UNIVERSITÉ DE LORRAINE

École doctorale IAEM Lorraine

Generating diverse synthetic data for ASR training data augmentation

THÈSE

présentée et soutenue publiquement le 10 octobre 2024

pour l'obtention du

Doctorat de l'Université de Lorraine

(mention informatique)

par

Sewade Olaolu Ogun

Composition du jury

Présidente :	Serena Ivaldi	Directrice de recherche, Centre Inria de l'U. de Lorraine
Rapporteurs :	Ralf Schlüter	Privatdozent, RWTH Aachen University
	Nicolas Obin	Maître de conférences, IRCAM, Sorbonne Université
Examinateur :	Emmanuel Dupoux	Directeur d'études, EHESS
Directeur de thèse :	Emmanuel Vincent	Directeur de recherche, Centre Inria de l'U. de Lorraine
Co-directeur de thèse :	Vincent Colotte	Maître de conférences, Université de Lorraine

Laboratoire Lorrain de Recherche en Informatique et ses Applications — UMR 7503

Acknowledgments

First and foremost, I thank God for giving me the strength and fortitude to complete this thesis. My family was also a pillar of support, my sisters, Senayon, Sesede, and Sedoten, and my dad and grandma. I couldn't have done it without their love and care. I will always love you.

Emmanuel and Vincent, you guided my research, supported me, revised my work, and posed critical questions that made this work the best version it can be. Our discussions have made me a better thinker and researcher, and it has birthed this thesis. Emmanuel, I cherish your exceptional experience and intellect. Vincent, I appreciate all your detailed feedback and our discussions. Thank you for believing in me.

Thanks to my jury members: Serena, Ralf, Nicolas, and Emmanuel Dupoux. I appreciate the time and effort you have dedicated to reviewing my thesis, as well as the insightful comments and questions posed during its defence. You have challenged my critical thinking. I am privileged to have you all on my jury.

My heartfelt thanks also go to my colleauges in the Multispeech team at INRIA, previous and current members, who became friends through our common values and discussions: Jean-Eudes Ayilo, Constance Douwes, Taous Latariene, Nasser-Eddine Monir, Mickaëlla Grondin-Verdon, Natalia Tomashenko, Théo Biasutto-Lervat, Sofiane Azzouz, Sam Bigeard, Aine Drelingyte, Imane Ghanem, Mayank Mishra, Nathan Olejniczak, Tulika Bose, Michel Olvera, Joris Cosentino, Hubert Nourtel, Louis Abel, Shakeel Sheikh, Pierre Champion, Sandipana Dowerah, Prerak Srivastava, Vinicius Ribeiro, Can Cui, Marina Krémé, Marie-Anne Lacroix, Yves Laprie, Emmanuelle Deschamps, Helene Cavallini, Pierre Champion, Nicolas Furnon, Soklong Him, Soklay Heng, Paul Magron, Louis Delebecque, Hugo Bergerat, Ashwin Geet D'Sa, Ajinkya Kulkarni, Mostafa Sadeghi, Paul Magron, Antoine Deleforge, Slim Ouini, and Denis Jouvet.

I would also like to express my gratitude to everyone in the Loria/Inria laboratory who fostered a culture of inclusivity and for recognizing the value of diversity in the workplace. Additionally, I am also grateful to the laboratory for having a great environment for work, relaxation, leisure, and lunch. The environment made me look forward to going to the laboratory daily.

I was surrounded by good friends during my PhD journey. My "Vamos à la playa" friends: Alexandre, Clémence, Anne-Catherine, Nicolas, Gabriel, Laura, Micaela, Mattia, Mariano, Belén, Mario, Elena, Nicolas L, Laura Z, and Francisca. You are my second family. You showed me friendship in diversity. I will forever cherish all the fun moments we shared.

My PhD journey also led to meeting wonderful friends like Linda, Can, Faustine, Amaury, and Clèvie. They occupy a special place in my heart. I learned a lot from our interactions, how to be tolerant of others and to appreciate others' cultures. Thank you for supporting me on my French learning journey.

Furthermore, to my French language teachers at the laboratory, Marie and Laura, I appreciate your efforts in helping me to surmount the beginners hurdle and then the intermediate plateau of language learning.

Lastly, I would like to congratulate myself for taking up the challenge to do a PhD and live the experience in a country far away from home. Along the way, I met beautiful and kind people, I learned to adapt, to be open-minded, to appreciate the moment, and to persevere.

In memory of my late beloved mother, Chief Mrs. J.O. Ogun, who gave me everything there is to give as a mother.

Contents

List of Figures

List of Tables

Glossary

AI: artificial intelligence
ANN: artificial neural network
ASR: automatic speech recognition
CER: character error rate
CTC: connectionist temporal classification
dBFS: decibels relative to full scale
GAN: generative adversarial network
GMM: Gaussian mixture model
GPU: graphics processing unit
HMM: hidden Markov model
LLM: large language model
LSTM: long short-term memory
MAS: monotonic alignment search
MCD: Mel-cepstral distance
MFCC: Mel-frequency cepstral coefficients
MHSA: multi-head self-attention
MLP: multi-layer perceptron
MOS: mean opinion score
MUSHRA: multiple stimuli with hidden reference and anchor
NLP: natural language processing
PESQ: perceptual evaluation of speech quality
RIR: room impulse response
RNN: recurrent neural network
RNN-T: recurrent neural network transducer
RTF: real-time factor
SNR: signal-to-noise ratio
STFT: short-time Fourier transform
TTS: text-to-speech
VAE: variational autoencoder
VC: voice conversion
WER: word error rate

1

Introduction

1.1 Motivation

The explosion of information and technology has made people increasingly reliant on digital devices such as smartphones, computers, digital assistants, and smartwatches for communication, scheduling events, working, etc. They have become ubiquitous. They can be found in our pockets, homes, places of work, cars, etc. Admittedly, they increase the efficiency of work and serve as human support in day-to-day activities.

Any digital device (or service) requires a process for collecting and understanding the user's request. Undoubtedly, interaction with it needs to be smooth, effective, and accessible. In other words, the interactive interface needs to be easy to use by the owner.

As depicted in Figure 1.1, the most common forms of interaction are through touch using mechanical buttons, touch buttons or a touch screen, typing using a keyboard and a display interface, and speech using an internal microphone and a loudspeaker. Although, every digital device does not necessarily require all these elements, some of them are important for specific services and for specific groups of people. The hands-free nature of speech makes it a smooth and natural form of interaction with digital devices like voice assistants, as it mimics spoken human-to-human interaction. Unlike touch and typing, it is also easily handled by children, the elderly, and the physically handicapped.[1] Finally, one develops the ability to speak from a young age whereas writing and digital literacy are developed at a later stage of life through education.

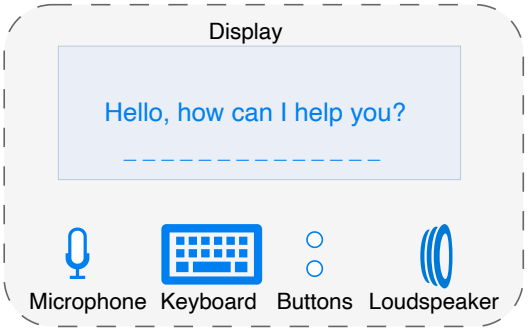


Figure 1.1: Analogy of a digital device interface.

[1] The handicapped includes the speech-impaired. They may need to use other forms of available interaction.

Technically, digital devices that use speech as a form of interaction are equipped with an automatic speech recognition (ASR) system that transcribes the speech. As shown in Figure 1.2, an ASR system, otherwise known as a speech-to-text system, converts the digital voice from the microphone to text for further processing of the request. In general, an ASR system is a key component of digital services built on top of speech data such as call centre analytics, production of automated meeting minutes, clinical documentation, podcast transcription, speech translation, language learning, etc. This implies that the accuracy of the service depends heavily on what the ASR system has transcribed.

Figure 1.2: Speech transcription using Automatic speech recognition (ASR).

Accurately transcribing speech is still a challenging task in general, and ASR transcription accuracy drops even further for low- and medium-resourced languages and in specific application domains where the data available for training the system is not large or diverse enough. Nevertheless, as digital technology and digital services continue to complement our work, making ASR systems that can accurately transcribe speech is very important. Also, it will no doubt make technology more inclusive and accessible to the majority of people. Likewise, it will close the digital divide for the handicapped, the elderly, speakers of low-resourced languages, and even non-native speakers of high-resourced languages who often have unsatisfactory experiences using these devices and services.

1.2 Context

ASR research has benefited from the development of machine learning algorithms and deep-learning-based methods in particular. In the last two decades, the error rate of large-scale multi-speaker ASR systems has drastically reduced from about 50 % to under 20 % for spontaneous speech and even lower for read speech (Srivastav et al., 2023; Radford et al., 2023), making them more useful in real-world applications. This improvement can be attributed to several factors including the creation of new ASR architectures, training algorithms, loss functions, inference algorithms, training datasets, and data augmentation, among others. Focusing on the training datasets, the datasets have become several times larger than what was customary. For instance, from LibriSpeech (Panayotov et al., 2015) of 960 hours to large multi-lingual datasets like MLS (Pratap et al., 2020) of 50,000 hours and VoxPopuli (Wang et al., 2021a) of over 400,000 hours. Additionally, research has shown that having a broader and larger dataset can positively improve ASR performance while the transcription accuracy is far reduced for small-sized and lower-quality training datasets (Szymański et al., 2020).

This is because human speech is dynamic and so many factors affect the speech transmission chain including the production, transmission, measurement, digital conversion, coding, reception, and perception of speech. Therefore, training data must cover these variability factors or diversity. Some examples of the typical variabilities in speech include speaking rate, loudness, pitch range, intonation, emotional state, reverberation, and noise, among others (Benzeghiba et al., 2007). Similar to the way the human ear has adapted to

understand speech despite these variabilities, ASR systems also need to be able to accurately transcribe speech irrespective of these variabilities. Hence, making an ASR system invariant to all these variables is a central task in learning good representations for ASR. A good representation of the data ensures that the ASR system can generalise to unobserved instances of speech in real-world applications. Interestingly, deep-learning-based ASR models, which are the most common ASR models deployed today, are even more data-hungry than previous approaches and require several views of training samples to generalise out of the training samples they have been trained on. In summary, to cover the mentioned variabilities, the training dataset must necessarily be diverse (Berrebbi et al., 2023).

Although large amounts of data have been curated for general domain applications and in highly resourced languages like English and French, many languages of the world and specific domains like medical speech still lack data of the magnitude that provides very low error rates in large-scale vocabulary ASR (Ardila et al., 2020). For example, specific applications like ASR for medical transcription typically have specialised vocabulary that is often under-represented in available ASR training data. Therefore, most large-scale vocabulary ASR systems perform poorly when deployed on this sort of target application. Furthermore, even though several of these problems can be solved by collecting more data, data collection is costly and time-consuming. In some situations, there is only a small population of speakers or they may not be accessible. Similarly, large-scale data collection is not often feasible in some ASR applications like ASR applied to children's speech, due to restrictions around privacy and consent. Therefore, most ASR training pipelines include data augmentation to create more diverse training instances (Kim et al., 2017; Park et al., 2019).

The method studied in this thesis involves ASR training data augmented with synthetic data generated from text-to-speech (TTS) systems and voice conversion (VC) systems. TTS systems can generate synthetic speech given a text prompt, and VC systems can change the characteristics of a given speech. Synthetic data has been explored for ASR data augmentation in recent years through speech chain (Tjandra et al., 2017, 2018), or by generating hidden states from text to train the ASR layers (Hayashi et al., 2018), or by directly generating synthetic data from text or VC (Mimura et al., 2018; Rosenberg et al., 2019). Although, these methods have shown significant improvement in ASR performance, directly generating synthetic data has been the most popular due to recent improvements in the quality of speech generated by TTS and VC systems (Kim et al., 2020; Tan et al., 2024). However, blindly generating more data does not lead to optimal results (Hu et al., 2022; Minixhofer et al., 2023) due to two major problems: a) the distribution of the synthetic data may not be similar to that of the real data, and b) one needs to consider what the real data lacks, and then try to fill the holes in the distribution of the real data. In other words, learning the distribution of the data could ensure that we can sample from this distribution to generate better and more diverse data for ASR data augmentation. In addition, limitations still exist in controlling the variabilities in the speech utterance during generation.

1.3 General problem and objectives

This thesis focuses on ASR data augmentation using diverse synthetic data generated by multi-speaker TTS systems and VC systems. The main objectives are severalfold.

First, we aim to train TTS/VC models using crowdsourced ASR datasets, as this is the type of data that is readily available for several applications and languages. ASR datasets contain a large number of speakers which can be useful for learning diverse speaking styles. However, directly using ASR datasets for speech synthesis does not yield high-quality utterances as the datasets are often noisy, reverberant, and recorded at low quality. In this regard, we examined the following questions: Can we train a high-quality TTS/VC system using general-purpose ASR data? How can we filter the ASR dataset to create a high-quality TTS dataset that still contains the required speaker diversity? How does the size and quality of the TTS dataset curated affect the naturalness and intelligibility of the generated utterances when used to train a TTS/VC model?

Secondly, we aim to increase the diversity of utterances generated by TTS and VC systems as this enables us to cover a larger distribution of variabilities in the data. Here, we focus on generative TTS models, and flow-based TTS models in particular, because they can inherently generate diverse utterances. Although they have been shown to generate diverse utterances (Valle et al., 2021; Kim et al., 2021), in the zero-shot multi-speaker TTS scenario, flow-based models are unable to learn the diversity of speaking styles inherent to each speaker given the text and speaker embeddings (Shih et al., 2021). Here, we considered the following questions: To what extent do generative TTS models capture the diversity in speech representation? Can we apply generative modelling to learn variabilities such as phoneme duration and pitch of the utterance and to what extent would this capture the distribution of the data? How much does this increase the diversity of the generated utterances? In addition, does the diversity of the utterances increase or decrease the naturalness and the quality of generated utterances? To achieve this goal, we modify a standard generative-based TTS model and introduce generative flow-based predictors for duration and pitch which enables us to answer these questions and to control some variances affecting speech.

Lastly, we aim to evaluate the significance of the diversity of the TTS and VC data for augmenting real data for the task of training ASR systems. Experimental methods that examine the speech variabilities in the synthetic data independently and jointly are an interesting direction to determine the most important variabilities to consider for ASR data augmentation. As such, we answer the following questions: Does the diversity of TTS utterances significantly impact the performance of ASR? What is the best method to select sentences for generating the TTS data? Which speaker characteristics in speech are the most important for improving ASR? What is the role of environmental noise and reverberation in the synthetic training data on the performance of the ASR system? Also, what are the best hyperparameters to combine these factors during generation? To answer these questions, we train ASR systems using a combination of real data and different sets of generated synthetic data containing the variabilities we wish to evaluate.

1.4 Contributions

1.4.1 Curating a multi-speaker TTS dataset from an ASR dataset

Instead of relying on studio-quality TTS datasets or TTS datasets curated from audiobooks for TTS training, we propose to use typical crowdsourced ASR datasets that are easily available for several speech applications and languages. Although large, crowdsourced ASR datasets inherently exhibit a large speaker variability (in terms of accent, speaking style, speaking rate, etc.) required for TTS systems to model diverse speakers,

the quality of the recordings varies and a significant amount of recordings contain noise and unintelligible content. In this line of research, we focus on automatically selecting high-quality TTS training samples from a crowdsourced dataset for generating high-quality and intelligible utterances for speakers seen and unseen during training. In this context, quality cannot be estimated via subjective listening tests since the ASR dataset is large or via objective metrics like PESQ that require a reference signal.

Therefore, we develop a method of dataset curation from an ASR-designed corpus for a TTS task. This method leverages the increasing accuracy of deep-learning-based, non-intrusive quality estimators (Lo et al., 2019; Cooper et al., 2022) to filter high-quality samples. We explore filtering the ASR dataset at different thresholds to balance the size of the dataset, number of speakers, and quality. We compare the dataset quality at different thresholds using a typical TTS training pipeline. That is, we evaluate the generated utterances from the multi-speaker TTS model trained individually on each of the filtered datasets. From the experiment, we show that by filtering the dataset, the highest quality dataset can obtain similar TTS dataset quality as a TTS dataset such as LibriTTS (Zen et al., 2019) curated from relatively higher-quality audiobooks. In addition, we show the effect of denoising the ASR utterances on the quality of the dataset.

1.4.2 Improving the naturalness and diversity of generated utterances

Previous TTS/VC systems have attempted to capture speech variabilities like pitch, speaking rate, accent, etc., by conditioning the system on a speaker embedding (Wang et al., 2018b) or by using discriminative models to learn these variabilities (Łańcucki, 2021). These two approaches are however not suitable for creating diverse datasets. In another direction, generative TTS/VC models are one type of TTS/VC models that are inherently capable of synthesizing diverse utterances (Kim et al., 2021; Miao et al., 2020) by sampling from a distribution to generate speech. Although they excel at generating high-quality utterances, the diversity is limited by the noise used during generation (Kim et al., 2020). Therefore, there is limited controllability.

Therefore, we tackle this problem by designing an improved flow-based TTS/VC architecture that learns the distribution of different speech variables that affect the diversity of speech. Here, we modify the TTS/VC model by integrating generative-based models for learning the variabilities. We find that our modifications significantly increase the diversity and naturalness of the generated utterances over baselines. In addition, we compare the distribution of speaker variability in real data to the distribution of this variability in the utterances generated by our newly improved TTS/VC model and the baselines. Here, we observe that our modifications to the TTS/VC model help to match the speaker variability in the real data more closely than other TTS/VC models.

1.4.3 Evaluating the impact of TTS data diversity on ASR performance

As opposed to naively generating synthetic data for ASR data augmentation, we examine different approaches to generating diverse data using our trained TTS/VC models. Diverse data includes data with a similar distribution to real speech data and data that files holes in the real data distribution. Therefore, we examine the data generation process for different speech variabilities.

Firstly, we develop a method to select the most informative sentences from a large text corpus. We show that ASR models trained with a combination of real data and utterances

generated using the selected text improve ASR performance over randomly selecting text. Secondly, we increase the diversity in the speaking rate and pitch by augmenting the phoneme duration and the predicted fundamental frequency. In addition, we develop a speaker selection method to select new speakers. Here, we find that our augmentation and selection methods can independently have a positive impact on the performance of the ASR model. Finally, we consider the effect of environmental factors such as noise and reverberation in the synthetic data on ASR performance. We observe that adding noise and reverberation after generating the utterances significantly improves the performance and robustness of the ASR system.

In conclusion, our experiments provide insights into the variabilities that are particularly important for ASR. With these insights, we generate different sizes of synthetic data that combine all the important variabilities and then combine them with real data to train ASR models. Our augmented data leads to a significant reduction in the character error rate (CER) and the word error rate (WER) of the trained models over individual experiments or baselines.

1.5 List of publications

Our contributions have been published in the following conference papers. We also specify in each chapter the publication related to the content presented.

- **S. Ogun**, V. Colotte, and E. Vincent. Can we use Common Voice to train a multi-speaker TTS system? In Proc. SLT. IEEE, 2023. pp. 900-905.
- **S. Ogun**, V. Colotte, and E. Vincent. Stochastic pitch prediction improves the diversity and naturalness of speech in Glow-TTS. In Proc. Interspeech 2023, 2023, pp. 4878–4882.
- **S. Ogun**, A. Owodunni, T. Olatunji, E. Alese, B. Oladimeji, T. Afonja, K. Olaleye, N. Etori, T. Adewumi. 1000 African Voices: Advancing inclusive multi-speaker multi-accent speech synthesis. In Proc. Interspeech 2024, 2024, pp. 1855-1859.

Other contributions during this thesis, but out of the scope of the main theme of this thesis, have led to the following secondary publications.

- T. Afonja, T. Olatunji, **S. Ogun**, N. Etori, A. Owodunni, M. Yekini. Performant ASR models for medical entities in accented speech. In Proc. Interspeech 2024, 2024, pp. 2315-2319.
- G. Coiffier, **S. Ogun**, L. Valque, P. Trivedi. State of the art. Think before loading, 2024, 978-2-9591975-0-5. hal-04509255

1.6 Organisation of the thesis

Chapter 2 provides an overview of the state of the art in the fields of speech synthesis and automatic speech recognition with a focus on augmentation methods for ASR training. As such, it introduces concepts related to speech, speech representations, variabilities in speech, TTS models, VC models, ASR models, and speech datasets.

Chapter 3 presents the methodology for curating a TTS dataset from an ASR dataset. We describe how our proposed method enables us to curate a high-quality dataset for training speech synthesis models. Then, we show how the method has been applied in other works for creating high-quality training datasets.

Chapter 4 deals with improving the diversity of utterances generated by the TTS/VC models. Here, we modify a Glow-TTS model and incorporate stochastic pitch and duration predictors to increase the diversity of utterances generated. We conduct comprehensive subjective and objective evaluations to measure the significance of these changes.

Chapter 5 evaluates the significance of TTS data diversity on ASR performance. In this chapter, we perform an experimental study to measure the performance gain that can be derived from synthetic data augmentation when the synthetic data is carefully created to be diverse. In particular, we independently and jointly evaluate the impact of the phonetic content, phoneme duration, pitch, number of speakers, data size, and the environmental characteristics of the synthetic data on the ASR performance.

Chapter 6 concludes this thesis by summarising the contributions and outlining some future research directions motivated by this work.

2

State of the art

In this chapter, we briefly present the fundamental concepts of speech as well as prominent speech attributes that contribute to the variability in speech. Then, we provide an overview of past studies to digitally generate speech using text-to-speech (TTS) systems and voice conversion (VC) systems. We also discuss the improvements that have been made to these systems to improve their performance and controllability. Alongside speech generation, we discuss previous and current methods for transcribing speech using automatic speech recognition (ASR) systems. Finally, we provide studies on improving the performance of ASR systems using data augmentation techniques including augmenting with synthetic speech data.

2.1 Speech variability

Speech is a system of communication among humans using spoken words or sounds. Hence, a speaker may perform several intentional speech acts, e.g., informing, declaring, asking, persuading, or directing while communicating through speech (Austin, 1975; Ingber, 1982). These speech acts affect the production of spoken words and are modulated by the spoken sounds (Searle, 1962). The speech acts can be formally described in terms of several speech attributes. Here, speech attributes refer to the rich information that facilitates human auditory perception and communication, beyond the sounds. This may include a set of fundamental speech sounds with their linguistic interpretations, a speaker profile encompassing gender, accent, emotional state, other speaker characteristics, and the speaking environment (Scherer, 1972; Krauss et al., 2002; Lee et al., 2007). Here, we attempt to broadly classify the speech attributes into three major categories namely linguistic attributes, paralinguistic attributes, and extralinguistic attributes.

2.1.1 Linguistic attributes

Oral language is employed in spoken communication to convey meaning or information to others. This meaning is organised symbolically through utterances and words, and then encoded acoustically through syllables and phonemes (a phoneme being the smallest distinctive speech unit in a given language). At all levels, the organisation of each unit is language-dependent, such as which phonemes, syllables, and words are employed and how they are allowed to follow each other. However, there are also certain common tendencies called linguistic universals (Mairal and Gil, 2006) which are a result of restrictions in

speech production and perception apparatus or due to other shared characteristics of natural languages.

Phonemes are defined in terms of their function, i.e., two sounds correspond to different phonemes if they can occur in the same context and distinguish different words of a given language, e.g., the phoneme sounds /f/ and /v/ in "fan" and "van" helps us to distinguish the two words in the English language. Additionally, phonemes can be divided into vowels and consonants. All vowels are voiced sounds while consonants are voiced or unvoiced, with a partial or full closure of the vocal tract (Kluender et al., 1988).

For a language where the phonemes, syllables, and words in that language are known in opposition to fully oral languages, phonetic transcription can be performed on words and often makes use of the International Phonetic Alphabet (IPA, 1999). For alphabetical languages like English, the number of characters is known and these character symbols can be used to compose syllables and words in a written form. In this case, similar combinations of characters may produce different phonetic sequences.

The goal of automatic speech recognition (ASR) is to focus on only the linguistic attributes while being invariant to the other attributes.

2.1.2 Paralinguistic attributes

Beyond the linguistic attributes of speech which focus on the literal transmitted message, additional information is conveyed in speech through paralinguistic attributes. These attributes are modulated onto or embedded into the verbal message, either in the acoustic content or in the linguistic content (Schuller et al., 2013). Paralinguistic attributes can include acoustic phenomena such as cough, laughter, filled pauses, which are often modelled in the same way as words in automatic speech processing. Additionally, they can denote the psychological state of a speaker in terms of emotion or mood (Mullennix et al., 1998; Wurm et al., 2001).

They can also be found in the prosody of the speech. Here, prosody refers to those patterns in speech that take place at time scales larger than individual sounds and includes variations in fundamental frequency (also referred to as F0 or pitch), duration of sounds, intensity, and voice quality (Bock and Mazzella, 1983; Bolinger and Bolinger, 1986; Kochanski et al., 2005; Xu, 2011). In a language like English, it can convey paralinguistic information such as speaking style. Also, it can convey linguistic and extra-linguistic information. For example, a high pitch can indicate anxiety and breathy voice may indicate attractivity, which can be embedded into the verbal message (Schuller et al., 2013). Also, in a language like English, a speaker who intends to ask a direct question will raise his pitch at the end of the sentence to convey this information (Delattre et al., 1962).

In addition to pitch, the duration of sounds can change its intended meaning. The length of a vowel in a word can convey surprise, confusion, or even change the meaning of the word in some languages such as Arabic and Finnish (Mitleb, 1984). For instance, the word "okay" in the English language can be interpreted differently depending on which vowel is emphasised since duration lengthening can be accompanied by a specific modification of pitch and loudness.

Likewise, the loudness level of speech can convey paralinguistic information. An example provided by Lago (2005) indicates that in Saudi Arabian cultures, loudness in discussions signifies strength and sincerity among Arabs; a soft tone implies weakness and deviousness, lower classes lower their voices, therefore respect is implied by lowering one's voice. Furthermore, the level of intensity may serve as an indicator of emotional state: a

speaker will tend to speak more loudly and at an unusually high pitch when he is excited or angry (Schuller et al., 2013).

2.1.3 Extralinguistic attributes

In addition to the discussed paralinguistic attributes, speech also contains some extralinguistic attributes. These include the speaker's identity, gender (Sweta et al., 2022), age (Shepstone et al., 2013), place of origin (Najafian et al., 2016), physical states such as alertness and sleepiness (Günsel et al., 2013), vigor or weakness, health or illness (Haulcy and Glass, 2021), etc. The length and thickness of the vocal folds depend on the gender and age, which in turn affect speech production. Also, the accent of a speaker is affected by the geographical origins and background of the speaker (the place where he lived or lives). Hence, one's place of origin may be determined through speech. Similarly, the state of one's health may affect speech production in several ways: production of some sounds may be affected by sleepiness, voice can become creaky due to obstruction in the vocal cavity, and speaking rate alongside intonation may change due to stress.

Finally, the environment where the utterance is made or recorded contributes to the perception of speech. It can significantly affect or occlude content in the received signal (Ris and Dupont, 2001). In general, the environment contains noise (Song et al., 2011) and reverberation, among others. Concretely, noise can be in the form of microphone noise, background music, speech, and stationary or non-stationary ambient noise. Also, the characteristics of a room affect the level of reverberation in the overlapping signal. Moreover, strong noisy conditions can imply modification of the pronunciation of sounds, popularly known as the Lombard effect (Junqua, 1993).

In conclusion, speech contains many attributes that ensure that there is no one-to-one mapping between a sentence and an utterance. However, ASR systems have to be able to transcribe a speech signal irrespective of the variability contained in the signal. The problem here is that it is impossible to cover all the variabilities in the real data used to train these ASR systems adequately. Therefore, this thesis will explore one method of augmenting the data by adding diverse synthetic data generated by speech synthesis systems. In addition, we will examine how some speech variabilities can be controlled when generating speech from a text prompt or during voice conversion.

2.2 Speech synthesis

Speech synthesis is a broad field that involves generating synthetic speech either from text (text-to-speech), linguistic features, or spectrograms (vocoding), or by changing the properties of speech, for example, speaker identity (voice conversion), emotion, speaking style, etc. (Donovan, 1996; Taylor, 2009; Tan et al., 2021). Before we discuss the speech synthesis systems themselves, we will briefly describe the input and output of the systems, and how they are processed.

2.2.1 General processing steps

The inputs and outputs of speech synthesis systems mainly include some processed form of text and speech. Although there have been different forms of inputs over the years, here, we focus on text-to-speech systems and voice conversion systems which use text and speech as inputs.

Text processing

The most popular text processing stages are text normalisation and text tokenisation.

Text normalisation is the process of transforming text data into a standardised form. It reduces variations in word forms to a common form when the variations mean the same thing. It involves converting text data into a consistent format by applying various transformations such as expanding titles, dates, and numbers into their written forms, expanding abbreviations into their full words if they have been pronounced as such in the paired speech, or converting text in different scripts into a universal script, etc. For example, the title "Mr." is normalised into "Mister" and the year "1980" is normalised into "nineteen eighty". There are open-source libraries for text normalisation, e.g., the NeMo text processing library (Zhang et al., 2021c) which can take into consideration the paired speech during text normalization and the uroman library which converts text in any script to the standard Latin alphabet (Hermjakob et al., 2018).

Text tokenisation involves breaking down the standardised text into smaller units called tokens. It entails converting the processed text into a form that eases pronunciation in speech synthesis (Tan et al., 2021). The most popular text tokenisation output format for TTS is the phoneme. To convert text to phonemes, also known as phonemisation, one method is to use manually curated pronunciation dictionaries such as the CMU Pronouncing Dictionary[2] for the English language or Phonemizer (Bernard and Titeux, 2021) which covers phonemisation for many other languages. However, these dictionaries cannot cover the pronunciations of all words. Hence, grapheme-to-phoneme (G2P) conversion models (Yao and Zweig, 2015) help to predict the phoneme sequence of words and can consider neighbouring words for correct phonemisation. In addition, some speech synthesis models have used characters (Jonathan Shen et al., 2018) and subwords (Casanova et al., 2024) as inputs to their TTS systems.

Acoustic features

Acoustic features are representations of audio signals extracted from speech using signal processing techniques and serve as crucial components in various applications of speech processing. They provide valuable information about the spectral, temporal, and perceptual aspects of sound, enabling the extraction and quantification of different acoustic properties. The most commonly derived acoustic features for speech synthesis include aperiodicity (Griffin and Lim, 1985), spectral envelope (Morise et al., 2016), mixed excitation (McCree and Barnwell, 1995), fundamental frequency (or F0) (Kawahara, 2006; Wang et al., 2018a), Mel-generalised cepstral coefficients (MGC) (Tokuda et al., 1994), and Mel-spectrograms (Stevens et al., 1937; Shen et al., 2018). Additionally, there is a family of speech synthesisers that control the synthesised speech from a parametric speech representation, such as fundamental frequency (F0) and intensity, and more generally by using the parameters of a source/filter model (Kawahara, 2006; Morise et al., 2016). Therefore, for these systems, acoustic features such as F0, energy values, formants, etc., may also be provided for speech generation. We will discuss more about speech representations in Section 2.5.3.

[2] http://www.speech.cs.cmu.edu/cgi-bin/cmudict

2.2.2 Pre-deep-learning era of speech synthesis

Speech synthesis has evolved over decades to become what it is today. There have been several methods of speech synthesis from the period of using mechanical devices like organ pipes to computational methods such as formant-based synthesis (Klatt, 1980), articulatory synthesis (Maeda, 1979; Hill et al., 1995), concatenative corpus-based synthesis (Hunt and Black, 1996; Balestri et al., 1999), and then statistical parametric methods (Tokuda et al., 2002). Hoffmann (2019) discussed the history of speech synthesis in depth from the first known use of tubes and electronic transducers until the period of concatenative corpus-based systems. Here, we mainly focus on concatenative corpus-based systems and statistical methods, and how they tackled the problem of improving naturalness and diversity in generated speech.

A concatenative system produces speech by selecting small segments of speech from a large database of pre-recorded speech. The system optimises the segment selection through a concatenation cost and combines the segments to synthetise full utterances. A selection module determines units to concatenate based on parameters such as pitch, duration, position in the syllable, and neighbouring phones. In addition, since all words and sentences cannot be covered in an existing corpus, the corpus is split into smaller segments such as phones, diphones, or words, and stored in a database. Unsurprisingly, using larger units such as diphones or words usually results in a more natural synthetic speech (Conkie, 1999), and a digital signal processing step can still be applied when necessary to smooth residual prosodic jumps at segment boundaries (Balestri et al., 1999). Yet, concatenative systems require a corpus of speech with good phonetic coverage and careful unit selection of speech to achieve naturalness. A new database is also needed for every new speaker, language, or speaking style. Therefore, it is somewhat difficult to obtain diverse speech synthesis for the same speaker. This has been explored by Obin et al. (2012) and Obin et al. (2015) using parametric approaches.

Statistical parametric methods use machine learning models such as Hidden Markov models (HMMs) to generate the sequence of parameters corresponding to a source/filter model such as duration of phonemes, pitch, intensity, and spectral envelope (Tokuda et al., 2002; Yu and Young, 2010) and tree-based models such as decision trees and random forests for duration modelling (Black and Muthukumar, 2015). These methods help to improve the diversity of synthetic speech by varying the sequence of speech units used for the synthesis. However, they are not as good in naturalness and quality as the concatenative methods (Hoffmann, 2019; King, 2014). Also, the HMM-based statistical models are known to have an over-smoothing problem (Zhang et al., 2008) which occurs when spectra are smoothed causing a muffled sound in speech. This is caused by the HMM training objective, therefore several training objectives including global variance normalisation (Toda and Tokuda, 2007) were introduced to tackle the problem. Global variance normalisation penalises the global variance of the statistical parameters of the model to achieve a better estimation of the speech spectrum. Additionally, researchers began to experiment with hybrid systems that combine statistical parametric methods with the unit-selection method to improve the naturalness of the synthetic speech (Sakai and Shu, 2005; Wilhelms-Tricarico et al., 2013; King, 2014).

Nowadays, neural networks and deep learning are the dominant methods for building speech synthesis systems due to their versatility and performance (Zen, 2015; Tan et al., 2021).

2.2.3 TTS with deep learning

Deep learning is the use of many layers of artificial neural networks to model a system's behaviour (LeCun et al., 2015). Unlike classical machine learning algorithms, it applies multiple learned linear and non-linear transformations to the input data, which provides rich representations of the data, making it easier to transform the input tokens into speech. Deep-learning-based methods also reduce human knowledge in designing TTS systems, and they encourage more end-to-end approaches (Wang et al., 2017b; Arık et al., 2017).

The earliest neural TTS models improved the naturalness of the synthesised speech over the concatenative and parametric approaches (Jonathan Shen et al., 2018). Nevertheless, earlier systems were not practical in production because they were slow in generation and could only generate spectrogram or speech frames auto-regressively, i.e., the current output of the model depends on the previously generated outputs. Many of these TTS systems relied on recurrent neural networks (RNNs) such as long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997) and are commonly known as *autoregressive TTS systems* in the literature.

While many RNN-based TTS systems used attention mechanisms to enable better alignment between the sequence of phonemes and the Mel-spectrogram or speech waveform, they were still known to be unstable and suffer from repeating or skipping segments of speech (Zhu et al., 2019). Some of the systems also relied on a post-processing module to reduce the over-smoothness of the spectrogram frames.

Another category of TTS systems called *parallel or non-autoregressive TTS systems* (Ren et al., 2020; Kim et al., 2020) reintroduced explicit phoneme duration modelling and alignment mechanisms (Badlani et al., 2022) to reduce skipping. In addition, there was also the introduction of a generative modelling framework to TTS which has helped to reduce the over-smoothness of the Mel-spectrograms (Ren et al., 2022).

TTS systems can also be categorised into *multi-stage TTS systems* and *end-to-end TTS systems* based on their intermediate representations.

Multi-stage TTS systems

Multi-stage TTS systems generate an intermediate representation of speech such as a Mel-spectrogram from tokenised text input. They comprise an encoder, a decoder, and a separate vocoder (Tan et al., 2021). As shown in Figure 2.1, the encoder processes the tokenised text to generate contextual representations of the tokenised text while the decoder converts the contextual tokenised text representations into an intermediate representation. The decoder may also be conditioned on a speaker representation and some prosodic variabilities like pitch, phoneme duration, and energy (Ren et al., 2020; Łańcucki, 2021). The encoder and decoder form the acoustic model of the TTS system. Finally, the vocoder converts the Mel-spectrogram into a speech waveform (Kong et al., 2020).

A rising choice of intermediate representations is the discretised acoustic representation popularly known as *discrete codes* or *acoustic tokens* (Wang et al., 2023). These discrete codes are a result of vector-quantisation of acoustic representations of speech using vector-quantised variational autoencoders (VQ-VAE) (Gârbacea et al., 2019). The representation has been used in diverse tasks such as language modelling and speech completion (Borsos et al., 2023), speech synthesis (Wang et al., 2023; Le et al., 2023), and speech translation (Barrault et al., 2023). It was also used in the XTTS model (Casanova et al., 2024) that we finetuned for the experiments in Appendix A.3.1.

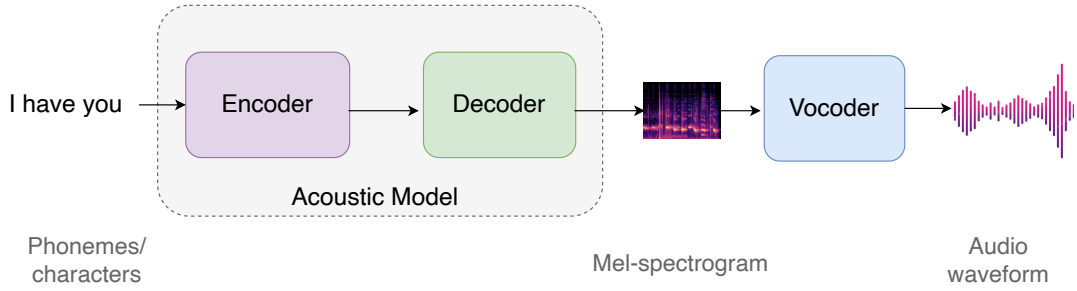


Figure 2.1: Components of a multi-stage neural TTS system.

Multi-stage speech synthesis systems are still in common use. An advantage of this system is that individual modules of the system can be independently trained and optimised for fast inference in different applications. However, errors from one module can easily cascade to other modules. In the deep-learning era, this can be reduced by fine-tuning one module in the TTS pipeline with the output of the previous module in the pipeline. For example, after first training the vocoder with Mel-spectrograms from real speech, the vocoder can be fine-tuned on samples of Mel-spectrograms generated by the acoustic model of the TTS system (Casanova et al., 2021).

Vocoder

Generally, the term vocoder (voice encoder and decoder) describes any electronic device, capable of analysing the speech signal, produce a parametric description of it, and resynthesise the speech signal from these parameters (Hoffmann, 2019). For instance, it was popular for decoding speech in telecommunications (Homer, 1955). Nowadays, a vocoder for speech synthesis uses software algorithms to convert an intermediate representation of speech into a speech waveform. This type of vocoding can be categorised into classical methods and deep-learning-based methods. Classical methods, such as the phase vocoders, were popular for estimating the phase of a spectrogram after pitch/duration modification of a sound signal (Kawahara, 2006).

In speech synthesis, the vocoder is used to recover the missing phase of the generated spectrogram. A classical iterative vocoder is the Fast Griffin-Lim algorithm (Griffin and Lim, 1984) which is an improvement based on the original Griffin-Lim (Griffin and Lim, 1984). It iteratively takes the prior estimate of the complex spectrogram, applies the short-term Fourier transform (STFT) backward and forward, and then reapplies the desired magnitude. Yet, a distortion caused by the phase reconstruction can be audibly noticed in the reconstructed speech, which reduces the naturalness of the speech. Therefore, deep-learning-based vocoders (van den Oord et al., 2016; Kalchbrenner et al., 2018) are now the preferred choice to synthesise natural and intelligible utterances from synthetic spectrograms.

Deep-learning-based vocoders convert intermediate representations to speech waveforms using deep-learning methods. Popular examples include MelGAN (Kumar et al., 2019) and HifiGAN (Kong et al., 2020) which enable the parallel generation of segments of an audio waveform from the spectrogram. The models use a Generative Adversarial Network (GAN) to better model the periodic patterns of the waveform. Additionally, they can produce high-quality speech with high sampling efficiency, even for languages or speakers that were unseen during training. They are currently the most common types of

vocoders for spectrogram inversion and are often used as universal vocoders (Song et al., 2022; Lee et al., 2023) due to their versatility.

Finally, although the input to vocoders has varied over time, the Mel-spectrogram has become the most popular intermediate representation in multi-stage TTS systems probably due to its compactness and interpretability (Wang et al., 2017b). More recently, deep-learning-based audio codecs have also been employed to reconstruct a speech signal from acoustic tokens, and therefore do not require training a vocoder (Défossez et al., 2023).

End-to-end TTS systems

An end-to-end TTS system produces a speech waveform given a tokenised text input (Donahue et al., 2020). The system can comprise an encoder and decoder, with a vocoder to convert its intermediate representation to audio as shown in Figure 2.2. Other architectures make use of an encoder-decoder or even just a decoder in addition to an audio codec to generate speech from text (Wang et al., 2023).

The intermediate representation of the audio does not need to be a human-constrained representation such as a spectrogram, allowing the model to learn rich features for better production of speech waveforms. Furthermore, the vocoder module is trained jointly with the rest of the system and may be fused into the decoder (Donahue et al., 2020) unlike in a multi-stage TTS system where the vocoder is separately trained. Nonetheless, there are two challenges with using the waveform as the target. First, the waveform contains more variance information (e.g., phase) than Mel-spectrograms, therefore the information gap between the input and output is larger than that in text-to-spectrogram generation (Ren et al., 2020). Second, it is difficult to train on the audio clip that corresponds to the full-text sequence due to the extremely long waveform samples and limited GPU memory (Kim et al., 2021). To illustrate the gap, consider that a speech waveform of 1 second at 16 kHz sampling frequency corresponds to about 3–5 words or 10 phonemes and 16,000 floating numbers. The first problem can be solved by applying adversarial training in the waveform generator to force it to implicitly recover the phase information by itself. Therefore, the vocoder part of the system is typically a modified GAN-based vocoder such as HifiGAN. Additionally, the vocoder layers of the system can be trained on short clips that correspond to a partial text sequence to reduce memory usage.

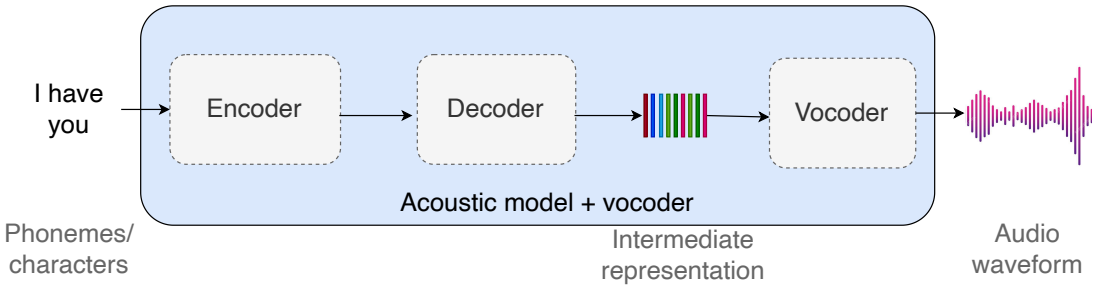


Figure 2.2: End-to-end TTS system pipeline.

2.2.4 Adding variability in speech synthesis systems

A range of methods has been proposed to vary the information added to the utterances generated by TTS systems. We first discuss variance adaptors that are used to add variation to speech, and then, talk about speaker representations in TTS systems. We wrap up the section with multi-speaker TTS systems and the generative models that are popularly used in state-of-the-art TTS architectures for increasing the naturalness and diversity of TTS utterances.

Variance adaptors

Variance adaptors are neural modules introduced into TTS models to explicitly learn to predict or generate prosodic attributes used in the speech synthesis pipeline (Ren et al., 2020; Łańcucki, 2021). They provide control over the individual prosodic attributes of speech and remove the constraint that the contextual tokenised text embeddings and speaker representation have to perfectly contain all the attribute descriptions in their limited embedding dimensions.

In a typical TTS system with variance adaptors, as depicted in Figure 2.3, the contextual tokenised text representation from the TTS encoder alongside the speaker representation is provided to each of the variance adaptors to predict phoneme-level or frame-level speech attributes like phoneme duration, fundamental frequency (F0), energy, etc. The predicted attributes can then be modified or varied before combining them with the tokenised text representation.

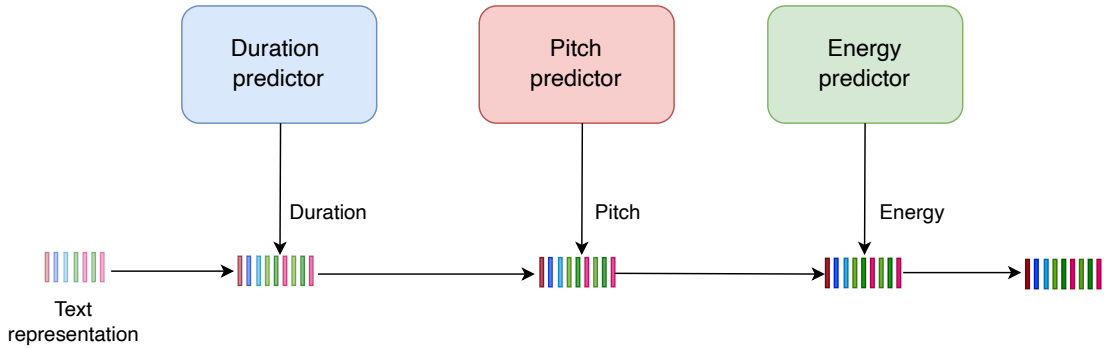


Figure 2.3: Adding speech attributes to tokenised text embeddings using variance adaptors.

The duration predictor is used to predict the number of times an intermediate representation has to be duplicated during inference. It is often trained using duration targets extracted from the speech. This target can be curated through an auxiliary pre-trained TTS model (Ren et al., 2019), through phoneme duration targets predicted by a forced alignment model (Ren et al., 2020), or using the duration computed from a simple alignment algorithm that aligns the spectrogram or intermediate representations to the tokenised text embeddings (Kim et al., 2020; Badlani et al., 2022). Alignment algorithms work by creating a path from the tokenised text input to the output spectrogram to discover the most probable path. They have the property that they are monotonic and they are stable, i.e., they do not skip any input token. They rely on similar principles as the connectionist temporal classification (CTC) algorithm (Graves et al., 2006). An example of such alignment algorithm is the Monotonic Alignment Search (MAS) algorithm used in

Glow-TTS (Kim et al., 2020). At inference time, the duration predictor simply predicts the phoneme duration directly.

Similar to the phoneme duration predictor, a pitch predictor predicts a representation of pitch such as the F0 at inference time. For training the pitch predictor, F0 values can be estimated for each Mel-spectrogram frame of the audio using a pitch extraction algorithm (Gerhard, 2003). The accuracy of the estimated F0 values depends on the pitch estimation algorithm and there can sometimes be a trade-off between the accuracy and the speed of estimation. Due to the large range of the F0 values, the values are transformed into log-F0 (Łańcucki, 2021) or Continuous Wavelet Transform (Ren et al., 2020) values when used as targets.

Variance adaptors can either be non-generative deterministic models (Łańcucki, 2021) or generative models (Lee et al., 2022) such as Normalising flows (described in Section 2.2.4). Generative variance adaptors have the advantage that they can increase the diversity of the generated speech utterance (Shih et al., 2021) by sampling from their latent distribution, which can be a useful feature in multi-speaker TTS synthesis where speakers can have different speaking styles, pronunciations, speaking rates, and intonations.

Hence, to control speech variabilities using variance adaptors, there are several possibilities: a) manually manipulate their representations, e.g., change the predicted duration of a phoneme, modify the predicted F0, etc,. b) sample diverse variance predictions using a generative modelling framework, or c) drive the variance adaptor prediction using a speaker representation.

Speaker representation in speech synthesis systems

A speaker vector represents speaker attributes in a speech utterance in a compact way, i.e., as a vector of fixed size, regardless of the length of the utterance. A speaker representation contains various pieces of information regarding a speaker (Wang et al., 2017a) including the voice timbre, speaking style, gender, etc. Over the years, different vector representations have been developed from the basic one-hot encoding vector to deep-learning features. I-vectors were a popular type of speaker representation based on statistical parametric modelling of speech features and dimensionality reduction of the learned features (Dehak et al., 2011; Reynolds et al., 2000) while the deep-learning-based speaker representations known as x-vectors or d-vectors are used nowadays due to their performance (Snyder et al., 2018). In the latter, a deep learning model is optimised in the context of speaker recognition or speaker verification, and the speaker representations from the optimisation are averaged to form d-vectors. They are useful for several tasks including speaker identification, language identification, and speaker verification, among others. In the context of speech synthesis, they add variance information regarding a speaker when generating speech in multi-speaker speech synthesis systems.

Learning a speaker's representation can be approached in different forms. First, a fixed-size embedding vector for each speaker can be learned jointly with the model during training (Kim et al., 2020). Although vectors learn to encode all the information regarding a speaker, new vectors have to be learned for new speakers not seen during training. As such, external speaker representations are desirable for generalisation of the system to unseen speakers (Casanova et al., 2021). Finally, it is also possible to train a speaker encoder module with the speech synthesis system (Wang et al., 2018b) which can provide rich representations tailored to the task and can still be used to extract speaker representations for speakers unseen during training.

The use of an external speaker embedding requires a neural speaker embedding system trained for this task. The information carried by the speaker embedding depends on the task the embedding system was trained on, which is typically a form of supervised speaker classification or speaker verification. For the speaker verification task, one common objective is to maximise intra-speaker embedding similarity while minimising inter-speaker embedding similarity (Wan et al., 2018). During training, positive labels from the same speaker and negative labels from other speakers are contrasted, therefore the model learns to push negative labels away in the embedding space while keeping positive labels close. When iteratively done for each speaker in the dataset, embeddings from similar speakers cluster around their centroid as shown in Figure 2.4.

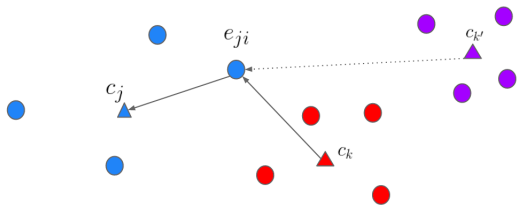


Figure 2.4: Intra-speaker embeddings are closer to their centroid c_n, $n \in \{j, k, k^{'}\}$. Embedding e_{ji} is made to be close to centroid c_j and far away from other centroids [source: Wan et al. (2018)].

Knowing that it is impossible to collect recordings from all speakers in the world for training and the number of speakers in a dataset is limited, some recent works have explored generating speaker embeddings for unseen speakers. This can be applied in speech synthesis systems to generate speech for unseen speakers. One such method aims to learn the speaker embedding distribution (Stanton et al., 2022; Watanabe et al., 2023) and to sample new speakers from it. Another method applied a GAN-based approach to generate realistic speaker embeddings different from those seen in training (Lux et al., 2023).

Multi-speaker TTS

Multi-speaker TTS systems add extra speaker-dependent conditioning to the TTS model to generate speech in the voice of a specific speaker. Several single-speaker TTS systems have been adapted to produce speech for multiple speakers with the addition of a speaker adaptation layer (Kim et al., 2020; Casanova et al., 2021). One advantage of multi-speaker TTS systems over single-speaker TTS systems is that we do not need to train independent systems for each speaker. Also, data from multiple speakers can be combined even when some speakers in the dataset only have a few samples (Luong et al., 2019). Using unseen speaker representations, we can also generate speech for speakers unseen during training (Casanova et al., 2021), which is popularly called *zero-shot TTS*.

As discussed above, it is usually not possible to control the different prosodic attributes of the generated speech using only the speaker representation. Therefore, introducing variance adaptors into the multi-speaker TTS model provides the model with more control of the generated speech attributes. Another method is to use generative models to learn the distribution of the data which can then be sampled from at inference.

Generative TTS models

Generative deep-learning models are models that learn the distribution of the training data and therefore have a latent representation or distribution that can be sampled from. The most common ones used in TTS are GANs (Bińkowski et al., 2019), Variational Autoencoders (VAE) (Sun et al., 2020), Normalising Flows (Kim et al., 2020, 2021; Shih et al., 2021), and Diffusion models (Popov et al., 2021). Another characteristic of generative models is that they relax the assumption in non-generative TTS models that there is a one-to-one mapping between the tokenised text input and the speech (Ren et al., 2020), bringing them closer to matching the intrinsic diversity of real-world speech. Flow-based generative models, in particular, can fit complex data distributions and generate diverse utterances with high quality. They have also been shown to be on par with recent diffusion probabilistic models for acoustic modelling (Zhang et al., 2023). In the following, we discuss more on normalising flows and also provide some mathematical formulation of the model as it is a main part of the TTS and VC system used in this thesis.

Normalising flow

Normalising flows (Kingma and Dhariwal, 2018) are a class of generative models that learn an invertible mapping between two probability distributions. Like other generative models, they make it possible to sample new data and to compute the likelihood of a sample. In many real-world applications, the data-generating distribution is either complicated to evaluate or completely unavailable. Therefore, an efficient and exact way to characterise such complex target distribution is to construct a map between the target distribution and a simple reference distribution (Kobyzev et al., 2021).

Given a random variable x and an invertible transformation f such that $x = f(z)$, the probability distributions of z and x are related via the classical change of variable formula:

$$p_x(x) = p_z(z) \left| \det \frac{\mathrm{d}z}{\mathrm{d}x} \right|^{-1} \tag{2.1}$$

where $\det \frac{\mathrm{d}x}{\mathrm{d}z}$ is the determinant of the Jacobian of the transformation that determines the amount of volume distortion.

As an example of this invertible transformation, let's consider a simple affine transformation of a 1-dimensional random variable. If $p_z(z)$ is the distribution Uniform(0, 1) and $x = a + b.z$ where $a \in \mathbb{R}$ and $b \in \mathbb{R}\backslash\{0\}$, then $\mathrm{d}x/\mathrm{d}z = b$ and $p_x(x)$ can be described as a simple affine (scale and shift) transformation of the source distribution $p_z(z)$.

For practical applications of these transformations in normalising flows, transformations should be invertible and should have easily computable Jacobians. Several transformations satisfying these properties have been proposed including Real-valued non-volume preserving (Real-NVP) transformations (Dinh et al., 2016), Generative Flow with Invertible 1x1 Convolutions (Glow) (Kingma and Dhariwal, 2018), Spline flows (Durkan et al., 2019), and Affine coupling transformations (Teshima et al., 2020).

Instead of a single transformation, normalising flows compose multiple transformations from one probability distribution to another probability distribution. We start from a simple distribution and gradually transform it into a more and more complex distribution. In this case, an appropriate simple distribution is a distribution we can sample from such as the multivariate Gaussian. Transforming this simple distribution into a more complex distribution enables us to generate data such as speech, as illustrated in Figure 2.5.

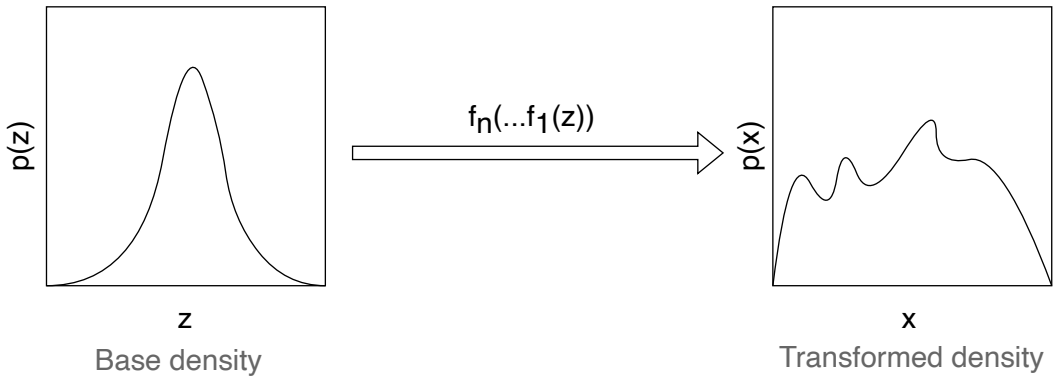


Figure 2.5: Transformation of probability density from one form to another using a series of invertible transformation functions.

Specifically, a normalising flow is a collection of nested invertible transformations $f = f_1.f_2.f_3...f_n$ such that $z_n = f_n(z_{n-1})$. Let p_i denote the probability distribution function of the ith random variable in the composite function. From Equation (2.1), applying the logarithm function to both sides of the equation,

$$\begin{aligned}
\log p_n(z_n) &= \log p_{n-1}(z_{n-1}) - \log \left| \det \frac{\mathrm{d}z_n}{\mathrm{d}z_{n-1}} \right| \\
&= \log p_{n-2}(z_{n-2}) - \log \left| \det \frac{\mathrm{d}z_{n-1}}{\mathrm{d}z_{n-2}} \right| - \log \left| \det \frac{\mathrm{d}z_n}{\mathrm{d}z_{n-1}} \right| \\
&= ... \\
&= \log p_0(z_0) - \sum_{i=1}^{n} \log \left| \det \frac{\mathrm{d}z_i}{\mathrm{d}z_{i-1}} \right| .
\end{aligned} \tag{2.2}$$

Equation (2.2) provides a direct way to compute the likelihood of an observation for some complex, real-data distribution given a prior p_0 and a set of invertible transformations f_1, f_2, f_3,..., f_n. We note that these transformations are usually parameterised as neural network layers that can be learned using maximum likelihood estimation.

Unlike VAEs that optimise the lower bound (ELBO) to infer a distribution over latent-variable values or GANs that learn to fool a discriminator network, normalising flows infer the exact latent-variable values, the representations of the data useful for downstream tasks. Normalising flows have also been extensively applied to speech synthesis and it is the decoder architecture used in popular TTS models like Glow-TTS (Kim et al., 2020) and VITS (Kim et al., 2021).

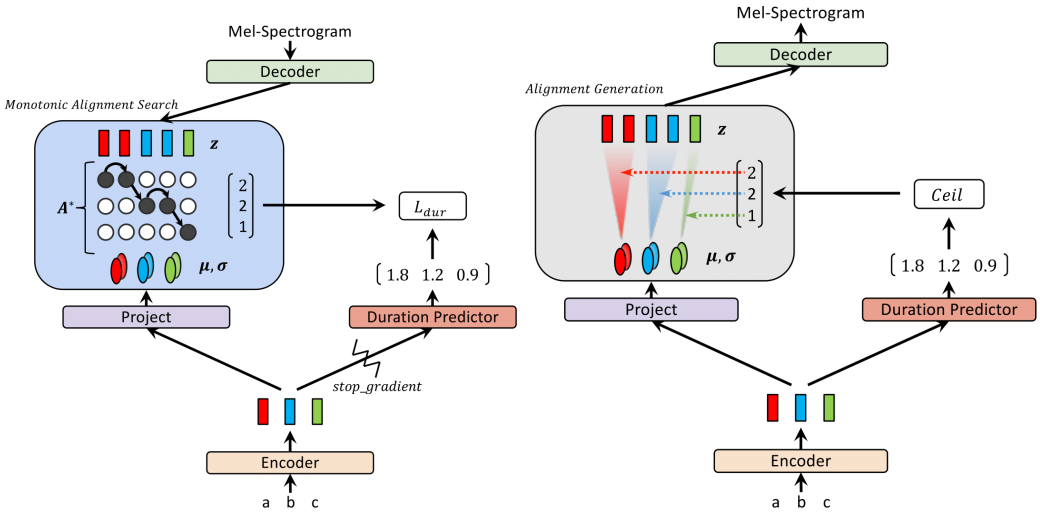
Glow-TTS

We have chosen to describe the Glow-TTS model in this thesis as it will be our baseline TTS model. Additional modifications are also made to the TTS model in other chapters. We have selected this multi-stage flow-based TTS model as it is capable of generating high-quality and diverse utterances, with a limited amount of training data and training time. Additionally, using a multi-stage TTS model helps us to focus on the acoustic model without influence from the vocoder.

The Glow-TTS architecture comprises a Transformer-based encoder, a flow-based decoder, and a deterministic duration predictor. The Transformer encoder generates contextual tokenised text embeddings from the tokenised text input, which are then linearly projected into an 80-dimensional representation of the mean μ of the prior distribution. Given μ, a scalar noise temperature T, and a random vector ε sampled from the standard normal distribution, the latent representation z sampled from the prior distribution can be expressed as

$$z = \mu + T\epsilon. \tag{2.3}$$

where $\epsilon \sim \mathcal{N}(0, 1)$. The decoder is therefore invertible, i.e., can invert the latent representation into a Mel-spectrogram and vice-versa. Figure 2.6 shows the Glow-TTS training and inference procedures.



(a) During training. (b) During inference.

Figure 2.6: Glow-TTS architecture [source: Kim et al. (2020)].

At training time, the statistics of the latent distribution, mean μ with standard deviation ε are predicted by projecting each tokenised text embedding to an 80-dimensional vector. The 80-dimensional vectors are then expanded to match the duration of the corresponding spectrogram using Monotonic Alignment Search. In practice, the standard deviation can be fixed at zero (as implemented in the standard model) which is equivalent to a T of 1 in Equation (2.3) at training time. At inference time, a certain value of T (often smaller than 1) is chosen, and a latent representation z is sampled from the prior distribution as input to the decoder to generate a Mel-spectrogram. The value of T has an impact on the pitch of the speaker while varying the value of ϵ produces different stress and intonation.

Glow-TTS can similarly be used for voice conversion, which is the theme of the next section. This allows us to train a single model for TTS and voice conversion, without complicating our synthetic data generation pipeline or increasing model training time.

2.3 Voice conversion

Voice conversion (VC) is the process of modifying an input utterance by changing the source speaker's voice or style into a target speaker's voice or style while preserving the

linguistic information (Abe et al., 1990). Depending on the system, it may preserve the original prosody or convert it to that of the target speaker or style (Kim et al., 2020). VC can also be used to change speech attributes thereby increasing the diversity of the speech data. We consider this approach for augmenting the ASR training data in this thesis.

A typical VC pipeline, as shown in Figure 2.7, includes a feature extraction module, a mapping module, and a resynthesis module (Sisman et al., 2020). The feature extraction module decomposes the speech signal into features that carry linguistic and paralinguistic information, the mapping module changes them into features of the target speaker or style, and the resynthesis module then generates the target speech waveform.

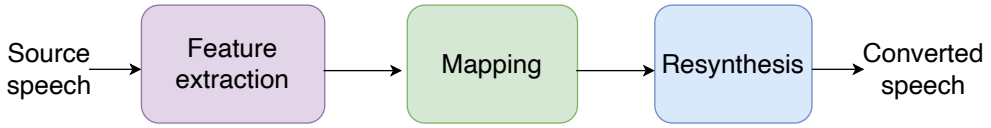


Figure 2.7: Typical voice conversion pipeline.

2.3.1 Pre-deep-learning era of voice conversion

There has been a continuous interest in the effective manipulation of speech properties since the debut of computer-based speech synthesis and the development of digital signal processing methods. Early VC systems include spectrum mapping methods based on vector quantisation (Abe et al., 1990) and dynamic time warping (Helander et al., 2008). They require parallel training data between the reference and target speakers. Statistical parametric approaches such as Gaussian mixture models (GMMs) (Toda et al., 2007), partial least square regression (Helander et al., 2010), and dynamic kernel partial least squares regression (DKPLS) (Helander et al., 2011) also benefit from more training data.

While it is more straightforward to train a mapping function from parallel training data than non-parallel training data, parallel training data is not always available and creating a large parallel corpus can be very cumbersome. However, there is an additional challenge of how to establish the mapping between non-parallel sources and target utterances. The INCA alignment technique by Erro et al. (2009) and the phonetic posteriogram (PPG) based approach (Sun et al., 2016) are some of the first methods developed for voice conversion with non-parallel training data. While the alignment technique doesn't use external resources, the PPG-based approach makes use of an automatic speech recogniser to generate an intermediate phonetic representation as the inter-lingual connection between the speakers. Several other statistical approaches to voice conversion include glottal source modelling (Childers, 1995), vocal tract length normalisation (Sundermann and Ney, 2003), negative matrix factorisation (NMF) (Takashima et al., 2012), and HMMs (Ye and Young, 2006). HMMs and GMMs have been greatly explored for voice conversion where the HMM is trained for a target speaker's data then the parameters of a GMM-based linear transformation function are estimated in such a way that the converted source vectors exhibit maximum likelihood with respect to the target HMM (Ye and Young, 2006). This approach showed comparable performance to methods using parallel data (Ye and Young, 2006) and did not require a large amount of training data, although the speech quality and similarity to the target speaker were not high. Deep-learning-based methods have advanced the state-of-the-art for voice conversion methods and have greatly improved the naturalness of generated utterances for the target speaker.

2.3.2 Voice conversion with deep learning

Deep learning eliminates the manual design of the analysis, mapping, and reconstruction functions, enabling complex non-linear transformations between the source and target speech. The non-linear transformations can be learned from parallel data or non-parallel data depending on the voice conversion method (Lorenzo-Trueba et al., 2018). It also allows the mapping module to learn from a large amount of speech data which significantly improves voice quality and similarity to the target speaker. Furthermore, as shown in Figure 2.8, in the deep learning era, the feature extraction module is called a neural encoder, the mapping function is called a neural decoder, usually incorporated with an alignment mechanism, and the resynthesis module is a neural vocoder. As the mapping function must align the input speaker's features with the target speaker's features, neural attention is one of the mechanisms for performing this alignment.

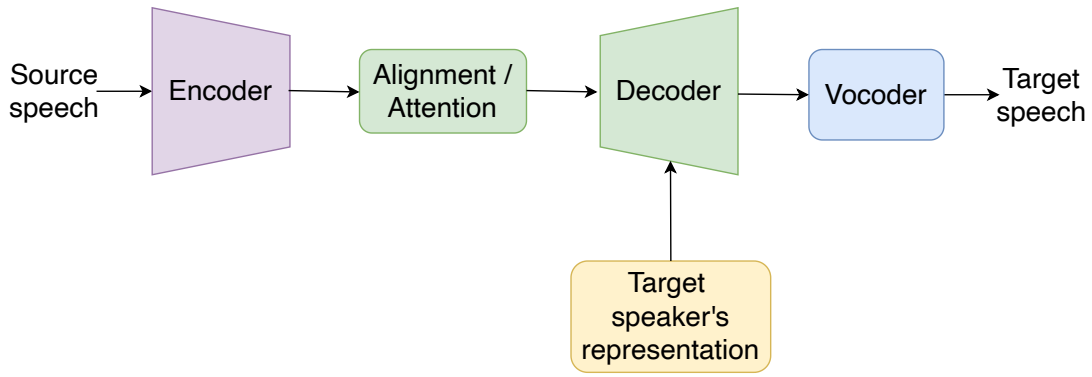


Figure 2.8: Auto-encoder pipeline for voice conversion.

Deep-learning-based methods have significantly benefited from non-parallel training data. As most current methods compress the representation of speech to extract only the linguistic content (and possibly the prosodic content) as depicted in Figure 2.8, the decoder of the VC model can re-add the missing content through the target speaker's representation. Therefore, the architecture typically takes the form of an autoencoder (Qian et al., 2019). Popular VC architectures such as Variational Auto Encoders (VAE) (Qian et al., 2019), Generative Adversarial Networks (GANs) (Wang et al., 2020b; Li et al., 2021), Normalising Flows (Merritt et al., 2022; Li et al., 2023), and vector quantised VC models (Wang et al., 2021b) use this training method. A multi-speaker dataset is still required for training the VC systems.

A neural vocoder in the VC pipeline ensures that the modified speech can be natural and non-robotic, without the need for tuning the vocoder for the target speaker. Vocoders have been discussed under TTS in Section 2.2.3.

Leveraging TTS models for voice conversion

An important aspect of voice conversion is to retain the linguistic content in the source speech when converting to the target speech. In this sense, VC systems and TTS systems are similar as they both aim to generate high-quality speech with the appropriate linguistic content. A TTS system provides a mechanism for the speech to adhere to the linguistic content, therefore the TTS knowledge can be leveraged by using a shared attention and/or a shared decoder in the TTS/VC system (Zhang et al., 2021b) as shown in Figure 2.9.

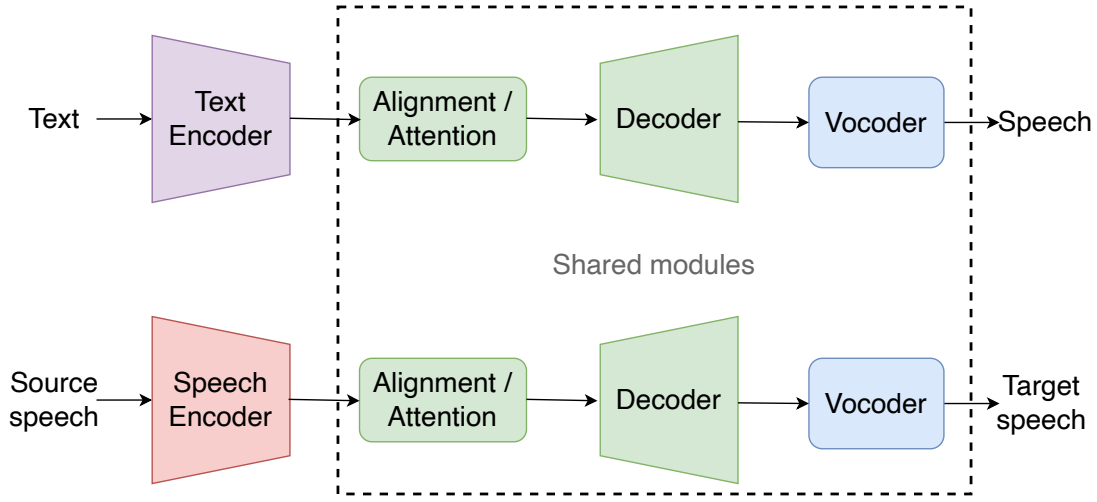


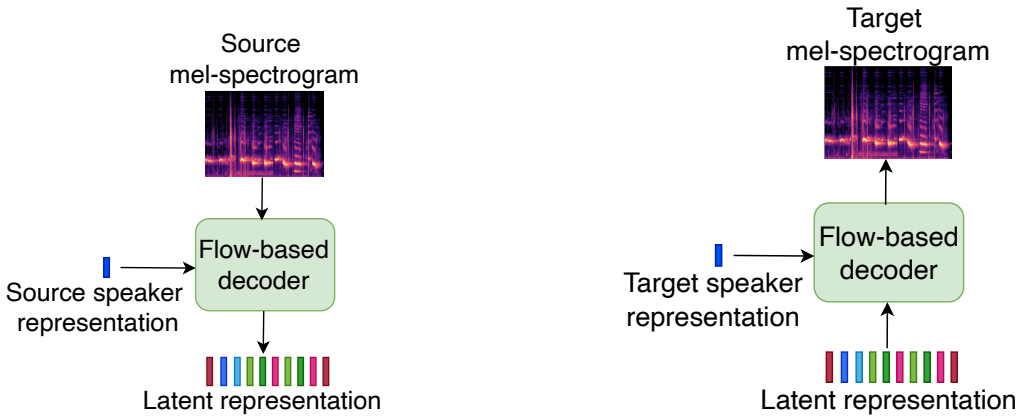
Figure 2.9: TTS/VC model where the voice conversion system leverages the TTS modules that are linguistically informed. The top arrows show the paths for TTS while the bottom arrows show the paths for VC. [source: Sisman et al. (2020)].

In this paradigm, two distinct content encoders encode the text and source speech into intermediate features. Then, a shared alignment/attention mechanism and a shared decoder are used to generate TTS speech or target VC speech. This also provides a mechanism to ensure that the representations derived from both encoders contain the linguistic content required for TTS and VC. A speaker representation and several other speech attributes can be provided to the decoder to add the necessary target speaker's attributes to the intermediate representation. Figure 2.9 shows the shared alignment and shared decoder architecture for VC where the VC system leverages the TTS modules for VC. The speaker representation and variance adaptors are not shown for brevity.

Similarly, for flow-based multi-speaker TTS/VC models like Glow-TTS (Kim et al., 2020, 2021), as shown in Figure 2.10a, the decoder is conditioned on the source speaker's representation during the inversion of the Mel-spectrogram into the latent representation of the speech. Therefore, the decoder learns to disentangle speaker information (timbre in particular) from the other content in the speech during training. The training objective reduces the Kulback-Leibler divergence between the intermediate representation of the Mel-spectrogram and the statistics derived from the contextual tokenised text embeddings, which shows that no speaker information is used to create the intermediate representation. At inference time, the source speaker's voice can be inverted into its latent representation as usual and then converted back to the target speaker's voice using the target speaker's representation as conditioning as depicted in Figure 2.10b. We leverage a flow-based multi-speaker TTS model for voice conversion in this thesis because the model provides a means to change some speaker properties in the latent distribution after inversion without affecting the phonetic content.

Disentangling linguistic content and prosody in voice conversion

The intermediate representations of the speech encoder used for VC may contain both linguistic information and prosodic attributes (Qian et al., 2020). Therefore, it is desirable to disentangle these two representations to better match the target speaker's prosody and



(a) Mel-spectrogram inversion

(b) Voice conversion

Figure 2.10: Flow-based VC showing the Mel-spectrogram inversion and voice conversion (VC) processes.

even for better controllability. The prosodic attributes may also provide some information regarding the source speaker, e.g., the speaking style of the speaker, which we would like to change to that of the target speaker. A method that has been explored involves drastically reducing the encoder dimension so that the model can prioritise retaining only the linguistic information from the source (Qian et al., 2020). In addition, extra speech information such as pitch is fed to the VC model so it doesn't have to store or recover it from its intermediate representation. Data augmentation such as pitch shifting, randomly dropping spectrogram frames, or randomly duplicating spectrogram frames helps to ensure that the model does not just memorise speaker attributes and must learn to use the provided information at inference time (Chan et al., 2022). Finally, a duration predictor can be used to modify the duration of the intermediate representation to match the target speaker's rhythm. In this thesis, we modify some prosodic attributes of the speech utterance such as phoneme duration and pitch during voice conversion to increase the diversity of VC generated utterances.

2.4 Evaluation of speech synthesis systems

Speech synthesis systems are difficult to evaluate due to the inherent variability of speech. Different evaluation methods have been proposed and each method may only capture an aspect of the evaluation. Therefore, it is customary to use several evaluation metrics. Table 2.1 lists some popular metrics.

The evaluation can take the form of a listening test where human evaluators rate the speech according to some specified criteria. These are called subjective tests because they depend on listeners' subjective perception. For naturalness and quality of speech, the most popular subjective method is the mean opinion score (MOS) (Loizou, 2011) evaluation. The 1–5 Likert scale is frequently used for MOS-like ratings where 1 is the worst rating and 5 is the best rating possible for that metric with increments of either 0.5 or 1.0. Typical scale labels for MOS ratings are: 1 - Bad, 2 - Poor, 3 - Fair, 4 - Good, and 5 - Excellent (Kirkland et al., 2023). A MOS evaluation can be used to quantify various criteria such as the noisiness, the naturalness, the intelligibility of speech or its similarity to the target

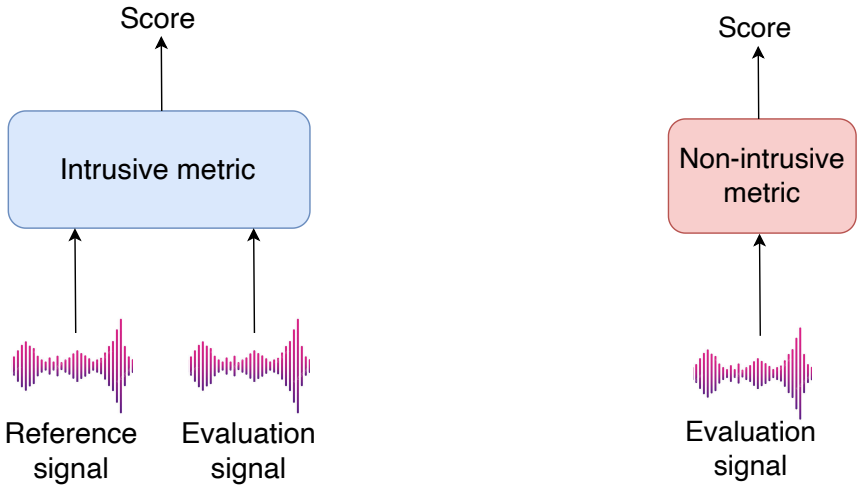
speaker, therefore the evaluation criterion must be specified to the listeners using clear instructions (Kirkland et al., 2023). Some speech synthesis evaluations use the Multiple Stimuli with Hidden Reference and Anchor (MUSHRA) listening test procedure (ITU-R, 2003). In MUSHRA, the listener is presented with a reference utterance, a certain number of test samples, a hidden version of the reference, and one or more anchors for evaluation. The anchors are customarily some low-pass versions of the reference that are expected to be given the lowest rating. Unlike MOS, MUSHRA uses a scale from 0 to 100. Subjective intelligibility tests are also performed using semantically unpredictable sentences (SUS) (Benoît et al., 1996) or minimal pairs of words (Hodge and Gotzke, 2011) which assess either the number of words a listener correctly transcribes or identify deficiencies in phoneme generation by the models.

Table 2.1: TTS metrics used for evaluating the quality, naturalness, speaker similarity to a target, intelligibility, and prosodic similarities of synthetic speech in comparison to natural speech.

Metrics	Method of evaluation
Quality and Naturalness	
Naturalness MOS	Listening test
Quality MOS	Listening test
MUSHRA	Listening test
PESQ	Intrusive test
MOSNet	Model based (non-intrusive)
WV-MOS	Model based (non-intrusive)
NISQA-TTS	Model based (non-intrusive)
Speaker similarity	
Speaker MOS	Listening test
MCD	Intrusive test
Cosine similarity	Model based (intrusive)
Comparative tests	
Comparative MOS	Listening test
Preference test	Listening test
Intelligibility	
Signal-to-noise ratio (SNR)	Non-intrusive
Word error rate (WER)	Model based (non-intrusive)
Character error rate (CER)	Model based (non-intrusive)
Minimal pair score	Listening test

Subjective evaluations are somewhat cumbersome as they require human input and are not always reproducible with separate groups of human evaluators and environments (Kirkland et al., 2023). Therefore, several quantitative metrics have been developed that usually correlate with human subjective evaluation. The metrics can be classified as either non-intrusive or intrusive.

An intrusive method, also called full-reference or double-ended method, compares the evaluation signal to a reference signal as depicted in Figure 2.11a. It estimates the speech quality by computing the distance between some features of the two signals and mapping this distance to a perceptual quality score (Shen et al., 2024). Some examples are the Perceptual Evaluation of Speech Quality (PESQ) (Rix et al., 2001) metric, and the Mel-



(a) Intrusive evaluation protocol

(b) Non-intrusive evaluation protocol

Figure 2.11: Intrusive and non-intrusive evaluation of utterances generated by speech synthesis systems.

cepstral Distance (MCD) (Kubichek, 1993) which compute the difference between the spectral features of the reference and the evaluation signal. A full-reference method is most applicable when the reference and the target speech are from the same speaker and the same text. The difficulty with this constraint is that the human speech generation process is a one-to-many process, not a strict one-to-one mapping, i.e., one text paired with one speaker can produce different prosody for the same style. Therefore, this method is not suitable for the evaluation of diversity in the generated speech. Non-intrusive methods, which are also called no-reference or single-ended methods, only rely on the evaluation signal to predict a quality score. This is depicted in Figure 2.11b. Non-intrusive speech quality predictors that have been recently developed use deep learning models to directly predict speech quality metrics like MOS and PESQ scores (Lo et al., 2019; Zhang et al., 2021d). These include neural models like AutoMOS (Patton et al., 2016), MOSNet (Lo et al., 2019), and NISQA (Mittag and Möller, 2020). In this approach, features are extracted from the speech using convolutional neural networks and recurrent networks, with a classification head used to summarise the features and to predict the quality score. Using a self-supervised feature extractor with a fully connected neural network classifier produced a higher accuracy and correlation with human scores (Cooper et al., 2022).

In the same direction, the intelligibility of the generated speech can be approximated by the word error rate (WER) or the character error rate (CER) of generated utterances measured against the reference text. Additionally, an objective speaker similarity measure is typically estimated using the cosine similarity (cos-sim) between a speaker embedding $\mathbf{r}$ extracted from the generated speech and the reference speaker's embedding $\mathbf{t}$,

$$\text{cos-sim}(\mathbf{r}, \mathbf{t}) = \frac{\mathbf{r} \cdot \mathbf{t}}{\|\mathbf{r}\| \|\mathbf{t}\|}. \tag{2.4}$$

The cosine similarity varies from -1 to 1 with values closer to 1 indicating that the speaker embeddings correspond to similar speakers.

2.5 Automatic speech recognition

Automatic speech recognition (ASR) is the ability to transcribe spoken language into text. An ASR system produces the most likely word sequence given a speech signal (Chen et al., 1996). Alongside TTS, ASR systems are an integral part of human-to-machine communication. Therefore, ASR needs to accurately transcribe speech, independently of the speech variabilities. In practice however ASR transcription accuracy varies due to variable speaker characteristics, recording conditions, and acoustic conditions.

ASR can be formally defined as follows: Given an input speech signal $\mathbf{x} = (x_1, x_2, ..., x_T)$ with length T, the ASR system outputs a label which is a sequence of phonemes, words or characters $\mathbf{y} = (y_1, y_2..., y_N)$, $y_n \in \mathbf{V}$, where $\mathbf{V}$ is the vocabulary. The most probable output sequence is given by

$$\hat{\mathbf{y}} = \arg\max_{y \in \mathbf{V}^*} \; p(\mathbf{y}|\mathbf{x}) \tag{2.5}$$

where $\mathbf{V}^*$ is the set of strings over $\mathbf{V}$. Following Bayes' rule, the posterior probability in the above equation can be expressed as the conditional probability $p(\mathbf{x}|\mathbf{y})$ of the word sequence given the acoustic observations multiplied by the prior probability $p(\mathbf{y})$ of the word sequence divided by the marginal likelihood $p(\mathbf{x})$ of observation sequences:

$$\hat{\mathbf{y}} = \arg\max_{y \in \mathbf{V}^*} \; \frac{p(\mathbf{x}|\mathbf{y})p(\mathbf{y})}{p(\mathbf{x})} \tag{2.6}$$

$$= \arg\max_{y \in \mathbf{V}^*} \; p(\mathbf{x}|\mathbf{y})p(\mathbf{y}). \tag{2.7}$$

A typical ASR model as shown in Figure 2.12 integrates this into its pipeline to form three different modules namely: a feature extractor, an acoustic model (encoder and decoder), and a language model, or this is directly learned as a single architecture as in end-to-end ASR systems (Prabhavalkar et al., 2024). The $p(\mathbf{x}|\mathbf{y})$ is calculated by the acoustic model and $p(\mathbf{y})$ is modelled by the language model. As shown in Equation (2.7), the marginal probability, $p(\mathbf{x})$, is discarded since it is constant with respect to the ranking of hypotheses and hence does not alter the search for the best hypothesis.

2.5.1 Pre-deep-learning era of ASR

ASR has long been a research topic of human-machine interaction. Early systems that recognised isolated digits for a single speaker were developed at Bell Laboratories (Davis et al., 1952) and phoneme recognisers were subsequently developed to recognise vowels and consonants (Fry, 1959). These ASR systems involved rule-based methods that require the knowledge of formants or template-matching using dynamic time warping. They work well for recognising digits, phonemes, and even isolated words, however, they fail when the variability in the speech signal is large or varies significantly from the template. Researchers at IBM developed large vocabulary speech recognition systems for different tasks for a period of almost two decades (Tappert et al., 1971; Jelinek et al., 1975). Additionally, researchers developed speaker-independent ASR systems for isolated words (Rabiner et al., 1979).

Statistical models such as Hidden Markov Models (HMMs) were later adopted for ASR to improve speech recognition for isolated words and continuous speech (Bahl et al., 1983). HMMs involve learning hidden or unobservable states (e.g., phonemes) to control

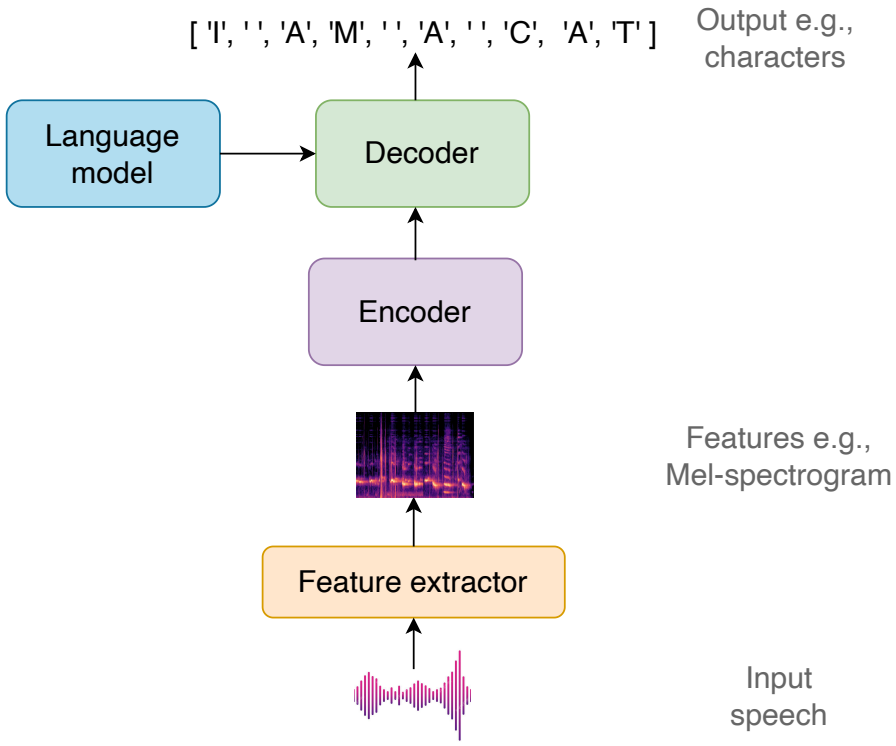


Figure 2.12: A typical neural ASR pipeline showing the feature extraction module, the encoder, the decoder, and the language model.

observable states (speech). The hidden states are not visible to the observer whereas the observed states are visible. As such, speech recognition was formulated as a problem of maximum likelihood decoding (Bahl et al., 1983).

HMMs were largely explored for ASR due to their high modelling accuracy in small data regimes and fast domain adaptation. Attempts were also made with Artificial Neural Networks (ANNs) for ASR and were successful in classifying short acoustic-phonetic units, such as individual phonemes, however, they were unsuccessful in dealing with longer speech signals (Lang et al., 1990). This is mainly due to inability to model long-term dependencies in ANNs. In the early 1990s, this led to the idea of combining HMMs and ANNs within a single model, broadly known as hybrid HMM/ANN model (Bourlard and Morgan, 1994; Chen et al., 1996; Chung and Un, 1996). As larger datasets became available and novel algorithms were discovered for deep learning, deep learning methods became the popular methods for ASR as they have a better generalisation power than HMMs or ANNs.

In the same period, the United States Department of Defense Advanced Research Projects Agency (DARPA) contributed significantly to ASR development by sponsoring a large research program aimed at achieving high word accuracy for 1011-word, continuous speech recognition, and database management tasks. This resulted in systems like the CMU SPHINX system (Lee et al., 1990) and the BBN continuous speech recognition system (Chow et al., 1987). Additionally, the curation of datasets such as the Wall Street Journal CSR corpus (Paul and Baker, 1992) and Switchboard corpus (Godfrey et al., 1992) presented more challenging datasets for continuous speech and spontaneous speech recognition.

2.5.2 Deep-learning-based ASR

Deep-learning-based ASR models use multilayer neural networks to extract useful features necessary for classifying segments of speech into phonemes, characters, or words. One of the first phoneme recognition systems based on neural networks was the time delay neural network (Waibel et al., 1989, 1995). It uses neural networks to classify patterns with shift-invariance (e.g., phonemes in speech, independent of time) and to model context in data.

Deep learning methods have been widely explored in creating large vocabulary ASR systems for continuous speech recognition (Vinyals et al., 2012) and in noisy environments (Barker et al., 2017). Recurrent neural networks, convolutional networks (Li et al., 2019), and self-attention networks (Sperber et al., 2018) are popular architectures for the ASR task. Given the architecture, one has to decide on a suitable alignment modelling approach between the input speech and text during training and inference. Nowadays, deep-learning architectures use an end-to-end training method where the models use a single objective to minimise the expected word error rate.

2.5.3 ASR model pipeline

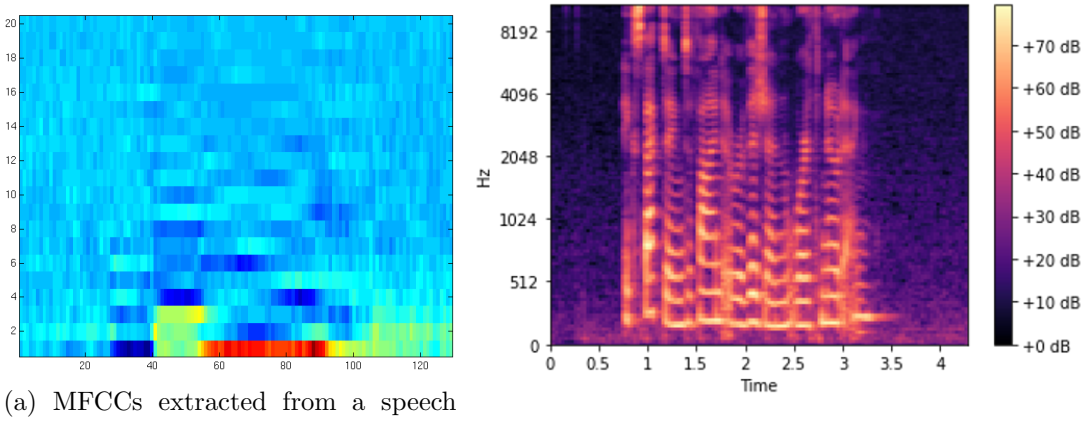
Feature extraction

Most speech-related tasks such as ASR and speech synthesis generally rely on good acoustic representation of the speech for training the model. The audio representations can either be created through the use of some signal processing or by extracting them from deep-learning-based feature extractors. Previously, many studies have focused on signal-processing methods to extract compact representations from speech and audio. The methods typically perform time-frequency analysis of speech segments along short windows of the audio signal. Then, a bank of filters that are designed to mimic the human auditory filter can be applied, e.g., the Mel-scale filter bank, the Bark-scale filter bank, or the Gammatone filter bank (Aertsen and Johannesma, 1980). For example, after pre-emphasising the speech signal, a bank of Gammatone filters can be applied to create Gammatone features (Schluter et al., 2007) for ASR training.

For Mel-scale features, the time-frequency analysis is typically performed in ASR and speech synthesis using a 25 millisecond or 30 millisecond Hann or Hamming window with a 10 millisecond shift. A discrete Fourier transform is then applied to the windowed signal. Subsequently, a Mel-scale filter bank is applied to create Mel-scale features such as Mel-frequency cepstral coefficients (MFCCs) and Mel-spectrogram described below.

Mel-frequency cepstral coefficients (MFCCs): MFCCs are a compact representation of the contribution of the vocal tract, i.e., the envelope of the spectrum, and have been used as input features for speech-related tasks such as automatic speech recognition (Davis and Mermelstein, 1980; Han et al., 2006) and speaker recognition (Sahidullah and Saha, 2012).

As shown in Figure 2.13b, MFCCs often comprise 13 or 20 coefficients instead of the 32–80 bands used in the Mel-spectrogram. They are also a bit more uncorrelated which can be beneficial with linear models like GMMs to simplify the estimation of the covariance parameters of the model. Nevertheless, they have been widely replaced by the Mel-spectrogram in the deep learning era. With lots of data and strong neural classifiers like convolutional neural networks, a Mel-spectrogram often performs better (Hafiz et al., 2023).



(a) MFCCs extracted from a speech sample

(b) Mel-spectrogram of a speech sample.

Figure 2.13: Speech representations used in deep-learning models.

Mel-spectrogram: The Mel-spectrogram is also created by applying a Mel-filterbank to the linear spectrum. As depicted in Figure 2.13b, it is interpretable and can provide information about speech such as pauses, pitch variations, formants, and formant trajectories. A typical dimension of the Mel-spectrum is 80. In addition, in some languages such as Mandarin, the Mel-spectrogram can be augmented with extracted frame-level F0 values (Miao et al., 2016). The Mel spectrogram is the major speech representation used in this thesis.

Deep-learning-based feature extraction: Recently, deep-learning-based feature extractors have also been in wider development and use in speech recognition tasks. These feature extractors take an audio waveform and produce rich feature representations of the audio (Tüske et al., 2018; Baevski et al., 2020). For example, wav2vec2, a feature extractor model (Baevski et al., 2020), learns rich representations of audio useful for several downstream speech classification and recognition tasks (Evain et al., 2021). As shown in Figure 2.14, the model comprises several layers of convolutional neural networks interleaved with layer normalisation layers and Gaussian Error Linear Unit (GELU) activation. The model produces a learned representation for every 25 milliseconds of audio with a 20 millisecond stride.

Encoder and Decoder

The encoder and decoder are essential parts of the neural ASR pipeline. The encoder performs non-linear transformations on the input features into rich contextual embeddings while the decoder's major function is to classify the embeddings into the target or labels. The encoder can take the form of a convolutional, LSTM, or even self-attention-based neural network (Prabhavalkar et al., 2024). A commonly used encoder architecture is the Conformer (Gulati et al., 2020), a convolution-augmented Transformer that sandwiches convolutional neural networks between self-attention layers. Furthermore, depending on the type of decoder, the ASR network can be broadly classified into three popular types namely Connectionist Temporal Classification (CTC) models (Graves et al., 2006), Recurrent Neural Network Transducers (RNN-T) (Graves, 2012), and Attention-based Encoder-Decoder (AED) (Dong et al., 2018). These different approaches have been lately popularised for training ASR systems.

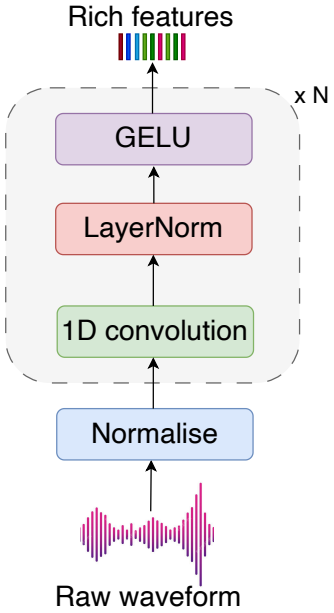


Figure 2.14: Wav2vec2 feature extraction model.

The CTC criterion makes a strong independence assumption that the models output at a time is conditionally independent of the outputs at other time steps, given the local encoder output at that time. This independence assumption makes CTC less predictive and often produces lower performance than other ASR networks.

The RNN-T framework relaxes the independence assumption in the CTC network since the output label at a time is conditionally dependent on the sequence of previous non-blank predictions (in the standard RNN-T case). The network generally has three modules namely an audio encoder, a prediction network, and a joint network that performs sequence-to-sequence modelling. Depending on the encoder architecture, different Transducer models can be derived, e.g., the Conformer-Transducer (Gulati et al., 2020) and Transformer-Transducer (Zhang et al., 2020) use a conformer or self-attention model as an audio encoder.

The attention-based encoder-decoder network uses an attention mechanism to implicitly identify and model the portions of the input acoustics that are relevant to each output label. Therefore, the model can use a standard cross-entropy criterion to learn the mapping, however it still needs a mechanism to determine that it has finished generating all the output labels. Therefore, an end-of-sentence token $< eos >$ is usually required to be predicted as the last token to indicate the end of prediction. Nowadays, it has become common practice to use an auxiliary criterion such as the CTC criterion on the audio encoder network, for example to reduce the flexibility in prediction in attention-based encoder-decoder networks when presented with long utterances (Watanabe et al., 2017) or generally as an auxiliary intermediate loss function (Lee and Watanabe, 2021) on the ASR encoder. In this thesis, we also include a CTC loss obtained through an extra convolution decoder on top of the Conformer encoder for ASR. We describe the Conformer model in detail below.

Conformer

The Transformer architecture which is based on self-attention has been widely used for modelling sequences such as text (Vaswani et al., 2017) and speech (Dong et al., 2018) due to its ability to capture long-range interactions and its high training efficiency. Similarly, convolutional neural networks have been successfully applied to ASR (Li et al., 2019). Convolutional neural networks work effectively by progressively capturing local context through a local receptive field in each layer of the network. Models with self-attention or convolutions each have their strengths and limitations. While Transformers are good at modelling long-range interactions, they are less capable of extracting fine-grained local feature patterns. Likewise, even though convolutional neural networks can exploit local information easily, they still require many more layers or parameters to capture global information due to their limited receptive field.

The Conformer combines the strengths of the two types of architectures. It is made up of a combination of self-attention and convolutional layers, sandwiched between a pair of feed-forward modules, as illustrated in Figure 2.15. The Conformer architecture was proposed for speech recognition by Gulati et al. (2020) and has been used for acoustic modelling of speech in several speech-related tasks (Pengcheng Guo et al., 2021).

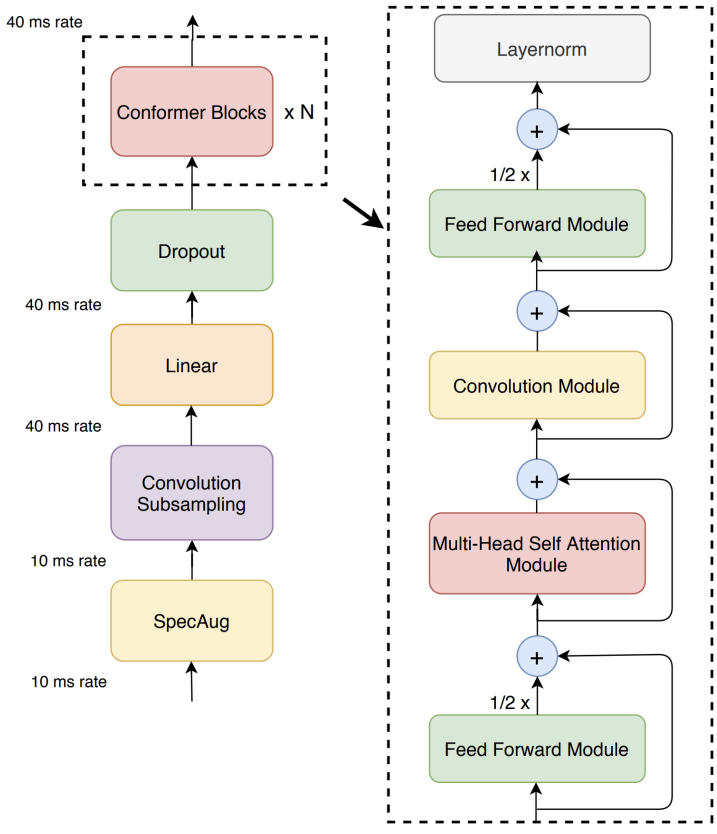


Figure 2.15: Conformer encoder architecture [source: Gulati et al. (2020)].

The Conformer encoder takes speech features as inputs and subsamples them using a convolution subsampling layer. The subsampling layer reduces the number of speech frames by k where k is often 2 or 4 depending on the task. The output of this layer is

then passed through several Conformer blocks. A Conformer block contains four modules stacked together as shown in the expanded section of Figure 2.15. The block contains two feed-forward modules sandwiching the multi-headed self-attention module and the convolution module. The entire block can be represented mathematically as:

$$\begin{aligned} x_{i1} &= x_{i0} + 0.5\,\mathrm{FFN}_1(x_{i0}) \\ x_{i2} &= x_{i1} + \mathrm{MHSA}(x_{i1}) \\ x_{i3} &= x_{i2} + \mathrm{Conv}(x_{i2}) \\ x_{i4} &= x_{i3} + 0.5\,\mathrm{FFN}_2(x_{i3}) \\ f_i &= \mathrm{LayerNorm}(x_{i4}). \end{aligned} \tag{2.8}$$

The input features in the first layer, x_{i0}, are passed through a Macaron-style half-step feed-forward module (FFN) (Lu et al., 2020). The Macaron-style feed-forward module multiplies the activations of the layer by half. The input features x_{i0} are then re-added

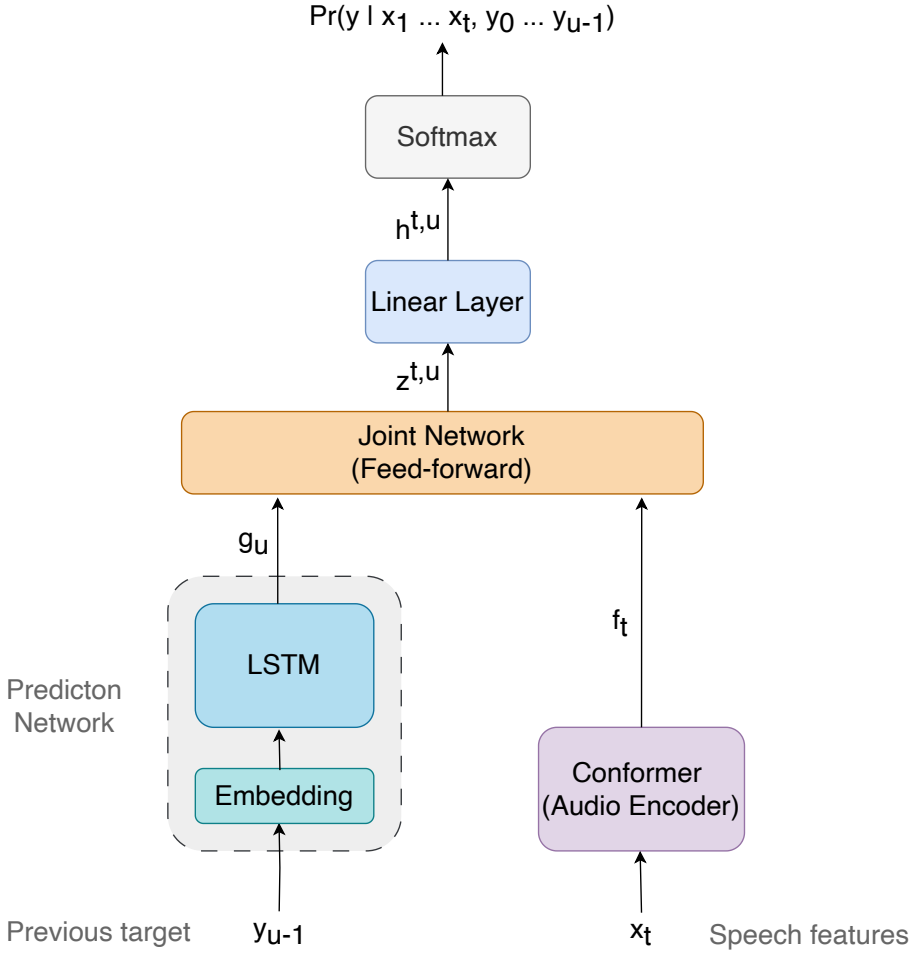


Figure 2.16: Conformer-Transducer architecture showing the Conformer encoder, the LSTM-based prediction network, and the feed-forward joint network.

to the halved activations through a residual connection to form x_{i1}. Afterwards, the feature in the first layer passes through a multi-head self-attention (MHSA) layer which is combined with a residual connection. Then, the feature in the second layer goes through

a convolutional module (Conv) which is combined with a residual connection. Finally, the feature in the third layer enters another Macaron-style feed-forward layer. A residual connection of the features is always added to the activations in all the layers, in addition to layer normalisation (LayerNorm) (Xiong et al., 2020) that is always applied to the output features x_{i4} of the Conformer block for training stability.

The Conformer-Transducer is a type of Transducer (Graves, 2012) that uses a Conformer encoder as the acoustic model, a single-LSTM-layer decoder as the prediction network, and a feed-forward module as a joint network. The ASR model architecture is shown in Figure 2.16. The audio encoder encodes a sequence of speech features into rich representations f_t. Similarly, the prediction network encodes the previously predicted token that is not a blank token into text embeddings. In this regard, the prediction network acts as a language model. The joint network combines the output of the audio encoder and the prediction network. This is followed by a linear layer that projects the joint network output $z^{t,u}$ into the dimension of the vocabulary, denoted as $h^{t,u}$. A softmax function is applied to this output to get a probability distribution over the output units.

Transducers have the advantage that they are trained to operate in streaming mode during inference. Therefore, they are the ASR models of choice on mobile devices and other voice assistants (He et al., 2019).

Self-supervised acoustic models

Self-supervised acoustic models leverage speech data to train a speech encoder to learn robust rich representations (Baevski et al., 2020). After training, the model can then be adapted for speech-related tasks using a few supervised examples (Yang et al., 2021). Self-supervised learning is typically applied when a large amount of unlabelled data is available and we would like to exploit this data for representation learning. The representations can be learned directly from the speech signal as in wav2vec2 (Baevski et al., 2020) or from features extracted from speech, e.g., Mel-spectrograms, as in HuBERT (Hsu et al., 2021).

Learning feature-rich representations from speech alone is an active area of research. Representations have been learned through predicting future steps of the output and by contrasting positive and negative labels (CPC) (van den Oord et al., 2018), using a vector-quantised codebook that constrains the output representation to be similar to embeddings in the codebook (wav2vec2) (Baevski et al., 2020), or using a masked language modelling approach (HuBERT) (Hsu et al., 2021) to predict some masked input features from context. Rich representations have also been learned by constraining the encoder output to be able to perform multiple speech-related classification and regression tasks, e.g., tasks involving the reconstruction of filter-bank, spectral, and prosody features (Ravanelli et al., 2020). More concretely, to learn representations in wav2vec2, a convolutional neural network-based feature encoder was used. During training, the representations from the feature encoder are quantised and contrasted with the context representations predicted by the masked Transformer as shown in Figure 2.17. The feature encoder was described in Section 2.5.3. After training, a randomly initialised decoder/classifier is normally added on top of the trained encoder for downstream applications such as ASR.

Language model

A language model is a model that learns the probability of occurrence of a word given surrounding words. It is an important part of an ASR system because it helps the ASR

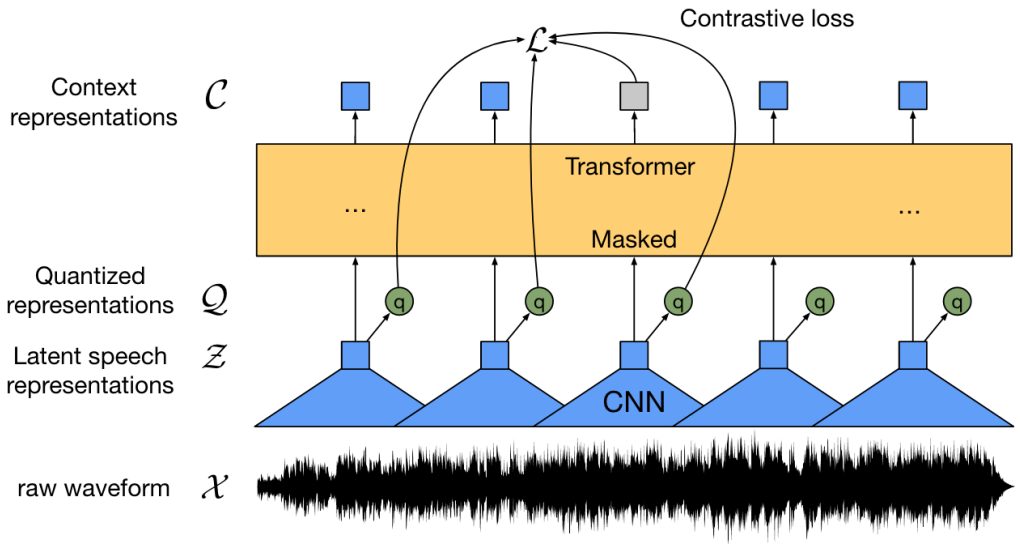


Figure 2.17: Illustration of the wav2vec framework for self-supervised learning of speech features [source: Baevski et al. (2020)].

model to fix the errors in its prediction, identify the most probable word among several similarly pronounced words, and also helps it to correctly predict rare words.

Today's sequence-to-sequence ASR architectures directly define the posterior models that do not distinguish between the language model and the acoustic model and thereby include an internal language model representation trained as part of the overall ASR model like the decoder in AED models or the prediction network of RNN-T. However, for improved performance in many applications, an external language model is still combined with the ASR model to improve its performance.

A language model can be integrated into the prediction of an ASR model in several ways. One simple way to combine the external language model scores with the ASR model scores is through a *shallow fusion* of their independent predicted probabilities. In this approach, a beam search algorithm is used to determine the most probable sequence of tokens (Toshniwal et al., 2018) with the integration of the probability scores from the language model. Another approach is to combine the ASR model features and language model features using a joint network using either *deep fusion* or *cold fusion* (Prabhavalkar et al., 2024).

Popular language models used with ASRs include n-grams, Recurrent neural networks like LSTMs, and Transformer-based language models such as BERT. N-gram language models are very fast to integrate into an ASR workflow since only a look-up table is required to compute the probabilities, however they are also the least accurate since they give low probability to unknown words, leading to poor generalisation. N-gram language models between 3-gram to 6-gram are common for word-based models and higher for subword or character-based models. ASR models today make use of deep-learning-based language models such as LSTMs and Transformer-based language models which can assign probability mass to unknown words through semantic similarity of words in their embedding space.

The language model can be trained on a massive amount of text data from a domain,

instead of being constrained to be trained on only the paired text accompanying the audio. Pretrained neural language models produce lower perplexity, and there is also a correlation between this perplexity and the utility of the language model in ASR. Furthermore, pretrained language models can additionally be used for refining the predictions of an ASR model. One such approach is called *N-best rescoring*. It involves a two-pass search algorithm. In the first pass, the ASR model decodes and outputs N-best ASR hypotheses of the input utterances, and then rescores the hypotheses in the second pass by combining the ASR and language model scores. The rescoring itself can be done with many different scoring approaches including shallow fusion. Similarly, in this current era of large language models (LLMs), LLMs have also been applied as a second step in correcting ASR predictions. In this approach, the final prediction from the ASR is sent to the LLM to correct transcription errors or add punctuation to ASR transcriptions (Adedeji et al., 2024). This first use case is very useful in niche domains such as the medical domain (Adedeji et al., 2024) where ASR models tend to wrongly transcribe medical named entities. For the use case of adding punctuation to the ASR transcriptions, this may not be a perfect approach since they do not make use of acoustic audio cues during refinement.

Decoding ASR outputs

Decoding algorithms estimate the most likely sequence of output labels among all possible sequences. The output representation from the decoder is used to compute probability scores which are used for the decoding task. Decoding can be done using a greedy search or beam search algorithm (Williams et al., 2018). The Greedy search algorithm selects the label with the highest probability at each decoding step, making it a candidate for fast, parallel computation. It ignores the dependency of the output labels, which causes relatively poor performance than beam search decoding. It is mainly used in CTC networks since the network ignores the dependency of the output labels. In beam search decoding, the model keeps a predefined number of possible paths, and at the end of decoding it selects the path with the maximum joint probability. Predicted labels can be characters, sub-words, or words.

2.5.4 ASR evaluation metrics

ASR systems are evaluated based on their transcription accuracy and sometimes based on the inference speed, especially when considering them for streaming applications. In terms of transcription accuracy, one has to evaluate how often words are wrongly transcribed. Therefore, for evaluation purposes, it is customary to have a test dataset that is disjoint from the dataset used for training the system or for tuning its hyperparameters. The test set is meant to mimic the testing conditions where the ASR model will be applied, e.g., the speakers in the test dataset are disjoint from those in the training dataset.

First, computing the errors requires that the test dataset audio files are transcribed, and then compared to the reference text using some metric. A common metric for determining the number of errors in the transcript is called the edit distance or the Levenshtein distance (Navarro, 2001). The metric counts the minimum number of operations required to transform one string into the other, considering edit operations such as insertion of a token, deletion of a token, and substitutions of a token in the text where a token can be a word or character in our case. Taking a word as an example, the edit operations can be described as follows:

- Insertion (ins) of a word: This implies that the prediction includes an extra word that cannot be found at that position in the reference text, and needs to be dropped,
- Deletion (del) of a word: This means that a word is missing at the current position in the predicted text to match the reference text,
- Substitution (sub) of a word: This occurs when a word in the predicted text has to be replaced by another word to get the reference text.

A dynamic programming algorithm is usually used to compute the edit distance. Given a sequence of predicted tokens $x = x_1, x_2, ..., x_m$ and reference tokens $y = y_1, y_2, ..., y_n$, the minimum edit distance is defined by a recursive algorithm, known as the WagnerFischer algorithm (Navarro, 2001), as follows:

$$
\begin{aligned}
d_{i0} &= \sum_{k=1}^{i} w_{\text{del}}(x_k), && \text{for } 1 \le i \le m \\
d_{0j} &= \sum_{k=1}^{j} w_{\text{ins}}(y_k), && \text{for } 1 \le j \le n \\
d_{ij} &= \begin{cases} d_{i-1,j-1} & \text{for } x_i = y_j \\ \min \begin{cases} d_{i-1,j} + w_{\text{del}}(x_i) \\ d_{i,j-1} + w_{\text{ins}}(y_j) \\ d_{i-1,j-1} + w_{\text{sub}}(x_i, y_j) \end{cases} & \text{for } x_i \neq y_j \end{cases} && \text{for } 1 \le i \le m, 1 \le j \le n.
\end{aligned}
\tag{2.9}
$$

The weights of the operations w_{ins}, w_{del}, and w_{sub} are usually set to 1, although different weights can be given to each operation in specific cases. The minimum edit distance can then be read off from the algorithm as d_{mn}.

Since the minimum edit distance provides us with the total number of errors, this accuracy of the transcription can be estimated on the test dataset at a sub-sentence level using the word error rates and character error rates.

Word error rate

The word error rate (WER) estimates the transcription error of the ASR model at the word level (Hunt, 1988). Additionally, it shows the ability of the ASR model to predict correct words in the context of other surrounding words. Therefore, it provides an estimate of the transcription accuracy of the ASR model.

The steps to compute the WER is described as follows: for each audio-text pair, we transcribe the audio using the trained ASR model and compare this to the reference text to compute the number of errors e_s in the predicted text. The WER is then calculated as the ratio of the sum of word errors in the predicted text to the total number of words in the reference text:

$$
\text{WER} = \frac{\sum_{s=1}^{n} e_s}{\sum_{s=1}^{n} r_s} \tag{2.10}
$$

where e_s represents the sum of insertion, deletion, and substitution errors computed using the edit distance algorithm, and r_s is the number of words in the s-th sentence of the test dataset.

The WER is often expressed as a percentage. It can exceed 100% in the case when the number of words in the predicted text differs significantly from the number of words in the ground truth due to insertion errors (Bisani and Ney, 2004). Importantly, a WER close to zero is usually desired for real-world ASR applications (Szymański et al., 2020).

Character error rate

The character error rate (CER) measures the ASR transcription errors at the character level. The CER is useful in evaluating mispronunciations in the predicted text since the WER penalises a misspelled word even if only some characters are different, or the spelling of the word differs in some way from the word in the reference text. Substituting errors at the word level for errors at the character level gives the CER:

$$\text{CER} = \frac{\sum_{s=1}^{n} e_{cs}}{\sum_{s=1}^{n} r_{cs}} \tag{2.11}$$

where e_{cs} represents the sum of insertion, deletion, and substitution errors in characters computed using the edit distance algorithm, and r_{cs} is the number of characters (including spaces) in the s-th sentence of the test dataset. The CER is often expressed as a percentage. It is always lower than the WER and closer to zero for good ASR systems.

Finally, since each metric can provide information regarding the different types of errors made by the ASR system, it is desirable to use both metrics in their evaluation. Therefore, the two metrics are used to determine the performance of the ASR systems in this thesis.

Real-time factor

The real-time factor (RTF) helps to determine the speed of inference of the ASR model. It is defined as:

$$\text{RTF} = \frac{\text{Time taken to process an audio in seconds}}{\text{Duration of the audio in seconds}}. \tag{2.12}$$

If RTF ≤ 1, then the input audio is said to have been processed in less than real-time or in real-time. In addition, the RTF is highly hardware-dependent and the value may vary for different hardware for the same audio (Malik et al., 2021), nevertheless it is still useful for comparing different models given the same hardware.

2.5.5 ASR data augmentation

Data augmentation is the process of changing the properties of the training dataset to improve the robustness of the model to small perturbations in the input data or increase the variance of the data to capture more diversity. In addition, data augmentation is also beneficial to deep learning models since they are known to easily overfit small training datasets. Similarly, the testing conditions of most ASR systems usually differ significantly from the conditions in the training data. Therefore, various techniques have been developed in the literature to increase the training data diversity and match the testing conditions.

The data augmentation strategies in ASR can be divided into different categories. Some data augmentations such as noise addition (Ko et al., 2017a) and increasing the reverberation of the audio (Kim et al., 2017) aim to vary the environmental conditions in

the data. Methods like pitch shifting, time shifting, and vocal tract length perturbation help to vary the prosodic attributes of the speech input (Ko et al., 2015). Others like SpecAugment (Park et al., 2019) delay model overfitting by zeroing out portions of the input features on the time and frequency axis of the features. These methods cannot however increase the phonetic diversity or speaker diversity in the dataset. Adding synthetic data to the real data has been a recent line of research explored to increase the diversity of the real data beyond the limits of the training data available.

2.5.6 ASR data augmentation using synthetic data

ASR training data augmentation using synthetic data is mostly done by combining real speech data and synthetic speech data during model training (Hu et al., 2022). The synthetic data can either be TTS generated or voice converted. Synthetic data are samples generated from a finite distribution of real data that was learned during TTS/VC training. Therefore, there is a distributional mismatch between real and synthetic data (Hu et al., 2022; Sharma et al., 2020). The mismatch is usually accounted for in several ways during the ASR training phase. Various approaches in the literature are discussed below.

Modify the synthetic data distribution

The first category of data augmentation modifies the synthetic data during ASR training. Examples include noise addition (Minixhofer et al., 2023), SpecAugment (Park et al., 2019), and mixup augmentation (Sharma et al., 2020). Adding noise to the synthetic speech simulates similar environmental conditions to the real speech during training. Mixup augmentation mixes the real and synthetic speech using a mixing ratio to create a more robust feature for the input data. In the case of mixup augmentation, the model is required to be able to label both the real and synthetic mixture at its output. Similarly, a rejection sampling strategy was used by Hu et al. (2022) to reject samples outside the distribution of typical real speech. Modifying the data directly increases the model's robustness to noise and environmental conditions, however it is data-centric.

Modify the ASR model layers

Another technique is to modify the ASR model's layers to account for the differences between the real and the synthetic datasets. For example, a separate batch normalisation statistic can be learned for the real and synthetic dataset (Hu et al., 2022). In this case, the batch normalisation statistics of the real dataset are used during inference while those of the synthetic dataset are discarded. The affine parameters of the statistics are still shared between the real and synthetic speech, which helps to reduce model overfitting. Changing the model layers increases robustness, however it still requires some knowledge of the model architecture and several modifications to determine which layers are most important for the task.

Modify the ASR model training objective

Modifying the training objective can be used to account for both the real and synthetic domains. For instance, a consistency loss was applied by Wang et al. (2020a) to ensure similar grapheme predictions for real and synthetic speech when the synthetic data was generated using the real data texts. Also, speech chain (Tjandra et al., 2020; Wang et al.,

2020a), which involves performing both ASR training and TTS training in a closed loop, using one part of the process to improve the other part, is another method that ensures consistency between the real and generated synthetic speech. It allows the ASR model and TTS model to learn from each other in a feedback loop. This method is quite experimental and requires the weight of the loss functions to be determined experimentally to get the best performance.

In this thesis, we mostly focus on modifying the synthetic data distribution during TTS/VC generation to perform ASR data augmentation. Hence we focus on the impact of the various attributes in the generated data, however it is possible to combine this with other discussed approaches.

2.6 Datasets

There have been several efforts in the past decade to create open-source datasets for training ASR models and speech synthesis models. Table 2.2 shows some popular open-source datasets with text labels that are currently available in the open domain.[3] Most of the corpora in Table 2.2 are hosted by the Linguistic Data Consortium (Liberman and Cieri, 1998), Common Voice's repository (Ardila et al., 2020), Hugging Face's models and datasets repository (Jain, 2022), and the Open Speech and Language Resources repository (OpenSLR, 2024).

Table 2.2: Popular datasets used for training ASR and TTS systems.

Dataset	Size (h)	Language	ASR/TTS
AISHELL-1 (Bu et al., 2017)	170	Mandarin	ASR
AMI Meeting corpus (Kraaij et al., 2005)	100	English	ASR
Common Voice (Ardila et al., 2020)	15,000	Multilingual	ASR
CSS10 (Park and Mulc, 2019)	140	Multilingual	TTS
GigaSpeech (Chen et al., 2021)	10,000	English	ASR
Libri-Light (Jacob Kahn et al., 2020)	60,000	English	ASR/TTS
LibriSpeech (Panayotov et al., 2015)	960	English	ASR
LibriTTS (Zen et al., 2019)	585	English	TTS
LJ Speech (Ito and Johnson, 2017)	24	English	TTS
M-AILABS (Solak, 2018)	1,000	Multilingual	ASR/TTS
MLS (Pratap et al., 2020)	50,000	Multilingual	ASR/TTS
MyST Dataset (Pradhan et al., 2023)	393	English	ASR
TED-LIUM (Rousseau et al., 2012)	452	English	ASR
The Spoken Wikipedia (Köhn et al., 2016)	1,005	Multilingual	ASR
VCTK (Yamagishi et al., 2019)	40	English	TTS

In general, ASR datasets can either be *read speech* (Ardila et al., 2020; Bu et al., 2017) or *spontaneous speech* (Maekawa et al., 2000). In read-speech datasets, the human recorders read out texts that are recorded during the recording activity. It may be necessary to split utterances and text into smaller segments using time-alignment methods like in the Spoken Wikipedia corpus (Köhn et al., 2016). In spontaneous speech datasets, the speaker does not use a well-defined script and instead talks about a given theme or

[3]For an exhaustive list of open-source speech datasets, we refer readers to Coqui AI's open speech corpora collection at https://github.com/coqui-ai/open-speech-corpora

topic. Spontaneous speech datasets can also be derived from podcasts, TV shows, and talks such as TED Talks (Rousseau et al., 2012). The collected spontaneous speech is then annotated after collection. In other domains, the dataset can also contain *continuous speech* (Kraaij et al., 2005) of several speakers in the same recording. Also, some datasets provide labelled and unlabelled data for use in unsupervised and semi-supervised training (Jacob Kahn et al., 2020; Pradhan et al., 2023; Chen et al., 2021).

For ASR, one notable project that birthed many ASR datasets is the LibriVox initiative (Kearns, 2014). It is an initiative "to make all books in the public domain available, narrated by real people and distributed for free, in audio format on the internet" (Kearns, 2014). Volunteers record chapters of books in many languages that are available in the public domain. Some datasets created from LibirVox include LibriSpeech, a curated corpus of English audiobooks. Some larger curated datasets such as M-AILABS (Solak, 2018), MLS (Pratap et al., 2020), and Libri-Light (Jacob Kahn et al., 2020) add more languages to the mix. Another more recent initiative is the Mozilla initiative that aims to record ASR datasets from many languages including low-resourced languages. The Common Voice (CV) corpus (Ardila et al., 2020), a derivative of that initiative, is a crowdsourced read speech corpus that covers over 100 languages and several hundreds of speakers.

For speech synthesis systems such as TTS systems and VC systems, corpora are typically smaller than for ASR. Several corpora have been created in controlled recording and room conditions due to the requirements of the dataset. The most popular of them is VCTK (Yamagishi et al., 2019), an English language corpus with over 100 speakers. To derive larger datasets, it is typical to extract clean samples from large ASR datasets or audio-book repositories. LibriTTS (Zen et al., 2019) is one such dataset derived from LibriSpeech by removing noisy samples and thresholding the signal-to-noise ratio (SNR) of the dataset. LJ Speech (Ito and Johnson, 2017) is also a popular TTS dataset derived from audio-book recordings of a single female English speaker, used for benchmarking TTS models. CSS10 (Park and Mulc, 2019) is a dataset of single speakers in 10 languages derived from LibriVox audiobooks to cover speech synthesis for more languages.

The datasets used for training ASR and speech synthesis systems are quite similar in some regard, and different in other aspects. Ordinarily, an ASR dataset is a multi-speaker dataset of speech and text pairs with some attributes that make them useful for the ASR task. Such attributes include noise, reverberation, varying speaking styles, varying recording conditions, diverse vocabulary, etc. A TTS dataset usually contains pairs of clean speech and text. It can either be *single-speaker* or *multi-speaker*. Both datasets contain speech and text pairs, however the dataset quality needed for training the models varies to some degree. In this thesis, we also aim to use existing ASR corpora for TTS/VC training for data augmentation purposes, therefore we select the Common Voice dataset for ASR and TTS training due to its large diversity of speakers and languages. We describe a method to curate the TTS dataset from this dataset in Chapter 3.

2.7 Summary

This chapter introduced speech in the context of human-machine interaction, and also the attributes of speech including prosodic attributes. Then, we reviewed speech synthesis systems and the various improvements that have been made to them through new deep learning architectures such as normalising flow-based generative models. We also discussed the changes to speech synthesis systems that help to increase their controllability as well

as performance. In addition, we discussed several ASR systems including Conformer-Transducer, a popular ASR architecture. Finally, we provided some studies that have been done to improve the performance of ASR systems by augmenting their training data with synthetic speech data.

3

Multi-speaker TTS dataset curation from an ASR dataset

Training text-to-speech (TTS) models requires paired speech and text data of high quality. For multi-speaker TTS training, there is also the requirement of having many speakers, with their individual speaker variability (in terms of accent, speaking style, speaking rate, etc.). These variabilities are necessary for the TTS system to learn diverse speaker characteristics that will enable it to generate convincing synthetic voices for speakers outside its training set. In the past decade, several efforts have been made to curate multi-speaker TTS datasets. Such dataset efforts fall into two categories, namely studio-quality TTS datasets such as VCTK (Yamagishi et al., 2019), and TTS datasets curated from audiobooks such as LibriTTS (Zen et al., 2019).

In the case of a studio-quality TTS dataset, the dataset has the advantage that all the recordings can be made in a controlled environment. The recording room can be designed to eliminate sound echoes, noise, and other external interference. Recording devices such as microphones and amplifiers can also be calibrated to the right distance and volume. The annotators can be selected to derive different characteristics in the dataset. Even the properties of the room can be modified to produce desired room characteristics. In addition, quality checks can be done to ensure that recordings meet the desired standards, for example, that words are pronounced accurately with the right stress and accent. This method of dataset curation is feasible for small-scale data curation, however this is not the case for large-scale data collection where sourcing for many speakers to physically record speech can be a cumbersome effort since speakers may not live in the same location as the recording site. In addition, the recording of high-quality data is expensive and also requires expert knowledge of studio recording and acoustics.

In recent efforts to curate TTS datasets, public domain audiobook stores like Librivox (Kearns, 2014) have provided free recordings of books made by volunteers who read long passages from books in any language of their choice. The books are sourced from Project Gutenberg (Rowberry, 2023), a web-based collection of books that have fallen out of copyright in the United States. The recordings then go through a processing stage to trim silence, remove noise, and eliminate low-quality recordings. The remaining recordings are then split into smaller chunks and aligned with the text from the corresponding book. This approach is less cumbersome and has been used to curate multi-speaker TTS datasets for the English language and several other highly-resourced languages (Solak, 2018). However, freely available public-domain books like these are non-existent for some languages and

domains, and volunteers are limited for some languages with only a few active volunteers participating. In addition, even for highly-resourced languages like English, the speaker diversity may be limited in terms of regional accents and other speaker characteristics. For example, LibriTTS has a concentration of speakers of US English accents, which are not representative of the entire spectrum of speakers and accents.

Recent automatic speech recognition (ASR) datasets are large and contain recordings from a much larger number of speakers. These recordings are usually collected through volunteers who record a short text in a comfortable environment using recording devices such as mobile phones and computers. Through this setup, a large number of speakers can participate, thereby making it possible to capture an inclusive and diverse set of speakers. Recordings can also be easily collected for several domains and languages provided there exists a text corpus to be spoken. The large speaker variability in terms of accent, speaking style, speaking rate, etc., and the large phonetic diversity inherently exhibited by these datasets makes them good candidates for training TTS models. However, they also exhibit lower-quality recording conditions (reverberation, background noise, variable recording device, etc.) which are desirable for ASR but can hinder their adoption as a TTS dataset. This necessitates a data curation effort to transform available crowdsourced ASR datasets into high-quality TTS datasets.

To deal with the noise and recording conditions, at the TTS modelling stage, the recordings can be enhanced by integrating a denoising module into the TTS architecture to directly take care of the noise in the model or factor it out during inference (Zhang et al., 2021a; Hsu et al., 2019). Another similar approach by Chang et al. (2022) adds an encoder to the TTS model to encode all the environmental characteristics of the speech. Therefore, different environmental characteristics can be added to the speech during generation for novel speech generation in different conditions. These approaches require modifications of the standard TTS architecture and do not deal with the dataset which may be required for other speech synthesis applications.

To make the ASR dataset useful for TTS training, it has to be processed by filtering out low-quality utterances or by performing some form of speech enhancement. For filtering out the low-quality utterances, speech quality measures can be used to measure the quality of the recordings and then, a threshold is set to accept good-quality recordings. For example, the signal-to-noise ratio (SNR) of audio can provide a measure of the level of noise in the audio. Although this filtering step is not perfect and allows a few noisy samples to remain uncaught, the resulting dataset can still be good enough for TTS. Even though metrics like PESQ (Rix et al., 2001) and Mel-cepstral distortion (MCD) (Kubichek, 1993) are useful for speech quality evaluation, they require a reference sample alongside the noisy speech which is not available in this case. Likewise, quality cannot be estimated via subjective listening tests, as this becomes intractable for a large number of audio samples. Besides the filtering out of noisy samples, transcripts from an ASR system can be used to compute the character error rate (CER) or word error rate (WER) and filter out samples with mispronunciations or bad transcription.

There has been a recent surge in non-intrusive model-based methods to automatically measure the quality of speech utterances (Lo et al., 2019; Patton et al., 2016). Although the earlier models do not always generalise well outside of the training corpus, the introduction of self-supervised models to the task has significantly improved the performance of quality prediction models resulting in higher correlation with human evaluators' scores (Cooper et al., 2022) than previous architectures. As an example, a self-supervised-based MOS quality predictor, WV-MOS (Andreev et al., 2023), developed by finetuning a wav2vec2

feature extractor stacked with a Multi-Layer Perceptron as a MOS regression head, generalises better to recordings and speakers unseen during training. It can be used to evaluate the performance of speech-processing systems for a variety of tasks.

In this chapter, we focus on automatically selecting high-quality training samples from a crowdsourced dataset, using Common Voice English (Ardila et al., 2020) as an example. Due to the size of the original dataset (1.4 million utterances), we leverage WV-MOS for evaluating the quality of recordings, and set up a pipeline to curate a high-quality TTS dataset.

The structure of this chapter is the following. We first describe the Common Voice dataset, its properties, and limitations for use as a TTS dataset in Section 3.1. Then, we describe our method to filter the dataset in Section 3.2. We evaluate the quality of the filtered dataset by training TTS models in Section 3.3 and provide objective and subjective evaluation results in Section 3.4. We finally conclude the chapter in Section 3.6. The contents of this chapter have been published at SLT 2023 (Ogun et al., 2023a).

3.1 Common Voice

Common Voice (Ardila et al., 2020) is the result of a Mozilla project to create a massively multilingual, multi-speaker collection of transcribed speech for speech technology research and development. The project employs crowd-sourcing for both data collection and data validation for scalability and sustainability. The corpus is designed to be applied to ASR, language identification, and among other speech-related tasks, keyword spotting. It is made available under a public domain license, the Creative Commons Zero license which allows developers to use the corpus without restrictions or costs.

Common Voice is one of the truly large crowdsourced audio corpora in the public domain with a large representation of contributors that cover under-resourced languages. For instance, through the efforts of contributors, the platform has been able to crowd-source over two thousand hours of data from 1,131 contributors for the Kinyarwanda language, a language spoken by the Bantu ethnic groups in Africa and a previously under-resourced language (Siminyu, 2022). As of July 2024, one hundred and twenty-nine languages including over 84,000 contributors are represented in the corpus, over 31,000 hours of audio have been recorded by contributors while 19,000 hours have been validated, and data collection is ongoing for several other languages. These statistics validate the power of dataset crowdsourcing.

Utterances are recorded through the Common Voice website at https://commonvoice.mozilla.org/en or a mobile application. The recordings are then verified by other contributors using a simple voting system. Each recording is up-voted or down-voted by contributors according to a list of criteria.[4] Recordings with more than two up-votes are marked as valid while recordings with at least two down-votes are marked as invalid since a maximum of three contributors rate each recording. The validated utterances are then split into train, development, and test sets, with non-overlapping speakers and sentences. These criteria for validation are not very restrictive, e.g., various kinds of background noises are allowed and down-votes are not considered in the validation decision in as much as a recording has received two up-votes.

The corpus is also regularly updated and versioned to include the latest releases. Although version 17.0 is the latest version of the corpus at the time of this writing, we used

[4] https://commonvoice.mozilla.org/en/criteria

version 7.0 in our experiments. All recordings from contributors are saved and released as single-channel, 16-bit mp3 files with a 32 kHz or 48 kHz sampling rate.

3.1.1 Analysing Common Voice corpus quality for TTS

The Common Voice Corpus is a corpus that is suited for ASR and has been widely used for training ASR systems (Rivière et al., 2020). On the one hand, the availability of the data has helped bridge the gap in performance between commercial systems and open-source ASR systems. On the other hand, to adapt it for TTS, we have to consider its undesirable properties:

- Noise: Several recordings exhibit electromagnetic noise or acoustic noise such as mouse clicks, low-frequency noise, background speakers, background music, etc. As described by Valentini-Botinhao et al. (2016), the quality of generated utterances from text-to-speech systems built from noisy recordings diminishes. Additionally, since the utterances are stored in mp3 format, quantisation noise can also sometimes be heard. Figure 3.1 shows the spectrogram of some samples in the Common Voice corpus that have been affected by noise. Lines and blurred spectrograms indicate the presence of noise affecting the main speech signal. The spectrogram shown mostly indicates electromagnetic noise with harmonics and white noise. We also show the WV-MOS score for each utterance below its spectrogram. The WV-MOS model outputs MOS scores in the range of 1.0 to 5.0. Here, it can be observed that the WV-MOS scores are typically low, which indicates low-quality utterances.

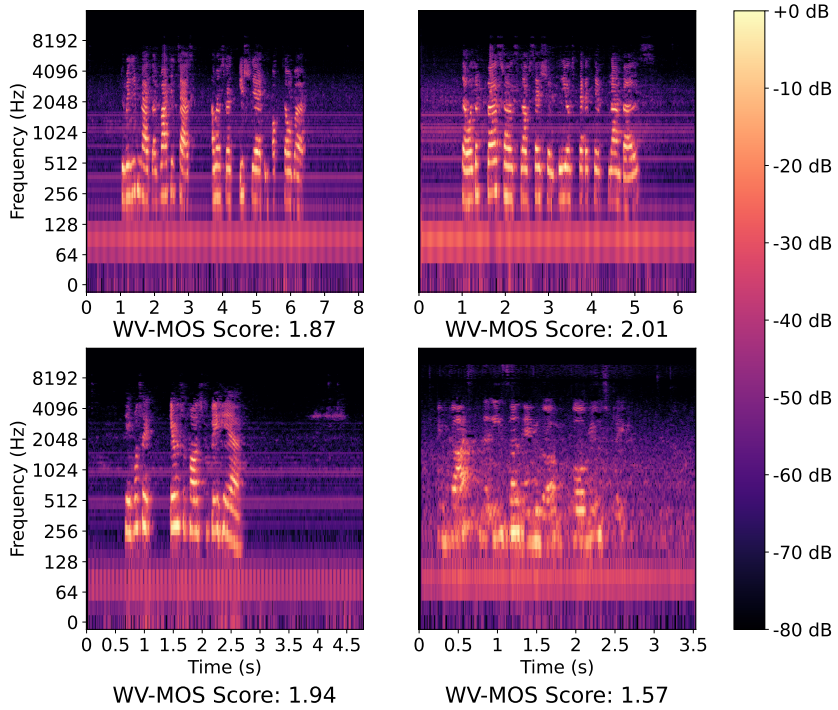


Figure 3.1: Spectrogram of some Common Voice samples showing noise distortions. The frequency axis is on a logarithmic scale. The audio files used for generating the plots are: `common_voice_en_{24018979, 24086062, 25375404, 27145348}.mp3`

- Low bandwidth: Due to recording choices or high compression, some audio files are low-pass filtered, with a cutoff frequency that varies from one file to another. For example, as shown in Figure 3.2, some files were processed with a cut-off frequency around 8 kHz or less. Other files show some missing bands which may be due to high compression of the recording.

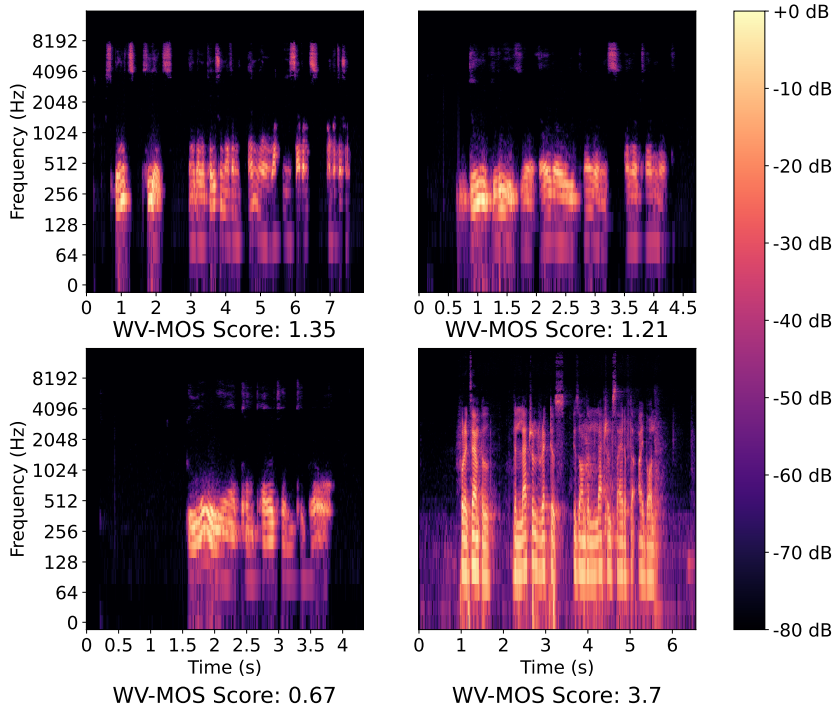


Figure 3.2: Spectrogram of some Common Voice samples showing low-pass filtering and high compression. The frequency axis is on a logarithmic scale. The audio files used for generating the plots are: `common_voice_en_{25829397, 25892038, 25829067, 20190776}.mp3`

- Mispronunciation: We observe mispronunciations of "unfamiliar" words, variations in the pronunciation of certain other words, and some utterances in other languages (e.g., German utterances in the English corpus).
- Unavailable speaker metadata: Age, gender, and accent information are not available for all speakers. For some TTS applications, this information is required to accurately select subsets of the dataset. For instance, for multi-accent TTS models, the accent metadata needs to be known to enable the system to learn individual accents during training.
- Other factors include variable recording characteristics (microphone, room, recording device), speaking rate, and volume. These factors affect the naturalness of synthetic voices generated from the TTS model trained on data containing several of these conditions.

As discussed earlier, the listed characteristics are generally not a limitation for training ASR systems, and they can even be desirable for robustness. For TTS, the inverse

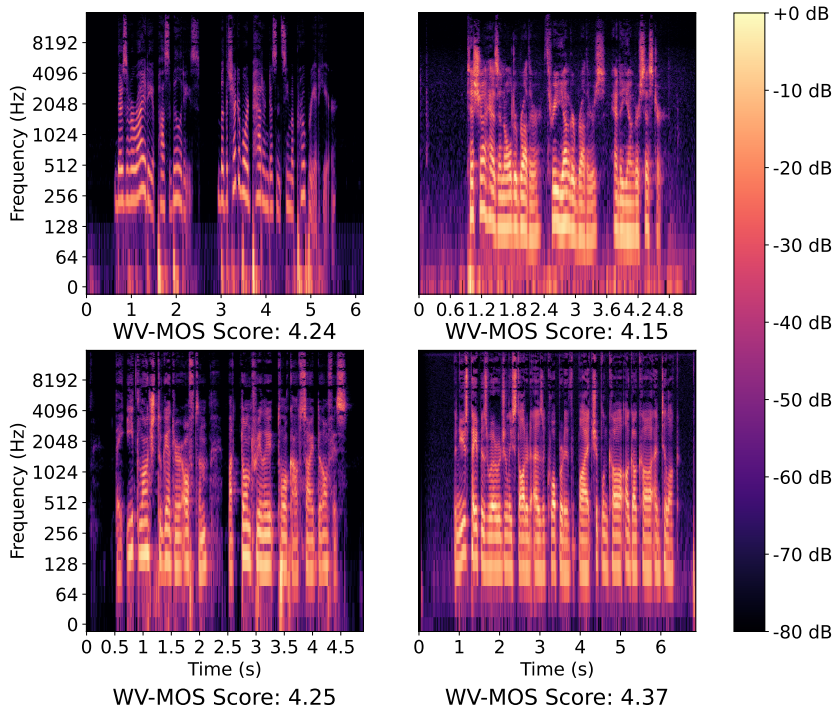


Figure 3.3: Spectrogram of some Common Voice samples with predicted WV-MOS scores greater than 4.0. The frequency axis is on a logarithmic scale. The audio files used for generating the plots are: `common_voice_en_{17653207, 19132573, 174179, 22372092}.mp3`

is usually the case. The dataset characteristics which have been observed to improve the quality of generated utterances in multi-speaker TTS include the following: high signal-to-noise ratio between the speech content and environmental noise, uncompressed signal, correct annotation of text to correspond to recording, normalised volume, and homogeneous speaking rate.

Exploring the dataset, we found several recordings that are of high quality, the spectrograms of some of these recordings are shown in Figure 3.3. Here, the MOS estimator outputs high WV-MOS scores for these recordings. Visually, the recordings do not show the presence of noise, low-pass filtering, etc. The goal is then to find an automatic method to select the best subset from the ASR dataset that still maintains the desired TTS properties, e.g., a large number of speakers.

3.1.2 Dataset preparation

In this work, we used the English subset of Common Voice (version 7.0).[5] Table 3.1 shows the statistics of the dataset. For TTS experiments, we excluded the predefined development and test sets from the validated set, leaving a 1,930-hour subset denoted as *validated-x*. The validated-x subset contains speakers in the train set including some speakers and utterances that have been originally excluded from the train, dev, and test splits. Therefore, we considered all the utterances in *validated-x* as candidate TTS

[5]The dataset can be freely downloaded from https://commonvoice.mozilla.org/en/datasets

training samples. The samples were preprocessed by resampling from 32 or 48 kHz to 16 kHz, and removing beginning and end silences using pydub[6] with a threshold of −50 decibels relative to full scale (dBFS).

Table 3.1: Dataset statistics for version 7 of the Common Voice English dataset. The validated-x set excludes the dev and test sets from the validated set.

Data split	Number of utterances	Duration (h)	Total speakers
train	759,975	1,209.31	27,135
dev	16,284	27.67	5,767
test	16,284	26.90	10,247
validated	1,425,784	1,985.20	65,709
validated-x	1,393,216	1,930.63	49,695

3.2 Methodology

3.2.1 MOS estimation

The MOS estimation of the utterances provides a method to rate utterances according to their overall quality. This is automatically performed using WV-MOS (Andreev et al., 2023), a pre-trained MOS estimation model[7] which is shown in Figure 3.4. The model combines a pre-trained wav2vec2.0 feature extractor and a two-layer multi-layer perceptron (MLP) head. The MLP head comprises two fully connected layers sandwiched with a ReLU activation and dropout.

The WV-MOS model processes raw audio files at a sampling rate of 16 kHz and outputs a prediction for each frame of the wav2vec2 feature extractor. The predictions are then averaged to get the final WV-MOS score corresponding to the audio. The model was trained using a mean squared error loss on the subjective evaluation scores released from the Voice Conversion Challenge 2018 (Lorenzo-Trueba et al., 2018). In an experimental evaluation, WV-MOS predictions correlated well with human quality judgement regarding noise and low bandwidth (Andreev et al., 2023).

As shown in Figure 3.5, we automatically estimated the MOS scores for each utterance in the validated-x dataset, then averaged the scores of utterances belonging to each speaker to have an estimate of each speaker's recording conditions. Filtering can then be done at the utterance level or using the average WV-MOS for each speaker. Filtering at the utterance level may keep more speakers, but will reduce the number of samples available per speaker. Our preliminary analysis showed a low standard deviation of intra-speaker WV-MOS scores, indicating that recording and environmental conditions are constant among utterances belonging to a speaker. Therefore, the selection was done at the speaker level.

To determine the right threshold to keep in the final TTS dataset, we had to determine a method to evaluate the quality of the selected utterances. Therefore, we trained different TTS models on the dataset filtered at different estimated MOS thresholds. Additionally, to evaluate the impact of denoising on the quality of utterances generated by the trained TTS models, we further denoised the filtered utterances using the pre-trained DPTNet

[6] https://github.com/jiaaro/pydub
[7] https://github.com/AndreevP/wvmos

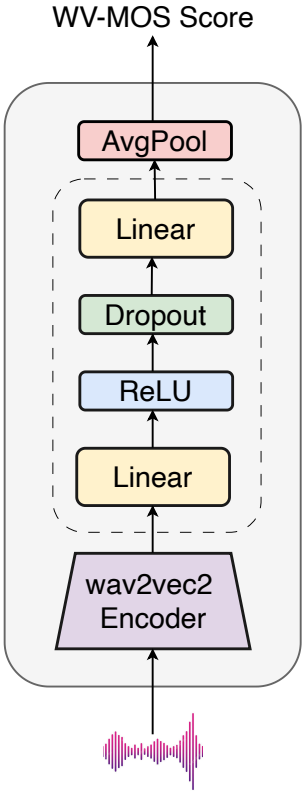


Figure 3.4: WV-MOS architecture.

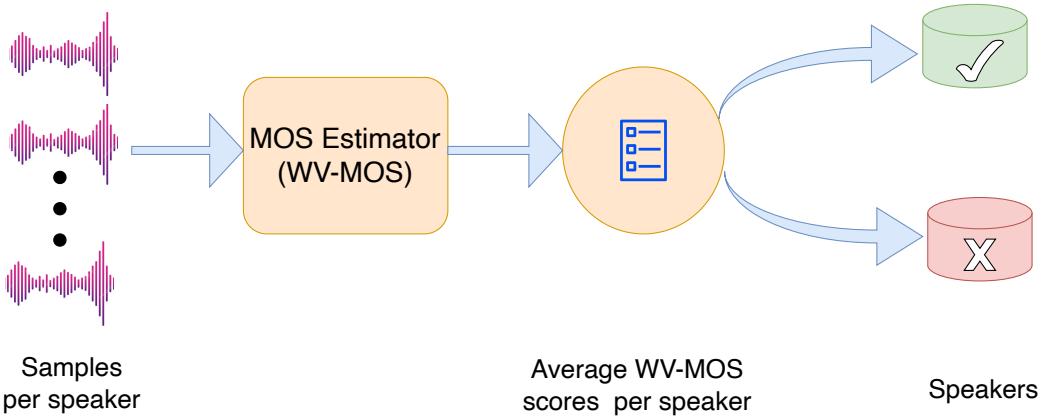


Figure 3.5: Dataset filtering pipeline.

model of Asteroid (Pariente et al., 2020). The DPTNet model had been trained on the LibriMix dataset (Cosentino et al., 2020) for single-channel speech separation. We ran separate TTS training experiments using the original and denoised datasets.

Recalling that WV-MOS scores are real numbers in the range of 1.0 to 5.0. We chose five thresholds WV-MOS $\geq \{2.0, 3.0, 3.5, 3.8, 4.0\}$ for analysis, with higher threshold values indicating higher quality. The thresholds were selected to balance data size and number of speakers. In the case of denoised training samples, we still made use of the data splits of the original dataset after denoising so that the list of selected speakers was not affected

and we can make comparisons among datasets before and after denoising.

In addition, to balance the speaker data size available, we limited the range of speaker duration to between 20 minutes and 10 hours in the first stage of filtering. Here, a random 10-hour subset of data belonging to the speakers with a total duration longer than 10 hours was selected and we discarded speakers with a total duration of less than 20 minutes. Utterances with very long duration (longer than 20 seconds) were also excluded from the data to allow large batch sizes during TTS training. Table 3.2 shows the training data duration and number of speakers corresponding to each WV-MOS threshold. We trained a TTS model on each dataset in Table 3.2. These models were also compared with a *baseline* TTS model trained on all available data that has not been filtered with the WV-MOS metric.

Table 3.2: Training data duration and number of speakers for various selected WV-MOS thresholds.

WV-MOS threshold	Duration (h)	Number of speakers
Baseline	636.27	633
WV-MOS $\geq$ 2.0	620.14	623
WV-MOS $\geq$ 3.0	532.14	537
WV-MOS $\geq$ 3.5	310.40	337
WV-MOS $\geq$ 3.8	187.40	183
WV-MOS $\geq$ 4.0	86.05	88

3.2.2 TTS models

We evaluated the dataset quality for TTS training at each defined WV-MOS threshold by training a multi-speaker TTS model. Our TTS model is a multi-stage TTS model with a separate Mel-spectrogram predictor and vocoder. In this section, we will detail the TTS architecture used in this chapter of the thesis.

Glow-TTS

We used a multi-speaker Glow-TTS model (Kim et al., 2020), conditioned on an external speaker embedding, similar to Casanova et al. (2021). This model comprises a Transformer encoder and a flow-based decoder, along with a phoneme duration prediction network. The model choice was influenced by its quality and its relatively short training time compared to other TTS models. Each utterance's speaker embedding and the corresponding sentence converted into phonemes were used as inputs to the model during training. The output is a Mel-spectrogram. We refer readers to Section 2.2.4 for a more detailed description of the model architecture.

Speaker embedding extractor

Since the Glow-TTS model requires speaker embeddings for speaker conditioning, we pre-computed a 256-dimensional speaker embedding from the Resemblyzer speaker extraction model[8] for each utterance. Given any speech waveform, Resemblyzer creates a summary vector of 256 values summarising the characteristics of the speaker. The Resemblyzer

[8] https://github.com/resemble-ai/Resemblyzer

model had been trained on the Voxceleb (Nagrani et al., 2017) dataset and has been widely used for speaker embedding extraction, e.g., by Casanova et al. (2021). In addition, the 256-dimensional embeddings extracted were L2-normalised. We refer readers to Section 2.2.4 of this thesis for a more detailed explanation of speaker representation in TTS models.

HiFi-GAN vocoder

A 16 kHz HiFi-GAN $V1$ vocoder (Kong et al., 2020) was trained on the LibriTTS dataset to convert the generated Mel-spectrograms into audio signals. The vocoder was fixed and used to evaluate all the TTS models considered. We decided to fix the vocoder and use the trained vocoder to convert all the Mel-spectrograms to audio to objectively evaluate the generated Mel-spectrograms. As such, the trained vocoder was not finetuned on Mel-spectrograms generated by Glow-TTS.

Finally, we show the training and evaluation pipeline in Figure 3.6. More details will be provided about the evaluation metrics used in Section 3.3. All our training experiments were carried out using the NeMo toolkit (Oleksii Kuchaiev et al., 2019).

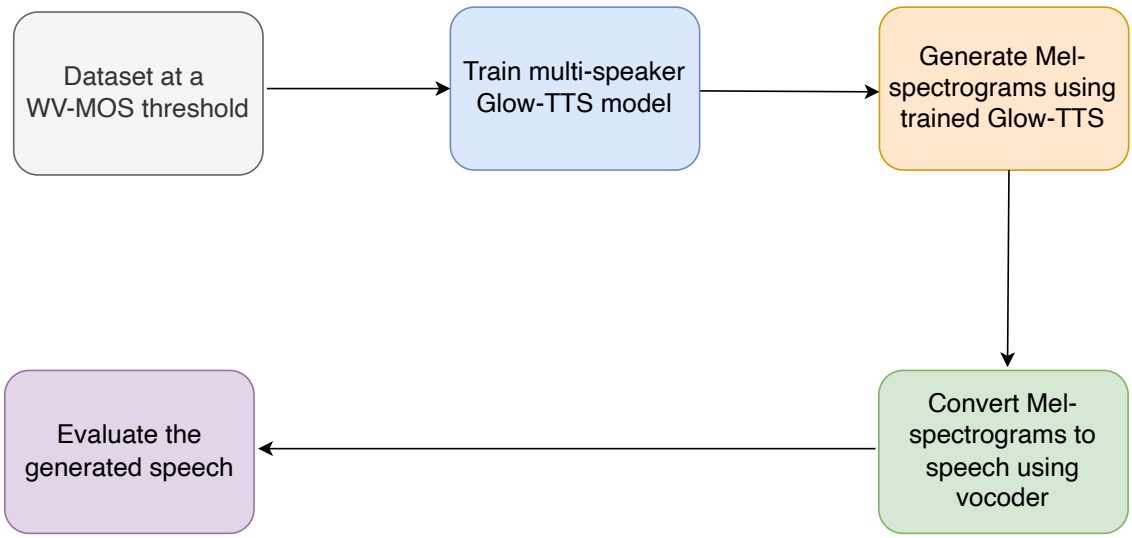


Figure 3.6: TTS training and model evaluation pipeline.

3.3 Experimental evaluation

3.3.1 Hyperparameters

An extensive list of hyperparameters used in this experiment is provided in Table 3.3. In summary, all TTS models were trained using a global batch size of 128 or 256 on 4 GPUs with a gradient accumulation of 2. Using 128 global batch size as an example, this implies that we load a batch of 16 samples on each GPU and accumulate the gradients for 2 iterations of the data before the models are updated (the noisier datasets only learn when the batch size is large). The model was optimised using the RAdam optimiser, a learning rate of 0.001, and a cosine-annealing scheduler with linear warm-up steps of 6,000. Each model was trained until the validation loss stopped decreasing for more than 10 epochs.

We selected 504 utterances randomly from each dataset for validation. At inference, the generation was done with a noise scale of 0.667 and length scale of 1.0, which were the best multi-speaker inference parameters reported by Kim et al. (2020). The value of the noise scale affects the pitch of the speaker while the length scale controls the speaking rate.

Table 3.3: Hyperparameters used to train the Glow-TTS models.

Training hyperparameters	
Early stopping patience	10 epochs
Monitor for early stopping	Validation loss
Gradient clipping value	5.0
Accumulate gradient batches	4
Epochs	150
Number of GPUs	4
Warmup steps	6,000
Optimiser	RAdam
Learning rate	10^{-3}
Weight decay	0.0
Learning rate scheduler	Cosine annealing
Dataset hyperparameters	
Batch size	32 or 64
Sample rate	16,000
Maximum frequency	8,000
Minimum frequency	0
Pad value	-11.52
Speaker embedding dimension	256
Number of Mel bins	80
Inference hyperparameters	
Noise scale	0.667
Length scale	1.0
Vocoder sample rate	16,000

3.3.2 Objective evaluation

To evaluate the TTS models objectively, we generated synthetic voices for *seen speakers* and *unseen speakers*. Seen speakers are all the speakers that have been included in the training dataset of the TTS model while unseen speakers are speakers that have not been used to train any of the TTS models. Eighty speakers were randomly selected from the smallest subset (WV-MOS $\geq$ 4.0) of Common Voice to represent *seen speakers*. By design, all speakers in that subset were also included in the lower WV-MOS threshold training subsets. We similarly selected 80 speakers from the VCTK corpus to represent *unseen speakers*. For each seen speaker, a single speaker embedding was extracted from a reference utterance of that speaker, which is a randomly selected utterance with a duration longer than 2 seconds. For unseen speakers, we extracted speaker embeddings from the fifth utterance (SpeakerID_005) of each speaker in the VCTK dataset as Casanova et al. (2021). The speaker embeddings were used to generate 25 utterances for each speaker,

using text sentences from the VCTK corpus with more than 20 words. This resulted in a total of 2,000 test utterances for both seen and unseen speakers.

We computed the following objective metrics for the generated synthetic voices.

- **WV-MOS score**: this measures the quality of the generated utterances objectively. This is computed using the WV-MOS prediction model for each utterance and then averaged over all the utterances belonging to a particular set to be evaluated.

- **cos-sim**: this measures the speaker similarity between the reference speaker and the speaker of the generated utterance. The cosine similarity (cos-sim) between the speaker embeddings of the generated utterance and the reference utterance is used to calculate the cos-sim. We refer readers to Equation (2.4) on how to compute the cos-sim. The speaker embeddings for the generated utterances were also extracted from the Resemblyzer speaker extraction model.

- **Character error rate (CER)**: this measures the intelligibility of the generated utterances objectively. The CER detects if the TTS model has made some deletions, insertions, or substitutions in the sentences used to generate the utterances or if some words have not been rendered by the TTS model during generation. The pre-trained ASR model QuartzNet15x5 (Kriman et al., 2020) from the NeMo toolkit was used to get predictions for each utterance, and each prediction is compared with the reference text provided to the TTS model to compute the CER. We remove all punctuation during CER computation except the apostrophe. A lower CER shows that the TTS model has made fewer errors and the utterances generated by the model are more intelligible compared to another TTS model with a higher CER. The ASR model was trained on LibriSpeech and Common Voice English "validated" set.

3.3.3 Subjective evaluation

We performed several listening tests that spanned about an hour per listener with 24 volunteers participating in the listening test. We asked listeners to use a headset and to set the volume so as to hear the audio files clearly without external interference. The listeners were also asked to perform the test in a quiet environment without long breaks, disturbance, or background noise.

Similar to the objective evaluation, we evaluated the TTS models subjectively using the following metrics.

- **MOS**: this measures the overall quality of the utterances. The listeners were asked to rate the utterances on a scale of 1–5 with increments of 1.0. A value of 1.0 denoted worse quality and a value of 5.0 meant excellent quality on the scale. We asked them to choose a score on the scale by considering how much the audio sounds natural, and non-robotic, and to consider how much the utterances contain noise, while also ignoring any form of mispronunciation of words in their decision. The utterances that have been generated by different TTS models using the same text were presented on the same page to the listener to prevent MOS evaluation drift. The evaluator still listens to each utterance on a page and rates them independently.

- **S-MOS**: The speaker MOS or S-MOS evaluates the speaker similarity between the generated and reference utterances. The listener listens to a reference utterance with the same sentence and then in turn listens to several utterances generated with

the speaker embedding belonging to the reference speaker. The listeners were then asked to rate the utterances on a scale of 1–5 with increments of 1.0, where a value of 1.0 denotes that the speakers are not the same, and 5.0 indicates that the reference speaker and speaker in the generated utterance are exactly the same. We asked listeners to ignore the speaking rate in their scoring, as the reference speaker often speaks faster than the generated utterances. A high S-MOS shows that the TTS model was able to generate utterances belonging to a speaker more accurately when trained on a particular TTS dataset.

- **Intelligibility score**: The intelligibility score measures the intelligibility of the generated utterances subjectively. For this score, we used the minimal pair approach (Hodge and Gotzke, 2011) of intelligibility evaluation. A minimal pair is a pair of words that vary by only one phoneme, e.g., "sea" vs. "she". This approach helps identify deficiencies in phoneme generation by the TTS models, and it is widely used by phoneticians. In the evaluation, a carrier audio containing one of the pairs was presented to the evaluator and the evaluator had to select the word heard after the carrier audio among three options: the correct word (e.g., "sea"), its minimal pair (e.g., "she") or none of these. The format used for the carrier audio is "The word to be recognised is [word]", e.g., "The word to be recognised is she". The proportion of correctly associated words was then computed as the *intelligibility score*.

For MOS and S-MOS evaluations, we arbitrarily selected speakers from the VCTK corpus to evaluate the models on unseen speakers. Concretely, we randomly selected two unseen male speakers (p245 and p254) and two unseen female speakers (p231 and p250) from the speakers in the corpus. In addition, we also arbitrarily selected five sentences from the text in the corpus that contain more than five words and yet are not too long, to generate short utterances between 3 and 5 seconds. We used these five sentences and the VCTK speakers to generate five utterances per speaker for all TTS models. Since this evaluation aims to compare different systems, we are confident that the number of speakers does not affect the comparison in terms of the quality of generated utterances and speaker similarity, and therefore enables us to keep the listening test duration sizeable.

Additionally, for S-MOS evaluation, we presented listeners with a reference utterance from the VCTK corpus and the utterances synthesised using the reference speakers' embedding and text. Each TTS model uses the reference utterance's text and speaker embedding as input to generate an utterance. In general, a rating is discarded in our evaluation if the listener does not play the audio corresponding to that rating and/or does not play the reference audio in the case of S-MOS evaluation.

For the intelligibility score evaluation, we selected 25 random minimal pairs from the list of minimal pairs provided in Table 3.4.[9]

3.4 Results

3.4.1 Impact of the WV-MOS threshold

Figure 3.7 shows the objective evaluation plots for objective quality (WV-MOS), speaker similarity (cos-sim), and intelligibility (CER). We explain the results of each of the metrics

[9]The minimal pairs were collected from https://www.englishclub.com/pronunciation/minimal-pairs.php

Table 3.4: List of minimal pairs used for the TTS evaluation.

Minimal pairs					
	Word A	Word B		Word A	Word B
1	sitting	suiting	26	pie	buy
2	sit	seat	27	rich	which
3	mall	mail	28	thing	thin
4	desk	disk	29	rat	hat
5	catch	catchy	30	alive	arrive
6	wet	wait	31	weight	gate
7	edge	edgy	32	catch	cat
8	bat	but	33	vow	wow
9	thing	think	34	she	sea
10	saw	so	35	do	two
11	thin	thing	36	fan	van
12	not	note	37	came	game
13	bus	buzz	38	fat	hat
14	bed	bad	39	past	fast
15	had	hat	40	free	three
16	first	fast	41	three	tree
17	am	an	42	think	sink
18	hard	had	43	day	they
19	back	bag	44	whizz	with
20	caught	cot	45	share	chair
21	eight	hate	46	pays	page
22	know	now	47	jeep	cheap
23	where	air	48	bad	badge
24	very	berry	49	quick	kick
25	jaw	your	50	copy	coffee

used below. Additionally, Table 3.5 shows the subjective and objective quality (MOS / WV-MOS) and speaker similarity (S-MOS / cos-sim) of utterances generated for four unseen speakers by TTS models trained on the baseline dataset and the WV-MOS ≥ 3.0 and WV-MOS ≥ 4.0 datasets. We subjectively evaluated three thresholds of the filtered data to show relative quality improvements in the TTS datasets as the threshold increases. In the table, we use a bold font to indicate the score of the best TTS model in each row and the TTS model statistically equivalent to it, which serves as an indication of the TTS dataset quality at the given threshold.

Increase in quality of generated utterances

In Figure 3.7a, we plot the WV-MOS scores averaged over the generated test samples for seen and unseen speakers. In addition, we plot the WV-MOS scores for the same unseen speakers where the TTS dataset used to train the TTS models has been denoised. Each of the WV-MOS thresholds denotes the threshold used for filtering the training dataset used to train the TTS model. Also, recall that a higher threshold value indicates that the selected speakers (and utterances) had a WV-MOS score above that threshold. Firstly, we observed that the WV-MOS scores in Figure 3.7a follow an increasing trend

Table 3.5: Subjective / objective quality (MOS / WV-MOS) and speaker similarity (S-MOS / cos-sim) of utterances generated for four unseen speakers by TTS models trained on the baseline dataset and the WV-MOS $\geq$ 3.0 and WV-MOS $\geq$ 4.0 datasets. Bold numbers denote the best system in each row and the systems statistically equivalent to it.

	Baseline	WV-MOS $\geq$ 3.0	WV-MOS $\geq$ 4.0
MOS	2.35	3.12	**3.69**
WV-MOS	3.63	3.59	**4.09**
S-MOS	2.69	**2.90**	**2.79**
cos-sim	**0.831**	**0.845**	**0.832**

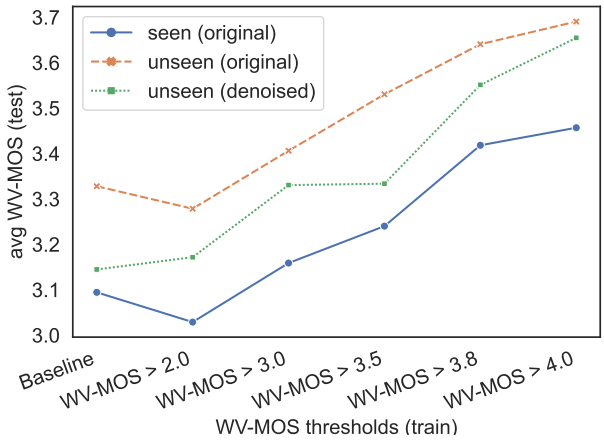
from the baseline to the WV-MOS $\geq$ 4.0 dataset. Subjective MOS results in Table 3.5 also corroborate this. This indicates that filtering based on WV-MOS scores improves the quality as expected. Additionally, we also observed that the TTS models were able to generate even higher-quality utterances for unseen speakers than speakers seen during training. This may indicate that the model learns to also associate deficiencies or noise in the training dataset with the speaker embeddings, and therefore affects the quality of generated utterances. Also, denoising the dataset shows to reduce the WV-MOS scores for the unseen speakers for all WV-MOS thresholds.

Noise affects speaker similarity

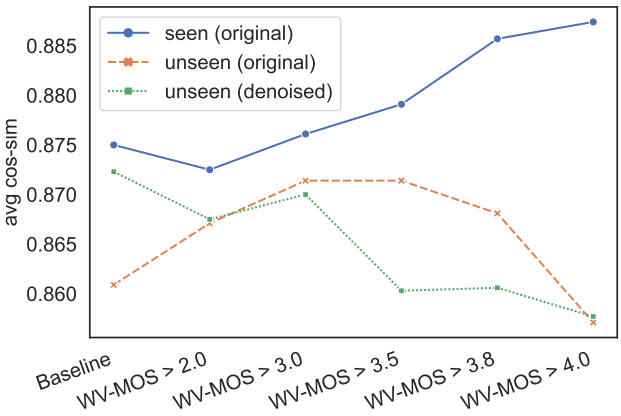
In Figure 3.7b, for models trained on original data, the cos-sim for seen speakers shows an increasing trend indicating that the model can learn to produce utterances with higher speaker similarity when the TTS dataset is of high quality. For unseen speakers, the cos-sim is lowest at the two ends of the plot probably due to the amount of noise in the dataset on one end and the low number of training speakers to learn from at the right end of the plot. We see a similar trend in Table 3.5 for the S-MOS of unseen speakers. This indicates that more training speakers are necessary for modelling speaker variability in the TTS model. For the denoised TTS dataset, the speaker similarities between the reference and generated speaker embeddings oddly show a decreasing trend, and indicate that denoising only improves speaker similarity for the very noisy TTS dataset, while reducing the speaker similarity as the dataset becomes high-quality.

Higher-quality TTS models generate more intelligible utterances

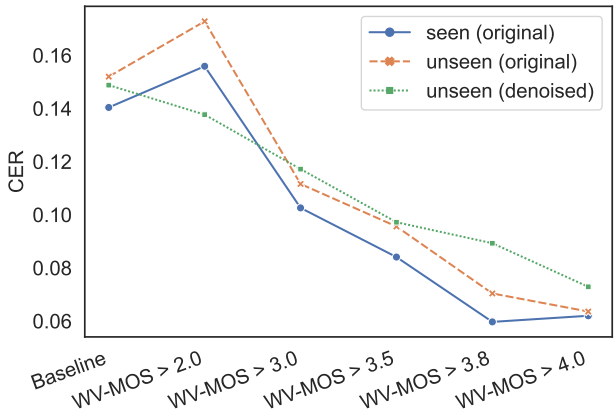
In Figure 3.7c, we show the CER computed between the ASR predictions for the generated utterances and the reference texts. The CER decreased steadily from the noisy baseline dataset to the less noisy WV-MOS datasets for both seen and unseen speakers. Here, we saw a reduction of more than 50 % in the CER from the baseline to the WV-MOS $\geq$ 4.0 dataset. This indicates that intelligibility is mostly affected by the quality of the dataset, not the size. The lowest quality datasets (Baseline vs. WV-MOS $\geq$ 2.0) do not follow these trends due to the higher variance in utterance-level WV-MOS scores for lower-quality speakers.



(a) WV-MOS scores for seen and unseen speakers.



(b) Cosine similarity between speaker embeddings.



(c) CER for seen and unseen speakers.

Figure 3.7: Objective quality (WV-MOS), speaker similarity (cos-sim), and intelligibility (CER) of utterances generated for 80 seen and 80 unseen speakers by TTS models trained on original or denoised samples above a given WV-MOS threshold.

Table 3.6: Subjective quality (MOS), speaker similarity (S-MOS) and intelligibility of utterances generated for four unseen speakers by TTS models trained on the baseline dataset, LibriTTS, and the WV-MOS $\geq$ 4.0-all dataset. Corresponding objective scores (WV-MOS and cos-sim) are included. Bold numbers denote the best system in each row and the systems statistically equivalent to it. Copy-synthesis on VCTK speech (VCTK-copy) provides an upper bound on the achievable speaker similarity.

		Baseline	LibriTTS	WV-MOS $\geq$ 4.0-all	VCTK-copy
MOS	Male	2.44	3.15	**3.70**	-
	Female	2.26	**3.38**	**3.52**	-
	Total	2.35	3.26	**3.61**	-
WV-MOS		3.63	**3.75**	**3.80**	-
S-MOS	Male	**2.92**	**2.95**	**3.02**	4.50
	Female	2.46	2.53	**2.73**	4.73
	Total	2.69	2.74	**2.88**	4.61
cos-sim		0.831	**0.861**	**0.861**	0.869
Intelligibility score		0.72	**0.82**	**0.82**	-

3.4.2 Impact of denoising training utterances

As seen in Figures 3.7a and 3.7c, the WV-MOS scores and CER for unseen speakers follow the same trend, whether the training data are denoised or not. Furthermore, denoising degrades the WV-MOS score and the CER, except for low-quality datasets in the case of the CER. In Figure 3.7b, we see that denoising also degrades the cos-sim score, except for the baseline. This shows that some speaker information may be lost during denoising. We conclude that denoising the dataset is not beneficial and instead, automatically selecting high-quality samples is the better strategy.

3.4.3 Common Voice vs. LibriTTS

We compared our dataset curation method to a standard TTS dataset: LibriTTS. Since LibriTTS has more speakers (2,484) and more data (492.68 hours after discarding utterances longer than 16.7 seconds) than the WV-MOS $\geq$ 4.0 dataset, we selected all speakers with WV-MOS above 4.0, without setting a 20 minute lower bound on total speaker duration. We call the resulting 4,469-speaker, 230.75 hour dataset WV-MOS $\geq$ 4.0-all. In Table 3.6, we provide the evaluation results for the utterances generated by the TTS models trained on this dataset vs. LibriTTS for unseen speakers. These datasets were also compared against the baseline of the available training dataset. In addition, we applied copy-synthesis on VCTK speech (VCTK-copy) to provide an upper bound on the achievable speaker similarity when the vocoder is constant. In copy-synthesis, a real speech utterance is converted into its Mel-spectrogram and then converted back into speech using a trained vocoder.

Both datasets have similar intelligibility

From Table 3.6, we can see that training on WV-MOS $\geq$ 4.0-all resulted in a similar intelligibility score to training on LibriTTS, while the baseline has a significantly lower intelligibility score indicating that noise affects intelligibility. This also shows that the

WV-MOS $\geq$ 4.0-all dataset is high-quality and can be used to train very intelligible TTS models.

WV-MOS $\geq$ 4.0-all improves speaker modelling for female speakers

Looking at the S-MOS scores in Table 3.6, the score was highest when training on WV-MOS $\geq$ 4.0-all. Comparing the S-MOS for each gender, we observe that WV-MOS $\geq$ 4.0-all produces significantly higher S-MOS scores for unseen female speakers and similar S-MOS scores for seen speakers. This shows that the WV-MOS $\geq$ 4.0-all can be useful for improving the modelling of female speaker characteristics in the TTS model. In addition, we noticed that male S-MOS scores are higher than female S-MOS scores for both datasets, which indicates that male speakers are better modelled by models trained on either dataset. We still saw a large gap in S-MOS between the TTS model trained on WV-MOS $\geq$ 4.0-all and the VCTK-copy topline, which leaves room for further improvement in speaker modelling. Also, the cos-sim was not statistically different between the two datasets, while still better than the baseline.

Similar overall quality between the two datasets.

Finally, while the WV-MOS scores for utterances generated by training on WV-MOS $\geq$ 4.0-all vs. training on LibriTTS were not statistically different, volunteers consistently gave higher MOS scores to the former, with an average improvement of 0.35 MOS point. Male speakers were assigned higher MOS scores, unlike LibriTTS where female voices were given higher MOS scores (in line with Zen et al. (2019)).

3.4.4 Other factors not captured by WV-MOS

We performed experiments on a medium-quality dataset, WV-MOS $\geq$ 3.5, to assess the impact of other factors. First, we removed sentences in a foreign language by using a language identification tool, LangID (Lui and Baldwin, 2012), to filter out sentences with English language probability lower than 0.8. Second, using the same pre-trained ASR model as above, we dropped utterances with a CER above 0.4 for better alignment of training text to utterances. Third, we filtered utterances according to their WV-MOS at the utterance level, instead of the speaker level. Finally, we removed pauses longer than 180 ms inside utterances using a voice activity detector[10] Each of these experiments resulted in discarding less than 1.5 % of the initial dataset. Informal listening tests did not show any improvement in quality and intelligibility. Although this would have required more formal tests to validate, we conclude that the other factors not captured by WV-MOS are not critical.

3.5 Application of the data selection method

The curation of high-quality TTS datasets from recordings is an area of research that is been explored widely in the TTS community. As earlier discussed, for some domains the only available dataset is likely to have been crowdsourced or recorded for other applications such as ASR. Re-purposing such a dataset for TTS requires extra effort, and in this work, we have shown that MOS estimation using quality estimators can be useful in the data

[10] https://github.com/wiseman/py-webrtcvad

curation pipeline. As an example, WV-MOS was used as one of the quality estimators to build a TTS corpus for the Arabic language using a news recording dataset (Baali et al., 2023). In our recent work, we also applied this method in a data processing step to select the utterances with high quality. In particular, we crowdsourced an African-accented speech dataset containing 86 African accents from 9 countries. The purpose of the data collection effort was to create a TTS model that is representative of the diverse accents on the African continent. As the dataset was recorded remotely on different devices, it needed to be processed to be suitable for training the TTS models. The recordings were first denoised to remove noise and room reverberation and then passed through a bandwidth-extension model to improve the quality of some of the highly degraded utterances, i.e., utterances recorded with low-resolution and clipping distortion. To keep the recording with the highest quality among the enhanced files per utterance, we applied the WV-MOS quality estimation method. The WV-MOS was used to estimate the quality of the samples and the file with the highest predicted WV-MOS score among the enhanced samples per recording was selected to be included in the TTS dataset. We have detailed the data curation process in Section A.2. The curated dataset was used to train an African-accented multi-speaker TTS model. More detail about the work has been provided in Appendix A.

3.6 Conclusion

In this chapter, we automatically selected high-quality training samples from a crowd-sourced dataset, using Common Voice English as an example. We estimated the quality of recordings using a non-intrusive self-supervised MOS estimator, WV-MOS, and then filtered the dataset at different thresholds of WV-MOS scores to evaluate the relative quality of the filtered dataset.

We successfully improved the overall quality, speaker similarity, and intelligibility of utterances generated by a multi-speaker TTS model trained on the Common Voice English dataset. Furthermore, we showed that denoising reduces the CER and increases the speaker similarity score (cos-sim) of generated utterances when the dataset is noisy, but degrades performance otherwise. Finally, we curated a high-quality 4,469-speaker, 230.75 hour dataset from the Common Voice corpus suitable for training TTS models as it even outperforms LibriTTS in terms of both subjective quality and speaker similarity. The applied approach is generic and could enable the creation of TTS training datasets for languages for which manual curation is not financially viable.

4

Improving the diversity of generated utterances

In the previous chapter, we examined the process of repurposing a crowdsourced ASR dataset as a TTS dataset. The curated TTS dataset is of high quality and has some desired properties, such as a large speaker representation and reduced noise, which will enable us to train TTS models for zero-shot multi-speaker TTS. Having successfully curated the TTS dataset, we shift our focus towards the TTS model in this chapter. Concretely, we examine how to improve the controllability of the TTS model to generate diverse and natural utterances. As we discussed earlier, given the one-to-many mapping between phonemes/characters and a speaker's utterance, we need to be able to model, control, and include the variabilities that improve the perceived quality of speech for speech synthesis. Some TTS systems model these variabilities by explicitly learning the speaker's characteristics such as pitch, energy, or rhythm (Ren et al., 2019; Łańcucki, 2021). For example, in TTS models like FastSpeech2 (Ren et al., 2020) and FastPitch (Łańcucki, 2021), the duration, pitch contour, and energy of the input tokens were explicitly learned. Since these systems are deterministic by design, they are unable to output diverse renderings for a given text input and speaker. One method of improving these models is to replace the deterministic pitch and energy predictors with normalising flows (Lee et al., 2022).

Flow-based TTS models like Glow-TTS can learn the distribution of the Mel-spectrogram (or audio), which includes all the speaker's characteristics, given the tokenised text inputs and a speaker embedding. This allows them to generate diverse utterances at inference time, however, this does not allow for external control of the variabilities in speech. The quality of generated utterances in the multi-speaker setup is also lower than for single-speaker TTS. For multi-speaker TTS, flow-based TTS models condition the model on speaker embeddings which are learned with the model (Kim et al., 2020) or extracted from a speaker extraction model (Casanova et al., 2021) for zero-shot TTS. The lower quality of generated utterances in the multi-speaker case is probably because flow-based models are unable to learn the diversity of speaking styles inherent to each speaker given only the speaker embeddings. In particular, when these models are used to generate speech for unseen speakers, the generated utterances often sound monotonic and less natural. Therefore, the task in this chapter is to design an improved flow-based TTS architecture that can match the diversity of real speech, including for unseen speakers.

In particular, we focus on a design that enables us to generate realistic and diverse utterances for speakers seen and unseen during training. We propose to achieve this by

explicitly learning the distribution of phoneme duration and pitch during training, to be able to sample from that distribution at inference.

The following sections are organised as follows: we present the proposed improved Glow-TTS architecture in Section 4.1, then we describe the experimental setup to validate our architecture in Section 4.2. In Section 4.3, we provide results and discuss the significance of the results. We finally conclude in Section 4.4. The contents of this chapter have been published at Interspeech 2023 (Ogun et al., 2023b).

4.1 TTS model

In this section, we first briefly describe the classical Glow-TTS architecture and then motivate several changes and improvements in the architecture.

4.1.1 Glow-TTS overview

The Glow-TTS architecture has been described in Section 2.2.4 with a breakdown of the model's architecture provided in Figure 2.6, however we briefly describe the model here to motivate the changes to the architecture. The classical Glow-TTS architecture as shown in shown in Figure 4.1 comprises a Transformer-based encoder, a flow-based decoder, and a deterministic duration predictor. The Transformer encoder generates contextual embeddings from the tokenised text input, which are then linearly projected into an 80-dimensional representation of the mean μ of the prior distribution through the projection layer. The prior distribution μ is then converted into a latent representation of

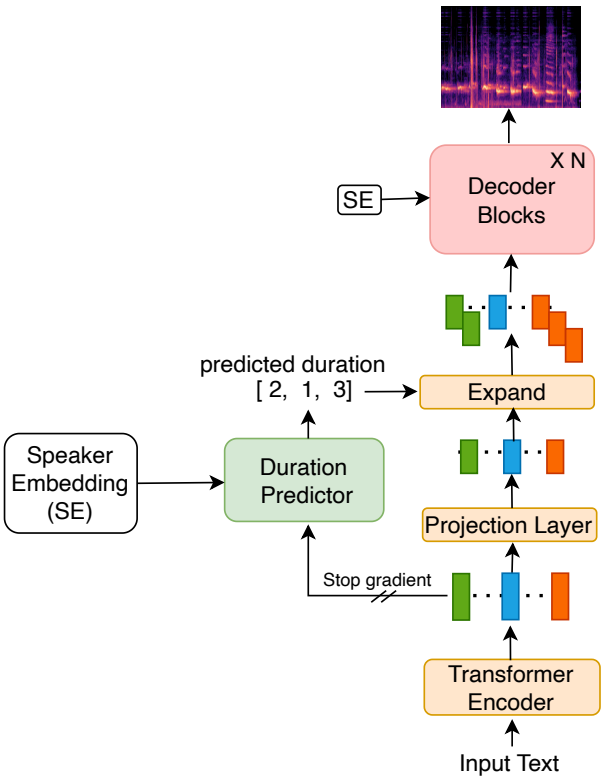


Figure 4.1: Glow-TTS architecture showing the computation path during inference.

the Mel-spectrogram by adding a randomly sampled Gaussian noise. At training time, the tokenised text input durations are estimated using the monotonic alignment search algorithm and are used to expand the latent representation. The duration predictor is also trained alongside the TTS model using the estimated duration as the target. Hence, at inference time the latent representation is expanded using the tokenised text input durations predicted by the duration predictor. The latent representation, alongside a speaker embedding, is then passed as input into the decoder to generate a Mel-spectrogram.

4.1.2 Modifications to the TTS architecture

Several modifications to the architecture were made to adapt the TTS system to our objective. Figure 4.2 illustrates the proposed improved architecture during inference. First, we describe the TTS decoder and the changes that were made in the decoder, then we discuss the incorporated stochastic duration predictor and the stochastic pitch predictor.

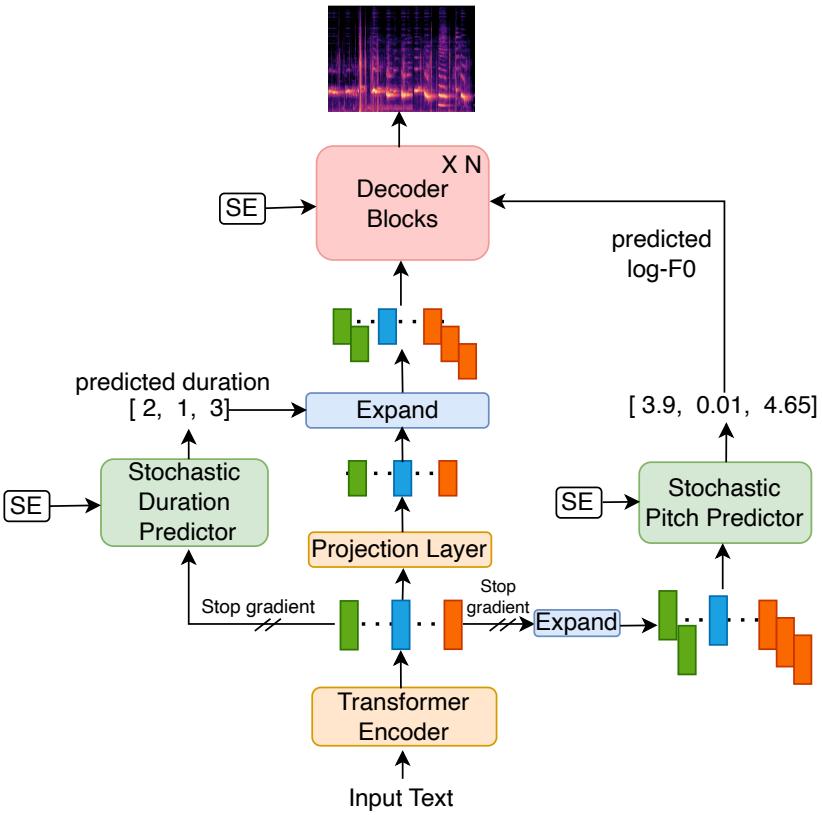


Figure 4.2: Proposed improved Glow-TTS architecture including a stochastic duration predictor and a stochastic pitch predictor. SE denotes the speaker embedding used to condition the model.

Decoder

As shown in Figure 4.3, the decoder is a stack of N normalising flow blocks, each consisting of an activation normalisation layer, an invertible 1x1 convolution layer, and an affine

coupling layer. The activation normalisation layer (ActNorm) performs a per-channel

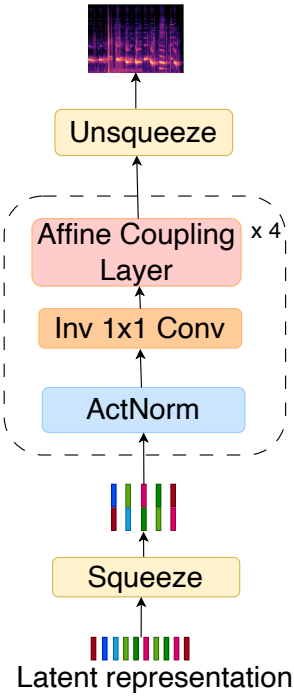


Figure 4.3: A breakdown of the flow-based decoder. The speaker and pitch conditioning are not shown.

affine transformation of the activations in that layer using a scale and bias parameter, similar to batch normalisation. These parameters are initialised such that the per-channel activations after applying ActNorm have zero mean and unit variance given an initial minibatch of data. After initialisation, the scale and bias are trained alongside other parameters in the TTS model. The invertible 1x1 convolution layer applies a permutation that reverses the ordering of the channels in order to ensure that after sufficient steps of flow, each dimension can affect every other dimension. The affine coupling layer is an affine transformation with efficient computations of the forward function, the reverse function, and the log-determinant.

As depicted in Figure 4.4, the affine coupling layers enable the external conditioning to be applied to the decoder features. In our multi-speaker TTS model, the decoder is also conditioned on both the speaker representation and pitch representation through the affine coupling layers. Furthermore, as shown in Figure 4.5a, for the speaker representation, the L2-normalised speaker embedding of the target speaker is passed through a convolution layer to adapt its channel dimension to the dimension of the decoder features, then weight normalised and finally expanded in the time dimension to match the time dimension of the decoder. For the pitch features, as shown in Figure 4.5b, we used the log-F0 values extracted from the ground-truth Mel-spectrograms as pitch targets and set all unvoiced values to zero during training. The log-F0 is first projected to match the dimension of the decoder features using a 1D convolution layer and normalisation layer. We squeezed the projected speaker representation and log-F0 features to match the squeeze ratio of the original decoder features to make them compatible with the features in the affine coupling layer.

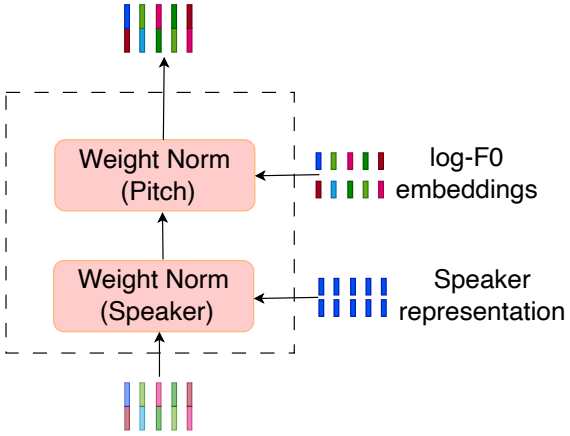
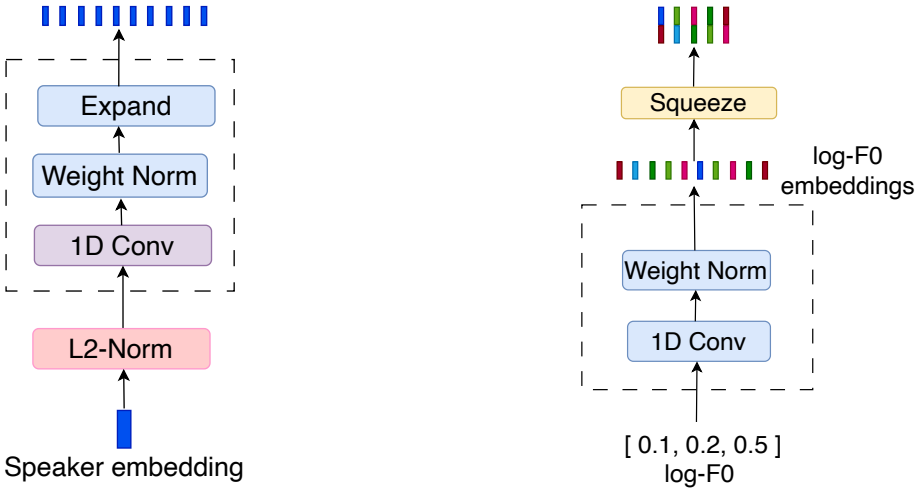


Figure 4.4: Affine coupling layer



(a) Speaker conditioning.

(b) Pitch conditioning.

Figure 4.5: Speaker and log-F0 processing for speaker and pitch conditioning.

Due to the nature of normalising flows, the decoder can perform transformations in two opposite directions depending on whether it is used during model training or during inference. At training time, the decoder inverts the Mel-spectrogram into a hidden representation, which is aligned to the latent representation from the encoder using an alignment search algorithm called Monotonic Alignment Search. The monotonic alignment search algorithm finds the most probable monotonic alignment between the hidden representation and the mean of the prior distribution.

At training or inference time, the predicted log-F0 values generated by the stochastic pitch predictor and the speaker representation are used for conditioning the decoder while the predicted phoneme duration is used to expand the prior distribution.

Duration and pitch predictors

The stochastic duration and pitch predictors are based on the flow-based duration predictor proposed in VITS (Kim et al., 2021).[11] The stochastic duration predictor helps to generate a more human-like rhythm while the stochastic pitch predictor helps to model the distribution of pitch contours to be able to generate pitch contours that closely match the distribution of real speech contours.

Unlike other flow-based generative models that are typically trained directly via maximum likelihood estimation, the flow-based predictor used here requires some changes. First, for duration modelling, the duration of each tokenised text input is a discrete integer, therefore has to be dequantised to be used with continuous normalising flows. Secondly, the duration of each input phoneme, as well as the log-F0 values, are scalars, hence this prevents high-dimensional transformation using high-capacity models.

To solve the first problem, variational dequantisation (Ho et al., 2019) was applied to the duration of the tokenised text input which makes the tokenised text inputs continuous. Additionally, the dimensions of the input scalars are increased to allow for high-dimensional transformation using variational data augmentation (Chen et al., 2020). Concretely, given the sequence of tokenised text inputs $w \in \mathbb{N}^n$, two new random variables, $u \in [0, 1)^n$ and $v \in \mathbb{R}^n$, which have the same dimension as that of the input sequence w, are introduced to the system. The difference $w - u$ becomes a sequence of positive real numbers that can be used in the continuous flow model. Additionally, w and v are concatenated to make a high-dimensional representation $z \in \mathbb{R}^{2xn}$.

We describe the differences in the use of the flow-based predictor for duration prediction and pitch prediction in the following sections. Also, the variable w is denoted as d for the input duration and p for the pitch contours.

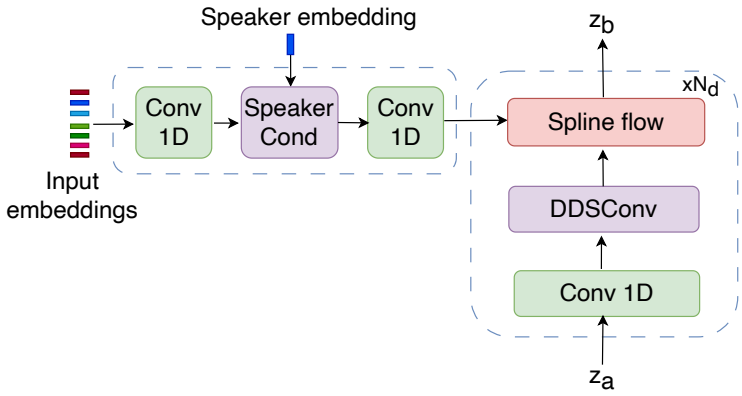


Figure 4.6: Flow module in the stochastic pitch and duration predictors showing the speaker and input embedding conditioning through the spline flow module.

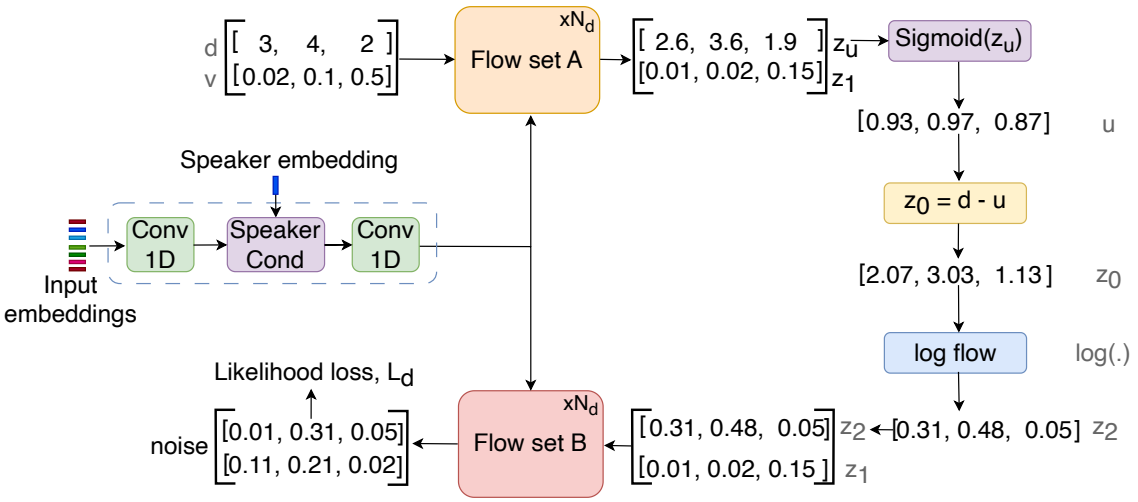
Duration prediction

For the duration prediction modelling, the architecture comprises two stages of normalising flows that learn the discrete phoneme duration. These discrete phoneme durations d are estimated using the monotonic alignment search algorithm during training. The two

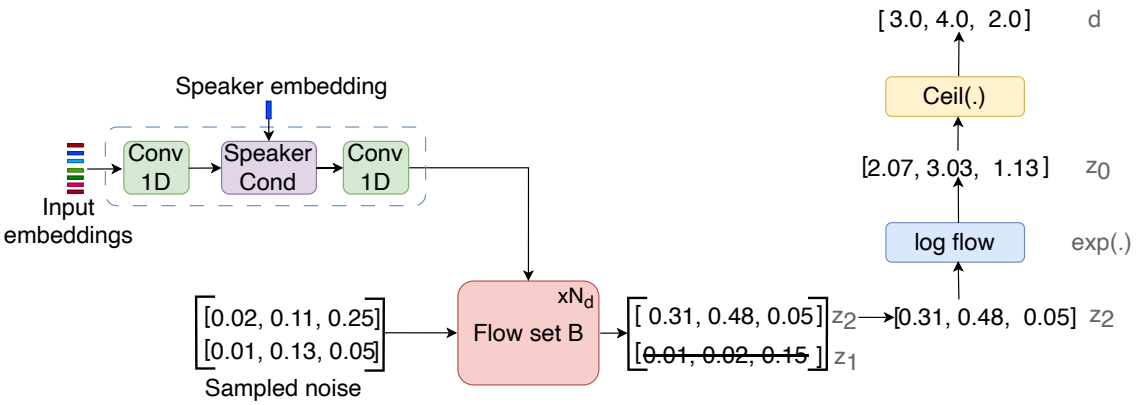
[11] Official implementation can be found at https://github.com/jaywalnut310/vits.

stages of normalising flow modules are each composed of N_d layers of flows that learn the transformation between the estimated duration and noise. Each block in the normalising flow architecture is made up of 1D convolution, dilated and depth-separable convolutions, and neural spline flows (Durkan et al., 2019) as shown in Figure 4.6. The neural spline flows act as coupling layers for external conditioning and use monotonic rational-quadratic splines as its invertible nonlinear transformations. We conditioned features in the flow module on the input embeddings and speaker embedding through the coupling layers in the spline flows.

As shown in Figure 4.7a, at training time, the first set of flows (flow set A) inverts the discrete duration, which is conditioned on the speaker and input embeddings, into a continuous distribution of duration z_0 using variational dequantisation. Then, the de-discretised duration z_0 is passed through the log-flow module to get the estimated continuous log-durations z_2. The log-flow module applies the natural logarithm function to the vector during the inversion process and applies the exponential function in the opposite direction of the flow computation during prediction. Finally, the second set of flows (flow set B) inverts this continuous vector into a Gaussian noise. Additionally, a stop-gradient operator



(a) Stochastic duration predictor during training



(b) Stochastic duration predictor during inference

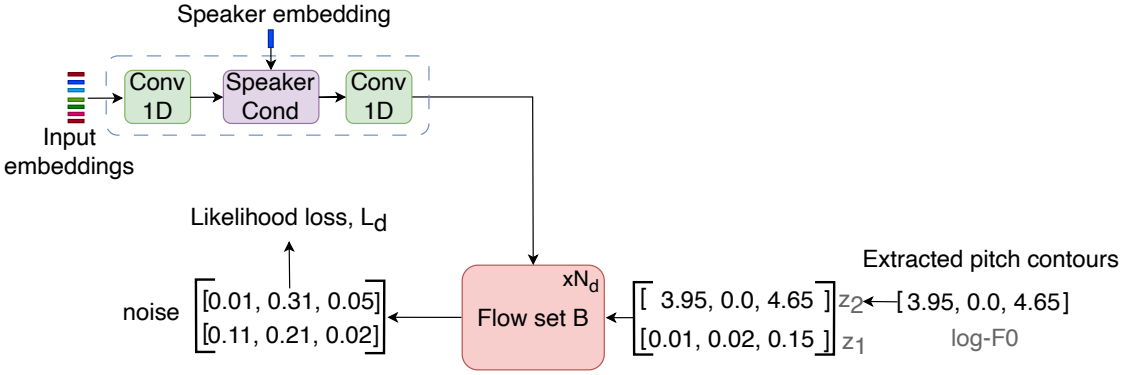
Figure 4.7: Stochastic duration prediction during training and inference. The crossed-out noise vector z_1 is discarded after computation at inference time.

is applied to the input embeddings when learning the duration as shown in Figure 4.2.

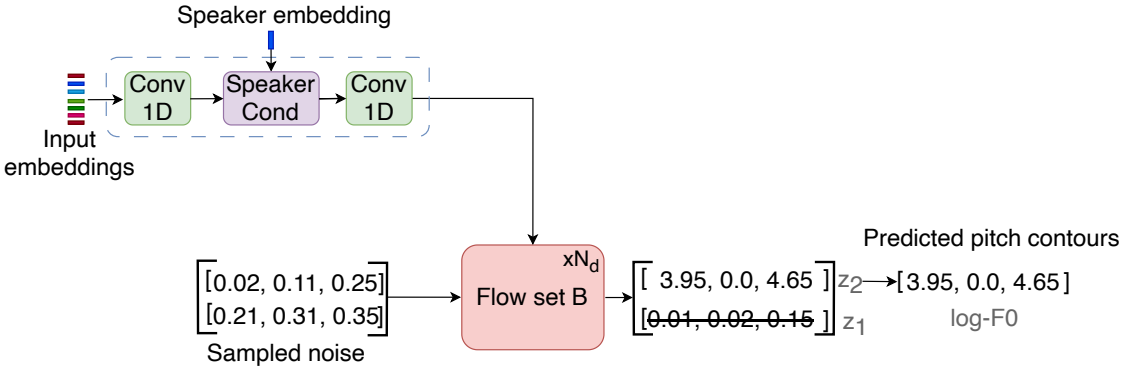
At inference time, the first set of flows is discarded. As shown in Figure 4.7b, two Gaussian noise vectors are simply sampled and passed through the second set of flows to generate the log-durations z_2, that can simply be converted to individual phoneme durations of integers by applying the log-flow and the ceiling function.

Pitch prediction

The stochastic pitch predictor architecture is similar to that of the stochastic duration predictor, except that the first set of flows (flow set A) and log-flow module are not used. This is because the F0 values are not discrete, unlike the duration values. However, we still used log-F0 as input to the second set of flows (flow set B) and applied variational data augmentation as described for the duration predictor. Figure 4.8a shows the structure of the stochastic pitch predictor during training. The logarithm function is applied to the F0 values extracted from the Mel-spectrogram for each frame to get the log-F0 values, which are concatenated with a Gaussian vector of the same length and passed through the second set of flows. The set of flows is conditioned on the speaker-conditioned input embeddings. As depicted in Figure 4.2, the speaker-conditioned input embeddings are expanded to match the duration of each tokenised text input estimated by the monotonic alignment search algorithm. Additionally, a stop-gradient operator is applied to the input embeddings when learning the log-F0 values as shown in Figure 4.2.



(a) Stochastic pitch predictor during training



(b) Stochastic pitch predictor during inference

Figure 4.8: Stochastic pitch prediction during training and inference. The crossed-out noise vector z_1 is discarded after computation at inference time.

At inference time, two random Gaussian noise vectors are sampled and passed through the flows in the reverse direction to generate the log-F0 values as shown in Figure 4.8b. The log-F0 values are directly provided to the decoder to generate Mel-spectrograms.

Loss function

The parameters in the various modules in the TTS architecture are learned using different losses:

$L_{\mathbf{kl}}$: the main loss for the latent representation estimation,

$L_{\mathbf{d}}$: the stochastic duration loss, and

$L_{\mathbf{p}}$: the stochastic pitch loss.

Since the three models are flow-based generative models, the losses minimise the negative log-likelihood of the data. We refer readers to Kim et al. (2020) and Kim et al. (2021) for more information on the log-likelihood estimation of the different modules.

The total loss L_{t} of the model is then computed by combining the three losses as follows:

$$L_{\mathrm{t}} = L_{\mathrm{kl}} + \alpha_1 \, L_{\mathrm{d}} + \alpha_2 \, L_{\mathrm{p}} \tag{4.1}$$

where α_1 and α_2 are scaler hyperparameters that control the weight given to the losses from the duration and pitch predictors. In our setup, the duration and pitch predictors are trained jointly with the remaining network and rely on the contextual text embeddings from the main TTS network as input during training. The scalar weights, α_1 and α_2, are typically far smaller than 1.

4.2 Experimental setup

First, we selected the classical *Glow-TTS* model, which contains 54.5 million parameters, as a baseline. To measure the impact of the stochastic duration and pitch models on the quality of the generated utterances, we developed two variants of the model:

- *GlowTTS-STD*: a Glow-TTS model with the standard duration predictor removed and replaced by a stochastic duration predictor. The model contains 55.3 million trainable parameters.
- *GlowTTS-STDP*: a Glow-TTS model with a stochastic duration predictor and an additional stochastic pitch predictor. The model contains 56.0 million trainable parameters.

The three models were trained on the TTS dataset, and objective and subjective evaluations were conducted to evaluate their performance.

4.2.1 Dataset preparation

TTS dataset

For this study, we continued with the high-quality TTS data we curated in Chapter 3, which we denoted as WV-MOS $\geq$ 4.0-all. The TTS dataset was curated from the Common

Voice dataset (English subset, version 7.0) as described in Chapter 3. We believe that the stochastic pitch predictor can benefit from learning a diverse distribution of fundamental frequency (F0) values for speakers in different contexts from the dataset. To briefly summarise how it was curated: we filtered the Common Voice English dataset by selecting only high-quality speakers based on speaker-level Mean Opinion Scores (MOS) estimated by a quality estimator. In this chapter, the curated 4,469-speaker, 230.75-hour dataset is used for model training, excluding 512 samples that were randomly selected from the dataset for model validation. The dataset is also resampled from 48 kHz or 32 kHz to 16 kHz, and silences and long pauses are removed using a voice activity detection tool.[12] Additionally, we use the VCTK dataset (Yamagishi et al., 2019) in evaluation.

F0 extraction

In the case of the GlowTTS-STDP model, we provided ground-truth pitch information to the TTS model in the form of frame-wise F0 values. The F0 values were extracted from the training and validation sets using the pYIN library (Mauch and Dixon, 2014). The pYIN library uses a modification of the YIN algorithm (De Cheveigné and Kawahara, 2002) for F0 estimation and tracking in speech and music. For every frame, several fundamental frequency candidates and their probabilities are obtained. Then, a second step was applied to estimate the most likely F0 sequence and voicing flags using Viterbi decoding. Additionally, we applied the logarithm function to the F0 values during training and we also set all non-voiced regions in the log-F0 values to zero. The window size and hop size of the algorithm were set to 64 ms and 16 ms to match the parameters for the Mel-spectrogram computation. This also aids the correct alignment of the extracted F0 values with the input embeddings during model training.

Speaker representation

For speaker representation in the TTS model, we precomputed a speaker embedding for each utterance using the Resemblyzer speaker extractor model[13] trained on Voxceleb (Nagrani et al., 2017). Resemblyzer summarises the speaker information in a given audio into an L2-normalised, 256-dimensional vector.

Text representation

Text representations in TTS are typically in the form of tokenised text. We refer readers to Section 2.2.1 where we discussed the process of text tokenisation. Our TTS models use phonemes and characters as tokenised text input, albeit in a mixed approach (Kastner et al., 2019). In this mixed approach, some phonemes are replaced by characters with a probability *phon_prob* during training to improve the robustness of the TTS model to uncommon words that do not have standard phoneme mappings in the dictionary. The following text processing steps were also applied to the text:

- Text normalisation: the texts were normalised using the NeMo text normalisation toolkit (Zhang et al., 2021c), with numbers and dates converted to their written forms.

[12] https://github.com/wiseman/py-webrtcvad
[13] https://github.com/resemble-ai/Resemblyzer

- Phonemisation of words: we used the CMU pronunciation dictionary to extract phonemes for the texts. The dictionary contains a set of 39 phonemes and we refer readers to the CMU pronunciation dictionary website for more details.[14] We also provide characters directly to the model for unknown words or for randomly selected words during training.

- Lexical stress removal: although vowels in the dictionary carry a lexical stress marker, these stress markers were excluded in our case since they were developed for North American English, while several speakers in the Common Voice dataset do not speak American English or have a North American English accent.

- Addition of blank tokens: the tokenised text inputs were interspersed with a blank token as was recommended for the classical Glow-TTS model.

At each training iteration, the model was provided with a batch of the following: a text representation (phonemes and characters), a Mel-spectrogram extracted from the corresponding speech, the per-frame F0 values extracted (in the case of GlowTTS-STDP), and the speaker embedding. At inference time, the model was only provided with the phoneme representation (or character for unknown words) and the speaker embeddings to be used to generate the utterance. The model generates the F0 values used for conditioning the GlowTTS-STDP model and then converts the latent representation into a Mel-spectrogram.

4.2.2 Training and inference hyperparameters

Table 4.1 lists the model, training, dataset, and inference hyperparameters chosen for the TTS models. For model hyperparameters, we used the hyperparameters of the original Glow-TTS where possible and only introduced new ones for the newly introduced layers in GlowTTS-STD and GlowTTS-STDP. We indicate newly introduced and new values of hyperparameters in bold in the table.

All TTS models were trained in automatic mixed-precision mode using a global batch size of 192 on 4 Nvidia RTX GPUs, i.e., a batch of 24 on each GPU with a gradient accumulation step of 2. Optimisation was done using the RAdam optimiser (Liu et al., 2020) with a learning rate of 0.001, and a cosine-annealing scheduler with a linear warm-up of 6,000 steps. It took approximately 4 days for the models with stochastic pitch prediction and stochastic duration prediction to completely train while the baseline Glow-TTS was trained for 2 days. Each model was set to train for a maximum of 1,000 epochs, however the best model was selected during the training run using an early stopping criterion of 10 epochs on the validation loss.

At inference time, the following parameters are also provided to the TTS model to generate Mel-spectrograms:

- the noise temperature of the prior distribution. This was set to 0.667 for all models as in Glow-TTS (Kim et al., 2020),

- the noise temperature for the pitch predictor. This was set to 0.8, similar to VITS (Kim et al., 2021),

[14] http://www.speech.cs.cmu.edu/cgi-bin/cmudict

Table 4.1: Hyperparameters for TTS models.

Model hyperparameters	
α_1	0.5
α_2	0.5
No. of decoder blocks N	12
No. of duration predictor flows N_d	**4**
No. of pitch predictor flows N_p	**4**
Input embedding dimension	192
No. of encoder layers	6
No. of attention heads	2
Squeeze ratio	2
Training hyperparameters	
Mixed precision	True
Early stopping patience	10 epochs
Monitor for early stopping	Validation loss
Gradient clipping value	5.0
Accumulate gradient batches	2
Maximum number of training epochs	**1,000**
Number of GPUs	4
Warmup steps	6,000
Optimizer	RAdam
Learning rate	$\mathbf{2 * 10^{-4}}$
Weight decay	0.0
Learning rate scheduler	Cosine annealing
Dataset hyperparameters	
Batch size	**24**
Sample rate	16,000
Maximum frequency	8,000
Minimum frequency	0
Speaker embedding dimension	256
Number of Mel bins	80
Phoneme probability *phon_prob*	**0.8**
Inference hyperparameters	
Noise scale	0.667
Length scale	1.0
Stochastic pitch noise scale	**0.8**
Stochastic duration noise scale	**0.8/1.0 (for reading style)**
Vocoder sample rate	16,000

- the noise temperature for the duration predictor. This was also set to 0.8 for speaking style evaluation and it was set to 1.0 for reading style evaluation (see Section 4.2.3).

The noise temperatures control the weight of the additive Gaussian noise added to the latent representations in each flow-based model.

Then, a 16 kHz HiFi-GAN V1 vocoder (Kong et al., 2020) trained on the LibriTTS dataset (Zen et al., 2019) is used to convert the generated Mel-spectrograms into audio. The vocoder was not finetuned on Mel-spectrograms generated by the models. We im-

plemented and trained the TTS models using the neural modules in the NeMo toolkit (Oleksii Kuchaiev et al., 2019).[15]

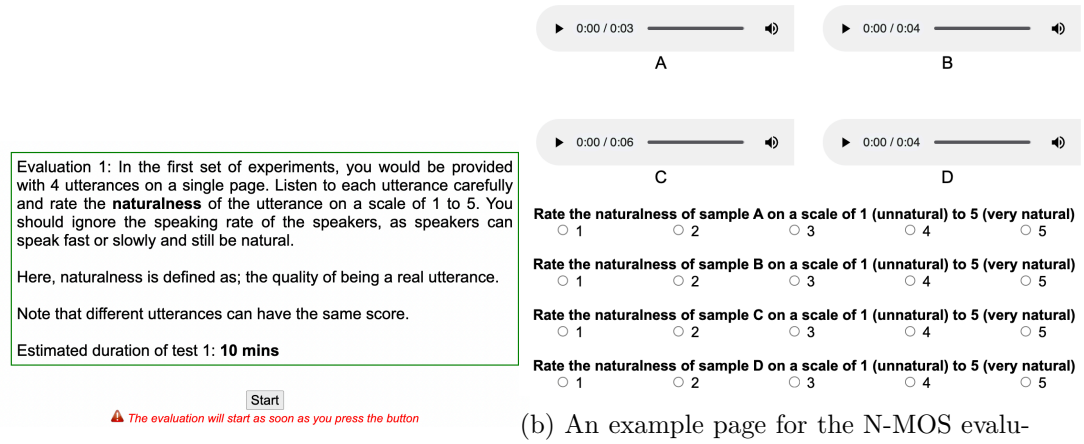
4.2.3 Model evaluation

After training the TTS models, we compared utterances generated by the classical Glow-TTS model baseline to the utterances generated by the two other models, GlowTTS-STD and GlowTTS-STDP using subjective and objective evaluation metrics. We describe the subjective and objective metrics below.

Subjective evaluation

Our subjective evaluation focused on evaluating the utterances generated by the TTS models for unseen speakers. Therefore, we selected speakers from the VCTK dataset (Yamagishi et al., 2019) for TTS evaluation. In particular, we arbitrarily selected three male speakers (p245, p254, and p263) and three female speakers (p228, p231, and p250). For each trained TTS model and speaker, we evaluated the following subjective metrics through listening tests.

- **N-MOS**: this measures the **naturalness** of the utterances generated by each model in **speaking** style. For this metric, the utterances are usually short, about 3–4 seconds long. We selected three short text sentences from VCTK for this evaluation. This allows us to generate 18 short sentences for each TTS model to be evaluated. Figure 4.9 shows the exact instructions provided to the evaluators for the N-MOS evaluation and the interface that was used to rate the utterances. The evaluators were asked to rate how the naturalness of the samples on a 1–5 Likert scale with 1.0 increments where naturalness was defined as the quality of being a real utterance.



(a) Instruction for N-MOS evaluation.

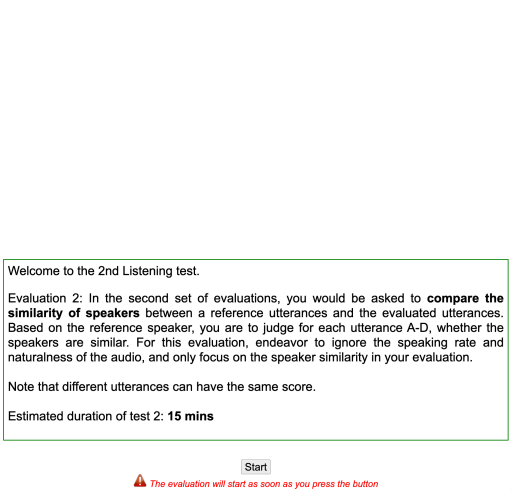
(b) An example page for the N-MOS evaluation.

Figure 4.9: Speaking-style MOS (N-MOS) evaluation instructions for the listening test.

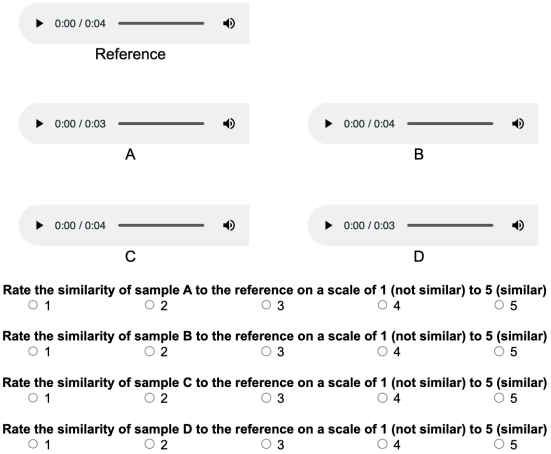
- **S-MOS**: this metric subjectively measures the speaker similarity of the generated utterances to a reference speaker's utterance. For this, three utterances that were recorded by each of the selected speakers in the VCTK dataset were used as the

[15]Training and inference code can be found at https://github.com/ogunlao/glowtts_stdp

reference utterances. The VCTK reference utterances were re-synthesised using the pre-trained vocoder (VCTK-copy) and were added to the subjective evaluations to determine the upper bound on the scores. Figure 4.10 shows the exact instructions provided to the evaluators for the S-MOS evaluation and the interface that was used to rate the utterances.



(a) Instruction for S-MOS evaluation.



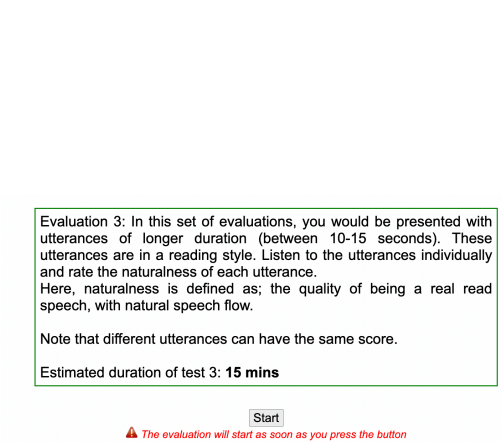
(b) An example page for the S-MOS evaluation.

Figure 4.10: Speaker similarity MOS (S-MOS) evaluation instructions for the listening test.

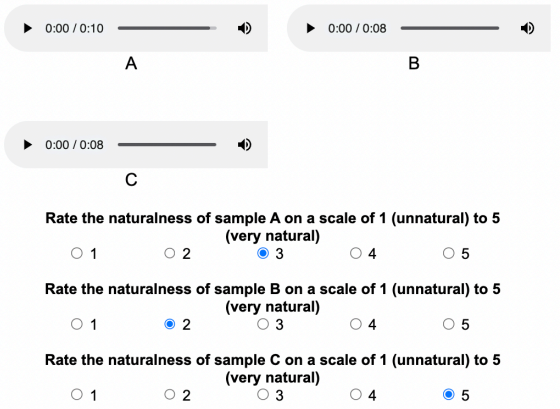
- **NR-MOS**: this subjectively measures the **naturalness** of the utterances generated in **reading** style. The utterances are longer (about 8–13 seconds long) and may contain punctuations. We selected three long texts with punctuation from LibriTTS for this evaluation. Recall that LibriTTS contains audiobooks and its text transcripts are more suitable for reading style evaluation. In addition, we generated 18 long utterances per model using the speakers selected from the VCTK corpus. Figure 4.11 shows the exact instructions provided to the evaluators for the S-MOS evaluation and the interface that was used to rate the utterances.

- **D-MOS**: it measures the **diversity** of generated utterances. Diversity here refers to the ability of the utterances to change the flow of intonation and pitch given the same input text. Therefore, in this thesis, we proposed to evaluate the diversity of the utterances using listening tests. To accomplish this, three utterances were generated for each speaker from the same text prompt and concatenated for the listening test. The evaluators were then asked to rate the diversity of the utterances in terms of the intonation (flow of pitch) of the speaker judging from the three concatenated utterances.

Following the evaluation methodology of SC-GlowTTS (Casanova et al., 2021), speaker embeddings in all of our evaluations were extracted from the fifth utterance (SpeakerID_005) of the VCTK speaker. The utterances are made with a long sentence containing 20 words and all the test speakers also uttered it. In addition, mean opinion score evaluations were performed on a web platform that loads the evaluations in the browser

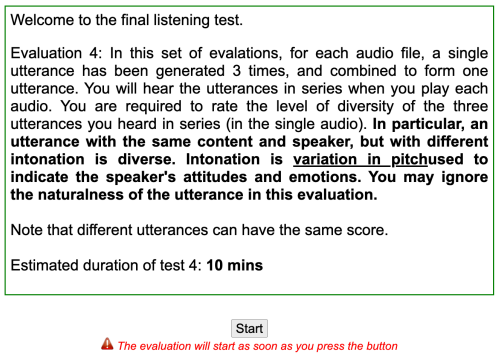


(a) Instruction for NR-MOS evaluation.

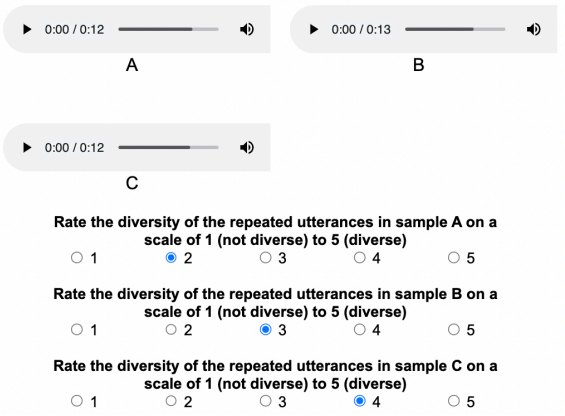


(b) An example page for the NR-MOS evaluation.

Figure 4.11: Reading-style MOS (NR-MOS) evaluation instructions for the listening test.



(a) Instruction for D-MOS evaluation.



(b) An example page for the D-MOS evaluation.

Figure 4.12: Diversity MOS (D-MOS) evaluation instructions for the listening test.

where the presentation of utterances had been randomised. The web platform was developed by our research team and it is dedicated to such evaluation. In total, 26 volunteers participated in the evaluation.

Objective evaluation

Objective evaluations were carried out similarly to subjective evaluations. For this evaluation, we selected 10 utterances from each of 10 male speakers and 10 female speakers in the VCTK dataset as unseen speakers. Then, we generated TTS utterances for each of the selected VCTK speakers and evaluated the following objective metrics on the datasets.

- **WV-MOS**: this objective metric measures the overall quality of the utterances generated by the TTS models. This is a model-based metric that estimates the MOS score that a human evaluator would have given to the utterance. We used the WV-MOS prediction model developed by Andreev et al. (2023) for predicting

WV-MOS values in the range of 1.0 to 5.0. We refer readers to Section 3.2.1 of this thesis for a more detailed description of the WV-MOS model.

- **cos-sim**: this is the cosine similarity between the generated utterances and the reference VCTK speaker embedding. The cosine similarity is typically used as an objective metric to evaluate the speaker similarities of the generated utterance and the reference speaker's utterance. We refer readers to Equation (2.4) on how to compute the cos-sim.

- **log-F0 distribution**: Here, we visually compared the distribution of log-F0 values of the generated utterances for males and females to the reference VCTK audio samples. We extracted F0 values for the VCTK test set and the generated utterances using pYIN (Mauch and Dixon, 2014), a fundamental frequency estimator. Then, we selected the voiced regions in the estimated F0 and applied the natural logarithm to the F0 to compute the log-F0 values. Afterward, we create a male and female kernel density estimate plot of the F0 values for each of the TTS models and the VCTK test set.

4.3 Results

Table 4.2 shows the results of subjective and objective evaluations. The best models have been selected (and highlighted) in each row by computing the 95 % confidence intervals (CI) on the difference between a pair of models. We selected the TTS model with the highest result in each row and computed the difference between its predictions and the predictions of other TTS models in the same row. We highlight the TTS model and the models statistically equivalent to it. The difference is statistically significant if the confidence interval lies fully on the positive side of the real axis and the TTS models are statistically equivalent otherwise.

In addition, the VCTK-copy served as a top-line during the listening test. In VCTK-copy, the real VCTK utterance is converted into its Mel-spectrogram and then converted back into speech using the trained vocoder.

Table 4.2: N-MOS, NR-MOS, S-MOS of utterances generated for 6 unseen speakers by the baseline Glow-TTS, GlowTTS-STD, GlowTTS-STDP, and VCTK-copy. Bold numbers denote the best system in each row and the systems statistically equivalent to it.

		Baseline	GlowTTS-STD	GlowTTS-STDP	VCTK-copy
N-MOS	Male	2.92	3.11	**3.40**	4.00
	Female	3.35	**3.51**	**3.51**	4.40
	Total	3.13	3.31	**3.45**	4.21
WV-MOS		4.11	**4.17**	**4.18**	4.32
NR-MOS	Male	2.91	**3.24**	**3.25**	-
	Female	2.98	3.28	**3.53**	-
	Total	2.95	3.26	**3.39**	-
S-MOS	Male	2.43	2.90	**3.10**	4.71
	Female	2.14	**3.07**	2.91	4.85
	Total	2.26	**2.98**	**2.99**	4.79
cos-sim		0.8287	**0.8364**	**0.8319**	0.8121

4.3.1 Naturalness and quality of utterances improved by stochastic duration and pitch prediction

Improved naturalness for speaking-style utterances

We found a significant improvement in the naturalness (N-MOS) of speaking utterances generated by GlowTTS-STD and GlowTTS-STDP over the baseline model in Table 4.2. This shows that the stochastic duration predictor more accurately learns the speaking style of speakers than the baseline duration predictor. In addition, stochastic pitch prediction does not degrade the naturalness of utterances, and in particular, improves or maintains the naturalness of utterances. For unseen male speakers, we noticed a significant improvement of 0.29 MOS points over GlowTTS-STD.

Improved naturalness for reading-style utterances

For longer sentences, the utterances generated by GlowTTS-STD and GlowTTS-STDP were rated higher than the baseline in general as measured by NR-MOS. Here, utterances generated by GlowTTS-STDP for female speakers were rated higher than utterances generated by GlowTTS-STD, but there was no significant difference in the ratings for male speakers. We observed that short utterances were rated higher than longer utterances in many cases. Nevertheless, GlowTTS-STDP still showed significant improvement in reading-style naturalness over the baseline.

Increased quality of utterances

In Table 4.2, the utterances generated by GlowTTS-STD and GlowTTS-STDP had higher overall quality (WV-MOS) than the baseline. Nevertheless, we did not find any statistically significant difference in the quality of the utterances generated by GlowTTS-STD and GlowTTS-STDP. However, there are still some observations in the result. First, the result indicates that the GlowTTS-STDP model was able to integrate pitch information correctly in the decoder without degrading the quality of the generated utterance. Second, it also shows that modelling the rhythm of the speech has a significant impact on speech quality as shown by the WV-MOS scores given to GlowTTS-STD. In general, the ratings were still however lower than the ratings given to the VCTK utterances with copy synthesis.

4.3.2 Higher speaker similarity

The utterances generated by models using a stochastic duration predictor (GlowTTS-STD and GlowTTS-STDP) exhibit significantly higher speaker similarity (S-MOS) than the baseline, especially for female speakers. In addition, we see two opposite trends between male and female S-MOS scores in GlowTTS-STD and GlowTTS-STDP models. For male speakers, the utterances generated by GlowTTS-STDP were rated higher in terms of S-MOS, indicating that the male speakers benefit more from pitch prediction in the model. This may indicate that the GlowTTS-STDP model was able to better capture the pitch of males than females. Conversely, the utterances generated by GlowTTS-STD were rated significantly higher in terms of S-MOS for female speakers, with a slight drop of 0.16 S-MOS points for GlowTTS-STDP. We believe that the rhythm of speech, controlled by the stochastic duration predictor, significantly impacts the subjective speaker similarity ratings given by volunteers. Similarly, the cos-sim values are higher for GlowTTS-STD and GlowTTS-STDP relative to the baseline's cos-sim value, however this is not as significant

as the S-MOS scores. In addition, we observed a low cos-sim value for VCTK-copy. This cos-sim value was computed between the speaker embedding extracted from the real speech and the speaker embeddings extracted from the vocoded versions of the real test utterances. This may indicate that some speaker information has been lost during the copy-synthesis process.

4.3.3 Diversity of utterances

Improved diversity of utterances

In Figure 4.13, we show the distribution of D-MOS scores for each model and gender in the plot. Here, we indicate the mean and 95 % confidence interval for the scores. Note that the diversity of Glow-TTS is not zero because it is a generative model and is also capable of generating diverse utterances. As such, we compare our architectural changes against the baseline. We observed an increasing trend in the diversity of utterances (D-MOS) from the baseline to GlowTTS-STDP, with GlowTTS-STDP being able to generate more diverse utterances than the other models without loss of naturalness. This indicates that the stochastic pitch prediction further aids in increasing the diversity of the utterances, in addition to the improved rhythm provided by the stochastic duration predictor. In general, female utterances were rated higher in diversity than male ones with more than 0.1 MOS points between the two genders.

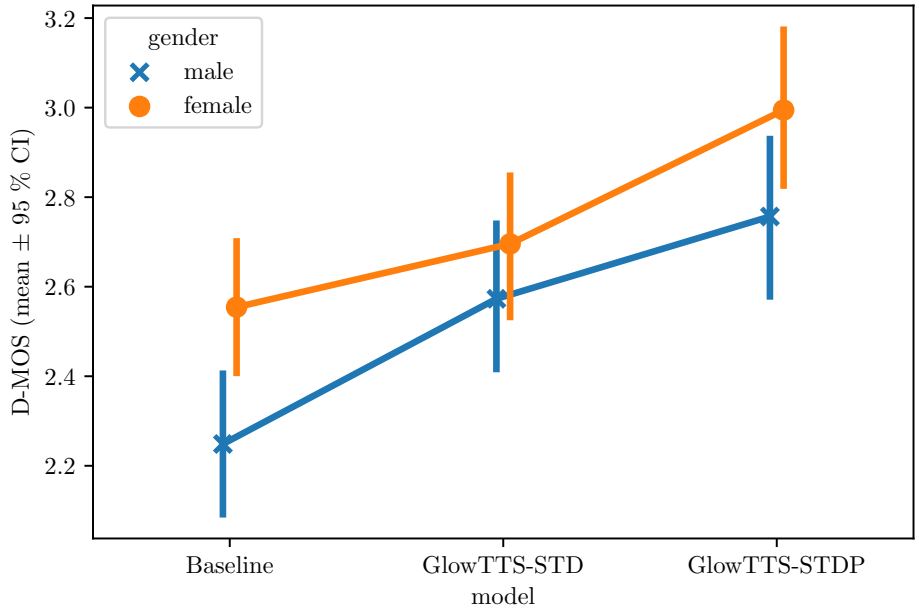


Figure 4.13: D-MOS of utterances generated by the evaluated models for both male and female unseen speakers.

Distribution of log-F0 values in GlowTTS-STDP is closer to real data

Figure 4.14 shows the distribution of log-F0 values computed for the generated utterances. Here, we show the distribution for the male and female genders on separate plots. In ad-

dition, we have removed outlier F0 values from the plots. These outlier values are mostly due to F0 estimation errors and do not contribute extra information to the understanding of the plot. Here, we observed that the distribution of log-F0 values for utterances generated by GlowTTS-STDP better matches those of real speech for both male and female speakers. Having the same distribution of log-F0 values in the real data and GlowTTS-STDP shows that the log-F0 predictions by the stochastic pitch predictor are realistic. In contrast, we observed that the log-F0 values are more spread-out compared to the baseline Glow-TTS and GlowTTS-STD which are both narrower. This seems normal because both TTS models do not explicitly generate pitch information. Therefore, they have less variability in predicting log-F0 values compared to GlowTTS-STDP which has an explicit pitch model for generating pitch values.

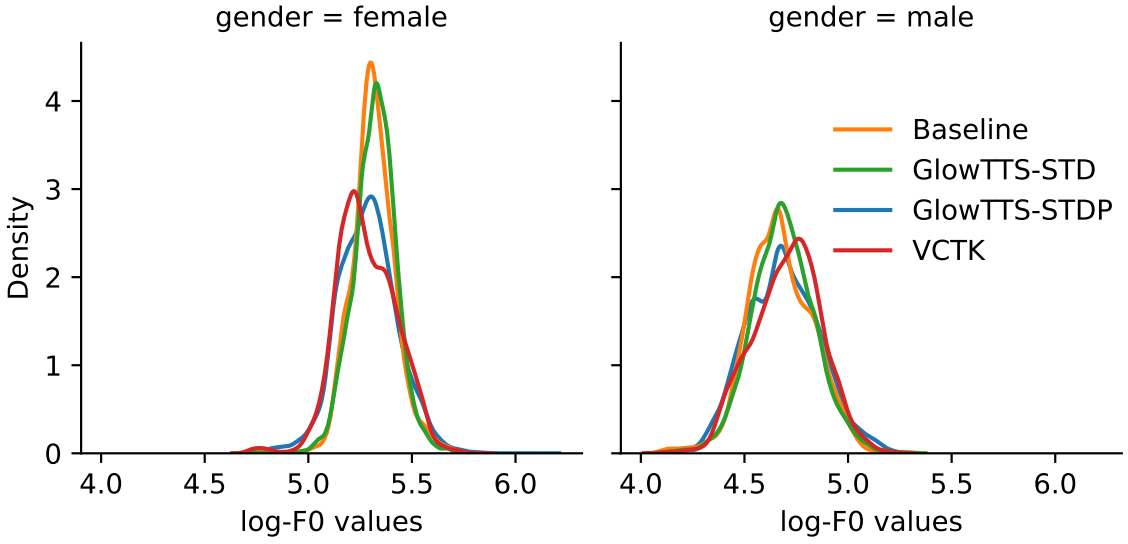


Figure 4.14: Log-F0 distribution of utterances generated with GlowTTS, GlowTTS-STD, GlowTTS-STDP, and the real VCTK utterances for both male and female speakers. Outliers due to F0 estimation errors have been removed.

In summary, the increased diversity of utterances in GlowTTS-STD while being realistic offers a means to generate more diverse data that can increase the coverage of speech distribution, without a reduction in naturalness. It will be interesting to determine if this diversity is also significant for ASR data augmentation, which usually uses augmentation techniques that are not so realistic, e.g., increasing the global pitch or duration of the utterances.

4.4 Conclusion

In this chapter, the naturalness of utterances was successfully improved using a stochastic duration predictor and stochastic pitch predictor. Similarly, we significantly improved the naturalness of long, reading-form utterances as shown by the NR-MOS evaluation. Improving the naturalness of long utterances was important to determine the capacity of the TTS systems to generate more difficult and long utterances with punctuation, in which many TTS systems often fail to generate realistic utterances. In addition, we increased the

diversity of generated utterances by integrating a stochastic pitch predictor into Glow-TTS in addition to the stochastic duration predictor. We observed that the stochastic duration prediction alone is not enough to produce very diverse utterances from our evaluations and the stochastic pitch prediction increases the diversity of the utterances without a reduction in naturalness. Additionally, we showed that the distribution of log-F0 values of our model better matches the distribution of log-F0 values of real utterances. Therefore, the improvement enables us to perform procedural manipulation of the rhythm and the pitch of generated utterances, without loss in the naturalness of the generated utterances. In Chapter 5, we will determine if this diversity as determined by the listening test is beneficial for ASR data augmentation.

5

Evaluating the impact of synthetic data diversity on ASR performance

5.1 Introduction

In Chapter 3 of this thesis, we showed a method for curating a TTS dataset from an ASR dataset. We also showed that the dataset is of high quality and more importantly, contains diverse speakers. In Chapter 4 of this thesis, we have built TTS models that improve the naturalness and diversity of utterances that the TTS models can generate. In addition, we trained the TTS models using the TTS dataset that we have curated from the ASR corpus. In this chapter, we shift our focus to applying the TTS models to generate synthetic data for the augmentation of ASR training data.

Training data augmentation is a widely used method for improving the performance of automatic speech recognition (ASR) systems (Ko et al., 2015; Hu et al., 2022). Data augmentation, in general, allows the model to learn features that are invariant to small perturbations in the input data or to synthesise testing conditions that are not covered in the real training data. Popular data augmentation methods include time masking and frequency masking (Park et al., 2019), changing the speaking rate (Ko et al., 2015), noise addition (Kim et al., 2017), or synthetic data augmentation (Baskar et al., 2019; Tjandra et al., 2020; Hu et al., 2022). Synthetic data has gained popularity as a data augmentation method in recent years due to the rapid improvement in the overall quality and naturalness of synthetic speech. Reported results even show that synthetic speech can sometimes be indistinguishable from real speech (Kim et al., 2021; Tan et al., 2024).

Synthetic data can be defined as data created to simulate real conditions. This includes synthetic utterances generated via text-to-speech (TTS) or voice conversion (VC). Synthetic data augmentation then involves combining real and synthetic data for ASR training. The use of synthetic data for data augmentation has already been explored for different applications such as keyword detection (Hu et al., 2022), code-switching ASR (Sharma et al., 2020), low-resourced language data augmentation (Zevallos et al., 2022; Casanova et al., 2022; Shahnawazuddin et al., 2020; Baas and Kamper, 2022), language adaptation (Robinson et al., 2022), or improving numeric sequence recognition (Peyser et al., 2019). Although these works have proven that this augmentation method is useful in improving model performance in different domains, naively generating synthetic data

does not always yield the best results (Hu et al., 2022; Sharma et al., 2020). This is because of the distributional gap between real speech and synthetic speech (Minixhofer et al., 2023).

One key to closing the gap is to generate synthetic data with a similar distribution to the real speech data (Hu et al., 2022), for example synthetic utterances with diverse acoustic and linguistic characteristics similar to real speech. The characteristics of speech that make an ASR dataset diverse are manifold. Large phonetic coverage of the data is important for the model's robustness to out-of-vocabulary words, or to correctly recognise uncommon words (Wu et al., 2007). Also, a diverse representation of speakers in the dataset ensures that the ASR model is invariant to individual voice characteristics. Similarly, attributes such as speaking style, emotion, or stress result in variations in prosody (Kim et al., 2021). Finally, the environmental conditions, including the various types of noises and reverberation, contribute to blurring or masking speech features (Minixhofer et al., 2023). Accounting for the mentioned attributes can help the synthetic dataset to match the real data domain or even fill holes that are not captured in the real training data.

To the best of our knowledge, only the works in Rossenbach et al. (2023) and Minixhofer et al. (2023) have recently examined the impact of acoustic diversity of synthesised speech on the final ASR results, and no recent work has explored the impact of linguistic diversity. Therefore, in this work, we perform an experimental study to measure the performance gains that can be obtained from synthetic data augmentation by varying the speech attributes in the dataset. In particular, we independently and jointly evaluate the impact of the phonetic content, phoneme duration, pitch, number of speakers, and the environmental characteristics of the synthetic data for different data sizes on the ASR performance.

Recent multi-speaker TTS systems can vary the acoustic characteristics of generated speech when conditioned on a specific prosodic attribute (Ren et al., 2020; Łańcucki, 2021). In particular, generative model-based TTS models learn the distribution of the acoustic and linguistic characteristics of speech from training data, which gives them the intrinsic ability to generate diverse utterances while being able to control the prosodic attributes of the speech implicitly (Kim et al., 2020) or explicitly using the method described in this thesis. Therefore, for data generation, we leverage the flow-based TTS models described in Section 5.2 which are capable of generating diverse and natural utterances.

The rest of the chapter is organised as follows. Section 5.2 describes the TTS/VC and ASR models used in this work as well as our experimental design. We explain the experimental methodology and results in Section 5.3 and conclude in Section 5.4.

5.2 Experimental design

This section describes the datasets, the TTS/VC models, and the ASR model used in our experiments. The procedure for systematically generating the synthetic data is also discussed in detail.

5.2.1 Datasets

In this study, the Common Voice dataset (Ardila et al., 2020), which is a popular ASR dataset, was used for the ASR experiments and for training the TTS/VC models. Common Voice is a crowdsourced, read-speech dataset. We refer readers to Chapter 3.1 for a more

detailed description of the dataset. As in other chapters, we use version 7 of the validated English subset of the dataset.

ASR datasets like Common Voice contain utterances with varying ambient noise from several speakers. This is a good attribute for general ASR, however noise can degrade TTS. For TTS/VC training, we select a subset of the utterances using a quality criterion as described in detail in Chapter 3. ASR datasets are the kind of data that is commonly available in most practical scenarios and languages, therefore we demonstrate how a noisy, multi-speaker ASR dataset can be useful for training TTS/VC systems for ASR data augmentation. Concretely, this involves selecting good-quality audio recordings from the CV dataset by filtering speakers with high automatically estimated mean opinion scores. The curated TTS dataset contains 4,469 speakers with a total duration of 230.75 h.

For our experiments, we consider the situation where the ASR dataset available in the language or domain is very small. This is typically the case in childrens ASR applications or low-resourced languages. Hence, the size of the real ASR training data was limited to 50 h or 100 h which enables us to evaluate the performance of the methods in the low data regime. The ASR data also contains only speakers that were used to train the TTS system to efficiently evaluate the contribution of speaker attributes.

Data preprocessing for TTS training

We maintain the same TTS data preprocessing methods as have been described in Section 4.2.1. Here, we briefly describe the process.

- Audio resampling: The recordings in the TTS dataset are resampled from 32 kHz or 48 kHz to 16 kHz sampling rate.
- Silence removal: Beginning and end silences are removed using pydub[16] with a threshold of -50 decibels relative to full scale (dBFS).

The paired texts are also processed by performing text normalisation and text tokenisation.

- Text normalisation: Numbers and abbreviations are converted to their written forms. We keep the case of the letters and common punctuations like ",.'!". Text normalisation was performed using the NeMo text-processing toolkit (Oleksii Kuchaiev et al., 2019).
- Text tokenisation: During training the texts are converted into phonemes (with stress markers excluded) and characters in a mixed approach. With a probability of 0.2, a word in a sentence is tokenised into characters or replaced with its phonetic representation. This improves the robustness of the TTS model to words without phoneme mapping in the CMU pronunciation dictionary or to out-of-vocabulary words. The input tokens were also interspersed with a blank token (Kim et al., 2020)

Data preprocessing for ASR training

Real training data curation

For ASR training, we selected all the 2,457 speakers in the TTS/VC data which are the speakers that intersect with the speakers in the Common Voice training set as our real

[16] https://github.com/jiaaro/pydub

training data. We randomly selected a 50-hour or 100-hour subset of the data that covers all the 2,457 speakers for our experimental study. We convert all the recordings to 16 kHz wav files from mp3 files. The remaining speakers in the training set were used as candidate ASR unseen speakers in our experiments.

Synthetic training data curation

To create the TTS datasets, we generated synthetic utterances using the three TTS/VC models that will be briefly described in Section 5.2.2. In the experiments covering individual attributes, we either generated 50 hours or 100 hours of synthetic data using the TTS/VC models. When we combined several speech attributes in the final experiments, we increased the size of the data until we did not see any improvements. Voice conversion was also performed on the real training data for data augmentation. Depending on the experiment, the speakers used as output targets by TTS/VC can either be speakers in the real ASR data or new speakers. Table 5.1 summarises the data duration and the number of speakers used in this experimental study.

Table 5.1: Data duration and number of speakers used for training the ASR and TTS/VC systems. We continuously increase the number of speakers by a factor of 2 for speaker diversity experiments described in Section 5.3.2.

Models	Real/synthetic	Duration (h)	Speakers
TTS	real	230.75	4,469
ASR	real	50	2,457
ASR	real	100	2,457
ASR	synthetic	50	2,457
ASR	synthetic	100	2,457 (x2, x4, x8)

5.2.2 Speech synthesis models

To create synthetic data with different speech characteristics, we leveraged the different TTS/VC models that were trained in Chapter 4: the classical speaker-conditioned Glow-TTS (Casanova et al., 2021), and the two other flow-based TTS/VC models proposed in Chapter 4, GlowTTS-STD and GlowTTS-STDP. They have been extensively described in Section 4.1.

- Multi-speaker Glow-TTS: it is a speaker-conditioned, non-autoregressive, flow-based generative TTS/VC model. The model uses a Transformer encoder and a flow-based decoder, along with a deterministic phoneme duration prediction network.

- GlowTTS-STD: it is a Glow-TTS model enhanced with an improved flow-based duration predictor. The stochastic duration predictor enables the model to have a more human-like rhythm than the monotonic rhythm of the classical Glow-TTS model.

- GlowTTS-STDP: Similar to GlowTTS-STD, the GlowTTS-STDP model incorporates a stochastic duration predictor and a stochastic pitch predictor into the TTS system providing it with more controllability.

Using the TTS/VC models for synthetic speech generation

The process used for generating speech using the different TTS/VC models only differs slightly. At inference, text and speaker embeddings are provided to the models to generate the Mel-spectrogram. Depending on the model the phoneme durations and log-F0 (for GlowTTS-STDP) are generated using the flow-based predictors, and used by the TTS models to generate the Mel-spectrogram.

The VC process is rather simple for the Glow-TTS and GlowTTS-STD models. They invert the Mel-spectrogram into latent representations using the flow-based decoder, then convert these representations back into Mel-spectrograms while conditioning the representations on the new speaker's representation. This has been described in Section 2.3.2 of this thesis. New phoneme durations may be predicted for the latent distribution using the model's duration predictor. In addition, the GlowTTS-STDP model also conditions the representation on the log-F0 values predicted for the new speaker when inverting the representation to generate a Mel-spectrogram.

A HiFi-GAN V1 (Kong et al., 2020) vocoder is then used for converting the generated Mel-spectrograms to speech at a 16 kHz sampling rate, which is suitable for ASR training. We used the vocoder that had been trained on the LibriTTS dataset for speech generation. Additionally, a speaker extraction model (Wan et al., 2018) was used to extract speaker embeddings.[17] We averaged each speaker's utterance embeddings to compute a speaker representation for inference.

5.2.3 ASR model

ASR models take speech features as inputs and output characters or words. A state-of-the-art ASR architecture, the Conformer-Transducer (Gulati et al., 2020) described in Section 2.5.3, was used in this study. It is composed of a convolution-augmented Transformer (Conformer) encoder, a Long Short Term Memory (LSTM) decoder, and a fully connected layer as a joint network. The Conformer encoder sandwiches convolutional layers between self-attention blocks. The convolutional layers better capture local information while the self-attention layers help to capture the global information from the hidden features. The encoder consists of 16 layers of Conformer blocks with 4 attention heads and a hidden dimension of 256. A 1-layer LSTM is used as the main decoder with the recurrent neural network transducer (RNN-T) loss (Graves, 2012). Following standard practice, we use an auxiliary Connectionist Temporal Classification (CTC) loss (Graves et al., 2006) obtained through an extra convolution decoder on top of the encoder as shown in Figure 5.1. The model contains 31.2 million trainable parameters. Hence, we combine the two losses to derive the ASR training loss $\mathcal{L}_{\text{ASR}}$ as

$$\mathcal{L}_{\text{ASR}} = \alpha\,\mathcal{L}_{\text{RNN-T}} + (1-\alpha)\,\mathcal{L}_{\text{CTC}}, \tag{5.1}$$

where $\mathcal{L}_{\text{RNN-T}}$ is the main loss from the transducer and $\mathcal{L}_{\text{CTC}}$ is the auxiliary loss from the CTC branch.

5.2.4 Training and inference hyperparameters

The ASR models and TTS models[18] were trained using the NeMo toolkit (Oleksii Kuchaiev et al., 2019). The ASR models use 80-dimensional log-Mel spectrograms as inputs and

[17] https://github.com/resemble-ai/Resemblyzer

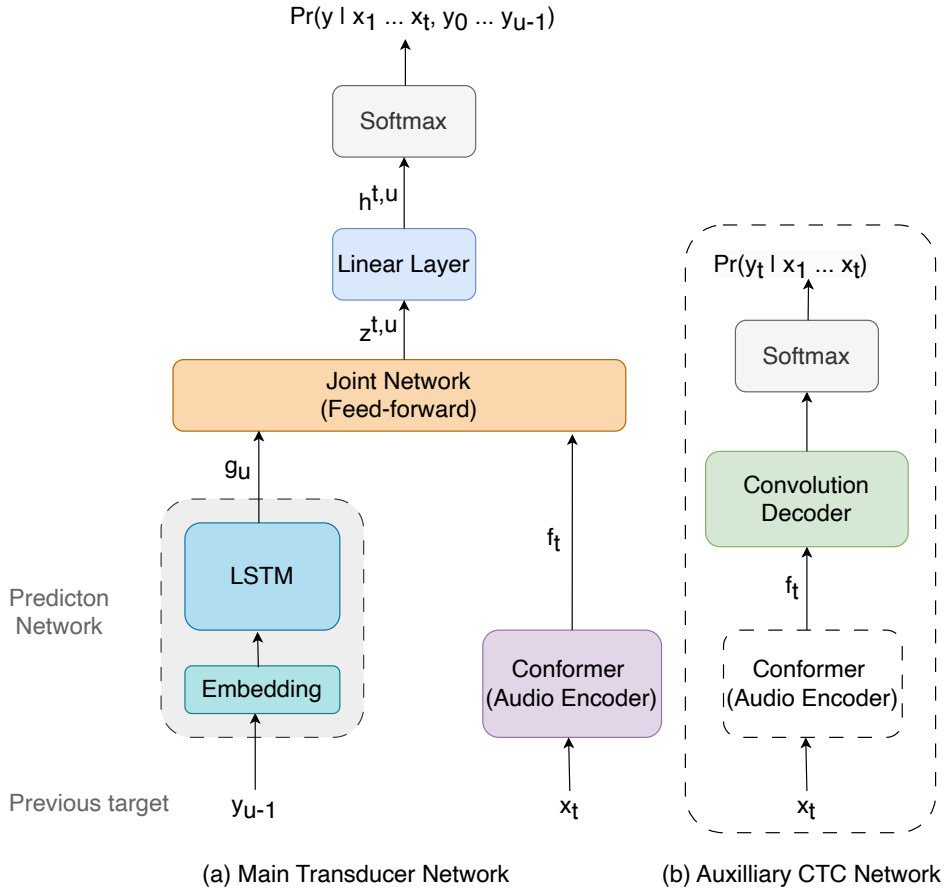


Figure 5.1: The Conformer-Transducer with an auxiliary CTC network added to the encoder path.

output characters in the set of the 26 English alphabetical characters, space, and apostrophe. Each ASR model was trained for 500 epochs with early stopping on the validation set after 50 epochs. The AdamW optimiser and the NoamAnnealing scheduler with a linear warm-up of 6,000 iterations were used for gradient optimisation. SpecAugment (Park et al., 2019) was also applied on the input features. We report the word error rate (WER) and character error rate (CER) on the Common Voice test set for each ASR model. The α value for combining the ASR losses was set to 0.7. No external language model was used for decoding.

To compare the performance of the ASR models, we performed statistical tests using the Matched Pairs Sentence-Segment Word Error (MAPSSWE) Test (Gillick and Cox, 1989) implemented in NIST's scoring toolkit. It is a standard parametric test that looks at the number of errors in segments of utterances of varying size that are specific to the output of the two ASR systems being compared. Bold values in the tables indicate the best system (for each combination of real and synthetic data durations) and those systems that are statistically equivalent to it at a 95% confidence level.

For TTS inference, the noise temperature of the prior distribution was set to 0.667 for all the models. The noise temperature for the pitch predictor and duration predictor

[18]Code for training TTS models can be found at https://github.com/ogunlao/glowtts_stdp

of GlowTTS-STD and GlowTTS-STDP was set to 0.8. The noise temperatures control the weight of the additive Gaussian noise added to the latent representations in each flow-based model.

5.3 Experimental methodology and results

The experiments were divided into subsections covering each of the factors we plan to evaluate and improve upon. The first subsection focuses on the linguistic content of speech, while the following subsections evaluate the impact of the speaker or prosodic characteristics, i.e., speakers, pitch, and phoneme duration. Then, we evaluate the impact of adding noise to the synthetic utterances. In the last subsection, we combine all the factors to create a diverse synthetic dataset for ASR training data augmentation. Each subsection describes the methodology used to generate the synthetic dataset and the corresponding ASR results.

For the ASR training data, we use the notation Ar_Btts or Ar_Bvc to denote that A hours of real speech and B hours of TTS/VC data were used in the specific experiment, e.g., 50r_50tts means that 50 hours of real ASR data and 50 hours of TTS data were combined for ASR training.

5.3.1 Increasing phonetic diversity

For languages or domains where a text corpus is more readily available than real speech data with corresponding texts, it may be useful to generate synthetic data to increase the phonetic content of the real ASR training data using the available texts. In our experiments, we applied an iterative greedy method to select sentences that we used to generate TTS data. The method is described below.

Methodology

Let $X = \{x_1, x_2, x_3, ..., x_n\}$ be a large set containing n sentences that covers the natural distribution of text in a specific language or domain. If a subset $Y \subset X$ of sentences (and the corresponding real utterances) is available as real training data, then the remaining texts in $Z = X \setminus Y$ are candidate sentences for generating synthetic utterances. Our goal is to select a subset of sentences $K \subset Z$ given a constraint on the size of the TTS data (e.g., a duration of T_s hours) that minimises the Kullback-Leibler (KL) divergence between the distribution of the combined real and synthetic data $Y \cup K$ and a desired distribution. In practice, we consider the distribution of pairs of adjacent phonemes, called di-phonemes, which account for the left or right context of every phoneme. Although it is also possible to use the distribution of tri-phonemes or word units, when Z is very large, the choice of di-phonemes increases the ratio of common to uncommon units.

Concretely, given a dataset D with all its sentences split into di-phonemes, for any di-phoneme $d_i \in D$, we compute its probability $P_D(d_i)$ in dataset D.

$$P_D(d_i) = \frac{\text{count of di-phoneme } d_i \text{ in } D}{\text{total no. of di-phonemes in } D}. \tag{5.2}$$

For discrete distributions and given an arbitrary set X, the KL divergence is defined as:

$$\text{KL}(P||Q) = \sum_{c \in \text{C}} P(c) \left(\log P(c) - \log Q(c)\right), \tag{5.3}$$

where $P(c)$ is the empirical distribution and $Q(c)$ is the model distribution. Usually, $P(c)$ is a fixed dataset and $Q(c)$ is a model whose parameters we want to adapt. In our case, it's the opposite: $Q(c)$ is a fixed desired distribution and we adapt the dataset distribution $P(c)$ to best fit it. Therefore, the KL divergence between the distribution of di-phonemes in $Y \cup K$ and the desired distribution Q becomes

$$\mathrm{KL}(P_{Y\cup K}||Q) = \sum_{d_i \in \mathrm{diph}} P_{Y\cup K}(d_i)(\log P_{Y\cup K}(d_i) - \log Q(d_i)). \tag{5.4}$$

In the following, we consider two choices for the desired distribution $Q(d_i)$: the *natural* distribution $P_X(d_i)$ of di-phonemes in X and *uniform* distribution, that is, the distribution with maximum entropy recommended by Wu et al. (2007).

We iteratively minimise the divergence by selecting utterances one at a time using Algorithm 1. The set of selected sentences K is used to synthesise speech, and the ASR system is then trained on the dataset corresponding to $Y \cup K$.

Algorithm 1 Algorithm for sentence selection.

Require: T_s, Z, Q
procedure SELECTSENTENCE(T_s, X, Y)
  Initialise $K = \emptyset,\ T_\mathrm{c} = 0$ ▷ T_c is the total duration of current data selected
  while $T_\mathrm{c} < T_\mathrm{s}$ **do**
    for each sentence z in Z **do**
      Compute $\mathrm{KL}_z := \mathrm{KL}\left(P_{Y\cup K\cup z}||Q\right)$
    end for
    $z_\mathrm{s} := z$ with minimum KL_z
    $Z \leftarrow Z \setminus \{z_\mathrm{s}\},\ K \leftarrow K \cup \{z_\mathrm{s}\}$
    $T_\mathrm{c} := T_\mathrm{c}$ + duration of clip corresponding to z_s
  end while
  return K
end procedure

The set of sentences in the Common Voice training set was used as X and di-phonemes were computed by first extracting the sequence of phonemes corresponding to the text using a grapheme-to-phoneme converter,[19] and then consider pairs of adjacent phonemes. In addition, we use the duration of real utterances corresponding to the text during iteration as that would not require us to generate TTS utterances when running the algorithm. Lastly, we optimise the algorithm to run on a 40-core CPU which takes about two days to select text equivalent to 50 hours of speech.

Results

In Table 5.2, we provide a baseline for ASR models trained on 50 hours and 100 hours of real data. The CER and WER of the baselines are higher than typical values reported for these data sizes due to the data selection method which restricts the selection of speakers to only speakers that intersect with the TTS training data. As such the ASR training dataset is "high-quality" but then makes it different from the significantly noisier test set.

[19] https://github.com/Kyubyong/g2p

We compare the proposed sentence selection method using the natural distribution of X or the uniform distribution as a target to *random* selection of sentences from set Z. For 50 hours of TTS data, the sentence selection algorithm selects 24,879, 28,664, and 24,879 sentences for natural, uniform, and random selection methods respectively. Table 5.2 shows the results for ASR models trained on utterances generated from 50 hours or 100 hours of selected sentences combined with 50 hours or 100 hours of real speech. The classical speaker-conditioned Glow-TTS model was used to generate the synthetic utterances for the 2,457 speakers in the real data. We cover all the speakers in the set during synthesis by continuously iterating through the set of 2,457 ASR speakers until the size of the TTS dataset has been reached.

Table 5.2: CER and WER of ASR models trained on real speech and TTS speech generated from sentences selected randomly or using the uniform or natural distribution criterion. Bold numbers denote the best selection method and the systems statistically equivalent to it for each amount of data.

Selection method	Data	% CER	% WER
-	50r	25.01	46.30
random	50r_50tts	23.28	42.79
uniform		22.56	42.13
natural		**22.24**	**41.52**
random	50r_100tts	22.23	40.98
uniform		22.83	41.92
natural		**21.91**	**39.89**
-	100r	22.10	40.81
random	100r_50tts	20.50	38.25
uniform		20.11	37.48
natural		**20.00**	**37.11**
random	100r_100tts	20.21	37.09
uniform		**20.00**	36.92
natural		**20.07**	**36.66**

In Table 5.2, by first considering the case when the sentences are randomly selected (random), it can be observed that augmenting 50 hours of real data with 50 hours or 100 hours of TTS data reduces the WER by 8% and 11% relative compared to training on real data only. Similarly, augmenting 100 hours of real data with 50 hours or 100 hours of TTS data reduces the WER by 6% and 9% relative. Similar reductions are achieved in terms of the CER. This shows that TTS based data augmentation can be effective for both small-sized and medium-sized data with greater performance gains obtained when starting from a smaller real dataset or when augmenting with more synthetic data.

Looking at the results for the different sentence selection methods, we observe a statistically significant relative improvement in CER and WER using the natural selection method over other selection methods. This is especially visible when the number of sentences selected is small, i.e., 50 hours. The CER and WER then drop by 4% and 3% relative to random selection. By contrast, the uniform selection method does not yield significantly better results than random selection. As Wu et al. (2007) used a combination of word-, character-, and phoneme-level distributions in their experiments, it is difficult to directly compare their increase in ASR performance using a uniform distribution to ours.

Also, an HMM-GMM-based model was used in their experiments which does not have the same behaviour as deep-learning-based models.

In conclusion, although adding synthetic data to the training dataset can reduce the CER and WER of the ASR model, considering the distribution of the text when selecting the sentences for generating the training utterances will further improve its performance.

To better understand the effect of the different sentence selection methods, we plotted in Figure 5.2 the KL divergence between the distribution of di-phonemes in the set of Common Voice training sentences and the distribution of di-phonemes in the combined set of real sentences and the subsequently added synthetic sentences for each of the selection methods considered compared to random selection. From Figure 5.2a, it can be observed that the natural selection method produces the lowest divergence from the Common Voice data. Although the random selection method reduces the KL divergence, one can extrapolate from Figure 5.2a that it will take twice more data to achieve a similar KL divergence. Similarly, using the uniform distribution reduces the KL divergence to some extent in Figure 5.2b but seems to saturate quickly without any further reduction. Additionally, the KL divergence becomes flat when the distribution of randomly selected text is compared to the uniform distribution indicating that the randomly selected data does not closely follow a uniform distribution of di-phonemes.

5.3.2 Increasing speaker diversity

The natural way of acquiring more speakers when there are few speakers in a dataset is to collect more data with more speakers. However, since TTS/VC models are capable of generating data when conditioned on arbitrary speakers, this approach may be used to increase the diversity of speakers for ASR data augmentation.

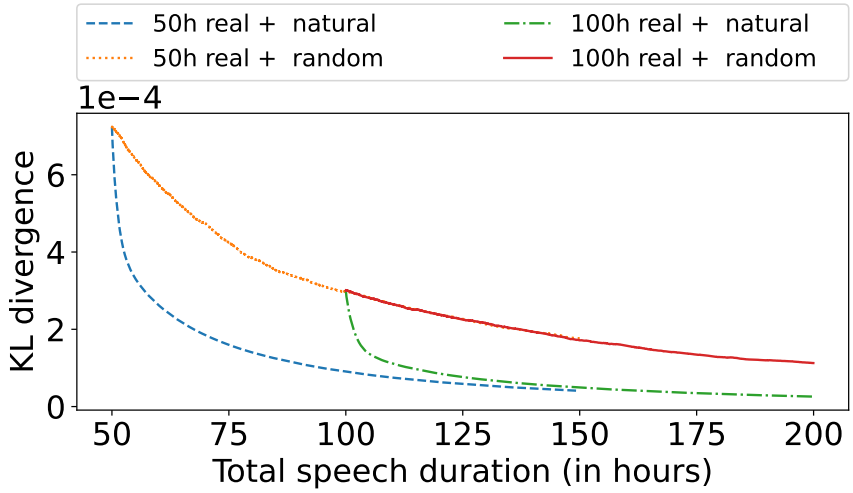
Methodology

Given the set of speakers $S = \{s_1, s_2, s_3, ..., s_n\}$ from the real ASR training dataset and a set of T unseen speakers, where $S \cap T = \emptyset$, the goal is to select the most informative speakers $R \subset T$ that will improve the ASR performance when used to synthesise additional training data. To achieve this, we explore a selection method that considers the distance of speakers to other speakers in their embedding space.

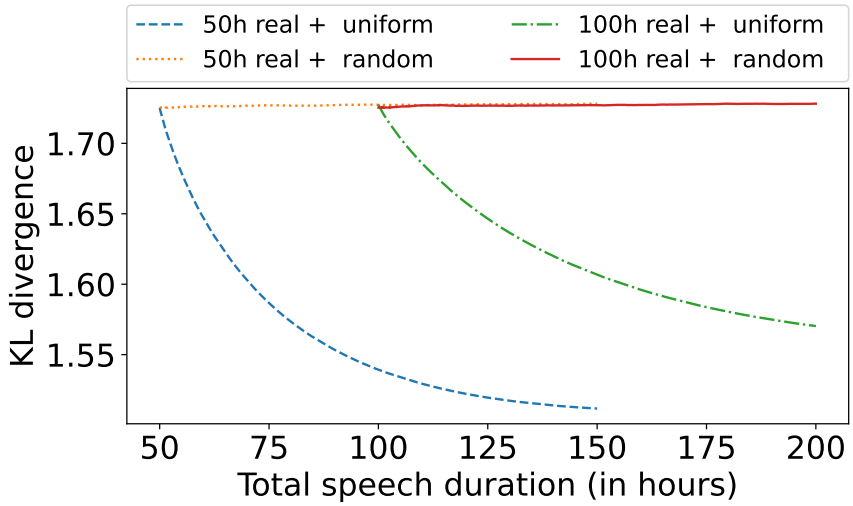
To maximise the speaker diversity, we successively select speakers in T according to their distance to the real speakers and the already selected speakers. Using a distance metric, we first compute the distance between each unselected speaker in T and the closest speaker in $S \cup R$. Then, we select the speaker corresponding to a given distance quantile: the closest speaker to any of the speakers in $S \cup R$ is denoted as *minmin*, the speaker with the median distance is denoted as *medmin*, and the farthest speaker is denoted as *maxmin*. The minmin based speaker selection is meant to avoid outliers while the medmin and maxmin based speaker selection are meant to select more novel speakers. The iterative algorithm for *maxmin* based speaker selection is described in Algorithm 2. Changing arg max to arg min or arg median in Algorithm 2 changes the algorithm to the *minmin* and *medmin* speaker selection method.

Concretely, we compute the cosine distance between the embeddings e_A and e_B of speaker A and speaker B as

$$\text{dist}(e_A, e_B) = 1 - \frac{e_A . e_B}{\|e_A\| \, \|e_B\|}. \tag{5.5}$$



(a) natural desired distribution



(b) uniform desired distribution

Figure 5.2: KL divergence between natural/uniform distribution of texts and combination of real speech texts and newly selected 50 hours / 100 hours equivalent of TTS data.

S is the set of speakers in the real training set and T are the speakers disjoint from S in the Common Voice training dataset. The speaker embeddings are computed by averaging across all the utterances corresponding to a speaker. The speaker embeddings are used to condition the Glow-TTS model to synthesise 50 hours of synthetic data. We iterate over the set of speaker embeddings available during TTS/VC inference to cover all speakers while using 50 hours worth of randomly selected sentences.

In addition to comparing the speaker selection methods, we iteratively increase the number of selected speakers by a factor of 2 to determine the number of synthetic speakers required to significantly improve the performance of the ASR model. Here, we increase the

Algorithm 2 Speaker selection algorithm using maxmin

Require: S, T, n

procedure SELECTSPEAKER(S, T, n) ▷ n is the desired number of speakers
  Initialise $R = \emptyset,\ K = \emptyset$
  while $|R| < n$ **do**
    for t in T **do**
      $d_{t,s} = \min_{s \in S \cup R} \text{dist}(t, s)$
      $K := K \cup \{d_{t,s}\}$
    end for
    $s = \arg\max_{k \in K} k$
    $R := R \cup \{s\}, T := T \setminus \{s\}, K = \emptyset$
  end while
  return R
end procedure

TTS and VC dataset size to 100 hours to have enough samples per speaker. The speakers are randomly selected from T with each smaller set being a subset of the larger set.

Results

We compare the three speaker selection methods (*maxmin*, *medmin* and *minmin*) to *random* speaker selection from the unseen speaker set. As a baseline, we also provide results for the ASR models trained on augmented data from *seen* speakers only, where the seen speakers are the 2,457 speakers in the real ASR training set.

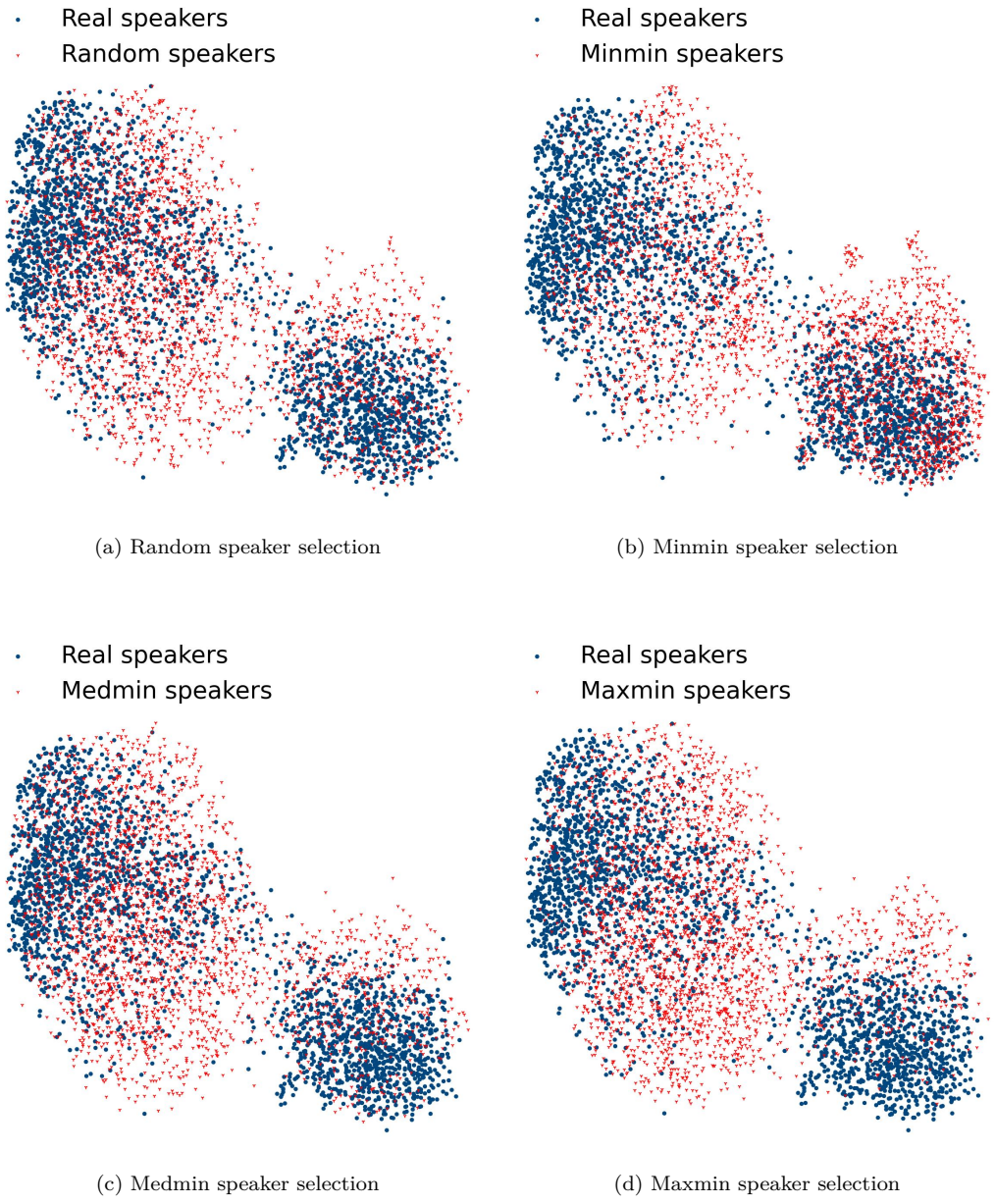
Table 5.3 shows the CER and WER of ASR models on the Common Voice test set when trained on a combination of 50 hours of real and 50 hours of TTS data that has been generated using speakers from the different speaker selection methods. Firstly, the results show a relative reduction in CER and WER of 3% and 2% of the ASR model trained on a combination of real data and TTS data generated using randomly selected unseen speakers compared to TTS data generated using only the speakers in the real ASR training set (seen). This indicates that ASR performance can be improved by randomly adding more speakers. Secondly, we see a statistically significant difference in the ASR performance between medmin and other sentence selection methods in terms of CER.

Table 5.3: CER and WER of ASR models trained using real data and data synthesised using speakers selected randomly or by using the minmin, medmin, or maxmin speaker selection criterion.

Speaker selection method	Data	CER (%)	WER (%)
Seen	50r_50tts	23.28	42.79
Random	50r_50tts	22.57	**42.02**
Minmin		22.55	**42.15**
Medmin		**22.23**	**41.65**
Maxmin		22.47	**41.79**

We plot the speaker embeddings of the real data and those of the selected speakers in Figure 5.3 using UMAP (McInnes et al., 2018). UMAP provides a two-dimensional representation of N-dimensional embeddings, to reflect the distance between the embeddings

as seen in Figure 5.3b. Each of the plots visibly has two main clusters indicating male and female groups. From these plots, we observe that the minmin selection criterion truly selects speakers around the real speakers and therefore only fills up holes in the speaker distribution. For this reason, its plot is denser than others which can be visually seen from the right side of the plot. Visually, the medmin and maxmin selection criteria produce a wider spread of newly selected speakers and therefore expanded the original speaker distribution. Visually, the randomly selected speakers also had a similar effect as medmin and maxmin on the speaker distribution.



(a) Random speaker selection

(b) Minmin speaker selection

(c) Medmin speaker selection

(d) Maxmin speaker selection

Figure 5.3: UMAP plot of speaker embeddings of real data speakers and speakers selected using the speaker selection algorithm.

In Table 5.4 where the ASR systems are trained on a combination of real data and TTS data generated with an increasing number of speakers, we similarly saw a significant

relative decrease in both CER and WER of the ASR models by 2%. Increasing the number of speakers above 2,457 does not yield an improvement in the ASR results. We observe that there is a trade-off between increasing the number of speakers in the dataset to the number of samples per speaker. The results seem to indicate that having more samples per speaker may also be as important as adding new speakers.

Table 5.4: CER and WER of ASR models trained on a combination of real data and TTS generated data when the TTS model is conditioned on different numbers of speakers. Speakers are randomly selected from the candidate set of ASR unseen speakers.

K = 2,457 speakers	Data	% CER	% WER
baseline		22.23	40.98
+ K		**21.84**	**40.21**
+ $2K$	50r_100tts	22.39	40.92
+ $4K$		22.34	40.95
+ $8K$		22.30	41.02

Table 5.5: CER and WER of ASR models trained on a combination of real data and VC-generated data when the VC model is conditioned on different numbers of speakers. Target speakers randomly selected from the candidate set of ASR unseen speakers.

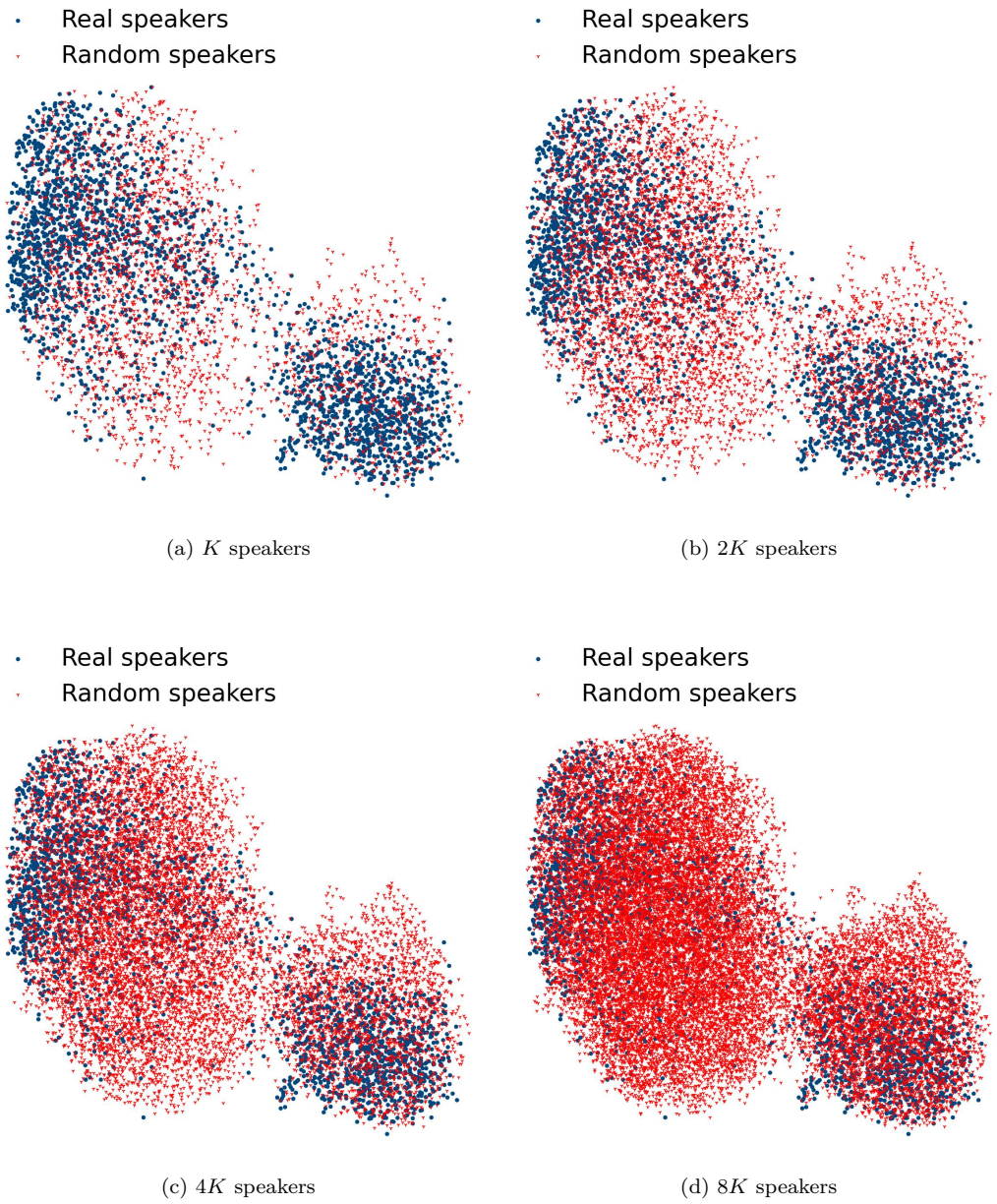
K = 2,457 speakers	Data	% CER	% WER
baseline		24.79	46.77
+ K		**24.60**	**46.04**
+ $2K$	50r_100vc	**24.33**	**46.09**
+ $4K$		**24.61**	**46.03**
+ $8K$		**24.41**	**46.09**

We observe similar results for ASR models trained on real and VC speech as shown in Table 5.5. For these experiments, VC is performed on each real utterance by iterating through the specified number of random ASR unseen speakers until a VC data size of 100 hours is reached. Here, we see a significant drop in the ASR's CER and WER by 1% and 2% relative when 2,457 TTS unseen speakers are used for VC. Increasing the number of speakers beyond 2,457 does not yield any statistical improvement in the results. In addition, VC experiments do not produce similar (or better ASR results) to TTS experiments perhaps due to the lack of augmentation of phonetic diversity.

In Figure 5.4, we plot the speaker embeddings of the real data with the speaker embeddings of K, $2K$, $4K$ or $8K$ randomly selected speakers that are not in the real ASR training set. The distribution of speakers became denser as more speakers were added to the real speaker set.

5.3.3 Increasing phoneme duration diversity

A well-known technique for augmenting an ASR dataset is to either increase or decrease the speaking rate of the utterances through speed perturbation (Ko et al., 2015). Speed perturbation changes the duration and the spectral content of the utterances but does not affect the relative duration of phonemes within each utterance. As a data augmentation strategy, we evaluate the impact of modifying relative phoneme durations instead by using



(a) K speakers

(b) $2K$ speakers

(c) $4K$ speakers

(d) $8K$ speakers

Figure 5.4: UMAP plot of speaker embeddings of real data speakers and K, $2K$, $4K$ or $8K$ randomly selected speakers not in the real dataset with $K = 2{,}457$.

a duration predictor to predict diverse phoneme durations in the TTS/VC model. We also consider time stretching to modify the duration of the utterances without affecting their spectral content.

Methodology

Our method involves only augmenting the phoneme duration while keeping the speaker characteristics and phonetic content of the utterances. Therefore, the duration augmentation method requires that we provide two different speaker embeddings to the TTS/VC model, the speaker embedding of the original real speech sample and another randomly

chosen speaker embedding from the pool of real speaker embeddings. As shown in Figure 5.5, the original or target speaker embedding is used to condition the decoder during generation and for MAS alignment of the original utterance (in the case of VC) while the duration predictor is conditioned on the randomly chosen speaker embedding to generate the phoneme duration. In the case of VC, the aligned phoneme duration of the original

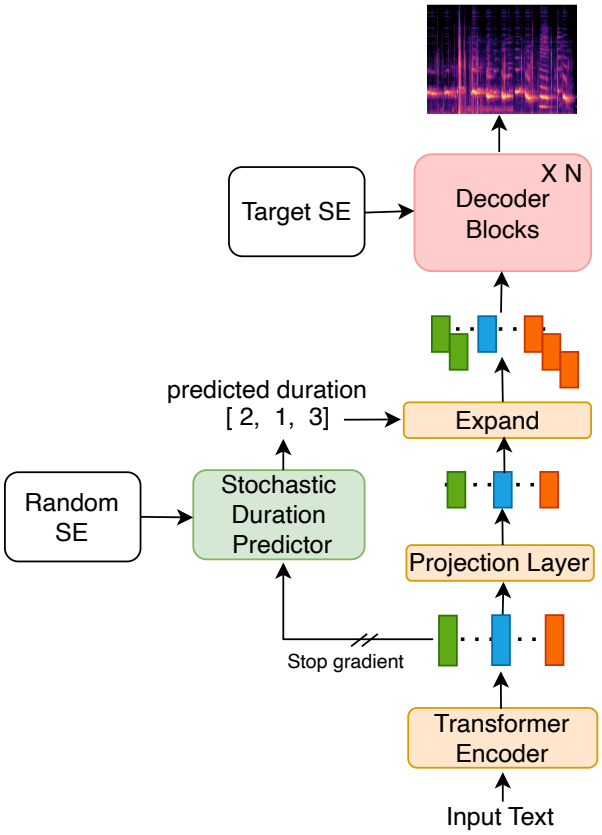


Figure 5.5: Proposed method to increase phoneme diversity in GlowTTS-STD. The method is also used for our trained Glow-TTS model. SE denotes speaker embedding.

real speaker and the predicted phoneme duration of a randomly chosen speaker are also required. Since the length of some phonemes for the original and the randomly chosen speaker will differ, we either duplicate or drop frames in the latent representation to match the newly predicted duration. Making the changes in the latent representation does not utterly degrade the naturalness of the synthesised utterances, except for some noticeable co-articulation errors. We therefore set a maximum threshold of five frames that can be dropped or added for any phoneme duration. Speed perturbation can also be performed by resampling the original speech signal, however this affects the spectral content. Here, since we want to evaluate only the duration, we apply time-stretching as implemented in the TimeStretch class in the Audiomentations library.[20]

In addition to the main experiments, we include four baselines which combine the real data with the following derived data:

- *50r_50rt*: time-stretching the real data to create an augmented time-stretched ver-

[20] https://github.com/iver56/audiomentations.git

sion of the same size,

- *50r_50rto*: time-stretching the real data on the fly with a probability of 0.5 on every dataset iteration,
- *50r_50rtts*: using the original real speaker embedding for predicting the durations for TTS augmentation, and
- *50r_50rvc*: using a random speaker embedding for VC without changing the duration of the original utterance.

We evaluate the duration-based data augmentation with 50 hours of real speech. Here, we denote the datasets augmented with the synthetic data using our method of generating diverse durations as $50r\_50tts$ and $50r\_50vc$.

We set the minimum stretching rate and maximum stretching rate to 0.9 and 1.1, following previous data augmentation studies by Ko et al. (2015) and Rossenbach et al. (2023). TTS/VC-based utterance generation is done using Glow-TTS and GlowTTS-STD. As previously described in Section 5.2.2, GlowTTS-STD uses a generative flow-based duration predictor to generate phoneme durations. We iterate over the set of real speakers during TTS/VC generation to cover all speakers.

Results

Table 5.6 shows the results of these experiments. Beginning with the ASR models trained with time-stretching, generating the time-stretched speech samples on the fly decreases the CER and WER by 4% and 5% relative compared to generating a 50-hour time-stretched version of the real data. We also observe that adding TTS data to the real data provides a greater improvement to the performance of the ASR systems. In particular, naively regenerating the real utterance's text using Glow-TTS with the same real speaker's embedding reduces the CER and WER of the ASR system by 4% and 3% relative over applying time-stretching to the real data with a probability of 0.5. Applying a randomly selected speaker to drive the duration predictor in Glow-TTS for duration diversity further reduces the WER by 1% relative, although we do not see a significant difference in the CER. While using the GlowTTS-STD model provides a significant improvement over the time-stretching baselines, it does not improve the ASR system relative to Glow-TTS. Lastly, ASR models using VC-generated utterances for data augmentation had the lowest performance.

5.3.4 Increasing pitch diversity

Pitch modification has been applied for data augmentation in some ASR studies, e.g., by Casanova et al. (2023). Besides, classical speed perturbation based data augmentation results in a modification of the pitch. For this reason, we examine the impact of pitch diversity in ASR by comparing augmentation using TTS/VC models to classical pitch modification methods.

Methodology

Similar to what was described in Section 5.3.3 for generating diverse phoneme durations, here we only want to change the pitch without changing the phonetic or prosodic attributes

Table 5.6: CER and WER on the Common Voice test set of ASR models trained on a combination of 50 hours real and synthetic data where different duration augmentation methods have been applied to increase duration diversity.

Model	Data	CER (%)	WER (%)
Baselines			
-	50r_50rt	25.09	47.28
-	50r_50rto	24.04	44.83
Glow-TTS	50r_50rtts	**23.19**	43.44
Glow-TTS	50r_50rvc	25.02	46.97
Text-to-speech			
Glow-TTS	50r_50tts	**23.28**	**42.95**
GlowTTS-STD		23.95	43.31
Voice conversion			
Glow-TTS	50r_50vc	24.44	45.06
GlowTTS-STD		25.85	47.68

of the utterance. As such, we provide two different speaker embeddings to the TTS/VC model as shown in Figure 5.6: the speaker embedding of the original or target real speech sample and another randomly chosen speaker embedding from the pool of real speaker embeddings. The original speaker embedding is used to generate phoneme durations and to condition the decoder during Mel-spectrogram generation or inversion (in the case of VC) while the randomly chosen speaker embedding is used to condition the pitch predictor for generating per-frame log-F0 values.

We also included two baselines which combine real data with the following derived data:

- *50r_50rp*: pitch-shifting the real data to create an augmented version of the same size,
- *50r_50rpo*: pitch-shifting the real data on the fly with a probability of 0.5 on every dataset iteration.

Along with the baselines, we train ASR models using the real data combined with the following data generated either by TTS or VC:

- *50r_50tts_osp*: TTS data generated by GlowTTS-STDP using the original real speaker embedding for pitch prediction,
- *50r_50tts_rsp*: TTS data generated by GlowTTS-STDP using a random speaker selected from the real speaker set,
- *50r_50vc_rsp*: VC data converted by GlowTTS-STDP by providing predicted log-F0 values of a random speaker. Here, we use the original speaker embedding for conditioning the vocoder.

For pitch shifting, a value is randomly chosen in the range of -2 to +2 semitones. The pitch-shifting augmentation is performed using the PitchShift class in the Audiomentations library.[20] TTS/VC-based pitch augmentation is done using GlowTTS-STDP as it is the only model that uses external pitch information or generates log-F0 values. Also, we iterate over the set of real speakers during TTS/VC generation to cover all speakers.

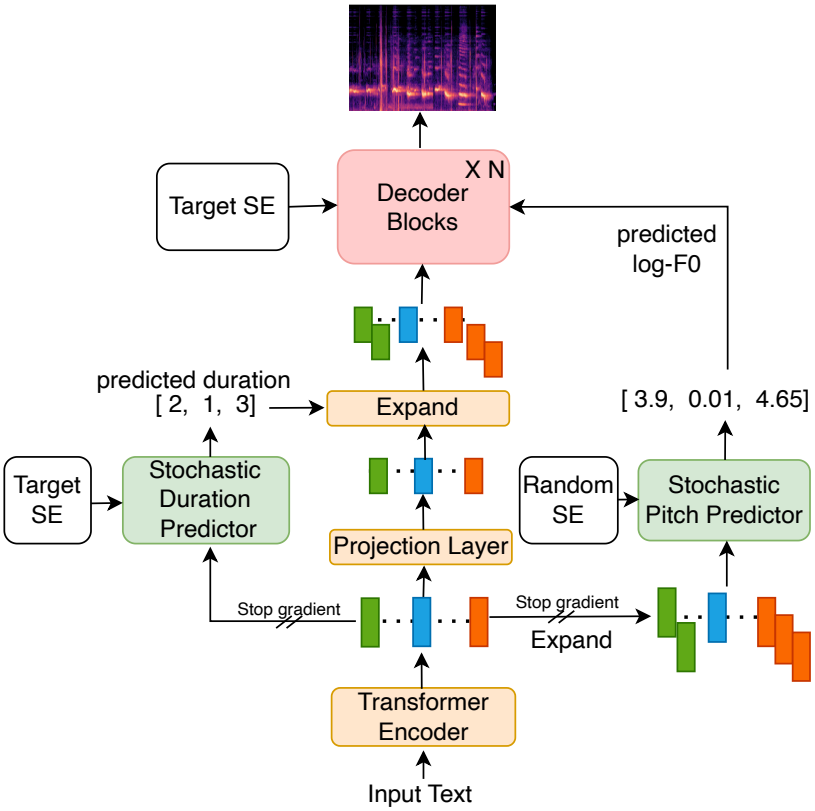


Figure 5.6: Proposed method to increase pitch diversity in GlowTTS-STDP. SE denotes speaker embedding.

Results

Table 5.7 shows the results of these experiments. Firstly, we observe that applying pitch shifting on the fly with a 0.5 probability negatively impacts the ASR results and performs worse than generating a 50 hour pitch-shifted version of the real data. This indicates that a large amount of pitch-shifting can have an unintended negative effect on ASR performance. Secondly, we observe that using a random speaker embedding for pitch prediction reduces both the CER and WER by 1% relative over using the original real speaker embedding for pitch prediction. Although this shows that the model benefits from pitch diversity, it is still worse than ASR models using a standard Glow-TTS without pitch modifications. Additionally, comparing Table 5.6 to Table 5.7 indicates that duration augmentation is more beneficial to ASR training data augmentation than pitch augmentation. Lastly, as has been the case with other experiments, the ASR model does not benefit from adding pitch-augmented VC data to the real data.

5.3.5 Matching the environmental conditions of real speech

In the previous experiments, all the ASR models were trained on a combination of real and synthetic data without considering the difference in environmental conditions of the combined datasets. Therefore, in this section, we consider the impact of synthetic data augmented with ambient noise and varying room characteristics on the performance of the

Table 5.7: CER and WER on the Common Voice test set of ASR models trained on a combination of 50 hours real and synthetic data where different pitch augmentation methods have been applied to increase pitch diversity.

Model	Data	CER (%)	WER (%)
Baselines			
-	50r	25.01	46.30
-	50r_50rp	24.96	46.66
-	50r_50rpo	26.04	48.19
Text-to-speech			
GlowTTS-STDP	50r_50tts_osp	24.30	44.59
	50r_50tts_rsp	**24.06**	**44.14**
Voice conversion			
GlowTTS-STDP	50r_50vc_rsp	25.48	47.33

ASR systems.

Methodology

To simulate a more noisy dataset, additive noise is randomly added to the synthesised utterances with a random signal-to-noise ratio (SNR) between 0 dB and 15 dB. Similarly, to simulate reverberation in speech, the synthetic utterances are randomly convolved with room impulse response (RIR) filters. We modified the TTS/VC-generated utterances with a probability of 0.25 or 0.5. Finally, we performed experiments covering both augmentation methods.

Concretely, in each training set batch, if a given synthetic utterance is selected for reverberation augmentation it is convolved with a random simulated RIR and if it is selected for noise augmentation a random noise sample is added to it (or to the reverberated utterance in case it was also selected for reverberation augmentation). The added noises are taken from the 6 hour noise subset of the MUSAN corpus (Snyder et al., 2015) which consists of technical and non-technical noises while the simulated RIRs are those provided by Ko et al. (2017b).

Results

Tables 5.8 and 5.9 show the results for ASR models trained with real data and augmented TTS/VC-generated data. Firstly, concerning TTS-generated data in Table 5.8, we observe that training on synthetic data with noise or reverberation improves the performance of the ASR models. With 25% noise augmentation probability, we observe a 3% and 5% relative drop in CER and WER over training without noise. Increasing the probability to 50% further reduces the CER and WER by 7% and 5% relative compared to training without noise. Secondly, adding reverberation to 25% of the TTS-generated utterances reduces the CER and WER by 1% and 2% relative. Increasing the probability to 0.5 does not improve the ASR performance. In general, adding noise or reverberation is better than using only the TTS-generated data without augmentation. Lastly, combining noise and reverberation at different probabilities reduces the CER and WER compared to using either noise or reverberation.

Table 5.8: CER and WER of ASR models trained on a combination of 50 hours real and 50 hours TTS generated data augmented with noise or RIRs with different probabilities (Prob).

Data	Prob	CER (%)	WER (%)
50r_50tts	0.0	23.19	43.44
Noise added to TTS data			
50r_50tts	0.25	22.02	42.12
	0.5	**21.47**	41.20
Reverb added to TTS data			
50r_50tts	0.25	22.92	42.62
	0.5	22.91	42.67
Noise + Reverb			
50r_50tts	0.25	21.89	41.59
	0.50	**21.55**	**40.56**

Similarly, concerning VC generated data in Table 5.9, while adding either noise or reverberation reduces the CER and WER of the ASR model, adding noise has a larger impact on ASR performance. In particular, adding noise and reverberation degraded the ASR performance compared to adding noise only. This indicates that VC-generated data mostly benefits from added noise and not from simulating reverberation.

Table 5.9: CER and WER of ASR models trained on a combination of 50 hours real and 50 hours VC-generated data augmented with noise or RIR at different probabilities (prob).

Data	prob	CER (%)	WER (%)
50r_50vc	0.0	25.02	46.97
Noise added to VC-generated data			
50r_50vc	0.25	22.97	44.28
	0.5	**22.66**	**43.63**
Reverb added to VC-generated data			
50r_50vc	0.25	24.43	45.88
	0.5	24.44	45.79
Noise + Reverb			
50r_50vc	0.25	23.18	44.51
	0.50	22.84	43.95

5.3.6 Combining all the attributes

Methodology

In a final set of experiments, we combine the augmentation methods explored above that independently reduced the CER and WER of the ASR models. Therefore, we select the following methods:

- Phonetic diversity: sentences selected using the natural selection strategy.
- Speaker diversity: 2,457 unseen speakers selected using the medmin speaker selection criterion.

- Duration diversity: augmented phoneme duration using random speaker embeddings from the 2,457 unseen speakers.

- Pitch diversity: not included in experiments because TTS/VC utterances generated by the GlowTTS-STDP model did not significantly reduce the WER relative to Glow-TTS.

- Matching the environmental conditions of real speech: applied reverberation and noise to TTS generated utterances with 50% probability on every data iteration.

- Model: using Glow-TTS for TTS only since VC-generated utterances do not reduce the WER significantly in comparison to TTS-generated utterances.

We iteratively increase the size of the synthetic data by a factor of 2 until we do not see a significant reduction in CER and WER.

Results

Table 5.10 shows the CER and WER of the ASR models trained on the combination of attributes. We have added some baselines to the table for easy comparison. We included a baseline of 50 and 100 hours of real data and a baseline of 50 hours of real data that has been time-stretched with a probability of 0.5. We copy the results from their respective tables. Firstly, we observe that combining the data augmentation methods reduces the CER and WER of an ASR model trained on either 50 hours or 100 hours of real data. It reduces the CER and WER of the ASR model trained on the 100 hours of real data by 4% and 1% relative. Here, we believe that noise addition to the synthetic data has contributed largely to this improvement. Increasing the size of the TTS data added to the 50 hours real data continues to improve the ASR model until 200 hours of data, indicating a limit to the improvement that can be achieved with the synthetic data.

Table 5.10: CER and WER of ASR models trained on a combination of 50 hours real data and different sizes of TTS generated data combining all the attributes that significantly reduced CER and WER.

Data	CER (%)	WER (%)
50r	25.01	46.30
50r_50rto	24.04	44.83
100r	22.10	40.81
50r_50tts	21.17	40.39
50r_100tts	20.05	38.32
50r_200tts	**19.70**	**37.07**
50r_400tts	**19.70**	37.31

Additionally, synthetic data augmentation can provide higher benefits in comparison to a classical data augmentation method (time-stretching). Here, we observe a 12% and 11% relative reduction in CER and WER of ASR model trained on a combination of 50 hours of real data and 50 hours of diverse synthetic data over an ASR model trained on the 50 hour time-stretched data.

5.4 Conclusion

This chapter explored different data augmentation methods to increase the diversity of TTS and VC-generated data that is combined with real data for ASR training. By combining the data augmentation methods that individually improve ASR performance, we were able to reduce the CER and WER of the ASR model trained with real and TTS-augmented data. In particular, our experiments showed that adding some individual attributes to the training data via TTS synthesis increases ASR performance. In particular, increasing the phonetic diversity of the dataset reduces the CER by 12% and WER by 14% relative compared to a dataset containing only real data. In addition, adding new speakers reduces the CER by 5% and WER by 3% relative compared to a dataset of real and TTS-generated data that contains only the speakers in the real dataset. Increasing the phoneme duration diversity relatively reduces the WER by 1% relative compared to a baseline dataset of real and TTS-generated data. VC conversion was not as beneficial as TTS for ASR data augmentation. Also, increasing the pitch diversity did not improve ASR performance, probably because pitch information is already adequate in the real data of 2,457 speakers. Adding noise and reverb to the TTS-generated dataset combined with the real data for ASR training reduces both the CER and WER by 7% relative compared to a baseline that does not add noise and reverb. Finally, combining all the attributes that help improve ASR performance seems to be beneficial and complementary. Therefore, this work further highlights the need for diversity of training utterances for better ASR performance, and this diversity can be provided using TTS-generated utterances. Controlling diversity is also not limitless as our work shows, and too much diversity can saturate the performance of the ASR system.

6

Conclusion and perspectives

This thesis explored the improvement of the performance of ASR models by training the models on real data augmented with synthetic data generated using text-to-speech (TTS) and voice conversion (VC) systems. The TTS and VC systems were created to generate more diverse utterances and were also trained on a dataset curated from an ASR corpus. This led us to touch on several tasks, including the curation of a high-quality TTS corpus from a crowdsourced ASR dataset the creation of a flow-based TTS system integrating stochastic duration and pitch predictors, and the evaluation of the impact of TTS and VC based ASR training data augmentation along different speech attributes on the resulting ASR performance. In this chapter, we conclude the thesis with a summary of our contributions and results in Section 6.1 and present future research directions in Section 6.2.

6.1 Conclusion

In Chapter 3, we developed a method for curating a TTS training dataset from an ASR dataset. This method leverages the increasing accuracy of deep-learning-based, non-intrusive quality estimators to filter high-quality samples. Specifically, we use a self-supervised model fine-tuned for mean opinion score (MOS) estimation, WV-MOS (Andreev et al., 2023), for estimating the average data quality for each speaker. We explored filtering the ASR dataset at different thresholds to balance the size of the dataset, the number of speakers, and the quality. Hence, we evaluated the generated utterances from the multi-speaker TTS model trained individually on each of the filtered datasets. From the experiment, we showed that at a WV-MOS threshold ≥ 4.0, the utterances generated by the TTS model trained on the filtered dataset improved in overall quality on average by 1.26 MOS point relative to a baseline of all the utterances. The utterances generated by TTS trained using the filtered dataset were also 7% more intelligible on average than the baseline as shown by the intelligibility score. In addition, the speaker similarity between reference speakers and the utterances generated by the filtered dataset increased by 0.19 MOS point relative to the baseline. These results validate our method of curating a high-quality dataset. In addition, we showed the effect of denoising the utterances on the quality of the dataset. We showed that denoising degrades the WV-MOS score and the CER, except for low-quality datasets in the case of the CER. This revealed that denoising the dataset is not beneficial and instead, automatically selecting high-quality samples is the better strategy. Also, comparing the curated 4,469-speaker, 230.75 hour

dataset to LibriTTS (Zen et al., 2019), we achieve similar overall quality and intelligibility for utterances generated from TTS trained on either of the datasets. This also shows that our dataset is of high quality and is comparable to widely used TTS datasets such as LibriTTS (Zen et al., 2019), which is curated from relatively higher-quality audiobooks.

In Chapter 4, we tackled the problem of capturing the diversity of speech in TTS/VC systems. Previous TTS/VC systems either develop methods to capture this diversity by conditioning the system on a speaker embedding alone or by using discriminative models to learn speech variabilities. In our approach, we designed an improved flow-based architecture that learns the distribution of different speech variables that affect the diversity of speech. Here, we modified the generative, flow-based Glow-TTS model (Kim et al., 2020) by integrating generative-based models for learning the distributions of pitch and duration. Although Glow-TTS has a speaker-dependent duration predictor, the predicted durations have been found to yield unnatural rhythm and cannot express how a person speaks at different speaking rates (Kim et al., 2021). Therefore, the stochastic duration predictor learns more realistic phoneme duration and gives better speech rhythm to the utterances while the stochastic pitch predictor learns the distribution of pitch contours in the dataset.

Our modifications led to the development of two variants of the Glow-TTS model: GlowTTS-STD which replaces the deterministic duration predictor in Glow-TTS with a stochastic duration predictor, and GlowTTS-STDP, Glow-TTS integrated with a stochastic duration predictor and stochastic pitch predictor. We find that our modifications significantly increase the diversity and naturalness of the generated utterances over the baseline Glow-TTS. In particular, the naturalness of the utterances generated by GlowTTS-STDP in the speaking style (shorter utterances) was 0.32 MOS points higher than the utterances generated by the baseline GlowTTS, and 0.14 MOS points higher than the utterances generated by GlowTTS-STD. In a similar direction, the naturalness of the utterances generated by GlowTTS-STDP in the reading style (long passage) was 0.44 MOS points higher than the utterances generated by the Glow-TTS baseline, and 0.13 MOS points higher than the utterances generated by GlowTTS-STD. These results show that integrating the stochastic duration predictor and the stochastic pitch predictor improves the naturalness of generated utterances. In terms of diversity, the utterances generated by the GlowTTS-STDP model were rated to be more diverse than both GlowTTS-STD and Glow-TTS. In particular, the utterances generated by the GlowTTS-STDP model were rated 0.25 MOS points higher than GlowTTS-STD. This shows that the stochastic pitch predictor also adds diversity to the utterances, and not only the stochastic duration predictor which generates rhythm for the utterances. In addition, we compared the distribution of log-F0 values in real data to the distribution in the generated utterances by the developed TTS/VC models and the baseline Glow-TTS. Here, we noticed that log-F0 values extracted from utterances generated by GlowTTS-STDP closely matched the distribution of log-F0 values in the real data. This shows that the log-F0 predictions by the stochastic pitch predictor are realistic.

In Chapter 5, we evaluated the impact of the diversity of TTS and VC data for augmenting real data for the task of training ASR systems. As opposed to naively generating synthetic data for ASR data augmentation, we examined different approaches to generating diverse data using our trained TTS/VC models. Firstly, we developed a method to select the most informative sentences from a large text corpus. We showed that ASR models trained with a combination of real data and utterances generated from the selected text improve ASR performance over randomly selecting text. In particular, increasing the

phonetic diversity of the dataset reduces the CER by 12% and WER by 14% relative compared to a dataset containing only real data. Secondly, we increased the diversity in the speaking rate and pitch by augmenting the phoneme duration and the predicted fundamental frequency. In addition, we developed a speaker selection method to select new speakers. Here, we found that our augmentation and selection methods can independently have a positive impact on the performance of the ASR model. Adding new speakers reduces the CER by 5% and WER by 3% relative compared to a dataset of real and TTS-generated data that contains only the speakers in the real dataset. Increasing the phoneme duration diversity reduces the WER by 1% relative compared to a baseline dataset of real and TTS-generated data. Additionally, VC conversion was not as beneficial as TTS for ASR data augmentation. Also, increasing the pitch diversity did not improve ASR performance, probably because pitch information is already adequate in the real data. Finally, we considered the effect of environmental factors such as noise and reverberation in the synthetic data on ASR performance. Adding noise and reverberation to the TTS-generated dataset combined with the real data for ASR training reduces both the CER and WER by 7% relative compared to a baseline that does not add noise and reverberation.

Finally, combining all the attributes that individually improved the ASR performance further reduces the CER and WER of the ASR models up to a point. This shows that dataset diversity can also saturate the performance of the ASR system like other augmentation methods. When compared with the classical time-stretching data augmentation method, we observed a 12% and 11% relative reduction in CER and WER of the ASR model trained on a combination of 50 hours of real data and 50 hours of diverse synthetic data over an ASR model trained on the 50 hour time-stretched data. In conclusion, our experiments provided insight into the variabilities that are particularly important for ASR and showed the effectiveness of using diverse synthetic data as a form of data augmentation.

6.2 Perspectives

The fields of TTS and ASR are fields that rapidly advance and improve on previous architectures, applications, and algorithms. Given the current landscape of research and technologies in this domain, we have identified five areas for further exploration.

6.2.1 Disentangling other speech factors in the TTS model for greater control of speech variabilities

In Chapter 4 of this thesis, we developed a TTS/VC system that disentangled rhythm, pitch, and speaker characteristics. However, several other factors like age, regional accent, and emotion, affect the statistical distribution of speech data and were not taken into account. Therefore, in future work, it could be interesting to examine how to control these factors during utterance generation to further cover the distribution of real data. For example, considering age, the acoustic properties of speech can be different for an elderly person, a teenager, or a child. TTS systems used for ASR training data augmentation would benefit from controlling these characteristics independently to generate utterances for the parts of the data distribution that are not represented in the TTS training data. In addition, regional or non-native accents and emotions affect the characteristics of speech. Disentangling accents and emotions can aid in increasing the diversity of the synthetic

data by enabling the combination of arbitrary accents or emotions with arbitrary speaker characteristics.

More generally, disentangling all variability factors in speech can be an interesting direction for both TTS/VC and ASR data augmentation. For TTS/VC, it could help control utterance generation for different characters in applications like audiobooks, podcasts, and role plays. Similarly, TTS/VC-generated utterances could be generated with greater control considering the most important aspects that improve general ASR performance so as to improve the ASR model on a specific domain where it has been found to have poor performance, e.g., some accent or some age group.

6.2.2 Using multilingual TTS models

In addition to disentangling speech characteristics in TTS/VC models, it could also be interesting to leverage multilingual TTS models like XTTS (Casanova et al., 2024) to cover speaker variabilities like accents. A multilingual model that has been trained on speakers in a language can be used to generate speech for another language if the TTS model is controllable and can be conditioned on the speaker variabilities that have been discussed (Casanova et al., 2023). For example, a TTS model that has been trained jointly on the English and Mandarin languages could then be used for generating utterances in English using Mandarin speakers or vice-versa. This method may inherently provide a way to learn the typical characteristics of Mandarin speakers when they speak the English language.

6.2.3 Better representation of speaker characteristics

Zero-shot TTS still shows that there is a large gap between real speech from a speaker and generated utterances using the speaker embedding of that speaker. One identified problem is that compressing the speaker's information into a single vector cannot perfectly represent the specific characteristics of the speaker, and a lot of information is lost (Kim et al., 2024). Some TTS models tackle this by providing several embeddings to the model to tackle different characteristics, e.g., accent, language, environment, etc., while some recent TTS models have adopted the approach of applying a small segment of the target speaker's speech as conditioning (Kim et al., 2024). The latter approach has two downsides; first, it assumes that there is always at least a recording of 3 s or more of speech from the target speaker, and second, it also makes it more difficult to disentangle the different speech characteristics since the linguistic information can easily mix with speaker information. Finding the best way to represent the acoustic characteristics of speakers and speech features can increase the similarity of real speakers' speech and those synthesised by TTS models while enabling higher controllability. This can include using a transformed version of the speech representation that reduces linguistic content while keeping much of the speaker and prosodic information.

6.2.4 Improved metrics for measuring synthetic data quality

Humans' perception of the world around them is biased, and this is also the case for the perceptive evaluation of synthetic audio. Studies have shown that subjective metrics like mean opinion scores may differ depending on the instructions given during the listening test (Kirkland et al., 2023). Therefore, model-based speech evaluation metrics have been developed in the past decade to automatically evaluate the naturalness and quality of

speech samples (Patton et al., 2016; Mittag and Möller, 2020). While these subjective and objective metrics currently show that synthetic speech is approaching the naturalness and quality of real speech, there still seems to be a gap between the distribution of real and synthetic speech (Minixhofer et al., 2023). In particular, ASR model performance drops drastically when trained on synthetic speech only compared to real speech (Minixhofer et al., 2023). Therefore, metrics that evaluate the closeness of the synthetic speech to real speech at the acoustic level could be developed in addition to the current subjective and objective metrics. Additionally, metrics can be developed to evaluate the controllability of TTS models, e.g., metrics that could measure the diversity of utterances generated by TTS models.

6.2.5 Using large language models to generate domain-specific text for ASR

In Section 5.3.1, we examined how to increase the phonetic diversity of the utterances by selecting more informative utterances. We were able to achieve significant improvement in the ASR performance by selecting text using a selection method that considers the natural distribution of text over random selection. Similarly, the current progress in large language models (LLMs) (Wei et al., 2022) could also be of benefit to ASR and TTS to generate text. With prompting techniques, an LLM can be provided with instructions to create texts that can be used to synthesise speech for TTS (Su et al., 2024). Specifications like domain, language, length of text, phonetic content, and exceptions can guide the LLM to create texts. This kind of text generation was previously possible but requires hours or days to select the text from a large available corpus. With LLMs as a reservoir of text and with more intelligent methods to retrieve and generate text, this LLM approach could reduce the text curation time, enable the fast generation of a text corpus for specific domains, etc., by using methods like few-shot learning and prompting. In the future, it may even be possible to teach LLMs to learn a new language fast to communicate in that language. With that, text could be quickly generated for speech synthesis and ASR augmentation.

6.2.6 ASR models invariant to real vs. synthetic data

In Section 2.5.6, we discussed different approaches that can be used to combine real data and synthetic data during the training of ASR models. Therefore, an interesting direction could be to develop ASR models and training methodologies that learn optimally from both real speech and synthetic speech. This requires that the model extracts the necessary information in the synthetic speech and does not focus on peculiarities that make it different from real speech. Hence, this approach may require changing the architecture of the ASR model to better accommodate synthetic speech or introduce new learning objectives. In addition, new paradigms may need to be developed to enable optimal training with real data and synthetic data. One idea could be to explore the use of neural modules, popularly known as adapters, that jointly learn synthetic data-specific features from the synthetic data, while some other parts of the ASR model learn the important linguistic features contained in the speech.

A

Advancing inclusive multi-speaker multi-accent speech synthesis

A.1 Introduction

Synthetic voices are used in everyday applications for text reading, content generation, voice-over, etc., and provide audio feedback in applications such as maps, language learning tools, smart speakers, and voice assistants. Synthetic voices have also found wider adoption as a plugin with the proliferation of large language models. With the high naturalness and high quality of synthetic voices (Kim et al., 2021; Wang et al., 2023), they have become more widely used on online platforms and social media.

Synthetic voices are usually generated by speech synthesis systems, and there are several open-source systems available for use (Casanova et al., 2022; Kim et al., 2021; Casanova et al., 2024). Several of these systems are provided in English, and other efforts have been made to cover over 1,000 languages of the world, including many African languages (Pratap et al., 2024).

Over 1.4 billion people reside in Africa, and more than 237 million people speak the English language at a bilingual or native level (Wikipedia contributors, 2024), including Nigeria, Uganda, South Africa, etc., and several other African countries speak it as a second or third language. Considering this large demography of speakers, the current representation of personas in synthetic voices does not show this diversity in terms of the typical African accent.

Generating synthetic speech with personas similar to one's demography is paramount for the widespread adoption and acceptability of the technology. For example, a Nigerian content creator would prefer to generate speech in an accent (or language) that is familiar and natural to their audience. Voice cloning methods have been created to generate speech in the voice of a reference speaker, however these systems still fail to generalise to the diverse African accents from our analysis. This is because speaker representation in speech synthesis systems favours more general accents like the American, British, or Australian accents. Even the widely used English text-to-speech (TTS) datasets are not representative of diverse demographies (Ito and Johnson, 2017; Yamagishi et al., 2019; Zen et al., 2019).

Therefore, in this work, we focus on expanding the diversity of synthetic personas in TTS systems to the typical African accent. Our work covers 747 speakers and 86 accents from 9 countries. We focus on improving several TTS systems in generating

voices with African-like personas, enabling a larger representation of African voices for use in applications like podcasts and content creation.

The remaining part of this appendix is structured as follows. Sections A.2 and A.3 describe the dataset, data preprocessing methods, TTS models, evaluation protocols, and experiments. We discuss the results in Section A.4 and conclude with limitations and the summary of the work in Sections A.5 and A.6 respectively. This work has been accepted for presentation at Interspeech 2024 (Ogun et al., 2024).

A.2 Methodology

A.2.1 Dataset: Afro-TTS

Curating a dataset of diverse African-accented English speakers is crucial for this work. Therefore, we initiated our research by online crowd-sourcing of data, following the methodology used in a previous work (Olatunji et al., 2023). Volunteers were asked to read and record a sequence of English texts. The texts contained general domain words and were specifically enriched with African-named entities to accurately represent the diversity of African names and organisations. Table A.1 shows the dataset statistics. The data collection process involved 747 paid contributors from 9 countries representing 86 accents. Contributors consented to have their audio used for TTS. The 136-hour, 16-bit, 48 kHz audio files, along with their metadata containing anonymised speaker identity, country, accent, age group, gender, have been open-sourced to facilitate African speech research.[21]

Table A.1: Afro-TTS dataset statistics. The country column indicates the ISO 3166-1 country code.

Country	No. of samples	No. of speakers	No. of accents	Duration (h)
NG	25,564	549	48	99.85
KE	5,307	58	8	16.45
ZA	4,279	125	20	16.41
GH	727	4	3	2.28
ZW	47	6	3	0.20
RW	40	1	1	0.15
SL	38	1	1	0.14
UG	26	2	1	0.08
ZM	8	1	1	0.04

Dataset preprocessing

As the dataset was recorded remotely on different devices, it needed to be processed to be suitable for TTS training. Directly filtering out low-quality utterances using a high WV-MOS threshold as was done in Chapter 3 will remove more than 70% of the dataset. Therefore, to keep a higher percentage of the dataset, we processed the utterances as follows:

- Denoising: The speech samples were denoised using a speech enhancement model (Défossez et al., 2020), which removes various background noise, including stationary

[21] Models and dataset can be accessed via https://huggingface.co/intronhealth/afro-tts.

and non-stationary noises, and room reverberation. The speech enhancement model by Défossez et al. (2020) is one of the few models that adds little to no artifacts to the speech after denoising,

- Bandwidth-extension: The denoised samples were passed through a bandwidth-extension model, VoiceFixer (Liu et al., 2021), to improve the quality of some of the highly degraded utterances, i.e., utterances recorded with low-resolution and clipping distortion. VoiceFixer has three audio processing modes which are suitable for different levels of degradation. To keep the audio file with the best quality per sample, we evaluated the quality of the denoised sample and the three enhanced samples using a quality estimator, WV-MOS (Andreev et al., 2023)[22], then we selected the sample with the highest predicted MOS score among the samples per utterance. All samples were processed at a sampling rate of 16 kHz,
- Volume normalization and voice activity detection: The speech samples were normalised to a volume level of -27 dBFS using the RMS-based normalisation method in ffmpeg[23] and a voice activity detection tool[24] was used to eliminate pauses longer than 30 milliseconds from the speech recordings.

We selected 736 samples covering speakers and accents with more than 20 minutes of data for testing. Then, the remaining samples were randomly split into training and development sets with 35,042 samples and 200 samples respectively. Additionally, we excluded recordings with a duration of over 50 seconds or samples with character counts exceeding 400 for faster training. The code for data processing described here can be accessed online.[25]

Furthermore, the text inputs were processed as follows:

- Abbreviations and name titles were expanded, e.g., "Alh → Alhaji", "Maj → Major",
- Numbers and dates were converted to their word forms using the NeMo text normalisation toolkit (Zhang et al., 2021c),
- Some punctuation marks (open bracket, closed bracket, colon, and semicolon), were also expanded to their word form, as they were read out in the dataset according to annotation instructions.

A.2.2 Evaluation protocol

For each utterance in the test set, we generated utterances using each of the TTS systems and then computed several subjective and objective metrics to compare the performance of the TTS systems.

Objective evaluation

For objective evaluation, we computed the following:

- **WV-MOS**: evaluates the overall quality of the utterances using a model trained for estimating the quality of an utterance,

[22] https://github.com/AndreevP/wvmos
[23] https://github.com/slhck/ffmpeg-normalize
[24] https://github.com/wiseman/py-webrtcvad
[25] https://github.com/intron-innovation/AfriSpeech-TTS/tree/vits/src/utils

- **NISQA**: evaluates the naturalness of the utterance using the NISQA model (Mittag and Möller, 2020) that has been trained to evaluate synthetic speech naturalness,
- **cos-sim**: evaluates the cosine similarity of the target speaker embedding to the reference speaker embedding. It measures the speakers' similarity objectively.
- **WER**: measures the intelligibility of the generated utterances. We computed the word error rate (WER) between the ground truth text and text predicted by Whisper-large-v3 (Radford et al., 2023) from the Hugging Face library.

Subjective evaluation

For subjective evaluation, we performed listening tests on the generated utterances using the Intron Speech Vault Platform.[26] We asked participants to evaluate the following objective metrics on a Likert scale of 1–5 with intervals of 1:

- **MOS**: the overall quality of the utterance. The participants were asked to consider the absence of noise and correct pronunciation in their evaluation,
- **Nat-MOS**: the naturalness of the utterance. The participants were asked to evaluate how natural or human-like the speech sounds, regardless of intelligibility or clarity.
- **Accent-MOS**: it measures the accentedness of the utterance, i.e., how likely the accent of the speaker belongs to one of the African accents,
- **Accent-Match**: Here, participants from the same country as the reference accent rated the closeness of the generated utterance to their own accent,
- **Country-Match**: participants from the same country as the reference accent rated the closeness of the generated utterance to their own country,
- **Gender-Match**: participants rated the closeness of the generated utterance's gender to the reference gender indicated for that reference speaker,
- **Preference test**: we performed a preference test[27] on our accented models using a modified Hugging Face TTS Arena tool (Vaibhav et al., 2024). Here, volunteers were shown a pair of utterances generated by two TTS models among the finetuned TTS models at every turn, then they were asked to select the more natural utterance between the pair.

A.3 Experiments

A.3.1 TTS models

The experiments in the chapter are based on two state-of-the-art, open-source, end-to-end TTS models, VITS (Kim et al., 2021) and XTTS (Casanova et al., 2024). We describe the TTS models and other modifications of the models that were trained or finetuned in this work.

[26] https://speech.intron.health
[27] https://huggingface.co/spaces/eniolaa/AfroTTS-Evaluation

- VITS: It is an end-to-end model that adopts variational inference augmented with normalising flows and an adversarial training process. It also uses a stochastic duration predictor to generate a human-like rhythm of speech. The VITS model contains 86.6 M parameters. It was trained on the VCTK dataset (44 hours of 109 native English speakers) for 500 k iterations.
- XTTS: It is a recent multilingual TTS system with cross-language voice cloning capabilities comprising three modules; a VQ-VAE module, a GPT module, and an audio decoder. We used version 2 of the XTTS model which contains 750 M parameters. The model had been trained on a dataset of over 16,000 hours comprising 16 languages,[28] including English.
- VITS-FT: A trained VITS model finetuned on the Afro-TTS processed dataset.
- XTTS-FT: A trained XTTS model finetuned on the training dataset. In our experiments, we only fine-tuned the GPT module.
- VITS-O: A randomly initialised VITS model trained from scratch on the training dataset to validate the quality of the data for TTS experiments
- VITS-EXT: A modified VITS model that takes an external 256-dimensional L2-normalised speaker embedding vector as speaker conditioning. The speaker embeddings were extracted from Resemblyzer,[29] a speaker embedding extractor. The weights of the speaker embedding layer was re-initialised during fine-tuning.

A.3.2 Training and finetuning hyper-parameters

All the VITS-like models namely, VITS-O, VITS-FT, and VITS-EXT, were trained with mixed-precision training on 4 GPUs for 350k iterations, with a global batch size of 64. We used the VITS implementation from the authors for training and inference, with the default training hyper-parameters.[30] Similarly, XTTS was fine-tuned on 1 GPU using the Coqui library[31] for about 250 iterations with default finetuning hyper-parameters, using a batch size of 2 and gradient accumulation of 128. The VITS-like models take phonemes (without stress markers) interspersed with a blank token as input and generate audio at a 16 kHz sampling rate while the XTTS-FT model was fine-tuned at 24 kHz by upsampling the 16 kHz processed training data. In addition, to mitigate regional bias during model training, we computed the average duration per speaker on the original dataset, and then we duplicated all the samples corresponding to a speaker to equal the average duration of 10 minutes if the speaker's duration was less than the average.

At inference time, we set the noise scale of the VITS-like models to 0.667, with the duration noise scale set to 0.8 following the original work. For speaker and accent representation, we either used the learned speaker embedding, the speaker embedding extracted from the speaker extractor model (for VITS-E), or the test utterance (for XTTS models). XTTS-based models generate speech at 24 kHz while VITS-like models generate speech at 16 kHz. Therefore, all the generated samples were resampled to 16 kHz for evaluation. Additionally, statistical significance tests were carried out on the presented results using

[28] https://docs.coqui.ai/en/latest/models/xtts.html
[29] https://github.com/resemble-ai/Resemblyzer
[30] https://github.com/jaywalnut310/vits
[31] https://github.com/coqui-ai/TTS

the bootstrap method. The best model or best models with similar scores among the trained and fine-tuned models are highlighted in bold.

A.4 Results and discussion

Aggregated results for subjective and objective evaluations are presented in Table A.2 and Table A.3. Furthermore, in Tables A.4 and A.5, we show a breakdown of the ratings for the best fine-tuned model where the raters' country/accent match the utterance country/accent enabling us to measure the similarity in accent, country, and gender of the utterances. Here, we only show results for three counties (NG, KE, and ZA) where we had significant evaluators to measure the similarity in accent, country, and gender of the utterances. Also, these three countries make up 97.8% of the dataset.

We generated 735 utterances per model, along with 162 speaker-interpolated utterances from VITS-based models. Also, 427 unique participants (64.6% female, 60 accents, from 9 countries) provided 26 974 human ratings representing roughly 5 ratings per utterance. Additionally, 26 116 ratings were provided across models and 858 ratings were provided for speaker-interpolated utterances. *GT denoised* are the Ground truth (GT) reference test samples.

Table A.2: Subjective evaluation results (with 95% confidence interval) for pre-trained and fine-tuned TTS models. Mean opinion score (MOS), naturalness MOS (Nat-MOS), accentedness MOS (Accent-MOS), and preference rankings are reported.

Model	MOS	Nat-MOS	Accent-MOS	Preference Ranking
GT denoised	3.75 ± 0.04	4.55 ± 0.03	4.49 ± 0.03	-
VITS	3.80 ± 0.04	2.84 ± 0.06	1.81 ± 0.05	-
XTTS	3.92 ± 0.04	3.31 ± 0.06	2.31 ± 0.06	-
Trained/fine-tuned models				
VITS-O	3.02 ± 0.05	4.00 ± 0.04	4.02 ± 0.04	-
VITS-FT	3.33 ± 0.04	4.18 ± 0.04	4.16 ± 0.04	(1192) 2rd
VITS-EXT	3.14 ± 0.05	4.07 ± 0.04	4.07 ± 0.04	(1168) 3rd
XTTS-FT	$\mathbf{3.77 \pm 0.04}$	$\mathbf{4.39 \pm 0.03}$	$\mathbf{4.35 \pm 0.03}$	**(1235) 1st**

Table A.3: Objective evaluation results (with 95% confidence interval) for pre-trained and fine-tuned TTS models. Model-based MOS (WV-MOS), model-based naturalness (NISQA), cosine similarity (cos-sim), accentedness MOS (Accent-MOS), and word error rate (% WER) are reported.

Model	WV-MOS	NISQA	cos-sim	WER (%)
GT denoised	2.85 ± 0.04	4.55 ± 0.03	-	-
VITS	4.26 ± 0.02	3.84 ± 0.04	-	32.75
XTTS	3.71 ± 0.02	3.00 ± 0.03	0.828	13.81
Trained/fine-tuned models				
VITS-O	2.93 ± 0.03	2.94 ± 0.02	0.834	66.77
VITS-FT	3.03 ± 0.03	2.97 ± 0.03	0.834	51.77
VITS-EXT	3.02 ± 0.03	$\mathbf{3.06 \pm 0.03}$	**0.914**	57.31
XTTS-FT	$\mathbf{3.31 \pm 0.03}$	$\mathbf{3.07 \pm 0.04}$	**0.889**	**19.20**

A.4.1 Naturalness and overall quality results

Table A.2 shows that although participants rated the pre-trained models higher in overall quality (MOS), the best fine-tuned model (XTTS-FT) was rated 1.08 MOS points higher than its pre-trained version, and 1.55 MOS points higher than the VITS baseline in terms of naturalness, probably due to the better pronunciation of African named entities in the reference text. Our results demonstrate that we are indeed able to generate speech that is more relatable to an African audience. However, model-based quality metrics (WV-MOS and NISQA) in Table A.3 show inverse results, i.e., that pre-trained models are better, possibly because underlying models lack exposure to African-sounding speech.

A.4.2 Accentedness and speaker similarity results

Most significantly in Table A.2, participants rated XTTS-FT only 0.14 MOS points lower in Accentedness (Accent-MOS) than the GT in contrast to the XTTS baseline which was rated 2.18 MOS points below the GT. This validates that our approach generates natural-sounding accented speech, bridging the current gap in the representation of African voices in speech synthesis. Speaker similarity results (cos-sim) also showed that generated utterances from fine-tuned models are closer to reference utterances than generated utterances from pre-trained models.

A.4.3 Preference scores

Preference test scores in Table A.2 show that raters prefer utterances generated by XTTS-FT. Preference test scores aligned well with MOS metrics where the XTTS-FT model outperformed the VITS-based models. Also, the pre-trained XTTS model had a higher MOS score on average than the pre-trained VITS model likely because of its multilingual pretraining.

A.4.4 Intelligibility

Utterances by XTTS models had lower WER than VITS-based models perhaps as a result of the greater diversity and quantity (363x) of its pretraining data. A higher WER after fine-tuning of XTTS was also due to noise artifacts in the Afro-TTS dataset.

A.4.5 Regional diversity considerations

Table A.4: Country-level MOS results (with 95% confidence interval) for the best model (XTTS-FT) showing naturalness (Nat-MOS), accentedness (A-MOS), country-match (Country-M), and accent-match (Accent-M).

	Nat-MOS	A-MOS	Country-M	Accent-M
KE	4.44 ± 0.09	4.37 ± 0.09	3.93 ± 0.52	3.90 ± 0.48
NG	4.37 ± 0.04	4.33 ± 0.04	4.24 ± 0.11	3.54 ± 0.15
ZA	4.46 ± 0.11	4.47 ± 0.10	3.40 ± 0.49	2.93 ± 0.53

Table A.4 reveals that although the naturalness and accentedness of generated utterances from our best model are close to the GT, regional differences surface. South Africans (ZA) rated South African-generated utterances lower in Accent-Match than West Africans

Table A.5: Accent-level: Best model (XTTS-FT) results (with 95% confidence interval) for ratings where the utterance accent is matched to the rater's accent.

Accent	Country-Match	Accent-Match
Afrikaans	4.80 ± 0.56	2.67 ± 5.17
Hausa	4.23 ± 0.22	3.93 ± 0.25
Igbo	4.12 ± 1.37	2.25 ± 1.24
Swahili	3.89 ± 0.56	3.77 ± 0.57
Tswana	3.75 ± 3.01	3.50 ± 1.59
Yoruba	4.25 ± 0.14	3.30 ± 0.20
Zulu	3.09 ± 0.59	2.88 ± 0.62

(NG) rated the generated utterances with West African accents. Although most participants agreed that the generated utterances represent the reference country from Table A.5, the generated accents do not always match the reference accent, e.g., generated speech in Afrikaans accent may sound like Zulu, and Igbo generated accent may sound like Yoruba. Indeed, in multilingual countries like NG and ZA, speaker accents are difficult to classify into binary accent classes (Owodunni et al., 2024), as many speakers have dual accents. Notably, although East African speakers have lower representation in the dataset compared to Southern Africans, East Africans (e.g., Swahili) generally rated Accent-Match higher than South Africans (e.g., Zulu, Afrikaans). These inconsistencies may reflect the accent imbalance in the Afro-TTS dataset and require further investigation.

A.5 Limitations

Although we included 86 distinct African accents, this is a small fraction of more than 3,000 languages and accents across the continent. Additionally, imbalanced accent representation in our dataset may yield biased performance favoring majority accents. Lastly, we acknowledge the privacy risk of releasing multi-speaker TTS systems that mimic voices in the source data increasing the risk of voice cloning or voice theft. We mitigate this by removing any speaker identifiers, making it more challenging to identify individuals. Finally, disentangling of speaker and accent characteristics is left for future work.

A.6 Conclusion

We developed an African-accented TTS system that achieves near-GT MOS for naturalness and accentedness using the pan-African TTS dataset, a 136-hour dataset containing 747 speakers with 86 African accents from 9 countries. Our work greatly improves the representation of African voices in speech synthesis and our best models have been open-sourced for research purposes.

B

Résumé étendu

B.1 Introduction

L'explosion des technologies numériques nous a rendu de plus en plus dépendants des appareils tels que les smartphones, les ordinateurs, les assistants personnels et les montres intelligentes pour communiquer, planifier des événements, travailler, etc. Ces appareils nécessitent un processus de collecte et de compréhension des demandes de l'utilisateur. Certains utilisent la parole comme forme d'interaction et sont équipés d'un système de reconnaissance automatique de la parole (ASR). Un système d'ASR convertit la parole enregistrée par un microphone en texte pour un traitement ultérieur de la demande. En général, l'ASR est un élément clé des services numériques nécessitant une transcription vocale tels que l'analyse de données de centres d'appels, la production automatisée de comptes-rendus de réunion ou de comptes-rendus cliniques, la transcription de podcasts, la traduction instantanée, l'apprentissage des langues, etc. Cela implique que la pertinence du service dépend fortement de la qualité de transcription du système d'ASR.

La transcription précise de la parole reste en général une tâche difficile, et la précision de l'ASR diminue davantage pour les langues peu ou moyennement dotées et dans les domaines d'application spécifiques où les données disponibles pour l'apprentissage ne sont pas assez volumineuses ou diversifiées. Néanmoins, cela reste une tâche importante, notamment afin de réduire la fracture numérique pour les handicapés, les personnes âgées, les locuteurs de langues peu dotées et même les locuteurs non-natifs de langues bien dotées.

B.2 Contexte

Ces vingts dernières années, le taux d'erreur des systèmes d'ASR s'est considérablement réduit, passant d'environ 50 % à moins de 20% pour la parole spontanée et encore moins pour la parole lue (Srivastav et al., 2023; Radford et al., 2023), ce qui les a rendus utilisables en pratique. Cette amélioration peut être attribuée à plusieurs facteurs, notamment la création de nouvelles architectures d'ASR, d'algorithmes d'apprentissage et d'inférence, de fonctions de coût, de jeux de données d'apprentissage et d'augmentation des données, entre autres.

Les données d'apprentissage doivent représenter plusieurs attributs vocaux. Quelques exemples d'attributs typiques de la parole incluent le débit de parole, l'intensité, l'intonation, l'état émotionnel, la réverbération et le bruit, entre autres.

Bien que de grandes quantités de données aient été conservées pour des applications

générales et dans des langues bien dotées comme l'anglais et le français, de nombreuses langues du monde et de nombreux domaines d'application manquent encore de données d'une ampleur suffisante pour obtenir des taux d'erreur d'ASR faibles (Ardila et al., 2020). Même si plusieurs de ces problèmes peuvent être résolus en collectant davantage de données, la collecte de données est coûteuse et prend du temps. Par conséquent, la plupart des pipelines d'apprentissage d'ASR incluent une augmentation des données pour créer un corpus d'apprentissage plus diversifié.

La méthode étudiée dans cette thèse consiste à augmenter un corpus d'apprentissage d'ASR constitué de données réelles par des données synthétiques générées par un système de synthèse de la parole (TTS) ou de conversion de voix (VC). Générer plus de données ne conduit pas toujours à des résultats optimaux en raison de deux problèmes majeurs : a) la distribution des données synthétiques peut ne pas être similaire à celle des données réelles, et b) il faut considérer ce qui manque aux données réelles, puis essayer de combler les trous dans la distribution des données réelles.

B.3 Obtention d'un jeu de données de TTS multi-locuteur à partir d'un jeu de données d'ASR

L'apprentissage des modèles de TTS nécessite des données vocales et textuelles appariées de haute qualité. Pour un modèle de TTS multi-locuteur, il est également nécessaire d'avoir de nombreux locuteurs avec leur variabilité propre en termes d'accent, de style d'expression, de débit de parole, etc. Ces variabilités sont nécessaires pour que le système de TTS puisse apprendre diverses caractéristiques des locuteurs qui lui permettront de générer des voix synthétiques convaincantes pour des locuteurs extérieurs à son ensemble d'apprentissage.

Les jeux de données récents pour l'apprentissage de l'ASR sont volumineux et contiennent des enregistrements d'un grand nombre de locuteurs, ce qui en fait de bons candidats pour l'apprentissage de modèles de TTS. Cependant, ils présentent des conditions d'enregistrement de moindre qualité (réverbération, bruit de fond, dispositif d'enregistrement variable, etc.) qui sont souhaitables pour l'ASR mais peuvent entraver leur adoption pour le TTS. Cela nécessite de filtrer les jeux de données d'ASR crowdsourcés disponibles pour les transformer en jeux de données de TTS de haute qualité.

Ici, nous nous concentrons sur la sélection automatique d'enregistrements de haute qualité, en utilisant Common Voice English (Ardila et al., 2020) comme exemple. En raison de la taille du jeu de données (1,4 million de phrases), nous exploitons un estimateur de qualité non-intrusif basé sur l'apprentissage profond, WV-MOS (Andreev et al., 2023) pour évaluer la qualité des enregistrements.

B.3.1 Common Voice

Common Voice (Ardila et al., 2020) est un jeu de données crowdsourcé massivement multilingue et multi-locuteur de parole transcrite pour la recherche et le développement de technologies vocales. Il est mis à disposition sous la licence Creative Commons Zero qui permet aux développeurs d'utiliser le corpus sans restrictions ni frais. Ce jeu de données est adapté à l'apprentissage de systèmes d'ASR (Rivière et al., 2020), cependant nous devons considérer ses propriétés indésirables telles que le bruit, la largeur de bande faible, les erreurs de prononciation, etc., pour l'apprentissage de TTS.

B.3.2 Méthodologie

La figure B.1 illustre le processus de filtrage du jeu de données. Nous avons estimé le score MOS de chaque phrase à l'aide de WV-MOS (Andreev et al., 2023), puis moyenné ces scores pour chaque locuteur et conservé uniquement les locuteurs dont le score est au-dessus d'un certain seuil. Pour déterminer le bon seuil, nous avons appris un modèle de TTS correspondant à chaque seuil.

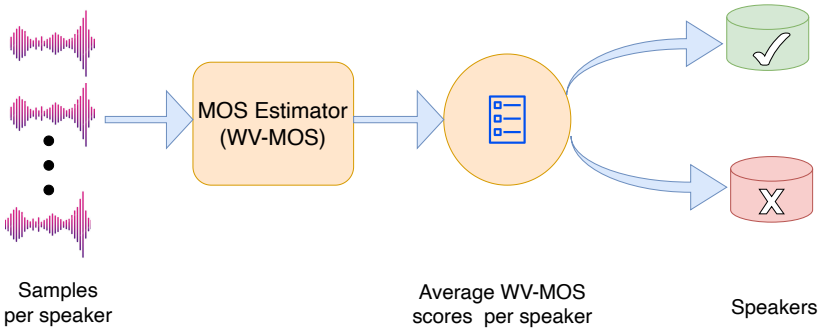


Figure B.1: Filtrage du jeu de données.

Comme les scores WV-MOS sont des nombres réels compris entre 1,0 et 5,0, nous avons choisi cinq seuils WV-MOS $\geq \{2.0, 3.0, 3.5, 3.8, 4.0\}$, où les seuils plus élevés indiquent une qualité supérieure. Ces seuils ont été sélectionnés pour équilibrer la taille des données et le nombre de locuteurs. Ces modèles ont également été comparés à un modèle de TTS de base appris sur l'ensemble des données non filtrées.

B.3.3 Évaluation

La procédure d'apprentissage et d'évaluation est illustrée dans la Figure B.2. Pour évaluer les modèles de manière objective et subjective, nous avons généré des voix synthétiques pour les locuteurs *vus* et *non-vus*. Pour une évaluation objective, nous mesurons la qualité sonore, la similarité du locuteur et l'intelligibilité des signaux générés en calculant le score WV-MOS moyen **(WV-MOS)**, la similarité cosinus **(cos-sim)** entre les plongements de locuteur du signal généré et de la référence et le taux d'erreur de caractère **(CER)**. Pour une évaluation subjective, nous mesurons la qualité globale **(MOS)**, la similarité du locuteur entre le signal généré et la référence **(S-MOS)** et l'intelligibilité (en utilisant des paires minimales).

B.3.4 Résultats

Le Tableau B.1 présente les résultats pour des locuteurs non-vus des modèles de TTS appris sur le jeu de données non-filtré (méthode de base), sur le jeu de données standard LibriTTS ou sur le jeu de données filtré de la plus haute qualité. Dans ce dernier cas, nous avons sélectionné tous les locuteurs avec un WV-MOS supérieur à 4.0 pour obtenir le jeu de données filtré WV-MOS $\geq$ 4.0-all, avec 4 469 locuteurs et une durée de 230,75 heures.

L'apprentissage sur WV-MOS $\geq$ 4.0-all aboutit à un score d'intelligibilité similaire à l'apprentissage sur LibriTTS, tandis que la méthode de base a un score d'intelligibilité significativement inférieur, indiquant que le bruit affecte l'intelligibilité. Cela montre également que le jeu de données WV-MOS $\geq$ 4.0-all est de haute qualité et peut être utilisé

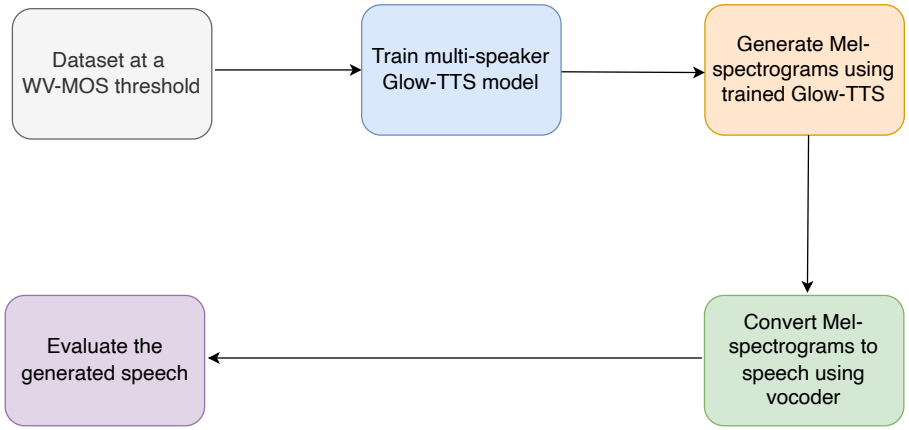


Figure B.2: Procédure d'apprentissage et d'évaluation de TTS.

Table B.1: Qualité perçue (MOS), similarité des locuteurs (S-MOS) et intelligibilité des signaux générés pour quatre locuteurs non-vus par des modèles TTS appris sur l'ensemble de données non-filtré, sur LibriTTS ou sur WV-MOS $\geq$ 4.0-all. Les chiffres en gras indiquent le meilleur système dans chaque ligne et les systèmes statistiquement équivalents à celui-ci. La synthèse par copie (VCTK-copy) fournit une borne supérieure de la similarité des locuteurs.

		Méthode de base	LibriTTS	WV-MOS $\geq$ 4.0-all	VCTK-copy
MOS	Homme	$2,44$	$3,15$	$\mathbf{3,70}$	-
	Femme	$2,26$	$\mathbf{3,38}$	$\mathbf{3,52}$	-
	Total	$2,35$	$3,26$	$\mathbf{3,61}$	-
WV-MOS		$3,63$	$\mathbf{3,75}$	$\mathbf{3,80}$	-
S-MOS	Homme	$\mathbf{2,92}$	$\mathbf{2,95}$	$\mathbf{3,02}$	$4,50$
	Femme	$2,46$	$2,53$	$\mathbf{2,73}$	$4,73$
	Total	$2,69$	$2,74$	$\mathbf{2,88}$	$4,61$
cos-sim		$0,831$	$\mathbf{0,861}$	$\mathbf{0,861}$	$0,869$
Intelligibilité		$0,72$	$\mathbf{0,82}$	$\mathbf{0,82}$	-

pour apprendre des modèles de TTS très intelligibles. En regardant les scores S-MOS, il apparaît que le score le plus élevé est obtenu par apprentissage sur WV-MOS $\geq$ 4.0-all. Les scores S-MOS masculins sont supérieurs aux scores S-MOS féminins pour les deux jeux de données, ce qui indique que les locuteurs masculins sont mieux modélisés. Nous constatons aussi un écart important en S-MOS entre le modèle de TTS appris sur WV-MOS $\geq$ 4.0-all et la borne supérieure VCTK-copy, ce qui laisse la place à de nouvelles améliorations dans la modélisation du locuteur. Enfin, bien que les scores WV-MOS pour les signaux générés par les systèmes appris sur WV-MOS $\geq$ 4.0-all et sur LibriTTS ne soient pas statistiquement différents, les évaluateurs ont systématiquement donné des scores MOS plus élevés aux premiers, avec une amélioration moyenne de 0,35 point MOS. Les locuteurs masculins se sont vus attribuer des scores MOS plus élevés, contrairement à LibriTTS où les voix féminines ont reçu des scores MOS plus élevés (conformément à Zen et al. (2019)).

B.3.5 Conclusion

Nous avons obtenu à partir du corpus Common Voice un jeu de données de haute qualité composé de 4 469 locuteurs et d'une durée de 230,75 heures adapté à l'apprentissage de modèles de TTS, qui surpasse même LibriTTS en termes de qualité subjective et de similarité des locuteurs. L'approche appliquée est générique et pourrait permettre la création de jeux de données d'apprentissage de TTS pour les langues pour lesquelles la création manuelle d'un jeu de données n'est pas financièrement viable.

B.4 Améliorer la diversité de la parole générée

Après avoir créé un jeu de données, nous nous concentrons sur le modèle de TTS dans cette partie. Nous examinons comment améliorer la contrôlabilité du modèle de TTS en concevant une architecture de modèle qui permet de générer des énoncés réalistes et diversifiés pour les locuteurs vus et non-vus. Pour cela, nous apprenons explicitement les distributions de la durée des phonèmes et de la hauteur, afin de pouvoir tirer des échantillons de ces distributions.

B.4.1 Méthodologie

La Figure B.3 illustre l'architecture proposée. Nous avons modifié l'architecture Glow-TTS comme suit. Premièrement, nous avons modifié le décodeur pour prendre en entrée le plongement de locuteur et la hauteur, ainsi que la représentation latente du texte. Deuxièmement, nous avons remplacé le modèle de durée déterministe par un modèle génératif de durée et ajouté un modèle de génératif de hauteur, tous deux basés sur une architecture de type Flow identique à celle du modèle de durée de VITS (Kim et al., 2021). Ces modèles génératifs aident à générer un rythme et une intonations plus réalistes.

B.4.2 Évaluation

Outre le modèle classique Glow-TTS comme référence, nous évaluons ses variantes Glow-TTS-STD (avec le modèle génératif de durée) et GlowTTS-STDP (avec les modèles génératifs de durée et de hauteur). Les trois modèles ont été appris sur le jeu de données de la Partie B.3. Pour l'évaluation subjective, nous avons utilisé plusieurs scores MOS évaluant le caractère naturel (N-MOS), la similarité de locuteur (S-MOS), le style de lecture (NR-MOS) et la diversité (D-MOS). Pour l'évaluation objective, nous avons calculé le score MOS de qualité (WV-MOS) et la similarité cosinus entre les plongements de locuteur du signal généré et de la référence (cos-sim).

B.4.3 Résultats

Les résultats dans le Tableau B.2 montrent que GlowTTS-STD et GlowTTS-STDP améliorent le caractère naturel et la similarité de locuteur. De plus, la Figure B.4 montre une tendance de diversité (D-MOS) croissante, GlowTTS-STDP étant capable de générer des signaux plus divers que Glow-TTS sans perte de naturel. Cela indique que les modèles génératifs de durée et de hauteur contribuent tous deux à augmenter la diversité des signaux générés.

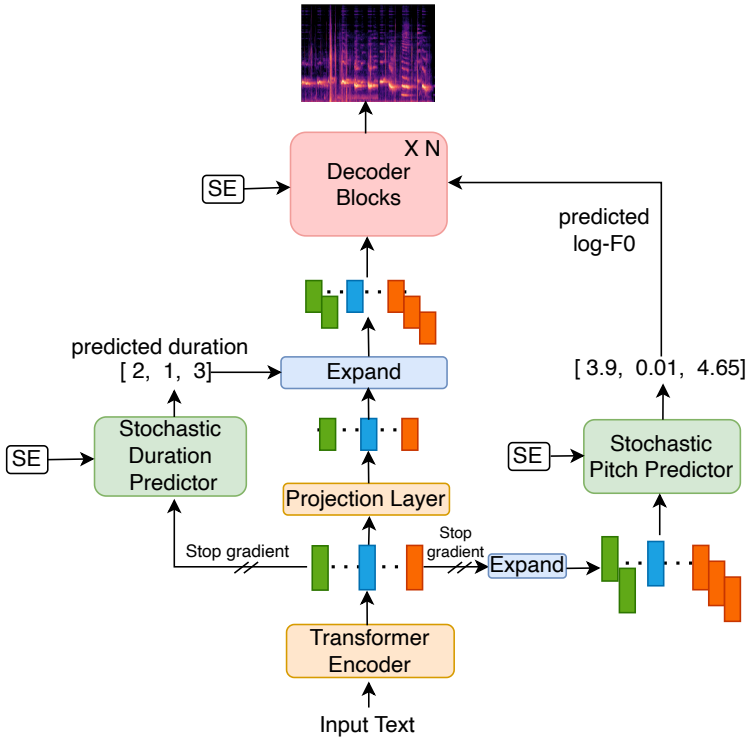


Figure B.3: Architecture Glow-TTS améliorée proposée, comprenant un modèle génératif de durée et un modèle génératif de hauteur. SE désigne le plongement de locuteur utilisé pour conditionner le modèle.

Table B.2: N-MOS, NR-MOS et S-MOS des signaux générés pour 6 locuteurs non-vus par la méthode Glow-TTS de base, GlowTTS-STD, GlowTTS-STDP et VCTK-copy. Les nombres en gras désignent le meilleur système dans chaque ligne et les systèmes statistiquement équivalents.

		Glow-TTS	GlowTTS-STD	GlowTTS-STDP	VCTK-copy
N-MOS	Homme	$2,92$	$3,11$	$\mathbf{3,40}$	$4,00$
	Femme	$3,35$	$\mathbf{3,51}$	$\mathbf{3,51}$	$4,40$
	Total	$3,13$	$3,31$	$\mathbf{3,45}$	$4,21$
WV-MOS		$4,11$	$\mathbf{4,17}$	$\mathbf{4,18}$	$4,32$
NR-MOS	Homme	$2,91$	$\mathbf{3,24}$	$\mathbf{3,25}$	-
	Femme	$2,98$	$3,28$	$\mathbf{3,53}$	-
	Total	$2,95$	$3,26$	$\mathbf{3,39}$	-
S-MOS	Homme	$2,43$	$2,90$	$\mathbf{3,10}$	$4,71$
	Femme	$2,14$	$\mathbf{3,07}$	$2,91$	$4,85$
	Total	$2,26$	$\mathbf{2,98}$	$\mathbf{2,99}$	$4,79$
cos-sim		$0,8287$	$\mathbf{0,8364}$	$\mathbf{0,8319}$	$0,8121$

B.4.4 Conclusion

Dans cette partie, le naturel et la diversité de la parole générée ont été améliorés avec succès en intégrant des modèles génératifs de durée et de hauteur dans le modèle GlowTTS. Cette

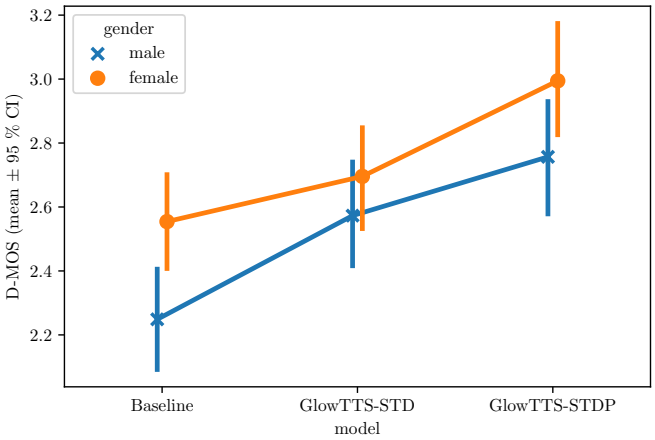


Figure B.4: D-MOS des signaux générés par les modèles évalués pour les locuteurs non-vus masculins et féminins.

amélioration nous permet de manipuler le rythme et la hauteur des signaux générés sans perte de naturel.

B.5 Évaluation de l'impact de la diversité des données synthétiques sur la performance d'ASR

Après avoir créé un jeu de données de TTS de haute qualité et construit un modèle de TTS capable de générer des énoncés naturels et diversifiés, nous nous concentrons sur l'utilisation de TTS et VC pour l'augmentation des données d'apprentissage d'ASR. TTS et VC ont gagné en popularité comme méthodes d'augmentation de données ces dernières années en raison de l'amélioration rapide de la qualité de la parole générée. L'utilisation de données synthétiques en complément de données réelles a déjà été explorée pour des tâches telles que la détection de mots-clés (Hu et al., 2022), l'ASR avec alternance codique (Sharma et al., 2020), l'augmentation des données pour les langues peu dotées (Zevallos et al., 2022; Casanova et al., 2022; Shahnawazuddin et al., 2020; Baas and Kamper, 2022), l'adaptation linguistique (Robinson et al., 2022) et l'amélioration de la reconnaissance de séquences numériques (Peyser et al., 2019), entre autres. Notre objectif est d'évaluer l'impact individuel et conjoint des différents attributs vocaux sur la performance d'ASR.

B.5.1 Méthodologie et évaluation

Nous utilisons les modèles de TTS/VC Glow-TTS, GlowTTS-STD et GlowTTS-STDP de la Partie B.4 pour générer les données synthétiques et évaluons l'impact sur l'apprentissage de l'ASR sur le jeu de données Common Voice (Ardila et al., 2020). Nous utilisons le modèle d'ASR Conformer-Transducer de l'état de l'art (Gulati et al., 2020) avec une branche de classification temporelle connexionniste (CTC) auxiliaire (Graves et al., 2006).

Nous évaluons séparément l'impact de la diversité phonétique, de la diversité de locuteurs, de la diversité de la durée et de la diversité de la hauteur des signaux générés par TTS/VC. Pour chacune de ces évaluations, nous sélectionnons la méthode qui fournit les

meilleurs résultats, puis nous combinons ces méthodes dans une expérience finale. Ces méthodes sont les suivantes :

- diversité phonétique : textes sélectionnés itérativement afin de minimiser la divergence de Kullback-Leibler avec la distribution naturelle des textes ;
- diversité de locuteurs : 2 457 locuteurs non-vus sélectionnés itérativement selon un critère de distance médiane aux locuteurs déjà sélectionnés ;
- diversité de durée : durée des phonèmes augmentée en utilisant des plongements de locuteur aléatoires parmi les 2 457 locuteurs non-vus ;
- diversité de hauteur : non incluse car GlowTTS-STDP ne réduit pas significativement le WER par rapport à Glow-TTS ;
- modèle : TTS uniquement, car VC ne réduit pas significativement le WER par rapport à TTS.

Nous augmentons de manière itérative la taille des données synthétiques d'un facteur 2 jusqu'à ce que nous ne constations pas de réduction significative du CER et du WER.

B.5.2 Résultats

Le Tableau B.3 montre que la combinaison des méthodes d'augmentation de données réduit le CER et le WER par rapport à un modèle d'ASR appris sur données réelles. Par exemple, cela réduit le CER et le WER du modèle d'ASR appris sur 100 heures de données réelles de 4 % et 1 % relatifs. L'augmentation de la taille des données TTS ajoutées aux 50 heures de données réelles continue d'améliorer le modèle d'ASR jusqu'à 200 heures, indiquant une limite à l'amélioration pouvant être obtenue avec les données synthétiques.

Table B.3: CER et WER des modèles d'ASR appris sur 50 ou 100 heures de données réelles combinées à 50, 100, 200 ou 400 heures de données générées par TTS à l'aide des méthodes ayant fourni les meilleurs résultats pour chaque attribut. Les nombres en gras désignent la meilleure combinaison et les systèmes statistiquement équivalents .

Données	CER (%)	WER (%)
50r	25.01	46.30
100r	22.10	40.81
50r_50tts	21.17	40.39
50r_100tts	20.05	38.32
50r_200tts	**19.70**	**37.07**
50r_400tts	**19.70**	37.31

B.5.3 Conclusion

Nous avons exploré différentes méthodes pour augmenter la diversité des données générées par TTS et VC pour l'apprentissage de l'ASR en combinaison avec des données réelles. En combinant les meilleures méthodes pour chaque attribut, nous avons pu réduire considérablement le CER et le WER. Ce travail souligne l'impact de la diversité des signaux d'apprentissage sur la performance d'ASR, et la possibilité d'assurer cette diversité à l'aide

de signaux générés par TTS/VC avec les bons locuteurs et les bons textes et notre méthode d'augmentation de durée. Cependant, l'impact de la diversité n'est pas illimité, et trop de diversité peut saturer les performances du système d'ASR.

B.6 Conclusion finale

Cette thèse a exploré l'amélioration de la performance des modèles ASR par apprentissage sur des données réelles combinées à des données synthétiques générées par TTS ou VC. Cela nous a amené à aborder trois tâches: l'obtention d'un jeu de données de TTS de haute qualité à partir d'un jeu de données crowdsourcé pour l'ASR, la création d'un système de TTS intégrant des modèles génératifs de durée et de hauteur et l'évaluation de l'impact de la diversité des données générées sur la performance d'ASR. Pour la première tâche, nous avons réussi à obtenir un jeu de données de 4 469 locuteurs et 230,75 heures dont la qualité est comparable à celle de LibriTTS (Zen et al., 2019). Pour la deuxième tâche, nous avons construit un modèle TTS/VC (GlowTTS-STDP) capable de générer des signaux plus naturels et plus diversifiés par rapport au modèle de base Glow-TTS grâce à des modèles génératifs de durée et de hauteur. Pour la dernière tâche, nous avons montré l'impact positif de la diversité des données générées par TTS pour chacun des attributs de la parole, à l'exception de la hauteur. Les données générées par VC n'ont cependant pas eu d'impact sur l'amélioration de la performance d'ASR.

Bibliography

Abe, M., Nakamura, S., Shikano, K., and Kuwabara, H. (1990). Voice conversion through vector quantization. *Journal of the Acoustical Society of Japan (E)*, 11(2):71–76.

Adedeji, A., Joshi, S., and Doohan, B. (2024). The Sound of Healthcare: Improving medical transcription ASR accuracy with large language models. *arXiv preprint arXiv:2402.07658*.

Aertsen, A. and Johannesma, P. I. (1980). Spectro-temporal receptive fields of auditory neurons in the grassfrog: I. Characterization of tonal and natural stimuli. *Biological Cybernetics*, 38(4):223–234.

Andreev, P., Alanov, A., Ivanov, O., and Vetrov, D. (2023). HiFi++: A unified framework for bandwidth extension and speech enhancement. In *International Conference on Acoustics, Speech and Signal Processing*, pages 1–5.

Ardila, R., Branson, M., Davis, K., Kohler, M., Meyer, J., Henretty, M., Morais, R., Saunders, L., Tyers, F., and Weber, G. (2020). Common Voice: A massively-multilingual speech corpus. In *Twelfth Language Resources and Evaluation Conference*, pages 4218–4222.

Arık, S. Ö., Chrzanowski, M., Coates, A., Diamos, G., Gibiansky, A., Kang, Y., Li, X., Miller, J., Ng, A., Raiman, J., Sengupta, S., and Shoeybi, M. (2017). Deep Voice: Real-time neural text-to-speech. In *International Conference on Machine Learning*, pages 195–204.

Austin, J. L. (1975). *How to do things with words*. Harvard University Press.

Baali, M., Hayashi, T., Mubarak, H., Maiti, S., Watanabe, S., El-Hajj, W., and Ali, A. (2023). Unsupervised data selection for TTS: Using Arabic broadcast news as a case study. *arXiv preprint arXiv:2301.09099*.

Baas, M. and Kamper, H. (2022). Voice conversion can improve ASR in very low-resource settings. In *Interspeech*, pages 3513–3517.

Badlani, R., Łańcucki, A., Shih, K. J., Valle, R., Ping, W., and Catanzaro, B. (2022). One TTS alignment to rule them all. In *International Conference on Acoustics, Speech and Signal Processing*, pages 6092–6096.

Baevski, A., Zhou, Y., Mohamed, A., and Auli, M. (2020). wav2vec 2.0: A framework for self-supervised learning of speech representations. In *Advances in Neural Information Processing Systems*, volume 33, pages 12449–12460.

Bahl, L. R., Jelinek, F., and Mercer, R. L. (1983). A maximum likelihood approach to continuous speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(2):179–190.

Balestri, M., Pacchiotti, A., Quazza, S., Salza, P. L., and Sandri, S. (1999). Choose the best to modify the least: A new generation concatenative synthesis system. In *Sixth European Conference on Speech Communication and Technology*.

Barker, J. P., Marxer, R., Vincent, E., and Watanabe, S. (2017). The CHiME challenges: Robust speech recognition in everyday environments. *New era for robust speech recognition: Exploiting deep learning*, pages 327–344.

Barrault, L., Chung, Y.-A., Meglioli, M. C., Dale, D., Dong, N., Duquenne, P.-A., Elsahar, H., Gong, H., Heffernan, K., Hoffman, J., Klaiber, C., Li, P., Licht, D., Maillard, J., Rakotoarison, A., Sadagopan, K. R., Wenzek, G., Ye, E., Akula, B., Chen, P.-J., Hachem, N. E., Ellis, B., Gonzalez, G. M., Haaheim, J., Hansanti, P., Howes, R., Huang, B., Hwang, M.-J., Inaguma, H., Jain, S., Kalbassi, E., Kallet, A., Kulikov, I., Lam, J., Li, D., Ma, X., Mavlyutov, R., Peloquin, B., Ramadan, M., Ramakrishnan, A., Sun, A., Tran, K., Tran, T., Tufanov, I., Vogeti, V., Wood, C., Yang, Y., Yu, B., Andrews, P., Balioglu, C., Costa-jussà, M. R., Celebi, O., Elbayad, M., Gao, C., Guzmán, F., Kao, J., Lee, A., Mourachko, A., Pino, J., Popuri, S., Ropers, C., Saleem, S., Schwenk, H., Tomasello, P., Wang, C., Wang, J., and Wang, S. (2023). SeamlessM4T-Massively Multilingual & Multimodal Machine Translation. *arXiv preprint arXiv:2308.11596*.

Baskar, M. K., Watanabe, S., Astudillo, R., Hori, T., Burget, L., and ernocký, J. (2019). Semi-supervised sequence-to-sequence ASR using unpaired speech and text. In *Interspeech*, pages 3790–3794.

Benoît, C., Grice, M., and Hazan, V. (1996). The SUS test: A method for the assessment of text-to-speech synthesis intelligibility using semantically unpredictable sentences. *Speech Communication*, 18(4):381–392.

Benzeghiba, M., De Mori, R., Deroo, O., Dupont, S., Erbes, T., Jouvet, D., Fissore, L., Laface, P., Mertins, A., Ris, C., Rose, R., Tyagi, V., and Wellekens, C. (2007). Automatic speech recognition and speech variability: A review. *Speech Communication*, 49(10-11):763–786.

Bernard, M. and Titeux, H. (2021). Phonemizer: Text to phones transcription for multiple languages in python. *Journal of Open Source Software*, 6(68):3958.

Berrebbi, D., Collobert, R., Jaitly, N., and Likhomanenko, T. (2023). More speaking or more speakers? In *International Conference on Acoustics, Speech and Signal Processing*, pages 1–5.

Bińkowski, M., Donahue, J., Dieleman, S., Clark, A., Elsen, E., Casagrande, N., Cobo, L. C., and Simonyan, K. (2019). High fidelity speech synthesis with adversarial networks. In *International Conference on Learning Representations*.

Bisani, M. and Ney, H. (2004). Bootstrap estimates for confidence intervals in ASR performance evaluation. In *International Conference on Acoustics, Speech and Signal Processing*, volume 1, pages I–409.

Black, A. W. and Muthukumar, P. K. (2015). Random forests for statistical speech synthesis. In *Interspeech*, pages 1211–1215.

Bock, J. K. and Mazzella, J. R. (1983). Intonational marking of given and new information: Some consequences for comprehension. *Memory & Cognition*, 11:64–76.

Bolinger, D. and Bolinger, D. L. M. (1986). *Intonation and its parts: Melody in spoken English*. Stanford University Press.

Borsos, Z., Marinier, R., Vincent, D., Kharitonov, E., Pietquin, O., Sharifi, M., Roblek, D., Teboul, O., Grangier, D., Tagliasacchi, M., and Zeghidour, N. (2023). AudioLM: A language modeling approach to audio generation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*.

Bourlard, H. A. and Morgan, N. (1994). *Connectionist Speech Recognition: A Hybrid Approach*. Kluwer Academic Publishers.

Bu, H., Du, J., Na, X., Wu, B., and Zheng, H. (2017). AISHELL-1: An open-source Mandarin speech corpus and a speech recognition baseline. In *20th Conference of the Oriental Chapter of the International Coordinating Committee on Speech Databases and Speech I/O Systems and Assessment (O-COCOSDA)*, pages 1–5.

Casanova, E., Davis, K., Gölge, E., Göknar, G., Gulea, I., Hart, L., Aljafari, A., Meyer, J., Morais, R., Olayemi, S., and Weber, J. (2024). XTTS: a massively multilingual zero-shot text-to-speech model. In *Interspeech*, pages 4978–4982.

Casanova, E., Shulby, C., Gölge, E., Müller, N. M., de Oliveira, F. S., Candido Jr., A., da Silva Soares, A., Aluisio, S. M., and Ponti, M. A. (2021). SC-GlowTTS: An efficient zero-shot multi-speaker text-to-speech model. In *Interspeech*, pages 3645–3649.

Casanova, E., Shulby, C., Korolev, A., Junior, A. C., da Silva Soares, A., Aluísio, S., and Ponti, M. A. (2023). ASR data augmentation in low-resource settings using cross-lingual multi-speaker TTS and cross-lingual voice conversion. In *Interspeech*, pages 1244–1248.

Casanova, E., Weber, J., Shulby, C. D., Junior, A. C., Gölge, E., and Ponti, M. A. (2022). YourTTS: Towards zero-shot multi-speaker TTS and zero-shot voice conversion for everyone. In *International Conference on Machine Learning*, pages 2709–2720.

Chan, C. H., Qian, K., Zhang, Y., and Hasegawa-Johnson, M. (2022). Speechsplit2.0: Unsupervised speech disentanglement for voice conversion without tuning autoencoder bottlenecks. In *International Conference on Acoustics, Speech and Signal Processing*, pages 6332–6336.

Chang, J.-H. R., Shrivastava, A., Koppula, H., Zhang, X., and Tuzel, O. (2022). Style Equalization: Unsupervised learning of controllable generative sequence models. In *International Conference on Machine Learning*, pages 2917–2937.

Chen, G., Chai, S., Wang, G., Du, J., Zhang, W.-Q., Weng, C., Su, D., Povey, D., Trmal, J., Zhang, J., Jin, M., Khudanpur, S., Watanabe, S., Zhao, S., Zou, W., Li, X., Yao, X., Wang, Y., Wang, Y., You, Z., and Yan, Z. (2021). GigaSpeech: An evolving, multi-domain ASR corpus with 10,000 hours of transcribed audio. In *Interspeech*, pages 4376–4380.

Chen, J., Lu, C., Chenli, B., Zhu, J., and Tian, T. (2020). VFlow: More expressive generative flows with variational data augmentation. In *International Conference on Machine Learning*, pages 1660–1669.

Chen, W.-Y., Chen, S.-H., and Lin, C.-J. (1996). A speech recognition method based on the sequential multi-layer perceptrons. *Neural Networks*, 9(4):655–669.

Childers, D. G. (1995). Glottal source modeling for voice conversion. *Speech Communication*, 16(2):127–138.

Chow, Y., Dunham, M., Kimball, O., Krasner, M., Kubala, G., Makhoul, J., Price, P., Roucos, S., and Schwartz, R. (1987). BYBLOS: The BBN continuous speech recognition system. In *ICASSP'87. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 12, pages 89–92. IEEE.

Chung, Y. J. and Un, C. K. (1996). Multilayer perceptrons for state-dependent weightings of HMM likelihoods. *Speech Communication*, 18(1):79–89.

Conkie, A. (1999). Robust unit selection system for speech synthesis. In *137th meeting of the Acoustical Society of America*, page 978.

Cooper, E., Huang, W.-C., Toda, T., and Yamagishi, J. (2022). Generalization ability of MOS prediction networks. In *International Conference on Acoustics, Speech and Signal Processing*, pages 8442–8446.

Cosentino, J., Pariente, M., Cornell, S., Deleforge, A., and Vincent, E. (2020). LibriMix: An open-source dataset for generalizable speech separation. *arXiv preprint arXiv:2005.11262*.

Davis, K. H., Biddulph, R., and Balashek, S. (1952). Automatic recognition of spoken digits. *The Journal of the Acoustical Society of America*, 24(6):637–642.

Davis, S. and Mermelstein, P. (1980). Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28(4):357–366.

De Cheveigné, A. and Kawahara, H. (2002). YIN, a fundamental frequency estimator for speech and music. *The Journal of the Acoustical Society of America*, 111(4):1917–1930.

Défossez, A., Copet, J., Synnaeve, G., and Adi, Y. (2023). High fidelity neural audio compression. *Transactions on Machine Learning Research*. Featured Certification, Reproducibility Certification.

Dehak, N., Torres-Carrasquillo, P. A., Reynolds, D. A., and Dehak, R. (2011). Language recognition via i-vectors and dimensionality reduction. In *Interspeech*, pages 857–860.

Delattre, P., Olsen, C., and Poenack, E. (1962). A comparative study of declarative intonation in American English and Spanish. *Hispania*, 45(2):233–241.

Dinh, L., Sohl-Dickstein, J., and Bengio, S. (2016). Density estimation using Real NVP. In *International Conference on Learning Representations*.

Donahue, J., Dieleman, S., Binkowski, M., Elsen, E., and Simonyan, K. (2020). End-to-end adversarial text-to-speech. In *International Conference on Learning Representations*.

Dong, L., Xu, S., and Xu, B. (2018). Speech-Transformer: A no-recurrence sequence-to-sequence model for speech recognition. In *International Conference on Acoustics, Speech and Signal Processing*, pages 5884–5888.

Donovan, R. E. (1996). *Trainable speech synthesis*. PhD thesis, Cambridge University.

Durkan, C., Bekasov, A., Murray, I., and Papamakarios, G. (2019). Neural spline flows. *Advances in Neural Information Processing Systems*, 32.

Défossez, A., Synnaeve, G., and Adi, Y. (2020). Real time speech enhancement in the waveform domain. In *Interspeech*, pages 3291–3295.

Erro, D., Moreno, A., and Bonafonte, A. (2009). INCA algorithm for training voice conversion systems from nonparallel corpora. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(5):944–953.

Evain, S., Nguyen, H., Le, H., Boito, M. Z., Mdhaffar, S., Alisamir, S., Tong, Z., Tomashenko, N., Dinarelli, M., Parcollet, T., Allauzen, A., Estève, Y., Lecouteux, B., Portet, F., Rossato, S., Ringeval, F., Schwab, D., and Besacier, L. (2021). LeBenchmark: A reproducible framework for assessing self-supervised representation learning from speech. In *Interspeech*, pages 1439–1443.

Fry, D. B. (1959). Theoretical aspects of mechanical speech recognition. *Journal of the British Institution of Radio Engineers*, 19(4):211–218.

Gârbacea, C., van den Oord, A., Li, Y., Lim, F. S., Luebs, A., Vinyals, O., and Walters, T. C. (2019). Low bit-rate speech coding with VQ-VAE and a WaveNet decoder. In *International Conference on Acoustics, Speech and Signal Processing*, pages 735–739.

Gerhard, D. (2003). Pitch extraction and fundamental frequency: history and current techniques. *Technical Report TR-CS*.

Gillick, L. and Cox, S. J. (1989). Some statistical issues in the comparison of speech recognition algorithms. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 532–535.

Godfrey, J. J., Holliman, E. C., and McDaniel, J. (1992). SWITCHBOARD: Telephone speech corpus for research and development. In *International Conference on Acoustics, Speech and Signal Processing*, volume 1, pages 517–520. IEEE Computer Society.

Graves, A. (2012). Sequence transduction with recurrent neural networks. In *ICML Workshop on Representation Learning*.

Graves, A., Fernández, S., Gomez, F., and Schmidhuber, J. (2006). Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *International Conference on Machine Learning*, pages 369–376.

Griffin, D. and Lim, J. (1984). Signal estimation from modified short-time Fourier transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 32(2):236–243.

Griffin, D. and Lim, J. (1985). A new model-based speech analysis/synthesis system. In *ICASSP'85. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 10, pages 513–516. IEEE.

Gulati, A., Qin, J., Chiu, C.-C., Parmar, N., Zhang, Y., Yu, J., Han, W., Wang, S., Zhang, Z., Wu, Y., and Pang, R. (2020). Conformer: Convolution-augmented Transformer for speech recognition. In *Interspeech*, pages 5036–5040.

Günsel, B., Sezgin, C., and Krajewski, J. (2013). Sleepiness detection from speech by perceptual features. In *International Conference on Acoustics, Speech and Signal Processing*, pages 788–792.

Hafiz, N. F., Mashohor, S., Shazril, M., Rasid, M. F. A., and Ali, A. (2023). Comparison of Mel frequency cepstral coefficient (MFCC) and Mel spectrogram techniques to classify industrial machine sound. In *15th International Conference on Software, Knowledge, Information Management and Applications*, pages 273–278.

Han, W., Chan, C.-F., Choy, C.-S., and Pun, K.-P. (2006). An efficient MFCC extraction method in speech recognition. In *IEEE International Symposium on Circuits and Systems (ISCAS)*, page 4.

Haulcy, R. and Glass, J. (2021). Classifying alzheimer's disease using audio and text-based representations of speech. *Frontiers in Psychology*, 11:624137.

Hayashi, T., Watanabe, S., Zhang, Y., Toda, T., Hori, T., Astudillo, R., and Takeda, K. (2018). Back-translation-style data augmentation for end-to-end ASR. In *IEEE Spoken Language Technology Workshop*, pages 426–433.

He, Y., Sainath, T. N., Prabhavalkar, R., McGraw, I., Alvarez, R., Zhao, D., Rybach, D., Kannan, A., Wu, Y., Pang, R., Liang, Q., Bhatia, D., Shangguan, Y., Li, B., Pundak, G., Sim, K. C., Bagby, T., Chang, S.-y., Rao, K., and Gruenstein, A. (2019). Streaming end-to-end speech recognition for mobile devices. In *International Conference on Acoustics, Speech and Signal Processing*, pages 6381–6385.

Helander, E., Schwarz, J., Nurminen, J., Silen, H., and Gabbouj, M. (2008). On the impact of alignment on voice conversion performance. In *Interspeech*, pages 1453–1456.

Helander, E., Silén, H., Virtanen, T., and Gabbouj, M. (2011). Voice conversion using dynamic kernel partial least squares regression. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(3):806–817.

Helander, E., Virtanen, T., Nurminen, J., and Gabbouj, M. (2010). Voice conversion using partial least squares regression. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(5):912–921.

Hermjakob, U., May, J., and Knight, K. (2018). Out-of-the-box universal Romanization tool uroman. In Liu, F. and Solorio, T., editors, *Proceedings of ACL 2018, System Demonstrations*, pages 13–18, Melbourne, Australia. Association for Computational Linguistics.

Hill, D., Manzara, L., and Schock, C. (1995). Real-time articulatory speech-synthesis-by-rules. In *14th Annual International Voice Technologies Applications Conference of the American Voice I/O Society*, volume 95, pages 11–14.

Ho, J., Chen, X., Srinivas, A., Duan, Y., and Abbeel, P. (2019). Flow++: Improving flow-based generative models with variational dequantization and architecture design. In *International Conference on Machine Learning*, pages 2722–2730.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Hodge, M. M. and Gotzke, C. L. (2011). Minimal pair distinctions and intelligibility in preschool children with and without speech sound disorders. *Clinical Linguistics & Phonetics*, 25(10):853–863.

Hoffmann, R. (2019). The long way of speech synthesis. In *Third International Workshop on the History of Speech Communication Research*.

Homer, D. (1955). Fundamentals of speech synthesis. *Journal of the Audio Engineering Society*, 3:170–185.

Hsu, W.-N., Tsai, Y.-H. H., Bolte, B., Salakhutdinov, R., and Mohamed, A. (2021). HuBERT: How much can a bad teacher benefit ASR pre-training? In *International Conference on Acoustics, Speech and Signal Processing*, pages 6533–6537.

Hsu, W.-N., Zhang, Y., Weiss, R. J., Chung, Y.-A., Wang, Y., Wu, Y., and Glass, J. (2019). Disentangling correlated speaker and noise for speech synthesis via data augmentation and adversarial factorization. In *International Conference on Acoustics, Speech and Signal Processing*, pages 5901–5905.

Hu, T.-Y., Armandpour, M., Shrivastava, A., Chang, J.-H. R., Koppula, H., and Tuzel, O. (2022). Synt++: Utilizing imperfect synthetic data to improve speech recognition. In *International Conference on Acoustics, Speech and Signal Processing*, pages 7682–7686.

Hunt, A. J. and Black, A. W. (1996). Unit selection in a concatenative speech synthesis system using a large speech database. In *International Conference on Acoustics, Speech and Signal Processing*, volume 1, pages 373–376.

Hunt, M. J. (1988). Evaluating the performance of connected-word speech recognition systems. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 457–460 vol.1.

Ingber, W. (1982). Linguistic communication and speech acts. *The Philosophical Review*, 91(1):134–138.

IPA (1999). *Handbook of the International Phonetic Association: A guide to the use of the International Phonetic Alphabet*. Cambridge University Press.

Ito, K. and Johnson, L. (2017). The LJ Speech dataset. https://keithito.com/LJ-Speech-Dataset/.

ITU-R (2003). *ITU-R Recommendation BS.1534-1. Method for the subjective assessment of intermediate quality level of audio systems*. ITU.

Jacob Kahn, M. R., Zheng, W., Kharitonov, E., Xu, Q., Mazaré, P.-E., Karadayi, J., Liptchinsky, V., Collobert, R., Fuegen, C., Likhomanenko, T., Synnaeve, G., Joulin, A., Mohamed, A., and Dupoux, E. (2020). Libri-Light: A benchmark for ASR with limited or no supervision. In *International Conference on Acoustics, Speech and Signal Processing*, pages 7669–7673.

Jain, S. M. (2022). Hugging Face. In *Introduction to Transformers for NLP: With The Hugging Face Library and Models to Solve Problems*, pages 51–67. Springer.

Jelinek, F., Bahl, L., and Mercer, R. (1975). Design of a linguistic statistical decoder for the recognition of continuous speech. *IEEE Transactions on Information Theory*, 21(3):250–256.

Jonathan Shen, R. P., Weiss, R. J., Schuster, M., Jaitly, N., Yang, Z., Chen, Z., Zhang, Y., Wang, Y., Skerry-Ryan, R., Saurous, R. A., Agiomyrgiannakis, Y., and Wu, Y. (2018). Natural TTS synthesis by conditioning WaveNet on Mel spectrogram predictions. In *International Conference on Acoustics, Speech and Signal Processing*, pages 4779–4783.

Junqua, J.-C. (1993). The lombard reflex and its role on human listeners and automatic speech recognizers. *The Journal of the Acoustical Society of America*, 93(1):510–524.

Kalchbrenner, N., Elsen, E., Simonyan, K., Noury, S., Casagrande, N., Lockhart, E., Stimberg, F., Oord, A., Dieleman, S., and Kavukcuoglu, K. (2018). Efficient neural audio synthesis. In *International Conference on Machine Learning*, pages 2410–2419.

Kastner, K., Santos, J. F., Bengio, Y., and Courville, A. (2019). Representation mixing for TTS synthesis. In *International Conference on Acoustics, Speech and Signal Processing*, pages 5906–5910.

Kawahara, H. (2006). STRAIGHT, exploitation of the other aspect of VOCODER: Perceptually isomorphic decomposition of speech sounds. *Acoustical science and technology*, 27(6):349–353.

Kearns, J. (2014). Librivox: Free public domain audiobooks. *Reference Reviews*, 28(1):7–8.

Kim, C., Misra, A., Chin, K., Hughes, T., Narayanan, A., Sainath, T. N., and Bacchiani, M. (2017). Generation of large-scale simulated utterances in virtual rooms to train deep-neural networks for far-field speech recognition in Google Home. In *Interspeech*, pages 379–383.

Kim, J., Kim, S., Kong, J., and Yoon, S. (2020). Glow-TTS: A generative flow for text-to-speech via monotonic alignment search. In *Advances in Neural Information Processing Systems*, volume 33, pages 8067–8077.

Kim, J., Kong, J., and Son, J. (2021). Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech. In *International Conference on Machine Learning*, pages 5530–5540.

Kim, S., Shih, K., badlani, r., Santos, J. F., Bakhturina, E., Desta, M., Valle, R., Yoon, S., and Catanzaro, B. (2024). P-Flow: A fast and data-efficient zero-shot TTS through speech prompting. In *Advances in Neural Information Processing Systems*, volume 36, pages 74213–74228.

King, S. (2014). Measuring a decade of progress in text-to-speech. *Loquens*, 1(1):e006–e006.

Kingma, D. P. and Dhariwal, P. (2018). Glow: Generative flow with invertible 1x1 convolutions. *Advances in Neural Information Processing Systems*, 31.

Kirkland, A., Mehta, S., Lameris, H., Henter, G. E., Székely, E., and Gustafson, J. (2023). Stuck in the MOS pit: A critical analysis of MOS test methodology in TTS evaluation. In *12th ISCA Speech Synthesis Workshop*, pages 41–47.

Klatt, D. H. (1980). Software for a cascade/parallel formant synthesizer. *The Journal of the Acoustical Society of America*, 67(3):971–995.

Kluender, K. R., Diehl, R. L., and Wright, B. A. (1988). Vowel-length differences before voiced and voiceless consonants: An auditory explanation. *Journal of Phonetics*, 16(2):153–169.

Ko, T., Peddinti, V., Povey, D., and Khudanpur, S. (2015). Audio augmentation for speech recognition. In *Interspeech*, pages 3586–3589.

Ko, T., Peddinti, V., Povey, D., Seltzer, M. L., and Khudanpur, S. (2017a). A study on data augmentation of reverberant speech for robust speech recognition. In *International Conference on Acoustics, Speech and Signal Processing*, pages 5220–5224.

Ko, T., Peddinti, V., Povey, D., Seltzer, M. L., and Khudanpur, S. (2017b). A study on data augmentation of reverberant speech for robust speech recognition. In *International Conference on Acoustics, Speech and Signal Processing*, pages 5220–5224.

Kobyzev, I., Prince, S. J., and Brubaker, M. A. (2021). Normalizing Flows: An introduction and review of current methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(11):3964–3979.

Kochanski, G., Grabe, E., Coleman, J., and Rosner, B. (2005). Loudness predicts prominence: Fundamental frequency lends little. *The Journal of the Acoustical Society of America*, 118(2):1038–1054.

Köhn, A., Stegen, F., and Baumann, T. (2016). Mining the spoken Wikipedia for speech data and beyond. In *Tenth International Conference on Language Resources and Evaluation*, pages 4644–4647.

Kong, J., Kim, J., and Bae, J. (2020). HiFi-GAN: Generative adversarial networks for efficient and high fidelity speech synthesis. *Advances in Neural Information Processing Systems*, 33:17022–17033.

Kraaij, W., Hain, T., Lincoln, M., and Post, W. (2005). The AMI meeting corpus. In *International Conference on Methods and Techniques in Behavioral Research*.

Krauss, R. M., Freyberg, R., and Morsella, E. (2002). Inferring speakers physical attributes from their voices. *Journal of experimental social psychology*, 38(6):618–625.

Kriman, S., Beliaev, S., Ginsburg, B., Huang, J., Kuchaiev, O., Lavrukhin, V., Leary, R., Li, J., and Zhang, Y. (2020). QuartzNet: Deep automatic speech recognition with 1D time-channel separable convolutions. In *International Conference on Acoustics, Speech and Signal Processing*, pages 6124–6128.

Kubichek, R. (1993). Mel-cepstral distance measure for objective speech quality assessment. In *IEEE Pacific Rim Conference on Communications Computers and Signal Processing*, volume 1, pages 125–128.

Kumar, K., Kumar, R., De Boissiere, T., Gestin, L., Teoh, W. Z., Sotelo, J., De Brebisson, A., Bengio, Y., and Courville, A. C. (2019). MelGAN: Generative adversarial networks for conditional waveform synthesis. *Advances in Neural Information Processing Systems*, 32.

Lago, C. (2005). *Race, culture and counselling*. McGraw-Hill Education (UK).

Łańcucki, A. (2021). Fastpitch: Parallel text-to-speech with pitch prediction. In *International Conference on Acoustics, Speech and Signal Processing*, pages 6588–6592.

Lang, K. J., Waibel, A. H., and Hinton, G. E. (1990). A time-delay neural network architecture for isolated word recognition. *Neural Networks*, 3(1):23–43.

Le, M., Vyas, A., Shi, B., Karrer, B., Sari, L., Moritz, R., Williamson, M., Manohar, V., Adi, Y., Mahadeokar, J., and Hsu, W.-N. (2023). Voicebox: Text-guided multilingual universal speech generation at scale. In *Advances in Neural Information Processing Systems*, volume 36, pages 14005–14034.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521(7553):436–444.

Lee, C.-H., Clements, M. A., Dusan, S., Fosler-Lussier, E., Johnson, K., Juang, B.-H., and Rabiner, L. R. (2007). An overview on automatic speech attribute transcription (ASAT). In *Interspeech*, pages 1825–1828.

Lee, J. and Watanabe, S. (2021). Intermediate loss regularization for CTC-based speech recognition. In *International Conference on Acoustics, Speech and Signal Processing*, pages 6224–6228.

Lee, K.-F., Hon, H.-W., and Reddy, R. (1990). An overview of the SPHINX speech recognition system. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 38(1):35–45.

Lee, S.-G., Ping, W., Ginsburg, B., Catanzaro, B., and Yoon, S. (2023). BigVGAN: A universal neural vocoder with large-scale training. In *International Conference on Learning Representations*.

Lee, Y., Yang, J., and Jung, K. (2022). Varianceflow: High-quality and controllable text-to-speech using variance information via normalizing flow. In *International Conference on Acoustics, Speech and Signal Processing*, pages 7477–7481.

Li, J., Lavrukhin, V., Ginsburg, B., Leary, R., Kuchaiev, O., Cohen, J. M., Nguyen, H., and Gadde, R. T. (2019). Jasper: An end-to-end convolutional neural acoustic model. In *Interspeech*, pages 71–75.

Li, J., Tu, W., and Xiao, L. (2023). FreeVC: Towards high-quality text-free one-shot voice conversion. In *International Conference on Acoustics, Speech and Signal Processing*, pages 1–5.

Li, Y. A., Zare, A., and Mesgarani, N. (2021). StarGANv2-VC: A diverse, unsupervised, non-parallel framework for natural-sounding voice conversion. In *Interspeech*, pages 1349–1353.

Liberman, M. and Cieri, C. (1998). The creation, distribution and use of linguistic data: the case of the linguistic data consortium. In *Language Resources and Evaluation Conference*, pages 159–166.

Liu, H., Kong, Q., Tian, Q., Zhao, Y., Wang, D., Huang, C., and Wang, Y. (2021). VoiceFixer: Toward general speech restoration with neural vocoder. *arXiv preprint arXiv:2109.13731*.

Liu, L., Jiang, H., He, P., Chen, W., Liu, X., Gao, J., and Han, J. (2020). On the variance of the adaptive learning rate and beyond. In *International Conference on Learning Representations*.

Lo, C.-C., Fu, S.-W., Huang, W.-C., Wang, X., Yamagishi, J., Tsao, Y., and Wang, H.-M. (2019). MOSNet: Deep learning-based objective assessment for voice conversion. In *Interspeech*, pages 1541–1545.

Loizou, P. C. (2011). Speech quality assessment. In *Multimedia Analysis, Processing and Communications*, pages 623–654. Springer.

Lorenzo-Trueba, J., Yamagishi, J., Toda, T., Saito, D., Villavicencio, F., Kinnunen, T., and Ling, Z. (2018). The Voice Conversion Challenge 2018: Promoting development of parallel and nonparallel methods. In *Odyssey*, pages 195–202.

Lu, Y., Li, Z., He, D., Sun, Z., Dong, B., Qin, T., Wang, L., and Liu, T.-y. (2020). Understanding and improving Transformer from a multi-particle dynamic system point of view. In *ICLR 2020 Workshop on Integration of Deep Neural Models and Differential Equations*.

Lui, M. and Baldwin, T. (2012). langid.py: An off-the-shelf language identification tool. In *ACL 2012 System Demonstrations*, pages 25–30.

Luong, H.-T., Wang, X., Yamagishi, J., and Nishizawa, N. (2019). Training multi-speaker neural text-to-speech systems using speaker-imbalanced speech corpora. In *Interspeech*, pages 1303–1307.

Lux, F., Tilli, P., Meyer, S., and Vu, N. T. (2023). Controllable generation of artificial speaker embeddings through discovery of principal directions. In *Interspeech*, pages 4788–4792.

Maeda, S. (1979). An articulatory model of the tongue based on a statistical analysis. *The Journal of the Acoustical Society of America*, 65(S1):S22–S22.

Maekawa, K., Koiso, H., Furui, S., and Isahara, H. (2000). Spontaneous speech corpus of Japanese. In *Language Resources and Evaluation Conference*, volume 6, pages 1–5.

Mairal, R. and Gil, J. (2006). *Linguistic Universals*. Cambridge University Press.

Malik, M., Malik, M. K., Mehmood, K., and Makhdoom, I. (2021). Automatic speech recognition: A survey. *Multimedia Tools and Applications*, 80:9411–9457.

Mauch, M. and Dixon, S. (2014). pYIN: A fundamental frequency estimator using probabilistic threshold distributions. In *International Conference on Acoustics, Speech and Signal Processing*, pages 659–663.

McCree, A. V. and Barnwell, T. P. (1995). A mixed excitation LPC vocoder model for low bit rate speech coding. *IEEE Transactions on Speech and audio Processing*, 3(4):242–250.

McInnes, L., Healy, J., Saul, N., and Großberger, L. (2018). UMAP: Uniform manifold approximation and projection. *Journal of Open Source Software*, 3(29).

Merritt, T., Ezzerg, A., Biliński, P., Proszewska, M., Pokora, K., Barra-Chicote, R., and Korzekwa, D. (2022). Text-free non-parallel many-to-many voice conversion using normalising flow. In *International Conference on Acoustics, Speech and Signal Processing*, pages 6782–6786.

Miao, C., Liang, S., Chen, M., Ma, J., Wang, S., and Xiao, J. (2020). Flow-TTS: A non-autoregressive network for text to speech based on flow. In *International Conference on Acoustics, Speech and Signal Processing*, pages 7209–7213.

Miao, Y., Gowayyed, M., Na, X., Ko, T., Metze, F., and Waibel, A. (2016). An empirical exploration of CTC acoustic models. In *International Conference on Acoustics, Speech and Signal Processing*, pages 2623–2627.

Mimura, M., Ueno, S., Inaguma, H., Sakai, S., and Kawahara, T. (2018). Leveraging sequence-to-sequence speech synthesis for enhancing acoustic-to-word speech recognition. In *IEEE Spoken Language Technology Workshop*, pages 477–484.

Minixhofer, C., Klejch, O., and Bell, P. (2023). Evaluating and reducing the distance between synthetic and real speech distributions. In *Interspeech*, pages 2078–2082.

Mitleb, F. (1984). Vowel length contrast in Arabic and English: A spectrographic test. *Journal of Phonetics*, 12(3):229–235.

Mittag, G. and Möller, S. (2020). Deep learning based assessment of synthetic speech naturalness. In *Interspeech*, pages 1748–1752.

Morise, M., Yokomori, F., and Ozawa, K. (2016). WORLD: a vocoder-based high-quality speech synthesis system for real-time applications. *IEICE Transactions on Information and Systems*, 99(7):1877–1884.

Mullennix, J. W., Mateljan, K., and Payne, D. (1998). Effects of mismatches in talker and emotion on speech perception. *The Journal of the Acoustical Society of America*, 104(3_Supplement):1759–1759.

Nagrani, A., Chung, J. S., and Zisserman, A. (2017). VoxCeleb: A large-scale speaker identification dataset. In *Interspeech*, pages 2616–2620.

Najafian, M., Safavi, S., Weber, P., and Russell, M. J. (2016). Identification of British English regional accents using fusion of i-vector and multi-accent phonotactic systems. In *Odyssey*, pages 132–139.

Navarro, G. (2001). A guided tour to approximate string matching. *ACM Computing Surveys*, 33(1):31–88.

Obin, N., Veaux, C., and Lanchantin, P. (2012). Making sense of variations: Introducing alternatives in speech synthesis. In *Speech Prosody*.

Obin, N., Veaux, C., and Lanchantin, P. (2015). Exploiting alternatives for text-to-speech synthesis: From machine to human. In *Speech Prosody in Speech Synthesis: Modeling and generation of prosody for high quality and flexible speech synthesis*, pages 189–202. Springer.

Ogun, S., Colotte, V., and Vincent, E. (2023a). Can we use Common Voice to train a multi-speaker TTS system? In *IEEE Spoken Language Technology Workshop*, pages 900–905.

Ogun, S., Colotte, V., and Vincent, E. (2023b). Stochastic pitch prediction improves the diversity and naturalness of speech in Glow-TTS. In *Interspeech*, pages 4878–4882.

Ogun, S., Owodunni, A. T., Olatunji, T., Alese, E., Oladimeji, B., Afonja, T., Olaleye, K., Etori, N. A., and Adewumi, T. (2024). 1000 African Voices: Advancing inclusive multi-speaker multi-accent speech synthesis. *arXiv preprint arXiv:2406.11727*.

Olatunji, T., Afonja, T., Yadavalli, A., Emezue, C. C., Singh, S., Dossou, B. F., Osuchukwu, J., Osei, S., Tonja, A. L., Etori, N., and Mbataku, C. (2023). AfriSpeech-200: Pan-African accented speech dataset for clinical and general domain ASR. *Transactions of the Association for Computational Linguistics*, 11:1669–1685.

Oleksii Kuchaiev, J. L., Nguyen, H., Hrinchuk, O., Leary, R., Ginsburg, B., Kriman, S., Beliaev, S., Lavrukhin, V., Cook, J., Castonguay, P., Popova, M., Huang, J., and Cohen, J. M. (2019). NeMo: A toolkit for building AI applications using neural modules. *arXiv preprint arXiv:1909.09577*.

OpenSLR (2024). Open speech and language resources. https://www.openslr.org/. [Online; accessed 03-April-2024].

Owodunni, A., Yadavalli, A., Emezue, C., Olatunji, T., and Mbataku, C. (2024). AccentFold: A journey through African accents for zero-shot ASR adaptation to target accents. In *Findings of the ACL: EACL 2024*, pages 2146–2161.

Panayotov, V., Chen, G., Povey, D., and Khudanpur, S. (2015). LibriSpeech: An ASR corpus based on public domain audio books. In *International Conference on Acoustics, Speech and Signal Processing*, pages 5206–5210.

Pariente, M., Cornell, S., Cosentino, J., Sivasankaran, S., Tzinis, E., Heitkaemper, J., Olvera, M., Stöter, F.-R., Hu, M., Martín-Doñas, J. M., Ditter, D., Frank, A., Deleforge, A., and Vincent, E. (2020). Asteroid: The pytorch-based audio source separation toolkit for researchers. In *Interspeech*, pages 2637–2641.

Park, D. S., Chan, W., Zhang, Y., Chiu, C.-C., Zoph, B., Cubuk, E. D., and Le, Q. V. (2019). SpecAugment: A simple data augmentation method for automatic speech recognition. In *Interspeech*, pages 2613–2617.

Park, K. and Mulc, T. (2019). CSS10: A collection of single speaker speech datasets for 10 languages. In *Interspeech*, pages 1566–1570.

Patton, B., Agiomyrgiannakis, Y., Terry, M., Wilson, K., Saurous, R. A., and Sculley, D. (2016). AutoMOS: Learning a non-intrusive assessor of naturalness-of-speech. In *NIPS End-to-end Learning for Speech and Audio Processing Workshop*.

Paul, D. B. and Baker, J. (1992). The design for the Wall Street Journal-based CSR corpus. In *Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York, February 23-26, 1992*.

Pengcheng Guo, F. B., Chang, X., Hayashi, T., Higuchi, Y., Inaguma, H., Kamo, N., Li, C., Garcia-Romero, D., Shi, J., Shi, J., Watanabe, S., Wei, K., Zhang, W., and Zhang, Y. (2021). Recent developments on ESPnet toolkit boosted by Conformer. In *International Conference on Acoustics, Speech and Signal Processing*, pages 5874–5878.

Peyser, C., Zhang, H., Sainath, T. N., and Wu, Z. (2019). Improving performance of end-to-end ASR on numeric sequences. In *Interspeech*, pages 2185–2189.

Popov, V., Vovk, I., Gogoryan, V., Sadekova, T., and Kudinov, M. (2021). Grad-TTS: A diffusion probabilistic model for text-to-speech. In *International Conference on Machine Learning*, pages 8599–8608.

Prabhavalkar, R., Hori, T., Sainath, T. N., Schlüter, R., and Watanabe, S. (2024). End-to-end speech recognition: A survey. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 32:325–351.

Pradhan, S. S., Cole, R. A., and Ward, W. H. (2023). My Science Tutor (MyST)–A large corpus of children's conversational speech. *arXiv preprint arXiv:2309.13347*.

Pratap, V., Tjandra, A., Shi, B., Tomasello, P., Babu, A., Kundu, S., Elkahky, A., Ni, Z., Vyas, A., Fazel-Zarandi, M., Baevski, A., Adi, Y., Zhang, X., Hsu, W.-N., Conneau, A., and Auli, M. (2024). Scaling speech technology to 1,000+ languages. *Journal of Machine Learning Research*, 25(97):1–52.

Pratap, V., Xu, Q., Sriram, A., Synnaeve, G., and Collobert, R. (2020). MLS: A large-scale multilingual dataset for speech research. In *Interspeech*, pages 2757–2761.

Qian, K., Zhang, Y., Chang, S., Hasegawa-Johnson, M., and Cox, D. (2020). Unsupervised speech decomposition via triple information bottleneck. In *International Conference on Machine Learning*, pages 7836–7846.

Qian, K., Zhang, Y., Chang, S., Yang, X., and Hasegawa-Johnson, M. (2019). AutoVC: Zero-shot voice style transfer with only autoencoder loss. In *International Conference on Machine Learning*, pages 5210–5219.

Rabiner, L., Levinson, S., Rosenberg, A., and Wilpon, J. (1979). Speaker-independent recognition of isolated words using clustering techniques. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 27(4):336–349.

Radford, A., Kim, J. W., Xu, T., Brockman, G., McLeavey, C., and Sutskever, I. (2023). Robust speech recognition via large-scale weak supervision. In *International Conference on Machine Learning*, pages 28492–28518.

Ravanelli, M., Zhong, J., Pascual, S., Swietojanski, P., Monteiro, J., Trmal, J., and Bengio, Y. (2020). Multi-task self-supervised learning for robust speech recognition. In *International Conference on Acoustics, Speech and Signal Processing*, pages 6989–6993.

Ren, Y., Hu, C., Tan, X., Qin, T., Zhao, S., Zhao, Z., and Liu, T.-Y. (2020). FastSpeech 2: Fast and high-quality end-to-end text to speech. In *International Conference on Learning Representations*.

Ren, Y., Ruan, Y., Tan, X., Qin, T., Zhao, S., Zhao, Z., and Liu, T.-Y. (2019). Fastspeech: Fast, robust and controllable text to speech. In *Advances in Neural Information Processing Systems*, volume 32.

Ren, Y., Tan, X., Qin, T., Zhao, Z., and Liu, T.-Y. (2022). Revisiting over-smoothness in text to speech. In *60th Annual Meeting of the ACL (Volume 1: Long Papers)*, pages 8197–8213.

Reynolds, D. A., Quatieri, T. F., and Dunn, R. B. (2000). Speaker verification using adapted Gaussian mixture models. *Digital Signal Processing*, 10(1-3):19–41.

Ris, C. and Dupont, S. (2001). Assessing local noise level estimation methods: Application to noise robust ASR. *Speech Communication*, 34(1-2):141–158.

Rivière, M., Joulin, A., Mazaré, P.-E., and Dupoux, E. (2020). Unsupervised pretraining transfers well across languages. In *International Conference on Acoustics, Speech and Signal Processing*, pages 7414–7418.

Rix, A., Beerends, J., Hollier, M., and Hekstra, A. (2001). Perceptual evaluation of speech quality (PESQ)-a new method for speech quality assessment of telephone networks and codecs. In *International Conference on Acoustics, Speech, and Signal Processing*, volume 2, pages 749–752.

Robinson, N. R., Ogayo, P., Gangu, S. R., Mortensen, D. R., and Watanabe, S. (2022). When is TTS augmentation through a pivot language useful? In *Interspeech*, pages 3538–3542.

Rosenberg, A., Zhang, Y., Ramabhadran, B., Jia, Y., Moreno, P., Wu, Y., and Wu, Z. (2019). Speech recognition with augmented synthesized speech. In *IEEE Automatic Speech Recognition and Understanding Workshop*, pages 996–1002.

Rossenbach, N., Hilmes, B., and Schlüter, R. (2023). On the relevance of phoneme duration variability of synthesized training data for automatic speech recognition. In *IEEE Automatic Speech Recognition and Understanding Workshop*, pages 1–8.

Rousseau, A., Deléglise, P., and Estève, Y. (2012). TED-LIUM: An automatic speech recognition dedicated corpus. In *Language Resources and Evaluation Conference*, pages 125–129.

Rowberry, S. (2023). *The Early Development of Project Gutenberg c. 1970–2000*. Cambridge University Press.

Sahidullah, M. and Saha, G. (2012). Design, analysis and experimental evaluation of block based transformation in MFCC computation for speaker recognition. *Speech Communication*, 54(4):543–565.

Sakai, S. and Shu, H. (2005). A probabilistic approach to unit selection for corpus-based speech synthesis. In *Interspeech*, pages 81–84.

Scherer, K. R. (1972). Judging personality from voice: A cross-cultural approach to an old issue in interpersonal perception 1. *Journal of personality*, 40(2):191–210.

Schluter, R., Bezrukov, I., Wagner, H., and Ney, H. (2007). Gammatone features and feature combination for large vocabulary speech recognition. In *International Conference on Acoustics, Speech and Signal Processing*, volume 4, pages 649–652.

Schuller, B., Steidl, S., Batliner, A., Burkhardt, F., Devillers, L., MüLler, C., and Narayanan, S. (2013). Paralinguistics in speech and language — state-of-the-art and the challenge. *Computer Speech & Language*, 27(1):4–39.

Searle, J. R. (1962). Meaning and speech acts. *The Philosophical Review*, 71(4):423–432.

Shahnawazuddin, S., Adiga, N., Kumar, K., Poddar, A., and Ahmad, W. (2020). Voice conversion based data augmentation to improve children's speech recognition in limited data scenario. In *Interspeech*, pages 4382–4386.

Sharma, Y., Abraham, B., Taneja, K., and Jyothi, P. (2020). Improving low resource code-switched ASR using augmented code-switched TTS. In *Interspeech*, pages 4771–4775.

Shen, J., Pang, R., Weiss, R. J., Schuster, M., Jaitly, N., Yang, Z., Chen, Z., Zhang, Y., Wang, Y., Skerrv-Ryan, R., Saurous, R. A., Agiomvrgiannakis, Y., and Wu, Y. (2018). Natural TTS synthesis by conditioning Wavenet on Mel spectrogram predictions. In *International Conference on Acoustics, Speech and Signal Processing*, pages 4779–4783.

Shen, K., Yan, D., Hu, J., and Ye, Z. (2024). Non-intrusive speech quality assessment: A survey. *Neurocomputing*, 580:127471.

Shepstone, S. E., Tan, Z.-H., and Jensen, S. H. (2013). Audio-based age and gender identification to enhance the recommendation of TV content. *IEEE Transactions on Consumer Electronics*, 59(3):721–729.

Shih, K. J., Valle, R., Badlani, R., Łańcucki, A., Ping, W., and Catanzaro, B. (2021). RAD-TTS: Parallel flow-based TTS with robust alignment learning and diverse synthesis. In *ICML Workshop on Invertible Neural Networks, Normalizing Flows, and Explicit Likelihood Models*.

Siminyu, K. (2022). Mozilla Foundation - lessons from building for kinyarwanda on Common Voice. https://foundation.mozilla.org/en/blog/lessons-from-building-for-kinyarwanda-on-common-voice/. [Online; accessed 17-May-2024].

Sisman, B., Yamagishi, J., King, S., and Li, H. (2020). An overview of voice conversion and its challenges: From statistical modeling to deep learning. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:132–157.

Snyder, D., Chen, G., and Povey, D. (2015). MUSAN: A music, speech, and noise corpus. *arXiv preprint arXiv:1510.08484*.

Snyder, D., Garcia-Romero, D., Sell, G., Povey, D., and Khudanpur, S. (2018). X-vectors: Robust DNN embeddings for speaker recognition. In *International Conference on Acoustics, Speech and Signal Processing*, pages 5329–5333.

Solak, I. C. (2018). The M-AILABS speech dataset. https://www.caito.de/2019/01/03/the-m-ailabs-speech-dataset/. [Online; accessed 19-March-2024].

Song, J. H., Skoe, E., Banai, K., and Kraus, N. (2011). Perception of speech in noise: neural correlates. *Journal of Cognitive Neuroscience*, 23(9):2268–2279.

Song, K., Cong, J., Wang, X., Zhang, Y., Xie, L., Jiang, N., and Wu, H. (2022). Robust MelGAN: A robust universal neural vocoder for high-fidelity TTS. In *13th International Symposium on Chinese Spoken Language Processing (ISCSLP)*, pages 71–75.

Sperber, M., Niehues, J., Neubig, G., Stüker, S., and Waibel, A. (2018). Self-attentional acoustic models. In *Interspeech*, pages 3723–3727.

Srivastav, V., Majumdar, S., Koluguri, N., Moumen, A., and Gandhi, S. (2023). Open automatic speech recognition leaderboard. https://huggingface.co/spaces/hf-audio/open_asr_leaderboard.

Stanton, D., Shannon, M., Mariooryad, S., Skerry-Ryan, R., Battenberg, E., Bagby, T., and Kao, D. (2022). Speaker generation. In *International Conference on Acoustics, Speech and Signal Processing*, pages 7897–7901.

Stevens, S. S., Volkmann, J., and Newman, E. B. (1937). A scale for the measurement of the psychological magnitude pitch. *The Journal of the Acoustical Society of America*, 8(3):185–190.

Su, H., Hu, T.-Y., Koppula, H. S., Vemulapalli, R., Chang, J.-H. R., Yang, K., Mantena, G. V., and Tuzel, O. (2024). Corpus synthesis for zero-shot ASR domain adaptation using large language models. In *International Conference on Acoustics, Speech and Signal Processing*, pages 12326–12330.

Sun, G., Zhang, Y., Weiss, R. J., Cao, Y., Zen, H., Rosenberg, A., Ramabhadran, B., and Wu, Y. (2020). Generating diverse and natural text-to-speech samples using a quantized fine-grained VAE and autoregressive prosody prior. In *International Conference on Acoustics, Speech and Signal Processing*, pages 6699–6703.

Sun, L., Li, K., Wang, H., Kang, S., and Meng, H. (2016). Phonetic posteriorgrams for many-to-one voice conversion without parallel data training. In *IEEE International Conference on Multimedia and Expo*, pages 1–6. IEEE.

Sundermann, D. and Ney, H. (2003). VTLN-based voice conversion. In *3rd IEEE International Symposium on Signal Processing and Information Technology*, pages 556–559.

Sweta, S., Babu, J. M., Palempati, A., and Kanhe, A. (2022). Cepstral coefficient-based gender classification using audio signals. In *International Conference on Sustainable Advanced Computing*, pages 81–90.

Szymański, P., Żelasko, P., Morzy, M., Szymczak, A., Żyła-Hoppe, M., Banaszczak, J., Augustyniak, L., Mizgajski, J., and Carmiel, Y. (2020). WER we are and WER we think we are. In *Findings of the ACL: EMNLP 2020*, pages 3290–3295.

Takashima, R., Takiguchi, T., and Ariki, Y. (2012). Exemplar-based voice conversion in noisy environment. In *IEEE Spoken Language Technology Workshop*, pages 313–317. IEEE.

Tan, X., Chen, J., Liu, H., Cong, J., Zhang, C., Liu, Y., Wang, X., Leng, Y., Yi, Y., He, L., Soong, F., Qin, T., Zhao, S., and Liu, T.-Y. (2024). NaturalSpeech: End-to-end text-to-speech synthesis with human-level quality. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46:4234–4245.

Tan, X., Qin, T., Soong, F., and Liu, T.-Y. (2021). A survey on neural speech synthesis. *arXiv preprint arXiv:2106.15561*.

Tappert, C. C., Dixon, N., Rabinowitz, A., and Chapman, W. (1971). Automatic recognition of continuous speech utilizing dynamic segmentation, dual classification, sequential decoding and error recovery. *Rome Air Dev. Cen, Rome, NY, Tech. Report TR-71*, 146.

Taylor, P. (2009). Text-to-speech synthesis.

Teshima, T., Ishikawa, I., Tojo, K., Oono, K., Ikeda, M., and Sugiyama, M. (2020). Coupling-based invertible neural networks are universal diffeomorphism approximators. *Advances in Neural Information Processing Systems*, 33:3362–3373.

Tjandra, A., Sakti, S., and Nakamura, S. (2017). Listening while speaking: Speech chain by deep learning. In *IEEE Automatic Speech Recognition and Understanding Workshop*, pages 301–308.

Tjandra, A., Sakti, S., and Nakamura, S. (2018). Machine Speech Chain with One-shot Speaker Adaptation. In *Interspeech*, pages 887–891.

Tjandra, A., Sakti, S., and Nakamura, S. (2020). Machine speech chain. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:976–989.

Toda, T., Black, A. W., and Tokuda, K. (2007). Voice conversion based on maximum-likelihood estimation of spectral parameter trajectory. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(8):2222–2235.

Toda, T. and Tokuda, K. (2007). A speech parameter generation algorithm considering global variance for HMM-based speech synthesis. *IEICE Transactions on Information and Systems*, 90(5):816–824.

Tokuda, K., Kobayashi, T., Masuko, T., and Imai, S. (1994). Mel-generalized cepstral analysis - a unified approach to speech spectral estimation. In *ICSLP*.

Tokuda, K., Zen, H., and Black, A. W. (2002). An HMM-based speech synthesis system applied to English. In *IEEE Speech Synthesis Workshop*, pages 227–230.

Toshniwal, S., Kannan, A., Chiu, C.-C., Wu, Y., Sainath, T. N., and Livescu, K. (2018). A comparison of techniques for language model integration in encoder-decoder speech recognition. In *IEEE Spoken Language Technology Workshop*, pages 369–375.

Tüske, Z., Schlüter, R., and Ney, H. (2018). Acoustic modeling of speech waveform based on multi-resolution, neural network signal processing. In *International Conference on Acoustics, Speech and Signal Processing*, pages 4859–4863.

Vaibhav, S., Clémentine, F., Pouget, L., Lacombe, Y., and Gandhi, S. (2024). TTS Arena: Benchmarking text-to-speech models in the wild. https://huggingface.co/blog/arena-tts. [Online; accessed 1-March-2024].

Valentini-Botinhao, C., Wang, X., Takaki, S., and Yamagishi, J. (2016). Speech enhancement for a noise-robust text-to-speech synthesis system using deep recurrent neural networks. In *Interspeech*, pages 352–356.

Valle, R., Shih, K. J., Prenger, R., and Catanzaro, B. (2021). Flowtron: an autoregressive flow-based generative network for text-to-speech synthesis. In *International Conference on Learning Representations*.

van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., and Kavukcuoglu, K. (2016). WaveNet: A generative model for raw audio. In *9th ISCA Workshop on Speech Synthesis Workshop*, page 125.

van den Oord, A., Li, Y., and Vinyals, O. (2018). Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30.

Vinyals, O., Ravuri, S. V., and Povey, D. (2012). Revisiting recurrent neural networks for robust ASR. In *International Conference on Acoustics, Speech and Signal Processing*, pages 4085–4088.

Waibel, A., Hanazawa, T., Hinton, G., Shikano, K., and Lang, K. (1989). Phoneme recognition using time-delay neural networks. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37(3):328–339.

Waibel, A., Hanazawa, T., Hinton, G., Shikano, K., and Lang, K. J. (1995). Phoneme recognition using time-delay neural networks. *Backpropagation: Theory, Architectures, and Applications*, pages 35–61.

Wan, L., Wang, Q., Papir, A., and Moreno, I. L. (2018). Generalized end-to-end loss for speaker verification. In *International Conference on Acoustics, Speech and Signal Processing*, pages 4879–4883.

Wang, C., Chen, S., Wu, Y., Zhang, Z., Zhou, L., Liu, S., Chen, Z., Liu, Y., Wang, H., Li, J., He, L., Zhao, S., and Wei, F. (2023). Neural codec language models are zero-shot text to speech synthesizers. *arXiv preprint arXiv:2301.02111*.

Wang, C., Riviere, M., Lee, A., Wu, A., Talnikar, C., Haziza, D., Williamson, M., Pino, J., and Dupoux, E. (2021a). VoxPopuli: A large-scale multilingual speech corpus for representation learning, semi-supervised learning and interpretation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 993–1003, Online. Association for Computational Linguistics.

Wang, D., Deng, L., Yeung, Y. T., Chen, X., Liu, X., and Meng, H. (2021b). VQMIVC: Vector quantization and mutual information-based unsupervised speech representation disentanglement for one-shot voice conversion. In *Interspeech*, pages 1344–1348.

Wang, G., Rosenberg, A., Chen, Z., Zhang, Y., Ramabhadran, B., Wu, Y., and Moreno, P. (2020a). Improving speech recognition using consistent predictions on synthesized

speech. In *International Conference on Acoustics, Speech and Signal Processing*, pages 7029–7033.

Wang, R., Ding, Y., Li, L., and Fan, C. (2020b). One-shot voice conversion using StarGAN. In *International Conference on Acoustics, Speech and Signal Processing*, pages 7729–7733.

Wang, S., Qian, Y., and Yu, K. (2017a). What does the speaker embedding encode? In *Interspeech*, pages 1497–1501.

Wang, X., Takaki, S., and Yamagishi, J. (2018a). Autoregressive neural F0 model for statistical parametric speech synthesis. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(8):1406–1419.

Wang, Y., Skerry-Ryan, R., Stanton, D., Wu, Y., Weiss, R. J., Jaitly, N., Yang, Z., Xiao, Y., Chen, Z., Bengio, S., Le, Q., Agiomyrgiannakis, Y., Clark, R., and Saurous, R. A. (2017b). Tacotron: Towards end-to-end speech synthesis. In *Interspeech*, pages 4006–4010.

Wang, Y., Stanton, D., Zhang, Y., Ryan, R.-S., Battenberg, E., Shor, J., Xiao, Y., Jia, Y., Ren, F., and Saurous, R. A. (2018b). Style Tokens: Unsupervised style modeling, control and transfer in end-to-end speech synthesis. In *International Conference on Machine Learning*, pages 5180–5189.

Watanabe, A., Takamichi, S., Saito, Y., Xin, D., and Saruwatari, H. (2023). Mid-attribute speaker generation using optimal-transport-based interpolation of Gaussian mixture models. In *International Conference on Acoustics, Speech and Signal Processing*, pages 1–5.

Watanabe, S., Hori, T., Kim, S., Hershey, J. R., and Hayashi, T. (2017). Hybrid CTC /attention architecture for end-to-end speech recognition. *Journal of Selected Topics in Signal Processing*, 11(8):1240–1253.

Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., Chi, E. H., Hashimoto, T., Vinyals, O., Liang, P., Dean, J., and Fedus, W. (2022). Emergent abilities of large language models. *Transactions on Machine Learning Research*.

Wikipedia contributors (2024). List of countries by english-speaking population — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=List_of_countries_by_English-speaking_population&oldid=1211690147. [Online; accessed 9-March-2024].

Wilhelms-Tricarico, R., Reichenbach, J., and Marple, G. (2013). The lessac technologies hybrid concatenated system for blizzard challenge 2013. In *Blizzard Challenge Workshop*.

Williams, I., Kannan, A., Aleksic, P. S., Rybach, D., and Sainath, T. N. (2018). Contextual speech recognition in end-to-end neural network systems using beam search. In *Interspeech*, pages 2227–2231.

Wu, Y., Zhang, R., and Rudnicky, A. (2007). Data selection for speech recognition. In *IEEE Automatic Speech Recognition and Understanding Workshop*, pages 562–565.

Wurm, L. H., Vakoch, D. A., Strasser, M. R., Calin-Jageman, R., and Ross, S. E. (2001). Speech perception and vocal expression of emotion. *Cognition & Emotion*, 15(6):831–852.

Xiong, R., Yang, Y., He, D., Zheng, K., Zheng, S., Xing, C., Zhang, H., Lan, Y., Wang, L., and Liu, T. (2020). On layer normalization in the Transformer architecture. In *International Conference on Machine Learning*, pages 10524–10533.

Xu, Y. (2011). Speech prosody: A methodological review. *Journal of Speech Sciences*, 1(1):85–115.

Yamagishi, J., Veaux, C., and MacDonald, K. (2019). CSTR VCTK corpus: English multi-speaker corpus for CSTR voice cloning toolkit (version 0.92). *University of Edinburgh. The Centre for Speech Technology Research (CSTR)*.

Yang, S., Chi, P.-H., Chuang, Y.-S., Lai, C.-I. J., Lakhotia, K., Lin, Y. Y., Liu, A. T., Shi, J., Chang, X., Lin, G.-T., Huang, T.-H., Tseng, W.-C., tik Lee, K., Liu, D.-R., Huang, Z., Dong, S., Li, S.-W., Watanabe, S., Mohamed, A., and yi Lee, H. (2021). SUPERB: Speech processing universal performance benchmark. In *Interspeech*, pages 1194–1198.

Yao, K. and Zweig, G. (2015). Sequence-to-sequence neural net models for grapheme-to-phoneme conversion. In *Interspeech*, pages 3330–3334.

Ye, H. and Young, S. (2006). Quality-enhanced voice morphing using maximum likelihood transformations. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(4):1301–1312.

Yu, K. and Young, S. (2010). Continuous F0 modeling for HMM based statistical parametric speech synthesis. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(5):1071–1079.

Zen, H. (2015). Acoustic modeling in statistical parametric speech synthesis-from HMM to LSTM-RNN. *Machine Learning in Speech and Language Processing Workshop*, 15.

Zen, H., Dang, V., Clark, R., Zhang, Y., Weiss, R. J., Jia, Y., Chen, Z., and Wu, Y. (2019). LibriTTS: A corpus derived from LibriSpeech for text-to-speech. In *Interspeech*, pages 1526–1530.

Zevallos, R., Bel, N., Cámbara, G., Farrús, M., and Luque, J. (2022). Data augmentation for low-resource Quechua ASR improvement. In *Interspeech*, pages 3518–3522.

Zhang, C., Ren, Y., Tan, X., Liu, J., Zhang, K., Qin, T., Zhao, S., and Liu, T.-Y. (2021a). DenoiSpeech: Denoising text to speech with frame-level noise modeling. In *International Conference on Acoustics, Speech and Signal Processing*, pages 7063–7067.

Zhang, G., Merritt, T., Ribeiro, S., Tura-Vecino, B., Yanagisawa, K., Pokora, K., Ezzerg, A., Cygert, S., Abbas, A., Bilinski, P., Barra-Chicote, R., Korzekwa, D., and Lorenzo-Trueba, J. (2023). Comparing normalizing flows and diffusion models for prosody and acoustic modelling in text-to-speech. In *Interspeech*, pages 27–31.

Zhang, M., Tao, J., Jia, H., and Wang, X. (2008). Improving HMM based speech synthesis by reducing over-smoothing problems. In *6th International Symposium on Chinese Spoken Language Processing*, pages 1–4.

Zhang, M., Zhou, Y., Zhao, L., and Li, H. (2021b). Transfer learning from speech synthesis to voice conversion with non-parallel training data. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:1290–1302.

Zhang, Q., Lu, H., Sak, H., Tripathi, A., McDermott, E., Koo, S., and Kumar, S. (2020). Transformer Transducer: A streamable speech recognition model with Transformer encoders and RNN-T loss. In *International Conference on Acoustics, Speech and Signal Processing*, pages 7829–7833.

Zhang, Y., Bakhturina, E., and Ginsburg, B. (2021c). NeMo (inverse) text normalization: From development to production. In *Interspeech*, pages 4857–4859.

Zhang, Z., Vyas, P., Dong, X., and Williamson, D. S. (2021d). An end-to-end non-intrusive model for subjective and objective real-world speech assessment using a multi-task framework. In *International Conference on Acoustics, Speech and Signal Processing*, pages 316–320.

Zhu, X., Zhang, Y., Yang, S., Xue, L., and Xie, L. (2019). Pre-alignment guided attention for improving training efficiency and model stability in end-to-end speech synthesis. *IEEE Access*, 7:65955–65964.

Résumé

Au cours des deux dernières décennies, le taux d'erreur des systèmes de reconnaissance automatique de la parole (RAP) a chuté drastiquement, les rendant ainsi plus utiles dans les applications réelles. Cette amélioration peut être attribuée à plusieurs facteurs, dont les nouvelles architectures utilisant des techniques d'apprentissage profond, les nouveaux algorithmes d'entraînement, les ensembles de données d'entraînement grands et diversifiés, et l'augmentation des données. En particulier, les jeux de données d'entraînement de grande taille ont été essentiels pour apprendre des représentations robustes de la parole pour les systèmes de RAP. Leur taille permet de couvrir efficacement la diversité inhérente à la parole, en terme de voix des locuteurs, de vitesse de parole, de hauteur, de réverbération et de bruit. Cependant, la taille et la diversité des jeux de données disponibles dans les langues bien dotées ne sont pas accessibles pour les langues moyennement ou peu dotées, ainsi que pour des domaines à vocabulaire spécialisé comme le domaine médical. Par conséquent, la méthode populaire pour augmenter la diversité des ensembles de données est l'augmentation des données. Avec l'augmentation récente de la naturalité et de la qualité des données synthétiques pouvant être générées par des systèmes de synthèse de la parole (TTS) et de conversion de voix (VC), ces derniers sont également devenus des options viables pour l'augmentation des données de RAP. Cependant, plusieurs problèmes limitent leur application. Premièrement, les systèmes de TTS/VC nécessitent des données de parole de haute qualité pour l'entraînement. Par conséquent, nous développons une méthode de curation dun jeux de données à partir d'un corpus conçu pour la RAP pour l'entraînement dun système de TTS. Cette méthode exploite la précision croissante des estimateurs de qualité non intrusifs basés sur l'apprentissage profond pour filtrer les échantillons de haute qualité. Nous explorons le filtrage du jeux de données de RAP à différents seuils pour équilibrer sa taille, le nombre de locuteurs et la qualité. Avec cette méthode, nous créons un ensemble de données interlocuteurs de haute qualité, comparable en qualité à LibriTTS. Deuxièmement, le processus de génération de données doit être contrôlable pour générer des données TTS/VC diversifiées avec des attributs spécifiques. Les systèmes TTS/VC précédents conditionnent soit le système sur l'empreinte du locuteur seule, soit utilisent des modèles discriminatifs pour apprendre les variabilités de la parole. Dans notre approche, nous concevons une architecture améliorée basée sur le flux qui apprend la distribution de différentes variables de la parole. Nous constatons que nos modifications augmentent significativement la diversité et la naturalité des énoncés générés par rapport à une référence GlowTTS, tout en étant contrôlables. Enfin, nous avons évalué l'importance de générer des données des TTS et VC diversifiées pour augmenter les données d'entraînement de RAP. Contrairement à la génération naïve des données TTS/VC, nous avons examiné indépendamment différentes approches telles que les méthodes de sélection des phrases et l'augmentation de la diversité des locuteurs, la durée des phonèmes et les contours de hauteur, en plus d'augmenter systématiquement les conditions environnementales des données générées. Nos résultats montrent que l'augmentation TTS/VC est prometteuse pour augmenter les performances de RAP dans les régimes de données faibles et moyen. En conclusion, nos expériences fournissent un aperçu des variabilités particulièrement importantes pour la RAP et révèlent une approche

systématique de l'augmentation des données de RAP utilisant des données synthétiques.

Mots-clés: synthèse de la parole, reconnaissance automatique de la parole, augmentation de données, conversion vocale, curation des données.

Abstract

In the last two decades, the error rate of automatic speech recognition (ASR) systems has drastically dropped, making them more useful in real-world applications. This improvement can be attributed to several factors including new architectures using deep learning techniques, new training algorithms, large and diverse training datasets, and data augmentation. In particular, the large-scale training datasets have been pivotal to learning robust speech representations for ASR. Their large size allows them to effectively cover the inherent diversity in speech, in terms of speaker voice, speaking rate, pitch, reverberation, and noise.

However, the size and diversity of datasets typically found in high-resourced languages are not available in medium- and low-resourced languages and in domains with specialised vocabulary like the medical domain. Therefore, the popular method to increase dataset diversity is through data augmentation. With the recent increase in the naturalness and quality of synthetic data that can be generated by text-to-speech (TTS) and voice conversion (VC) systems, these systems have also become viable options for ASR data augmentation. However, several problems limit their application.

First, TTS/VC systems require high-quality speech data for training. Hence, we develop a method of dataset curation from an ASR-designed corpus for training a TTS system. This method leverages the increasing accuracy of deep-learning-based, non-intrusive quality estimators to filter high-quality samples. We explore filtering the ASR dataset at different thresholds to balance the size of the dataset, number of speakers, and quality. With this method, we create a high-quality multi-speaker dataset which is comparable to LibriTTS in quality.

Second, the data generation process needs to be controllable to generate diverse TTS/VC data with specific attributes. Previous TTS/VC systems either condition the system on the speaker embedding alone or use discriminative models to learn the speech variabilities. In our approach, we design an improved flow-based architecture that learns the distribution of different speech variables. We find that our modifications significantly increase the diversity and naturalness of the generated utterances over a GlowTTS baseline while also being controllable.

Lastly, we evaluated the significance of generating diverse TTS and VC data for augmenting ASR training data. As opposed to naively generating the TTS/VC data, we independently examined different approaches such as sentence selection methods and increasing the diversity of speakers, phoneme duration, and pitch contours, in addition to systematically increasing the environmental conditions of the generated data. Our results show that TTS/VC augmentation holds promise in increasing ASR performance in low- and medium-data regimes. In conclusion, our experiments provide insight into the variabilities that are particularly important for ASR, and reveal a systematic approach to ASR data augmentation using synthetic data.

Keywords: text-to-speech, automatic speech recognition, data augmentation, voice conversion, data curation.