

Developing Disentangled Speech Representation Algorithms Using Information Theory with Application in Speech Synthesis

Thomas Kassiotis

Thesis submitted in partial fulfillment of the requirements for the
Masters' of Science degree in Computer Science and Engineering

University of Crete
School of Sciences and Engineering
Computer Science Department
Voutes University Campus, 700 13 Heraklion, Crete, Greece

Thesis Advisor: Principal Researcher *Yannis Pantazis*

This work has been performed at the University of Crete, School of Sciences and Engineering, Computer Science Department.

The work has been supported by the Foundation for Research and Technology - Hellas (FORTH), Institute of Computer Science (ICS).

UNIVERSITY OF CRETE
COMPUTER SCIENCE DEPARTMENT

**Developing Disentangled Speech Representation Algorithms Using
Information Theory with Application in Speech Synthesis**

Thesis submitted by
Thomas Kassiotis
in partial fulfillment of the requirements for the
Masters' of Science degree in Computer Science

THESIS APPROVAL

Author:




Thomas Kassiotis

Committee approvals:



Yannis Stylianos
Professor, Committee Member



Yannis Pantazis
Principal Researcher, Thesis Supervisor



Grigorios Tsagkatakis
Assistant Professor, Committee Member

Departmental approval:



Grigorios Tsagkatakis
Assistant Professor, Director of Graduate Studies

Heraklion, November 11, 2024

Developing Disentangled Speech Representation Algorithms Using Information Theory with Application in Speech Synthesis

Abstract

This thesis explores the development and evaluation of disentangled representation learning algorithms for speech signal representation and conditional speech synthesis using tools from information theory. The primary objective is to enhance the ability to separate different attributes of speech, such as content, speaker identity, and style, to improve the controllability and quality of speech synthesis systems.

To achieve this, we extended the FastSpeech 2 [1] model, a state-of-the-art text-to-speech synthesis model developed by researchers at Microsoft, incorporating advanced disentanglement techniques. The research utilizes the Espresso [2] dataset, provided by Meta AI, which includes diverse speech samples essential for training and evaluating the proposed methods.

We implemented several disentanglement methods, including a data-driven approach using a Gradient Reversal Layer [3] with dual classifiers for adversarial training, and various mutual information estimators such as MINE [4], INFO_NCE [5], CLUB [6], as well as two novel estimators: Convex Conjugated Rényi Divergence and Worst Case Regret Rényi Divergence [7]. These techniques were integrated into the FastSpeech 2 model to enable the separation of content, speaker identity, and style attributes in speech signals.

The models were trained and evaluated using a comprehensive set of metrics to assess the quality of disentanglement and the naturalness of the synthesized speech. These metrics include cosine similarity matrices and average inter-cluster distances, which quantify the degree of separation between embeddings, offering insight into how well the model disentangles speaker and style information. Furthermore, we utilized dimensionality reduction techniques (PCA) to visualize the embeddings in a lower-dimensional space, providing a clear visual representation of the model's ability to cluster different speaker and style attributes. For perceptual and intelligibility evaluation, PESQ (Perceptual Evaluation of Speech Quality) and STOI (Short-Time Objective Intelligibility) were employed. Additionally, Word Error Rate (WER) was used to assess the accuracy of the generated speech in terms of content. Among the methods evaluated, the Convex Conjugated Rényi Divergence (CCR) and the combination of Convex Conjugated Rényi Divergence combined with Gradient Reversal Layer (CCR & GRL) demonstrated the most promising results in achieving effective disentanglement.

This research contributes to the field of speech processing by providing a framework for disentangled representation learning, which can be applied to various applications, including personalized speech synthesis and speaker adaptation. Future work will focus on further refining these techniques and exploring their applicability to other domains of speech and audio processing.

Ανάπτυξη Αλγορίθμων Εκμάθησης Ανεξάρτητων Αναπαραστάσεων Ομιλίας Χρησιμοποιώντας Θεωρία Πληροφορίας με Εφαρμογή στη Σύνθεση Ομιλίας

Περίληψη

Η παρούσα διπλωματική εργασία εξετάζει την ανάπτυξη και αξιολόγηση αλγορίθμων εκμάθησης ανεξάρτητων αναπαραστάσεων για την αναπαράσταση σήματος ομιλίας και τη συνθετική ομιλία, χρησιμοποιώντας εργαλεία από τη θεωρία πληροφορίας. Ο πρωταρχικός στόχος είναι η ενίσχυση της ικανότητας διαχωρισμού διαφορετικών χαρακτηριστικών της ομιλίας, όπως το περιεχόμενο, η ταυτότητα του ομιλητή και το στυλ, προκειμένου να βελτιωθεί η ελεγχόμενη δυνατότητα και η ποιότητα των συστημάτων σύνθεσης ομιλίας.

Για την επίτευξη αυτού του στόχου, επεκτείναμε το μοντέλο FastSpeech 2 [1], ένα υπερσύγχρονο μοντέλο μετατροπής κειμένου σε ομιλία, το οποίο αναπτύχθηκε από ερευνητές της Microsoft, ενσωματώνοντας εξελιγμένες μεθόδους διαχωρισμού αναπαραστάσεων. Το σύνολο δεδομένων που χρησιμοποιήθηκε για την εκπαίδευση είναι το Expresso [2], το οποίο παρέχεται από τη Meta AI και περιλαμβάνει μεγάλο εύρος δειγμάτων ομιλίας, απαραίτητα για την εκπαίδευση και αξιολόγηση των προτεινόμενων μεθόδων.

Εφαρμόστηκαν διάφοροι μέθοδοι διαχωρισμού αναπαραστάσεων, συμπεριλαμβανομένης μιας προσέγγισης με τη χρήση του Gradient Reversal Layer [3], σε συνδυασμό με διπλούς ταξινομητές για ανταγωνιστική εκπαίδευση, και διάφορους εκτιμητές αμοιβαίας πληροφορίας όπως οι MINE [4], INFO NCE [5], CLUB [6], καθώς και δύο νέους εκτιμητές: τη Convex Conjugated Rényi Divergence και τη Worst Case Regret Rényi Divergence [7]. Αυτές οι τεχνικές ενσωματώθηκαν στο μοντέλο FastSpeech 2 για να επιτευχθεί ο διαχωρισμός των χαρακτηριστικών περιεχομένου, ταυτότητας ομιλητή και στυλ σε σήματα ομιλίας.

Τα μοντέλα εκπαιδεύτηκαν και αξιολογήθηκαν χρησιμοποιώντας ένα ολοκληρωμένο σύνολο μετρικών για την αποτίμηση της ποιότητας του διαχωρισμού των αναπαραστάσεων και της φυσικότητας της παραγόμενης ομιλίας. Αυτές οι μετρικές περιλαμβάνουν πίνακες συσχέτισης συνημιτόνων και μέσες αποστάσεις μεταξύ συστάδων, οι οποίες ποσοτικοποιούν το βαθμό διαχωρισμού μεταξύ των αναπαραστάσεων. Επιπλέον, χρησιμοποιήθηκαν τεχνικές μείωσης διαστάσεων (PCA) για την οπτικοποίηση των αναπαραστάσεων σε χώρο χαμηλότερης διάστασης, παρέχοντας μια σαφή οπτική αναπαράσταση της ικανότητας του μοντέλου να ομαδοποιεί τα χαρακτηριστικά ομιλητή και στυλ. Για την αξιολόγηση της ποιότητας και της καταληπτότητας, χρησιμοποιήθηκαν οι δείκτες PESQ (Perceptual Evaluation of Speech Quality) και STOI (Short-Time Objective Intelligibility), ενώ για την ακρίβεια της παραγόμενης ομιλίας σε σχέση με το περιεχόμενο χρησιμοποιήθηκε ο δείκτης Word Error Rate (WER). Μεταξύ των μεθόδων που αξιολογήθηκαν, οι Convex Conjugated Rényi Divergence (CCR) και ο συνδυασμός Convex Conjugated Rényi Divergence με Gradient Reversal Layer (CCR & GRL) απέδωσαν τα πιο ενθαρρυντικά αποτελέσματα στην επίτευξη

αποτελεσματικού διαχωρισμού.

Αυτή η έρευνα συνεισφέρει στον τομέα της επεξεργασίας ομιλίας, παρέχοντας ένα πλαίσιο για την εκμάθηση διαχωρισμένων αναπαραστάσεων, το οποίο μπορεί να εφαρμοστεί σε διάφορες εφαρμογές, όπως η εξατομικευμένη σύνθεση ομιλίας και η προσαρμογή ομιλητή. Η μελλοντική εργασία θα εστιάσει στην περαιτέρω βελτίωση αυτών των τεχνικών και στην εξερεύνηση της εφαρμοσιμότητάς τους σε άλλους τομείς επεξεργασίας ομιλίας και ήχου.

Acknowledgements

I would like to express my deepest gratitude to my supervisor, Prof. Yannis Pantazis, for his invaluable guidance, mentorship, and unwavering support throughout this journey. His insights and encouragement have significantly influenced the direction and quality of this work.

I am deeply grateful to my friends for the unforgettable moments we've shared during this journey, which added joy and balance to this experience.

Finally, I would like to express my heartfelt thanks to my family for their unwavering love and support throughout the years.

Exploration is really the essence of the human spirit.

Contents

1	Introduction	1
1.1	Background	1
1.2	Problem Statement	2
1.3	Objectives	4
1.4	Scope of the Study	5
2	Literature Review	7
2.1	Overview of Speech Signal Representation	7
2.1.1	Traditional Methods of Speech Signal Representation	7
2.1.2	Advanced Neural Representations	8
2.1.3	Hybrid Approaches	9
2.1.4	Emerging Trends and Future Directions	9
2.2	Conditional Speech Synthesis	9
2.2.1	Early Methods	9
2.2.2	Deep Learning-based Approaches	10
2.2.3	Conditional Speech Synthesis Techniques	11
2.3	Text-To-Speech (TTS) Systems	11
2.3.1	Text To Speech Systems in the Era of Deep Learning	11
2.3.2	Components of TTS Systems	12
2.3.2.1	Text Analysis	12
2.3.2.2	Acoustic Modeling	14
2.3.2.3	Speech Synthesis - Vocoder	15
2.4	Disentangled Representation Learning	15
2.4.1	Techniques and Models in Disentangled Representation Learning	16
2.5	Information Theory in Machine Learning	17
2.5.1	Key Concepts in Information Theory	17
2.6	Previous Work on Disentanglement in Speech Processing	18
3	Preliminary Work	21
3.1	Dataset	21
3.1.1	Description of the Espresso Dataset	21
3.1.2	Data Statistics	22

3.1.3	Directory Structure	23
3.1.4	Dataset Samples	24
3.1.5	Preprocessing Steps	24
3.2	Models and Algorithms	28
3.2.1	Overview of FastSpeech 2	28
3.2.1.1	FastSpeech 2 Architecture	28
3.2.1.2	FastSpeech 2 Vocoder	30
3.2.1.3	FastSpeech 2 Loss Function	31
4	Disentanglement in Speech Representation	33
4.1	Disentanglement Methods	33
4.1.1	Gradient Reversal Layer and Dual Classifiers	33
4.1.2	Mutual Information Estimators	36
4.1.2.1	MINE	38
4.1.2.2	INFO_NCE	39
4.1.2.3	CLUB	39
4.2	Novel Disentanglement Methods	40
4.2.0.1	Convex Conjugate Rényi	41
4.2.0.2	Worst Case Regret	42
4.2.0.3	Convex Conjugate Rényi & Gradient Reversal Layer (Hybrid Method)	43
5	Extending FastSpeech 2	45
5.1	Embedding Layers for Speaker Identity and Style	45
5.2	Integrating Disentanglement Methods	45
5.3	Data Loader and Preprocessing for Espresso Dataset	47
6	Experiments and Results	49
6.1	Experimental Setup	49
6.2	Cosine Similarity & Average Inter Cluster Distances	50
6.3	Evaluation of Model Performance on Speaker and Style Embeddings	51
6.4	Visualization and Analysis of Embeddings Using Principal Component Analysis (PCA)	53
6.5	Objective Speech Intelligibility Assessment Using STOI	53
6.6	Objective Speech Quality Assessment Using PESQ	55
6.7	Word Error Rate Analysis for Speech Recognition Accuracy	56
7	Conclusion	59
7.1	Summary of Contributions	59
7.2	Challenges and Limitations	60
7.3	Future Work	60

8	Appendices	63
8.1	TextGrid File	63
8.2	Gradient Reversal Layer	67
8.3	Speaker Classifier	67
8.4	Style Classifier	68
8.5	MINE Model	68
8.6	INFO_NCE Model	69
8.7	CLUB Model	69
8.8	Convex Conjugate Rényi Model	70
8.9	Worst Case Regret Model	71
8.10	Cosine Similarity Matrices	72
8.11	PCA Plots for Style &Speaker Embeddings	76
8.12	GPU Details	78
8.13	requirements.txt	78
8.14	Sentence Examples for Word Error Rate (WER) Evaluation	80
8.15	Sentence Examples for STOI &PESQ Evaluation	81
8.16	STOI Results Across Different Models	83
8.17	PESQ Results Across Different Models	84
8.18	Word Error Rate - Convex Conjugate Rényi &GRL	86
8.19	Word Error Rate - Convex Conjugate Rényi	88
8.20	Word Error Rate - Worst Case Regret	91
8.21	Word Error Rate - CLUB	92
8.22	Word Error Rate - INFO_NCE	94
8.23	Word Error Rate - MINE	96
8.24	Word Error Rate - GRL	98
8.25	Flask Application	100
	Bibliography	103

List of Tables

2.1	Acoustic Models	14
2.2	Vocoder Models	15
3.1	Speech Data by Style	22
3.2	Audio Samples for Different Style Categories	24
3.3	Encoder Configuration for FastSpeech 2	30
3.4	Components and Loss Functions of HiFi-GAN	31
6.1	Hardware Specifications	49
6.2	Software Specifications	49
6.3	Average Inter-Cluster Distance for Speakers across Different Models	52
6.4	Average Inter-Cluster Distance for Styles across Different Models	52
6.5	STOI Mean and Standard Deviation across Different Models	54
6.6	STOI Results for Different Styles and Models	55
6.7	PESQ Mean and Standard Deviation across Different Models	55
6.8	PESQ Scores for Different Models and Styles	56
6.9	WER Mean and Standard Deviation across Different Models	57
8.1	STOI Results Across Different Models for Each Sentence	84
8.2	Model PESQ scores across different models and sentences.	86
8.3	WER CCR &GRL	88
8.4	WER CCR	90
8.5	WER WC	92
8.6	WER CLUB	94
8.7	WER INFO_NCE	96
8.8	WER MINE	98
8.9	WER GRL	100

List of Figures

1.1	The five linguistics branches	4
2.1	An example of speech waveform and spectrogram. (a) Waveform (b) Linear-spectrogram (c) Mel-spectrogram	8
2.2	Key Components of TTS	12
3.1	Aligment Pipeline	24
3.2	Montreal Forced Aligner	25
3.3	Preprocess Pipeline	26
3.4	Process Utterance Module	26
3.5	Scaling and Nomalization Module	27
3.6	Metadata Generation Module	27
3.7	FastSpeech 2 Architecture. LR in subfigure (b) denotes the length regulator operation. LN in subfigure (c) denotes layer normaliza- tion. Variance predictor represents duration/pitch/energy predictor.	28
4.1	FastSpeech 2 Architecture with Gradient Reversal Layer and Dual Classifiers for Speaker and Style Disentanglement	35
4.2	Disentanglement Procedure	35
4.3	Joint and Product of Marginals Distribution	37
4.4	FastSpeech 2 Architecture with Mutual Information Estimators for Speaker and Style Disentanglement	37
4.5	Mutual Information Estimation	38
4.6	Step function (red line) versus Lipschitz continuous function (blue line), illustrating the contrast between discontinuous and smoothly varying behaviors	41
4.7	Hybrid FastSpeech 2 Architecture Combining Convex Conjugate Rényi Divergence and Gradient Reversal Layer for Enhanced Speaker and Style Disentanglement	44
5.1	Embedding Layers for Speaker and Style	46
6.1	Cosine Similarity Matrix for Speakers (left) and Styles (right) using the Convex Conjugate Rényi with GRL model.	51
6.2	Speaker and Style Embeddings for CCR &GRL model.	53

8.1	Cosine Similarity Matrix for Speakers (left) and Styles (right) using the MINE model.	72
8.2	Cosine Similarity Matrix for Speakers (left) and Styles (right) using the INFO_NCE model.	72
8.3	Cosine Similarity Matrix for Speakers (left) and Styles (right) using the Gradient Reversal Layer model.	73
8.4	Cosine Similarity Matrix for Speakers (left) and Styles (right) using the Worst Case Regret model.	73
8.5	Cosine Similarity Matrix for Speakers (left) and Styles (right) using the Convex Conjugate Rényi model.	74
8.6	Cosine Similarity Matrix for Speakers (left) and Styles (right) using the CLUB model.	74
8.7	Comparison of Speaker Embeddings Across Different Models Using PCA.	76
8.8	Comparison of Style Embeddings Across Different Models Using PCA.	77
8.9	Screenshot of Flask Application	101

Chapter 1

Introduction

1.1 Background

Speech signal representation and conditional speech synthesis are fundamental areas within the field of speech processing, underlining numerous contemporary applications such as virtual assistants, text-to-speech (TTS) and speech-to-text (STT) systems, personalized voice synthesis, noise removal and various other interactive technologies. These applications leverage advanced speech processing techniques to enhance user interaction and improve accessibility. For instance, virtual assistants like Siri [8] and Alexa [9] rely on accurate speech recognition and synthesis to understand and respond to user commands effectively.

Speech signal representation pertains to the conversion of speech into a format that machines can efficiently analyze and process. This conversion often utilizes features such as Mel-frequency cepstral coefficients (MFCCs) [10] and Mel spectrograms [11], which are traditional methods based on signal processing, or more advanced neural representations that capture the intricate nuances of human speech. Recent advancements have highlighted the critical importance of high-quality speech signal representations in various applications. For instance, the ICASSP 2024 Speech Signal Improvement Challenge [12] emphasized improvements in speech signal quality and representations by addressing distortions such as noise and reverberation, thus enhancing the overall intelligibility and quality of communication systems. Another study explored the robustness of speaker embeddings for speaker identification, demonstrating superior performance compared to traditional methods under challenging acoustic conditions [13].

Conditional speech synthesis, on the other hand, involves generating human-like speech based on specific inputs or conditions, such as text, speaker identity, or emotional tone. The rise of deep learning has significantly advanced this technology, enabling the production of high-quality, natural-sounding synthetic voices. Notable models like Tacotron [14] and FastSpeech [15] have established new benchmarks in this domain, thus enhancing the robustness and versatility of text-to-speech systems.

Disentangled representation learning is crucial for both speech signal representation and conditional speech synthesis. This learning approach involves isolating different underlying factors of variation within the data, such as content, speaker identity, and speaking style. Such disentanglement is vital for improving the flexibility and interpretability of models. Extensive research highlights the advantages of disentangled representations in TTS systems, enabling the modification of one attribute (e.g., speaker identity) without affecting others (e.g., content or emotion). This capability allows for more precise and personalized speech synthesis. Studies such as "Speech Resynthesis from Discrete Disentangled Self-Supervised Representations" [16] showcases the benefits of separating low-bitrate representations for different speech components, thus achieving better control over synthesized speech. Similarly, the work "Learning Disentangled Speech Representations" [17] by Brima et al. on disentangled speech representations provides insights into the strengths and weaknesses of current disentanglement methods, emphasizing the need for improved techniques to achieve better modularity and compactness in representations. These studies underscore the critical need for advancing disentangled representation learning to address the persistent issues of content and style leakage in speech synthesis models. The importance of disentangled representation learning in speech processing is underscored by several key benefits:

- **Enhanced Model Control:** By isolating attributes like content, speaker identity, and style, models can generate speech with specific desired characteristics, thus enhancing user satisfaction and personalization.
- **Improved Generalization:** Disentangled representations aid models in generalizing better to new, unseen data by independently capturing the essential aspects of speech.
- **Robustness to Noise:** These representations enhance the robustness of speech systems to variations and noise, as the model learns to segregate and disregard irrelevant factors.

Overall, disentangled representation learning significantly advances the capabilities and applications of speech processing technologies, paving the way for more adaptive and intelligent systems.

1.2 Problem Statement

Despite significant advancements in speech signal representation and conditional speech synthesis, several challenges remain in achieving highly controllable and interpretable speech synthesis systems. Traditional methods often fail to separate different attributes of speech, such as content, speaker identity, and style, which limits the ability to control these attributes independently. This limitation affects the quality and personalization of synthesized speech, particularly in applications like text-to-speech systems where precise control over various speech attributes is

crucial. Additionally, current models struggle with maintaining the integrity of non-target attributes when modifying specific ones, leading to unnatural or inconsistent speech outputs. A notable issue in this context is the phenomenon of "content leakage", where content information unintentionally influences style embeddings, and "style leakage", where speaker information permeates style embeddings. These problems compromise the effectiveness and clarity of the synthesized speech, posing significant hurdles to achieving truly disentangled representations.

Language is a structured system used by humans for communication, comprising spoken, written, and sign language forms. Speech is the vocalized form of language, serving as a medium for expressing spoken language. Linguistics, the scientific study of language, covers several branches: phonetics, phonology, morphology, syntax, semantics, and pragmatics [18]. These branches deal with various elements of linguistic systems, such as speech sounds, basic units like phonemes and morphemes, sentence structures, meanings, and language usage. Disentangling speaker identity from style is particularly challenging because both attributes exist within the same speech domain and are deeply correlated. Speaker identity includes unique vocal characteristics such as tone, pitch, and rhythm, which are easily conflated with stylistic elements. This overlap makes it difficult for models to isolate one attribute without affecting the other. Consequently, speaker-specific features may unintentionally leak into style embeddings and vice versa, complicating the process of achieving effective disentanglement. The difficulty in disentangling speaker identity, content, and style arises due to their interdependent nature:

1. **Speaker Identity and Style:** Both involve vocal characteristics like pitch and tone, which means changing one can inadvertently affect the other.
2. **Content and Style:** Content pertains to the meaning conveyed by the speech, while style relates to how that content is expressed. Stylistic elements like intonation and pacing can influence the perceived meaning of the content.
3. **Speaker Identity and Content:** Speaker-specific traits can influence how content is delivered and perceived, further complicating the separation.

In the framework of linguistic structures Figure 1.1, various aspects of speech are influenced by distinct attributes. At the core, phonetics focuses on speaker identity, serving as the foundation for distinguishing individual voices. Phonology, also emphasizes speaker identity, refining vocal characteristics. Morphology, the third layer, represents the intersection between speaker identity and style, blending these elements to shape speech patterns. Syntax, illustrates the overlap between content and style, highlighting how sentence structure conveys nuances of meaning and expression. Semantics, is dedicated to content, ensuring clear communication of ideas. Finally, pragmatics, captures the essence of style, influencing how speech is perceived.

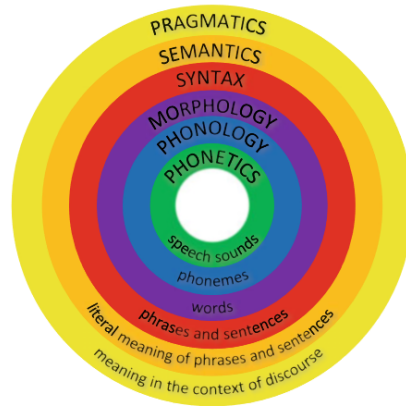


Figure 1.1: The five linguistics branches

Further complicating the problem, current evaluation metrics for disentanglement are often inadequate, failing to fully capture the nuances of attribute separation and their impact on overall speech quality. Moreover, there is a lack of comprehensive datasets that are specifically designed to test and benchmark disentangled representation learning in speech synthesis, which hinders the development and comparison of new methodologies. This inadequacy in existing evaluation metrics has led researchers to create novel frameworks for assessment. For instance, the paper "Theory and Evaluation Metrics for Learning Disentangled Representations" [19] provides a framework for evaluation metrics, emphasizing the need for metrics that capture informativeness, separability, and interpretability.

Additionally, the computational complexity and resource requirements for training advanced models that can effectively disentangle attributes in speech synthesis are substantial. This makes it challenging to implement these models in real-world applications where computational resources may be limited. Ensuring that these models are both efficient and scalable remains a critical challenge.

Addressing these challenges is crucial for advancing the field of speech processing. Developing robust evaluation metrics and comprehensive datasets, along with efficient and scalable models, is imperative for the future of speech synthesis technology.

1.3 Objectives

The primary objectives of this research will be on achieving effective disentanglement using novel techniques and robust evaluation metrics. Specifically, the objectives are:

1. **Develop Advanced Disentanglement Methods**

Implement innovative disentanglement techniques to effectively decouple different speech attributes such as content, speaker identity, and style. Explore and refine methods such as Gradient Reversal Layer (GRL) with dual classifiers, and various mutual information estimators (MINE, INFO_NCE, CLUB, Convex Conjugated Rényi Divergence and Worst Case Regret Rényi Divergence).

2. Integrate and Extend Existing Models

Extend the FastSpeech 2 model to incorporate disentanglement techniques, enabling the separation of content, speaker identity, and style attributes in speech signals.

3. Evaluate and Validate Disentanglement Techniques

Implement evaluation metrics that capture the nuances of attribute separation and their impact on overall speech quality. Utilize metrics to assess the informativeness, separability, and interpretability of disentangled representations.

4. Achieve High-Quality, Personalized Speech Synthesis

Aim to produce high-quality, natural-sounding speech that closely matches desired attributes such as speaker identity and emotional tone.

1.4 Scope of the Study

The scope of this study involves the development, implementation, and evaluation of advanced disentangled representation learning techniques for speech signal representation and conditional speech synthesis. The study aims to address the challenges of achieving controllable and interpretable speech synthesis system by focusing on several key areas:

Theoretical Foundations and Methodologies

- **Exploration of Theoretical Foundations:** This research will delve into the theoretical underpinnings of disentangled representation learning, with an emphasis on methods that can effectively isolate different speech attributes such as content, speaker identity, and style. This includes a thorough examination of the mathematical and algorithmic principles that enable effective disentanglement.
- **Examination of State-of-the-Art Methods:** The study will investigate state-of-the-art techniques such as Gradient Reversal Layer (GRL) with dual classifiers and various mutual information estimators, including MINE, INFO_NCE, CLUB, Convex Conjugated Rényi Divergence, and Worst Case Regret Rényi Divergence. The goal is to refine and enhance these methods for improved

performance in speech synthesis, ultimately aiming to achieve a low-variance mutual information estimator. This improvement will contribute to better disentanglement of speech attributes such as speaker identity and emotional tone, leading to high-quality, natural-sounding speech synthesis.

Dataset Development and Preprocessing

- **Utilization of the Espresso Dataset:** Utilizing the Espresso dataset provided by Meta AI, which includes diverse speech samples crucial for training and evaluating the proposed methods. Comprehensive preprocessing steps will be applied to the dataset for effective training and testing of the models.

Model Implementation and Extension

- **Extension of FastSpeech 2 Model:** FastSpeech 2 model will be extended to incorporate advanced disentanglement techniques, focusing on separating content, speaker identity, and style attributes in speech signals. This involves modifying the model architecture to enhance its capability in handling multiple speech attributes independently.

Experimental Setup and Results Analysis

- **Detailed Experimental Setup:** The research will include a detailed experimental setup, specifying hardware and software configurations, dataset splitting methods, and evaluation protocols. This ensures reproducibility and transparency in the research methodology.
- **Analysis and Interpretation of Results:** The results of the experiments will be analyzed to compare the performance of different disentanglement techniques.

Limitations and Future Work

- **Discussion of Limitations:** The study will discuss the limitations encountered during the research, providing a critical analysis of the challenges and potential areas for improvement.
- **Suggestions for Future Work:** Based on the findings, directions for future research will be suggested, identifying areas that require further investigation and development.

Chapter 2

Literature Review

2.1 Overview of Speech Signal Representation

Speech signal representation is a fundamental concept in the field of speech processing, playing a crucial role in various applications such as speech recognition, speaker identification, and text-to-speech synthesis. This section will provide an overview of key methodologies and advancements in speech signal representation, emphasizing traditional techniques, recent developments, and their implications for speech processing tasks.

2.1.1 Traditional Methods of Speech Signal Representation

Mel-Frequency Cepstral Coefficients (MFCCs) [10] have been a fundamental component in speech signal processing for several decades. They are derived from the Fourier transform of the signal and provide a compact representation of the speech spectrum. By modeling the human ear's perception of sound frequencies, MFCCs effectively capture the important features of speech signals. However, they also have disadvantages, such as sensitivity to noise and a lack of robustness to variations in speech conditions, which can affect their performance in real-world applications.

Linear Predictive Coding (LPC) [20] analyzes the speech signal by estimating the formants and eliminating their effects from the speech waveform. Formant estimation refers to determining formant parameters such as resonant frequency and bandwidth. This method is based on the source-filter model of speech production and provides a parametric representation of the spectral envelope of the speech signal. However, LPC has disadvantages, such as sensitivity to pitch and errors in formant estimation, which can lead to inaccuracies in the spectral representation, especially in the presence of noise or rapid speech variations.

Mel spectrograms [11] are another traditional method used for representing speech signals. They convert the speech signal into a time-frequency representation, where the frequency axis is scaled according to the Mel scale, which approximates the human ear's response to different frequencies. This method provides a

more detailed and perceptually relevant visualization of the speech signal’s spectral content, making it useful for various speech processing tasks. However, Mel spectrograms also have drawbacks, including high computational complexity and sensitivity to noise, which can impact their accuracy and efficiency in practical applications.

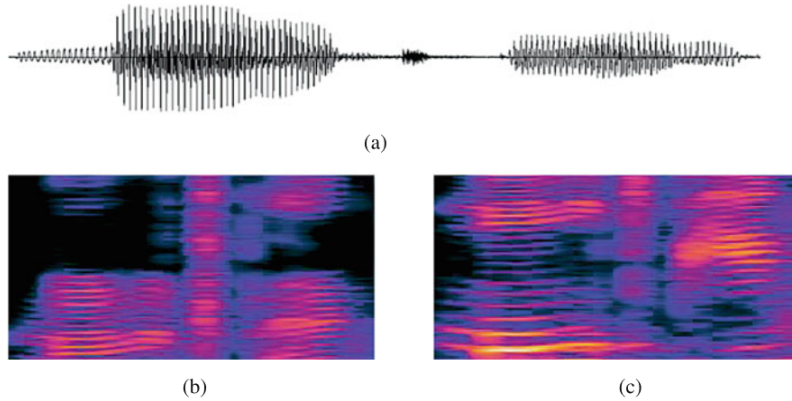


Figure 2.1: An example of speech waveform and spectrogram. (a) Waveform (b) Linear-spectrogram (c) Mel-spectrogram

2.1.2 Advanced Neural Representations

Deep Neural Networks (DNNs) have revolutionized speech signal representation by learning hierarchical features from raw audio data. These models, such as convolutional neural networks (CNNs) [21] and recurrent neural networks (RNNs) [22], can automatically extract complex patterns and representations that are highly effective for various speech processing tasks [23].

WaveNet [24], developed by DeepMind, is a generative model for raw audio waveforms that leverages deep learning to produce highly realistic speech signals. It models the probability distribution of the waveform samples directly, capturing intricate temporal dependencies in the data .

Transformers models [25], particularly those like SpeechBERT [26], have shown significant promise in capturing long-range dependencies in speech signals. By using self-attention mechanisms, transformers can learn robust representations that improve performance in tasks such as speech recognition and synthesis [27].

2.1.3 Hybrid Approaches

Combination of Traditional and Neural Methods Recent research [28] has explored hybrid approaches that combine the strengths of traditional signal processing techniques with advanced neural representations. For instance, features like MFCCs are often used as inputs to DNNs to leverage their interpretability alongside the powerful learning capabilities of deep network.

Self-Supervised Learning approaches [29, 30, 31], such as Contrastive Predictive Coding (CPC), leverage large amounts of unlabeled data to learn meaningful representations. These methods have shown great potential in capturing useful speech features without extensive labeled datasets.

2.1.4 Emerging Trends and Future Directions

Multimodal Speech Representations The integration of speech with other modalities such as text, video, and sensor data is an emerging trend. Multimodal approaches can provide richer and more contextually relevant representations, improving the performance of speech processing systems [32].

2.2 Conditional Speech Synthesis

Conditional speech synthesis focuses on producing human-like speech driven by specific inputs, such as text, speaker identity, or emotional tone. This field has seen substantial progress with the advent of deep learning, resulting in models that can generate high-quality, natural-sounding speech. In this section, we explore key methods and models in the domain of conditional speech synthesis, highlighting their impact and advancements.

2.2.1 Early Methods

Concatenative Synthesis was one of the earliest methods used in TTS systems. It involves piecing together segments of recorded speech to form complete utterances. While it can produce natural-sounding speech, it is limited by the variability and size of the speech database, making it less flexible and scalable. Panda and Nayak [33] describe the integration of rule-based methods with waveform concatenation to improve the naturalness of synthesized speech, yet they note limitations in flexibility due to database constraints. Additionally, a lot of research has been done to propose improvements to the efficiency of concatenative TTS systems. For instance, "An efficient unit-selection method for concatenative text-to-speech synthesis" [34] suggests enhancements to optimize the performance of these systems.

Formant Synthesis generates speech by simulating the resonant frequencies of the human vocal tract. This method allows for more control over speech parameters but often results in less natural-sounding speech compared to concatenative

synthesis. Recent advancements in formant synthesis have focused on leveraging deep learning techniques to improve the naturalness and flexibility of the generated speech. For instance, the study on "Speaker-independent neural formant synthesis" [35] discusses a deep learning approach that uses a small set of phonetically meaningful speech parameters to generate high-quality speech through neural vocoders like WaveNet and HiFi-GAN [36].

Articulatory synthesis [37, 38] generates speech by mimicking the movements of human speech organs such as the lips, tongue, glottis, and vocal tract. In theory, this method could be the most effective approach to speech synthesis since it replicates the natural process of human speech production. However, modeling these articulatory behaviors accurately is extremely challenging. For instance, acquiring the necessary data for simulating these articulatory movements is difficult. Consequently, the speech quality produced by articulatory synthesis is generally lower compared to later methods like formant synthesis and concatenative synthesis.

2.2.2 Deep Learning-based Approaches

Deep learning-based approaches have revolutionized conditional speech synthesis by leveraging neural networks to generate high-quality, natural-sounding speech. These methods typically use models like neural vocoders, autoencoders, and generative adversarial networks (GANs) to synthesize speech conditioned on various inputs such as text, speaker identity, and emotional tone. Deep learning approaches also enable fine-grained control over speech characteristics. For instance, models can be conditioned on various attributes, such as speaker identity, to generate speech in different voices, or on prosodic features to alter the rhythm and intonation of speech. This makes them highly versatile for applications in text-to-speech systems, voice cloning, and emotional speech synthesis.

WaveNet, developed by DeepMind, marked a significant breakthrough in speech synthesis. It is a deep generative model that produces raw audio waveforms and is capable of generating highly realistic and natural-sounding speech. WaveNet models the probability distribution of audio samples directly, capturing intricate temporal dependencies

Tacotron is an end-to-end neural network architecture for TTS. It converts text into mel spectrograms, which are then transformed into waveforms using a vocoder. Tacotron and its successor, Tacotron 2 [39], have set new benchmarks in the field by producing high-quality speech with natural prosody and intonation

FastSpeech addresses the speed and robustness limitations of previous models like Tacotron. It is a non-autoregressive model that predicts mel spectrograms from text in parallel, significantly reducing inference time while maintaining high speech quality. **FastSpeech 2** further improves the model by incorporating more accurate duration predictions and additional conditioning information.

Glow-TTS [40] is a flow-based generative model for TTS that offers high-quality speech synthesis with the advantage of being invertible and having a

tractable likelihood. It uses normalizing flows to model the distribution of mel spectrograms conditioned on text.

NaturalSpeech 3 [41] is another state-of-the-art TTS model that focuses on enhancing the naturalness and expressiveness of synthesized speech. It leverages advancements in neural network architectures and training techniques to produce speech that closely mimics human-like prosody and intonation. The model incorporates sophisticated conditioning mechanisms and a high-fidelity vocoder to generate clear and natural audio, setting new standards in the field of TTS.

2.2.3 Conditional Speech Synthesis Techniques

Conditional speech synthesis techniques are pivotal in creating advanced TTS systems that can adapt to various linguistic and paralinguistic features. These techniques enable the generation of more natural, expressive, and contextually appropriate speech by conditioning the synthesis process on specific attributes or conditions.

Speaker Adaptation and Transfer Learning Speaker adaptation [42] involves fine-tuning a pre-trained TTS model on a small dataset of a target speaker’s voice. This enables the model to synthesize speech in the target speaker’s voice with high fidelity. Transfer learning techniques are often used to adapt the model efficiently without requiring extensive data from the target speaker.

Prosody control techniques aim to manipulate aspects of speech such as pitch, duration, and intensity to produce expressive and natural-sounding speech. By conditioning the TTS model on prosodic features, it is possible to generate speech that conveys different emotions and speaking styles [43].

Emotion and style transfer in speech synthesis involve conditioning the model on attributes like emotional state or speaking style. This allows the TTS system to produce speech that reflects specific emotions or mimics particular styles, enhancing the expressiveness and versatility of synthesized speech [44, 45].

Multi-Speaker and Cross-Lingual Synthesis Multi-speaker TTS models can synthesize speech in different voices by conditioning on speaker identity, while cross-lingual synthesis enables TTS systems to generate speech in multiple languages [46]. These models use speaker embeddings or language embeddings to condition the synthesis process, allowing for the generation of speech in various voices and languages with a single model.

As research in this field continues to evolve, we can expect even greater innovations that will push the boundaries of what is possible in speech synthesis.

2.3 Text-To-Speech (TTS) Systems

2.3.1 Text To Speech Systems in the Era of Deep Learning

With the advent of deep learning, neural network-based text-to-speech systems have emerged, utilizing deep neural networks as the core technology for speech

synthesis. A significant breakthrough came with the introduction of WaveNet, which directly generates waveforms from linguistic features and is considered the first modern neural TTS model.

Subsequently, models like DeepVoice 1 [47] and DeepVoice 2 [48] are structured using three main components (text analysis, acoustic model and vocoder) with each component being enhanced through neural network-based solutions. Moreover, end-to-end models such as Char2Wav [49], Tacotron 1, Tacotron 2, Deep Voice 3 [50], FastSpeech 1 and FastSpeech 2 were developed. These models streamline the text analysis modules by directly using character or phoneme sequences as input and simplifying acoustic feature generation with mel-spectrograms. More recently, fully end-to-end TTS systems have been created that generate waveforms directly from text input. Examples of these advanced models include ClariNet [51], FastSpeech 2, EATS [52], and NaturalSpeech [53]. Compared to earlier TTS methods based on concatenative, formant synthesis and articulatory Synthesis, neural network-based TTS offers significant advantages, including superior voice quality in terms of intelligibility and naturalness, as well as reduced need for extensive human preprocessing and feature engineering.

2.3.2 Components of TTS Systems

A neural TTS system is composed of three fundamental components: a text analysis module, an acoustic model, and a vocoder. As depicted in Figure 2.2, the text analysis module translates a text sequence into linguistic features, the acoustic model generates acoustic features from these linguistic features, and the vocoder synthesizes the waveform from the acoustic features. Specifically, we begin by outlining the main taxonomy of the fundamental components of neural TTS, followed by detailed discussions on text analysis, acoustic models, and vocoders.

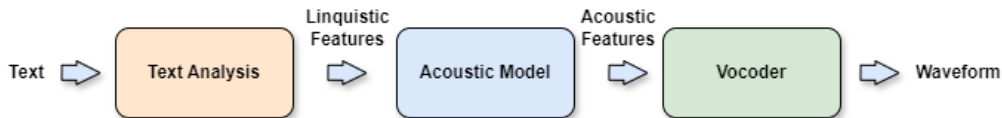


Figure 2.2: Key Components of TTS

2.3.2.1 Text Analysis

The text analysis component is a critical part of a TTS system. It processes raw text input to prepare it for further stages of speech synthesis. Text analysis involves multiple steps to accurately convert text into speech. Among these steps, two fundamental methods are grapheme-to-phoneme (G2P) conversion and prosodic analysis.

Grapheme-to-Phoneme Conversion is the process of translating written text (graphemes) into their corresponding pronunciations (phonemes). This step

is crucial because it bridges the gap between text and its phonetic representation, enabling the TTS system to know how words should be pronounced. For instance the word 'phone' is converted into 'f ow n'. A significant amount of research has been conducted in this area, leading to the development of neural network approaches for grapheme-to-phoneme conversion. These approaches leverage the power of deep learning to handle the complexities and irregularities of non-phonetic languages [54, 55, 56].

Prosodic Analysis examines the rhythmic and intonational aspects of speech, which are crucial for constructing synthesized speech that sounds natural. This involves analyzing prosodic elements such as rhythm, stress, and intonation, which correspond to variations in phoneme duration, loudness, and pitch.

1. **Pitch (Fundamental Frequency)**: Refers to the variations in the pitch of the voice, which help in distinguishing between questions, statements, and emotions.
2. **Duration**: The length of time each sound is held, which can emphasize certain words or indicate natural pauses in speech.
3. **Intensity (Loudness)**: The volume of speech, which can be used to stress important words and convey emotions.

Prosodic features are extracted from text to guide the modulation of speech synthesis, ensuring that the generated speech sounds natural and expressive.

Text Analysis in Neural TTS

In neural TTS systems, the advanced modeling capabilities of neural networks allow character or phoneme sequences to be used directly as input for speech synthesis. This method significantly reduces the complexity of the text analysis module, compared with traditional TTS systems, where text analysis often involved multiple intricate steps including part-of-speech tagging, syntactic parsing, and extensive rule-based processing to handle various linguistic nuances. When characters are utilized as input, only text normalization is required to transform raw text into a standard word format. On the other hand, if phonemes are employed as input, an additional grapheme-to-phoneme conversion step is necessary to derive phonetic representations from the standardized word format. Text analysis has been integrated into neural TTS systems in several ways:

- **Neural Network-Based Text Analysis Modules**: Char2Wav and Deep-Voice 1 & 2 have implemented character-to-linguistic feature conversion directly into their pipelines using neural networks. Unified models by Pan et al. [57] and Zhang et al. [58] have adopted a multi-task paradigm to cover all text analysis tasks, achieving excellent results.

- **Prosody Prediction:** Prosody is vital for the naturalness of synthesized speech. Despite the simplification of text analysis in neural TTS, prosody prediction features have been incorporated into the text encoder. These include the prediction of pitch, duration, phrase breaks [59], breath or filled pauses, and prosodic style features, enhancing the overall speech synthesis quality.
- **Reference Encoders:** Some models utilize reference encoders to learn prosody representations from reference speech [60, 61]. This approach helps in capturing the nuances of natural speech prosody and integrating them into synthesized speech.
- **Text Pre-Training:** Several works focus on learning robust text representations with implicit prosody information through self-supervised pre-training methods [62, 63, 64]. This method helps in capturing the subtleties of prosody, contributing to more natural-sounding speech synthesis.
- **Incorporating Syntax Information:** Dedicated modeling methods, such as graph networks [65], have been employed to incorporate syntax information into neural TTS systems. This approach ensures that the syntactic structure of the text is considered, enhancing the coherence and natural flow of the generated speech.

2.3.2.2 Acoustic Modeling

Acoustic models are responsible for generating acoustic features from linguistic inputs or directly from phonemes. Over the evolution of TTS systems, a variety of acoustic models have been utilized, hidden Markov Models (HMMs) [66], Deep Neural Networks (DNNs), sequence-to-sequence models based on the encoder-attention-decoder framework, incorporating architectures like RNNs, CNNs, and Transformers. More recent advancements include feed-forward networks, specifically utilizing CNNs and Transformers [67], as well as cutting-edge generative models such as GANs [68], VAEs [69], and Diffusion models [70]. Below, there is a table that provides examples of generative models used for acoustic modeling in TTS systems.

Model Type	Models
GAN-Based Models	GAN [71], TTS-Stylization [72], Multi-SpectroGAN [68]
Flow-Based Models	Flowtron [73], Flow-TTS [74], Glow-TTS [40]
VAE-Based Models	BVAE-TTS [75], VAE-TTS [76], Para. Tacotron 1 [77] & 2 [78]
Diffusion-Based Models	Diff-TTS [79], Grad-TTS [80], PriorGrad [81]
Transformer-Based Models	TransfomerTTS [82], FastSpeech [15], FastSpeech 2 [1]

Table 2.1: Acoustic Models

In a later section, we will analyze in depth the acoustic model of FastSpeech 2, exploring its architecture and the innovations it brings to TTS systems.

2.3.2.3 Speech Synthesis - Vocoder

Vocoders are responsible for generating waveforms from acoustic features or directly from linguistic features. Vocoder technology has evolved significantly alongside the development of TTS systems. Initially, various signal processing-based vocoders were employed [83, 84]. However, advancements in neural networks have led to the adoption of neural network-based vocoders [24, 85, 86, 87] which offer enhanced performance and quality.

Vocoders in Neural TTS

Early neural vocoders, such as WaveNet, Parallel WaveNet [88, 85], and WaveRNN, directly take linguistic features as input to generate waveforms. Later models, such as WaveGlow, FloWaveNet, MelGan [89], use mel-spectrograms as input for waveform generation. Given that speech waveforms are extensive, autoregressive waveform generation can be time-consuming. To address this, various deep generative models have been employed for waveform generation, including Normalizing Flows such as NICE [90, 91] Generative Adversarial Networks [92], Variational Autoencoders [93], and Denoising Diffusion Probabilistic Models (DDPM) [94]. Below is a table that provides examples of different vocoder implementations in TTS systems.

Vocoder Type	Models
Autoregressive	WaveNet, SampleRNN [95], WaveRNN, LPCNet [96], FFTNet [97]
Flow-Based	Par. WaveNet, WaveGlow, FloWaveNet, SqueezeWave [98]
GAN-Based	WaveGAN [99], GAN-TTS [100], HiFi-GAN, VocGAN [101], GED [102]
Diffusion-Based	WaveGrad [103], DiffWave, InferGrad [104]

Table 2.2: Vocoder Models

In a later section, we will analyze the HiFi-GAN vocoder that FastSpeech 2 uses.

2.4 Disentangled Representation Learning

Disentangled representation learning [105] aims to separate underlying factors of variation in data into distinct and interpretable components. This approach is crucial in fields like computer vision, natural language processing, and speech synthesis, as it helps models understand and manipulate specific attributes without interference from others.

In the context of speech synthesis, this technique offers significant advantages. For instance, in voice conversion, disentangling speaker identity from linguistic content makes it possible to convert one person’s speech to sound like another’s while maintaining the original content. Additionally, by disentangling prosody (which includes intonation, stress, and rhythm), it becomes feasible to generate speech with varying emotional tones and expressions, enhancing the expressiveness and versatility of synthesized speech.

2.4.1 Techniques and Models in Disentangled Representation Learning

Various techniques and models have been developed to achieve disentanglement, each leveraging different methodologies to address the challenges inherent in diverse application domains such as computer vision, natural language processing, and speech synthesis. Below, we will discuss some of the prominent techniques and models that have been developed to achieve disentanglement.

1. **Variational Autoencoders (VAEs)** are commonly used for disentangled representation learning by forcing the latent space to adhere to a predefined distribution, making it easier to separate different factors of variation. Examples include β -VAE [106] and Factor-VAE [107], which introduce modifications to the standard VAE framework to encourage disentanglement.
2. **Generative Adversarial Networks (GANs)** can also be adapted for disentangled representation learning. Models like InfoGAN [108] maximize the mutual information between subsets of the latent variables and the observations to encourage disentanglement. StyleGAN [109] achieves disentanglement through style transfer techniques in the generator network.
3. **Self-Supervised Learning** Techniques like contrastive learning have been employed to achieve disentanglement without labeled data. Contrastive Learning [110] is a deep learning technique for unsupervised representation learning. The goal is to learn a representation of data such that similar instances are close together in the representation space, while dissimilar instances are far apart. By contrasting different views of the same data, models learn to separate meaningful variations.
4. **Mutual Information-Based Methods** Methods based on mutual information aim to maximize the mutual information between latent variables and observed data, helping to ensure that each latent variable captures distinct and meaningful aspects of the data. Mutual information measures the dependency between variables. In the context of disentangled representation learning, MI quantifies how much information a latent variable contains about a specific aspect of the data. By maximizing MI between latent variables and observed data, these methods encourage each latent variable to

capture distinct information, leading to more interpretable and useful representations.

2.5 Information Theory in Machine Learning

This section will explore how information theory principles are applied in machine learning, highlighting key concepts, applications, and significant research contributions.

2.5.1 Key Concepts in Information Theory

Mutual information quantifies the amount of information gained about one random variable through another, effectively measuring the reduction in uncertainty of one variable given the knowledge of another. In machine learning, mutual information is instrumental for feature selection and clustering. It helps identify features that contribute the most to the predictive power of a model by determining which features share the most information with the target variable. This process aids in constructing simpler, more interpretable models without sacrificing performance. Additionally, mutual information enhances clustering methods by ensuring that the formed clusters are highly informative about the data's inherent structure. Consequently, mutual information-based disentangled representation learning can create representations where different aspects of the data are clearly separated, improving model performance and interpretability.

Kullback-Leibler Divergence (D_{KL}) measures the difference between two probability distributions, specifically quantifying the information lost when one distribution approximates another. Additionally, D_{KL} can be employed to estimate mutual information for two random variables using neural estimators.

Neural Network Estimators for Mutual Information leverage the power of neural networks to estimate mutual information by learning the dependencies between high-dimensional continuous variables. Various techniques have been developed to approximate mutual information, utilizing the expressive capabilities of neural networks to model complex relationships within the data. For example, Belghazi et al. introduced a neural estimator for mutual information [4] that uses neural networks to approximate the mutual information between high-dimensional continuous variables. Similarly, Poole et al. [111] proposed techniques that employ neural networks to estimate bounds on mutual information, thereby facilitating the integration of information-theoretic objectives into the training of deep learning models.

2.6 Previous Work on Disentanglement in Speech Processing

Two primary techniques in disentangled representation learning for speech processing, are Disentangling with Adversarial Training and Disentangling with Semi-Supervised Learning.

1. Disentangling with Adversarial Training

Various studies have explored this area using adversarial training techniques. For instance, Ma et al. [72] enhanced content-style disentanglement ability and controllability through adversarial and collaborative games. Hsu et al. [112] leveraged the VAE framework with adversarial training to disentangle noise from speaker information. Qian et al. [113] proposed SpeechFlow, a method that disentangles rhythm, pitch, content, and timbre using three bottleneck reconstructions. Zhang et al. [114] focused on disentangling noise from speakers using frame-level noise modeling and adversarial training.

2. Disentangling with Semi-Supervised Learning

Controlling speech synthesis attributes such as pitch, duration, energy, prosody, emotion, speaker, and noise can be easier when labels for each attribute are available. However, the challenge arises when there are no tags or only partial labels. When partial labels are available, [115] proposed a semi-supervised learning method to learn the latent variables of the VAE model, enabling control over attributes like affect or speaking rate. In scenarios where no labels are available, [69] introduced Gaussian mixture VAE models to disentangle different attributes. Furthermore, adversarial training techniques have been employed to handle noisy data; for instance, [116] and [117] utilized gradient reversal or adversarial training to disentangle speaker timbre from noise, thereby facilitating the synthesis of clean speech for noisy speakers.

In this section we provide several notable studies that focus on disentanglement representation learning on the domain of speech.

Below

1. Voice Conversion and Speaker Identity

Chou et al. [118] introduces a novel framework for voice conversion that operates without the need for parallel data. Unlike traditional methods that require individual models for each target speaker, this approach allows a single model to convert voices to multiple different speakers. The key innovation lies in disentangling speaker characteristics from the linguistic content in speech signals. Additionally, Hsu et al. [119] presents a generative adversarial network based approach to disentangle speaker identity from phonetic content, enhancing voice conversion systems. These advancements significantly improve the flexibility and efficiency of voice conversion technologies

by enabling better separation of speech components, thereby enhancing the quality and naturalness of the synthesized speech.

2. Prosody and Content Disentanglement

Research done by L. Qu et al. [120] addresses the challenging problem of disentangling prosodic information from speech, which is intrinsically associated with other attributes like timbre and rhythm. The authors propose an unsupervised approach to extract emotional prosody from speech through a novel model called Prosody2Vec. This model integrates three key components to effectively disentangle prosody from semantic content and speaker identity. Skerry-Ryan et al. [39].

3. Speaker Adaptation and Emotional Tone

Akuzawa et al. [121] explores using Variational Autoencoders for expressive speech synthesis, enabling the separation of speaker identity and emotional tone.

Chapter 3

Preliminary Work

In this chapter, we outline the methodology employed to conduct our research. This includes a detailed description of the dataset utilized, the preprocessing steps undertaken to prepare the data for analysis, an overview of the FastSpeech 2 model and its components.

3.1 Dataset

3.1.1 Description of the Espresso Dataset

The Espresso Dataset [2] serves as the primary data source for our research. It comprises a collection of high-quality (48kHz) expressive speech waveforms (36GB), annotated audio recordings specifically designed for speech synthesis and analysis tasks. The dataset includes a diverse range of speakers and multiple linguistic contexts to ensure robustness and generalization of the models developed.

Key features of the Espresso Dataset The Espresso dataset includes a collection of 40 hours of speech data from 4 speakers (2 males, 2 females), encompassing multiple speech styles. This dataset captures a wide range of speech variations, including different speaking rates and emotional tones, ensuring diversity. It includes speech recordings with various speaking styles such as confused, default, enunciated, happy, laughing, sad, and whisper. The audio was recorded in a professional recording studio with minimal background noise at 48kHz/24bit. Additionally, transcriptions of the speech are provided. The Espresso dataset is distributed under the **CC BY-NC 4.0** license.

3.1.2 Data Statistics

Below are the statistics of Expresso's expressive styles:

Table 3.1: Speech Data by Style

Style	Read (min)	Improvised (min)	Total (hrs)
angry	-	82	1.4
animal	-	27	0.4
animal_directed	-	32	0.5
awe	-	92	1.5
bored	-	92	1.5
calm	-	93	1.6
child	-	28	0.4
child_directed	-	38	0.6
confused	94	66	2.7
default	133	158	4.9
desire	-	92	1.5
disgusted	-	118	2.0
enunciated	116	62	3.0
fast	-	98	1.6
fearful	-	98	1.6
happy	74	92	2.8
laughing	94	103	3.3
narration	21	76	1.6
non_verbal	-	32	0.5
projected	-	94	1.6
sad	81	101	3.0
sarcastic	-	106	1.8
singing*	-	4	0.07
sleepy	-	93	1.5
sympathetic	-	100	1.7
whisper	79	86	2.8
Total	11.5h	34.4h	45.9h

Read speech involves speakers reading pre-written text scripts and **Improvised** speech includes spontaneous and unscripted dialogue or monologues.

3.1.3 Directory Structure

The espresso dataset directory has the following structure:

```

espresso/
|-- README.txt
|-- LICENSE.txt
|-- read_transcriptions.txt
|-- VAD_segments.txt
|-- splits/
|   |-- train.txt
|   |-- dev.txt
|   |-- test.txt
|   |-- README
|-- audio_48khz/
|   |-- conversational/
|   |   |-- ex04-ex01/
|   |   |   |-- laughing/ # both channels have the same style
|   |   |   |   |-- ex04-ex01_laughing_001.wav
|   |   |   |   |-- ex04-ex01_laughing_002.wav
|   |   |   |   |-- ...
|   |   |   |-- ...
|   |   |-- ...
|   |-- read/
|   |   |-- ex03/ # speaker
|   |   |   |-- default/ # style
|   |   |   |   |-- longform/ # recorded in long format
|   |   |   |   |   |-- ex03_default_longform_00001.wav
|   |   |   |   |   |-- base/ # recorded in short sentences
|   |   |   |   |   |-- ex03_default_00003.wav
|   |   |   |   |   |-- ex03_default_emphasis_00010.wav
|   |   |   |   |   |-- ex03_default_essentials_00005.wav
|   |   |   |   |   |-- ...
|   |   |   |-- happy/
|   |   |   |   |-- base/
|   |   |   |   |   |-- ex04_happy_00085.wav
|   |   |   |   |   |-- ex04_happy_00091.wav
|   |   |   |   |   |-- ...
|   |   |   |-- ...
|   |-- ...

```

The conversational audio directory has the following structure:

```
conversational/{speaker pair}/{styles}/{speaker pair}_{styles}_{id}.wav
```

The read audio directory has the following structure:

```
read/{speaker}/{style}/{corpus}/{speaker}_{style or substyle}_{id}.wav
```

Table 3.2: Audio Samples for Different Style Categories

Sample Type	Speaker ID	Audio
Confused	Speaker 1	Play
Default	Speaker 3	Play
Enunciated	Speaker 1	Play
Happy	Speaker 4	Play
Laughing	Speaker 4	Play
Sad	Speaker 3	Play
Whisper	Speaker 2	Play

3.1.4 Dataset Samples

In table 3.2 we provide some random samples for different style categories (**Adobe Acrobat Reader** - Click on the "Play" button to listen to the audio samples or visit the Espresso website).

3.1.5 Preprocessing Steps

Before the main preprocessing module, we need to prepare the alignment of our data. The alignment preparation process involves several key steps, as illustrated in the Figure 3.1. Below is a detailed breakdown of each step:

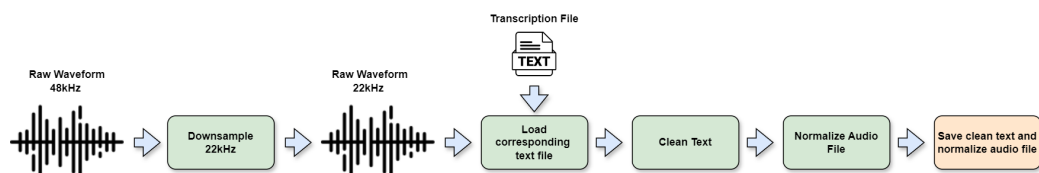


Figure 3.1: Alignment Pipeline

1. **Downsample Audio:** The first step is to downsample the audio files to the target sampling rate of 22kHz. This ensures that all audio data is consistent and suitable for further processing.
2. **Load Corresponding Text:** For each audio file, we load the corresponding text file. This text file contains the content of the audio file, and it is crucial for aligning the audio with its transcription.
3. **Clean Text:** The loaded text undergoes a series of cleaning operations to normalize and standardize it. Examples of text cleaning include:
 - Converting all text to lowercase to maintain uniformity.

- Removing any non-alphanumeric characters (except spaces) to focus on the actual content.
 - Replacing multiple spaces with a single space to avoid unnecessary gaps.
 - Splitting the text into tokens (words or subwords) that can be processed by the phoneme conversion tools.
4. **Normalize Audio:** The audio waveform is normalized to ensure consistent amplitude levels across all files. This involves scaling the audio signal to a standard range, typically using the maximum WAV value specified in the configuration.
 5. **Save Clean Text and Normalized Audio:** Finally, the cleaned text and normalized audio are saved to the target directory. This organized and standardized data is then ready for the main preprocessing module.

Creating TextGrids with Montreal Forced Aligner The next step in our preprocessing pipeline is to create TextGrids for each audio file. TextGrids are annotation files that contain time-aligned transcriptions of the audio data. To obtain these alignments, we use the Montreal Forced Aligner (MFA) tool. Montreal Forced Aligner [122] is a tool that provides accurate alignments between spoken utterances and their corresponding phoneme sequences. It leverages acoustic models and pronunciation dictionaries to perform forced alignment, producing precise timings for each phoneme in the utterance. The resulting TextGrids are crucial for training models like FastSpeech 2, which require detailed phoneme-level alignments. An example of a TextGrid file created using MFA can be found in Appendix 8.1. The TextGrid file sample in the Appendix section, aligns the spoken sentence "how about the age of innocence or vanity fair" with its corresponding phonemes. The file contains two tiers: "words" and "phones," each providing precise timing intervals for words and phonemes respectively.

```
(tom_conda) /mnt/raid1/assiotis:/media/hdd2/assiotis/project/models/custom/FastSpeech2_spr_emb4 mfa align /media/hdd2/assiotis/project/data/raw/Expresso/train-clean-360/d1 /media/hdd2/assiotis/project/models/custom/FastSpeech2_spr_emb/lexicon/Librispeech-lexicon.txt english-us-arpa /media/hdd2/assiotis/project/models/custom/FastSpeech2_spr_emb/preprocessed_data/Expresso/TextGrid/d1
INFO Setting up corpus information ...
INFO Loading corpus from source files ...
INFO Found 1 speaker across 2899 files, average number of utterances per speaker: 2899.0
INFO Initializing multiprocessing jobs
WARNING Number of jobs was specified as 3, but due to only having 1 speakers, MFA will only use 1 jobs. Use the --single_speaker flag if you would like to split utterances across jobs regardless of their speaker.
INFO Normalizing text ...
INFO Generating MFCCs ...
INFO Calculating CMVN ...
INFO Generating final features ...
INFO Creating corpus split ...
INFO Compiling training graphs ...
INFO Performing first-pass alignment ...
INFO Generating alignments ...
INFO Calculating PMLR for speaker adaptation ...
INFO Performing second-pass alignment ...
INFO Generating alignments ...
INFO Collecting phone and word alignments from alignment lattices ...
WARNING Alignment analysis not available without using postgresql
INFO Exporting alignment TextGrids to /media/hdd2/assiotis/project/models/custom/FastSpeech2_spr_emb/preprocessed_data/Expresso/TextGrid/d1
INFO Finished exporting TextGrids to /media/hdd2/assiotis/project/models/custom/FastSpeech2_spr_emb/preprocessed_data/Expresso/TextGrid/d1
INFO Done! Everything took 168.769 seconds
```

Figure 3.2: Montreal Forced Aligner

Preprocessing Main Pipeline The primary preprocessing pipeline comprises three key modules: the process utterance module, the scaling and normalization

module, and the metadata generation module. Each of these modules plays a crucial role in preparing the data for subsequent stages. As illustrated in Figure 3.3, these components work together to ensure high-quality input for model training. In the following sections, we will analyze each module in detail, providing a comprehensive understanding of their functions and implementations.

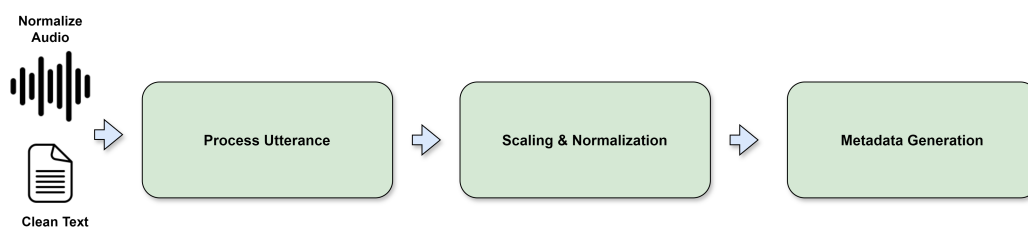


Figure 3.3: Preprocess Pipeline

The first module is the Process Utterance module Figure 3.4, where the following preprocessing steps occur:

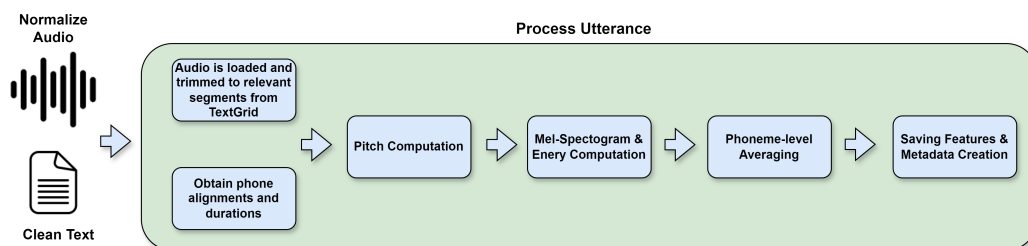


Figure 3.4: Process Utterance Module

1. **TextGrid Reading:** The TextGrid file is read to obtain phone alignments and durations.
2. **Audio Loading and Trimming:** The audio file is loaded and trimmed to the relevant segment using the start and end times from the TextGrid.
3. **Pitch Computation:** The fundamental frequency (pitch) is computed using pyworld.
4. **Mel-spectrogram and Energy Computation:** The mel-spectrogram and energy are computed using the short-time Fourier transform (STFT).
5. **Phoneme-level Averaging:** The pitch and energy values are averaged over the duration of each phoneme.
6. **Saving Features and Metadata Creation:** The computed features (duration, pitch, energy, mel-spectrogram) are saved as .npy files and metadata for the utterance is collected and returned.

The next module is called Scaling and Normalization Figure 3.5, where the following steps occur:

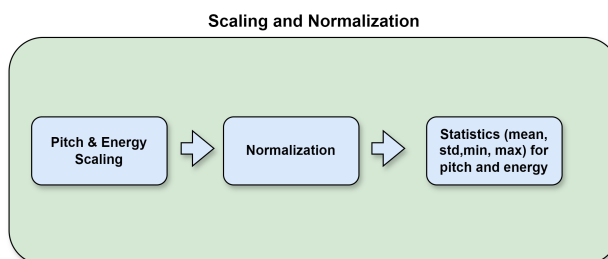


Figure 3.5: Scaling and Normalization Module

1. **Pitch and Energy Scaling:** The StandardScaler from sklearn is used to compute mean and standard deviation for pitch and energy across all utterances.
2. **Normalization:** Pitch and energy values are normalized by subtracting the mean and dividing by the standard deviation.
3. **Statistics Saving:** The computed statistics (mean, std, min, max) for pitch and energy are saved to a JSON file.

The last module is Metadata Generation Figure 3.6, where the following steps occur:

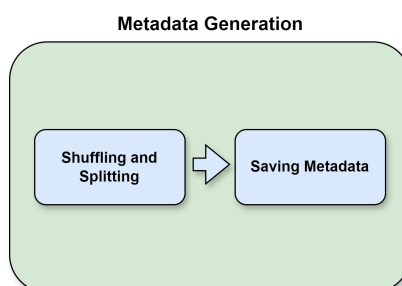


Figure 3.6: Metadata Generation Module

1. **Shuffling and Splitting:** The collected metadata is shuffled and split into training and validation sets based on the specified validation size.
2. **Saving Metadata:** The metadata for training and validation sets is saved to text files (train.txt and val.txt).

3.2 Models and Algorithms

In this section, we delve into the core methodologies and frameworks employed in our study. We will begin by exploring the FastSpeech 2 model.

3.2.1 Overview of FastSpeech 2

FastSpeech 2 was introduced to improve upon its predecessor, FastSpeech, in two major aspects. First, it uses ground-truth mel-spectrograms as training targets instead of the distilled mel-spectrograms derived from an autoregressive teacher model as in FastSpeech. This modification simplifies the two-stage teacher-student distillation process of FastSpeech and prevents the information loss that can occur during the distillation of target mel-spectrograms. Second, FastSpeech 2 integrates additional variance information, including pitch, duration, and energy, into the decoder input. This enhancement addresses the one-to-many mapping problem in text-to-speech synthesis, as noted in previous studies. Overall, FastSpeech 2 provides better voice quality than FastSpeech while maintaining its advantages of fast, robust, and controllable speech synthesis.

3.2.1.1 FastSpeech 2 Architecture

The Figure 3.7 illustrates the architecture of FastSpeech 2, which can be divided into several key components:

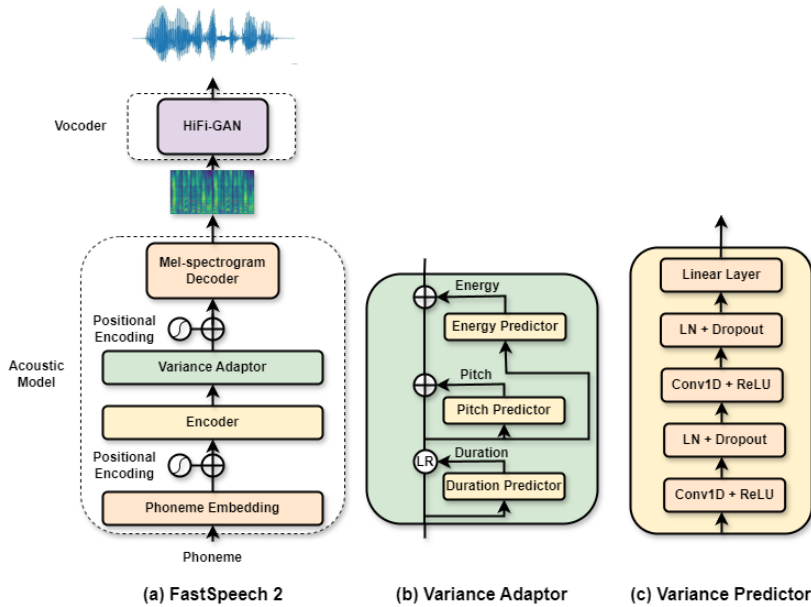


Figure 3.7: FastSpeech 2 Architecture. LR in subfigure (b) denotes the length regulator operation. LN in subfigure (c) denotes layer normalization. Variance predictor represents duration/pitch/energy predictor.

1. **Phoneme Embedding and Positional Encoding:** This component converts the input phoneme sequence into embeddings. Positional encoding is added to these embeddings to provide information about the position of each phoneme in the sequence, which helps the model understand the order of phonemes.
2. **Encoder:** The encoder processes the phoneme embeddings to generate hidden representations. This step involves capturing the relationships between phonemes and transforming the input into a form that the subsequent layers can use effectively.
3. **Variance Adaptor:** The variance adaptor includes predictors for duration, pitch, and energy:
 - **Duration Predictor:** Predicts the duration of each phoneme, which helps in aligning the phoneme sequence with the time domain. For the duration predictor, the output is the length of each phoneme in the logarithmic domain.
 - **Pitch Predictor:** Estimates the pitch contour, which is essential for natural-sounding speech. For the pitch predictor, the output sequence is the frame-level fundamental frequency sequence (F0).
 - **Energy Predictor:** Determines the energy level for each phoneme, contributing to the expressiveness of the generated speech. For the energy predictor, the output is a sequence of the energy of each mel-spectrogram frame.
 - **All predictors share the same model structure (variance predictor) but not model parameters.**
4. **Variance Predictor:** It includes a linear layer followed by layer normalization (LN) and dropout. Then, it uses a Conv1D layer with ReLU activation and another set of layer normalization and dropout. This predictor ensures the accurate prediction of variance information.
5. **Mel-Spectrogram Decoder:** This component generates the mel-spectrogram from the adjusted hidden representations. The mel-spectrogram serves as an intermediate representation of the audio signal.
6. **Post-Net:** is a convolutional neural network module, that serves as a post-processing step to enhance the quality of the synthesized speech. After the decoder generates the initial mel-spectrogram, it is passed through the Post-Net to refine and smooth the spectrogram, reducing artifacts and improving the overall naturalness and intelligibility of the output speech. This additional processing step ensures that the final synthesized speech closely matches the desired audio quality.

7. **Vocoder:** The vocoder is an essential component that converts the refined mel-spectrogram into the final waveform. In our implementation of FastSpeech 2, we utilize the HiFi-GAN vocoder for this task. HiFi-GAN (High-Fidelity Generative Adversarial Network) is designed to produce high-quality, natural-sounding speech waveforms.

Endoer Architecture The core of the encoder consists of 4 Transformer layers. Each Transformer layer includes:

- **Multi-Head Attention:** This mechanism allows the model to focus on different parts of the input sequence simultaneously, capturing various aspects of the phoneme relationships.
- **Feed-Forward Network (FFN):** This component consists of two linear transformations with a ReLU activation in between. It helps in further transforming the representations.
- **Layer Normalization and Dropout:** To stabilize the training and prevent overfitting, layer normalization and dropout are applied.

Parameter	Value
Number of Layers	4 (encoder_layer)
Number of Attention Heads	2 (encoder_head)
Hidden Size	256 (encoder_hidden)
Convolution Filter Size	1024 (conv_filter_size)
Convolution Kernel Size	[9, 1] (conv_kernel_size)
Dropout Rate	0.2 (encoder_dropout)

Table 3.3: Encoder Configuration for FastSpeech 2

3.2.1.2 FastSpeech 2 Vocoder

HiFi-GAN, known for its efficiency and high fidelity, is a generative adversarial network-based vocoder that excels in producing realistic and clear speech. A GAN comprises two main components: a generator, which is responsible for creating data, and a discriminator, which assesses the authenticity of the generated data. The optimization of a GAN involves an adversarial loss function, which can be expressed as follows:

$$\min_{\theta} \max_{\phi} \mathbb{E}_{x \sim p_{data}} \log D(x; \phi) + \mathbb{E}_{x \sim p_z} \log(1 - D(G(z; \theta); \phi)) \quad (3.1)$$

where θ and ϕ denote the parameter of generator and discriminator respectively, and p_{data} and p_z denote the true data distribution and standard Gaussian distribution.

GAN	Generator	Discriminator	Loss
HiFi-GAN [36]	Multi-Receptive Field Fusion	Multi-Period D, Multi-Scale D	LS-GAN, STFT Loss, Feature Matching Loss

Table 3.4: Components and Loss Functions of HiFi-GAN

HiFi-GAN processes various patterns of different lengths in parallel using a multi-receptive field fusion module, allowing it to balance between synthesis efficiency and sample quality. This module enhances flexibility, making it possible to trade off between these factors. Additionally, HiFi-GAN incorporates multi-period discriminators that use multiple discriminators to process equally spaced samples of the input audio over different periods. Specifically, the 1D waveform sequence of length T is reshaped into a 2D data array of dimensions $[P, T/p]$, where p is the period, and then processed by a 2D convolution. This approach enables the model to capture various implicit structures by analyzing different parts of the input audio at different periods.

These embedding layers function as lookup tables, where each entry corresponds to a specific speaker or style. During the training process, the values in these embedding layers are updated to minimize the loss and achieve the desired synthesis outcomes. This means that the model learns to adjust the embeddings to accurately represent the unique characteristics of each speaker and style.

3.2.1.3 FastSpeech 2 Loss Function

The FastSpeech 2 model employs a composite loss function designed to train the neural network effectively. The loss function combines multiple components, each targeting different aspects of the model’s output, to ensure the generated speech matches the desired attributes and quality. Below we will analyze the components of the Loss Function.

- **Mean Squared Error (MSE) Loss:** Used for predicting pitch, energy, and duration. MSE measures the average squared difference between the predicted and target values, which is suitable for continuous data.

$$\text{MSE}(y, \hat{y}) = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad (3.2)$$

- **Mean Absolute Error (MAE) Loss:** Applied to mel-spectrogram predictions. MAE calculates the average absolute differences between the predicted and target values, providing a measure of the overall error magnitude.

$$\text{MAE}(x, y) = \frac{1}{n} \sum_{i=1}^n |x_i - y_i| \quad (3.3)$$

Loss Calculation

- **Mel Loss and Postnet Mel Loss:** The loss between the predicted mel-spectrogram and the target mel-spectrogram. This is calculated using MAE loss for both the initial mel-spectrogram predictions and the refined postnet mel-spectrogram predictions.
- **Pitch Loss:** The loss between the predicted and target pitch values. Depending on the feature level (phoneme or frame), the loss is calculated using MSE on the masked sequences.
- **Energy Loss:** The loss between the predicted and target energy values. Similar to pitch, it uses MSE loss and is applied to the masked sequences based on the feature level.
- **Duration Loss:** The loss between the predicted and target log durations. The logarithm of the durations is taken to stabilize training, and MSE loss is used to measure the differences.
- **Overall Loss:** The total loss is the sum of all individual losses, ensuring that the model learns to predict accurate mel-spectrograms, pitch, energy, and duration. This composite loss function helps in training a robust and high-quality TTS model.

$$\mathcal{L}_{total} = \mathcal{L}_{mel} + \mathcal{L}_{postnet\ mel} + \mathcal{L}_{pitch} + \mathcal{L}_{energy} + \mathcal{L}_{duration} \quad (3.4)$$

Chapter 4

Disentanglement in Speech Representation

4.1 Disentanglement Methods

This section delves into the primary techniques employed for disentanglement, highlighting Gradient Reversal Layer and Dual Classifiers, as well as various Mutual Information Estimators.

4.1.1 Gradient Reversal Layer and Dual Classifiers

Gradient Reversal Layer During forward propagation, the Gradient Reversal Layer (GRL) (Appendix 8.2) functions as an identity transform. However, during backpropagation, it behaves differently. The GRL takes the gradient from the subsequent layer, multiplies it by $-\lambda$, and passes it to the preceding layers. This reversal of the gradients during backpropagation forces the model to learn representations that are invariant to certain attributes, effectively promoting the disentanglement of these attributes.

Speaker and Style Classifiers We implemented two classifiers: one for speaker identity (Appendix 8.3) and one for speaking style (Appendix 8.4). Both classifiers are feed-forward neural networks with a single linear projection layer. The primary difference between them lies in the output layer. The speaker classifier has an output layer with four neurons, each corresponding to one of the four possible speakers, while the style classifier has an output layer with seven neurons, each representing one of the seven speaking styles. This design allows us to effectively map the 256-dimensional embedding vectors to their respective speaker identities and speaking styles by classifying them into the appropriate categories.

Additional Loss Term for Disentanglement We use cross-entropy loss for both the speaker and style classifiers. The cross-entropy loss for a single example

is given by:

$$\text{CrossEntropyLoss}(y, \hat{y}) = - \sum_{c=1}^C y_c \log(\hat{y}_c) \quad (4.1)$$

where y is the true label, \hat{y} is the predicted probability, C is the number of classes. For the speaker classifier, the cross-entropy loss, $\mathcal{L}_{speaker}$, is computed between the predictions of the speaker classifier and the true speaker labels. Similarly, for the style classifier, the cross-entropy loss, \mathcal{L}_{style} , is computed between the predictions of the style classifier and the true style labels.

To promote disentanglement between speaker identity and speaking style, we introduce two classifiers: one for speaker identity and one for style, each trained to minimize its own classification error through cross-entropy losses $L_{speaker}$ and L_{style} . However, to ensure that the embeddings are disentangled, we employ an adversarial training approach using a gradient reversal layer (GRL). In this setup, the GRL is applied only to the embedding layer, such that when the gradients from $L_{speaker}$ and L_{style} are backpropagated, they pass through the GRL. This effectively reverses the gradients' directions before updating the embedding weights. The overall adversarial term in the objective function becomes:

$$L_{GRL} = \lambda(L_{speaker} + L_{style}) \quad (4.2)$$

where λ is a regularization coefficient that controls the influence of the reversed gradient contributions on the embeddings. The FastSpeech loss optimizes the model's performance for accurate speech synthesis, training it to match the target outputs in terms of content, prosody, and naturalness. This loss encourages the model to learn high-quality embeddings that can generate realistic speech. In parallel, we have the adversarial loss from the GRL path. This component includes the speaker and style classification losses $L_{speaker}$ and L_{style} which pass through the GRL before reaching the embedding layer. By minimizing this adversarial loss, the model is forced to produce embeddings that are uninformative about speaker identity and style, promoting disentanglement of these attributes in the learned representations. Thus, by jointly minimizing these two losses, we encourage the model to learn representations that are invariant to variations in both speaker identity and style, achieving robust and disentangled embeddings suitable for generating high-quality speech while remaining flexible to speaker and style changes.

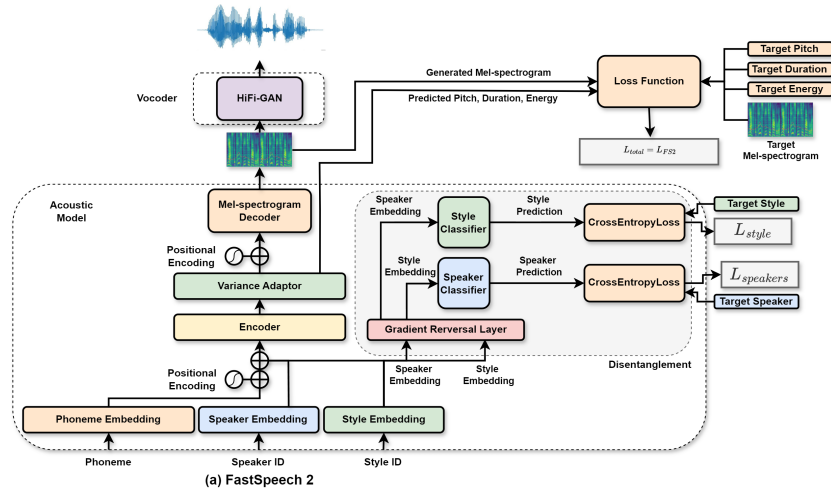


Figure 4.1: FastSpeech 2 Architecture with Gradient Reversal Layer and Dual Classifiers for Speaker and Style Disentanglement

Training Procedure The Figure 4.2 illustrates the process of disentangling speaker and style representations using a Gradient Reversal Layer with dual classifiers.

In the forward pass, the speaker and style embeddings are first passed through the GRL. During this phase, the GRL acts as an identity unit, meaning it simply passes the embeddings through without any changes.

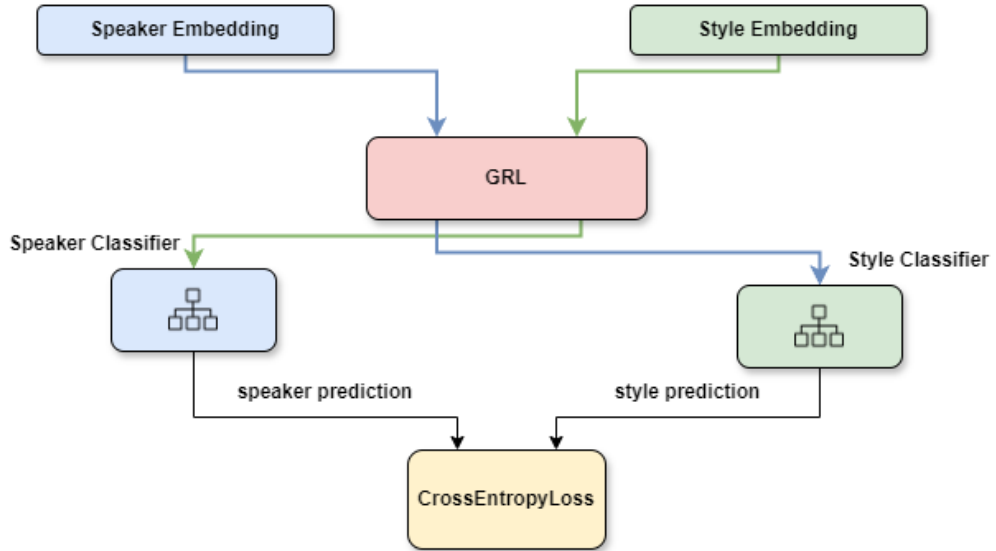


Figure 4.2: Disentanglement Procedure

Following this, the speaker embedding, after passing through the GRL, is fed

into the style classifier. This setup aims to predict the speaking style based on the speaker embedding. Similarly, the style embedding, after passing through the GRL, is fed into the speaker classifier. Here, the objective is to predict the speaker identity from the style embedding.

Both the speaker prediction and the style prediction are then passed to the Cross Entropy Loss function along with their corresponding true labels. This loss function calculates the discrepancy between the predicted labels and the true labels, which is used to update the network weights during back propagation.

In the backward pass, the Gradient Reversal Layer (GRL) modifies the gradients flowing back through the network. Specifically, the GRL multiplies the gradient by $-\lambda$ ($\lambda = 1$) before passing it to the preceding layers. This operation effectively reverses the gradient direction, which serves to penalize the network during backpropagation. By doing so, the GRL encourages the model to learn representations that are invariant to the attributes being disentangled. Essentially, while the classifiers attempt to predict the speaker identity from the style embedding and the style category from the speaker embedding, the reversed gradients counteract this process. This penalization forces the model to reduce any information about the speaker in the style embedding and vice versa, thus achieving better disentanglement.

4.1.2 Mutual Information Estimators

As we discussed in Section 2.5, mutual information (MI) is a measure of the mutual dependence between two variables. It quantifies the amount of information obtained about one variable through the other variable. In our approach, we utilize deep neural networks to estimate the mutual information between two random variables, specifically the speaker and style embeddings in our model.

To estimate MI, we construct two distributions: the joint distribution and the product of marginals. The joint distribution is formed by concatenating the speaker embedding and the style embedding Figure 4.3. The product of marginals is constructed by concatenating the speaker embedding with a randomly permuted style embedding, effectively breaking any inherent correlation between the two. The joint distribution preserves the true correlation between speaker and style embeddings, while the product of marginals, simulates the scenario where there is no dependence between the two variables.

Since we do not have access to the true distributions and can only sample from the training data, we approximate these distributions using the available samples. By calculating the MI between these two distributions, we aim to enforce the mutual information to approach zero during the whole training process, thereby ensuring that the speaker and style embeddings occupy orthogonal spaces. When the mutual information is zero, it indicates that there is no correlations between the speaker and style embeddings, making them independent.

Now, we will analyze the different Mutual Information (MI) estimators that we have implemented. This analysis will focus on understanding how each model

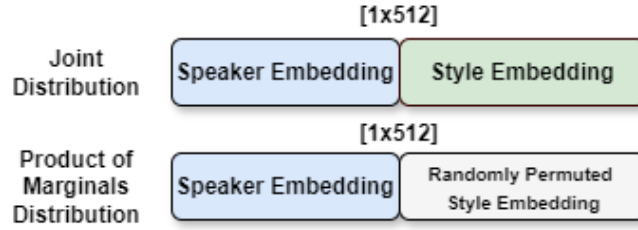


Figure 4.3: Joint and Product of Marginals Distribution

computes the MI, whether as an upper or lower bound. Each of these models approaches MI estimation differently, leveraging various techniques to approximate the true mutual dependence between variables. One significant issue with MI estimators is the high variance they exhibit. High variance in MI estimators means that the estimated values can fluctuate widely depending on the specific sample or batch of data used during training. This instability can make it difficult to obtain reliable and consistent measurements of mutual information, leading to inaccurate representations of the true dependency between variables. High variance can also affect the convergence and performance of models that rely on these estimators, resulting in slower training times and potentially less robust outcomes. Therefore, managing and reducing this variance is crucial for improving the effectiveness and reliability of MI-based disentanglement methods.

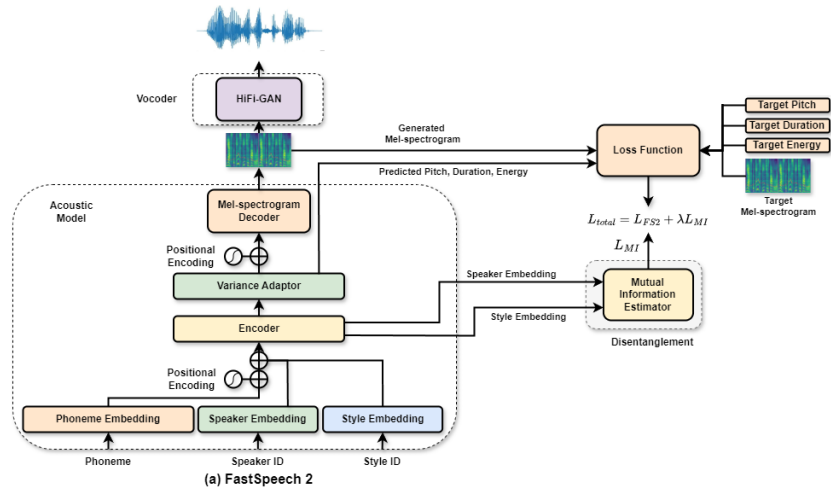


Figure 4.4: FastSpeech 2 Architecture with Mutual Information Estimators for Speaker and Style Disentanglement

4.1.2.1 MINE

Mutual Information Neural Estimation (MINE) [4] is a method for estimating the mutual information between two variables using neural networks. In our implementation, the MINE model is configured to handle the two distribution vectors (joint and product of marginal distributions, as illustrated in Figure 4.3). The MINE model uses a neural network (Appendix 8.5) to estimate MI by computing the lower bound Figure 4.5 of the Kullback-Leibler (KL) divergence between these distributions. The Kullback-Leibler divergence is given by

$$D_{\text{KL}}(P \parallel Q) = \sum_x P(x) \log \frac{P(x)}{Q(x)} \quad (4.3)$$

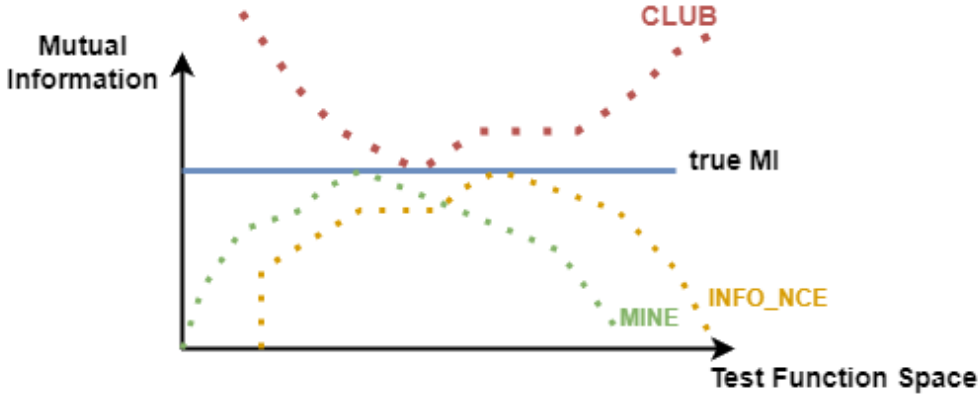


Figure 4.5: Mutual Information Estimation

where P and Q are the joint distribution and product of marginals distribution respectively. Our goal is to minimize the mutual information using the MINE estimator, which provides a lower bound estimate of MI. When the MI is zero, it indicates that there are no correlations between the speaker and style embeddings. This independence means that the latent spaces for these embeddings are orthogonal, thereby achieving disentanglement. We add the MI loss term ($\lambda_{MI} * \mathcal{L}_{MI}$) to the overall loss function, which already includes other components such as mel-spectrogram loss, duration loss, pitch loss, energy loss, and gradient penalty:

$$\begin{aligned} \mathcal{L}_{total} &= \mathcal{L}_{mel} + \mathcal{L}_{postnet\ mel} + \mathcal{L}_{pitch} + \mathcal{L}_{energy} + \mathcal{L}_{duration} + \lambda_{MI} * \mathcal{L}_{MI} \\ &= \mathcal{L}_{FS2} + \lambda_{MI} * \mathcal{L}_{MI} \end{aligned} \quad (4.4)$$

By adding the MI loss term to the total loss function, we enforce the model to learn representations where speaker and style information are disentangled. To enhance numerical stability, the maximum value of the product of marginals vector is subtracted before applying the logarithm and exponential functions. This technique, known as the "log-sum-exp trick", helps to avoid overflow and underflow

issues during the computation of exponential functions, ensuring more stable and reliable calculations. Again we use λ_{mi} as a regularization coefficient to scale the mutual information (MI) loss, in the total loss function. The primary purpose of this coefficient is to balance the contribution of the MI loss relative to other loss components in the training objective.

4.1.2.2 INFO_NCE

INFO_NCE [5], is another technique to estimate a lower bound on mutual information between two random variables. INFO_NCE (Appendix 8.6) uses contrastive learning to maximize the mutual information between two variables by contrasting the true joint distribution against the product of marginals.

The goal of the InfoNCE method is to distinguish between true pairs of embeddings (positive pairs) and random pairs of embeddings (negative pairs).

Positive pairs are formed by directly concatenating the corresponding embeddings from X and Y. For negative pairs, we shuffle the Y embeddings and concatenate them with the X embeddings (all possible combinations of pairs). This shuffling ensures that the correlation between X and Y is broken. Both the positive pairs and the negative pairs are fed into the neural network (INFO_NCE), which processes these pairs and outputs a score.

The lower bound on mutual information is calculated using the scores from positive and negative pairs.

$$MI_{\text{lower bound}} = \frac{1}{N} \sum_{i=1}^N T0_i - \left(\log \left(\frac{1}{N} \sum_{i=1}^N \exp(T1_i - T1_{\max}) \right) + T1_{\max} - \log(N) \right) \quad (4.5)$$

Where $T0_i$ represents the score of the positive pair for sample i , $T1_i$ represents the scores of the negative pairs for sample i , $T1_{\max}$ is the maximum value in $T1$ for numerical stability, and N is the sample size.

4.1.2.3 CLUB

CLUB [6] aims to provide an upper bound 4.5 on the mutual information between two random variables X and Y (speaker and style embeddings). The key idea is to use a neural network to approximate the conditional distribution $q(Y|X)$ and then compute the log-likelihood of the data under this model to estimate the MI.

The CLUB model Appendix 8.7 consists of two main components: the mean prediction network (p_{μ}) and the log variance prediction network ($p_{\log var}$). The p_{μ} network predicts the mean of the conditional distribution $q(Y|X)$, while the $p_{\log var}$ network predicts the log variance of this distribution.

First, we take sample from X and feed them through these networks to obtain the predicted mean (μ) and log variance ($\log \sigma^2$).

Positive samples are pairs (X, Y) drawn from the joint distribution, and their log-likelihood is calculated using the predicted mean and variance.

$$positive = -\frac{(\mu - Y)^2}{2\sigma^2 + 1\epsilon^{-6}} \quad (4.6)$$

Negative samples are created by permuting Y while keeping X the same, effectively simulating the product of marginals distribution. This breaks any correlation between X and Y .

$$negative = -\frac{(Y_{shuffled} - \mu)^2}{2\sigma^2 + 1\epsilon^{-6}} \quad (4.7)$$

The mutual information estimation is obtained by subtracting the average log-likelihood of the negative samples from the average log-likelihood of the positive samples. This difference is averaged to provide the final MI estimate.

$$MI = \frac{(\sum_i positive_i - \sum_i negative_i)}{N} \quad (4.8)$$

Same as before, we add the MI loss term $(\lambda_{MI} * \mathcal{L}_{MI})$ to the overall loss function.

4.2 Novel Disentanglement Methods

In this section, we introduce two novel approaches for disentangling speaker and style embeddings using divergence-based methods: Convex Conjugate Rényi Divergence and Worst-Case Regret Divergence. These methods aim to improve the robustness of mutual information estimations by addressing key challenges, such as high variance.

The **Rényi divergence** (Equation 4.9) is a generalization of the **Kullback-Leibler (KL) divergence**, which provides a way to measure the difference between two probability distributions. It introduces an additional parameter α , which controls the sensitivity of the divergence to different parts of the distributions, such as their tails. When $\alpha = 1$ the Rényi divergence simplifies to the Kullback-Leibler (KL) divergence.

$$R_\alpha(P||Q) = \frac{1}{\alpha - 1} \log\left(\int p(x)^\alpha q(x)^{1-\alpha} dx\right) \quad (4.9)$$

where P and Q are the two probability distributions, $p(x)$ and $q(x)$ are the probability density functions (PDFs) of P and Q , respectively. The variable α is the order of the Rényi divergence. By tuning the parameter α , we can adjust how sensitive the divergence is to different regions of the probability distributions, such as the tails or more central values. This flexibility makes Rényi divergence particularly useful in disentanglement tasks, where we seek to decorrelate two embeddings, such as speaker and style embeddings, by minimizing their mutual information.

The parameter α allows us to fine-tune how we measure the overlap between these distributions, providing more control over the disentanglement process.

The two novel methods we propose, Worst Case (WC) and Convex Conjugate Rényi (CCR), are low-variance estimators due to their use of functions from a Lipschitz-1 continuous space. Previous estimators are characterized by high variance, as they use functions from a test function space that permits step functions 4.6. As shown in the diagram, a step function has an abrupt change at point 0.5, resulting in a large derivative and high variance in the Mutual Information (MI) estimates. In contrast, our proposed methods restrict the function gradients, avoiding sharp changes in value. This Lipschitz continuity ensures more stable MI estimates, with controlled slopes that prevent large fluctuations.

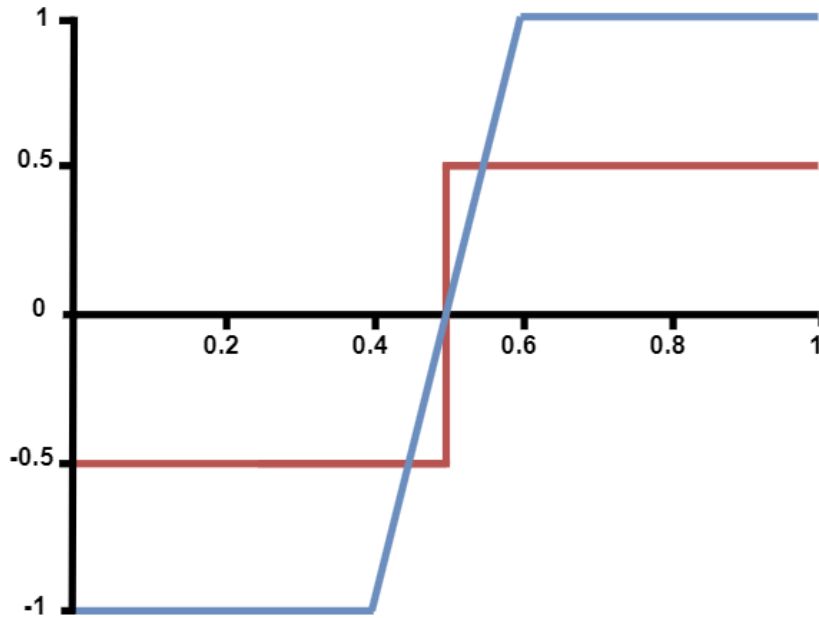


Figure 4.6: Step function (red line) versus Lipschitz continuous function (blue line), illustrating the contrast between discontinuous and smoothly varying behaviors

4.2.0.1 Convex Conjugate Rényi

The Convex Conjugate Rényi Divergence provides a reformulation of the traditional Rényi divergence in a way that eliminates risk-sensitive terms. This is particularly useful for our disentanglement problem, where high variance in MI estimators can lead to unstable training. By leveraging this divergence, we can obtain a more stable lower-bound estimator for the mutual information between speaker and style embeddings.

The **Convex Conjugate Rényi** Formula is as follows

$$R_\alpha^\Gamma(P||Q) = \sup_{g \in \Gamma: g < 0} \left\{ \int g dQ + \frac{1}{\alpha - 1} \log \int |g|^{\frac{\alpha-1}{\alpha}} dP \right\} + \alpha^{-1}(\log a + 1) \quad (4.10)$$

Let g be a function from $\Gamma = Lip_1(\Omega)$, indicating that g is defined on a space of Lipschitz functions on Ω with Lipschitz constant 1. Function g is parameterized by a neural network (Appendix 8.8) that serves as a test function and is learned as part of the optimization process. In order to achieve low variance, we apply a Lipschitz constraint to the neural network test function g , which ensures that the function does not change too rapidly. This constraint is enforced via a gradient penalty during training, which penalizes the model if the gradients of g exceed a set threshold.

The distributions P and Q represent the product of marginals and the joint distribution. Since we do not have access to the true distributions, we approximate these distributions by sampling from the training data. The integral $\int g dQ$ can be approximated by a statistical average, specifically $\frac{1}{n} \sum_{i=1}^n g(x_i)$, where $x_i \sim Q$.

- Q represents the joint distribution, which is constructed by concatenating the speaker and style embeddings.
- P represents the product of marginals, which is obtained by concatenating the speaker embedding with a randomly permuted style embedding, effectively breaking any inherent correlation between the two variables.

As before, we compute the mutual information (MI) between the speaker and style embeddings using the Convex Conjugate Rényi divergence. This computed MI is then added as an additional regularization term to the overall loss function of the FastSpeech2 model. To control the influence of this regularization, we introduce a tuning parameter λ , which scales the contribution of the MI term. By incorporating this extra term, we encourage the disentanglement of speaker and style embeddings, ensuring that the model effectively reduces the mutual dependence between these two factors during training.

4.2.0.2 Worst Case Regret

The Worst-Case Regret divergence is another approach we use to estimate mutual information between the speaker and style embeddings. Unlike traditional divergences, which may be sensitive to specific regions of the probability distributions, the Worst-Case Regret focuses on the maximum possible divergence between two distributions. This provides a more robust measure of divergence, especially in scenarios where we want to account for the worst-case mutual dependence between the embeddings.

The formula for the **Worst-Case Regret Divergence** is given by:

$$D_\infty(P||Q) = \sup_{g \in \Gamma: g < 0} \left\{ \int g dQ + \log \int |g| dP \right\} + 1 \quad (4.11)$$

Here, g is again parameterized by a neural network (Appendix 8.9) that serves as a test function to measure the maximum divergence between the joint and marginal distributions. The neural network is learned as part of the optimization process. To ensure stability and prevent overfitting to extreme outliers, we apply a Lipschitz constraint to the neural network test function g . This constraint ensures that the function does not change too abruptly by bounding the gradients. To enforce the Lipschitz constraint, we apply a gradient penalty during training, which penalizes the model if the gradients of g exceed a predefined threshold. This helps to reduce variance and ensure more robust mutual information estimation during the disentanglement process.

The distributions P and Q represent the joint distribution and the product of marginals, respectively, as defined previously:

- Q represents the joint distribution, constructed by concatenating the speaker and style embeddings.
- P represents the product of marginals, constructed by concatenating the speaker embedding with a randomly permuted style embedding, breaking any inherent correlation.

As before, the mutual information (MI) estimated through the Worst-Case Regret divergence is added as an additional regularization term to the overall loss function of the FastSpeech2 model. To further enhance control over the impact of this regularization, we introduce a tuning parameter, λ , that scales the contribution of the MI term. This allows us to adjust the strength of the disentanglement objective during training, ensuring that the model can balance the trade-off between speaker and style separation and the primary task performance.

4.2.0.3 Convex Conjugate Rényi & Gradient Reversal Layer (Hybrid Method)

In our hybrid approach, we integrate both the convex conjugate Rényi divergence and the gradient reversal layer (GRL) method to enhance disentanglement of speaker identity and speaking style in the FastSpeech 2 model. The convex conjugate Rényi divergence term provides a measure for reducing mutual information between speaker and style attributes, encouraging more independent representations. Simultaneously, the GRL applies adversarial training by reversing gradients from speaker and style classifiers, discouraging the embedding layer from encoding either attribute. This combined approach leverages the strengths of both methods, achieving a more robust disentanglement that improves model flexibility and generalization across varied speaker identities and styles.

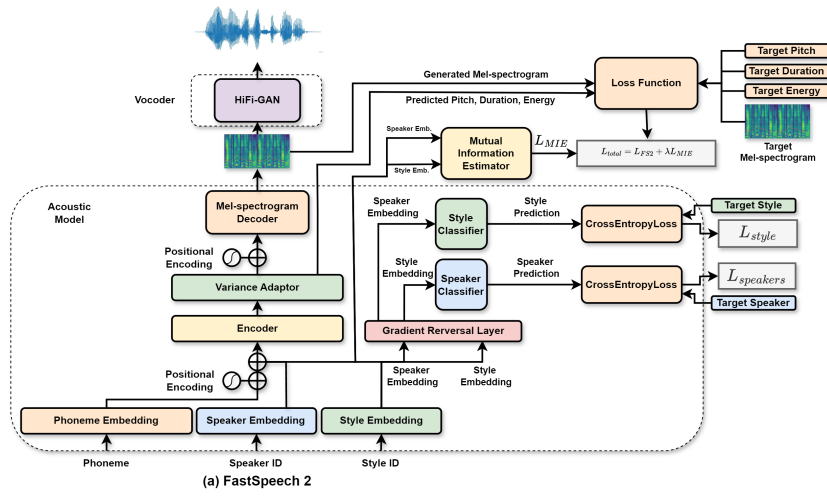


Figure 4.7: Hybrid FastSpeech 2 Architecture Combining Convex Conjugate Rényi Divergence and Gradient Reversal Layer for Enhanced Speaker and Style Disentanglement

Chapter 5

Extending FastSpeech 2

An additional extension in our FastSpeech 2 implementation involves adding speaker and style embeddings to the input. Our focus in disentanglement is to train these embeddings effectively to decouple speaker identity and style.

5.1 Embedding Layers for Speaker Identity and Style

To enhance the FastSpeech 2 model for our specific application, we introduced two additional embedding layers [123] to account for speaker identity and speech style. An embedding layer in neural networks is a lookup table that maps indices from a fixed vocabulary to dense vectors of fixed size. It is commonly used in natural language processing and other tasks where categorical data needs to be transformed into continuous vectors.

The speaker identity embedding layer is designed to support the four speakers in our dataset. This layer has a dimension of 4×256 Figure 5.1, where the number 4 corresponds to the four different speakers, and 256 is the size of each embedding vector. Each speaker is represented by a unique 256-dimensional vector, allowing the model to effectively capture and utilize speaker-specific characteristics during the speech synthesis process.

Similarly, the style embedding layer is configured to learn and represent seven distinct speech styles. This layer has a dimension of 7×256 Figure 5.1, with the number 7 corresponding to the seven different styles we aim to synthesize (confused, default, enunciated, happy, laughing, sad and whisper). Each style is represented by a 256-dimensional vector, enabling the model to generate speech with the appropriate stylistic variations.

5.2 Integrating Disentanglement Methods

To enhance the FastSpeech 2 model with disentangled representations, we integrated several advanced mutual information estimation techniques and adversarial

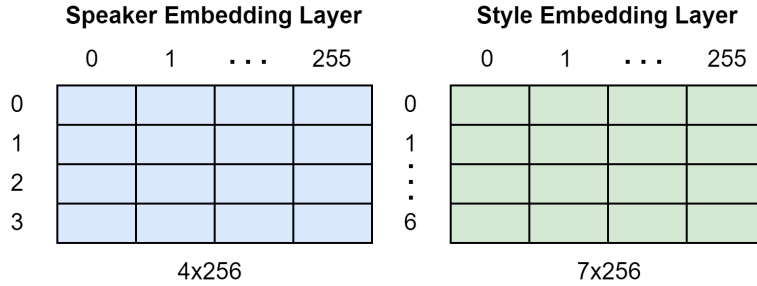


Figure 5.1: Embedding Layers for Speaker and Style

training methods into its architecture. The methods implemented and integrated include:

- **MINE (Mutual Information Neural Estimation)**: We incorporated a neural network-based MI estimator that leverages the joint distribution and product of marginals to compute a lower bound on MI.
- **INFO_NCE (Information Noise Contrastive Estimation)**: This method computes a lower bound on MI using a contrastive loss approach with positive and negative sample pairs.
- **CLUB (Contrastive Learning Upper Bound)**: This model provides an upper bound on MI by approximating the conditional distribution $q(Y|X)$ and computing the log-likelihood of the data under this model.
- **GRL (Gradient Reversal Layer and Dual Classifiers)**: The GRL is used in adversarial training to promote the disentanglement of speaker and style embeddings. Dual classifiers are employed to predict speaker identity and style category, with the GRL reversing the gradient to enforce invariance.
- **CCR (Convex Conjugate Rényi Divergence)**: This novel method estimates MI by leveraging convex conjugate properties to provide a more robust estimation.
- **WC (Worst Case Regret Rényi Divergence)**: Another novel approach, this method estimates MI by considering the worst-case scenario in Rényi divergence, ensuring a more conservative MI estimation.
- **CCR & GRL (Hybrid Method)**: The hybrid approach combines Convex Conjugate Rényi (CCR) divergence and Gradient Reversal Layer (GRL) techniques.

We extended the loss function of FastSpeech 2 to incorporate the MI loss term for each method, promoting the disentanglement of speaker and style embeddings.

This adjustment ensures that the model learns representations where these attributes occupy orthogonal latent spaces, leading to improved disentanglement and more accurate speech synthesis.

5.3 Data Loader and Preprocessing for Espresso Dataset

To accommodate the unique structure of the Espresso dataset, we developed custom scripts for data loading, preprocessing, and corresponding utility functions. The dataset consists of multi-modal data, including speech recordings with varying attributes such as speaker identity and style. For effective training, we ensured that the data pipeline correctly formats and prepares the input.

1. **Data Preprocessing:** We implemented preprocessing functions to handle various forms of input data, including converting audio files into Mel-spectrograms, text tokenization, and extracting speaker and style labels from the metadata. The preprocessing also handles padding and normalization, ensuring that input sequences are of consistent length and form, crucial for batch training.
2. **Custom DataLoader:** Our DataLoader is built to efficiently handle large datasets and batch processing. It integrates seamlessly with PyTorch’s DataLoader class, enabling random shuffling, batching, and loading of the pre-processed data. Additionally, the loader can dynamically fetch speaker and style embeddings based on the indices provided in the Espresso dataset, ensuring that the model receives the correct input features during training.

By building a robust data pipeline, we ensured that the Espresso dataset is fully compatible with the enhanced FastSpeech 2 model. This step was critical in facilitating the training of the model and achieving accurate predictions of speaker identity and style during inference.

Chapter 6

Experiments and Results

6.1 Experimental Setup

The hardware and software specifications are as follows:

Component	Specification
Processor	AMD Ryzen Threadripper 2950X 16-Core Processor
Processor Arch	x86_64
Processor CPUs	32
RAM	125 GB DDR4
GPU	(3x) NVIDIA GeForce RTX 2080 Ti with 11 GB GDDR6 memory
Storage	2.7T HDD

Table 6.1: Hardware Specifications

For more detailed information regarding the GPU specifications and configuration, please refer to the Appendix section 8.12.

Component	Specification
Operating System	Ubuntu 18.04.5 LTS (GNU/Linux 4.15.0-213-generic x86_64)
Python Version	Python 3.6.9 (default, Mar 10 2023, 16:46:00)
PyTorch Version	PyTorch version: 1.10.1+cu102
CUDA Version	Release 10.0, V10.0.130
Other Libraries	Requirements 8.13

Table 6.2: Software Specifications

6.2 Cosine Similarity & Average Inter Cluster Distances

After training, the learned embeddings for both speakers and styles were extracted and saved. These embeddings represent the features that the model has identified to differentiate between speakers and styles in the latent space. To assess the relationships and similarities between these embeddings, a cosine similarity matrix was computed for each model. The cosine similarity matrix provides a quantitative measure of how similar or distinct the embeddings are. Additionally, the embeddings were visualized in 2D spaces using dimensionality reduction techniques such as PCA, offering further insights into the structure of the learned representations. Finally, to better understand the separability of different embeddings, we computed the average inter-cluster distance based on the cosine similarity matrix. This metric provides a summary of how distant the different clusters (speakers or styles) are from each other on average, giving us a measure of the model’s ability to disentangle these factors.

Cosine Similarity is a measure used to determine the similarity between two non-zero vectors in a high-dimensional space, such as the learned embeddings in our models. It is defined as the cosine of the angle between two vectors, with values ranging from -1 (indicating complete dissimilarity) to 1 (indicating identical vectors). Mathematically, the cosine similarity between two vectors \mathbf{u} and \mathbf{v} is given by the formula:

$$\text{Cosine Similarity}(u, v) = \frac{u * v}{\|u\| \|v\|} \quad (6.1)$$

where $u * v$ is the dot product of the vectors u and v and $\|u\|, \|v\|$ are the Euclidean norms of the vectors u and v respectively. The cosine similarity matrix is constructed by computing the cosine similarity between every pair of embeddings. This matrix allows us to compare how closely related the different embeddings are, with higher values indicating higher similarity. Below, we present the cosine similarity matrix for our best model, CCR & GRL. This matrix offers a visual representation of the relationships between the learned embeddings for speakers and styles. The cosine similarity matrices for the other models are provided in the Appendix 8.10 for reference.

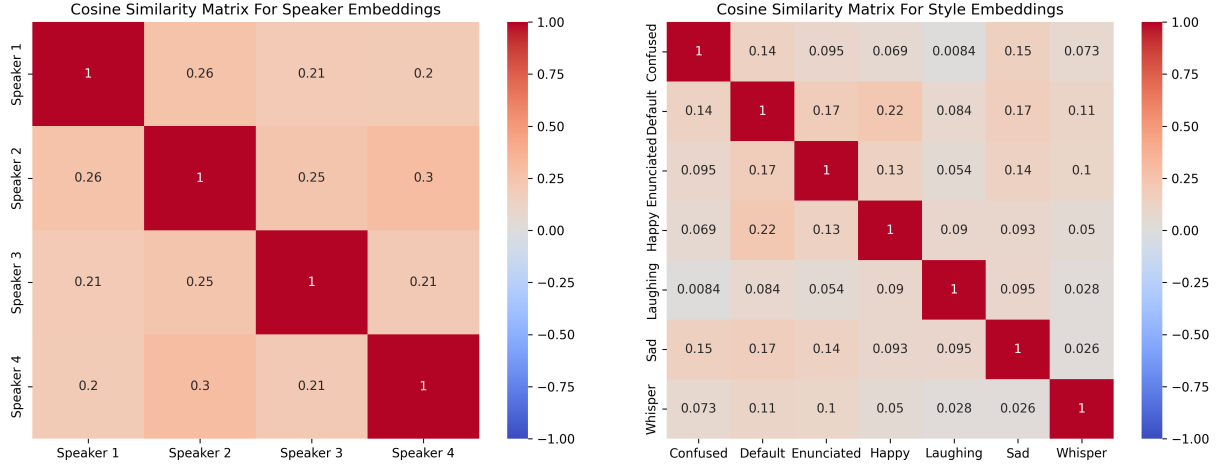


Figure 6.1: Cosine Similarity Matrix for Speakers (left) and Styles (right) using the Convex Conjugate Rényi with GRL model.

Average Inter-Cluster Distance is a metric used to quantify the overall separability between different clusters of embeddings (e.g., speakers or styles) based on their cosine distances. The cosine distance is computed as:

$$\text{Cosine Distance} = 1 - \text{Cosine Similarity} \quad (6.2)$$

To compute the average inter-cluster distance, we first convert the cosine similarity matrix into a distance matrix by applying the above formula to every pair of embeddings. We then consider only the distances between different clusters (i.e., inter-cluster distances) by taking the upper triangular part of the distance matrix, excluding the diagonal (which represents self-distances). The average inter-cluster distance is the mean of these inter-cluster distances, and it is given by:

$$\text{Average Inter-Cluster Distance} = \frac{1}{N} \sum_{\substack{i,j \\ i \neq j}} \text{Cosine Distance}(u_i, u_j) \quad (6.3)$$

where N is the number of unique pairs where $i \neq j$ and u_i, u_j are embeddings.

6.3 Evaluation of Model Performance on Speaker and Style Embeddings

The Tables 6.3 and 6.4 provides insights into the average inter-cluster distances for both speaker and style embeddings across various models. This metric captures the degree of separation between different clusters, which is key for assessing the model’s ability to disentangle embeddings.

Model	Average Inter-Cluster Distance Speaker Embeddings
CCR & GRL	0.9002
CCR	0.8754
GRL	0.8749
WC	0.8484
CLUB	0.8444
MINE	0.7528
INFO	0.6576

Table 6.3: Average Inter-Cluster Distance for Speakers across Different Models

Model	Average Inter-Cluster Distance Style Embeddings
CCR & GRL	0.7616
CCR	0.7566
GRL	0.7316
MINE	0.7200
WC	0.6950
CLUB	0.6911
INFO	0.6266

Table 6.4: Average Inter-Cluster Distance for Styles across Different Models

- Speakers vs. Styles:** The average inter-cluster distances for speakers are generally higher than for styles across all models. This suggests that the models are able to learn more distinct, separable embeddings for different speakers than for different styles. In other words, the speaker embeddings are more spread out in the latent space compared to style embeddings.
- Model Performance on Speakers:** The CCR & GRL model has the highest average inter-cluster distance for speakers (0.9002). This indicates that the CCR& GRL model has learned the most distinguishable speaker embeddings, suggesting that it performs better at separating speaker identities in the latent space.
- Model Performance on Styles:** The CCR & GRL model has the highest average inter-cluster distance for styles (0.7616), meaning it provides the most separable style embeddings, indicating that it captures style variations more distinctly than other models.

6.4 Visualization and Analysis of Embeddings Using Principal Component Analysis (PCA)

In this section, we visualize the style and speaker embeddings for all models using Principal Component Analysis (PCA) to reduce the high-dimensional data into two dimensions. These plots provide a clear illustration of how the different models handle the separation and disentanglement of style and speaker information. Each plot highlights how distinct clusters form for different speakers and styles, offering valuable insights into the models' ability to disentangle these two critical aspects of speech synthesis.

The Figure 6.2 focuses on the style and speaker embeddings for our best-performing model, CCR & GRL. As shown in the PCA plot, this model achieves a remarkable degree of separation between different styles and speakers, with well-defined clusters that emphasize its effectiveness in disentangling style and speaker information. We also perform a comparative analysis by generating similar plots for other models. This comparison enables us to evaluate the level of separation and disentanglement achieved by each model. The plot for the CCR & GRL model is shown in Figure 6.2, while the corresponding plots for the other models can be found in the Appendix 8.11.

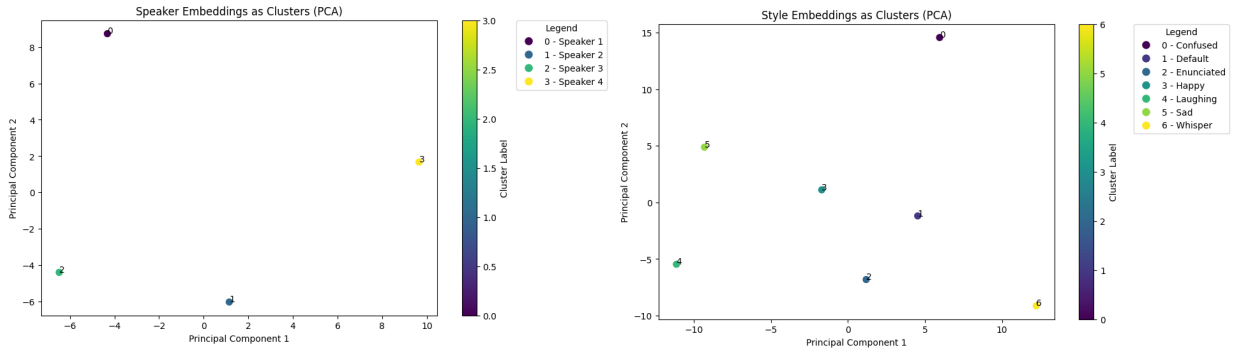


Figure 6.2: Speaker and Style Embeddings for CCR & GRL model.

As we can see from the plots, the CCR & GRL model demonstrates more distinct and widely separated clusters for speakers and styles compared to the other models.

6.5 Objective Speech Intelligibility Assessment Using STOI

Short-Time Objective Intelligibility (STOI) is a widely-used metric for evaluating the intelligibility of speech signals. It provides an objective assessment by comparing a clean reference speech signal with a degraded version. The STOI score is a

value between 0 and 1, where a higher score indicates better speech intelligibility. To compute the STOI, we randomly selected 40 sentences from the training set (detailed in the Appendix 8.15). For each of these sentences, we compared the original WAV file with the corresponding generated version to calculate the STOI score. The table 6.5 presents the results, while a full breakdown of the individual STOI scores for all sentences can be found in the Appendix 8.16.

Model	Mean	Standard Deviation
CCR & GRL	0.3011	0.1202
CCR	0.2537	0.1771
WC	0.2379	0.1661
GRL	0.2272	0.0967
MINE	0.2163	0.1207
INFO	0.2147	0.1202
CLUB	0.1864	0.0924

Table 6.5: STOI Mean and Standard Deviation across Different Models

Among the evaluated models, CCR & GRL model consistently achieves the highest STOI scores across all styles when compared to other models, indicating better intelligibility and more natural voice generation.

An additional evaluation has been conducted, comparing the STOI scores of various models on the sentence ”*What room has no walls*”. The models, listed as V1/2000_V1/S, V1/2000_E/SE, BASE/2000_E/S, and BASE/500_LS/S, are provided by META and pretrained on the Espresso dataset. These models are based on a HuBERT encoder. Specifically V1 their best model, is trained on a mixture of read and spontaneous speech corpora. The base model has been trained on Librispeech. The HuBERT embeddings are extracted from layer 12 for V1 and layer 9 for the base model. These embeddings are followed by k-means clustering, with $k = 2000$ or $k = 500$, on either Espresso (E) or the same training set used for the HuBERT model (V1 or LS, respectively). A HifiGAN vocoder, trained on the Espresso, VCTK, and LJ Speech datasets, is used to convert these discrete units back to speech. The vocoder is either conditioned on the speaker (S) or on both the speaker and expression (S+E).

Our HiFiGAN vocoder, having been trained solely on the Espresso dataset, results in lower STOI scores (CCR, WC, GRL, CCR & GRL cases). This is consistent with the observed outcomes, as the vocoder has not been trained on external datasets, which limits its ability to generalize to broader speech intelligibility scenarios.

Model	Confused	Happy	Sad
V1/2000_V1/S	0.5028	0.4443	0.5591
V1/2000_E/SE	0.6076	0.4753	0.5710
V1/2000_E/S	0.6038	0.4722	0.5638
BASE/2000_E/S	0.4711	0.4019	0.5469
BASE/500_LS/S	0.2094	0.4227	0.5751
CCR	0.2837	0.2365	0.2012
CCR & GRL	0.2921	0.2375	0.2212
WC	0.2437	0.2108	0.1771
GRL	0.1846	0.2794	0.1382

Table 6.6: STOI Results for Different Styles and Models

6.6 Objective Speech Quality Assessment Using PESQ

PESQ (Perceptual Evaluation of Speech Quality) offers an objective method for measuring the perceived quality of speech signals. Instead of relying on subjective human judgment, PESQ models human auditory perception to compare a clean reference signal with a degraded one. This metric quantifies speech quality on a scale ranging from 0 to 4.5 for wideband audio, where higher scores reflect better quality. To compute the PESQ scores, we used the same 40 randomly selected sentences from the training set (detailed in the Appendix 8.15). The Table 6.7 presents the aggregated results, while individual PESQ scores for all sentences can be found in the Appendix 8.17 for further reference.

Model	Mean	Standard Deviation
CCR & GRL	1.1576	0.2748
CCR	1.1162	0.2117
CLUB	1.1114	0.2559
GRL	1.0907	0.2995
WC	1.0837	0.2009
INFO	1.0795	0.1870
MINE	1.0722	0.1905

Table 6.7: PESQ Mean and Standard Deviation across Different Models

From the results the CCR & GRL model consistently outperforms other models in terms of PESQ, achieving the highest mean score with the lowest variability. This suggests that the combination of CCR and GRL provides better perceived speech quality and more consistent performance across different conditions compared to the other models.

As an additional evaluation, we also compared the PESQ scores across different models for the sentence "What room has no walls," similar to the approach taken for the STOI analysis. The results for this specific sentence, along with the model comparisons, are presented below.

Model	Confused	Happy	Sad
V1/2000_V1/S	1.0322	1.0395	1.0496
V1/2000_E/SE	1.0460	1.0500	1.0458
V1/2000_E/S	1.0412	1.0451	1.0558
BASE/2000_E/S	1.0365	1.0396	1.0410
BASE/500_LS/S	1.0418	1.0317	1.0495
CCR	1.2650	1.1398	1.0322
CCR & GRL	1.2721	1.1402	1.0398
WC	1.1044	1.1312	1.1179
GRL	1.0674	1.0382	1.1078

Table 6.8: PESQ Scores for Different Models and Styles

The Table 6.8 shows the PESQ scores for different models across various speech styles. From the results, it can be observed that the CCR & GRL model achieves the highest score for the "Confused" style (1.2721), indicating superior quality in this case, while the v1/2000_e/se model consistently performs well across all styles. PESQ is primarily concerned with speech quality, focusing on how natural or clean the audio sounds to human listeners. STOI, on the other hand, is a measure of speech intelligibility. It evaluates how understandable the speech is, particularly in the presence of noise or other degradation. The CCR & GRL model likely enhances the overall clarity and sound quality of the speech, reducing noise and improving signal consistency, which results in higher PESQ scores. However, it may not preserve the exact intelligibility of the words as effectively, leading to lower STOI scores.

6.7 Word Error Rate Analysis for Speech Recognition Accuracy

Word Error Rate (WER) is a standard metric for evaluating the accuracy of automatic speech recognition (ASR) systems. It measures the number of word-level transcription errors—substitutions, deletions, and insertions—relative to the total number of words in the reference transcription. A lower WER indicates higher transcription accuracy. In this section, we present an analysis of the WER results across various models, highlighting their effectiveness in transcribing the provided speech data. To compute the WER, we generated 50 sentences (detailed in the Appendix 8.14) that were not part of the training set. We used Google's Speech Recognition API to evaluate the WER. The process involved providing the generated wav files as input and comparing them to the original text that was used to generate the speech. The overall results are presented in the Table 6.9, with detailed WER results for each sentence available in the Appendix 8.18-8.24.

From the results, the CCR & GRL model exhibited the best performance, with the lowest mean WER of 0.1250 and a standard deviation of 0.1630. This suggests that the model achieved high accuracy while maintaining relatively low variability, indicating more consistent performance across different speakers and sentences.

6.7. WORD ERROR RATE ANALYSIS FOR SPEECH RECOGNITION ACCURACY57

Model	Mean	Standard Deviation
CCR & GRL	0.1250	0.1630
CCR	0.1909	0.1573
INFO	0.2361	0.2000
GRL	0.2500	0.2218
MINE	0.2613	0.1988
WC	0.2857	0.1806
CLUB	0.3333	0.2238

Table 6.9: WER Mean and Standard Deviation across Different Models

The CCR model also showed good performance, with a mean WER of 0.1909 and a standard deviation of 0.1573. This indicates slightly higher error rates compared to the combined CCR & GRL model, but with lower variability, showing that the model is still reliable across different conditions. Overall, the combination of CCR & GRL provided the best balance between low error rates and performance consistency, making it the most effective model in this evaluation.

Chapter 7

Conclusion

7.1 Summary of Contributions

This work presents several key advancements in the field of speech synthesis, focusing on disentangling speaker and style representations through innovative methodologies. Below, we summarize the main contributions of the research and highlight their impact on improving model performance and practical applicability.

1. **Development of Extended FastSpeech 2 Model:** This master’s thesis presents an enhanced version of the FastSpeech 2 model through the integration of disentangled speaker and style embeddings. This enables the generation of diverse and natural-sounding speech for multiple speakers and styles.
2. **Integration of Mutual Information Estimators:** We introduced advanced mutual information estimation techniques such as MINE, INFO NCE, CLUB, Convex Conjugate Rényi, Worst Case Regret and Gradient Reversal Layer to ensure disentangled representations. These methods improved the separation between speaker and style embeddings.
3. **Comprehensive Evaluation:** The work includes a thorough evaluation of the learned embeddings using cosine similarity matrices, inter-cluster distances, and dimensionality reduction techniques (PCA). Additionally, perceptual and intelligibility metrics like STOI (Speech Intelligibility), PESQ (Perceptual Evaluation of Speech Quality), and WER (Word Error Rate) were used to further assess the speech synthesis quality and intelligibility across different models. The results demonstrated that certain models, especially the Convex Conjugate Rényi & Gradient Reversal Layer model, provided more distinct clusters for both speakers and styles, while also achieving notable scores in perceptual and intelligibility metrics.
4. **Practical Contribution:** A flask-based application was developed to compare the results of our models with baseline models from the Espresso dataset.

This allowed for a more interactive demonstration of the model’s performance (Appendix 8.25).

7.2 Challenges and Limitations

Despite the promising results achieved through our model adaptations and experiments, several challenges and limitations were encountered throughout the process. These challenges impact the effectiveness, generalizability, and computational efficiency of the models. Some of these challenges are listed below.

- **Data Challenges** One significant challenge in this work arises from limited or imbalanced datasets. In speech synthesis tasks, it’s crucial to have a large, diverse set of training samples to ensure that the model generalizes well across different speakers and styles. However, for certain styles or speaker identities, the dataset might have fewer examples, leading to biased learning. This imbalance can make it difficult for the model to accurately learn the distribution of features related to these underrepresented classes, resulting in lower quality or less reliable synthesis for these categories.
- **Disentangling Features** Another challenge is disentangling different features, such as speaker identity and style. However, this is often difficult because the two factors may overlap in the latent space. For an easier example of disentangling two features, consider a model tasked with distinguishing between color and shape in object recognition. Separating a ”red circle” from a ”blue square” is relatively straightforward for the model, as color and shape are distinct features with minimal overlap.
- **Generalization to Unseen Data** The model’s ability to generalize to unseen speakers, styles, or languages is often constrained by the diversity of the training data. If the model is only trained on a limited range of speakers and styles, its ability to generate accurate and natural-sounding speech for new, unseen combinations of speaker and style may be compromised.
- **Vocoder Quality** The vocoder (e.g., HiFi-GAN) that converts the model’s intermediate outputs (such as mel-spectrograms) into audio is another source of potential limitation. If the vocoder is not properly trained or fine-tuned on the same dataset as the main model, it can degrade the quality of the generated audio, producing unnatural or robotic-sounding speech.

7.3 Future Work

Future work can address the existing limitations and explore new methodologies to improve performance, efficiency, and generalization. The following items outline potential directions for continued development.

- **Fine-tuning Hyperparameters of FastSpeech2 and MI Estimators** Future work can explore optimizing several key hyperparameters to enhance the performance and efficiency of the FastSpeech2 model and the Mutual Information (MI) Estimators. In FastSpeech2, parameters such as the learning rate, batch size, number of layers, and attention heads play critical roles in determining how well the model learns the intricate patterns of speech. Fine-tuning these values can help improve the quality of the synthesized speech and allow for better generalization across diverse datasets. In MI Estimators, parameters like the learning rate, the number and types of layers (e.g., fully connected, convolutional), and other architectural elements could also be optimized to better disentangle speaker and style features.
- **Optimizing Embedding Dimensions** Determining the optimal size for the speaker and style embeddings is a crucial aspect of improving model performance. Future work can explore advanced techniques like bottleneck layers, which force the model to represent information using a compressed feature space, encouraging it to learn only the most relevant features. Other dimensionality reduction techniques, such as Principal Component Analysis (PCA) and autoencoders, could also be used to identify the optimal embedding size. Additionally, leveraging state-of-the-art methods like Neural Architecture Search (NAS) can automate the search for the best embedding dimensions, striking a balance between model complexity and accuracy.
- **Training the Vocoder on More Diverse Datasets** To enhance the naturalness of synthesized speech, future work can focus on training the vocoder on a more diverse range of datasets. A vocoder trained on a variety of speaking styles, accents, and languages will be better equipped to capture the intricacies of natural speech, resulting in more lifelike and expressive voice outputs. By incorporating diverse datasets, the vocoder can learn a broader range of acoustic patterns, improving its ability to generalize and produce more natural-sounding speech across different speakers and styles.
- **Scaling Up the Training Dataset** Increasing the size and diversity of the training dataset can significantly enhance the performance of the model. A larger dataset allows the model to capture more variations in speaker characteristics, styles, and acoustic environments, leading to improved generalization. By training on more data, the model can better handle edge cases and produce higher-quality, more natural speech across a broader range of scenarios. Additionally, incorporating datasets with diverse accents, languages, and speaking conditions could further refine the model's robustness and versatility.

Chapter 8

Appendices

8.1 TextGrid File

```
1 File type = "ooTextFile"
2 Object class = "TextGrid"
3
4 xmin = 0
5 xmax = 3.830476
6 tiers? <exists>
7 size = 2
8 item []:
9   item [1]:
10     class = "IntervalTier"
11     name = "words"
12     xmin = 0
13     xmax = 3.830476
14     intervals: size = 13
15     intervals [1]:
16       xmin = 0.0
17       xmax = 0.12
18       text = ""
19     intervals [2]:
20       xmin = 0.12
21       xmax = 0.48
22       text = "how"
23     intervals [3]:
24       xmin = 0.48
25       xmax = 0.91
26       text = "about"
27     intervals [4]:
28       xmin = 0.91
29       xmax = 1.07
30       text = "the"
31     intervals [5]:
```

```
32         xmin = 1.07
33         xmax = 1.1
34         text = ""
35     intervals [6]:
36         xmin = 1.1
37         xmax = 1.37
38         text = "age"
39     intervals [7]:
40         xmin = 1.37
41         xmax = 1.51
42         text = "of"
43     intervals [8]:
44         xmin = 1.51
45         xmax = 2.18
46         text = "innocence"
47     intervals [9]:
48         xmin = 2.18
49         xmax = 2.31
50         text = ""
51     intervals [10]:
52         xmin = 2.31
53         xmax = 2.67
54         text = "or"
55     intervals [11]:
56         xmin = 2.67
57         xmax = 3.13
58         text = "vanity"
59     intervals [12]:
60         xmin = 3.13
61         xmax = 3.68
62         text = "fair"
63     intervals [13]:
64         xmin = 3.68
65         xmax = 3.830476
66         text = ""
67 item [2]:
68     class = "IntervalTier"
69     name = "phones"
70     xmin = 0
71     xmax = 3.830476
72     intervals: size = 34
73     intervals [1]:
74         xmin = 0.0
75         xmax = 0.12
76         text = ""
77     intervals [2]:
78         xmin = 0.12
79         xmax = 0.24
80         text = "HH"
```

```
81         intervals [3]:
82             xmin = 0.24
83             xmax = 0.48
84             text = "AW1"
85         intervals [4]:
86             xmin = 0.48
87             xmax = 0.52
88             text = "AHO"
89         intervals [5]:
90             xmin = 0.52
91             xmax = 0.61
92             text = "B"
93         intervals [6]:
94             xmin = 0.61
95             xmax = 0.81
96             text = "AW1"
97         intervals [7]:
98             xmin = 0.81
99             xmax = 0.91
100            text = "T"
101         intervals [8]:
102             xmin = 0.91
103             xmax = 0.96
104             text = "DH"
105         intervals [9]:
106             xmin = 0.96
107             xmax = 1.07
108             text = "IYO"
109         intervals [10]:
110             xmin = 1.07
111             xmax = 1.1
112             text = ""
113         intervals [11]:
114             xmin = 1.1
115             xmax = 1.29
116             text = "EY1"
117         intervals [12]:
118             xmin = 1.29
119             xmax = 1.37
120             text = "JH"
121         intervals [13]:
122             xmin = 1.37
123             xmax = 1.45
124             text = "AHO"
125         intervals [14]:
126             xmin = 1.45
127             xmax = 1.51
128             text = "V"
129         intervals [15]:
```

```
130         xmin = 1.51
131         xmax = 1.59
132         text = "IH1"
133     intervals [16]:
134         xmin = 1.59
135         xmax = 1.64
136         text = "N"
137     intervals [17]:
138         xmin = 1.64
139         xmax = 1.71
140         text = "AH0"
141     intervals [18]:
142         xmin = 1.71
143         xmax = 1.84
144         text = "S"
145     intervals [19]:
146         xmin = 1.84
147         xmax = 1.94
148         text = "AH0"
149     intervals [20]:
150         xmin = 1.94
151         xmax = 2.02
152         text = "N"
153     intervals [21]:
154         xmin = 2.02
155         xmax = 2.18
156         text = "S"
157     intervals [22]:
158         xmin = 2.18
159         xmax = 2.31
160         text = ""
161     intervals [23]:
162         xmin = 2.31
163         xmax = 2.55
164         text = "A01"
165     intervals [24]:
166         xmin = 2.55
167         xmax = 2.67
168         text = "R"
169     intervals [25]:
170         xmin = 2.67
171         xmax = 2.77
172         text = "V"
173     intervals [26]:
174         xmin = 2.77
175         xmax = 2.89
176         text = "AE1"
177     intervals [27]:
178         xmin = 2.89
```

```
179         xmax = 2.94
180         text = "N"
181     intervals [28]:
182         xmin = 2.94
183         xmax = 3.0
184         text = "AHO"
185     intervals [29]:
186         xmin = 3.0
187         xmax = 3.06
188         text = "T"
189     intervals [30]:
190         xmin = 3.06
191         xmax = 3.13
192         text = "IYO"
193     intervals [31]:
194         xmin = 3.13
195         xmax = 3.26
196         text = "F"
197     intervals [32]:
198         xmin = 3.26
199         xmax = 3.43
200         text = "EH1"
201     intervals [33]:
202         xmin = 3.43
203         xmax = 3.68
204         text = "R"
205     intervals [34]:
206         xmin = 3.68
207         xmax = 3.830476
208         text = ""
```

8.2 Gradient Reversal Layer

```
1     class GradientReversalLayer(Function):
2         @staticmethod
3         def forward(ctx, x):
4             ctx.save_for_backward(x)
5             return x
6
7         @staticmethod
8         def backward(ctx, grad_output):
9             return -grad_output
```

8.3 Speaker Classifier

```

1 class SpeakerClassifier(nn.Module):
2     def __init__(self, embedding_dim, num_speakers):
3         super(SpeakerClassifier, self).__init__()
4         # num_speakers = 4
5         self.fc = nn.Linear(embedding_dim, num_speakers)
6
7     def forward(self, x):
8         return self.fc(x)

```

8.4 Style Classifier

```

1 class StyleClassifier(nn.Module):
2     def __init__(self, embedding_dim, num_styles):
3         super(StyleClassifier, self).__init__()
4         # num_styles = 7
5         self.fc = nn.Linear(embedding_dim, num_styles)
6
7     def forward(self, x):
8         return self.fc(x)

```

8.5 MINE Model

```

1 class MINE(nn.Module):
2     def __init__(self, x_dim, y_dim, hidden_size):
3         super(MINE, self).__init__()
4         self.T_func = nn.Sequential(
5             nn.Linear(x_dim + y_dim, hidden_size),
6             nn.ReLU(),
7             nn.Linear(hidden_size, hidden_size),
8             nn.ReLU(),
9             nn.Linear(hidden_size, 1)
10        )
11
12    def mi_est(self, x_samples, y_samples):
13        return self.dkl(x_samples, y_samples)
14
15    def dkl(self, x_samples, y_samples):
16        sample_size = y_samples.shape[0]
17        random_index = torch.randperm(sample_size)
18        y_shuffle = y_samples[random_index]
19        T0 = self.T_func(torch.cat([x_samples, y_samples], dim=-1))
20        T1 = self.T_func(torch.cat([x_samples, y_shuffle], dim=-1))
21        T1_max = torch.max(T1)

```



```

22         lower_bound = T0.mean()-(T1_max+torch.log(torch.mean(torch.exp(T1-T1_max))))
23         return lower_bound

```

8.6 INFO_NCE Model

```

1 class InfoNCE(nn.Module):
2     def __init__(self, x_dim, y_dim, hidden_size):
3         super(InfoNCE, self).__init__()
4         self.F_func = nn.Sequential(
5             nn.Linear(x_dim + y_dim, hidden_size),
6             nn.ReLU(),
7             nn.Linear(hidden_size, hidden_size),
8             nn.ReLU(),
9             nn.Linear(hidden_size, 1),
10            nn.Softplus()
11        )
12
13    def mi_est(self, x_samples, y_samples):
14        sample_size = y_samples.shape[0]
15        x_tile = x_samples.unsqueeze(0).repeat((sample_size, 1, 1))
16        y_tile = y_samples.unsqueeze(1).repeat((1, sample_size, 1))
17        T0 = self.F_func(torch.cat([x_samples,y_samples], dim = -1))
18        T1 = self.F_func(torch.cat([x_tile, y_tile], dim = -1))
19        lower_bound = T0.mean() - (T1.logsumexp(dim = 1).mean() - np.log(sample_size))
20        return lower_bound

```

8.7 CLUB Model

```

1 class CLUB(nn.Module):
2     def __init__(self, x_dim, y_dim, hidden_size):
3         super(CLUB, self).__init__()
4         # p_mu outputs mean of q(Y|X)
5         self.p_mu = nn.Sequential(
6             nn.Linear(x_dim, hidden_size // 2),
7             nn.ReLU(),
8             nn.Linear(hidden_size // 2, y_dim)
9         )
10        # p_logvar outputs log of variance of q(Y|X)
11        self.p_logvar = nn.Sequential(
12            nn.Linear(x_dim, hidden_size // 2),
13            nn.ReLU(),
14            nn.Linear(hidden_size // 2, y_dim),
15            nn.Tanh()
16        )

```

```

17
18 def get_mu_logvar(self, x_samples):
19     mu = self.p_mu(x_samples)
20     logvar = self.p_logvar(x_samples)
21     return mu, logvar
22
23 def mi_est(self, x_samples, y_samples):
24     mu, logvar = self.get_mu_logvar(x_samples)
25
26     # log of conditional probability of positive sample pairs
27     positive = - (mu - y_samples)**2 / (2.0 * logvar.exp() + 1e-6)
28
29     prediction_1 = mu.unsqueeze(1)
30     y_samples_1 = y_samples.unsqueeze(0)
31
32     # log of conditional probability of negative sample pairs
33     negative = - ((y_samples_1 - prediction_1)**2).mean(dim=1) / (2.0 * logvar.exp() + 1e-6)
34
35
36 def loglikeli(self, x_samples, y_samples):
37     # unnormalized loglikelihood
38     mu, logvar = self.get_mu_logvar(x_samples)
39     return -(mu - y_samples)**2 / (logvar.exp() + 1e-6) - logvar.sum(dim=1).mean(dim=0)

```

8.8 Convex Conjugate Rényi Model

```

1 class CCR(nn.Module):
2     def __init__(self, x_dim, y_dim, hidden_size):
3         super(CCR, self).__init__()
4         self.T_func = nn.Sequential(
5             nn.Linear(x_dim + y_dim, hidden_size),
6             nn.ReLU(),
7             nn.Linear(hidden_size, hidden_size),
8             nn.ReLU(),
9             nn.Linear(hidden_size, 1)
10        )
11
12    def mi_est(self, x_samples, y_samples):
13        return self.conjugate(x_samples, y_samples)
14
15    def conjugate(self, x_samples, y_samples):
16        sample_size = y_samples.shape[0]
17        random_index = torch.randperm(sample_size)
18        y_shuffle = y_samples[random_index]
19        T0 = self.T_func(torch.cat([x_samples, y_samples], dim=-1))
20        T1 = self.T_func(torch.cat([x_samples, y_shuffle], dim=-1))
21        T_max = torch.max(torch.max(T0), torch.max(T1)).to(x_samples.device)

```

```

22     T0_shifted = T0 - T_max
23     T1_shifted = T1 - T_max
24     # Convex-Conjugate Renyi Variational Formula
25     alpha = torch.tensor(2.0, device=x_samples.device)
26     epsilon = torch.tensor(1e-8, device=x_samples.device)
27     term1 = T0_shifted.mean()
28     term2_intermediate = torch.pow(
29         torch.abs(T1_shifted)+ epsilon, (alpha - 1) / alpha
30     )
31     term2 = (1.0 / (alpha - 1.0))*torch.log(
32         torch.mean(term2_intermediate) + epsilon
33     )
34     term3 = torch.pow(alpha, -1) * (torch.log(alpha + epsilon) + 1)
35     cc_renyi_divergence = term1 + term2 + term3
36     return cc_renyi_divergence

```

8.9 Worst Case Regret Model

```

1     class WCR(nn.Module):
2         def __init__(self, x_dim, y_dim, hidden_size):
3             super(WCR, self).__init__()
4             self.T_func = nn.Sequential(
5                 nn.Linear(x_dim + y_dim, hidden_size),
6                 nn.ReLU(),
7                 nn.Linear(hidden_size, hidden_size),
8                 nn.ReLU(),
9                 nn.Linear(hidden_size, 1)
10            )
11
12        def mi_est(self, x_samples, y_samples):
13            return self.worst_case(x_samples, y_samples)
14
15        def worst_case(self, x_samples, y_samples):
16            sample_size = y_samples.shape[0]
17            random_index = torch.randperm(sample_size)
18            y_shuffle = y_samples[random_index]
19            T0 = self.T_func(torch.cat([x_samples, y_samples], dim=-1))
20            T1 = self.T_func(torch.cat([x_samples, y_shuffle], dim=-1))
21            T_max = torch.max(torch.max(T0), torch.max(T1)).to(x_samples.device)
22            T0_shifted = T0 - T_max
23            T1_shifted = T1 - T_max
24            # Worst-case Regret Variational Formula
25            alpha = torch.tensor(2.0, device=x_samples.device)
26            term1 = T0.mean()
27            term1 = T0_shifted.mean()
28            term2 = torch.log(torch.mean(torch.abs(T1))) + 1

```

```

29     worst_case_regret_div = term1 + term2
30     return worst_case_regret_div

```

8.10 Cosine Similarity Matrices

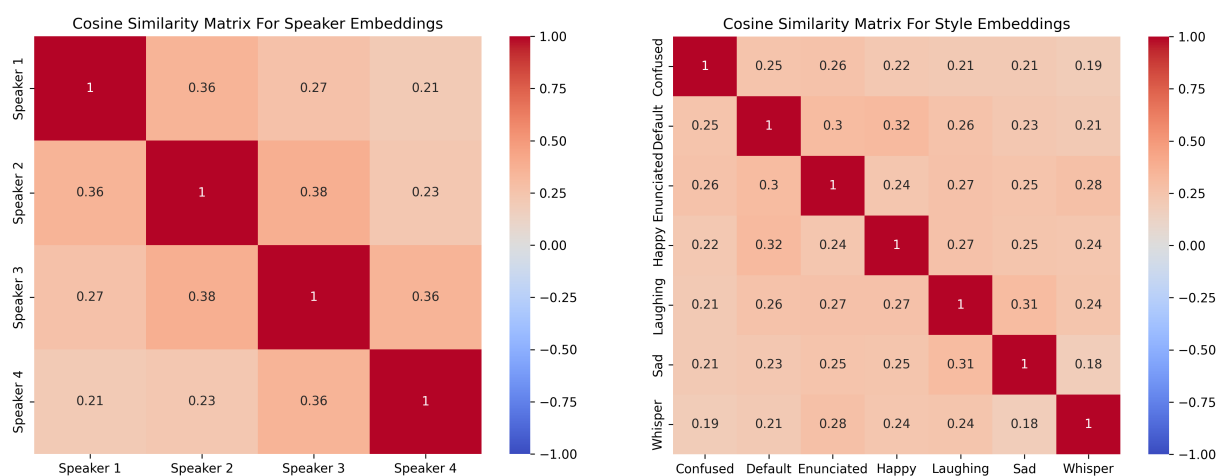


Figure 8.1: Cosine Similarity Matrix for Speakers (left) and Styles (right) using the MINE model.

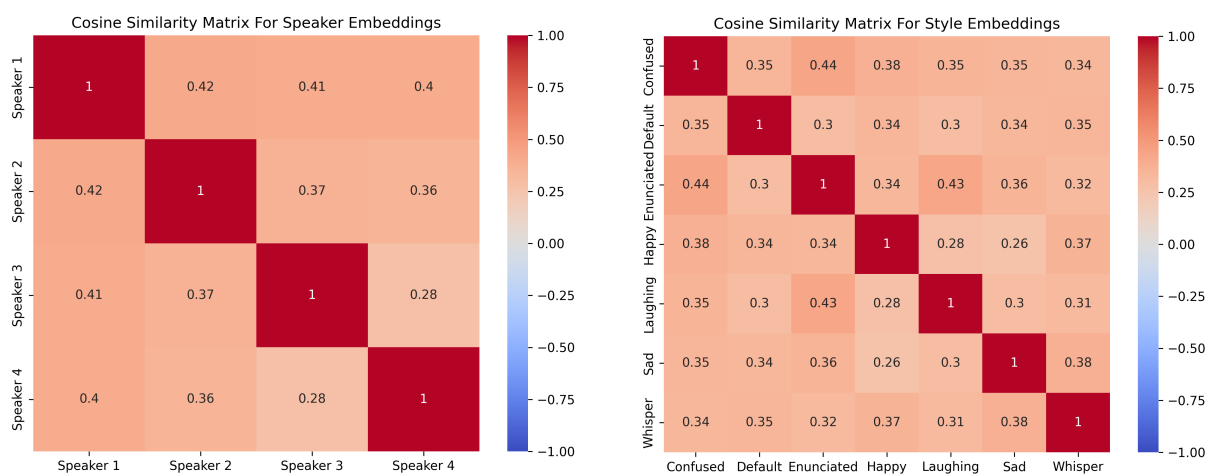


Figure 8.2: Cosine Similarity Matrix for Speakers (left) and Styles (right) using the INFO_NCE model.

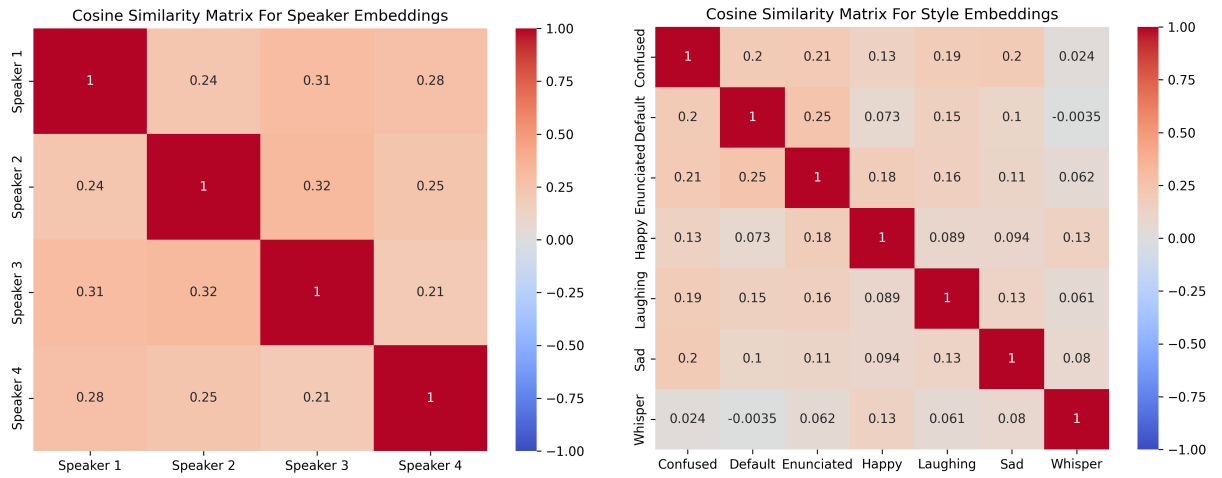


Figure 8.3: Cosine Similarity Matrix for Speakers (left) and Styles (right) using the Gradient Reversal Layer model.

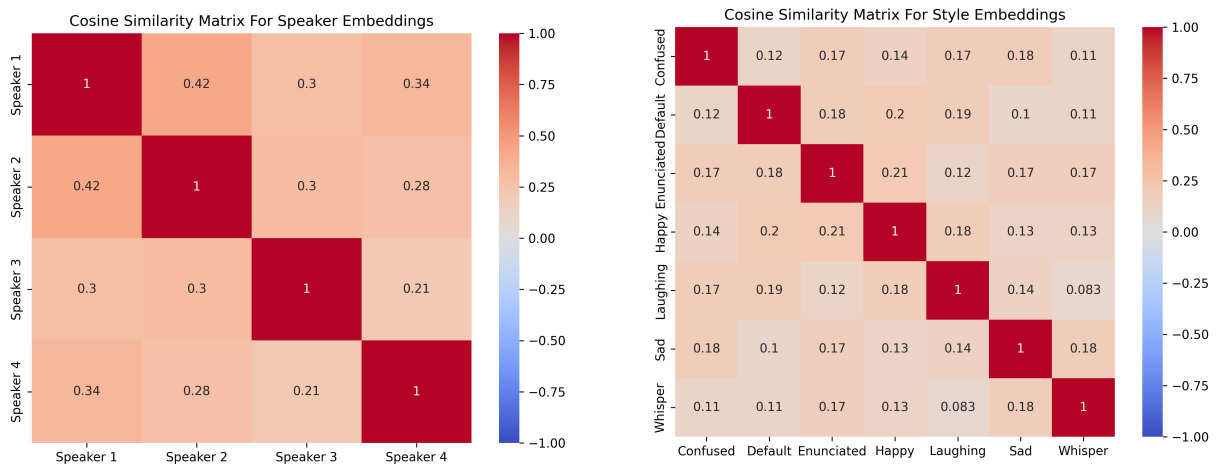


Figure 8.4: Cosine Similarity Matrix for Speakers (left) and Styles (right) using the Worst Case Regret model.

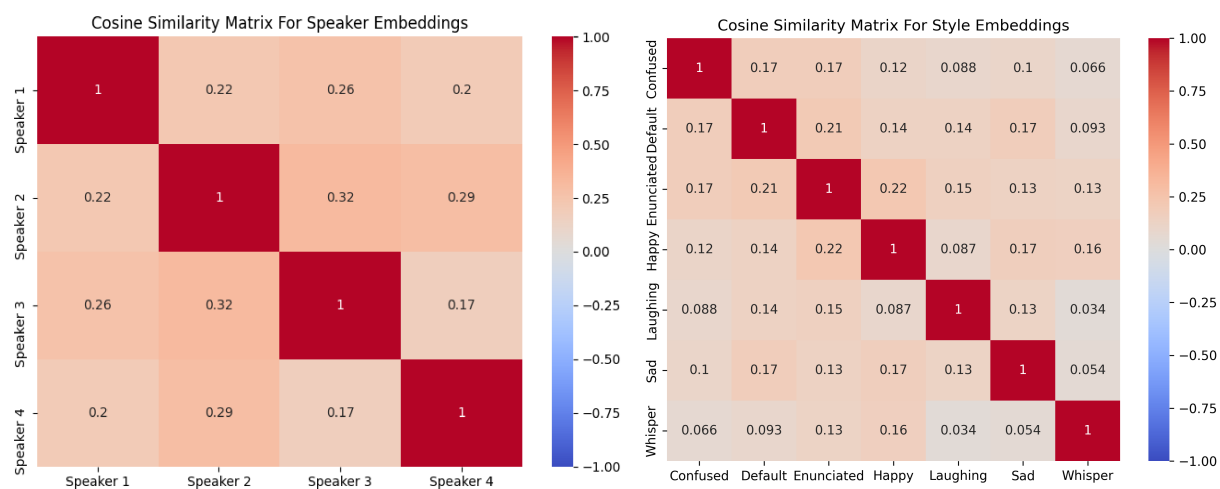


Figure 8.5: Cosine Similarity Matrix for Speakers (left) and Styles (right) using the Convex Conjugate Rényi model.

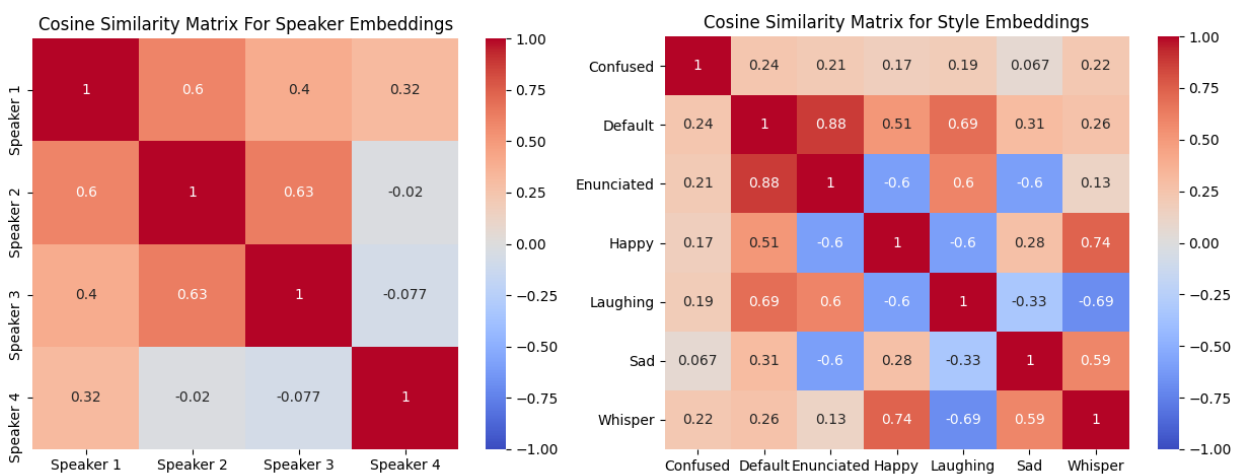


Figure 8.6: Cosine Similarity Matrix for Speakers (left) and Styles (right) using the CLUB model.

8.11 PCA Plots for Style & Speaker Embeddings

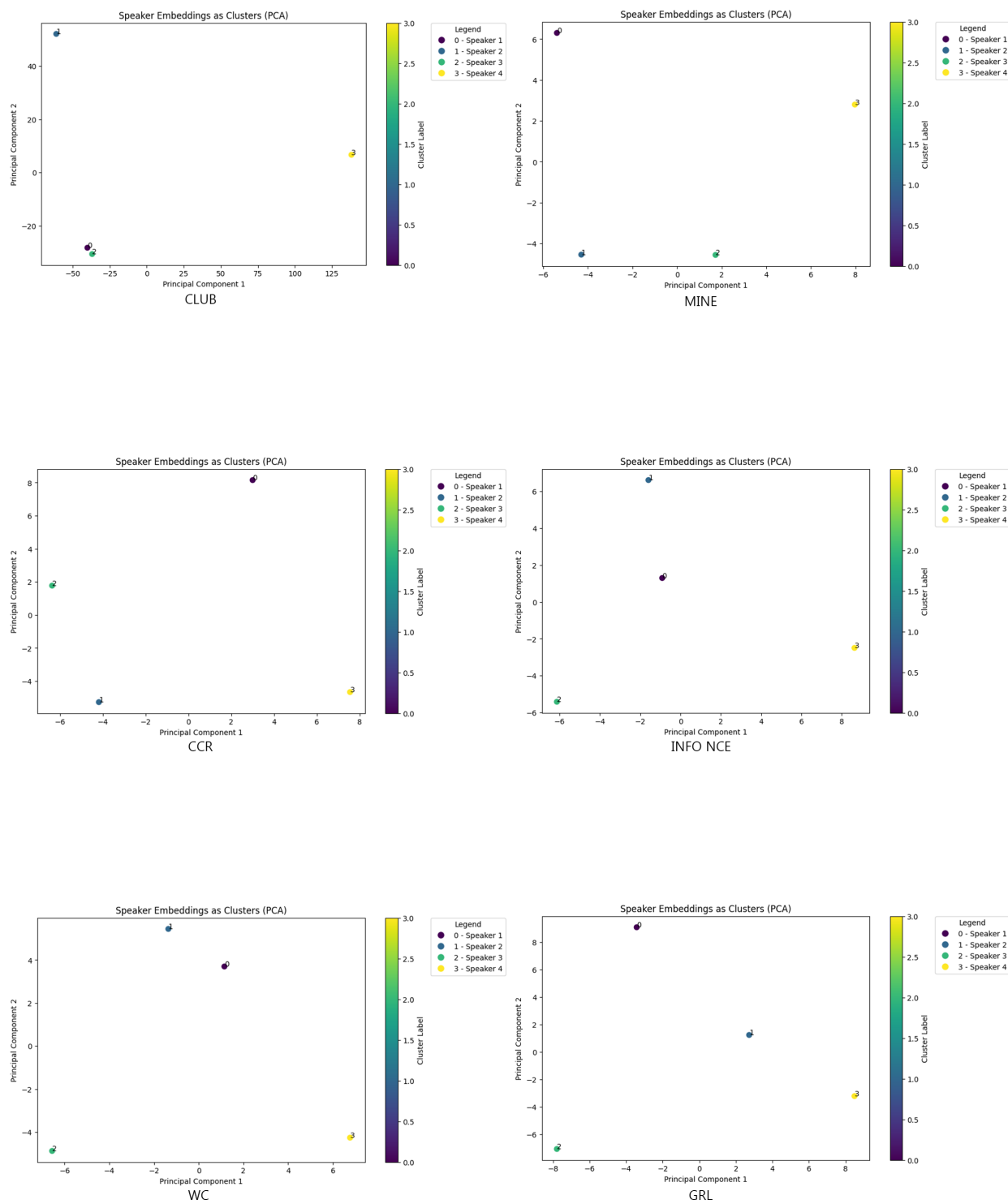


Figure 8.7: Comparison of Speaker Embeddings Across Different Models Using

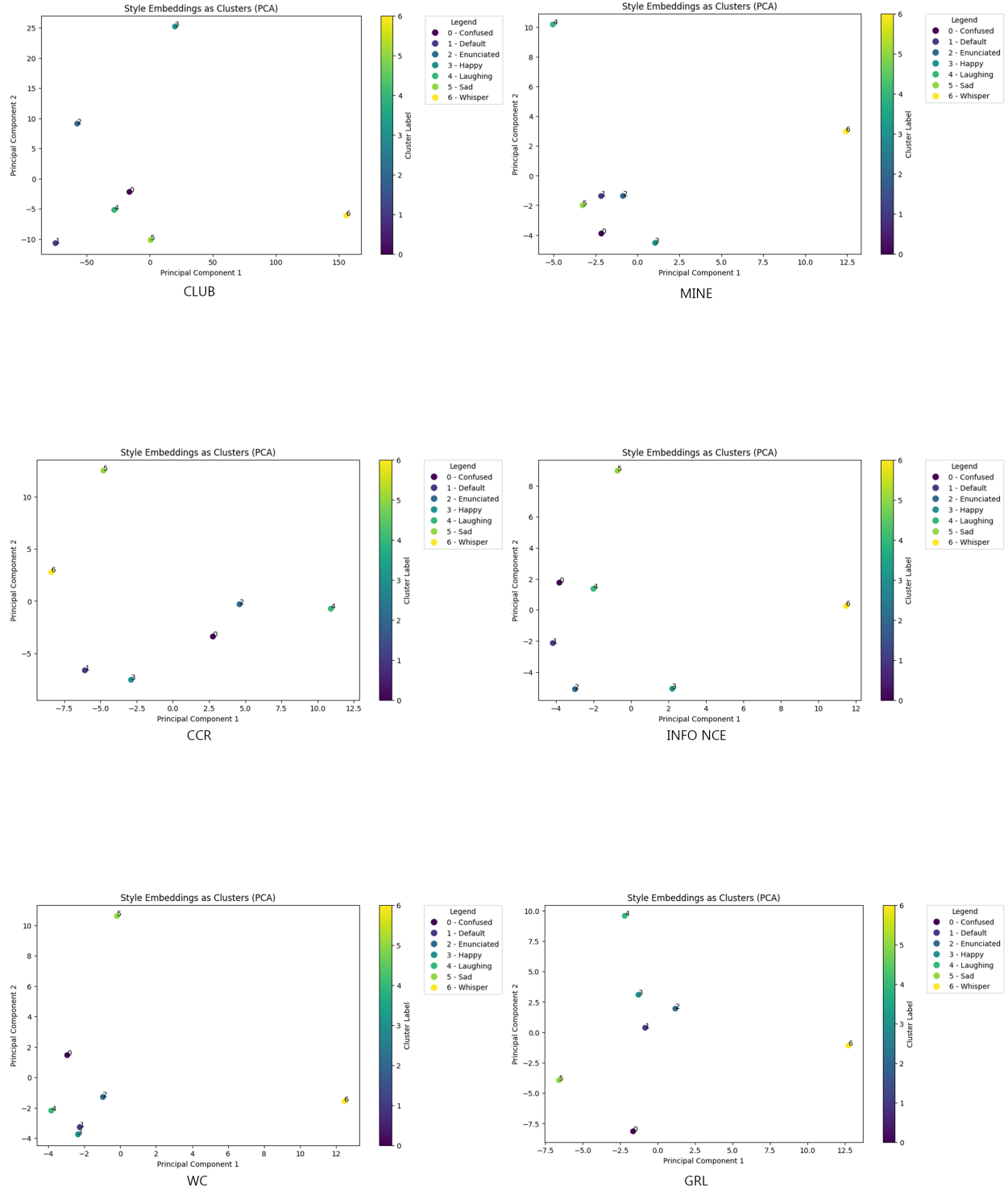


Figure 8.8: Comparison of Style Embeddings Across Different Models Using PCA.

8.12 GPU Details

```

+-----+
| NVIDIA-SMI 450.119.03   Driver Version: 450.119.03   CUDA Version: 11.0   |
+-----+-----+-----+-----+-----+-----+
| GPU  Name          Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf    Pwr:Usage/Cap|      Memory-Usage | GPU-Util  Compute M. |
|                                           MIG M.         |
+-----+-----+-----+-----+-----+-----+
|   0   GeForce RTX 208...  Off   | 00000000:08:00:0  Off   |                     N/A |
| 29%   42C    P0     49W / 250W |    0MiB / 11019MiB |      1%    Default |
|                                           |                     N/A |
+-----+-----+-----+-----+-----+-----+
|   1   GeForce RTX 208...  Off   | 00000000:41:00:0  Off   |                     N/A |
| 35%   47C    P0     56W / 250W |    0MiB / 11019MiB |      0%    Default |
|                                           |                     N/A |
+-----+-----+-----+-----+-----+-----+
|   2   GeForce RTX 208...  Off   | 00000000:42:00:0  Off   |                     N/A |
| 38%   50C    P0     44W / 250W |    0MiB / 11019MiB |      0%    Default |
|                                           |                     N/A |
+-----+-----+-----+-----+-----+-----+

```

8.13 requirements.txt

```

1  absl-py (1.4.0)
2  appdirs (1.4.4)
3  audioread (3.0.1)
4  cachetools (4.2.4)
5  certifi (2024.2.2)
6  cffi (1.15.1)
7  charset-normalizer (2.0.12)
8  click (8.0.4)
9  cycpler (0.11.0)
10 Cython (3.0.10)
11 dataclasses (0.8)
12 decorator (5.1.1)
13 Distance (0.1.3)
14 future (1.0.0)
15 g2p-en (2.1.0)
16 google-auth (1.35.0)
17 google-auth-oauthlib (0.4.6)
18 grpcio (1.48.2)
19 idna (3.7)
20 importlib-metadata (4.8.3)
21 importlib-resources (5.4.0)
22 inflect (4.1.0)

```

```
23  joblib (1.1.1)
24  kiwisolver (1.3.1)
25  librosa (0.7.2)
26  llvmlite (0.31.0)
27  Markdown (3.3.7)
28  matplotlib (3.2.2)
29  nltk (3.6.7)
30  numba (0.48.0)
31  numpy (1.19.5)
32  oauthlib (3.2.2)
33  packaging (21.3)
34  Pillow (8.3.2)
35  pip (9.0.1)
36  pkg-resources (0.0.0)
37  pooch (1.6.0)
38  protobuf (3.19.6)
39  pyasn1 (0.5.1)
40  pyasn1-modules (0.3.0)
41  pycparser (2.21)
42  pyparsing (3.1.2)
43  pypinyin (0.51.0)
44  python-dateutil (2.9.0.post0)
45  pyworld (0.3.4)
46  PyYAML (5.4.1)
47  regex (2023.8.8)
48  requests (2.27.1)
49  requests-oauthlib (2.0.0)
50  resampy (0.4.3)
51  rsa (4.9)
52  scikit-learn (0.23.2)
53  scipy (1.5.0)
54  setuptools (59.5.0)
55  six (1.16.0)
56  SoundFile (0.10.3.post1)
57  tensorboard (2.2.2)
58  tensorboard-data-server (0.6.1)
59  tensorboard-plugin-wit (1.8.1)
60  tgt (1.4.4)
61  threadpoolctl (3.1.0)
62  torch (1.10.1)
63  torchaudio (0.10.1)
64  tqdm (4.46.1)
65  typing-extensions (4.1.1)
66  Unidecode (1.1.1)
67  urllib3 (1.26.18)
68  Werkzeug (2.0.3)
69  wheel (0.37.1)
70  zipp (3.6.0)
```

8.14 Sentence Examples for Word Error Rate (WER) Evaluation

1. The sky turned pink as the sun set behind the mountains.
2. The dog bark at the mailman.
3. He placed the book gently on the dusty shelf.
4. The dog barked excitedly at the passing cars.
5. What goes up but never comes down?
6. The fish swam in the clear water.
7. The stormy sea roared against the rocky shore.
8. They cheered when the final whistle blew.
9. The stars twinkled in the clear night sky.
10. She whispered a secret into the dark night.
11. The wind howled through the trees all night long.
12. He read a book before going to bed.
13. They danced in the rain without a care.
14. The old man sighed and closed his eyes.
15. The clock ticked loudly in the silent room.
16. She painted the sky in shades of blue and purple.
17. What has keys but can't open doors?
18. He ate an apple for his snack.
19. The cake was soft and sweet, melting in her mouth.
20. The baby giggled as the puppy chased its tail.
21. The sun peeked through the clouds after the rain.
22. She placed the last piece of the puzzle in place.
23. the cat jumped onto the couch.
24. The day ended as quietly as it had begun with a soft breeze.
25. The water in the lake was calm and still.
26. What gets wetter as it dries?
27. The leaves crunched under her boots as she walked.
28. The phone rang, but no one was on the other end.
29. The moonlight glowed softly on the forest floor.
30. She held the fragile flower in her hand.
31. The lion roared, shaking the ground beneath them.

32. He scribbled a quick note before rushing out the door.
33. The stars fell like confetti from the sky.
34. The train whistled as it pulled into the station.
35. The air smelled of rain and fresh earth.
36. The kitten purred as it snuggled in her lap.
37. The river flowed gently through the valley.
38. The paper airplane soared across the classroom.
39. She carefully folded the letter and placed it in her pocket.
40. The old oak tree swayed gently in the evening breeze.
41. She glanced at her watch and hurried down the street.
42. A penny saved is a penny earned, they say.
43. The old swing creaked as it swayed in the breeze.
44. His fingers danced over the piano keys with ease.
45. The mountain loomed in the distance, tall and proud.
46. What has legs but cannot walk?
47. The candle flickered as the wind blew through the window.
48. The baby's first steps were met with cheers and smiles.
49. The city lights twinkled like stars in the distance.
50. The sound of the ocean calmed her restless mind.

8.15 Sentence Examples for STOI & PESQ Evaluation

1. Monday, there's gonna be haze.
2. Go to hell.
3. Nothing like a good laugh!
4. Why aren't you up there objecting?
5. Hold the elevator, please!
6. Bring popcorn and a blanket!
7. Who am I to be modest?
8. So it's like someone gives you a horse.
9. What's the best time to prune azalea?
10. Yes, tomorrow in Gorges there should be a light drizzle.
11. That was an amazing book!
12. Who's the wealthiest YouTuber?
13. Let her have her say.

14. Happy reading!
15. The advances in technology.
16. Who sent the note?
17. Rough night at the casino tables?
18. She kept saying what?
19. So what'll it be?
20. Why did the teddy bear not want dessert?
21. It's already in Montreal, remember?
22. No, snow is not expected on Saturday.
23. There's an alarm tomorrow at twelve fifteen AM.
24. In Rome this evening, it's gonna be partly cloudy.
25. Remain on Alma street for three miles.
26. Setting seven timers, What timer duration do you want?
27. Comparing and contrasting the various offender types is illuminating.
28. Temperatures should go from thirteen to five degrees.
29. Your alarm is tomorrow at nine thirty PM.
30. There's nothing to worry about.
31. The Flying Scotsman has Johnny Lee Miller in it and it's about a bobsledding team.
32. Why should he go anywhere?
33. Set up and get the puck!
34. Did Tierra deserve her bad girl reputation?
35. Find it now, before it disappears and somebody accuses you of suppressing evidence.
36. Blimy are you threatening me?
37. She is well balanced.
38. Sorry, I still do not have that information.
39. So that's it?
40. Yes, The King's Speech is popular.

Original Text	CCR & GRL	CCR	WCR	MINE	INFO	CLUB	GRL
---------------	-----------	-----	-----	------	------	------	-----

8.16 STOI Results Across Different Models

Original Text	CCR & GRL	CCR	WCR	MINE	INFO	CLUB	GRL
Monday, there's gonna be haze.	0.3591	0.3351	0.2591	0.2131	0.1319	0.1650	0.2198
Go to hell.	0.1604	0.1434	0.1142	0.1006	0.1130	0.0764	0.1487
Nothing like a good laugh!	0.1982	0.1859	0.0691	0.1210	0.1037	0.1205	0.1839
Why aren't you up there objecting?	0.5105	0.4601	0.3179	0.4171	0.0635	0.3983	0.2716
Hold the elevator, please!	0.2189	0.1034	0.1358	0.1570	0.0153	0.0909	0.1726
Bring popcorn and a blanket!	0.1067	1.0945	1.0908	0.0054	0.0074	0.0391	0.1054
Who am I to be modest?	0.3094	0.2100	0.3021	0.2187	0.1881	0.2659	0.3087
So it's like someone gives you a horse.	0.4156	0.3507	0.2614	0.3007	0.1953	0.1882	0.3738
What's the best time to prune azalea?	0.1403	0.1297	0.0845	0.0747	0.0947	0.1060	0.0560
Yes, tomorrow in Gorges there should be a light drizzle.	0.4329	0.3198	0.3088	0.2549	0.3413	0.3058	0.2500
That was an amazing book!	0.2634	0.1442	0.1437	0.1141	0.0935	0.1128	0.1336
Who's the wealthiest YouTuber?	0.2077	0.1896	0.0789	0.1541	0.1743	0.1556	0.1293
Let her have her say.	0.2378	0.1220	0.1872	0.0697	0.2261	0.0729	0.1903
Happy reading!	0.2588	0.1104	0.0843	0.0788	0.1586	0.0431	0.0894
The advances in technology.	0.1972	0.1884	0.1880	0.0358	0.0717	0.0655	0.0901
Who sent the note?	0.3971	0.0494	0.2217	0.2687	0.2877	0.0107	0.0983
Rough night at the casino tables?	0.4134	0.3690	0.3636	0.4346	0.4733	0.2653	0.3678
She kept saying what?	0.6284	0.5350	0.4070	0.3007	0.3889	0.3761	0.2546
So what'll it be?	0.3174	0.1653	0.2553	0.2961	0.2643	0.2621	0.2598
Why did the teddy bear not want dessert?	0.4447	0.3691	0.4143	0.2569	0.4835	0.3291	0.4052
It's already in Montreal, remember?	0.2633	0.1798	0.1767	0.2705	0.2139	0.1926	0.1642
No, snow is not expected on Saturday.	0.5465	0.4917	0.3142	0.5159	0.4637	0.2175	0.4720
There's an alarm tomorrow at twelve fifteen AM.	0.3093	0.2655	0.2442	0.1571	0.1621	0.2294	0.1999
In Rome this evening, it's gonna be partly cloudy.	0.3395	0.1799	0.1527	0.3643	0.2414	0.2318	0.3263
Remain on Alma street for three miles.	0.2976	0.2227	0.2719	0.2435	0.2677	0.2527	0.2598

Original Text	CCR & GRL	CCR	WCR	MINE	INFO	CLUB	GRL
Setting seven timers, What timer duration do you want?	0.3076	0.4105	0.3752	0.2237	0.3915	0.1870	0.2919
Comparing and contrasting the various offender types is illuminating.	0.3443	0.2143	0.2448	0.4017	0.2143	0.1231	0.2054
Temperatures should go from thirteen to five degrees.	0.3752	0.3246	0.3723	0.2311	0.3489	0.3346	0.3420
Your alarm is tomorrow at nine thirty PM.	0.3318	0.1597	0.1310	0.4728	0.1547	0.2398	0.3953
There's nothing to worry about.	0.1075	0.1782	0.1903	0.1247	0.1397	0.1800	0.2113
The Flying Scotsman has Johnny Lee Miller in it and it's about a bobsled-ding team.	0.1731	0.1158	0.1061	0.2376	0.1512	0.1819	0.1678
Why should he go anywhere?	0.1430	0.1888	0.1515	0.1430	0.2812	0.1566	0.2506
Set up and get the puck!	0.2868	0.1215	0.2022	0.2171	0.2739	0.1922	0.2272
Did Tierra deserve her bad girl reputation?	0.1195	0.1189	0.1633	0.1342	0.1641	0.1313	0.1506
Find it now, before it disappears and somebody accuses you of suppressing evidence.	0.2989	0.1695	0.1595	0.2266	0.1830	0.1501	0.2402
Blimy are you threatening me?	0.3869	0.3771	0.2569	0.1224	0.2993	0.2106	0.1119
She is well balanced.	0.2539	0.2704	0.2476	0.2452	0.3039	0.2248	0.1950
Sorry, I still do not have that information.	0.1819	0.1336	0.1471	0.1715	0.0952	0.1557	0.1692
So that's it?	0.3390	0.2657	0.2503	0.2527	0.2670	0.3106	0.2761
Yes, The King's Speech is popular.	0.4201	0.1852	0.0721	0.0227	0.0613	0.1049	0.3214

Table 8.1: STOI Results Across Different Models for Each Sentence

8.17 PESQ Results Across Different Models

Original Text	CCR & GRL	CCR	WCR	MINE	INFO	CLUB	GRL
Monday, there's gonna be haze.	1.2384	1.2056	0.1847	0.1685	0.1455	0.1974	0.1854
Go to hell.	1.3527	1.3169	1.2845	1.1995	1.2204	1.1603	1.3065
Nothing like a good laugh!	1.1931	1.1677	1.1369	1.1367	1.0922	1.1968	1.1209
Why aren't you up there objecting?	1.0637	1.0456	1.0583	1.0398	1.0397	1.0452	1.037

Original Text	CCR & GRL	CCR	WCR	MINE	INFO	CLUB	GRL
Hold the elevator, please!	1.1431	1.0496	1.1196	1.1099	1.0783	1.0788	1.0518
Bring popcorn and a blanket!	1.1715	1.1387	1.1647	1.0910	1.1718	1.1262	1.1385
Who am I to be modest?	1.1235	1.1028	1.1402	1.0503	1.1264	1.1461	1.0891
So it's like someone gives you a horse.	1.1168	1.0833	1.0734	1.0719	1.0797	1.0742	1.0972
What's the best time to prune azalea?	1.2813	1.1349	1.1203	1.0890	1.1893	1.2448	1.1972
Yes, tomorrow in Gorges there should be a light drizzle.	1.0440	1.0349	1.0410	1.0400	1.0761	1.0362	1.0390
That was an amazing book!	1.0749	1.0621	1.0592	1.0770	1.0426	1.0646	1.0681
Who's the wealthiest YouTuber?	1.0392	1.1222	1.0807	1.0441	1.0620	1.0567	1.0550
Let her have her say.	1.2430	1.1715	1.1524	1.1727	1.1805	1.1860	1.1356
Happy reading!	1.1992	1.1113	1.1129	1.1622	1.1478	1.1433	1.1281
The advances in technology.	1.0416	1.0283	1.0267	1.0305	1.0327	1.0315	1.0402
Who sent the note?	1.0722	1.0762	1.0635	1.0724	1.0729	1.1376	1.0786
Rough night at the casino tables?	1.0665	1.0604	1.0522	1.0450	1.0654	1.0829	1.0603
She kept saying what?	1.2687	1.0869	1.0822	1.2637	1.3134	1.2418	1.0629
So what'll it be?	1.0533	1.0595	1.0393	1.0575	1.0332	1.0714	1.0592
Why did the teddy bear not want dessert?	1.0686	1.0424	1.0475	1.0483	1.0524	1.0519	1.0373
It's already in Montreal, remember?	1.0863	1.0299	1.0484	1.0451	1.0347	1.0335	1.0289
No, snow is not expected on Saturday.	1.0506	1.0390	1.0480	1.0370	1.0448	1.0585	1.0355
There's an alarm tomorrow at twelve fifteen AM.	1.3022	1.0417	1.0438	1.0417	1.0421	1.0424	1.0435
In Rome this evening, it's gonna be partly cloudy.	2.7599	2.3898	1.9109	1.8023	1.6980	2.1302	2.7171
Remain on Alma street for three miles.	1.0267	1.0179	1.0219	1.0220	1.0250	1.0201	1.0177
Setting seven timers, What timer duration do you want?	1.0604	1.0307	1.0310	1.0711	1.0395	1.0428	1.0313
Comparing and contrasting the various offender types is illuminating.	1.0226	1.0262	1.0279	1.0203	1.0196	1.0098	1.0198
Temperatures should go from thirteen to five degrees.	1.0403	1.0499	1.0354	1.0404	1.0591	1.0316	1.0399
Your alarm is tomorrow at nine thirty PM.	1.0724	1.1152	1.0824	1.0789	1.1207	1.0857	1.0737

Original Text	CCR & GRL	CCR	WCR	MINE	INFO	CLUB	GRL
There's nothing to worry about.	1.0356	1.0524	1.0889	1.0365	1.1018	1.0777	1.0379
The Flying Scotsman has Johnny Lee Miller in it and it's about a bobsled-ding team.	1.0954	1.0810	1.2324	1.0595	1.0912	1.0591	1.0723
Why should he go anywhere?	1.0676	1.0521	1.0748	1.0547	1.0596	1.0914	1.0701
Set up and get the puck!	1.1147	1.1113	1.0679	1.1125	1.0665	1.0862	1.0684
Did Tierra deserve her bad girl reputation?	1.0939	1.0679	1.1779	1.0754	1.0652	1.2490	1.0539
Find it now, before it disappears and somebody accuses you of suppressing evidence.	1.0859	1.0565	1.0864	1.0709	1.0725	1.0727	1.0812
Blimy are you threatening me?	1.0469	1.0469	1.0529	1.0335	1.0497	1.0351	1.0466
She is well balanced.	1.1050	1.1101	1.0705	1.0986	1.0862	1.1275	1.0327
Sorry, I still do not have that information.	1.0834	1.0942	1.0802	1.1122	1.1029	1.1130	1.0808
So that's it?	1.0379	1.0559	1.0431	1.0319	1.0421	1.0455	1.0336
Yes, The King's Speech is popular.	1.0623	1.0790	1.0842	1.0727	1.1355	1.0761	1.0550

Table 8.2: Model PESQ scores across different models and sentences.

8.18 Word Error Rate - Convex Conjugate Rényi & GRL

Original	Transcribed	WER
The sky turned pink as the sun set behind the mountains.	the sky turn pink as the sun set behind the mountain	0.1818
The dog bark at the mailman.	the dog bark at the mailman	0.0000
He placed the book gently on the dusty shelf.	he plays the book gently on the dusty Shelf	0.1111
The dog barked excitedly at the passing cars.	the dog barked excitedly at the passing powers	0.1250
What goes up but never comes down?	what goes up but never comes down	0.0000
The fish swam in the clear water.	the fish swim in the clear	0.2857
The stormy sea roared against the rocky shore.	the stormy sea road against the rock	0.2500
They cheered when the final whistle blew.	they shared won the final whistle	0.4286

The stars twinkled in the clear night sky.	the stars twinkled in the clear night	0.1250
She whispered a secret into the dark night.	she whispered a secret into the dark night	0.0000
The wind howled through the trees all night long.	the wind howled through the trees all night	0.1111
He read a book before going to bed.	He read a book before going to bed	0.0000
They danced in the rain without a care.	the danced in the rain without a	0.2500
The old man sighed and closed his eyes.	the old man's side and closed his eyes	0.2500
The clock ticked loudly in the silent room.	the clock technology in the silent	0.3750
She painted the sky in shades of blue and purple.	she painted the sky in shades of blue and purple	0.0000
What has keys but can't open doors?	what has keys but can't open doors	0.0000
His footsteps echoed in the empty corridor.	his footsteps settled in the empty	0.2857
The cake was soft and sweet, melting in her mouth.	the cake was soft and sweet melting in her mouth	0.0000
The baby giggled as the puppy chased its tail.	the baby giggled as the poppy chased its	0.2222
The sun peeked through the clouds after the rain.	the sun peek through the clouds after the	0.2222
She placed the last piece of the puzzle in place.	she placed the last piece of the puzzle in place	0.0000
The cat jumped onto the couch.	the cat jumped onto the couch	0.0000
The day ended as quietly as it had begun with a soft breeze.	the day in the desert quietly as it had bacon with a soft bre	0.3846
The water in the lake was calm and still.	the water in the lake was calm and still	0.0000
What gets wetter as it dries?	what gets wetter as it dries	0.0000
The leaves crunched under her boots as she walked.	the leaves crunched under her boots as she walked	0.0000
The phone rang, but no one was on the other end.	the phone rang but no one was in the other	0.2727
The moonlight glowed softly on the forest floor.	the Moonlight glowed Softly on the forest floor	0.0000
She held the fragile flower in her hand.	she held the fragile flower in her hand	0.0000
The lion roared shaking the ground beneath them.	the lion guard shaking the ground beneath	0.2500
He scribbled a quick note before rushing out the door.	he scribbled a quick note before rushing out the door	0.0000

The stars fell like confetti from the sky.	the Stars felt like confetti from the	0.2500
The train whistled as it pulled into the station.	the train whistles as it pulled into the station	0.1111
The air smelled of rain and fresh earth.	bear smell the rain and fresh	0.6250
The kitten purred as it snuggled in her lap.	the kitten purred as it snuggled in her lap	0.0000
The river flowed gently through the valley.	the river flow gently through the valley	0.1429
The paper airplane soared across the classroom.	the paper airplane sort across the class	0.2827
She carefully folded the letter and placed it in her pocket.	she carefully folded the letter and placed it in her	0.0909
The treasure chest creaked open, revealing gold coins.	the treasure chest creaked open revealing gold coins	0.0000
She glanced at her watch and hurried down the street.	she glanced at her watch and hurried down the street	0.0000
A penny saved is a penny earned they say.	a penny saved is a penny and they say	0.1111
The old swing creaked as it swayed in the breeze.	golf swing Creek as its weight in the bre	0.6000
His fingers danced over the piano keys with ease.	his fingers danced over the piano keys with	0.0000
The mountain loomed in the distance tall and proud.	the mountain lived in the distance to all land	0.4444
What has legs but cannot walk?	what has legs but can't	0.3333
The candle flickered as the wind blew through the window.	the candle flicker does the wine blew through the window	0.3000
The baby's first steps were met with cheers and smiles.	the baby's first steps were met with chairs and smiles	0.2000
The city lights twinkled like stars in the distance.	the city lights twinkle lights stars in the distance	0.2222
The sound of the ocean calmed her restless mind.	the sound of the ocean called her Restless	0.2222

Table 8.3: WER CCR & GRL

8.19 Word Error Rate - Convex Conjugate Rényi

Original Text	Transcribed Text	WER
The sky turned pink as the sun set behind the mountains.	the sky turn pink as the sun set behind the mountain	0.1818
The dog bark at the mailman.	the dog bark at the mail	0.1667
He placed the book gently on the dusty shelf.	he plays the book gently on the dusty Shelf	0.1111

Original Text	Transcribed Text	WER
The dog barked excitedly at the passing cars.	the dog barked excitedly at the passing powers	0.1250
What goes up but never comes down?	what does up but never comes	0.2857
The fish swam in the clear water.	the fifth swam in the clear	0.2857
The stormy sea roared against the rocky shore.	the stormy sea Road against the Rocky	0.2500
They cheered when the final whistle blew.	they shared won the final whistle	0.4286
The stars twinkled in the clear night sky.	the stars twinkled in the clear night	0.1250
She whispered a secret into the dark night.	she whispered a secret into the dark	0.1250
The wind howled through the trees all night long.	the wine called through the trees all night	0.3333
He read a book before going to bed.	He read a book before going to bed	0.0000
They danced in the rain without a care.	the danced in the rain without a	0.2500
The old man sighed and closed his eyes.	the old man's side and closed his eyes	0.2500
The clock ticked loudly in the silent room.	the clock to cloudy in the silent	0.3750
She painted the sky in shades of blue and purple.	she painted the sky in shades of blue and purple	0.0000
What has keys but can't open doors?	what has keys but can't open do	0.2857
His footsteps echoed in the empty corridor.	his footsteps settled in the empty	0.2857
The cake was soft and sweet, melting in her mouth.	the cake was soft and sweet melting in her mouth	0.0000
The baby giggled as the puppy chased its tail.	the baby giggled as the poppy chased its	0.2222
The sun peeked through the clouds after the rain.	the sun peek through the clouds after the	0.2222
She placed the last piece of the puzzle in place.	she plays the last piece of the puzzle in place	0.1000
The cat jumped onto the couch.	the cat jumped onto the	0.1667
The day ended as quietly as it had begun with a soft breeze.	The Day of the Dead quietly as it had been in the soft bre	0.5385
The water in the lake was calm and still.	the water in the lake was calm and	0.1111
What gets wetter as it dries?	what gets wetter as it dries	0.0000
The leaves crunched under her boots as she walked.	the leaves crunched under her boots as she walked	0.0000
The phone rang, but no one was on the other end.	the phone rang but no one was in the other	0.2727

Original Text	Transcribed Text	WER
The moonlight glowed softly on the forest floor.	the Moonlight Softly on the forest floor	0.1250
She held the fragile flower in her hand.	she held the fragile flower in her	0.125
The lion roared shaking the ground beneath them.	the lion guard shaking the ground beneath	0.2500
He scribbled a quick note before rushing out the door.	he scribbled a quick note before rushing out the door	0.0000
The stars fell like confetti from the sky.	the Stars felt like confetti from the	0.2500
The train whistled as it pulled into the station.	the train whistled as it pulled into the station	0.1111
The air smelled of rain and fresh earth.	they're small the rain and fresh	0.6250
The kitten purred as it snuggled in her lap.	the kitten purred as it snuggled in her lap	0.0000
The river flowed gently through the valley.	the river flow gently through the valley	0.1429
The paper airplane soared across the classroom.	the paper airplane sword across the classroom	0.1429
She carefully folded the letter and placed it in her pocket.	she carefully folded the letter and placed it in her	0.0909
The treasure chest creaked open, revealing gold coins.	the treasure chest creaked open revealing gold coins	0.0000
She glanced at her watch and hurried down the street.	she glanced at her watch and hurry down the	0.2000
A penny saved is a penny earned they say.	a penny saved is a penny and they say	0.1111
The old swing creaked as it swayed in the breeze.	adult Twin Creeks as its weight in the breeze	0.6000
His fingers danced over the piano keys with ease.	his fingers danced over the piano keys with	0.1111
The mountain loomed in the distance tall and proud.	the Martin wound in the distance to all and	0.5556
What has legs but cannot walk?	what is likes but cannot	0.5000
The candle flickered as the wind blew through the window.	the candle flicker does the wind blew through the window	0.2000
The baby's first steps were met with cheers and smiles.	the baby's first steps were met with chairs and smiles	0.2000
The city lights twinkled like stars in the distance.	the city lights twinkle lights stars in the distance	0.2222
The sound of the ocean calmed her restless mind.	the sound of the ocean called her Restless	0.2222

Table 8.4: WER CCR

8.20 Word Error Rate - Worst Case Regret

Original Text	Transcribed Text	WER
The sky turned pink as the sun set behind the mountains.	the sky turned pink as the sun set behind the mountain	0.3000
The dog bark at the mailman.	the dog bark at the	0.3333
He placed the book gently on the dusty shelf.	can you place the book gently on the dusty Shelf	0.3333
The dog barked excitedly at the passing cars.	the dog barked excitedly at the passing cars	0.0000
What goes up but never comes down?	what does up but never comes	0.2857
The fish swam in the clear water.	the thing swam in the clear	0.2857
The stormy sea roared against the rocky shore.	this to me see your word against the rocky Shore	0.7500
They cheered when the final whistle blew.	Richard won the final whistle	0.5714
The stars twinkled in the clear night sky.	the star twinkle twinkle	0.8750
She whispered a secret into the dark night.	cheap whisper to secret into the dark	0.5000
The wind howled through the trees all night long.	the White House for the trees all night	0.4444
He read a book before going to bed.	can you read a book before going to bed	0.2500
They danced in the rain without a care.	they danced in the rain without	0.2500
The old man sighed and closed his eyes.	old man's side and closed his eyes	0.3750
The clock ticked loudly in the silent room.	aquatic loudly in the silent	0.5000
She painted the sky in shades of blue and purple.	she painted the sky in shades of blue and purple	0.0000
What has keys but can't open doors?	what is keys but can't open doors	0.2857
His footsteps echoed in the empty corridor.	his footsteps settled in the empty	0.2857
The cake was soft and sweet melting in her mouth.	the cake was soft and sweet melting in her mouth	0.0000
The baby giggled as the puppy chased its tail.	the baby giggle does the puppy chase this	0.5556
The sun peeked through the clouds after the rain.	the song peek through the clouds after the rain	0.2222
She placed the last piece of the puzzle in place.	she plays the last piece of the puzzle in place	0.1000
The cat jumped onto the couch.	the cat jumped on the	0.3333
The day ended as quietly as it had begun with a soft breeze.	today and today is quietly as it had bacon with a soft bre	0.4615
The water in the lake was calm and still.	the water in the lake was calm	0.2222
What gets wetter as it dries?	what gets wetter as a	0.3333
The leaves crunched under her boots as she walked.	the leaves crunched under her but says she	0.3333

Original Text	Transcribed Text	WER
The phone rang, but no one was on the other end.	the phone ran but no one was another	0.5455
The moonlight glowed softly on the forest floor.	the Moonlight close Softly on the forest floor	0.1250
She held the fragile flower in her hand.	she held the fragile flower in her	0.1250
The lion roared shaking the ground beneath them.	the lion roar shaking the ground beneath	0.2500
He scribbled a quick note before rushing out the door.	he's crippled a quick note before rushing at the	0.4000
The stars fell like confetti from the sky.	the Stars felt like confetti from the	0.2500
The train whistled as it pulled into the station.	the train whistle does it pulled into the station	0.2222
The air smelled of rain and fresh earth.	bear smell the rain and fresh	0.6250
The kitten purred as it snuggled in her lap.	the kitten put as its snuggled in her	0.3333
The river flowed gently through the valley.	the river flow gently through the	0.2857
The paper airplane soared across the classroom.	the paper airplane sword across the classroom	0.1429
She carefully folded the letter and placed it in her pocket.	she carefully folded the letter and the place that in her Pok	0.3636
The treasure chest creaked open, revealing gold coins.	the treasure chest freaked open revealing gold card	0.2500
She glanced at her watch and hurried down the street.	she glanced at her watch and hurry down the street	0.1000
A penny saved is a penny earned they say.	a penny saved is a penny and they say	0.1111
The old swing creaked as it swayed in the breeze.	the old Twin Creek has its weight in the breeze	0.5000
His fingers danced over the piano keys with ease.	his fingers stand still over the piano keys with	0.3333
The mountain loomed in the distance tall and proud.	the mountain moved in the distance to all and	0.4444
What has legs but cannot walk?	what is legs but cannot walk	0.1667
The candle flickered as the wind blew through the window.	the candle flickered as the wine blew through the window	0.1000
The baby's first steps were met with cheers and smiles.	the baby's first steps were met with yours and smiles	0.2000
The city lights twinkled like stars in the distance.	the city lights twinkle lights stars in the distance	0.2222
The sound of the ocean calmed her restless mind.	the sound of ocean calmed her Restless	0.2222

Table 8.5: WER WC

8.21 Word Error Rate - CLUB

Original Text	Transcribed Text	WER
The sky turned pink as the sun set behind the mountains.	the sky turn pink has the sunset behind the mountain	0.4545
The dog bark at the mailman.	the dog bark at the mail	0.1667
He placed the book gently on the dusty shelf.	he plays the buck gently on the dusty Shelf	0.2222
The dog barked excitedly at the passing cars.	the dog barked excitedly at the passing	0.1250
What goes up but never comes down?	what does a banana comes down	0.5714
The fish swam in the clear water.	the fifth swam in the clear	0.2857
The stormy sea roared against the rocky shore.	the story against the Rocky	0.5000
They cheered when the final whistle blew.	nature when the final whistle	0.4286
The stars twinkled in the clear night sky.	the stars twinkle twinkle little star	0.7500
She whispered a secret into the dark night.	sheath whispered a secret into the dark	0.2500
The wind howled through the trees all night long.	the White House for the trees all night	0.4444
He read a book before going to bed.	he read a book before going to	0.1250
They danced in the rain without a care.	play dancing in the rain without	0.5000
The old man sighed and closed his eyes.	the old man's side and closest	0.6250
The clock ticked loudly in the silent room.	aquatic flowey in the silent	0.6250
She painted the sky in shades of blue and purple.	she painted the sky and shades of blue and purple	0.1000
What has keys but can't open doors?	what is but can't open	0.7143
His footsteps echoed in the empty corridor.	is but stop sicker than the empty	0.8571
The cake was soft and sweet melting in her mouth.	the cake was soft and sweet melting in her mouth	0.000
The baby giggled as the puppy chased its tail.	the baby giggle does the poppy Taste of	0.6667
The sun peeked through the clouds after the rain.	the sun peeked through the clouds after the	0.1111
She placed the last piece of the puzzle in place.	she plays the last piece of the puzzle and place	0.2000
The cat jumped onto the couch.	the cat jumped onto the	0.1667
The day ended as quietly as it had begun with a soft breeze.	with a soft Breeze	0.6923
The water in the lake was calm and still.	the water in the lake was called man	0.3333
What gets wetter as it dries?	what does Twitter as a	0.6667
The leaves crunched under her boots as she walked.	delete scratched under her boots as she	0.4444
The phone rang, but no one was on the other end.	the phone ring but no one was on	0.3636

Original Text	Transcribed Text	WER
The moonlight glowed softly on the forest floor.	the Moonlight Softly on the forest floor	0.1250
She held the fragile flower in her hand.	she held the fragile flower in her	0.1250
The lion roared shaking the ground beneath them.	the lion roars shaking the ground	0.3750
He scribbled a quick note before rushing out the door.	before rushing out the door	0.5000
The stars fell like confetti from the sky.	the Stars felt like confetti from the	0.2500
The train whistled as it pulled into the station.	train whistled as it pulled into the station	0.0000
The air smelled of rain and fresh earth.	bear smell the rain and fresh	0.6250
The kitten purred as it snuggled in her lap.	the kitten part as it snuggled in	0.3333
The river flowed gently through the valley.	the river flow gently through the valley	0.1429
The paper airplane soared across the classroom.	the paper airplanes would across the class	0.4286
She carefully folded the letter and placed it in her pocket.	she carefully folded the letter and placed it in her	0.0909
The treasure chest creaked open, revealing gold coins.	Patricia Chesapeake open reviewing gold	0.7500
She glanced at her watch and hurried down the street.	she glanced at her watch and hurried down the street	0.0000
A penny saved is a penny earned they say.	a penny saved is a penny and they	0.2222
The old swing creaked as it swayed in the breeze.	adult Twin Creeks as its weight in the breeze	0.6000
His fingers danced over the piano keys with ease.	his fingers danced over the piano keys with	0.1111
The mountain loomed in the distance tall and proud.	the mountains in the distance doll and	0.4444
What has legs but cannot walk?	what is legs but cannot	0.3333
The candle flickered as the wind blew through the window.	the candle flicker does the wine blew through the window	0.3000
The baby's first steps were met with cheers and smiles.	the baby's first steps were met with chairs and smile	0.3000
The city lights twinkled like stars in the distance.	the city lights twinkle black stars in the distance	0.2222
The sound of the ocean calmed her restless mind.	the sound of the ocean called the Restless	0.3333

Table 8.6: WER CLUB

8.22 Word Error Rate - INFO_NCE

Original Text	Transcribed Text	WER
The sky turned pink as the sun set behind the mountains.	the sky turn pink has the sunset behind the	0.4545
The dog bark at the mailman.	the dog bark at the	0.1667
He placed the book gently on the dusty shelf.	he plays the book gently on the dusty	0.2222
The dog barked excitedly at the passing cars.	he dog barked excitedly at the passing car	0.1250
What goes up but never comes down?	what does a but never comes down	0.2857
The fish swam in the clear water.	the fish swim in the Clearwater	0.4286
The stormy sea roared against the rocky shore.	Rocky Shore	0.7500
They cheered when the final whistle blew.	Richard won the final whistle	0.5714
The stars twinkled in the clear night sky.	the stars twinkled in the clear night	0.1250
She whispered a secret into the dark night.	she whispered a secret into the dark	0.1250
The wind howled through the trees all night long.	the wind held for the trees all night	0.3333
He read a book before going to bed.	can you read a book before going	0.5000
They danced in the rain without a care.	play dance in the rain without a	0.3750
The old man sighed and closed his eyes.	the old man's side and closed his eyes	0.2500
The clock ticked loudly in the silent room.	the clock ticks loudly in the silent	0.2500
She painted the sky in shades of blue and purple.	she painted the sky in shades of blue and purple	0.0000
What has keys but can't open doors?	what is keys by Kent open do	0.5714
His footsteps echoed in the empty corridor.	his footstep second in the empty quarter	0.4286
The cake was soft and sweet melting in her mouth.	the cake was stuffed and sweet melting in her mouth	0.1000
The baby giggled as the puppy chased its tail.	the baby giggled as the puppy chased its	0.1111
The sun peeked through the clouds after the rain.	the sun heat through the clouds after the	0.2222
She placed the last piece of the puzzle in place.	she plays the last piece of the puzzle in place	0.1000
The cat jumped onto the couch.	the cat jump on the	0.5000
The day ended as quietly as it had begun with a soft breeze.	the day and the day is quietly as in a bag in with a soft br	0.6923
The water in the lake was calm and still.	the water in the lake was call and	0.2222
What gets wetter as it dries?	what gets wetter as a price	0.3333
The leaves crunched under her boots as she walked.	the leaves crunched under her boots as	0.2222
The phone rang, but no one was on the other end.	the phone rang but no one was another	0.3636

Original Text	Transcribed Text	WER
The moonlight glowed softly on the forest floor.	the Moonlight glowed Softly on the forest floor	0.0000
She held the fragile flower in her hand.	she held the fragile flower in her hand	0
The lion roared shaking the ground beneath them.	the lion roar shaking the ground Ben	0.3750
He scribbled a quick note before rushing out the door.	before running out the door	0.6000
The stars fell like confetti from the sky.	the Stars fell like confetti from the sky	0.0000
The train whistled as it pulled into the station.	the train whistled as it pulled into the station	0.0000
The air smelled of rain and fresh earth.	Bears smell of rain and fresh	0.5000
The kitten purred as it snuggled in her lap.	the kitten part as it snuggled in her	0.2222
The river flowed gently through the valley.	the river flow gently through the valley	0.1429
The paper airplane soared across the classroom.	the paper airplane sword across the classroom	0.1429
She carefully folded the letter and placed it in her pocket.	she carefully folded the letter and placed it in her	0.0909
The treasure chest creaked open, revealing gold coins.	the treasure chest cracked open revealing gold Co	0.2500
She glanced at her watch and hurried down the street.	she glanced at her watch and hurried down the	0.1000
A penny saved is a penny earned they say.	a penny saved is a penny earned a	0.2222
The old swing creaked as it swayed in the breeze.	the old Twin Creek does its weight in the bre	0.6000
His fingers danced over the piano keys with ease.	his fingers danced over the piano keys with e	0.1111
The mountain loomed in the distance tall and proud.	the mountain moved in the distance to all and	0.4444
What has legs but cannot walk?	what is lags but can it	0.6667
The candle flickered as the wind blew through the window.	the candle flicker does the wine Boo for the window	0.5000
The baby's first steps were met with cheers and smiles.	the baby's first steps were met with chairs and smiles	0.2000
The city lights twinkled like stars in the distance.	stars in the distance	0.5000
The sound of the ocean calmed her restless mind.	the sound of the ocean called her Restless	0.2222

Table 8.7: WER INFO_NCE

8.23 Word Error Rate - MINE

Original Text	Transcribed Text	WER
The sky turned pink as the sun set behind the mountains.	behind the mountain	0.8182
The dog bark at the mailman.	the dog bark at the	0.1667
He placed the book gently on the dusty shelf.	can you place the book gently on the dusty Shelf	0.3333
The dog barked excitedly at the passing cars.	the dog barked excitedly at the passing cars	0.0000
What goes up but never comes down?	what does but never comes	0.4286
The fish swam in the clear water.	the fish swim in a clear water	0.2857
The stormy sea roared against the rocky shore.	the strongest Heroes against the Rocky	0.5000
They cheered when the final whistle blew.	date sheet won the final whist	0.7130
The stars twinkled in the clear night sky.	the stars twinkle in the clear night	0.2500
She whispered a secret into the dark night.	she whispered a secret into the dark	0.1250
The wind howled through the trees all night long.	the White House for the trees all night	0.4444
He read a book before going to bed.	he read a book before going to	0.1250
They danced in the rain without a care.	they danced in the rain without	0.25000
The old man sighed and closed his eyes.	the old man sides and clothes his	0.3750
The clock ticked loudly in the silent room.	the classic loudly in the silent	0.3750
She painted the sky in shades of blue and purple.	she painted the stye and shades of blue and purple	0.2000
What has keys but can't open doors?	what has keys but can't open	0.2857
His footsteps echoed in the empty corridor.	his phone stopped settled in the empty quart	0.5714
The cake was soft and sweet melting in her mouth.	the cake was soft and sweet melting in her mouth	0.0000
The baby giggled as the puppy chased its tail.	the baby get old as the puppy Chase sit	0.5556
The sun peeked through the clouds after the rain.	the Sun Beat through the clouds after the	0.2222
She placed the last piece of the puzzle in place.	she plays the last piece of the puzzle in	0.2000
The cat jumped onto the couch.	the cat jumped onto the couch	0.0000
The day ended as quietly as it had begun with a soft breeze.	today in the dance quietly as it had bacon with a soft bre	0.4615
The water in the lake was calm and still.	the water in the lake was calm and	0.1111
What gets wetter as it dries?	what gets wetter as it	0.1667
The leaves crunched under her boots as she walked.	the leaves crunched under her but says she	0.3333
The phone rang, but no one was on the other end.	the phone ran but no one was on the	0.2727

Original Text	Transcribed Text	WER
The moonlight glowed softly on the forest floor.	the Moonlight glowed Softly on a forest	0.2500
She held the fragile flower in her hand.	she held the fragile flower in her	0.125
The lion roared shaking the ground beneath them.	the lion Rod shaking the ground been	0.3750
He scribbled a quick note before rushing out the door.	he's crippled a quick note before rushing out the	0.3000
The stars fell like confetti from the sky.	the stars that like confetti from the sky	0.1250
The train whistled as it pulled into the station.	the train whistled as it pulled into the station	0.0000
The air smelled of rain and fresh earth.	bear smell the rain and fresh	0.6250
The kitten purred as it snuggled in her lap.	the kitten Prairie does it's snuggled in her lap	0.3333
The river flowed gently through the valley.	the river flow gently through the valley	0.1429
The paper airplane soared across the classroom.	the paper airplane sort of cross the classroom	0.4286
She carefully folded the letter and placed it in her pocket.	she got a folded the ladder and placed it in her	0.3636
The treasure chest creaked open, revealing gold coins.	the traffic just creaked open revealing goal	0.5000
She glanced at her watch and hurried down the street.	she glanced at her watch and hurried down the street	0.0000
A penny saved is a penny earned they say.	a penny saved is a penny earned	0.2222
The old swing creaked as it swayed in the breeze.	those Twin Creeks has its weight in the bre	0.8000
His fingers danced over the piano keys with ease.	has fingers stand still over the piano keys with the	0.4444
The mountain loomed in the distance tall and proud.	the mountain lived in the distance stolen	0.4444
What has legs but cannot walk?	what is legs but cannot walk	0.1667
The candle flickered as the wind blew through the window.	the candle flickered as the wine blew through the window	0.1000
The baby's first steps were met with cheers and smiles.	the baby's first steps were met with Jersey and smiles	0.2000
The city lights twinkled like stars in the distance.	the city lights wrinkled like stars in the distance	0.1111
The sound of the ocean calmed her restless mind.	the sound of the ocean called her Restless	0.2222

Table 8.8: WER MINE

8.24 Word Error Rate - GRL

Original Text	Transcribed Text	WER
The sky turned pink as the sun set behind the mountains.	the sky turn pink as the sunset behind the mountain	0.3636
The dog bark at the mailman.	the dog bark at the mail	0.1667
He placed the book gently on the dusty shelf.	he plays the book gently on the dusty Shelf	0.1111
The dog barked excitedly at the passing cars.	the dog barked excitedly at the passing	0.1250
What goes up but never comes down?	what goes up but never comes	0.1429
The fish swam in the clear water.	the fish swim in a clear water	0.2857
The stormy sea roared against the rocky shore.	this time is your own against the Rocky	0.7500
They cheered when the final whistle blew.	they shared from the final whistle	0.4286
The stars twinkled in the clear night sky.	the stars twinkled in the clear night	0.1250
She whispered a secret into the dark night.	she whispered a secret into the dark	0.1250
The wind howled through the trees all night long.	the White House of the trees all night	0.4444
He read a book before going to bed.	here at the book the photo on the	0.8750
They danced in the rain without a care.	play dance in the rain without	0.5000
The old man sighed and closed his eyes.	the old man's side and closed his eyes	0.2500
The clock ticked loudly in the silent room.	the classic lovely and the silent	0.6250
She painted the sky in shades of blue and purple.	she painted the sky in shades of blue and purple	0.0000
What has keys but can't open doors?	what is keys but can't open doors	0.2857
His footsteps echoed in the empty corridor.	is food stop second in the MP	0.8571
The cake was soft and sweet melting in her mouth.	the cake was soft and sweet melting in her mouth	0.0000
The baby giggled as the puppy chased its tail.	the baby Giggles as the puppy chased it	0.3333
The sun peeked through the clouds after the rain.	the sun peeked through the clouds after the	0.1111
She placed the last piece of the puzzle in place.	she placed the last piece of the puzzle in place	0.0000
The cat jumped onto the couch.	the cat jump onto the couch	0.1667
The day ended as quietly as it had begun with a soft breeze.	the day Ended as quietly as it had bacon with a soft Breeze	0.0769
The water in the lake was calm and still.	the water in the lake was calm and	0.1111
What gets wetter as it dries?	what gets wetter as a	0.3333
The leaves crunched under her boots as she walked.	the leaves crunch on the habitat she	0.6667
The phone rang, but no one was on the other end.	the phone ring but no one was on the other	0.1818

Original Text	Transcribed Text	WER
The moonlight glowed softly on the forest floor.	the moon light glowed Softly on the forest floor	0.2500
She held the fragile flower in her hand.	she held a fragile flower in her	0.2500
The lion roared shaking the ground beneath them.	the lion or shaking the ground beneath	0.2500
He scribbled a quick note before rushing out the door.	he scribbled a quick note before rushing out the door	0.0000
The stars fell like confetti from the sky.	the stars fall like confetti from the	0.2500
The train whistled as it pulled into the station.	the train whistle desert pulled into the station	0.3333
The air smelled of rain and fresh earth.	bear smelled of rain and fresh	0.3750
The kitten purred as it snuggled in her lap.	the Kitten Party has its nickels in her lap.	0.5556
The river flowed gently through the valley.	the river flow gently through the valley	0.1429
The paper airplane soared across the classroom.	the paper airplane sword across the class	0.2857
She carefully folded the letter and placed it in her pocket.	be carefully folded the letter and placed it in her Pok	0.1818
The treasure chest creaked open, revealing gold coins.	the treasure chest Creek open reviewing gold	0.3750
She glanced at her watch and hurried down the street.	she glanced at her watch and hurried down the street	0.0000
A penny saved is a penny earned they say.	a penny saved is a penny earned	0.2222
The old swing creaked as it swayed in the breeze.	Bill's Twin Creek has its weight in the breez	0.8000
His fingers danced over the piano keys with ease.	his fingers Dan stove for the piano keys with	0.4444
The mountain loomed in the distance tall and proud.	the mountain loom in the distance salt and	0.3333
What has legs but cannot walk?	what has legs but cannot	0.1667
The candle flickered as the wind blew through the window.	the candle flicker does the wine blew through the window	0.3000
The baby's first steps were met with cheers and smiles.	the baby's first steps were met with chairs and smiles	0.2000
The city lights twinkled like stars in the distance.	the city lights twinkled like stars in the distance	0.0000
The sound of the ocean calmed her restless mind.	the sound of the ocean called the Restless	0.3333

Table 8.9: WER GRL

8.25 Flask Application

Disentangled Speech Representation Algorithms Using Information Theory with Application in Speech Synthesis



Figure 8.9: Screenshot of Flask Application

Bibliography

- [1] Yi Ren, Chenxu Hu, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. FastSpeech 2: Fast and High-Quality End-to-End Text to Speech, August 2022. URL <http://arxiv.org/abs/2006.04558>. arXiv:2006.04558 [cs, eess] version: 8.
- [2] Tu Anh Nguyen, Wei-Ning Hsu, Antony D’Avirro, Bowen Shi, Itai Gat, Maryam Fazel-Zarani, Tal Remez, Jade Copet, Gabriel Synnaeve, Michael Hassid, Felix Kreuk, Yossi Adi, and Emmanuel Dupoux. EXPRESSO: A Benchmark and Analysis of Discrete Expressive Speech Resynthesis, August 2023. URL <http://arxiv.org/abs/2308.05725>. arXiv:2308.05725 [cs, eess].
- [3] Yaroslav Ganin and Victor Lempitsky. Unsupervised Domain Adaptation by Backpropagation, February 2015. URL <http://arxiv.org/abs/1409.7495>. arXiv:1409.7495 [cs, stat].
- [4] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. Mutual Information Neural Estimation. In *Proceedings of the 35th International Conference on Machine Learning*, pages 531–540. PMLR, July 2018. URL <https://proceedings.mlr.press/v80/belghazi18a.html>. ISSN: 2640-3498.
- [5] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation Learning with Contrastive Predictive Coding, January 2019. URL <http://arxiv.org/abs/1807.03748>. arXiv:1807.03748 [cs, stat].
- [6] Pengyu Cheng, Weituo Hao, Shuyang Dai, Jiachang Liu, Zhe Gan, and Lawrence Carin. CLUB: A Contrastive Log-ratio Upper Bound of Mutual Information, July 2020. URL <http://arxiv.org/abs/2006.12013>. arXiv:2006.12013 [cs, stat].
- [7] Jeremiah Birrell, Yannis Pantazis, Paul Dupuis, Markos A. Katsoulakis, and Luc Rey-Bellet. Function-space regularized Rényi divergences, February 2023. URL <http://arxiv.org/abs/2210.04974>. arXiv:2210.04974 [cs, stat].
- [8] Deep Learning for Siri’s Voice: On-device Deep Mixture Density Networks for Hybrid Unit Selection Synthesis, . URL <https://machinelearning.apple.com/research/siri-voices>.
- [9] Alexa’s speech recognition research at ICASSP 2022, May 2022. URL <https://www.amazon.science/blog/alexas-speech-recognition-research-at-icassp-2022>.

- [10] S. Davis and P. Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28(4):357–366, August 1980. ISSN 0096-3518. doi: 10.1109/TASSP.1980.1163420. URL <https://ieeexplore.ieee.org/document/1163420>. Conference Name: IEEE Transactions on Acoustics, Speech, and Signal Processing.
- [11] J. Volkmann, S. S. Stevens, and E. B. Newman. A Scale for the Measurement of the Psychological Magnitude Pitch. *The Journal of the Acoustical Society of America*, 8(3.Supplement):208, January 1937. ISSN 0001-4966. doi: 10.1121/1.1901999. URL <https://doi.org/10.1121/1.1901999>.
- [12] Nicolae Catalin Ristea, Ando Saabas, Ross Cutler, Babak Naderi, Sebastian Braun, and Solomiya Branets. ICASSP 2024 Speech Signal Improvement Challenge, January 2024. URL <http://arxiv.org/abs/2401.14444>. arXiv:2401.14444 [cs, eess].
- [13] Walter Kellermann, Rainer Martin, and Nobutaka Ono. Signal processing and machine learning for speech and audio in acoustic sensor networks. *EURASIP Journal on Audio, Speech, and Music Processing*, 2023(1):54, December 2023. ISSN 1687-4722. doi: 10.1186/s13636-023-00322-6. URL <https://doi.org/10.1186/s13636-023-00322-6>.
- [14] Yuxuan Wang, R. J. Skerry-Ryan, Daisy Stanton, Yonghui Wu, Ron J. Weiss, Navdeep Jaitly, Zongheng Yang, Ying Xiao, Zhifeng Chen, Samy Bengio, Quoc Le, Yannis Agiomyrgiannakis, Rob Clark, and Rif A. Saurous. Tacotron: Towards End-to-End Speech Synthesis, April 2017. URL <http://arxiv.org/abs/1703.10135>. arXiv:1703.10135 [cs] version: 2.
- [15] Yi Ren, Yangjun Ruan, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. FastSpeech: Fast, Robust and Controllable Text to Speech, November 2019. URL <http://arxiv.org/abs/1905.09263>. arXiv:1905.09263 [cs, eess].
- [16] Adam Polyak, Yossi Adi, Jade Copet, Eugene Kharonov, Kushal Lakhotia, Wei-Ning Hsu, Abdelrahman Mohamed, and Emmanuel Dupoux. Speech Resynthesis from Discrete Disentangled Self-Supervised Representations, July 2021. URL <http://arxiv.org/abs/2104.00355>. arXiv:2104.00355 [cs, eess].
- [17] Yusuf Brima, Ulf Krumnack, Simone Pika, and Gunther Heidemann. Learning Disentangled Speech Representations, November 2023. URL <http://arxiv.org/abs/2311.03389>. arXiv:2311.03389 [cs, eess].
- [18] Xu Tan. Basics of Spoken Language Processing. In Xu Tan, editor, *Neural Text-to-Speech Synthesis*, pages 17–36. Springer Nature, Singapore, 2023. ISBN 978-981-9908-27-1. doi: 10.1007/978-981-99-0827-1_2. URL https://doi.org/10.1007/978-981-99-0827-1_2.
- [19] Kien Do and Truyen Tran. Theory and Evaluation Metrics for Learning Disentangled Representations, March 2021. URL <http://arxiv.org/abs/1908.09961>. arXiv:1908.09961 [cs, stat].
- [20] J. Makhoul. Linear prediction: A tutorial review. *Proceedings of the IEEE*, 63(4):561–580, April 1975. ISSN 1558-2256. doi: 10.1109/PROC.1975.9792. URL <https://doi.org/10.1109/PROC.1975.9792>.

- [//ieeexplore.ieee.org/document/1451722](http://ieeexplore.ieee.org/document/1451722). Conference Name: Proceedings of the IEEE.
- [21] Keiron O’Shea and Ryan Nash. An Introduction to Convolutional Neural Networks, December 2015. URL <http://arxiv.org/abs/1511.08458>. arXiv:1511.08458 [cs].
- [22] Robin M. Schmidt. Recurrent Neural Networks (RNNs): A gentle Introduction and Overview, November 2019. URL <http://arxiv.org/abs/1912.05911>. arXiv:1912.05911 [cs, stat].
- [23] Geoffrey Hinton, Li Deng, Dong Yu, George Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara Sainath, and Brian Kingsbury. Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups. *IEEE Signal Processing Magazine*, 29(6):82–97, November 2012. ISSN 1053-5888. doi: 10.1109/MSP.2012.2205597. URL <http://ieeexplore.ieee.org/document/6296526/>.
- [24] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. WaveNet: A Generative Model for Raw Audio, September 2016. URL <http://arxiv.org/abs/1609.03499>. arXiv:1609.03499 [cs].
- [25] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need, August 2023. URL <http://arxiv.org/abs/1706.03762>. arXiv:1706.03762 [cs].
- [26] Yung-Sung Chuang, Chi-Liang Liu, Hung-Yi Lee, and Lin-shan Lee. Speech-BERT: An Audio-and-text Jointly Learned Language Model for End-to-end Spoken Question Answering, August 2020. URL <http://arxiv.org/abs/1910.11559>. arXiv:1910.11559 [cs, eess].
- [27] Yibin Zheng, Xinhui Li, Fenglong Xie, and Li Lu. Improving End-to-End Speech Synthesis with Local Recurrent Neural Network Enhanced Transformer. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6734–6738, February 2020. doi: 10.1109/ICASSP40776.2020.9054148. URL <https://ieeexplore.ieee.org/document/9054148>. ISSN: 2379-190X.
- [28] Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, Jie Chen, Jingdong Chen, Zhijie Chen, Mike Chrzanowski, Adam Coates, Greg Diamos, Ke Ding, Niandong Du, Erich Elsen, Jesse Engel, Weiwei Fang, Linxi Fan, Christopher Fougner, Liang Gao, Caixia Gong, Awni Hannun, Tony Han, Lappi Johannes, Bing Jiang, Cai Ju, Billy Jun, Patrick LeGresley, Libby Lin, Junjie Liu, Yang Liu, Weigao Li, Xiangang Li, Dongpeng Ma, Sharan Narang, Andrew Ng, Sherjil Ozair, Yiping Peng, Ryan Prenger, Sheng Qian, Zongfeng Quan, Jonathan Raiman, Vinay Rao, Sanjeev Satheesh, David Seetapun, Shubho Sengupta, Kavya Srinet, Anuroop Sriram, Haiyuan Tang, Liliang Tang, Chong Wang, Jidong Wang, Kaifu Wang, Yi Wang, Zhijian Wang, Zhiqian Wang, Shuang Wu, Likai Wei, Bo Xiao, Wen Xie, Yan Xie, Dani Yogatama, Bin Yuan, Jun Zhan, and Zhenyao Zhu. Deep Speech 2 : End-to-End Speech Recognition in English and Mandarin. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 173–182.

- PMLR, June 2016. URL <https://proceedings.mlr.press/v48/amodei16.html>. ISSN: 1938-7228.
- [29] Saurabhchand Bhati, Jesús Villalba, Piotr Żelasko, Laureano Moro-Velazquez, and Najim Dehak. Regularizing Contrastive Predictive Coding for Speech Applications, April 2023. URL <http://arxiv.org/abs/2304.05974>. arXiv:2304.05974 [eess].
- [30] Chao Wang, Zhonghao Li, Benlai Tang, Xiang Yin, Yuan Wan, Yibiao Yu, and Zejun Ma. Towards High-fidelity Singing Voice Conversion with Acoustic Reference and Contrastive Predictive Coding, October 2021. URL <http://arxiv.org/abs/2110.04754>. arXiv:2110.04754 [cs, eess].
- [31] Zhouyuan Huo, Dongseong Hwang, Khe Chai Sim, Shefali Garg, Ananya Misra, Nikhil Siddhartha, Trevor Strohman, and Françoise Beaufays. Incremental Layer-wise Self-Supervised Learning for Efficient Speech Domain Adaptation On Device, September 2021. URL <http://arxiv.org/abs/2110.00155>. arXiv:2110.00155 [cs, eess].
- [32] Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. Multimodal Machine Learning: A Survey and Taxonomy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(2):423–443, October 2019. ISSN 1939-3539. doi: 10.1109/TPAMI.2018.2798607. URL <https://ieeexplore.ieee.org/document/8269806>. Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence.
- [33] Soumya Priyadarsini Panda and Ajit Kumar Nayak. A waveform concatenation technique for text-to-speech synthesis. *International Journal of Speech Technology*, 20(4):959–976, December 2017. ISSN 1572-8110. doi: 10.1007/s10772-017-9463-8. URL <https://doi.org/10.1007/s10772-017-9463-8>.
- [34] Jerneja Zganec Gros and Mario Zganec. An Efficient Unit-selection Method for Concatenative Text-to-speech Synthesis Systems. *Journal of Computing and Information Technology*, 16(1):69, 2008. ISSN 1330-1136, 1846-3908. doi: 10.2498/cit.1001049. URL <http://cit.srce.unizg.hr/index.php/CIT/article/view/1660>.
- [35] Pablo Pérez Zarazaga, Zofia Malisz, Gustav Eje Henter, and Lauri Juvela. Speaker-independent neural formant synthesis, June 2023. URL <http://arxiv.org/abs/2306.01957>. arXiv:2306.01957 [eess].
- [36] Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae. HiFi-GAN: Generative Adversarial Networks for Efficient and High Fidelity Speech Synthesis, October 2020. URL <http://arxiv.org/abs/2010.05646>. arXiv:2010.05646 [cs, eess].
- [37] C.H. Coker. A model of articulatory dynamics and control. *Proceedings of the IEEE*, 64(4):452–460, April 1976. ISSN 1558-2256. doi: 10.1109/PROC.1976.10154. URL <https://ieeexplore.ieee.org/document/1454423/similar#similar>. Conference Name: Proceedings of the IEEE.
- [38] C. H. Shadle and R. I. Damper. Prospects for articulatory synthesis: A position paper. pages 121–126, 2002. URL <https://eprints.soton.ac.uk/256064/>.

- [39] Jonathan Shen, Ruoming Pang, Ron J. Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, Yu Zhang, Yuxuan Wang, R. J. Skerry-Ryan, Rif A. Saurous, Yannis Agiomyrgiannakis, and Yonghui Wu. Natural TTS Synthesis by Conditioning WaveNet on Mel Spectrogram Predictions, February 2018. URL <http://arxiv.org/abs/1712.05884>. arXiv:1712.05884 [cs].
- [40] Jaehyeon Kim, Sungwon Kim, Jungil Kong, and Sungroh Yoon. Glow-TTS: A Generative Flow for Text-to-Speech via Monotonic Alignment Search, October 2020. URL <http://arxiv.org/abs/2005.11129>. arXiv:2005.11129 [cs, eess].
- [41] Zeqian Ju, Yuancheng Wang, Kai Shen, Xu Tan, Detai Xin, Dongchao Yang, Yanqing Liu, Yichong Leng, Kaitao Song, Siliang Tang, Zhizheng Wu, Tao Qin, Xiangyang Li, Wei Ye, Shikun Zhang, Jiang Bian, Lei He, Jinyu Li, and Sheng Zhao. NaturalSpeech 3: Zero-Shot Speech Synthesis with Factorized Codec and Diffusion Models, April 2024. URL <http://arxiv.org/abs/2403.03100>. arXiv:2403.03100 [cs, eess].
- [42] Raviraj Joshi and Nikesh Garera. Rapid Speaker Adaptation in Low Resource Text to Speech Systems using Synthetic Data and Transfer learning, December 2023. URL <http://arxiv.org/abs/2312.01107>. arXiv:2312.01107 [cs].
- [43] R. J. Skerry-Ryan, Eric Battenberg, Ying Xiao, Yuxuan Wang, Daisy Stanton, Joel Shor, Ron J. Weiss, Rob Clark, and Rif A. Saurous. Towards End-to-End Prosody Transfer for Expressive Speech Synthesis with Tacotron, March 2018. URL <http://arxiv.org/abs/1803.09047>. arXiv:1803.09047 [cs, eess].
- [44] Yi Lei, Shan Yang, Xinsheng Wang, and Lei Xie. MsEmoTTS: Multi-scale emotion transfer, prediction, and control for emotional speech synthesis, January 2022. URL <https://arxiv.org/abs/2201.06460v1>.
- [45] MM-TTS: Multi-Modal Prompt Based Style Transfer for Expressive Text-to-Speech Synthesis, . URL <https://arxiv.labs.arxiv.org/html/2312.10687>.
- [46] Yuke Li, Xinfu Zhu, Yi Lei, Hai Li, Junhui Liu, Danming Xie, and Lei Xie. Zero-Shot Emotion Transfer For Cross-Lingual Speech Synthesis, October 2023. URL <http://arxiv.org/abs/2310.03963>. arXiv:2310.03963 [cs, eess].
- [47] Sercan O. Arik, Mike Chrzanowski, Adam Coates, Gregory Diamos, Andrew Gibiansky, Yongguo Kang, Xian Li, John Miller, Andrew Ng, Jonathan Raiman, Shubho Sengupta, and Mohammad Shoeybi. Deep Voice: Real-time Neural Text-to-Speech, March 2017. URL <http://arxiv.org/abs/1702.07825>. arXiv:1702.07825 [cs].
- [48] Sercan Arik, Gregory Diamos, Andrew Gibiansky, John Miller, Kainan Peng, Wei Ping, Jonathan Raiman, and Yanqi Zhou. Deep Voice 2: Multi-Speaker Neural Text-to-Speech, September 2017. URL <http://arxiv.org/abs/1705.08947>. arXiv:1705.08947 [cs].
- [49] Jose M. R. Sotelo, Soroush Mehri, Kundan Kumar, J. F. Santos, Kyle Kastner, Aaron C. Courville, and Yoshua Bengio. Char2Wav: End-to-End Speech Synthesis. February 2017. URL <https://www.semanticscholar.org/paper/Char2Wav%3A-End-to-End-Speech-Synthesis-Sotelo-Mehri/9203d6c076bffe87336f2ea91f5851436c02dbe6>.

- [50] Wei Ping, Kainan Peng, Andrew Gibiansky, Sercan O. Arik, Ajay Kannan, Sharan Narang, Jonathan Raiman, and John Miller. Deep Voice 3: Scaling Text-to-Speech with Convolutional Sequence Learning, February 2018. URL <http://arxiv.org/abs/1710.07654>. arXiv:1710.07654 [cs, eess].
- [51] Wei Ping, Kainan Peng, and Jitong Chen. ClariNet: Parallel Wave Generation in End-to-End Text-to-Speech, February 2019. URL <http://arxiv.org/abs/1807.07281>. arXiv:1807.07281 [cs, eess].
- [52] Jeff Donahue, Sander Dieleman, Mikolaj Bińkowski, Erich Elsen, and Karen Simonyan. End-to-End Adversarial Text-to-Speech, March 2021. URL <http://arxiv.org/abs/2006.03575>. arXiv:2006.03575 [cs, eess].
- [53] Xu Tan, Jiawei Chen, Haohe Liu, Jian Cong, Chen Zhang, Yanqing Liu, Xi Wang, Yichong Leng, Yuanhao Yi, Lei He, Frank Soong, Tao Qin, Sheng Zhao, and Tie-Yan Liu. NaturalSpeech: End-to-End Text to Speech Synthesis with Human-Level Quality, May 2022. URL <http://arxiv.org/abs/2205.04421>. arXiv:2205.04421 [cs, eess].
- [54] Kanishka Rao, Fuchun Peng, Hasim Sak, and Francoise Beaufays. Grapheme-to-phoneme conversion using Long Short-Term Memory recurrent neural networks. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4225–4229, South Brisbane, Queensland, Australia, April 2015. IEEE. ISBN 978-1-4673-6997-8. doi: 10.1109/ICASSP.2015.7178767. URL <http://ieeexplore.ieee.org/document/7178767/>.
- [55] Moon-Jung Chae, Kyubyong Park, Jinhyun Bang, Soobin Suh, Jonghyuk Park, Namju Kim, and Longhun Park. Convolutional Sequence to Sequence Model with Non-Sequential Greedy Decoding for Grapheme to Phoneme Conversion. *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2486–2490, April 2018. doi: 10.1109/ICASSP.2018.8462678. URL <https://ieeexplore.ieee.org/document/8462678/>. Conference Name: ICASSP 2018 - 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) ISBN: 9781538646588 Place: Calgary, AB Publisher: IEEE.
- [56] Maximilian Bisani and Hermann Ney. Joint-sequence models for grapheme-to-phoneme conversion. *Speech Communication*, 50(5):434–451, May 2008. ISSN 0167-6393. doi: 10.1016/j.specom.2008.01.002. URL <https://www.sciencedirect.com/science/article/pii/S0167639308000046>.
- [57] Junjie Pan, Xiang Yin, Zhiling Zhang, Shichao Liu, Yang Zhang, Zejun Ma, and Yuxuan Wang. A unified sequence-to-sequence front-end model for Mandarin text-to-speech synthesis, November 2019. URL <http://arxiv.org/abs/1911.04111>. arXiv:1911.04111 [cs, eess].
- [58] Yang Zhang, Liqun Deng, and Yasheng Wang. Unified Mandarin TTS Front-end Based on Distilled BERT Model, December 2020. URL <http://arxiv.org/abs/2012.15404>. arXiv:2012.15404 [cs].
- [59] Rui Liu, Berrak Sisman, Feilong Bao, Guanglai Gao, and Haizhou Li. Modeling Prosodic Phrasing with Multi-Task Learning in Tacotron-based TTS. *IEEE Signal Processing Letters*, 27:1470–1474, 2020. ISSN 1070-9908, 1558-2361. doi: 10.1109/

- LSP.2020.3016564. URL <http://arxiv.org/abs/2008.05284>. arXiv:2008.05284 [cs, eess].
- [60] Yuxuan Wang, Daisy Stanton, Yu Zhang, R. J. Skerry-Ryan, Eric Battenberg, Joel Shor, Ying Xiao, Fei Ren, Ye Jia, and Rif A. Saurous. Style Tokens: Unsupervised Style Modeling, Control and Transfer in End-to-End Speech Synthesis, March 2018. URL <http://arxiv.org/abs/1803.09017>. arXiv:1803.09017 [cs, eess].
- [61] Mingjian Chen, Xu Tan, Bohan Li, Yanqing Liu, Tao Qin, Sheng Zhao, and Tie-Yan Liu. AdaSpeech: Adaptive Text to Speech for Custom Voice, March 2021. URL <http://arxiv.org/abs/2103.00993>. arXiv:2103.00993 [cs, eess].
- [62] Tomoki Hayashi, Shinji Watanabe, Tomoki Toda, Kazuya Takeda, Shubham Toshniwal, and Karen Livescu. Pre-trained text embeddings for enhanced text-to-speech synthesis: 20th Annual Conference of the International Speech Communication Association: Crossroads of Speech and Language, INTERSPEECH 2019. *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, 2019-September:4430–4434, 2019. ISSN 2308-457X. doi: 10.21437/Interspeech.2019-3177. URL <http://www.scopus.com/inward/record.url?scp=85074724712&partnerID=8YFLogxK>.
- [63] Haohan Guo, Frank K. Soong, Lei He, and Lei Xie. Exploiting Syntactic Features in a Parsed Tree to Improve End-to-End TTS, April 2019. URL <http://arxiv.org/abs/1904.04764>. arXiv:1904.04764 [cs].
- [64] Guangyan Zhang, Kaitao Song, Xu Tan, Daxin Tan, Yuzi Yan, Yanqing Liu, Gang Wang, Wei Zhou, Tao Qin, Tan Lee, and Sheng Zhao. Mixed-Phoneme BERT: Improving BERT with Mixed Phoneme and Sup-Phoneme Representations for Text to Speech, July 2022. URL <http://arxiv.org/abs/2203.17190>. arXiv:2203.17190 [cs, eess].
- [65] Rui Liu, Berrak Sisman, and Haizhou Li. GraphSpeech: Syntax-Aware Graph Attention Network For Neural Speech Synthesis, March 2021. URL <http://arxiv.org/abs/2010.12423>. arXiv:2010.12423 [cs, eess].
- [66] K. Tokuda, T. Yoshimura, T. Masuko, T. Kobayashi, and T. Kitamura. Speech parameter generation algorithms for HMM-based speech synthesis. In *2000 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.00CH37100)*, volume 3, pages 1315–1318 vol.3, June 2000. doi: 10.1109/ICASSP.2000.861820. URL <https://ieeexplore.ieee.org/document/861820>. ISSN: 1520-6149.
- [67] Kainan Peng, Wei Ping, Zhao Song, and Kexin Zhao. Non-Autoregressive Neural Text-to-Speech, June 2020. URL <http://arxiv.org/abs/1905.08459>. arXiv:1905.08459 [cs, eess].
- [68] Sang-Hoon Lee, Hyun-Wook Yoon, Hyeong-Rae Noh, Ji-Hoon Kim, and Seong-Whan Lee. Multi-SpectroGAN: High-Diversity and High-Fidelity Spectrogram Generation with Adversarial Style Combination for Speech Synthesis, December 2020. URL <http://arxiv.org/abs/2012.07267>. arXiv:2012.07267 [eess].

- [69] Wei-Ning Hsu, Yu Zhang, Ron J. Weiss, Heiga Zen, Yonghui Wu, Yuxuan Wang, Yuan Cao, Ye Jia, Zhifeng Chen, Jonathan Shen, Patrick Nguyen, and Ruoming Pang. Hierarchical Generative Modeling for Controllable Speech Synthesis, December 2018. URL <http://arxiv.org/abs/1810.07217>. arXiv:1810.07217 [cs, eess].
- [70] Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. DiffWave: A Versatile Diffusion Model for Audio Synthesis, March 2021. URL <http://arxiv.org/abs/2009.09761>. arXiv:2009.09761 [cs, eess, stat].
- [71] Haohan Guo, Frank K. Soong, Lei He, and Lei Xie. A New GAN-based End-to-End TTS Training Algorithm, April 2019. URL <http://arxiv.org/abs/1904.04775>. arXiv:1904.04775 [cs].
- [72] ShuangMa, Daniel McDuff, and Yale Song. Neural TTS Stylization with Adversarial and Collaborative Games. April 2019. URL <https://www.microsoft.com/en-us/research/publication/neural-tts-stylization-with-adversarial-and-collaborative-games/>.
- [73] Rafael Valle, Kevin Shih, Ryan Prenger, and Bryan Catanzaro. Flowtron: an Autoregressive Flow-based Generative Network for Text-to-Speech Synthesis, July 2020. URL <http://arxiv.org/abs/2005.05957>. arXiv:2005.05957 [cs, eess].
- [74] Chenfeng Miao, Shuang Liang, Minchuan Chen, Jun Ma, Shaojun Wang, and Jing Xiao. Flow-TTS: A Non-Autoregressive Network for Text to Speech Based on Flow. *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7209–7213, May 2020. doi: 10.1109/ICASSP40776.2020.9054484. URL <https://ieeexplore.ieee.org/document/9054484/>. Conference Name: ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) ISBN: 9781509066315 Place: Barcelona, Spain Publisher: IEEE.
- [75] Yoonhyung Lee, Joongbo Shin, and Kyomin Jung. Bidirectional Variational Inference for Non-Autoregressive Text-to-Speech. October 2020. URL <https://openreview.net/forum?id=o3iritJHLf0>.
- [76] Ya-Jie Zhang, Shifeng Pan, Lei He, and Zhen-Hua Ling. Learning latent representations for style control and transfer in end-to-end speech synthesis, February 2019. URL <http://arxiv.org/abs/1812.04342>. arXiv:1812.04342 [cs, eess].
- [77] Isaac Elias, Heiga Zen, Jonathan Shen, Yu Zhang, Ye Jia, Ron Weiss, and Yonghui Wu. Parallel Tacotron: Non-Autoregressive and Controllable TTS, October 2020. URL <http://arxiv.org/abs/2010.11439>. arXiv:2010.11439 [cs, eess].
- [78] Isaac Elias, Heiga Zen, Jonathan Shen, Yu Zhang, Ye Jia, R. J. Skerry-Ryan, and Yonghui Wu. Parallel Tacotron 2: A Non-Autoregressive Neural TTS Model with Differentiable Duration Modeling, August 2021. URL <http://arxiv.org/abs/2103.14574>. arXiv:2103.14574 [cs, eess].
- [79] Myeonghun Jeong, Hyeongju Kim, Sung Jun Cheon, Byoung Jin Choi, and Nam Soo Kim. Diff-TTS: A Denoising Diffusion Model for Text-to-Speech, April 2021. URL <http://arxiv.org/abs/2104.01409>. arXiv:2104.01409 [cs, eess].

- [80] Vadim Popov, Ivan Vovk, Vladimir Gogoryan, Tasnima Sadekova, and Mikhail Kudinov. Grad-TTS: A Diffusion Probabilistic Model for Text-to-Speech, August 2021. URL <http://arxiv.org/abs/2105.06337>. arXiv:2105.06337 [cs, stat].
- [81] Sang-gil Lee, Heeseung Kim, Chaehun Shin, Xu Tan, Chang Liu, Qi Meng, Tao Qin, Wei Chen, Sungroh Yoon, and Tie-Yan Liu. PriorGrad: Improving Conditional Denoising Diffusion Models with Data-Dependent Adaptive Prior, February 2022. URL <http://arxiv.org/abs/2106.06406>. arXiv:2106.06406 [cs, eess, stat].
- [82] Naihan Li, Shujie Liu, Yanqing Liu, Sheng Zhao, and Ming Liu. Neural Speech Synthesis with Transformer Network. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):6706–6713, July 2019. ISSN 2374-3468. doi: 10.1609/aaai.v33i01.33016706. URL <https://ojs.aaai.org/index.php/AAAI/article/view/4642>. Number: 01.
- [83] STRAIGHT, exploitation of the other aspect of VOCODER: Perceptually isomorphic decomposition of speech sounds, . URL https://www.jstage.jst.go.jp/article/ast/27/6/27_6_349/_article.
- [84] Yang Ai and Zhen-Hua Ling. A Neural Vocoder with Hierarchical Generation of Amplitude and Phase Spectra for Statistical Parametric Speech Synthesis. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:839–851, 2020. ISSN 2329-9290, 2329-9304. doi: 10.1109/TASLP.2020.2970241. URL <http://arxiv.org/abs/1906.09573>. arXiv:1906.09573 [cs, eess].
- [85] Nal Kalchbrenner, Erich Elsen, Karen Simonyan, Seb Noury, Norman Casagrande, Edward Lockhart, Florian Stimberg, Aaron van den Oord, Sander Dieleman, and Koray Kavukcuoglu. Efficient Neural Audio Synthesis, June 2018. URL <http://arxiv.org/abs/1802.08435>. arXiv:1802.08435 [cs, eess].
- [86] Ryan Prenger, Rafael Valle, and Bryan Catanzaro. Waveglow: A Flow-based Generative Network for Speech Synthesis. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3617–3621, February 2019. doi: 10.1109/ICASSP.2019.8683143. URL <https://ieeexplore.ieee.org/document/8683143>. ISSN: 2379-190X.
- [87] Sungwon Kim, Sang-gil Lee, Jongyoon Song, Jaehyeon Kim, and Sungroh Yoon. FloWaveNet : A Generative Flow for Raw Audio, May 2019. URL <http://arxiv.org/abs/1811.02155>. arXiv:1811.02155 [cs, eess].
- [88] Aaron van den Oord, Yazhe Li, Igor Babuschkin, Karen Simonyan, Oriol Vinyals, Koray Kavukcuoglu, George van den Driessche, Edward Lockhart, Luis C. Cobo, Florian Stimberg, Norman Casagrande, Dominik Grewe, Seb Noury, Sander Dieleman, Erich Elsen, Nal Kalchbrenner, Heiga Zen, Alex Graves, Helen King, Tom Walters, Dan Belov, and Demis Hassabis. Parallel WaveNet: Fast High-Fidelity Speech Synthesis, November 2017. URL <http://arxiv.org/abs/1711.10433>. arXiv:1711.10433 [cs].
- [89] Kundan Kumar, Rithesh Kumar, Thibault de Boissiere, Lucas Gestin, Wei Zhen Teoh, Jose Sotelo, Alexandre de Brebisson, Yoshua Bengio, and Aaron Courville. MelGAN: Generative Adversarial Networks for Conditional Waveform Synthesis, December 2019. URL <http://arxiv.org/abs/1910.06711>. arXiv:1910.06711 [cs, eess].

- [90] Laurent Dinh, David Krueger, and Yoshua Bengio. NICE: Non-linear Independent Components Estimation, April 2015. URL <http://arxiv.org/abs/1410.8516>. arXiv:1410.8516 [cs].
- [91] Diederik P. Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improving Variational Inference with Inverse Autoregressive Flow, January 2017. URL <http://arxiv.org/abs/1606.04934>. arXiv:1606.04934 [cs, stat].
- [92] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Networks, June 2014. URL <http://arxiv.org/abs/1406.2661>. arXiv:1406.2661 [cs, stat].
- [93] Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes, December 2022. URL <http://arxiv.org/abs/1312.6114>. arXiv:1312.6114 [cs, stat].
- [94] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising Diffusion Probabilistic Models, December 2020. URL <http://arxiv.org/abs/2006.11239>. arXiv:2006.11239 [cs, stat].
- [95] Soroush Mehri, Kundan Kumar, Ishaan Gulrajani, Rithesh Kumar, Shubham Jain, Jose Sotelo, Aaron Courville, and Yoshua Bengio. SampleRNN: An Unconditional End-to-End Neural Audio Generation Model, February 2017. URL <http://arxiv.org/abs/1612.07837>. arXiv:1612.07837 [cs].
- [96] Jean-Marc Valin and Jan Skoglund. LPCNet: Improving Neural Speech Synthesis Through Linear Prediction, February 2019. URL <http://arxiv.org/abs/1810.11846>. arXiv:1810.11846 [cs, eess].
- [97] Zeyu Jin, Adam Finkelstein, Gautham J. Mysore, and Jingwan Lu. Fftnet: A real-time speaker-dependent neural vocoder. In *2018 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2018 - Proceedings*, pages 2251–2255. Institute of Electrical and Electronics Engineers Inc., September 2018. doi: 10.1109/ICASSP.2018.8462431. URL <https://collaborate.princeton.edu/en/publications/fftnet-a-real-time-speaker-dependent-neural-vocoder>.
- [98] Bohan Zhai, Tianren Gao, Flora Xue, Daniel Rothchild, Bichen Wu, Joseph E. Gonzalez, and Kurt Keutzer. SqueezeWave: Extremely Lightweight Vocoders for On-device Speech Synthesis, January 2020. URL <http://arxiv.org/abs/2001.05685>. arXiv:2001.05685 [cs, eess].
- [99] Chris Donahue, Julian McAuley, and Miller Puckette. Adversarial Audio Synthesis, February 2019. URL <http://arxiv.org/abs/1802.04208>. arXiv:1802.04208 [cs].
- [100] Mikolaj Bińkowski, Jeff Donahue, Sander Dieleman, Aidan Clark, Erich Elsen, Norman Casagrande, Luis C. Cobo, and Karen Simonyan. High Fidelity Speech Synthesis with Adversarial Networks, September 2019. URL <http://arxiv.org/abs/1909.11646>. arXiv:1909.11646 [cs, eess].
- [101] Jinhyeok Yang, Junmo Lee, Youngik Kim, Hoonyoung Cho, and Injung Kim. VocGAN: A High-Fidelity Real-time Vocoder with a Hierarchically-nested Adversarial

- Network, July 2020. URL <http://arxiv.org/abs/2007.15256>. arXiv:2007.15256 [cs, eess].
- [102] Alexey A. Gritsenko, Tim Salimans, Rianne van den Berg, Jasper Snoek, and Nal Kalchbrenner. A Spectral Energy Distance for Parallel Speech Synthesis, October 2020. URL <http://arxiv.org/abs/2008.01160>. arXiv:2008.01160 [cs, eess, stat].
- [103] Nanxin Chen, Yu Zhang, Heiga Zen, Ron J. Weiss, Mohammad Norouzi, and William Chan. WaveGrad: Estimating Gradients for Waveform Generation, October 2020. URL <http://arxiv.org/abs/2009.00713>. arXiv:2009.00713 [cs, eess, stat].
- [104] Zehua Chen, Xu Tan, Ke Wang, Shifeng Pan, Danilo Mandic, Lei He, and Sheng Zhao. InferGrad: Improving Diffusion Models for Vocoder by Considering Inference in Training, February 2022. URL <http://arxiv.org/abs/2202.03751>. arXiv:2202.03751 [cs, eess].
- [105] Xin Wang, Hong Chen, Si'ao Tang, Zihao Wu, and Wenwu Zhu. Disentangled Representation Learning, June 2024. URL <http://arxiv.org/abs/2211.11695>. arXiv:2211.11695 [cs].
- [106] Christopher P. Burgess, Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. Understanding disentangling in β -VAE, April 2018. URL <http://arxiv.org/abs/1804.03599>. arXiv:1804.03599 [cs, stat].
- [107] Hyunjik Kim and Andriy Mnih. Disentangling by Factorising, July 2019. URL <http://arxiv.org/abs/1802.05983>. arXiv:1802.05983 [cs, stat].
- [108] Xi Chen, Yan Duan, Rein Houthoofd, John Schulman, Ilya Sutskever, and Pieter Abbeel. InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets, June 2016. URL <http://arxiv.org/abs/1606.03657>. arXiv:1606.03657 [cs, stat].
- [109] Tero Karras, Samuli Laine, and Timo Aila. A Style-Based Generator Architecture for Generative Adversarial Networks, March 2019. URL <http://arxiv.org/abs/1812.04948>. arXiv:1812.04948 [cs, stat].
- [110] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum Contrast for Unsupervised Visual Representation Learning, March 2020. URL <http://arxiv.org/abs/1911.05722>. arXiv:1911.05722 [cs].
- [111] Ben Poole, Sherjil Ozair, Aaron van den Oord, Alexander A. Alemi, and George Tucker. On Variational Bounds of Mutual Information, May 2019. URL <http://arxiv.org/abs/1905.06922>. arXiv:1905.06922 [cs, stat].
- [112] Wei-Ning Hsu, Yu Zhang, Ron J. Weiss, Yu-An Chung, Yuxuan Wang, Yonghui Wu, and James Glass. Disentangling Correlated Speaker and Noise for Speech Synthesis via Data Augmentation and Adversarial Factorization. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5901–5905, February 2019. doi: 10.1109/ICASSP.2019.8683561. URL <https://ieeexplore.ieee.org/abstract/document/8683561>. ISSN: 2379-190X.

- [113] Kaizhi Qian, Yang Zhang, Shiyu Chang, David Cox, and Mark Hasegawa-Johnson. Unsupervised Speech Decomposition via Triple Information Bottleneck, March 2021. URL <http://arxiv.org/abs/2004.11284>. arXiv:2004.11284 [cs, eess].
- [114] Chen Zhang, Yi Ren, Xu Tan, Jinglin Liu, Kejun Zhang, Tao Qin, Sheng Zhao, and Tie-Yan Liu. DenoiSpeech: Denoising Text to Speech with Frame-Level Noise Modeling, December 2020. URL <http://arxiv.org/abs/2012.09547>. arXiv:2012.09547 [cs, eess].
- [115] Raza Habib, Soroosh Mariooryad, Matt Shannon, Eric Battenberg, R. J. Skerry-Ryan, Daisy Stanton, David Kao, and Tom Bagby. Semi-Supervised Generative Modeling for Controllable Speech Synthesis, October 2019. URL <http://arxiv.org/abs/1910.01709>. arXiv:1910.01709 [cs, eess].
- [116] Seungwoo Choi, Seungju Han, Dongyoung Kim, and Sungjoo Ha. Attentron: Few-Shot Text-to-Speech Utilizing Attention-Based Variable-Length Embedding, August 2020. URL <http://arxiv.org/abs/2005.08484>. arXiv:2005.08484 [cs, eess].
- [117] Siddharth Gururani, Kilol Gupta, Dhaval Shah, Zahra Shakeri, and Jervis Pinto. Prosody Transfer in Neural Text to Speech Using Global Pitch and Loudness Features, May 2020. URL <http://arxiv.org/abs/1911.09645>. arXiv:1911.09645 [cs, eess].
- [118] Ju-chieh Chou, Cheng-chieh Yeh, Hung-yi Lee, and Lin-shan Lee. Multi-target Voice Conversion without Parallel Data by Adversarially Learning Disentangled Audio Representations, June 2018. URL <http://arxiv.org/abs/1804.02812>. arXiv:1804.02812 [cs, eess].
- [119] Chin-Cheng Hsu, Hsin-Te Hwang, Yi-Chiao Wu, Yu Tsao, and Hsin-Min Wang. Voice Conversion from Unaligned Corpora using Variational Autoencoding Wasserstein Generative Adversarial Networks, June 2017. URL <http://arxiv.org/abs/1704.00849>. arXiv:1704.00849 [cs].
- [120] Leyuan Qu, Taihao Li, Cornelius Weber, Theresa Pekarek-Rosin, Fuji Ren, and Stefan Wermter. Disentangling Prosody Representations with Unsupervised Speech Reconstruction, September 2023. URL <http://arxiv.org/abs/2212.06972>. arXiv:2212.06972 [cs, eess].
- [121] Kei Akuzawa, Yusuke Iwasawa, and Yutaka Matsuo. Expressive Speech Synthesis via Modeling Expressions with Variational Autoencoder, February 2019. URL <http://arxiv.org/abs/1804.02135>. arXiv:1804.02135 [cs, eess].
- [122] Michael McAuliffe, Michaela Socolof, Sarah Mihuc, Michael Wagner, and Morgan Sonderegger. Montreal Forced Aligner: Trainable Text-Speech Alignment Using Kaldi. pages 498–502, 2017. doi: 10.21437/Interspeech.2017-1386. URL https://www.isca-archive.org/interspeech_2017/mcauliffe17_interspeech.html.
- [123] PyTorch Documentation. torch.nn.embedding, 2023. URL <https://pytorch.org/docs/stable/generated/torch.nn.Embedding.html>.