

國立臺灣大學電機資訊學院電信工程學研究所

博士論文

Graduate Institute of Communication Engineering
College of Electrical Engineering and Computer Science

National Taiwan University

Doctoral Dissertation

高效率語音生成：運算效率、資料效率及其在語音自監督學習中的應用

Efficient Speech Generation: Computational Efficiency,
Data Efficiency, and Its Application in Speech
Self-Supervised Learning

許博竣

Po-chun Hsu

指導教授：李琳山 教授

共同指導教授：李宏毅 教授

Advisor: Lin-shan Lee, Ph.D.

Co-Advisor: Hung-yi Lee, Ph.D.

中華民國 113 年 7 月

July, 2024





誌謝

2024年夏，博士論文的完成標誌著人生邁入下一個里程碑。大學四年，碩博連讀六年，不知不覺已在台北生活十年。回首這段求學路，自己是何其幸運，總在山窮水盡之時，值遇師長與同行善友，指引我走向一村柳暗花明。感念一路上得之於人者太多，僅能略記隻言片語，以表謝意。

當初在猶豫是否逕讀博士班時，父母便表示會全力支持我的選擇。爾後在面對投稿失利、實驗瓶頸，以及對於能否順利畢業充滿憂慮和不安時，他們始終給予我無限的包容和精神支持，成為我無後顧之憂追求學位的堅強後盾。感謝他們的無私奉獻。

在實驗室期間，感謝同儕的陪伴與支持。無論是修課及研究的討論，或是日常瑣事的分享、抱怨和互相幫助，這段生活因此而飽滿。特別感謝達融和淞楓學長，他們在期刊投稿、畢業論文撰寫，以及面對壓力時的心態調適等方面，提供了許多具體的建議與幫助，使我得以堅持到底。也感謝彤恩姐，她的辛勞打點讓實驗室所有的研究和活動順利進行。

最後，感謝博士班期間誨我甚深的三位導師——李琳山老師、李宏毅老師，以及我在實習期間的主管 Abdelrahman Mohamed 博士。李琳山老師及李宏毅老師為實驗室創造了良好的學習環境，除了提供充足的資源，同學間自由研究和積極討論的氛圍，促使我在這幾年不斷成長。此外，更能從三位學業界導師身上看到

一流學者所應具備的能力及風範，包含做研究的態度、問題的發現與解決以及表達能力的重要性。能與他們合作研究並受其指導，無疑是這六年來最大的收穫。

感謝這一路上所有給予我支持與幫助的人，你們的存在讓這段旅程更加豐富和充實。我會將這份感激銘記於心，繼續努力前行。






摘要

近年來，隨著深度學習的進步，許多語音生成模型展現了出色的表現。儘管取得了亮眼的成果，語音生成技術的發展也伴隨了對運算和資料資源的更大需求，導致其效率受到了限制。本論文旨在從各個方面解決高效語音生成所面臨的挑戰，包括運算效率、資料效率及其在資料高效的語音自監督學習（Self-Supervised Learning, SSL）中的應用。

我們首先關注語音生成的運算效率。我們提出了一種高度壓縮的非自迴歸神經聲碼器 (Neural Vocoder)，顯著減少了模型大小和訓練所需的運算資源。將改進的架構與額外的後置濾波器相結合，提出的模型無需依賴 GPU 加速，即可實現實時推理和高品質語音輸出。該模型不僅在生成 44 kHz 語音方面展現了卓越的性​​能，更為高效語音合成樹立了新的基準。

接下來，我們探索自迴歸生成機制，並提高其推理效率。我們引入了創新方法，頻率自迴歸生成 (Frequency-wise Autoregressive Generation, FAR) 和位自迴歸生成 (Bit-wise Autoregressive Generation, BAR)，它們分別在不同的域進行自迴歸生成。這些方法大大提高了推理速度，同時保持了良好的語音品質。除了用於神經聲碼器之外，所提出的技術亦可能適用於其他語音生成任務，包括用於自迴歸模型以提高推理效率和用於非自迴歸模型以提高輸出品質，從而擴大其影響。

隨後我們將重點轉向資料效率，解決在收集用於文字引導語音轉換 (Text-



Guided Voice Conversion) 的標記資料時所面臨的高成本問題。我們引入強化學習 (Reinforcement Learning, RL) 和基於人類回饋的強化學習 (Reinforcement Learning from Human Feedback, RLHF) 來增強生成語音的表現力。我們的方法減少了對大型標記資料集的依賴，並提高了模型處理複雜風格的文字描述和產生富有表現力的語音的能力，從而在客觀和主觀評估方面展現了顯著改進。

最後，我們擴展了研究範圍，透過語音生成技術提高語音 SSL 中的資料效率。我們利用高品質文字轉語音系統產生的合成語音資料來增強低資源的 (Low-Resource) 預訓練 (Pre-training) 語料庫，減少對大量現實世界語音資料的需求。提出的方法表明，合成資料可以有效地補充真實資料，從而以更少的資源達到具有競爭力的性能。

總體而言，本論文對提高語音生成效率及其在語音處理中的應用做出了重大貢獻。我們引入新穎的架構、生成方法及學習範式來解決運算和資料效率的挑戰，為該領域的未來進步奠定基礎。

關鍵字：語音生成、運算效率、資料效率

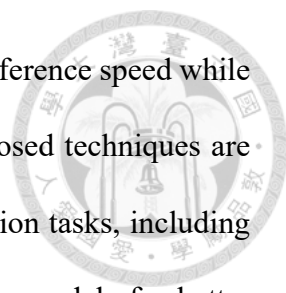


Abstract

Speech generation models have achieved outstanding performance with the advancement of deep learning in recent years. Despite the remarkable achievements, the development of speech generation technology has also been accompanied by greater demands on computational and data resources, resulting in limitations in its efficiency. This thesis aims to address the challenge of achieving efficient speech generation from various aspects, including computational efficiency, data efficiency, and its application for data-efficient speech self-supervised learning (SSL).

We first focus on the computational efficiency of speech generation. We propose a highly compressed non-autoregressive neural vocoder, significantly reducing model size and computational resources for training. By integrating the improved architecture with an additional post-filter, the proposed model achieves high-quality speech output with real-time inference capabilities without relying on GPU acceleration. This model not only demonstrates superior performance in generating 44 kHz speech but also sets a new benchmark for efficient speech synthesis.

Next, we explore autoregressive generation mechanisms and enhance inference efficiency. We introduce innovative methods, Frequency-wise Autoregressive Generation (FAR) and Bit-wise Autoregressive Generation (BAR), which perform the autoregressive



processes in different domains. These methods drastically improve inference speed while maintaining high speech quality. Besides neural vocoders, the proposed techniques are versatile and have the potential to be applied to other speech generation tasks, including autoregressive models for efficient inference and non-autoregressive models for better quality, thereby broadening their impact.

Then, we shift the focus to data efficiency, addressing the high costs associated with collecting labeled data for text-guided voice conversion. We introduce reinforcement learning (RL) and reinforcement learning from human feedback (RLHF) to enhance the expressiveness of generated speech. Our approach reduces the dependency on large, labeled datasets and improves the model's ability to handle text descriptions of complex speech styles and generate expressive speech, achieving significant improvements in both objective and subjective evaluations.

Lastly, we extend the scope of our research to improve data efficiency in speech SSL with speech generation techniques. By leveraging synthetic speech data generated from a high-quality text-to-speech system, we augment the low-resource pre-training corpus, reducing the need for extensive real-world speech data. The proposed approach demonstrates that synthetic data can effectively supplement real data, enabling competitive performance with significantly fewer resources.

Overall, this thesis makes substantial contributions to enhancing the efficiency of speech generation and its applications in speech processing. We introduce novel architectures, generation methods, and learning paradigms that address computational and data efficiency challenges, setting the stage for future advancements in the field.

Keywords: Speech Generation, Computational Efficiency, Data Efficiency



Contents

誌謝	i
摘要	iii
Abstract	v
Contents	vii
List of Figures	xiii
List of Tables	xvii
Chapter 1 Introduction	1
1.1 Motivation	1
1.2 Contribution	4
1.3 Overview	7
Chapter 2 Background	9
2.1 Neural Vocoder	9
2.1.1 Overview	9
2.1.2 Autoregressive Neural Vocoder	10
2.1.3 Non-Autoregressive Neural Vocoder	12
2.1.4 Efficiency, Quality, and Stability of Neural Vocoders	13
2.2 Text-Guided Speech Generation	16
2.2.1 Overview	16



2.2.2	Text-Guided Text-to-Speech Synthesis	17
2.2.3	Text-Guided Voice Conversion	18
2.2.4	Data Resources of Text-Guided Speech Generation	22
2.3	Self-Supervised Learning in Speech Processing	22
2.3.1	Overview	22
2.3.2	Data Resources of Self-Supervised Learning in Speech Processing	24
Chapter 3 Computational Efficiency in Speech Generation: Better Architecture Design for Non-Autoregressive Neural Vocoder		25
3.1	Introduction	25
3.2	Related Work	27
3.2.1	WaveGlow: A Flow-Based Neural Vocoder	27
3.3	Method	31
3.3.1	Highly Compressed WaveGlow	31
3.3.2	WaveNet-Based Post-Filter	33
3.4	Experimental Setup	35
3.4.1	Datasets	35
3.4.2	Model Details	35
3.4.3	Evaluation Metrics	37
3.5	Results	38
3.5.1	Speed and Computational Cost	38
3.5.2	Audio Quality Comparison	39
3.5.3	High-Fidelity Audio Generation	40
3.5.4	Text-to-Speech	42



3.6	Summary	43
Chapter 4 Computational Efficiency in Speech Generation: Applying Non-Autoregressive Mechanism for Efficient Autoregressive Generation		45
4.1	Introduction	45
4.2	Method	50
4.2.1	Rethinking the Direction for Autoregressive Generation	50
4.2.2	Frequency-wise Autoregressive Generation (FAR)	52
4.2.3	Bit-wise Autoregressive Generation (BAR)	54
4.2.4	Proposed Vocoder Architecture	56
4.2.5	Post-filtering for Posterior Sampling	59
4.3	Experimental Setup	63
4.3.1	Dataset	63
4.3.2	Acoustic Feature	64
4.3.3	Model Details	64
4.3.4	Evaluation Metrics	70
4.4	Results	71
4.4.1	Objective Evaluation	71
4.4.2	Subjective Evaluation	84
4.4.3	Generalization Evaluation	87
4.5	Summary	90
Chapter 5 Data Efficiency in Speech Generation: Improving Text-Guided Voice Conversion with Reinforcement Learning and Human Feedback		93
5.1	Introduction	93

5.2	Related Work	96
5.2.1	Diffusion-Based Text-to-Audio Generation	96
5.2.2	Reinforcement Learning in Speech Processing	97
5.2.3	Reinforcement Learning and Human Feedback in Audio Generation	98
5.3	Text-Guided Voice Conversion	99
5.3.1	Modifying a Pre-Trained Model for Text-Guided Voice Conversion .	100
5.3.2	Duration Model Fine-Tuning	104
5.4	Improving Model Performance with Reinforcement Learning and Human Feedback	104
5.4.1	Reward Model	105
5.4.2	Denoising Diffusion Policy Optimization	107
5.4.3	Reinforcement Learning from Human Feedback	109
5.5	Datasets	110
5.5.1	PromptSpeech Dataset	110
5.5.2	Emotion Datasets	111
5.5.3	Accent Dataset	112
5.5.4	Sound Event Dataset	113
5.6	Experimental Setup	113
5.6.1	Data Preprocessing	113
5.6.2	Models and Training	114
5.6.3	Evaluation Tasks	117
5.7	Results	119
5.7.1	Objective Evaluation	119



5.7.2	Subjective Evaluation	125
5.8	Summary	127



Chapter 6	Enhancing Data Efficiency in Self-Supervised Learning with Speech Generation	129
6.1	Introduction	130
6.2	Related Work	131
6.2.1	Self-Supervised Learning with Unpaired Text Data for Speech Processing	131
6.2.2	Speech Generation for Data Augmentation	132
6.3	Method	133
6.3.1	Overview	133
6.3.2	Extracting Discrete Speech Representation	134
6.3.3	Text-to-speech Modules	135
6.4	Experimental Setup	139
6.4.1	Datasets	139
6.4.2	Comparing Systems	140
6.4.3	Evaluation Methods	141
6.5	Results	142
6.5.1	Performance without Synthetic Data	142
6.5.2	Performance of Off-the-Shelf TTS Methods	143
6.5.3	Performance of the Proposed System	145
6.6	Summary	149

Chapter 7 Conclusion

7.1 Thesis Summary 151

7.1.1 Achievements of Each Chapter 152

7.1.2 Impact to Subsequent Works 153

7.2 Future Work 155

7.2.1 Exploring Applications and New Domains for Redesigned Autoregressive Generation 155

7.2.2 Expanding Applications of Reinforcement Learning in Speech Generation 156

7.2.3 Extending Iterative Training for Self-Supervised Learning and Speech Generation 157

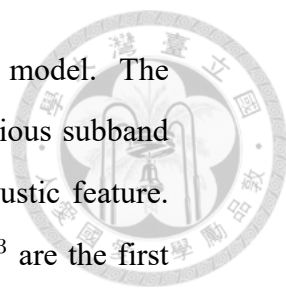
References **159**





List of Figures

3.1	(a) WaveGlow network. (b) WaveGlow network (inverted). The WN module also takes Mel-spectrograms as input, which are omitted in this figure.	29
3.2	(a) Architecture of WG-WaveNet. (b) Training of WG-WaveNet.	32
3.3	Trade-off between MOS and GPU inference speed.	40
4.1	Overview of different autoregressive methods and their orders of generation. (a) Conventional autoregressive generation. Samples at different time steps are generated sequentially. (b) Frequency-wise autoregressive generation (FAR). Subbands are first generated autoregressively and combined to form the full-band waveform. (c) Bit-wise autoregressive generation (BAR). The spectrograms in (b) and Mel-spectrograms in (c) are only for visualization and not for generation.	47
4.2	(a) Conventional autoregressive model. Each blue circle represents a scalar. (b) Proposed autoregressive model. The speech is generated iteratively in the frequency domain or the bit precision domain. Each green block is the model illustrated in Figure 4.5 (a), and each orange block represents a time series.	51
4.3	Frequency-wise autoregressive generation.	52
4.4	Bit-wise autoregressive generation.	54



4.5 (a) Overview and detailed block diagram of the proposed model. The model predicts the i th subband x^i conditioned on the previous subband x^{i-1} . h^i is the hidden state, and f^i is the upsampled acoustic feature. BAR is integrated in each FAR prediction, and $b^{i,1}, b^{i,2}, b^{i,3}$ are the first three bits of x^i . Channel sizes are shown in gray and parenthesized. (b) WN module. 55

4.6 Different methods to sample output from posteriorgram. **Argmax**: Choosing the category with the greatest probability. **Sample**: Sampling according to the probability distribution. **Post-Filtering**: Using a network to predict 16-bit samples. 58

4.7 Post-filter and two paths for training. Channel sizes are shown in gray and parenthesized. 60

4.8 Average pooling in L_S . (a) STFT magnitude. (b) Band-limited STFT magnitude. 62

4.9 WN module with the grouping mechanism (g-5). Some operations are omitted for simplicity. (c, t) represents a t -length sequence of c -dimensional vectors. 68

4.10 Mel-spectrograms of generated speech using ground truth acoustic features from LJ Speech (LJ050-0241). 79

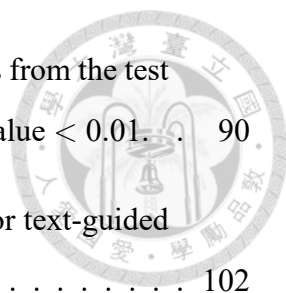
4.11 MUSHRA results when using ground truth acoustic features from the test set of LJ Speech. *: p-value < 0.05; **: p-value < 0.01. 83

4.12 MUSHRA results when using ground truth acoustic features from the test set of LJ Speech. To compare with LPCNet, ground truth and generated utterances were downsampled to 16 kHz before evaluating. **: p-value < 0.01. 85

4.13 MUSHRA results when using acoustic features generated from Tacotron 2. **: p-value < 0.01. 86

4.14 MUSHRA results when using ground truth acoustic features from speaker *bdl* and *slt* in CMU ARCTIC. The vocoders were trained on VCTK corpus. **: p-value < 0.01. 89

4.15	MUSHRA results when using ground truth acoustic features from the test set of our 44 kHz internal mandarin speech corpus. **: p-value < 0.01. . .	90
5.1	Overview of modifying a pre-trained text-to-audio model for text-guided voice conversion.	102
5.2	(a) Architecture of the unit encoder. (b) Architecture of the duration model. Note that ground-truth instead of predicted duration is used as input to the range predictor and the Gaussian upsampling module at training time. . .	103
5.3	(a) Overview of the RL pipeline. The text-guided voice conversion model is optimized with the reward model and RL algorithm. (b) Overview of the RLHF process, including collecting human preferences and building a reward model with human feedback. The reward model learns from human feedback and is used to optimize the text-guided voice conversion model with the same RL algorithm as in (a).	108
6.1	Overview of the proposed system.	133

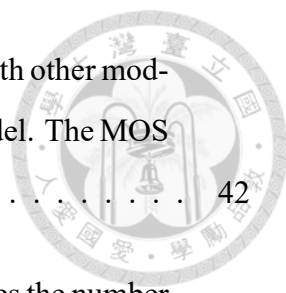






List of Tables

2.1	Example text descriptions in different works. Descriptions in other languages (Lang.) are translated into English.	19
2.2	Details of text-guided speech generation models, including task, language (Lang.), attributes specified in text descriptions (Attr.). The asterisks (*) indicate that the training datasets or codes for building datasets are open-source. -: not reported.	20
2.3	Details of text-guided speech generation datasets reported in different works, including language (Lang.), data source (Src.), total duration (Dur., in hours), number of utterances (Utr.), and number of speakers (Spk.). -: not reported.	21
3.1	Parameters for calculating L_s (reported in samples).	36
3.2	Comparison of model sizes.	36
3.3	Comparison of computational cost and speed during training and inference time. The units of memory, time, and speed are GB, days, and kHz, respectively.	38
3.4	MOS and MCD results compared with other models. Mel-spectrograms were extracted from the ground truth. The MOS results are reported with 95% confidence intervals.	39
3.5	MOS results of high-fidelity audio generation with 95% confidence intervals. Mel-spectrograms were extracted from the ground truth sampled at 44 kHz.	41



3.6 MOS results and GPU inference speed (in kHz) compared with other models. Mel-spectrograms were generated by the Tacotron 2 model. The MOS results are reported with 95% confidence intervals. 42

4.1 Details of the multi-resolution STFT auxiliary loss. m denotes the number in L_{sc}^m and L_{mag}^m 67

4.2 Details of different models. For Parallel WaveGAN, only the parameters of the generator are counted. 69

4.3 Details and inference speed (w/o and w/ GPU) of different models. The values in the brackets are the standard deviations. 72

4.4 Objective evaluation results (means and standard deviations) when using ground truth acoustic features from the test set of LJ Speech. p-value < 0.05 ; p-value < 0.01 74

4.5 Objective evaluation results (means and standard deviations) when using ground truth acoustic features from the test set of LJ Speech. To compare with LPCNet, ground truth and generated utterances were downsampled to 16 kHz before evaluating. p-value < 0.05 ; p-value < 0.01 75

4.6 Ablation study results (means and standard deviations) when using ground truth acoustic features from the test set of LJ Speech. p-value < 0.05 ; p-value < 0.01 78

4.7 Objective evaluation results (means and standard deviations) when using ground truth acoustic features from speaker *bdl* and *slt* in CMU ARCTIC. The vocoders were trained on VCTK corpus. p-value < 0.05 ; p-value < 0.01 . 88

4.8 Objective evaluation results (means and standard deviations) when using ground truth acoustic features from the test set of our 44 kHz internal mandarin speech corpus. p-value < 0.05 ; p-value < 0.01 89

5.1 Details of datasets used in this work, including total duration (Dur., in hours), number of utterances (Utrr.), number of speakers (Spk.), and contained styles. 111



5.2 Configurations to train the reward model with human feedback. **Size:** size of the human feedback dataset. **lr.:** learning rate. **bsz.:** batch size to sample data from the human feedback dataset. **Start Step:** step to start updating the model with L_{hf} . **Update Steps:** every how many steps to update the model with L_{hf} . **Max Steps:** total training steps. **Best Step:** step of the best checkpoint. 116

5.3 Style classification and speaker similarity results on different datasets. . . 120

5.4 Style classification results of the proposed model. **Base** denotes the model after the joint training and duration model fine-tuning. **b32** uses a batch size of 32. **Base-RL** applies RL and only updates the LoRA layers in the diffusion model. **+ue** and **+dur.** further apply LoRA to the SMHA pooling layers for the unit encoder and the duration model, respectively. . 121

5.5 Classification results of the reward models trained or fine-tuned with different amounts of human feedback. Human Feedback is a validation set separated from the collected human feedback data. **orig.** denotes the original reward model built without human feedback data. Details of other validation sets and reward models can be found in Section 5.5 and Table 5.2, respectively. 123

5.6 Style classification results of the proposed models without RL, with RL, and with RLHF. 124

5.7 Speaker similarity results of the proposed models without RL, with RL, and with RLHF. 125

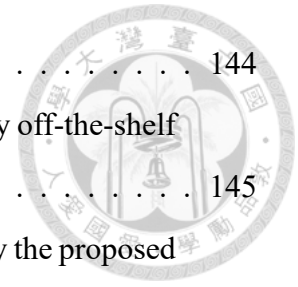
5.8 Subjective evaluation results of the proposed models without RL, with RL, and with RLHF. All scores are reported with 95% confidence intervals. 126

5.9 Subjective pairwise preference test results. A percent preference greater than 50% indicates the proposed RLHF model is preferred over **Base-RL**. All results are reported with 95% confidence intervals. 126

6.1 Average length ratios (unit length / phoneme length) with different post-processing methods. 134

6.2 Results of pre-training on natural speech data. 143

6.3	Quality analysis of discrete units.	144
6.4	Results of pre-training on synthetic speech data generated by off-the-shelf TTS methods.	145
6.5	Results of pre-training on synthetic speech data generated by the proposed method.	147
6.6	Results of pre-training and fine-tuning on synthetic speech data generated by the proposed method.	148





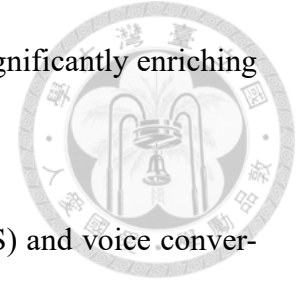
Chapter 1

Introduction

1.1 Motivation

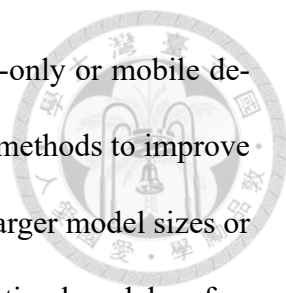
Speech is one of the most commonly used mediums for human communication. Compared with writing and reading, speaking and listening skills are mastered by most individuals earlier when they grow up. In daily life, dialogue is also a more efficient and preferred method of communication in most scenarios than text messages. The high efficiency of speech communication is not solely attributed to the ability of humans to rapidly produce, perceive, and comprehend sounds. The nuances in intonation, intensity, and speaking rate embedded within speech enhance the transmission of messages, such as accents or emotions, that are challenging to encapsulate in written form. In addition to the way humans communicate with each other, recently, the way humans interact with machines is shifting gradually to more intuitive spoken languages from traditional programming languages, buttons, or text menus. Speech generation technologies play an increasingly important role in human-computer interaction. These technologies not only facilitate seamless communication between digital entities and humans but also enable machines to deliver a wider

variety of information more swiftly and naturally through speech, significantly enriching user experience by emulating human-like interactions.



In the field of speech generation, text-to-speech synthesis (TTS) and voice conversion (VC) stand out as the primary tasks, with TTS focusing on converting plain text into audible speech and VC on modifying voice characteristics such as speaker identity, accent, speaking style, or emotional tone. Typically, the process for these tasks involves two key components: a task-specific **synthesizer** to generate target acoustic features and a **vocoder** to recover speech waveforms from these acoustic features. These components can be built either separately [1, 2] or jointly [3]. With deep learning achieving prominent performance in computer vision (CV) and natural language processing (NLP), neural networks (NN) have also dominated the field of speech processing and become the mainstream speech generation architecture. This evolution has led to significantly better quality of generated speech, making it more natural, expressive, and closer to human-like speech than ever before. These advancements can be attributed to innovations in computing hardware (particularly GPUs), advanced architectures, growing model sizes, and the availability of high-quality datasets. Despite the remarkable achievements, such developments have also introduced new challenges and limitations to the efficiency of different aspects:

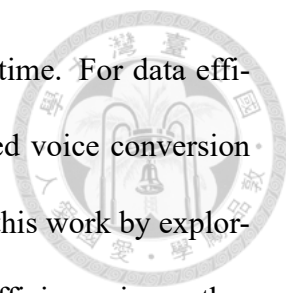
1. **Computational Efficiency:** The early adoption of neural network-based vocoders, or neural vocoders, marked a significant leap in speech generation quality [1]. These neural vocoders apply an autoregressive architecture to generate waveform samples one by one [4, 5], making them with extremely slow inference speed since there are tens of thousands of samples in a one-second utterance. The low inference efficiency makes real-time speech generation challenging, especially on de-



vices without powerful computational resources, such as CPU-only or mobile devices. Some subsequent works focused on non-autoregressive methods to improve inference speed [6, 7]. However, these methods often require larger model sizes or increased computational resources at training time to achieve optimal model performance. The eager demand for computational resources at training time raises the threshold and difficulty of building models for different speech generation tasks, and the low efficiency at inference time directly affects the user experience and limits the applicability of speech generation technologies.

2. **Data Efficiency:** The capability of neural networks to model complex data across various speech tasks is well-recognized. However, this effectiveness heavily depends on the availability of large and high-quality real-world datasets, which often necessitate considerable efforts to collect for specific tasks. For text-to-speech synthesis, clean and well-labeled text-speech paired data is essential, while voice conversion tasks require datasets with diverse speaker styles. More complicated speech generation tasks, such as text-guided TTS [8, 9] or VC [10, 11], require human efforts to collect speech data of different speaking styles and carefully label text descriptions for them. When building these versatile speech generation models, in addition to architecture design and computational efficiency optimization, it is challenging to reduce efforts for data collection, alleviate reliance on large datasets, and utilize available data more effectively.

This thesis focuses on investigating and tackling the inefficiency of speech generation from different aspects, mitigating the challenges posed by computational resource demands and data scarcity when constructing speech generation applications. We start with improving the architectures of neural vocoders and inventing better generation mech-

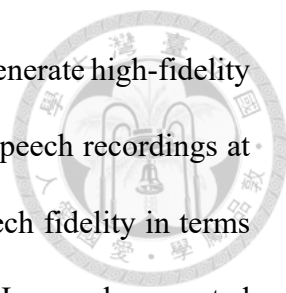


anisms, boosting computational efficiency at training and inference time. For data efficiency, we introduce an innovative learning paradigm for text-guided voice conversion under a low-resource scenario. Furthermore, we extend the scope of this work by exploring the application of speech generation methods to improve data efficiency in another speech task, speech self-supervised learning (SSL) [12, 13], extending the contributions of this thesis from optimizing the efficiency of speech generation to enhancing data efficiency with speech generation techniques.

1.2 Contribution


Focusing on different aspects of efficiency, the main contributions made by this thesis are summarized as follows:

1. **Computational efficiency: designing a more compact non-autoregressive neural vocoder architecture for faster training, real-time inference, and high-quality 44 kHz speech.** We propose a hybrid neural vocoder model containing two modules. The first is a compact WaveGlow model [6], which is highly compressed to reduce model size and required training GPU memory. To further improve the speech quality and convergence speed, we adopt a lightweight WaveNet-based post-filter [4]. These modules are optimized jointly with the specially designed losses and training process to ensure convergence. The proposed model is efficient and economical at both training and inference time. In particular, the proposed model is 97.2% smaller in size compared with the original WaveGlow, not only requiring significantly less GPU memory and training time but also reaching 1.5 times faster than real-time without any GPU acceleration during inference.



Furthermore, we explore the ability of the proposed method to generate high-fidelity 44 kHz speech. We first assess the naturalness of real-world speech recordings at different sampling rates, demonstrating the importance of speech fidelity in terms of human perception. Then, we evaluate the quality of the 44 kHz speech generated from neural vocoders and improve the performance by providing acoustic features with higher temporal resolution when training neural vocoders.

- 2. Computational efficiency: rethinking the autoregressive generation mechanism and refining its inference efficiency.** We explore the properties and the effectiveness of autoregressive generation. Based on our observations and assumptions, we proposed a novel concept for autoregressive generation. Instead of in the time domain, the proposed model conducts autoregressive speech generation in the frequency and bit precision domains more efficiently. In addition, we combine the characteristics of the proposed autoregressive methods to build a post-filter for better speech quality. Compared with its conventional autoregressive counterpart, the proposed model exhibits comparable high speech quality while achieving inference speeds up to 244 and 9669 times faster with and without GPU acceleration, respectively, which matches the speeds of non-autoregressive methods.
- 3. Data efficiency: introducing a more data-efficient learning paradigm for text-guided voice conversion.** We enhance the data efficiency of text-guided voice conversion by introducing reinforcement learning (RL) and reinforcement learning with human feedback (RLHF). To address the challenge of collecting expressive speech data and extensive human-labeled style descriptions, we first utilize diverse, publicly available datasets to build a text-guided voice conversion model and a reward model. The reward model is built with contrastive learning and can do zero-shot



classification for more complex speech styles unseen during training. We specially design a process to use the reward model to assess the speech styles of converted utterances. Then, by utilizing the RL algorithm and the discrimination from the reward model, we improve the voice conversion model's ability to generate speech when given more complex style descriptions, facilitating the model to learn more and better from the limited resources without collecting more high-quality labeled data. Lastly, we further introduce human feedback to enhance the reward model, which then guides the voice conversion model to generate expressive speech that is more aligned with human preferences. The proposed method improves the objective metrics by up to 100% and reaches better subjective scores with statistical significance. The experimental results indicate that the proposed approach reduces the reliance on costly, human-annotated data, optimizes the training process, and expands the model's versatility in interpreting and applying various speech styles effectively.

- 4. Data efficiency: achieving low-resource speech self-supervised learning with the aid of speech generation.** We demonstrate the impact of data scarcity for building a speech SSL model, including the challenges of overfitting and worse performance. In such a low-resource scenario, we propose a process to leverage discrete acoustic representations to construct a speech generation system more effectively. The speech generation system then generates large amounts of synthetic speech data, augmenting the pre-training corpora for speech SSL. Experimental results show that the proposed approach effectively reduces the demand for speech data by 90% with only slight performance degradation.

1.3 Overview



This thesis is organized as follows:

- Chapter 2 reviews the literature on topics related to this thesis, including neural vocoders, text-guided speech generation, and speech self-supervised learning.
- Chapter 3 introduces a compact, non-autoregressive neural vocoder that is computationally efficient at training and inference time.
- Chapter 4 introduces an innovative concept for autoregressive generation and proposes an autoregressive neural vocoder with faster inference speed.
- Chapter 5 explores applying reinforcement learning to text-guided voice conversion to improve data efficiency.
- Chapter 6 expands the focus on data efficiency from speech generation to speech self-supervised learning.
- Chapter 7 summarizes this thesis and discusses future research directions.





Chapter 2

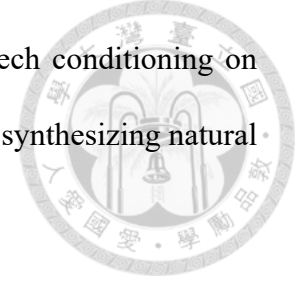
Background

2.1 Neural Vocoder

2.1.1 Overview

Neural speech generation has recently achieved remarkable audio qualities in different speech tasks such as text-to-speech (TTS) [2, 4, 14] and voice conversion (VC) [15]. These systems typically comprise two separate models: a synthesizer and a vocoder. A synthesizer is usually designed for some specific speech task and outputs acoustic features such as linear-scaled spectrograms, Mel-spectrograms, F0 frequencies, spectral envelopes, or aperiodicity information [1, 16–18]. A vocoder is designed to reconstruct audio waveforms from the acoustic features [4, 5, 19]. In the early era of neural speech generation, a hand-crafted vocoder [19–21] or the Griffin-Lim algorithm [22] was adopted to reconstruct speech waveforms [16, 23–25]. However, speech signals have a high temporal resolution, and neither conventional vocoders nor heuristic methods can reconstruct high-quality natural speech, remaining modeling raw audio a challenging problem. In Tacotron

2 [1], a neural network [4] is used as the vocoder to generate speech conditioning on Mel-spectrograms and has shown the potential of neural vocoder for synthesizing natural human speech.



Neural vocoders, while proposed for the TTS system, are not limited to this usage. A neural vocoder can be used for a wide range of applications associated with speech generation, such as TTS [2, 16, 26], VC [15, 27], speech bandwidth extension (SBE) [28], and speech compression (SC) [29, 30]. With high-quality vocoders involved, these models can focus only on processing acoustic features for different speech tasks instead of directly manipulating speech waveforms. As more and more models process and output acoustic features while using another neural network to reconstruct speech, the research about using neural vocoders for waveform modeling becomes influential on the naturalness of generated speech [31–33].

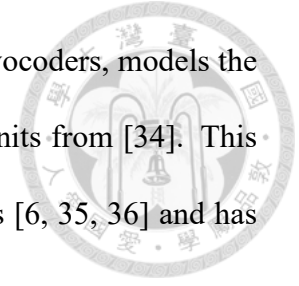
2.1.2 Autoregressive Neural Vocoder

In the beginning, most of the neural vocoders for speech generation are autoregressive. Given a waveform $\mathbf{x} = \{x_1, x_2, \dots, x_T\}$ and its acoustic features c , where x_t is the audio sample at time t , the joint probability of \mathbf{x} given c can be factorized as follows:

$$p(\mathbf{x}|c) = \prod_{t=1}^T p(x_t|c, x_1, x_2, \dots, x_{t-1}). \quad (2.1)$$

An autoregressive vocoder is built based on this factorization and trained to maximize $p(\mathbf{x}|c)$, where c can be extracted from real waveforms or predicted by a synthesizer. It generates future audio samples conditioning on previous ones to model long-term dependencies in speech waveform, resulting in an iterative generation process. WaveNet [4],

as one of the earliest and most representative autoregressive neural vocoders, models the waveform using dilated convolutional layers and gated activation units from [34]. This architecture has shown effectiveness in modeling speech waveforms [6, 35, 36] and has been widely applied in various speech processing tasks [37–39].



Some variants of autoregressive vocoders are later proposed to improve synthesis speed by leveraging more lightweight architectures and more efficient generation processes. FFTNet [40] improves efficiency by an architecture resembling the Fast Fourier Transform. In WaveRNN [5], the deep convolutional networks in WaveNet are replaced with smaller recurrent neural networks to model long-term dependencies. This work also introduces Sparse WaveRNN and Subscale WaveRNN to reduce the complexity at inference time. Inheriting the compact architecture of WaveRNN, LPCNet [41] adopts linear prediction to enhance the efficiency of speech generation. Based on LPCNet, Bunched LPCNet [42] proposed sample bunching and bit bunching to increase inference speed, and [43] applies tensor decomposition to reduce model parameters further.

Another branch applies the subband analysis technique and significantly improves efficiency [44–47]. By splitting a full-band waveform into several subband signals, autoregressive models can iterate on different subbands simultaneously and generate multiple samples in parallel. Based on subband LPCNet [47], [48] proposed to predict a subband signal conditioning on generated samples from the current and other subbands.

In addition to modifying model architectures and generation methods, some other works turn from Python frameworks to highly optimized implementations in C to improve inference efficiency^{1 2}. The efforts above have successfully improved the efficiency of

¹<https://github.com/NVIDIA/nv-wavenet>

²<https://github.com/xiph/LPCNet>

autoregressive vocoders. However, these autoregressive models are inherently serial and can not fully utilize parallel processors like GPUs or TPUs.



2.1.3 Non-Autoregressive Neural Vocoder

Recently, many non-autoregressive models have been proposed to better improve the inference efficiency of the neural vocoder. These methods discard the iterative generation process and target to simultaneously predict all samples x_1, x_2, \dots, x_T in a waveform \mathbf{x} . To this goal, various approaches are proposed to estimate and optimize the probability of \mathbf{x} given acoustic features c , $p(\mathbf{x}|c)$, without the factorization used in autoregressive generation.

Some works try to distill outputs of autoregressive models to non-autoregressive ones, for example, Parallel WaveNet [49] and Clarinet [50]. The student networks underlying both Parallel WaveNet and Clarinet are based on inverse autoregressive flow (IAF) [51]. Though the IAF network can run in parallel at inference time, the teacher-student-based knowledge distillation strategy makes the training process inefficient and the whole framework complicated for users to implement.

Inspired by the success of the flow-based model in image generation [52], WaveGlow [6] is proposed. Instead of the autoregressive process in IAF, WaveGlow adopts the affine coupling layer, which is more efficient, and the synthesis speed is 25 times faster than real-time. Another group of vocoders utilizes generative adversarial networks (GAN), which calculate the probability implicitly, for example, MelGAN [53] and Parallel WaveGAN [35]. Aside from the flow-based and GAN-base models, diffusion-based (or score-based) methods, such as DiffWave [36] and WaveGrad [54], model the multi-

step generation as a Markov process and refine the output waveform over steps, making it possible to trade off between inference speed and speech quality.



2.1.4 Efficiency, Quality, and Stability of Neural Vocoders

Autoregressive and non-autoregressive vocoders have different generation processes, resulting in their respective advantages and limitations regarding computational efficiency, quality, and stability.

Computational Efficiency

Autoregressive vocoders inherently generate samples sequentially, inevitably requiring tens of thousands of predictions to generate an utterance of a few seconds. The inference speed is thus drastically slower than real-time. Even in the faster variants[5, 40–48], efficiency remains a concern as these models cannot generate all samples in parallel due to the nature of autoregressive generation in the time domain. In contrast, non-autoregressive models generate all signal samples in one computation, allowing for real-time synthesis even without a GPU when the architectures are compact. With the parallel synthesis capability, non-autoregressive vocoders demonstrate a significantly faster inference speed than their autoregressive counterparts. Nevertheless, many non-autoregressive methods use more computational resources during training to reach decent performance [6, 7, 54], making it challenging to build applications with limited GPU budgets.

Quality



Even though non-autoregressive models are much faster at inference time, empirical evidence suggests that most of these methods are less performant than autoregressive ones.

To better support this claim, we survey the subjective evaluations in different famous non-autoregressive works, including flow-based, GAN-based, and diffusion-based models. We organize our observations as follows. All the comparisons are in terms of the naturalness of the generated speech.

• Flow-Based Model

- FloWaveNet and WaveFlow do not outperform WaveNet in their original papers [55, 56].
- Though the performance of WaveGlow is better than WaveNet in the original paper [6] and [53], it usually performs worse than WaveNet in most other recent works [7, 36, 56, 57].

• GAN-Based Model

- MelGAN performs worse than WaveNet in the original paper [53] and [7]. Also, in [54], both MelGAN and its improved version, MB-MelGAN [58], are less performant than WaveRNN.
- Although Parallel WaveGAN outperforms WaveNet in the original paper [35], we found the WaveNet used in the experiments is smaller than in [57], [36], and [7]. In [57] and [54], both WaveNet and WaveRNN show better results than Parallel WaveGAN. Furthermore, in another work to improve Parallel WaveGAN by the same authors [59], WaveNet with noise-shaping [60] out-

performs both the original and the improved Parallel WaveGAN models.

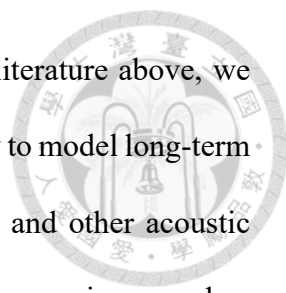
- HiFi-GAN is a recently proposed vocoder and outperforms WaveNet in the original paper [7]. Yet works that include HiFi-GAN and other autoregressive methods as baseline comparisons are currently limited.

- **Diffusion-Based Model**

- In DiffWave [36], WaveNet outperforms all proposed BASE models and other non-autoregressive baselines. The proposed LARGE model is only slightly better than WaveNet in the subjective evaluation.
- In WaveGrad [54], WaveRNN outperforms all proposed BASE models and other non-autoregressive baselines. The proposed LARGE model is only slightly better than WaveRNN in the subjective evaluation.

From the evaluation results of the literature, the comparative autoregressive baselines, mostly WaveNet or WaveRNN, usually perform better than non-autoregressive models. Furthermore, most autoregressive models in these works are their original versions. They are not implemented with improving techniques proposed later [40, 41, 45, 60]. The observations above indicate that non-autoregressive vocoders still struggle to consistently outperform autoregressive ones regarding speech quality, whether evaluated in the original papers [36, 53–56, 61] or as comparative models in other works [7, 36, 54, 56, 57, 59, 61].

A similar tendency has been observed in natural language processing (NLP), where studies have indicated the effectiveness of autoregressive models in capturing sequential dependencies and contextual information [62–65]. Moreover, a comprehensive survey [66] on multiple tasks, including neural machine translation (NMT), automatic speech recognition (ASR), and TTS, demonstrates that the difficulty of non-autoregressive gen-



eration correlates with the target token dependency. In light of the literature above, we attribute the superior quality of autoregressive vocoders to their ability to model long-term dependencies and capture time-dependent nuances in pitch, energy, and other acoustic features. Conditioning future audio samples on previous ones, autoregressive vocoders capture temporal coherence, producing more natural speech with fine-grained acoustic details. Consequently, despite the high efficiency of non-autoregressive models, autoregressive vocoders gain an advantage in speech quality.

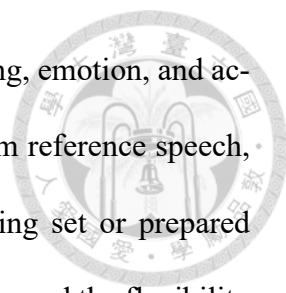
Stability

Besides efficiency and quality, autoregressive and non-autoregressive methods also differ in stability. While autoregressive models generally excel in speech quality, they can sometimes experience error propagation, which hampers their stability. Error propagation [67] refers to the phenomenon where errors in earlier predictions can propagate and amplify throughout the generation process, potentially affecting the overall stability of the output. On the other hand, non-autoregressive methods, with the ability of parallel synthesis, are not affected by this issue and offer improved stability. Though autoregressive models may encounter error propagation in some scenarios, it is worth noting that their overall quality is still better, as mentioned previously.

2.2 Text-Guided Speech Generation

2.2.1 Overview

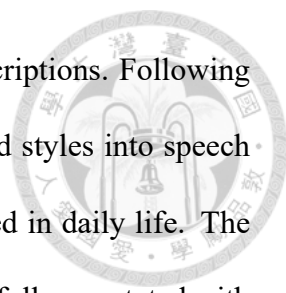
Recent years have witnessed significant advancements in speech generation technology, evolving from robotic articulations to systems capable of producing natural speech with



various styles, including specified speaker identity, manner of speaking, emotion, and accent. However, these styles are typically predefined or extracted from reference speech, limiting the model's expressiveness to style categories in the training set or prepared speech prompts. Consequently, the diversity of generated speech styles and the flexibility for users to specify desired styles are constrained. On the contrary, in image and music generation, generative models that condition output styles on text descriptions have flourished, demonstrating remarkable achievements and allowing users to specify a broader range of styles more freely. This discrepancy has led to a paradigm shift towards text-guided speech generation, where natural language provides a more intuitive and flexible interface for defining and customizing voice styles. As a result, text-guided speech generation is emerging as a prominent research direction and is increasingly applied to tasks such as text-to-speech synthesis and voice conversion.

2.2.2 Text-Guided Text-to-Speech Synthesis

A typical expressive multi-speaker TTS model produces speech with content and style determined by the input text sequence and additional specified conditions, respectively. These conditions contain features of the target style, such as speaker identity and emotion. The style can correspond to a class seen during the training stage and specified by the class ID at inference time. Alternatively, it can represent a new, unseen style derived from a reference utterance. Recent works explore the extraction of conditions from human-written descriptions. [68] proposes to take short written-style tags as input and model the relationship between linguistic embeddings and speech styles. Similarly, PromptTTS [8] introduces two datasets (real and synthesized versions, denoted as PromptSpeech-R and PromptSpeech-S, respectively) to develop a TTS model capable of adjusting speech char-



acteristics such as speed, pitch, volume, and emotion through text descriptions. Following these works, InstructTTS [9] endeavors to encapsulate more nuanced styles into speech from more subjective and abstract descriptions akin to sentences used in daily life. The authors create a speech dataset with varied and expressive styles, carefully annotated with detailed text prompts in free-form natural language. The resulting InstructTTS model allows users to unrestrictedly describe diverse speech attributes, including speed, emotion, and conceptual scenarios. Besides focusing on the form of text descriptions, some advanced research uses specialized speaker prompts only for describing speaker characteristics [69, 70], leaving the rest of the speech variation handled by other model components. Table 2.1 lists text prompts used to describe styles in different works, demonstrating different aspects of speech characteristics to control. Complementary research directions include applying multi-modal inputs to specify styles [71], a unified architecture that understands linguistic and acoustic features [72], and leveraging large language models to reduce the reliance on human-written style descriptions [72–75].

2.2.3 Text-Guided Voice Conversion

The goal of speech conversion is to transform the input speech into an utterance of a specific speaker without altering the original spoken content. Like in the TTS task, traditionally, the target speaker can be specified by a class ID or referring to a speech utterance. Recently, progress has been made in utilizing text descriptions to characterize target speakers and styles. In PromptVC [10], the authors propose using a diffusion model to learn the mapping from linguistic features to style embeddings extracted from given reference speech. In Kuan et al. (2023) [11], a combination of TTS systems and speech processing toolkits are employed to create a dataset with a large amount of paired data, which

Table 2.1: Example text descriptions in different works. Descriptions in other languages (Lang.) are translated into English.



Name	Lang.	Example
PromptSpeech-R [8]	en	Her sound height is really high, the volume is normal, but she speaks very slowly
PromptSpeech-S [8]	en	A lady whispers to her friend slowly
Coco-Nut [76]	ja	A young man is speaking in a high-pitched voice, as if he is excited.
FSNR0 [68]	ko	with affection
NLSpeech [9]	zh	There was a sense of joy in the words, an expression of joy in the heart, mixed with pride.
PromptTTS++ [69]	en	(style) A woman speaks slowly with low volume and low pitch. (speaker) The speaker identity is described as soft, adult-like, gender-neutral and slightly muffled.
PromptSpeaker [70]	zh	a husky voice from a middle-aged man

includes text descriptions of target styles and the corresponding pre- and post-conversion speech utterances. The authors adopt an architecture similar to AudioBox [72], building a unified model to concurrently learn to interpret the style specified by linguistic features and convert acoustic features accordingly. By integrating textual guidance, the advancements of TTS and VC tasks highlight the shift towards more flexible, controllable, and customizable speech generation technologies.

Table 2.2: Details of text-guided speech generation models, including task, language (Lang.), attributes specified in text descriptions (Attr.). The asterisks (*) indicate that the training datasets or codes for building datasets are open-source. -: not reported.

Name	Task	Lang.	Attr.
Kim et al. (2021) [68]	TTS	ko	emotion, intention, voice tone, speed
PromptTTS* [8]	TTS	en	gender, pitch, speed, volume, emotion
InstructTTS [9]	TTS	zh	overall perceived emotion, emotion level, style
PromptStyle [77]	TTS	zh	speaking style
PromptTTS 2* [73]	TTS	en	gender, pitch, speed, volume
PromptTTS++ [69]	TTS	en	speaker characteristics, pitch, speed, volume
PromptSpeaker [70]	TTS	zh	speaker characteristics
Zhang et al. (2023) [74]	TTS	zh	speaking scenario, emotion, loudness, pitch, speed
MM-TTS [71]	TTS	en	gender, emotion, emotion level
Audiobox [72]	TTS	multilingual	age, gender, audio quality, pitch, speed, accent, emotion, environment
Lyth et al. (2024) [75]	TTS	en	gender, accent, speed, pitch, audio quality
PromptVC [10]	VC	zh	-
Kuan et al. (2023) [11]	VC	en	gender, pitch, speed, volume, emotion, SoX effects [78]

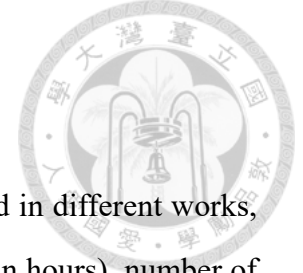


Table 2.3: Details of text-guided speech generation datasets reported in different works, including language (Lang.), data source (Src.), total duration (Dur., in hours), number of utterances (Utr.), and number of speakers (Spk.). -: not reported.

Name	Lang.	Src.	Dur.	Utr.	Spk.
<i>Publicly Available</i>					
PromptSpeech-R [8]	en	LibriTTS [79]	-	27893	1191
PromptSpeech-S [8]	en	TTS	-	160124	4
Coco-Nut [76]	ja	YouTube	8	7600	-
<i>Proprietary</i>					
FSNR0 [68]	ko	professional voice actors	26	18700	8
NLSpeech [9]	zh	internal	44	32000	7
PromptStyle [77]	zh	internal	12	-	8
PromptTTS++ [69]	en	LibriTTS-R [80]	585	-	2456
PromptSpeaker [70]	zh	internal, AISHELL-3 [81], DiDiSpeech [82]	-	21760	792
Zhang et al. (2023) [74]	zh	artistic storytelling shows	60	30000	1
MM-TTS [71]	en	MEAD	36	31055	47
PromptVC [10]	zh	internal	-	50000	6
Kuan et al. (2023) [11]	en	internal, PromptSpeech [8]	700	405000	2460

2.2.4 Data Resources of Text-Guided Speech Generation

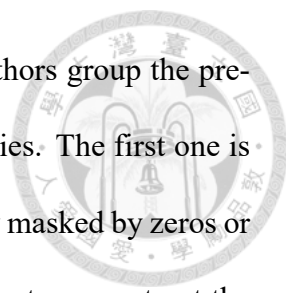


Table 2.2 shows details of text-guided speech generation models, highlighting a significant reliance on proprietary datasets, particularly for works in languages other than English and involving complicated styles [9, 10, 70, 74, 77]. This reliance often leads to the utilization of extensive, private collections of data, limiting the reproducibility and generalization of such models. Table 2.3 lists details of datasets used in text-guided speech generation works. These datasets are mainly closed-source, with only a few being publicly available. Moreover, the open-source datasets typically encompass less expressive speech from a TTS dataset or model [8], resulting in reduced diversity. Given these constraints, some works have sought to mitigate the need for human-annotated data by leveraging large language models for data retrieval or augmentation. Despite these efforts, how to efficiently utilize existing datasets to enhance the diversity of text descriptions and speech data, thereby constructing a more robust model, remains a significant challenge.

2.3 Self-Supervised Learning in Speech Processing

2.3.1 Overview

Self-supervised learning (SSL) in speech processing has evolved as a significant advancement, enabling models to learn from unlabeled data. An SSL model first undergoes pre-training on extensive corpora; then, it extracts robust representations useful across various speech tasks. This learning paradigm is facilitated through pretext tasks, specifically designed to assist the model in capturing high-level abstract information from the raw audio signals. During pre-training, the model is tasked with predicting some aspects of the input



utterance based on other parts of the same utterance. In [83], the authors group the pretext tasks for speech self-supervised learning into three main categories. The first one is generative approaches, in which the input speech features are partially masked by zeros or other features at randomly selected frames [84–86]. The model learns to reconstruct the missing features with the information from the unmasked parts. The second is contrastive approaches, which introduce contrastive learning methods to extract representations that can distinguish designed positive samples from negative ones [12, 87–89]. The last category is predictive approaches, in which the model aims to generate speech features of future frames given features from previous frames [90–93]. These pretext tasks encourage the model to understand underlying patterns and structures in speech data, such as phonetic nuances or speaker characteristics, without relying on explicit annotation.

After an SSL model is trained on a large dataset of unlabeled speech, it can be fine-tuned using labeled data for specific downstream tasks such as automatic speech recognition, speaker identification, or sentiment analysis [94]. This fine-tuning process adapts the general-purpose representations learned during the pre-training to the specifics of the targeted task. The representations can be the outputs of the SSL model or hidden outputs from the intermediate layers. Importantly, SSL has shown to be particularly effective in speech processing, outperforming traditional supervised learning methods, especially in scenarios where labeled data is scarce or in multi-modal contexts. This has led to significant improvements over traditional supervised methods, which often require extensive labeled datasets that are costly and labor-intensive to create.

2.3.2 Data Resources of Self-Supervised Learning in Speech Processing



Despite significant advances in SSL for speech processing, these methods exhibit a pronounced dependence on extensive datasets during pre-training. While WavLM and HuBERT have achieved state-of-the-art (SOTA) performance with little labeled data [13, 94, 95], the best models are pre-trained using datasets comprising tens of thousands of hours of speech. This heavy reliance on vast amounts of unlabeled data poses substantial challenges for applying these SOTA methods to specific knowledge domains or low-resource languages with limited available speech data. Additionally, using large-scale speech datasets raises concerns regarding copyright and privacy issues, as they often contain proprietary or personally identifiable information. Furthermore, despite SSL achieving notable successes across computer vision, natural language processing, and speech processing domains, research on reducing the dependence on extensive unlabeled datasets remains limited. This scarcity of solutions exacerbates the issue, maintaining it as a significant challenge within the field. Consequently, developing methods to enhance the data efficiency of SSL with smaller, more domain-specific datasets without compromising performance remains a pivotal area of research.



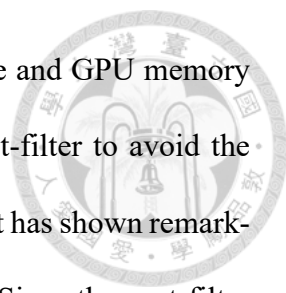
Chapter 3

Computational Efficiency in Speech Generation: Better Architecture Design for Non-Autoregressive Neural Vocoder

3.1 Introduction

In the previous chapters, we identified limitations in the computational efficiency of speech generation systems. First is the slow inference speed of autoregressive neural vocoders. Subsequently, although some non-autoregressive alternatives, such as WaveGlow [6] and Parallel WaveGAN [35], are proposed to improve inference efficiency, these methods have deeper model architectures [6] or more complicated training frameworks [7, 35], which instead result in higher demands for training resources.

To address the latter limitation, in this chapter, we aim to design an efficient, high-quality, and small-footprint waveform generation model. Starting with a non-autoregressive WaveGlow vocoder, we first compress the model by applying the weight-sharing tech-



nique across different layers, significantly minimizing the model size and GPU memory required during training. Then, we adopt a WaveNet-based [4] post-filter to avoid the compression harming the speech quality. The architecture of WaveNet has shown remarkable performance in various speech processing tasks [6, 35, 37, 38]. Since the post-filter only needs to amend the output of the compressed WaveGlow, a small network is competent, keeping the overall model fast and lightweight. With the streamlined architecture and specially designed losses in the frequency domain, the proposed model, referred to as WG-WaveNet, possesses improved convergence speed while maintaining high-quality output. Besides, WG-WaveNet requires much less computational cost at both training and inference time. The contributions of this study are summarized as follow:

- We propose a hybrid neural vocoder model, which is composed of a highly compressed WaveGlow model and a WaveNet-based post-filter. The proposed model, WG-WaveNet, is efficient and economical during training. WG-WaveNet has only 2.5 M parameters, which is 2.8% of those in the original WaveGlow. It can be trained on an NVIDIA 1080Ti GPU (using less than 8 GB GPU memory) in 4 days, while 8 NVIDIA GV100 GPUs were used in the original WaveGlow paper [6].
- The proposed methods significantly improve the inference efficiency. In particular, the inference speed of the proposed WG-WaveNet is higher than 960 kHz using an NVIDIA 1080Ti GPU and 1.5 times faster than real-time even without any GPU acceleration.
- For speech quality, perceptual experiments show that the proposed model can generate speech with similar quality compared with WaveNet, WaveGlow, Squeeze-Wave [96], and Parallel WaveGAN [35].

- We also study the quality of 44 kHz audio waveforms (high-fidelity audio) generated by neural vocoders. We explore the performances of ground truth recordings with various sampling rates and the effects of different parameters of short-time Fourier transform for training vocoders. The proposed method not only makes it possible to synthesize 44 kHz audio samples on a single CPU 1.2 times faster than real-time but also achieves a score of 4.01 in the MOS test, which even betters 16 kHz recordings.

3.2 Related Work

3.2.1 WaveGlow: A Flow-Based Neural Vocoder

WaveGlow [6] is a flow-based neural vocoder. It learns a bijective mapping f between real-world speech data and a predefined distribution, a zero mean spherical Gaussian here. This mapping can be considered as a series of bijective transformations, $f = f_k \circ f_{k-1} \circ \dots \circ f_1$, where k is the total number of transformations. The mapping process can be formulated as

$$z = f_k \circ f_{k-1} \circ \dots \circ f_1(x), \quad (3.1)$$

where $z \sim \mathcal{N}(0, \mathbf{I})$, and x is a speech waveform from the dataset. Following that, since each transformation f_i is invertible, we have

$$x = f_1^{-1} \circ f_2^{-1} \circ \dots \circ f_k^{-1}(z). \quad (3.2)$$

For each speech waveform x from the training set, a WaveGlow model with parameters θ is trained to minimize the negative log-likelihood of x . The log-likelihood can be

calculated by applying the change of variables technique to Eq. 3.2:

$$\log p_{\theta}(x) = \log p_{\theta}(z) + \sum_{i=1}^k \log |\det(J(f_i(x)))|, \quad (3.3)$$



Where J is the Jacobian. The first term in Eq. 3.3 can be easily derived from the Gaussian distribution, $\mathcal{N}(z; 0, \mathbf{I})$. To make the second term, the sum of the log-determinant of the Jacobian of each transformation, easy to calculate, WaveGlow adopts multiple techniques to build invertible and tractable transformations.

Figure 3.1 (a) shows the architecture of WaveGlow. The input speech signals are first reshaped into groups of 8 audio samples, followed by 12 transformations, each consisting of an invertible 1x1 convolution [52] and an affine coupling layer [97].

1x1 Invertible Convolution

An 1x1 invertible convolution f_{conv} is a simple transformation to mix information across different channels, formulated as

$$f_{conv}(x) = Wx, \quad (3.4)$$

where x is the input of each transformation, and W is the weight matrix, which is initialized to be orthonormal and invertible. The log-determinant of the Jacobian of this transformation can be simply calculated from W :

$$\log |\det(J(f_{conv}(x)))| = \log |\det W|. \quad (3.5)$$

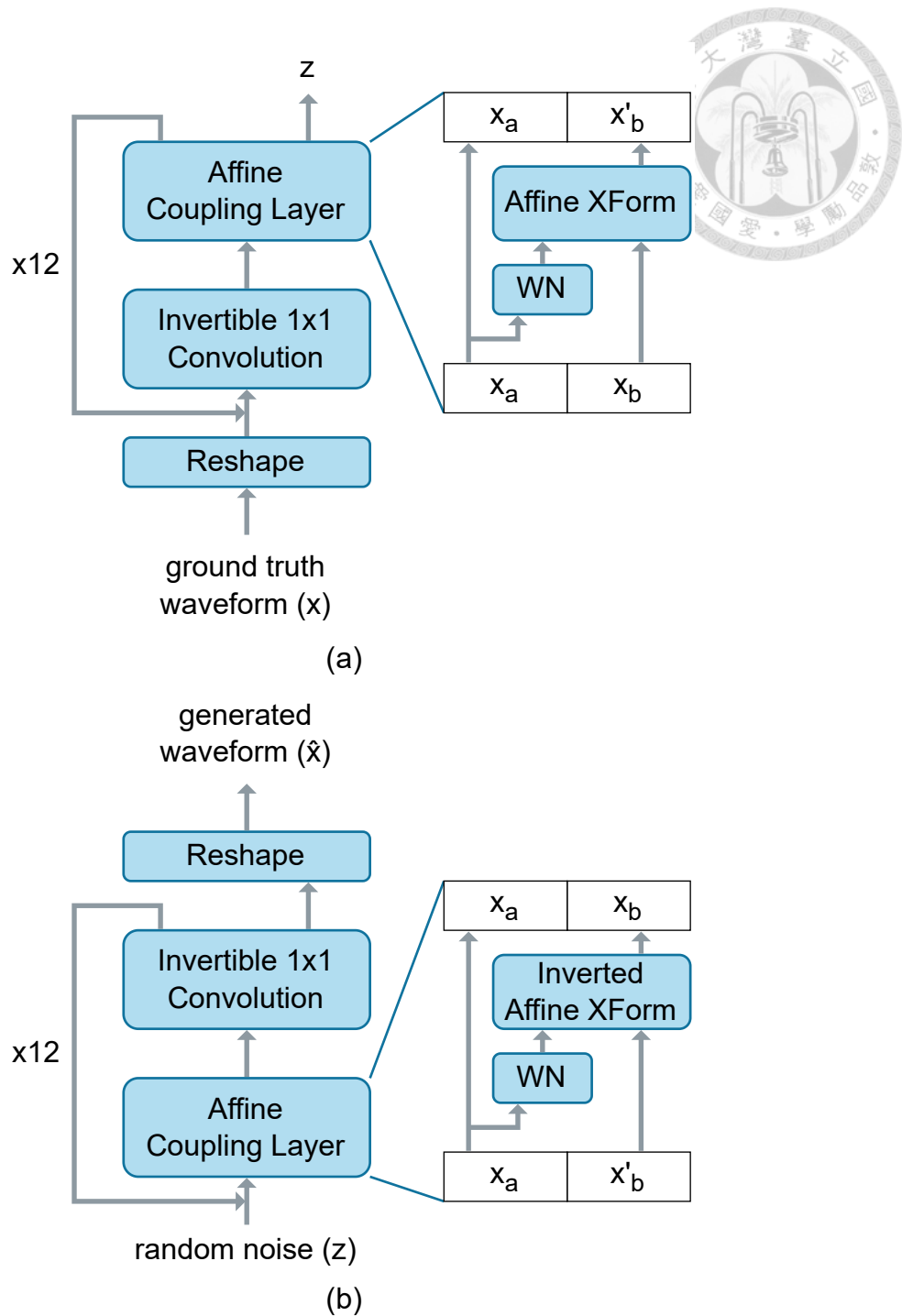


Figure 3.1: (a) WaveGlow network. (b) WaveGlow network (inverted). The WN module also takes Mel-spectrograms as input, which are omitted in this figure.

Affine Coupling Layer

In an affine coupling layer f_{acl} , the input x is first split along the channels into two parts:

$$x_a, x_b = split(x). \quad (3.6)$$

x_a serves as the partial output and the input of another module WN , calculating parameters to transform x_b as the remaining output:

$$\log s, t = WN(x_a, mel), \quad (3.7)$$

$$x'_b = s \odot x_b + t, \quad (3.8)$$

$$f_{acl}(x) = \text{concat}(x_a, x'_b), \quad (3.9)$$

where mel is a Mel-spectrogram used to provide acoustic information of speech. The process of f_{acl} is designed to be invertible, regardless of the architecture of WN . WaveGlow applies WaveNet [4] as the backbone of WN , which consists of convolutional layers and gated activation units [34]. When the model is inverted to generate speech, as formulated in Eq. 3.2, the affine transformation (Eq. 3.8) is also inverted as

$$x_b = \frac{x'_b - t}{s}. \quad (3.10)$$

Figure 3.1 (b) shows the complete inverted process.

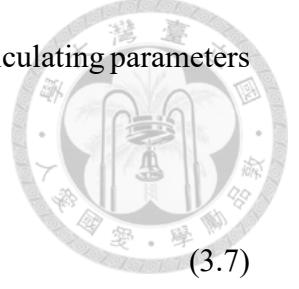
Due to the simplicity of affine transformation, the log-determinant of the Jacobian of f_{acl} can be written as

$$\log |\det(J(f_{acl}(x)))| = \log |s|. \quad (3.11)$$

After deriving the log-determinant of the Jacobian of each transformation, we can rewrite Eq. 3.3 as follows:

$$\log p_\theta(x) = -\frac{z^\top z}{2\sigma^2} + \sum_{i=1}^k (\log s_i + \log |\det W_i|), \quad (3.12)$$

where $z = f(x)$ is the model output during WaveGlow training, σ is the standard deviation



of the predefined Gaussian distribution. The model is then trained to optimize this log-likelihood.



3.3 Method

The proposed WG-WaveNet is composed of two components, shown in Figure 3.2 (a). The first part is a highly compressed WaveGlow model, which will be introduced in Section 3.3.1. In Section 3.3.2, to further improve the sound quality, we employ a WaveNet-based post-filter trained with loss functions on the frequency domains.

3.3.1 Highly Compressed WaveGlow

A WaveGlow model consists of several transformations to progressively map speech data to the Gaussian space. A transformation is composed of an affine coupling layer [97] and an invertible 1x1 convolution layer [52]. Each affine coupling layer in WaveGlow adopts a deep WaveNet-like module. Consequently, the overall model is huge and hard to train.

We apply cross-layer parameter sharing to reduce parameters and make the model more compact. The cross-layer parameter sharing has shown to be helpful in NLP task pre-training [98] and source separation [99]. As shown in Figure 3.2 (a), transformations in the compressed WaveGlow share the same affine coupling layer¹. This approach keeps the model from drastically growing in size when it gets deeper. Considering these transformations are processes of gradually mapping data from one distribution to another, invertible 1x1 convolution layers remain different across transformations to keep variability.

¹To make the affine coupling layer shareable here, we remove the early-output mechanism used in the original WaveGlow to keep the output shape the same across layers.

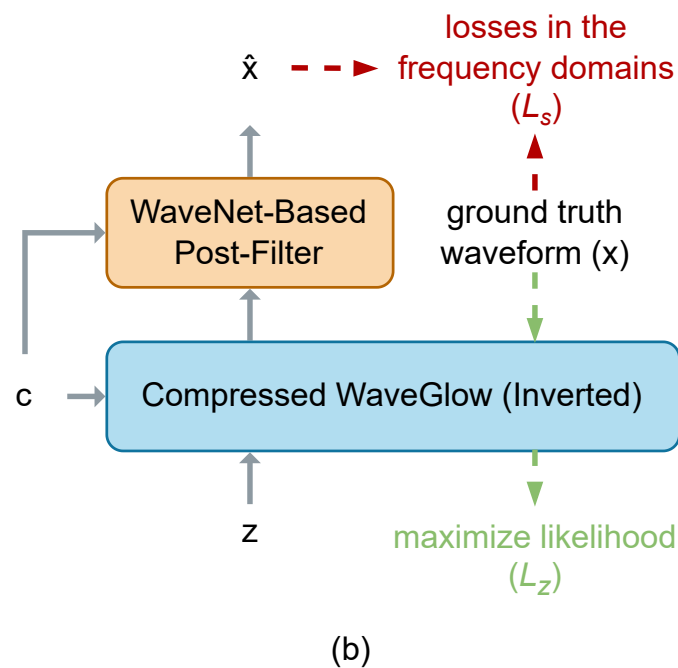
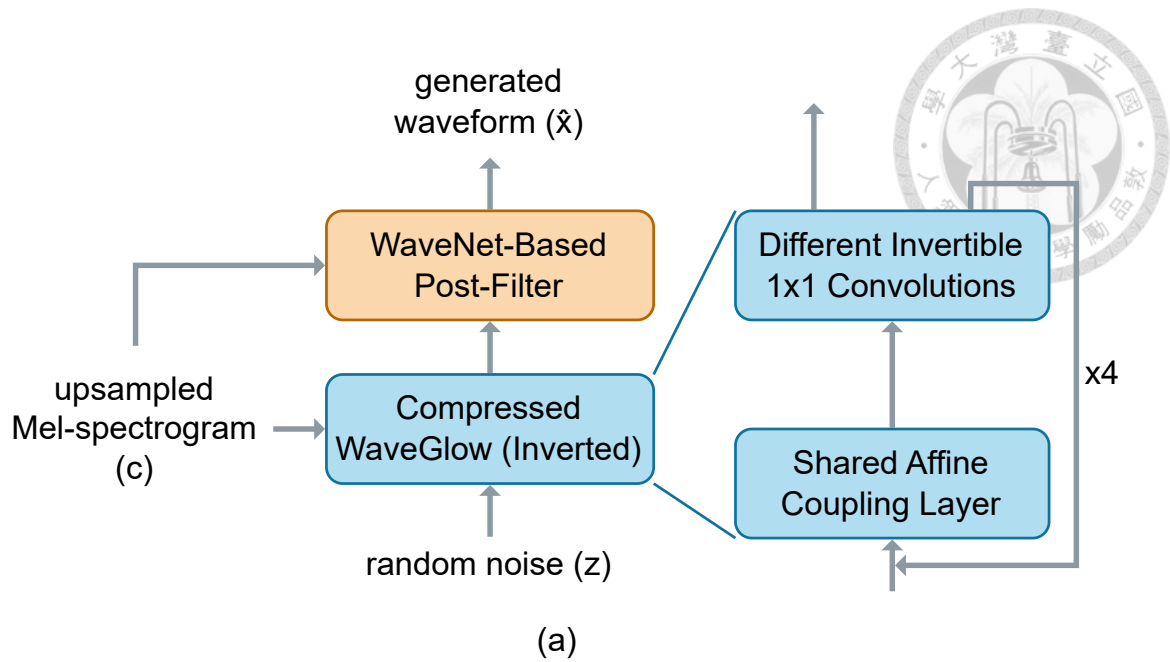


Figure 3.2: (a) Architecture of WG-WaveNet. (b) Training of WG-WaveNet.

We found that this improved the quality of generated speech in preliminary experiments. We also change the upsampling method from deconvolution to layers of duplication and convolution to further reduce parameters.

The training process is the same as mentioned in [6], shown by the green path in

Figure 3.2 (b). The loss function, denoted as L_z , is the negative log-likelihood of the training data. The proposed compression approach reduces the number of parameters in the WaveGlow and considerably cuts down the requirements of GPU memory. In the following section, we propose to use a post-filter to further speed up the convergence and improve the performance of the compressed WaveGlow.

3.3.2 WaveNet-Based Post-Filter

A random noise z is sampled from the Gaussian distribution as the input of the inverted compressed WaveGlow. The output of WaveGlow is then used as the input of the WaveNet-based post-filter to generate \hat{x} in parallel [100, 101] conditioned on an upsampled Mel-spectrogram. The WaveNet-based post-filter is trained by minimizing the loss function $L_s(x, \hat{x})$, in which x is the ground truth samples, while \hat{x} is the output of the post-filter. The WaveNet-based post-filter and the inverted compressed WaveGlow are jointly learned to minimize $L_s(x, \hat{x})$ ². Since the WaveNet here synthesizes audio samples based on the output of the inverted compressed WaveGlow, its parameters can also be highly reduced.

For L_s , we utilize loss functions on the different frequency domains. Spectral losses have been shown effective for training waveform generation models in [102], [103], and [35]. We modify the multi-resolution short-time Fourier transform (STFT) auxiliary loss in [35] as follows:

$$L_s(x, \hat{x}) = \frac{1}{M} \sum_{i=1}^M (L_{sc}^i(x, \hat{x}) + L_{mag}^i(x, \hat{x}) + L_{mel}^i(x, \hat{x})), \quad (3.13)$$

where M is the number of different parameter sets of STFT; L_{sc} and L_{mag} are the spectral

²Since the WaveNet-based post-filter is irreversible, it can not be trained jointly by maximizing the likelihood as WaveGlow.

convergence loss and the log STFT-magnitude loss from [104]:

$$L_{sc}(x, \hat{x}) = \frac{\| |STFT(x)| - |STFT(\hat{x})| \|_F}{\| |STFT(x)| \|_F}, \quad (3.14)$$

$$L_{mag}(x, \hat{x}) = \frac{1}{N_{mag}} \|\log |STFT(x)| - \log |STFT(\hat{x})|\|_1, \quad (3.15)$$

where $\|\cdot\|_F$ is the Frobenius norm, $\|\cdot\|_1$ is the L_1 norm, $|STFT(\cdot)|$ is the STFT magnitude, and N_{mag} is the number of elements in the magnitude. To make L_s more representative of human perception, we add a Mel-scale STFT-magnitude loss:

$$L_{mel}(x, \hat{x}) = \frac{1}{N_{mel}} \|\log |MEL(x)| - \log |MEL(\hat{x})|\|_1, \quad (3.16)$$

where $|MEL(\cdot)|$ and N_{mel} denote the Mel-scaled STFT magnitude and the number of elements in the magnitude, respectively. The number of Mel bands differs in different STFT parameter sets.

The WaveNet-based post-filter is trained jointly with the inverted compressed WaveGlow, as shown by the red path in Figure 3.2 (b). The loss function for training WG-WaveNet is a linear combination of L_z and L_s :

$$L_{total} = \lambda L_z + L_s, \quad (3.17)$$

where λ is a scalar to balance the loss terms. In practice, L_s is calculated every n iterations.

Eventually, the overall WG-WaveNet (compressed WaveGlow plus WaveNet post-filter) is 2.8% of the original WaveGlow in model size. Model details will be discussed in Section 3.4.2.





3.4 Experimental Setup

3.4.1 Datasets

Two datasets were used in the experiments. One was the LJ Speech Dataset [105]. This English dataset consists of 13100 clean audio clips (about 24 hours) of a female speaker. The sampling rate is 22050. The other was an internal Mandarin corpus, which contains 9004 utterances (about 6.8 hours) from a female speaker. The recordings were sampled at 44 kHz. 100 utterances were selected from each dataset for evaluation.

We used the 80-band Mel-spectrogram as the condition to synthesize audio. For WG-WaveNet, the FFT size, hop size, and window size for STFT are 2048, 200, and 800, respectively.

3.4.2 Model Details

The WaveNet-based post-filter in the proposed WG-WaveNet is composed of 7 layers of dilated convolution blocks with 64 channels. The original WaveGlow has 12 transformations. With the help of the post-filter, the compressed WaveGlow consists of only 4 transformations. The WaveNet-like module in the shared affine coupling layer has 7 layers with 128 channels. The WG-WaveNet model was trained for 1 M steps using the Adam optimizer [106] with a batch size of 8. The learning rate was $4e^{-4}$ and reduced by half every 200 K steps. We set $\lambda = 1$ and $n = 3$ based on preliminary experiments. The parameters for calculating L_s in Section 3.3.2 are listed in Table 3.1. We also built a faster version of WG-WaveNet, denoted as g-20. In WaveGlow and original WG-WaveNet, the input is reshaped to groups of 8 samples [6]. Inspired by [96], the input of g-20 is reshaped



Table 3.1: Parameters for calculating L_s (reported in samples).

FFT size	4096, 2048, 1024, 512, 256
hop size	400, 200, 100, 50, 25
window size	1600, 800, 400, 200, 100
# of Mel bands	640, 320, 160, 80, 40

Table 3.2: Comparison of model sizes.

Model	Size
WaveNet	24.7 M
WaveGlow	87.9 M
SqueezeWave	23.7 M
Parallel WaveGAN	1.3 M
WG-WaveNet (ours)	2.5 M
WG-WaveNet (g-20)	3.1 M

to groups of 20 samples.

We compared our method with four baseline models: WaveNet, WaveGlow, SqueezeWave, and Parallel WaveGAN. To ensure that the models were consistent compared to the original models, for the first three, we used pre-trained models from public implementations^{3 4 5}. Note that the pre-trained models of WaveGlow and SqueezeWave were released by the official. We followed the setup in [35] to train the Parallel WaveGAN.

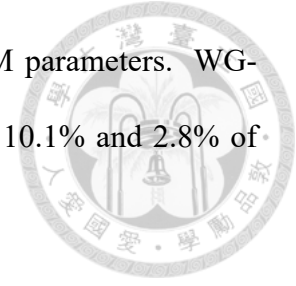
The numbers of parameters of different models are listed in Table 3.2. Both the WaveNet-based post-filter and the compressed WaveGlow have fewer layers and channels than the original WaveNet and WaveGlow, making WG-WaveNet much more com-

³https://github.com/r9y9/wavenet_vocoder

⁴<https://github.com/NVIDIA/waveglow>

⁵<https://github.com/tianrengao/SqueezeWave>

pact. WaveNet has 24.7 M parameters, and WaveGlow has 87.9 M parameters. WG-WaveNet, on the other hand, has only 2.5 M parameters, which is 10.1% and 2.8% of those in WaveNet and WaveGlow, respectively.

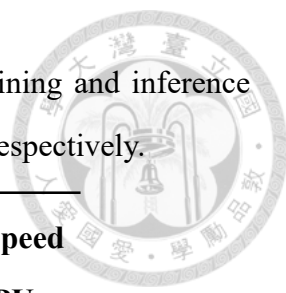


3.4.3 Evaluation Metrics

To evaluate the performance of the proposed model and the baseline models, we conducted subjective and objective evaluations to assess speech quality.

- **Subjective Evaluation:** We conducted Mean Opinion Score (MOS) tests to rate the quality of generated speech under human perception. In these MOS tests, raters were asked to score each speech utterance on a five-point scale based on naturalness. A higher score denotes higher quality and a closer resemblance to authentic human speech. We randomly selected 10 utterances from the evaluation set to generate, and each utterance was rated by at least 20 raters.
- **Objective Evaluation:** We calculated Mel-cepstral distortion (MCD, reported in dB) [107] as the objective metric to evaluate the distortion in the frequency domain. We first extracted 25-dimensional Mel-Frequency Cepstral Coefficients (MFCCs) from both ground truth and synthesized speech signals. Subsequently, the root mean square error (RMSE) was calculated on a frame-by-frame basis. The average RMSE per frame was then used to determine the overall MCD score. A lower MCD score implies greater similarity between the generated and ground truth speech, indicating higher quality in the frequency domain. This evaluation was conducted using 10 utterances randomly selected from the evaluation set.

Table 3.3: Comparison of computational cost and speed during training and inference time. The units of memory, time, and speed are GB, days, and kHz, respectively.



Model	Training	Inference Speed
	Memory / Time	CPU / GPU
WaveNet	-	0.1 / 0.12
WaveGlow	-	10 / 279
SqueezeWave	-	330 / 4486
Parallel WaveGAN	14.4 / 2.7	18 / 841
WG-WaveNet (ours)	7.7 / 3.5	33 / 967
WG-WaveNet (g-20)	5.2 / 2.5	53 / 1634

3.5 Results

3.5.1 Speed and Computational Cost

We evaluated the speed and memory usage of different models during training and inference. Parallel WaveGAN and WG-WaveNet were trained on the same server using an Nvidia V100 16GB RAM GPU to fairly evaluate the computational cost at the training stage. The testing environment was a personal computer with an Intel i7-6700K CPU and an Nvidia 1080Ti GPU. Since the computational cost of parallel synthesis methods might be affected by the output length at the inference stage, we tested the models using utterances with various lengths uniformly distributed from 2 to 9 seconds.

The results are shown in Table 3.3. Though the training time of WG-WaveNet is slightly longer than that of Parallel WaveGAN, the training memory is 47% less. The inference speed of WG-WaveNet is at a rate of 967 kHz with GPU and 1.5 times faster than real-time without GPU. Moreover, the faster WG-WaveNet variant, g-20, can be

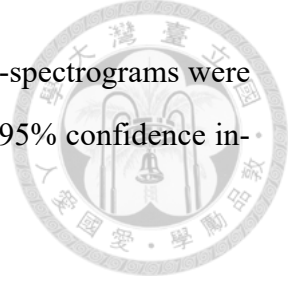


Table 3.4: MOS and MCD results compared with other models. Mel-spectrograms were extracted from the ground truth. The MOS results are reported with 95% confidence intervals.

Model	MOS	MCD
WaveNet	4.49±0.101	4.619
WaveGlow	3.71±0.159	4.393
SqueezeWave	2.96±0.121	3.608
Parallel WaveGAN	4.24±0.108	4.026
WG-WaveNet (ours)		
$\lambda = 1, n = 3$	4.08±0.118	3.783
$\lambda = 1, n = 1$	3.23±0.159	2.948
$\lambda = 0, n = 1$	3.65±0.164	2.407
g-20	3.75±0.124	3.848
Ground Truth	4.61±0.096	-

optimized with much fewer computational resources and generate 22 kHz speech 2.4 times faster than real-time without GPU.

3.5.2 Audio Quality Comparison

The subjective and objective evaluation results are shown in Table 3.4. To assess the effects of L_z and L_s on model performance, we trained WG-WaveNet with different λ and n . Figure 3.3 shows the trade-off between audio quality and inference speed.

The observations based on Table 3.4 and Figure 3.3 are concluded as follows: (1) WaveNet has the highest MOS, which is close to that of the ground truth data, yet there is a gap between the performance of parallel and autoregressive synthesis methods. (2) MCD

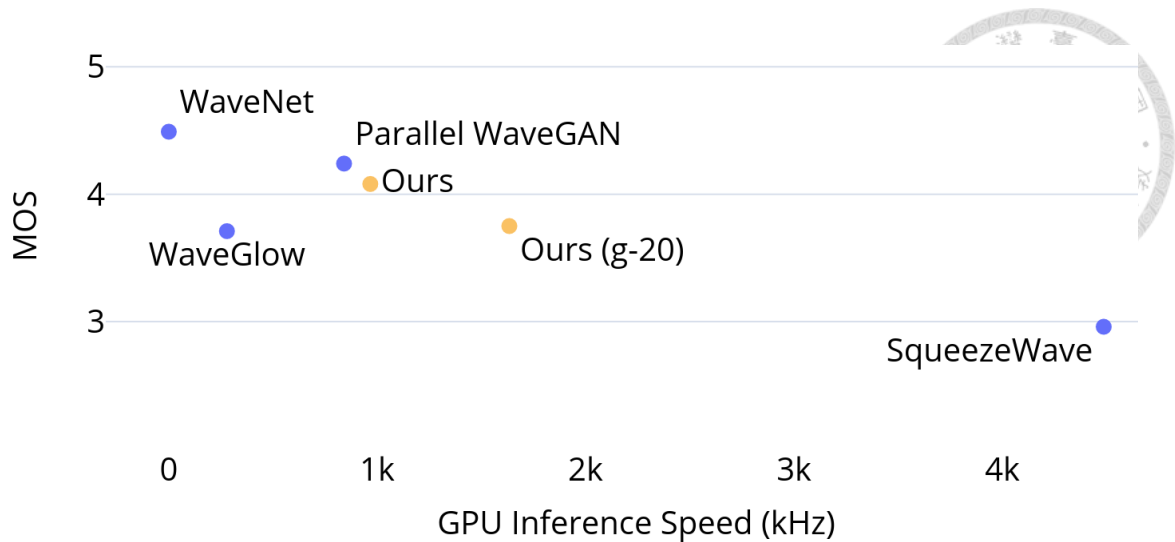


Figure 3.3: Trade-off between MOS and GPU inference speed.

is not strongly related to human perception. Training a model using L_s ($\lambda = 0, n = 1$) leads to the lowest MCD but not the best MOS, while WaveNet has the highest MOS and MCD. A similar contradictory result was also found in [40]. (3) Though we used the officially released models to synthesize utterances, WaveGlow and SqueezeWave did not perform well. Subjects reported there were noise and reverberation effects in the generated speech. (4) The ablation study shows that both L_z and L_s are crucial for training WG-WaveNet. We found that training using only L_s ($\lambda = 0, n = 1$) led to good quality at the voiced parts of speech but significant high-frequency glitches at the unvoiced parts. (5) The MOS decreases rapidly when the generating efficiency improves. WG-WaveNet, however, has a faster speed and an MOS of 4.08, which is close to that of Parallel WaveGAN. This indicates that the proposed WG-WaveNet can greatly increase the synthesis speed while preserving a comparable performance.

3.5.3 High-Fidelity Audio Generation

Due to the fast inference speed and high quality of WG-WaveNet as shown in Sections 3.5.1 and 3.5.2, we show that WG-WaveNet can generate high-fidelity audio (44 kHz) in this



Table 3.5: MOS results of high-fidelity audio generation with 95% confidence intervals. Mel-spectrograms were extracted from the ground truth sampled at 44 kHz.

Model	MOS
Parallel WaveGAN	
w1600	3.12±0.134
w800	3.04±0.126
WG-WaveNet (ours)	
w1600	3.15±0.148
w800	3.71±0.131
w800 (g-20)	4.01±0.110
Ground Truth (16 kHz)	
Ground Truth (16 kHz)	3.72±0.147
Ground Truth (22 kHz)	4.15±0.127
Ground Truth (44 kHz)	4.44±0.105

subsection. To evaluate the performance, we trained WG-WaveNet and Parallel WaveGAN on the 44 kHz speech dataset mentioned in Section 3.4.1. We only compared WG-WaveNet with Parallel WaveGAN here because only Parallel WaveGAN and SqueezeWave are efficient enough to synthesize 44 kHz audio, and the audio quality of SqueezeWave is not comparable with Parallel WaveGAN. MOS tests with the same setups as those in Section 3.4.3 were conducted on the generated waveform and ground truth data with different sampling rates.

The results are shown in Table 3.5. "w800" denotes that the window size for extracting Mel-spectrograms is set to 800. The FFT size, hop size, and the number of Mel bands are also the same as mentioned in 3.4.1. "w1600" denotes that the window size is doubled to 1600, and the other parameters are also doubled. Since the sampling rate is changed from 22050 to 44100, doubling STFT parameters (w1600) makes the temporal

Table 3.6: MOS results and GPU inference speed (in kHz) compared with other models. Mel-spectrograms were generated by the Tacotron 2 model. The MOS results are reported with 95% confidence intervals.

Model	MOS	Inference Speed
Tacotron 2+GL	2.11±0.139	-
Tacotron 2+WaveNet	3.96±0.116	0.12
Tacotron 2+Parallel WaveGAN	3.72±0.127	841
Tacotron 2+WG-WaveNet (ours)	3.68±0.133	967
Ground Truth	4.36±0.108	-

resolution of extracted features the same as in Section 3.5.2, while the temporal resolution is doubled in "w800". Similarly, parameters for calculating L_s in "w800" are the same as in Table 3.1, while they are doubled in "w1600". We first found that the sampling rates of the ground truth samples significantly affect their perceptual scores. The raters considered the ground truths with higher sampling rates to be better. Experiments reveal that when the temporal resolution of acoustic features is fixed (w1600), it is harder to generate 44 kHz speech than to generate 22 kHz one. We observed that Mel-spectrograms with higher temporal resolution (w800) helped improve the performance of WG-WaveNet (w800). WG-WaveNet outperformed Parallel WaveGAN in both "w800" and "w1600" cases. Eventually, the faster WG-WaveNet reached 4.01 MOS, which is even better than that of 16 kHz ground truth speech.

3.5.4 Text-to-Speech

We combined WG-WaveNet with a Tacotron 2 model to evaluate the proposed method as a vocoder. The Tacotron 2 was built following [1]. Data preprocessing for training the

TTS model and vocoders were set to the same as mentioned in Section 3.4.1.

The results of MOS tests and GPU inference speed of vocoders are reported in Table 3.6. Note that the ground truth inherently has better prosody and quality than those of the speech generated by Tacotron 2. We found that the performance gap between WaveNet and the parallel synthesis methods narrowed. WG-WaveNet has an MOS comparable to that of Parallel WaveGAN, and the inference speed is faster than other methods, which shows the advantage of WG-WaveNet as a vocoder for fast, high-quality speech synthesis.

3.6 Summary

This chapter aims to address the computational efficiency challenges of non-autoregressive vocoders. We introduce WG-WaveNet, a flow-based vocoder with a compressed WaveGlow model and a WaveNet-based post-filter. By applying the weight-sharing technique to compress WaveGlow, we significantly reduce the model size and GPU memory required during training. The proposed post-filter further speeds up the convergence, leading to less training time. Besides training efficiency, we show that WG-WaveNet achieves a high inference speed without sacrificing speech quality. Finally, we investigate the quality of high-fidelity audio and show that the proposed WG-WaveNet can real-time generate high-quality 44 kHz audio samples without GPU acceleration.



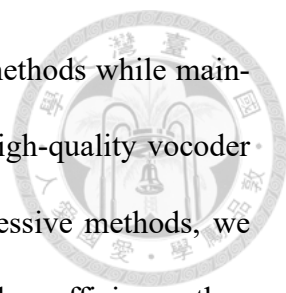


Chapter 4

Computational Efficiency in Speech Generation: Applying Non-Autoregressive Mechanism for Efficient Autoregressive Generation

4.1 Introduction

In the previous chapters, we discussed the evolution of neural vocoders, the emergence of non-autoregressive models designed to improve the speed of speech generation, and the associated challenges in training efficiency. Chapter 3 specifically studied reducing the computational efficiency required for training a flow-based non-autoregressive model. The literature review in Chapter 2 and the experimental results from Chapter 3 underscored the superior quality provided by autoregressive models. Building on these observations, Chapter 4 shifts the focus back to autoregressive models, investigates their characteristics, and rebuilds the autoregressive algorithm to achieve a more efficient generation process.



This chapter aims to enhance the efficiency of autoregressive methods while maintaining their superior speech quality, thereby establishing a rapid, high-quality vocoder for speech generation. To better explain the properties of autoregressive methods, we first explore and hypothesize the reason for the high quality and the low efficiency, then introduce the proposed methods motivated by the hypothesis.

Autoregressive methods have been successfully applied in various fields and achieved high-quality results. These methods predict targets word by word [108–110], frame by frame [1, 16], or pixel by pixel [34, 111–113]. Similarly, in waveform generation, conventional autoregressive vocoders generate speech signals sample by sample [4, 5]. All these methods predict only a small part of the whole target in each generation, conditioned on the previous predictions. We hence hypothesize that, **in a generative task, dividing the target into multiple smaller parts and predicting one part at a time conditioned on the predicted parts reduce the complexity and difficulty of modeling real data.** The predicted parts can provide more detailed information for the subsequent generation process, resulting in better next predictions. For speech synthesis, by leveraging the conditioning mechanism, autoregressive vocoders can better capture and model time dependencies in a waveform, leading to superior speech quality. The hypothesis above also explains the inefficiency of autoregressive methods in speech synthesis, primarily attributed to dividing the target in the time domain. This division significantly increases the computational cost at inference, forming a predicting process that iteratively generates tens of thousands of samples. Figure 4.1 (a) illustrates the predicting process of the conventional autoregressive methods. To address the inefficiency problem while maintaining speech quality, we turn to explore alternative domains to divide a speech waveform for iterative generation.

Motivated by the success of subband analysis in speech synthesis [44–48, 58], we

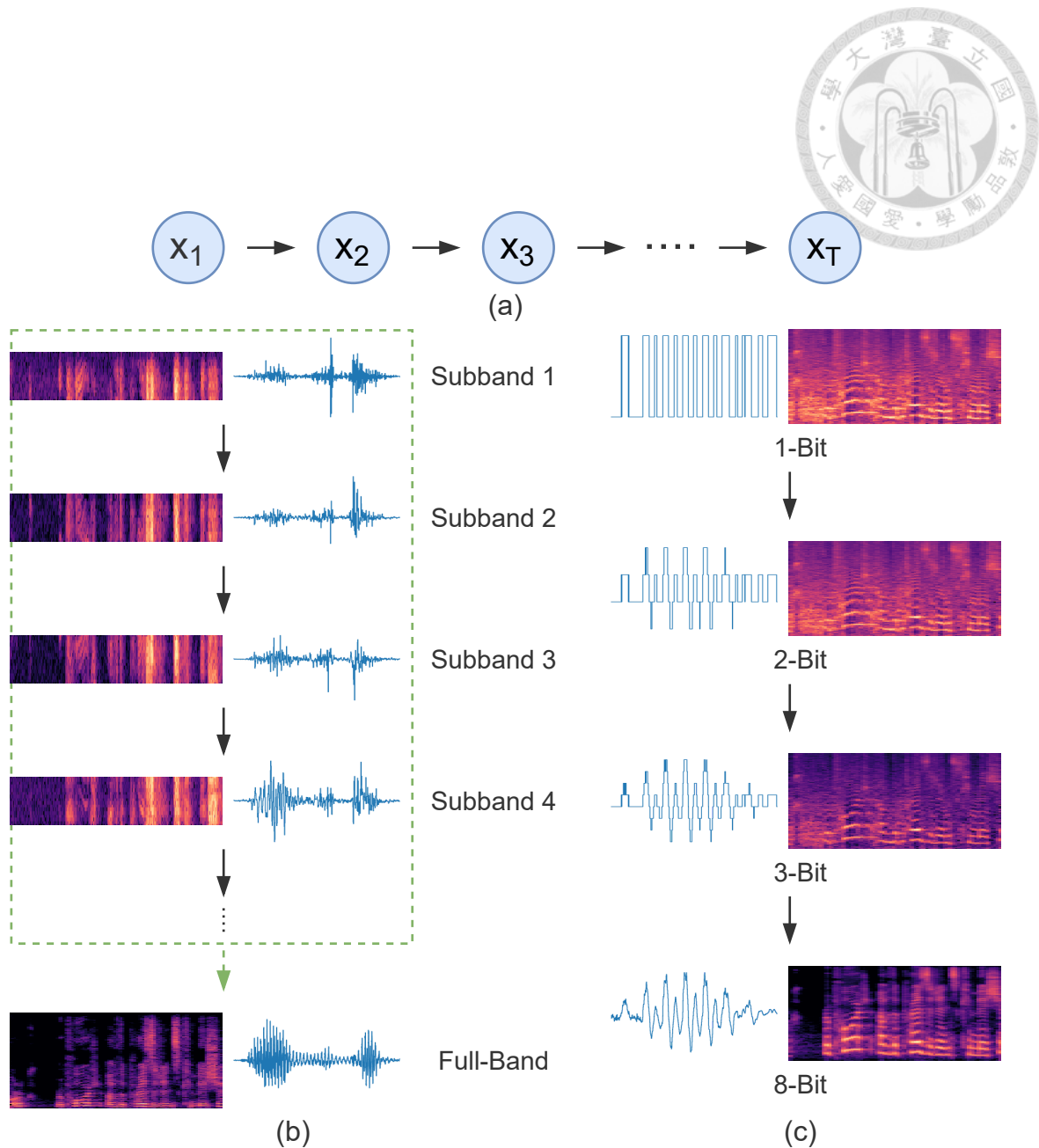
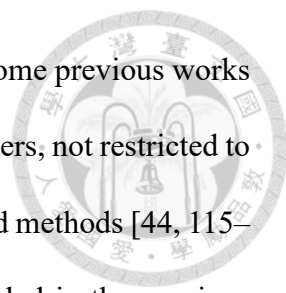
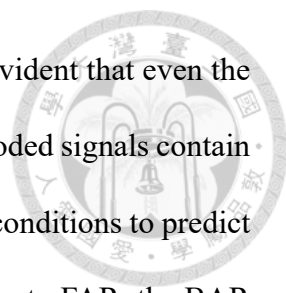


Figure 4.1: Overview of different autoregressive methods and their orders of generation. (a) Conventional autoregressive generation. Samples at different time steps are generated sequentially. (b) Frequency-wise autoregressive generation (FAR). Subbands are first generated autoregressively and combined to form the full-band waveform. (c) Bit-wise autoregressive generation (BAR). The spectrograms in (b) and Mel-spectrograms in (c) are only for visualization and not for generation.



first shift the focus from the time domain to the frequency domain. Some previous works have shown correlations between speech subbands [114, 115], and others, not restricted to speech processing, leveraged the subband correlation in their proposed methods [44, 115–117]. Combined with our hypothesis, the subband correlation revealed in the previous works inspires a thought of dividing and sequentially predicting signals in the frequency domain. Following the idea, we propose frequency-wise autoregressive generation (FAR). In FAR, a full-band speech utterance is divided into multiple frequency subbands by analysis filters. Autoregressively in the frequency domain, the model learns to predict a subband given the previous one. As shown in Figure 4.1 (b), each subband contains partial information of the full-band waveform, and successive subbands are correlated, sharing similar time-dependent acoustic information provided in conventional autoregressive vocoders, such as energy, f_0 , formants, or voicing status. A subband signal can thus provide helpful information for the next subband prediction. The FAR model generates the whole subband utterance in a single computation. As a result, the time needed to derive the full-band waveform is not dependent on the speech length. Instead, it is proportional to the number of subbands, which typically is considerably less than the speech length.

We further probe other possible domains for autoregressive generation. Inspired by the idea of dividing signals into different precision in [5], we investigate autoregressive generation in the bit precision domain and propose bit-wise autoregressive generation (BAR). In BAR, each speech sample is quantized into an 8-bit representation using μ -law companding transformation. The value of an 8-bit sample ranges from 0 to 255. The first bit, which represents whether the value of the speech sample is greater than 127, is predicted first. Then the second and the third bits are sequentially predicted, conditioned on the previously predicted bits. Finally, the first three bits are used to generate a com-



plete 8-bit sample. Figure 4.1 (c) shows the overview of BAR. It is evident that even the 1-bit waveform shows clear f_0 and formant contours. Since low-bit-coded signals contain partial acoustic information of the 8-bit waveform, they can serve as conditions to predict signals in other bit precision, implying the feasibility of BAR. Similar to FAR, the BAR model also generates signals in the same bit precision in parallel, and the time for inference is only proportional to the number of precision we divided.

The novel FAR and BAR greatly enhance efficiency by minimizing the number of required iterations, yet they retain the iterative generative process to ensure the high quality of generated speech. Furthermore, FAR and BAR can be combined and applied to the same model. We show in experiments the effectiveness of combining FAR and BAR and the importance of the generation orders.

The contributions of this chapter are twofold. First, this work proposes to explore and design new directions for autoregressive methods to improve efficiency. Instead of in the time domain, the proposed model conducts autoregressive speech generation in the frequency and bit precision domain. Second, a post-filter is applied for sampling from output posteriors and restores high-fidelity 16-bit samples instead of 8-bit ones. We combine the characteristics of the proposed autoregressive methods to design the training objective for the post-filter. To validate our hypothesis and assess the effectiveness of the proposed FAR, BAR, and post-filtering methods, we conducted comprehensive experiments comparing them with existing autoregressive and non-autoregressive vocoders. Objective and subjective experimental results consistently demonstrate that the proposed method, augmented with a grouping mechanism, achieves real-time synthesis and matches the quality of state-of-the-art neural vocoders. Furthermore, in addition to neural vocoders, the proposed methods have the potential to be applied to autoregressive models in various speech

synthesis tasks and improve efficiency.



4.2 Method

4.2.1 Rethinking the Direction for Autoregressive Generation

The conventional autoregressive neural vocoder, as shown in Figure 4.2 (a), formulates the synthesis problem as maximizing the joint probability of a waveform $\mathbf{x} = \{x_1, x_2, \dots, x_T\}$, which can be factorized as follows:

$$p(\mathbf{x}) = \prod_{t=1}^T p(x_t | x_1, x_2, \dots, x_{t-1}), \quad (4.1)$$

where x_t is the audio sample at time t and generated conditioned on signals from previous time steps. Teacher forcing can be easily applied for parallel computing during training. While at inference time, the signal x_t at time t is not able to be predicted until all x_1, x_2, \dots, x_{t-1} are inferred. The total iterations and time are inevitably proportional to the target audio length. Twenty-two thousand calculations are conducted iteratively to generate a one-second speech with a 22 kHz sampling rate.

To increase inference efficiency and preserve the quality of the audio output, we redesign the autoregressive method to compute in domains other than the temporal one. A waveform sequence \mathbf{x} is first split into N subsequences, x^1, x^2, \dots, x^N , where N is some fixed number, and each x^n is a time series. As shown in Figure 4.2 (b), the prediction of the n th subsequence x^n is conditioned on the previous subsequence x^{n-1} . Eq. 4.1 can be reformulated as follows:

$$p(\mathbf{x}) = \prod_{n=1}^N p(x^n | x^{n-1}). \quad (4.2)$$

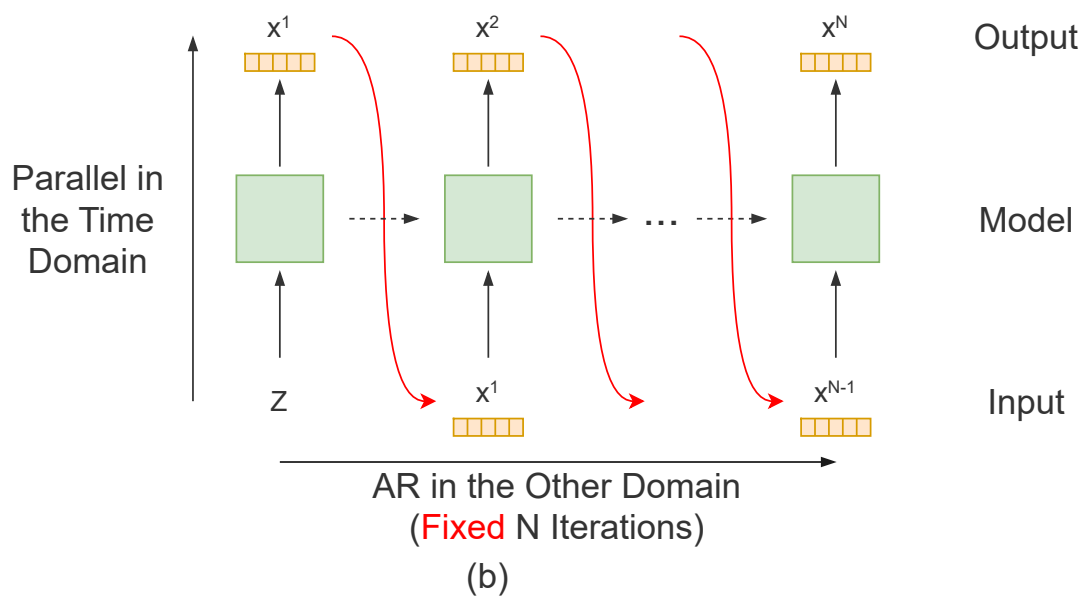
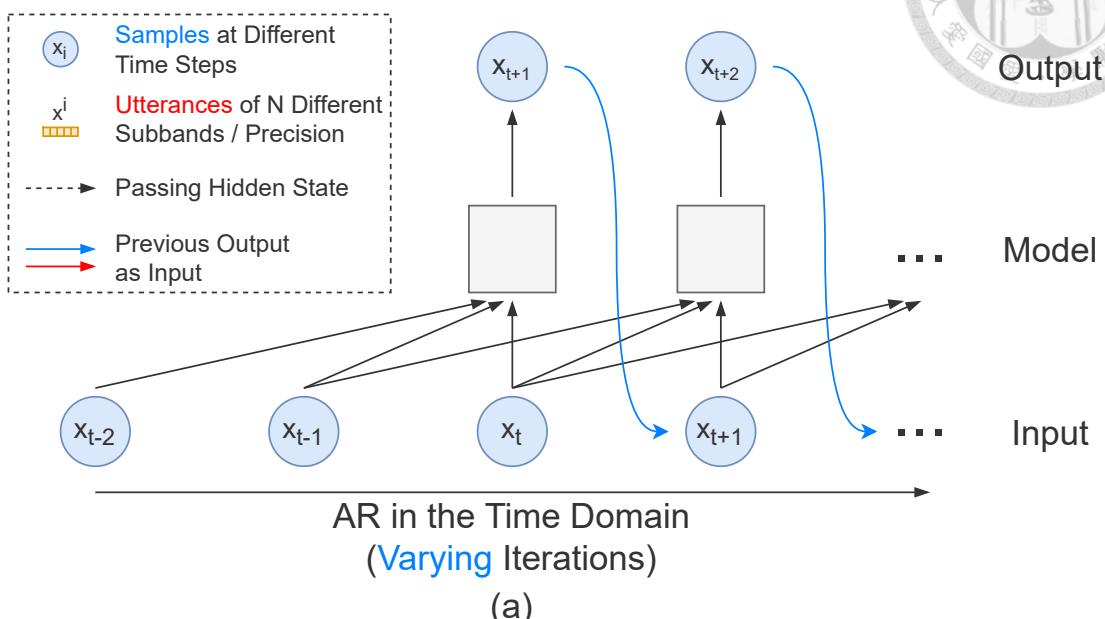


Figure 4.2: (a) Conventional autoregressive model. Each blue circle represents a scalar. (b) Proposed autoregressive model. The speech is generated iteratively in the frequency domain or the bit precision domain. Each green block is the model illustrated in Figure 4.5 (a), and each orange block represents a time series.

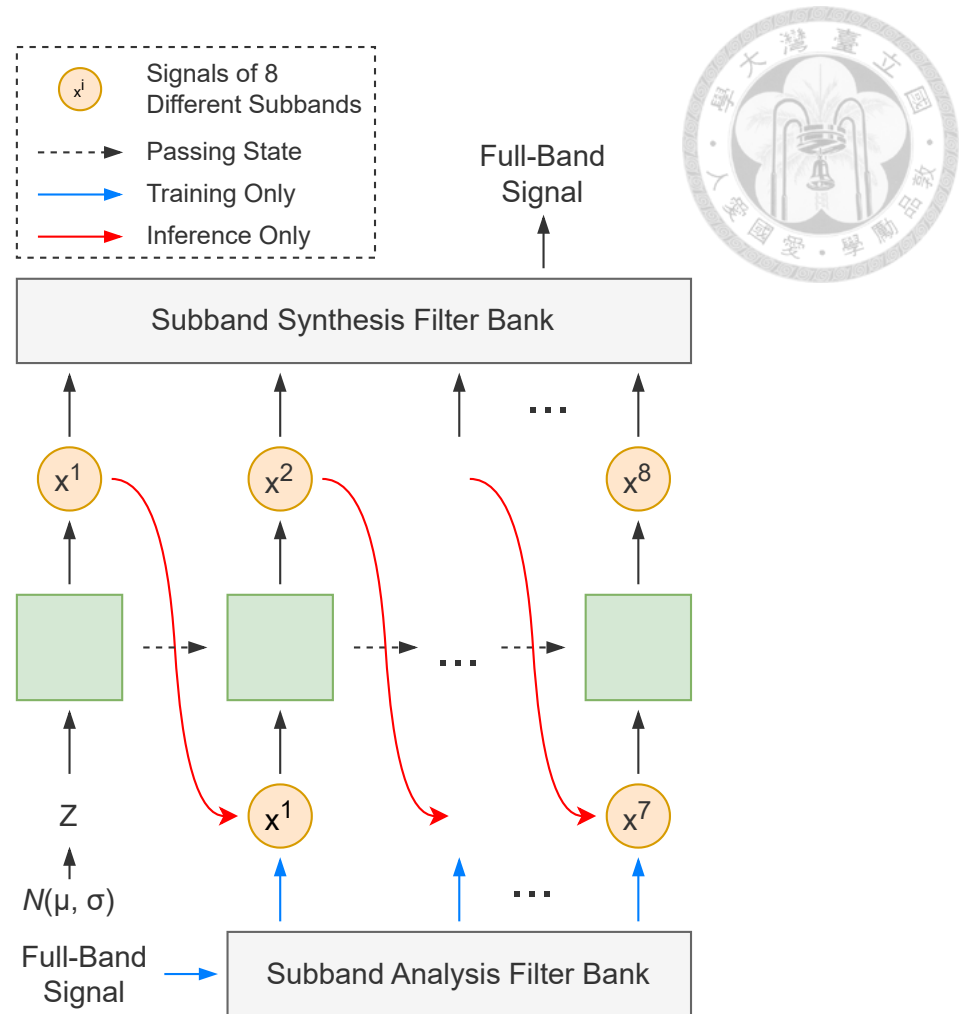


Figure 4.3: Frequency-wise autoregressive generation.

The redesigned process takes fixed N iterations in total and can be parallel in the time domain by designing the model architecture, e.g., using only layers of CNN. We will discuss the splitting methods in Section 4.2.2 and 4.2.3. Note that the conditioning acoustic features are omitted and will be detailed in Section 4.2.4.

4.2.2 Frequency-wise Autoregressive Generation (FAR)

The first splitting method is subband analysis, which divides a speech utterance into multiple subband signals using a subband analysis filter bank, and each represents information

in different frequency bands. The analysis process can be written as follows:

$$\{x^1, x^2, \dots, x^N\} = \phi(\mathbf{x}), \quad (4.3)$$



where ϕ is the analysis filter bank, N is the number of subbands, and x^i is the i th subband utterance. While splitting a signal with length L into N subband signals, each length is shortened to L/N . Though some works leverage this method to improve the efficiency of autoregressive vocoders [44, 45, 47], they only make use of the property of length shortening to enable models to synthesize N shorter subband utterances in parallel. The inference time still grows substantially as the signal length increases.

In FAR, we replace the time domain with the frequency domain for autoregressive generation, making the model compute parallelly in the time domain and instead perform autoregressive synthesis in the frequency domain. As shown in Figure 4.3, the model first takes as input a noise Z from the normal distribution $N(\mu, \sigma)$ and generates the signal of the first subband, x^1 . In each forward operation, x^i serves as a condition to generate the next subband signal, x^{i+1} . Finally, a full-band L -length speech \mathbf{x} can be generated with N different subband signals and a synthesis filter bank ψ :

$$\mathbf{x} = \psi(x^1, x^2, \dots, x^N). \quad (4.4)$$

We follow [118] to design analysis and synthesis filter banks. In our proposed model, a speech utterance is divided into eight subband signals, and the model iteratively generates from the subband with the highest frequency to the one with the lowest. The total number of iterations is consequently fixed to eight, which is much less than that in traditional autoregressive methods and makes the model more efficient.

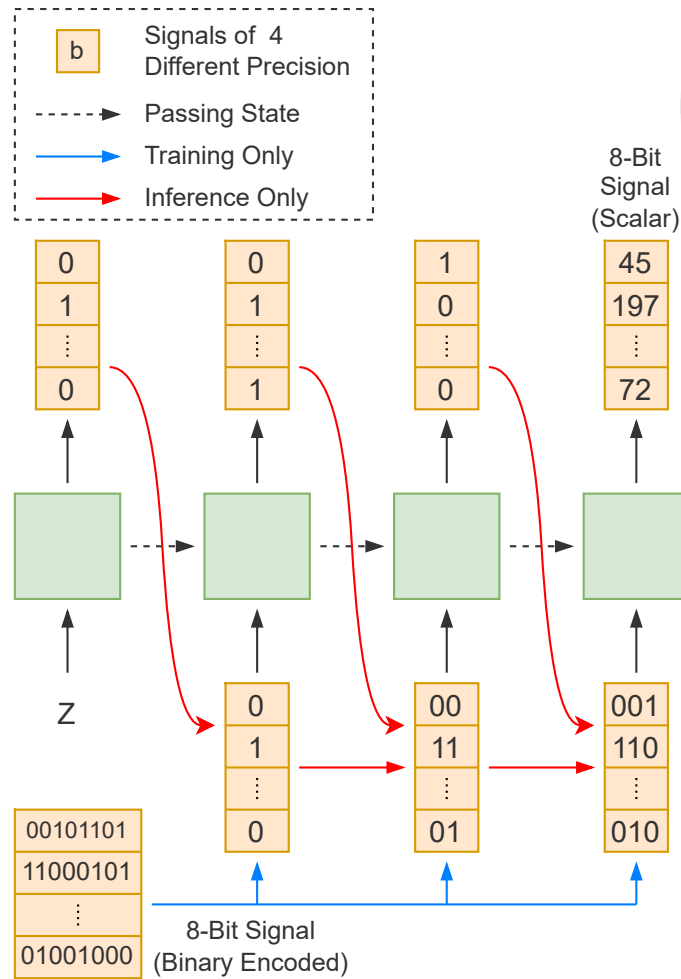


Figure 4.4: Bit-wise autoregressive generation.

4.2.3 Bit-wise Autoregressive Generation (BAR)

FAR enables models to generate speech samples with a fixed number of iterations while conditioned on the previous and future information. To provide more information for autoregressive speech generation, in this section, we further investigated another domain to split the signals. The proposed BAR is an autoregressive method in the bit precision domain.

In BAR, we follow [4] to model the output as 8-bit samples transformed by the μ -law algorithm. In a transformed 8-bit signal, samples are integers ranging from 0 to 255, denoted as 8-Bit Signal (Scalar) in Figure 4.4. Samples can also be represented as binary

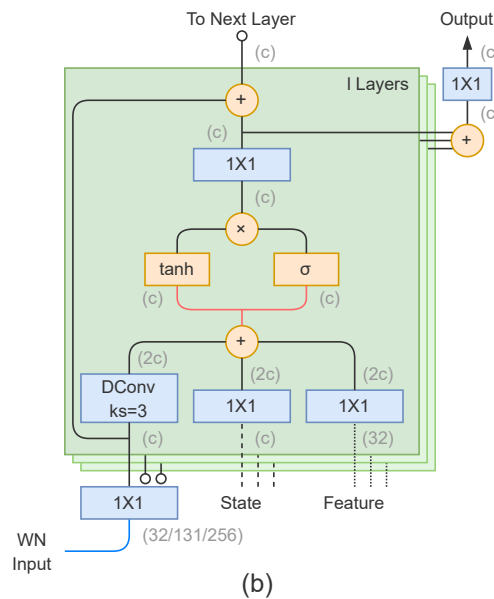
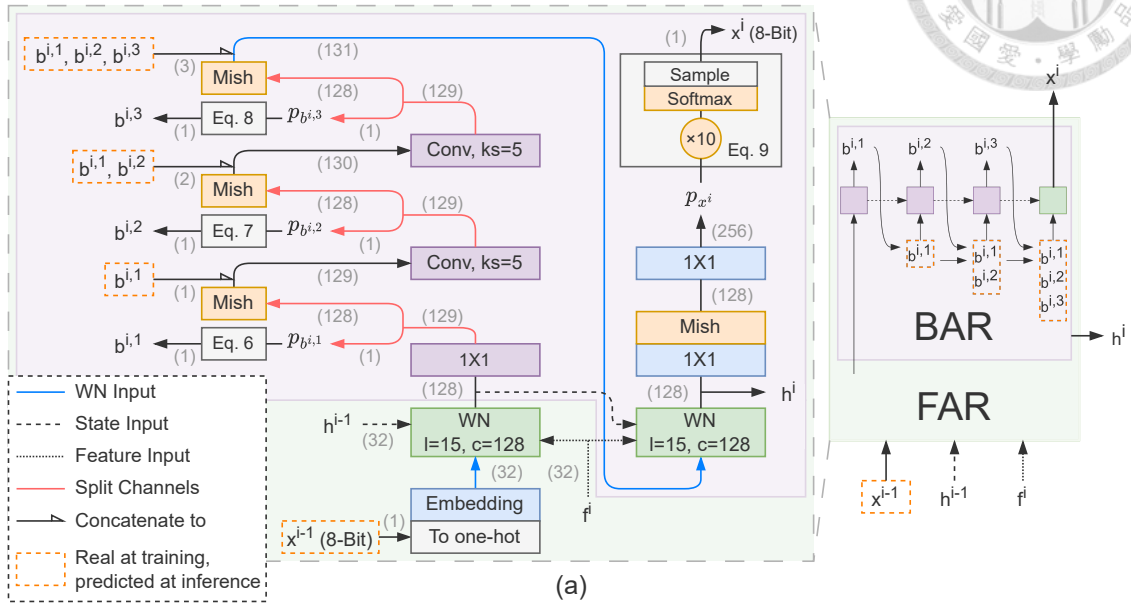
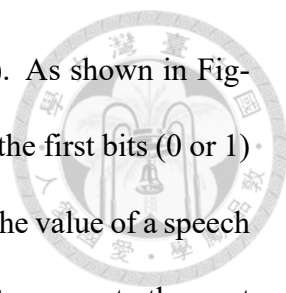


Figure 4.5: (a) Overview and detailed block diagram of the proposed model. The model predicts the i th subband x^i conditioned on the previous subband x^{i-1} . h^i is the hidden state, and f^i is the upsampled acoustic feature. BAR is integrated in each FAR prediction, and $b^{i,1}, b^{i,2}, b^{i,3}$ are the first three bits of x^i . Channel sizes are shown in gray and parenthesized. (b) WN module.

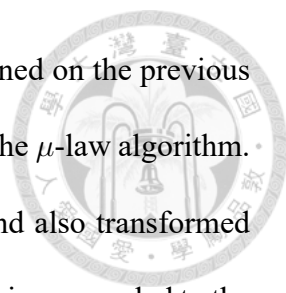


sequences of length eight, denoted as 8-Bit Signal (Binary Encoded). As shown in Figure 4.4, the first step is to take an initial vector Z as input to generate the first bits (0 or 1) of the samples at different time steps. The first bit represents whether the value of a speech sample is greater than 127. All previous bits are used as conditions to generate the next bits. The output signal gradually becomes more precise as the bits are predicted. Although the model can iteratively generate each bit in the binary sequences, we empirically found that the 8-bit integers in a signal can be directly predicted by conditioning solely on the first three bits, thereby fixing the total number of iterations at four. In our implementation, we integrate BAR into each FAR iteration, and the initial vector Z is the hidden output in the model. Besides, we employ four separate layers for predicting the first three bits and the 8-bit integers. Further implementation details will be provided in the next section.

4.2.4 Proposed Vocoder Architecture

In the previous sections, we introduced a new concept for autoregressive generation and demonstrated FAR and BAR. The two proposed autoregressive methods can be combined and applied to the same model. We will show in Section 4.4.1 that the model with both FAR and BAR outperforms the models with only either method. This section details how to combine FAR and BAR as an autoregressive neural vocoder.

Figure 4.5 (a) shows an overview and a detailed block diagram of the model, consisting of two WaveNet-based modules (WN) and convolutional layers (Conv) with kernel sizes (ks) of 5 and 1 (1X1). Figure 4.5 (b) shows the WN module, which is mainly composed of dilated convolutional layers (DConv) and gated activation units [34]. The WN module was originally proposed in WaveNet [4] and has been widely adopted in many works [6, 35, 101, 119].



To conduct FAR, the model predicts the i th subband x^i conditioned on the previous subband x^{i-1} . Each x^i is a sequence of 8-bit integers transformed by the μ -law algorithm. The input for predicting x^1 is randomly sampled from $N(0, 0.25)$ and also transformed into 8-bit integers. The acoustic feature, a full-band Mel-spectrogram, is upsampled to the same length as the subband signals in the time domain by an upsampling network. The network consists of four convolutional layers, followed by Mish activation layers [120] and with kernel sizes of 2, 5, 5, and 1, respectively. Nearest neighbor upsampling is applied before the second and third convolutional layers. The upsampled feature is split along the channel into eight partitions. For the i th subband generation, the i th partition is used as the conditioning feature, denoted as f^i . Inspired by [5], we use h^i as a hidden state to pass information across different iterations. All subbands are predicted by the same model with the same weights. We denote the model as FAR and formulate the process as follows:

$$x^i, h^i = FAR(x^{i-1}, h^{i-1}, f^i). \quad (4.5)$$

BAR is integrated into each FAR iteration. In Figure 4.5 (a), the first output channel of each purple blocks is used to predict the first three bits of the i th subband sequence x^i , denoted as $b^{i,1}$, $b^{i,2}$, and $b^{i,3}$, respectively. The remaining channels are passed to Mish activation layers and are concatenated with predicted bits (ground-truth bits at training) as input of the next layer. Conditioned on the first three bits, the 8-bit integers in x^i are then predicted using a WN module followed by 1X1 and Mish activation layers. Note that instead of using the same WN module to predict the first three bits and the 8-bit integers, for better efficiency, we adopt three single layers to predict the first three bits, respectively. The idea of using different functions in autoregressive prediction has been introduced in [121, 122].

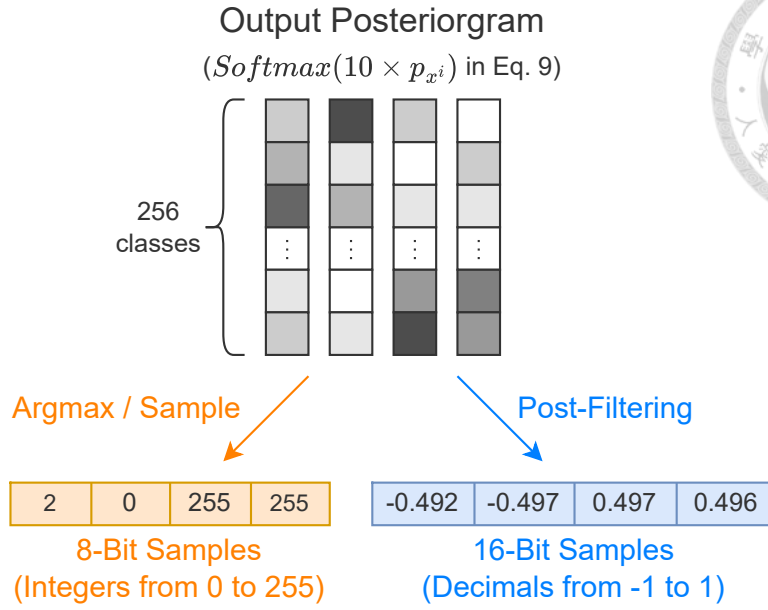


Figure 4.6: Different methods to sample output from posteriorgram. **Argmax**: Choosing the category with the greatest probability. **Sample**: Sampling according to the probability distribution. **Post-Filtering**: Using a network to predict 16-bit samples.

Denoting the output for predicting $b^{i,1}$, $b^{i,2}$, $b^{i,3}$, and x^i as $p_{b^{i,1}}$, $p_{b^{i,2}}$, $p_{b^{i,3}}$, and p_{x^i} , we formulate the processes of sampling from the posterior as follows:

$$b^{i,1} = \text{Sample}(\sigma(10 \times p_{b^{i,1}})), \quad (4.6)$$

$$b^{i,2} = \text{Sample}(\sigma(10 \times p_{b^{i,2}})), \quad (4.7)$$

$$b^{i,3} = \text{Sample}(\sigma(5 \times p_{b^{i,3}})), \quad (4.8)$$

$$x^i = \text{Sample}(\text{Softmax}(10 \times p_{x^i})), \quad (4.9)$$

where *Sample* is randomly sampling from probabilities and can be a post-filter network when predicting x^i , which will be detailed in the next section. Similar as in [45], each posterior is steepened, and the prediction becomes less noisy. We use the cross-entropy loss for training [4, 5, 41].



4.2.5 Post-filtering for Posterior Sampling

The output of a conventional autoregressive vocoder is usually a sequence of probability distributions [4, 5, 41], and waveform signals can be generated by two different methods: (1) Selecting the category with the most significant probability (Argmax). (2) Random sampling according to the probability distributions (Sample). The orange path in Figure 4.6 represents these two sampling methods. [40] has shown that Argmax and random sampling from the output distribution introduce noise and distortion. In [40] and [41], the authors leveraged voicing or pitch information and designed different rules for sampling from the distribution. While in this work, instead of carefully handcrafted sampling rules, we proposed using a neural network as a post-filter for sampling (the blue path in Figure 4.6). The proposed post-filter aims to solve two problems: (1) It reduces the effort of manually designing sampling rules by learning to restore high-quality speech from probability distributions. A well-trained post-filter can generate more accurate samples than sampling according to the distributions (2) It mitigates the noise and distortion introduced when transforming originally 16-bit samples into 8-bit representations. By directly predicting 16-bit samples, the network improves the detail and fidelity of the signal. Instead of posterior distributions of 2^{16} classes, the post-filter outputs the values of 16-bit samples, which are decimals ranging from -1 to 1. The process is formulated as:

$$x^i = PF(AR(x^{i-1})), \quad (4.10)$$

where PF is the post-filter, and x^i denotes the i th subband. AR is the proposed autoregressive model (h^{i-1} and f^i are omitted), and the output of AR is a posteriorgram, i.e., $Softmax(10 \times p_{x^i})$ in Eq. 4.9. All subbands are predicted by the same post-filter with

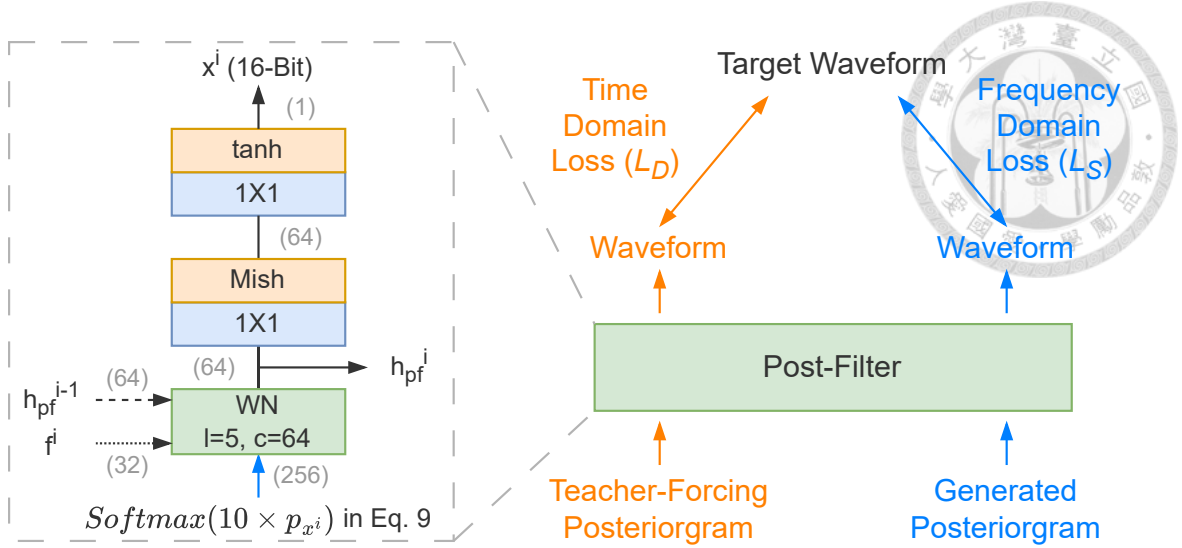


Figure 4.7: Post-filter and two paths for training. Channel sizes are shown in gray and parenthesized.

the same weights. The architecture of the post-filter is shown in Figure 4.7.

An autoregressive model is trained under teacher-forcing manners, which means x^i is generated conditioned on its previous ground truth subband \hat{x}^{i-1} . Two different types of posteriorgrams, $AR(\hat{x}^{i-1})$ and $AR(x_{ntf}^{i-1})$, can be used for training PF , formulated as follows:

$$x_{tf}^i = PF(AR(\hat{x}^{i-1})), \quad (4.11)$$

$$x_{ntf}^i = PF(AR(x_{ntf}^{i-1})), \quad (4.12)$$

where tf and ntf denote the sequence is generated with and without teacher forcing, respectively. Note that at inference time, PF works the same as Eq. 4.12.

We train the post-filter using two different losses in the time and frequency domains, respectively. Figure 4.7 shows the two training paths. In each step of the post-filter training, we perform both paths and apply different loss functions as the input type changes. When calculating the time domain loss, if the frequency domain condition (Mel-spectrogram) is the only ground truth information provided, generating a waveform close

to the ground truth in the time domain is challenging. Using a teacher-forcing posteriorgram as input (the orange path) makes it more feasible to optimize the time domain loss, as partial information excluded from a Mel-spectrogram, such as phase, is provided in the ground truth input \hat{x}^{i-1} . The model learns to utilize this information to generate a waveform close to the real signal in the time domain. The time domain loss L_D is written as follows:

$$L_D = \frac{1}{N+1} (MAE(\mathbf{x}_{tf}, \hat{\mathbf{x}}) + \sum_{i=1}^N MAE(x_{tf}^i, \hat{x}^i)), \quad (4.13)$$

where $\mathbf{x}_{tf} = \psi(x_{tf}^1, x_{tf}^2, \dots, x_{tf}^N)$, $\hat{\mathbf{x}} = \psi(\hat{x}^1, \hat{x}^2, \dots, \hat{x}^N)$, and MAE is the mean absolute error.

For calculating the frequency domain loss, since frequency information is provided in the input Mel-spectrogram, the post-filter can take a generated posteriorgram as input (the blue path) and minimize the distance between \mathbf{x}_{ntf} and $\hat{\mathbf{x}}$ in the frequency domain, where $\mathbf{x}_{ntf} = \psi(x_{ntf}^1, x_{ntf}^2, \dots, x_{ntf}^N)$. This kind of measurement has been shown to be effective [7, 35, 57, 102]. The frequency domain loss L_S is modified from the multi-resolution short-time Fourier transform (STFT) auxiliary loss [35] as follows:

$$L_S = \frac{1}{M} \sum_{m=1}^M (L_{sc}^m(\mathbf{x}_{ntf}, \hat{\mathbf{x}}) + L_{mag}^m(\mathbf{x}_{ntf}, \hat{\mathbf{x}})), \quad (4.14)$$

where M is the number of different parameter sets of STFT; L_{sc} and L_{mag} are the spectral convergence loss and log STFT-magnitude loss from [104]:

$$L_{sc}(a, b) = \frac{\| |STFT(a)| - |STFT(b)| \|_F}{\| |STFT(a)| \|_F}, \quad (4.15)$$

$$L_{mag}(a, b) = MAE(\log |STFT(a)|, \log |STFT(b)|), \quad (4.16)$$

where $\|\cdot\|_F$ is the Frobenius norm, and $|STFT(\cdot)|$ is the band-limited STFT magnitude.

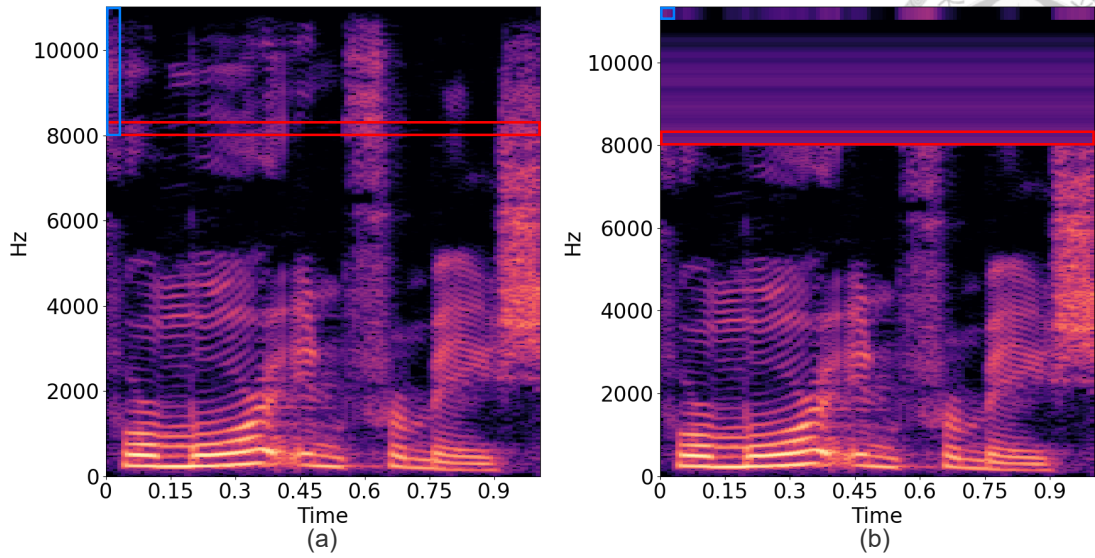


Figure 4.8: Average pooling in L_S . (a) STFT magnitude. (b) Band-limited STFT magnitude.

We use the magnitude information with the frequency ranging from 0 to 8000 Hz. Magnitudes with higher frequency are average pooled in the frequency and time axes to extract only local and global energy information. As shown in Figure 4.8 (a), the blue box represents calculating the average per frame; the red box represents calculating the average per frequency. The frame-level averages are concatenated to each frame, and the frequency-level averages replace all values on each frequency. Figure 4.8 (b) is the extracted magnitude. The blue box represents an average per frame (duplicated along the frequency axis for visualization); the red box represents an average per frequency, and the value is duplicated along the time axis for calculating L_S . The average pooling is to reduce the synthetic noise resulting from applying L_S , as mentioned in [57, 61, 123].

In each training step, the model generates both x_{tf}^i and x_{ntf}^i using the same mini-batch data. Then different loss functions, L_D and L_S , are applied for the two cases, respectively.

The final loss function L_{PF} is a linear combination of L_D and L_S :

$$L_{PF} = 100L_D + 0.1L_S. \quad (4.17)$$



We empirically decided the scalars in Eq. 4.17 in preliminary experiments. A large scalar for L_D and a small scalar for L_S made the training more stable and led to better performance. If the scalar for L_S is too large, the generated utterances will be with more synthetic noise mentioned in [57, 61, 123].

4.3 Experimental Setup

4.3.1 Dataset

Four datasets were used in our experiments.

- **LJ Speech** LJ Speech [105] is a high-quality speech dataset widely used for training and evaluating TTS models and neural vocoders. The audio clips are recorded with a sampling rate of 22 kHz by a female English speaker. It contains 13100 utterances, and the total duration is approximately 24 hours.
- **VCTK** CSTR's VCTK corpus [124] is a multi-speaker English speech dataset containing about 44k utterances and transcriptions. Approximately 400 sentences are read by 109 English speakers with different accents. The total duration is about 44 hours. The audio clips are with a sampling rate of 48 kHz and downsampled to 22 kHz in our experiments.
- **CMU ARCTIC** Initially designed for unit selection speech synthesis, the CMU

ARCTIC databases [125] consist of clean audio clips of different English speakers. We used utterances from a male (*bdl*) and a female (*slt*) speaker as a test set for experiments in Section 4.4.3. All the utterances were downsampled to 22 kHz.

- **Internal Mandarin Speech Corpus** The corpus is designed for building Mandarin TTS systems. It consists of 9004 high-quality speech utterances of a Mandarin female speaker. The total duration is about 7 hours. We used the audio clips with a sampling rate of 44 kHz for high-fidelity speech synthesis.

4.3.2 Acoustic Feature

We used an 80-dimensional Mel-spectrogram as the conditioning acoustic feature for speech synthesis. For computing the STFT, the FFT size, hop size and window size were set to 1024 (46 ms), 200 (9 ms), and 800 (36 ms), respectively.

4.3.3 Model Details

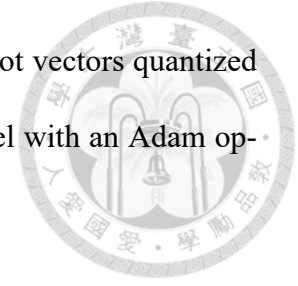
Baseline and TTS Models

Five baseline vocoder models were used in our experiments. We trained three vocoder models, including autoregressive and non-autoregressive methods. We also trained a TTS model to evaluate their performance for speech synthesis.

- **WaveNet** An 8-bit WaveNet from public implementation ¹ were used. The model was with 30 layers, 3 dilation cycles, 128 residual channels, 256 gate channels, and 128 skip channels. The upsampling layers for acoustic features had upsam-

¹https://github.com/r9y9/wavenet_vocoder

pling rates of $\{5,8,5\}$. Both input and output were 8-bit one-hot vectors quantized using μ -law companding transformation. We trained the model with an Adam optimizer [106] for 500k iterations.



- **WaveRNN** We used the public implementation ² to build an 8-bit WaveRNN as a faster autoregressive baseline. The dual softmax layer and efficiency optimization techniques proposed in [5] were not adopted. Two GRU layers in the models had 512 channels. The upsampling layers for acoustic features had upsampling rates of $\{5,8,5\}$. The input and output were 8-bit quantized vectors. The network is trained with an Adam optimizer for 500k iterations.
- **Parallel WaveGAN** Though there were many GAN-based neural vocoders proposed recently [7, 35, 123, 126], we chose Parallel WaveGAN from public implementation ³ as a non-autoregressive baseline for its stable and efficient training. The upsampling rates for acoustic features were $\{5,8,5\}$. We followed [35] to set the parameters of the generator and discriminator. The discriminator was fixed for the first 100k steps and jointly trained with the generator for 400k steps. An Adam optimizer were used during training.
- **Tacotron 2** A Tacotron 2 [1] was built as a frontend for text-to-speech synthesis. The model was modified from the public implementation by NVIDIA ⁴. We applied the reduction factor in [16] to improve convergence speed. The implementation is publicly available ⁵.

The other two vocoder models were LPCNet and WaveGlow. The implementations

²<https://github.com/fatchord/WaveRNN>

³<https://github.com/kan-bayashi/ParallelWaveGAN>

⁴<https://github.com/NVIDIA/tacotron2>

⁵<https://github.com/BogiHsu/Tacotron2-PyTorch>

of these two models were officially released by the authors. Note that the original training settings were not exactly the same as those mentioned above. However, changing the settings without further parameter tuning may lead to suboptimal performance, and for these two models, it's not easy tuning the parameters and extensively retraining. The input of LPCNet is specially designed acoustic features [41], including 18-dimensional BFCC and two pitch parameters (period and correlation), which are different from a Mel-spectrogram. Training a WaveGlow model takes eight GPUs and several days [6]. Hence, using the official pre-trained models is a simple and practical way to show the best performance of these methods, and we only used them in the single speaker evaluations.

- **LPCNet** The pre-trained LPCNet model was from the official implementation ⁶. The acoustic features used and the detailed parameters were set following [41]. The model was implemented in Python using Keras and trained on 16 kHz utterances. At inference time, the model generated speech in either Python or C.
- **WaveGlow** The pre-trained WaveGlow model was from the official implementation ⁷, built in Python using PyTorch, and trained on LJ Speech. The model took Mel-spectrograms as input and generated 22 kHz utterances. The detailed parameters were set following [6].

Proposed Models

Figure 4.5 and Figure 4.7 show detailed parameters of the proposed model and post-filter, including kernel sizes, channel sizes, and numbers of layers. The WN modules in Figure 4.5 (a) and Figure 4.7 had the same dilation size growth rate. The dilation sizes of

⁶<https://github.com/xiph/LPCNet>

⁷<https://github.com/NVIDIA/waveglow>

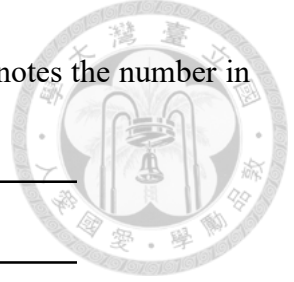


Table 4.1: Details of the multi-resolution STFT auxiliary loss. m denotes the number in L_{sc}^m and L_{mag}^m .

m	1	2	3
FFT size	2048 (93 ms)	1024 (46 ms)	512 (23 ms)
Hop size	400 (18 ms)	200 (9 ms)	100 (5 ms)
Window size	2000 (91 ms)	1000 (45 ms)	500 (23 ms)

different layers grew exponentially in cycles, i.e., $[1, 2, 4, \dots, 32, 1, 2, 4, \dots]$. The upsampling network had upsampling rates of $\{5, 5\}$ and 80 channels, except for the output layer, which had 256 channels.

We first trained all modules but the post-filter with the cross-entropy loss for 500k steps. In each training step, the model generated subbands from the first to the eighth with ground truth previous subbands as input (i.e., the teacher-forcing mechanism). Then we fixed the weights and followed the same training process to train only the post-filter with L_{PF} for 500k steps. We set $M = 3$ for calculating L_S . Table 4.1 lists the detailed parameters for the STFT. We used a batch size of 8 and the Ranger optimizer [127] during training.

To further improve the synthesis speed, we applied the grouping mechanism in [6] to both WN modules in Figure 4.5 (a). The modified WN module is shown in Figure 4.9. The WN input sequence, hidden state, and acoustic feature are first reshaped. The channel sizes increase while the lengths are shortened inversely. We denote the model with this mechanism as $g-i$, where i indicates the factor for reshaping. Note that we did not apply the grouping mechanism to the post-filter to preserve the performance. The method was initially used to build a flow-based model for 1-D speech signals. It has shown effectiveness in improving inference efficiency since the input length is significantly reduced while

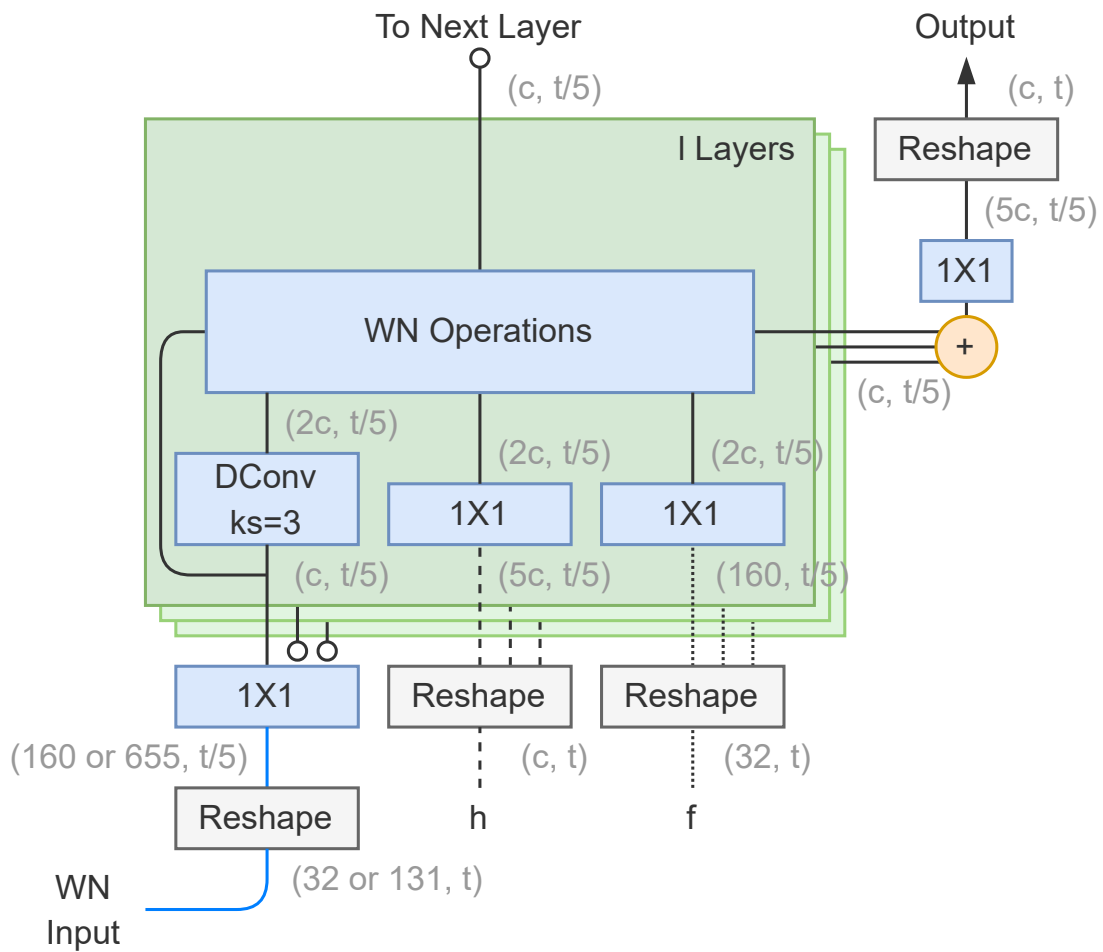
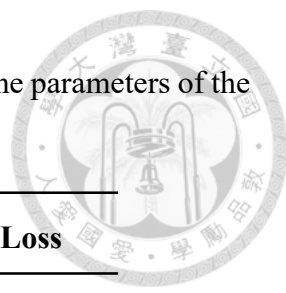


Figure 4.9: WN module with the grouping mechanism (g-5). Some operations are omitted for simplicity. (c, t) represents a t -length sequence of c -dimensional vectors.

Table 4.2: Details of different models. For Parallel WaveGAN, only the parameters of the generator are counted.



Model	Label	Type	Arch.	Loss
WaveNet	WN	AR	WN	CE
WaveRNN	WR	AR	RNN	CE
LPCNet	LPC	AR	RNN	CE
WaveGlow	WG	NAR	WN+Flow	Flow
Parallel WaveGAN	PWG	NAR	WN	Adv.+Aux.
Proposed	Proposed	AR	WN	CE+Aux.
-PF	-PF	AR	WN	CE
g-5	g-5	AR	WN	CE+Aux.
g-10	g-10	AR	WN	CE+Aux.

the channel sizes of the hidden layers remain the same [57, 96].

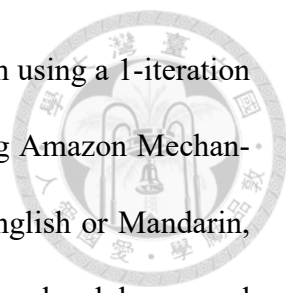
All the baseline, TTS, and proposed models were trained on a single NVIDIA V100 GPU using PyTorch. Table 4.2 lists the details of different models. The third column shows whether the model is autoregressive (AR) or non-autoregressive (NAR). The fourth column describes the main architecture used to build the model, the WN module mentioned in Section 4.2.4, RNN models (RNN), or a specially designed flow-based model (Flow) [6]. The fifth column lists the loss functions for training, including cross-entropy loss (CE), adversarial loss (Adv.), auxiliary loss in [35] (Aux.), and specially designed loss for the flow-based model (Flow) [6]. As described in Section 4.2.4, we applied FAR and BAR to WaveNet to build the proposed models. Consequently, the details of the proposed models are the most similar to the WaveNet model.



4.3.4 Evaluation Metrics

We used four objective and one subjective metrics in the following experiments to evaluate the distortion of the generated speech in different perspectives and the quality under human perception.

- **Objective Evaluation** We used Mel-cepstral distortion (**MCD**, reported in dB) [107] to evaluate the distortion in the frequency domain. Mel-cepstrums were first extracted from ground truth and generated speech. The root mean square error (RMSE) was then calculated frame-by-frame, and the average RMSE per frame represented the distortion. To further show the pitch accuracy of generated speech, we calculated the RMSE of F0 (**F0-RMSE**, reported in Hz) and $\log_2 F0$ (**LogF0-RMSE**, reported in $10^{-2} \log_2 \text{Hz}$). We also calculated the error rate of voiced/unvoiced (V/UV) flags (**V/UV Error**, reported in %), which was the percentage of the frames with mismatched V/UV flags. The F0 and V/UV information were extracted using pYIN algorithm [128].
- **Subjective Evaluation** We conducted modified Multiple Stimuli with Hidden Reference and Anchor (**MUSHA**) tests [129] to rate the quality of generated speech under human perception. In our MUSHRA test, samples with the same speech content but generated by different systems were presented to the raters side by side. The raters then scored the samples from 0 to 100 based on their naturalness. A higher score indicates better quality in naturalness. Similar to the setup in [31, 33], and [130–134], when evaluating samples with the same speech content, the raters were not forced to score at least one system with 100. We used the ground truth sample as the upper anchor and an extra system to generate the lower anchor. The system



simply reconstructed the speech signal from a Mel-spectrogram using a 1-iteration Griffin-Lim algorithm [22]. All the raters were recruited using Amazon Mechanical Turk. The raters should self-report as native speakers (English or Mandarin, depending on the evaluation). We also asked the raters to wear headphones and stay in a quiet environment while taking the test. We randomly selected 20 utterances for the evaluation, and each utterance was scored by at least 15 subjects.

After the objective and the subjective evaluations, paired t-tests were conducted to show the statistical significance. In the ablation study, we calculated the p-values between the proposed method without PF and other systems for better comparison. For the rest of the experiments, we calculated the p-values between the proposed method and other systems.

4.4 Results

4.4.1 Objective Evaluation

In this section, we used different objective metrics to evaluate the efficiency and performance of the proposed methods and other vocoders. All the models were trained using the LJ Speech corpus. We separated 100 utterances as the test set.

Model Sizes and Inference Speed

Table 4.3 shows the model sizes of different methods and their inference speed. The second column reports the number of model parameters (in millions). Among all methods, Parallel WaveGAN was the smallest in size. The adversarial network [135] enabled

Table 4.3: Details and inference speed (w/o and w/ GPU) of different models. The values in the brackets are the standard deviations.

Model	Size (M)	Code	CPU (kHz)	GPU (kHz)
WN	4.7	py	0.19 (0.00)	0.13 (0.00)
WR	4.3	py	1.14 (0.02)	1.87 (0.02)
LPC	1.8	py/c	0.04 (0.00)/ 103.7 (1.5)	0.04 (0.00)/-
WG	87.7	py	6.2 (0.2)	1203.8 (487.2)
PWG	1.3	py	19.5 (1.5)	1325.8 (62.6)
Proposed	5.8	py	8.9 (0.3)	393.1 (4.0)
-PF	5.6	py	9.2 (0.3)	415.6 (3.4)
g-5	7.0	py	27.9 (0.8)	891.6 (205.1)
g-10	7.3	py	46.3 (1.1)	1257.0 (480.4)

the lightweight generator in Parallel WaveGAN to learn effectively with a discriminator. WaveGlow, on the other hand, was the largest model. The flow-based architecture made the network inevitably deep [6, 52]. The proposed model had about 30% more parameters than WaveNet since the network was deeper. The sizes of those with the grouping mechanism (g-5 and g-10) were with more parameters since their channel sizes were 5 times and 10 times larger in the hidden layers, respectively, as shown in the blue path of Figure 4.9.

As for measuring the inference speed, we tested these models using an Intel i7-6700K CPU and an NVIDIA 2080Ti GPU. Since the efficiency of non-autoregressive models might be affected by the length of the target speech at the inference stage, we selected 8 utterances with various lengths. The duration of the audio clips ranged from 2 to 10 seconds, and the average duration was 5.8 seconds. The fourth and fifth columns of Table 4.3 shows the generating rates (number of generated samples per second, in kHz) with and without GPU, respectively. We concluded our observations as follows:

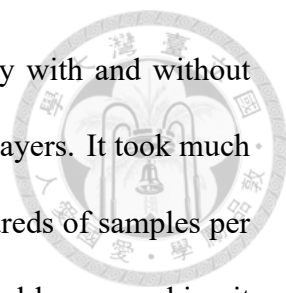
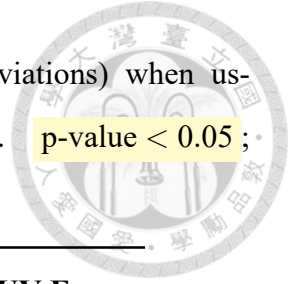
- 
- The autoregressive models in Python generated speech slowly with and without GPU. WaveNet was a deep network with many convolutional layers. It took much computing time in each iteration and could only generate hundreds of samples per second. WaveRNN used two GRUs to replace the convolutional layers, making it faster and more lightweight, but both methods were far from real-time (22 kHz). LPCNet was built and trained in a different framework (Keras). It was inefficient while generating in Python. However, with the highly optimized implementation in C, the inference speed without GPU reached 103.7 kHz, even faster than all the other non-autoregressive methods. Since there are no other C implementations of the non-autoregressive models for comparison, the results of LPCNet are listed only to show the improvement of the C implementation over the Python implementation.
 - The non-autoregressive methods generated utterances more efficiently with and without GPU. Parallel WaveGAN was at 1325.8 kHz and reached 60 times faster than real-time with GPU. In the case of inference only with CPU, it was still at a rate of 20 kHz. The high efficiency came from Parallel WaveGAN's lightweight generator and its parallel synthesis architecture. WaveGlow, when with GPU, reached a similar result to Parallel WaveGAN. The inference speed was much slower without GPU since the model size was larger, making the power of parallel computing more critical.
 - The proposed model increased the autoregressive inference speed by changing from the time domain to the frequency domain and the bit precision domain. We also showed in the fourth and fifth rows that using a compact PF to sample signals from posteriorgrams had only a little degradation on speed. The proposed model was 18 times faster than real-time with GPU, shortening the performance gap between

Table 4.4: Objective evaluation results (means and standard deviations) when using ground truth acoustic features from the test set of LJ Speech. p-value < 0.05 ;

p-value < 0.01 .



Model	MCD (dB)	F0-RMSE (Hz)	LogF0-RMSE ($10^{-2}\log_2\text{Hz}$)	V/UV Error (%)
WN	3.82 (0.37)	8.83 (16.86)	4.51 (3.97)	6.91 (2.45)
WR	2.98 (0.30)	3.97 (1.63)	2.61 (2.00)	4.42 (2.41)
WG	3.16 (0.22)	4.90 (2.38)	3.10 (1.70)	5.14 (2.29)
PWG	2.41 (0.11)	5.06 (2.67)	3.47 (2.76)	6.55 (3.49)
Proposed	1.93 (0.13)	4.05 (1.66)	2.66 (1.82)	5.09 (3.12)
-PF	2.59 (0.22)	4.14 (1.37)	2.67 (1.51)	4.64 (2.18)
g-5	1.94 (0.12)	4.20 (1.09)	2.67 (0.58)	5.03 (2.59)
g-10	2.26 (0.14)	4.29 (1.22)	2.73 (0.69)	5.15 (2.58)

autoregressive and non-autoregressive methods. Finally, we measured the rate of the proposed models with the grouping mechanism and showed the mechanism's effectiveness in improving efficiency. When inference only with CPU, g-5 and g-10 achieved 1.3 and 2.1 times faster than real-time, respectively.

- Unlike the autoregressive models, which had a fixed input length, the input length of the non-autoregressive models varied according to the length of the target speech. Hence, for the non-autoregressive models, the standard deviations of the inference speed were much larger, indicating that their efficiencies were affected more by the various speech lengths.

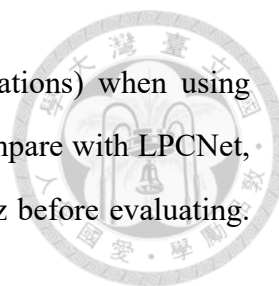


Table 4.5: Objective evaluation results (means and standard deviations) when using ground truth acoustic features from the test set of LJ Speech. To compare with LPCNet, ground truth and generated utterances were downsampled to 16 kHz before evaluating.

p-value < 0.05 ; p-value < 0.01 .

Model	MCD	F0-RMSE	LogF0-RMSE	V/UV Error
	(dB)	(Hz)	($10^{-2}\log_2\text{Hz}$)	(%)
WN	3.70 (0.38)	7.82 (12.70)	4.13 (2.96)	6.51 (3.03)
WR	2.60 (0.29)	4.00 (1.92)	2.52 (1.16)	4.70 (2.72)
LPC	8.45 (0.37)	15.87 (4.97)	9.99 (2.00)	17.62 (3.64)
WG	2.11 (0.20)	5.00 (2.12)	3.03 (1.19)	5.18 (2.55)
PWG	2.45 (0.12)	5.07 (2.21)	3.19 (1.34)	6.67 (2.92)
Proposed	1.91 (0.15)	5.00 (2.88)	3.15 (2.03)	4.61 (2.65)
g-5	1.91 (0.13)	5.85 (6.11)	3.34 (2.45)	4.79 (2.61)
g-10	2.24 (0.16)	5.07 (4.91)	2.85 (1.00)	5.23 (2.59)

Evaluation Results

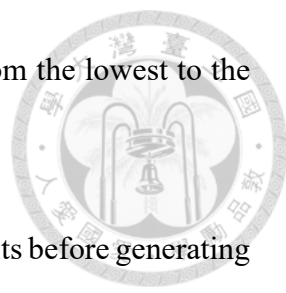
We took all the test set of LJ Speech to evaluate the models using different objective metrics. The results are listed in Table 4.4. The proposed models (Proposed, g-5, and g-10) and Parallel WaveGAN had a lower MCD, which indicated the magnitude information was better restored. We attributed the performance to the STFT auxiliary loss [35] since it directly optimized the magnitude distortion in the frequency domain. Regarding F0-RMSE, WaveRNN outperformed the others, but the difference was small since the values of F0 varied at a much larger scale. Besides, the difference between WaveRNN and the proposed models is not significant. Similarly, for LogF0-RMSE and V/UV Error, the results of most models were close.

Since LPCNet generated 16 kHz speech, we also downsampled the ground truth and the generated utterances to 16 kHz for another evaluation. The results are listed in Table 4.5. We found that LPCNet performed worse than the others. The distortion may be introduced when applying the block-sparse matrices to reduce the model complexity [41].

Ablation Study

We conducted a comprehensive ablation study to examine the effectiveness of the proposed methods and validate our hypothesis in Section 4.1. The different settings based on the proposed model without *PF* (-PF) are denoted as follows:

- **BAR-2** The BAR in -PF was modified to generate only the first two bits before generating the 8-bit signal.
- **-BAR** The BAR in -PF was removed. The architecture remained the intact; only each channel to predict the first three bits and to take them as input was removed.
- **-FAR, MB** The FAR in -PF was removed while keeping the architecture intact. The same model independently predicted one of the eight subbands (MB) at a time.
- **-FAR, FB** The FAR in -PF was removed while keeping the architecture intact. The model directly predicted a full-band (FB) waveform.
- **-AR, MB** The FAR and BAR in -PF were removed while keeping the architecture intact. The same model independently predicted one of the eight subbands (MB) at a time.
- **-AR, FB** The FAR and BAR in -PF were removed while keeping the architecture intact. The model directly predicted the 8-bit signal of a full-band (FB) waveform.

- 
- **FL2H** The FAR in -PF was modified to generate subbands from the lowest to the highest frequency.
 - **BL3B** The BAR in -PF was modified to generate the last three bits before generating the complete 8-bit signal.
 - **FGT n and BGT n** For $i \leq n$, after a sequence was generated in the i th prediction, it was replaced by the ground truth and used as input for the next prediction. For example, in BGT1, the real first bits of each subband replaced the generated ones; in FL2H, FGT1, the real lowest-frequency subband replaced the generated one to predict the next subband and to synthesize the full-band waveform.
 - **FGT4mix** Similar to FGT n , four subbands were replaced by the ground truth for the next prediction. Instead of replacing the first four subbands, the first (the highest-frequency), third, fifth, and seventh subbands were replaced.

The results are listed in Table 4.6, and Figure 4.10 shows the Mel-spectrograms of the generated utterances. We first discuss the effectiveness of the proposed methods and how the results support our hypothesis, concluded as follows:

- The performance degraded when removing FAR (-PF \rightarrow -FAR, MB/FB), BAR (-PF \rightarrow -BAR), or *PF* (Proposed \rightarrow -PF) or when reducing the number of different bit precision in BAR (-PF \rightarrow BAR-2), indicating the effectiveness of the proposed methods. Applying FAR, BAR, or PF effectively improved performance, and combining the three methods led to the best results.
- Directly predicting a waveform without autoregressive methods (-AR, FB) yielded severely distorted results, whereas dividing the target waveform into subbands (-

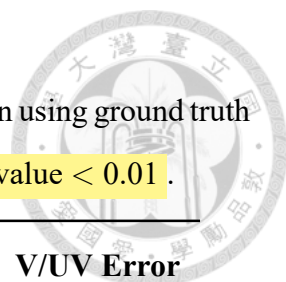


Table 4.6: Ablation study results (means and standard deviations) when using ground truth acoustic features from the test set of LJ Speech. **p-value < 0.05** ; **p-value < 0.01** .

Model	MCD (dB)	F0-RMSE (Hz)	LogF0-RMSE ($10^{-2}\log_2\text{Hz}$)	V/UV Error (%)
Proposed	1.93 (0.13)	4.05 (1.66)	2.66 (1.83)	5.09 (3.12)
-PF	2.59 (0.22)	4.14 (1.37)	2.67 (1.51)	4.64 (2.18)

Ablation Study on the Proposed Methods (based on -PF)

BAR-2	2.70 (0.21)	4.31 (1.80)	2.77 (1.58)	5.30 (2.63)
-BAR	4.79 (0.36)	4.61 (3.18)	2.95 (2.54)	5.00 (2.31)
-FAR, MB	6.91 (0.58)	7.11 (2.01)	4.76 (1.32)	27.28 (9.74)
-FAR, FB	11.57 (0.54)	9.61 (3.01)	7.00 (1.94)	52.54 (11.67)
-AR, MB	9.91 (0.54)	5.34 (2.86)	3.47 (2.37)	7.04 (3.10)
-AR, FB	15.41 (0.61)	7.03 (3.17)	4.55 (1.76)	17.86 (7.87)

Ablation Study on the Generation Order (based on -PF)

FL2H	6.15 (0.57)	8.22 (6.62)	5.50 (3.94)	36.43 (10.75)
FL2H, FGT1	3.68 (0.37)	3.04 (1.13)	1.89 (0.69)	2.73 (1.89)
FL2H, FGT2	1.95 (0.22)	2.42 (1.04)	1.48 (0.60)	2.13 (1.85)
FGT1	2.52 (0.22)	3.42 (1.02)	2.16 (0.57)	4.66 (2.58)
FGT2	2.54 (0.21)	3.53 (1.44)	2.20 (0.88)	4.70 (2.43)
FGT4mix	3.39 (0.29)	3.28 (2.16)	2.12 (1.69)	4.11 (2.29)
BL3B	6.20 (0.43)	3.89 (1.10)	2.46 (0.62)	5.40 (2.52)
BL3B, BGT1	6.28 (0.42)	4.30 (2.16)	2.70 (1.23)	5.32 (2.50)
BL3B, BGT2	5.87 (0.41)	4.47 (4.11)	2.80 (2.79)	5.14 (2.53)
BGT1	2.74 (0.38)	2.54 (1.24)	1.57 (0.71)	3.74 (2.37)
BGT2	2.49 (0.24)	2.31 (1.29)	1.46 (0.74)	2.71 (2.11)
-FAR, FB, BGT1	9.99 (0.53)	4.63 (1.26)	2.95 (0.72)	8.33 (3.51)
-FAR, FB, BGT2	6.63 (0.53)	2.07 (0.93)	1.28 (0.55)	2.71 (2.04)

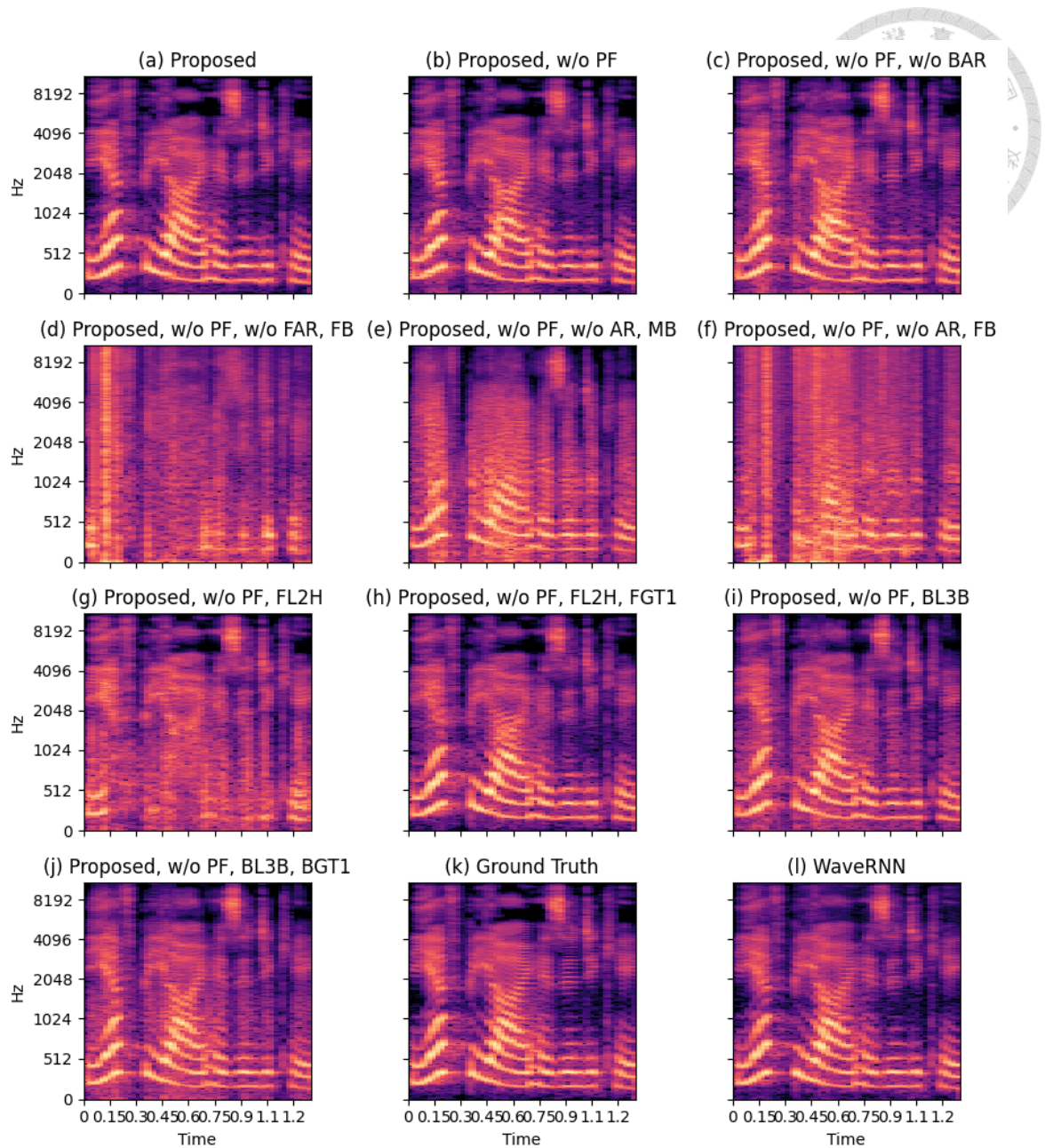


Figure 4.10: Mel-spectrograms of generated speech using ground truth acoustic features from LJ Speech (LJ050-0241).

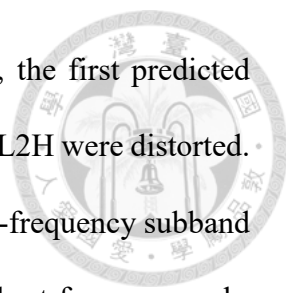
AR, MB) improved performance. Additionally, applying an autoregressive mechanism further enhanced speech quality. Specifically, FAR (-AR, MB \rightarrow -BAR) demonstrated improvements across all metrics, and BAR (-AR, MB/FB \rightarrow -FAR, MB/FB) led to a better MCD. The findings above strongly support our hypothesis that dividing a target into multiple smaller parts and iteratively generating each part

conditioned on predicted parts reduce prediction complexity.

- Although -FAR, MB/FB exhibited worse pitch-related results (F0-RMSE, LogF0-RMSE, and V/UV Error) than -AR, MB/FB, the degradation was due to the architecture instead of the autoregressive mechanism. In -FAR, MB/FB, 1-bit signals were first predicted without using the second WN module, as shown in Figure 4.5 (a). Besides, as mentioned in Section 4.1, 1-bit signals contain precise F0 contours. The two facts above suggest that -FAR, MB/FB predicted much pitch information leveraging only a portion of the model capacity, leading to less accurate results. In contrast, -AR, MB/FB directly predicted the complete 8-bit signals, fully utilizing the entire model to produce pitch information. The BAR in -PF was not affected by this issue, which will be discussed together with the generation order of BAR in the subsequent paragraphs.

The ablation study results also demonstrate the importance of the generation orders in FAR and BAR. Specifically, inverting the generation order of FAR from generating the highest-frequency subband first to the lowest-frequency one first (-PF \rightarrow FL2H) resulted in increased distortion and errors. This can be explained based on the following facts:

- Since the lower-frequency subbands contain more speech information, such as pitch, energy, and voicing status, these subbands have a greater impact on speech quality.
- In autoregressive generation, it is commonly observed that predictions in the early steps are less accurate due to the lack of information from previous predictions or hidden states. Without previous information, the first generation process is the same as predicting without an autoregressive mechanism.

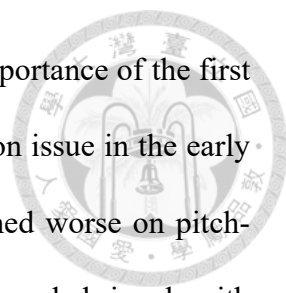


Without sufficient previous information in the first generation step, the first predicted highest-frequency subband in -PF and lowest-frequency subband in FL2H were distorted. However, FL2H was much less performant since the degraded lowest-frequency subband had a more negative impact on speech quality than the degraded highest-frequency subband.

In FL2H, the lowest-frequency subband was first predicted without previous predictions, hence of low quality. The distorted lowest-frequency subband and the lack of information in the early steps would negatively affect subsequent predictions. Consequently, the distorted lower-frequency subbands led to worse speech quality. To demonstrate the importance of lower-frequency subbands, we evaluated FL2H, FGT1 and FL2H, FGT2. With ground truth low-frequency subbands provided, the improvements were significant, indicating the importance of lower-frequency subbands to high-quality speech.

Although the higher-frequency subbands in -PF were first generated and could also be distorted, the performance was less affected as these subbands contain much less speech information. Also, the improvements were not as significant as in FL2H when ground truth high-frequency subbands were provided (-PF \rightarrow FGT1 and -PF \rightarrow FGT2). These findings indicate that the higher-frequency subbands have minimal impact on speech quality and can be generated in the early steps.

Regarding BAR, we conducted similar experiments. Inverting the generation order from generating the first three bits first to the last three bits first (-PF \rightarrow BL3B) also showed worse results, while providing the ground truth last few bits (BL3B, BGT1, and BL3B, BGT2) did not improve the performance. On the other hand, providing the ground truth first few bits (BGT1, and BGT2) led to better pitch-related results, indicating that the



first few bits are more crucial for accurately predicting pitch. The importance of the first few bits, combined with the previously mentioned lack of information issue in the early steps, provides another explanation for why -FAR, MB/FB performed worse on pitch-related metrics, which is that BAR in -FAR, MB/FB predicted low-bit-coded signals with less accurate pitch information in the early steps. In contrast, the BAR in -PF was less affected by the lack of information. Despite generating the first three bits in the early steps, -PF still produced high-quality speech since it combined FAR and BAR. In -PF, the lower-frequency subbands, which were more critical to high-quality speech, were predicted with more acoustic information from previous subbands and hidden states. The hidden states possessed rich information about previous predictions since they were iteratively computed throughout the generation process. Sufficient information helped the BAR in -PF to predict high-quality low-bit-coded signals of the lower-frequency subbands, leading to better results.

We also observed that in BAR, the MCD increased when the real first target was given (-PF \rightarrow BGT1 and BL3B \rightarrow BL3B, BGT1) and decreased when the real first two targets were given (-PF \rightarrow BGT2 and BL3B \rightarrow BL3B, BGT2)⁸. We inferred this was due to the mismatch between the consecutive subbands when the given ground truth information was insufficient. BGT1 generated a subband with only partial ground truth information (the first bits). The generated subband might still have differences, such as phase discrepancy, from the ground truth, leading to a mismatch with the ground truth first bits given in the next subband prediction. In BGT2, with more ground truth information (the first and the second bits), the generated subband was closer to the ground truth, decreasing the mismatch. The explanation above also applies to BL3B, BGT1 and BL3B, BGT2.

⁸The first two targets of -PF were the first two bits of the 8-bit signal, and the first two targets of BL3B were the last two bits.

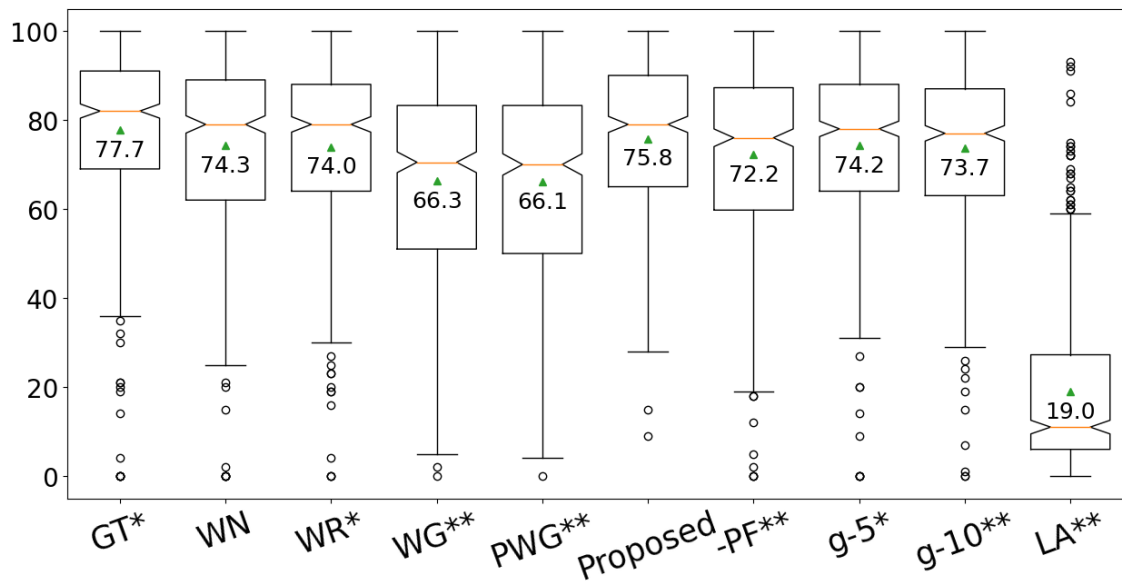


Figure 4.11: MUSHRA results when using ground truth acoustic features from the test set of LJ Speech. *: p-value < 0.05; **: p-value < 0.01.

To verify that a mismatch may occur when consecutive subbands are partial ground truth and partial generated, we evaluated FGT4mix. In FGT4mix, a generated subband mismatched the ground truth next subband since the former was not predicted conditioned on the latter. The MCD results hence worsened despite more information being provided, indicating that the mismatch between subbands affects speech quality. Note that FGT1 and FGT2 did not suffer any mismatches since the generated parts were predicted conditioned on the ground truth subbands. Furthermore, for -FAR, FB, which applies only BAR and directly predicts a full-band waveform, there was no mismatch since the sub-band mechanism was removed. All metrics were improved when part of the ground truth were given (-FAR, FB \rightarrow -FAR, FB, BGT1 and -FAR, FB \rightarrow -FAR, FB, BGT2). These observations well support the claim that mismatches between subbands degraded speech quality.

The comprehensive ablation study shows the importance of the generation orders and explains how we decided the orders for FAR and BAR according to the results.

4.4.2 Subjective Evaluation



Evaluation on Ground Truth Acoustic Features

We randomly selected 20 utterances from the test set of LJ Speech for the subjective evaluation. The MUSHRA results of different models are listed in Figure 4.11. We concluded our observations as follows:

- Among the baseline models, the autoregressive models bettered the non-autoregressive ones with significant differences ($p\text{-value} < 0.01$). Besides, the baseline models with the same type (autoregressive or non-autoregressive) performed similarly, and there was no significant difference ($p\text{-value} > 0.05$).
- The proposed model outperformed the others and reached a MUSHRA score of 75.8. There was no significant difference between the proposed method and WaveNet, showing that both models generated utterances closest to natural human speech, yet the proposed method reached thousands of times more efficient than WaveNet according to Table 4.3.
- WaveGlow and Parallel WaveGAN achieved intermediate results in the objective evaluation. However, they were still relatively low-quality to the competitive systems, and subjects in the MUSHRA test tended to give lower scores. The audio samples on the demo⁹ page better show the quality of different vocoders.
- The score of the proposed model without PF dropped by 3.6, indicating the importance of PF in synthesizing natural speech. As for the grouping mechanism, the

⁹<https://bogihsu.github.io/TASLP2021-Parallel/Demo/demo.html>

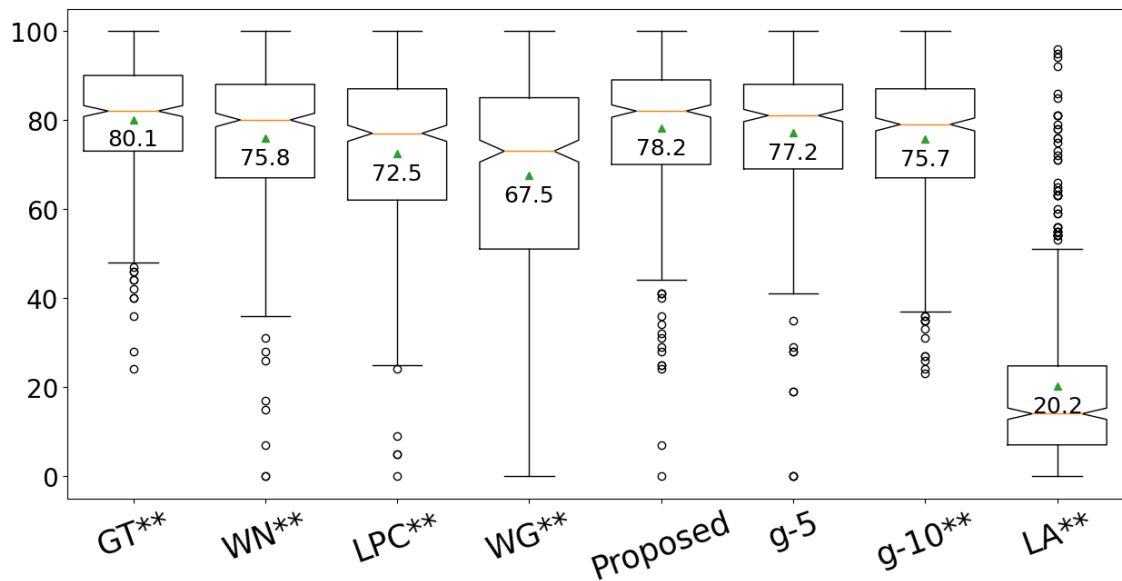


Figure 4.12: MUSHRA results when using ground truth acoustic features from the test set of LJ Speech. To compare with LPCNet, ground truth and generated utterances were downsampled to 16 kHz before evaluating. **: p-value < 0.01.

scores of g-5 and g-10 degraded slightly but still close to WaveNet and WaveRNN, showing the ability to improve the synthesis efficiency while preserving the quality.

Similar to in Section 4.4.1, to compare with LPCNet, we also downsampled the utterances to 16 kHz and ran another MUSHRA test. The results are listed in Figure 4.12. LPCNet performed worse than WaveNet and the proposed methods but still bettered non-autoregressive WaveGlow.

Evaluation on Acoustic Features from Tacotron 2

We combined vocoders with a Tacotron 2 model as complete TTS systems for further evaluation. The Tacotron 2 was trained using the training set of LJ Speech. During inference, we selected 20 sentences from Harvard Sentences [136] as the text input for the systems.

The results are shown in Figure 4.13. Compared with the evaluation using ground

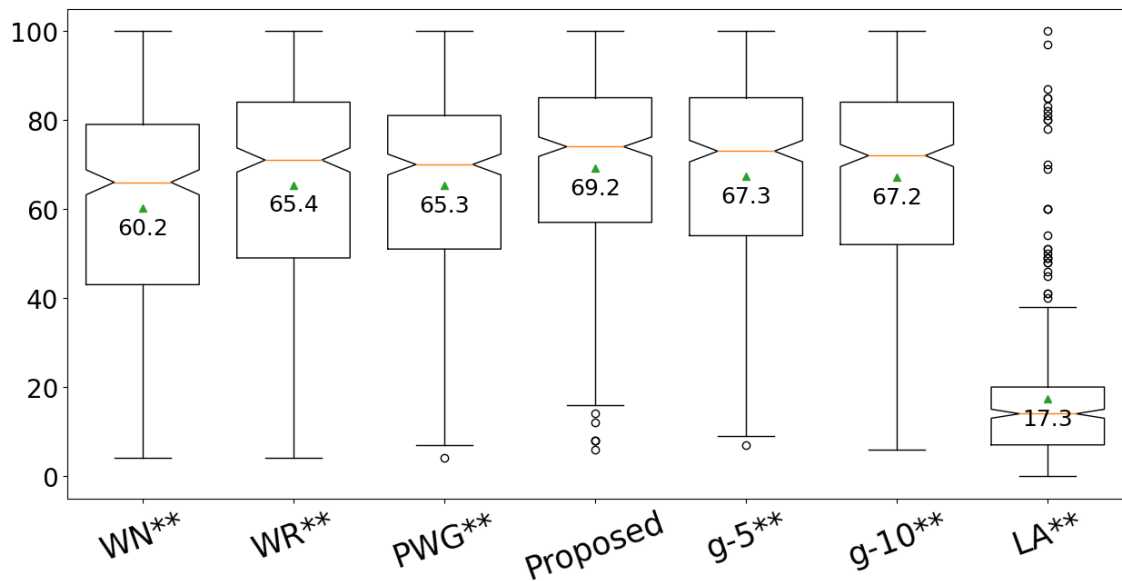


Figure 4.13: MUSHRA results when using acoustic features generated from Tacotron 2. **: p-value < 0.01.

truth acoustic features, the scores between the baseline methods become closer. We found that features from Tacotron 2 had more distortion than ground truth features. In this case, WaveNet and WaveRNN were unstable and less performant. The reduced robustness of the autoregressive models to artificial acoustic features could be attributed to the error propagation from autoregressive properties in the time domain. When a suboptimal sample is generated with distorted acoustic features, it is used as the condition for the next prediction. Even if the acoustic features are clean in subsequent time steps, a flawed sample from the previous step can lead to further erroneous predictions, causing the errors to propagate and resulting in speech with noise and artifacts.

In contrast, the non-autoregressive Parallel WaveGAN was much more stable regardless of the quality of the input features. The proposed models, possessing the properties of autoregressive and non-autoregressive methods, were not affected by error propagation, either. All three models maintained high quality and outperformed the baseline models with significant differences (p-value < 0.05).



4.4.3 Generalization Evaluation

In this section, we studied the generalization abilities of the baseline and proposed models under different situations, including synthesizing speech of unseen speakers and high-fidelity speech with a 44 kHz sampling rate.

Generalization to Unseen Speakers

To build a multi-speakers version of the vocoders, we used the VCTK dataset for training. We use utterances from speaker *bdl* and *slt* in CMU ARCTIC as a male and a female test set, respectively. For each test set, we took 30 utterances (arctic_b0500~arctic_b0529) for objective evaluation and randomly selected 10 audio clips for subjective evaluation.

Table 4.7 and Figure 4.14 list the results. The tendency of the objective scores between different models is similar to the results in Section 4.4.1. WaveRNN performed slightly worse in MCD, reflected in generated speech with more noise and a lower perceptual score. The MUSHRA results showed that WaveNet generalized best. The proposed model also had a higher score than WaveRNN and Parallel WaveGAN, indicating its ability to generalize to unseen speakers.

Generalization to High-Fidelity Dataset

To study the performance when generalization to high-fidelity dataset, we trained the vocoders on our 44 kHz internal Mandarin speech corpus. All STFT parameters in samples remained the same as described in Section 4.3 and Table 4.1, i.e., STFT parameters in milliseconds were halved in this 44 kHz experiment. We took 30 utterances from the test set for objective evaluation and randomly selected 20 utterances for subjective evaluation.



Table 4.7: Objective evaluation results (means and standard deviations) when using ground truth acoustic features from speaker *bdl* and *slt* in CMU ARCTIC. The vocoders were trained on VCTK corpus. p-value < 0.05 ; p-value < 0.01 .

Model	MCD (dB)	F0-RMSE (Hz)	LogF0-RMSE ($10^{-2}\log_2\text{Hz}$)	V/UV Error (%)
<i>bdl (male)</i>				
WN	2.58 (0.21)	3.20 (2.52)	4.01 (3.61)	8.22 (4.93)
WR	2.64 (0.21)	2.95 (0.66)	3.64 (0.85)	11.00 (6.02)
PWG	2.90 (0.31)	3.68 (2.53)	4.62 (3.09)	7.93 (4.62)
Proposed	1.98 (0.23)	3.01 (1.33)	3.80 (1.83)	7.22 (5.01)
<i>slt (female)</i>				
WN	2.49 (0.23)	3.81 (1.37)	2.88 (1.00)	4.81 (3.10)
WR	3.56 (0.33)	3.34 (1.11)	2.69 (0.85)	2.79 (2.51)
PWG	2.71 (0.15)	4.35 (1.46)	3.49 (1.09)	4.03 (3.19)
Proposed	2.16 (0.18)	3.04 (1.42)	2.42 (1.09)	2.75 (2.39)

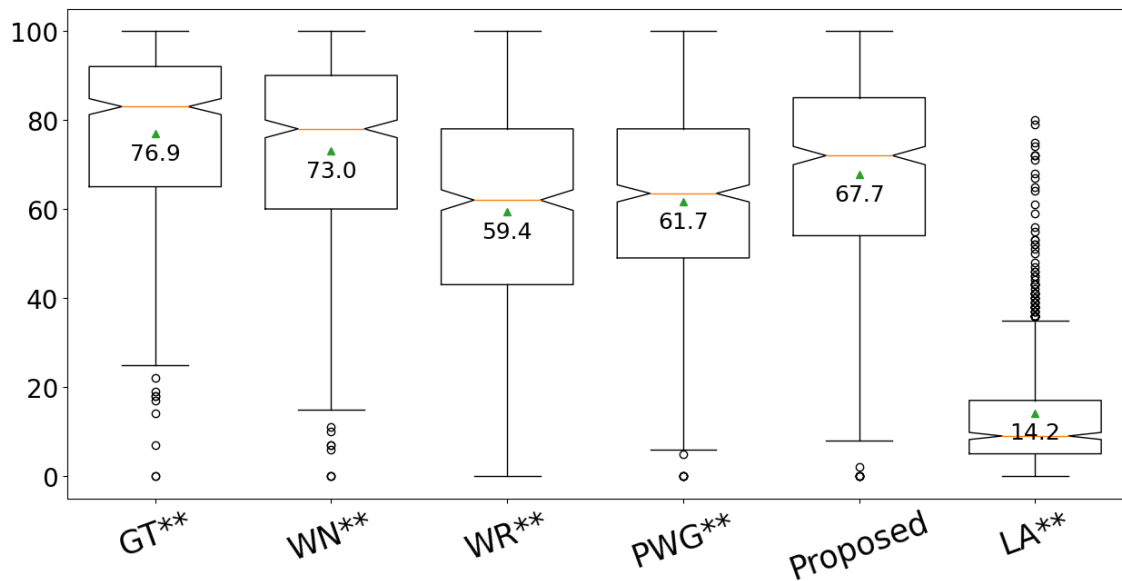


Figure 4.14: MUSHRA results when using ground truth acoustic features from speaker *bdl* and *slt* in CMU ARCTIC. The vocoders were trained on VCTK corpus. **: p-value < 0.01.

Table 4.8: Objective evaluation results (means and standard deviations) when using ground truth acoustic features from the test set of our 44 kHz internal mandarin speech corpus. p-value < 0.05 ; p-value < 0.01 .

Model	MCD (dB)	F0-RMSE (Hz)	LogF0-RMSE ($10^{-2}\log_2\text{Hz}$)	V/UV Error (%)
WN	4.32 (0.88)	31.10 (45.57)	18.94 (19.43)	15.43 (6.63)
WR	5.11 (0.37)	4.59 (3.00)	3.23 (3.70)	7.12 (4.08)
PWG	3.25 (0.20)	9.32 (10.21)	5.71 (4.48)	11.06 (6.08)
Proposed	2.10 (0.26)	6.70 (13.37)	2.49 (1.10)	6.07 (3.48)

The results are shown in Table 4.8 and Figure 4.15. The objective evaluation results showed a similar tendency as previous. We found that speech volume from WaveNet varied in the same utterance, leading to worse objective evaluation results. As for the MUSHRA results, utterances from WaveRNN had a bit more noise but were still graded with good scores. Besides, Parallel WaveGAN failed to generate natural results. On the

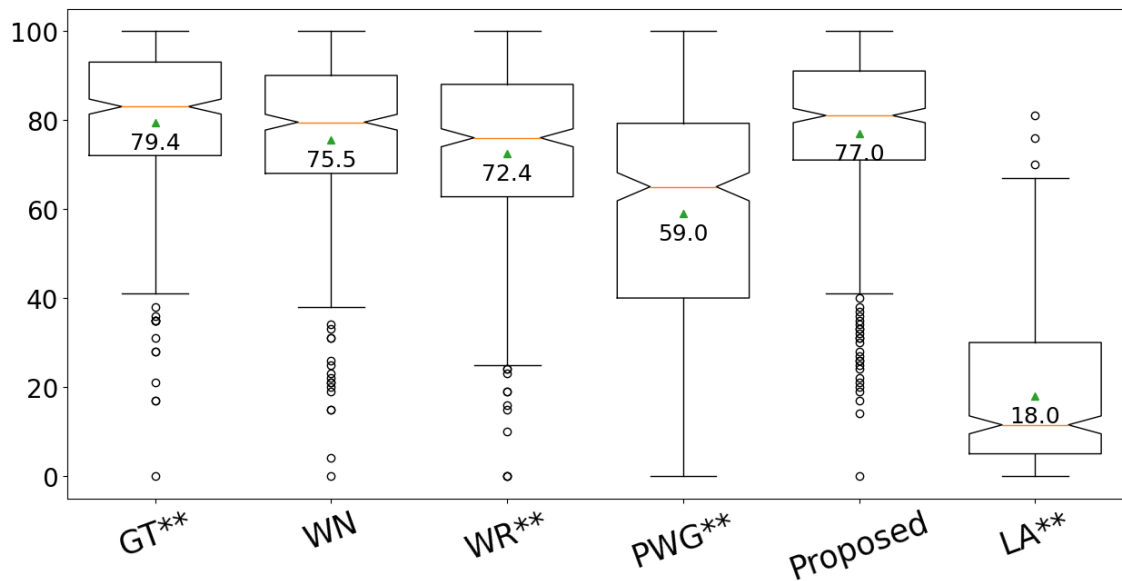


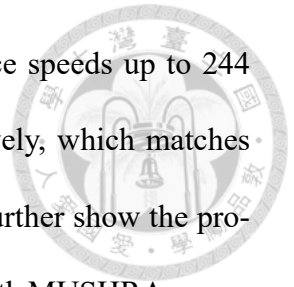
Figure 4.15: MUSHRA results when using ground truth acoustic features from the test set of our 44 kHz internal mandarin speech corpus. **: p-value < 0.01.

other hand, both WaveNet and the proposed model achieved close scores to the ground truth, and there was no significant difference between these two systems, indicating their ability to synthesize natural high-fidelity speech.

4.5 Summary

This chapter addresses the limitations of conventional autoregressive vocoders by introducing innovative methods: frequency-wise autoregressive generation (FAR) and bit-wise autoregressive generation (BAR). FAR iteratively generates utterances across different frequency subbands, while BAR progressively generates utterances with varying bit precision. We further enhance speech quality through a post-filter applied for posterior sampling. Unlike traditional autoregressive methods, our proposed approaches operate with a fixed number of iterations, making inference time independent of speech length. Consequently, our model demonstrates high inference efficiency. Compared with the au-

autoregressive WaveNet model, the proposed model achieves inference speeds up to 244 and 9669 times faster with and without GPU acceleration, respectively, which matches the speeds of non-autoregressive vocoders. Perceptual evaluations further show the proposed model consistently outperforms non-autoregressive methods with MUSHRA scores higher than or close to other autoregressive methods, confirming that the proposed vocoder can produce natural, high-quality speech in various scenarios.





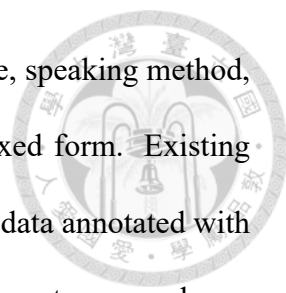


Chapter 5

Data Efficiency in Speech Generation: Improving Text-Guided Voice Conversion with Reinforcement Learning and Human Feedback

5.1 Introduction

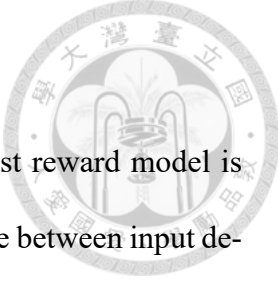
In the previous chapters, we discussed and studied the challenges of computational efficiency in speech generation from various aspects, examining both non-autoregressive and autoregressive neural vocoders. We introduced methods to significantly reduce the computational resources required during training and inference. Expanding our exploration of efficient speech generation, in this chapter, we aim to improve the data efficiency of speech generation. Specifically, we focus on text-guided speech generation and set text-guided voice conversion as our primary task in this study. Text-guided voice conversion involves converting the style of source speech based on text descriptions while preserv-



ing the content. These styles include variations in pitch, speaking rate, speaking method, and emotion, described using human-written sentences without a fixed form. Existing text-guided generation works typically require collecting rich speech data annotated with detailed descriptions to build a model that can comprehend complex sentences and produce expressive speech. In order to alleviate the heavy dependency on human-curated private datasets, this work aims to leverage only publicly available datasets and explore a more effective learning paradigm for developing a text-guided voice conversion model. To this end, we build our model based on a pre-trained open-source model and explore two advanced learning strategies for text-guided voice conversion: reinforcement learning and reinforcement learning from human feedback.

Reinforcement learning (RL) introduces a paradigm where models learn through interacting with specially designed environments. These environments incorporate appropriate metrics and feedback mechanisms to reward model outputs. Since models can learn to maximize rewards through continuous trial and error rather than learn from paired data, this approach improves model performance without additionally collecting more real-world data for the task. Based on RL, reinforcement learning from human feedback (RLHF) [137, 138] uses a small amount of human feedback on model outputs and builds a reward model with similar preferences to humans. This reward model is used as the environment in RL to assess the output of the target model, which is then guided to make predictions more aligned with human preferences. RLHF has shown remarkable results in the field of NLP, and related works are booming [139–141]. In contrast, research applying RL and RLHF to enhance speech generation models has yet to be explored. We argue that RL and RLHF can be more data-efficient methods to improve text-guided voice conversion models since the models can be refined by interacting with environments without the

need to collect more paired text-speech data.



To effectively apply RL and RLHF to our task, building a robust reward model is essential. This reward model needs to accurately evaluate the relevance between input descriptions and output speech, guiding the learning process of the voice conversion model. Given the scarcity of high-quality, diverse text-speech data in our setting, we develop our reward model using contrastive learning. A model built with contrastive learning, such as CLIP [142], has shown powerful zero-shot classification ability competitive with fully supervised models without the necessity for training on labeled data. Our reward model scores the similarity between a speech utterance and a text description, and we show in the experiment that it outperforms traditional classifiers in classifying speech style. Following the RL algorithm and the discrimination from the reward model, we fine-tune the proposed model and significantly improve the conversion results when given complex style descriptions as input. Moreover, we study the help of RLHF. We collect preferences annotated by human raters based on the relevance between generated speech and text descriptions. Then, we explore different strategies to apply human feedback in RL and the impact of using different amounts of human feedback. The subjective evaluation results show that RLHF can improve the model to generate expressive speech more aligned with human preferences.

The contributions of this work are summarized as follow:

- This study pioneers the application of RL and RLHF in speech generation to enrich expressiveness and styles of generated speech. We explore various ways to build a performant reward model and investigate how to leverage human feedback effectively. With the RL algorithm and the guidance of the reward model, we refine

the proposed text-guided voice conversion model, setting a foundation for future advancements in applying RL and RLHF in speech technology.

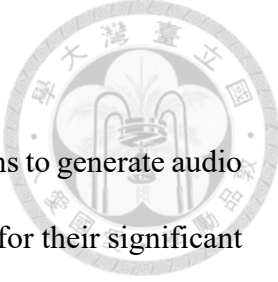
- Experimental results validate the effectiveness of the proposed methods. In the objective evaluation, both RL and RLHF greatly enhance the expressiveness and style accuracy of the output speech by up to 100%, ensuring it better follows the textual descriptions provided. Subjective evaluation results also confirm that after applying RL or RLHF, the proposed model can generate speech that is much more consistent with the target style, and the improvements are statistically significant. In addition, RLHF outperforms RL in pairwise preference tests, indicating a superior alignment with human preferences. Both objective and subjective evaluation results show that the proposed methods learn from the limited data more effectively and efficiently without the need to collect more high-quality, diverse, and paired text-speech data.

5.2 Related Work

5.2.1 Diffusion-Based Text-to-Audio Generation

Research using text to guide speech generation or voice conversion is limited due to the lack of publicly available datasets, as mentioned in Section 2. In contrast, another closely related field, text-to-audio (TTA) generation, has seen significant development. With plenty of public training data, building models to generate audio that matches given text prompts describing styles or events is a popular and growing research direction. In this work, we leverage the ability of a well-trained diffusion-based TTA model to initialize the proposed model. In the following paragraph, we introduce recent works related to

diffusion-based TTA generation.

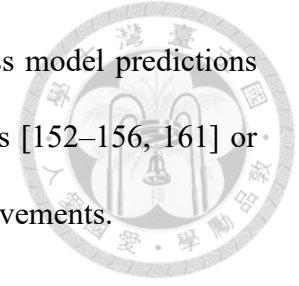


In TTA, given a text description of the target sound, the model aims to generate audio corresponding to this description. Recently, diffusion models, known for their significant progress in image generation [143], have been widely applied to the TTA task and demonstrated promising results. Diffsound [144] introduces a vector-quantized variational autoencoder (VQ-VAE) [145] to convert audio into discrete acoustic units. An autoregressive model is built to generate these acoustic units based on text descriptions, followed by a diffusion model that reconstructs audio Mel-spectrograms from the generated units. In [146] and AudioLDM [147], the authors adopt latent diffusion models [143] (LDM) and show high-quality results. Building upon these innovations, TANGO [148] leverages FLAN-T5 [149], an instruction-tuned large language model, as the text encoder to further improve the ability of the TTA model. Beyond the standard TTA task, [146] and AudioLDM 2 [150] extend the capabilities of TTA models to multimodal scenarios, allowing the models to generate audio from text, images, or video inputs. These approaches, utilizing various pre-trained models as encoders for different modalities, enhance the versatility of TTA models and enable them to produce high-quality audio in diverse contexts.

5.2.2 Reinforcement Learning in Speech Processing

In reinforcement learning (RL), a model learns by interacting with the environment that provides rewards according to model outputs. During training, models target to maximize expected rewards from these environments and progressively refine their predictions. Recently, many works have applied RL techniques to various speech tasks [151], including automatic speech recognition [152–157], speaker recognition [158], speech emotion recognition [159, 160], and speech enhancement [161, 162]. These methods employ ob-

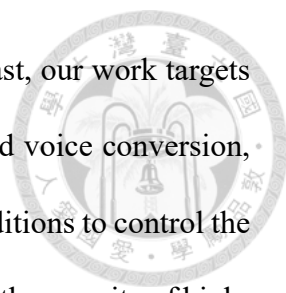
jective metrics [159, 161] or well-optimized systems [162] to assess model predictions and provide feedback rewards. Algorithms such as policy gradients [152–156, 161] or proximal policy optimization [158] are then adopted to ensure improvements.



For the application of RL in speech generation, [163] utilizes speech emotion recognition and RL to improve TTS, enhancing the emotional expressiveness of the synthesized speech. Another work leverages RL and human preferences to improve the naturalness of TTS outputs [164]. These works focus on simple goals, such as improving solely emotional expressiveness or overall speech quality. In contrast, our study aims for a more complex setting, where the model generates speech with specified compound styles. A compound style is a combination of speech styles in different aspects, such as speaking method and emotion. Given the impracticality and cost of collecting data for all style combinations for supervised learning, we introduce RL as an effective solution. Furthermore, this work focuses on text-guided speech generation rather than TTS, a relatively unexplored area compared to the well-studied field of TTS.

5.2.3 Reinforcement Learning and Human Feedback in Audio Generation

Some concurrent works to ours recently explore the use of RL or human feedback to improve the TTA task. Tango 2 [165] proposes an approach to generate a pairwise text-audio preference dataset and apply direct preference optimization (DPO) [140] to optimize the TTA model. With a similar goal, BATON [166] collects human feedback to finetune an off-the-shelf TTA model. Both works are built on the previously proposed text-to-audio model [148], aiming to make the generated audio more aligned with the complex scenario described in the input text. Besides, for TTA, plenty of paired text-audio data is available



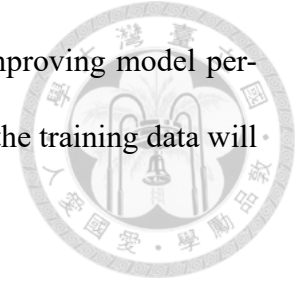
to build a good model even before further finetuning [167]. In contrast, our work targets a different and limitedly explored speech generation task, text-guided voice conversion, where we need to design an architecture that takes multiple input conditions to control the content, speaker identity, and style of the output speech. Furthermore, the scarcity of high-quality data highlights the challenge of this task. In this work, we introduce RL to design a more effective learning pipeline under such a low-resource scenario and build a better text-guided voice conversion model. Moreover, the concurrent works leverage either RL or human feedback to refine their models, whereas we adopt both and benefit the proposed model. We study how reinforcement learning with human feedback (RLHF) helps speech generation and how to apply it, which has yet to be discussed before in speech or audio generation.

5.3 Text-Guided Voice Conversion

In text-guided voice conversion, the model takes source speech, a text description of the target style, and a target speaker embedding as input to generate converted speech. The converted speech retains the linguistic content of the source speech, while its speech style and speaker identity are modified according to the text description and speaker embedding. The speech styles can be different speaking rates, speaking methods, or emotions, and the text descriptions are human-written sentences in any format. The training data for building a text-guided voice conversion model consists of speech utterances in diverse styles paired with corresponding text descriptions, and text transcriptions of the speech content are not required for this task.

This section introduces how we build the proposed text-guided voice conversion

model based on a pre-trained model. Section 5.4 further studies improving model performance with reinforcement learning and human feedback. Lastly, the training data will be detailed in Section 5.5.

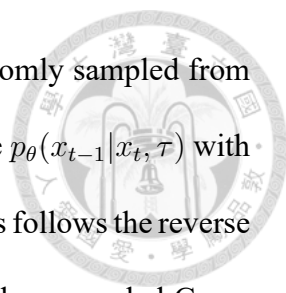


5.3.1 Modifying a Pre-Trained Model for Text-Guided Voice Conversion

We utilize a pre-trained text-to-audio (TTA) model as the backbone of the proposed model. In TTA, there is plenty of public data [167] available, and many open-source models are built with remarkable performance in comprehending text sentences and producing sound events [147, 148, 150]. Hence, we take advantage of their text understanding and sound generation abilities in our text-guided voice conversion model. While TTA models conditioned on textual descriptions can mimic human-like sounds, they lack the capability to generate meaningful speech content. To apply a TTA model in text-guided voice conversion, we need to adapt its architecture to control the output speech content. In the following paragraphs, we first introduce the TTA model we adopt, TANGO [148]. Then, we detail the proposed architecture for text-guided voice conversion.

TANGO

TANGO is a TTA model employing an instruction-tuned large language model, FLAN-T5 [149], to comprehend and encode input text, followed by a latent diffusion model (LDM) [143]. The LDM iteratively generates sounds from x_T to x_0 , where T is the number of generation steps, x_T is random Gaussian noise, and x_0 is the latent representation of the output sound. During training, x_0 is derived by encoding the target sound using a pre-trained VAE encoder [147]. $x_T, x_{T-1}, x_{T-2}, \dots, x_1$ are then calculated through



the forward diffusion process [168]. At each training step, x_t is randomly sampled from $\{x_T, x_{T-1}, x_{T-2}, \dots, x_1\}$ as input, and the model is trained to optimize $p_\theta(x_{t-1}|x_t, \tau)$ with encoded text embeddings τ as extra conditions. The generation process follows the reverse diffusion process [168] to generate x_t one-by-one, starting from random sampled Gaussian noise x_T . After the iterative generation process, the final output x_0 is passed to the pre-trained VAE decoder and a HiFi-GAN [7] vocoder to reconstruct Mel-spectrograms and waveforms, respectively. The original TANGO was built for general-purpose TTA generation. It can produce sounds consistent with the scenarios described in the text, but it cannot precisely control the style and content of the output speech. Hence, we modify the architecture to take a speech utterance as additional input, which provides detailed content information.

Proposed Model

Figure 5.1 depicts an overview of the proposed model modified from pre-trained TANGO. We first convert the source speech, which specifies the content of the output speech, into HuBERT units [13]. These discrete representations have shown a good ability to remove speaker information and retain content information [30], allowing the model to specify different target speakers for the output speech through additional speaker embeddings. The HuBERT units specifying the speech content, the speaker embedding of the target speaker, and the text prompt describing the target style are encoded by a unit encoder. The output unit embeddings ϕ are concatenated with the LDM input, which is the intermediate (x_t) in the reverse diffusion process, and the text embeddings τ provide style information through the cross-attention mechanism [148]. We use a pre-trained d-vector [169] model ¹

¹<https://github.com/yistLin/dvector>

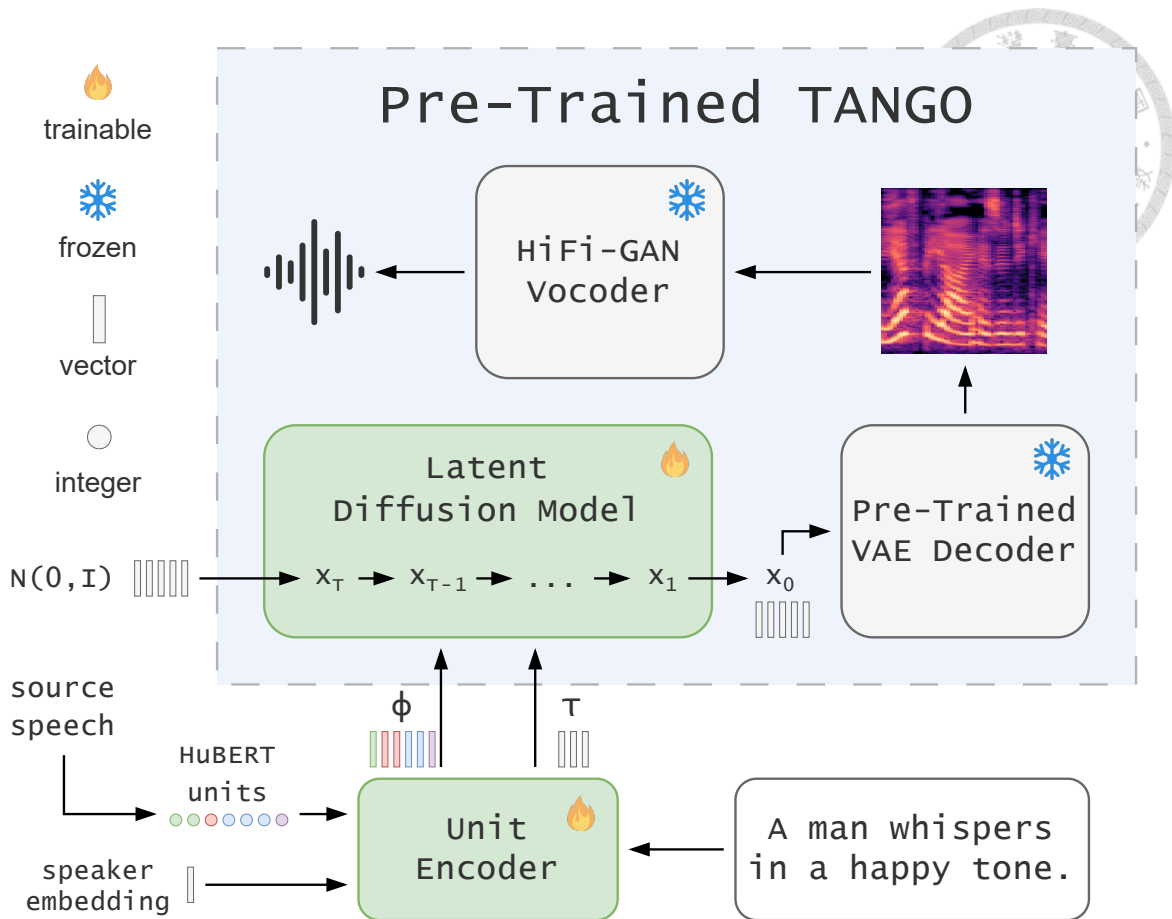


Figure 5.1: Overview of modifying a pre-trained text-to-audio model for text-guided voice conversion.

to extract speaker embeddings. At each training step, we randomly select five utterances of the same speaker but with different styles to extract the average speaker embedding, making the embeddings contain less information about a specific speech style and urging the model to control the output styles following text descriptions. We follow the same training process except for the augmentation trick in the original TANGO paper [148] to optimize the LDM. The loss function is denoted as L_{ldm} . All input conditions are from the same utterance during training and can be specified separately at inference time.

The unit encoder is illustrated in Figure 5.2 (a), primarily composed of CNN and bidirectional LSTM layers. Following TANGO, we use pre-trained FLAN-T5-LARGE [149] to encode the input text, combined with the speaker embedding as the output τ . We re-

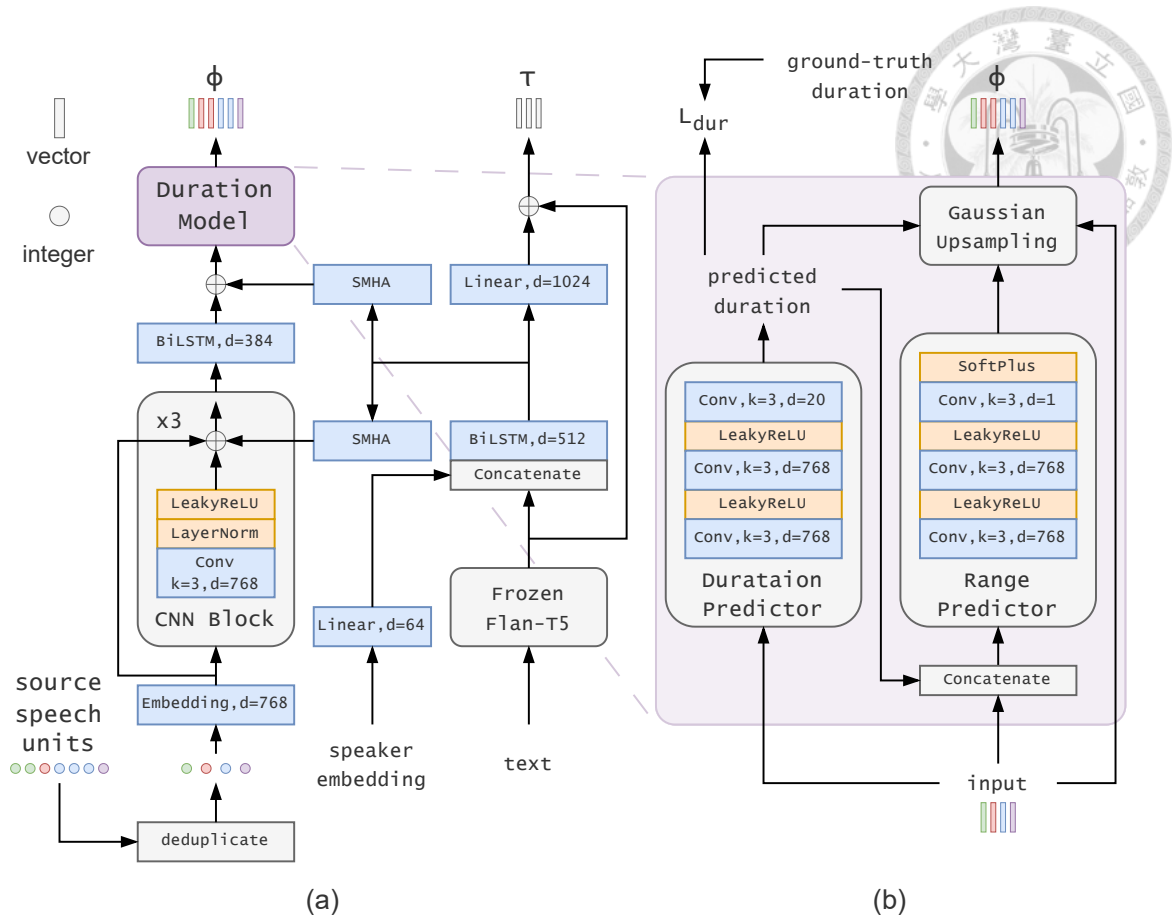


Figure 5.2: (a) Architecture of the unit encoder. (b) Architecture of the duration model. Note that ground-truth instead of predicted duration is used as input to the range predictor and the Gaussian upsampling module at training time.

move the repetitions in the unit sequence to eliminate prosody-related and speed-related information in the source speech. The information will be re-specified by a duration model according to the text description and speaker embedding. We apply a self multi-head attention pooling (SMHA) [170] layer to extract a vector representing style and speaker information. The duration model is illustrated in Figure 5.2 (b). We follow [171] to construct a duration predictor, a range predictor, and a differentiable Gaussian upsampling module. During training, we use ground-truth duration as input to the range predictor and the Gaussian upsampling module. For the duration predictor, we clip the minima and the maxima of the target duration to 1 and 20, respectively, making it a 20-class classification model trained to minimize the cross-entropy loss L_{dur} . Finally, all modules are jointly

trained to minimize the loss $L_{total} = L_{ldm} + L_{dur}$.



5.3.2 Duration Model Fine-Tuning

During the joint training process, we found that the duration and diffusion models have different convergence speeds. Therefore, after the joint training, we freeze the model and separately fine-tune the duration predictor in the duration model and the SMHA pooling layer before it. Besides, we add a new GRU and linear layers upon the CNN block in the duration predictor. The modified duration predictor predicts duration autoregressively. The input and output of the CNN block and the duration of the previous time step are concatenated together as the input of the GRU to predict the next duration. During training, ground truth duration is used as input in a teach-forcing manner, and the SMHA pooling layer, the CNN block, and the GRU layer are updated to minimize L_{dur} .

5.4 Improving Model Performance with Reinforcement Learning and Human Feedback

The proposed model built in Section 5.3 has a preliminary voice conversion ability to produce utterances of the target speaker with the specified style and the exact content of the source speech. In the previous training stage, the training data is a combination of publicly available datasets, and each dataset contains different aspects of speech style, such as accent or emotion. During training, each input description only specifies styles from one dataset. The model has never seen a description simultaneously specifying multiple styles from different datasets. Hence, in this section, we explore leveraging reinforcement learning (RL) to improve the model performance when using more complex style descriptions,

making it more effectively benefit from the diverse datasets with various kinds of styles. The datasets used in this work will be introduced in Section 5.5. Note that in the following, we combine two aspects of speech styles to generate complex style descriptions, but the proposed method can be generalized to cases with more styles.

We select different styles from different datasets to combine and generate their style descriptions. Specifically, we select three speaking styles (speak, shout, whisper) and eight emotions (neutral, happy, angry, sad, disgust, sleepiness, surprise, fear), combining them into 24 compound styles. These combinations simultaneously specify multiple styles from different datasets, which are unseen during the previous training. Besides, due to the lack of real speech data of these compound styles, we cannot directly apply the supervised joint training process mentioned in Section 5.3. Consequently, our model so far cannot convert speech into such complex styles well. To address this problem, we introduce reinforcement learning for this scenario. For simplicity, during training, we fix the descriptions for these 24 styles in the form of “*A man/woman <speaking style> in a/an <emotion style> tone.*”² We will show in multiple experiments that the model can still generalize well when using more diverse forms of descriptions³ during inference.

5.4.1 Reward Model

We first build a reward model for the subsequent reinforcement learning. Given a text description x_{text} and a speech utterance x_{speech} , the reward model RM aims to score their relevance, i.e., how consistent the speech style is with what the text describes. The reward model consists of a text encoder E_{text} and a speech encoder E_{speech} . The text encoder

²E.g., “*A man whispers in a happy tone.*”

³E.g., “*Listen to the man as he communicates his joy in a whisper.*”

includes a pre-trained and frozen FLAN-T5-LARGE [149], followed by a bidirectional LSTM layer, a linear layer, and an average pooling layer to output a vector v_{text} representing the global information. The speech encoder has the same architecture except that FLAN-T5-LARGE is replaced by WavLM Base+ [95], and the output is denoted as v_{speech} . The reward model outputs the cosine similarity between v_{text} and v_{speech} to score their relevance. The scoring process is formulated as

$$RM(x_{text}, x_{speech}) = \text{cosine_similarity}(v_{text}, v_{speech}), \quad (5.1)$$

$$v_{text} = E_{text}(x_{text}), \quad (5.2)$$

$$v_{speech} = E_{speech}(x_{speech}). \quad (5.3)$$

Following CLIP [142] and CLAP [172], we optimize the reward model through contrastive learning. At each training step, we randomly sample a mini-batch of N speech utterances and the corresponding text descriptions from the same dataset⁴ for encoding. Next, we calculate the similarities between v_{text} and v_{speech} vectors, resulting in a similarity matrix $C \in R^{N \times N}$. The similarities on the diagonal of C represent the relevances between the paired text and speech samples in a mini-batch. The loss function is formulated as

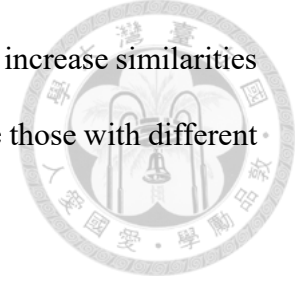
$$L_{cl} = 0.5 \times (l_{text}(C) + l_{speech}(C)), \quad (5.4)$$

$$l = \frac{1}{N} \sum_{i=1}^N \log \text{diag}_i(\text{softmax}(C \times \rho)), \quad (5.5)$$

where ρ is a learnable temperature scalar to scale C and diag_i is the i th element on the diagonal. l_{text} and l_{speech} indicate that the softmax is operated along the text and speech

⁴Section 5.5 introduces the seven datasets used in this work, and speech utterances in a mini-batch are from the same dataset.

axes of C , respectively. Eventually, the reward model is optimized to increase similarities between text and speech encodings with the same styles and decrease those with different styles.



After training the reward model, given a speech utterance generated by the voice conversion model with one of the 24 style descriptions, we first follow Eq. 5.1 to calculate similarities between the utterance and the 24 style descriptions. Then, we apply softmax to the 24 similarities. Among the resulting 24 values, we define the reward as the one corresponding to the target style of the output utterance, indicating the relevance between the generated speech and the input text description.

5.4.2 Denoising Diffusion Policy Optimization

Figure 5.3 (a) shows the overview of the proposed RL process. Our RL algorithm follows denoising diffusion policy optimization (DDPO) [173], which investigates the application of reinforcement learning to diffusion models in image generation. Given an input c (containing source speech, a text description, and a target speaker) and the trained voice conversion model with parameters θ , the diffusion model outputs x_0 conditioned on c . We use the reward model to calculate the reward and score the relevance between the generated speech and the text description in speech style. The RL objective is to update θ to maximize the reward, which is formulated as

$$J(\theta) = \mathbb{E}_{x_0 \sim p_\theta(x_0|c)}[r(x_0, c_{text})], \quad (5.6)$$

where c_{text} is the text description in c , r is the process illustrated in the last paragraph of Section 5.4.1 to calculate rewards, and $p_\theta(x_0|c)$ can be derived during the reverse diffusion

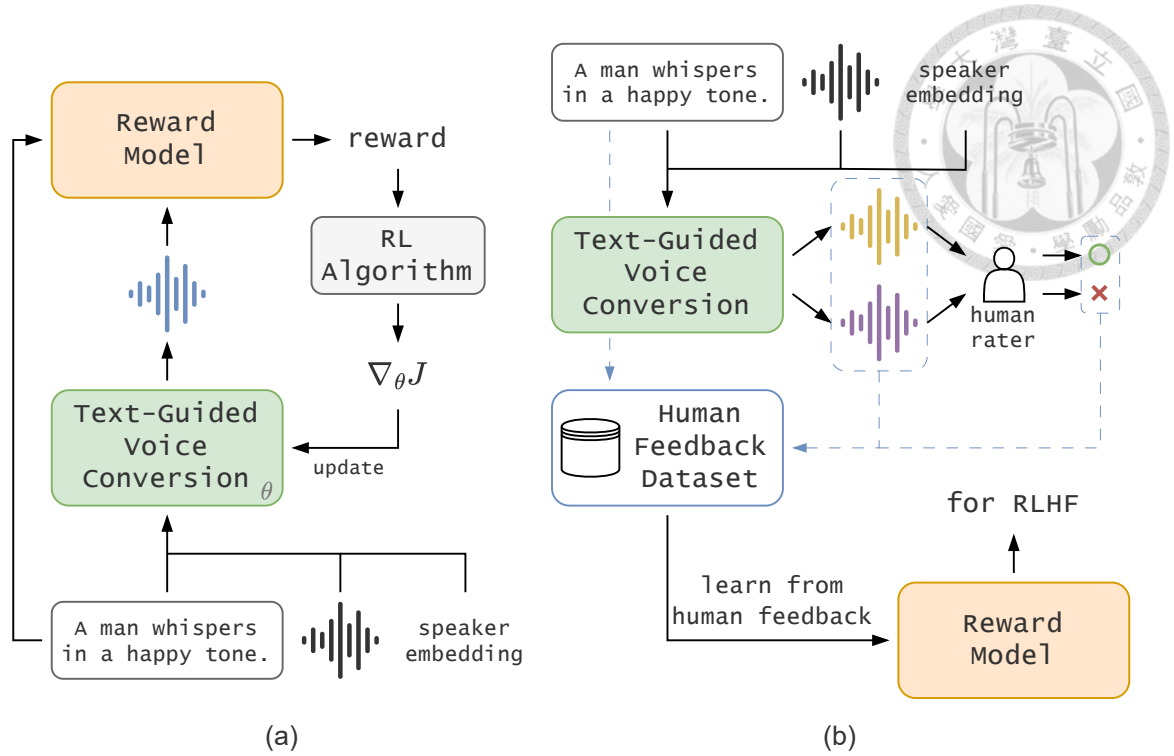


Figure 5.3: (a) Overview of the RL pipeline. The text-guided voice conversion model is optimized with the reward model and RL algorithm. (b) Overview of the RLHF process, including collecting human preferences and building a reward model with human feedback. The reward model learns from human feedback and is used to optimize the text-guided voice conversion model with the same RL algorithm as in (a).

process. Note that x_0 is converted to speech through the VAE decoder and HifiGAN vocoder before being scored through r . We omit this process for simplicity.

We update θ through estimating $\nabla_{\theta} J$. At training step k , we use the model parameterized with θ_k to generate m samples, each of which is a denoising trajectory $\{x_T, x_{T-1}, \dots, x_0\}$ from the reverse diffusion process. These trajectories and the conditions to generate them are kept to update model parameters. We use the same samples to update the model for n steps and generate new samples at step $k + n$. At step $k + i$, the gradient can be estimated with importance sampling [174]:

$$\nabla_{\theta_{k+i}} J = \mathbb{E} \left[\sum_{t=1}^T \frac{p_{\theta_{k+i}}(x_{t-1}|x_t, c)}{p_{\theta_k}(x_{t-1}|x_t, c)} \nabla_{\theta_{k+i}} \log p_{\theta_{k+i}}(x_{t-1}|x_t, c) r(x_0, c_{text}) \right], \quad (5.7)$$

where each x_t is an element in the trajectory generated by θ_k . The practical implementation follows DDPO and proximal policy optimization (PPO) [175], which apply trust region [176] by clipping.

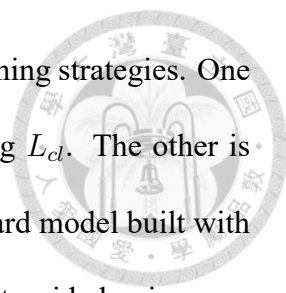


5.4.3 Reinforcement Learning from Human Feedback

We further adopt a more advanced RL algorithm for speech generation, reinforcement learning from human feedback (RLHF) [137, 138], which introduces real-world human feedback to the learning paradigm of RL. The proposed RLHF procedure is modified from [139], including three steps: (1) collecting human preferences for generated speech, (2) building a reward model with human feedback, and (3) applying DDPO with the new reward model. The overview of the RLHF process is illustrated in Figure 5.3 (b).

We first prepare input samples from the training data to generate speech. Each input sample contains randomly selected source speech, a style description, and a target speaker. For each input sample, we use the voice conversion model trained with DDPO to generate two speech utterances. Then, we recruit human raters. Given a description and two converted utterances, the raters are asked to label the utterance that has a style more relevant to the description. The collected human feedback data is denoted as D_{hf} . Each sample in D_{hf} is a set of $(x_{text}, x_{speech}^w, x_{speech}^l)$, where x_{text} is the text description and x_{speech}^w is the preferred speech utterance compared with x_{speech}^l . Following [139], the reward model is optimized to score the output speech in a way more aligned with human preferences. The loss function is formulated as

$$L_{hf} = -\mathbb{E}_{(x_{text}, x_{speech}^w, x_{speech}^l) \sim D_{hf}} [\log(\sigma(RM(x_{text}, x_{speech}^w) - RM(x_{text}, x_{speech}^l)))], \quad (5.8)$$



where $RM(\cdot)$ is calculated following Eq. 5.1. We investigate two training strategies. One is optimizing L_{hf} only and fine-tuning a reward model trained using L_{cl} . The other is jointly optimizing $L_{cl} + L_{hf}$ from scratch. Finally, we adopt the reward model built with human feedback and follow the DDPO algorithm to optimize the text-guided voice conversion model.

5.5 Datasets

Developing a text-guided voice conversion model that can comprehend complex text descriptions and generate speech with diverse styles requires a large amount of high-quality, expressive speech recordings and human-written text descriptions. Previous works involve professionals for recording and annotating, expensively building proprietary datasets specifically for text-guided speech generation. In contrast, this study aims to accomplish the task by leveraging only publicly available datasets. We extensively collect seven speech datasets for various tasks, forming a collection with over 200 hours of speech data. We group these datasets into four main categories for the following introduction. Details of the datasets are listed in Table 5.1.

5.5.1 PromptSpeech Dataset

PromptSpeech [8] was initially proposed for text-guided text-to-speech generation and consists of rich speech data and corresponding style descriptions. It is composed of two subsets. One is derived from the LibriTTS [79] dataset, denoted as PromptSpeech-R. Each speech utterance has a sentence describing the styles in four aspects (gender, pitch, speed,

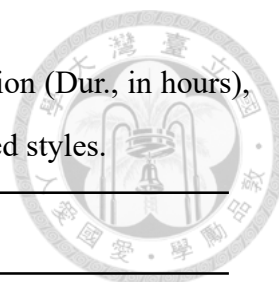


Table 5.1: Details of datasets used in this work, including total duration (Dur., in hours), number of utterances (Utrr.), number of speakers (Spk.), and contained styles.

Name	Dur.	Utrr.	Spk.	Styles
<i>PromptSpeech Dataset [8]</i>				
PromptSpeech-R	38	27k	1191	gender, pitch, speed, volume
PromptSpeech-S	97	50k	4	gender, pitch, speed, volume, 5 emotions
<i>Emotion Datasets</i>				
EmoV-DB [177]	9	7k	4	gender, 5 emotions
ESD [178]	13	18k	10	gender, 5 emotions
CREMA-D [179]	5	7k	91	gender, 6 emotions, 4 emotion levels
<i>Accent Dataset</i>				
VCTK [124]	41	44k	110	gender, 12 accents
<i>Sound Event Dataset</i>				
AudioCaps [167]	13	5k		sound events

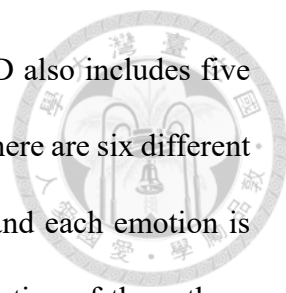
volume). The other subset, denoted as PromptSpeech-S, is synthesized by Azure TTS⁵ and annotated with text descriptions of styles in five aspects (general, shout, whisper, cheerful, sad). For the synthesized subset, We use the same 50k utterances as in [11]. We only use the officially released training splits of the two subsets and exclude 100 utterances from each as the validation sets.

5.5.2 Emotion Datasets

We collect three emotion datasets, EmoV-DB [177], ESD [178], and CREMA-D [179], which contain various emotion styles and expressive speech data. EmoV-DB includes five

⁵<https://azure.microsoft.com/en-us/products/ai-services/text-to-speech/>

#overview



emotion styles (neutral, amused, anger, sleepiness, disgust), and ESD also includes five emotions (neutral, happy, angry, sad, surprise) with some overlaps. There are six different emotions (neutral, happy, anger, sad, disgust, fear) in CREMA-D, and each emotion is presented at four different levels (xx, low, mid, high). The combination of these three datasets includes totally eight unique emotion styles (neutral, happy, angry, sad, disgust, sleepiness, surprise, fear). We exclude 90, 100, and 100 utterances from EmoV-DB, ESD, and CREMA-D, respectively, as the validation sets.

The speech data in these datasets are only labeled with emotion tags, while building a text-guided voice conversion model requires sentences describing target speech styles. Hence, we use a large language model (LLM) to convert labels into text descriptions. Specifically, we provide the gender and emotion information and ask ChatGPT ⁶ to produce different text prompts describing the same given styles. For EmoV-DB and ESD, we generate 20 descriptions per gender-emotion pair. For CREMA-D, we generate four descriptions per gender-emotion-level pair.

5.5.3 Accent Dataset

VCTK [124] is a high-quality dataset containing recordings of 110 speakers with 12 different accents. Similarly, we use ChatGPT to generate text descriptions based on gender and accent information. We generate eight descriptions per gender-accent pair. We split 110 utterances (one per speaker) as the validation set. Note that the use of VCTK only aims to increase the data diversity during training, i.e., providing more speakers with different accents. Although we prepare text prompts describing accents as model input, the accent of the output speech is primarily affected by the target speaker embedding that we

⁶<https://chat.openai.com/>

additionally provide. Section 5.3 illustrates more details about speaker embedding.



5.5.4 Sound Event Dataset

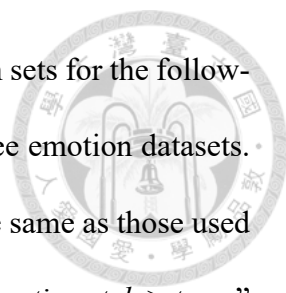
AudioCaps [167] is derived from AudioSet [180], a large-scale dataset for sound event classification. Audio clips in AudioCaps are annotated with a sentence describing the content of the sound. Since the audio data is collected from YouTube with extensive topics and scenarios, the vocabulary of the text descriptions is much richer than other datasets mentioned above. We use the subsets annotated as human speech and exclude 100 utterances as the validation set. Similar to the accent dataset, AudioCaps is used only to augment the diversity of text descriptions and speech styles during training. In our setting, the content of the converted speech is decided by the input source speech instead of the text description.

5.6 Experimental Setup

5.6.1 Data Preprocessing

We use all seven datasets introduced in Section 5.5 for the joint training, the duration model fine-tuning, and the contrastive learning of the reward model. For RL and RLHF, due to the small total training steps, we only use the three emotion datasets. All speech utterances are resampled at 16 kHz. Speech units are extracted using the official HuBERT implementation and the pre-trained HuBERT Base model.⁷ We extract HuBERT features from the 9th layer and run k-means clustering on our speech data with $k = 100$.

⁷<https://github.com/facebookresearch/fairseq/tree/main/examples/hubert>



We prepare in-domain (ID) and out-of-domain (OOD) validation sets for the following experiments. The ID set consists of the validation sets of the three emotion datasets. All speakers are seen during training, and the text descriptions are the same as those used in RL, i.e., in the form of “*A man/woman <speaking style> in a/an <emotion style> tone.*” The OOD set contains speech utterances randomly selected from the *train-clean* split of LibriTTS [79]. These utterances are not included in PromptSpeech [8], and all speakers are unseen. Besides, we use ChatGPT to randomly generate unseen text descriptions given different genders, speaking styles, and emotions. For both validation sets, we randomly generate (*source speech, target speaker, style description*) combinations as model input conditions. Elements in a combination can be from different ground truth utterances. The ID and OOD validation sets contain 290 and 177 combinations, respectively.

For RLHF, We follow the process introduced in Section 5.4 to use the voice conversion model trained with vanilla RL to generate utterances and recruit raters to label them. The input units, text descriptions, and speaker embeddings are randomly selected from different utterances of the three emotion datasets. A total of 12600 samples are collected, of which 12000 are for training and 600 for validation.

5.6.2 Models and Training

Figure 5.1 and Figure 5.2 show the details of the proposed model, including channel sizes and kernel sizes of different layers. The pre-trained TANGO model is from the official GitHub repository.⁸ We use the *tango-full-ft-audiocaps* checkpoint, and the pre-trained VAE and HifiGAN vocoder are adopted from [147]. The joint training process follows the setting in [148], except that we do not apply the data augmentation method proposed in

⁸<https://github.com/declare-lab/tango>

the original paper. We use a per GPU batch size of 2 with 4 gradient accumulation steps. We train the model on one V100 GPU for 10 epochs, resulting in an effective batch size of $2(\text{sample}) \times 4(\text{accumulation}) \times 1(\text{GPU}) = 8$.



After the joint training, we freeze the model and fine-tune only the duration predictor and the SMHA pooling layer with the cross-entropy loss for another 10 epochs. The effective batch size and the learning rate are set to 64 and $3e-4$, respectively. The remaining settings are the same as the joint training. Since works about text-guided voice conversion and aiming to convert both speaking styles and emotions are limited, we use this fine-tuned model as the baseline model in this work. The diffusion model uses DDPM [168] with 1000 diffusion steps at training time. We apply DDIM [181] with 100 steps at inference time and with 30 steps in reinforcement learning for better efficiency.

For RL and RLHF, we apply LoRA [182] with a rank of 4 to the attention layers in the diffusion model and update only the LoRA layers. The remaining parameters are kept frozen. We found this stabilized the RL process compared with updating the whole model. We set $m = 128$ and $n = 4$ for DDPO, i.e., we generate 128 outputs and then use them to update the model for 4 steps with an effective batch size of 32. To avoid out-of-memory, we use a batch size of 1 and 32 gradient accumulation steps in our implementation. The training is done on one NVIDIA V100 GPU with the AdamW optimizer [183] and a fixed learning rate of $3e-4$ for 2000 steps. We found that training for too many steps leads to worse quality and distortions in output speech, as reported in [173]. We keep the checkpoint with the best validation loss as the final model.

The bidirectional LSTM and linear layers in the reward model have 256 and 512 channels, respectively. For contrastive learning, we train the reward model on the collected

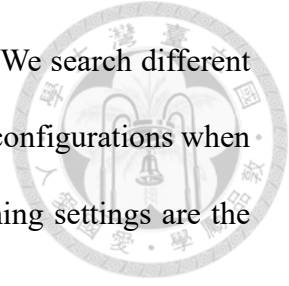
Table 5.2: Configurations to train the reward model with human feedback. **Size**: size of the human feedback dataset. **lr.**: learning rate. **bsz.**: batch size to sample data from the human feedback dataset. **Start Step**: step to start updating the model with L_{hf} . **Update Steps**: every how many steps to update the model with L_{hf} . **Max Steps**: total training steps. **Best Step**: step of the best checkpoint.

Label	Size	lr.	bsz.	Start Step	Update Steps	Max Steps	Best Step
<i>Fine-tuning</i>							
ft-4k	4k	5e-5	2	1	1	8k	3.1k
ft-8k	8k	3e-4	1	1	1	8k	6.3k
ft-12k	12k	1e-4	2	1	1	8k	6.1k
<i>Training from scratch</i>							
tfs-4k	4k	3e-4	2	25001	5	50k	40k
tfs-8k	8k	3e-4	1	1	5	50k	40k
tfs-12k	12k	3e-4	1	1	4	50k	45k

seven datasets. We set the batch size to 8 and gradient accumulation steps to 4, resulting in an effective batch size of 32. As mentioned in 5.4, the eight samples in each mini-batch are randomly selected from the same dataset. We use the Adam optimizer [106] with a learning rate of 3e-4 and a cosine learning rate scheduler with 5000 warm-up steps. The model is trained for 50k steps on one 2080Ti GPU, and we keep the best checkpoint at step 45k as the reward model for vanilla RL.

For RLHF, we try two training strategies to leverage the human feedback dataset: fine-tuning and training from scratch. For fine-tuning, we directly fine-tune the trained reward model with the objective L_{hf} using the Adam optimizer and a fixed learning rate. For training from scratch, we train a randomly initialized model with the objective $L_{cl} + L_{hf}$. To study the impact of data size on RLHF, we divide the human feedback dataset

into three splits with 4k, 8k, and 12k training samples, respectively. We search different training parameters to ensure the best results. Table 5.2 lists the best configurations when training with different strategies and on different splits. The remaining settings are the same as those in contrastive learning.



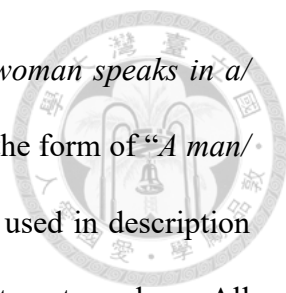
5.6.3 Evaluation Tasks

We use objective and subjective evaluations to assess the ability to convert styles and show the benefits of applying RL and RLHF.

Objective Evaluation

We conduct the objective evaluation on both ID and OOD validation sets. For each input condition in the validation sets, we generate three speech utterances and evaluate them in the following tasks:

- **Style Classification with the Reward Model** We use 3-class speaking style classification (SS), 8-class emotion classification (Emo.), and 24-class description classification (Des.) tasks to evaluate converted speech. Following CLIP [142], we use our reward model to classify speech utterances. The reward model is the same as in RL, trained with contrastive learning and without human feedback data. For speaking style classification, we extract encodings of the generated speech and three different style descriptions in the form of “*A man/woman <speaking style> tone.*” Then we calculate the cosine similarity between the speech and text encodings. The pair with the highest similarity indicates that the generated speech is classified to the corresponding speaking style. Emotion classification follows the same process,



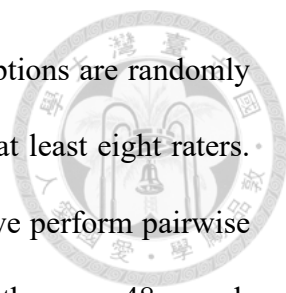
except that the style descriptions are in the form of “*A man/woman speaks in a/an <emotion style> tone.*”. Similarly, 24 style descriptions in the form of “*A man/woman <speaking style> in a/an <emotion style> tone.*” are used in description classification. Genders in all descriptions are aligned with the target speakers. All classification results are reported in accuracy (%).

- **Style Classification with Style Classifiers** For better reference in the SS and Emo. tasks, we train a 3-class speaking style classifier and an 8-class emotion classifier to evaluate converted speech.⁹ The architectures are the same as the speech encoder in the reward model. We only modify the channel sizes of the last linear layers and optimize these classifiers with the cross-entropy loss. All classification results are reported in accuracy (%).
- **Speaker Similarity (SSIM)** To assess the speaker similarity between the generated speech and the input speech specifying the target speaker, we use the d-vector model to extract the speaker embeddings of the two utterances. Then, we calculate their cosine similarity. The result ranges from -1 to 1, and a higher score indicates a better speaker similarity.

Subjective Evaluation

Following [147], [148], and [150], we ask human raters to assess speech utterances on a 5-point scale in two aspects: overall audio quality (OVL) and relevance to the style descriptions (REL). For each voice conversion model, we generate 48 speech utterances with different target style combinations (two genders, three speaking styles, and eight

⁹For the Des. task, it is infeasible to build a 24-class classifier with supervised learning due to the lack of paired data, as mentioned in Section 5.4.



emotion styles). The source speech, target speakers, and text descriptions are randomly selected from the OOD validation set. Each utterance is scored by at least eight raters. To better demonstrate the benefits of introducing human feedback, we perform pairwise preference tests between models trained with RL and RLHF. We use the same 48 speech utterances in the tests. In each test, two utterances, which are generated using the same input conditions but different models, are provided to human raters. The raters are asked to decide which utterance is more relevant to the input style description. Each pair is scored by at least 15 raters. All raters in the subjective evaluation are recruited using Amazon Mechanical Turk.

5.7 Results

5.7.1 Objective Evaluation

Performances of the Evaluation Models

We first evaluate ground-truth speech data in the objective evaluation tasks and show the performances of the evaluation models. For emotion classification, since the classifier has an 8-class output while each emotion dataset contains only five to six emotion styles, we ignore those irrelevant classes when calculating accuracies. For SSIM, we first calculate the average speaker embedding of a speaker. Then, we calculate the similarity between the speaker embedding of each utterance and the average embedding. The results are listed in Table 5.3. We conclude our observations as follows:

- The reward model reaches accuracies of over 97% on PromptSpeech-S, EmoV-DB, and ESD, indicating its ability to discriminate different speaking styles and

Table 5.3: Style classification and speaker similarity results on different datasets.

Dataset	# of Styles	Classification Acc.		SSIM
		Reward Model	Classifier	
PromptSpeech-S	3	98.3	49.2	0.761
EmoV-DB	5	98.9	53.3	0.622
ESD	5	97.0	55.0	0.704
CREMA-D	6	77.0	52.0	0.885

emotions. Although the accuracy drops on CREMA-D, we attribute the degradation to the lower recording quality and more noisy speech data in this dataset.

- We found that the style classifiers easily overfit on the training set, making parameter searching and fewer training steps crucial to reaching optimal results. However, the reward model still performs much better than the style classifiers, showing the effectiveness of contrastive learning on the collected datasets with various styles.
- All speaker similarities are over 0.6. PromptSpeech-S consists of clean synthesized speech, resulting in a higher average similarity of 0.761. As for CREMA-D, there are more speakers, and some have only one utterance in the validation set, leading to multiple speaker similarities of 1 and the highest average SSIM.

Classification Results of Generated Speech

Table 5.4 lists the classification results of the proposed models before and after applying reinforcement learning. **Base** denotes the proposed model after the joint training and duration model fine-tuning. The default batch size is set to 8 in the joint training. We study the impact of the batch size by setting a larger batch size of 32, denoted as **b32**. **Base-**

Table 5.4: Style classification results of the proposed model. **Base** denotes the model after the joint training and duration model fine-tuning. **b32** uses a batch size of 32. **Base-RL** applies RL and only updates the LoRA layers in the diffusion model. **+ue** and **+dur.** further apply LoRA to the SMHA pooling layers for the unit encoder and the duration model, respectively.

Model	Reward Model			Classifier	
	SS	Emo.	Des.	SS	Emo.
<i>ID validation set</i>					
Base	40.9	27.7	10.5	38.7	20.6
Base (b32)	39.5	27.8	12.2	33.9	20.9
Base-RL	44.0	46.8	21.5	41.6	29.9
Base-RL (+ue)	42.3	45.3	18.2	38.6	31.3
Base-RL (+ue, +dur.)	39.8	58.5	26.2	38.3	36.9
<i>OOD validation set</i>					
Base	42.4	37.3	16.6	34.8	27.1
Base (b32)	41.2	37.3	18.1	34.5	26.2
Base-RL	49.9	51.2	24.7	40.1	29.0
Base-RL (+ue)	47.6	52.5	26.6	36.0	28.1
Base-RL (+ue, +dur.)	45.8	60.8	30.5	36.2	33.5

RL is the model with RL. We also show the effects of tuning more parameters in RL by applying LoRA to the SMHA pooling layers for the unit encoder and the duration model.

Our observations are concluded as follows:

- The results evaluated with the reward model on the ID validation set show that the proposed model possesses a preliminary ability to convert speaking styles and emotions. The model reaches 40.9%, 27.7%, and 10.5% in SS, Emo., and Des.,

respectively.¹⁰

- Using a larger batch size of 32 does not lead to significant improvements but requires four times more computational resources. Hence, we set the default batch size to 8 in this work.
- Applying RL shows better classification results, implying that the converted speech is much more expressive and aligned with the target styles. The improvements are especially significant in the Emo. and Des. tasks. The accuracies relatively increase by approximately 70% and 100%. Trained on the same amount of data, the model greatly improves after RL, indicating that the proposed learning paradigm helps build a text-guided voice conversion model more efficiently and effectively.
- Tuning more parameters in RL does not lead to much better results. Although **Base-RL (+ue, +dur.)** shows higher accuracies in Emo. and Des., we found the converted speech has a longer duration, and the speaking rate is unnaturally slower. The duration predictor of this model is negatively affected after RL. We also tried optimizing all model parameters in RL. However, the training became less stable, and the loss exploded easily.
- The results evaluated with the style classifiers show lower accuracies, which can be attributed to the worse performances of the classifiers, as shown in Table 5.3. Besides, since speech utterances in the OOD validation set are from LibriTTS, which are cleaner and high-quality, the results on this validation set are slightly better. Despite the discrepancies between validation sets or evaluation models, all results demonstrate a similar tendency as the above observations.

¹⁰The accuracies of random guesses in SS, Emo., and Des. are 33.3%, 12.5%, and 4.2%, respectively.

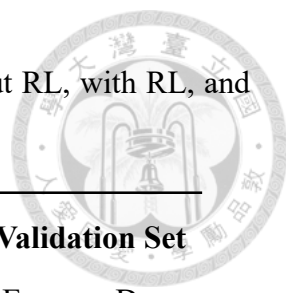
Table 5.5: Classification results of the reward models trained or fine-tuned with different amounts of human feedback. Human Feedback is a validation set separated from the collected human feedback data. **orig.** denotes the original reward model built without human feedback data. Details of other validation sets and reward models can be found in Section 5.5 and Table 5.2, respectively.

		Reward Model						
		orig.	ft-4k	ft-8k	ft-12k	tfs-4k	tfs-8k	tfs-12k
Validation Set	Human Feedback	52.5	61.7	65.3	66.0	56.0	57.3	61.0
	PromptSpeech-S	98.3	91.5	52.5	83.1	98.3	100	100
	EmoV-DB	98.9	94.4	37.8	86.7	100	97.8	98.9
	ESD	97.0	77.0	41.0	62.0	96.0	97.0	97.0
	CREMA-D	77.0	59.0	17.0	55.0	78.0	71.0	78.0

Performances of Reward Models with Human Feedback

Table 5.5 lists the performances of the reward models when training with different strategies and amounts of human feedback. We found that fine-tuning leads to the best result for predicting human feedback. However, it also introduces performance degradation in style classification tasks, especially for **tfs-4k**, which is trained for more steps with a larger learning rate. On the contrary, training from scratch by jointly optimizing L_{cl} and L_{hf} shows lower accuracies in predicting human feedback but maintains good performances in other tasks. Based on the results, we select three reward models, **ft-12k**, **tfs-4k**, and **tfs-12k**, for RLHF in the following experiments to study the impact of different training strategies and data sizes.

Table 5.6: Style classification results of the proposed models without RL, with RL, and with RLHF.



Model	Reward Model	ID Validation Set			OOD Validation Set		
		SS	Emo.	Des.	SS	Emo.	Des.
Base	-	40.9	27.7	10.5	42.4	37.3	16.6
Base-RL	orig.	44.0	46.8	21.5	49.9	51.2	24.7
Base-RLHF-v1	ft-12k	34.4	37.7	13.4	36.9	49.2	18.1
Base-RLHF-v2	tfs-4k	43.2	46.2	19.5	51.6	48.8	24.7
Base-RLHF-v3	tfs-12k	42.0	51.8	21.5	45.0	53.1	26.0

Classification Results of Applying RLHF

Table 5.6 shows the style classification results of the proposed models with RLHF. Similar to applying RL, applying RLHF greatly improves the proposed models, outperforming **Base** by up to 100% relatively. We found that fine-tuning the reward model leads to less performant results, which reflect the worse ability of **ft-12k** in style classification shown in Table 5.5. On the contrary, the reward models trained from scratch with human feedback are comparable to the original one, and therefore, **Base-RLHF-v2** and **Base-RLHF-v3** perform similarly to **Base-RL**. We hence conclude that the objective evaluation results are mainly affected by the style classification abilities of the reward models. We will demonstrate how human feedback helps model performance in the subjective evaluation.

SSIM Results of Generated Speech

Table 5.7 shows the SSIM results in speaker conversion. For better reference, we include results of YourTTS [184], a popular text-to-speech and voice conversion model. Note that YourTTS does not support converting speaking styles or emotions. We only use it

Table 5.7: Speaker similarity results of the proposed models without RL, with RL, and with RLHF.

Model	Reward Model	SSIM (ID)	SSIM (OOD)
Base	-	0.389	0.513
Base-RL	orig.	0.354	0.485
Base-RLHF-v1	ft-12k	0.372	0.473
Base-RLHF-v2	tfs-4k	0.323	0.461
Base-RLHF-v3	tfs-12k	0.329	0.443
YourTTS [184]	-	0.355	0.650

to perform speaker conversion, so it achieves good results more easily. The proposed models show SSIM results comparable to YourTTS, demonstrating its ability to convert the speaker of the output speech according to the input speaker embedding. Since this work primarily focuses on voice conversion guided by text descriptions, we do not specifically design or apply other tricks for speaker conversion. We leave improving the speaker conversion ability as a future research direction.

5.7.2 Subjective Evaluation

Table 5.8 lists the subjective evaluation results. We report OVL and REL scores for ground-truth speech as well as speech generated by the proposed models without RL, with RL, and with RLHF. Our observations are summarized as follows:

- Both OVL and REL scores of ground-truth utterances are worse than those of generated ones. We attribute this to the fact that each ground-truth utterance contains only one style, either a speaking style or an emotion. On the contrary, the generated speech contains one of the 24 compound styles, making it much more expressive.

Table 5.8: Subjective evaluation results of the proposed models without RL, with RL, and with RLHF. All scores are reported with 95% confidence intervals.

Model	Reward Model	OVL	REL
Base	-	3.25±0.091	3.03±0.098
Base-RL	orig.	3.22±0.094	3.26±0.098
Base-RLHF-v1	ft-12k	3.21±0.094	3.25±0.104
Base-RLHF-v2	tfs-4k	3.18±0.096	3.22±0.101
Base-RLHF-v3	tfs-12k	3.15±0.090	3.28±0.096
Ground Truth	-	2.46±0.101	2.65±0.102

Table 5.9: Subjective pairwise preference test results. A percent preference greater than 50% indicates the proposed RLHF model is preferred over **Base-RL**. All results are reported with 95% confidence intervals.

Model	Reward Model	Pref.
Base-RLHF-v1	ft-12k	0.521±0.037
Base-RLHF-v2	tfs-4k	0.525±0.037
Base-RLHF-v3	tfs-12k	0.559±0.036

Besides, we found that some ground-truth utterances are noisy or with echoes, and some are not that similar to the emotions they are labeled with. The results highlight the challenge of building a text-guided voice conversion model using only publicly available datasets rather than using collected high-quality datasets like in other works.

- For REL, models with RL or RLHF are much better than **Base** with significance (p-value < 0.05). **Base-RLHF-v3** reaches the highest score, although no significant difference exists between models with RL or RLHF. These results imply that the

proposed RL and RLHF approaches can effectively help the model learn to convert speech styles according to guidance from text descriptions.

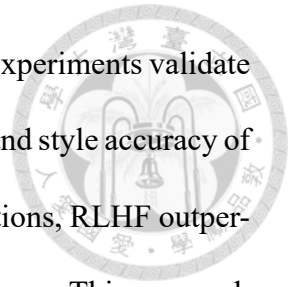
- For OVL, the generated speech has higher scores. The proposed models may benefit from the high-quality TTS data, such as PromptSpeech-R and VCTK, in the training set. Since we do not include quality metrics to calculate rewards in RL, all proposed models have no significant difference ($p\text{-value} > 0.05$).

To better show the improvements made by leveraging human feedback, we conduct pairwise preference tests and list the results in Table 5.9. All three models with RLHF are preferred over **Base-RL**. In addition, strategies to leverage human feedback and build reward models also affect performance. Training from scratch outperforms fine-tuning and reaches the best with sufficient human feedback data. Overall, the subjective evaluation results underline the difficulty of text-guided voice conversion with only public datasets and the effectiveness of using RL and RLHF under such a limited scenario.

5.8 Summary

This chapter addresses the data efficiency challenges in text-guided voice conversion by utilizing reinforcement learning (RL) and reinforcement learning from human feedback (RLHF) to refine speech generation models. We introduce a novel approach that leverages publicly available datasets to train a text-guided voice conversion model, significantly reducing the reliance on costly, proprietary datasets. We adapt a pre-trained text-to-audio model for our specific needs and demonstrate how to enhance model performance using RL techniques that facilitate the model to learn from another reward model built via contrastive learning. For RLHF, we examine the effects of varying the amount of human

feedback and explore different strategies to incorporate it in RL. Our experiments validate that both RL and RLHF can significantly improve the expressiveness and style accuracy of the generated speech. Particularly, as confirmed by subjective evaluations, RLHF outperforms standard RL and shows a closer alignment with human preferences. This approach showcases the potential of using RL and RLHF in speech generation and highlights their effectiveness in making voice conversion models more data-efficient and responsive to textual guidance.





Chapter 6

Enhancing Data Efficiency in Self-Supervised Learning with Speech Generation

In the preceding chapters, we studied the challenges and advancements related to speech generation efficiency, focusing on the computational efficiency of neural vocoders and the data efficiency of text-guided voice conversion. In Chapter 6, we extend the scope of this thesis by exploring the application of speech generation methods to improve the data efficiency of self-supervised learning (SSL) in speech processing. Specifically, we study the potential of speech generation as a powerful method for data augmentation in SSL, aiming to mitigate the substantial demand for unlabeled speech data. This research not only enhances the data efficiency of SSL in speech processing but also broadens the contributions of this thesis from optimizing the efficiency of speech generation to enhancing data efficiency with speech generation techniques.

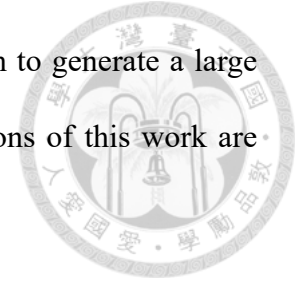
6.1 Introduction



Using enormous amounts of unlabeled data and tiny fractions of labeled ones helped self-supervised learning methods achieve remarkable results in the field of computer vision [185, 186], natural language processing [109, 110], and speech processing [13, 95]. Such dependency, however, has raised concerns for certain applications [187–189], may be incorporated at pre-training, leading to potential leakage to downstream applications. In the field of speech processing, the leakage can be related to speaker identity or unique timbre in speech utterances [190]. The text content, on the other hand, is simpler to anonymize by using only public domain text data [191]; hence, it doesn't face the same level of scrutiny as speech data that deeply entangles speaker identity and style into the spoken content.

Most recent SSL speech models are pre-trained on thousands to tens of thousands of hours of real-world speech data to achieve outstanding performance [12, 13, 95]. This study aims to alleviate this heavy reliance on large amounts of real speech data by leveraging synthetic data to augment the training corpus. For example, can we train a competitive SSL speech model solely using 100 hours of audio-only data and 10 hours of paired audio and labeled data? The proposed approach involves utilizing the high-quality discrete representation units learned by an SSL model to build a high-quality text-to-speech (TTS) system, which subsequently augments speech corpus for pre-training a better SSL model. We first pre-train an SSL model on a small amount of real-world speech data. Discrete units are then extracted from the speech data using the SSL model to construct a multi-speaker unsupervised unit-to-speech model. Next, we build a text-to-unit model using limited text-speech paired data. Following that, a TTS system is built by integrating

the text-to-unit and unit-to-speech models. We use this TTS system to generate a large synthetic corpus and pre-train a better SSL model. The contributions of this work are summarized as follows:

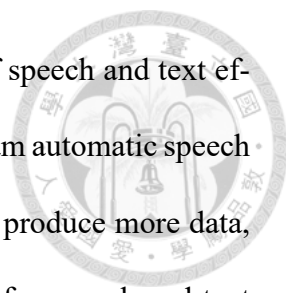


- We investigate the performance of a speech SSL model pre-trained solely on a 100 hours audio-only dataset and demonstrate the impact of data scarcity on SSL models, including the challenge of overfitting and the resulting degradation in performance due to the limited pre-training data.
- We propose a TTS system to augment the limited data for SSL pre-training. By leveraging discrete units obtained from the SSL model pre-trained on 100 hours of audio-only data, we build unit-to-speech and text-to-unit models to construct a high-quality multi-speaker TTS system, which is then used to generate a large pre-training speech corpus.
- Experimental results show that the proposed method effectively reduces the demand for real-world pre-training data by 90% with only slight performance degradation compared to SOTA SSL approaches on the standard LibriSpeech benchmark.

6.2 Related Work

6.2.1 Self-Supervised Learning with Unpaired Text Data for Speech Processing

Many works have explored enhancing speech SSL models by incorporating additional unpaired text data. In [192], the authors propose using a well-trained TTS model and text data to create a paired dataset and combine it with speech-only data for SSL model



pre-training. With rich data resources, the joint learning paradigm of speech and text effectively improves the performance of the SSL model in the downstream automatic speech recognition (ASR) task. Besides using text data and a TTS model to produce more data, some other works [193–197] proposes to learn joint representations for speech and text modalities and build multimodality SSL models. The resulting models demonstrate prominent improvements in various speech-text tasks such as ASR and speech translation.

Even though pre-training SSL models with speech and text data has demonstrated remarkable improvements in speech processing, the previous approaches still heavily rely on rich paired or unpaired data. None of these works have explored the development of an SSL model with limited data resources.

6.2.2 Speech Generation for Data Augmentation

Leveraging synthetic speech to enhance ASR performance has been an established and effective strategy [198–201, 201–204]. These methods employ text data and TTS models to create extensive paired datasets, thereby augmenting the training data for ASR. By generating speech data with increased acoustic and lexical diversity, these approaches effectively lead to better ASR results, demonstrating the benefits of utilizing synthetic speech data in the development of ASR systems, especially in low-resource settings.

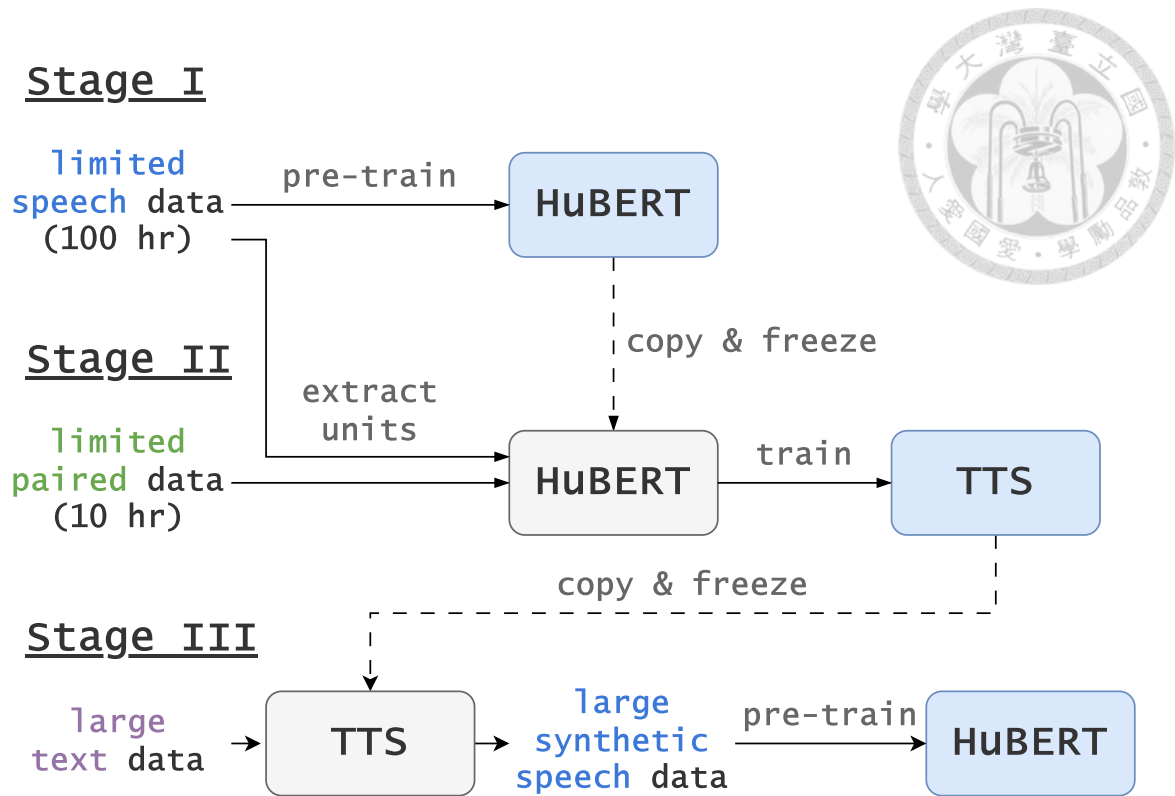


Figure 6.1: Overview of the proposed system.

6.3 Method

6.3.1 Overview

In this work, we choose HuBERT [13] as the SSL model to improve in a low-resource setting. The training datasets are in three settings: limited text-speech paired data, limited speech data, and rich-resource text data. More information about the datasets is detailed in Section 6.4.1.

The high level idea of this work is illustrated in Figure 6.1. In **Stage I**, we first pre-train a HuBERT model on the limited unpaired speech data. This model is used in **Stage II** to extract discrete speech representation [13]. We use the SSL representation to help build a multi-speaker TTS system and generate a large synthetic dataset from the rich-resource



Table 6.1: Average length ratios (unit length / phoneme length) with different postprocessing methods.

Remove repetitions	DPDP	Ratio
✗	✗	4.2
✓	✗	2.1
✓	✓	1.7

text dataset. The synthetic dataset is then augmented and used for pre-training a better HuBERT model (**Stage III**).

6.3.2 Extracting Discrete Speech Representation

In **Stage II**, we first extract speech representations using a HuBERT model. The model is a 12-layered BASE model [13] and pre-trained for three iterations on the limited speech dataset in our setting. We then extract the representations from the 9th layer and apply k-means clustering to convert continuous features into discrete units. We set $k = 500$ in our experiments. Lastly, we adopt the duration-penalized dynamic programming (DPDP) algorithm proposed in [205] and remove repeated tokens [13]. The duration penalty is set to 1.0.

Table 6.1 shows the average length ratios between unit and phoneme sequences with different postprocessing methods. The DPDP algorithm smooths the unit sequence, resulting in a shorter sequence after removing repetitions, which we found helpful for building the TTS system.

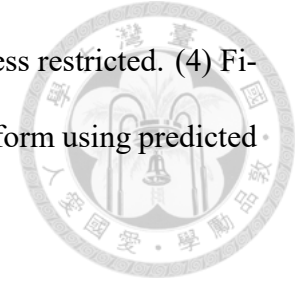


6.3.3 Text-to-speech Modules

A common multi-speaker TTS pipeline usually consists of an acoustic feature predictor and a vocoder to restore waveform from acoustic features [1, 2]. The former is usually trained on a large amount of paired data, and the latter only on speech data. In our setting, the total speech data for training is limited to about 100 hours, and the text-speech paired data is even scarcer (about 10 hours). With only 10 hours of paired data, it is hence challenging to build a multi-speaker TTS from scratch. Some low-resource TTS works leverage additional data from rich resource languages [206] or apply ASR methods to help TTS training [207, 208], while in this work, we convert speech into discrete representation to reduce the difficulty of training.

The proposed TTS system comprises four components: (1) A *session encoder* (SE) encodes each speech utterance into a vector, representing information such as speaking style in the utterance. The representation is used during training and generation. (2) A *text-to-unit* model (T2U) predicts the discrete units representing speech information from given text. In preliminary experiments, we found that it was significantly easier to predict discrete units than continuous acoustic features such as Mel-spectrograms. With only a limited amount of paired data, a multi-speaker text-to-Mel model failed to converge and failed to generate intelligible speech. (3) Two variance predictors predict more detailed prosody and pitch information from discrete units. A *duration predictor* (DP) predicts the duration of each discrete unit, and a *pitch predictor* (PP) predicts the $\log(F_0)$ at each frame. It is worth noting that only speech data is required to train both predictors. Compared with a common TTS model, which takes paired data to train the whole acoustic feature predictor [1, 2], the proposed method allows training parts of the acoustic feature

predictors without text labels; hence, the amount of training data is less restricted. (4) Finally, a *unit-to-speech* model (U2S) is trained to synthesize the waveform using predicted discrete units and acoustic features.



Session Encoder (SE)

We follow [209] to build SE and extract x-vectors as session embedding.¹ The model is trained on the limited speech dataset in our setting. To preserve the diversity of speech utterances, we do not average the x-vectors of the same speaker. Instead, we extract an x-vector for each speech utterance and use the utterance-level x-vectors for training and synthesis.

Text-to-unit Model (T2U)

T2U is implemented based on Tacotron 2 [1]. A text sequence is first converted into a phoneme sequence and passed as input to T2U, which then generates a unit sequence. The session embedding from SE is concatenated to the output of the encoder, allowing T2U to generate units in different speaking styles. We append end-of-sentence (EOS) tokens to the end of both input and output sequences, guiding the decoder to predict a stop token when attending to the last input or an EOS token has been predicted at the previous timestep. We remove repeated tokens in target unit sequences during training, and the model follow [16] to predict two units at each timestep. This process shortens the output length and makes it closer to the input length, leading to faster convergence speed and more stable attention alignments, as mentioned in [16].

¹Session encoder training is done on NTU infrastructure on publicly available data.



Duration Predictor (DP) and Pitch Predictor (PP)

Similarly to [210, 211] a duration predictor is trained to restore repeated tokens in unit sequences removed in training T2U. The model takes a deduplicated unit sequence as input and predicts the number of repetitions for each unit.

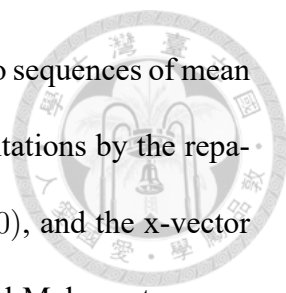
To generate more acoustic information before restoring the speech waveform, we train a pitch predictor. The model takes a unit sequence (with repetitions) as input and predicts the log-scaled fundamental frequency ($F0$) at each frame.

Both predictors adopt the same architecture. The input sequence is first encoded by convolution layers with skip connections, followed by an LSTM layer and a combination of several convolution layers and layer normalization. The session embedding is concatenated to each timestep of the input. The predictors are trained to minimize the mean absolute error, and $\log(F0)$ is used for calculating the loss.

Unit-to-speech Model (U2S)

We follow [30, 210] to build a unit-to-speech model with some modifications. The original unit-based HifiGAN from [30] uses the normalized and quantized $\log(F0)$ to provide additional pitch information, while in this work, we simply use the unnormalized and continuous $\log(F0)$. U2S takes in units, $\log(F0)$, and the x-vector to reconstruct the waveform.

To further enhance the quality of the generated speech, we add a variational auto-encoder (VAE) to model the acoustic information that is not explicitly captured in units, $\log(F0)$, and x-vectors. During training, a Mel-spectrogram is first extracted from the tar-



get waveform, and the VAE encoder encodes the Mel-spectrogram into sequences of mean and variance vectors, which are then used to generate latent representations by the reparameterization trick. The latent representations, the units, the $\log(F0)$, and the x-vector are concatenated and passed to the decoder to reconstruct the original Mel-spectrogram. The output of the last hidden layer is used as an additional condition for U2S. At inference time, we discard the encoder and use the normal Gaussian distribution as the prior of the latent representations. The VAE is built using convolution layers with skip connections, and we train the VAE and U2S jointly.

Data Augmentation and Oversampling

To enrich the diversity of the synthetic dataset, we augment the generated utterances with two methods. First, we observed that the generated utterances exhibit a length 80% shorter than natural speech. Consequently, for each utterance, we multiply the durations predicted by DP with a scalar uniformly sampled between 1.0 to 1.5. This process stretches the speech, making the speaking rate various and closer to natural speech. Secondly, we add background noise from our noise dataset to the generated speech with a random SNR between 0 to 15.

Finally, we combine the augmented synthetic data and the limited real-world speech data as the training set to build the HuBERT model. We found that oversampling the natural speech utterances improves the performance. Specifically, with an oversampling rate of r , the natural utterances are sampled for pre-training r times more frequently than synthetic ones.



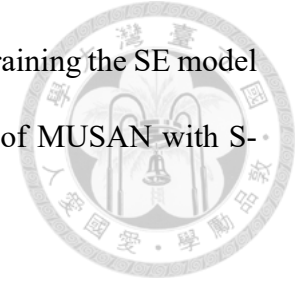
6.4 Experimental Setup

6.4.1 Datasets

We use LibriSpeech [191] for pre-training, fine-tuning, and evaluation. The dataset is split into several splits for different purposes. All the speech data is downsampled to 16 kHz.

- **S-100hr.** In our setting, we limited the total duration of speech data to 100 hours. Instead of directly using the *train-clean-100* split in LibriSpeech, we select utterances from the *train-clean-100*, *train-clean-360*, and *train-other-500* splits, making a new split with clean and noisy speech data. This split contains 29721 speech clips from 245 speakers (123 males and 122 females) without text transcriptions.
- **ST-10hr.** We use the 10-hour split of Libri-light [212] as the low resource paired dataset. This split contains 2763 utterances from 24 speakers, and the total duration is about 10 hours. Speech utterances in *ST-10hr* are included in *S-100hr*.
- **S-960hr.** This split includes all training splits in LibriSpeech, which contains 281241 utterances from 2338 speakers with a total duration of about 960 hours. We use the split to pre-train the topline HuBERT model in our experiments.
- **T-960hr.** This split includes all text transcriptions of the total 960 hours data in LibriSpeech. These transcriptions are used to generate the synthetic dataset.
- **T-LM.** This split includes the text used to train the language model in LibriSpeech. The data is used to generate the large synthetic dataset in Section 6.5.3.
- **dev-clean and dev-other.** We use the *dev-clean* and *dev-other* splits of LibriSpeech for validation while building the systems and for evaluating the performance.

We use the MUSAN dataset [213] for data augmentation when training the SE model and augmenting the synthetic dataset. We replaced the *speech* part of MUSAN with S-100hr to prevent using additional speech data.

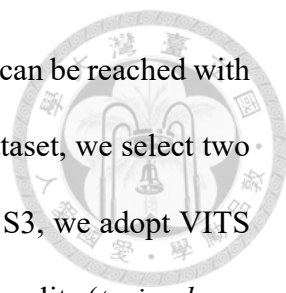


6.4.2 Comparing Systems

- **Baseline HuBERT (S0).** The baseline in this work is a HuBERT BASE model pre-trained only on limited speech data, *S-100hr*. We pre-train the model for three iterations. Models in different iterations are pre-trained for 200k steps, 400k steps, and 400k steps, respectively. After pre-training, the model is fine-tuned for the ASR task on *ST-10hr* for 40k steps. In each iteration, we experiment with different settings and select the best model as the teacher [13] for the next iteration, as shown in Table 6.2. In this study, we denote different S0 models as *S0- x^{th} -kmy-last (best)-lz*, which means the model is from the x^{th} iteration, the cluster number for k-means is set to y , the last (best) checkpoint during pre-training is selected, and we are referring to the z^{th} layer of the model.

We use the official HuBERT implementation and configurations provided in the Fairseq toolkit [214] for pre-training and fine-tuning. The model is pre-trained on 32 GPUs and fine-tuned on 8 GPUs. If not specified, the HuBERT models in the following systems are pre-trained and fine-tuned with the same setting of *S0-1st-km100*.

- **Topline HuBERT (S1).** Similar to the BASE model in [13], S1 is pre-trained on *S-960hr*. This model is considered a topline system built with high-resource speech data.

- 
- **Off-the-shelf TTS (S2 and S3).** To show the performance that can be reached with a high-quality TTS model trained on the high-resource TTS dataset, we select two off-the-shelf TTS models released in ESPnet [215]. In S2 and S3, we adopt VITS models [3] trained on the VCTK dataset [124] and on the clean splits (*train-clean-100* and *train-clean-360*) of LibriTTS [79], respectively. We use the pre-trained TTS models and *T-960hr* to generate two synthetic datasets. While generating, the target speakers are randomly sampled from the seen speakers. There are 108 speakers in S2 and 1151 in S3. Two HuBERT models are then pre-trained on the different synthetic datasets, respectively.
 - **Proposed (S4).** To build the proposed system, we first extract discrete units from the speech data in *S-100hr* and *ST-10hr*. We use the features from the best *S0-3rd* model (refer to Section 6.5.1) and generate units as described in Section 6.3.2. T2U is trained on limited paired data, *ST-10hr*, while SE, DP, PP, and U2S are trained on limited speech data, *S-100hr*. All modules are trained on 8 NVIDIA V100 GPUs, except for SE on 1 NVIDIA 2080Ti GPU. We use *T-960hr* to generate the dataset. Target speakers are randomly selected from the 245 speakers in *S-100hr*. The synthetic dataset is then augmented and combined with *S-100hr* for HuBERT pre-training.

6.4.3 Evaluation Methods

To evaluate the quality of the discrete units extracted using *S0* in different iterations, we calculate phone purity (PP) and cluster purity (CP) [13] on *dev-clean* and *dev-other*. For the fine-tuning results, we report the word error rate (WER) on *dev-other*, decoded with a 4-gram language model trained on the official LibriSpeech language modeling data.

All the evaluation metrics are from the official HuBERT implementation in the Fairseq toolkit.



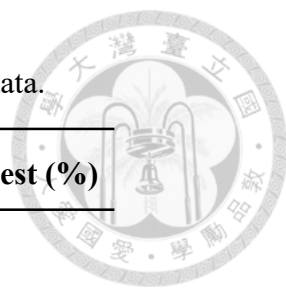
6.5 Results

6.5.1 Performance without Synthetic Data

We first evaluate the performance of the baseline model, S0, pre-trained on mere 100 hours of speech data of *S-100hr*. The results are presented in Table 6.2. The column **Feat.** indicates the type of the pre-training target, which is either mfcc or representations extracted from the model of the previous iteration with the best WER result. **K** denotes the number of clusters for k-means clustering. We report the WERs on the *dev-other* split of LibriSpeech. **-last** and **-best** represent that the last and the best model checkpoints during pre-training are used for fine-tuning, respectively. It is worth noting that the **WER-last** gets worse in the 2nd and 3rd iterations, deviating from the findings in [13]. This divergence could be attributed to overfitting as we constrain the amount of training data to 100 hours only. Even fine-tuning with the best checkpoints during pre-training fails to enhance the performance. Thus, we can deduce that a HuBERT BASE model can only reach a WER of approximately 25% when pre-trained on limited speech data. For comparison, the topline model (S1) attains 14.2% when pre-trained on 960 hours of speech data.

As the proposed system relies on discrete units generated using S0 and k-means clustering, we first evaluate the quality of the units. We select S0 models in different iterations with the best WERs, which are *S0-1st-km100-last*, *S0-2nd-km100-best*, and *S0-3rd-km500-best*. We also consider the officially released pre-trained HuBERT BASE model

Table 6.2: Results of pre-training on natural speech data.



Feat.	K	WER-last (%)	WER-best (%)
<i>S0-1st</i>			
mfcc	100	25.0 / 25.1	31.1 / 33.8
<i>S0-2nd</i>			
S0-1st-km100-last-l6	100	33.6 / 32.4	27.6 / 26.7
S0-1st-km100-last-l6	500	38.2 / 37.8	27.1 / 27.6
<i>S0-3rd</i>			
S0-2nd-km100-best-l9	100	34.1	26.5
S0-2nd-km100-best-l9	500	34.6	25.4
<i>S1-1st</i>			
mfcc	100	14.2	14.4

for comparison. The features of different layers are extracted from *S-100hr*, then k-means clustering with different k is conducted on these features. The results are listed in Table 6.3. Unlike our previous findings, the quality of units improves with more pre-training iterations. We select *S0-3rd-km500-best-l9* and $k = 500$ to generate units for the proposed system as the setting is the most performant.

6.5.2 Performance of Off-the-Shelf TTS Methods

We use off-the-shelf TTS methods to generate synthetic datasets and pre-train a HuBERT models, as described in Section 6.4.2. The results are listed in Table 6.4. **Nat.** and **Synth.** denote the total duration of the natural and the synthetic data for pre-training, respectively. The dataset generated by S2 is smaller and contains only 652 hours of speech data. The WERs are also much worse than the baseline. For S3, the TTS model is trained on a

Table 6.3: Quality analysis of discrete units.

Layer	K	PP (%)	CP (%)
<i>S0-1st-km100-last</i>			
6	100	48.36	21.20
6	500	58.21	8.59
<i>S0-2nd-km100-best</i>			
9	100	54.69	26.25
9	500	64.32	9.42
<i>S0-3rd-km500-best</i>			
6	100	55.43	23.87
9	100	59.03	27.53
11	100	58.38	27.22
12	100	60.49	28.03
6	500	61.93	7.82
9	500	67.32	9.39
11	500	67.05	9.54
12	500	66.97	9.57
<i>Pre-trained HuBERT BASE</i>			
6	500	68.12	8.99



dataset with 1151 speakers and adopts an x-vector to specify the target speaker. When synthesizing a dataset, we randomly sample x-vectors of 24, 245, and all 1151 speakers from the training data and all 2338 speakers from *S-960hr*. The results show that speaker diversity of the dataset is crucial for pre-training a good HuBERT model. With the number of speakers increasing from 24 (same as *ST-10hr*), 245 (same as *S-100hr*), to 1151, the WER improves accordingly from 27.0%, 26.0%, to 23.2%. The best result of S3 is about

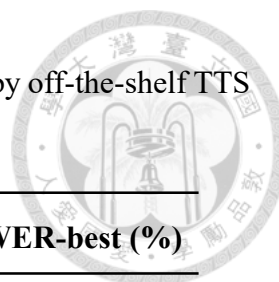


Table 6.4: Results of pre-training on synthetic speech data generated by off-the-shelf TTS methods.

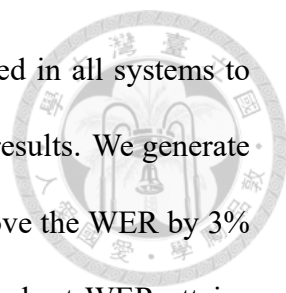
Sys.	# of spk.	Nat. (hr)	Synth. (hr)	WER-last (%)	WER-best (%)
S0	245	100	-	25.0	31.1
S1	2338	960	-	14.2	14.4
S2	108	-	652	34.3	50.0
S3	24	-	800	27.3	27.0
S3	245	-	805	26.0	26.0
S3	1151	-	809	23.2	23.2
S3	2338	-	826	22.3	22.2

22.2%.

6.5.3 Performance of the Proposed System

Table 6.5 shows the performance of the proposed system. We report **WER-last** except for the 2nd iteration of S0. **Spk. per uttr.** indicates how many speech utterances of different speakers are generated for each text utterance. **Synth. FT** denotes the size of additional synthetic data used for fine-tuning. We conclude our observations as follows: (1) augmenting the 100 hours natural speech with background noise, denoted as S0n, does not improve the performance; (2) the proposed TTS method attains a similar WER to S3 (23.5% vs. 23.2%), while the latter uses a large high-quality dataset to build the TTS model; (3) generating the same utterance with different speaker styles and a higher over-sampling rate for the natural speech effectively improve the WER to 20.4%; (4) increasing the pre-training steps from 200k to 400k slightly helps for a larger synthetic dataset.

To further improve the performance, we augment the fine-tuning data with the paired



data synthesized from *T-LM*. Note that this text dataset has been used in all systems to build the 4-gram language model for decoding. Table 6.6 shows the results. We generate extra 100, 1k, and 10k hours of paired data for fine-tuning and improve the WER by 3% to 17.5%. Lastly, we run the pre-training for the 2nd iteration. The best WER attains 15.8%, which is close to the performance of the topline. Compared with S0, the proposed system decreases the WER by about 10% (from 25.0% to 15.8%). Compared with S1, the proposed system reduces the demand for speech data by approximately 90% (from 960 hours to 100 hours) with only slight degradation in WER.

Table 6.5: Results of pre-training on synthetic speech data generated by the proposed method.

Sys.	# of spk.	Spk. per uttr.	Nat. (hr)	Synth. (hr)	Oversampling rate	Steps	Iter.	Synth. FT (hr)	WER (%)
<i>Fine-tuning on ST-10hr</i>									
S0	245	-	100	-	-	200k	1	-	25.0
S0	245	-	100	-	-	400k	2	-	26.7
S0n	245	-	100	-	-	200k	1	-	26.7
S1	2338	-	960	-	-	200k	1	-	14.2
S2	108	1	-	652	-	200k	1	-	34.3
S3	1151	1	-	809	-	200k	1	-	23.2
S4	245	1	-	1.1k	-	200k	1	-	23.5
S4	245	1	100	1.1k	1	200k	1	-	21.5
S4	245	1	100	1.1k	1	400k	1	-	22.2
S4	245	10	100	11k	1	200k	1	-	23.8
S4	245	10	100	11k	10	200k	1	-	21.5
S4	245	10	100	11k	10	400k	1	-	21.0
S4	245	10	100	11k	100	200k	1	-	20.4



Table 6.6: Results of pre-training and fine-tuning on synthetic speech data generated by the proposed method.

Sys.	# of spk.	Spk. per uttr.	Nat. (hr)	Synth. (hr)	Oversampling rate	Steps	Iter.	Synth. FT (hr)	WER (%)
<i>Fine-tuning on ST-10hr and synthetic data</i>									
S4	245	10	100	11k	100	200k	1	100	19.2
S4	245	10	100	11k	100	200k	1	1k	17.9
S4	245	10	100	11k	100	200k	1	10k	17.5
S4	245	1	100	1.1k	100	200k	2	10k	15.8
S4	245	10	100	11k	100	200k	2	10k	17.2



6.6 Summary



This chapter aims to improve the data efficiency of self-supervised learning in speech processing by utilizing synthetic speech to augment a low-resource pre-training corpus. We first construct a high-quality TTS system with limited data resources. The TTS system is subsequently employed to generate large amounts of speech data for pre-training a HuBERT model. Our experiments show a significant performance loss when pre-training a HuBERT model with only 100 hours of speech data, about 10% of the original setting. On the contrary, by incorporating synthetic data for pre-training, the proposed approach drastically alleviates the substantial demand for real-world speech data by 90% with only minimal performance degradation.





Chapter 7

Conclusion

7.1 Thesis Summary

This thesis centers on efficient speech generation, a crucial component in human-computer interaction. We explore different aspects of efficiency: training, inference, and data efficiency, each of which is studied through a speech generation task. Subsequently, we extend our research by applying speech generation to improve data efficiency for another speech-related task. For training and inference efficiency, we optimize model architectures and generation methods for non-autoregressive and autoregressive neural vocoders. For data efficiency, we study utilizing reinforcement learning and human feedback for data-efficient text-guided voice conversion without extensive labeled data. Lastly, we extend our contributions to enhance data efficiency in speech self-supervised learning, demonstrating the broader applicability of efficient speech generation. We summarize the contributions of this thesis from two perspectives: the achievements of each chapter and the impact on subsequent works, setting the stage for future advancements in efficient speech generation.

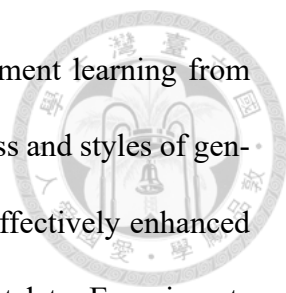


7.1.1 Achievements of Each Chapter

In Chapter 3, we proposed an improved architecture for the non-autoregressive WaveG-low vocoder to address the challenge of computational efficiency in speech generation. The proposed architecture reduced the model size by 97% and the required computational resources through weight-sharing and an additional post-filter, achieving faster training and inference capabilities without compromising speech quality. Our experiments demonstrated that this streamlined architecture generated high-quality 44 kHz speech in real-time without GPU acceleration, significantly improving over traditional methods that relied on more resource-intensive models.

Chapter 4 introduced innovative methods to enhance the inference efficiency of autoregressive neural vocoders. From rethinking the generation direction of the conventional autoregressive approach, we redesigned new techniques, frequency-wise autoregressive generation (FAR) and bit-wise autoregressive generation (BAR), enabling faster generation while maintaining high speech quality. The proposed methods allowed for the integration of non-autoregressive models to speed up the autoregressive generation process. Experiments revealed that the proposed model achieved significantly faster inference speed than conventional autoregressive methods, matching the speeds of non-autoregressive vocoders. Perceptual evaluations further confirmed that the proposed model generated speech with higher quality than other competitive models, demonstrating a novel and effective solution to the trade-off between speed and quality for speech generation technologies.

In Chapter 5, we focused on improving the data efficiency of text-guided voice conversion. The proposed model was built only on publicly available datasets. We pioneered

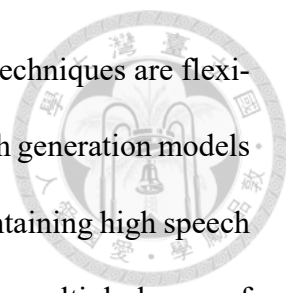


the research of adopting reinforcement learning (RL) and reinforcement learning from human feedback (RLHF) in speech generation to enrich expressiveness and styles of generated speech, demonstrating that the integration of RL and RLHF effectively enhanced performance without requiring additional human-annotated speech-text data. Experiments showed that models with RL or RLHF improved objective evaluation results up to two times better and generated much preferred converted speech in subjective evaluations. The proposed approach provided a more effective learning paradigm to reduce the dependency on large, costly datasets and enhance the model's ability to handle more complex style descriptions unseen in the training datasets.

Chapter 6 explored how speech generation techniques can help data efficiency in other speech processing tasks. We studied building a text-to-speech (TTS) system with discrete speech units and low data resources. The TTS system generated synthetic datasets and augmented training data for speech self-supervised learning. By using synthetic data to supplement real speech datasets, we demonstrated a reduction in the need for extensive labeled datasets. Our results showed that the proposed method significantly decreased the reliance on real-world data by 90% without substantial losses in performance, thus extending the achievements of this thesis from efficiency improvements in speech generation to broader efficiency advancements in speech processing.

7.1.2 Impact to Subsequent Works

The achievements in Chapter 3 and Chapter 4 demonstrate how modifying model architecture and redesigning the generation process can drastically reduce computational resources. The proposed methods, such as model compression, post-filtering, and re-designed autoregressive generation, are not only applicable to the models used in these

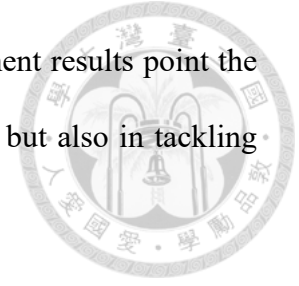


chapters but also hold great potential for broader application. These techniques are flexible enough to be adopted in advanced neural vocoders and other speech generation models to reduce the number of parameters or speed up generation while maintaining high speech quality. For example, weight-sharing can be applied to models with multiple layers of identical structures. This approach has been shown as an effective technique for compressing transformer-based models for various tasks [98, 216], implying that our method in Chapter 3 can potentially be applied to transformer-based text-to-speech models, which are widely used in current speech generation research. Besides, similar to BAR in Chapter 4, some recent speech generation models iteratively generate discrete speech units from coarse to fine precisions, achieving a balanced trade-off between inference speed and speech quality with remarkable success. These results highlight the potential of our methods to inspire future work in achieving more efficient speech generation with fewer computational resources.

In Chapter 5, we identified the challenges posed by limited data resources in text-guided voice conversion and demonstrated how to apply RL and RLHF as a solution to facilitate more effective learning. Given the limited research applying RL to speech generation, this pioneering work showcases not only the feasibility of integrating RL with speech generation but also extensive experiment results, including how to construct effective reward models, the necessary amount of human feedback, and various strategies to utilize human feedback, laying a crucial foundation for future research.

Chapter 6 expands the scope of efficient speech generation to broader applications. We demonstrated how to leverage less speech data, SSL features, and discrete speech units to construct a high-quality text-to-speech model, confirming that speech generation can serve as an effective data augmentation method in low-resource settings to enhance the

performance of speech models. The proposed methods and experiment results point the way for future research not only in speech self-supervised learning but also in tackling low data resource challenges across various speech processing tasks.



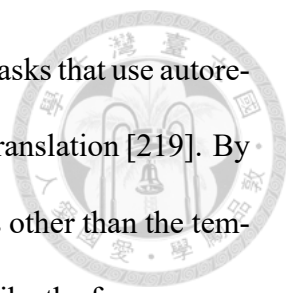
7.2 Future Work

7.2.1 Exploring Applications and New Domains for Redesigned Autoregressive Generation

The FAR and BAR proposed in Chapter 4 have shown significant promise in efficiency and quality. One direction for future work is to extend this success to later autoregressive or diffusion-based waveform generation models [36, 54, 217] to enhance inference speeds. Similarly, non-autoregressive methods such as GAN-based vocoders [7, 35] can adopt FAR and BAR to introduce autoregressive generation paradigms for better speech quality.

Another research direction is to find new domains other than the frequency and bit precision domains for autoregressive waveform generation. For example, one can follow the principle of latent diffusion models [143] to first transform speech into the latent space of a pre-trained variational autoencoder. Within this latent space, autoregressive generation is then conducted along the channel axis. Additionally, by employing specialized loss functions or architectural designs, one can encode different granularities of speech information into different channels. This design can enable autoregressive generation to iteratively produce speech representations in the latent space from coarse to fine granularities, potentially enhancing the quality of generated speech.

Extending to the idea of finding new domains, the proposed autoregressive generation



concept applies not only to speech generation but also to other speech tasks that use autoregressive models, like automatic speech recognition [218] and speech translation [219]. By following the core principle of “autoregressive generation in domains other than the temporal one,” these speech models can utilize appropriate domains, just like the frequency or bit precision domains in waveform generation, and adapt their autoregressive generation processes to drastically enhance the generation speed without compromising quality.

7.2.2 Expanding Applications of Reinforcement Learning in Speech Generation

RL and RLHF for speech generation introduced in Chapter 5 have achieved notable success. However, existing research integrating RL and audio generation typically only uses text prompts describing simple sound events or combinations of speech styles. A future research direction can explore generating or converting speech based on more detailed descriptions. For example, specifications in descriptions can include speaking environments, background noise, delicate emotional variations in various real-world situations, and nuances in intonation or volume across different parts of an utterance. This direction involves designing and generating intricate style descriptions and developing reward models to accurately assess the relevance between the speech and these detailed descriptions.

Another potential direction involves incorporating multi-faceted rewards in the existing RL framework, which can simultaneously evaluate speech in various aspects besides text-speech relevance, such as naturalness and speaker similarity. The multi-reward RL can optimize model performance in terms of different evaluation metrics. This direction focuses on designing learning algorithms to integrate multiple types of rewards while retaining effective and stable training.

7.2.3 Extending Iterative Training for Self-Supervised Learning and Speech Generation



Chapter 6 demonstrated the effectiveness of leveraging speech generation to support low-resource SSL in speech processing. The proposed method uses a low-resource SSL model to extract features to build a TTS model. The TTS model then generates synthetic data to augment the SSL training dataset, which is used to train a better SSL model. Inspired by HuBERT [13], future research can further extend this iterative learning process. The improved SSL model can again be used to extract features and build a better TTS model, leading to another round of data augmentation and SSL pre-training. By repeating the cycle of SSL pre-training, building a TTS model on SSL features, and data augmentation, one can iteratively enhance the performance of both TTS and SSL models.


Inspired by the recent success of using a single decoder-only transformer for various speech tasks [220, 221], another research direction is to incorporate a unified model with the proposed method, iteratively learning and augmenting datasets for different tasks. One can first build a decoder-only transformer on the SSL task. Then, this model is used to extract SSL features and is trained for other speech tasks, such as TTS, voice conversion, and ASR. Subsequently, this model can generate more speech-only or paired text-speech data to augment training data. Similarly, this process can be repeated multiple times to enhance model performance and build better synthetic datasets. This approach aims to learn more efficiently when the data sizes of different modalities are imbalanced (e.g., scarce speech data and rich text data), creating a unified model that benefits from multitask training and generates better training data in a self-training manner [222, 223].

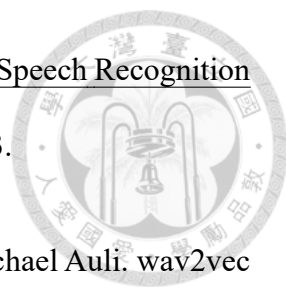


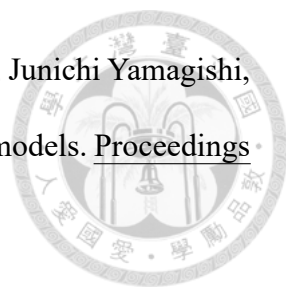



References

- [1] Jonathan Shen, Ruoming Pang, Ron J Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, Yu Zhang, Yuxuan Wang, Rj Skerrv-Ryan, et al. Natural tts synthesis by conditioning wavenet on mel spectrogram predictions. In 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 4779–4783. IEEE, 2018.
- [2] 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. In International Conference on Learning Representations, 2021.
- [3] Jaehyeon Kim, Jungil Kong, and Juhee Son. Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech. In International Conference on Machine Learning, pages 5530–5540. PMLR, 2021.
- [4] 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. arXiv preprint arXiv:1609.03499, 2016.
- [5] Nal Kalchbrenner, Erich Elsen, Karen Simonyan, Seb Noury, Norman Casagrande, Edward Lockhart, Florian Stimberg, Aaron Oord, Sander Dieleman, and Koray


- 
- Kavukcuoglu. Efficient neural audio synthesis. In International Conference on Machine Learning, pages 2410–2419. PMLR, 2018.
- [6] 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. IEEE, 2019.
- [7] Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae. Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis. Advances in Neural Information Processing Systems, 33:17022–17033, 2020.
- [8] Zhifang Guo, Yichong Leng, Yihan Wu, Sheng Zhao, and Xu Tan. Prompttts: Controllable text-to-speech with text descriptions. In ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 1–5. IEEE, 2023.
- [9] Dongchao Yang, Songxiang Liu, Rongjie Huang, Chao Weng, and Helen Meng. Instructtts: Modelling expressive tts in discrete latent space with natural language style prompt, 2023.
- [10] Jixun Yao, Yuguang Yang, Yi Lei, Ziqian Ning, Yanni Hu, Yu Pan, Jingjing Yin, Hongbin Zhou, Heng Lu, and Lei Xie. Promptvc: Flexible stylistic voice conversion in latent space driven by natural language prompts. arXiv preprint arXiv:2309.09262, 2023.
- [11] Chun-Yi Kuan, Chen-An Li, Tsu-Yuan Hsu, Tse-Yang Lin, Ho-Lam Chung, Kai-Wei Chang, Shuo-Yiin Chang, and Hung-yi Lee. Towards general-purpose text-

- 
- instruction-guided voice conversion. In 2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), pages 1–8. IEEE, 2023.
- [12] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. Advances in neural information processing systems, 33:12449–12460, 2020.
- [13] Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 29:3451–3460, 2021.
- [14] 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. In Proc. Interspeech 2021, pages 141–145, 2021.
- [15] Kaizhi Qian, Yang Zhang, Shiyu Chang, Mark Hasegawa-Johnson, and David Cox. Unsupervised speech decomposition via triple information bottleneck. In International Conference on Machine Learning, pages 7836–7846. PMLR, 2020.
- [16] 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. In Proc. Interspeech 2017, pages 4006–4010, 2017.
- [17] Heiga Ze, Andrew Senior, and Mike Schuster. Statistical parametric speech synthesis using deep neural networks. In 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, pages 7962–7966. IEEE, 2013.

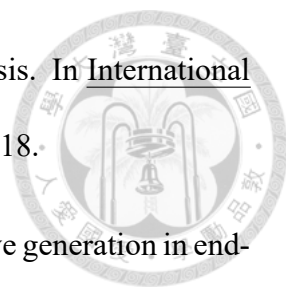
- 
- [18] Keiichi Tokuda, Yoshihiko Nankaku, Tomoki Toda, Heiga Zen, Junichi Yamagishi, and Keiichiro Oura. Speech synthesis based on hidden markov models. Proceedings of the IEEE, 101(5):1234–1252, 2013.
- [19] Masanori Morise, Fumiya Yokomori, and Kenji Ozawa. World: a vocoder-based high-quality speech synthesis system for real-time applications. IEICE TRANSACTIONS on Information and Systems, 99(7):1877–1884, 2016.
- [20] Hideki Kawahara, Jo Estill, and Osamu Fujimura. Aperiodicity extraction and control using mixed mode excitation and group delay manipulation for a high quality speech analysis, modification and synthesis system STRAIGHT. In Second International Workshop on Models and Analysis of Vocal Emissions for Biomedical Applications, 2001.
- [21] Zhizheng Wu, Oliver Watts, and Simon King. Merlin: An open source neural network speech synthesis system. In SSW, pages 202–207, 2016.
- [22] Daniel Griffin and Jae Lim. Signal estimation from modified short-time fourier transform. IEEE Transactions on Acoustics, Speech, and Signal Processing, 32(2):236–243, 1984.
- [23] 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. In Proc. Interspeech 2018, pages 501–505, 2018.
- [24] Cheng-chieh Yeh, Po-chun Hsu, Ju-chieh Chou, Hung-yi Lee, and Lin-shan Lee. Rhythm-flexible voice conversion without parallel data using cycle-gan over phoneme posteriorgram sequences. In 2018 IEEE Spoken Language Technology Workshop (SLT), pages 274–281. IEEE, 2018.

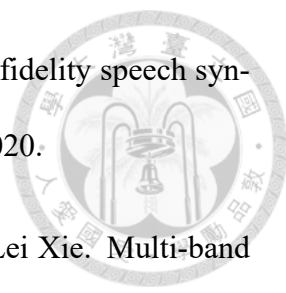
- 
- [25] Andy T. Liu, Po chun Hsu, and Hung-Yi Lee. Unsupervised end-to-end learning of discrete linguistic units for voice conversion. In Proc. Interspeech 2019, pages 1108–1112, 2019.
- [26] Chung-Ming Chien, Jheng-Hao Lin, Chien-yu Huang, Po-chun Hsu, and Hung-yi Lee. Investigating on incorporating pretrained and learnable speaker representations for multi-speaker multi-style text-to-speech. In ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 8588–8592. IEEE, 2021.
- [27] Kaizhi Qian, Yang Zhang, Shiyu Chang, Xuesong Yang, and Mark Hasegawa-Johnson. Autovc: Zero-shot voice style transfer with only autoencoder loss. In International Conference on Machine Learning, pages 5210–5219. PMLR, 2019.
- [28] Zhen-Hua Ling, Yang Ai, Yu Gu, and Li-Rong Dai. Waveform modeling and generation using hierarchical recurrent neural networks for speech bandwidth extension. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 26(5):883–894, 2018.
- [29] W Bastiaan Kleijn, Andrew Storus, Michael Chinen, Tom Denton, Felicia SC Lim, Alejandro Luebs, Jan Skoglund, and Hengchin Yeh. Generative speech coding with predictive variance regularization. In ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 6478–6482. IEEE, 2021.
- [30] Adam Polyak, Yossi Adi, Jade Copet, Eugene Kharonov, Kushal Lakhotia, Wei-Ning Hsu, Abdelrahman Mohamed, and Emmanuel Dupoux. Speech resynthesis

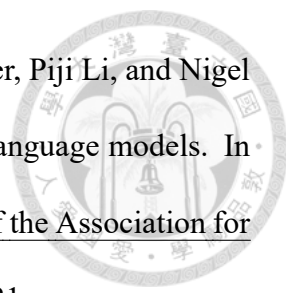
- from discrete disentangled self-supervised representations. In Proc. Interspeech 2021, pages 3615–3619, 2021.
- [31] Jaime Lorenzo-Trueba, Thomas Drugman, Javier Latorre, Thomas Merritt, Bartosz Putrycz, Roberto Barra-Chicote, Alexis Moinet, and Vatsal Aggarwal. Towards achieving robust universal neural vocoding. In Proc. Interspeech 2019, pages 181–185, 2019.
- [32] Po-chun Hsu, Chun-hsuan Wang, Andy T Liu, and Hung-yi Lee. Towards robust neural vocoding for speech generation: A survey. arXiv preprint arXiv:1912.02461, 2019.
- [33] Yunlong Jiao, Adam Gabryś, Georgi Tinchev, Bartosz Putrycz, Daniel Korzekwa, and Viacheslav Klimkov. Universal neural vocoding with parallel wavenet. In ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 6044–6048. IEEE, 2021.
- [34] Aaron Van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with pixelcnn decoders. Advances in neural information processing systems, 29, 2016.
- [35] Ryuichi Yamamoto, Eunwoo Song, and Jae-Min Kim. Parallel wavegan: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram. In ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 6199–6203. IEEE, 2020.
- [36] Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. Diffwave:

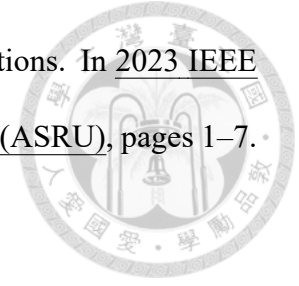
- 
- A versatile diffusion model for audio synthesis. In International Conference on Learning Representations, 2021.
- [37] Sandeep Kumar Pandey, Hanumant Singh Shekhawat, and SRM Prasanna. Emotion recognition from raw speech using wavenet. In TENCON 2019-2019 IEEE Region 10 Conference (TENCON), pages 1292–1297. IEEE, 2019.
- [38] Alice Coucke, Mohammed Chlieh, Thibault Gisselbrecht, David Leroy, Mathieu Poumeyrol, and Thibaut Lavril. Efficient keyword spotting using dilated convolutions and gating. In ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 6351–6355. IEEE, 2019.
- [39] Hui Wang, Fei Gao, Yue Zhao, and Licheng Wu. Wavenet with cross-attention for audiovisual speech recognition. IEEE Access, 8:169160–169168, 2020.
- [40] 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), pages 2251–2255. IEEE, 2018.
- [41] Jean-Marc Valin and Jan Skoglund. Lpcnet: Improving neural speech synthesis through linear prediction. In ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 5891–5895. IEEE, 2019.
- [42] Ravichander Vipperla, Sangjun Park, Kihyun Choo, Samin Ishtiaq, Kyoungbo Min, Sourav Bhattacharya, Abhinav Mehrotra, Alberto Gil C.P. Ramos, and Nicholas D.

- 
- Lane. Bunched lpcnet: Vocoder for low-cost neural text-to-speech systems. In Proc. Interspeech 2020, pages 3565–3569, 2020.
- [43] Hiroki Kanagawa and Yusuke Ijima. Lightweight lpcnet-based neural vocoder with tensor decomposition. In INTERSPEECH, pages 205–209, 2020.
- [44] Takuma Okamoto, Kentaro Tachibana, Tomoki Toda, Yoshinori Shiga, and Hisashi Kawai. Subband wavenet with overlapped single-sideband filterbanks. In 2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), pages 698–704. IEEE, 2017.
- [45] Takuma Okamoto, Tomoki Toda, Yoshinori Shiga, and Hisashi Kawai. Improving fftnet vocoder with noise shaping and subband approaches. In 2018 IEEE Spoken Language Technology Workshop (SLT), pages 304–311. IEEE, 2018.
- [46] Chengzhu Yu, Heng Lu, Na Hu, Meng Yu, Chao Weng, Kun Xu, Peng Liu, Deyi Tuo, Shiyin Kang, Guangzhi Lei, Dan Su, and Dong Yu. Durian: Duration informed attention network for speech synthesis. In Proc. Interspeech 2020, pages 2027–2031, 2020.
- [47] Qiao Tian, Zewang Zhang, Heng Lu, Ling-Hui Chen, and Shan Liu. Featherwave: An efficient high-fidelity neural vocoder with multi-band linear prediction. In Proc. Interspeech 2020, pages 195–199, 2020.
- [48] Yang Cui, Xi Wang, Lei He, and Frank K. Soong. An efficient subband linear prediction for lpcnet-based neural synthesis. In Proc. Interspeech 2020, pages 3555–3559, 2020.
- [49] Aaron Oord, Yazhe Li, Igor Babuschkin, Karen Simonyan, Oriol Vinyals, Koray Kavukcuoglu, George Driessche, Edward Lockhart, Luis Cobo, Florian Stim-

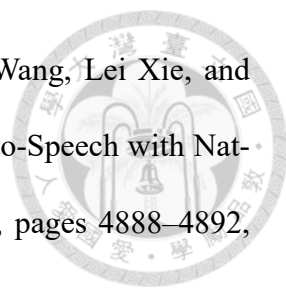
- 
- berg, et al. Parallel wavenet: Fast high-fidelity speech synthesis. In International conference on machine learning, pages 3918–3926. PMLR, 2018.
- [50] Wei Ping, Kainan Peng, and Jitong Chen. Clarinet: Parallel wave generation in end-to-end text-to-speech. In International Conference on Learning Representations, 2019.
- [51] Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. In Advances in Neural Information Processing Systems, volume 29, 2016.
- [52] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. Advances in neural information processing systems, 31, 2018.
- [53] Kundan Kumar, Rithesh Kumar, Thibault De Boissiere, Lucas Gestein, Wei Zhen Teoh, Jose Sotelo, Alexandre de Brébisson, Yoshua Bengio, and Aaron C Courville. Melgan: Generative adversarial networks for conditional waveform synthesis. Advances in neural information processing systems, 32, 2019.
- [54] Nanxin Chen, Yu Zhang, Heiga Zen, Ron J Weiss, Mohammad Norouzi, and William Chan. Wavegrad: Estimating gradients for waveform generation. In International Conference on Learning Representations, 2021.
- [55] Sungwon Kim, Sang-Gil Lee, Jongyoon Song, Jaehyeon Kim, and Sungroh Yoon. Flowwavenet: A generative flow for raw audio. In International Conference on Machine Learning, pages 3370–3378. PMLR, 2019.
- [56] Wei Ping, Kainan Peng, Kexin Zhao, and Zhao Song. Waveflow: A compact flow-based model for raw audio. In International Conference on Machine Learning, pages 7706–7716. PMLR, 2020.

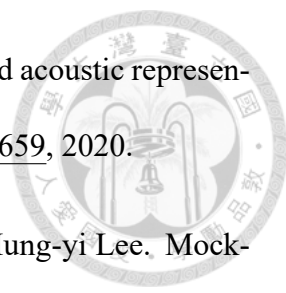
- 
- [57] Po-chun Hsu and Hung-yi Lee. Wg-wavenet: Real-time high-fidelity speech synthesis without gpu. Proc. Interspeech 2020, pages 210–214, 2020.
- [58] Geng Yang, Shan Yang, Kai Liu, Peng Fang, Wei Chen, and Lei Xie. Multi-band melgan: Faster waveform generation for high-quality text-to-speech. In 2021 IEEE Spoken Language Technology Workshop (SLT), pages 492–498. IEEE, 2021.
- [59] Eunwoo Song, Ryuichi Yamamoto, Min-Jae Hwang, Jin-Seob Kim, Ohsung Kwon, and Jae-Min Kim. Improved parallel wavegan vocoder with perceptually weighted spectrogram loss. In 2021 IEEE Spoken Language Technology Workshop (SLT), pages 470–476. IEEE, 2021.
- [60] Kentaro Tachibana, Tomoki Toda, Yoshinori Shiga, and Hisashi Kawai. An investigation of noise shaping with perceptual weighting for wavenet-based speech generation. In 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 5664–5668. IEEE, 2018.
- [61] Alexey Gritsenko, Tim Salimans, Rianne van den Berg, Jasper Snoek, and Nal Kalchbrenner. A spectral energy distance for parallel speech synthesis. Advances in Neural Information Processing Systems, 33:13062–13072, 2020.
- [62] Jiatao Gu, James Bradbury, Caiming Xiong, Victor OK Li, and Richard Socher. Non-autoregressive neural machine translation. In International Conference on Learning Representations, 2018.
- [63] Jason Lee, Elman Mansimov, and Kyunghyun Cho. Deterministic non-autoregressive neural sequence modeling by iterative refinement. In 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018, pages 1173–1182. Association for Computational Linguistics, 2018.

- 
- [64] Yixuan Su, Deng Cai, Yan Wang, David Vandyke, Simon Baker, Piji Li, and Nigel Collier. Non-autoregressive text generation with pre-trained language models. In Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, pages 234–243, 2021.
- [65] Yisheng Xiao, Lijun Wu, Junliang Guo, Juntao Li, Min Zhang, Tao Qin, and Tie-yan Liu. A survey on non-autoregressive generation for neural machine translation and beyond. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2023.
- [66] Yi Ren, Jinglin Liu, Xu Tan, Zhou Zhao, Sheng Zhao, and Tie-Yan Liu. A study of non-autoregressive model for sequence generation. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 149–159, 2020.
- [67] Lijun Wu, Xu Tan, Di He, Fei Tian, Tao Qin, Jianhuang Lai, and Tie-Yan Liu. Beyond error propagation in neural machine translation: Characteristics of language also matter. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 3602–3611, 2018.
- [68] Minchan Kim, Sung Jun Cheon, Byoung Jin Choi, Jong Jin Kim, and Nam Soo Kim. Expressive Text-to-Speech Using Style Tag. In Proc. Interspeech 2021, pages 4663–4667, 2021.
- [69] Reo Shimizu, Ryuichi Yamamoto, Masaya Kawamura, Yuma Shirahata, Tatsuya Komatsu, Kentaro Tachibana, et al. Prompttts++: Controlling speaker identity in prompt-based text-to-speech using natural language descriptions. arXiv preprint arXiv:2309.08140, 2023.
- [70] Yongmao Zhang, Guanghou Liu, Yi Lei, Yunlin Chen, Hao Yin, Lei Xie, and Zhifei

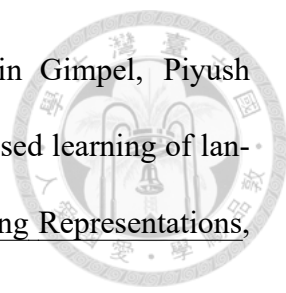



- Li. Promptspeaker: Speaker generation based on text descriptions. In 2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), pages 1–7. IEEE, 2023.
- [71] Wenhao Guan, Yishuang Li, Tao Li, Hukai Huang, Feng Wang, Jiayan Lin, Lingyan Huang, Lin Li, and Qingyang Hong. Mm-tts: Multi-modal prompt based style transfer for expressive text-to-speech synthesis. arXiv preprint arXiv:2312.10687, 2023.
- [72] Apoorv Vyas, Bowen Shi, Matthew Le, Andros Tjandra, Yi-Chiao Wu, Baishan Guo, Jiemin Zhang, Xinyue Zhang, Robert Adkins, William Ngan, et al. Audiobox: Unified audio generation with natural language prompts. arXiv preprint arXiv:2312.15821, 2023.
- [73] Yichong Leng, Zhifang Guo, Kai Shen, Xu Tan, Zeqian Ju, Yanqing Liu, Yufei Liu, Dongchao Yang, Leying Zhang, Kaitao Song, et al. Prompttts 2: Describing and generating voices with text prompt. arXiv preprint arXiv:2309.02285, 2023.
- [74] Hanglei Zhang, Yiwei Guo, Sen Liu, Xie Chen, and Kai Yu. Expressive tts driven by natural language prompts using few human annotations. arXiv preprint arXiv:2311.01260, 2023.
- [75] Dan Lyth and Simon King. Natural language guidance of high-fidelity text-to-speech with synthetic annotations. arXiv preprint arXiv:2402.01912, 2024.
- [76] Aya Watanabe, Shinnosuke Takamichi, Yuki Saito, Wataru Nakata, Detai Xin, and Hiroshi Saruwatari. Coco-nut: Corpus of japanese utterance and voice characteristics description for prompt-based control. In 2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), pages 1–8. IEEE, 2023.

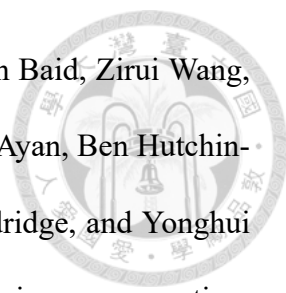
- 
- [77] Guanghou Liu, Yongmao Zhang, Yi Lei, Yunlin Chen, Rui Wang, Lei Xie, and Zhifei Li. PromptStyle: Controllable Style Transfer for Text-to-Speech with Natural Language Descriptions. In Proc. INTERSPEECH 2023, pages 4888–4892, 2023.
- [78] Benjamin Barras. Sox: Sound exchange, 2012.
- [79] Heiga Zen, Viet Dang, Rob Clark, Yu Zhang, Ron J. Weiss, Ye Jia, Zhifeng Chen, and Yonghui Wu. LibriTTS: A Corpus Derived from LibriSpeech for Text-to-Speech. In Proc. Interspeech 2019, pages 1526–1530, 2019.
- [80] Yuma Koizumi, Heiga Zen, Shigeki Karita, Yifan Ding, Kohei Yatabe, Nobuyuki Morioka, Michiel Bacchiani, Yu Zhang, Wei Han, and Ankur Bapna. LibriTTS-R: A Restored Multi-Speaker Text-to-Speech Corpus. In Proc. INTERSPEECH 2023, pages 5496–5500, 2023.
- [81] Yao Shi, Hui Bu, Xin Xu, Shaoji Zhang, and Ming Li. Aishell-3: A multi-speaker mandarin tts corpus and the baselines. arXiv preprint arXiv:2010.11567, 2020.
- [82] Tingwei Guo, Cheng Wen, Dongwei Jiang, Ne Luo, Ruixiong Zhang, Shuaijiang Zhao, Wubo Li, Cheng Gong, Wei Zou, Kun Han, et al. Didispeech: A large scale mandarin speech corpus. In ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 6968–6972. IEEE, 2021.
- [83] Abdelrahman Mohamed, Hung-yi Lee, Lasse Borgholt, Jakob D Havtorn, Joakim Edin, Christian Igel, Katrin Kirchhoff, Shang-Wen Li, Karen Livescu, Lars Maaløe, et al. Self-supervised speech representation learning: A review. IEEE Journal of Selected Topics in Signal Processing, 16(6):1179–1210, 2022.


- 
- [84] Shaoshi Ling and Yuzong Liu. Decoar 2.0: Deep contextualized acoustic representations with vector quantization. arXiv preprint arXiv:2012.06659, 2020.
- [85] Andy T Liu, Shu-wen Yang, Po-Han Chi, Po-chun Hsu, and Hung-yi Lee. Mockingjay: Unsupervised speech representation learning with deep bidirectional transformer encoders. In ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 6419–6423. IEEE, 2020.
- [86] Andy T Liu, Shang-Wen Li, and Hung-yi Lee. Tera: Self-supervised learning of transformer encoder representation for speech. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 29:2351–2366, 2021.
- [87] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. arXiv preprint arXiv:1807.03748, 2018.
- [88] Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli. wav2vec: Unsupervised Pre-Training for Speech Recognition. In Proc. Interspeech 2019, pages 3465–3469, 2019.
- [89] Alexei Baevski, Steffen Schneider, and Michael Auli. vq-wav2vec: Self-supervised learning of discrete speech representations. In International Conference on Learning Representations, 2020.
- [90] Yu-An Chung, Wei-Ning Hsu, Hao Tang, and James Glass. An Unsupervised Autoregressive Model for Speech Representation Learning. In Proc. Interspeech 2019, pages 146–150, 2019.
- [91] Yu-An Chung and James Glass. Generative pre-training for speech with autoregressive predictive coding. In ICASSP 2020-2020 IEEE International Conference


- 
- on Acoustics, Speech and Signal Processing (ICASSP), pages 3497–3501. IEEE, 2020.
- [92] Yu-An Chung, Hao Tang, and James Glass. Vector-Quantized Autoregressive Predictive Coding. In Proc. Interspeech 2020, pages 3760–3764, 2020.
- [93] Shaoshi Ling, Yuzong Liu, Julian Salazar, and Katrin Kirchhoff. Deep contextualized acoustic representations for semi-supervised speech recognition. In ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 6429–6433. IEEE, 2020.
- [94] Shu wen Yang, Po-Han Chi, Yung-Sung Chuang, Cheng-I Jeff Lai, Kushal Lakhotia, Yist Y. Lin, Andy T. Liu, Jiatong Shi, Xuankai Chang, Guan-Ting Lin, Tzu-Hsien Huang, Wei-Cheng Tseng, Ko tik Lee, Da-Rong Liu, Zili Huang, Shuyan Dong, Shang-Wen Li, Shinji Watanabe, Abdelrahman Mohamed, and Hung yi Lee. SUPERB: Speech Processing Universal PERFORMANCE Benchmark. In Proc. Interspeech 2021, pages 1194–1198, 2021.
- [95] Sanyuan Chen, Chengyi Wang, Zhengyang Chen, Yu Wu, Shujie Liu, Zhuo Chen, Jinyu Li, Naoyuki Kanda, Takuya Yoshioka, Xiong Xiao, et al. Wavlm: Large-scale self-supervised pre-training for full stack speech processing. IEEE Journal of Selected Topics in Signal Processing, 16(6):1505–1518, 2022.
- [96] 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. arXiv preprint arXiv:2001.05685, 2020.
- [97] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real NVP. In International Conference on Learning Representations, 2017.

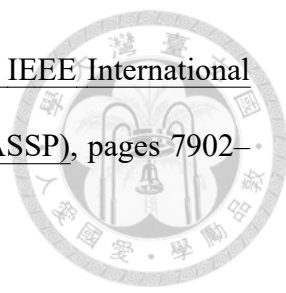
- 
- [98] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. In International Conference on Learning Representations, 2020.
- [99] Chao-I Tuan, Yuan-Kuei Wu, Hung-yi Lee, and Yu Tsao. Mitas: A compressed time-domain audio separation network with parameter sharing. arXiv preprint arXiv:1912.03884, 2019.
- [100] Francesc Lluís, Jordi Pons, and Xavier Serra. End-to-End Music Source Separation: Is it Possible in the Waveform Domain? In Proc. Interspeech 2019, pages 4619–4623, 2019.
- [101] Dario Rethage, Jordi Pons, and Xavier Serra. A wavenet for speech denoising. In 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 5069–5073. IEEE, 2018.
- [102] Shinji Takaki, Toru Nakashika, Xin Wang, and Junichi Yamagishi. Stft spectral loss for training a neural speech waveform model. In ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 7065–7069. IEEE, 2019.
- [103] Shinji Takaki, Hirokazu Kameoka, and Junichi Yamagishi. Training a neural speech waveform model using spectral losses of short-time fourier transform and continuous wavelet transform. arXiv preprint arXiv:1903.12392, 2019.
- [104] Sercan Ö Arık, Heewoo Jun, and Gregory Diamos. Fast spectrogram inversion using multi-head convolutional neural networks. IEEE Signal Processing Letters, 26(1):94–98, 2019.

- 
- [105] Keith Ito and Linda Johnson. The lj speech dataset. <https://keithito.com/LJ-Speech-Dataset/>, 2017.
- [106] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In International Conference on Learning Representations, 2015.
- [107] Robert Kubichek. Mel-cepstral distance measure for objective speech quality assessment. In Proceedings of IEEE Pacific Rim Conference on Communications Computers and Signal Processing, volume 1, pages 125–128. IEEE, 1993.
- [108] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. Advances in neural information processing systems, 30, 2017.
- [109] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. Advances in neural information processing systems, 33:1877–1901, 2020.
- [110] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 7871–7880, 2020.
- [111] Aaron Van Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. In International conference on machine learning, pages 1747–1756. PMLR, 2016.

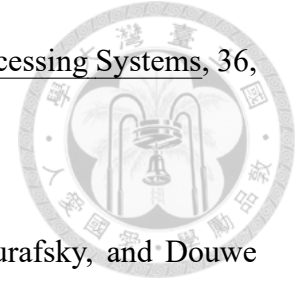
- 
- [112] Jiahui Yu, Yuanzhong Xu, Jing Yu Koh, Thang Luong, Gunjan Baid, Zirui Wang, Vijay Vasudevan, Alexander Ku, Yinfei Yang, Burcu Karagol Ayan, Ben Hutchinson, Wei Han, Zarana Parekh, Xin Li, Han Zhang, Jason Baldridge, and Yonghui Wu. Scaling autoregressive models for content-rich text-to-image generation. Transactions on Machine Learning Research, 2022.
- [113] Charlie Nash, Yaroslav Ganin, SM Ali Eslami, and Peter Battaglia. Polygen: An autoregressive generative model of 3d meshes. In International conference on machine learning, pages 7220–7229. PMLR, 2020.
- [114] Ji Ming, Peter Jancovic, and Francis Jack Smith. Robust speech recognition using probabilistic union models. IEEE Transactions on Speech and Audio Processing, 10(6):403–414, 2002.
- [115] James McAuley, Ji Ming, Darryl Stewart, and Philip Hanna. Subband correlation and robust speech recognition. IEEE Transactions on Speech and Audio Processing, 13(5):956–964, 2005.
- [116] Yinji Piao, HyunWook Park, et al. Image resolution enhancement using inter-subband correlation in wavelet domain. In 2007 IEEE International Conference on Image Processing, volume 1, pages I–445. IEEE, 2007.
- [117] Mohammed Imamul Hassan Bhuiyan and Anindya Bijoy Das. A subband correlation-based method for the automatic detection of epilepsy and seizure in the dual tree complex wavelet transform domain. In 2014 IEEE Conference on Biomedical Engineering and Sciences (IECBES), pages 811–816. IEEE, 2014.
- [118] T.Q. Nguyen. Near-perfect-reconstruction pseudo-qmf banks. IEEE Transactions on Signal Processing, 42(1):65–76, 1994.

- 
- [119] Jan Chorowski, Ron J Weiss, Samy Bengio, and Aäron van den Oord. Unsupervised speech representation learning using wavenet autoencoders. IEEE/ACM transactions on audio, speech, and language processing, 27(12):2041–2053, 2019.
- [120] Diganta Misra. Mish: A self regularized non-monotonic activation function. In 31st British Machine Vision Conference (BMVC 2020). British Machine Vision Association, 2020.
- [121] Chris S Xie. Dynamic vs static autoregressive models for forecasting time series. Available at SSRN 1268910, 2008.
- [122] Rajesh Wadhvani et al. Review on various models for time series forecasting. In 2017 International Conference on Inventive Computing and Informatics (ICICI), pages 405–410. IEEE, 2017.
- [123] Qiao Tian, Yi Chen, Zewang Zhang, Heng Lu, Linghui Chen, Lei Xie, and Shan Liu. Tfgan: Time and frequency domain based generative adversarial network for high-fidelity speech synthesis. arXiv preprint arXiv:2011.12206, 2020.
- [124] Junichi Yamagishi, Christophe Veaux, Kirsten MacDonald, et al. Cstr vctk corpus: English multi-speaker corpus for cstr voice cloning toolkit, 2017.
- [125] John Kominek and Alan W Black. The cmu arctic speech databases. In Fifth ISCA workshop on speech synthesis, 2004.
- [126] Jinhyeok Yang, Junmo Lee, Youngik Kim, Hoon-Young Cho, and Injung Kim. Vocgan: A high-fidelity real-time vocoder with a hierarchically-nested adversarial network. In Proc. Interspeech 2020, pages 200–204, 2020.

- 
- [127] Less Wright. Ranger - a synergistic optimizer. <https://github.com/lessw2020/Ranger-Deep-Learning-Optimizer>, 2019.
- [128] Matthias Mauch and Simon Dixon. pyin: A fundamental frequency estimator using probabilistic threshold distributions. In 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 659–663. IEEE, 2014.
- [129] ITUR Recommendation. Method for the subjective assessment of intermediate sound quality (mushra). ITU, BS, pages 1543–1, 2001.
- [130] Thomas Merritt, Bartosz Putrycz, Adam Nadolski, Tianjun Ye, Daniel Korzekwa, Wiktor Dolecki, Thomas Drugman, Viacheslav Klimkov, Alexis Moinet, Andrew Breen, et al. Comprehensive evaluation of statistical speech waveform synthesis. In 2018 IEEE Spoken Language Technology Workshop (SLT), pages 325–331. IEEE, 2018.
- [131] Marius Cotescu, Thomas Drugman, Goeric Huybrechts, Jaime Lorenzo-Trueba, and Alexis Moinet. Voice conversion for whispered speech synthesis. IEEE Signal Processing Letters, 27:186–190, 2019.
- [132] Javier Latorre, Jakub Lachowicz, Jaime Lorenzo-Trueba, Thomas Merritt, Thomas Drugman, Srikanth Ronanki, and Viacheslav Klimkov. Effect of data reduction on sequence-to-sequence neural tts. In ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 7075–7079. IEEE, 2019.
- [133] Adam Gabryś, Goeric Huybrechts, Manuel Sam Ribeiro, Chung-Ming Chien, Julian Roth, Giulia Comini, Roberto Barra-Chicote, Bartek Perz, and Jaime Lorenzo-Trueba. Voice filter: Few-shot text-to-speech speaker adaptation using voice con-

- 
- version as a post-processing module. In ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 7902–7906. IEEE, 2022.
- [134] Kamil Deja, Ariadna Sanchez, Julian Roth, and Marius Cotescu. Automatic Evaluation of Speaker Similarity. In Proc. Interspeech 2022, pages 2348–2352, 2022.
- [135] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. Advances in neural information processing systems, 27, 2014.
- [136] EH Rothausser. Ieee recommended practice for speech quality measurements. IEEE Trans. on Audio and Electroacoustics, 17:225–246, 1969.
- [137] Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. Advances in neural information processing systems, 30, 2017.
- [138] Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. Learning to summarize with human feedback. Advances in Neural Information Processing Systems, 33:3008–3021, 2020.
- [139] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. Advances in neural information processing systems, 35:27730–27744, 2022.
- [140] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is

secretly a reward model. Advances in Neural Information Processing Systems, 36, 2024.



[141] Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. Kto: Model alignment as prospect theoretic optimization. arXiv preprint arXiv:2402.01306, 2024.

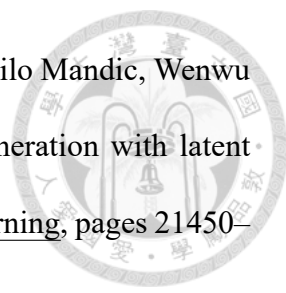
[142] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In International conference on machine learning, pages 8748–8763. PMLR, 2021.


[143] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 10684–10695, 2022.

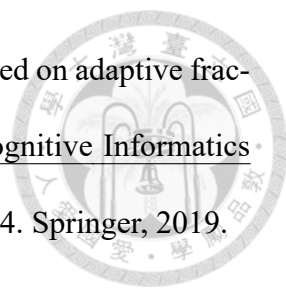
[144] Dongchao Yang, Jianwei Yu, Helin Wang, Wen Wang, Chao Weng, Yuexian Zou, and Dong Yu. Diffsound: Discrete diffusion model for text-to-sound generation. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 2023.

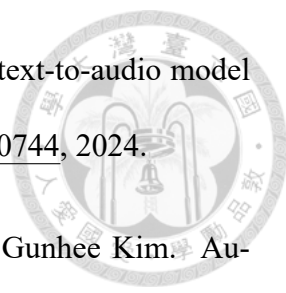
[145] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. Advances in neural information processing systems, 30, 2017.

[146] Rongjie Huang, Jiawei Huang, Dongchao Yang, Yi Ren, Luping Liu, Mingze Li, Zhenhui Ye, Jinglin Liu, Xiang Yin, and Zhou Zhao. Make-an-audio: Text-to-audio generation with prompt-enhanced diffusion models. In International Conference on Machine Learning, pages 13916–13932. PMLR, 2023.

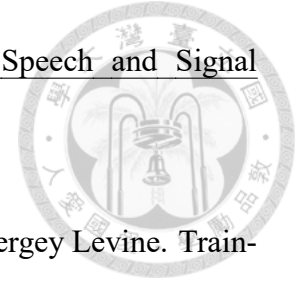
- 
- [147] Haohe Liu, Zehua Chen, Yi Yuan, Xinhao Mei, Xubo Liu, Danilo Mandic, Wenwu Wang, and Mark D Plumbley. Audioldm: Text-to-audio generation with latent diffusion models. In International Conference on Machine Learning, pages 21450–21474. PMLR, 2023.
- [148] Deepanway Ghosal, Navonil Majumder, Ambuj Mehrish, and Soujanya Poria. Text-to-audio generation using instruction guided latent diffusion model. In Proceedings of the 31st ACM International Conference on Multimedia, pages 3590–3598, 2023.
- [149] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling instruction-finetuned language models. arXiv preprint arXiv:2210.11416, 2022.
- [150] Haohe Liu, Qiao Tian, Yi Yuan, Xubo Liu, Xinhao Mei, Qiuqiang Kong, Yuping Wang, Wenwu Wang, Yuxuan Wang, and Mark D Plumbley. Audioldm 2: Learning holistic audio generation with self-supervised pretraining. arXiv preprint arXiv:2308.05734, 2023.
- [151] Siddique Latif, Heriberto Cuayáhuitl, Farrukh Pervez, Fahad Shamshad, Hafiz Shehbaz Ali, and Erik Cambria. A survey on deep reinforcement learning for audio-based applications. Artificial Intelligence Review, 56(3):2193–2240, 2023.
- [152] Taku Kala and Takahiro Shinozaki. Reinforcement learning of speech recognition system based on policy gradient and hypothesis selection. In 2018 IEEE international conference on acoustics, speech and signal processing (ICASSP), pages 5759–5763. IEEE, 2018.
- [153] Andros Tjandra, Sakriani Sakti, and Satoshi Nakamura. Sequence-to-sequence asr

- 
- optimization via reinforcement learning. In 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 5829–5833. IEEE, 2018.
- [154] Shigeki Karita, Atsunori Ogawa, Marc Delcroix, and Tomohiro Nakatani. Sequence training of encoder-decoder model using policy gradient for end-to-end speech recognition. In 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 5839–5843. IEEE, 2018.
- [155] Yingbo Zhou, Caiming Xiong, and Richard Socher. Improving end-to-end speech recognition with policy learning. In 2018 IEEE international conference on acoustics, speech and signal processing (ICASSP), pages 5819–5823. IEEE, 2018.
- [156] Andros Tjandra, Sakriani Sakti, and Satoshi Nakamura. End-to-end speech recognition sequence training with reinforcement learning. IEEE Access, 7:79758–79769, 2019.
- [157] Hoon Chung, Hyeong-Bae Jeon, and Jeon Gue Park. Semi-supervised training for sequence-to-sequence speech recognition using reinforcement learning. In 2020 international joint conference on neural networks (IJCNN), pages 1–6. IEEE, 2020.
- [158] Mathieu Seurin, Florian Strub, Philippe Preux, and Olivier Pietquin. A Machine of Few Words: Interactive Speaker Recognition with Reinforcement Learning. In Proc. Interspeech 2020, pages 4323–4327, 2020.
- [159] Egor Lakomkin, Mohammad Ali Zamani, Cornelius Weber, Sven Magg, and Stefan Wermter. Emorl: continuous acoustic emotion classification using deep reinforcement learning. In 2018 IEEE International Conference on Robotics and Automation (ICRA), pages 4445–4450. IEEE, 2018.

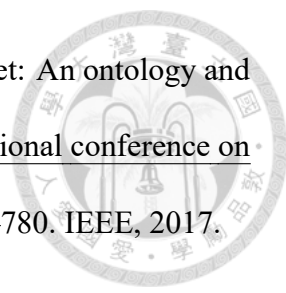
- 
- [160] J Sangeetha and T Jayasankar. Emotion speech recognition based on adaptive fractional deep belief network and reinforcement learning. In Cognitive Informatics and Soft Computing: Proceeding of CISC 2017, pages 165–174. Springer, 2019.
- [161] Yuma Koizumi, Kenta Niwa, Yusuke Hioka, Kazunori Kobayashi, and Yoichi Haneda. Dnn-based source enhancement to increase objective sound quality assessment score. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 26(10):1780–1792, 2018.
- [162] Yih-Liang Shen, Chao-Yuan Huang, Syu-Siang Wang, Yu Tsao, Hsin-Min Wang, and Tai-Shih Chi. Reinforcement learning based speech enhancement for robust speech recognition. In ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 6750–6754. IEEE, 2019.
- [163] Rui Liu, Berrak Sisman, and Haizhou Li. Reinforcement Learning for Emotional Text-to-Speech Synthesis with Improved Emotion Discriminability. In Proc. Interspeech 2021, pages 4648–4652, 2021.
- [164] Dong Zhang, Zhaowei Li, Shimin Li, Xin Zhang, Pengyu Wang, Yaqian Zhou, and Xipeng Qiu. Speechalign: Aligning speech generation to human preferences. arXiv preprint arXiv:2404.05600, 2024.
- [165] Navonil Majumder, Chia-Yu Hung, Deepanway Ghosal, Wei-Ning Hsu, Rada Mihalcea, and Soujanya Poria. Tango 2: Aligning diffusion-based text-to-audio generations through direct preference optimization. arXiv preprint arXiv:2404.09956, 2024.
- [166] Huan Liao, Haonan Han, Kai Yang, Tianjiao Du, Rui Yang, Zunnan Xu, Qinmei

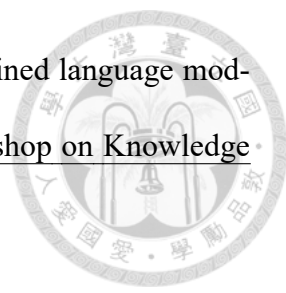
- 
- Xu, Jingquan Liu, Jiasheng Lu, and Xiu Li. Baton: Aligning text-to-audio model with human preference feedback. [arXiv preprint arXiv:2402.00744](https://arxiv.org/abs/2402.00744), 2024.
- [167] Chris Dongjoo Kim, Byeongchang Kim, Hyunmin Lee, and Gunhee Kim. Audiocaps: Generating captions for audios in the wild. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 119–132, 2019.
- [168] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. Advances in neural information processing systems, 33:6840–6851, 2020.
- [169] Ehsan Variani, Xin Lei, Erik McDermott, Ignacio Lopez Moreno, and Javier Gonzalez-Dominguez. Deep neural networks for small footprint text-dependent speaker verification. In 2014 IEEE international conference on acoustics, speech and signal processing (ICASSP), pages 4052–4056. IEEE, 2014.
- [170] Miquel Àngel India Massana, Pooyan Safari, and Francisco Javier Hernandez Pericás. Self multi-head attention for speaker recognition. In Interspeech 2019: the 20th Annual Conference of the International Speech Communication Association: 15-19 September 2019: Graz, Austria, pages 4305–4309. International Speech Communication Association (ISCA), 2019.
- [171] Jonathan Shen, Ye Jia, Mike Chrzanowski, Yu Zhang, Isaac Elias, Heiga Zen, and Yonghui Wu. Non-attentive tacotron: Robust and controllable neural tts synthesis including unsupervised duration modeling. [arXiv preprint arXiv:2010.04301](https://arxiv.org/abs/2010.04301), 2020.
- [172] Benjamin Elizalde, Soham Deshmukh, Mahmoud Al Ismail, and Huaming Wang. Clap learning audio concepts from natural language supervision. In ICASSP

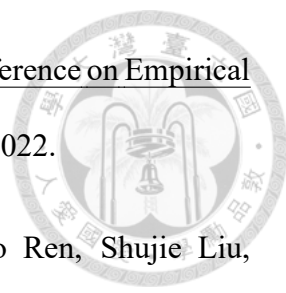
2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 1–5. IEEE, 2023.

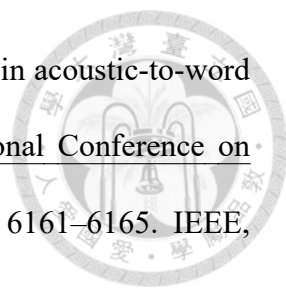


- [173] Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. In The Twelfth International Conference on Learning Representations, 2024.
- [174] Sham Kakade and John Langford. Approximately optimal approximate reinforcement learning. In Proceedings of the Nineteenth International Conference on Machine Learning, pages 267–274, 2002.
- [175] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347, 2017.
- [176] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In International conference on machine learning, pages 1889–1897. PMLR, 2015.
- [177] Adaeze Adigwe, Noé Tits, Kevin El Haddad, Sarah Ostadabbas, and Thierry Dutoit. The emotional voices database: Towards controlling the emotion dimension in voice generation systems. arXiv preprint arXiv:1806.09514, 2018.
- [178] Kun Zhou, Berrak Sisman, Rui Liu, and Haizhou Li. Emotional voice conversion: Theory, databases and esd. Speech Communication, 137:1–18, 2022.
- [179] Houwei Cao, David G Cooper, Michael K Keutmann, Ruben C Gur, Ani Nenkova, and Ragini Verma. Crema-d: Crowd-sourced emotional multimodal actors dataset. IEEE transactions on affective computing, 5(4):377–390, 2014.
- [180] Jort F Gemmeke, Daniel PW Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence,

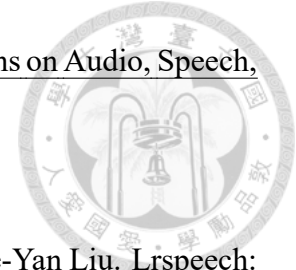
- 
- R Channing Moore, Manoj Plakal, and Marvin Ritter. Audio set: An ontology and human-labeled dataset for audio events. In 2017 IEEE international conference on acoustics, speech and signal processing (ICASSP), pages 776–780. IEEE, 2017.
- [181] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In International Conference on Learning Representations, 2021.
- [182] Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In International Conference on Learning Representations, 2022.
- [183] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In International Conference on Learning Representations, 2019.
- [184] Edresson Casanova, Julian Weber, Christopher D Shulby, Arnaldo Candido Junior, Eren Gölge, and Moacir A Ponti. Yourtts: Towards zero-shot multi-speaker tts and zero-shot voice conversion for everyone. In International Conference on Machine Learning, pages 2709–2720. PMLR, 2022.
- [185] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 15750–15758, 2021.
- [186] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent—a new approach to self-supervised learning. Advances in neural information processing systems, 33:21271–21284, 2020.

- 
- [187] Jie Huang, Hanyin Shao, and Kevin Chang. Are large pre-trained language models leaking your personal information? In ICML 2022 Workshop on Knowledge Retrieval and Language Models, 2022.
- [188] Xudong Pan, Mi Zhang, Shouling Ji, and Min Yang. Privacy risks of general-purpose language models. In 2020 IEEE Symposium on Security and Privacy (SP), pages 1314–1331. IEEE, 2020.
- [189] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. arXiv preprint arXiv:2107.03374, 2021.
- [190] Wei-Cheng Tseng, Wei-Tsung Kao, and Hung yi Lee. Membership Inference Attacks Against Self-supervised Speech Models. In Proc. Interspeech 2022, pages 5040–5044, 2022.
- [191] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: an asr corpus based on public domain audio books. In 2015 IEEE international conference on acoustics, speech and signal processing (ICASSP), pages 5206–5210. IEEE, 2015.
- [192] Zhehuai Chen, Yu Zhang, Andrew Rosenberg, Bhuvana Ramabhadran, Gary Wang, and Pedro Moreno. Injecting text in self-supervised speech pretraining. In 2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), pages 251–258. IEEE, 2021.
- [193] Ziqiang Zhang, Long Zhou, Junyi Ao, Shujie Liu, Lirong Dai, Jinyu Li, and Furu Wei. Speechcut: Bridging speech and text with hidden-unit for encoder-decoder

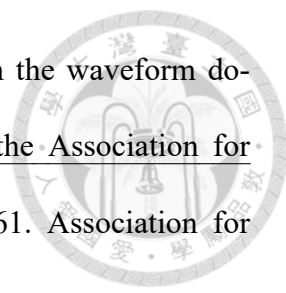
- 
- based speech-text pre-training. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 1663–1676, 2022.
- [194] Ziqiang Zhang, Sanyuan Chen, Long Zhou, Yu Wu, Shuo Ren, Shujie Liu, Zhuoyuan Yao, Xun Gong, Lirong Dai, Jinyu Li, et al. Speechlm: Enhanced speech pre-training with unpaired textual data. arXiv preprint arXiv:2209.15329, 2022.
- [195] Ankur Bapna, Yu-an Chung, Nan Wu, Anmol Gulati, Ye Jia, Jonathan H Clark, Melvin Johnson, Jason Riesa, Alexis Conneau, and Yu Zhang. Slam: A unified encoder for speech and language modeling via speech-text joint pre-training. arXiv preprint arXiv:2110.10329, 2021.
- [196] Junyi Ao, Rui Wang, Long Zhou, Chengyi Wang, Shuo Ren, Yu Wu, Shujie Liu, Tom Ko, Qing Li, Yu Zhang, et al. Specht5: Unified-modal encoder-decoder pre-training for spoken language processing. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 5723–5738, 2022.
- [197] Zhehuai Chen, Yu Zhang, Andrew Rosenberg, Bhuvana Ramabhadran, Pedro J. Moreno, Ankur Bapna, and Heiga Zen. MAESTRO: Matched Speech Text Representations through Modality Matching. In Proc. Interspeech 2022, pages 4093–4097, 2022.
- [198] Masato Mimura, Sei Ueno, Hirofumi Inaguma, Shinsuke Sakai, and Tatsuya Kawahara. Leveraging sequence-to-sequence speech synthesis for enhancing acoustic-to-word speech recognition. In 2018 IEEE Spoken Language Technology Workshop (SLT), pages 477–484. IEEE, 2018.
- [199] Sei Ueno, Masato Mimura, Shinsuke Sakai, and Tatsuya Kawahara. Multi-speaker


- 
- sequence-to-sequence speech synthesis for data augmentation in acoustic-to-word speech recognition. In ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 6161–6165. IEEE, 2019.
- [200] Nick Rossenbach, Albert Zeyer, Ralf Schlüter, and Hermann Ney. Generating synthetic audio data for attention-based speech recognition systems. In ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 7069–7073. IEEE, 2020.
- [201] Sei Ueno, Masato Mimura, Shinsuke Sakai, and Tatsuya Kawahara. Data augmentation for asr using tts via a discrete representation. In 2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), pages 68–75. IEEE, 2021.
- [202] Xianrui Zheng, Yulan Liu, Deniz Gunceler, and Daniel Willett. Using synthetic audio to improve the recognition of out-of-vocabulary words in end-to-end asr systems. In ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 5674–5678. IEEE, 2021.
- [203] Amin Fazel, Wei Yang, Yulan Liu, Roberto Barra-Chicote, Yixiong Meng, Roland Maas, and Jasha Droppo. SynthASR: Unlocking Synthetic Data for Speech Recognition. In Proc. Interspeech 2021, pages 896–900, 2021.
- [204] Manuel Giollo, Deniz Gunceler, Yulan Liu, and Daniel Willett. Bootstrap an End-to-End ASR System by Multilingual Training, Transfer Learning, Text-to-Text Mapping and Synthetic Audio. In Proc. Interspeech 2021, pages 2416–2420, 2021.
- [205] Herman Kamper. Word segmentation on discovered phone units with dynamic pro-

gramming and self-supervised scoring. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 31:684–694, 2022.



- [206] Jin Xu, Xu Tan, Yi Ren, Tao Qin, Jian Li, Sheng Zhao, and Tie-Yan Liu. Lrspeech: Extremely low-resource speech synthesis and recognition. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pages 2802–2812, 2020.
- [207] Alexander H Liu, Tao Tu, Hung-yi Lee, and Lin-shan Lee. Towards unsupervised speech recognition and synthesis with quantized speech representation learning. In ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 7259–7263. IEEE, 2020.
- [208] Yi Ren, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. Almost unsupervised text to speech and automatic speech recognition. In International conference on machine learning, pages 5410–5419. PMLR, 2019.
- [209] David Snyder, Daniel Garcia-Romero, Gregory Sell, Daniel Povey, and Sanjeev Khudanpur. X-vectors: Robust dnn embeddings for speaker recognition. In 2018 IEEE international conference on acoustics, speech and signal processing (ICASSP), pages 5329–5333. IEEE, 2018.
- [210] Felix Kreuk, Adam Polyak, Jade Copet, Eugene Kharonov, Tu Anh Nguyen, Morgan Rivière, Wei-Ning Hsu, Abdelrahman Mohamed, Emmanuel Dupoux, and Yossi Adi. Textless speech emotion conversion using discrete & decomposed representations. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 11200–11214, 2022.

- 
- [211] Gallil Maimon and Yossi Adi. Speaking style conversion in the waveform domain using discrete self-supervised units. In Findings of the Association for Computational Linguistics: EMNLP 2023, pages 8048–8061. Association for Computational Linguistics, 2023.
- [212] Jacob Kahn, Morgane Riviere, Weiyi Zheng, Evgeny Kharitonov, Qiantong Xu, Pierre-Emmanuel Mazaré, Julien Karadayi, Vitaliy Liptchinsky, Ronan Collobert, Christian Fuegen, et al. Libri-light: A benchmark for asr with limited or no supervision. In ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 7669–7673. IEEE, 2020.
- [213] David Snyder, Guoguo Chen, and Daniel Povey. Musan: A music, speech, and noise corpus. arXiv preprint arXiv:1510.08484, 2015.
- [214] Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. In Proceedings of NAACL-HLT 2019: Demonstrations, 2019.
- [215] Tomoki Hayashi, Ryuichi Yamamoto, Katsuki Inoue, Takenori Yoshimura, Shinji Watanabe, Tomoki Toda, Kazuya Takeda, Yu Zhang, and Xu Tan. Espnet-tts: Unified, reproducible, and integratable open source end-to-end text-to-speech toolkit. In ICASSP 2020-2020 IEEE international conference on acoustics, speech and signal processing (ICASSP), pages 7654–7658. IEEE, 2020.
- [216] Po-Han Chi, Pei-Hung Chung, Tsung-Han Wu, Chun-Cheng Hsieh, Yen-Hao Chen, Shang-Wen Li, and Hung-yi Lee. Audio albert: A lite bert for self-supervised learning of audio representation. In 2021 IEEE Spoken Language Technology Workshop (SLT), pages 344–350. IEEE, 2021.

- 
- [217] Sangjun Park, Kihyun Choo, Joohyung Lee, Anton V. Porov, Konstantin Osipov, and June Sig Sung. Bunched LPCNet2: Efficient Neural Vocoders Covering Devices from Cloud to Edge. In Proc. Interspeech 2022, pages 808–812, 2022.
- [218] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In International Conference on Machine Learning, pages 28492–28518. PMLR, 2023.
- [219] Eliya Nachmani, Alon Levkovitch, Yifan Ding, Chulayuth Asawaroengchai, Heiga Zen, and Michelle Tadmor Ramanovich. Translatotron 3: Speech to speech translation with monolingual data. In ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 10686–10690. IEEE, 2024.
- [220] Paul K Rubenstein, Chulayuth Asawaroengchai, Duc Dung Nguyen, Ankur Bapna, Zalán Borsos, Félix de Chaumont Quitry, Peter Chen, Dalia El Badawy, Wei Han, Eugene Kharitonov, et al. Audiopalm: A large language model that can speak and listen. arXiv preprint arXiv:2306.12925, 2023.
- [221] Dong Zhang, Shimin Li, Xin Zhang, Jun Zhan, Pengyu Wang, Yaqian Zhou, and Xipeng Qiu. Speechnpt: Empowering large language models with intrinsic cross-modal conversational abilities. In The 2023 Conference on Empirical Methods in Natural Language Processing, 2023.
- [222] Qiantong Xu, Tatiana Likhomanenko, Jacob Kahn, Awni Hannun, Gabriel Synnaeve, and Ronan Collobert. Iterative Pseudo-Labeling for Speech Recognition. In Proc. Interspeech 2020, pages 1006–1010, 2020.
- [223] Daniel S. Park, Yu Zhang, Ye Jia, Wei Han, Chung-Cheng Chiu, Bo Li, Yonghui

Wu, and Quoc V. Le. Improved Noisy Student Training for Automatic Speech Recognition. In Proc. Interspeech 2020, pages 2817–2821, 2020.

