MambaMuse: An End-To-End Framework For Efficient One-Shot Timbre Transfer

Austin Pham (ap4460)
Columbia University
ap4460@columbia.edu

***Abstract*—This paper presents a novel U-Net-based architecture that integrates Adaptive Instance Normalization (AdaIN) and Mamba, a Selective State Space Model (SSM), for efficient musical timbre transfer. The proposed method addresses limitations in existing approaches, such as high computational demand, instability, and slow inference speed, by enabling real-time, one-shot timbre transfer with expressive and natural outputs. The integration of AdaIN allows user-driven customization of content-style blending, while Mamba efficiently captures long-term dependencies to maintain coherence over extended musical phrases. By operating directly in the time domain, as opposed to using spectrograms or encoded features, our approach aims to preserve fine-grained audio details for higher fidelity. We hope to demonstrate the model's performance through a comparative evaluation against state-of-the-art methods, highlighting its efficiency and versatility. The code is publicly available [1].**

***Index Terms*—Timbre transfer, state-space model, adaptive instance normalization, signal processing**

I. Introduction

The growing demand for creative audio manipulation has led to significant progress in the field of musical timbre transfer. Timbre transfer involves taking the unique stylistic characteristics of one audio signal and applying them to another. This process allows musicians and producers to transform the expressive and stylistic elements of audio, making it a powerful tool for generating new and unique compositions. As the demand for personalized high-quality audio experiences grows, it has become increasingly important to develop methods that can effectively capture detailed audio characteristics while also being adaptable in real time.

Current approaches to timbre transfer use a variety of deep learning methods, including Generative Adversarial Networks (GAN) [1], Variational Autoencoders (VAE) [2], and Diffusion Probabilistic Models [3]. Each of these methods has its own strengths, such as high-quality output and detailed control over features, but they also come with challenges. GANs, for example, are known for their ability to generate high-quality audio, but require a lot of computational power and can be unstable during training. Diffusion models, on the other hand, produce realistic audio with great detail, but their slow, step-by-step generation process makes them impractical for real-time use.

To address these challenges, we propose a U-Net-based architecture that integrates Adaptive Instance Normalization (AdaIN) for musical timbre transfer [4]. Inspired by the work of [5], the U-Net structure uses an encoder-decoder setup to extract and reconstruct features efficiently, while the AdaIN layers make it possible to adapt the timbral features of a style audio input in real time. Unlike GAN-based models, our approach supports one-shot timbre transfer, which means it can quickly adapt to new styles without needing extra training or fine-tuning. Additionally, the AdaIN integration allows for user-driven customization, giving users dynamic control over how much content and style blend together, which makes the model more versatile and easy to use in creative applications.

Furthermore, we incorporate the Mamba architecture within our U-Net framework to efficiently model long sequences. Mamba, a type of Selective State Space Model (SSM) [6], excels at capturing long-term dependencies in audio, which is essential for maintaining coherence over extended musical phrases. Additionally, Mamba allows for efficient linear-time sequence modeling, which allows us to work within the time domain.

The main contributions of this paper are as follows: (1) We introduce an efficient U-Net-based architecture for timbre transfer. (2) We use custom AdaIN layers to enable flexible, real-time style adaptation without the need for further training.

II. Related Works

A. Generative Adversarial Networks (GANs)

GAN-based approaches have shown notable success in generating high-quality timbre transformations. Models such as WaveGAN [7] and VAE-GAN type architecture [8] utilize adversarial training to synthesize realistic audio by learning from paired datasets of content and style inputs. GANs are particularly powerful at generating expressive and diverse audio outputs; however, they require significant computational resources and are known for their training instability due to the need for balancing the generator and discriminator networks. Additionally, GAN-based models often struggle to generalize to unseen timbres without further fine-tuning, limiting their adaptability for one-shot learning scenarios.

B. Variational Autoencoders (VAEs)

Variational Autoencoders are another class of generative models applied to timbre transfer [8]. VAEs learn a probabilistic latent representation of audio, allowing for smooth transitions between different timbres. These models are advantageous for their latent space representation, which supports interpolations and structured control over generated audio [9].

[1]Code: https://github.com/aphamm/mamba-muse

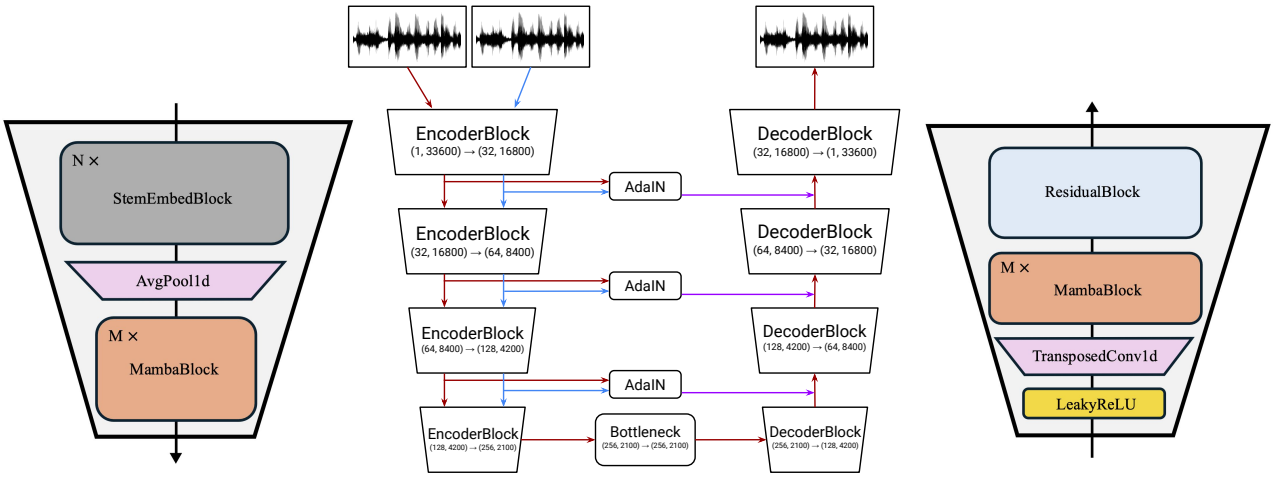


Fig. 1: The architecture of EncoderBlock (Left), MambaMuse (Middle), and DecoderBlock (Right).

However, VAEs tend to sacrifice audio fidelity due to the compression process, resulting in lower quality compared to GANs or diffusion models.

C. Diffusion Probabilistic Models

Diffusion models, like DiffWave [10], represent a newer approach to audio generation, capable of achieving very high-quality outputs by modeling the progressive refinement of noisy signals. These models have shown promise in generating highly realistic audio with rich temporal details. However, their inference process is inherently slow, as it involves iterative denoising over multiple steps. This makes them impractical for real-time applications, especially for dynamic timbre transfer tasks where low latency is critical. Moreover, diffusion models typically require substantial training time and data, making them less accessible for use with limited computational resources.

D. State Space Models

Recent advances in State Space Models (SSMs), such as the Mamba architecture, have demonstrated the ability to model long-term dependencies in sequential data effectively and efficiently [11]. The use of SSMs in audio applications has been driven by their capacity to capture global temporal structures without the quadratic complexity of traditional recurrent models [12]. Mamba, in particular, excels in maintaining coherence over extended sequences, which is crucial for musical timbre transfer where consistency across long phrases is desired.

E. Time Domain Speech Processing

Most previous works have focused on operating within the time-frequency domain using features like Mel-Frequency Cepstral Coefficients (MFCC) [13] and Power-Normalized Cepstral Coefficients (PNCC) [14]. Spectrogram-based methods are often used because they reduce the temporal resolution and make it easier for neural networks to focus on frequency-based transformations. However, the conversion process from waveform to spectrogram—and back—inevitably results in a loss of fine-grained information, which can adversely affect audio fidelity. However, time domain speech processing has been studied in various tasks including source separation [15], [16], representation learning [17], multimodal processing [18], speech generation [11], [19], and speech super-resolution [20]. Similarly, we propose a method for timbre transfer working directly in the time domain, allowing for the preservation of subtle details that are critical to producing a high-quality audio signal.

F. Adaptive Instance Normalization

AdaIN has been widely used in image style transfer for its ability to dynamically adapt the style of one image to another by aligning feature statistics. In the context of timbre transfer, AdaIN allows for intuitive blending between the content and style input signals, providing a semantically meaningful way for users to control the transfer process [4]. Unlike other conditioning methods that require pre-computed embeddings or complex attention mechanisms, AdaIN offers a lightweight

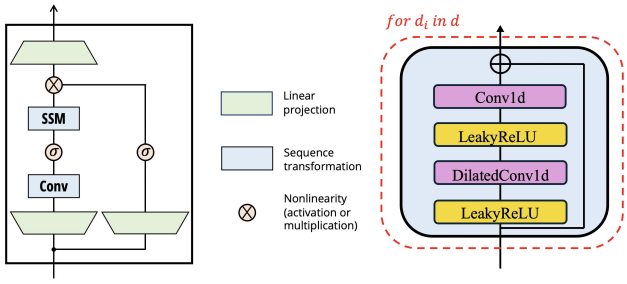


Fig. 2: The architecture of MambaBlock (Left) and ResidualBlock (Right)

and efficient solution that supports real-time, user-driven customization.

III. MambaMuse

MambaMuse uses a U-Net style autoencoder inspired by the work of Wave-U-Mamba [5]. The architecture takes raw waveform as an input, encodes it through a bottleneck representation and then attempts to reconstruct the original waveform. A residual connection is employed at every block to enhance stability [21]. We use Mamba as a global feature extractor as a means to efficiently and effectively learn long-range patterns. We use a MambaBlock that integrates Mamba with Layer Normalization (LN) and residual connections.

A. Encoding

To help stabilize Mamba training, we use a 1D version of stem embeddings to map input representations into a high-dimensional latent space [22]. The raw waveform is encoded through a series of 4 EncoderBlocks, which consist of a StemEmbeddingBlock, AvgPool1D and MambaBlock. The StemEmbeddingBlock consists of a 1D convolution with kernel size of 4 and stride of 1, followed by LN, LeakyReLU, and residual connections. Average pooling effectively halves the sequence length, allowing for a compression of features in a natural manner.

After pooling, the sequence is then processed through $N = 2$ MambaBlocks. This iterative downsampling process allows the MambaBlocks to model long-term dependencies at different temporal resolutions. After the EncoderBlocks, the sequence is processed through the bottleneck. It has the same core architecture, without the AvgPool1D and number of MambaBlocks set to $N = 3$.

B. Decoding

For the decoding process, we similarly employ 4 consecutive upsampling blocks that take in residual connections from each corresponding EncoderBlock of the same dimension. We use transposed convolution, in which kernel sizes are set as a multiple of strides to remove checkerboard artifacts common in transposed convolution [23]. Additionally, ResidualBlocks are used to improve the connectivity of representations, each with a kernel size of 3 with dilations set to $[1, 3]$. Finally, the sequence is passed through a convolution layer and $\tanh$ activation to predict the original input waveform. Additionally, all convolution layers are weight-normalized [24].

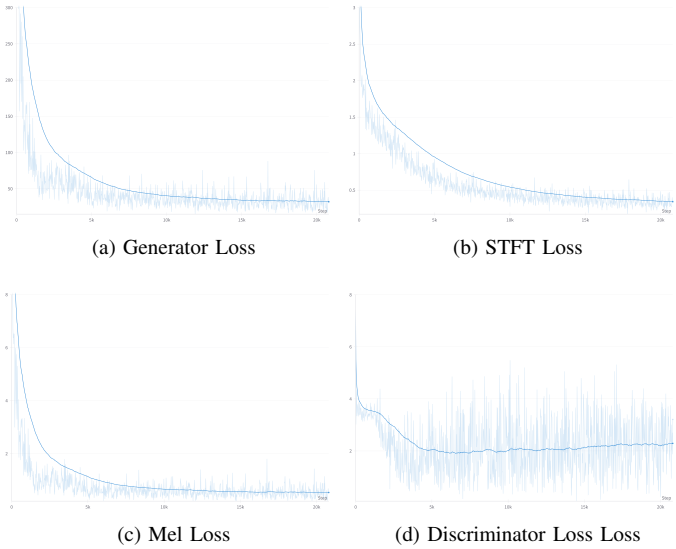
C. AdaIN

For timbre transfer, we applied AdaIN layers at strategic points in the U-Net architecture. During pre-training, only one input is passed through the AdaIN layer. The input sequence is just passed through normally, effectively ignoring the AdaIN layer. However, at inference time we simultaneously pass in two inputs through the AdaIN layer to align the feature statistics of content input to the style input. AdaIN layers are placed at each residual connection from EncoderBlock to its corresponding upsampling module. AdaIN receives a content input x and a style input y, and simply aligns the channelwise mean and variance of x to match those of y. There are no learnable affine parameters, but rather adaptively computes the affine parameters from the style input in the following fashion:

$$AdaIN(x, y) = \sigma(y)(\frac{x - \mu(x)}{\sigma(x)}) + \mu(y)$$

D. Discriminator

Following early works on incorporating GAN for vocoding, we use Multi-Period Discriminator (MPD) and Multi-Scale Discriminator (MSD) to guide adversarial training of MambaMuse. MPD first reshapes speech signal into 2D data and then applies 2D convolution to capture periodic patterns in the signal [25]. MPD consists of 5 sub-discriminators with different periods to reshape speech signals. MSD is a 1D convolution-based discriminator that operates on smoothed waveform [26]. MSD consists of 3 sub-discriminators that operate on raw waveform and two down-sampled waveforms. By



(a) Generator Loss (b) STFT Loss

(c) Mel Loss (d) Discriminator Loss Loss

Fig. 3: Training Loss with time weighted EMA smoothing

combining two modules, our generator is expected to capture both periodic patterns and continuous nature of speech signals, rendering natural and high-quality waveform generation.

E. Training Objective

1) Mel Spectrogram Loss: For the natural and realistic audio generation, it is crucial that perceptual quality be measured thoroughly. The $L1$-loss between mel spectrograms of the generated and original audio is used. In equation 1, Y and $\hat{Y}$ refer to the original waveform and the predicted waveform, respectively, where ψ refers to the mel transformation.

$$L_{\text{mel}} = \mathbb{E}[||\psi(Y) - \psi(\hat{Y})||_1] \tag{1}$$

2) Multi-Resolution Short-Time Fourier Transform (STFT) Loss: We incorporate Multi-Resolution Short-Time Fourier Transform (STFT) Loss [27] to encourage our model to learn detailed characteristics of high-frequency bands in the spectrogram, typically more perceptible to the human ear. Multi-Resolution STFT Loss is the sum of the STFT losses with different parameters (window size, hop length). Each STFT loss L_{s} is the sum of spectral convergence loss and log STFT magnitude loss. It also helps prevent learning the patterns of specific hyper-parameters used in STFT.

$$L_{\text{STFT}}(Y, \hat{Y}) = \mathbb{E}[L_{\text{s}}(Y, \hat{Y})] \tag{2}$$

3) GAN Loss: We use the same formulations for adversarial loss for both the generator and discriminator as in [25].

$$\begin{aligned} L_{\text{GAN}}(D; G) &= \mathbb{E}\left[(D(x) - 1)^2 + (D(G(s)))^2\right] \\ L_{\text{GAN}}(G; D) &= \mathbb{E}\left[(D(G(s)) - 1)^2\right] \end{aligned} \tag{3}$$

4) Final Loss: The final loss for our MambaMuse generator and discriminator are L_{Gen} and L_{Dis}, respectively.

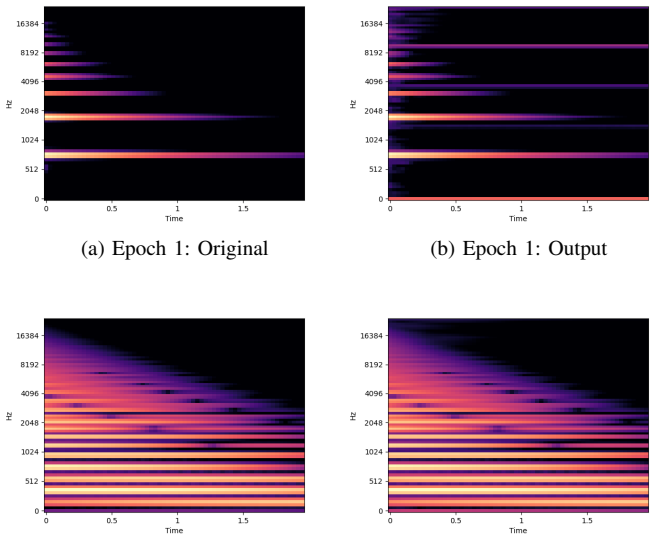
$$\begin{aligned} L_{\text{G}} &= \lambda_{\text{mel}} L_{\text{mel}} + \lambda_{\text{STFT}} L_{\text{STFT}} + L_{\text{GAN}}(G; D) \\ L_{\text{D}} &= L_{\text{GAN}}(D; G) \end{aligned} \tag{4}$$

We set $\lambda_{\text{mel}} = 45$ and $\lambda_{\text{STFT}} = 10$ to balance the values among loss terms. The $L1$-loss between the two waveform is not used since it does not lead to better perceptual quality not high-frequency components.

IV. Results

A. Training

To train the MambaMuse architecture, we use a subset of the NSynth to capture a diverse range of musical timbres and styles [28]. The audio dataset contains 305,979 musical notes, each with a unique pitch, timbre, and envelope. Data is sampled at $16kHz$ audio snippets, consisting of 1,006 instruments from commercial sample libraries. The data is preprocessed to remove all synthetic instruments and samples with a $velocity \leq 50$. We only kept random 2.1 second segments of audio starting within the first second to reduce the overall training computation requirements. At a sample rate of $16kHz$, this results in samples of length $33,600$. In total, we keep $41,600$, $3,200$, 128 total samples for a train, validation, test split. The audio is further normalized before processed in our model. The data is released here [2].



(a) Epoch 1: Original (b) Epoch 1: Output

(c) Epoch 15: Original (d) Epoch 15: Output

Fig. 4: Mel spectrogram through training. The top and bottom row show Epoch 1 and 15 respectively, The left and right column show the original signal and generated signal, respectively.

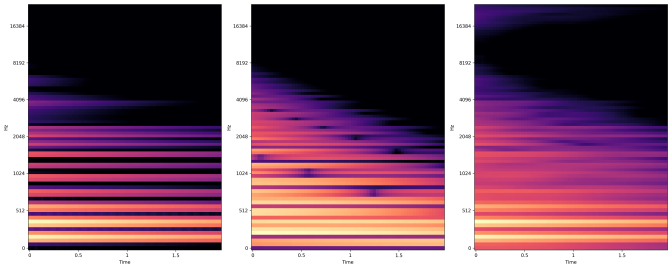
The model is trained for 25 epochs with early stopping to prevent overfitting. The AdamW optimizer is used with $\beta_1 = 0.6$ and $\beta_2 = 0.99$. The initial learning rate is set at $2e{-}4$ with a warmup of 5 epochs, followed by exponential weight decay of $\gamma = 0.999$. Training was done with a single NVIDIA A100 80GB with $batch\_size = 32$, which took roughly 6 hours to train.

B. Results

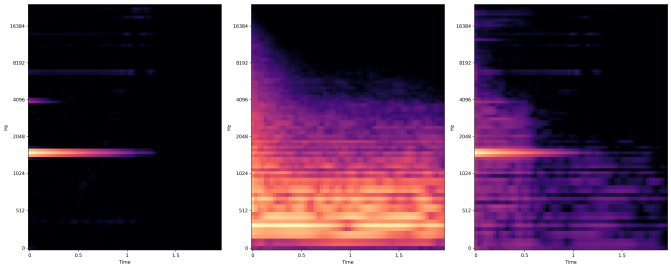
The results of our experiments demonstrate the potential of MambaMuse in enabling efficient one-shot timbre transfer. The model successfully incorporated the stylistic characteristics of the source timbre into the target audio in a one-shot fashion, showcasing its adaptability and speed. However, while the outputs were expressive and natural, they lacked some nuance in certain scenarios. For example, subtle timbral variations in the source material were not always captured with full fidelity, indicating room for improvement in preserving fine-grained stylistic details. Despite these limitations, the real-time inference capability and one-shot adaptability highlight the practicality of MambaMuse for creative applications. Further refinement and experimentation could enhance the nuance and depth of the generated outputs.

In figure 5, we see sample generation of our trained model on random inputs. The resulting mel spectrogram of the output

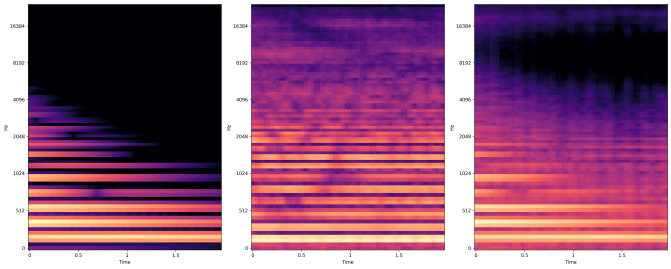
[2]Dataset: https://huggingface.co/datasets/aphamm/mamba-muse



(a) Example 1



(b) Example 2



(c) Example 3

Fig. 5: Model generation: Mel spectorgrams of the content (Left), style (middle) and output (right) waveform

waveform show a smooth blending between the content and style inputs.

C. *Efficiency*

The efficiency of MambaMuse is evident from the model's parameter and memory requirements. With a total trainable parameter count of $4.2\ M$, the architecture maintains a compact footprint suitable for real-time applications. Importantly, the computational cost is minimal, with total multiply-add operations amounting to only $0.02\ MB$. This low computational overhead makes the model accessible even on devices with limited processing capabilities.

In terms of memory usage, with a 2.1 second input length, we have an input size of $0.20\ MB$. The forward and backward pass together require $275.25\ MB$. This efficiency underscores the model's suitability for real-time processing, as it operates with minimal resource demands while maintaining high-quality outputs. Inference speed is measured using a single A100 GPU.

V. Future Work

Due to time constraints, model architecture design was not fully explored. Based on the audio pre-training, we can see the high fidelity reconstruction of the original signal with our existing MambaMuse architecture. However, the application to style transfer yields different results. Though it is efficient in processing two simultaneous samples, the output waveform lacks nuance and stylistic quality. More exploration on the position of AdaIN layers should be investigated. A possible explanation for this degenerate audio generation could be that AdaIN layers are only applied at residual connections. Thus, the content feature statistics are only aligned based on the features of the style passed through the encoder. However, the style is then not used in the decoder process, which could help align the content feature statistics during the upsampling process.

VI. Conclusion

In this paper, we presented MambaMuse, an end-to-end framework for efficient one-shot musical timbre transfer. By integrating Adaptive Instance Normalization (AdaIN) and Mamba into a U-Net-based architecture, we addressed key challenges in the field, including computational inefficiency, instability, and slow inference speeds. Our approach operates directly in the time domain to preserve fine-grained audio details, enabling real-time, expressive timbre transfer with user-driven customization.

While our results demonstrate the efficiency and versatility of MambaMuse, there remain avenues for further exploration. Ablation studies could investigate the optimal placement of AdaIN layers within the U-Net architecture to maximize performance and flexibility. Additionally, more robust evaluation metrics, beyond empirical examples, could provide a deeper understanding of the model's capabilities and limitations. For example, quantitative metrics for fidelity and coherence could complement subjective listening tests.

REFERENCES

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[2] D. P. Kingma, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[3] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Advances in neural information processing systems*, vol. 33, pp. 6840–6851, 2020.

[4] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *Proceedings of the IEEE international conference on computer vision*, pp. 1501–1510, 2017.

[5] Y. Lee and C. Kim, "Wave-u-mamba: An end-to-end framework for high-quality and efficient speech super resolution," *arXiv preprint arXiv:2409.09337*, 2024.

[6] A. Gu and T. Dao, "Mamba: Linear-time sequence modeling with selective state spaces," *arXiv preprint arXiv:2312.00752*, 2023.

[7] C. Donahue, J. McAuley, and M. Puckette, "Adversarial audio synthesis," *arXiv preprint arXiv:1802.04208*, 2018.

[8] R. Sammut Bonnici, C. Saitis, and M. Benning, "Timbre transfer with variational auto encoding and cycle-consistent adversarial networks," *arXiv e-prints*, pp. arXiv–2109, 2021.

[9] Z.-S. Liu, V. Kalogeiton, and M.-P. Cani, "Multiple style transfer via variational autoencoder," in *2021 IEEE International Conference on Image Processing (ICIP)*, pp. 2413–2417, IEEE, 2021.

[10] Z. Kong, W. Ping, J. Huang, K. Zhao, and B. Catanzaro, "Diffwave: A versatile diffusion model for audio synthesis," *arXiv preprint arXiv:2009.09761*, 2020.

[11] K. Goel, A. Gu, C. Donahue, and C. Ré, "It's raw! audio generation with state-space models," in *International Conference on Machine Learning*, pp. 7616–7633, PMLR, 2022.

[12] M. H. Erol, A. Senocak, J. Feng, and J. S. Chung, "Audio mamba: Bidirectional state space model for audio representation learning," *arXiv preprint arXiv:2406.03344*, 2024.

[13] C. K. On, P. M. Pandiyan, S. Yaacob, and A. Saudi, "Mel-frequency cepstral coefficient analysis in speech recognition," in *2006 International Conference on Computing & Informatics*, pp. 1–5, IEEE, 2006.

[14] C. Kim and R. M. Stern, "Power-normalized cepstral coefficients (pncc) for robust speech recognition," *IEEE/ACM Transactions on audio, speech, and language processing*, vol. 24, no. 7, pp. 1315–1329, 2016.

[15] Y. Luo and N. Mesgarani, "Tasnet: time-domain audio separation network for real-time, single-channel speech separation," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 696–700, IEEE, 2018.

[16] D. Stoller, S. Ewert, and S. Dixon, "Wave-u-net: A multi-scale neural network for end-to-end audio source separation," *arXiv preprint arXiv:1806.03185*, 2018.

[17] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," *Advances in neural information processing systems*, vol. 33, pp. 12449–12460, 2020.

[18] H. Akbari, L. Yuan, R. Qian, W.-H. Chuang, S.-F. Chang, Y. Cui, and B. Gong, "Vatt: Transformers for multimodal self-supervised learning from raw video, audio and text," *Advances in Neural Information Processing Systems*, vol. 34, pp. 24206–24221, 2021.

[19] A. Van Den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, K. Kavukcuoglu, *et al.*, "Wavenet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, vol. 12, 2016.

[20] V. Kuleshov, S. Z. Enam, and S. Ermon, "Audio super resolution using neural networks," *arXiv preprint arXiv:1708.00853*, 2017.

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[22] T. Xiao, M. Singh, E. Mintun, T. Darrell, P. Dollár, and R. Girshick, "Early convolutions help transformers see better," *Advances in neural information processing systems*, vol. 34, pp. 30392–30400, 2021.

[23] A. Odena, V. Dumoulin, and C. Olah, "Deconvolution and checkerboard artifacts," *Distill*, vol. 1, no. 10, p. e3, 2016.

[24] T. Salimans and D. P. Kingma, "Weight normalization: A simple reparameterization to accelerate training of deep neural networks," *Advances in neural information processing systems*, vol. 29, 2016.

[25] J. Kong, J. Kim, and J. Bae, "Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis," 2020.

[26] K. Kumar, R. Kumar, T. de Boissiere, L. Gestin, W. Z. Teoh, J. Sotelo, A. de Brebisson, Y. Bengio, and A. Courville, "Melgan: Generative adversarial networks for conditional waveform synthesis," 2019.

[27] R. Yamamoto, E. Song, and J. Kim, "Parallel wavegan: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram," 2020.

[28] J. Engel, C. Resnick, A. Roberts, S. Dieleman, M. Norouzi, D. Eck, and K. Simonyan, "Neural audio synthesis of musical notes with wavenet autoencoders," in *International Conference on Machine Learning*, pp. 1068–1077, PMLR, 2017.