly for 2000 iterations, and stabilized between 1.628 and 2.008.

To further assess how well the training process unfolded, encoder-decoder alignment plots were analyzed. Also, for a given test utterance, the original and synthetic spectral representations were compared. Given that the training corpus comprises utterances of neutral prosody, it was adequate to compare spectral representations. Figure 3.2 illustrates the predicted and the original Mel-spectrograms obtained for the same test sentence. When analyzing both spectral representations, one difference stands out the most: the duration and placement of pauses. In the depicted spectral representations, pauses can be identified by predominantly dark-blue colored frames. As referred in section 3.3, pauses are represented by punctuation marks in phonetic transcriptions. Since pauses do not have fixed lengths, the same symbol, a comma for example, may represent longer or shorter pauses. Thus, for a given test sentence, the model may predict slightly longer or shorter pauses than those in the original utterance.



(a) Predicted Mel-spectrogram for 8250 iterations



(b) Mel-spectrogram extracted from the original utterance

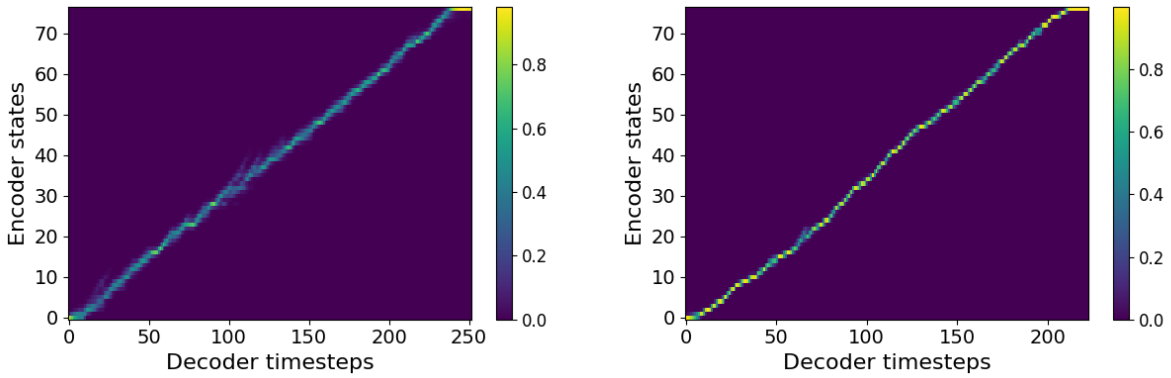
Figure 3.2: Predicted and original Mel-spectrograms for the following test sentence: *Em mil novecentos e oitenta e oito, a obra é adjudicada com a indicação de que a construção da unidade deveria iniciar-se até mil novecentos e oitenta e nove.*

3.5.3.1 Encoder-decoder alignment plot

Encoder-decoder alignment plots illustrate how well Seq2Seq models map an input sequence of one type, to an output sequence of another type. In this case, the model aims to map a phonetic input sequence to an acoustic output sequence. Each encoder state corresponds to an element (phoneme) of the input sequence, and decoder timesteps corresponds to acoustic frames. Each decoding timestep can be interpreted as an array of weights for each encoder state. These weights add up to one, and their magnitude is represented by a color (the larger the magnitude, the brighter the color).

A good alignment means that at each decoding timestep most of the weight is focused at a specific encoder state such that a diagonal line is formed. This diagonal line indicates that the input sequence order is preserved in the output. Hence, the brighter and less blurry the diagonal line is, the better is the alignment.

Figure 3.3 illustrates the encoder-decoder alignments obtained for the same test sentence, at different checkpoints. It is clear that at 750 iterations, the model still has not trained enough: although the diagonal line is noticeable, it is still blurry, meaning that at each decoding timestep several encoder states have similar weights, and that the model could not distinctively select an encoder state. At checkpoint 8250, the alignment plot is brighter and more focused, suggesting that the model unambiguously selected an encoder state for each decoding timestep, while preserving the input sequence order.



(a) Obtained alignment for 750 training iterations

(b) Obtained alignment for 8250 training iterations

Figure 3.3: Encoder-decoder alignment plots for the same test sentence at different checkpoints.

3.6 Neural vocoder — implementation

The universal vocoding procedure (repository 1 in table 3.1), which serves as the waveform generation step in the proposed model, has some differences in regard to the original universal vocoding approach [26] presented by Lorenzo-Trueba and colleagues (2019). The former produces 9-bit mu-law audio, sampled at 16 kHz, while the latter generates 10-bit mu-law audio, sampled at 24 kHz. Furthermore, the one-hot audio vector — employed in the autoregressive side of the original model — is replaced with an embedding layer in the default implementation adopted for the proposed model.

Given that the vocoder default implementation was already tuned to predict 16 kHz audio samples at inference time, training the model was a straightforward procedure. Except for silence trimming, the same audio preprocessing stages as in the Seq2Seq regressive model were ensured. This enabled using acoustic outputs predicted by the Seq2Seq model as inputs to the neural vocoder.

3.6.1 Model training

The same hyperparameters were employed as in the base implementation of the model. These are specified in table 3.8. The model was trained for 100 000 iterations, starting at a learning rate of 4×10^{-4} that decayed by 50% every 20 000 iterations. As the training process unfolded, synthesis quality was perceptually evaluated from generated audio samples.

Table 3.8: Neural Vocoder hyperparameters.

Parameter name	Value	Description
conditioning_channels	128	Conditioning network GRU hidden size
embedding_dim	256	Embedding layer dimension
rnn_channels	896	Autoregressive network GRU hidden size
fc_channels	512	Output dimension of the affine layers (corresponds to the number of quantization levels)
learning_rate	4e-4	Learning rate
step_size	20000	Period of learning rate decay
gamma	0.5	Multiplicative factor of learning rate decay
batch_size	32	Batch size
checkpoint_interval	20000	Number of iterations per checkpoint
num_steps	100000	Number of training iterations
sample_frames	40	Number of Mel-spectrogram frames that are fed to the conditioning network
audio_slice_frames	8	Number of conditioning network output middle frames that are upsampled and used to condition the autoregressive network GRU

3.7 Final remarks

The topics presented in this chapter allowed to: 1) identify the basic structure of a proposed multi-speaker TTS model, and how its main components interconnect; 2) characterize the corpora designated to train the model and determine which traits inherent to these played an important role in the system's performance; 3) detail the text and audio preprocessing steps in context of the subsequent training stages; and 4) understand how each training stage unfolded and was assessed.

Regarding text preprocessing, the model benefited from the inclusion of punctuation, from a more concise phonetic notation (SAMPA rather than IPA), and from a more comprehensive GtoP tool (Festival-based GtoP). The latter, in particular, was critical to ensure the model could synthesize homographs and non-standard words correctly. Concerning audio preprocessing, the removal of leading and trailing silence from audio files was fundamental to ensure convergence of the Seq2Seq regressive model.

At this point, the model underwent the pre-train stage, conditioned on prosodically neutral data from 100 adult speakers. The following chapter will address speaker adaptation (fine-tune stage) for data of the same type — neutral prosody and adult speakers — and of different type — expressive prosody, and children and adolescent speakers. Performance will be evaluated for the former case.

Chapter 4

Experiments

The starting point for the speaker adaptation work described in this chapter is a TTS system for European Portuguese, pre-trained on the BD-PUBLICO dataset, that could synthesize any prosodically neutral sentence using the voices of 100 young adult speakers. The experiments that were performed regarding speaker adaptation (see Section 4.1) include: the details of each fine-tune configuration, their differences regarding training data, and an analysis of their spectral outputs. Additionally, an assessment on the proposed model's performance is detailed in Section 4.2.

4.1 Experimental setup

As stated in the previous chapter (see Section 3.1), the main reason for employing a universal vocoder is to reduce the amount of training stages during the speaker adaptation stage. As such, speaker adaptation only involves fine-tuning the Seq2Seq regressive model, instead of the whole system. The model was fine-tuned for three distinct settings: 1) two adult speakers; 2) two adolescent speakers; and 3) two child speakers. For each setting, the pair of speakers comprised one male and one female voice. All configurations employed non-parallel data.

The adult–speaker fine-tune setting is the standard implementation of the proposed model, since the characteristics of data (namely the prosody, and type of sentences) and the speakers' traits (namely pitch values) are in line with the speech corpus employed during the pre-train stage. Fine-tuning employed 300 utterances, of which 33 for validation, for each speaker *sp_01* and *sp_02* of the BDFALA corpus. Synthetic speech was assessed in terms of naturalness, voice similarity, and intelligibility (see Section 4.2) for both speakers. Additionally, the same setting with reduced data — 20 utterances per speaker, of which four for validation — was tested for naturalness and similarity.

The fine-tune configurations for child and adolescent voices were addressed separately and did not take part in subjective assessments, given that synthetic speech was perceptually worse than for the adult–speaker setting. Characteristics of both the data and the speakers were substantially different from those used in pre-training, hence, these configurations were analyzed as out-of-domain (OOD) scenarios (see Section 4.1.1). In each of these settings, the amount of fine-tuning data was considerably

Table 4.1: Amount of fine-tuning data per configuration.

Configuration	Speaker	# Utterances/files	Total Duration [hh:mm:ss]
Adults-standard	*sp_01*	300	00:51:36
	sp_02	300	
Adolescents	*sp_03*	34	00:03:10
	sp_04	33	
Children	*sp_11*	19	00:02:44
	sp_36	21	
Adults-reduced	*sp_01*	20	00:03:18
	sp_02	20	

lower than in the standard adult–speaker setting. The models for each child voice were fine-tuned with 19 and 21 utterances, of which four for validation, taken from the EUROM.1 corpus, for speakers *sp_11* and *sp_36*, respectively. The voice models of adolescent speakers were fine-tuned with 33 and 34 utterances, of which eight for validation, taken from a subset of BDFALA, for speakers *sp_04* and *sp_03*, respectively.

4.1.1 Differences in training data: mean pitch and prosody

The most notable differences among the data of each fine-tuning configuration in comparison to pre-training data are the mean pitch values of the speakers and the prosodic contours of the utterances they recorded. The differences in pitch were, unsurprisingly, most prominent in the voices of children, since these had the highest pitch. Prosodic contours were analyzed according to two factors: 1) the standard deviation (SD) of pitch for each speaker; and 2) the type of sentences — declarative, interrogative, or exclamatory — employed in each training setting.

Table 4.2 specifies the pitch values of pre-train speakers and each fine-tune speaker. When interpreting the data in table 4.2 referring to BD-PUBLICO, one must bear in mind that BD-PUBLICO comprises 100 speakers, therefore it is not practical to display the pitch data of all its speakers in the table — instead, it is more sensible to specify the range of values for each column of the table.

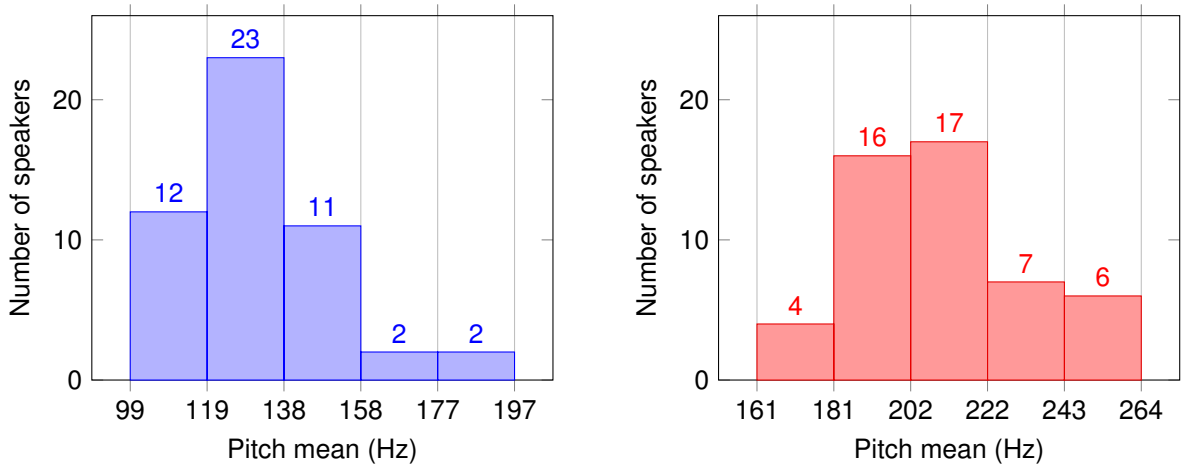
Table 4.2: Pitch values for EP speakers in different age groups. Values are rounded to the nearest unit.

Training stage	Speech corpus	Speaker	Age	Pitch [Hz]			
				Min.	Max.	Mean	SD
Pre-train	BD-PUBLICO	*o000-o049* (M)	19-28	72-115	141-399	99-197	6-39
		o050-o099 (F)		71-99	250-399	161-264	15-38
Fine-tune	BDFALA	*sp_02* (M)	35	75	221	114	15
		sp_01 (F)	41	90	389	200	33
	BDFALA02	*sp_03* (M)	14	79	392	165	38
		sp_04 (F)	13	92	398	254	49
	EUROM.1	*sp_11* (M)	10	132	398	291	52
		sp_36 (F)	9	132	398	272	56

Regarding mean pitch values, both children (*sp_11* and *sp_36*) standout with higher values than the

remaining fine-tune speakers, and more importantly, pre-train speakers. Children voices are clearly an OOD scenario in terms of mean pitch as these yield a mean pitch outside the range of values of pre-train data.

Histograms displayed in figure 4.1 complement the data of table 4.2 by illustrating how mean pitch values are distributed across the pre-train corpus. It is clear that the majority of male speakers registered values under 138 Hz, and only four out of 50 speakers had mean pitch values of 158 Hz or higher. Similarly, only six female speakers recorded a mean pitch equal or higher than 243 Hz. Hence, in terms of mean pitch, adolescent speakers (*sp_03* and *sp_04*) only were similar to a small minority of voices employed during pre-training.

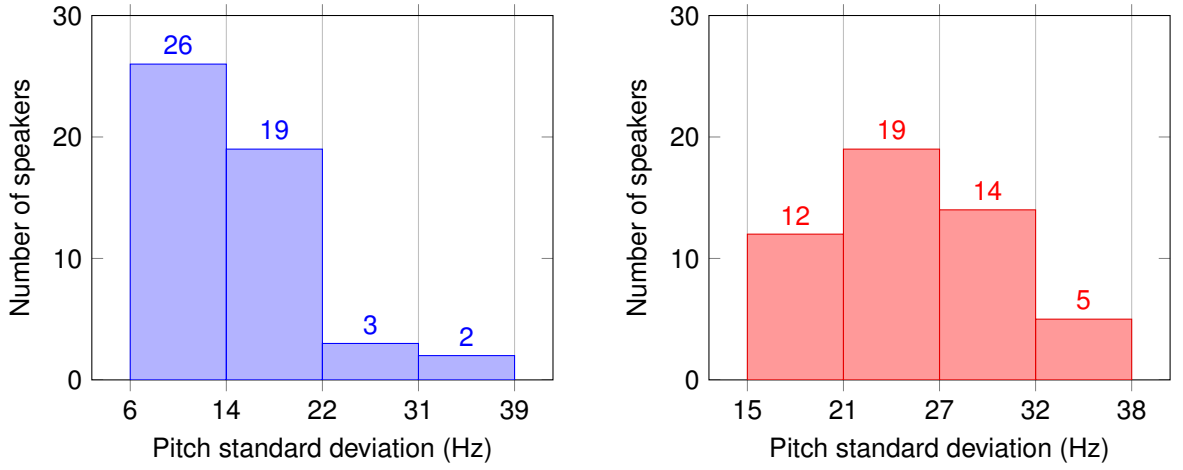


(a) Male speakers

(b) Female speakers

Figure 4.1: Histograms of pitch mean values of speakers in the BD-PUBLICO corpus.

Histograms in figure 4.2 specify the SD values across the BD-PUBLICO corpus to complement table 4.2. Although SD values for adolescents seem relatively close to those of BD-PUBLICO (for speaker *sp_03*, the pitch SD is even within the range of values for BD-PUBLICO voices), only two male and five female speakers from BD-PUBLICO achieved an SD higher than 31 Hz and 32 Hz, respectively.



(a) Male speakers

(b) Female speakers

Figure 4.2: Histograms of pitch SD values of all speakers in the BD-PUBLICO corpus.

Table 4.3 specifies the number of sentences of each type in the datasets employed in each training configuration.

In terms of sentence type, the adolescent–speaker fine-tune setting differs the most from the pre-train configuration, since over 80% of the sentences are either exclamatory or interrogative. Also, contrarily to pre-training data, each utterance and corresponding text transcription files of adolescent–speaker data often included more than one sentence — this is why the total number of sentences greatly exceeds the number of utterances employed for fine-tuning. Given these discrepancies, the adolescent–speaker fine-tune setting was also considered as an OOD scenario.

Although child and adult fine-tune settings employed similar data in terms of sentence type, the former setting presented more expressive prosody as it achieved the highest pitch SD values, as seen in table 4.2.

This section addressed the main differences in terms of pitch mean values and prosody among the data for each training configuration. From tables 4.2 and 4.3, one may infer that both the adolescent and child fine-tune configurations comprised expressive prosody as opposed to the remaining. From the four described fine-tune configurations, both adult–speaker settings (standard and reduced) are the most similar to pre-train data both in terms of pitch and sentence type.

Table 4.3: Type of sentences in the dataset of each training setting. Number of declarative, interrogative and exclamatory sentences in each dataset.

Training setting		**# Declarative**	**# Interrogative**	**# Exclamatory**	**TOTAL**
Pre-train		8021 (99.16%)	47 (0.58%)	21 (0.26%)	8089 (100%)
Fine-tune	adults-standard	527 (87.83%)	70 (11.67%)	3 (0.50%)	600 (100%)
	adults-reduced	34 (85.00%)	6 (15.00%)	—	40 (100%)
	adolescents	22 (19.64%)	25 (22.32%)	65 (58.04%)	112 (100%)
	children	34 (85.00%)	6 (15.00%)	—	40 (100%)

4.1.2 Seq2Seq regressive model — fine-tune implementation

Overall, the fine-tuning process was very similar to the pre-train stage. Fine-tuning contemplated the whole Seq2Seq regressive model except for the speaker encoder, as stated in the original paper [13].

For the fine-tune stage, the same hyperparameters were used as in the original implementation proposed by the authors. In comparison to the pre-train stage, only the batch size was changed from 32 to 8, the number of training epochs at constant LR was changed from 70 to 7, and weighting factors of speaker encoder and speaker adversarial losses were changed to 0 and 0.2, respectively. The remaining fine-tune stage hyperparameters are specified in annex A, table A.1.

Prior to fine-tuning the model, all data underwent the same text and audio preprocessing steps as before pre-training, except for the silence removal stage, as audio files did not include long sections of leading or trailing silence.

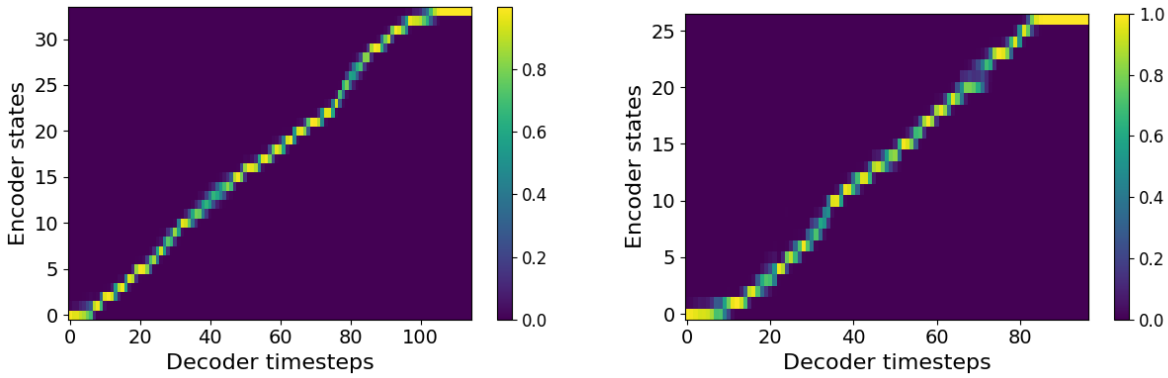
During the training process, trainable speaker embeddings were employed for each speaker. Each embedding was initialized with the average of speaker encoder outputs for all training utterances of

the speaker in question — it is clear that the fine-tuning process of the Seq2Seq regressive model follows the speaker adaptation framework, described in chapter 2, section 2.2.2. At inference, contrarily to the default fine-tune implementation, the proposed model directly employs the average of speaker encoder outputs. For all fine-tune configurations, no difference was perceived in synthetic speech when employing the trained embeddings instead of the average speaker encoder outputs. As such, for the sake of simplicity, we chose to solely use the pre-trained speaker encoder for embedding extraction at fine-tuning. However, upon different pre-train data, trainable embeddings may bring clear improvements to acoustic feature estimation.

Similarly to the pre-train stage, fine-tuning was assessed based on encoder-decoder alignment plots and Mel-spectrograms. For each voice, the alignments and spectral representations were generated from test sentences (unseen during training). To minimize prosodic differences that otherwise could standout in Mel-spectrograms, we selected declarative sentences without pauses.

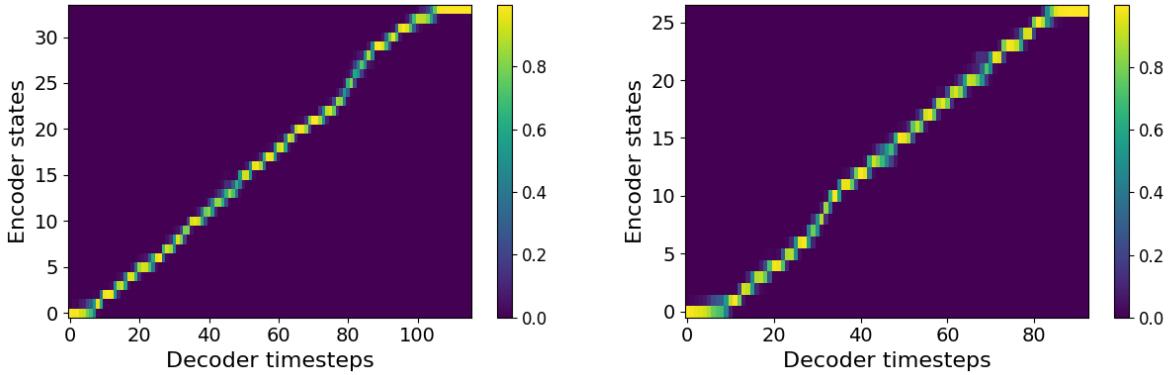
4.1.2.1 Adult–speaker configurations

Standard and reduced adult–speaker configurations were trained for 1600 iterations (approximately 23 epochs), and 200 iterations (50 epochs), respectively. Figure 4.3 illustrates the encoder-decoder alignments obtained for both configurations.



(a) Standard setting, speaker *sp_01*

(b) Standard setting, speaker *sp_02*



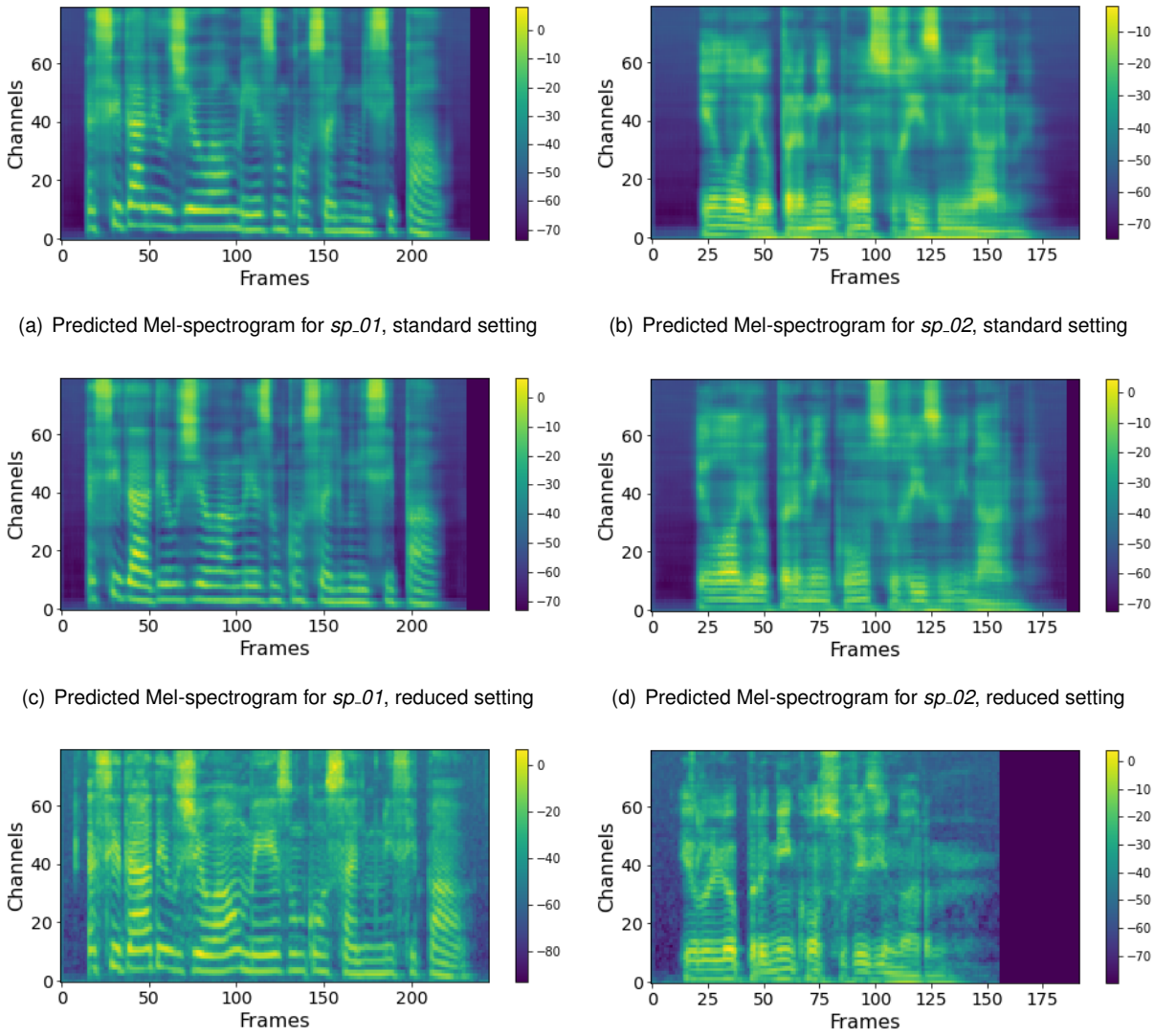
(c) Reduced setting, speaker *sp_01*

(d) Reduced setting, speaker *sp_02*

Figure 4.3: Encoder-decoder alignments for fine-tuning test sentences — adult speakers.

For each speaker, the same test sentences were employed in both configurations. For these sentences, despite the difference in amount of fine-tuning data, encoder-decoder alignments remained very similar.

Figure 4.4 illustrates the predicted and original Mel-spectrograms obtained from the same test sentences. Predicted Mel-spectrograms were similar for both configurations regardless of the amount of training data, suggesting that the Seq2Seq regressive model can produce reasonable results in a reduced data setting.



(a) Predicted Mel-spectrogram for *sp_01*, standard setting

(b) Predicted Mel-spectrogram for *sp_02*, standard setting

(c) Predicted Mel-spectrogram for *sp_01*, reduced setting

(d) Predicted Mel-spectrogram for *sp_02*, reduced setting

(e) Original Mel-spectrogram for *sp_01*

(f) Original Mel-spectrogram for *sp_02*

Figure 4.4: Mel-spectrograms for fine-tuning test sentences — adult speakers.

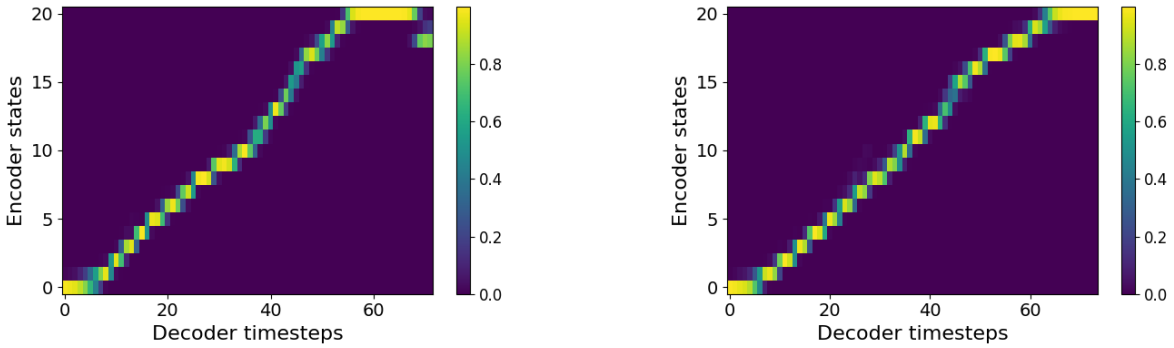
4.1.2.2 OOD configurations

Fine-tune configurations for adolescents — *sp_03* and *sp_04* — and children — *sp_11 and sp_36* — were trained for 300 iterations (approximately 47 epochs) and 184 iterations (46 epochs), respectively. Figures 4.5 and 4.6 depict the encoder-decoder alignments and Mel-spectrograms obtained for test

sentences of each configuration. Despite the differences in pitch and prosody in comparison to pre-train data, both OOD configurations were capable of generating reasonable alignments and spectral representations.

Adolescent–speaker configuration

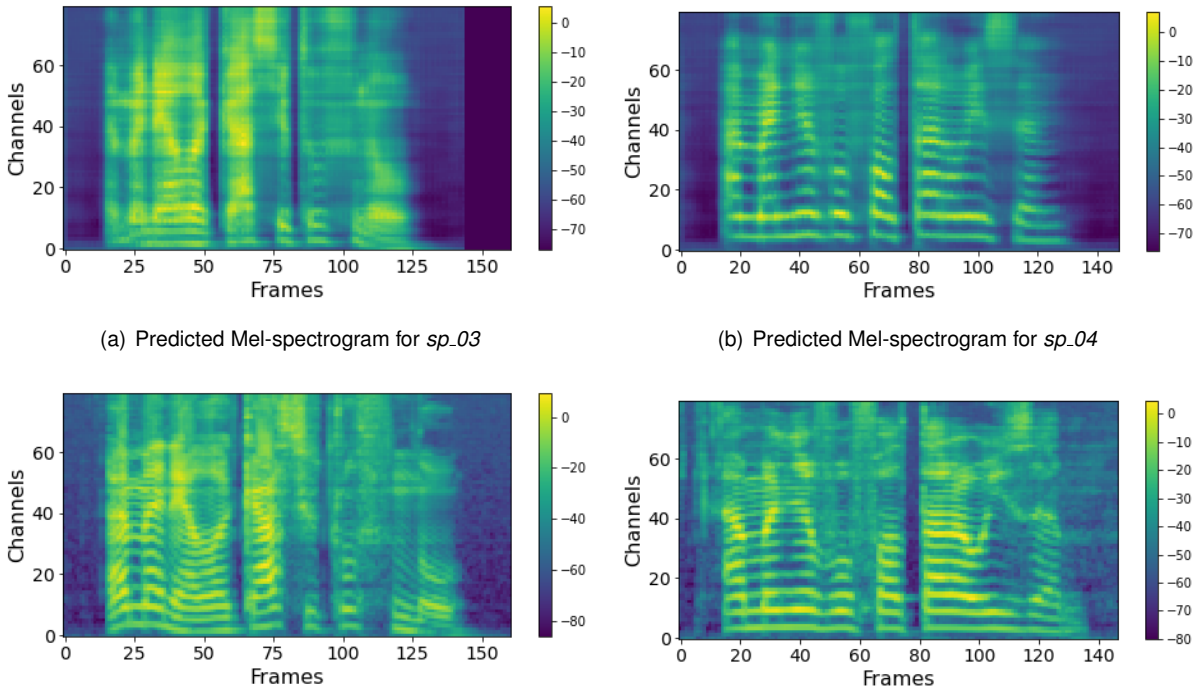
Regarding the adolescent–speaker configuration, one can notice that the predicted spectrogram for *sp_04* is sharper than *sp_03*'s in context of original spectral representations — for *sp_03*, contours can be clearly distinguished in the original spectrogram, up to higher frequencies at channels 40-60, which does not yield for the predicted spectrogram. Accordingly, synthetic speech quality was noticeably better for *sp_04* than for *sp_03* — *sp_03* synthetic speech samples sounded noisier.



(a) Alignment for test sentence of speaker *sp_03*

(b) Alignment for test sentence of speaker *sp_04*

Figure 4.5: Encoder-decoder alignments for fine-tuning test sentences — adolescent speakers.



(a) Predicted Mel-spectrogram for *sp_03*

(b) Predicted Mel-spectrogram for *sp_04*

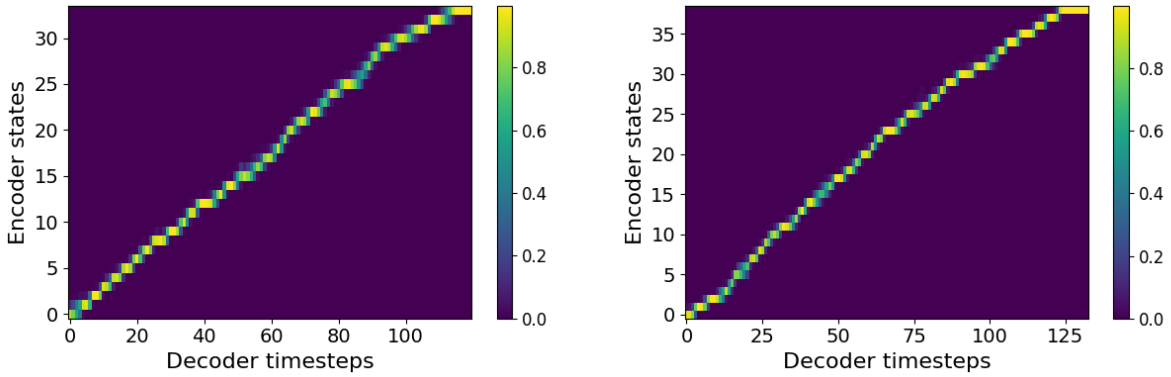
(c) Original Mel-spectrogram for *sp_03*

(d) Original Mel-spectrogram for *sp_04*

Figure 4.6: Mel-spectrograms for fine-tuning test sentences — adolescent speakers.

Child–speaker configuration

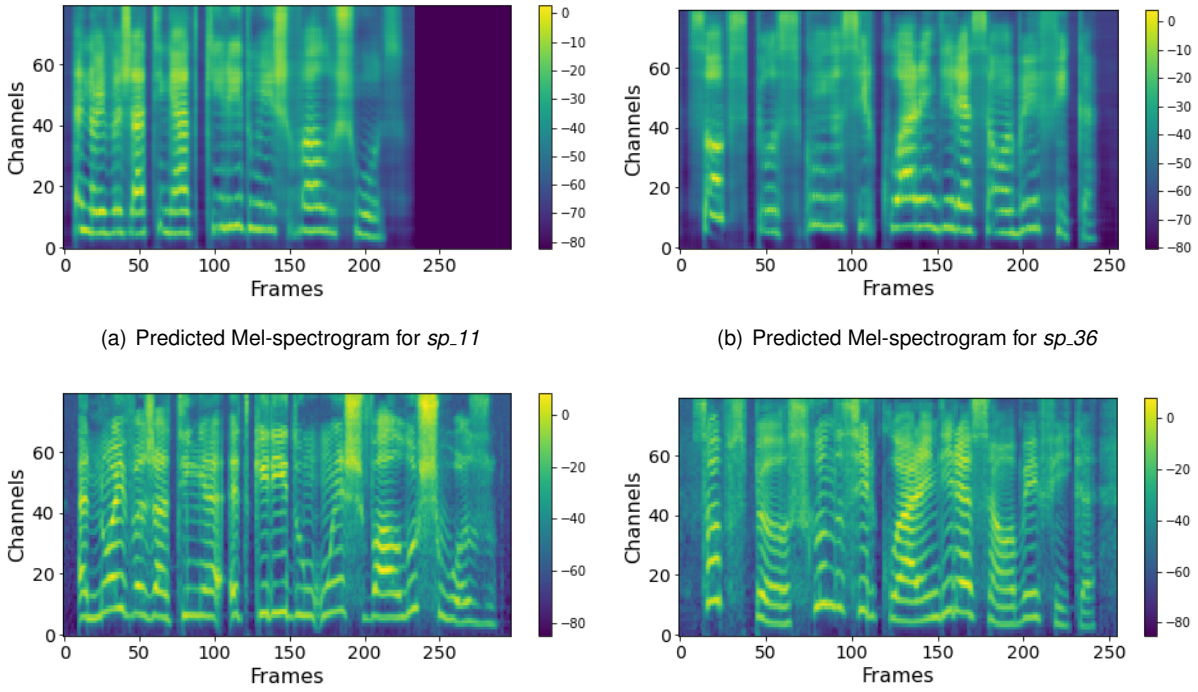
For the voice models of children, two differences stood out between predicted and original spectral representations: 1) The difference in the contours' shape — in original Mel-spectrograms contours have a more curved shape, covering a wider range of frequencies, while in predicted spectrograms contours display a distinctively flatter shape; 2) High-frequency contours that are clearly distinguishable in original Mel-spectrograms are not recognizable in predicted spectral representations.



(a) Alignment for test sentence of speaker *sp_11*

(b) Alignment for test sentence of speaker *sp_36*

Figure 4.7: Encoder-decoder alignments for fine-tuning test sentences — child speakers.



(a) Predicted Mel-spectrogram for *sp_11*

(b) Predicted Mel-spectrogram for *sp_36*

(c) Original Mel-spectrogram for *sp_11*

(d) Original Mel-spectrogram for *sp_36*

Figure 4.8: Mel-spectrograms for fine-tuning test sentences — child speakers.

Since the sentences uttered by children are prosodically expressive, their spectral representations are expected to present a more curvaceous shape than for the remaining fine-tune configurations. The contrast in prosody in regard to pre-train data leads the model to predict flatter pitch contours in spectral

representations. The model was pre-trained on data with a lower range of mean pitch values, thus having more difficulty in predicting the high frequency components present in the voices of children.

4.1.3 Speaker identity discrimination

For the model proposed in chapter 3 to operate in a multi-speaker fashion, it is crucial for it to distinguish speakers successfully. To demonstrate that the model can effectively discriminate among different voices, we extracted speaker embeddings from different speakers and analyzed these with t-SNE, a technique that allows to visualize high–dimensional data in a two or three–dimensional space [62].

Figure 4.9(a) depicts the embeddings generated from 12 different speakers, using 10 utterances per speaker. It is worth noticing that different samples uttered by the same speaker produce very similar embeddings, which almost entirely overlap. This proves that the model can successfully distinguish different speaker identities, independently of the linguistic content of each utterance. When visualizing a wider range of speaker identities, illustrated in figures 4.9(b), (c), and (d), one can notice that the placement of the embeddings relatively to one another allows to identify the speakers' gender. Furthermore, the embeddings of fine-tune speakers are also represented.

Pitch is known to be one of the most common features in speaker identification tasks. In fact, the presence of pitch–related information is noticeable in the speaker embeddings employed in the proposed model, since it is possible to correlate the relative location of an embedding in the plot, with the mean pitch of the speaker associated to that embedding. To prove this, sections 4.1.3.1, 4.1.3.2, and 4.1.3.3 detail the analysis of the deviation of fine-tune embeddings relatively to pre-train speaker identities, taking into account the pitch data studied in section 4.1.1.

4.1.3.1 Adult speakers: *sp_01* and *sp_02*

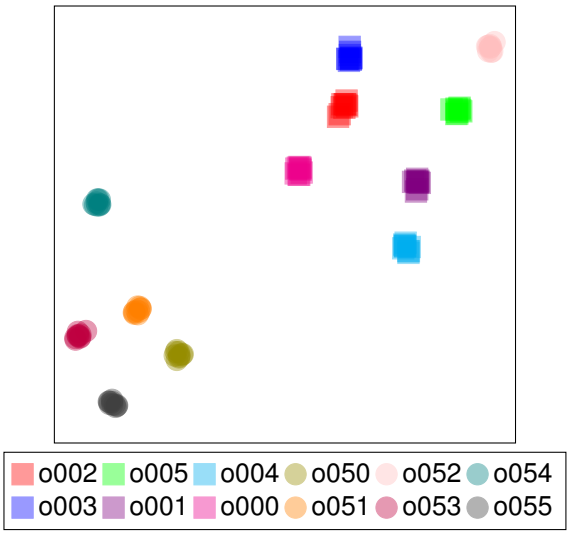
Besides all the embeddings of pre-train speakers, figure 4.9(b) depicts the embeddings of adult fine-tune speakers. From a gender viewpoint, it is noticeable that the fine-tune embeddings are placed in accordance with the pre-train speakers' embedding distribution.

In comparison to the range of pitch values of pre-train speakers, the male speaker *sp_02* registered a relatively low mean pitch value — 114 Hz — quite far from the lowest mean pitch registered for pre-train female speakers — 161 Hz. Female speaker *sp_01* displayed a mean pitch of 200 Hz which is very close to the highest mean pitch value recorded for pre-train male speakers — 197 Hz. This scenario suggests that the embedding of *sp_01* is closer to male–speaker embeddings than the embedding of *sp_02* is to female–speaker embeddings, and can be observed in figure 4.9(b).

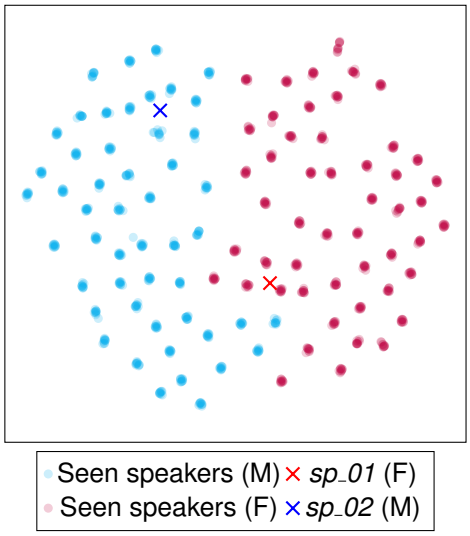
4.1.3.2 Adolescent speakers: *sp_03* and *sp_04*

The embeddings of adolescent fine-tune speakers are illustrated in figure 4.9(c). Regarding gender, fine-tune embeddings are placed in conformity with the distribution of pre-train speakers.

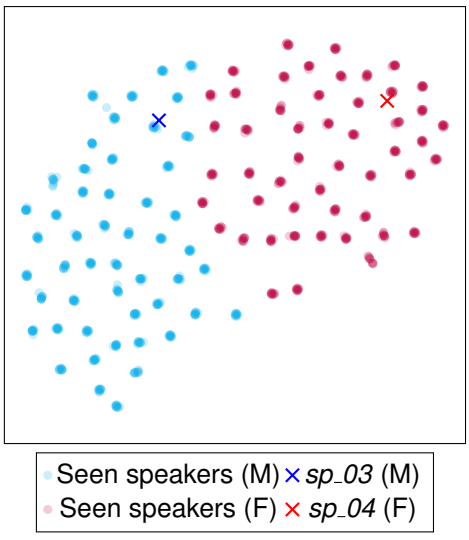
Both fine-tune speakers registered relatively large values of mean pitch in comparison to pre-train speakers of the same gender. Male speaker *sp_03* recorded a mean pitch of 165 Hz, among the highest for male pre-train speakers. Moreover, this value is close to the lowest pitch value for pre-train female speakers. This suggests that the embedding of *sp_03* is one of the closest to embeddings of female speakers. On the other hand, given that female speaker *sp_04* recorded a mean pitch of 254 Hz, which is one of the highest among female pre-train speakers, it is expected that the corresponding embedding is one of the farthest from male–speaker embeddings. As seen in the figure, the embedding of speaker *sp_03* is significantly closer to female speaker embeddings than the embedding of speaker *sp_04* is to male speaker embeddings.



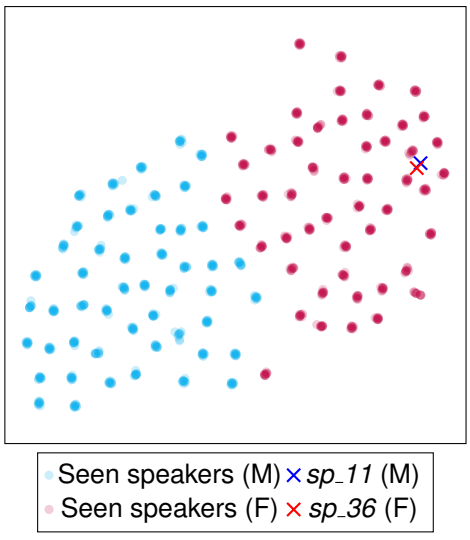
(a) Some embeddings of pre-train speakers (squares and circles represent male and female speakers, respectively)



(b) Embeddings of all pre-train speakers and adult fine-tune speakers



(c) Embeddings of all pre-train speakers and adolescent fine-tune speakers



(d) Embeddings of all pre-train speakers and child fine-tune speakers

Figure 4.9: Visualization of speaker embeddings.

4.1.3.3 Child speakers: *sp_11* and *sp_36*

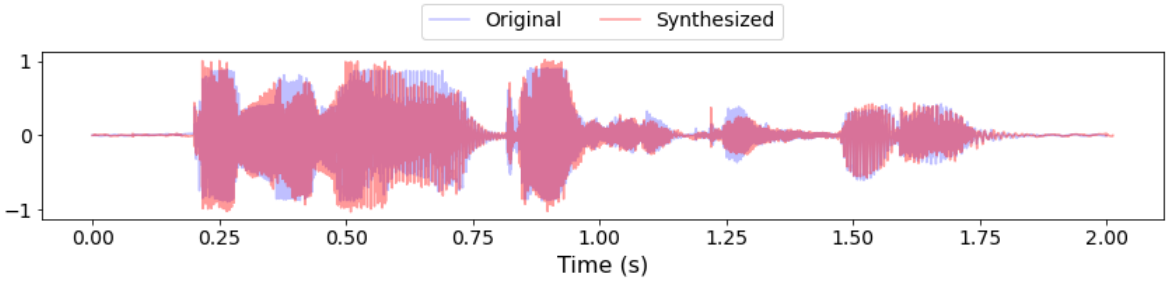
Embeddings of child fine-tune speakers are depicted in figure 4.9(d). Both child voices registered the highest mean pitch values among all speakers.

The placement of these embeddings relative to pre-train embeddings suggests that both are female speakers, even though only speaker *sp_36* is. The misplacement of the male speaker embedding could be expected due to pitch similarity between the voices of children. Furthermore, the fact that both children displayed a very high mean pitch suggests that their embeddings are among the farthest from male–speaker embeddings, which in fact occurs.

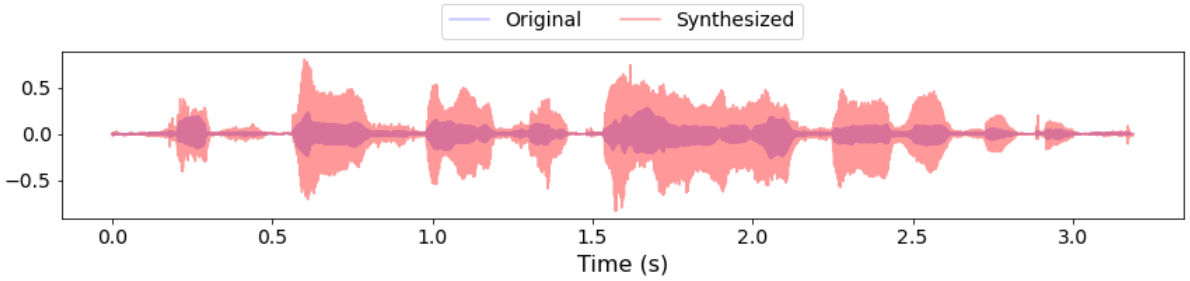
4.1.4 Synthetic speech for OOD configurations

The Seq2Seq regressive model is able to generate meaningful encoder-decoder alignments, that translate into intelligible speech, for both OOD fine-tune configurations. Regarding the voice models of these configurations, section 4.1.2 identified the patterns present in Mel-spectrograms that negatively influenced synthetic speech quality. Besides the quality of spectral outputs, one must also consider that the amount of data was very scarce for each OOD scenario — under five minutes of speech for each configuration.

The last component in the multi-speaker TTS pipeline — the neural vocoder — is determinant in the final output quality since it generates the synthetic speech signal. To identify the best possible output the neural vocoder could generate for each OOD scenario, the vocoder was fed with Mel-spectrograms of original utterances. Then, the corresponding output was compared to the original speech signal. While for adolescents the synthetic audio resembled the original signal, for children, it was noticeably different from the original signal — the synthesized audio was amplified, as depicted in figure 4.10(b), and sounded noisier. Original audio signals of children recordings registered notably lower amplitudes than audio signals of the remaining configurations, which could have caused the amplitude difference visible in figure 4.10(b).



(a) Original and synthesized speech signals for adolescent speaker *sp_03*



(b) Original and synthesized speech signals for child speaker *sp_36*

Figure 4.10: Original speech signals and synthesized signals given the Mel-spectrograms of original speech.

4.2 Evaluation

This section focuses on assessing the performance of the proposed model in comparison to other systems.

Performance tests were divided into three groups: 1) naturalness and similarity; 2) intelligibility; and 3) synthesis speed. The proposed model was tested for the adult–speaker standard configuration. Each test is described in the subsections that follow.

4.2.1 Naturalness and similarity

Naturalness and similarity were rated using AB and ABX preference tests, respectively. “A” and “B” refer to synthetic utterances, and X to a sentence uttered by the target speaker, used as reference.

Three unseen sentences were randomly selected from the test set of each speaker. Ideally, one should use a larger number of utterances per speaker for testing, but that would have made the tests very lengthy and time-consuming.

To provide a context of the model’s performance in the scope of TTS, naturalness and similarity tests were extended to four different TTS approaches besides the proposed model: 1) EP-Tacotron-2: a DL-based model inspired by the original Tacotron 2 system, developed in the context of a previous Master Thesis [40]; 2) EP-Merlin: an SPSS-based model, also developed in the context of a previous Master Thesis [39]; 3) DIXI+: an in-house concatenative synthesis system; and 4) the reduced adult–speaker configuration of the proposed model.

Originally, the BDFALA corpus comprised 600 phonetically rich sentences for both speakers *sp_01*

and *sp_02*. Nevertheless, the reference configuration of the proposed model was trained with only 300 utterances per speaker, as the sentences were the same for both speakers, meaning that the corpus was made of parallel data. Since the model is fine-tuned to both speakers simultaneously, the dataset was split in two halves of 300 utterances, one for each speaker, to avoid training with parallel data.

The EP-Tacotron-2 configuration [40] was trained separately for each speaker, thus not raising the issue of parallel data. The training data for this configuration comprised fully parallel data for both speakers, including the 600 sentences used in the proposed model configuration, plus another 128, making up for a total of 728 utterances. The EP-Merlin configuration [39] was trained separately for each speaker, with the whole the 600–sentence dataset. DIXI+ [55] was not ”fine-tuned” for speakers *sp_01* and *sp_02*, but for other individual voices, and the burden of building concatenative systems for new voices was a major handicap of this type of synthesizers. Thus, DIXI+ only took part in naturalness assessments.

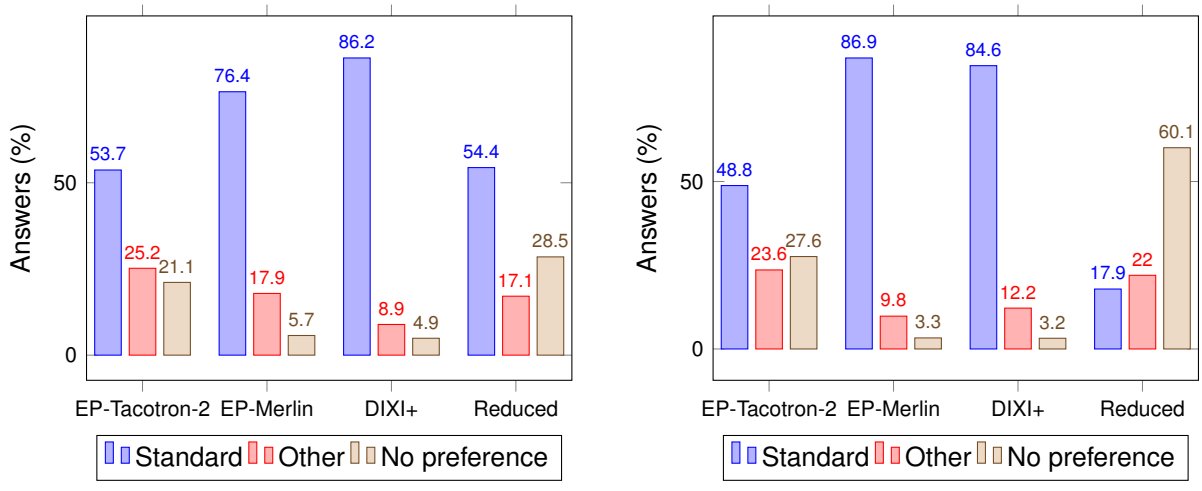
In preference tests, users were asked to listen beyond noise or potential distortion in the speech samples, and to focus solely on voice traits to choose which sample best resembled human speech, and the target voice, for naturalness and similarity, respectively.

4.2.1.1 Results

Results for naturalness and similarity tests are displayed in figures 4.11 and 4.12, respectively. In total, 41 listeners participated in these tests. The standard configuration of the proposed model is referred to as “Standard” in the caption of the figures, and “Other” refers to each of the four alternative approaches that were tested. Each approach is specified across the horizontal axis of the bar charts.

From the obtained results one can conclude that the proposed model needed much less data for fine-tuning than EP-Merlin and EP-Tacotron-2: the standard configuration used, at most, as much as half of the data for fine-tuning in comparison to these, while achieving meaningful results.

Regarding the naturalness test, the proposed model produced distinctively better results for both voices than EP-Merlin, and DIXI+, as expected. Surprisingly, the standard configuration of the proposed



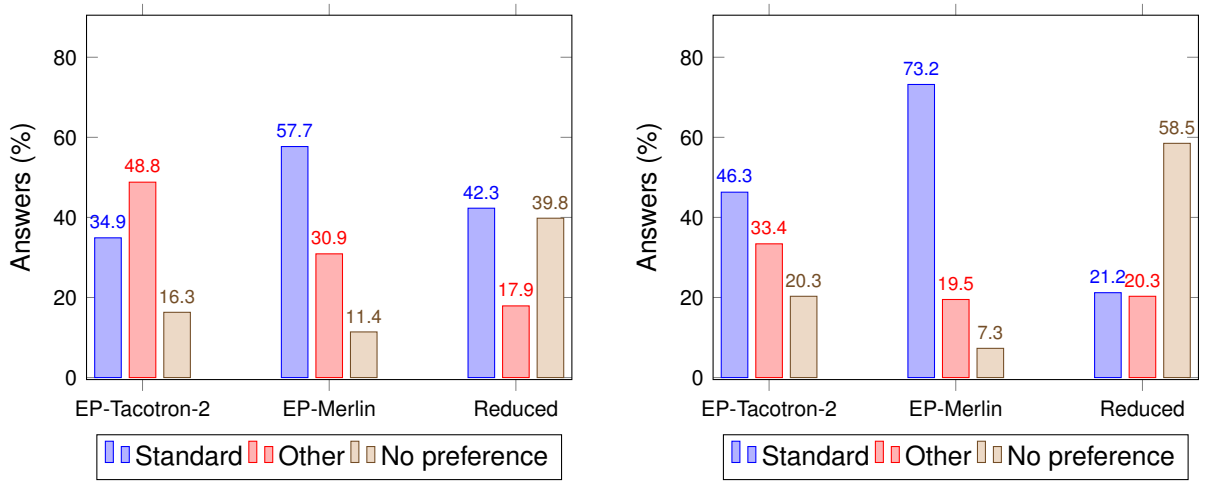
(a) Female voice *sp_01*

(b) Male voice *sp_02*

Figure 4.11: Naturalness AB preference test results.

model was also chosen over the DL-based EP-Tacotron-2 more times, for both voices. One must notice, however, that in the latter case the percentage of "no preference" answers was considerably larger than for EP-Merlin and DIXI+.

Regarding similarity, results showed that the proposed model clearly performed better than EP-Merlin for both voices, although not as blatantly as in the naturalness test. This could be due to the fact that EP-Merlin produced samples with noticeably better audio quality than the proposed model. Listeners may have been influenced by voice–unrelated features during assessments. In comparison with EP-Tacotron-2, the results for the female voice *sp_01* were as expected — EP-Tacotron-2 achieved a clear, but not overwhelming edge over the proposed model. Regarding the male voice *sp_02*, the proposed model achieved better results, even though over 20% of listeners did not prefer one model over the other. The discrepancy in similarity results from one speaker to the other suggests that EP-Tacotron-2 performed better for the female voice *sp_01* than for the male voice *sp_02*.



(a) Female voice *sp_01*

(b) Male voice *sp_02*

Figure 4.12: Similarity ABX preference test results.

Synthetic samples generated by EP-Tacotron-2 occasionally contained bursts of subtle noise and/or loudness. The latter phenomenon was particularly noticeable in fricative consonants — synthetic speech samples suddenly sounded louder from a specific timestep onward. Some participants may have been influenced by these factors when assessing the naturalness and similarity of EP-Tacotron-2 speech samples, thus preferring the proposed model.

In regard to the reduced configuration of the proposed model, results show that synthetic speech quality decreased for the female voice *sp_01* upon fine-tune data reduction. Nevertheless, the high percentage of "no preference" answers, especially in the similarity test, mitigates this phenomenon. For the male voice *sp_02*, both in naturalness and similarity assessments, no noticeable difference was registered in comparison to the standard configuration since the majority of participants expressed no preference. Also, the percentage of listeners that selected one configuration over the other is very balanced for both configurations.

4.2.2 Intelligibility

Intelligibility was assessed from listeners' textual transcriptions of semantically unpredictable sentences. Participants were asked to transcribe 10 sentences for each test speaker while listening to each sentence only once. The first test sentence of each speaker served as a dry run and therefore was assessed separately. The word error rate (WER) was determined from the transcriptions to measure how accurately the sentences were perceived by the listeners. The target test participants were native or fluent Portuguese speakers.

4.2.2.1 Results

In total, 33 participants took the intelligibility test. Table 4.4 displays the WER obtained from the transcriptions of participants. In some sentences several participants misheard specific words, which increased the WER. This was the case for the dry run sentence for speaker *sp_01* — one third of the participants could not perceive the word *colunas* —, hence the abnormally large WER for this sentence. Also, this sentence presented very little leading silence, which may have degraded its intelligibility. Table 4.5 displays the most frequently misheard words. Words *invocam/evocam* are only distinguished by the presence/absence of nasality, and some participants may have been influenced by the context: *evocam* would be in context within the sentence, while *invocam* is out of context.

Table 4.4: Intelligibility test results.

Speaker	**WER (%)**	
	Dry run	Remaining
sp_01 (F)	16.67	3.74
sp_02 (M)	1.01	4.22

Table 4.5: Frequently mistaken words.

Original word	**Incorrect transcription(s)**	**Total Occurrences**
colunas	*runas*, *com umas*	11
invocam	*evocam*	4
abstraída	*distraída*	24
do	*no*	31

4.2.3 Synthesis speed

Waveform generation is the most time-consuming stage in the DL-based TTS pipeline, hence, TTS synthesis speed is mainly determined by it. As such, only the performance of the waveform generation module was considered for the synthesis speed test. The proposed model's neural vocoder was compared with its counterpart in the EP-Tacotron-2 implementation, developed in a previous Master Thesis [40].

For each model, synthetic speech was generated for the female voice *sp_01* from 10 sentences, and synthesis times were compared for each sentence. On average, the proposed model's neural vocoder synthesized speech 11 times faster than its counterpart in EP-Tacotron-2, with an NVIDIA GeForce GTX TITAN X GPU.

4.3 Summary

Speaker adaptation was performed to six different voices, distributed among four distinct fine-tune configurations: 1) standard adult–speaker; 2) adolescent–speaker; 3) child–speaker; and 4) reduced adult–speaker. Results show that the proposed model was superior than previous–generation systems in terms on naturalness and similarity. Furthermore, the proposed model was preferred over EP-Tacotron-2 for both speakers in terms of naturalness, and for speaker *sp_02* in terms of similarity. Still, the preponderance of the proposed model over Tacotron-2 is mitigated by two factors: 1) for both speakers, the percentage of "no preference" answers was higher than for previous–generation systems in all tests; and 2) for speaker *sp_01* exclusively, the naturalness result indicates that the proposed model was preferred over EP-Tacotron-2, while the similarity result indicates the opposite, thus not being clear which system ensures better overall quality.

Fine-tuning with reduced data is a key aspect for efficient speaker adaptation — less data with specific characteristics is easier to obtain, and the process itself unfolds faster. For voice *sp_02*, results show that there was no difference regarding naturalness and similarity between standard and reduced configurations. Regarding voice *sp_01*, although results indicate a clear preference for the standard configuration over the reduced setting, the amount of "no preference" answers was the highest among all tests for this voice. Still, one can conclude that the reduced and standard settings achieved the same performance for one of the voices, meaning that it is possible to preserve output quality while reducing the amount of training data.

Overall, the model generated intelligible speech for EP, and performed, on average, 11 times faster than EP-Tacotron-2.

Chapter 5

Conclusions

This study proposed a multi-speaker TTS system capable of generating intelligible speech for EP, based on state-of-the-art Speech Synthesis methodologies. Three factors proved to have a prominent impact in EP text-preprocessing: 1) the phonetic transcriptions of homographs; 2) text normalization; and 3) the phonetic alphabet. From the experimented text-preprocessing procedures, the Festival–based GtoP offered the best performance for these factors, ensuring coherent phonetic transcriptions for most homographs and a robust text normalization, while employing a concise phonetic alphabet (SAMPA). The superior performance of this GtoP module (that is, the better precision of generated phonetic transcriptions) ensured better convergence during training stages, and improved the pronunciation and intelligibility of synthesized speech.

The ultimate goal of TTS is to provide an end-to-end architecture that avoids separate and often intricate training processes, in which text pre-processing is included. However, the detached nature of different stages within a TTS system is useful for changing or updating specific modules while maintaining the overall structure of the system. Regarding text pre-processing, this allowed to switch between different GtoP approaches effectively, and select the most suitable one.

The pre-train stage of the regressive Seq2Seq model produced better acoustic representations when trained with data comprising a large number of speakers (at least 100), distributed in balanced quantity across speakers. Likewise, training data comprising shorter utterances allowed to establish a better mapping between phonetic sequences and acoustic sequences.

Employing a universal neural vocoder for waveform generation instead of a speaker dependent neural vocoder simplified the speaker adaptation process — within the TTS framework, only the regressive model required additional training for speaker adaptation.

In what concerns the incorporation of new speakers, the proposed model gained from employing a framework similar to non-parallel voice conversion, as it allowed to add an arbitrary number of new speakers simultaneously. Moreover, only considering the pre-train stage, it ensured TTS synthesis for 100 different voices. This model registered better performance than previous–generation speaker-dependent TTS systems, and achieved results comparable to a different state-of-the-art speaker dependent system. Furthermore, the proposed model posed a more viable option for speaker adaptation, both

in data quantity and training procedure, than the aforementioned alternatives.

The proposed model was able to perform speaker adaptation with reduced data for declarative and prosodically neutral sentences, since pre-train and fine-tuning data was predominantly of this type. Nevertheless, to generate prosodically expressive synthetic speech, the whole model (including the neural vocoder) should be trained from scratch with expressive data, comprising a wider range pitch values, and a more balanced distribution in terms of sentence type.

Regarding synthesis speed, the proposed model proved to be significantly faster than EP-Tacotron-2. The WaveRNN–based vocoder employed in the proposed model was, on average, 11 times faster than the WaveNet employed in EP-Tacotron-2, and thus more practical.

5.1 Future work

The Festival–based text preprocessing tool as well as text transcriptions of all speech corpora followed the old EP orthography, prior to the 1990 Orthographic Agreement. Given that the two orthographies are frequently employed together in written text, future TTS systems for EP should support both of them.

The presented model performed multi-speaker TTS, and allowed to incorporate new, unseen voices. Nevertheless, the generation of synthetic speech was limited to a specific speaking style. Available training data was mostly made of declarative sentences, uttered with neutral prosody, which prevented the model from learning exclamatory and interrogative intonations. TTS systems' architectures should be extended to retain and model prosodic information from input data. Another nuisance common to most of TTS systems including the proposed model, is that these require high-quality data (in terms of audio and recording conditions) to synthesize natural-sounding, intelligible speech. Obtaining high-quality data is often difficult, which poses a drawback in the feasibility of these systems.

Recent studies presented at the INTERSPEECH 2020 conference have detailed relevant advancements in the scope of TTS synthesis. Regarding prosody modeling and control, we chose two that provided meaningful contributions: 1) a model that aims to learn prosodic representations from speech data [63]; and 2) a method that generates prosody from syntactic and semantic information in input text [64]. The former model operates at syllable level, rather than phoneme level. It extracts prosodic representations from reference speech, and concatenates these with encoded text representations. Then, it decodes the resultant representation into a synthetic Mel-spectrogram. The latter method measures the linguistic embedding similarity between the input sentence and all train sentences, and selects the train sentence with highest similarity. The synthetic Mel-spectrogram is predicted based on the acoustic embedding of the selected train sentence, sharing its prosodic traits. In both methodologies, predicted Mel-spectrograms are fed to waveform generation modules to synthesize speech.

Regarding speech synthesis for different audio quality and recording conditions, two other techniques attained additional improvements: 1) a WaveRNN-based universal vocoder that extends its performance to unseen recording conditions [65]; and 2) a data efficient voice cloning system from noisy samples [66]. The universal vocoder leverages from additional information in the form of speaker em-

beddings and ensures better performance than the baseline WaveRNN. The voice cloning system employs domain-adversarial training to extricate noise from noisy speech samples, and thereby generate synthetic speech.

References

[1] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986. ISSN 1476-4687. doi: 10.1038/323533a0. URL https://doi.org/10.1038/323533a0.

[2] J. Holmes and W. Holmes. *Speech Synthesis and Recognition*. Taylor & Francis, Inc., USA, 2nd edition, 2002. ISBN 0-7484-0857-6.

[3] S. King. What is ”end-to-end” text-to-speech synthesis?, 2019. URL http://media.speech.zone/images/Simon_King_Lancaster_2019_compressed_for_publication.pdf.

[4] H. Mullah. A Comparative Study of Different Text-to- Speech Synthesis Techniques. *International Journal of Scientific & Engineering Research*, 6:287–292, 2015.

[5] S. Jothilakshmi and V. N. Gudivada. Chapter 10 - Large Scale Data Enabled Evolution of Spoken Language Research and Applications. In V. N. Gudivada, V. V. Raghavan, V. Govindaraju, and C. R. Rao, editors, *Cognitive Computing: Theory and Applications*, volume 35 of *Handbook of Statistics*, pages 301 – 340. Elsevier, 2016. doi: 10.1016/bs.host.2016.07.005. URL http://www.sciencedirect.com/science/article/pii/S0169716116300463.

[6] H. Zen, K. Tokuda, and A. W. Black. Statistical parametric speech synthesis. *Speech Communication*, 51(11):1039 – 1064, 2009. ISSN 0167-6393. doi: https://doi.org/10.1016/j.specom.2009.04.004. URL http://www.sciencedirect.com/science/article/pii/S0167639309000648.

[7] S. King. An introduction to statistical parametric speech synthesis. *Sadhana*, 36(5):837–852, 2011. ISSN 0973-7677. doi: 10.1007/s12046-011-0048-y. URL https://doi.org/10.1007/s12046-011-0048-y.

[8] H. Zen, A. Senior, and M. Schuster. Statistical parametric speech synthesis using deep neural networks. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 7962–7966, May 2013. doi: 10.1109/ICASSP.2013.6639215. ISSN: 2379-190X.

[9] Y. Ning, S. He, Z. Wu, C. Xing, and L.-J. Zhang. A Review of Deep Learning Based Speech Synthesis. *Applied Sciences*, 9(19):4050, Jan. 2019. doi: 10.3390/app9194050. URL https://www.mdpi.com/2076-3417/9/19/4050. Number: 19 Publisher: Multidisciplinary Digital Publishing Institute.

[10] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to Sequence Learning with Neural Networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3104–3112. Curran Associates, Inc., 2014. URL http://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf.

[11] J. Nurminen, H. Silén, V. Popa, E. Helander, and M. Gabbouj. Voice Conversion. In S. Ramakrishnan, editor, *Speech Enhancement, Modeling and Recognition- Algorithms and Applications*. IntechOpen, Rijeka, 2012. doi: 10.5772/37334. URL https://doi.org/10.5772/37334. Section: 5.

[12] M. Abe, S. Nakamura, K. Shikano, and H. Kuwabara. Voice conversion through vector quantization. In *ICASSP-88., International Conference on Acoustics, Speech, and Signal Processing*, pages 655–658 vol.1, 1988. doi: 10.1109/ICASSP.1988.196671.

[13] J.-X. Zhang, Z.-H. Ling, and L.-R. Dai. Non-Parallel Sequence-to-Sequence Voice Conversion with Disentangled Linguistic and Speaker Representations. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:540–552, 2020. ISSN 2329-9290, 2329-9304. doi: 10.1109/TASLP.2019.2960721. URL http://arxiv.org/abs/1906.10508. arXiv: 1906.10508.

[14] S. Arik, J. Chen, K. Peng, W. Ping, and Y. Zhou. Neural Voice Cloning with a Few Samples. *CoRR*, abs/1802.06006, 2018. URL http://arxiv.org/abs/1802.06006. _eprint: 1802.06006.

[15] K. Chen, B. Chen, J. Lai, and K. Yu. High-quality Voice Conversion Using Spectrogram-Based WaveNet Vocoder. In *Interspeech 2018*, pages 1993–1997. ISCA, Sept. 2018. doi: 10.21437/Interspeech.2018-1528. URL http://www.isca-speech.org/archive/Interspeech_2018/abstracts/1528.html.

[16] H. Lu, Z. Wu, D. Dai, R. Li, S. Kang, J. Jia, and H. Meng. One-Shot Voice Conversion with Global Speaker Embeddings. In *Interspeech 2019*, pages 669–673. ISCA, Sept. 2019. doi: 10.21437/Interspeech.2019-2365. URL http://www.isca-speech.org/archive/Interspeech_2019/abstracts/2365.html.

[17] J.-c. Chou, C.-c. Yeh, H.-y. Lee, and L.-s. Lee. Multi-target Voice Conversion without Parallel Data by Adversarially Learning Disentangled Audio Representations. *arXiv:1804.02812 [cs, eess]*, June 2018. URL http://arxiv.org/abs/1804.02812. arXiv: 1804.02812.

[18] S. Wang, Y. Qian, and K. Yu. What Does the Speaker Embedding Encode? In *INTERSPEECH*, 2017. doi: 10.21437/Interspeech.2017-1125.

[19] M. Blaauw, J. Bonada, and R. Daido. Data Efficient Voice Cloning for Neural Singing Synthesis. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6840–6844, May 2019. doi: 10.1109/ICASSP.2019.8682656. ISSN: 2379-190X.

[20] J. Lorenzo-Trueba, J. Yamagishi, T. Toda, D. Saito, F. Villavicencio, T. Kinnunen, and Z. Ling. The Voice Conversion Challenge 2018: Promoting Development of Parallel and Nonparallel Methods. *arXiv:1804.04262 [cs, eess, stat]*, Apr. 2018. URL http://arxiv.org/abs/1804.04262. arXiv: 1804.04262.

[21] J. Latorre, J. Lachowicz, J. Lorenzo-Trueba, T. Merritt, T. Drugman, S. Ronanki, and K. Viacheslav. Effect of data reduction on sequence-to-sequence neural TTS. *arXiv:1811.06315 [cs, eess]*, Nov. 2018. URL http://arxiv.org/abs/1811.06315. arXiv: 1811.06315.

[22] T. Merritt, B. Putrycz, A. Nadolski, T. Ye, D. Korzekwa, W. Dolecki, T. Drugman, V. Klimkov, A. Moinet, A. Breen, R. Kuklinski, N. Strom, and R. Barra-Chicote. Comprehensive evaluation of statistical speech waveform synthesis. *arXiv:1811.06296 [cs, eess]*, Dec. 2018. URL http://arxiv.org/abs/1811.06296. arXiv: 1811.06296.

[23] C. Mendonça and S. Delikaris-Manias. Statistical tests with MUSHRA data. In *144th Audio Engineering Society International Convention 2018*, pages 859–868, United States, Jan. 2018. Audio Engineering Society. ISBN 978-1-5108-6474-0.

[24] Y. He, H. Zhang, and Y. Wang. RawNet: Fast End-to-End Neural Vocoder. *arXiv:1904.05351 [cs, eess, stat]*, Apr. 2019. URL http://arxiv.org/abs/1904.05351. arXiv: 1904.05351.

[25] Y. Ai and Z.-H. Ling. A Neural Vocoder with Hierarchical Generation of Amplitude and Phase Spectra for Statistical Parametric Speech Synthesis. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:839–851, 2020. ISSN 2329-9290, 2329-9304. doi: 10.1109/TASLP.2020.2970241. URL http://arxiv.org/abs/1906.09573. arXiv: 1906.09573.

[26] J. Lorenzo-Trueba, T. Drugman, J. Latorre, T. Merritt, B. Putrycz, R. Barra-Chicote, A. Moinet, and V. Aggarwal. Towards achieving robust universal neural vocoding. *arXiv:1811.06292 [cs, eess]*, July 2019. URL http://arxiv.org/abs/1811.06292. arXiv: 1811.06292.

[27] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu. WaveNet: A Generative Model for Raw Audio. *arXiv:1609.03499 [cs]*, Sept. 2016. URL http://arxiv.org/abs/1609.03499. arXiv: 1609.03499.

[28] D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling. Improving Variational Inference with Inverse Autoregressive Flow. *arXiv:1606.04934 [cs, stat]*, Jan. 2017. URL http://arxiv.org/abs/1606.04934. arXiv: 1606.04934.

[29] A. v. d. Oord, Y. Li, I. Babuschkin, K. Simonyan, O. Vinyals, K. Kavukcuoglu, G. v. d. Driessche, E. Lockhart, L. C. Cobo, F. Stimberg, N. Casagrande, D. Grewe, S. Noury, S. Dieleman, E. Elsen, N. Kalchbrenner, H. Zen, A. Graves, H. King, T. Walters, D. Belov, and D. Hassabis. Parallel WaveNet: Fast High-Fidelity Speech Synthesis. *arXiv:1711.10433 [cs]*, Nov. 2017. URL http://arxiv.org/abs/1711.10433. arXiv: 1711.10433.

[30] N. Kalchbrenner, E. Elsen, K. Simonyan, S. Noury, N. Casagrande, E. Lockhart, F. Stimberg, A. v. d. Oord, S. Dieleman, and K. Kavukcuoglu. Efficient Neural Audio Synthesis. *arXiv:1802.08435 [cs, eess]*, June 2018. URL http://arxiv.org/abs/1802.08435. arXiv: 1802.08435 version: 2.

[31] J.-M. Valin and J. Skoglund. LPCNet: Improving Neural Speech Synthesis Through Linear Prediction. *arXiv:1810.11846 [cs, eess]*, Feb. 2019. URL http://arxiv.org/abs/1810.11846. arXiv: 1810.11846.

[32] O. Watts, G. Eje Henter, J. Fong, and C. Valentini-Botinhao. Where do the improvements come from in sequence-to-sequence neural TTS? In *10th ISCA Speech Synthesis Workshop*, pages 217–222. ISCA, Sept. 2019. doi: 10.21437/SSW.2019-39. URL http://www.isca-speech.org/archive/SSW_2019/abstracts/SSW10_O_6-2.html.

[33] P. Taylor, A. W. Black, and R. Caley. The Architecture Of The Festival Speech Synthesis System. In *IN THE THIRD ESCA WORKSHOP IN SPEECH SYNTHESIS*, pages 147–151, 1998.

[34] Y. Wang, R. J. Skerry-Ryan, D. Stanton, Y. Wu, R. J. Weiss, N. Jaitly, Z. Yang, Y. Xiao, Z. Chen, S. Bengio, Q. Le, Y. Agiomyrgiannakis, R. Clark, and R. A. Saurous. Tacotron: Towards End-to-End Speech Synthesis. *arXiv:1703.10135 [cs]*, Apr. 2017. URL http://arxiv.org/abs/1703.10135. arXiv: 1703.10135.

[35] E. Shriberg. Higher-Level Features in Speaker Recognition. In C. Müller, editor, *Speaker Classification I*, volume 4343, pages 241–259. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007. ISBN 978-3-540-74186-2 978-3-540-74200-5. doi: 10.1007/978-3-540-74200-5_14. URL http://link.springer.com/10.1007/978-3-540-74200-5_14.

[36] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. J. Skerry-Ryan, R. A. Saurous, Y. Agiomyrgiannakis, and Y. Wu. Natural TTS Synthesis by Conditioning WaveNet on Mel Spectrogram Predictions. *arXiv:1712.05884 [cs]*, Feb. 2018. URL http://arxiv.org/abs/1712.05884. arXiv: 1712.05884.

[37] A. Polyak and L. Wolf. Attention-based Wavenet Autoencoder for Universal Voice Conversion. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6800–6804, Brighton, United Kingdom, May 2019. IEEE. ISBN 978-1-4799-8131-1. doi: 10.1109/ICASSP.2019.8682589. URL https://ieeexplore.ieee.org/document/8682589/.

[38] L. C. Oliveira, M. d. C. G. V. Ribeiro, and I. Trancoso. DIXI - Portuguese Text-to-Speech System. In *EUROSPEECH'91 - European Conference on Speech Technology*. ESCA, 1991. event-place: Genoa, Italy.

[39] A. C. R. Gonçalves. Text-to-Speech Synthesis in European Portuguese using Deep Learning. Master's thesis, Instituto Superior Técnico - Universidade de Lisboa, 2018.

[40] S. Quintas. Portuguese Speech Synthesis for Amyotrophic Lateral Sclerosis. Master's thesis, Instituto Superior Técnico - Universidade de Lisboa, 2019.

[41] Z. Wu, O. Watts, and S. King. Merlin: An Open Source Neural Network Speech Synthesis System. In *SSW*, 2016. doi: 10.21437/SSW.2016-33.

[42] W. Ping, K. Peng, A. Gibiansky, S. O. Arik, A. Kannan, S. Narang, J. Raiman, and J. Miller. Deep Voice 3: Scaling Text-to-Speech with Convolutional Sequence Learning. *arXiv:1710.07654 [cs, eess]*, Feb. 2018. URL http://arxiv.org/abs/1710.07654. arXiv: 1710.07654.

[43] Y. Taigman, L. Wolf, A. Polyak, and E. Nachmani. VoiceLoop: Voice Fitting and Synthesis via a Phonological Loop. *arXiv:1707.06588 [cs]*, Feb. 2018. URL http://arxiv.org/abs/1707.06588. arXiv: 1707.06588.

[44] T. Hayashi, R. Yamamoto, K. Inoue, T. Yoshimura, S. Watanabe, T. Toda, K. Takeda, Y. Zhang, and X. Tan. ESPnet-TTS: Unified, Reproducible, and Integratable Open Source End-to-End Text-to-Speech Toolkit. *arXiv:1910.10909 [cs, eess]*, Feb. 2020. URL http://arxiv.org/abs/1910.10909. arXiv: 1910.10909.

[45] T. Kaneko, H. Kameoka, K. Tanaka, and N. Hojo. StarGAN-VC2: Rethinking Conditional Methods for StarGAN-Based Voice Conversion. *arXiv:1907.12279 [cs, eess, stat]*, Aug. 2019. URL http://arxiv.org/abs/1907.12279. arXiv: 1907.12279.

[46] Y. Zhang, R. J. Weiss, H. Zen, Y. Wu, Z. Chen, R. J. Skerry-Ryan, Y. Jia, A. Rosenberg, and B. Ramabhadran. Learning to Speak Fluently in a Foreign Language: Multilingual Speech Synthesis and Cross-Language Voice Cloning. *arXiv:1907.04448 [cs, eess]*, July 2019. URL http://arxiv.org/abs/1907.04448. arXiv: 1907.04448.

[47] Y. Jia, R. J. Weiss, F. Biadsy, W. Macherey, M. Johnson, Z. Chen, and Y. Wu. Direct speech-to-speech translation with a sequence-to-sequence model. *arXiv:1904.06037 [cs, eess]*, June 2019. URL http://arxiv.org/abs/1904.06037. arXiv: 1904.06037.

[48] L. Bernardo. Exploring the future of voice tech, Unbabel, Lisbon, Dec. 2019.

[49] C. Veaux, J. Yamagishi, and K. MacDonald. *SUPERSEDED - CSTR VCTK Corpus: English Multi-speaker Corpus for CSTR Voice Cloning Toolkit*. University of Edinburgh, 2017. doi: 10.7488/DS/1994. URL http://datashare.is.ed.ac.uk/handle/10283/2651.

[50] J. Kominek and A. Black. The CMU Arctic speech databases. *SSW5-2004*, 2004.

[51] C. A. D. Martins, M. I. d. V. Mascarenhas, H. Meinedo, J. P. d. S. Neto, L. C. Oliveira, C. E. M. Ribeiro, I. Trancoso, and M. d. C. G. V. Ribeiro. Spoken Language Corpora for Speech Recognition and Synthesis in European Portuguese. In *RECPAD'98 - 10th Portuguese Conference on Pattern Recognition*, Mar. 1998. event-place: Lisbon, Portugal.

[52] Y. Bengio, J. Louradour, R. Collobert, and J. Weston. Curriculum learning. In *Journal of the American Podiatry Association*, volume 60, page 6, 2009. doi: 10.1145/1553374.1553380.

[53] E. Dunbar, R. Algayres, J. Karadayi, M. Bernard, J. Benjumea, X.-N. Cao, L. Miskic, C. Dugrain, L. Ondel, A. W. Black, L. Besacier, S. Sakti, and E. Dupoux. The Zero Resource Speech Challenge 2019: TTS without T. *arXiv:1904.11469 [cs, eess]*, July 2019. URL http://arxiv.org/abs/1904.11469. arXiv: 1904.11469.

[54] M. Bernard, hadware, R. Riad, A. Greyber, J. Benjumea, lsn0gud, and S. Li. bootphon/phonemizer: phonemizer-2.2.1, July 2020. URL https://doi.org/10.5281/zenodo.3959035.

[55] S. Paulo, L. C. Oliveira, C. Mendes, L. Figueira, R. Cassaca, C. Viana, and H. Moniz. DIXI – A Generic Text-to-Speech System for European Portuguese. In A. Teixeira, V. L. S. de Lima, L. C. de Oliveira, and P. Quaresma, editors, *Computational Processing of the Portuguese Language*, volume 5190, pages 91–100. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008. ISBN 978-3-540-85979-6 978-3-540-85980-2. doi: 10.1007/978-3-540-85980-2_10. URL http://link.springer.com/10.1007/978-3-540-85980-2_10. Series Title: Lecture Notes in Computer Science.

[56] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention Is All You Need. *arXiv:1706.03762 [cs]*, Dec. 2017. URL http://arxiv.org/abs/1706.03762. arXiv: 1706.03762.

[57] R. Vergin and D. O'Shaughnessy. Pre-emphasis and speech recognition. In *Proceedings 1995 Canadian Conference on Electrical and Computer Engineering*, volume 2, pages 1062–1065 vol.2, Sept. 1995. doi: 10.1109/CCECE.1995.526613. ISSN: 0840-7789.

[58] F. Ernawan and N. Abu. Efficient Discrete Tchebichef on SpectrumAnalysis of Speech Recognition. *International Journal of Machine Learning and Computing*, 1:1–6, 2011. doi: 10.7763/IJMLC.2011.V1.1.

[59] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto. librosa: Audio and music signal analysis in python. In *Proceedings of the 14th python in science conference*, volume 8, 2015.

[60] H. Bredin, R. Yin, J. M. Coria, G. Gelly, P. Korshunov, M. Lavechin, D. Fustes, H. Titeux, W. Bouaziz, and M.-P. Gill. pyannote.audio: neural building blocks for speaker diarization. In *ICASSP 2020, IEEE International Conference on Acoustics, Speech, and Signal Processing*, Barcelona, Spain, 2020.

[61] D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. *arXiv:1412.6980 [cs]*, Jan. 2017. URL http://arxiv.org/abs/1412.6980. arXiv: 1412.6980.

[62] L. V. D. Maaten and G. E. Hinton. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[63] G. Zhang, Y. Qin, and T. Lee. Learning Syllable-Level Discrete Prosodic Representation for Expressive Speech Generation. In *Interspeech 2020*, pages 3426–3430. ISCA, Oct. 2020. doi: 10.21437/Interspeech.2020-2228. URL http://www.isca-speech.org/archive/Interspeech_2020/abstracts/2228.html.

[64] S. Tyagi, M. Nicolis, J. Rohnke, T. Drugman, and J. Lorenzo-Trueba. Dynamic Prosody Generation for Speech Synthesis Using Linguistics-Driven Acoustic Embedding Selection. In *Interspeech 2020*, pages 4407–4411. ISCA, Oct. 2020. doi: 10.21437/Interspeech.2020-1411. URL http://www.isca-speech.org/archive/Interspeech_2020/abstracts/1411.html.

[65] D. Paul, Y. Pantazis, and Y. Stylianou. Speaker Conditional WaveRNN: Towards Universal Neural Vocoder for Unseen Speaker and Recording Conditions. In *Interspeech 2020*, pages 235–239. ISCA, Oct. 2020. doi: 10.21437/Interspeech.2020-2786. URL http://www.isca-speech.org/archive/Interspeech_2020/abstracts/2786.html.

[66] J. Cong, S. Yang, L. Xie, G. Yu, and G. Wan. Data Efficient Voice Cloning from Noisy Samples with Domain Adversarial Training. In *Interspeech 2020*, pages 811–815. ISCA, Oct. 2020. doi: 10.21437/Interspeech.2020-2530. URL http://www.isca-speech.org/archive/Interspeech_2020/abstracts/2530.html.

Appendix A

Fine-tune hyperparameters

Table A.1: Multi-speaker Seq2Seq fine-tune stage hyperparameters.

Type	Parameter name	Value	Description
Experiment parameters	epochs	70	Number of training epochs
	iters_per_checkpoint	100	Number of iterations per checkpoint
Data parameters	training_list	–	Path to file listing data for training
	validation_list	–	Path to file listing data for validation
	test_list	–	Path to file listing data for testing
	n_mel_channels	80	Number of Mel bands
	n_symbols	47	Number of symbols in the phoneme list, 47 using Festival, 54 using Seq2Seq toolkit, and 53 using eSpeak
	n_speakers	2	Number of voices for speaker adaptation
	predict_spectrogram	False	Set as false to use Mel-spectrograms instead of linear spectrograms
Training parameters	learning_rate	1e-3	Learning rate
	weight_decay	1e-6	Weight decay coefficient
	grad_clip_thresh	5.0	Gradient norms above this value are clipped
	batch_size	8	Batch size
	warmup	7	Number of epochs with constant LR
	decay_rate	0.5	LR penalizing factor
	decay_every	7	LR decays every decay_every epochs
	contrastive_loss_w	30.0	Contrastive loss weighting factor
	speaker_encoder_loss_w	0	Speaker encoder loss weighting factor
	text_classifier_loss_w	1.0	Text classifier loss weighting factor
	speaker_adversarial_loss_w	0.2	Adversarial loss weighting factor
	speaker_classifier_loss_w	0.1	Auxiliary classifier loss weighting factor
	ce_loss	False	Set the adversarial loss as the symmetric of the auxiliary classifier loss

Appendix B

Semantically unpredictable sentences

Table B.1: Intelligibility test sentences – speaker *sp_01*

#	Sentence
1	*Colunas experimentais sobem pela chave.*
2	*A primavera sacode frustrações aquáticas.*
3	*O fumo zangado indicava ideias amarelas.*
4	*Bibliotecas elétricas são exploradas por relógios.*
5	*Mochilas gasosas falam por cima de tesouras.*
6	*Caixotes tristes voaram amanhã.*
7	*Padrões prateados invocam fogueiras.*
8	*A minha lancheira surrealista está abstraída.*
9	*Montei uma alface delimitada.*
10	*Os parafusos comem tijolos felizes.*

Table B.2: Intelligibility test sentences – speaker *sp_02*

#	Sentence
1	*Círculos recortados na parede brincam comigo.*
2	*Garrafas ecléticas preenchem o meu cérebro.*
3	*A suspensão escondeu-se do meu estojo.*
4	*O arvoredo usa facas de lava congeladas.*
5	*As maçãs do avião ficaram curadas.*
6	*O palhaço comeu gatos aéreos.*
7	*Poderes culinários desmantelaram arquitetos azuis.*
8	*A bipolaridade desenhou uma gaveta social luminosa.*
9	*O submarino bebia vinho no aeroporto.*
10	*A porta do saco de vidro é inteligente.*