Accent Conversion and Pronunciation Improvement

Zur Erlangung des akademischen Grades eines
Doktors der Ingenieurwissenschaften
von der KIT-Fakultät für Informatik
des Karlsruher Instituts für Technologie (KIT)

genehmigte

DISSERTATION

von

TUAN NAM NGUYEN
aus Hanoi, Vietnam

Tag der mündlichen Prüfung: 26.06.2025
Referent: Prof. Alexander Waibel
Korreferent: Prof. Berrak Sisman

Abstract

Second-language learners often exhibit accents and mispronunciations, which can significantly affect their communication proficiency. Accents and pronunciation difficulties can create language barriers between speakers. This challenge motivated our research on "Accent Conversion and Pronunciation Improvement," which aims to make non-native speech sound more native, enhance intelligibility while preserving the original content, emotion and speaker identity. The scope of this thesis focuses specifically on accent conversion for English, and mostly converting from non-native accent to American(native) accent. Our approach focus mainly on various English accents, the approach is generally applicable to any pair of accented or non-accented speech. To date, we have published five research papers and two patents on this topic, each tackling different challenges in accent conversion step by step.

The one of most challenge of Accent Conversion is lacking ground truth data. There has not been a parallel corpus that contains pairs of audios having the same contents yet coming from the same speakers in different accents. In my first work, we provide a solution to create these parallel data. The approach involves synthesizing training data using a voice conversion model that retains pronunciation patterns and prosody while altering speaker identity.

In the second work, we have a motivation that create the TTS which can synthesize audio from transcripts with any voices and many accents without demanding a lot of accented training resources. We proposed SYNTACC, the adapts conventional multi-speaker text-to-speech (TTS) models to synthesize speech in multiple accents. The method factorizes model weights into a shared component and an accent-dependent component, which reduces the number of training parameters needed per accent. This approach enables high-quality multi-accent speech synthesis in low-resource training conditions. The evaluation on Indian, Spanish, Chinese, and Vietnamese accents demonstrates that SYNTACC effectively generates accented speech while preserving speaker identity.

After the first two works, we gained the ability to generate a larger amount of accented speech data. With this increased data availability, we can explore training a more advanced and robust model for this task. In the third research, we approach accent conversion in a way similar to speech-to-speech translation, treating each accent as a distinct language and converting between accents in the same manner as translating between languages. To bridge the acoustic variability between accents and enable language-like modeling, we introduce discrete speech units as an intermediate representation. These units are obtained by quantizing hidden representations from neural network model, capturing high-level linguistic content

while abstracting away speaker. By using discrete units, we can apply the state-of-the-art speech-to-speech translation model for solving the accent conversion model. The approach significantly improves non-native fluency and convert accent while preserving speaker identity. The evaluation shows that this method outperforms previous AC approaches in fluency and naturalness.

In the first three works, our approaches basically builds the mapping function from one accent to another accent. In fourth work, we come up with different approach: disentangle-resynthesis, which can leverage non-parallel data by decomposing speech into components like speaker identity, content, prosody, and accent, which are then recombined for synthesis. We assume TTS system trained solely on native speech will produce accent-independent linguistic representations. By generating high-quality ground-truth audio, native TTS ensures correct pronunciation, consistent speaker identity, aligned duration, and synchronized prosody with the original non-native speech. Additionally, this native TTS system is able to generate ideal ground-truth data for non-native speakers, ensuring native pronunciation, same speaker identity, duration, prosody, and precise alignment with the original non-native audio. Their AC model leverages TTS text representations to learn accent-independent features and uses synthetic ground-truth to learn the mapping function from non-native accent speech to native-like speech. This approach outperforms previous methods in term of preserving speaker identity, prosody (emotion) and duration.

Although the model in the fourth research achieves fast inference due to its non-autoregressive architecture, it still requires the entire input for prediction, making it unsuitable for streaming applications. Recognizing the previous research as one of the best for accent conversion, in the fifth work, we focus on essential architectural and training modifications to develop a streaming-capable AC model while maintaining the performance of its non-streaming counterpart.

In this thesis, we address the challenge of ground-truth data in accent conversion by developing solutions for generating high-quality training corpora. Furthermore, we categorize accent conversion approaches into two main paradigms: (1) mapping-based models that learn a direct transformation between accents, and (2) the disentangle-resynthesis framework, which decomposes speech into multiple components such as content, accent, and speaker identity, enabling controlled modification and synthesis. Additionally, we propose a streaming accent conversion model, making streaming applications feasible. Our contributions advance the field by improving data availability, enhancing model performance, and enabling more natural and practical accent conversion.

Zusammenfassung

Zweitsprachlerner weisen häufig Akzente und Aussprachefehler auf, die ihre Kommunikationsfähigkeit erheblich beeinträchtigen können. Diese Herausforderungen motivierten unsere Forschung zur „Akzentkonvertierung und Aussprachеverbesserung“, mit dem Ziel, nicht-native Sprache nativer klingen zu lassen, die Verständlichkeit zu erhöhen und dabei Inhalt, Emotion und Sprecheridentität zu bewahren. Der Fokus dieser Dissertation liegt speziell auf der Akzentkonvertierung für Englisch, insbesondere der Umwandlung von nicht-nativen Akzenten in den amerikanischen (nativen) Akzent. Obwohl unser Ansatz hauptsächlich auf verschiedene englische Akzente abzielt, ist er grundsätzlich auf jedes Paar von akzentbehafteter oder akzentfreier Sprache anwendbar. Bis heute haben wir fünf Forschungsarbeiten und zwei Patente zu diesem Thema veröffentlicht, die jeweils unterschiedliche Herausforderungen der Akzentkonvertierung schrittweise angehen.

Eine der größten Herausforderungen bei der Akzentkonvertierung ist das Fehlen von Ground-Truth-Daten. Es existiert kein paralleles Korpus, das Paare von Audiodateien mit identischem Inhalt, aber unterschiedlichen Akzenten desselben Sprechers enthält. In unserer ersten Arbeit präsentieren wir eine Lösung zur Erstellung solcher paralleler Daten. Der Ansatz beinhaltet die Synthese von Trainingsdaten mithilfe eines Voice-Conversion-Modells, das Aussprachemuster und Prosodie beibehält, während die Sprecheridentität verändert wird.

In der zweiten Arbeit verfolgen wir das Ziel, ein Text-to-Speech-System (TTS) zu entwickeln, das Audiodaten aus Transkripten mit verschiedenen Stimmen und Akzenten erzeugen kann, ohne umfangreiche akzentbezogene Trainingsressourcen zu benötigen. Wir schlagen SYNTACC vor, das konventionelle Multi-Sprecher-TTS-Modelle anpasst, um Sprache in mehreren Akzenten zu synthetisieren. Die Methode faktorisiert Modellgewichte in eine gemeinsame Komponente und eine akzentabhängige Komponente, wodurch die Anzahl der pro Akzent benötigten Trainingsparameter reduziert wird. Dieser Ansatz ermöglicht hochwertige Multi-Akzent-Sprachsynthese unter Bedingungen mit begrenzten Trainingsressourcen. Die Bewertung für indische, spanische, chinesische und vietnamesische Akzente zeigt, dass SYNTACC effektiv akzentbehaftete Sprache generiert und dabei die Sprecheridentität bewahrt.

Nach den ersten beiden Arbeiten konnten wir eine größere Menge an akzentbehafteten Sprachdaten generieren. Mit dieser erhöhten Datenverfügbarkeit können wir ein fortschrittlicheres und robusteres Modell für diese Aufgabe trainieren. In der dritten Forschung behandeln wir die Akzentkonvertierung ähnlich wie die Sprach-zu-Sprach-Übersetzung, wobei jeder Akzent als eigene Sprache betrachtet wird. Um die

akustische Variabilität zwischen Akzenten zu überbrücken und eine sprachähnliche Modellierung zu ermöglichen, führen wir diskrete Spracheinheiten als Zwischenrepräsentation ein. Diese Einheiten werden durch Quantisierung von versteckten Repräsentationen eines neuronalen Netzwerks gewonnen und erfassen hochrangige linguistische Inhalte, während sie sprecherspezifische Details abstrahieren. Durch die Verwendung diskreter Einheiten können wir ein hochmodernes Sprach-zu-Sprach-Übersetzungsmodell auf die Akzentkonvertierung anwenden. Dieser Ansatz verbessert die Flüssigkeit nicht-nativer Sprache erheblich und konvertiert Akzente, während die Sprecheridentität erhalten bleibt. Die Evaluation zeigt, dass diese Methode frühere Ansätze in Bezug auf Flüssigkeit und Natürlichkeit übertrifft.

In den ersten drei Arbeiten bauen unsere Ansätze im Wesentlichen eine Abbildungsfunktion von einem Akzent zu einem anderen auf. In der vierten Arbeit verfolgen wir einen anderen Ansatz: Disentangle-Resynthese, der nicht-parallele Daten nutzt, indem er Sprache in Komponenten wie Sprecheridentität, Inhalt, Prosodie und Akzent zerlegt, die dann für die Synthese wieder kombiniert werden. Wir nehmen an, dass ein TTS-System, das ausschließlich auf nativer Sprache trainiert wurde, akzentunabhängige linguistische Repräsentationen erzeugt. Durch die Generierung hochwertiger Ground-Truth-Audiodaten stellt das native TTS-System korrekte Aussprache, konsistente Sprecheridentität, abgestimmte Dauer und synchronisierte Prosodie mit der ursprünglichen nicht-nativen Sprache sicher. Zusätzlich kann dieses native TTS-System ideale Ground-Truth-Daten für nicht-native Sprecher erzeugen, die native Aussprache, gleiche Sprecheridentität, Dauer, Prosodie und präzise Ausrichtung mit dem ursprünglichen nicht-nativen Audio gewährleisten. Unser AC-Modell nutzt TTS-Textrepräsentationen, um akzentunabhängige Merkmale zu lernen, und verwendet synthetische Ground-Truth-Daten, um die Abbildungsfunktion von nicht-nativer akzentbehafteter Sprache zu nativer Sprache zu erlernen. Dieser Ansatz übertrifft frühere Methoden hinsichtlich der Erhaltung der Sprecheridentität, Prosodie (Emotion) und Dauer.

Obwohl das Modell in der vierten Forschung aufgrund seiner nicht-autoregressiven Architektur eine schnelle Inferenz ermöglicht, erfordert es dennoch die gesamte Eingabe für die Vorhersage, was es für Streaming-Anwendungen ungeeignet macht. Da die vorherige Forschung als eine der besten für die Akzentkonvertierung anerkannt ist, konzentrieren wir uns in der fünften Arbeit auf wesentliche architektonische und trainingsbezogene Modifikationen, um ein Streaming-fähiges AC-Modell zu entwickeln, das die Leistung seines nicht-Streaming-Gegenstücks beibehält.

In dieser Dissertation adressieren wir die Herausforderung fehlender Ground-Truth-Daten in der Akzentkonvertierung, indem wir Lösungen zur Generierung hochwertiger Trainingskorpora entwickeln. Darüber hinaus kategorisieren wir Akzentkonvertierungsansätze in zwei Hauptparadigmen: (1) Mapping-basierte Modelle, die eine direkte Transformation zwischen Akzenten erlernen, und (2) das Disentangle-Resynthese-Framework, das Sprache in mehrere Komponenten wie Inhalt, Akzent und Sprecheridentität zerlegt und eine kontrollierte Modifikation und Synthese ermöglicht. Zusätzlich schlagen wir ein Streaming-fähiges Akzentkonvertierungsmodell vor, das Streaming-Anwendungen ermöglicht. Unsere

Beiträge fördern das Feld, indem sie die Datenverfügbarkeit verbessern, die Modellleistung steigern und eine natürlichere und praktischere Akzentkonvertierung ermöglichen.

Acknowledgments

I would like to thank everyone who has supported and helped me during the work on this thesis.

First and foremost, I am deeply grateful to Professor Alexander Waibel for accepting me into the ASR Team six years ago and giving me the opportunity to pursue research in this exciting field. Initially, my focus was on improving speech recognition for accented speech. However, through his innovative ideas, and visionary thinking, he encouraged me to think further , to imagine a system that can change accents of human speech.

From the very beginning, I found the idea of accent conversion fascinating. I dreamed of creating a system capable of performing accent conversion in streaming application, and thanks to Professor Waibel's continuous support, advice, and visionary leadership, this dream has gradually come true. His influence has been invaluable throughout my entire research journey. Working at the Interactive Systems Lab has been a truly rewarding and joyful experience.

This research was supported by a grant from Zoom Video Communications, Inc. I gratefully acknowledge their generous support, which helped make this work possible. I would also like to thank Professor Waibel for facilitating this collaboration and for his trust in me to carry out this research direction.

I would also like to sincerely thank Professor Berrak Sisman for kindly agreeing to co-review this thesis and for providing valuable feedback. Her expertise in speech synthesis has been particularly insightful and has greatly enriched the evaluation and positioning of this work.

My special thanks go to Ngoc-Quan Pham for carefully proofreading parts of my thesis as well as many of my research papers. The time I spent working with him in the ASR team was truly enjoyable and invaluable, I learned so much from him during that period.

I am also grateful to all the past and present members with whom I had the pleasure of working over the years: Thanh Le Ha, Thai Son Nguyen, Juan Hussain, Professor Jan Niehues, Stefan, Dogucan, Irem, Enes, Thai-Binh, Tu-Anh, Fabian, Leonard, Zhaolin, Yining, Sai and Seymanur. They have been not only wonderful colleagues but also good friends who shared memorable moments — from working together, having lunch and dinner, to playing sports. Their support made my journey at the lab both productive and joyful.

My thanks also go to all the secretarial, technical, and administrative staff, with special thanks to Silke Dannenmaier and Margit Rödder for their invaluable support. I would also like to thank Laura Kernahan, Maximilian Eifried, and Ilya Shibayev for their assistance and help throughout my time at the lab.

Last but not least, I would like to express my deepest gratitude to my beloved family: my wife, Thuy-Lam, and my daughter, Nam-Tuong, for sharing every moment with me and making my PhD journey so much more wonderful over these years. Their endless support, love, and encouragement have been an infinite source of strength, helping me stay motivated and making every challenge feel lighter and every success even more meaningful.

Contents

1. Introduction

1.1. What is accent

An accent refers to the unique way in which individuals or groups pronounce words, which often reflects their geographic, cultural, or social background. It encompasses variations in phonetic elements such as intonation, rhythm, stress, and the articulation of sounds. Accents are not only linked to the native language of a speaker, they can also develop as a result of learning a second language, with speakers carrying over phonological features from their first language.

Accents do not affect the grammatical structure of a language but influence how words and sounds are pronounced. These variations are a natural consequence of human language diversity, and, although they are often tied to national or regional identities, other factors such as age, socioeconomic status, and social group affiliations can also shape an individual's accent.

1.2. Applications of accent conversion

Second-language learners often exhibit accents and mispronunciations, which can significantly affect their communication proficiency. The accent not only associates with the second-language learner, but also present in native speaker, such as British accent, Indian accent, Philippino accent, etc. In general, accent and pronunciation challenges can create language barriers between speakers from diverse regions. This challenge motivated our research on this thesis.

One key motivation for accent conversion is to improve speech intelligibility in scenarios where a strong regional or foreign accent may hinder comprehension. For example, in customer service or call center interactions, accent conversion technology can enhance communication clarity, ensuring that non-native speakers are better understood by native listeners. This has practical applications in multinational companies, healthcare, education, and more.

The ability to convert or modify accents has significant implications in various domains of speech technology, language learning, and global communication Waibel et al. (2023); Nguyen and Waibel (2025). As the world becomes more interconnected, spoken language barriers remain a challenge, and accents can sometimes act as a source of misunderstanding or bias. Accent conversion seeks to bridge this gap by enabling better communication between different linguistic and cultural backgrounds.

In the realm of language learning, accent conversion serves as a tool for learners to acquire more native-like pronunciation. By hearing and practicing with accent-adapted speech models, learners can fine-tune their phonetic and prosodic elements more effectively. This contributes to better oral proficiency and the development of communication skills in a second language.

As accent conversion continues to evolve, the goal is not only to modify speech but also to maintain the speaker's voice identity while changing the accent. This balance between maintaining individuality and changing accent is central to the motivation behind developing accent conversion models, particularly in the context of creating inclusive and accessible speech technologies.

1.3. Challenges of Accent Conversion Models

Accent conversion models face several challenges related to both acoustic and linguistic aspects, as well as the availability of ground-truth data.

Acoustic Challenge

The primary acoustic challenge lies in modifying non-native speech to sound more like native speech while preserving the speaker's identity and emotional expression. This requires precise control over prosodic features such as pitch, rhythm, and intonation, as well as spectral characteristics like timbre. Striking a balance between accent modification and the preservation of speaker-specific traits is a complex task, as overly aggressive modifications can result in unnatural-sounding speech or the loss of the original speaker's identity.

Linguistic Challenge

On the linguistic front, the focus is on enhancing the intelligibility of non-native speech, ensuring that it is clearer and easier to understand. This involves addressing pronunciation errors, improving phoneme articulation, and adapting speech patterns to align with native norms. Achieving this without altering the semantic content of the speech is a significant challenge, as linguistic modifications must be context-aware and linguistically accurate.

Ground-Truth Challenge

One of the most significant challenges in accent conversion is the lack of ground-truth data. Currently, there is no parallel corpus that contains pairs of audio recordings with the same linguistic content spoken by the same speaker in different accents. This absence of high-quality, aligned training data makes it difficult to train models effectively, as they require precise supervision to learn the subtle differences between accents. As a result, researchers often rely on non-parallel data or synthetic datasets, which can introduce additional complexities and limitations.

Addressing these challenges requires innovative approaches in model design, data augmentation, and evaluation methodologies, which will be discussed in detail in the subsequent chapters.

1.4. Contributions

The main contribution of this thesis is a systematic approach and the necessary techniques to construct and evaluate some of the very first accent conversion models. We address the accent conversion's challenges through innovative methodologies, detailed evaluation strategies, and practical applications.

Key Contributions

1. **Systematic Frameworks for Accent Conversion:** We categorize accent conversion approaches into two main paradigms:

- **Direct-Mapping Framework:** This approach learns a direct transformation between accents, enabling efficient and straightforward conversion.
- **Disentangle-Resynthesis Framework:** This paradigm decomposes speech into multiple components, such as content, accent, and speaker identity, allowing for controlled modification and synthesis.

2. **Addressing the Ground-Truth Challenge:** One of the most significant challenges in accent conversion is the lack of ground-truth data. To overcome this, we develop solutions to generate high-quality training corpora, enabling the training of robust and accurate models. Our work in this area has resulted in five published papers and two filed patents, each exploring different data augmentation strategies and model architectures.
3. **Streaming Accent Conversion:** We propose a streaming accent conversion model, making real-time applications feasible. This innovation addresses the challenges of low-latency processing and seamless integration into applications such as virtual conferences and meetings.

Significance

The contributions of this thesis advance the field of accent conversion by providing a comprehensive framework for model development, addressing critical challenges such as ground-truth data scarcity, and enabling real-time applications. Our work not only improves the state-of-the-art in accent conversion but also opens new avenues for research and practical deployment in diverse domains. The works in the thesis are published in InterSpeech and ICASSP conferences Nguyen et al. (2022, 2023, 2024, 2025)

1.5. Thesis organization

In the next chapter (Chapter 2), we focus on reviewing the background and state-of-the-art literature relevant to the thesis topic. The first part (Section 2.1) of this chapter covers speech processing, including feature extraction techniques that are fundamental for various speech-related tasks. These techniques play a crucial role in speech recognition, synthesis, and accent conversion.

After reviewing some speech processing background, we explore the literature on relevant problems, including text-to-speech (TTS) synthesis, voice conversion, speech recognition (Section 2.2). State-of-the-art TTS systems generate natural-sounding speech from text with various reference voices. Voice conversion focuses on modifying speaker identity while preserving linguistic content, which shares similarities with accent conversion. We not only inherit some state-of-the-art (SOTA) architectures from TTS and voice conversion but also use these models for augmenting additional training data for accent conversion.

Linguistic challenges are a crucial aspect of accent conversion; therefore, understanding spoken language is essential (Section 2.3). Automatic Speech (ASR) recognition models enable machines to transcribe spoken language into text, serving as a fundamental component for evaluating pronunciation improvements in accent conversion. Additionally, in this thesis, ASR helps the AC model comprehend speech content and extract content representation.

Next, we present self-supervised speech representation learning, which have achieved state-of-the-art results across many speech processing tasks (Section 2.4). These methods, such as Wav2Vec 2.0 Baevski et al. (2020a), HuBERT Hsu et al. (2021a), and WavLM Chen et al. (2022), enable models to learn powerful and generalizable speech representations from large amounts of unlabeled audio. By capturing both acoustic and linguistic information, self-supervised models greatly benefit tasks like automatic speech recognition, speech translation, voice conversion, and accent conversion . We include self-supervised speech representation in the Background section, because it forms a fundamental building block for many techniques used throughout this thesis. Understanding how self-supervised methods improve feature extraction, weight initialization, robustness to noise, and cross-task transferability is essential for following the design choices made later.

Voice Conversion (VC) is closely related to accent conversion. Specifically, VC models provide techniques for disentangling speaker identity from linguistic content, a property that is crucial for accent conversion tasks. In this thesis, voice conversion is not only employed as a data augmentation strategy, but also leveraged as a backbone for developing accent conversion models. In Section 2.5, we review crucial research works in voice conversion that have inspired and formed the development of the methods proposed in this thesis.

Lastly, in Section 2.6, we present a literature review on accent conversion (AC), beginning with a discussion of how it differs from voice conversion (VC), which, despite their similarities, differ in their problem definitions, challenges, and target objectives. These differences necessitate distinct modeling strategies and evaluation methods. Following this, we review notable previous works in the field of accent conversion and discuss the methodologies commonly used to evaluate the performance of such systems. Finally, we describe the datasets and evaluation framework employed in this thesis. These resources provide a basis for training, testing, and comparing AC models, ensuring that our proposed methods are thoroughly validated across both objective and subjective performance metrics.

In our very first works (Chapter 3), we approach the accent conversion problem as aiming to build a direct mapping function from one accent to another. However, obtaining training parallel data, which has utterances from the same speakers in different accents is challenging due to its scarcity. To address this, we explore data augmentation techniques such as using voice conversion to generate high-quality ground-truth audio for non-native speakers. In the chapter 4, we develop SYNTACC - Synthesizing Multi-Accent Speech by Weight Factorization- a system that enables controllable accented text-to-speech synthesis with minimal dependence on accented speech data. In the chapter 5, by leveraging this capability to generate extensive parallel training data and considering the accent conversion as speech-to-speech translation, we train a state-of-the-art speech-to-speech model that serves as a mapping function to convert speech from many source accents to a target accent.

In the chapter 6, we approach the accent conversion as a disentangle-resynthesis framework, which leverages non-parallel data, making it more flexible and widely applicable. This method decomposes speech into separate components—such as speaker identity, content, prosody, and accent—which are then recombined to synthesize the output waveform. Content features are often represented by bottleneck features (BNFs), extracted from self-supervised models , ASR bottlenecks , or ASR logits . However, BNFs inherently capture accent-related features and pronunciation errors. Therefore, in this work, we focus on disentangling accent-related features and improving pronunciation in the content representation. Finally, in Chapter 7, we make modifications to our disentangle-resynthesis model with the goal of adapting them for streaming purpose. In Chapter8, we present a comprehensive comparison and evaluation of the proposed accent conversion approaches in real-world scenarios. Specifically, we analyze their ability to generalize to out-of-domain accents that are unseen during training and investigate their scalability as the amount of training data increases. These evaluations provide deeper insights into the robustness and practical applicability of the different modeling paradigms. Finally, Chapter 9 concludes this dissertation work and discusses possible future research directions.

2. BackGround

2.1. Speech Processing

Speech processing plays a crucial role in both speech synthesis and accent conversion by enabling more natural, intelligible, and controllable speech generation. In speech synthesis, instead of directly synthesizinh waveforms, modern systems can generate intermediate audio features, such as mel-spectrograms or linguistic embeddings, which are then converted back to waveforms using a vocoder. This two-step approach offers several advantages: it simplifies modeling by reducing the complexity of waveform generation, improves generalization across different speakers and accents, and allows for more flexible transformations, such as modifying prosody, pronunciation, and speaker characteristics.

One key reason for using audio features in training is that directly optimizing a loss function between two waveforms is highly sensitive and ineffective. Waveforms are high-dimensional and oscillate rapidly, meaning that even small misalignments can lead to large numerical differences in loss calculations, despite perceptually similar sounds. This issue, known as the high sensitivity of time-domain loss functions, makes it difficult to train models that generate high-quality speech. By working with intermediate audio features, such as spectrograms, the loss function operates on a more structured and perceptually meaningful representation, reducing sensitivity to phase shifts and fine temporal misalignments. The vocoder then reconstructs the waveform in a way that preserves the naturalness and consistency of speech.

In addition to spectral features like Mel-spectrograms, modern speech processing systems often extract prosodic and speaker-related attributes—most notably, **pitch (fundamental frequency)** and **speaker embeddings**—to further enhance controllability and naturalness.

Pitch extraction plays a crucial role in capturing the intonation and rhythm of speech, which are key components of prosody. Accurate modeling of pitch allows systems to generate speech that sounds expressive and emotionally appropriate. In accent conversion, pitch is particularly important for matching the intonational patterns of the target accent, which can differ significantly from those of the source accent. For example, tonal languages or stress-timed languages may impose different pitch dynamics on speech, and being able to capture and adapt these variations is essential for natural-sounding accent conversion.

Speaker embeddings encode unique speaker characteristics such as vocal timbre and pitch range. This disentanglement of speaker identity from content and prosody enables multi-speaker synthesis, voice conversion, and speaker-preserving accent conversion.

2.1.1. Mel-spectrogram

In this section, we describe the *Mel-spectrogram*, a widely used time-frequency feature of speech signals. Mel-spectrograms play a crucial role in various speech processing tasks, including speech synthesis and recognition, by transforming raw audio into a format that captures both spectral and temporal dynamics aligned with human auditory perception.

Motivation and Mel Scale

Unlike traditional spectrograms that use a linear frequency scale, Mel-spectrograms apply the *Mel scale* Wikipedia (2025a), which compresses higher frequencies and expands lower ones. This mimics how human ears perceive sound, being more sensitive to pitch differences at lower frequencies.
The mapping from a frequency f in Hz to its Mel-scale counterpart $m(f)$ is given by:

$$m(f) = 2595 \log_{10}\left(1 + \frac{f}{700}\right) \tag{2.1}$$

This non-linear transformation allows the model to focus on the regions most important for speech understanding and perception, especially relevant in tasks like speech modeling.

Computing the Mel-Spectrogram

The Mel-spectrogram is computed through the following pipeline:

1. **Short-Time Fourier Transform (STFT):** The input audio signal is divided into overlapping short frames, and the Fourier transform is applied to each frame to obtain the frequency content. This results in the time-frequency representation known as a spectrogram Wikipedia (2025c)

2. **Mapping to Mel Scale:** While the Mel scale defines how humans perceive pitch, the practical transformation from the linear spectrogram to a Mel-spectrogram is done using a set of triangular filters known as a *Mel filterbank*. These filters are linearly spaced on the Mel scale and are applied to the power spectrum obtained from the STFT. Each filter aggregates energy from a specific range of frequencies and contributes to one Mel-frequency bin. This operation emphasizes perceptually meaningful frequency regions and compresses irrelevant detail, especially in higher frequencies.

3. **Logarithmic Compression:** The magnitude of each Mel bin is compressed using a logarithmic function to match human loudness perception, yielding the final Mel-spectrogram $M(m,t)$.

Mathematically, the transformation is expressed as:

$$M(m,t) = \log\left(\sum_f H(m,f)|X(f,t)|^2\right) \tag{2.2}$$

where $H(m,f)$ is the Mel filter bank, and $|X(f,t)|^2$ is the power of the STFT.

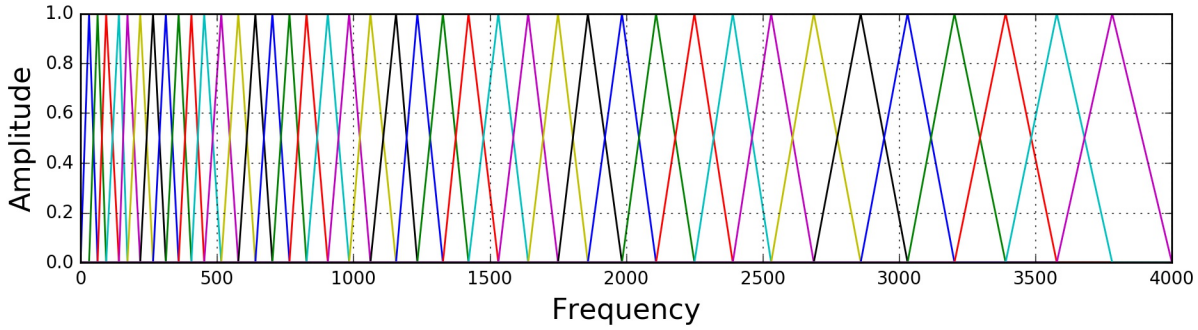

Figure 2.1.: Visualization of Mel filter banks applied to a linear spectrogram. Each triangular filter corresponds to one Mel-frequency bin.

Visualization and Interpretation

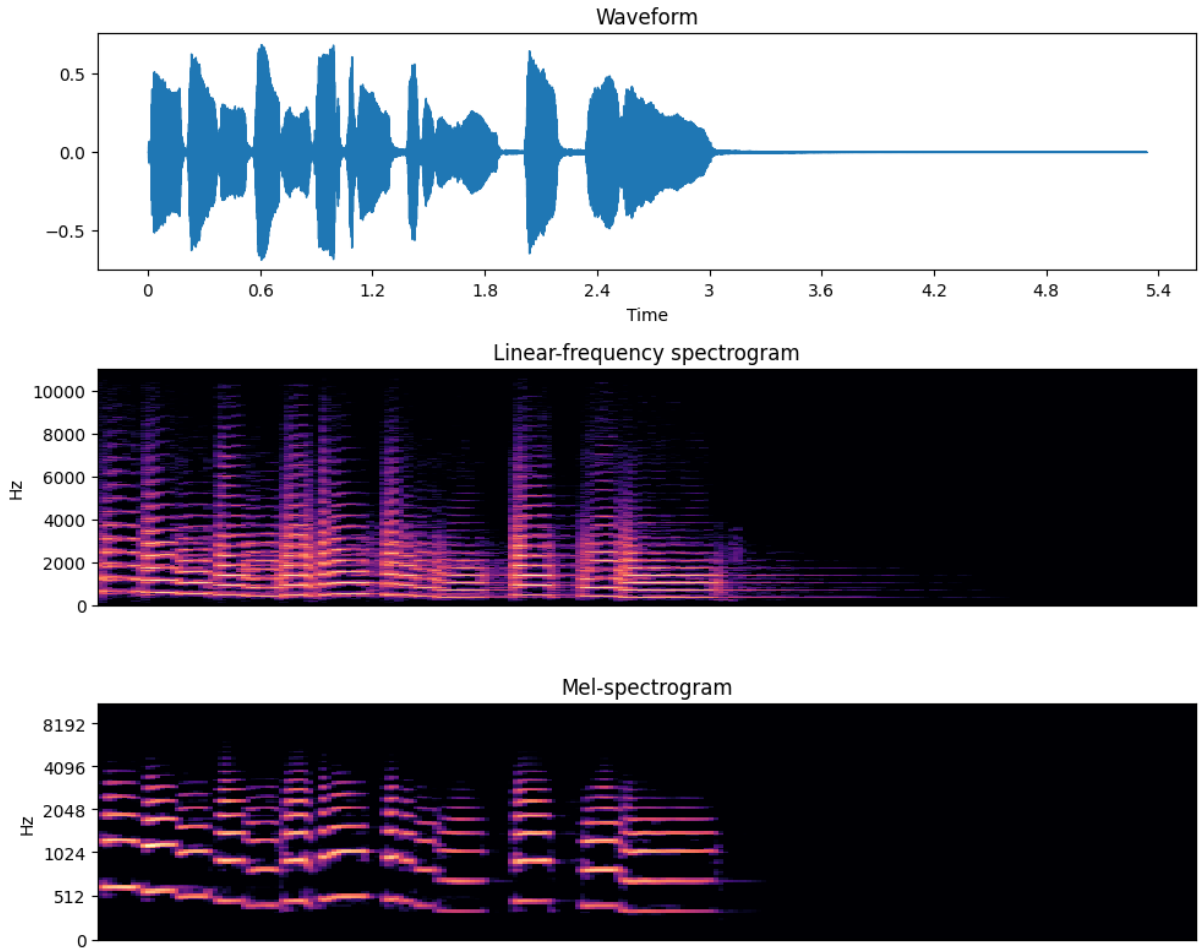

Figure 2.2.: Illustration of a waveform (top), a linear-frequency spectrogram (middle), and a Mel-spectrogram (bottom). The Mel-spectrogram emphasizes perceptually important frequency bands.

The Mel-spectrogram is commonly visualized as a 2D heatmap where:

- The x-axis represents time,
- The y-axis corresponds to Mel-scaled frequency bins,
- The color intensity indicates the log-scaled magnitude of each bin.

This representation provides intuitive insights into the temporal and spectral patterns of speech, such as phonetic content, intonation, and rhythm.

Summary

In summary, the Mel-spectrogram is a perceptually motivated and computationally efficient representation that bridges raw waveform data and high-level speech features. Its compact and informative structure makes it an indispensable tool for training models in speech modeling, allowing for high-fidelity audio synthesis.

2.1.2. Pitch extraction

Controlling pitch is essential in speech synthesis to achieve natural-sounding prosody, as variations in pitch contribute to expressiveness, emphasis, and speaker characteristics. Pitch extraction Wikipedia (2025b) is a crucial task in speech processing that involves estimating the fundamental frequency (F0) of a speech signal. The fundamental frequency represents the vibration rate of the vocal cords and serves as a key factor in prosody, speaker identity, and speech intelligibility. Accurately extracting and manipulating F0 allows speech synthesis systems to generate more natural and expressive speech, ensuring that synthesized voices sound dynamic rather than monotonous. This is particularly important in tasks such as accent conversion, where modifying pitch patterns can help better align the speech characteristics with the target accent while preserving the speaker's identity. Several algorithms have been developed for robust and accurate pitch tracking, among which YIN and YAAPT are two widely used methods.

YIN Algorithm

The YIN algorithm de Cheveigné and Kawahara (2002) is a time-domain method designed for accurate and robust pitch detection. It is based on the autocorrelation principle but introduces several modifications to reduce common pitch estimation errors.

- **Difference function:** YIN computes a difference function instead of autocorrelation to emphasize local minima, which correspond to pitch periods.
- **Cumulative mean normalization:** The difference function is normalized to suppress false minima due to noise.

- **Parabolic interpolation:** The minimum of the difference function is refined using interpolation for more accurate pitch period estimation.

YIN is known for its accuracy, low latency, and robustness to moderate noise, making it suitable for offline and real-time speech applications.

YAAPT Algorithm

YAAPT (Yet Another Algorithm for Pitch Tracking) Kasi and Zahorian (2002) is a hybrid approach that combines time-domain and frequency-domain features to achieve high robustness in noisy and complex speech environments.

- **Multi-domain analysis:** YAAPT uses both spectral and temporal cues, including a spectral correlation function and dynamic programming to track pitch.
- **Voicing decision:** The algorithm explicitly detects voiced/unvoiced frames using energy and spectral features.
- **Tracking and smoothing:** A dynamic programming-based path search is used to select the best pitch contour across time, improving continuity.

YAAPT is particularly effective in handling degraded or noisy signals and provides smooth and reliable pitch trajectories, which are highly useful in expressive speech modeling and accent conversion. In this thesis, we choose both YIN and YAAPT for pitch detection algorithm.

2.1.3. Speaker Attribute Modeling

Speaker Embeddings for Voice and Accent Conversion

Speaker embeddings are compact, fixed-dimensional representations that capture essential speaker-specific characteristics, such as timbre, pitch, and speaking style, from variable-length speech inputs. These embeddings enable models to disentangle speaker identity from linguistic content, playing a critical role in voice conversion (VC) and accent conversion (AC) systems.
Several types of speaker embeddings have been developed over time:

- **i-vector:** Guo et al. (2018) Introduced initially for speaker recognition, the i-vector framework represents an utterance as a low-dimensional vector derived from a total variability space. While effective, i-vectors rely on Gaussian Mixture Models (GMMs) and are less robust to domain shifts.
- **x-vector:** Snyder et al. (2018) An advancement over i-vectors, x-vectors are deep neural network-based embeddings extracted from frame-level features. They offer better performance in speaker recognition and are more adaptable across different conditions.

- **d-vector:** Li et al. (2015) Derived from deep speaker verification models, d-vectors are computed by averaging the hidden layer activations corresponding to an input utterance. d-vectors have been widely used in Voice Conversion tasks due to their robustness and ability to generalize across unseen speakers.

Importance of Speaker Embeddings in Accent Conversion

In accent conversion, maintaining the speaker's identity while altering pronunciation patterns is crucial. Speaker embeddings provide a powerful conditioning mechanism that allows models to preserve speaker attributes even as the linguistic features are modified. Specifically:

- **Identity Preservation:** Liu et al. (2021a) By conditioning the speech generation model on speaker embeddings, the system ensures that the converted output retains the original speaker's voice characteristics, making the converted speech sound authentic.
- **Speaker-Independent Feature Transformation:** Speaker embeddings allow the AC model to focus exclusively on modifying pronunciation or accent patterns without inadvertently altering speaker-specific traits.
- **Generalization to Unseen Speakers:** Wang et al. (2023) Pretrained speaker embedding extractors enable accent conversion systems to operate effectively even with speakers unseen during training, thus supporting zero-shot or few-shot adaptation.

It is important to distinguish the roles that speaker embeddings and accent features play in the accent conversion process.

Speaker embeddings are designed to capture **global, time-invariant** characteristics of a speaker. These include features such as voice timbre, pitch range, speaking style, and vocal tract characteristics, which typically remain consistent throughout an utterance or even across different utterances from the same speaker. As such, speaker embeddings serve as a global conditioning signal that defines the overall identity of the speaker across the entire generated waveform.

In contrast, accent features are inherently **local, time-varying** phenomena. Accents affect pronunciation at the phoneme or syllable level, influencing aspects like vowel shifts, consonant articulation, rhythm, and intonation. These variations occur dynamically as the speech unfolds over time and cannot be effectively captured by a static global embedding.

Thus, a well-designed accent conversion system should treat speaker identity and accent patterns differently:

- Use speaker embeddings to provide a *global* representation that remains fixed across the entire utterance, ensuring consistent speaker identity.

- Convert accent as *time-dependent* transformations that operate at a fine-grained level, allowing dynamic modification of pronunciation and prosody while preserving the speaker's core attributes.

Failing to maintain this separation may lead to artifacts such as speaker identity loss Chen et al. (2024), unnatural prosody. Therefore, careful disentanglement of speaker and accent representations is crucial for achieving high-quality, natural accent conversion.

Challenges and Limitations

Despite their advantages, speaker embeddings can also present challenges, especially when they are not sufficiently powerful or when facing unseen speakers:

- **Poor Generalization to Unseen Speakers:** If the embedding extractor is not trained on a diverse dataset, embeddings may fail to capture the true identity of speakers who were not present during training. This can lead to significant speaker identity leakage or blending in the converted speech.
- **Insufficient Speaker Representation:** Weak embeddings may not fully encode the fine-grained speaker attributes such as prosody, emotional tone, or subtle voice timbres. As a result, the synthesized speech may sound generic or lose personal voice characteristics.
- **Sensitivity to Recording Conditions:** Speaker embeddings can be affected by background noise, microphone variability, and recording environments. These factors introduce noise into the embedding, which can degrade the quality and consistency of accent conversion.

Interestingly, in speech synthesis, models often exhibit reasonable performance even when the speaker embeddings are not particularly strong or robust Liu (2024). This phenomenon can be attributed to the implicit memorization of speaker characteristics during training. When the model is trained on a fixed set of speakers, it can learn to associate certain patterns in the acoustic or latent space with specific speakers, essentially "baking in" speaker identity information into the model parameters themselves. As a result, even if the speaker embedding does not provide a rich or highly discriminative description of the speaker, the model can still generate speech that sounds like one of the training speakers by relying on this learned association.

However, this reliance on memorization presents a major limitation: when encountering unseen speakers during inference, the model struggles to generalize, leading to degraded speaker similarity or even mode collapse (i.e., generating voices that sound like the closest training speaker rather than the true target speaker). This highlights the importance of having strong, generalizable speaker embeddings—especially for tasks like accent conversion or voice conversion that demand speaker preservation under more challenging conditions, such as accent shifts, unseen speakers, and diverse speaking styles.

In this thesis, we adopt state-of-the-art speaker embedding model to achieve robust speaker identity modeling for accent conversion. These embeddings have been extensively validated in the literature and provide a reliable way to encode speaker-specific information globally across the utterance.

It is important to note that improving speaker embedding extraction itself, especially for better generalization to unseen speakers, lies beyond the scope of this thesis. Instead, our focus is on leveraging existing speaker embedding techniques to build effective accent conversion pipelines. Future work could explore developing more powerful, adaptive speaker representations to further enhance the naturalness and speaker consistency in accent-converted speech.

2.1.4. Vocoder

A vocoder (voice coder) Wikipedia (2025d) is a fundamental component in speech synthesis that reconstructs a waveform from intermediate speech representations, such as mel-spectrograms ,acoustic features, or even hidden representations produced by neural networks. Instead of directly generating raw audio waveforms, modern speech synthesis systems first predict intermediate features, which are then converted into high-quality waveforms using a vocoder. This two-step approach enhances the efficiency and flexibility of the synthesis process while maintaining naturalness and intelligibility in the output speech.

In speech synthesis, generating a waveform directly instead of generating mel-spectrogram or other acoustic feature is computationally expensive and challenging due to the high dimensionality of audio signals. Instead, vocoders provide a structured way to synthesize waveforms from feature representations, ensuring smoother and more natural speech. Key advantages of using a vocoder include:

1. **Dimensionality Reduction**: Mel-spectrograms and other features have lower dimensions than raw waveforms, making them easier to model.
2. **Improved Generalization**: Feature-based synthesis captures essential speech characteristics while filtering out noise and redundancy
3. **Efficient and High-Quality Speech Generation**: Neural vocoders can generate speech in real-time while maintaining high fidelity
4. **Robustness to Alignment Errors:** Directly generating raw waveforms is highly sensitive to small temporal shifts; even a slight misalignment can cause significant errors in the waveform domain. Using feature-based targets like spectrograms mitigates this sensitivity, providing a more stable and forgiving learning signal during training

Types of Vocoders

1. Traditional Vocoders : These include Griffin-Lim Wang et al. (2017) Algorithm and WORLD Morise (2016) Vocoder, which model speech as a combination of excitation and spectral envelopes.

2. Neural Vocoders: Recent advnces leverage deep learning models, such as WaveNetShen et al. (2018), WaveGlow Prenger et al. (2018), Parallel WaveGAN Yamamoto et al. (2020), HiFi-GAN Kong et al. (2020a), and VocGAN Yang et al. (2020), to generate high-quality, natural-sounding speech. These models significantly improve speech synthesis realism by learning complex patterns in audio waveforms.

With the rise of deep learning, neural vocoders have largely replaced traditional vocoders in high-quality speech synthesis, enabling more natural and expressive speech generation. On the downside, the neural vocoder needs to be trained with a relatively large amount of speech data, and the training process can be slow. The WaveNet vocoder, the first neural vocoder, has slow inference speed due to its autoregressive nature. However, this issue can be addressed using alternative approaches like Parallel-WaveNet, WaveGlow, and HiFi-GAN. Among these, HiFi-GAN stands out as a state-of-the-art vocoder due to its ability to generate high-fidelity waveforms in real-time while maintaining computational efficiency. In this thesis, we adopt HiFi-GAN for synthesizing audio waveforms, ensuring high-quality and natural-sounding speech in our speech synthesis and accent conversion tasks. Since HiFi-GAN is closely related to speech synthesis, we will provide a more detailed explanation in Section 2.2

2.2. Text-to-Speech Synthesis

An AC and a speech synthesis share the same output modality, meaning that the architectural advancements in speech synthesis can be effectively leveraged for accent conversion models. Neural text-to-speech synthesis, in particular, has emerged as a powerful tool, outperforming statistical TTS methods Mehta et al. (2022). Neural text-to-speech synthesis can be broadly categorized into autoregressive and non-autoregressive architectures, each addressing different challenges in speech generation.

TTS is inherently a one-to-many problem, as the same text input can correspond to multiple valid speech outputs due to variations in prosody, intonation, and speaker style. Addressing this variability has been a key focus in TTS research.

For autoregressive architectures, the most widely used model is Tacotron Wang et al. (2017); Shen et al. (2018) , which generates speech frame by frame in a sequential manner. Tacotron-based models, such as Tacotron 2, employ an encoder-decoder structure where the decoder predicts the next frame based on previously generated frames. While this approach ensures high-quality and natural-sounding speech, it suffers from slow inference speed and potential error accumulaion due to its sequential nature.

To address the inference speed and stability issues of autoregressive models, researchers have developed non-autoregressive architectures, such as Variational Inference Text-to-Speech (VITS) Kim et al. (2021a).

VITS extends TTS capabilities by explicitly modeling the one-to-many problem through stochastic latent variables that capture prosody and speaker-specific variations more effectively than Tacotron. A key component of VITS is the conditional prior distribution from text, which enhances the model's ability to generate diverse speech outputs from the same text input.

In this thesis, the Tacotron Shen et al. (2018) and VITS Kim et al. (2021a) architectures are used as the speech synthesis components of the accent conversion system. This section reviews Tacotron-2 and VITS as key literature relevant to the thesis.

2.2.1. Tacotron

Tacotron Wang et al. (2017) is an autoregressive neural sequence-to-sequence model that synthesizes speech from text by generating mel-spectrograms, followed by a Griffin-Lim vocoder for waveform reconstruction. Tacotron-2 Shen et al. (2018)is an updated version that improves upon the original by incorporating a more advanced vocoder called WaveNet, which enables the generation of higher-quality speech waveforms with improved naturalness and clarity. The key difference in Tacotron-2 is its end-to-end pipeline, where the model directly predicts the mel-spectrogram from the input text, and a separate neural vocoder (WaveNet van den Oord et al. (2016)) is used for the final waveform generation. This eliminates the need for intermediate post-processing and enhances the overall efficiency and quality of the synthesis process.

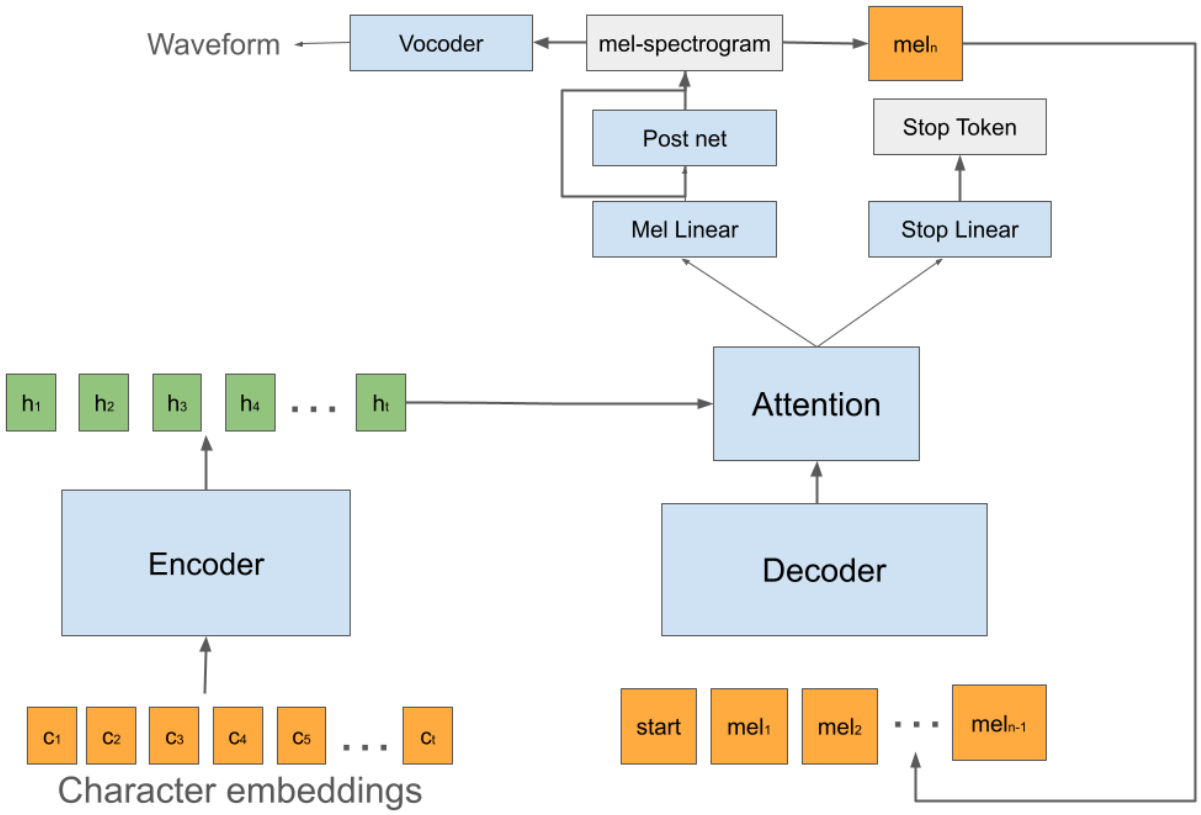

Figure 2.3.: Tacotron 2 architecture Wang et al. (2017); Shen et al. (2018)

Model Architecture

Tacotron-2 architecture consists of:

- **Encoder**: Converts the input text sequence into a high-level latent feature representation.
- **Attention Mechanism**: Dynamically aligns the input text features with the output mel-spectrogram frames.
- **Decoder**: Autoregressively predicts mel-spectrogram frames conditioned on previous outputs and context vectors.
- **Post-Net**: Refines the initially predicted mel-spectrogram by applying a convolutional post-processing network.
- **Stop Token Prediction**: Predicts a binary token at each timestep indicating whether to stop decoding.

The overall Tacotron architecture operates as follows:

The encoder transforms an input sequence $X = \{c_1, c_2, ..., c_t\}$ (phoneme or character embeddings) into a hidden representation $H = \{h_1, h_2, ..., h_t\}$:

$$H = \text{Encoder}(X) \tag{2.3}$$

The attention mechanism calculates a context vector a_n for each decoder step by attending over H based on the previous decoder hidden state s_{n-1}:

$$a_n = \text{Attention}(H, s_{n-1}) \tag{2.4}$$

The decoder autoregressively predicts the mel-spectrogram frames:

$$mel_n = \text{Decoder}(mel_{1:n-1}, a_n) \tag{2.5}$$

where mel_n denotes the predicted mel-spectrogram frame at time step n.
To improve the quality of the predicted mel-spectrogram, a **PostNet**, typically composed of several convolutional layers, is applied:

$$\tilde{Y} = Y + \text{PostNet}(Y) \tag{2.6}$$

where $\tilde{Y}$ represents the refined mel-spectrogram , Y represents the mel-spectrogram $mel_1, mel_2, ..., mel_n\}$
Additionally, Tacotron predicts a **stop token** $stop_n$ at each timestep:

$$stop_n = \text{StopLinear}(s_{n-1}, a_n) \tag{2.7}$$

where $stop_n$ is a scalar between 0 and 1, indicating whether to stop decoding (usually using a threshold during inference).
Finally, the resulting mel-spectrogram $\tilde{Y}$ is converted to a waveform $\hat{s}$ using an external neural vocoder, such as Griffin-Lim, WaveNet, or HiFi-GAN:

$$\hat{s} = \text{Vocoder}(\tilde{Y}) \tag{2.8}$$

The complete Tacotron model is trained jointly with a combination of the following loss functions:

- **Mel-Spectrogram Reconstruction Loss**: Typically an L1 or L2 loss between predicted and ground-truth mel-spectrograms.
- **Post-Net Loss**: A separate reconstruction loss between refined mel-spectrograms and ground-truth mel-spectrograms.
- **Stop Token Loss**: A binary cross-entropy loss for predicting the correct stop point.

Advantages of Tacotron:

- **End-to-end training:** Tacotron 2 eliminates the need for manual feature extraction by learning text-to-speech mappings directly from data, reducing the complexity of the pipeline.
- **High-Quality Speech Generation:** By combining a sequence-to-sequence model with an attention mechanism and a neural vocoder (such as WaveNet), Tacotron 2 produces highly natural and expressive speech with improved fidelity and intelligibility.
- **Post-Net Refinement:** The Post-Net further improves the mel-spectrogram predictions, resulting in smoother and more accurate outputs
- **Flexible and Adaptive:** The model can be easily adapted to various speaker styles or emotional expressions by incorporating speaker embeddings or style token.

By these advantages, the components of Tacotron are utilized as part of the accent conversion architecture, as discussed in Chapter 3

2.2.2. VITS: Variational Inference Text-to-Speech

VITS Kim et al. (2021a) is a non-autoregressive model that synthesizes speech via variational inference, enabling diverse and high-quality speech synthesis. Before introducing VITS, it is essential to understand the three key components that it integrates: **Normalizing Flows** Rezende and Mohamed (2015), **Variational Autoencoders (VAEs)** Kingma and Welling (2019), and **Generative Adversarial Networks (GANs)** Goodfellow et al. (2014). These components collectively enable VITS to generate high-quality and natural-sounding speech.

Normalizing Flows

Normalizing Flows (NFs) are a powerful technique for modeling complex probability distributions by transforming a simple prior distribution (e.g., a standard Gaussian) into a more expressive one through a sequence of invertible, differentiable transformations. Given a random variable z drawn from a simple prior distribution $p(z)$, a normalizing flow applies a series of transformations to obtain a more complex distribution:

$$x = f_\theta(z), \tag{2.9}$$

where f_θ is a bijective function parameterized by θ. By applying the change of variable formula, the transformed probability density function can be computed as:

$$p(x) = p(z) \left| \det \frac{\partial f_\theta^{-1}(x)}{\partial x} \right|, \tag{2.10}$$

where $\det \frac{\partial f_\theta^{-1}(x)}{\partial x}$ is the determinant of the Jacobian matrix of the inverse transformation.
Normalizing flows allow for flexible latent space modeling by successively applying multiple transformations:

$$z_K = f_K \circ f_{K-1} \circ \ldots \circ f_1(z_0), \tag{2.11}$$

where each f_k is an invertible function, and z_0 follows a known simple prior (e.g., Gaussian). The final latent variable z_K can model complex distributions, enabling better data representation.
Normalizing flows are typically trained using the principle of Maximum Likelihood Estimation (MLE). Given a dataset $D = \{x^{(i)}\}_{i=1}^N$ of observed data points, the goal is to find the parameters θ that maximize the likelihood of observing this data under the model distribution $p(x;\theta)$. This is equivalent to maximizing the log-likelihood function:

$$\mathscr{L}(\theta) = \sum_{i=1}^{N} \log p(x^{(i)};\theta). \tag{2.12}$$

Using the change of variables formula for the composition of functions, the log-likelihood for a single data point x can be expressed as:

$$\log p(x;\theta) = \log p_{z_0}(z_0) + \log \left| \det \left(\frac{\partial x}{\partial z_0} \right) \right|^{-1} \tag{2.13}$$

$$= \log p_{z_0}(z_0) - \log \left| \det \left(\frac{\partial x}{\partial z_0} \right) \right| \tag{2.14}$$

$$= \log p_{z_0}(z_0) - \sum_{k=1}^{K} \log \left| \det \left(\frac{\partial z_k}{\partial z_{k-1}} \right) \right|, \tag{2.15}$$

where $z_0 = f_\theta^{-1}(x) = f_1^{-1} \circ \cdots \circ f_K^{-1}(x)$, and $\frac{\partial z_k}{\partial z_{k-1}}$ is the Jacobian of the k-th transformation f_k.
The training objective is therefore to minimize the negative log-likelihood (NLL) averaged over the dataset:

$$\text{minimize}_\theta \quad -\frac{1}{N} \sum_{i=1}^{N} \left(\log p_{z_0}(z_0^{(i)}) - \sum_{k=1}^{K} \log \left| \det \left(\frac{\partial f_k}{\partial z_{k-1}^{(i)}} \right) \right| \right). \tag{2.16}$$

This optimization is typically performed using gradient-based methods such as Stochastic Gradient Descent (SGD) or Adam. The key requirements for normalizing flows are that each transformation f_k must be invertible, and both the forward computation f_k and the logarithm of the absolute determinant of its Jacobian (log-det-Jacobian) must be computationally efficient. This allows for efficient calculation of the likelihood and its gradients with respect to the parameters θ.

Variational Autoencoders (VAEs)

Variational Autoencoders (VAEs) are a type of deep generative model that learns a structured latent space by optimizing a probabilistic lower bound on the data likelihood. Given an input data point x, VAEs introduce a latent variable z and model the marginal likelihood as:

$$p_\theta(x) = \int p_\theta(x,z)dz. \tag{2.17}$$

However, directly computing the marginal likelihood $\log p_\theta(x)$ is typically intractable due to the integral over the latent variable z. To address this, Variational Autoencoders (VAEs) introduce a variational distribution $q_\phi(z \mid x)$ to approximate the true posterior $p_\theta(z \mid x)$, and instead maximize a lower bound on the log-likelihood known as the *Evidence Lower Bound (ELBO)*.
We start by rewriting the marginal log-likelihood as:

$$\log p_\theta(x) = \int \log p_\theta(x)\, q_\phi(z \mid x)\, dz \quad \text{since} \int q_\phi(z \mid x)\, dz = 1 \tag{2.18}$$

Therefore

$$\log p_\theta(x) = \log \mathbb{E}_{q_\phi(z|x)}[p_\theta(x)] \tag{2.19}$$

Furthermore, using the identity

$$p_\theta(x) = \frac{p_\theta(x,z)}{p_\theta(x \mid z)} \tag{2.20}$$

Combine 2.19 and 2.20, we have:

$$\begin{aligned}
\log p_\theta(x) &= \mathbb{E}_{q_\phi(z|x)}\left[\log \frac{p_\theta(x,z)}{p_\theta(x \mid z)}\right] \\
&= \mathbb{E}_{q_\phi(z|x)}\left[\log \frac{p_\theta(x,z)\, q_\phi(z \mid x)}{p_\theta(x \mid z)\, q_\phi(z \mid x)}\right] \\
&= \mathbb{E}_{q_\phi(z|x)}\left[\log \frac{p_\theta(x,z)}{q_\phi(z \mid x)}\right] + \mathbb{E}_{q_\phi(z|x)}\left[\log \frac{q_\phi(z \mid x)}{p_\theta(x \mid z)}\right] \\
&= \mathbb{E}_{q_\phi(z|x)}\left[\log \frac{p_\theta(x,z)}{q_\phi(z \mid x)}\right] + D_{\mathrm{KL}}(q_\phi(z|x)||p_\theta(x \mid z)).
\end{aligned} \tag{2.21}$$

Note that the second term does not correspond to a KL divergence between two distributions over z, so instead we use the correct form:

$$D_{\mathrm{KL}}\left(q_\phi(z \mid x) \,\|\, p_\theta(z \mid x)\right) \geq 0. \tag{2.22}$$

Therefore, we can derive the following inequality:

$$\log p_\theta(x) \geq \mathbb{E}_{q_\phi(z|x)}\left[\log \frac{p_\theta(x,z)}{q_\phi(z \mid x)}\right], \tag{2.23}$$

where the right-hand side is referred to as the **Evidence Lower Bound (ELBO)**. By maximizing the ELBO, we indirectly maximize a lower bound on the true data likelihood $\log p_\theta(x)$.
We now derive an equivalent and more interpretable form of the ELBO. Starting from:

$$\mathscr{L}_{\text{ELBO}} = \mathbb{E}_{q_\phi(z|x)}\left[\log \frac{p_\theta(x,z)}{q_\phi(z \mid x)}\right], \tag{2.24}$$

we apply the factorization $p_\theta(x,z) = p_\theta(x \mid z)p(z)$, which gives:

$$\begin{aligned}\mathscr{L}_{\text{ELBO}} &= \mathbb{E}_{q_\phi(z|x)}\left[\log \frac{p_\theta(x \mid z)p(z)}{q_\phi(z \mid x)}\right] \\ &= \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x \mid z)\right] + \mathbb{E}_{q_\phi(z|x)}\left[\log \frac{p(z)}{q_\phi(z \mid x)}\right] \\ &= \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x \mid z)\right] - D_{\text{KL}}\left(q_\phi(z \mid x) \,\|\, p(z)\right).\end{aligned} \tag{2.25}$$

which separates the objective into two key components:

- **Reconstruction term:** The expectation $\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x \mid z)]$ encourages the decoder $p_\theta(x \mid z)$ to reconstruct the input x well from the latent representation z. This term measures how well the model can explain the observed data given the sampled latent variables.
- **Regularization term:** The KL divergence $D_{\text{KL}}(q_\phi(z \mid x) \,\|\, p(z))$ acts as a regularizer that penalizes the divergence between the approximate posterior $q_\phi(z \mid x)$ and the prior $p(z)$. This prevents the latent space from overfitting to the training data and encourages smooth, structured latent representations.

Maximizing the ELBO thus balances the trade-off between accurate data reconstruction and latent space regularity. In practice, the prior $p(z)$ is often chosen to be a standard normal distribution $\mathscr{N}(0,I)$, which simplifies both sampling and computation of the KL divergence.
This form of the ELBO is particularly useful for optimization, as it can be directly estimated using Monte Carlo sampling during training:

$$\mathscr{L}_{\text{ELBO}} \approx \frac{1}{L}\sum_{l=1}^{L} \log p_\theta(x \mid z^{(l)}) - D_{\text{KL}}(q_\phi(z \mid x) \,\|\, p(z)), \quad \text{where } z^{(l)} \sim q_\phi(z \mid x). \tag{2.26}$$

This formulation allows VAEs to be trained end-to-end using stochastic gradient descent, with gradients backpropagated through both the encoder and decoder networks.

The encoder maps input speech features x into a latent distribution:

$$z \sim q_\phi(z|x) = \mathscr{N}(\mu_\phi(x), \sigma_\phi(x)), \tag{2.27}$$

where $\mu_\phi(x)$ and $\sigma_\phi(x)$ are the learned mean and variance functions. The decoder then reconstructs x from z using:

$$x \sim p_\theta(x|z). \tag{2.28}$$

In speech synthesis, VAEs provide a structured representation of speech variability, such as speaker identity, emotion, and intonation, which improves the diversity and naturalness of generated speech. VITS uses a conditional VAE framework to model latent speech representations conditioned on text.

Generative Adversarial Networks (GANs)

Generative Adversarial Networks (GANs) are a class of generative models that train a generator network G and a discriminator network D in an adversarial framework. The generator attempts to synthesize realistic samples, while the discriminator tries to distinguish real samples from generated ones. The training objective follows a min-max optimization:

$$\min_G \max_D \mathbb{E}_{x\sim p_{\text{data}}(x)}[\log D(x)] + \mathbb{E}_{z\sim p(z)}[\log(1 - D(G(z)))]. \tag{2.29}$$

The generator learns to produce realistic outputs by fooling the discriminator, while the discriminator continuously improves its ability to differentiate real and fake samples.

In speech synthesis, GANs are used to improve the quality of generated waveforms by enforcing a more realistic distribution. Specifically, GAN-based models like HiFi-GAN Kong et al. (2020a) introduce a discriminator that evaluates generated speech at multiple scales (e.g., waveform segments, frequency representations) to ensure high-fidelity synthesis.

HiFi-GAN consists of a generator and multiple discriminators. The generator converts an input mel-spectrogram into a high-fidelity waveform, while the discriminators assess the quality of the generated speech by comparing it to real speech signals. The training process involves a combination of adversarial loss, feature matching loss, and mel-spectrogram loss to improve the perceptual quality of the synthesized speech.

The generator G takes speech features $\mathbf{M}$ as input and produces a waveform $\hat{\mathbf{x}} = G(\mathbf{M})$. The discriminators $\{D_k\}_{k=1}^K$ are designed to distinguish between real speech signals $\mathbf{x}$ and generated waveforms $\hat{\mathbf{x}}$. The adversarial loss $\mathscr{L}_{\text{adv}}$ encourages the generator to produce realistic waveforms by minimizing:

$$\mathscr{L}_{\text{adv}} = \mathbb{E}_{\mathbf{x}}[\log D_k(\mathbf{x})] + \mathbb{E}_{\hat{\mathbf{x}}}[\log(1 - D_k(\hat{\mathbf{x}}))].$$

Additionally, the feature matching loss $\mathscr{L}_{\text{fm}}$ ensures that the intermediate feature representations of the generated and real waveforms are similar:

$$\mathscr{L}_{\text{fm}} = \mathbb{E}_{\mathbf{x},\hat{\mathbf{x}}}\left[\sum_{i=1}^{T} \frac{1}{N_i} \|D_k^{(i)}(\mathbf{x}) - D_k^{(i)}(\hat{\mathbf{x}})\|_1\right],$$

where T is the number of layers in the discriminator, N_i is the number of features in the i-th layer, and $D_k^{(i)}$ denotes the i-th layer of the k-th discriminator.
Finally, the mel-spectrogram loss $\mathscr{L}_{\text{mel}}$ ensures that the generated waveform closely matches the target mel-spectrogram:

$$\mathscr{L}_{\text{mel}} = \|\phi(\mathbf{x}) - \phi(\hat{\mathbf{x}})\|_1,$$

where $\phi(\cdot)$ represents the mel-spectrogram transformation. The overall loss function for the generator is a weighted combination of these losses:

$$\mathscr{L}_G = \mathscr{L}_{\text{adv}} + \lambda_{\text{fm}}\mathscr{L}_{\text{fm}} + \lambda_{\text{mel}}\mathscr{L}_{\text{mel}},$$

where λ_{fm} and λ_{mel} are hyperparameters controlling the contribution of each loss term.
VITS incorporates adversarial training to refine speech synthesis quality, ensuring that generated waveforms closely resemble natural speech. The GAN-based approach enhances prosody, speaker characteristics, and intelligibility while reducing artifacts common in traditional speech synthesis models.

Variational Inference Text-to-Speech (VITS) is a state-of-the-art neural speech synthesis model that combines **Variational Autoencoders (VAEs)**, **Normalizing Flows**, and **Generative Adversarial Networks (GANs)** into a single end-to-end trainable framework. Unlike conventional TTS systems that separately model phonetic duration, acoustic features, and waveform generation, VITS learns a latent speech representation and directly generates high-fidelity waveforms.

Overview of VITS Framework

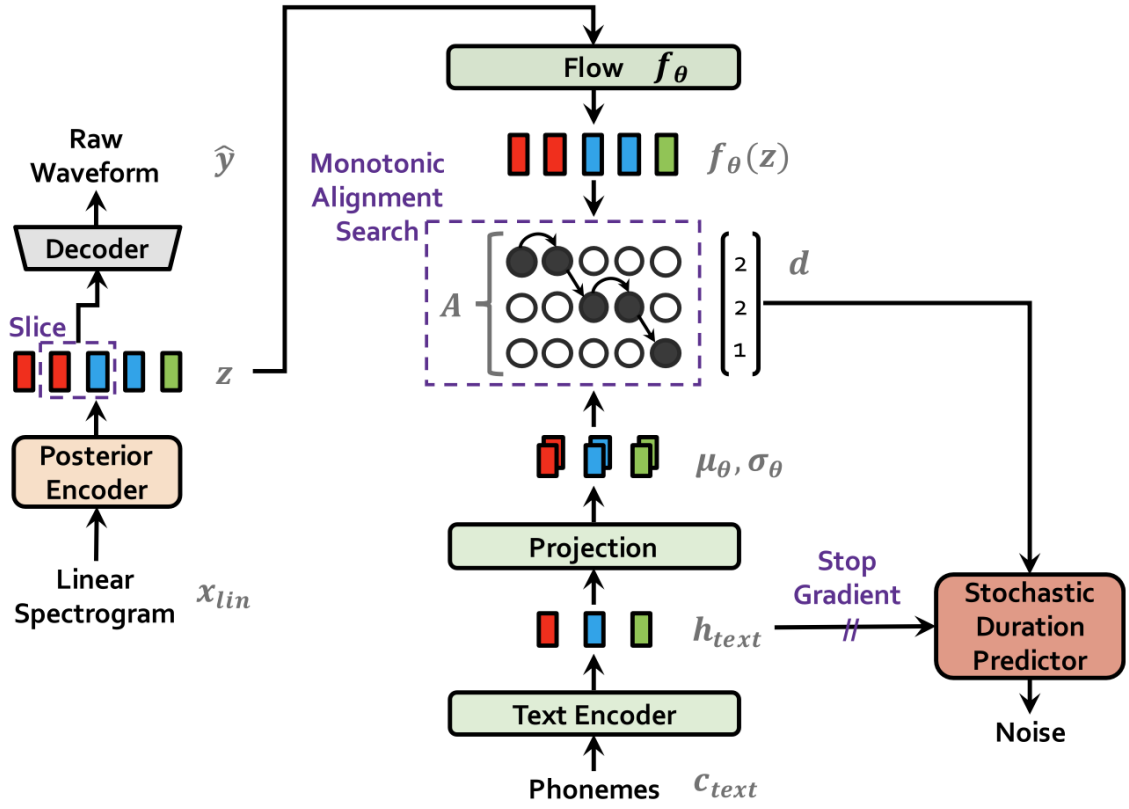

Figure 2.4.: VITS training process Kim et al. (2021a)

VITS (Variational Inference Text-to-Speech) models speech synthesis as a latent-variable probabilistic generative process. It aims to model the conditional distribution:

$$p(x|c) = \int p(x|z)p(z|c)dz, \tag{2.30}$$

where:

- x is the speech waveform,
- c is the text input (e.g., phoneme sequence),
- z is a latent variable capturing intermediate speech representations.

Since direct marginalization over z is intractable, VITS employs variational inference to approximate the posterior distribution $p(z|x,c)$. The model consists of these major components:

- **Text Conditional Prior Network**: Encodes text embeddings into a prior distribution over the latent space

- **Posterior Encoder with Normalizing Flow**: Maps speech waveforms to a latent space using normalizing flows.

- **Monotonic Alignment Module and Duration Predictor**: Provides soft and differentiable alignment between text and speech.

- **Waveform Generator (HiFi-GAN)**: Generates high-quality waveforms from latent variables via adversarial training

Text Conditional Prior

One of the key innovations of VITS is the use of a **conditional prior distribution** over the latent variable z. Given a phoneme sequence c, the model learns a structured prior $p(z|c)$ using a normalizing flow. This conditional prior enables flexible modeling of the many-to-one problem in speech synthesis, where multiple speech waveforms can correspond to the same text. The prior distribution over the latent space is modeled as:

$$p(z_0|c) = \mathcal{N}(z_0; \mu_\theta(c), \sigma_\theta^2(c), \tag{2.31}$$

where $\mu_\theta(y)$ and $\sigma_\theta(y)$ are predicted from the text. To model more complex latent distributions, VITS applies a normalizing flow transformation f:

$$z = f(z_0), \tag{2.32}$$

allowing computation of the prior using the change of variables formula as:

$$p(z|c) = p(z_0|c) \left| \det \left(\frac{\partial f^{-1}(z)}{\partial z} \right) \right|. \tag{2.33}$$

Posterior Latent Encoding

To capture the complex variations in speech, VITS employs a **posterior encoder** that maps an observed speech waveform x to a latent variable z:

$$q_\phi(z|x) = \mathcal{N}(z; \mu_\phi(x), \sigma_\phi^2(x)), \tag{2.34}$$

Since the true posterior $p(z|x,c)$ is intractable, this approximation ensures that latent variables encode meaningful speech characteristics while remaining close to the prior. The KL divergence between the posterior and prior distributions is minimized to regularize latent space representations:

$$D_{\mathrm{KL}}(q_\phi(z|x)||p_\theta(z|c). \tag{2.35}$$

Monotonic Alignment and Duration Predictor

VITS enforces a monotonic alignment between text and speech using a duration predictor $\Delta(c)$. Durations $\{d_1, d_2, ..., d_T\}$ are predicted for each token, and the alignment path π maximizes the accumulated attention probability under a monotonic constraint:

$$\pi^* = \arg \max_{\pi \in \text{MonoAlign}} \sum_{(t,n) \in \pi} \log \alpha_{t,n}, \tag{2.36}$$

where $\alpha_{t,n}$ represents the alignment probability between text token t and speech frame n.
This design allows fully end-to-end training without external alignments or complex attention mechanisms.

Waveform Generation with Hifi-GAN

The final step in VITS is speech synthesis via Hifi-GAN. Instead of using an explicit vocoder, VITS integrates a Hifi-GAN vocoder to directly generate waveforms from the latent variable z. The generator G takes z as input and produces a speech waveform:

$$x = G_\psi(z) \tag{2.37}$$

A discriminator network D is trained to distinguish real speech samples from generated ones, optimizing the adversarial objective:

$$\min_G \max_D \mathbb{E}_{x \sim p_{\text{data}}(x)}[\log D(x)] + \mathbb{E}_{z \sim p(z)}[\log(1 - D(G(z)))]. \tag{2.38}$$

Final Training Objective

The overall training objective of VITS combines three loss functions:

- Reconstruction Loss (Variational Autoencoder loss):

$$\mathscr{L}_{\text{recon}} = \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)]. \tag{2.39}$$

- KL Divergence Regularization:

$$\mathscr{L}_{\text{KL}} = D_{\text{KL}}(q_\phi(z|x) || p_\theta(z|c)). \tag{2.40}$$

- Adversarial Loss (GAN-based loss for waveform generation): $\mathscr{L}_{\text{GAN}}$

- Duration Predictor Loss: $\mathscr{L}_{\text{Dur}}$

This results in fast, high-quality speech generation without autoregressive sampling.
Thus, the final loss function is:

$$\mathscr{L} = \mathscr{L}_{\text{recon}} + \lambda_{\text{KL}}\mathscr{L}_{\text{KL}} + \lambda_{\text{duration}}\mathscr{L}_{\text{Dur}} + \lambda_{\text{GAN}}\mathscr{L}_{\text{GAN}}, \tag{2.41}$$

where λ_{KL} and λ_{GAN} are weighting factors that balance the different loss terms..

VITS Inference (Fast and Deterministic Generation)

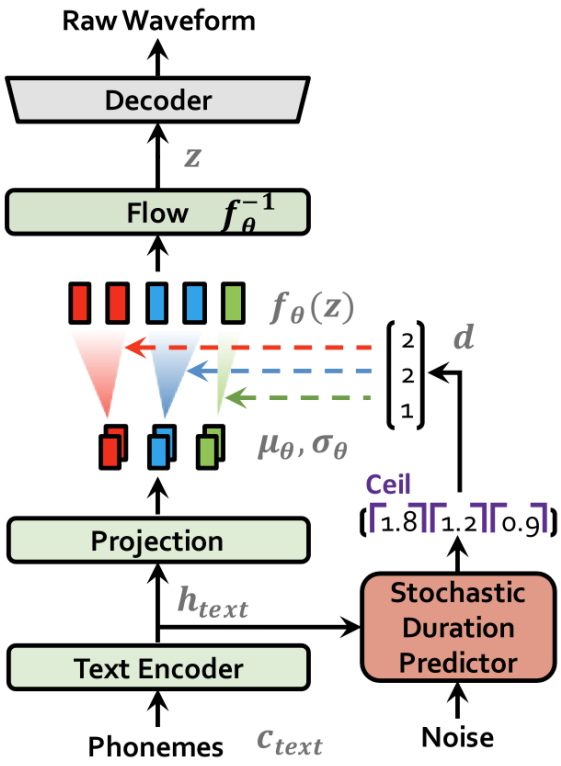

Figure 2.5.: VITS inference process Kim et al. (2021a)

During inference, the speech waveform is generated deterministically through the following steps:

1. Encode text c into prior parameters $(\mu_\theta, \sigma_\theta)$.
2. Predict the phoneme durations using the duration predictor $\Delta(c)$, which determines the alignment between text tokens and speech frames.
3. Expand the prior parameters $(\mu_\theta, \sigma_\theta)$ according to the predicted durations to match the target frame length.
4. Sample latent variable z_0 from $\mathcal{N}(\mu_\theta, \sigma_\theta)$.

5. Transform z_0 via normalizing flow: $z = f(z_0)$.

6. Generate waveform: $x = G_\psi(z)$.

This results in fast, high-quality speech generation without autoregressive sampling.

Advantages of VITS

VITS introduces several improvements over conventional text-to-speech models:

- **End-to-end training:** Eliminates the need for vocoder modules.
- **Expressive speech synthesis:** The learned latent space allows for modeling speaker identity, prosody, and style.
- **High-quality waveform generation:** Adversarial training improves speech naturalness and reduces artifacts.
- **Fast inference:** VITS enables efficient and deterministic generation of speech waveforms, allowing for faster synthesis due to non-autoregressive inference.

Conclusion

By leveraging HiFi-GAN for waveform generation, VITS achieves superior speech quality while maintaining efficiency. The adversarial training framework ensures that synthesized speech is perceptually natural, making HiFi-GAN an essential component in modern speech synthesis pipelines. Due to its advantages, VITS serves as the backbone for the controllable accented TTS model presented in Chapter 4.

2.3. Speech Recognition

The goal of accent conversion is to improve the intelligibility of speech; therefore, understanding and successfully capturing the content of speech is crucial for effective accent conversion. In this thesis, the speech recognition model has been utilized multiple times as a part of the content encoder in the accent conversion process. Additionally, speech recognition is also used to evaluate the performance of speech synthesis systems in general, which can be leveraged to assess the performance of the accent conversion system. This section focuses on neural network-driven approaches to speech recognition, which have been shown to outperform traditional statistical methods Raissi et al. (2022). Among them, sequence-to-sequence (Seq2Seq) Chan et al. (2016) Nguyen et al. (2020c) models and connectionist temporal classification (CTC)-based models Graves et al. (2006)have gained prominence due to their ability to handle variable-length input and output sequences.

2.3.1. Sequence-to-Sequence (Seq2Seq) Models

Seq2Seq-based speech recognition systems Chan et al. (2016); Pham et al. (2020); Nguyen (2021); Nguyen et al. (2020a,b) typically follow an encoder-decoder architecture, often augmented with attention mechanisms. These models have demonstrated state-of-the-art performance in various speech recognition tasks due to their ability to model long-range dependencies and directly map acoustic features to text. The main components of such systems include:

- **Encoder**: Processes the input speech features (e.g., mel spectrograms) and generates a high-dimensional representation.
- **Decoder**: Takes the encoder output and generates a sequence of words, characters, or phonemes autoregressively.
- **Attention Mechanism**: Helps the decoder focus on relevant parts of the input sequence, improving accuracy.

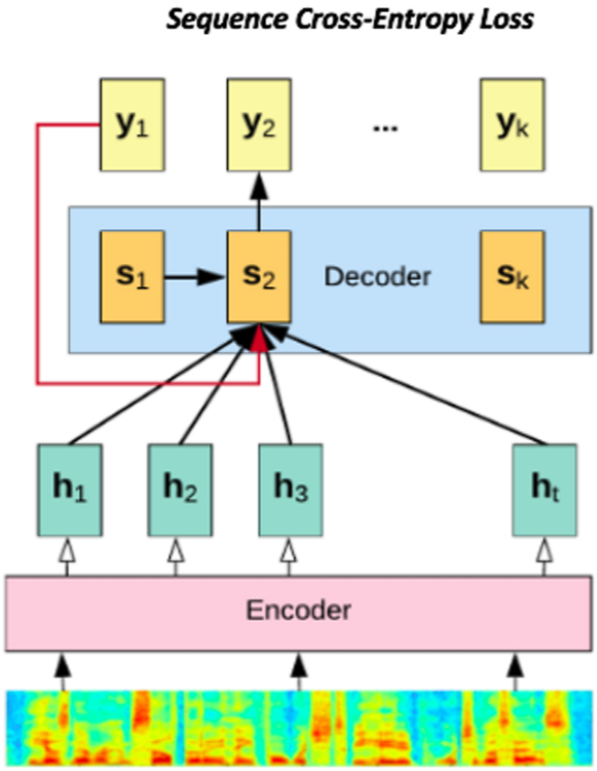

Figure 2.6.: Sequence-to-sequence speech recognition Nguyen (2021)

Mathematically, given an input sequence of feature frames X, the encoder transforms it into a latent representation H:

$$H = f_{\text{enc}}(X) \tag{2.42}$$

where f_{enc} represents the encoder function.

At each decoding step, the decoder does not operate on the entire encoder output directly. Instead, it uses an attention mechanism to dynamically select and weigh the most relevant parts of H based on the current decoding context. Specifically, the decoder maintains a recurrent hidden state s_n and computes an attention context vector c_n as a weighted sum over the encoder outputs:

$$c_n = \sum_t \alpha_{t,n} h_t \tag{2.43}$$

where the attention weights $\alpha_{t,n}$ are computed via a scaled dot-product attention:

$$\alpha_{t,n} = \frac{\exp(e_{t,n}/\sqrt{d})}{\sum_{t'} \exp(e_{t',n}/\sqrt{d})}, \tag{2.44}$$

and the alignment score $e_{t,n}$ measures the compatibility between the encoder output h_t and the decoder state s_n:

$$e_{t,n} = \frac{h_t \cdot s_n}{\sqrt{d}}. \tag{2.45}$$

The decoder then generates the next token y_n conditioned on both the previously generated outputs $y_{1:n-1}$ and the attention context vector c_n:

$$P(y_n|H, y_{1:n-1}) = f_{\text{dec}}(s_n, c_n) \tag{2.46}$$

where f_{dec} denotes the decoder function.

Thus, the attention mechanism serves as a bridge that connects the encoder outputs to the decoder at each time step, ensuring that the decoder focuses on the most relevant parts of the input sequence for generating accurate predictions.

During training, the model is optimized to minimize the cross-entropy loss between the predicted output sequence and the ground-truth labels. The cross-entropy loss for a given training example is formulated as:

$$\mathscr{L}_{CE} = -\sum_{n=1}^{N} \log P(y_n^*|H, y_{1:n-1}^*) \tag{2.47}$$

where y_n^* denotes the ground-truth token at position n, and $P(y_n^*|H, y_{1:n-1}^*)$ is the predicted probability of the correct token given the encoder output and the previous ground-truth tokens.

Advantages of Seq2Seq Models:

- **End-to-End Training:** Sequence-to-sequence models allow the entire speech recognition pipeline—traditionall split into acoustic, language, and pronunciation modeling—to be trained jointly. This simplification reduces system complexity and enables better global optimization.

- **Modeling Long-Range Dependencies:** Thanks to the encoder-decoder architecture and attention mechanisms, Seq2Seq models can effectively capture long-range temporal dependencies, which are essential for understanding context across long utterances.

- **Direct Mapping without Alignment:** Unlike classical systems that require frame-level phoneme alignments, Seq2Seq models can learn to map audio features (e.g., Mel-spectrograms) directly to textual outputs, streamlining data preparation and training.

Challenges:

- **Large Data Requirements** Achieving strong performance typically demands large amounts of labeled training data, which can be expensive and time-consuming to collect, especially for low-resource languages or accented speech.

- **High Computational Costs:** Training and inference with autoregressive decoders are often slower and more computationally intensive compared to non-autoregressive or conventional hybrid models, posing difficulties for deployment in real-time or resource-constrained environments.

- **Attention Instability on Long Sequences:** When dealing with long input sequences, the attention mechanism can become unstable, leading to degraded performance. Additionally, memory and computational costs grow quadratically with sequence length, limiting scalability.

Seq2Seq models initially achieved success in machine translation. Later, they were adopted by the speech recognition community to develop end-to-end automatic speech recognition systems. Similarly, the speech synthesis community leveraged Seq2Seq models to create end-to-end TTS synthesizers. Given the sequential and monotonic nature of speech signals, Seq2Seq models and attention mechanisms are particularly well-suited for speech-related tasks. This thesis extensively utilizes this model type for accent conversion.

2.3.2. Connectionist Temporal Classification (CTC)-Based Speech Recognition

CTC-based models Graves et al. (2006); Nguyen (2021) provide an alternative approach to speech recognition by addressing the alignment issue between input audio and output text. Unlike Seq2Seq models, CTC does not require explicit alignment between speech frames and target labels. Instead, it introduces a special blank token and relies on probabilistic alignment learning.

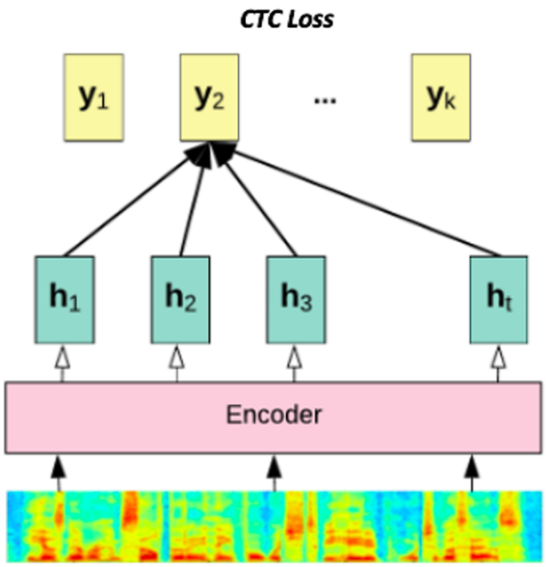

Figure 2.7.: CTC-based speech recognition Nguyen (2021)

The CTC model defines a probability distribution over all possible alignments between input feature sequences and output sequences :

$$P(Y|X) = \sum_{A \in \mathscr{A}(Y)} P(A|X) \tag{2.48}$$

where is the set of all valid alignments allowing repeated and blank tokens.
CTC loss is computed by summing over all possible alignments:

$$\mathscr{L}_{CTC} = - \sum_{A \in \mathscr{A}(Y)} P(A|X) \tag{2.49}$$

which is efficiently computed using the forward-backward algorithm.
Key components of CTC-based models:

- **Acoustic Encoder**: Extracts high-level features from speech signals, typically using convolutional or recurrent layers.
- **CTC Loss Function**: Computes the probability of all possible alignments and optimizes the most likely transcription.
- **Decoding with Beam Search**: Finds the best word sequence by considering multiple possible alignments.

Advantages of CTC models:

- **Efficient end-to-end training:** CTC models eliminate the need for frame-level alignments or phoneme annotations. The model learns to align speech frames to output labels implicitly through the CTC loss function.

- **Real-time inference capability:** Since CTC is non-autoregressive, it enables parallel computation and fast decoding, making it suitable for streaming or low-latency applications.

- **Simplified architecture:** A typical CTC-based system includes only an encoder and a softmax output layer, allowing for a compact and easily trainable model.

Challenges:

- **Conditional independence assumption:** CTC assumes that the output predictions at each time step are conditionally independent. This limits the model's ability to capture dependencies between tokens in the output sequence.

- **Reliance on language model:** Due to the above limitation, an external language model (LM) is often required during decoding to improve grammaticality and fluency.

Integration with Language Models:
To compensate for the conditional independence assumption, CTC decoding is commonly enhanced using external language models via *shallow fusion* Wick et al. (2022). In this method, the score for a hypothesis Y given input X is computed as:

$$\log P_{\text{combined}}(Y|X) = \log P_{\text{CTC}}(Y|X) + \lambda \log P_{\text{LM}}(Y) \tag{2.50}$$

where $P_{\text{CTC}}(Y|X)$ is the probability of sequence Y from the CTC model, $P_{\text{LM}}(Y)$ is the score from an external language model, and λ is a tunable weight balancing the two components.

This fusion helps produce more fluent and grammatically correct outputs while preserving the efficiency of the CTC framework.

Due to its non-autoregressive nature, CTC-based speech recognition can process input very fast, making it highly efficient for streaming applications. This makes it particularly suitable for capturing speech content in fast and streaming accent conversion tasks.

2.4. Self-Supervised Learning for Speech Representation

2.4.1. Overview of Self-Supervised Speech Learning

Self-supervised learning (SSL) has revolutionized the field of speech processing by enabling models to learn powerful representations from unlabeled audio data. Instead of relying on costly manual transcriptions, SSL models are trained with tasks that allow them to predict masked or future portions of raw speech signals. Prominent examples include Wav2Vec 2.0 Baevski et al. (2020b), HuBERT Hsu et al. (2021a), and WavLM Chen et al. (2022), all of which have demonstrated outstanding performance across a wide range of downstream speech tasks.

2.4.2. Advantages of Self-Supervised Speech Learning

Benefits for Feature Extraction

SSL models learn hierarchical, multi-layered representations that capture rich acoustic, phonetic, and even semantic information Li et al. (2023). Lower layers tend to encode fine-grained acoustic features (e.g., pitch, energy), while higher layers abstract linguistic content and speaker characteristics. These representations serve as highly informative feature extractors, enabling downstream systems to access meaningful information without requiring large supervised datasets. In particular, features extracted from SSL models have proven highly effective for tasks such as automatic speech recognition (ASR), speaker recognition, and speech synthesis(voice conversion and text-to-speech).

Weight Initialization for Downstream Models

Beyond feature extraction, SSL models also offer excellent initialization for training larger systems Baevski et al. (2020a). Fine-tuning downstream models from a pretrained SSL checkpoint leads to faster convergence, improved generalization, and often superior final performance compared to training from scratch. By initializing the encoder with SSL-pretrained weights, models can start from a rich latent space, rather than random parameters, thus requiring less supervised data to achieve good performance.

Advantages in Low-Resource Settings

One of the most transformative impacts of SSL is its ability to enable strong performance in low-resource settings Hsu et al. (2021a). Traditional supervised learning requires thousands of hours of labeled speech, but SSL models trained on unlabeled corpora allow fine-tuning with only a few hours of labeled data. This democratizes access to speech technology for low-resource languages and aceents, making it possible to develop effective systems even where labeled data is scarce or expensive to obtain.

Robustness to Noisy Conditions

Self-supervised speech models are typically trained on large and diverse datasets containing a wide range of acoustic environments, including noisy, reverberant, and multi-speaker conditions Chen et al. (2022). As a result, the learned feature representations tend to be more robust against various types of distortions compared to features derived from supervised models trained on clean data only. This robustness is highly beneficial for downstream tasks like speech recognition or voice conversion, where real-world input speech may contain background noise, microphone variability, or other artifacts. By leveraging self-supervised features, models can generalize better to unseen and noisy conditions without requiring explicit noise modeling or additional denoising steps.

Cross-Task Transferability

Another key advantage of self-supervised learning is the broad transferability of the learned representations across multiple speech processing tasks. Since SSL models are optimized to capture general speech patterns without being constrained to a specific task, their representations can be effectively adapted for various downstream applications, including speech recognition, speaker verification, emotion recognition, prosody prediction and speech synthesis. This cross-task adaptability reduces the need for task-specific feature engineering and enables a unified framework where a single set of features can support multiple objectives.

2.4.3. Application in This Thesis

In this thesis, we leverage SSL-pretrained models such for multiple purposes: (1) content feature extraction to obtain robust, speaker-aware or speaker-independent representations, (2) weight initialization for faster and more stable model training, and (3) facilitating strong performance under low-resource training conditions. Improving SSL pretraining itself is beyond the scope of this work; instead, we focus on effectively applying existing state-of-the-art SSL models to downstream accent conversion tasks. We did use the three state-of-the-art SSL-pretrained models such as Wav2vec, WavLM and Hubert.

Self-Supervised Learning with Wav2vec 2.0

Wav2vec 2.0 is a self-supervised learning framework designed to extract meaningful speech representations from unlabeled audio data. The model consists of two main components: a feature encoder and a context encoder.

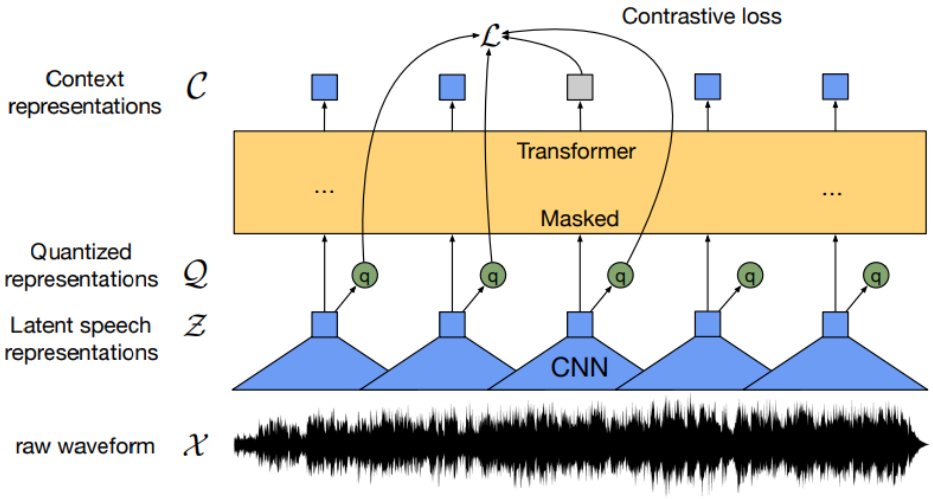

Figure 2.8.: Wav2vec 2.0 architecture Baevski et al. (2020b)

Feature Encoder: The feature encoder CNN_{enc} consists of multiple temporal convolutional layers that process raw waveform input X into a sequence of latent feature representations:

$$Z = CNN_{enc}(X), \quad Z \in \mathbb{R}^{T' \times d} \tag{2.51}$$

where X is the input waveform, T' is the number of extracted frames, and d is the dimensionality of the latent features.

Context Encoder: The latent features Z are then fed into a transformer-based context encoder $\mathscr{T}_{enc}$, which models long-range dependencies and enhances the speech representations:

$$C = \mathscr{T}_{enc}(Z), \quad C \in \mathbb{R}^{T' \times d} \tag{2.52}$$

where C represents the contextualized speech representations, which are crucial for downstream tasks.

Vector Quantization (VQ) Module. To enable a discrete contrastive learning objective, Wav2vec 2.0 uses a quantization module van den Oord et al. (2017) that maps continuous features Z to a finite set of codebook embeddings:

$$Q = f_{vq}(Z), \quad Q \in \mathbb{R}^{T' \times d} \tag{2.53}$$

These quantized embeddings serve as targets during pretraining.

Contrastive Learning Objective. Wav2vec 2.0 is trained using a contrastive objective. A portion of the latent sequence Z is masked, and the model is trained to identify the correct quantized representation Q_t corresponding to each masked timestep t, among a set of negative distractors $\mathscr{N}_t$:

$$L_{contrast} = -\sum_t \log \frac{\exp(\text{sim}(C_t, Q_t)/\tau)}{\sum_{\tilde{Q} \in \mathscr{N}_t} \exp(\text{sim}(C_t, \tilde{Q})/\tau)} \tag{2.54}$$

where $\text{sim}(\cdot,\cdot)$ represents cosine similarity, τ is the temperature scaling parameter, and $\mathscr{N}_t$ is the set of negative samples.

Advantages of Wav2Vec 2.0 : Wav2Vec 2.0 offers several advantages for speech representation learning. Its contrastive self-supervised objective enables learning from large amounts of unlabeled speech data, making it highly scalable. The Transformer context encoder captures long-range dependencies, resulting in rich and robust representations. Additionally, the pretrained features significantly improve data efficiency, robustness to noise, and transferability across various speech processing tasks.

Self-Supervised Learning with HuBERT

HuBERT (Hidden-Unit BERT) is a self-supervised learning model for speech representation learning, introduced by Hsu et al. (2021a). It leverages a masked prediction objective inspired by BERT (Bidirectional Encoder Representations from Transformers) to learn effective speech representations. The model consists of several key stages, which we detail below.

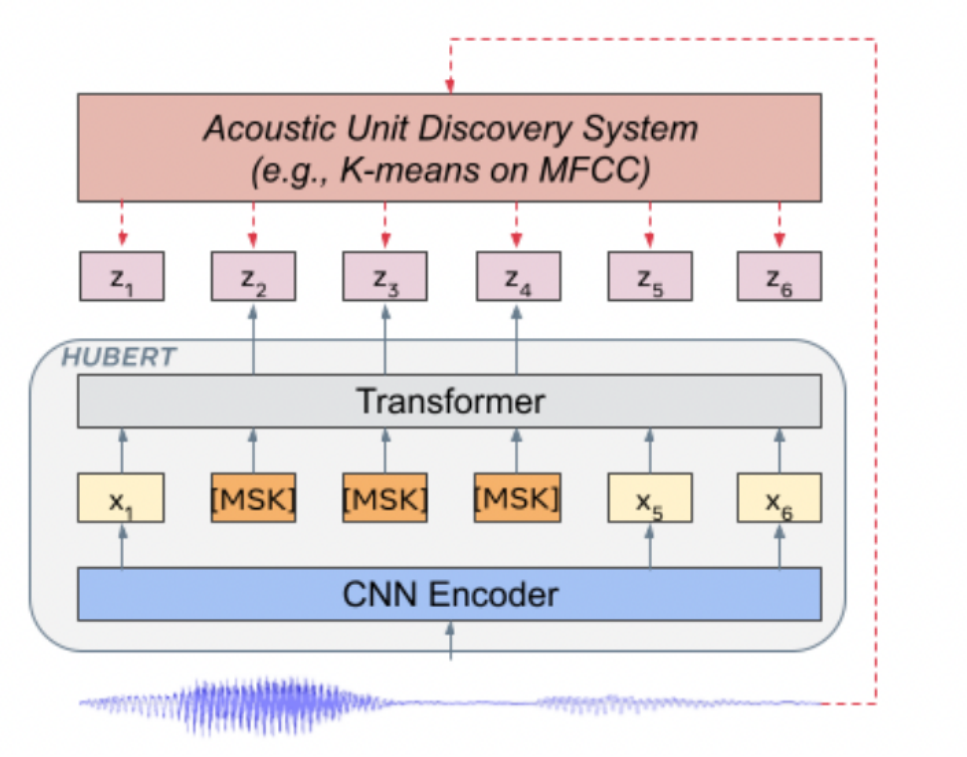

Figure 2.9.: HuBERT architecture Hsu et al. (2021a)

Preprocessing and Feature Extraction: HuBERT processes raw speech waveforms by first extracting a sequence of feature representations using a convolutional neural network (CNN). This is done in the first step, where an input waveform x is passed through a convolutional encoder to produce the feature representations f:

$$f = \text{CNN}_{\text{enc}}(x). \tag{2.55}$$

The extracted features, f, are then fed into a transformer encoder for further processing.

Contextualized Representations via Transformer Encoder: The features obtained from the convolutional encoder are passed through a Transformer encoder $\mathscr{T}_{enc}$ to produce contextualized representations of the audio. The transformer model allows HuBERT to capture long-range dependencies in speech, which is crucial for understanding the phonetic and linguistic structure of the audio:

$$h = \mathscr{T}(f), \tag{2.56}$$

where h represents the output of the transformer encoder, containing the rich, context-aware speech representations.

Clustering with K-means: To group the continuous representations into discrete units, HuBERT employs an iterative clustering technique using K-means clustering. For each frame h_t, the model assigns

it to one of the clusters c_t by minimizing the squared distance between the frame representation and the centroids of the clusters:

$$c_t = \arg\min_k \|h_t - \mu_k\|^2, \tag{2.57}$$

where μ_k denotes the centroid of the k-th cluster. This step discretizes the continuous representations into a set of symbolic cluster assignments, which are treated as pseudo-labels for the next phase of training.

Masked Prediction Objective: HuBERT is trained using a masked prediction objective, where a portion of the cluster assignments is randomly masked, and the model is trained to predict these masked assignments. This approach is similar to masked language model, but applied to the speech domain. The model learns to predict the masked cluster labels $C = \{c_1, c_2, \ldots, c_T\}$ by minimizing a cross-entropy loss:

$$\mathscr{L}_{\text{HuBERT}} = -\sum_{t \in \mathscr{M}} \log P(c_t | h_t), \tag{2.58}$$

where $\mathscr{M}$ is the set of masked time steps. This objective forces the model to learn useful representations of speech that can be used for various downstream tasks, such as phoneme classification, speaker recognition, and speech-to-text.

Iterative Refinement: One of the distinctive features of HuBERT is its iterative refinement process. After the initial clustering step, the model is trained to predict the masked labels, and the resulting cluster assignments are used to improve the model's pseudo-labels. This iterative process allows HuBERT to refine its speech representations over multiple training stages, leading to better performance on downstream tasks.

Advantages of HuBERT : HuBERT has several advantages over previous speech representation learning models. By using a masked prediction objective, it can learn from unlabelled speech data, which makes it highly scalable to large speech datasets. Additionally, the use of a Transformer encoder allows HuBERT to capture complex dependencies in speech, leading to more accurate and robust representations. Finally, the iterative refinement process helps the model to continually improve the quality of its learned representations.

Self-Supervised Learning with WavLM

WavLM Chen et al. (2022) extends the ideas of Wav2Vec 2.0 and HuBERT by incorporating a unified framework for speech representation learning that is optimized for both speech recognition and speech understanding tasks. It is trained with a masked speech prediction objective, similar to HuBERT, but introduces an innovative training strategy called "utterance mixing" to better model speaker and noise variations.

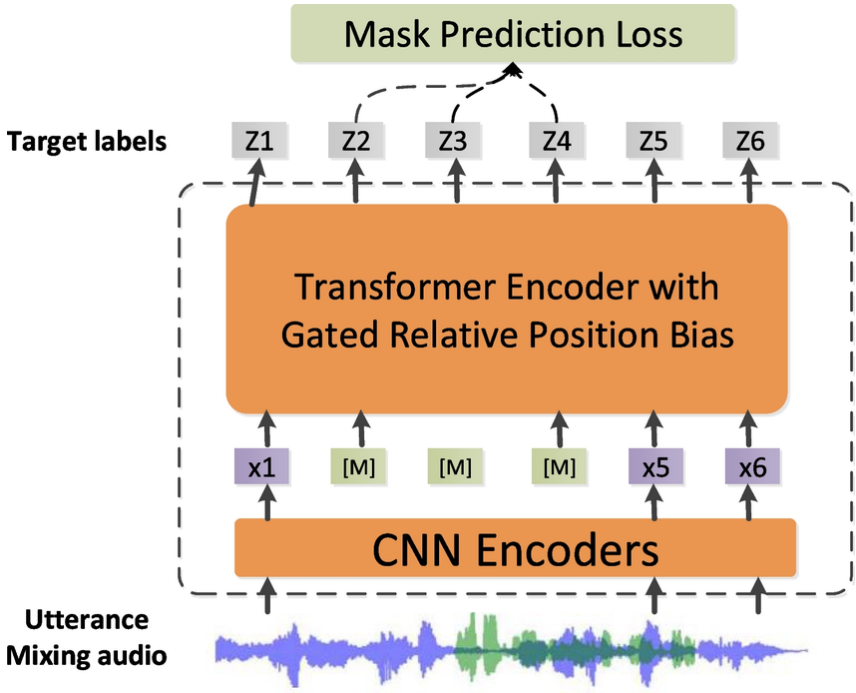

Figure 2.10.: WavLM architecture Chen et al. (2022)

Feature Encoder: Similar to Wav2Vec 2.0, WavLM uses a multi-layer convolutional feature encoder that transforms raw audio signals into latent representations:

$$Z = \text{CNN}_{enc}(X), \quad Z \in \mathbb{R}^{T' \times d} \tag{2.59}$$

Transformer Context Encoder: The extracted latent features Z are then passed through a deep Transformer-based context encoder $\mathscr{T}_{enc}$ to produce contextualized representations:

$$C = \mathscr{T}_{enc}(Z), \quad C \in \mathbb{R}^{T' \times d} \tag{2.60}$$

Utterance Mixing: During training, WavLM randomly mixes two audio segments and trains the model to predict the masked tokens from the mixed signal. This approach teaches the model to disentangle speaker, content, and noise information, leading to more robust and generalized representations.
Masked Prediction Objective: WavLM adopts a masked prediction loss, where a portion of the input is masked, and the model is tasked with predicting the correct quantized targets. This helps the model learn meaningful speech representations without requiring manual labels.

Advantages of WavLM : WavLM brings several improvements over earlier SSL models. The utterance mixing technique enhances robustness to speaker variations and background noise. Its unified pretraining objective enables strong performance across diverse downstream tasks, including ASR, speaker verification, and emotion recognition. Moreover, WavLM achieves excellent results in low-resource settings, demonstrating superior data efficiency and generalization ability compared to previous models.

2.5. Voice conversion

Voice conversion aims to modify a source speaker's voice to sound like a target speaker while preserving the linguistic content. Various approaches have been developed to tackle this task, typically categorized by how they handle the decomposition and transformation of speech features. In this section, we summarize two prominent paradigms in VC research: the disentangle-resynthesis approach and the direct transformation approach.

2.5.1. Disentangle-Resynthesis Approach

The disentangle-resynthesis approach, exemplified by models such as *VQMIVC* Wang et al. (2021a), *AutoVC* Qian et al. (2019), **FreeVC** Li et al. (2023) and *AdainVC* chieh Chou et al. (2019), involves separating the speaker identity and linguistic content from the source speech and then recombining them with the target speaker's characteristics. This is typically achieved using models such as Variational Autoencoders (VAEs) or Generative Adversarial Networks (GANs). For instance, in a VAE-based system, the encoder extracts a latent representation that captures the speaker-independent content, while the decoder reconstructs the speech using the target speaker's voice characteristics. Mathematically, this process can be represented as:

$$\mathbf{z} = \text{Encoder}(\mathbf{x}_{\text{source}}), \quad \hat{\mathbf{x}}_{\text{target}} = \text{Decoder}(\mathbf{z}, \mathbf{s}_{\text{target}}),$$

where $\mathbf{z}$ is the latent representation of the linguistic content, $\mathbf{x}_{\text{source}}$ is the source speech, $\mathbf{s}_{\text{target}}$ is the target speaker's embedding, and $\hat{\mathbf{x}}_{\text{target}}$ is the converted speech.

In this thesis, we employ two disentangle-resynthesis models, VQMIVC and FreeVC, to augment additional data for training the accent conversion system, as will be discussed later. Moreover, FreeVC also serves as the backbone architecture for the disentangle-resynthesis accent conversion method described in Chapter 6.

VQMIVC: Vector Quantization Autoencoder-Based Voice Conversion

VQMIVC (Vector Quantization Mutual Information Voice Conversion) takes an alternative approach by leveraging an autoencoder architecture to achieve voice conversion while disentangling speaker, content, and prosodic features. The model consists of four core modules:

- **Content Encoder** E_c: Extracts phonetic content embeddings z_c from input speech x.
- **Speaker Encoder** E_s: Produces a speaker identity embedding z_s (D-vector) from input speech x.
- **Pitch Encoder** E_p: Extracts prosody-related information to create a prosody embedding z_p.
- **Decoder** $\mathscr{D}$: Reconstructs speech from the content, speaker, and prosody embeddings.

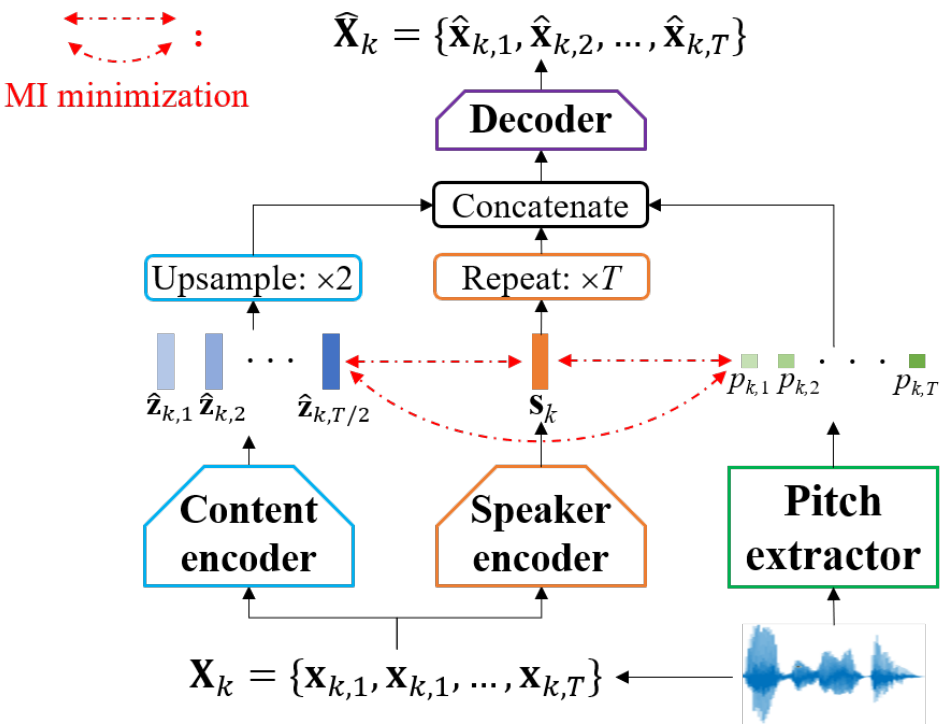

Figure 2.11.: VQMIVC architecture Wang et al. (2021a)

The content embedding z_c is computed as:

$$z_c = E_c(x) \tag{2.61}$$

and is further discretized using vector quantization:

$$\hat{z}_c = \mathrm{VQ}(z_c) \tag{2.62}$$

The speaker embedding z_s is obtained by:

$$z_s = E_s(x) \tag{2.63}$$

The pitch (prosody) embedding z_p is extracted as:

$$z_p = E_p(x) \tag{2.64}$$

Finally, the reconstruction of the speech waveform $\hat{x}$ is obtained as follows:

$$\hat{x} = \mathscr{D}(\hat{z}_c, z_p, z_s) \tag{2.65}$$

To ensure the proper disentanglement of representations, VQMIVC employs a mutual information loss $\mathscr{L}_{MI}$ that minimizes the dependence between speaker, prosody, and phonetic representations:

$$\mathscr{L}_{MI} = -I(z_c, z_s) - I(z_c, z_p) \tag{2.66}$$

where $I(a,b)$ denotes the mutual information between two latent variables a and b.
During inference, the voice conversion process follows these steps:

1. The **source speech** x_{src} is passed through the content encoder and pitch encoder to extract z_c and z_p.
2. The **target speech** x_{tgt} is fed into the speaker encoder to extract z_s.
3. The **decoder** reconstructs the converted speech:

$$\hat{x} = \mathscr{D}(z_c, z_p, z_s) \tag{2.67}$$

In conclusion, VQMIVC effectively addresses the disentanglement problem in voice conversion by leveraging vector quantization and mutual information. The model's architecture disentangles speaker identity, content, and prosody, enabling more accurate and flexible voice conversion. By using mutual information loss, VQMIVC ensures that these features remain independent, facilitating high-quality conversion between different speakers while preserving the phonetic content and prosodic characteristics. This approach provides a robust solution to the challenges in voice conversion, improving both the naturalness and variability of the generated speech.

FreeVC: Voice Conversion Based on the VITS Framework

FreeVC Li et al. (2023) is a non-autoregressive voice conversion model that extends the VITS framework, specifically adapted for voice conversion. The core architecture of FreeVC retains the key components of VITS—namely, the prior encoder, posterior encoder, normalizing flows, and HiFi-GAN decoder—but modifies them to suit the voice conversion task. Unlike VITS, the Prior Encoder in FreeVC directly processes raw waveforms instead of relying on text annotations, and it adopts a different architectural design. The inference process is then conditioned on the target speaker embedding and content representation rather than textual information.

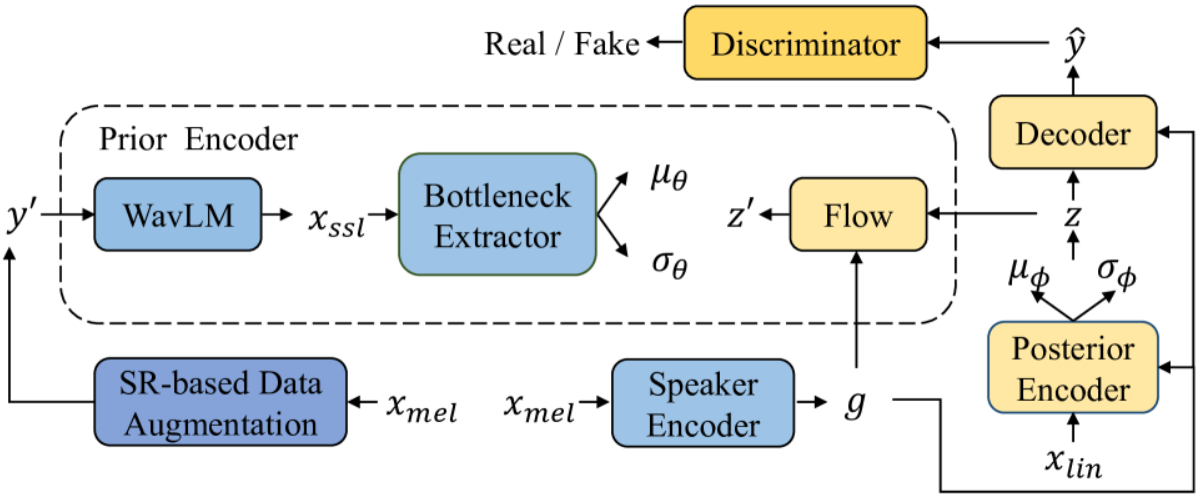

Figure 2.12.: FreeVC architecture and training process Li et al. (2023)

Key Components of FreeVC

- **Prior Encoder**: The prior encoder in FreeVC is composed of three main components: a content encoder, a bottleneck extractor, and a normalizing flow. The content encoder is responsible for extracting content representations from the input speech while discarding speaker identity information. Typically, a pretrained SSL model such as WavLM or wav2vec is used for this purpose. Specifically, the WavLM model takes raw waveforms as input and outputs 1024-dimensional SSL features, denoted as x_{ssl}, which contain both content and speaker information.

  To remove the unwanted speaker-related information embedded in x_{ssl}, the features are passed through a bottleneck extractor that compresses them into a much smaller d-dimensional representation, where $d \ll 1024$. This significant dimensionality reduction imposes an information bottleneck, forcing the resulting low-dimensional features to discard content-irrelevant information such as noise and speaker traits.

  Subsequently, similar to VITS, the d-dimensional hidden representation is projected into a $2d$-dimensional space and then split into two d-dimensional vectors, representing the parameters of a Gaussian distribution $\mathcal{N}(z;\mu,\sigma^2)$. To enhance the expressiveness of the prior distribution, a normalizing flow conditioned on the speaker embedding g is applied. Following the design in VITS, the normalizing flow is built from multiple affine coupling layers and is constructed to be volume-preserving, with a Jacobian determinant $\left|\frac{\partial z}{\partial z_0}\right| = 1$.

- **Speaker Encoder**: Encodes the target speaker identity into a fixed-dimensional vector. During inference, this vector guides the generation of the target speaker's voice.

- **Posterior Encoder**: Encodes the source speech waveform into a latent space distribution to match the prior during training.

- **Decoder**: A HiFi-GAN based generator synthesizes the waveform from the latent variable z.

Training Objective

The training loss of FreeVC is very similar to that of VITS, combining several components:

$$\mathscr{L}_{\text{FreeVC}} = \mathscr{L}_{\text{recon}} + \lambda_{\text{KL}}\mathscr{L}_{\text{KL}} + \lambda_{\text{GAN}}\mathscr{L}_{\text{GAN}}, \tag{2.68}$$

where:

- $\mathscr{L}_{\text{recon}}$ is the reconstruction loss between the real and generated waveforms.
- $\mathscr{L}_{\text{KL}}$ is the KL divergence between the posterior and prior distributions
- $\mathscr{L}_{\text{GAN}}$ is the adversarial loss from the discriminator.

Inference Process

During inference, FreeVC performs the following steps:

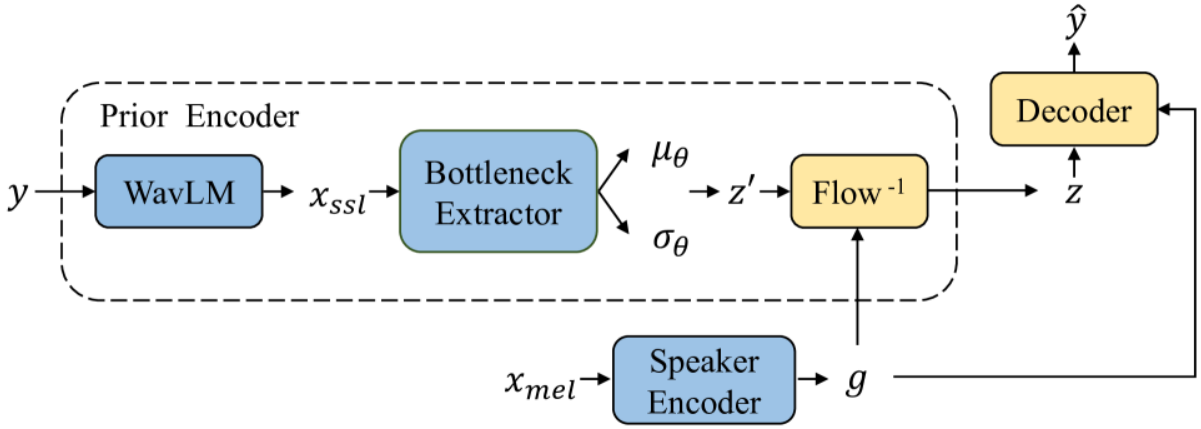

Figure 2.13.: FreeVC inference process Li et al. (2023)

1. Encode the source speech prior parameters $(\mu_\theta, \sigma_\theta)$ using the content encoder and bottleneck extractor.
2. Sample latent variables $z^{'}$ from $\mathscr{N}(\mu_\theta, \sigma_\theta)$
3. Transform $z^{'}$ via normalizing flow: $z = f(z^{'})$.
4. Generate the final converted waveform through the decode: $y = G_\psi(z)$

Because FreeVC avoids autoregressive sampling and attention-based alignment, it enables fast, high-quality, and robust voice conversion even in zero-shot scenarios.

2.5.2. Direct Mapping Approach

The direct mapping approach, as seen in models like *FragmentVC* Lin et al. (2021), *CycleGAN-VC* Kaneko et al. (2019) , directly maps the source speech to the target speech without explicitly disentangling the speaker and content components. This is often achieved using attention-based models trained on paired data or Generative adversarial network trained on non-parallel data. In this thesis, we employ FragmentVC, to augment additional data for training the accent conversion system in Chapter 3

FragmentVC: Attention-based Voice Conversion

FragmentVC is a attention-based VC model designed to convert speaker identity while preserving phonetic content. The architecture consists of three main components: a source encoder, a target encoder, and a decoder. The decoder receives three feature sequences:

- **Q (Query)**: The output feature sequence from the source encoder.
- **K (Key)**: The first output feature sequence from the target encoder.
- **V (Value)**: The second output feature sequence from the target encoder.

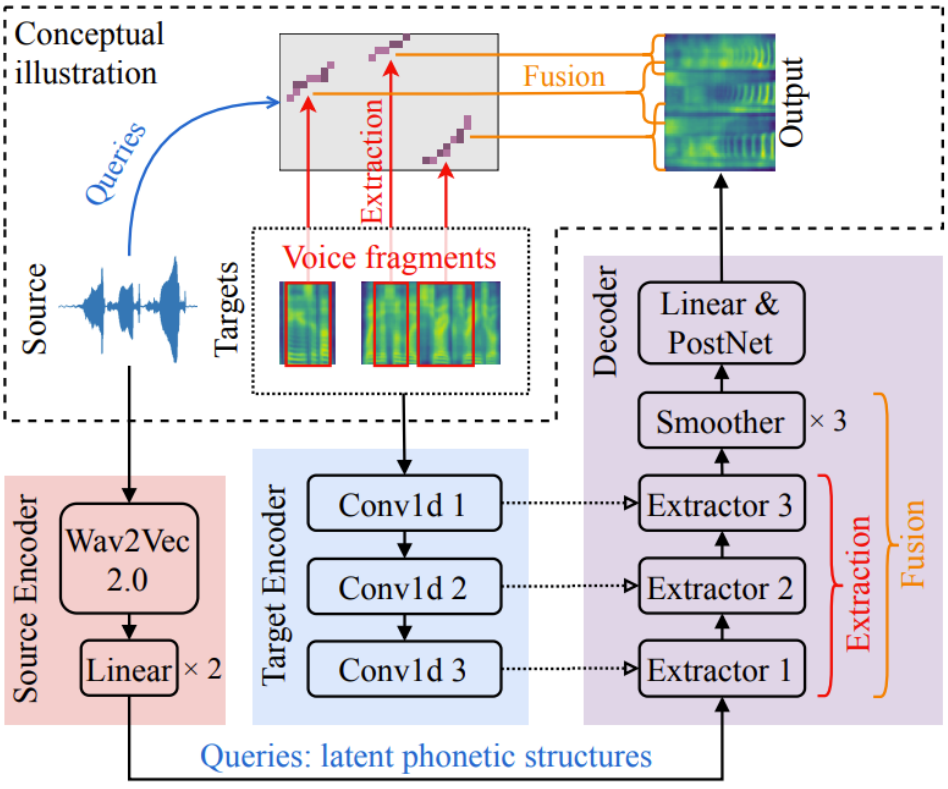

Figure 2.14.: FragmentVC architecture Lin et al. (2021)

The decoder utilizes a cross-attention mechanism, where the source encoder's output (Q) attends to the key sequence (K) from the target encoder. The attention weight $\alpha_{i,j}$ between query q_i and key k_j is computed as:

$$\alpha_{i,j} = \frac{\exp(q_i^\top k_j)}{\sum_{j'} \exp(q_i^\top k_{j'})} \tag{2.69}$$

where q_i and k_j are feature vectors at time step i and j, respectively.
The attended feature vector $\tilde{q}_i$ is then obtained by weighting the value sequence v_j:

$$\tilde{q}_i = \sum_j \alpha_{i,j} v_j \tag{2.70}$$

The final converted Mel-spectrogram $\hat{S}$ is generated by the decoder:

$$\hat{S} = \mathscr{D}(\tilde{Q}) \tag{2.71}$$

where $\mathscr{D}$ is the decoder function, and $\tilde{Q}$ is the attended feature sequence.
During training, the same utterance is used for the source encoder, target encoder, and decoder reconstruction target, allowing the encoders to automatically learn to disentangle phonetic content from speaker identity without requiring explicit constraints. The reconstruction loss is given by:

$$\mathscr{L}_{rec} = ||S - \hat{S}||_1 \tag{2.72}$$

where S is the ground-truth Mel-spectrogram, and $\hat{S}$ is the reconstructed one.

2.6. Accent Conversion

2.6.1. Comparison Between Accent Conversion and Voice Conversion

Although accent conversion and voice conversion are both subfields of speech-to-speech transformation, they differ significantly in their objectives, challenges, and modeling strategies.

Objective Differences. Voice conversion aims to change the speaker identity of an utterance while preserving its linguistic content. The main goal is to generate speech that sounds like it was spoken by a different person. In contrast, accent conversion modifies the accent or pronunciation style of the input speech to match a desired target accent, while maintaining both the speaker identity and the linguistic content. Thus, AC has a finer-grained objective, focusing on phonetic and prosodic nuances without altering speaker characteristics.

Data Requirements. Voice Conversion systems can be effectively trained with either parallel or non-parallel data using speaker embeddings and disentanglement techniques. Accent conversion, however,

often struggles with limited high-quality parallel accent data, especially from non-native speakers, making the collection and alignment of training data more difficult. Furthermore, accent conversion requires careful phonetic alignment and possibly intermediate linguistic representations to ensure accurate pronunciation improvement.

Challenge of Subtle and Time-Varying Variation. A key challenge that distinguishes accent conversion from voice conversion lies in the subtle and temporally dynamic nature of accents. While speaker identity in VC is typically modeled as a global representation—consistent across an entire utterance or even multiple utterances, accent is inherently a time-varying attribute. It manifests locally at the phoneme or frame level through minor phonetic deviations, pronunciation patterns, and intonation shifts. These variations are often more subtle and less prominent than speaker-specific features, making them harder to detect, model, and modify accurately.

This temporal and subtle nature of accent requires accent conversion models to have fine-grained control over speech content representation. They must dynamically adjust pronunciation-related features while preserving the speaker's global identity and linguistic content. Any excessive or imprecise modification risks distorting intelligibility or altering speaker characteristics. In contrast, VC models can rely on more straightforward global conditioning techniques to shift speaker identity, making them comparatively easier to implement. As a result, the need for localized transformations and high sensitivity to error renders accent conversion a significantly more challenging task than voice conversion.

Evaluation Complexity. Objective evaluation of voice conversion models is relatively straightforward using speaker similarity and content preservation metrics. Accent conversion evaluation, however, requires accent similarity assessments, which are often subjective and demand expert judgment or accent classification models.

2.6.2. Previous Work on Accent Conversion

One of the primary difficulties in AC is the lack of ground-truth parallel data, where the same speaker utters the same sentence in different accents. In contrast, VC can often be trained on paired data of different speakers saying identical content. Due to this limitation, previous researchers have explored various strategies to achieve accent conversion effectively.

Reference-Based Accent Conversionn

Phonetic posteriorgrams (PPGs) Churchwell et al. (2024) have been extensively used in both voice conversion and accent conversion tasks. PPGs are frame-level phoneme probability distributions extracted from speech using automatic speech recognition models. They offer a speaker-independent representation by focusing purely on the phonetic content of speech while suppressing speaker-specific character-

istics. This property makes PPGs particularly valuable in AC systems, where the goal is to modify the pronunciation style. By serving as an intermediate linguistic representation, PPGs enable the system to disentangle accent and speaker information more effectively.

Despite their utility, previous PPG-based methods present notable limitations. A common approach Zhao et al. (2019) in accent conversion involves utilizing native-accented reference utterances during the conversion process. The system extracts PPGs from a native speaker's recording and synthesizes speech that retains the target speaker's voice characteristics but reflects the native pronunciation patterns. However, this reliance on reference utterances at inference time reduces practicality in real-world scenarios, where native-accented recordings may not be readily available. As a result, this reference-based system often face challenges in achieving fully autonomous accent conversion without external pronunciation references.

Reference-Free Accent Conversion

To address the impractical reliance on reference utterances, recent research has focused on developing reference-free AC systems. Unlike reference-based methods, reference-free approaches aim to perform accent conversion without requiring a direct native-accented speech sample during inference.

Only a few prior works have explored reference-free accent conversion. One of the first notable approaches is by Liu et al. (2020a), which employed a pipeline combining text-to-speech and automatic speech recognition components to generate the output speech. While this method proved effective in converting accents, it relied on complex multi-stage processing. Specifically, the system first transcribed the input speech using ASR and then synthesized new speech from the transcription using TTS. This architecture introduces a significant challenge of error accumulation—errors in the ASR transcription (e.g., misrecognized words or punctuation) are propagated into the TTS output, degrading the final speech quality and intelligibility. Furthermore, the dependency on high-quality ASR and TTS systems makes the approach computationally expensive, slow in inference.

A second study Quamer et al. (2022) introduced a more streamlined approach by eliminating the need for explicit TTS and ASR components. Instead, the authors used a speech synthesizer with speech embeddings as input to generate synthetic training data. Their AC model, based on an LSTM-driven sequence-to-sequence (Seq2Seq) architecture, was trained from scratch using the generated synthetic data. However, a key limitation of this approach was the need to train a speech synthesizer for each speaker in the training set, making it less flexible for real-world deployment.

Towards a More Generalized Approach

Building upon previous work, this thesis aims to develop a more generalizable reference-free AC system. Instead, a single, unified model will be leveraged for accent conversion across many speakers and

many accents, improving scalability and practical usability. Furthermore, recent advancements in self-supervised learning, speech generative model, and disentangled speech representations offer promising directions for improving AC performance without relying on training parallel data or reference utterances during inference.

2.6.3. Evaluation of accent conversion

Evaluating the performance of an accent conversion system is crucial to understanding its effectiveness in modifying speech and pronunciation while preserving speaker identity and maintaining speech quality. The evaluation process typically involves both subjective and objective metrics, which complement each other to provide a comprehensive evaluation of the performance of the Ac model.

Subjective Metrics of Accent Conversion

Subjective evaluation is crucial as accent is inherently a perceptual trait. Three commonly used subjective evaluation methods include **Mean Opinion Score (MOS)**, **Nativeness (or Accentedness)**, and **Speaker Similarity Mean Opinion Score (SimMOS)**.

Mean Opinion Score (MOS): MOS assesses the overall quality and naturalness of the converted speech. Human listeners rate speech samples on a 5-point scale:

- **1:** Bad (Highly unnatural, robotic, or noisy)
- **2:** Poor (Partially understandable but with significant distortions)
- **3:** Fair (Moderate quality with noticeable artifacts)
- **4:** Good (Mostly natural but with minor imperfections)
- **5:** Excellent (Highly natural and free of artifacts)

The MOS is computed by averaging scores from multiple participants, giving an overall indication of how well the accent-converted speech aligns with natural human speech.

Nativeness (or Accentedness) : This test evaluates how native-like the converted speech sounds, using the same 5-point scale:

- **1:** Strongly accented (Very difficult to understand)
- **2:** Moderately accented (Noticeable accent but understandable)
- **3:** Mildly accented (Slight accent but mostly native-like)
- **4:** Nearly native (Minimal accent)

- **5:** Fully native (Indistinguishable from native speech)

A well-performing AC model should produce speech that receives high nativeness scores, indicating successful accent modification.

Speaker Similarity (SimMOS): This test evaluates how well the converted speech retains the original speaker's identity. Listeners are presented with pairs of speech samples (one original and one converted) and asked to rate their similarity on a 5-point scale::

- **1:** Completely different speaker
- **2:** Mostly different with some similarities
- **3:** Somewhat similar but noticeable differences
- **4:** Mostly similar with minor variations
- **5:** Nearly identical

A high speaker similarity score indicates that the AC model preserves the speaker's unique vocal characteristics while modifying the accent.
To ensure reliable evaluation, at least 10 human participants are typically recruited to assess these subjective metrics. Their responses provide valuable insights into the perceptual quality of the accent-converted speech.

Objective Metrics of Accent Conversion

While subjective evaluations provide human judgments on quality and similarity, objective metrics offer a quantitative and reproducible way to measure accent conversion performance. Three widely used metrics are **Word Error Rate (WER)**, **Accent Classifier Accuracy (ACC)**, and **Speaker Embedding Cosine Similarity (SECS)**.

Word Error Rate (WER): WER measures how well an automatic speech recognition system transcribes speech. It is computed as:

$$\text{WER} = \frac{S + D + I}{N} \tag{2.73}$$

where:

- S is the number of substituted words,
- D is the number of deleted words,
- I is the number of inserted words,

- N is the total number of words in the reference transcription.

A lower WER for converted speech indicates improved pronunciation and intelligibility.

Accent Classifier Accuracy (ACC): Accent classification accuracy measures how well the AC model transforms the accent by using a pre-trained accent classifier. This classifier is trained on a large dataset of speech samples labeled by accent type. The classification process follows these steps:

- The original non-native and converted speech are fed into the classifier.
- The classifier predicts the accent label for each sample.
- The accuracy (ACC) is computed by comparing predicted labels to expected labels.

A successful AC model should produce a converted speech that is classified with high confidence as native-accented speech, thereby reducing the accent classifier's ability to detect non-native speech.

Speaker Embedding Cosine Similarity (SECS):
SECS is an objective measure used to quantify speaker similarity before and after accent conversion. This metric is computed as follows steps:

- A state-of-the-art speaker embedding model (e.g., ECAPA-TDNN, x-vector, or WavLM-based embeddings) extracts speaker embedding from both original and converted speech.
- The cosine similarity between these speaker embeddings is computed using:

$$\text{SECS} = \frac{\mathbf{e}_1 \cdot \mathbf{e}_2}{\|\mathbf{e}_1\|\|\mathbf{e}_2\|} \tag{2.74}$$

  where $\mathbf{e}_1$ and $\mathbf{e}_2$ are the speaker embeddings

- The similarity score ranges from -1 to 1, where:
  - 1.0 indicates identical speaker identity
  - 0.0 suggests no correlation
  - -1.0 represents completely different speakers

A SECS score greater than > 0.85 indicates that the AC model successfully preserves speaker identity while modifying pronunciation.

Summary

Subjective and objective metrics together provide a comprehensive evaluation:

- High **MOS** and **Nativeness** scores indicate high-quality and native-like speech.

- Low **WER** suggests better intelligibility.
- High **ACC** ensures effective accent transformation.
- High **SECS** and **SimMOS** confirms speaker identity retention.

These evaluations ensure that an AC model is both effective and practical for real-world applications.

2.6.4. Experimental setup

Dataset

In this thesis, we utilize a range of well-established speech datasets to support different stages of training, fine-tuning, and evaluation of accent conversion systems.

The **LibriSpeech** corpus Panayotov et al. (2015) is a large-scale dataset derived from public domain audiobooks read by native English speakers. It contains approximately 1000 hours of speech sampled at 16 kHz and is primarily used for training automatic speech recognition. Due to its clean, well-aligned transcriptions and high-quality recordings, it serves as a reliable source of native speech, providing consistent pronunciation patterns.

Building upon LibriSpeech, the **LibriTTS** corpus Zen et al. (2019) was developed to better support text-to-speech and voice conversion research. LibriTTS offers approximately 585 hours of English speech recordings, sampled at 24 kHz for higher audio fidelity compared to LibriSpeech. The dataset consists of aligned audio-text pairs, multi-speaker recordings, and both clean and noisy speech segments, making it versatile for a variety of speech synthesis tasks. Importantly, LibriTTS preserves the speaker diversity of LibriSpeech while improving the prosodic naturalness and intelligibility of the recordings, addressing limitations observed in audiobook-style speech. It provides phoneme-level alignments, normalized and carefully processed texts, and multiple utterances per speaker, which are essential features for training robust accent conversion, TTS, and voice conversion models.

The **VCTK** dataset Yamagishi et al. (2019) consists of speech data from 109 native English speakers with various regional accents. Each speaker reads about 400 sentences, allowing us to capture speaker-dependent prosody and accent variation.

The **LJSpeech** corpus Ito (2017), containing over 13,000 utterances from a single American female speaker. Its consistent speaking style and recording conditions make it particularly suitable for modeling clean, native-like target speech.

To incorporate non-native speech dataset, we employ the **L2-ARCTIC** dataset Zhao et al. (2018), which features recordings from 24 non-native English speakers, each with different first-language backgrounds (e.g., Hindi, Korean, Mandarin, Spanish). Each speaker reads the same set of phonetically balanced

utterances as in CMU-ARCTIC, making it a valuable resource for supervised accent conversion training. This dataset is central to our modeling and evaluation of pronunciation errors and non-native accent patterns. We also include the **CMU-ARCTIC** corpus Kominek and Black (2004b), a parallel dataset containing recordings from multiple native English speakers, each reading approximately 1150 phonetically balanced sentences. Its parallel structure with L2-ARCTIC enables the construction of aligned native-nonnative pairs, which are crucial for training and evaluating parallel accent conversion systems and computing objective metrics such as mel-cepstral distortion (MCD) or ABX error rates.

Together, these datasets provide a rich and diverse foundation for training robust models that generalize across accents, speaker identities, and phonetic variability.

Framework and toolkit

To support the implementation and experimentation of various models in this thesis, we leverage several state-of-the-art open-source toolkits.

Coqui-TTS Gölge et al. (2021) is utilized as the main framework for building and training text-to-speech and voice conversion models. It offers a modular and extensible architecture with support for various acoustic models (e.g., Tacotron2, FastSpeech2) and vocoders (e.g., HiFi-GAN, WaveGlow), which facilitates rapid prototyping and integration of accent conversion components.

Hugging Face Transformers Wolf et al. (2019) is employed to access pre-trained models and fine-tuning utilities, particularly for language and embedding tasks. Its extensive model zoo and ease of integration allow for quick experimentation with multilingual and self-supervised learning models relevant to peech processing.

NMTGMinor Ngoc-Quan (2019), a neural machine translation and speech recognition toolkit, is used for implementing and training sequence-to-sequence models for accent conversion. It provides a powerful framework suitable for research-oriented customization.

Together, these toolkits form the software foundation of our research, allowing us to build, modify, and evaluate many models efficiently in this thesis.

3. Data Augmentation by using Voice Conversion

There is currently no parallel corpus available that consists of audio pairs with identical content spoken by the same speakers in different accents. In this chapter, to address this limitation, we propose a method for generating parallel training data. The approach involves two main steps: first, we develop a voice conversion model to synthesize a dataset containing audio pairs that maintain the same speaker identity while exhibiting different accents. Then, we train an accent conversion (AC) model using the synthesized data to transform speech from the source accented speech to the target accented one. Additionally, building on the proven effectiveness of self-supervised speech representation learning, particularly with wav2vec 2.0, in tasks such as voice conversion, speech recognition, speech translation, we incorporate this pre-trained model with specific modifications to train the accent conversion model in the second step. With only nine hours of synthesized training data, the encoder, initialized with the pre-trained wav2vec 2.0 weights, achieves superior performance compared to an LSTM-based encoder Hochreiter and Schmidhuber (1997).

3.1. General idea of generating synthetic data

In the first step, we employ a voice conversion (VC) model that maintains the pronunciation pattern and prosody of the source audio while modifying only the speaker's identity, specifically timbre and pitch, to match a target speaker. Since prosody and pronunciation patterns are the key components of an accent, preserving both ensures that the accent remains unchanged. This approach allows us to generate paired utterances that share the same voice and linguistic content but differ in accent, providing valuable training data for accent conversion.

The goal of generating synthetic data is to create paired audio samples that facilitate training the sequence-to-sequence (seq2seq) accent conversion model. Consider two recordings of the same content spoken by two different speakers: Andy, who has an American accent, and Rishu, who has an Indian accent. By applying voice conversion to Andy's original speech, we aim to produce an audio sample that retains the American accent while transforming the speaker identity to match Rishu's voice. This is consistent to a VC system which only changes a speaker's identity while preserves an original accent.

As a result, we obtain a pair of utterances in Rishu's voice but with two distinct accents: the original Indian-accented speech and the converted American-accented speech. Similarly, if we perform the voice conversion other way round on Rishu's audio to synthesize Andy's voice, the outcome is to have an

audio pair in Andy's voice but two different accents- (original) American and (converted) Indian. The goal of generating synthetic data is to create paired audio samples that facilitate training the sequence-to-sequence (seq2seq) accent conversion model.The overall voice conversion process for synthetic data generation is illustrated in Figure 3.1.

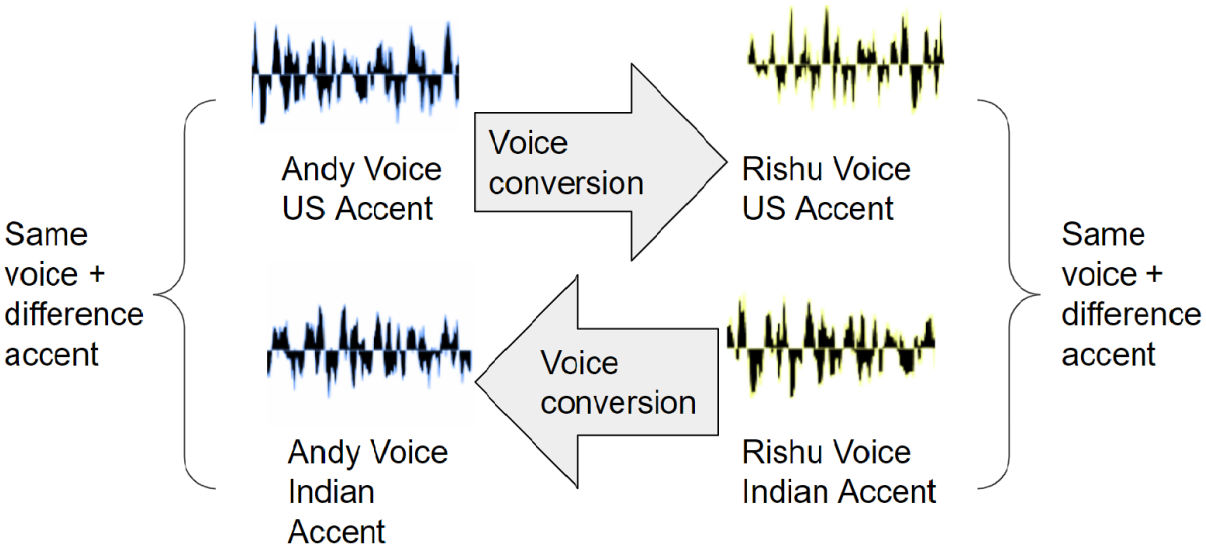

Figure 3.1.: Synthetic data process

3.2. Selecting a Voice Conversion Model

To generate synthetic data for accent conversion, we require a voice conversion model that can modify speaker identity while preserving unique accent characteristics, including pronunciation patterns and prosody. The primary challenge is ensuring that the transformed speech retains the original accent, as prosody and pronunciation are key components of an accent. To this end, we explore two advanced VC architectures: FragmentVC, which is based on a direct mapping framework Lin et al. (2021), and VQMIVC, which employs a disentangle variational autoencoder-based approach Wang et al. (2021a). After thoroughly fine-tuning these models with our dataset using the same hyperparameter settings as in their respective original papers, we employ the most effective VC model to generate synthetic parallel data.

Both FragmentVC and VQMIVC (as describe in chapter 2.5) are promising candidates for generating synthetic data for accent conversion, primarily because they effectively preserve the accent characteristics of the input speech during conversion. For FragmentVC, the attention mechanism is designed such that the phonetic structure of the speech remains largely unchanged throughout the voice conversion process. As a result, key elements that define an accent—such as pronunciation patterns and prosody—are preserved while the speaker identity is adapted. VQMIVC adopts a disentanglement strategy that explicitly separates phonetic and prosodic information from speaker identity. During conversion, only the

speaker embedding is altered, while the phonetic and prosodic features are kept intact. This ensures that the resulting speech maintains the original accent characteristics of the source speaker. These properties make both FragmentVC and VQMIVC highly suitable for accent conversion data augmentation, as they allow the creation of parallel data without compromising accent-related features. Selecting the most effective VC model for synthetic data generation is crucial for training a high-quality accent conversion system.

3.3. Model architecture

Baseline

The baseline model follows an encoder-decoder framework with an attention mechanism, as illustrated in Figure 3.2. The encoder comprises two pyramid bi-directional LSTM (bi-LSTM) layers, similar to the structure proposed in Liu et al. (2020a). Each bi-LSTM layer consists of 768 units, with a down-sampling factor of 2 to reduce sequence length while preserving essential temporal information. The input to the encoder is an 80-dimensional Mel-filterbank feature representation. Additionally, a phoneme classifier layer is integrated on top of the encoder, and a connectionist temporal classification (CTC) loss is applied to enhance phonetic representation learning.

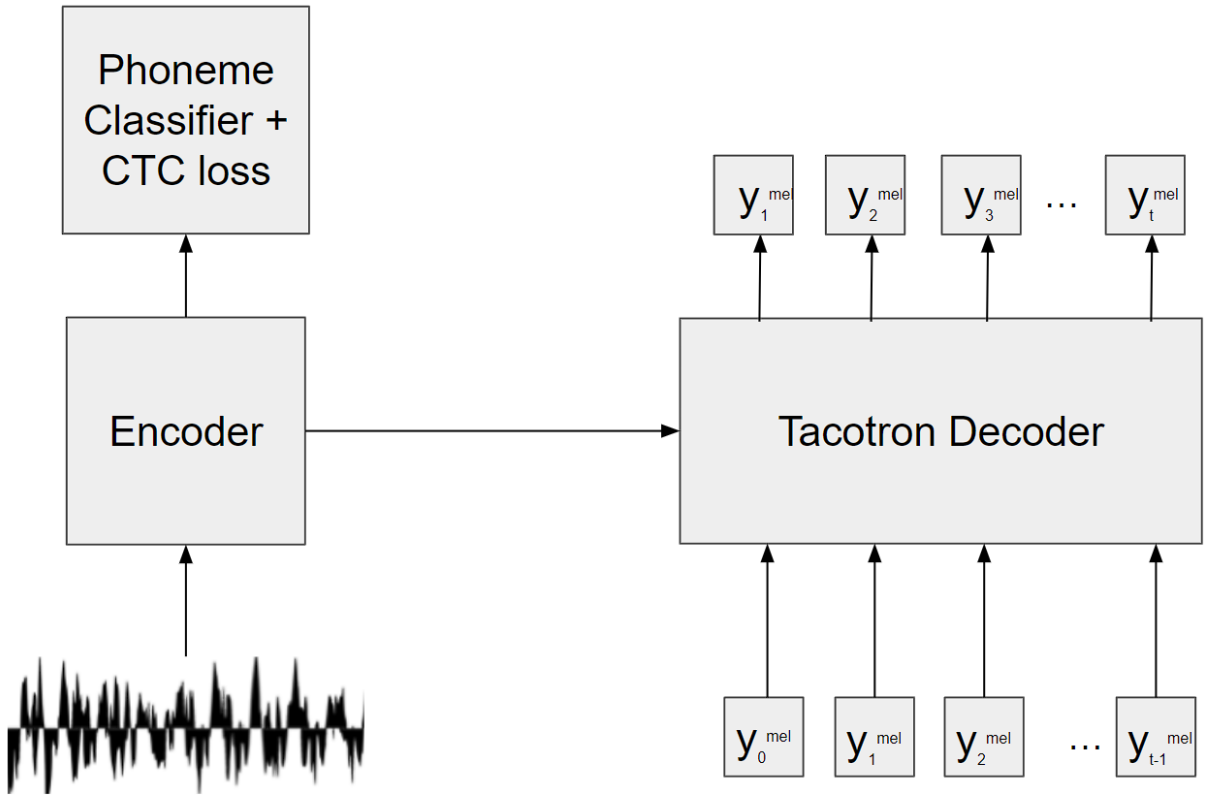

Figure 3.2.: Accent conversion architecture

The baseline decoder follows the Tacotron speech synthesizer decoder Shen et al. (2018), incorporating an attention mechanism consisting of a query layer, key-value layer, and alignment layer. The attention module aligns query, key, and value vectors in the same space, using previous attention weights α_{i-1} to enhance alignment (Eq 3.3). This makes the mechanism both content-based and location-sensitive.

The decoding process starts with the query computation (Eq 3.2), followed by attention weight estimation (Eq 3.3). The weighted sum of key-value vectors forms the context vector (Eq 3.4), which is combined with the decoder RNN hidden state (Eq 3.5). The final mel-spectrogram prediction y_i^{mel} is computed (Eq 3.6) and refined using a convolutional PostNet module (Eq 3.7). We adopt the hyperparameters from NVIDIA's Tacotron2 implementation[1].

$$h^x_{1\ldots n} = Encoder(x_1 \ldots x_N) \tag{3.1}$$

$$s_i = RNNAtt(s_{i-1}, [c_i - 1; Prenet(y^{mel}_{i-1})]) \tag{3.2}$$

$$\alpha_i = Att(Query(s_i), Mem(h^x_{1\ldots n}), Location(\alpha_{i-1})) \tag{3.3}$$

$$c_i = \sum_j \alpha_{i,j} * h_j \tag{3.4}$$

$$d_i = RNNDecoder(d_{i-1}, [s_i; c_i]) \tag{3.5}$$

$$y^{mel}_i = Linear([d_i; c_i]) \tag{3.6}$$

$$y^{Postnet}_i = Postnet(y^{mel}_i) \tag{3.7}$$

Proposed Approach

Our proposed model integrates a pre-trained wav2vec 2.0 Baevski et al. (2020c) encoder with a Tacotron-based speech synthesis decoder, similar to the baseline model. Instead of training the encoder from scratch, we leverage self-supervised learning to obtain robust speech representations from raw audio.

Advantages of Wav2vec 2.0 Over Training from Scratch: As described in Section 2.4.2, compared to training an encoder from scratch, incorporating a pre-trained wav2vec 2.0 encoder offers several advantages:

- **Better Data Efficiency:** Self-supervised learning allows the model to leverage large-scale unlabeled datasets, reducing the reliance on annotated speech data.
- **Improved Feature Representation:** The pre-trained encoder captures both phonetic and prosodic information, leading to better generalization on accent conversion tasks.

[1] https://github.com/NVIDIA/tacotron2

- **Effective Downstream Adaptation:** Pre-trained Wav2vec representations can be fine-tuned on specific downstream tasks, such as accent conversion, using relatively small amounts of labeled data. This makes it especially valuable in low-resource settings

- **Improved Generalization:** Pre-trained models demonstrate strong generalization across different domains and accents, making them highly suitable for cross-accent tasks.

- **Faster Convergence:** Since the wav2vec encoder is already trained to extract meaningful speech features, the fine-tuning process requires fewer epochs compared to training an encoder from scratch.

- **Robustness to Noise and Variability:** Pre-trained representations are more invariant to speaker and environmental noise, improving the robustness of the accent conversion model.

Integration with Accent Conversion Model

We use the pre-trained Wav2vec 2.0 Base model trained on LibriSpeech, which extracts 768-dimensional feature representations. Unlike the baseline encoder, which operates on mel-spectrograms, the wav2vec encoder directly processes raw waveforms. The implementation from Fairseq[2] is used. On top of the wav2vec encoder, we add a phoneme classifier and compute a connectionist temporal classification (CTC) loss:

$$\mathscr{L}_{ctc} = -\sum_{i} \log P(Y_i|X) \tag{3.8}$$

where $P(Y_i|X)$ represents the probability of the target phoneme sequence given the input audio.
During training, we freeze the feature encoder while fine-tuning both the context encoder and the Tacotron decoder.

Loss Function

Both the baseline and proposed systems use the following loss function:

$$\mathscr{L} = ||Y_{mel} - Y_{mel}^{Decoder}||_2 + ||Y_{mel} - Y_{Postnet}^{Decoder}||_2 + \mathscr{L}_{CTC} \tag{3.9}$$

where Y_{mel} is the ground-truth mel-spectrogram, $Y_{mel}^{Decoder}$ is the Tacotron decoder's output, and $Y_{Postnet}^{Decoder}$ is the enhanced mel-spectrogram after PostNet processing.

We exclude the stop token loss because, during inference, we assume that the output speech has the same duration as the input speech.

[2]https://github.com/pytorch/fairseq/blob/main/fairseq/models/wav2vec

For waveform synthesis, we use the Hifi-GAN neural vocoder Prenger et al. (2019), utilizing its open-source PyTorch implementation. Since mel-spectrograms contain all relevant details for high-quality speech synthesis, we train the Hifi-GAN vocoder using ground-truth 80-dimensional mel-filterbank spectrograms.

3.4. Experiments

To train the voice conversion model, we utilized audio data from the CMU-ARCTIC corpus Kominek and Black (2004a) and the L2-ARCTIC Hindi-accented corpus Zhao et al. (2018). These datasets collectively consist of speech recordings from eight speakers—four native English speakers and four speakers with an Indian accent. Each speaker recorded the same set of 1,152 sentences. Instead of training a VC model from scratch, we leveraged pre-trained VC models from Wang et al. (2021a) and Lin et al. (2021) and fine-tuned them using our datasets to enhance the performance across all speakers.

To generate parallel data for training our accent conversion model, we first divided the 1,152 sentences into three subsets: a training set (1,052 sentences), a validation set (50 sentences), and a test set (50 sentences). Each sentence was spoken by all eight speakers, including both native and accented speakers. Following the data synthesis method outlined in the previous section, we generated eight audio pairs per sentence, resulting in approximately 9,000 audio pairs, totaling around nine hours of speech data. All recordings were downsampled to 16 kHz.

Both the baseline and proposed AC models were trained on a single GPU with a batch size of 64. Gradient updates were performed every eight mini-batches using the Adam optimizer. The learning rate was set to 0.5 with a warm-up period of 4,000 steps, following the same learning rate schedule as the Transformer model Vaswani et al. (2017a). The training process was capped at a maximum of 20,000 steps.

We conduct the evaluation on all models by using two kinds of metrics: objective and subjective. We use 50 sentences (200 American-accented audio files and 200 Indian-accented audio files) in the test set for evaluation. Sample evaluation audios can be found here [3].

Subjective tests

Accentedness test and speaker similarity test : Two tests are conducted by 10 Indian participants who listen to the provided audios and evaluate their accent features on a 5-point scale: 1-bad, 2-poor, 3-fair, 4-good, 5-excellent. To generate converted audios with an Indian accent, we apply our AC models on 200 American-accent audio files. For the accentedness test, the participants give a score out of 5 for the degree to which the converted audios sound like Indian-accented speeches. For the speaker similarity

[3] https://tuannamnguyenkit.github.io

test, they rate the similarity between the voice identity of the input audio and the converted audio, then give a score out of 5 as above.

Objective tests

Word error rates (WER) and Accent classifier accuracy (ACC) : We compute a WER of both original Indian-accented audios and converted Indian-accented audios by our competitive ASR system Nguyen et al. (2021). If the gap between two WERs is lower, the quality of the converted audio is implied to be better. In addition, we design an accent classifier, which takes audio as an input and predict the accent of that input. The accent classifier is a 4-LSTM-layer network with 512 hidden units with an accent classifier layer on the top and trained on our training data set. We compute ACC on both converted and original audios. A better conversion model is expected to have a smaller gap between two ACCs

3.5. Results

Models	Accentness	Speaker Similarity
Original Indian Accented Audio	4.8	-
Our Proposed System		
+FragmentVC Synthetic Data	2.5	3.8
+VQMIVC Synthetic Data	2.3	**4.1**
Wav2vec-based		
+FragmentVC Synthetic Data	**3.8**	3.1
+VQMIVC Synthetic Data	**3.7**	**3.6**

Table 3.1.: Subjective evaluation results. A lower accentness score indicates better accent conversion, while a higher speaker similarity score indicates better speaker identity preservation.

Models	WER (%)	ACC (%)
Original Indian Accented Audio	9.7	96
Original US Accented Audio	6.3	98
LSTM-based		
+FragmentVC Synthetic Data	42	63
+VQMIVC Synthetic Data	35	47
Wav2vec-based		
+FragmentVC Synthetic Data	24	**88**
+VQMIVC Synthetic Data	**18**	**87**

Table 3.2.: Objective evaluation results. WER denotes word error rate, and ACC represents accent classification accuracy. Lower WER and higher ACC indicate better performance.

Subjective Evaluation

To assess the perceived quality of the converted speech, we conducted a subjective evaluation with native Indian speakers. Table 7.1 presents the results for accentness and speaker similarity.

In terms of accentness, the Wav2vec-based model outperforms the LSTM-based model under all synthetic data conditions, achieving significantly lower scores (3.8 vs. 2.5 and 3.7 vs. 2.3). However, for speaker similarity, the LSTM-based model performs better, indicating that the Wav2vec-based model modifies the speech more extensively. While this results in improved accent conversion, it also makes listeners perceive greater deviations from the original speaker identity.

Comparing the two synthetic data approaches, FragmentVC and VQMIVC exhibit similar performance in accent conversion (2.5 vs. 2.3 and 3.8 vs. 3.7). However, in speaker similarity, the VQMIVC method outperforms FragmentVC, suggesting that it is more effective at preserving speaker identity. This indicates that VQMIVC produces higher-quality voice conversion results that retain speaker characteristics more effectively.

Objective Evaluation

To quantitatively evaluate the effectiveness of our approach, we applied automatic speech recognition (ASR) and accent classification models to both original and synthesized speech.

For the original Indian-accented audio, the ASR model achieved a word error rate (WER) of 9.7%, while the WER for the US-accented audio was lower at 6.3%. The accent classifier also demonstrated high accuracy, identifying Indian- and US-accented speech with 96% and 98% accuracy, respectively.

Table 7.2 shows the WER and accent classification accuracy (ACC) results for different AC models. Within the LSTM-based models, training with VQMIVC synthetic data consistently yields better WER than using FragmentVC synthetic data (35% vs. 42%). A similar trend is observed in the Wav2vec-based models, where VQMIVC achieves the lowest WER of 18%. These results indicate that VQMIVC generates higher-quality training data than FragmentVC.

Regarding accent classification accuracy, the Wav2vec-based model performs similarly with both FragmentVC and VQMIVC synthetic data (88% and 87%, respectively). However, the LSTM-based model exhibits a significant drop in performance when trained on VQMIVC data (63% vs. 47%). However, this does not suggest that the Fragment VC preserves the accent better than the VQMIVC, since the LSTM-based model is not strong enough to generate a good quality output.

Overall Comparison of AC Models

Across all evaluation metrics, the Wav2vec-based AC model demonstrates superior performance compared to the LSTM-based model. It consistently achieves lower accentness scores, lower WER, and

higher ACC, confirming its effectiveness in accent conversion. This highlights the advantage of leveraging self-supervised speech representations for improved accent conversion.

3.6. Conclusions

In this chapter, we have demonstrated that a well-trained voice conversion model can serve as an effective solution for generating high-quality training data for accent conversion . This approach mitigates the challenge of obtaining ground-truth parallel data, which is often scarce and costly to collect. Additionally, we have shown that leveraging a Wav2vec pre-trained encoder can significantly enhance AC performance, particularly in low-resource conditions, by providing robust speech representations learned through self-supervised training.

While our focus has been on a uni-directional (one-to-one) AC model, future work will explore alternative approaches to improving AC performance beyond the current framework. Instead of directly extending this model, we aim to investigate other architectures, training strategies, and pre-trained models that may further enhance accent conversion. Advances in self-supervised learning and representation learning, such as HuBERT and WavLM, present promising directions for improving the adaptability and generalization of AC models. These explorations will provide deeper insights into the effectiveness of different methodologies in tackling accent conversion challenges in the next chapters.

4. Synthesizing Multi-Accented Text-to-Speech

In the previous chapter, we demonstrated that voice conversion can effectively generate parallel training data. However, since this data is derived from a limited original dataset containing just over 1,000 sentences, this approach alone does not expand the dataset with additional sentences or diverse content (transcripts). As a result, the generated speech remains constrained by the linguistic coverage and phonetic diversity of the original dataset, which may not be sufficient for training a robust accent conversion system.

To overcome this limitation, an alternative approach is to leverage a Multi-accented Text-to-Speech (TTS) system. Unlike voice conversion, which primarily modifies the timbre of existing recordings, a well-trained TTS model can generate high-quality speech with diverse linguistic content and speaker characteristics. By conditioning the TTS model on different accents, we can synthesize a more extensive dataset that captures a broader range of phonetic variations and sentence structures. This, in turn, facilitates more effective training of accent conversion models, particularly in low-resource scenarios where authentic multi-accented speech data is limited.

4.1. Motivation

Conventional multi-speaker text-to-speech systems are capable of synthesizing speech for multiple voices; however, they are not designed to generate speech in different accents. This limitation motivates the development of **SYNTACC** (Synthesizing Speech with Accents), an approach that extends conventional multi-speaker TTS to produce multi-accented speech.

Most TTS systems are trained using a single speaker's voice, but recent advancements have enabled the synthesis of speech for multiple speakers, known as multi-speaker TTS Casanova et al. (2022). While these systems allow voice cloning for different speakers, they do not inherently model accent variations. Addressing this limitation is crucial, as the ability to modify accents is highly desirable in modern TTS applications, especially for users who wish to adapt synthetic voices for effective communication across linguistic and cultural boundaries. Furthermore, multi-accented TTS plays a important role in generating non-native accented speech for training augmented speech recognition models and producing paired speech samples for accent conversion models.

A straightforward approach to achieving accent control in TTS is to train separate models for each accent. However, this requires a vast amount of accent-specific training data, which is often unavailable for

many regional accents. To address this challenge, we propose a **multi-accent TTS system** that enables the synthesis of multiple accents within a single model. Our approach fine-tunes a conventional multi-speaker TTS system to incorporate accent conditioning while minimizing the demand for large-scale accented speech datasets. The model is evaluated on four distinct accents: *Indian, Spanish, Chinese, and Vietnamese*.

Transformer-based TTS models have gained prominence in recent years due to their ability to capture long-range dependencies efficiently Vaswani et al. (2017b). Additionally, adaptive weight mechanisms have demonstrated effectiveness in various multilingual speech processing tasks, including speech recognition, machine translation, and speech-to-speech translation Pham et al. (2022a). In this chapter, our research on **SYNTACC** aims to contribute to this growing body of work by exploring the effectiveness of such adaptive mechanisms in multi-accented TTS. The method works by decomposing each weight matrix into a shared component and an accent-dependent component, with the former being initialized by the pretrained multi-speaker TTS model and the latter being factorized into vectors using rank-1 matrices to reduce the number of training parameters per accent. We implement this approach using **YourTTS** Casanova et al. (2022), an advanced version of the **VITS** model Kim et al. (2021b). Since the primary operation in this approach relies on matrix-vector multiplication, it can be seamlessly integrated into various neural TTS architectures, including **FastSpeech** Ren et al. (2019) and **GlowTTS** Kim et al. (2020).

4.2. Multi-accent adaptive weight component

The idea of adaptive weight is motivated by the belief that there are features shared between accents that must be selectively represented, and networks are expected to switch between different "modes" depending on the input accent being processed. In a multi-accent scheme, the phoneme set, the character set and the word set are the same for every accent. Meanwhile, accents may differ in various aspects such as phonetics (acoustic realisation of the same phoneme), prosody, and pronunciation. From a phonological point of view, accent differences may be reflected in their letter-to-sound mapping Kolluru et al. (2014). These differences can consist of substitution of some phonemes. For example ‘bath’ in British English */b a /* versus in General American English (GA) */b æ /*; or in insertions/deletions of some phones, for example in ‘herb’ */ b/* vs */h b/*. Therefore, if we can change the "mode" of the letter-to-sound component of a TTS model, we can expect the accent to alter accordingly. In this paper, the adaptive weight that adds scales and biases to each weight matrix of the letter-to-sound component is selected for investigation.

Given these variations, we hypothesize that changing the "mode" of the letter-to-sound component in a TTS model will lead to corresponding accent changes. In this work, we explore the adaptive weight components, which adds scaling factors and biases to each weight matrix within the letter-to-sound component.

The idea of adaptive weight factorization is based on the fact that matrix multiplication is a fundamental operation in neural networks Pham et al. (2021b). Many architectures, including Transformer models, utilize linear combinations of lower-level features $X \in \mathbb{R}^D$, expressed as matrix multiplications between the input X and weight matrices. For instance, in the self-attention mechanism, the queries Q, keys K, and values V are generated by:

$$Q = W_Q^\top X, \quad K = W_K^\top X, \quad V = W_V^\top X \tag{4.1}$$

with the self-attention operation defined as:

$$SelfAtt(X) = \text{softmax}(QK^\top)V. \tag{4.2}$$

Given that matrix multiplication is central to these operations, we propose a decomposition of the weight matrix W into a shared component W_S and accent-dependent factors: a multiplicative scaling W_{ML} and a bias W_{BL}. Thus, any linear transformation of the form $Y = W^\top X$ can be rewritten as:

$$Y = (W_S \cdot W_{ML} + W_{BL})^\top X \tag{4.3}$$

$$= (W_S \cdot W_{ML})^\top X + W_{BL}^\top X. \tag{4.4}$$

In this formulation, W_{ML} modulates the magnitude and direction of the shared weights W_S, while W_{BL} introduces a content-based bias depending on the input features X. Each accent is associated with a unique set of W_{ML} and W_{BL}, ensuring that the model maintains a semi-shared structure. This decomposition allows for efficient adaptation to multiple accents while preserving the core structure of the neural network.

One challenge in accent-dependent weight adaptation is that both W_{ML} and W_{BL} must have the same size as W_S, which risks making the accent-dependent components dominate the shared component. To address this, we leverage rank-1 matrices $\bar{W} \in \mathbb{R}^{D_{in} \times D_{out}}$, which can be factorized into two vectors Wen et al. (2020); Yoon et al. (2019). Specifically, we represent $\bar{W}$ as:

$$\bar{W} = rs^\top, \tag{4.5}$$

where $r \in \mathbb{R}^{D_{in}}$ and $s \in \mathbb{R}^{D_{out}}$, effectively reducing the number of parameters from $D_{in} \times D_{out}$ to $D_{in} + D_{out}$. However, a single rank-1 decomposition may lack sufficient representational power. To enhance expressiveness while maintaining parameter efficiency, we extend this approach by introducing k independent weight factors per accent, summing them to increase the rank of the additional weight matrices:

$$\bar{W} = \sum_{i=1}^{k} r_i s_i^{\top}. \tag{4.6}$$

This formulation ensures that accent-dependent variations are captured effectively without significantly increasing the number of parameters.

4.3. Adaptive multi-accent speech synthesis

YourTTS forms the foundational architecture of our SYNTACC system, incorporating an adaptive component for accents Casanova et al. (2022). Built as an improved version of VITS, YourTTS introduces several enhancements that support zero-shot synthesis across multiple speakers and languages. It stands out as one of the few end-to-end, non-autoregressive TTS models capable of generating high-quality speech. Unlike VITS, which relies on phoneme-level input, YourTTS directly consumes raw text. This feature is particularly advantageous for languages or accent variations where grapheme-to-phoneme (G2P) tools are either unavailable or unreliable. In the context of SYNTACC, where accent-specific G2P mappings can vary widely, avoiding dependence on phoneme inputs simplifies the modeling process and boosts generalization. Although the original YourTTS was trained on English, Portuguese, and French, our use case narrows the focus to English, using a training setup that incorporates multiple accents.

Similar to YourTTS, SYNTACC adopts a conditional variational autoencoder framework augmented with a normalizing flow mechanism (see Fig. 4.1). The overall architecture is composed of three principal components: a posterior encoder, a prior decoder, and a waveform generator, which correspond to the distributions $q_\phi(z|x)$, $p_\theta(z|c)$, and $p_\psi(x|z)$, respectively. $q_\theta(z|x)$ and $p_\psi(z|c)$ are the posterior and data distributions, parameterized by neural posterior encoder's parameters ϕ and Hifi-GAN Kong et al. (2020b) waveform generator's paremeter ψ respectively, where x is the speech input and z is the latent variables. In this setup, x denotes the input speech signal, and z represents the latent variables inferred by the model. The Posterior Encoder receives linear spectrograms and speaker embeddings as input and produces the latent representation z. This latent vector, combined with the speaker embedding, is subsequently passed into the HiFi-GAN vocoder to synthesize the corresponding speech waveform.

The prior distribution $p_\theta(z|c)$ is conditioned on the input text c, where the prior is defined through a text encoder coupled with a normalizing flow decoder f. This flow-based decoder is responsible for transforming the latent variable z in a manner that is conditioned on both the speaker embeddings and the encoded textual information. To ensure alignment between the output sequences of the flow decoder and the text encoder, the Monotonic Alignment Search (MAS) algorithm is utilized. Additionally, a stochastic duration predictor, conditioned on speaker embeddings, estimates the duration d derived from the MAS alignment. During training, the model seeks to maximize the likelihood of the speech output x conditioned on the input text c, expressed as $p(x|c)$, by optimizing the evidence lower bound (ELBO) objective.

$$\log p(x|c) > \mathbb{E}_{q_\phi(z|x)}[\log p_\psi(x|z)] - D_{KL}(q_\phi(z|x)||p_\theta(z|c)) \tag{4.7}$$

Similar to prior approaches, our text encoder employs a Transformer architecture. To enable multi-accent training, we augment each input character embedding with a 16-dimensional trainable accent embedding. The baseline configuration consists of 10 Transformer blocks with 196 hidden channels. Our proposed model extends this architecture by incorporating an adaptive factorized weight component of rank 2. This modification dynamically adjusts layer-specific operations, including the QKV-projection in the self-attention mechanism.

The primary architectural distinction between the baseline and our proposed model resides in the text encoder, which serves as the letter-to-sound module. The system processes the Transformer-encoded text through a flow-based decoder that performs invertible transformations from simple to complex distributions, following the change-of-variable principle. All other components—including the flow-based decoder, Monotonic Alignment Search, speaker encoder, posterior encoder, duration predictor, and waveform generator—remain unchanged from the original YourTTS architecture Casanova et al. (2022).

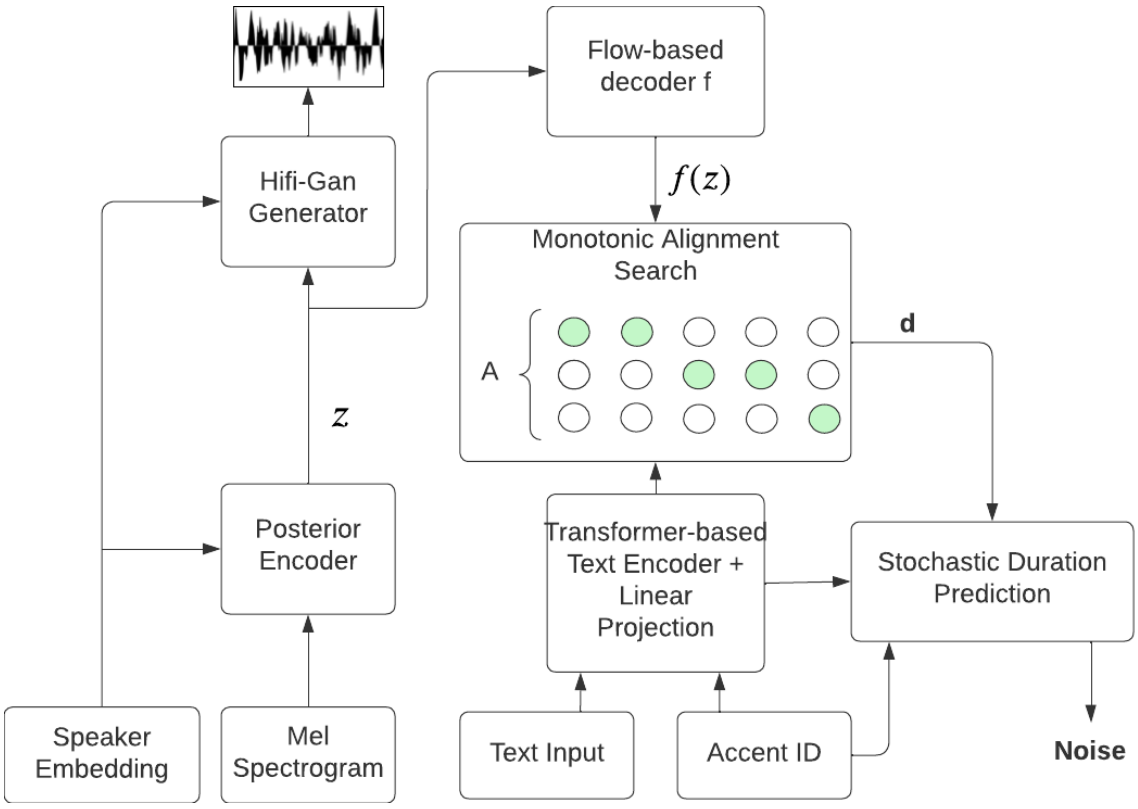

Figure 4.1.: Multi-accent TTS architecture

4.3.1. Experiments

4.3.2. Training and Experiment Setup

The YourTTS model is initially pretrained using the VCTK corpus, which contains 44 hours of speech data recorded at 48kHz from 109 different speakers. Prior to training, all audio samples are resampled to 16kHz to ensure consistency. Our pretraining procedure strictly adheres to the methodology described in the original YourTTS paper Casanova et al. (2022). The model undergoes training for 150,000 steps, after which the resulting pretrained model serves as the foundational checkpoint for all subsequent experimental models in our study.

For our investigation of multi-accent text-to-speech synthesis, we utilize the publicly available L2-Arctic dataset Zhao et al. (2018). This comprehensive dataset includes speech samples with Hindi, Spanish, and Vietnamese accents in English, featuring a total of 12 speakers - comprising 4 native English speakers and 4 speakers with Indian accents. Each speaker's contribution consists of 1,152 carefully selected sentences. To maintain uniformity across our experiments, we implement a standardized preprocessing pipeline for all audio data. This includes amplitude normalization to -27dB using RMS-based normalization (implemented via the ffmpeg-normalize Python package) and resampling all audio files to 16kHz. After preprocessing, we obtain approximately 3 hours of speech data for each of the four accent categories under study: Chinese-accented, Spanish-accented, Indian-accented, and Vietnamese-accented English. This carefully curated dataset represents a low-resource scenario, making it particularly suitable for evaluating our proposed approach.

In this phase of our research, we conduct a systematic comparison between dedicated single-accent models and our unified multi-accent approach. Specifically, we fine-tune four separate instances of the pretrained YourTTS model (from the initial pretraining phase) on each of the accent-specific datasets. This process yields four specialized TTS systems: Indian Accent TTS, Spanish Accent TTS, Chinese Accent TTS, and Vietnamese Accent TTS. These single-accent models serve as important baselines for evaluating the performance of our multi-accent synthesis approach.

Both our baseline implementation and the proposed SYNTACC model are initialized using weights from the pretrained YourTTS model. To preserve the model's capability for multi-speaker synthesis while adapting to accented speech, we strategically combine the accented speech data with samples from Indian speakers in the VCTK dataset during the fine-tuning process. We conduct a comprehensive ablation study to evaluate the impact of pretrained weights on SYNTACC's performance through three distinct training configurations:

(1) Training from scratch without utilizing any pretrained weights,

(2) Fine-tuning with frozen pretrained weights (only updating accent-dependent parameters),

(3) Full fine-tuning where all model parameters are updated.

All models are implemented using the CoquiAI framework Gölge et al. (2021), with training conducted for approximately 10,000 steps on NVIDIA A100 GPUs with 48GB memory, using a substantial batch size of 128 samples to ensure stable and efficient training.

4.3.3. Evaluation Metrics

We assess model performance through comprehensive objective and subjective evaluations. For testing, we synthesize 10 representative sentences in each of the four target accents (totaling 40 test cases). All evaluation samples are publicly available for review[1].

Subjective Tests

We conduct three perceptual tests with 10 native English speakers using 5-point Likert scales (1: poor, 5: excellent):

- **Accentedness**: Measures the authenticity of the synthesized accent compared to native speakers of the target accent
- **Similarity Mean Opinion Score (Sim-MOS)**: Evaluates voice similarity between synthesized and reference speaker samples
- **Mean Opinion Score (MOS)**: Assesses overall speech naturalness and quality

All tests are conducted in controlled listening environments with randomized sample presentation to minimize bias.

Objective Tests

Word Error Rate (WER): WER comparisons between our models and the original YourTTS baseline reveal the impact of accent adaptation on speech clarity, where lower WER values indicate superior performance.

4.3.4. Results and Analysis

Tables 7.1 and 7.2 present comprehensive evaluation results across Chinese (ZH), Indian (IND), Vietnamese (VN), and Spanish (ES) accents, including average scores (AVG) for all accents. Our analysis reveals several key findings:

[1] https://accenttts.github.io/

Subjective Evaluation Results

The SYNTACC model fails to generate intelligible speech when trained without pretrained weights, producing completely incomprehensible output. This failure mode stems from the insufficient training data available for learning accent-dependent components from scratch, establishing this configuration as our worst-performing baseline.

Notably, single-accent YourTTS models achieve superior Accentedness scores (mean = 4.2) but demonstrate significantly poorer performance on Sim-MOS tests (mean = 2.9). We attribute this to their specialized fine-tuning on only four accented speakers, which compromises their multi-speaker generalization capability.

Our proposed SYNTACC model with pretrained weights demonstrates significant improvements, outperforming the baseline multi-accent TTS by 39 % in Accentedness (3.9 vs. 2.8 average) while maintaining comparable Sim-MOS scores. This performance gap suggests that our adaptive weighting mechanism more effectively captures accent-specific phonetic features. All models achieve consistently strong MOS ratings (3.78-3.99), indicating robust preservation of speech naturalness across configurations.

Interestingly, the Indian accent achieves markedly higher Accentedness scores (4.3 vs 3.6 average for other accents), likely due to more consistent pronunciation patterns in Indian English that facilitate model learning. The similar performance between SYNTACC configurations with frozen versus fine-tuned shared weights suggests that accent-specific adaptation occurs primarily in the specialized components.

Objective Evaluation Results

Models	WER				
	IND	ES	VN	ZH	AVG
YourTTS specific-accented	2.4	3.5	3.7	3.9	3.4
Baseline	1.9	3.2	3.3	3.5	3.0
YourTTS	_	_	_	_	0.8
SYNTACC					
+freeze shared weights	2.1	3.2	3.5	3.8	3.15
+fine-tune all	1.8	3.3	3.1	3.2	**2.85**

Table 4.2.: Objective metrics

All experimental models exhibit elevated WER compared to the original YourTTS, with SYNTACC (full fine-tuning) achieving the lowest WER (2.85). The Indian accent again shows superior performance (WER = 2.4) compared to other accents, which we hypothesize results from greater representation of Indian-accented speech in the ASR training data Nguyen et al. (2021).

4.4. Conclusion

To conclude, this chapter has presented our SYNTACC model with a weight factorization method, which shows great promise for integration into any TTS neural architecture. We also demonstrated how pre-trained YourTTS weights can enable effective training of the SYNTACC model under low-resource conditions. While our experiments were conducted with a 4-accent TTS model, the method can easily be extended to additional accents by increasing the size of the accent-dependent components before fine-tuning. In the next chapter, we aim to explore accent conversion using the SYNTACC model to generate additional training data.

5. Accent conversion with parallel data synthesized from SYNCTACC

In the previous chapter, we demonstrated the effectiveness of SYNTACC in multi-accent TTS synthesis by leveraging adaptive weight factorization. Building upon this capability, we now explore the potential of SYNTACC as a tool for generating large-scale accented speech data.

A major limitation in existing accent conversion (AC) models, including the approach in Chapter 3, is the scarcity of parallel training data—paired speech samples. With SYNTACC, we can overcome this challenge by generating large-scale parallel training data, synthesizing identical sentences in multiple accents. This significantly enhances data availability, enabling the training of a more robust and accurate accent conversion model.

In this chapter, we investigate how the combination of SYNTACC-generated parallel data and a larger, more powerful speech-to-speech model can enhance accent conversion quality. We present our model architecture, training methodology, and evaluation metrics to assess the effectiveness of our approach.

5.1. Overview

This chapter introduces an autoregressive, reference-free, zero-shot, many-to-one accent conversion system designed to be trained in low-resource non-native conditions while enhancing fluency in non-native speech. One key aspect of this chapter is leveraging the ability of SYNTACC to generate diverse accented speech with the same text content, enabling the training of parallel AC models, a novel data augmentation strategy that distinguishes our approach from previous studies.

In chapter 3, we train successfully seq2seq AC on parallel data with mel-spectrogram target. However, while training with mel-spectrogram targets yields high-quality results under parallel settings, it faces several limitations. First, mel-spectrograms carry fine-grained acoustic details that are highly speaker-dependent, making it challenging for the model to generalize across many different speakers and accents. Second, the continuous nature of spectrograms increases the complexity of the learning task, requiring the model to precisely predict detailed acoustic features frame by frame.

To address these challenges, we move beyond mel-spectrograms and adopt discrete speech representations as the target for accent conversion. In recent years, the discrete representation Lakhotia et al. (2021)

have emerged as a compelling intermediate representation in the field of speech processing . These units, derived from raw speech signals through various self-supervised learning techniques, offer a compact and informative representation that bridges the gap between continuous acoustic waveforms and higher-level linguistic information. This discrete nature facilitates the application of techniques originally developed for natural language processing (NLP) to speech-related tasks, leading to significant advancements in speech recognition, synthesis, translation, and more Chang et al. (2024).

5.1.1. Overview of Speech discrete units

Speech discrete units can be broadly categorized into two main types, each capturing different aspects of the speech signal: semantic units and acoustic units.

Semantic units Semantic units aim to capture the linguistic content and meaning embedded within the speech. These units are typically derived from models trained using masked language modeling (MLM) or contrastive learning objectives. Prominent examples of such models include:

- **HuBERT (Hidden Unit BERT):** Hsu et al. (2021a) Building upon the success of masked language modeling, this model learns contextualized representations by predicting masked spans of hidden units from a convolutional neural network (CNN) that processes the raw audio waveform. The resulting discrete units capture high-level semantic information.
- **Wav2Vec 2.0:** Baevski et al. (2020a) Wav2Vec 2.0 employs a contrastive objective, where the model distinguishes between the true masked speech segment and distractor segments. The learned representations, and subsequently derived discrete units, exhibit strong semantic properties.

Semantic units often exhibit a closer relationship with the phonemic or even lexical content of the speech. They tend to be more robust to variations in speaker identity, and acoustic conditions, focusing on the underlying message.

Acoustic units Acoustic units, on the other hand, focus on capturing the fine-grained acoustic details of the speech signal. These units are typically obtained from auto-encoder based models, which learn to compress and reconstruct the audio. Key examples include:

- **SoundStream:** Zeghidour et al. (2021) This neural audio codec employs a residual vector quantization (RVQ) technique to compress audio into a sequence of discrete codes. The resulting acoustic units represent the detailed acoustic characteristics of the speech, enabling high-fidelity reconstruction.
- **Encodec:** Défossez et al. (2022) Similar to SoundStream, Encodec is also a neural audio codec that utilizes a multi-scale vector quantization approach to produce discrete representations. These units effectively capture the nuances of the audio signal, including prosody and speaker characteristics.

Acoustic units excel at preserving the acoustic fidelity of the speech and are often used in tasks where the fine-grained details of the audio are crucial, such as high-quality speech synthesis.

Applications of Speech Discrete Units The emergence of robust speech discrete units has spurred advancements across various speech processing applications:

- **Speech Recognition (ASR):** Discrete units provide a more manageable and linguistically meaningful input representation for ASR systems compared to continuous acoustic features. They can be used to train acoustic models more efficiently and potentially improve robustness.
- **Speech Synthesis (TTS):** Acoustic units, with their ability to capture fine-grained acoustic details, are particularly valuable for high-fidelity and natural-sounding speech synthesis. Semantic units can guide the linguistic content of the synthesized speech.
- **Speech Translation:** Discrete units can serve as a pivot representation, allowing for translation between languages without relying on direct acoustic-to-acoustic mapping. Semantic units are particularly useful for capturing the meaning that needs to be translated. Huber et al. (2023); Waibel and Fugen (2008); Niehues et al. (2016); Ha et al. (2016); Waibel and Fuegen (2012); Niehues et al. (2018); Waibel (2019b,a)
- **Spoken Language Understanding (SLU):** The semantic information encoded in discrete units can be leveraged for downstream SLU tasks such as intent recognition and slot filling.
- **Speaker Recognition and Verification:** While semantic units are speaker-invariant by design, acoustic units retain speaker-specific information that can be utilized for speaker recognition and verification tasks.
- **Audio Compression:** The discrete nature of these representations naturally lends itself to efficient audio compression techniques, as demonstrated by models like SoundStream and Encodec.

By transforming continuous speech signals into sequences of semantic discrete units, these units effectively capture linguistic content while minimizing speaker-specific variations. Previous studies Lakhotia et al. (2021) show that HuBERT discrete unit is a good representation for speech content and removes most of the speaker identity, verified by good performance in voice conversion task. But there is no clear clarification on how HuBERT discrete unit reacts on the accent speech. In the section 5.1.2, we are going to show our observations on the effect of the accent on HuBERT unit.

5.1.2. Semantic discrete units on accented speech

In this section, we investigate how accent influences the semantic discrete units extracted by HuBERT, which have been successfully utilized in other speech-to-speech tasks such as speech translation and voice conversion.

Influencing factors	LCSR
Different Speaker but same accent	0.85
Different accent	0.62

Table 5.1.: HuBERT units on accent speech

HuBERT discrete units on accent speech

We employ the HuBERT model Hsu et al. (2021a), trained on general American English data, combined with K-means clustering (with $K = 500$) to extract semantic discrete units. In our experiments, general American English is designated as the target accent, while Indian English serves as the source accent. We construct a parallel dataset using the CMU-L2 ARCTIC corpus Kominek and Black (2004b); Zhao et al. (2018), where both Indian-English and general American speakers read identical scripts. Each speaker's recordings are processed through HuBERT to generate sequences of semantic discrete units. To evaluate the similarity between unit sequences, we utilize the Longest Common Subsequence (LCS) metric. To mitigate the impact of speaking rate variations, consecutive duplicate units are removed from the sequences. Given that utterance durations vary slightly, we compute the Longest Common Subsequence Ratio (LCSR), defined as $LCSR = \frac{LCS}{\text{utterance length}}$, using the shorter utterance length in each pair. Each audio sample ranges between 3 to 5 seconds, and the experiment is conducted over random 1000 data pairs selected from CMU-L2 ARCTIC corpus .Furthermore, we categorize the pair data into two groups based on whether differences from accent changes or speaker changes within the same accent.

The experimental results are summarized in Table 5.1. As shown in the table, the difference between accents (e.g., Indian English vs. American English) leads to substantial alterations in the HuBERT unit sequences compared to the variations observed between different speakers within the same accent. This accent mismatch results in a noticeable drop in LCSR, from 0.85 to 0.62 .

These findings suggest that the patterns formed by semantic discrete units are sensitive to accent differences. Specifically, even though HuBERT is primarily trained to model high-level linguistic information, variations in pronunciation between accents are reflected in the unit sequences. The reduced LCSR indicates that accent-induced pronunciation changes introduce meaningful perturbations in the semantic representation space. This observation aligns with the intuition that while semantic discrete units abstract away speaker identity to a certain extent, they still preserve sufficient phonetic detail to differentiate accents. Consequently, modeling accent conversion at the level of discrete units offers a promising direction, as it allows for focusing on adjusting pronunciation while minimizing disruption to the underlying linguistic content and speaker characteristics.

5.1.3. Proposed framework

We regard accent conversion as a sequence-to-sequence task on the semantic discrete units. As illustrated in Figure 5.1, we propose a two-stage generative framework for accent conversion, consisting of a *conversion stage* that performs pronunciation correction (PC) using a sequence-to-sequence model, and a *speaking stage* that synthesizes speech through a native multi-speaker unit-to-speech (U2S) model conditioned on the discrete units and speaker embeddings.

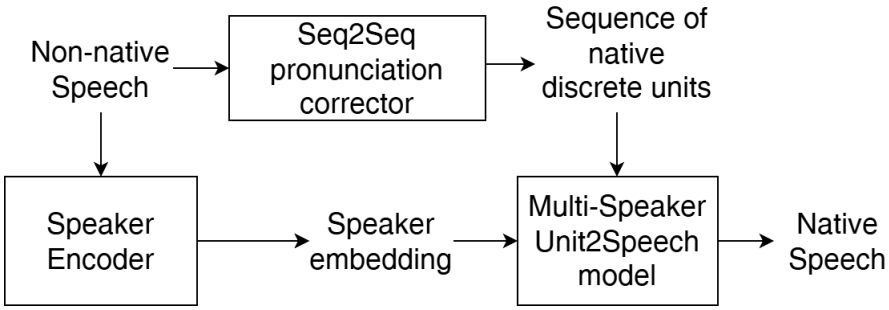

Figure 5.1.: The workflow of the proposed system

The PC model converts non-native accented speech into discrete representation units in native speech. Since these discrete units Lakhotia et al. (2021), which are derived from self-supervised clustering on native speech, effectively separate speech content from speaker identity, they allow for accurate conversion back into native speech while maintaining the original speaker's voice. The speaking stage employs a unit-to-speech (U2S) model that synthesizes speech with the prosody and characteristics of the target accent, conditioned on the converted discrete units and a speaker embedding to preserve the original speaker's identity. Furthermore, these units extracted from native speech are inherently easier to convert back to natural-sounding native speech. While our approach leverages synthetic data for training, all evaluations are conducted on real speech samples.

By dividing the accent conversion process into two stages and utilizing discrete units extracted from raw speech as the intermediary representation, our framework allows the *speaking* stage to operate independently of parallel data. This design enables the use of large-scale, unlabeled speech data in the target accent to train high-quality and diverse speech synthesis models. At the same time, it simplifies the conversion stage, which can focus solely on learning pronunciation or phoneme-level differences using parallel data, without requiring the same speaker across source and target samples.

To summary, the main contributions of this chapter are as follows:

- We propose an AC system capable of transforming speech with different accents into native-like pronunciation while preserving both speaker identity and speech content. Additionally, the proposed framework is capable of prosody conversion and enhances the fluency of non-native speech.

- Since seq2seq pronunciation correction models typically require substantial parallel training data, we introduce a novel data augmentation strategy to generate additional training samples under limited non-native data conditions. Parallel synthetic training data is introduced to effectively train the Voice Conversion (VC) system Ma et al. (2021). We believe it can also be used to train any AC systems in a parallel fashion without an initial demand for such kind of data.
- We pioneer the use of discrete units in training transformer-based AC models. Drawing inspiration from the success of self-supervised pretraining in Speech Recognition Pham et al. (2022b) and Speech-to-Speech Translation Lee et al. (2021), we demonstrate that initializing AC models with pretrained encoder-decoder architectures significantly improves training efficiency compared to random initialization.

5.2. Methodology

The below sections presents the three steps for developing the first unit-based accent conversion system. Initially, we train the Speech-to-Units (S2U) and Unit-to-Speech (U2S) models using a native speech corpus. Next, we generate parallel training data by training a multi-speaker, multi-accented TTS model alongside a mono-speaker native TTS model. Finally, the synthesized parallel data is utilized to train the sequence-to-sequence (seq2seq) pronunciation correction (PC) model.

5.2.1. Speech2Unit model and Multi-speaker Unit2Speech model

Following the optimal configuration proposed in Lakhotia et al. (2021), we utilize the HuBERT model Hsu et al. (2021a) to extract high-level semantic representations from native speech. HuBERT operates in a self-supervised manner to learn contextualized embeddings that capture not just acoustic patterns, but also underlying linguistic and phonemic content. Specifically, we extract continuous features from every 20-ms frame using the sixth transformer layer of a pre-trained HuBERT model, as intermediate layers have been empirically shown to provide the most semantically rich and stable representations.

To convert these continuous features into discrete semantic units, we apply K-means clustering with $K = 500$ centroids trained on the extracted features. Each frame-level embedding is then quantized by assigning it to the nearest cluster center using Euclidean distance. To reduce redundancy and form a more compact and informative sequence, we remove consecutive duplicate cluster indices. The resulting sequence of discrete units serves as a symbolic approximation of the spoken content, capturing the core semantic structure of the utterance while abstracting away speaker-specific and prosodic variations.

These HuBERT-based semantic units are particularly well-suited for accent conversion tasks, as they encapsulate linguistic meaning in a speaker-invariant format. This allows the pronunciation correction module to focus on modifying accent-specific patterns without affecting the overall speech content or speaker identity.

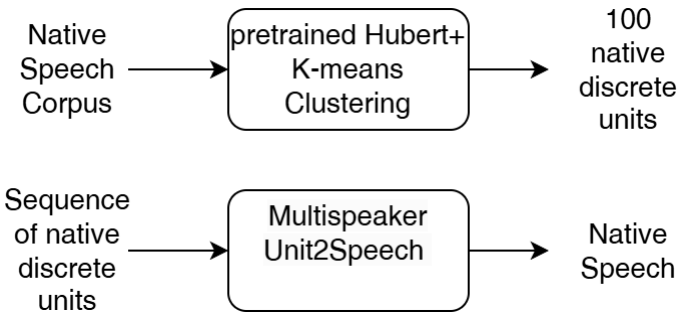

Figure 5.2.: Step 1 - Traning Speech2Unit(S2U) and multispeaker Unit2Speech(U2S)

YourTTS Casanova et al. (2021), a zero-shot multi-speaker TTS model capable of achieving high voice similarity for unseen speakers, is selected as the architecture for our Unit-to-Speech (U2S) model. We treat the discrete units extracted from native speech as text input and train YourTTS on a native speech corpus with a diverse set of speakers. Consequently, our U2S model can reconstruct native speech from discrete unit sequences while preserving the original speaker's identity.

5.2.2. Data augmentation using Multi-accented TTS and Native TTS

The goal of producing synthetic data is to create parallel data that can be used to train the discrete units seq2seq model. Our approach is to start with a strong base TTS model and enhance it through a multi-accented adaptation step, allowing control over accent generation. Building on SYNTACC in Section 6, we adopt YourTTS as the base model and employ weight factorization Pham et al. (2021a) to achieve multi-accented synthesis. As a result, SYNTACC can generate speech in both multiple voices and varied accents.

We train YourTTS on the native speaker corpus to generate output audios. The seq2seq PC model's outputs are discrete units, hence these training sequences should be consistent. To ensure consistent voice, style, and accent in synthetic output, we train YourTTS with audio from a single native speaker. Then applying S2U on synthesized output audios to creates discrete native unit sequences.

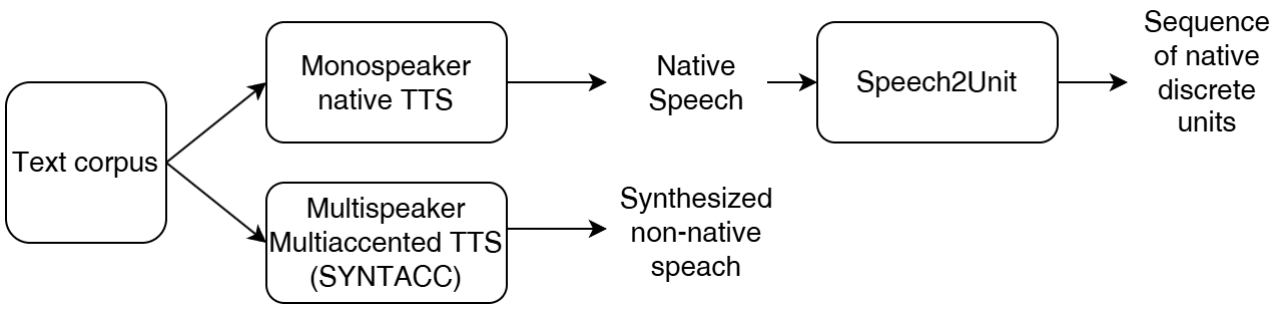

Figure 5.3.: Step 2 - Data augmentation using Multi-accented TTS and Native TTS

After training the Native YourTTS and SYNTACC models, we use them to create parallel training data for the PC. First, we require a large text corpus. Then, for each sentence in this text corpus, we generate corresponding input and output audio. We produce input audio for each sentence using SYNTACC with a random speaker and accent. We can additionally modify the synthesized audio by randomizing the duration noise scale and the inference noise scale of SYNTACC to make the seq2seq PC model more robust on a variety of input sounds. In contrast to the SYNTACC model, we fix these noise scales when doing inference native YourTTS, then use S2U to construct a sequence of discrete native units for output audio. Finally, in the following section, the input non-native audios and matching sequence of discrete units can be utilized to train the seq2seq PC.

5.2.3. Training the Pronunciation Corrector

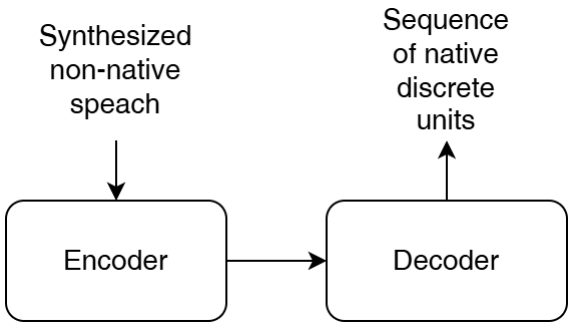

Figure 5.4.: Step 3 -Training seq2seq Pronunciation corrector

Our proposed pronunciation corrector (PC) is a Transformer-based sequence-to-sequence model consisting of a speech encoder and a discrete unit decoder. In this work, we investigate the use of both pretrained encoders and decoders to enhance model performance.

Pretrained Encoder: Wav2Vec 2.0 and HuBERT

The encoder in our PC model is initialized with a self-supervised speech representation model. We explore two widely used pretrained models: Wav2Vec 2.0 and HuBERT.

HuBERT and Wav2Vec 2.0 apply self-supervised learning to extract meaningful speech representations. Wav2vec 2.0 uses contrastive learning to distinguish true audio representations from distractors. HuBERT predicts masked frame labels derived from clustered hidden representations, enabling it to learn without explicit negative samples. While both models share a Transformer-based structure, HuBERT's iterative clustering approach makes it particularly effective for discrete unit extraction.

Since our discrete units are extracted from the representations of HuBERT, we use this pretrained model to initialize the encoder weights. Additionally, we employ Wav2Vec 2.0 with relative attention Pham et al. (2020) to further improve performance.

Pretrained Decoder: MBart50

BART (Bidirectional and Auto-Regressive Transformers) was introduced as a denoising autoencoder for text, leveraging the Transformer architecture to reconstruct text sequences from noisy inputs. The core idea behind BART is to apply a bidirectional encoder to understand the context of a given sentence, followed by an autoregressive decoder to generate a reconstructed version of the sentence. This structure allows the model to capture both local and global dependencies in the text, making it particularly effective for tasks like text generation and sequence-to-sequence learning.

The multilingual extension of BART Lewis et al. (2019), MBART50 Liu et al. (2020b), extends this denoising autoencoding principle to 50 languages, making it a versatile model for cross-lingual tasks such as machine translation. By pretraining on a large set of multilingual data, MBART50 learns to generalize across diverse languages and sentence structures, providing strong cross-linguistic capabilities. This multilingual pretraining allows MBART50 to effectively map between different languages, enabling it to perform tasks like translation and text generation in various languages with minimal task-specific fine-tuning.

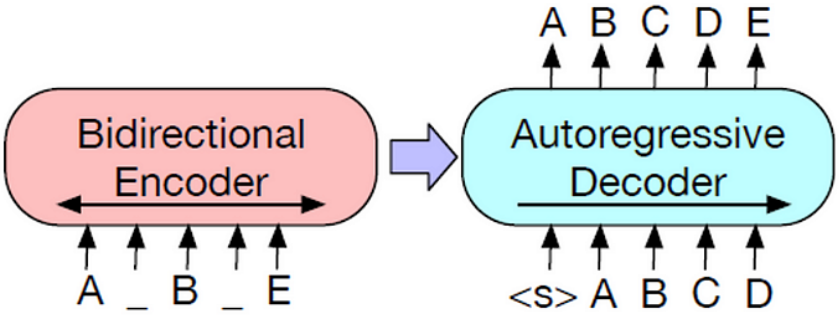

Figure 5.5.: MBart architecture Liu et al. (2020b)

MBART is trained using a denoising autoencoding objective, where random noise is introduced into the input text. The model learns to reconstruct the original text by applying random word masking and shuffling. The training process consists of encoding a corrupted sentence $\tilde{x}$ into a hidden representation h using a bidirectional encoder, followed by autoregressive decoding to predict the next token in the sequence:

$$h = \mathscr{T}_{\text{enc}}(\tilde{x}), \tag{5.1}$$

$$P(y_t|y_{<t},h) = \text{Softmax}(\mathscr{T}_{\text{dec}}(y_{<t},h)), \tag{5.2}$$

where h represents the encoded hidden state, and y_t denotes the predicted token at timestep t.

In our approach, the discrete units are derived from English speech data, which we treat as a symbolic language with structural similarities to written text. This allows us to leverage MBART50's pretrained

decoder, which has been fine-tuned to handle a broad range of language structures. Given MBART's ability to generalize across different languages, we adapt it to our task by replacing its original embedding layer with a new matrix specifically trained on discrete units extracted from speech. This adaptation enables the model to handle speech-based discrete units in the same way it processes textual data.

To further optimize the system for accent conversion, we fine-tune the MBART50 model in combination with Wav2Vec and HuBERT pretrained features. These models are adapted to generate discrete units, enabling robust pronunciation correction through fine-tuned discrete unit generation. This process enhances our system's ability to perform accent conversion by leveraging MBART50's powerful sequence-to-sequence generation capabilities and multilingual generalization, making it suitable for multi-accent speech synthesis and translation tasks.

5.3. Experiment

5.3.1. Data and training description

5.4. Data and Model Training

The Speech2Unit (S2U) and Unit2Speech (U2S) models are trained using different datasets: the LJSpeech corpus Ito (2017) for S2U and the LibriTTS-R corpus Zen et al. (2019); Koizumi et al. (2023) for U2S. LJSpeech consists of 13,100 recordings of a single native English female speaker reading sentences aloud, providing a clean and consistent speech source. Following the methodology outlined in Section 5.2.1, we develop the S2U model specifically for English speech. To build a multi-speaker U2S system, we train a YourTTS model using discrete units from S2U as input, along with corresponding speaker embeddings generated by a pretrained speaker encoder Wan et al. (2020), and map them to target speech waveforms.

For model training and data augmentation, we utilize audio from the LJSpeech corpus to train YourTTS and fine-tune the SYNTACC model using L2-Arctic Zhao et al. (2018). The L2-Arctic dataset contains non-native English speech from 24 speakers, covering six different accents: Hindi, Mandarin, Vietnamese, Korean, Spanish, and Arabic. Each speaker records the same 1,152 sentences, resulting in approximately three hours of speech per accent, making it a low-resource scenario. The SYNTACC model is fine-tuned, and YourTTS is trained from scratch using hyperparameter settings from their respective original implementations. Additionally, transcripts from the CommonVoice corpusArdila et al. (2020) are used to generate parallel training data, as described in Section 5.2.2.

To examine the impact of data augmentation on PC performance, we generate one million synthetic utterances using two different strategies. In the *non-overlapped sentence strategy*, we randomly sample one million unique sentences from a text corpus and synthesize one non-native speech sample per sentence, each assigned a random accent and speaker. In contrast, the *overlapped sentence strategy* selects

166,000 sentences and generates six versions of each—one for each accent—using different randomly selected speakers. In both cases, we generate a single native speech recording per sentence as the corresponding target audio. The second strategy is expected to improve the PC model's ability to generalize across diverse accents. Before synthesizing the dataset, we partition the text corpus into training and validation sets at a 1000:1 ratio. For testing, we use an in-house dataset consisting of approximately 1,000 sentences (three hours of speech) recorded by speakers with Chinese, Indian, Arabic, and Vietnamese accents. Since none of these speakers appear in the training set, this evaluation setup simulates a zero-shot learning scenario.

For PC training, we utilize Wav2Vec 2.0 and HuBERT with the Large configuration, both having a hidden size of 1024. The MBart50 decoder also employs the same hidden size. We adopt a training batch size equivalent to approximately one hour of speech and apply gradient accumulation to stabilize updates. The learning rate follows a linear decay schedule, starting at 0.001, and the model converges after approximately 20,000 updates (around five to six hours of training on a single NVIDIA RTX A6000 GPU with 48GB RAM). During inference, we use beam search with a beam size of 8 to generate corrected pronunciations. The implementation of the PC model is based on the Hugging Face framework[1].

5.4.1. Evaluation Metrics

Test Perplexity For the PC, we estimate the perplexity on our in-house test data. Perplexity is a measure of how efficiently a model predicts the next unit in a sequence of units. In our case, it also indicates how well the PC learns the patterns of native speech and decodes them into discrete units. The perplexity is computed as:

$$PPL = \exp\left(-\frac{1}{N}\sum_{i=1}^{N}\log P(u_i|u_{1:i-1})\right) \tag{5.3}$$

where N is the number of units in a sequence, and $P(u_i|u_{1:i-1})$ represents the model's predicted probability of the i-th unit given the preceding units.

For each sentence in the test set, we use the native YourTTS model to synthesize a native audio with the same content, followed by Speech2Unit to generate a ground-truth sequence of units. These ground-truth sequences serve as a reference for estimating the perplexity of the PC on the test data.

Furthermore, to assess how data augmentation and pretrained encoder-decoder configurations impact performance, we compare different data augmentation strategies and different weight initialization methods (with and without a pretrained model). The best PC configuration, determined by the lowest test perplexity, is selected for evaluating the subjective metrics of the entire system.

[1] `https://github.com/huggingface`

Accentedness, Fluency, Speaker Similarity Mean Opinion Score (Sim-MOS), and Mean Opinion Score (MOS). To assess the overall performance of our system, we employ three subjective evaluation metrics. A total of 50 randomly selected sentences from the test set are used for evaluation. Each test sentence undergoes three different evaluations conducted by 20 American participants, who listen to the synthesized speech and rate its quality on a 5-point scale: 1 (bad), 2 (poor), 3 (fair), 4 (good), and 5 (excellent).

In the Accentedness test, participants evaluate how closely the generated speech resembles native pronunciation. The Fluency test measures the naturalness and smoothness of speech delivery. For the Sim-MOS test, listeners assess how similar the synthesized speech is to the original target speaker's voice based on vocal timbre. After collecting the ratings, we compute the mean scores along with a 95% confidence interval for all subjective metrics.

We consider three recent baseline systems for comparison. The first one Quamer et al. (2022) is also based on a sequence-to-sequence architecture: it transforms non-native speech into the mel-spectrogram of native speech, followed by an external vocoder to reconstruct the waveform. The other two baselines Jia et al. (2023); Jin et al. (2023a) adopt non-autoregressive approaches with disentanglement networks that separate accent attributes from the original speech. As their source codes are not publicly available, we evaluated our system by comparing its outputs with their published audio samples.

5.4.2. Results

Table 7.1 shows that the sentence-overlapping configuration consistently achieves lower text perplexity than the non-overlapping setting across all weight initialization scenarios. This suggests that datasets with diverse accents and multiple speakers per sentence are particularly well-suited for the accent conversion task, as they may enable the pronunciation converter (PC) to more effectively learn to distinguish among various accents. Regarding weight initialization, the combination of the Wav2Vec encoder and MBart decoder yields the lowest perplexity. Therefore, this configuration, together with the sentence-overlapping setting, is chosen for the subjective evaluation.

As shown in Table 7.2, our model outperforms Baseline-1 in all audio quality metrics. This suggests that native discrete units may serve as a more effective representation than mel-spectrograms for the accent conversion task. Compared to the other two baselines, our model achieves better performance in terms of accentedness and fluency, although it is slightly inferior to Baseline-3 in speaker similarity. This observation indicates that our model might alter the audio more substantially, resulting in outputs that sound more native but may cause listeners to perceive a change in speaker identity. *b*reak The two non-autoregressive baselines preserve the input audio's duration while inducing minimal changes in fluency. This ability to synchronize input and output makes them particularly suitable for applications like accent-aware video dubbing. In contrast, our system significantly enhances fluency, making it more

Models	PPL
No pretrained	
+ non-overlapped sentence	3.63
+ overlapped sentence	2.45
Wav2vec encoder + no pretrained decoder	
+ non-overlapped sentence	3.21
+ overlapped sentence	2.25
HuBERT encoder + no pretrained decoder	
+ non-overlapped sentence	3.28
+ overlapped sentence	2.32
Wav2vec encoder + MBart decoder	
+ no sentence overlapped	3.11
+ sentence overlapped	**2.16**
HuBERT encoder + MBart decoder	
+ non-overlapped sentence	3.23
+ overlapped sentence	2.24

Table 5.2.: Test perplexity

Models	Accentness	Sim MOS	Fluency
Input	2.12 ± 0.05		3.31 ± 0.03
Proposed			
In-house test	$\mathbf{4.46} \pm 0.12$	3.89 ± 0.09	$\mathbf{4.55} \pm 0.07$
Public test	$\mathbf{4.42} \pm 0.11$	3.95 ± 0.07	$\mathbf{4.51} \pm 0.06$
Baseline-1	3.67 ± 0.11	3.61 ± 0.06	3.83 ± 0.08
Baseline-2	3.65 ± 0.09	3.92 ± 0.05	3.94 ± 0.09
Baseline-3	3.93 ± 0.10	$\mathbf{4.23} \pm 0.10$	3.84 ± 0.10

Table 5.3.: Subjective metrics

appropriate for applications focused on language comprehension. Furthermore, our model demonstrates consistent performance across both in-house and publicly available test sets.

5.5. Conclusion

In this paper, we have demonstrated that a well-designed controllable accented TTS system can serve as an efficient tool for generating large-scale parallel training data for accent conversion). Furthermore, we believe that this data augmentation approach can be extended to improve the availability of accented speech data for tasks such as speech recognition and translation. Additionally, we have shown how leveraging a pretrained encoder-decoder architecture with native discrete units enhances the training of a many-to-one AC system. Experimental results confirm that our proposed method effectively transforms speech from unseen speakers into a native accent while improving both fluency and accent accuracy.

In this chapter, we have successfully trained a robust model for the mapping framework, which shows great promise for accent conversion tasks. However, the mapping framework faces several limitations, such as slow inference times and mismatches between input and output lengths, which can hinder video conference applications. In the next chapter, we will explore alternative approaches, focusing on the disentangle-resynthesis framework, to address these issues and further enhance the flexibility and performance of our system.

6. Disentangle-Resynthesis Framework for Accent Conversion

In the previous chapters, our approaches primarily focused on building a direct mapping function from one accent to another. While this method has shown promising results, it also comes with inherent limitations, such as restricted flexibility in accent control and potential distortions due to direct transformation. In this chapter, we explore an alternative approach: *disentangle-resynthesis*. Instead of learning a direct mapping, this method decomposes speech into distinct components, such as speaker identity, linguistic content, prosody, and accent. By explicitly modeling these factors, we gain more control over the synthesis process, allowing for more precise accent conversion without affecting speaker identity or fluency. We begin by detailing the theoretical foundation of disentanglement in speech processing, followed by our model architecture and training methodology. Finally, we evaluate the performance of this framework and compare it to the mapping-based approach discussed in previous chapters.

6.1. Overview

In this chapter, we introduce a novel accent conversion (AC) approach that goes beyond accent conversion by explicitly improving the pronunciation of non-native speakers. By leveraging both the non-native audio and its corresponding transcript, we generate an ideal ground-truth audio with native-like pronunciation while maintaining the original duration , emotion and prosody. This ground-truth data helps the model learn not only to eliminate accents but also to establish a direct mapping between accented and native speech.

To achieve high-quality waveform reconstruction for AC, we adopt the end-to-end VITS framework, which integrates variational inference and adversarial training for natural and expressive speech synthesis. As a result, our system generates speech that closely resembles native pronunciation while preserving the original speaker's identity. Evaluation results further confirm that our approach significantly improves both accent conversion and pronunciation quality.

Direct-mapping AC models from previous chapters, typically based on autoregressive sequence-to-sequence (seq2seq) architectures, rely heavily on parallel data for training. However, such data—utterances from the same speakers in different accents—is scarce and difficult to obtain. To mitigate this issue, data augmentation techniques such as text-to-speech (TTS) (in Section 4) and voice conversion (VC) (in Section 3) are employed to generate ground-truth audio for non-native speakers. However, these synthesized ground-truth recordings often originate from different speakers and exhibit variations in duration and

prosody compared to the original non-native utterances. While encoder-decoder models with attention mechanisms can help align input and output sequences, the differences in length make these models less suitable for applications requiring precise time synchronization, such as real-time video conferencing and dubbing.

The disentangle-resynthesis AC leverages non-parallel data, which is abundant and diverse. To effectively utilize this data, these models decompose speech into separate features such as speaker identity, content, prosody, and accent, which are then recombined to generate the original waveform. Content features are often represented by bottleneck features (BNFs), extracted from self-supervised pre-trained models Li et al. (2023) (e.g., WavLM Chen et al. (2022), Wav2vec Baevski et al. (2020c), HuBERT Hsu et al. (2021b)) , ASR bottlenecks Jia et al. (2023), or ASR output in logits form Jin et al. (2023b). However, BNFs also capture accent-related feature and non-native pronunciation issues. While methods like adversarial training Jin et al. (2023b)or Pseudo-Siamese networks Jia et al. (2023) attempt to disentangle accent information, they still struggle to improve pronunciations due to the absence of ground-truth.

Building on the strengths and limitations of existing AC models, we propose a novel framework for training a disentangle-resynthesis AC model using generated parallel data. Our key hypothesis is that a TTS system trained exclusively on native speech can learn accent-independent linguistic representations. Furthermore, this native TTS system serves as a reliable source for generating ideal ground-truth data for non-native speakers, ensuring native pronunciation while preserving speaker identity, duration, prosody, and precise alignment with the original non-native audio. These accent-independent representations not only facilitate the creation of high-quality training data but also enable effective knowledge distillation into our AC model, enhancing its ability to learn accent-independent features.

Previous studies Zhou et al. (2023); Chen et al. (2024) have used native TTS models to guide AC models, often by sharing the same decoder and distilling knowledge from the TTS text encoder to the AC encoder. A key challenge in these approaches is aligning the outputs of the TTS and AC encoders due to their differing lengths. In Zhou et al. (2023), both models use the autoregressive Tacotron Shen et al. (2018) architecture, necessitating an external unstable attention mechanism for alignment. Another approach Chen et al. (2024) employs the non-autoregressive FastSpeech 2 Ren et al. (2021) TTS model. In this approach, alignment is achieved through an external length regulator based on Montreal Force Alignment, which uses a GMM-HMM ASR model to upsample the TTS encoder's output, enabling both encoders to produce outputs of similar length. This latter method closely aligns with our proposed approach. These approaches use a separate vocoder to convert mel-spectrograms into waveforms.

The non-autoregressive VITS framework has proven successful in both TTS Kim et al. (2021a) and Voice Conversion Li et al. (2023). Inspired by this success, we adopt a modified VITS framework for our AC system due to several advantages. VITS uses internal monotonic alignment, simplifying audio-text alignment and improving training efficiency. Additionally, as an end-to-end architecture, VITS

eliminates the need for a separate vocoder to convert mel-spectrograms into waveforms, setting it apart from previous AC methods.

Overall, to further enhance the AC model's training process, we introduced several key improvements. In the first stage, we simultaneously pre-train the AC model while training the VITS-based native TTS, allowing the AC model to learn the distribution of native audio data comprehensively. After training the native VITS-based TTS, we leverage it to generate ideal ground-truth data for non-native audio, as described in the next section. In the second stage, we fine-tune the pre-trained AC model using the non-native input alongside the generated ground-truth output and also distillate knowledge from the native TTS. In summary, we train the VITS-based TTS model to assist in initializing the weights, generating ideal ground-truth data, and facilitating knowledge distillation for the AC model.

6.2. Methodology

6.2.1. Training native VITS and pretrained AC model

In the first stage, we pre-train the AC model and train the native TTS model using the VITS framework. VITS Kim et al. (2021a) is a conditional variational autoencoder architecture enhanced with a normalizing flow mechanism Rezende and Mohamed (2015). It comprises three main components: a posterior encoder, a prior encoder, and a waveform generator. These modules model the distributions $q_\phi(z|x)$, $p_\theta(z|c)$, and $p_\psi(y|z)$, respectively.

Specifically, $q_\phi(z|x)$ and $p_\psi(y|z)$ represent the posterior and data distributions, parameterized by the posterior encoder ϕ and the HiFi-GAN waveform generator ψ Kong et al. (2020a), where x denotes the speech input, z is the latent variable, and y is the generated waveform. The prior distribution $p_\theta(z|c)$ is parameterized by the prior encoder θ and further refined by a normalizing flow f. Here, the conditioning input c can be either audio or text, depending on the specific task configuration.

Our AC and TTS models share several key components, including the HiFi-GAN decoder, posterior encoder, speaker encoder, F0 encoder, and normalizing flow, as illustrated in Fig. 6.3a. The posterior encoder receives linear spectrograms x_{lin} as input and samples a latent variable z. This latent representation z, together with the speaker embedding g produced by the speaker encoder and the F0 sequence embeddings $F0$ obtained from the F0 encoder, is passed to the HiFi-GAN vocoder to synthesize the output waveform.

The AC and TTS models each have separate prior encoders. The AC model's prior encoder θ_{audio} includes a content encoder which extracts speech features and a bottleneck extractor to generate a normal distribution out of BNFs. Previous works have utilized self-supervised pre-trained representations Li et al. (2023) or ASR features (bottleneck or logits)Jin et al. (2023b) as content encoder outputs. These features contain information related to content, speaker, and accent. To filter out unwanted information,

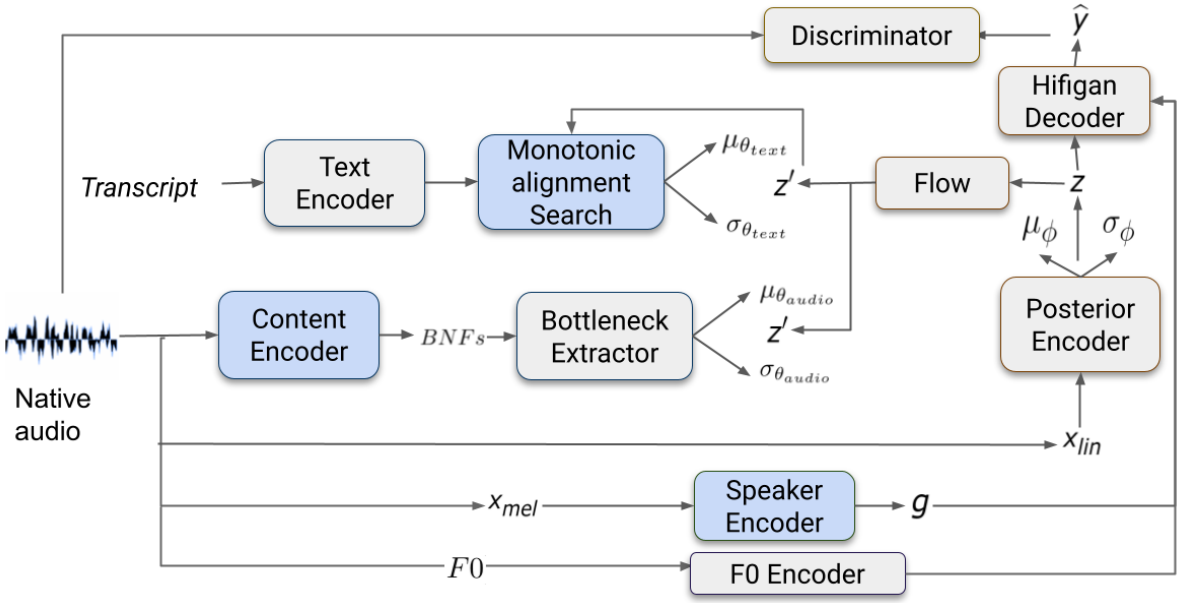

Figure 6.1.: Training TTS and pretraining Accent Conversion model

the $BNFs$ are processed by the bottleneck extractor. The TTS model's prior encoder consists of a text encoder θ_{text} , which takes the transcript as input. Monotonic Alignment Search (MAS) aligns the text encoder output with the normalizing flow output, to match the differing lengths for text representations and audio representations.

The training loss is split into Variational Auto-encoder (VAE) loss and Generative Adversarial Network (GAN) loss. GAN loss includes adversarial loss $\mathscr{L}_{adv}(D)$ and $\mathscr{L}_{adv}(G)$ for the discriminator and generator, along with feature matching loss $\mathscr{L}_{fm}(D)$ for the generator G . VAE loss consists of reconstruction loss $\mathscr{L}_{rec}$ (L1 distance between target and predicted mel-spectrogram) and $\mathscr{L}_{kl}$ (KL divergence between the prior and the posterior). To address training-inference mismatch of VITS, we integrate the full end-to-end inference during training to improve voice quality. Specifically, L1 distance between ground-truth x_{mel} and inference output x_{mel}^{e2e} generated by $p_\psi(x|f^{-1}(z^{'}),g,F0)$, where the latent $z^{'}$ is sampled from $p_{\theta_{audio}}(z^{'}|c_{\text{audio}})$ or $p_{\theta_{text}}(z^{'}|c_{\text{text}})$, and both g and $F0$ are extracted from speaker encoder and F0 encoder respectively, is added to reconstruction loss. KL loss $\mathscr{L}_{kl}$, which sums the KL divergence between the prior distributions $p_{\theta_{text}}(z|c_{\text{text}})$ and $p_{\theta_{audio}}(z|c_{\text{audio}})$, and the posterior distribution $q_\phi(z|x)$. The overall training loss for pre-training phase can be expressed:

$$\mathscr{L}_{kl} = KL(q_\phi(z|x)||p_{\theta_{text}}(z|c_{text})) + KL(q_\phi(z|x)||p_{\theta_{audio}}(z|c_{audio})) \tag{6.1}$$

$$\mathscr{L}_{recon} = ||x_{mel} - \hat{x}_{mel}|| + ||x_{mel} - x_{mel}^{e2e}|| \tag{6.2}$$

$$\mathscr{L}(G) = \mathscr{L}_{rec} + \mathscr{L}_{kl} + \mathscr{L}_{adv}(G) + \mathscr{L}_{fm}(G) \tag{6.3}$$

$$\mathscr{L}(D) = \mathscr{L}_{adv}(D) \tag{6.4}$$

6.2.2. Generating ideal ground-truth

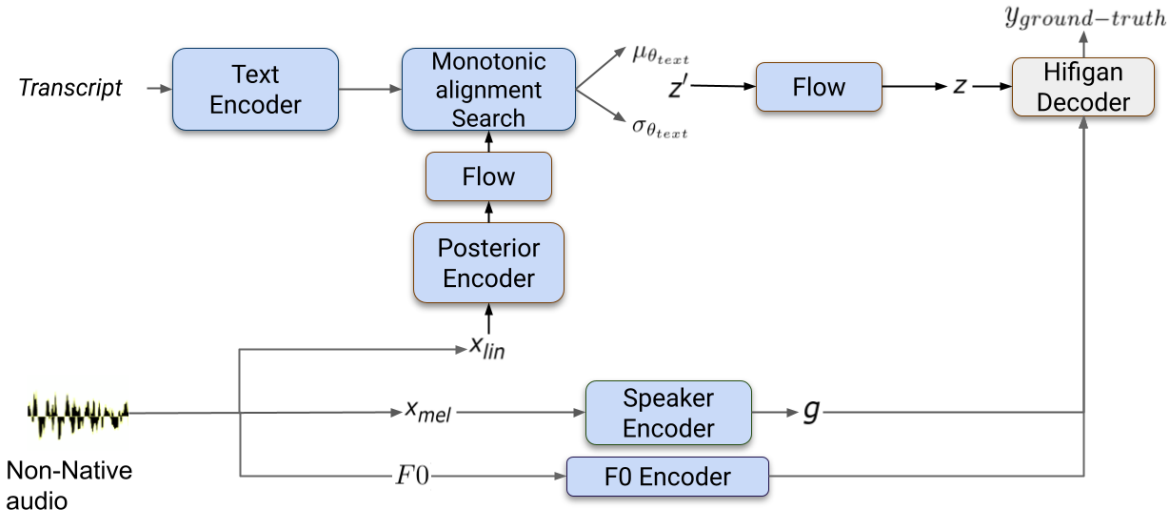

Figure 6.2.: Generating ideal ground-truth

The native TTS model trained in the earlier stage is used to generate ground-truth audio corresponding to each non-native input. We begin by sampling the latent variable z from the posterior distribution of the non-native audio x. Monotonic Alignment Search (MAS) is then employed to determine the alignment A that maximizes the log-likelihood of the transformed latent variable $f(z)$ under the distribution $p_{\theta_{\text{text}}}(z' \mid c_{\text{text}}, A)$. This alignment is subsequently used to upsample the text input.

$$z \sim q_\phi(z|x_{non-native}) = N(z; \mu_\phi(x), \sigma_\phi(x)) \tag{6.5}$$

$$A = argmax \log N(f(z); \mu_{\theta_{text}}(c_{text}, A), \sigma_{\theta_{text}}(c_{text}, A)) \tag{6.6}$$

$$c_{upsample} = upsampling(c_{text}, A) \tag{6.7}$$

Next, following the inference procedure of VITS, we obtain the latent variable z' by sampling from the prior distribution $p_{\theta_{\text{text}}}(z' \mid c_{\text{upsample}})$, where c_{upsample} denotes the upsampled text input. This latent variable is then passed through the inverted normalizing flow f^{-1} to refine its representation. Finally, the ground-truth audio $\hat{y}_{\text{ground-truth}}$ is synthesized using the HiFi-GAN decoder from the distribution $p_\psi(x \mid f^{-1}(z'), g, F0)$, where the speaker embedding g and F0 contour are extracted from the corresponding non-native audio.

In summary, the synthetic ground-truth audio $y_{\text{ground-truth}}$ is generated from the native transcript with perfect pronunciation, while leveraging MAS alignment, F0, and speaker embedding from the non-native audio to preserve the original duration, prosody, and speaker identity.

$$z^{'} \sim p_{\theta_{text}}(z^{'}|c_{upsample}) \tag{6.8}$$

$$y_{ground-truth} = Hifigan(f^{-1}(z^{'}), g, F0) \tag{6.9}$$

6.2.3. Finetuning AC model with ideal ground-truth and knowledge distillation from native TTS

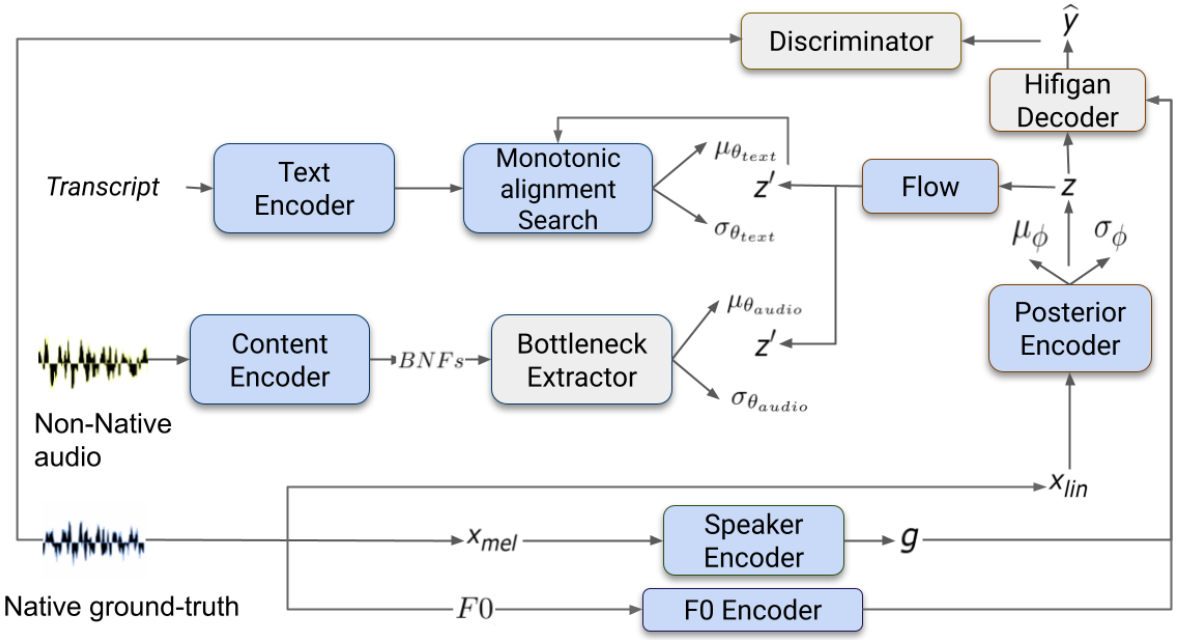

Figure 6.3.: Fine-tuning Accent Conversion model with ground-truth and knwoledge distillation from TTS

In the third step, we fine-tune the AC model using the synthetic ground-truth data. The primary goal of this stage is to enable the model to generate high-quality native speech from non-native input. Specifically, the content encoder processes the non-native speech, while all other components utilize the synthetic ground-truth data, as illustrated in Fig. 6.3. During this stage, we freeze the parameters of all components except for the bottleneck extractor and the HiFi-GAN decoder. The bottleneck extractor is fine-tuned to capture accent-independent content representations from the non-native inputs.

To guide this learning, we introduce a distillation loss, formulated as the Kullback–Leibler (KL) divergence between the two prior distributions: $p_{\theta_{\text{text}}}(z \mid c_{\text{text}})$ and $p_{\theta_{\text{audio}}}(z \mid c_{\text{audio}})$. The overall training objective during this fine-tuning stage is defined as:

$$\mathscr{L}_{distill} = KL(p_{\theta_{audio}}(z|c_{audio})||p_{\theta_{text}}(z|c_{text})) \tag{6.10}$$

$$\mathscr{L}(G) = \mathscr{L}_{rec} + \mathscr{L}_{kl} + \mathscr{L}_{distill} + \mathscr{L}_{adv}(G) + \mathscr{L}_{fm}(G) \tag{6.11}$$

6.2.4. Overall model architecture detail

Our posterior encoder, HiFi-GAN decoder, normalizing flow, and text prior encoder follow the original VITS framework Kim et al. (2021a). For the content encoder, we fine-tune a pretrained Wav2Vec 2.0 model on the LibriSpeech and L2-Arctic datasets using CMUdict phoneme labels, and use the bottleneck features from its final layer as content representations. After fine-tuning, the parameters of the content encoder are frozen.

To improve text-audio alignment, we use phoneme labels instead of raw text as input to the text encoder. Both the bottleneck extractor and text encoder share the same Transformer-based architecture as used in VITS. We retain Wav2Vec 2.0's downsampling rate of 320 for extracting mel-spectrograms, F0, and linear spectrograms, and set the HiFi-GAN decoder's upsampling rate accordingly.

F0 sequences are computed using the YIN algorithm with the same downsampling factor, and the F0 encoder follows the architecture from Wang et al. (2021b). For speaker representation, we adopt the pretrained speaker encoder from Liu et al. (2021b).

6.2.5. Inference

Our framework supports inference both with and without a transcript. In the absence of a transcript, the model extracts content information directly from the non-native audio using the content encoder and the bottleneck extractor within the prior encoder. A latent variable is then sampled from the prior distribution and used to generate the waveform, conditioned on speaker and prosody embeddings.
When a transcript is available during inference, the system can leverage the text prior encoder instead of the audio prior encoder, as described in Section 6.2.2, enabling the generation of speech with native-like pronunciation. This dual-mode inference capability improves the flexibility and quality of the system, particularly when high-quality transcripts are provided.

6.3. Experiment and Result

6.3.1. Data

We use the LJspeech dataset Ito (2017), which features consistent pronunciation from a single native speaker. To augment the dataset, we employ FreeVC Li et al. (2023), a voice conversion model, to generate multi-speaker native-accented utterances from the original voices. The augmented multi-speaker dataset used in our work consisted of approximately 65,000 utterances from 100 VCTK speakers. These were generated by augmenting the 13,000 utterances from the LJspeech dataset, with each utterance paired with five random speaker embeddings from the VCTK corpus Yamagishi et al. (2019). This ensured a diverse representation of voices with one specific native-like pronunciation. This augmented multi-speaker dataset is utilized, enabling the system to be trained in a multi-speaker setting during the

first step. In the second stage, we fine-tune the AC model on the L2ARCTIC dataset Zhao et al. (2018), featuring 24 accented speakers across 6 accents. For each accent, we select 3 speakers for training and the remaining speakers for testing. Each speaker's utterances are split into a training set of 1,032 non overlap utterances, a validation set of 50, and a test set of 50. The test set, chosen using our competive ASR model Pham et al. (2020), focuses on utterances with the high average WER (larger than 10) across all speakers, with assumption that the higher WER rates mean stronger accents.

6.3.2. Evaluation metrics

Subjective tests

Nativeness and Speaker Similarity Tests: Ten participants conducted two evaluations using a 5-point scale (1-bad, 2-poor, 3-fair, 4-good, 5-excellent). In the Nativeness test, they rated how closely the converted audios resembled native speech. In the speaker similarity test, participants assessed the similarity between the voice identity of the original input and the converted audio, providing a score out of 5 for each.

Objective tests

Word error rates (WER), Accent classifier accuracy (ACC) and Speaker Embedding Cosine Similarity (SECS) : To evaluate the improvement in pronunciation, we use the Word Error Rate (WER) from our competitive seq2seq Transformer ASR model Pham et al. (2020). A lower WER indicates better intelligibility in the conversion process. SECS is employed to measure speaker similarity by computing the cosine similarity between speaker embeddings of the original speech and the converted speech. We utilize the state-of-the-art speaker verification model from Chen et al. (2022) to extract speaker embeddings, which suggests that a cosine similarity greater than 0.85 indicates that both audios likely originate from the same speaker.

Additionally, we train an accent classifier to determine whether an input audio is native or non-native. The classifier architecture and training setting is similar to Nguyen et al. (2022). We compute two accuracy scores, one for original non-native audio set and one for converted native audio set. A better AC is expected to show a larger gap between these two ACCs.

6.3.3. Experimental setup

Trained on both native and non-native audio with ASR loss, the final layer of the content encoder produces representations that are closely aligned with linguistic content. This results in a model that is more accent- and speaker-independent, enabling precise mapping from non-native pronunciation to the correct linguistic information. Consequently, we use the model pre-trained in the first stage as the baseline. The proposed model refers to the variant fine-tuned in the second stage.

To evaluate the impact of fine-tuning on the synthetic ground-truth data, we train a variant that only incorporates the knowledge distillation loss L_{distill} between the text and audio encoders during the second stage. Additionally, we assess the audio quality of the synthetic ground-truth to evaluate system performance when the correct transcript is available.

The training hyperparameters are set similarly to those in the original VITS, with the system trained for 600,000 steps in the first stage and 200,000 steps in the second stage. Sample evaluation audio files are available in the accompanying GitHub repository. [1]

6.3.4. Result

Table 6.1.: Objective metrics

Models	WER	ACC	SECS
Original	18.3	98.3	1.0
Synthetic ground-truth	5.1	13.9	0.83
Baseline	17.1	18.9	0.82
w/o Synthetic ground-truth	14.3	17.0	0.84
Proposed	**12.4**	17.2	0.83

The objective metric evaluation, presented in Table 6.1, shows that the proposed method outperforms all other approaches in terms of Word Error Rate (WER). This indicates that the method successfully enhances pronunciation, bringing it closer to native speech. Even without the synthetic ground-truth, the model still benefits from knowledge distillation from the native TTS, though the pronunciation improvement is less pronounced. The synthetic ground-truth, generated from transcripts, achieves a WER of 5.1, confirming the high quality of these ground-truth audios.

Furthermore, the Speaker Embedding Consistency Score (SECS) remains stable between 0.82 and 0.84 across all configurations, demonstrating that the speaker identity is consistently preserved across all methods. In terms of Accent Consistency (ACC), the results are somewhat weaker compared to other metrics. Since the goal is to preserve the prosody of the original audio, which can sometimes reflect the speaker's accent, the model occasionally classifies the converted audio as non-native. This prosody retention may, in some cases, result in less effective accent conversion.

Table 6.2.: Subjective metrics

Models	Nativeness	Sim-MOS
Original	1.67 ± 0.05	-
Synthetic ground-truth	3.93 ± 0.05	3.84 ± 0.15
Baseline	3.32 ± 0.08	3.94 ± 0.18
w/o Synthetic ground-truth	3.71 ± 0.08	3.9 ± 0.19
Proposed	$\mathbf{3.87} \pm 0.08$	3.85 ± 0.19

[1] `https://accentconversion.github.io/demo`

The subjective metric evaluation, also shown in Table 6.2, reveals that our proposed method performs best in the nativeness test. Similar to the objective evaluation, the Sim-MOS results remain consistent across all configurations.

6.4. Conclusion

This work presents a significant advancement in accent conversion and native-like pronunciation correction. Through the introduction of a two-stage training process that leverages VITS framework, we enable the AC model to better capture the accent-independent feature of speech while maintaining the speaker's identity. The use of generated ground-truth data, alongside knowledge distillation from native TTS, further enhances the system's ability to correct non-native pronunciations effectively. Objective and subjective evaluations demonstrate that our approach not only improves accent conversion but also addresses pronunciation issues, providing a more natural and comprehensible output.

7. Streaming Accent Conversion

In the previous chapter, we introduced a two-stage training framework leveraging the VITS model, enabling the accent conversion (AC) system to capture accent-independent speech features while preserving speaker identity. By integrating native TTS for ground-truth generation and knowledge distillation, our approach significantly improved both AC and pronunciation correction, as demonstrated by extensive evaluations.
However, the AC model from the previous chapter is not suitable for streaming applications due to its reliance on a global attention mechanism, which requires processing entire utterances at once. In this chapter, we extend the previous work to real-time applications by modifying the architecture by integrating an Emformer encoder and an optimized inference mechanism, ensuring low-latency processing. Similarly to the previous chapter, we leverage a native TTS model to generate high-quality ground-truth data, facilitating efficient training. Experimental results show that our streaming AC model achieves performance comparable to state-of-the-art AC models while maintaining stable latency, making it the first AC system capable of real-time streaming.

7.1. Overview

People often exhibit accents and mispronunciations while communicating. Accent conversion aims to enhance intelligibility while preserving content, emotion, and speaker identity. *Streaming AC* is particularly important for real-time applications such as video conferencing Waibel and Fuegen (2012), where seamless interaction is crucial. Traditional AC models process full utterances, leveraging long-range context for accurate accent conversion and pronunciation correction.

However, streaming AC requires real-time speech conversion, posing challenges for existing AC models that rely on processing entire utterances at once. Unlike non-streaming models, a streaming AC system must operate incrementally on speech frames or chunks while maintaining naturalness and intelligibility. In this work, we adapt the previous non-streaming AC model for streaming by modifying its architecture and training process while leveraging knowledge distillation. To the best of our knowledge, this is the first streaming AC model capable of both accent conversion and pronunciation improvement.

The previous AC combine both disentangling and mapping approaches into a unified framework in Chapter 6. We hypothesized that a TTS system trained solely on native speech will produce accent-independent linguistic representations. Additionally, this native TTS system is able to generate ideal

ground-truth data for non-native speakers, ensuring native pronunciation, same speaker identity, duration, prosody, and precise alignment with the original non-native audio. Their AC model leverages TTS text representations to learn accent-independent features and uses synthetic ground-truth to learn the mapping function from non-native accent speech to native-like speech. This approach enables both AC and pronunciation correction while maintaining fast inference due to its non-autoregressive architecture.

Although this model achieves fast inference, it still requires the entire input for prediction, making it unsuitable for streaming applications. We adopt this approach of using native TTS to generate ideal ground truth. Recognizing this model as one of the best for AC, we focus on architectural and training modifications to develop a streaming-capable AC framework while maintaining the performance of its non-streaming counterpart.

In this chapter, our work proposes the streaming AC by adapting and enhancing the non-streaming AC models in the previous chapter to operate in a streaming context. Our major contributions include significant architectural and training modifications that enable the model to process speech incrementally while maintaining high performance comparable to non-streaming models. Furthermore, we provide the implementation source code for streaming inference. This marks a substantial step forward in the field, as our streaming AC model is the first to effectively perform AC and pronunciation improvement in streaming fashion.

7.2. Methodology

This section outlines the step-by-step training procedure for our streaming AC model. We first train a native TTS model, which plays a crucial role in generating high-quality ground-truth data for non-native speech. Next, we describe the necessary architectural adjustments for streaming AC and explain the training process using synthetic ground-truth audio. Finally, we detail the inference mechanism for real-time speech conversion.

7.2.1. Training a Native TTS Model with Prosody and Speaker Preservation

Model Training

We employ the VITS framework Kim et al. (2021a) to train the native TTS model. VITS is a conditional variational autoencoder (CAVE) enhanced with normalizing flow, consisting of three primary components: a posterior encoder $q_\phi(z|x)$, a prior encoder $p_\theta(z|c)$, and a waveform generator $p_\psi(y|z)$, as illustrated in Figure 7.1.

To achieve precise text-audio alignment, VITS utilizes Monotonic Alignment Search (MAS). However, to accelerate training and bypass alignment learning, we leverage the Montreal Forced Aligner (MFA) McAuliffe et al. (2017) to upsample text representations to match the duration of the audio. The prior encoder conditions on both the upsampled transcript and the fundamental frequency (F0) sequence.

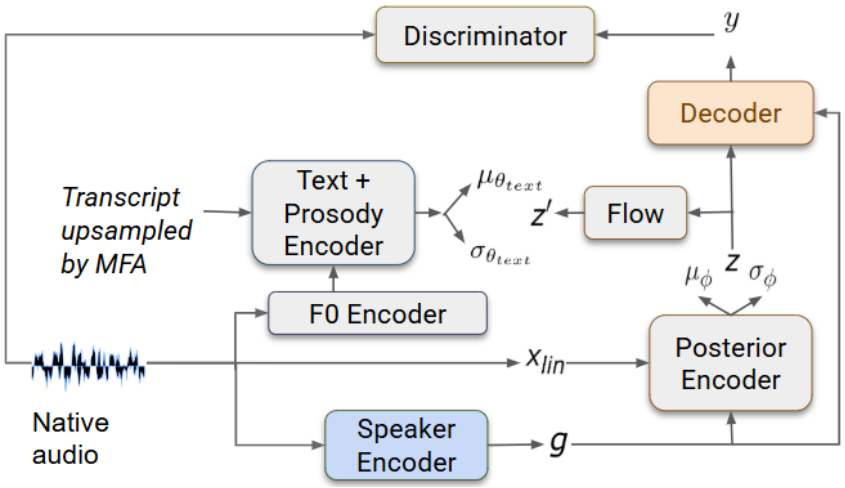

Figure 7.1.: Native TTS model preserving prosody and speaker identity.

The F0 encoder extracts frame-level F0 embeddings, which are fused with text embeddings before being processed by a Transformer-based encoder Vaswani et al. (2017c). This encoder, augmented with normalizing flow, generates the prior distribution $p_{\theta_{text}}(z|c)$ conditioned on text and prosody features. The posterior encoder takes linear spectrograms x_{lin} and speaker embeddings g to sample the latent variable $q_\phi(z|x)$, while the HiFi-GAN Kong et al. (2020a) decoder reconstructs the waveform y from z.

Mel-spectrograms, F0, and linear spectrograms are computed using a downsampling rate of 320, and the HiFi-GAN decoder applies a corresponding upsampling rate. The YIN algorithm Kroon (2022) extracts F0 sequences at the same downsampling rate, and the F0 encoder follows the approach in Wang et al. (2021b). We use a pre-trained speaker encoder from Liu et al. (2021b). Our training loss and hyperparameters are aligned with those in Kim et al. (2021a).

Generating Ground-Truth Audio from Native TTS

The trained native TTS model generates high-quality ground-truth audio for each non-native input. First, MFA aligns the phoneme transcript to the non-native audio and upsamples it to match the audio duration. Then, following the VITS inference process, a latent variable $z^{'}$ is sampled from $p_{\theta_{text}}(z^{'}|c_{upsample})$ and refined via an inverted normalizing flow f^{-1}. Finally, the HiFi-GAN decoder generates the synthetic ground-truth audio $\hat{y}_{\text{ground-truth}}$ using $p_\psi(x|f^{-1}(z^{'}),g,F0)$, where g and $F0$ are extracted from the original non-native speech.

In summary, the generated ground-truth audio $y_{\text{ground-truth}}$ retains the speaker's identity, prosody, and duration while ensuring native-like pronunciation. This process is formalized by the following equations:

$$A = MFA(x_{non-native}, c_{text}) \tag{7.1}$$

$$c_{upsample} = upsampling(c_{text}, A) \tag{7.2}$$

$$z^{'} \sim p_{\theta_{text}}(z^{'}|c_{upsample}, F_0) \tag{7.3}$$

$$y_{ground-truth} = Hifigan(f^{-1}(z^{'}), g, F0) \tag{7.4}$$

This native TTS-based ground-truth generation serves as a crucial component in training our streaming AC model, enabling efficient learning with high-quality training data.

7.2.2. Streaming Non-Autoregressive Model

Architecture

Conditional Variational Autoencoder (CVAE) architectures have been widely adopted in speech-to-speech tasks such as voice conversion (VC) Li et al. (2023) and accent conversion Nguyen et al. (2025). Unlike TTS, which uses text input for the prior encoder, AC and VC models take a content representation extracted from self-supervised speech models or ASR systems. While TTS involves a one-to-many mapping from text to audio, AC and VC map continuous content representations to audio in a one-to-one manner. Consequently, a CVAE structure is not strictly necessary.

To simplify training and inference for streaming AC, we adopt a conventional autoencoder architecture with four main components: a content encoder, a bottleneck extractor, a HiFi-GAN decoder, and a speaker encoder. Our content encoder extracts both linguistic content and frame-level prosody features, eliminating the need for a dedicated prosody encoder. This encoder is pre-trained and remains frozen during AC model training. The bottleneck extractor removes accents and refines pronunciation, while the HiFi-GAN decoder reconstructs native-like audio using the extracted content and speaker embedding.

To enable low-latency streaming inference, we replace the Transformer-based ASR model in the original AC framework with Emformer Hao et al. (2024), a streaming-optimized Transformer variant. Unlike conventional Transformers, which apply global self-attention to the entire sequence, Emformer divides the input into fixed-length segments and applies attention within a restricted local context. This ensures efficient computation while maintaining sufficient contextual information for accurate content extraction. Emformer processes the input sequence X by partitioning it into overlapping segments:

$$X = \{X_1, X_2, \ldots, X_T\}, \quad X_i \in \mathbb{R}^{L \times d} \tag{7.5}$$

where X_i is the i-th segment of length L, and d is the feature dimension.

Each segment is processed using an attention mechanism constrained to a left context of C_l segments and a right look-ahead of C_r segments:

$$Q_i, K_i, V_i = W_Q X_i, W_K X_{i-C_l:i+C_r}, W_V X_{i-C_l:i+C_r} \tag{7.6}$$

where W_Q, W_K, W_V are learned projection matrices for queries, keys, and values, respectively.

The attention scores are computed as:

$$A_i = \text{softmax}\left(\frac{Q_i K_i^T}{\sqrt{d_k}}\right) \tag{7.7}$$

where d_k is the key dimension. The output of the attention layer is then:

$$Z_i = A_i V_i \tag{7.8}$$

To maintain efficiency, Emformer incorporates relative positional encoding and a memory mechanism that caches past segment outputs, allowing the model to retain long-range dependencies without excessive computation. The final encoded representation is obtained by applying a feed-forward network (FFN) and residual connections:

$$H_i = \text{LayerNorm}\left(Z_i + X_i\right) \tag{7.9}$$

$$Y_i = \text{LayerNorm}\left(\text{FFN}(H_i) + H_i\right) \tag{7.10}$$

Our Emformer implementation consists of 24 layers, each with a hidden size of 1024, a left context of 30 frames, a right look-ahead of 8 frames, and a segment size of 4 frames. These settings strike a balance between latency and model accuracy, enabling real-time accent conversion with minimal delay. To ultilize advantage of self supervised learning representation, we initialize the weight of our Emformer by Wav2vec 2.0 weight.

Furthermore, we optimize the Torch Emformer implementation by replacing the standard multi-head attention with a Flex Attention function Dong et al. (2024), reducing computational overhead. This modification improves training efficiency by a factor of three and increases batch size by eight times compared to the original implementation.

The bottleneck extractor was originally based on a WaveNet van den Oord et al. (2016) with non-causal dilated convolutions, which performed well in prior implementations. Although the bottleneck extractor and HiFi-GAN decoder were not explicitly designed for streaming, their fully convolutional Waibel et al. (1989); Sugiyama et al. (1991); Waibel et al. (1987a,b) nature enables chunk-by-chunk processing, making them viable for real-time applications.
To enhance inference speed, we replace HiFi-GAN V1 with HiFi-GAN V2 and optimize the source code for efficient streaming inference across all components.

Training

The content encoder is trained on the Librispeech Panayotov et al. (2015) and L2Arctic datasets using CMU-dict phoneme labels. It is optimized using a combination of CTC and prosody losses:

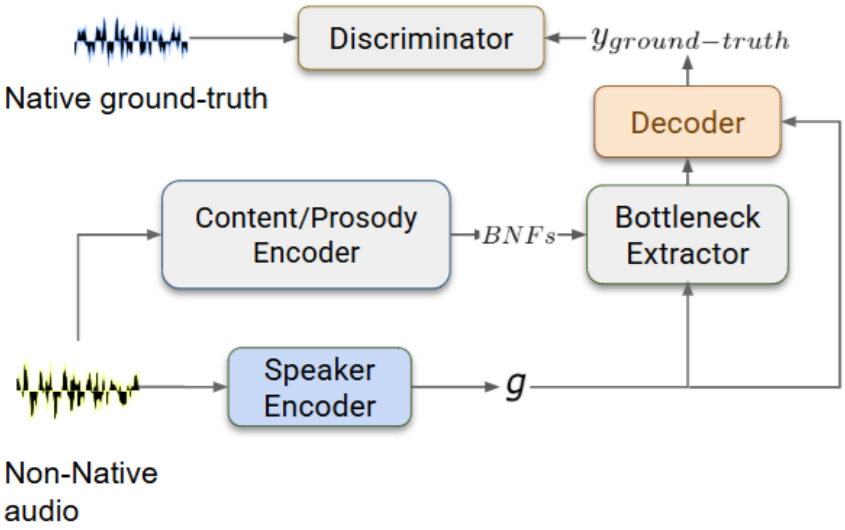

Figure 7.2.: Training the streaming Accent Conversion model with ideal ground-truth generation.

$$\mathscr{L}_{ASR} = (1-\alpha)\mathscr{L}_{ctc} + \alpha\mathscr{L}_{prosody} \tag{7.11}$$

where $\mathscr{L}_{prosody}$ is the L1 loss between predicted and ground truth log-F0 values, with an empirically determined weighting factor $\alpha = 0.2$.

Our Emformer implementation is optimized for efficiency. The original Torch Emformer exhibited 70% sparsity in its attention padding mask, leading to unnecessary computations. To mitigate this, we reimplemented it using the Flex Attention function Dong et al. (2024), reducing training time threefold and enabling an eightfold increase in batch size.

After training the content encoder, we pre-train the streaming AC model exclusively on native speech, ensuring it learns native speech characteristics. In the second stage, we fine-tune it on non-native inputs paired with native ground-truth outputs. We use a 3:1 ratio of non-native to native data to balance adaptation and retention of native speech properties. The HiFi-GAN loss, incorporating mel-spectrogram, feature, and discriminator losses, is used throughout both training stages.

Streaming Inference

During training, the model leverages future frames to predict accent-converted speech. However, for real-time inference, we use a chunk-based approach to maintain low latency while ensuring consistent output.

Due to the model's non-causal nature, it requires a look-ahead of 0.64s, corresponding to the right context window of the Emformer, HiFi-GAN, and WaveNet models. During inference, the input is processed in 0.08s chunks (matching Emformer's segment length). The system aims to minimize latency while maintaining smooth output, ensuring streaming and full-segment inference yield identical results.

Inference relies on a Voice Activity Detector (VAD) Sarfjoo et al. (2021) to determine speech segments. Once speech is detected, the system introduces an initial delay of 0.8s (10 chunks of 0.08s) before outputting the first two converted chunks and extracting the speaker embedding. This embedding remains fixed throughout the segment. Audio playback begins once the second chunk is available, ensuring smooth and uninterrupted streaming. The complete inference procedure is detailed in Algorithm 1.

Algorithm 1 Streaming AC Inference

```
0: Initialize: output_chunks ← [], input_chunks ← [], cache ← None
0: while start stream do
0:     chunk ← receive_new_chunk()
0:     input_chunks.append(chunk)
0:     if len(input_chunks) >= 10 then
0:         if cache == None then
0:             out, cache ← model.forward(input_chunks)
0:         else
0:             out, cache ← model.forward_with_cache(chunk, cache)
0:         end if
0:         output_chunks.append(out)
0:     end if
0:     if len(output_chunks) == 2 then
0:         player_thread.start_play()
0:     end if
0: end while=0
```

Our streaming framework maintains a system-wide real-time factor (RTF) below 1 for each chunk, typically achieving around 0.25 on a GTX 1060 6GB. This ensures that the output thread consistently stays at least two chunks ahead of the playback thread, maintaining a stable latency of approximately 0.8 seconds across various computational devices. This design enables smooth and efficient audio streaming. Although latency could be further reduced by decreasing the look-ahead receptive field, such optimization is beyond the scope of this work. Instead, we prioritize an efficient streaming architecture that preserves high audio quality.

Importantly, streaming inference is performed without any padding, ensuring output consistency with offline full-segment processing. After each chunk is processed, model caches—including those for the Emformer, WaveNet-based modules, and HiFi-GAN—are updated and stored, allowing seamless processing of subsequent chunks.

7.3. Experiment and Result

7.3.1. Dataset and Evaluation metrics

We use the same datasets and evaluation metrics as described in the previous chapter on non-streaming accent conversion. Specifically, we evaluate our model using the L2-ARCTIC corpus for non-native speech and the Librispeech dataset for native speech. The evaluation metrics include Word Error Rate

(WER), Mel-Cepstral Distortion (MCD), Mean Opinion Score (MOS), and Accent-MOS, as previously defined. This consistency ensures a fair comparison between the streaming and non-streaming models

7.3.2. Experimental setup

Our streaming AC model is based on the fine-tuned version in the second stage. To assess its performance, we compare it with the non-streaming model introduced in the previous chapter Nguyen et al. (2025).

Some causal streaming voice conversion models have been proposed Yang et al. (2024). However, in our accent conversion and pronunciation enhancement task, a look-ahead context is essential for accurate pronunciation improvement. To demonstrate this, we also train a causal streaming variant by replacing the non-causal CNN and transposed CNN of the Wavenet-based and Hifigan decoder with their causal counterparts. We analyze the audio quality of synthetic ground truth from the native TTS model. The training hyperparameters follow those of the original HiFi-GAN Kong et al. (2020a), with the system trained for 600,000 steps in the pre-training stage and 200,000 steps in the fine-tuning stage. In the both stages, the mel-spectrogram loss converges to approximately 0.3. Sample evaluation audios are available in [1].

7.3.3. Result

The objective metric evaluation in Table 7.2 shows that the streaming model performs comparably to the non-streaming model. Surprisingly, MOSNet assigns a higher score to our accent conversion output than to the original audio. This suggests that pre-trained MOSNet, which is primarily trained on non-accented speech, may give lower scores to non-native speech, even when it is real. A MOSNet score of approximately 4.1 indicates that our generated audio maintains good quality in terms of naturalness. Additionally, SECS remains stable at 0.85 and 0.84, verifying that the speaker identity is well-preserved in both settings. The ACC results are weaker compared to other metrics. Since our model aims to retain the original audio's prosody, which can sometimes reflect the speaker's accent, it occasionally classifies the converted audio as non-native. This prosody preservation may, in some cases, reduce the effectiveness of accent conversion. The causal streaming model performs worst across all metrics, highlighting the importance of look-ahead context. Subjective evaluation (Table 7.1) shows similar scores for streaming and non-streaming models, while strong WER performance indicates high-quality synthetic ground truth.

7.4. Conclusion

This chapter presents the first streaming accent conversion model, demonstrating its ability to synthesize high-quality audio with native-like pronunciation in a real-time setting. Our results show that the streaming model achieves performance comparable to its non-streaming counterpart. We introduced an

[1] `https://accentconversion.github.io/streaming_demo`

Table 7.1.: Subjective metrics

Models	Nativeness	Sim-MOS
Original	1.12 ± 0.15	-
Synthetic ground-truth	3.93 ± 0.15	4.02 ± 0.17
Causal streaming Model	1.23 ± 0.12	3.75 ± 0.12
Streaming model	3.78 ± 0.18	3.92 ± 0.18
Non-Streaming model	3.87 ± 0.20	3.96 ± 0.19

Table 7.2.: Objective metrics

Models	WER	ACC	SECS	MOSNet
Original	18.3	98.3	1.0	3.97
Synthetic ground-truth	4.7	11.5	0.83	4.18
Causal streaming Model	33.5	25.1	0.75	3.01
Streaming model	14.1	16.3	0.85	4.12
Non Streaming model	14.3	17.0	0.84	4.08

effective inference strategy tailored for non-autoregressive accent conversion in a streaming context. Although the current end-to-end latency remains at 0.8 seconds, this can be further reduced by minimizing the model's look-ahead receptive field. Future work will focus on lowering latency while preserving audio quality, as well as optimizing inference efficiency for deployment on low-power devices such as CPUs.

8. Comparison and Evaluation in Real-World Scenarios

This chapter presents a systematic comparison between two main approaches for this thesis: direct mapping approaches, inspired by speech-to-speech translation (presented in chapter 3 and 5), and disentangle–resynthesis approaches, which decompose speech into intermediate representations before resynthesis (presented in chapter 6 and 7). Although both paradigms have shown promising results, their generalization ability and scalability under different data conditions remain underexplored. In this chapter, we focus on two critical evaluation that are highly relevant for real-world accent conversion systems.

Generalization to Out-of-Domain Accents

A fundamental challenge in accent conversion is the vast diversity of accents worldwide. It is infeasible for any model to observe all possible accents during training. Therefore, out-of-domain evaluation, where models are tested on accents unseen during training, is a crucial indicator of robustness and practical applicability.

In our evaluation, we compare the two approaches on unseen accents, i.e., accents that do not appear in the training data. This setting simulates real-world deployment scenarios, where systems must handle speakers with novel or underrepresented accent characteristics.

Scalability with Increasing Training Data

Another key aspect of accent conversion systems is how effectively they leverage additional training data. Most prior studies, including earlier chapters of this thesis, primarily rely on the L2-ARCTIC dataset, which, while valuable, is limited in both size and accent diversity. We invested substantial effort in collecting and recording a large-scale accented speech corpus that covers a wide range of accents and speakers from different regions of the world. This expanded dataset enables a more comprehensive evaluation of accent conversion models under realistic and diverse conditions, and allows us to systematically study how different approaches benefit from increased data volume and accent variability.

In this chapter, we present how the accent conversion system of this thesis improved with additional data. Through the external effort of Interactive AI, LLC, Large database of accented speech was collected and the data was made available for research purpose and this thesis research. The database comprised approximately 3,000 hours of speech data. The dataset includes speakers with a wide range of non-native accents, such as Indian, Japanese, Chinese, Vietnamese, French, and Filipino accents, among others. The recordings cover diverse speakers, speaking styles, and acoustic conditions, making the corpus substantially more varied than commonly used benchmark datasets. This large and diverse dataset

allows us to move beyond small-scale, controlled experimental settings and to evaluate accent conversion systems under more realistic conditions.

To evaluate the generalization ability of accent conversion models to out-of-domain accents, we conduct a controlled cross-accent experiment using the L2-ARCTIC dataset. L2-ARCTIC contains speech data from six distinct non-native accents. In each experimental run, we randomly select four accents for training and reserve the remaining two accents as unseen test accents.

The accent conversion model is trained exclusively on speech from the selected four accents and is then evaluated on the two held-out accents, which are not observed during training. This setup directly measures the model's ability to generalize beyond the accent distributions seen during training and to handle accent characteristics that differ from those it was optimized for.

To reduce the bias introduced by a particular accent split and to ensure statistical robustness, we repeat this random selection process ten times, each time using a different combination of four training accents and two test accents. Performance metrics are then averaged across the ten runs. This repeated random split strategy provides a more reliable estimate of out-of-domain generalization performance and allows us to assess how consistently each accent conversion approach performs when confronted with previously unseen accents.

8.1. Generalization to Out-of-Domain Accents

8.1.1. Experiment

To evaluate the generalization ability of accent conversion models to out-of-domain accents, we conduct a controlled cross-accent experiment using the L2-ARCTIC dataset. L2-ARCTIC contains speech data from six distinct non-native accents. In each experimental run, we randomly select four accents for training and reserve the remaining two accents as unseen test accents.

The accent conversion model is trained exclusively on speech from the selected four accents and is then evaluated on the two held-out accents, which are not observed during training. This setup directly measures the model's ability to generalize beyond the accent distributions seen during training and to handle accent characteristics that differ from those it was optimized for.

To reduce the bias introduced by a particular accent split and to ensure statistical robustness, we repeat this random selection process ten times, each time using a different combination of four training accents and two test accents. Performance metrics are then averaged across the ten runs. This repeated random split strategy provides a more reliable estimate of out-of-domain generalization performance and allows us to assess how consistently each accent conversion approach performs when confronted with previously unseen accents.

For this evaluation, we select the best-performing model from each of the two proposed approaches introduced in Chapter 3 and Chapter 5, respectively. These models represent the strongest configurations within their corresponding frameworks and are therefore suitable for a fair and meaningful comparison.

To assess their performance, we modify the evaluation protocol used in the previous chapters. Specifically, we focus on quantifying the improvement of both objective and subjective evaluation metrics in percentage when the models are trained with more data. The objective metrics include word error rate (WER), accent recognition accuracy (Acc), and speaker similarity, measured using cosine similarity in the speaker embedding space. In addition, we conduct subjective listening tests to evaluate perceptual quality, including accentedness mean opinion scores (MOS) and speaker similarity MOS.
Using both objective and subjective metrics allows us to comprehensively analyze the models' performance in terms of intelligibility, accent reduction, speaker identity preservation, and perceived naturalness, particularly in the challenging out-of-domain accent setting.

8.1.2. Result

Overall, both approaches are able to improve the accent characteristics of speech in out-of-domain test conditions. However, the subjective evaluation 8.1shows that the mapping approach achieves substantially higher perceived nativeness. This can be attributed to the inherent properties of the sequence-to-sequence architecture, in which the decoder is trained to generate speech patterns that closely resemble native speech. In contrast, the disentangle-and-resynthesis approach attempts to remove accent-related information through an ASR-based encoder. In out-of-domain scenarios, this disentanglement is imperfect, leading to residual accent leakage and reduced nativeness in the converted speech.
With respect to naturalness and speaker similarity, the disentangle-and-resynthesis approach performs better because it explicitly preserves the original prosody and speaker-related characteristics of the input speech. In contrast, the mapping approach discards most of the original prosodic and speaker information during the encoding process. As a result, although it achieves higher accent nativeness, its output speech exhibits reduced naturalness and weaker speaker similarity compared to the disentangle-and-resynthesis approach.

Table 8.1.: Subjective metrics

Models	Nativeness	MOS	Sim-MOS
Original	1.67 ± 0.05	-	-
Mapping approach	4.01 ± 0.08	3.7	3.75 ± 0.19
Disentanglement-Resynthesis approach	3.57 ± 0.08	4.05	4.06 ± 0.19

Table 8.2.: Objective metrics

Models	WER	ACC	SECS	MOSNet
Original	20.3	98.3	1.0	3.97
Mapping approach	16.1	13.3	0.85	4.04
Disentanglement-Resynthesis approach	15.7	27.0	0.88	4.18

In terms of objective evaluation, the mapping approach exhibits a higher word error rate (WER). This is because the translation framework must first interpret the non-native input speech before generating native-like output. In out-of-domain conditions, mismatches between training and test accents can lead the model to misrecognize the linguistic content, causing errors in the generated speech, even when the output sounds fluent and native-like.

Consistent with the nativeness results, the mapping approach achieves higher accent classifier accuracy, indicating that it produces outputs that are more consistent with the target accent in out-of-domain conditions. In contrast, both SECS and MOSNet scores align with the speaker similarity and naturalness evaluations, showing that the disentangle-and-resynthesis approach better preserves speaker identity and overall speech quality compared to the mapping approach.

8.2. Scalability with Increasing Training Data

Through the efforts of Interactive AI, LLC, a large database of accented speech was collected and made available for research purposes. This database, comprising approximately 3,000 hours of speech, was used in this thesis research. The corpus covers a broad range of non-native accents, including but not limited to Indian, Japanese, Chinese, Vietnamese, French, and Filipino accents, and includes speakers from diverse linguistic backgrounds, age groups, and recording conditions. Compared to commonly used benchmark datasets such as L2-ARCTIC, this corpus provides substantially greater diversity in both accent characteristics and speaker variability.

Building on this dataset, we conduct our experiments using the full accented speech corpus, leveraging all available training data. By training the accent conversion models on the complete 3,000-hour corpus, we focus on evaluating their scalability and robustness in a large-scale, real-world setting. This setup reflects practical deployment conditions, where maximizing data utilization is often preferable to operating under constrained data regimes.

Through this experiment, we aim to examine how accent conversion models behave when trained in a large-scale data regime and to compare how different approaches exploit extensive accented speech data. Evaluation is conducted using the same objective and subjective metrics described in previous sections to ensure consistency and comparability across experimental results.

8.2.1. Data collection

The dataset consists of English speech from speakers with diverse non-native accents, including Indian, Korean, Chinese, Arabic, Filipino, Japanese, French, and Vietnamese accents. For the initial release, we recruited a large number of speakers for each accent group to ensure sufficient intra-accent variability. Speakers were recruited from different regions around the world, with ages ranging from 18 to 43 years. All participants demonstrated at least a basic level of English proficiency. Participants were allowed to record speech using their own devices, including laptop microphones and smartphone microphones (e.g.,

iPhones), resulting in recordings collected under a wide range of acoustic conditions. This setup reflects realistic recording environments, such as home offices and personal rooms, and introduces natural variability in channel characteristics, background noise, and microphone quality. Such diversity is important for building accent conversion models that are robust to real-world deployment scenarios rather than being limited to studio-quality speech. All speakers were instructed to speak in a natural manner. The recorded speech was sampled at 24kHz and stored in WAV format.

To construct the corpus, we used the full set of sentence prompts from the LibriSpeech dataset. This choice was motivated by several factors. First, the LibriSpeech prompts are open source and can be used to produce approximately one hour of curated speech per speaker. Second, the LibriSpeech corpus has been widely adopted and has demonstrated strong performance in speech synthesis and voice conversion tasks, making it a reliable source of linguistically rich and phonetically balanced material.

Because recordings were collected in uncontrolled environments with a high degree of freedom in recording devices and acoustic conditions, a rigorous data filtering process was required to ensure transcription accuracy and overall data quality. We therefore employed two independently developed automatic speech recognition systems—Whisper and NVIDIA NeMo ASR—to verify the consistency between each audio sample and its corresponding LibriSpeech prompt.

For each utterance, we decoded the audio using several independently trained ASR systems and compared the recognized text with the original prompt. Utterances for which the ASR outputs showed strong agreement with the prompt were retained, while samples with large mismatches—indicating misread, skipped, or incorrectly spoken sentences—were discarded. This multi-ASR consensus strategy significantly reduces the inclusion of incorrectly labeled or poorly articulated recordings.

In addition to transcript verification, we also applied signal-level quality checks, including silence detection, duration thresholds, and basic signal-to-noise ratio filtering, to remove recordings with excessive noise, clipping, or abnormal speaking rate. Together, these filtering steps yield a high-quality accented speech corpus suitable for training robust accent conversion. Overall, a total of 1,352 accented speech hours were collected from 108 accented speakers.[1]

8.2.2. Experiment

To examine the performance of the proposed models when trained with significantly more data, we select the best-performing model from each of the two approaches introduced in Chapter 3 and Chapter 5. Both models are retrained using the newly collected accented speech data combined with the L2-ARCTIC training dataset, excluding a small subset reserved for testing. This allows us to study how the additional training data impacts their ability to convert accents.

For evaluation, we use a combined test set consisting of the original L2-ARCTIC dataset and the reserved small subset of newly collected accented speech. In this subset, one speaker is randomly selected for each

[1] MAcciDat. This corpus was collected externally by Datoid LLC to support this research. For licensing inquiries, please contact the Interactive AI LLC, Alexander Waibel (alex@waibel.com, alex@interactive-ai.com).

Table 8.3.: Statistics of the accented speech corpus

Accent	**Number of Speakers**	**Number of Hours**
Indian	29	421
Chinese	14	150
Arabic	5	5
Filipino	12	101
Japanese	17	297
French	10	97
Vietnamese	21	282
Total	108	1353

accent, and 100 audio samples are extracted from each chosen speaker. This setup enables us to assess the models' generalization to unseen speakers.

All models are evaluated on this combined dataset, and performance metrics are reported to quantify the improvements resulting from training with the additional accented speech data.

To assess their performance, we modify the evaluation protocol used in the previous chapters. Specifically, we focus on quantifying the improvement of both objective and subjective evaluation metrics in percentage when the models are trained with more data. The objective metrics include word error rate (WER), accent recognition accuracy (Acc), and speaker similarity, measured using cosine similarity in the speaker embedding space. In addition, we conduct subjective listening tests to evaluate perceptual quality, including accentedness mean opinion scores (MOS) and speaker similarity MOS.

8.2.3. Result

Table 8.4.: Subjective metrics

Models	Nativeness	Sim-MOS
Mapping approach		
Public dataset	3.85	4.01
MaAcciDat	4.05	4.14
Relative improvement	+5.7%	+3.3%
Disentanglement-Resynthesis approach		
Public dataset	3.65	3.75
MaAcciDat	4.01	4.22
Relative improvement	+10.1%	+12.6%

According to the results summarized in Table 8.5, increasing the amount of training data leads to consistent performance improvements across all evaluation metrics. In particular, models trained with substantially more data show clear gains in intelligibility, speech naturalness, and speaker similarity. This trend indicates that the proposed accent conversion frameworks are able to effectively leverage additional data, resulting in more fluent and perceptually native converted speech while better preserving the original speaker identity. In terms of intelligibility, the mapping approach achieves larger improvements than the

Table 8.5.: Objective metrics

Models	WER	ACC	SECS
Mapping approach			
Public dataset	15.1	13.3	0.85
MaAcciDat	10.7	13.01	0.855
Relative improvement	30%	2%	5.1 %
Disentanglement-Resynthesis approach			
Public dataset	12.4	18	0.88 %
MaAcciDat	10.6	17.56%	0.883 %
Relative improvement	15.7%	3%	5.2 %

disentangle–resynthesis approach (30% vs 15-7% in term of WER), largely because it relies on a substantially larger model with stronger sequence modeling capacity. This enables more accurate modeling of linguistic content and pronunciation, which directly contributes to improved intelligibility, especially when sufficient training data are available.

There is no significant improvement in terms of nativeness for the mapping approach, as it already achieves high nativeness even with limited training data. In contrast, the disentangle–resynthesis approach benefits substantially from additional training data: as more data become available, the model becomes better at disentangling non-native accent characteristics and resynthesizing speech that sounds more native. With respect to speaker similarity, both approaches benefit similarly from additional training data, showing no significant difference in their relative gains.

8.2.4. Conclusion

This chapter presented a comprehensive comparison of two major accent conversion paradigms—direct mapping based on speech-to-speech translation and the disentangle–resynthesis framework—under realistic and challenging evaluation conditions. In particular, we focused on two critical dimensions for practical deployment: generalization to out-of-domain accents and scalability with increasing amounts of training data.

The experimental results demonstrate that the mapping approach benefits significantly from large-scale training data, showing notable improvements in intelligibility when exposed to diverse accented speech. Its strong modeling capacity enables effective learning of pronunciation patterns across accents; however, its performance gain saturates more quickly in terms of speech naturalness, which is already high even under limited data conditions.

In contrast, the disentangle–resynthesis approach exhibits stronger improvements in naturalness and accent correction as more training data becomes available. With increased data diversity, the model better separates accent-related factors from content and speaker identity, resulting in more native-like pronunciation while maintaining prosodic and speaker characteristics. This behavior highlights the advantage of explicit factorization when scaling to broader accent coverage.

Both approaches demonstrate comparable gains in speaker similarity, indicating that speaker identity can be preserved effectively across paradigms when sufficient training data is provided. Importantly, the out-of-domain evaluations reveal that disentangle–resynthesis models tend to generalize more robustly to unseen accents, while direct mapping approaches remain sensitive to accent distributions observed during training.

Overall, the findings of this chapter emphasize that no single approach universally dominates across all evaluation criteria. Instead, the choice between direct mapping and disentangle–resynthesis should be guided by application constraints, available training data, and deployment requirements. These insights provide valuable guidance for designing scalable and robust accent conversion systems and motivate future research toward hybrid architectures that combine the strengths of both paradigms.

9. Conclusion and future works

This thesis presents a comprehensive exploration of accent conversion (AC), addressing its core challenges through a series of innovative models and frameworks designed to improve data availability, conversion quality, and real-time applicability. We approached AC from both data generation and model architecture perspectives, demonstrating that leveraging voice conversion (VC) and text-to-speech (TTS) systems can significantly enhance AC performance, especially in low-resource conditions.

In the first part of this work, we proposed a novel data augmentation using VC models to synthesize high-quality parallel accent data. The integration of self-supervised learning via pretrained Wav2vec encoders proved essential in improving performance under limited data conditions. We then introduced SYNTACC—a lightweight, using weight factorization for TTS—which facilitates accent control and can be effectively trained even with limited accented traning data.

Following this, we demonstrated how controllable accented TTS systems can be used to generate large-scale synthetic datasets to train many-to-one AC models. By leveraging native speech discrete units and pretrained encoder-decoder frameworks, we showed that it is possible to improve both accent accuracy and naturalness in non-native speech transformation tasks.

To overcome the limitations of the mapping-based AC approach, we investigated a disentangle-resynthesis framework using VITS. By introducing a two-stage training method and leveraging both generated ground-truth and teacher-student distillation from native TTS, we achieved accurate pronunciation correction while preserving speaker identity, emotion and phonetic alignment.

Finally, we presented the first streaming accent conversion system, which enables real-time inference while maintaining competitive performance with non-streaming models. Our streaming method introduced a non-autoregressive architecture and an efficient inference mechanism tailored for online applications. Although the current latency of 0.8 seconds is not yet optimal, it lays the foundation for future research in low-latency AC systems suitable for deployment on edge devices.

After completing five research papers and reflecting on them collectively, We realize that a central contribution of this thesis is the introduction of flexible frameworks and concepts that can be applied across a wide range of architectures for accent conversion. For example, rather than relying solely on conventional sequence-to-sequence models for direct mapping, we propose exploring the use of decoder-only transformers translate discrete speech units from non-native to native variants. In the disentangle-resynthesis

paradigm, we suggest replacing traditional variational autoencoders with more expressive generative models, such as diffusion-based flow matching Lou et al. (2024), to more effectively synthesize native-like speech from accent-independent features. Moreover, instead of relying on fixed speaker embeddings, future work can explore leveraging short speech prompts extracted directly from the input audio to more effectively preserve speaker identity during conversion. This approach can overcome the limitations of traditional speaker embeddings, as short speech prompts provide more detailed about the speaker's voice characteristics, leading to more accurate identity preservation Liu (2024). The speech recognition transducer model Zhou et al. (2022) can be considered a promising new direction for the content encoder architecture, as it effectively combines linguistic, acoustic, and alignment information into a unified representation, enabling streaming. However, its performance in accent conversion needs to be evaluated in future work. Finally, optimizing for real-time streaming accent conversion on edge devices or CPUs is an important goal for future improvements.

In conclusion, this thesis contributes multiple practical advancements to the field of accent conversion. From data augmentation to architecture design and streaming deployment, each of these areas tackles a crucial challenge in accent conversion. Future work will focus on further reducing latency, enhancing robustness across more diverse accents.

A. Bibliography

Ardila, R., M. Branson, K. Davis, M. Henretty, M. Kohler, J. Meyer, R. Morais, L. Saunders, F. M. Tyers, and G. Weber, 2020: Common voice: A massively-multilingual speech corpus. URL https://arxiv.org/abs/1912.06670, 1912.06670.

Baevski, A., H. Zhou, A. Mohamed, and M. Auli, 2020a: wav2vec 2.0: A framework for self-supervised learning of speech representations. *CoRR*, **abs/2006.11477**, URL https://arxiv.org/abs/2006.11477, 2006.11477.

———, 2020b: wav2vec 2.0: A framework for self-supervised learning of speech representations. 2006.11477.

Baevski, A., Y. Zhou, A. Mohamed, and M. Auli, 2020c: wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in Neural Information Processing Systems*, Larochelle, H., M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., Curran Associates, Inc., Vol. 33, 12 449–12 460.

Casanova, E., J. Weber, C. Shulby, A. C. Júnior, E. Gölge, and M. A. Ponti, 2021: Yourtts: Towards zero-shot multi-speaker TTS and zero-shot voice conversion for everyone. *CoRR*, **abs/2112.02418**, URL https://arxiv.org/abs/2112.02418, 2112.02418.

Casanova, E., J. Weber, C. D. Shulby, A. C. Junior, E. Gölge, and M. A. Ponti, 2022: YourTTS: Towards zero-shot multi-speaker TTS and zero-shot voice conversion for everyone. *Proceedings of the 39th International Conference on Machine Learning*, Chaudhuri, K., S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, Eds., PMLR, Proceedings of Machine Learning Research, Vol. 162, 2709–2720, URL https://proceedings.mlr.press/v162/casanova22a.html.

Chan, W., N. Jaitly, Q. Le, and O. Vinyals, 2016: Listen, attend and spell. *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 4960–4964.

Chang, X., J. Shi, J. Tian, Y. Wu, Y. Tang, Y. Wu, S. Watanabe, Y. Adi, X. Chen, and Q. Jin, 2024: The interspeech 2024 challenge on speech processing using discrete units. URL https://arxiv.org/abs/2406.07725, 2406.07725.

Chen, S., C. Wang, Z. Chen, Y. Wu, S. Liu, Z. Chen, J. Li, N. Kanda, T. Yoshioka, X. Xiao, J. Wu, L. Zhou, S. Ren, Y. Qian, Y. Qian, J. Wu, M. Zeng, X. Yu, and F. Wei, 2022: Wavlm: Large-scale

self-supervised pre-training for full stack speech processing. *IEEE Journal of Selected Topics in Signal Processing*, **16 (6)**, 1505–1518.

Chen, X., J. Pei, L. Xue, and M. Zhang, 2024: Transfer the linguistic representations from tts to accent conversion with non-parallel data. *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 12 501–12 505.

chieh Chou, J., C. chieh Yeh, and H. yi Lee, 2019: One-shot voice conversion by separating speaker and content representations with instance normalization. URL `https://arxiv.org/abs/1904.05742`, `1904.05742`.

Churchwell, C., M. Morrison, and B. Pardo, 2024: High-fidelity neural phonetic posteriorgrams. URL `https://arxiv.org/abs/2402.17735`, `2402.17735`.

de Cheveigné, A. and H. Kawahara, 2002: Yin, a fundamental frequency estimator for speech and music. *J Acoust Soc Am*, **111**, 1917–1930, URL `http://audition.ens.fr/adc/pdf/2002_JASA_YIN.pdf`.

Dong, J., B. Feng, D. Guessous, Y. Liang, and H. He, 2024: Flex attention: A programming model for generating optimized attention kernels. URL `https://arxiv.org/abs/2412.05496`, `2412.05496`.

Défossez, A., J. Copet, G. Synnaeve, and Y. Adi, 2022: High fidelity neural audio compression. URL `https://arxiv.org/abs/2210.13438`, `2210.13438`.

Goodfellow, I. J., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, 2014: Generative adversarial networks. URL `https://arxiv.org/abs/1406.2661`, `1406.2661`.

Graves, A., S. Fernández, F. Gomez, and J. Schmidhuber, 2006: Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. *Proceedings of the 23rd International Conference on Machine Learning (ICML)*, ACM, 369–376.

Guo, J., N. Xu, K. Qian, Y. Shi, K. Xu, Y. Wu, and A. Alwan, 2018: Deep neural network based i-vector mapping for speaker verification using short utterances. URL `https://arxiv.org/abs/1810.07309`, `1810.07309`.

Gölge, E., E. Casanova, A. Korolev, T. Werkmeister, WeberJulian, T. Müller, R. Morais, K. Guiller, B. Gerazov, T. Hellweg, A. Chaurasia, J. Thalheim, N. Stoker, K. Iida, N. Müller, R. (), A. Pujols, M. Hansen, bgerazov, mittimithai, A. Hilmkil, M. Toman, geneing, G. Elsmore-Paddock, M. Weinelt, Q. Hou, jyegerlehner, a froghyar, and Anand..., 2021: coqui-ai/tts: v0.2.1. URL `https://doi.org/10.5281/zenodo.5343925`.

Ha, T.-L., J. Niehues, and A. Waibel, 2016: Toward multilingual neural machine translation with universal encoder and decoder. *Proceedings of the 13th International Conference on Spoken Language Translation*.

Hao, Z., H. Quan, and Y. Lu, 2024: EMF-former: An Efficient and Memory-Friendly Transformer for Medical Image Segmentation . *proceedings of Medical Image Computing and Computer Assisted Intervention – MICCAI 2024*, Springer Nature Switzerland, Vol. LNCS 15008.

Hochreiter, S. and J. Schmidhuber, 1997: Long short-term memory. *Neural Comput.*, **9 (8)**, 1735–1780, URL `https://doi.org/10.1162/neco.1997.9.8.1735`.

Hsu, W., B. Bolte, Y. H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed, 2021a: Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *CoRR*, **abs/2106.07447**, URL `https://arxiv.org/abs/2106.07447`, `2106.07447`.

Hsu, W.-N., B. Bolte, Y.-H. H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed, 2021b: Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, **29**, 3451–3460, URL `https://doi.org/10.1109/TASLP.2021.3122291`.

Huber, C., T. A. Dinh, C. Mullov, N.-Q. Pham, T.-B. Nguyen, F. Retkowski, S. Constantin, E. Ugan, D. Liu, Z. Li, et al., 2023: End-to-end evaluation for low-latency simultaneous speech translation. *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 12–20.

Ito, K., 2017: The lj speech dataset. `https://keithito.com/LJ-Speech-Dataset/`.

Jia, D., Q. Tian, K. Peng, J. Li, Y. Chen, M. Ma, Y. Wang, and Y. Wang, 2023: Zero-shot accent conversion using pseudo siamese disentanglement network. `2212.05751`.

Jin, M., P. Serai, J. Wu, A. Tjandra, V. Manohar, and Q. He, 2023a: Voice-preserving zero-shot multiple accent conversion. *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1–5.

———, 2023b: Voice-preserving zero-shot multiple accent conversion. *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1–5.

Kaneko, T., H. Kameoka, K. Tanaka, and N. Hojo, 2019: Cyclegan-vc2: Improved cyclegan-based non-parallel voice conversion. URL `https://arxiv.org/abs/1904.04631`, `1904.04631`.

Kasi, K. and S. A. Zahorian, 2002: Yet another algorithm for pitch tracking. *2002 IEEE International Conference on Acoustics, Speech, and Signal Processing*, Vol. 1, I–361–I–364.

Kim, J., S. Kim, J. Kong, and S. Yoon, 2020: Glow-tts: A generative flow for text-to-speech via monotonic alignment search. *Proceedings of the 34th International Conference on Neural Information Processing Systems*, Curran Associates Inc., Red Hook, NY, USA, NIPS'20.

Kim, J., J. Kong, and J. Son, 2021a: Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech. *International Conference on Machifi-ne Learning*.

———, 2021b: Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech. *Proceedings of the 38th International Conference on Machine Learning*, Meila, M. and T. Zhang, Eds., PMLR, Proceedings of Machine Learning Research, Vol. 139, 5530–5540, URL `https://proceedings.mlr.press/v139/kim21f.html`.

Kingma, D. P. and M. Welling, 2019: An introduction to variational autoencoders. *CoRR*, **abs/1906.02691**, URL `http://arxiv.org/abs/1906.02691`, `1906.02691`.

Koizumi, Y., H. Zen, S. Karita, Y. Ding, K. Yatabe, N. Morioka, M. Bacchiani, Y. Zhang, W. Han, and A. Bapna, 2023: Libritts-r: A restored multi-speaker text-to-speech corpus. `2305.18802`.

Kolluru, B., V. Wan, J. Latorre, K. Yanagisawa, and M. J. F. Gales, 2014: Generating multiple-accent pronunciations for TTS using joint sequence model interpolation. *Proc. Interspeech 2014*, 1273–1277.

Kominek, J. and A. Black, 2004a: The cmu arctic speech databases. *SSW5-2004*.

Kominek, J. and A. W. Black, 2004b: The cmu arctic speech databases. *5th ISCA Workshop on Speech Synthesis (SSW 5)*, 223–224.

Kong, J., J. Kim, and J. Bae, 2020a: Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis. *Advances in Neural Information Processing Systems*, Larochelle, H., M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., Curran Associates, Inc., Vol. 33, 17 022–17 033.

———, 2020b: Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis. *Advances in Neural Information Processing Systems*, Larochelle, H., M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., Curran Associates, Inc., Vol. 33, 17 022–17 033, URL `https://proceedings.neurips.cc/paper/2020/file/c5d736809766d46260d816d8dbc9eb44-Paper.pdf`.

Kroon, A., 2022: Comparing conventional pitch detection algorithms with a neural network approach. URL `https://arxiv.org/abs/2206.14357`, `2206.14357`.

Lakhotia, K., E. Kharitonov, W.-N. Hsu, Y. Adi, A. Polyak, B. Bolte, T.-A. Nguyen, J. Copet, A. Baevski, A. Mohamed, and E. Dupoux, 2021: Generative spoken language modeling from raw audio. `2102.01192`.

Lee, A., P. Chen, C. Wang, J. Gu, X. Ma, A. Polyak, Y. Adi, Q. He, Y. Tang, J. M. Pino, and W. Hsu, 2021: Direct speech-to-speech translation with discrete units. *CoRR*, **abs/2107.05604**, URL https://arxiv.org/abs/2107.05604, 2107.05604.

Lewis, M., Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, 2019: BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *CoRR*, **abs/1910.13461**, URL http://arxiv.org/abs/1910.13461, 1910.13461.

Li, J., W. Tu, and L. Xiao, 2023: Freevc: Towards hifigagh-quality text-free one-shot voice conversion. *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1–5.

Li, L., D. Wang, Z. Zhang, and T. F. Zheng, 2015: Deep speaker vectors for semi text-independent speaker verification. URL https://arxiv.org/abs/1505.06427, 1505.06427.

Lin, Y. Y., C.-M. Chien, J.-H. Lin, H. yi Lee, and L. shan Lee, 2021: Fragmentvc: Any-to-any voice conversion by end-to-end extracting and fusing fine-grained voice fragments with attention. 2010.14150.

Liu, S., 2024: Zero-shot voice conversion with diffusion transformers. URL https://arxiv.org/abs/2411.09943, 2411.09943.

Liu, S., Y. Cao, D. Wang, X. Wu, X. Liu, and H. Meng, 2021a: Any-to-many voice conversion with location-relative sequence-to-sequence modeling. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, **29**, 1717–1728, URL http://dx.doi.org/10.1109/TASLP.2021.3076867.

———, 2021b: Any-to-many voice conversion with location-relative sequence-to-sequence modeling. *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, **29**, 1717–1728, URL https://doi.org/10.1109/TASLP.2021.3076867.

Liu, S., D. Wang, Y. Cao, L. Sun, X. Wu, S. Kang, Z. Wu, X. Liu, D. Su, D. Yu, and H. Meng, 2020a: End-to-end accent conversion without using native utterances. *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 6289–6293.

Liu, Y., J. Gu, N. Goyal, X. Li, S. Edunov, M. Ghazvininejad, M. Lewis, and L. Zettlemoyer, 2020b: Multilingual denoising pre-training for neural machine translation. *CoRR*, **abs/2001.08210**, URL https://arxiv.org/abs/2001.08210, 2001.08210.

Lou, H., H. Paik, P. D. Haghighi, W. Hu, and L. Yao, 2024: Latentspeech: Latent diffusion for text-to-speech generation. URL https://arxiv.org/abs/2412.08117, 2412.08117.

Ma, D., W.-C. Huang, and T. Toda, 2021: Investigation of text-to-speech-based synthetic parallel data for sequence-to-sequence non-parallel voice conversion. *2021 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, 870–877.

McAuliffe, M., M. Socolof, S. Mihuc, M. Wagner, and M. Sonderegger, 2017: Montreal forced aligner: Trainable text-speech alignment using kaldi. *Interspeech 2017*, 498–502.

Mehta, S., E. Szekely, J. Beskow, and G. E. Henter, 2022: Neural hmms are all you need (for high-quality attention-free tts). *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 7457–7461, URL `http://dx.doi.org/10.1109/ICASSP43922.2022.9746686`.

Morise, M., 2016: D4c, a band-aperiodicity estimator for high-quality speech synthesis. *Speech Communication*, **84**, 57–65, URL `https://www.sciencedirect.com/science/article/pii/S0167639316300413`.

Ngoc-Quan, P., 2019: GitHub - quanpn90/NMTGMinor: A Neural Machine Translation toolkit for research purpose — github.com. `https://github.com/quanpn90/NMTGMinor`.

Nguyen, T. N., S. Akti, N. Q. Pham, and A. Waibel, 2025: Improving pronunciation and accent conversion through knowledge distillation and synthetic ground-truth from native tts. *ICASSP 2025 - 2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1–5.

Nguyen, T. N., T. S. Nguyen, C. Huber, N.-Q. Pham, T.-L. Ha, F. Schneider, and S. Stüker, 2021: KIT's IWSLT 2021 offline speech translation system. *Proceedings of the 18th International Conference on Spoken Language Translation (IWSLT 2021)*, Association for Computational Linguistics, Bangkok, Thailand (online), 125–130, URL `https://aclanthology.org/2021.iwslt-1.13`.

Nguyen, T. N., N.-Q. Pham, and A. Waibel, 2022: Accent Conversion using Pre-trained Model and Synthesized Data from Voice Conversion. *Proc. Interspeech 2022*, 2583–2587.

Nguyen, T.-N., N.-Q. Pham, and A. Waibel, 2023: Syntacc : Synthesizing multi-accent speech by weight factorization. *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1–5.

Nguyen, T.-N., Q. Pham, and A. Waibel, 2024: Accent conversion using discrete units with parallel data synthesized from controllable accented tts. *Synthetic Data's Transformative Role in Foundational Speech Models*, 51–55.

Nguyen, T. N. and A. Waibel, 2025: Accent conversion for virtual conferences. US Patent 12,375,624.

Nguyen, T.-S., 2021: High performance neural networks for online speech recognizer. Ph.D. thesis, Karlsruher Institut für Technologie (KIT).

Nguyen, T. S., J. Niehues, E. Cho, T.-L. Ha, K. Kilgour, M. Muller, M. Sperber, S. Stueker, and A. Waibel, 2020a: Low latency asr for simultaneous speech translation. *arXiv preprint arXiv:2003.09891*.

Nguyen, T.-S., S. Stüker, and A. Waibel, 2020b: Super-human performance in online low-latency recognition of conversational speech. *arXiv preprint arXiv:2010.03449*.

Nguyen, T.-S., S. Stüker, J. Niehues, and A. Waibel, 2020c: Improving sequence-to-sequence speech recognition training with on-the-fly data augmentation. *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 7689–7693.

Niehues, J., T. S. Nguyen, E. Cho, T.-L. Ha, K. Kilgour, M. Müller, M. Sperber, S. Stüker, and A. Waibel, 2016: Dynamic transcription for low-latency speech translation. *Interspeech*, 2513–2517.

Niehues, J., N.-Q. Pham, T.-L. Ha, M. Sperber, and A. Waibel, 2018: Low-latency neural speech translation. *arXiv preprint arXiv:1808.00491*.

Panayotov, V., G. Chen, D. Povey, and S. Khudanpur, 2015: Librispeech: An asr corpus based on public domain audio books. *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 5206–5210.

Pham, N., T. Nguyen, S. Stüker, and A. Waibel, 2021a: Efficient weight factorization for multilingual speech recognition. *CoRR*, **abs/2105.03010**, URL `https://arxiv.org/abs/2105.03010`, `2105.03010`.

Pham, N.-Q., T.-L. Ha, T.-N. Nguyen, T.-S. Nguyen, E. Salesky, S. Stüker, J. Niehues, and A. Waibel, 2020: Relative Positional Encoding for Speech Recognition and Direct Translation. *Proc. Interspeech 2020*, 31–35.

Pham, N.-Q., T.-N. Nguyen, S. Stüker, and A. Waibel, 2021b: Efficient Weight Factorization for Multilingual Speech Recognition. *Proc. Interspeech 2021*, 2421–2425.

Pham, N.-Q., A. Waibel, and J. Niehues, 2022a: Adaptive multilingual speech recognition with pretrained models. *Proc. Interspeech 2022*, 3879–3883.

———, 2022b: Adaptive multilingual speech recognition with pretrained models. `2205.12304`.

Prenger, R., R. Valle, and B. Catanzaro, 2018: Waveglow: A flow-based generative network for speech synthesis. *CoRR*, **abs/1811.00002**, URL `http://arxiv.org/abs/1811.00002`, `1811.00002`.

———, 2019: Waveglow: A flow-based generative network for speech synthesis. *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 3617–3621.

Qian, K., Y. Zhang, S. Chang, X. Yang, and M. Hasegawa-Johnson, 2019: Autovc: Zero-shot voice style transfer with only autoencoder loss. 1905.05879.

Quamer, W., A. Das, J. Levis, E. Chukharev-Hudilainen, and R. Gutierrez-Osuna, 2022: Zero-Shot Foreign Accent Conversion without a Native Reference. *Proc. Interspeech 2022*, 4920–4924.

Raissi, T., W. Zhou, S. Berger, R. Schlüter, and H. Ney, 2022: Hmm vs. ctc for automatic speech recognition: Comparison based on full-sum training from scratch. URL https://arxiv.org/abs/2210.09951, 2210.09951.

Ren, Y., C. Hu, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T.-Y. Liu, 2021: Fastspeech 2: Fast and high-quality end-to-end text to speech. *International Conference on Learning Representations*, URL https://openreview.net/forum?id=piLPYqxtWuA.

Ren, Y., Y. Ruan, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T.-Y. Liu, 2019: Fastspeech: Fast, robust and controllable text to speech. *Advances in Neural Information Processing Systems*, Wallach, H., H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., Curran Associates, Inc., Vol. 32, URL https://proceedings.neurips.cc/paper/2019/file/f63f65b503e22cb970527f23c9ad7db1-Paper.pdf.

Rezende, D. and S. Mohamed, 2015: Variational inference with normalizing flows. *Proceedings of the 32nd International Conference on Machine Learning*, Bach, F. and D. Blei, Eds., PMLR, Lille, France, Proceedings of Machifigane Learning Research, Vol. 37, 1530–1538, URL https://proceedings.mlr.press/v37/rezende15.html.

Sarfjoo, S. S., S. Madikeri, and P. Motlicek, 2021: Speech activity detection based on multilingual speech recognition system. *Interspeech 2021*, 4369–4373.

Shen, J., R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerrv-Ryan, R. A. Saurous, Y. Agiomvrgiannakis, and Y. Wu, 2018: Natural tts synthesis by conditioning wavenet on mel spectrogram predictions. *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 4779–4783.

Snyder, D., D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, 2018: X-vectors: Robust dnn embeddings for speaker recognition. *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 5329–5333.

Sugiyama, M., H. Sawai, and A. Waibel, 1991: Review of tdnn (time delay neural network) architectures for speech recognition. *1991 IEEE International Symposium on Circuits and Systems (ISCAS)*, 582–585 vol.1.

van den Oord, A., S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, 2016: Wavenet: A generative model for raw audio. URL https://arxiv.org/abs/1609.03499, 1609.03499.

van den Oord, A., O. Vinyals, and K. Kavukcuoglu, 2017: Neural discrete representation learning. *CoRR*, **abs/1711.00937**, URL http://arxiv.org/abs/1711.00937, 1711.00937.

Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, 2017a: Attention is all you need. *Advances in Neural Information Processing Systems*, 5998–6008.

Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, 2017b: Attention is all you need. *Advances in Neural Information Processing Systems*, Guyon, I., U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., Curran Associates, Inc., Vol. 30, URL https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

———, 2017c: Attention is all you need. *Advances in Neural Information Processing Systems*, Guyon, I., U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., Curran Associates, Inc., Vol. 30, URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Waibel, A., 2019a: Connecting humans with humans: Multimodal, multilingual, multiparty mediation. *2019 International Conference on Multimodal Interaction*, 3–4.

———, 2019b: Multimodal dialogue processing for machine translation. *The Handbook of Multimodal-Multisensor Interfaces: Language Processing, Software, Commercialization, and Emerging Directions-Volume 3*, 577–620.

Waibel, A., M. Behr, D. Yaman, F. I. Eyiokur, T.-N. Nguyen, C. Mullov, M. A. Demirtas, A. Kantarci, S. Constantin, and H. K. Ekenel, 2023: Face-dubbing++: Lip-synchronous, voice preserving translation of videos. *2023 IEEE International Conference on Acoustics, Speech, and Signal Processing Workshops (ICASSPW)*, IEEE, 1–5.

Waibel, A. and C. Fuegen, 2012: Simultaneous translation of open domain lectures and speeches. US Patent 8,090,570.

Waibel, A. and C. Fugen, 2008: Spoken language translation. *IEEE Signal Processing Magazine*, **25 (3)**, 70–79.

Waibel, A., T. Hanazawa, G. Hinton, K. Shikano, and K. Lang, 1989: Phoneme recognition using time-delay neural networks. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **37 (3)**, 328–339.

Waibel, A., T. Hanazawa, G. Hinton, K. Shikano, and K. J. Lang, 1987a: Phoneme recognition using time-delay neural networks. *Meeting of the Institute of Electrical, Information and Communication Engineers (IEICE)*, Tokyo, Japan, 19–24.

———, 1987b: Phoneme recognition using time-delay neural networks. Technical Report TR-I-0006, ATR Interpreting Telephony Research Laboratories, Kyoto, Japan.

Wan, L., Q. Wang, A. Papir, and I. L. Moreno, 2020: Generalized end-to-end loss for speaker verification. 1710.10467.

Wang, D., L. Deng, Y. T. Yeung, X. Chen, X. Liu, and H. Meng, 2021a: Vqmivc: Vector quantization and mutual information-based unsupervised speech representation disentanglement for one-shot voice conversion. 2106.10132.

———, 2021b: Vqmivc: Vector quantization and mutual information-based unsupervised speech representation disentanglement for one-shot voice conversion. *Interspeech 2021*, 1344–1348.

Wang, H., S. Zheng, Y. Chen, L. Cheng, and Q. Chen, 2023: Cam++: A fast and efficient network for speaker verification using context-aware masking. URL https://arxiv.org/abs/2303.00332, 2303.00332.

Wang, Y., R. J. Skerry-Ryan, D. Stanton, Y. Wu, R. J. Weiss, N. Jaitly, Z. Yang, Y. Xiao, Z. Chen, S. Bengio, Q. V. Le, Y. Agiomyrgiannakis, R. Clark, and R. A. Saurous, 2017: Tacotron: A fully end-to-end text-to-speech synthesis model. *CoRR*, **abs/1703.10135**, URL http://arxiv.org/abs/1703.10135, 1703.10135.

Wen, Y., D. Tran, and J. Ba, 2020: Batchensemble: an alternative approach to efficient ensemble and lifelong learning. *arXiv preprint*.

Wick, C., J. Zöllner, and T. Grüning, 2022: Rescoring sequence-to-sequence models for text line recognition with ctc-prefixes. URL https://arxiv.org/abs/2110.05909, 2110.05909.

Wikipedia, 2025a: Mel scale — Wikipedia, the free encyclopedia. http://en.wikipedia.org/w/index.php?title=Mel%20scale&oldid=1222316940, [Online; accessed 29-April-2025].

———, 2025b: Pitch detection algorithm — Wikipedia, the free encyclopedia. http://en.wikipedia.org/w/index.php?title=Pitch%20detection%20algorithm&oldid=1240271906, [Online; accessed 29-April-2025].

———, 2025c: Short-time Fourier transform — Wikipedia, the free encyclopedia. http://en.wikipedia.org/w/index.php?title=Short-time%20Fourier%20transform&oldid=1278583871, [Online; accessed 29-April-2025].

———, 2025d: Vocoder — Wikipedia, the free encyclopedia. http://en.wikipedia.org/w/index.php?title=Vocoder&oldid=1286293123, [Online; accessed 29-April-2025].

Wolf, T., L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, and J. Brew, 2019: Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, **abs/1910.03771**, URL http://arxiv.org/abs/1910.03771, 1910.03771.

Yamagishi, J., C. Veaux, and K. MacDonald, 2019: CSTR VCTK Corpus: English multi-speaker corpus for CSTR voice cloning toolkit (version 0.92).

Yamamoto, R., E. Song, and J.-M. Kim, 2020: Parallel wavegan: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram. URL https://arxiv.org/abs/1910.11480, 1910.11480.

Yang, J., J. Lee, Y. Kim, H. Cho, and I. Kim, 2020: Vocgan: A high-fidelity real-time vocoder with a hierarchically-nested adversarial network. URL https://arxiv.org/abs/2007.15256, 2007.15256.

Yang, Y., Y. Kartynnik, Y. Li, J. Tang, X. Li, G. Sung, and M. Grundmann, 2024: Streamvc: Real-time low-latency voice conversion. *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 11 016–11 020.

Yoon, J., S. Kim, E. Yang, and S. J. Hwang, 2019: Scalable and order-robust continual learning with additive parameter decomposition. *arXiv preprint arXiv:1902.09432*.

Zeghidour, N., A. Luebs, A. Omran, J. Skoglund, and M. Tagliasacchi, 2021: Soundstream: An end-to-end neural audio codec. URL https://arxiv.org/abs/2107.03312, 2107.03312.

Zen, H., V. Dang, R. Clark, Y. Zhang, R. J. Weiss, Y. Jia, Z. Chen, and Y. Wu, 2019: Libritts: A corpus derived from librispeech for text-to-speech. *CoRR*, **abs/1904.02882**, URL http://arxiv.org/abs/1904.02882, 1904.02882.

Zhao, G., S. Ding, and R. Gutierrez-Osuna, 2019: Foreign Accent Conversion by Synthesizing Speech from Phonetic Posteriorgrams. *Proc. Interspeech 2019*, 2843–2847.

Zhao, G., S. Sonsaat, A. Silpachai, I. Lucic, E. Chukharev-Hudilainen, J. Levis, and R. Gutierrez-Osuna, 2018: L2-arctic: A non-native english speech corpus. *Proc. Interspeech*, 2783–2787, URL http://dx.doi.org/10.21437/Interspeech.2018-1110.

Zhao, G., S. Sonsaat, A. Silpachai, I. Lucic, E. Chukharev-Hudilainen, J. Levis, and R. Gutierrez-Osuna, 2018: L2-arctic: A non-native english speech corpus. *Interspeech 2018*, 2783–2787.

Zhou, W., W. Michel, R. Schlüter, and H. Ney, 2022: Efficient training of neural transducer for speech recognition. *Interspeech 2022*, ISCA, 2058–2062, interspeech$_2$022, *URL*.

Zhou, Y., Z. Wu, M. Zhang, X. Tian, and H. Li, 2023: Tts-guided training for accent conversion without parallel data. *IEEE Signal Processing Letters*, **30**, 533–537.

B. List of Figures

C. List of Tables

Since I am not a native English speaker, my written English is not yet perfect. Therefore, I used ChatGPT to help paraphrase some paragraphs of my thesis. I did not simply give my published papers to a language model to rewrite them. Instead, I reorganized the content of my research to build a coherent thesis and wrote it paragraph by paragraph based on my own ideas and the connections between the works. After writing each paragraph myself, I used ChatGPT to correct grammar and improve the wording to better match a scientific writing style.

I confirm that all scientific content, ideas, and the overall structure of the thesis are entirely my own work. I have five published papers on this topic, all based on my original ideas. ChatGPT was used only for language improvement and clarity. In addition to the technical parts, I also used ChatGPT to help describe how my accent conversion project could contribute to the world, such as possible applications of my research. This was limited to a small part of the Introduction section.