Coarse-to-Fine Text-to-Music Latent Diffusion

Luca A. Lanzendörfer[*]
ETH Zurich
`lanzendoerfer@ethz.ch`

Tongyu Lu[*]
ETH Zurich
`tonglu@ethz.ch`

Nathanaël Perraudin
Swiss Data Science Center
`nathanael.perraudin@sdsc.ethz.ch`

Dorien Herremans
Singapore University of Technology and Design
`dorien_herremans@sutd.edu.sg`

Roger Wattenhofer
ETH Zurich
`wattenhofer@ethz.ch`

Abstract

We introduce `DiscoDiff`, a text-to-music generative model that utilizes two latent diffusion models to produce high-fidelity 44.1kHz music hierarchically. Our approach significantly enhances audio quality through a coarse-to-fine generation strategy, leveraging residual vector quantization from the Descript Audio Codec. We consolidate this coarse-to-fine design through an important observation that the audio latent representation can be splitted into primary and secondary part, controlling music contents and details accordingly. We validate the effectiveness of our approach and text-audio alignment through various objective metrics. Furthermore, we provide access to high-quality synthetic captions for MTG-Jamendo and FMA datasets, as well as open-sourcing `DiscoDiff`'s codebase and model checkpoints.

1 Introduction

Generating coherent long-form music at the waveform level is challenging due to its intricate, multi-scale structure, ranging from short rhythmic patterns to long melodies. Recent advancements address this by using neural audio codecs [1, 2, 3] to compress audio, significantly reducing the frame rate. These compressed representations allow generative models like large language models and diffusion models to efficiently synthesize high-quality audio. Our work is built on the Descript Audio Codec, which encodes audio into nine hierarchical tokens through Residual Vector Quantization (RVQ). Each token represents different scales of the audio, encoded in an 8-dimensional space, contributing additively to the final latent embedding.

In this work, we introduce `DiscoDiff`, a novel coarse-to-fine method for text-to-music generation on the continuous latent representation of the Descript Audio Codec [3]. `DiscoDiff` contains two 1D diffusion models capable of generating high-fidelity music at 44.1 kHz. Our method differs from previous latent diffusion approaches [4, 5], by directly leveraging the hierarchical RVQ patterns.

We exploit these properties by employing a coarse-to-fine generative approach, inspired by text-to-speech methods [6]. The first diffusion model generates the latent embedding in the coarsest level, while the second model, conditioned on the first, refines the remaining tokens. To train `DiscoDiff`, we generate high-quality captions for the MTG-Jamendo [7] and FMA [8] and filter sub-par FMA samples using likelihood estimation. Our contributions can be summarized as follows:

- We propose the application of a coarse-to-fine latent audio diffusion process, leading to enhanced sample quality in music generation.

[*]Equal contribution

Neural Information Processing Systems (NeurIPS) 2024 Workshop on AI-Driven Speech, Music, and Sound Generation.

- We curate and provide a quality ranking of the FMA dataset, filtered according to the likelihood ranking on a high-quality music distribution. Additionally, we open-source high-quality synthetic captions generated for MTG-Jamendo and FMA datasets.
- We release model checkpoints and accompanying code as open-source resources, supporting reproducibility and further advancements in the field of generative music.[1]

2 Related Work

Text-to-music models require two necessary parts: a generative model and a text-conditioning module. Auto-regressive models such as WaveNet [9] initiate music generation by synthesizing waveforms sample-by-sample. Recent advancements, including JukeBox [10], AudioLM [11], and MusicGen [12], use transformers to generate waveform latents, achieving high-quality samples. Denoising Diffusion Probabilistic Models (DDPM) [13] and Latent Diffusion Models (LDM) [14] offer faster sampling than auto-regressive models. DiffWave [15], Noise2Music [4], and Riffusion [16] demonstrate the effectiveness of diffusion models for waveform and spectrogram generation.

Recent latent diffusion models outperformed direct waveform or spectrogram generation [17, 18, 19, 20]. Stable Audio [5] achieved excellent results, thanks to the advances in compressed audio latent representation with audio encoder-decoders (codecs) such as SoundStream [1], EnCodec [2], and Descript Audio Codec (DAC) [3]. Residual vector quantization (RVQ) is normally used for further compression.

For text conditioning, T5 [21] is widely used in text-to-audio models like Make-an-Audio [22] and Noise2Music [4]. Mustango [23] employs FLAN-T5 [24], an instruction-tuned version of T5. Another approach, audio-text embedding, differs from pure-text embedding. The w2v-BERT model [25], used by AudioLM [11], produces joint audio-text embedding. MuLan [26] introduces unsupervised learning for MusicLM [27], while CLAP [28],used by Stable Audio, recently emerged as an open-source model for generating 128-dim joint embeddings for audio and text.

3 Method

Instead of generating raw waveforms, we model neural-codec-compressed continuous latent sequences. These latents offer a structured, efficient representation of audio, enabling coarse-to-fine generation. While using autoregressive models is intuitive for handling such sequences with token-by-token prediction, it is not the only option.

3.1 Neural Codec: DAC

For the audio representation, we employ the Descript Audio Codec (DAC) to compress the audio waveform. DAC initially converts waveform $\mathbf{w} \in \mathbb{R}^{512L}$, sampled at $44.1kHz$, into an audio embedding sequence $\mathbf{Z} \in \mathbb{R}^{D\times L}$ of frame rate $44.1kHz/512 \approx 86.1fps$. Subsequently, these embeddings are further compressed by Residual Vector Quantization (RVQ) with $K = 9$ codebooks.

A distinctive feature of DAC that significantly contributes to our method is the dimensionality reduction of the embedding from $D = 1024$ to $d = 8$ using learned linear projections. Specifically in latent query process as shown in Fig. 1, we have computation for $i = 0, \dots K - 1$:

$$\begin{aligned} \mathbf{X}_i = \arg\min_{\mathbf{X}\in\mathcal{X}_i} \|\mathbf{X} - \hat{\mathbf{X}}_i\|_2 \quad (\hat{\mathbf{X}}_i = \mathbf{P}_i\mathbf{Z}_i,\ \mathbf{P}_i \in \mathbb{R}^{d\times D},\ \mathbf{Z}_0 = \mathbf{Z}) \\ \mathbf{Z}_{i+1} = \mathbf{Z}_i - \mathbf{Q}_i\mathbf{X}_i, \quad (\mathbf{Q}_i \in \mathbb{R}^{D\times d}) \end{aligned} \tag{1}$$

where $\mathbf{P}_i, \mathbf{Q}_i$ is the learned in/out-projection weight in DAC, $\mathbf{Z}_i$ is the i^{th} residual and $\mathcal{X}_i$ is the i^{th} codebook. This mechanism was introduced to address the unbalanced codebook visitation problem inherent in the RVQ process. The smaller embeddings $\mathbf{X}_i$ are concatenated into the DAC latent

$$\mathbf{X} = \mathrm{concat}(\mathbf{X}_0, \dots, \mathbf{X}_{K-1}) \in \mathbb{R}^{Kd\times L} \tag{2}$$

which is our generation target. Importantly, this reduction allows us to generate 9 smaller latents of size 8 instead of a single embedding of size 1024, simplifying the generation task. We call this procedure "DAC latent query process".

[1]`https://github.com/ETH-DISCO/discodiff`

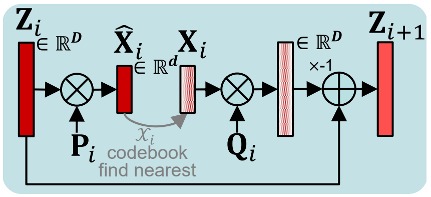

Figure 1: The i^{th} latent query step, as given in Eq. 1. Iterating this process starting from audio embedding $\mathbf{Z}_0$, we obtain residuals $\mathbf{Z}_i$ and dim-reduced latents $\mathbf{X}_i, i = 0, \ldots, K-1$. In-projection $\mathbf{P}_i$ maps the D-dim residual into a smaller latent $\hat{\mathbf{X}}_i$ of dimension d. $\mathbf{X}_i$ is found through the VQ process and then projected out through $\mathbf{Q}_i$ back into the D-dim space, forming the next residual.

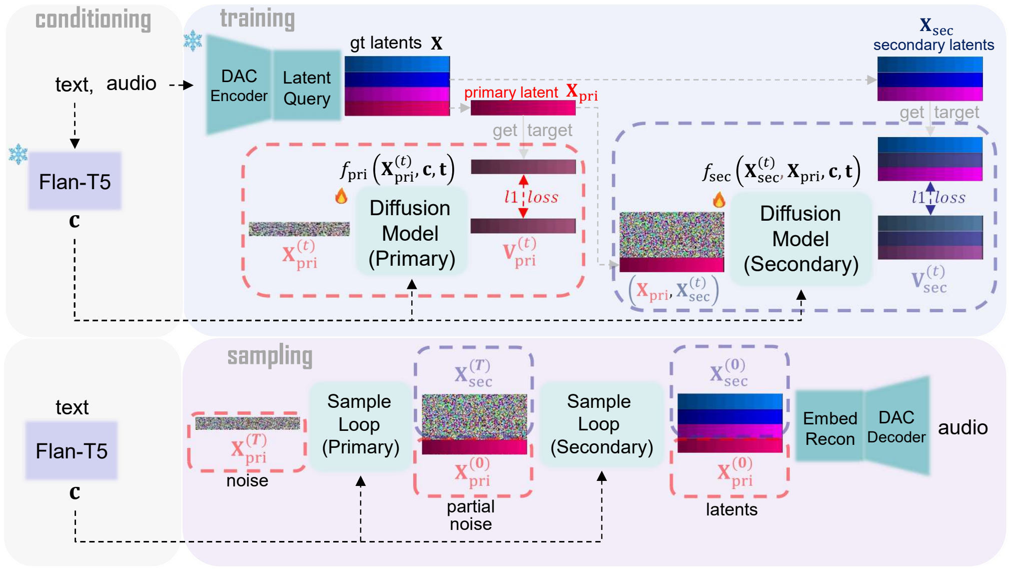

Figure 2: `DiscoDiff` training and sampling pipeline. Our diffusion model is trained to generate the continuous DAC latents, which are obtained through first encoding the audio and then applying the latent query. "Latent query" is the process described in Eq. 1 and Eq. 2, while "embed recon" refers to the computation described in Eq. 4. The term $\mathbf{V}^{(t)}$, the direct output of diffusion denoising U-net, refers to the "v" objective in v-diffusion.

3.2 Generation Patterns

Due to the nature of the RVQ process, it is reasonable to expect a hierarchical structure among the $K = 9$ embeddings: the first embedding $\mathbf{X}_0$ is likely to capture the most significant and coarse features of $\mathbf{Z}$, while subsequent embeddings $\mathbf{X}_{1,\ldots,8}$ capture increasingly refined details. This inherent structure raises an important question: in what order should the 9 smaller embeddings be generated?

Several patterns have been explored in prior work [12]. We refer the reader to Figure 1 of [12] for a schematic representation of these possible generation patterns. Through experimentation, we found out that the "Coarse First Pattern" proposed in VALL-E [6] performs the best for diffusion models given a fixed training compute budget. This pattern involves splitting the $K = 9$ embeddings into two groups: the primary and secondary latent

$$\mathbf{X}_{\text{pri}} = \mathbf{X}_0 \in \mathbb{R}^{d \times L}, \quad \mathbf{X}_{\text{sec}} = \text{concat}(\mathbf{X}_1, \ldots, \mathbf{X}_{K-1}) \in \mathbb{R}^{(K-1)d \times L} \tag{3}$$

3.3 `DiscoDiff`

Fig. 2 illustrates our general pipeline, in which two diffusion U-nets with 1.2B parameter each, are trained from scratch, while the parameters of the DAC and conditioning models remain frozen.

Conditioning: Our approach employs text embedding **c** obtained from Flan-T5 Large [24] model, which is introduced into the diffusion models via cross-attention mechanisms. The weights of the Flan-T5 model are kept frozen during the training process.

Training: The primary and secondary models $f_{\text{pri}}, f_{\text{sec}}$ are trained independently, but both using an ℓ_1 loss function. The training input for the secondary model is the concatenation of the ground-truth primary latent $\mathbf{X}_0^{(0)}$ with the noisy secondary latents $\mathbf{X}_{\text{sec}}^{(t)}$. The secondary model is specifically trained to denoise the secondary latents.

Sampling: Given a text input, text embedding $\mathbf{c}$ is first computed with Flan-T5 model. Then, the sampling process involves two diffusion sampling loops—primary and secondary—to produce the DAC latents. First, we sample the primary latent $\tilde{\mathbf{X}}_{\text{pri}} \in \mathbb{R}^{d \times L}$, or equivalently $\tilde{\mathbf{X}}_0$, through a T-step diffusion sampling loop with primary model, starting from Gaussian noise $\mathbf{X}_{\text{pri}}^{(T)} \sim \mathcal{N}(0, \mathbf{I}_{d \times L})$. Next, we sample the secondary latents $\tilde{\mathbf{X}}_{\text{sec}}$ with the secondary model through a T-step diffusion sampling loop, starting from Gaussian noise $\mathbf{X}_{\text{sec}}^{(T)} \sim \mathcal{N}(0, \mathbf{I}_{(K-1)d \times L})$, which yields the next latents $\tilde{\mathbf{X}}_1, \ldots, \tilde{\mathbf{X}}_{K-1}$. Finally, all latents are combined to reconstruct the embedding $\tilde{\mathbf{Z}} \in \mathbb{R}^{D \times L}$ using:

$$\tilde{\mathbf{Z}} = \sum_{i=0}^{K-1} \tilde{\mathbf{Z}}_i, \quad (\tilde{\mathbf{Z}}_i = \mathbf{Q}_i \check{\mathbf{X}}_i,\ \check{\mathbf{X}}_i = \arg\min_{\mathbf{X} \in \mathcal{X}_i} \|\mathbf{X} - \tilde{\mathbf{X}}_i\|_2,\ \mathbf{Q}_i \in \mathbb{R}^{D \times d},\ i = 0, \ldots K-1) \tag{4}$$

where $\mathcal{X}_i$ is the i^{th} codebook. Finally, $\tilde{\mathbf{Z}}$ is fed into the DAC decoder to generate the final waveform.

Table 1: Comparison of FAD and CLAP scores between existing models and our model on MusicCaps. `DiscoDiff` w/ PL represents the secondary model only, using ground-truth primary latent. Inference refers to sample generation time, duration to sample length, and SR to sampling rate. We also note whether the checkpoint is publicly available and if the model was trained on open-source data.

Model	$\text{FAD}_{\text{vgg}} \downarrow$	$\text{CLAP}_{\text{score}} \uparrow$	Inference (s)	Duration (s)	SR (kHz)	Checkpoint	OS Data
DAC Enc-Decoded	1.1	0.46	-	-	44.1	-	-
`DiscoDiff` w/ PL	1.3	0.43	8 s	29 s	44.1	-	-
MusicGen-L [12]	3.8	0.31	242 s	95 s	32	yes	no
Riffusion [16]	14.8	0.19	25 s	5 s	44.1	yes	no
Noise2Music [4]	2.1	-	36 s	30 s	24	no	no
AudioLDM2 [18]	3.1	0.22	107 s	10 s	48	yes	yes
Moûsai [17]	7.5	0.23	49.2 s	43 s	48	no	no
`DiscoDiff`	4.1	0.34	16 s	29 s	44.1	yes	yes

4 Experiments

4.1 Datasets

To develop a fully open-source model, we train `DiscoDiff` using two publicly available music datasets: MTG-Jamendo [7] and Free Music Archive (FMA) [8]. Jamendo includes over 55,000 Creative Commons-licensed tracks (3,778 hrs), annotated with genre, instrument, and mood/theme tags. FMA offers 106,574 tracks (8,232 hrs) with genre and artist tags. During training, audio tracks in both datasets are chunked into 29 second segments.

Music Captioning: Since the training sets lack captions, we create them via the following pipeline: 1. Use SALMONN [29] to generate raw descriptions; 2. Summarize tags into concise strings (e.g., "Genre: classical; instruments: harp"); 3. Use ChatGPT-3.5-turbo to create captions by rephrasing raw SALMONN descriptions guided by tags, with preference given to tag over raw descriptions.

Data Cleaning: To avoid degrading model quality, we excluded FMA tracks with low musicality (in common aesthetics) such as experimental music. Hence, we created a high-quality FMA subset resembling the Jamendo dataset in musical characteristics. This involved selecting FMA tracks with a high likelihood of matching Jamendo's profile: 1. Extract CLAP embeddings for each Jamendo track, creating a set $\mathcal{D}_{\text{clap}} = \{\mathbf{c}_{\text{aud}}^{(i)}\}$; 2. Fit a d_c-dim Gaussian distribution $\mathcal{N}(\bar{\mathbf{c}}_{\text{aud}}, \Sigma_{\text{aud}})$ to approximate Jamendo's distribution; 3. Compute log-likelihood $\log p$ of each FMA track's CLAP embedding under this distribution; 4. Rank FMA tracks by their log-likelihood values and select the top 20%.

Based on listening tests, we selected the top 20% of FMA files according to their log-likelihood scores. We release the likelihood scores of the FMA dataset.[2]

Test Dataset: For evaluation, we use the MusicCaps dataset [27], which contains 5,521 10-sec captioned music examples. These examples are a curated subset of the AudioCaps dataset [30], with 2,858 samples from the evaluation set and 2,663 samples from the AudioSet training split.

[2]`https://huggingface.co/datasets/disco-eth/jamendo-fma-captions`

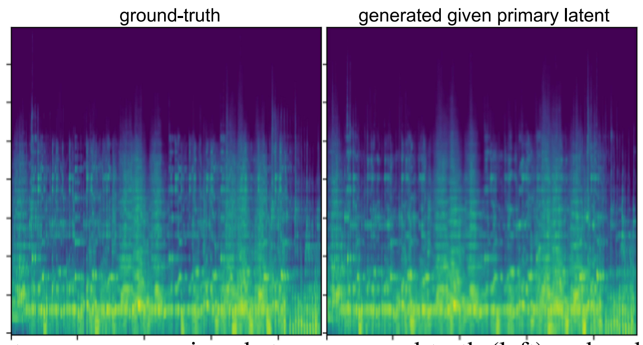

Figure 3: Mel-spectrogram comparison between ground-truth (left) and audio generated by the secondary model given ground-truth primary latents (right).

4.2 Effectiveness of Secondary Model

During evaluation, we found that the secondary model, trained to predict secondary latents $\mathbf{X}\mathrm{sec}$ from primary latent $\mathbf{X}\mathrm{pri}$, can nearly reconstruct the original audio. As shown in Fig. 3, the generated audio is almost identical to the ground truth, based on mel-spectrograms. This suggests that primary latents capture core content, while secondary latents refine details, validating our coarse-to-fine approach where the primary model generates coarse contents and the secondary model refines them.

4.3 Evaluation

We test `DiscoDiff` on the evaluation set of MusicCaps with two main metrics, as shown in Table 1.

FAD (Fréchet Audio Distance): FAD [31] assesses the quality of generated audio by comparing audio embeddings between generated samples and ground-truth audio, using VGGish for extraction. Lower FAD values indicate better quality.

CLAP Score: This cosine similarity between CLAP audio and text embeddings [28] measures alignment between generated audio and its caption. A higher score means better semantic relevance.

Despite being trained on a limited amount of publicly available data, `DiscoDiff` demonstrates competitive performance when compared to closed-source diffusion models. In terms of audio quality and conditioning effectiveness, `DiscoDiff` surpasses both Riffusion and Mousai. However, due to the smaller and less diverse training data compared to models such as Stable Audio and MusicGen, our model produces audio with less diversity, which accounts for its comparatively lower performance in this area. Thanks to the use of a diffusion-based approach, `DiscoDiff` achieves faster inference times than most existing models, particularly the autoregressive architectures.

5 Conclusion

We introduced a coarse-to-fine generation approach within a latent diffusion framework to generate DAC latents, which can be decoded to 44.1 kHz audio. This approach builds on the key insight that the primary latent dictates the core audio content, while the secondary latents refine the finer audio details. Consequently, we first generate the primary latent and then conditionally generate the secondary latents. Furthermore, we release high-quality synthetic captions and a cleaned version of the FMA dataset, improving its musical quality.

Acknowledgment

We would like to thank LAION, Hessian.ai, and TU Darmstadt for providing their high-performance computation infrastructure. Furthermore, we thank Christoph Schuhmann, Vishal Vishal, Anouar Nabil, Wolfgang Stille, and Björn Plüster for their technical support. The DiscoDiff codebase is built on `Denoising-Diffusion-Pytorch`[3] along with `PyTorch` and `PyTorch Lightning`.

[3]`https://github.com/lucidrains/denoising-diffusion-pytorch`

References

[1] N. Zeghidour, A. Luebs, A. Omran, J. Skoglund, and M. Tagliasacchi, “Soundstream: An end-to-end neural audio codec,” *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, vol. 30, p. 495–507, nov 2021. [Online]. Available: https://doi.org/10.1109/TASLP.2021.3129994

[2] A. Défossez, J. Copet, G. Synnaeve, and Y. Adi, “High fidelity neural audio compression,” *Trans. on Mach. Learn. Res.*, 2023.

[3] R. Kumar, P. Seetharaman, A. Luebs, I. Kumar, and K. Kumar, “High-fidelity audio compression with improved rvqgan,” *ArXiv*, vol. abs/2306.06546, 2023.

[4] Q. Huang, D. S. Park, T. Wang, T. I. Denk, A. Ly, N. Chen, Z. Zhang, Z. Zhang, J. Yu, C. H. Frank, J. Engel, Q. V. Le, W. Chan, and W. Han, “Noise2music: Text-conditioned music generation with diffusion models,” *ArXiv*, vol. abs/2302.03917, 2023.

[5] Z. Evans, C. Carr, J. Taylor, S. H. Hawley, and J. Pons, “Fast timing-conditioned latent audio diffusion,” *ArXiv*, vol. abs/2402.04825, 2024.

[6] S. Chen, S. Liu, L. Zhou, Y. Liu, X. Tan, J. Li, S. Zhao, Y. Qian, and F. Wei, “Neural codec language models are zero-shot text to speech synthesizers,” *ArXiv*, vol. abs/2301.02111, 2023.

[7] D. Bogdanov, M. Won, P. Tovstogan, A. Porter, and X. Serra, “The mtg-jamendo dataset for automatic music tagging,” in *Int. Conf. on Machine Learning*, 2019.

[8] M. Defferrard, K. Benzi, P. Vandergheynst, and X. Bresson, “FMA: A dataset for music analysis,” in *18th Int. Society for Music Information Retrieval Conf. (ISMIR)*, 2017. [Online]. Available: https://arxiv.org/abs/1612.01840

[9] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. W. Senior, and K. Kavukcuoglu, “Wavenet: A generative model for raw audio,” in *The 9th ISCA Speech Synthesis Workshop, Sunnyvale, CA, USA*. ISCA, 2016, p. 125.

[10] P. Dhariwal, H. Jun, C. Payne, J. W. Kim, A. Radford, and I. Sutskever, “Jukebox: A generative model for music,” *ArXiv*, vol. abs/2005.00341, 2020.

[11] Z. Borsos, R. Marinier, D. Vincent, E. Kharitonov, O. Pietquin, M. Sharifi, D. Roblek, O. Teboul, D. Grangier, M. Tagliasacchi, and N. Zeghidour, “Audiolm: A language modeling approach to audio generation,” *IEEE ACM Trans. Audio Speech Lang. Process.*, vol. 31, pp. 2523–2533, 2023.

[12] J. Copet, F. Kreuk, I. Gat, T. Remez, D. Kant, G. Synnaeve, Y. Adi, and A. Défossez, “Simple and controllable music generation,” *ArXiv*, vol. abs/2306.05284, 2024.

[13] J. Ho, A. Jain, and P. Abbeel, “Denoising diffusion probabilistic models,” in *Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 6840–6851.

[14] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, “High-resolution image synthesis with latent diffusion models,” in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2022, pp. 10 684–10 695.

[15] Z. Kong, W. Ping, J. Huang, K. Zhao, and B. Catanzaro, “Diffwave: A versatile diffusion model for audio synthesis,” *ArXiv*, vol. abs/2009.09761, 2020.

[16] S. Forsgren and H. Martiros, “Riffusion - Stable diffusion for real-time music generation,” 2022. [Online]. Available: https://riffusion.com/about

[17] F. Schneider, O. Kamal, Z. Jin, and B. Scholkopf, “Moûsai: Text-to-music generation with long-context latent diffusion,” *ArXiv*, vol. abs/2301.11757, 2023.

[18] H. Liu, Z. Chen, Y. Yuan, X. Mei, X. Liu, D. Mandic, W. Wang, and M. D. Plumbley, “AudioLDM: Text-to-audio generation with latent diffusion models,” in *Proc. of the 40th Int. Conf. on Machine Learning*, vol. 202. PMLR, 23–29 Jul 2023, pp. 21 450–21 474.

[19] K. Chen, Y. Wu, H. Liu, M. Nezhurina, T. Berg-Kirkpatrick, and S. Dubnov, “Musicldm: Enhancing novelty in text-to-music generation using beat-synchronous mixup strategies,” *ArXiv*, vol. abs/2308.01546, 2023.

[20] J. Nistal, M. Pasini, C. Aouameur, M. Grachten, and S. Lattner, “Diff-a-riff: Musical accompaniment co-creation via latent diffusion models,” 2024. [Online]. Available: https://arxiv.org/abs/2406.08384

[21] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, “Exploring the limits of transfer learning with a unified text-to-text transformer,” *J. Mach. Learn. Res.*, vol. 21, no. 140, pp. 1–67, 2020. [Online]. Available: http://jmlr.org/papers/v21/20-074.html

[22] R. Huang, J.-B. Huang, D. Yang, Y. Ren, L. Liu, M. Li, Z. Ye, J. Liu, X. Yin, and Z. Zhao, “Make-an-audio: Text-to-audio generation with prompt-enhanced diffusion models,” *ArXiv*, vol. abs/2301.12661, 2023.

[23] J. Melechovský, Z. Guo, D. Ghosal, N. Majumder, D. Herremans, and S. Poria, “Mustango: Toward controllable text-to-music generation,” *ArXiv*, vol. abs/2311.08355, 2023.

[24] H. W. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, Y. Li, X. Wang, M. Dehghani, and S. B. et al., “Scaling instruction-finetuned language models,” *ArXiv*, vol. abs/2210.11416, 2022.

[25] Y.-A. Chung, Y. Zhang, W. Han, C.-C. Chiu, J. Qin, R. Pang, and Y. Wu, “w2v-bert: Combining contrastive learning and masked language modeling for self-supervised speech pre-training,” *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 244–250, 2021.

[26] Q. Huang, A. Jansen, J. Lee, R. Ganti, J. Y. Li, and D. P. W. Ellis, “Mulan: A joint embedding of music audio and natural language,” in *Int. Society for Music Information Retrieval Conf.*, 2022.

[27] A. Agostinelli, T. I. Denk, Z. Borsos, J. Engel, M. Verzetti, A. Caillon, Q. Huang, A. Jansen, A. Roberts, M. Tagliasacchi, M. Sharifi, N. Zeghidour, and C. Frank, “Musiclm: Generating music from text,” *ArXiv*, vol. abs/2301.11325, 2023.

[28] Y. Wu, K. Chen, T. Zhang, Y. Hui, T. Berg-Kirkpatrick, and S. Dubnov, “Large-scale contrastive language-audio pretraining with feature fusion and keyword-to-caption augmentation,” *ICASSP 2023 - 2023 IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5, 2022.

[29] T. Changli, Y. Wenyi, S. Guangzhi, C. Xianzhao, T. Tian, L. Wei, L. Lu, M. Zejun, and Z. Chao, “SALMONN: Towards generic hearing abilities for large language models,” *arXiv:2310.13289*, 2023.

[30] C. D. Kim, B. Kim, H. Lee, and G. Kim, “AudioCaps: Generating captions for audios in the wild,” in *Proc. of the 2019 Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1*, Jun. 2019, pp. 119–132.

[31] K. Kilgour, M. Zuluaga, D. Roblek, and M. Sharifi, “Fréchet audio distance: A metric for evaluating music enhancement algorithms,” *ArXiv*, vol. abs/1812.08466, 2018.

[32] A. Q. Nichol and P. Dhariwal, “Improved denoising diffusion probabilistic models,” in *Proc. of the 38th Int. Conf. on Machine Learning*, vol. 139. PMLR, 2021, pp. 8162–8171.

A Diffusion Model Choice

In this work, we use Denoising Diffusion Probabilistic Models (DDPM) [13, 32]. DDPMs learn to reverse a Markov chain of T steps. Starting from the original data sample $\mathbf{x}^{(0)}$, Gaussian noise is gradually added in the forward process until the data becomes nearly indistinguishable from pure noise $\mathbf{x}^{(T)} \sim \mathcal{N}(0, \mathbf{I})$. The model is trained to reverse this process, denoising $\mathbf{x}^{(T)}$ step by step to reconstruct $\mathbf{x}^{(0)}$. During generation, the model begins with pure noise $\mathbf{x}^{(T)}$ and iteratively refines it to produce a coherent sample. DDPMs offer advantages over autoregressive models in parallel sample generation, accelerating generation times for long sequences and enhances sample diversity. In our model, we used v-diffusion, where the model learns to predict a velocity term V_t defined as

$$\mathbf{v}^{(t)} = \sqrt{\bar{\alpha}_t}\epsilon^{(t)} - \sqrt{1-\bar{\alpha}_t}\mathbf{x}^{(0)} \tag{5}$$

where $\epsilon^{(t)}$ is the noise term at timestep t and $\bar{\alpha}_t$ is the scheduling coefficient defined in [13]. Sampling can be implemented following the following relationships:

$$\begin{aligned} \hat{\mathbf{x}}^{(0)} &= \sqrt{\bar{\alpha}_t}\mathbf{x}^{(t)} - \sqrt{1-\bar{\alpha}_t}\hat{\mathbf{v}}^{(t)} \\ \hat{\epsilon}_t^{(t)} &= \sqrt{1-\bar{\alpha}_t}\mathbf{x}^{(t)} + \sqrt{\bar{\alpha}_t}\hat{\mathbf{v}}^{(t)} \end{aligned} \tag{6}$$