A Mathematical Background for Dual-Path Diffusion

A.1 Forward Diffusion Process

In dual-path diffusion (DPD), we consider a Gaussian diffusion process [16] that continuously diffuses our generation target $\mathbf{z}$ into increasingly noisy versions of $\mathbf{z}$, denoted as $\mathbf{z}_t$ with $t \in [0, 1]$ running from $t = 0$ (least noisy) to $t = 1$ (most noisy). This forward diffusion process is formally defined as

$$q(\mathbf{z}_t|\mathbf{z}) = \mathcal{N}(\mathbf{z}_t; \alpha_t \mathbf{z}, \sigma_t^2 \mathbf{I}), \tag{14}$$

where two strictly positive scalar-valued, continuously differentiable functions α_t, σ_t define the noise schedule [1] of this forward diffusion process. Building upon the nice properties of Gaussian distributions, we can express $q(\mathbf{z}_t|\mathbf{z}_s)$, for any $0 \leq s < t \leq 1$, as another Gaussian distribution:

$$q(\mathbf{z}_t|\mathbf{z}_s) = \mathcal{N}\left(\mathbf{z}_t; \frac{\alpha_t}{\alpha_s}\mathbf{z}_s, \left(\sigma_t^2 - \frac{\alpha_t}{\alpha_s}\sigma_s^2\right)\mathbf{I}\right). \tag{15}$$

Regarding the choice of noise scheduling functions, we consider the typical setting used in [1, 15]: $\alpha_t = \sqrt{1 - \sigma_t^2}$, which gives rise to a *variance-preserving* diffusion process [16]. Specifically, we employ the trigonometric functions in [48], defined as follows:

$$\alpha_t := \cos(\pi t/2) \quad \sigma_t := \sin(\pi t/2) \quad \forall t \in [0, 1] \tag{16}$$

$$\Leftrightarrow \ \alpha_\delta := \cos(\delta) \qquad \sigma_\delta := \sin(\delta) \qquad \forall \delta \in [0, \pi/2]. \tag{17}$$

With this re-parameterization, the diffusion process can now be defined in terms of angle $\delta \in [0, \pi/2]$:

$$\mathbf{z}_\delta = \cos(\delta)\mathbf{z} + \sin(\delta)\boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{18}$$

where $\mathbf{z}_\delta$ gets noisier as δ increases from 0 to $\pi/2$.

A.2 Prediction of Diffusion Velocity

The diffusion velocity of $\mathbf{z}_\delta$ at δ [48] is defined as:

$$\mathbf{v}_\delta := \frac{d\mathbf{z}_\delta}{d\delta} = \frac{d\cos(\delta)}{d\delta}\mathbf{z} + \frac{d\sin(\delta)}{d\delta}\boldsymbol{\epsilon} = \cos(\delta)\boldsymbol{\epsilon} - \sin(\delta)\mathbf{z}. \tag{19}$$

Based on $\mathbf{v}_\delta$, we can compute $\mathbf{z}$ and $\boldsymbol{\epsilon}$ from a noisy latent $\mathbf{z}_\delta$:

$$\mathbf{z} = \cos(\delta)\mathbf{z}_\delta - \sin(\delta)\mathbf{v}_\delta = \alpha_\delta \mathbf{z}_\delta - \sigma_\delta \mathbf{v}_\delta; \tag{20}$$

$$\boldsymbol{\epsilon} = \sin(\delta)\mathbf{z}_\delta + \cos(\delta)\mathbf{v}_\delta = \sigma_\delta \mathbf{z}_\delta + \alpha_\delta \mathbf{v}_\delta, \tag{21}$$

which suggests $\mathbf{v}_\delta$ a feasible target for network prediction $\hat{\mathbf{v}}_\theta(\mathbf{z}_\delta; \mathbf{c})$ given a collection of conditions $\mathbf{c}$, as an alternative to the $\mathbf{z}$ prediction ($\hat{\mathbf{z}}_\theta(\mathbf{z}_\delta; \mathbf{c})$), e.g., in [16], and the $\boldsymbol{\epsilon}$ prediction ($\hat{\boldsymbol{\epsilon}}_\theta(\mathbf{z}_\delta; \mathbf{c})$), e.g., in [1, 50, 74]. As reported by Salimans and Ho [48] and Schneider et al. [23], training the neural network θ with a mean squared error (MSE) loss as in the pioneering work [1] remains effective:

$$\mathcal{L} := \mathbb{E}_{\mathbf{z}\sim p_{\text{data}}(\mathbf{z}), \boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\mathbf{I}), \delta\sim\text{Uniform}[0,1]}\left[\|\cos(\delta)\boldsymbol{\epsilon} - \sin(\delta)\mathbf{z} - \hat{\mathbf{v}}_\theta(\cos(\delta)\mathbf{z} + \sin(\delta)\boldsymbol{\epsilon}; \mathbf{c})\|_2^2\right], \tag{22}$$

which forms the basis of DPD's training loss, i.e., the simplest case of considering only a single chunk per input ($M = 1$) in Eq. (7). We can easily extend this to a multi-chunk version by sampling M different angles $\delta_1, \ldots, \delta_M \sim$ Uniform$[0, 1]$, where the m-th sampled angle is applied to the corresponding chunk of the latent, i.e., $\mathbf{z}[(m-1)L/M : mL/M]$.

A.3 Generative Diffusion Process

Generation is done by inverting the forward process from a noise vector randomly drawn from $\mathcal{N}(\mathbf{0}, \mathbf{I})$. One efficient way to accomplish this is to take advantage of DDIM [26], which enables running backward from angle δ to angle $\delta - \omega$, for any step size $0 < \omega < \delta$:

$$p_\theta(\mathbf{z}_{\delta-\omega}|\mathbf{z}_\delta) := q\left(\mathbf{z}_{\delta-\omega}\middle|\mathbf{z} = \frac{\mathbf{z}_\delta - \sigma_\delta\hat{\boldsymbol{\epsilon}}_\theta(\mathbf{z}_\delta; \mathbf{c})}{\alpha_\delta}\right) = \alpha_{\delta-\omega}\left(\frac{\mathbf{z}_\delta - \sigma_\delta\hat{\boldsymbol{\epsilon}}_\theta(\mathbf{z}_\delta; \mathbf{c})}{\alpha_\delta}\right) + \sigma_{\delta-\omega}\boldsymbol{\epsilon}, \tag{23}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Song et al. [26] considered $\boldsymbol{\epsilon} \equiv \hat{\boldsymbol{\epsilon}}_\theta(\mathbf{z}_\delta; \mathbf{c})$, leading to a deterministic update rule:

$$\mathbf{z}_{\delta-\omega} = \frac{\alpha_{\delta-\omega}}{\alpha_\delta}\mathbf{z}_\delta + \left(\sigma_{\delta-\omega} - \frac{\alpha_{\delta-\omega}\sigma_\delta}{\alpha_\delta}\right)\hat{\boldsymbol{\epsilon}}_\theta(\mathbf{z}_\delta; \mathbf{c}). \tag{24}$$

Building upon the diffusion velocity, Salimans and Ho [48] re-parameterized DDIM as

$$\begin{aligned} p_\theta(\mathbf{z}_{\delta-\omega}|\mathbf{z}_\delta) :=& q\left(\mathbf{z}_{\delta-\omega}\,|\mathbf{z} = \alpha_\delta\mathbf{z}_\delta - \sigma_\delta\hat{\mathbf{v}}_\theta(\mathbf{z}_\delta; \mathbf{c})\right) && (25)\\ =& \alpha_{\delta-\omega}\left(\alpha_\delta\mathbf{z}_\delta - \sigma_\delta\hat{\mathbf{v}}_\theta(\mathbf{z}_\delta; \mathbf{c})\right) + \sigma_{\delta-\omega}\boldsymbol{\epsilon}, && (26)\end{aligned}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Here, we can similarly consider a parameterized noise vector $\boldsymbol{\epsilon} \equiv \sigma_\delta\mathbf{z}_\delta + \alpha_\delta\hat{\mathbf{v}}_\theta(\mathbf{z}_\delta; \mathbf{c})$ based on Eq. (21), yielding a simplified deterministic update rule:

$$\begin{aligned} \mathbf{z}_{\delta-\omega} =& \alpha_{\delta-\omega}\left(\alpha_\delta\mathbf{z}_\delta - \sigma_\delta\hat{\mathbf{v}}_\theta(\mathbf{z}_\delta; \mathbf{c})\right) + \sigma_{\delta-\omega}\left(\sigma_\delta\mathbf{z}_\delta + \alpha_\delta\hat{\mathbf{v}}_\theta(\mathbf{z}_\delta; \mathbf{c})\right) && (27)\\ =& \left(\alpha_{\delta-\omega}\alpha_\delta - \sigma_{\delta-\omega}\sigma_\delta\right)\mathbf{z}_\delta + \left(\sigma_{\delta-\omega}\alpha_\delta - \alpha_{\delta-\omega}\sigma_\delta\right)\hat{\mathbf{v}}_\theta(\mathbf{z}_\delta; \mathbf{c}) && (28)\\ =& \cos(\omega)\mathbf{z}_\delta - \sin(\omega)\hat{\mathbf{v}}_\theta(\mathbf{z}_\delta; \mathbf{c}) && (29)\end{aligned}$$

where the last equation is obtained by applying the trigonometric identities:

$$\begin{aligned} \alpha_{\delta-\omega}\alpha_\delta - \sigma_{\delta-\omega}\sigma_\delta &= \cos(\delta-\omega)\cos(\delta) - \sin(\delta-\omega)\sin(\delta) = \cos(\omega); && (30)\\ \sigma_{\delta-\omega}\alpha_\delta - \alpha_{\delta-\omega}\sigma_\delta &= \sin(\delta-\omega)\cos(\delta) - \cos(\delta-\omega)\sin(\delta) = \sin(\omega). && (31)\end{aligned}$$

Building upon this angular update rule and having specified the angle step sizes $\omega_1, \ldots, \omega_T$ with $\sum_{t=1}^{T}\omega_t = \pi/2$, we can generate samples from $\mathbf{z}_{\pi/2} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ after T steps of sampling:

$$\mathbf{z}_{\delta_t-\omega_t} = \cos(\omega_t)\mathbf{z}_{\delta_t} - \sin(\omega_t)\hat{\mathbf{v}}_\theta(\mathbf{z}_{\delta_t}; \mathbf{c}), \quad \delta_t = \begin{cases} \frac{\pi}{2} - \sum_{i=t+1}^{T}\omega_i, & 1 \le t < T; \\ \frac{\pi}{2}, & t = T, \end{cases} \tag{32}$$

running from $t = T$ to $t = 1$.

B Training and Implementation Details

B.1 Audio VAE-GAN

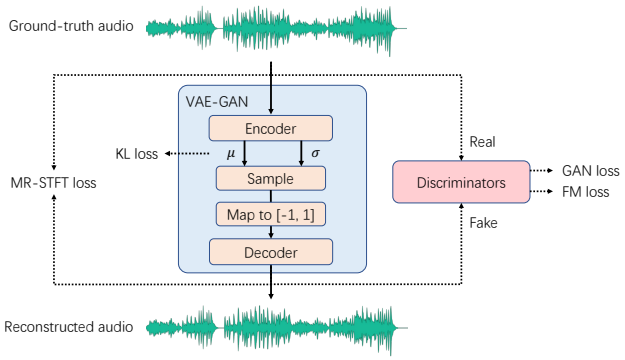


Figure 4: The audio VAE-GAN trained for dual-path diffusion models

As shown in Figure 4, we train a VAE-GAN to extract 250Hz 16-dimensional latent $\mathbf{z} \in \mathbb{R}^{L\times 16}$ from a 24kHz audio $\mathbf{x} \in \mathbb{R}^{T_{\text{wav}}}$ with $L = \lceil T_{\text{wav}}/96 \rceil$. The audio VAE-GAN mainly comprises three trainable modules: (i) a variational Gaussian encoder $\mathcal{E}_\phi(\mathbf{x}) \equiv \mathcal{N}(\mu_\phi(\mathbf{x}), \sigma_\phi(\mathbf{x})\mathbf{I})$, (ii) a decoder $\mathcal{D}_\phi(\mathbf{z})$, and (iii) a set of n discriminators $\{D^{(i)}\}_{i=1}^{n}$.

Regarding network architecture, we use the ResNet-style convolutional neural networks (CNNs) in HiFi-GAN [66] as the backbone.[6] For the encoder, we replace the up-sampling blocks in HiFi-GAN with convolution-based down-sampling blocks, with down-sampling rates of $[2, 3, 4, 4]$, output dimensions of $[32, 64, 128, 256]$ and kernel sizes of $[5, 7, 9, 9]$ in four down-sampling blocks, giving 40M

[6] Our implementation is similar to that in https://github.com/jik876/hifi-gan.

parameters. The final layer of the encoder maps the 256-dimensional output to two 16-dimensional latent sequences, respectively for the mean and variance of diagonal Gaussian sampling.[7] As shown in Figure 4, to match the normal range of targets for diffusion models [1, 2], we map the sampled outputs to $[-1, 1]$ by $\mathbf{z}_{(i,j)} := \min\left\{\max\left\{\bar{\mathbf{z}}_{(i,j)}/3, -1\right\}, 1\right\} \forall i, j$, where the subscript (i, j) denotes the value on the i-th row and j-th column, and the choice of 3 in practice would sieve extreme values occupying $< 0.1\%$. For the architecture setting of the decoder, it inherits the same architecture of HiFi-GAN, and uses up-sampling rates of $[4, 4, 3, 2]$, kernel sizes of $[9, 9, 5, 7]$ and larger number of output channels ($[768, 384, 192, 96]$) for four up-sampling blocks, taking 60.1M parameters.

For adversarial training, we use the multi-period discriminators in [66] and the multi-resolution spectrogram discriminators in [67]. The training scheme is similar to that in [66]. The training loss for the encoder and the decoder comprises four components:

$$\mathcal{L}_{\text{vae-gan}}(\phi) := \mathbb{E}_{\mathbf{x}\sim p_{\text{data}}(\mathbf{x})}\left[\mathbb{E}_{\mathbf{z}\sim\mathcal{E}_\phi(\mathbf{x})}\left[\lambda_{\text{mr-stft}}\mathcal{L}_{\text{mr-stft}} + \lambda_{\text{fm}}\mathcal{L}_{\text{fm}} + \lambda_{\text{gan}}\mathcal{L}_{\text{gan}} + \lambda_{\text{kl}}\mathcal{L}_{\text{kl}}\right]\right] \tag{33}$$

$$\mathcal{L}_{\text{mr-stft}} := \sum_{r=1}^{R} \left\|\text{STFT}_r(\mathbf{x}) - \text{STFT}_r\left(\mathcal{D}_\phi\left(\mathbf{z}\right)\right)\right\|_1 \tag{34}$$

$$\mathcal{L}_{\text{fm}} := \sum_{i=1}^{n} \frac{1}{|D^{(i)}|} \sum_{l=1}^{|D^{(i)}|} \left\|D_l^{(i)}(\mathbf{x}) - D_l^{(i)}(\mathcal{D}_\phi\left(\mathbf{z}\right))\right\|_1 \tag{35}$$

$$\mathcal{L}_{\text{gan}} := \sum_{i=1}^{n} \left(D^{(i)}\left(\mathcal{D}_\phi\left(\mathbf{z}\right)\right) - 1\right)^2 \tag{36}$$

$$\mathcal{L}_{\text{kl}} := \text{KL}\left(\mathcal{E}_\phi(\mathbf{x}) || \mathcal{N}(\mathbf{0}, \mathbf{I})\right), \tag{37}$$

where STFT_r computes the magnitudes after the r-th short-time Fourier transform (STFT) out of $R = 7$ STFTs (the number of FFTs $= [8192, 4096, 2048, 512, 128, 64, 32]$; the window sizes $= [4096, 2048, 1024, 256, 64, 32, 16]$; the hop sizes $= [2048, 1024, 512, 128, 32, 16, 8]$), $|D^{(i)}|$ denotes the number of hidden layers used for feature matching in discriminator $D^{(i)}$, $D_l^{(i)}$ denotes the outputs of the l-th hidden layers in discriminator $D^{(i)}$, and $\lambda_{\text{mr-stft}}$, λ_{fm}, λ_{gan}, λ_{kl} are the weights, respectively, for the multi-resolution STFT loss $\mathcal{L}_{\text{mr-stft}}$, the feature matching loss $\mathcal{L}_{\text{fm}}$, the GAN's generator loss $\mathcal{L}_{\text{gan}}$, and the Kullback–Leibler divergence based regularization loss $\mathcal{L}_{\text{kl}}$. To balance the scale of different losses, we set $\lambda_{\text{mr-stft}} = 50$, $\lambda_{\text{fm}} = 20$, $\mathcal{L}_{\text{gan}} = 1$, and $\lambda_{\text{kl}} = 5 \times 10^{-3}$ in our training. In practice, we find it critical to lower the scale of the KL loss for a better reconstruction, though the distribution of the latents can still be close to zero mean and unit variance.

B.2 Wav2Vec2-Conformer

Our implementation of Wav2Vec2-Conformer was based on an open-source library.[8] In particular, Wav2Vec2-Conformer follows the same architecture as Wav2Vec2 [44], but replaces the Transformer structure with the Conformer [45]. This model with 199.5M parameters was trained in self-supervised learning (SSL) manner similar to [44] using our prepared 257k hours of music data.

B.3 MuLan

Our reproduced MuLan [43] is composed of a music encoder and a text encoder. For music encoding, we rely on a publicly accessible Audio Spectrogram Transformer (AST) model pre-trained on AudioSet,[9] which gives promising results on various audio classification benchmarks. For text encoding, we employ the BERT [8] base model pre-trained on a large corpus of English data using a masked language modeling (MLM) objective.[10] These two pre-trained encoders, together having 195.3M parameters, were subsequently fine-tuned on the 257k hours of music data with a text augmentation technique similar to [6]. In particular, we enriched the tag-based texts to generate music captions by asking ChatGPT [68]. At training time, we randomly paired each audio with either

[7] The Gaussian sampling is referred to LDMs' implementation at https://github.com/CompVis/latent-diffusion/blob/main/ldm/modules/distributions/distributions.py

[8] https://huggingface.co/docs/transformers/model_doc/wav2vec2-conformer

[9] https://huggingface.co/MIT/ast-finetuned-audioset-10-10-0.4593

[10] https://huggingface.co/bert-base-uncased

Algorithm 1 Music Generation

given $\mathcal{D}_\phi$, $\hat{\mathbf{v}}_\theta$, T, $\omega_1, \ldots, \omega_T$
input Music/text prompt $\mathcal{P}$

Initialize $\delta_T = \pi/2$
Compute the MuLan tokens for $\mathcal{P}$: $\mathbf{c}_{1:T_{\text{cnd}}}$
Generate $\mathbf{u}_{1:T_{\text{ST}}}$ from $\mathbf{c}_{1:T_{\text{cnd}}}$ with LM ▷ (1)
Sample $\mathbf{z}_{\delta_T} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for $t = T$ to 1 **do**
  Prepare condition: $\mathbf{c} = \{\mathbf{u}_{1:T_{\text{ST}}}, [\delta_t]_{r=1}^{L}\}$ ▷ (8)
  Update angle: $\delta_{t-1} = \delta_t - \omega_t$ ▷ (3)
  $\mathbf{z}_{\delta_{t-1}} = \cos(\omega_t)\mathbf{z}_{\delta_t} - \sin(\omega_t)\hat{\mathbf{v}}_\theta(\mathbf{z}_{\delta_t}; \mathbf{c})$ ▷ (9)
end for
repeat
  pass $\mathbf{c}_{1:T_{\text{cnd}}}$, $\mathbf{u}_{1:T_{\text{ST}}}$ and $\mathbf{z}_0$ to **Algorithm 2**
until $\mathbf{z}_0$ reaches the desired length
return $\mathcal{D}_\phi(\mathbf{z}_0)$

Algorithm 2 Music Continuation

given $\mathcal{D}_\phi$, $\hat{\mathbf{v}}_\theta$, T, M, $\omega_1, \ldots, \omega_T$
input Music $\mathbf{z}_0$ and $\mathbf{c}_{1:T_{\text{cnd}}}$, $\mathbf{u}_{1:T_{\text{ST}}}$ (if provided)

Denote $M_{\text{ST}} = \lceil T_{\text{ST}}/M \rceil$, $L_M = \lceil L/M \rceil$
Initialize $\delta_T = \pi/2$
Generate $\mathbf{u}_{T_{\text{ST}}:T_{\text{ST}}+M_{\text{ST}}}$ from $\mathbf{c}_{1:T_{\text{cnd}}} \oplus \mathbf{u}_{M_{\text{ST}}:T_{\text{ST}}}$
Sample $\mathbf{z}_{\text{new}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \in \mathbb{R}^{L_M}$
Save first chunk: $\mathbf{z}_{\text{save}} = \mathbf{z}_0[:L_M]$
$\mathbf{z}_{\delta_T} = \mathbf{z}_0[L_M:] \oplus \mathbf{z}_{\text{new}}$
for $t = T$ to 1 **do**
  Update $\boldsymbol{\delta}_{\text{new}} = [0]_{r=1}^{L-L_M} \oplus [\delta_t]_{r=1}^{L_M}$
  Prepare condition: $\mathbf{c} = \{\mathbf{u}_{M_{\text{ST}}:T_{\text{ST}}+M_{\text{ST}}}, \boldsymbol{\delta}_{\text{new}}\}$
  Update angle: $\delta_{t-1} = \delta_t - \omega_t$
  $\mathbf{z}_{\delta_{t-1}} = \cos(\omega_t)\mathbf{z}_{\delta_t} - \sin(\omega_t)\hat{\mathbf{v}}_\theta(\mathbf{z}_{\delta_t}; \mathbf{c})$
end for
return $\mathbf{z}_{\text{save}} \oplus \mathbf{z}_0$

the generated caption or its respective tags. In practice, this could robustly improve the model's capability of handling free-form text.

C Algorithms for MeLoDy

MeLoDy supports music or text prompting for music generation, as illustrated in Figure 1. We concretely detail the sampling procedures in Algorithm 1, where the algorithm starts by generating the latent sequence of length L and then recursively prolongs the latent sequence using Algorithm 2 until it reaches the desired length.

We further explain how music continuation can be effectively done in DPD. Recall that the inputs for training DPD are the concatenated chunks of noisy latents in different noise scales. To continue a given music audio, we can add a new chunk composed of random noises and drop the first chunk. This is feasible since the conditions (i.e., the semantic tokens and the angles) defined for DPD have an autoregressive nature. Based on the semantic LM, we can continue the generation of $\lceil T_{\text{ST}}/M \rceil$ semantic tokens for the new chunk. Besides, it is sensible to keep the chunks other than the new chunk to have zero angles: $\boldsymbol{\delta}_{\text{new}} := [0]_{r=1}^{L-\lceil L/M \rceil} \oplus [\delta_t]_{r=1}^{\lceil L/M \rceil}$, as shown in Algorithm 2.

In addition, music inpainting can be done in a similar way. We replace the inpainting partition of the input audio with random noise and partially set the angle vector to zeros to mark the positions where the denoising operations are not needed. Yet, in this case, the semantic tokens can only be roughly estimated using the remaining part of the music audio.

Table 4: The objective measures for the ablation study on angle schedules.

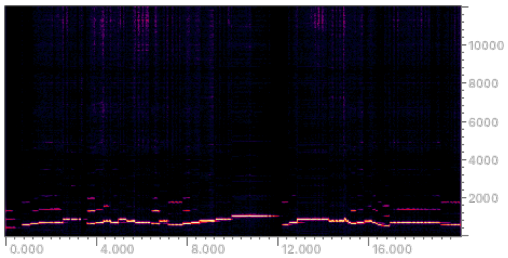
Angle schedule (ω_t)	**Steps** (T)	**FAD** (↓)	**MCC** (↑)
Uniform [23]: $\omega_t = \frac{\pi}{2T}$	10	8.52	0.45
	20	6.31	0.49
Ours proposed in Eq. (4): $\omega_t = \frac{\pi}{6T} + \frac{2\pi t}{3T(T+1)}$	10	**5.93**	**0.52**
	20	**5.41**	**0.53**

D Ablation Study on Angle Schedules

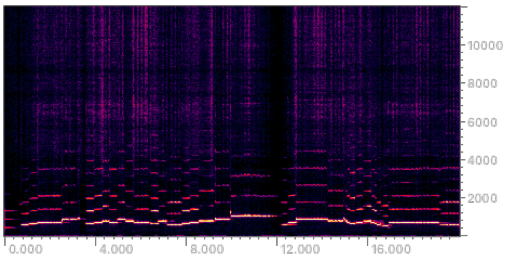
We conduct an ablation study on angle schedules to validate the effectiveness of our proposed angle schedule $\omega_1, \ldots, \omega_T$ in Eq. (4) in comparison to the previous uniform angle schedule [23] also used for angle-parameterized continuous-time diffusion models. In particular, the same pre-trained DPD model $\hat{\mathbf{v}}_\theta$ and was used to sample with two different angle schedules with 10 steps and 20 steps,

respectively, conditional on the same semantic tokens generated for the text prompts in MusicCaps. Table 4 shows their corresponding objective measures in terms of FAD and MCC. We observe a significant improvement, especially when taking a small number of sampling steps, by using the proposed sampling method. This is aligned with our expectations that taking larger steps at the beginning of the sampling followed by smaller steps could improve the quality of samples, similar to the findings in previous diffusion scheduling methods [50, 51].

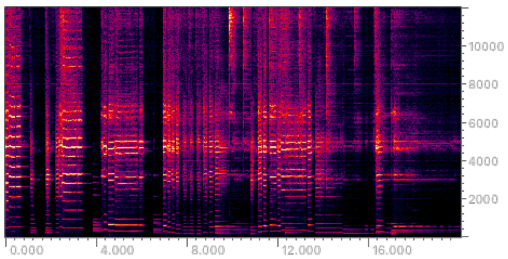
We further qualitatively analyze the quality of the generated samples using some simple text prompts of instruments, i.e., flute, saxophone, and acoustic guitar, by pair-wise comparing their spectrograms as illustrated in Figure 5. In the case of "flute", sampling with the proposed angle schedule results in a piece of naturally sound music, being more saturated in high-frequency bands and even remedying the breathiness of flute playing, as shown in Figure 5b. On the contrary, we can observe from the spectrogram in Figure 5a that the sample generated with a uniform angle schedule is comparatively monotonous. In the case of "saxophone", the uniform angle schedule leads to metallic sounds that are dissonant, as revealed by the higher energy in 3kHz to 6kHz frequency bands shown in Figure 5c. In comparison, the frequency bands are more consistent in Figure 5d, when the proposed schedule is used. While the comparatively poorer sample quality using the uniform schedule could be caused by the limited number of sampling steps, we also show the spectrograms after increasing the sampling steps from 10 to 20. In the case of "acoustic guitar", when taking 20 sampling steps, the samples generated with both angle schedules sound more natural. However, in Figure 5e, we witness a horizontal line around the 4.4kHz frequency band, which is unpleasant to hear. Whereas, the sample generated by our proposed schedule escaped such an acoustic issue, as presented in Figure 5f.



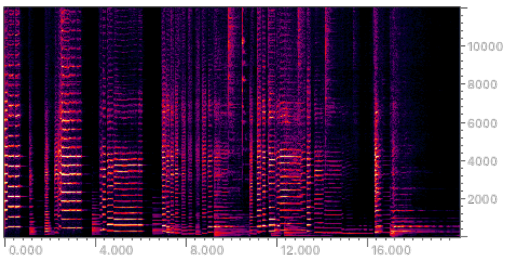
(a) 10-step sampling with uniform angle schedule for text prompt: "flute"



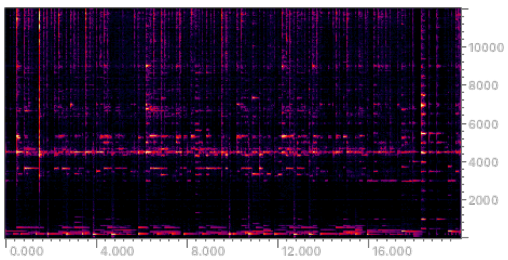
(b) 10-step sampling with our proposed angle schedule for text prompt: "flute"



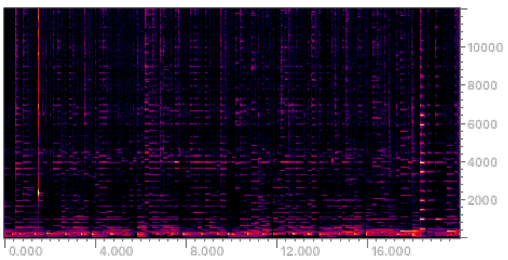
(c) 10-step sampling with uniform angle schedule for text prompt: "saxophone"



(d) 10-step sampling with our proposed angle schedule for text prompt: "saxophone"



(e) 10-step sampling with uniform angle schedule for text prompt: "acoustic guitar"



(f) 10-step sampling with our proposed angle schedule for text prompt: "acoustic guitar"

Figure 5: Spectrograms of generated samples with uniform (left) and our proposed (right) angle schedules

Table 5: The objective measures for the ablation study on architectures.

Architecture	**Velocity MSE** (↓)	**SI-SNRi** (↑)
UNet-1D [23]	0.13	5.33
UNet-2D [31]	0.15	4.96
DPD (Ours)	**0.12**	**6.15**

E Ablation Study on Architectures

To examine the superiority of our proposed dual-path model in Figure 2, we also study the ablation of network architectures. In particular, to focus on the denoising capability of different architectures, we only take a subset of the training data (approximately 5k hours of music data) to train different networks with the same optimization configurations – 100k training steps using AdamW optimizer with a learning rate of 5×10^{-4} and a batch size of 96 on 8 NVIDIA V100 GPUs. For a fair comparison, we train the UNet-1D[11] and the UNet-2D[12] with comparable numbers of parameters (approximately 300M). Note that the FAD and MCC measures are not suitable for evaluating the performance of each forward pass of the trained network for denoising. In addition to the training objective, i.e., the velocity MSE, we use the scale-invariant signal-to-noise ratio (SNR) improvements (SI-SNRi) [35, 37] between the true latent $\mathbf{z}$ and the predicted latent $\hat{\mathbf{z}} = \alpha_\delta \mathbf{z}_\delta - \sigma_\delta \hat{\mathbf{v}}_\theta(\mathbf{z}_\delta; \mathbf{c})$. The results are shown in Table 5, where our proposed dual-path architecture outperforms the other two widely used UNet-style architectures in terms of both the velocity MSE and SI-SNRi.

F Qualitative Evaluation

To conduct a pair-wise comparison, each music producer is asked to fill in the form composed of three questions. Specifically, we present the user interface for each pair-wise comparison in Figure 6.

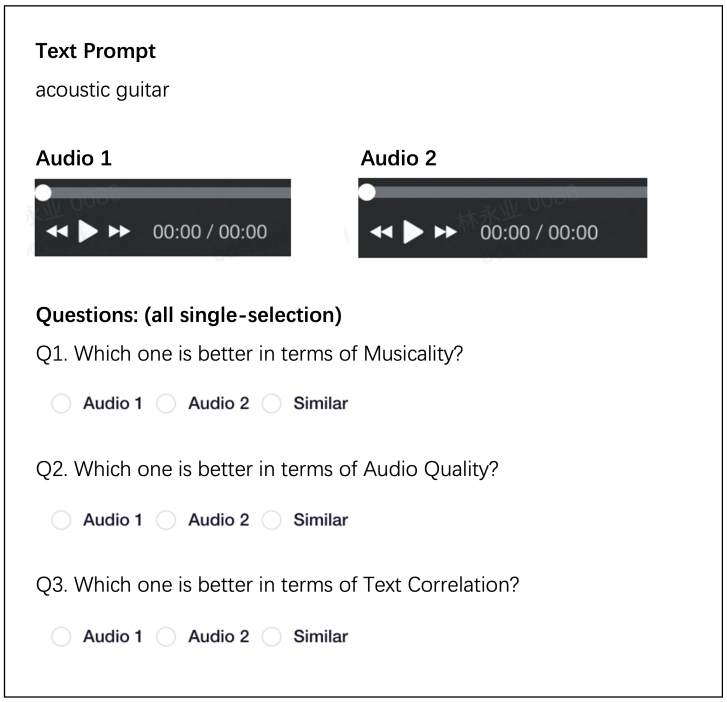


Figure 6: The user interface for music producers in each pair-wise comparison

[11] Our implementation of UNet-1D relied on https://github.com/archinetai/a-unet.
[12] Our implementation of UNet-2D relied on https://huggingface.co/riffusion/riffusion-model-v1.

References

[1] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[2] 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.

[3] Zalán Borsos, Raphaël Marinier, Damien Vincent, Eugene Kharitonov, Olivier Pietquin, Matt Sharifi, Olivier Teboul, David Grangier, Marco Tagliasacchi, and Neil Zeghidour. Audiolm: a language modeling approach to audio generation. *arXiv preprint arXiv:2209.03143*, 2022.

[4] Prafulla Dhariwal, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford, and Ilya Sutskever. Jukebox: A generative model for music. *arXiv preprint arXiv:2005.00341*, 2020.

[5] Andrea Agostinelli, Timo I Denk, Zalán Borsos, Jesse Engel, Mauro Verzetti, Antoine Caillon, Qingqing Huang, Aren Jansen, Adam Roberts, Marco Tagliasacchi, et al. Musiclm: Generating music from text. *arXiv preprint arXiv:2301.11325*, 2023.

[6] Qingqing Huang, Daniel S Park, Tao Wang, Timo I Denk, Andy Ly, Nanxin Chen, Zhengdong Zhang, Zhishuai Zhang, Jiahui Yu, Christian Frank, et al. Noise2music: Text-conditioned music generation with diffusion models. *arXiv preprint arXiv:2302.03917*, 2023.

[7] 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.

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[9] 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.

[10] Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, et al. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*, 2022.

[11] 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.

[12] Felix Kreuk, Gabriel Synnaeve, Adam Polyak, Uriel Singer, Alexandre Défossez, Jade Copet, Devi Parikh, Yaniv Taigman, and Yossi Adi. Audiogen: Textually guided audio generation. *arXiv preprint arXiv:2209.15352*, 2022.

[13] Chengyi Wang, Sanyuan Chen, Yu Wu, Ziqiang Zhang, Long Zhou, Shujie Liu, Zhuo Chen, Yanqing Liu, Huaming Wang, Jinyu Li, et al. Neural codec language models are zero-shot text to speech synthesizers. *arXiv preprint arXiv:2301.02111*, 2023.

[14] Eugene Kharitonov, Damien Vincent, Zalán Borsos, Raphaël Marinier, Sertan Girgin, Olivier Pietquin, Matt Sharifi, Marco Tagliasacchi, and Neil Zeghidour. Speak, read and prompt: High-fidelity text-to-speech with minimal supervision. *arXiv preprint arXiv:2302.03540*, 2023.

[15] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. *In International Conference on Machine Learning*, pages 2256–2265, 2015.

[16] Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *Advances in neural information processing systems*, 34:21696–21707, 2021.

[17] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34, 2021.

[18] Rongjie Huang, Max WY Lam, Jun Wang, Dan Su, Dong Yu, Yi Ren, and Zhou Zhao. Fastdiff: A fast conditional diffusion model for high-quality speech synthesis. *arXiv preprint arXiv:2204.09934*, 2022.

[19] Sungwon Kim, Heeseung Kim, and Sungroh Yoon. Guided-tts 2: A diffusion model for high-quality adaptive text-to-speech with untranscribed data. *arXiv preprint arXiv:2205.15370*, 2022.

[20] Kai Shen, Zeqian Ju, Xu Tan, Yanqing Liu, Yichong Leng, Lei He, Tao Qin, Sheng Zhao, and Jiang Bian. Naturalspeech 2: Latent diffusion models are natural and zero-shot speech and singing synthesizers. *arXiv preprint arXiv:2304.09116*, 2023.

[21] 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. *arXiv preprint arXiv:2301.12503*, 2023.

[22] 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. *arXiv preprint arXiv:2301.12661*, 2023.

[23] Flavio Schneider, Zhijing Jin, and Bernhard Schölkopf. Moûsai: Text-to-music generation with long-context latent diffusion. *arXiv preprint arXiv:2301.11757*, 2023.

[24] David Holz et al. Midjourney. *Artificial Intelligence platform. Accessible at https://www.midjourney.com/Accessed November 1st*, 2023.

[25] Zhisheng Xiao, Karsten Kreis, and Arash Vahdat. Tackling the generative learning trilemma with denoising diffusion gans. *arXiv preprint arXiv:2112.07804*, 2021.

[26] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[27] Qiuqiang Kong, Yin Cao, Turab Iqbal, Yuxuan Wang, Wenwu Wang, and Mark D Plumbley. Panns: Large-scale pretrained audio neural networks for audio pattern recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:2880–2894, 2020.

[28] Antoine Caillon and Philippe Esling. Rave: A variational autoencoder for fast and high-quality neural audio synthesis. *arXiv preprint arXiv:2111.05011*, 2021.

[29] Marco Pasini and Jan Schlüter. Musika! fast infinite waveform music generation. *arXiv preprint arXiv:2208.08706*, 2022.

[30] Mubert Inc. Mubert. *URL https://mubert.com/*, 2023.

[31] S Forsgren and H Martiros. Riffusion - stable diffusion for real-time music generation. *URL https://riffusion.com/*, 2023.

[32] 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.

[33] Zalân Borsos, Matt Sharifi, Damien Vincent, Eugene Kharitonov, Neil Zeghidour, and Marco Tagliasacchi. Soundstorm: Efficient parallel audio generation. *arXiv preprint arXiv:2305.09636*, 2023.

[34] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. Maskgit: Masked generative image transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11315–11325, 2022.

[35] Yi Luo, Zhuo Chen, and Takuya Yoshioka. Dual-path rnn: efficient long sequence modeling for time-domain single-channel speech separation. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 46–50. IEEE, 2020.

[36] Jingjing Chen, Qirong Mao, and Dong Liu. Dual-path transformer network: Direct context-aware modeling for end-to-end monaural speech separation. *arXiv preprint arXiv:2007.13975*, 2020.

[37] Max WY Lam, Jun Wang, Dan Su, and Dong Yu. Effective low-cost time-domain audio separation using globally attentive locally recurrent networks. In *2021 IEEE Spoken Language Technology Workshop (SLT)*, pages 801–808. IEEE, 2021.

[38] Max WY Lam, Jun Wang, Dan Su, and Dong Yu. Sandglasset: A light multi-granularity self-attentive network for time-domain speech separation. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5759–5763. IEEE, 2021.

[39] Cem Subakan, Mirco Ravanelli, Samuele Cornell, Mirko Bronzi, and Jianyuan Zhong. Attention is all you need in speech separation. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 21–25. IEEE, 2021.

[40] Shengkui Zhao and Bin Ma. Mossformer: Pushing the performance limit of monaural speech separation using gated single-head transformer with convolution-augmented joint self-attentions. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.

[41] Yu-An Chung, Yu Zhang, Wei Han, Chung-Cheng Chiu, James Qin, Ruoming Pang, and Yonghui Wu. W2v-bert: Combining contrastive learning and masked language modeling for self-supervised speech pre-training. In *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 244–250. IEEE, 2021.

[42] Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi. Soundstream: An end-to-end neural audio codec. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:495–507, 2021.

[43] Qingqing Huang, Aren Jansen, Joonseok Lee, Ravi Ganti, Judith Yue Li, and Daniel PW Ellis. Mulan: A joint embedding of music audio and natural language. *arXiv preprint arXiv:2208.12415*, 2022.

[44] 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.

[45] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, et al. Conformer: Convolution-augmented transformer for speech recognition. *arXiv preprint arXiv:2005.08100*, 2020.

[46] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pages 8162–8171. PMLR, 2021.

[47] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[48] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022.

[49] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *arXiv preprint arXiv:2206.00364*, 2022.

[50] 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*, 2020.

[51] Max WY Lam, Jun Wang, Dan Su, and Dong Yu. Bddm: Bilateral denoising diffusion models for fast and high-quality speech synthesis. In *International Conference on Learning Representations*, 2022.

[52] Ken Shoemake. Animating rotation with quaternion curves. In *Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, pages 245–254, 1985.

[53] Biao Zhang and Rico Sennrich. Root mean square layer normalization. *Advances in Neural Information Processing Systems*, 32, 2019.

[54] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

[55] 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.

[56] 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.

[57] Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*, 2021.

[58] Tao Lei, Yu Zhang, Sida I Wang, Hui Dai, and Yoav Artzi. Simple recurrent units for highly parallelizable recurrence. *arXiv preprint arXiv:1709.02755*, 2017.

[59] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[60] Robin Rombach and Patrick Esser. Stable diffusion v2-1. *URL https://huggingface.co/stabilityai/stable-diffusion-2-1*, 2023.

[61] Lvmin Zhang and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. *arXiv preprint arXiv:2302.05543*, 2023.

[62] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[63] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning*, pages 1278–1286. PMLR, 2014.

[64] Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A Bharath. Generative adversarial networks: An overview. *IEEE signal processing magazine*, 35(1):53–65, 2018.

[65] 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.

[66] 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.

[67] Won Jang, Dan Lim, Jaesam Yoon, Bongwan Kim, and Juntae Kim. Univnet: A neural vocoder with multi-resolution spectrogram discriminators for high-fidelity waveform generation. *arXiv preprint arXiv:2106.07889*, 2021.

[68] OpenAI. Chatgpt. *URL https://chat.openai.com/*, 2023.

[69] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[70] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

[71] Shawn Hershey, Sourish Chaudhuri, Daniel PW Ellis, Jort F Gemmeke, Aren Jansen, R Channing Moore, Manoj Plakal, Devin Platt, Rif A Saurous, Bryan Seybold, et al. Cnn architectures for large-scale audio classification. In *2017 ieee international conference on acoustics, speech and signal processing (icassp)*, pages 131–135. IEEE, 2017.

[72] Kevin Kilgour, Mauricio Zuluaga, Dominik Roblek, and Matthew Sharifi. Fréchet audio distance: A reference-free metric for evaluating music enhancement algorithms. In *INTERSPEECH*, pages 2350–2354, 2019.

[73] 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. *arXiv preprint arXiv:2106.09685*, 2021.

[74] Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. Diffwave: A versatile diffusion model for audio synthesis. *arXiv preprint arXiv:2009.09761*, 2020.