MusicLDM: Enhancing Novelty in text-to-music Generation Using Beat-Synchronous mixup Strategies

Ke Chen, Yusong Wu, Haohe Liu, Marianna Nezhurina, Taylor Berg-Kirkpatrick, Shlomo Dubnov

MUSICLDM: ENHANCING NOVELTY IN TEXT-TO-MUSIC GENERATION USING BEAT-SYNCHRONOUS MIXUP STRATEGIES

Ke Chen[*1], *Yusong Wu*[*2], *Haohe Liu*[*3], *Marianna Nezhurina*[4], *Taylor Berg-Kirkpatrick*[1], *Shlomo Dubnov*[1]

[1]University of California San Diego (UCSD)
[2]Mila, Quebec Artificial Intelligence Institute, Université de Montréal
[3]Centre for Vision Speech and Signal Processing, University of Surrey
[4]LAION

ABSTRACT

Diffusion models have shown promising results in cross-modal generation tasks, including text-to-image and text-to-audio generation. However, generating music, as a special type of audio, presents unique challenges due to limited availability of music data and sensitive issues related to copyright and plagiarism. In this paper, to tackle these challenges, we first construct a state-of-the-art text-to-music model, MusicLDM, that adapts Stable Diffusion and AudioLDM architectures to the music domain. Then, to address the limitations of training data and to avoid plagiarism, we leverage a beat tracking model and propose two different mixup strategies for data augmentation: beat-synchronous audio mixup and beat-synchronous latent mixup, which recombine training audio directly or via a latent embeddings space, respectively. Such mixup strategies encourage the model to interpolate between musical training samples and generate new music within the convex hull of the training data, making the generated music more diverse while still staying faithful to the corresponding style. In addition to popular evaluation metrics, we design several new evaluation metrics based on CLAP score to demonstrate that our proposed MusicLDM and beat-synchronous mixup strategies improve both the quality and novelty of generated music, as well as the correspondence between input text and generated music.

Index Terms— Music Generation, Audio Synthesis, Diffusion Model, Text-to-Music

1. INTRODUCTION

Text-guided generation tasks have gained increasing attention in recent years and have been applied to various modalities, including text-to-image, text-to-video, and text-to-audio generation. As a special type of audio generation, *text-to-music* generation has many practical applications [1, 2]. For instance, musicians use text-to-music generation to quickly build samples to speed up their creative process and amateur music lovers leverage generated pieces for the purpose of musical education [3]. Diffusion models have shown superior performance in these types of cross-modal generation tasks, including systems like DALLE-2 [4] and Stable Diffusion [5] for text-to-image; AudioLDM [6] and Make-an-Audio [7] for text-to-audio. In the domain of music, text-to-music models include the retrieval-based MuBERT [8], language-model-based MusicLM [9], MusicGen [10], diffusion-based Riffusion [11] and Noise2Music [12].

However, text-to-music generation presents several specific challenges. One of the main concerns is the limited availability of text-music parallel training data [9] compared to other modalities such as text-to-image, making it difficult to train a high-quality conditional model. Further, the effectiveness of diffusion models trained on more modest training sets has not been fully explored. Finally, a potential issue with text-to-music generation is the risk of plagiarism and limited novelty. Diffusion models are capable of memorizing and combining different image objects from training images to create replicas [13, 14], which can lead to highly similar or even identical samples to the training data. Music is often protected by copyright laws, and generating new music that sounds too similar to existing music can lead to legal issues. Therefore, it is important to develop models that can generate novel music while avoiding plagiarism, even when trained on relatively small training datasets.

In this paper, we propose new methods for generating novel text-conditioned musical audio from limited training data. We first construct a text-to-music generation model, MusicLDM, which adapts the AudioLDM [6] architectures. Next, to address the data limitation and encourage novel generations, we adapt an idea from past work in other modalities: mixup [15], which augments data by recombining existing training points through linear interpolation. This augmentation encourages models to interpolate between training data rather than simply memorizing individual training examples, thus is useful in addressing data limitations and plagiarism in music generation. However, for **music generation**, the naive application of mixup is problematic. Simply combining waveforms from two distinct musical pieces leads unnatural and ill-formed music: tempos and beats (as well as other musical elements) are unlikely to match. Thus, we propose two mixup strategies, specifically designed for music generation: beat-synchronous audio mixup (BAM) and beat-synchronous latent mixup (BLM), which first analyze and beat-align training samples before interpolating between audio samples directly or encoding and then interpolating in a latent space, respectively.

We design new metrics that leverage contrastive language-audio pretraining (CLAP) to test for plagiarism and novelty in text-to-music generation. Experimental results demonstrate that our beat-synchronous mixup augmentation strategies substantially reduce the amount of copying in generated outputs. Further, our proposed MusicLDM[1], in combination with mixup, achieves better overall musical audio quality as well as better correspondence between output audio and input text. In both objective and subjective evaluations, MusicLDM stands as a SoTA model at the task of text-to-music generation while only being trained on 9K text-music sample pairs.

*The first three authors have equal contribution. We would like to thank IRCAM — Project REACH for supporting this project.

[1]music samples: https://musicldm.github.io
code: https://github.com/RetroCirce/MusicLDM/

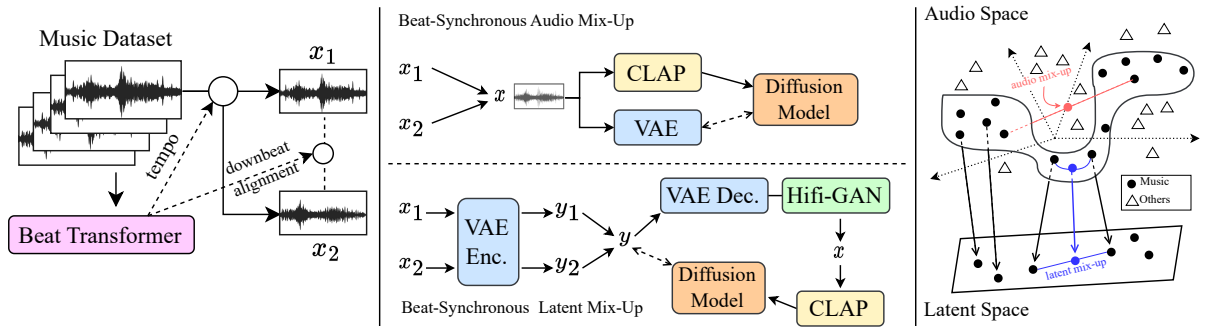

Fig. 1: Mixup strategies in MusicLDM. Left: tempo grouping and downbeat alignment via Beat Transformer. Middle: BAM and BLM mixup strategies, and MusicLDM training components (dot arrows indicate the training process). Right: How BAM and BLM are applied in the feature space of audio and latent variables.

2. METHODOLOGY

2.1. MusicLDM

As shown in the middle of Figure 1, MusicLDM has similar architecture as AudioLDM [6] : a contrastive language-audio pretraining (CLAP) module [16], a latent diffusion module [6] with a pretrained audio variational auto-encoder (VAE) [17], and a Hifi-GAN neural vocoder [18]. Given an audio waveform $\boldsymbol{x} \in \mathbb{R}^T$ and corresponding text, where T the sample length, we feed them into three modules:

1. We pass $\boldsymbol{x}$ through the audio encoder [19] of CLAP $f_{audio}(\cdot)$, to obtain a D-dimensional semantic audio embedding $\boldsymbol{E}_x^a \in \mathbb{R}^D$.
2. We pass the text of x through the text encoder [20] of CLAP $f_{text}(\cdot)$, to to obtain the semantic text embedding $\boldsymbol{E}_x^t \in \mathbb{R}^D$.
3. We transform $\boldsymbol{x}$ into in the mel-spectrogram $\boldsymbol{x}_{mel} \in \mathbb{R}^{T \times F}$. Then we pass $\boldsymbol{x}_{mel}$ into the VAE encoder, to obtain an audio latent representation $\boldsymbol{y} \in \mathbb{R}^{C \times \frac{T}{P} \times \frac{F}{P}}$, where T the mel-spectrogram frame size, F the number of mel bins, C the latent channel size of VAE, and P the downsampling rate of VAE.

In MusicLDM, the latent diffusion model has a UNet architecture where each encoder or decoder block is composed of a ResNet layer [21] and a spatial transformer layer [5]. The output of the diffusion model is the estimated noise $\boldsymbol{\epsilon}_\theta(\boldsymbol{z}_n, n, \boldsymbol{E}_x)$ from n to $(n-1)$ time step in the reverse process, where θ is the parameter group of the diffusion model, and $\boldsymbol{z}_n$ is the n-step feature generated by the forward process. We adopt the training objective [22,23] as the mean square error (MSE) loss function:

$$L_n(\theta) = \mathbb{E}_{\boldsymbol{z}_0, \boldsymbol{\epsilon}, n} ||\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\boldsymbol{z}_n, n, \boldsymbol{E}_x)||_2^2 \tag{1}$$

where $\boldsymbol{z}_0 = \boldsymbol{y}$ is the audio latent representation from VAE (i.e., the groundtruth), and $\boldsymbol{\epsilon}$ is the target noise for training.

For MusicLDM, we make two changes from the original AudioLDM to enhance its performance on text-to-music generation. First, we retrained the CLAP on text-music pair datasets to improve its understanding of music data and corresponding texts. We also retrained the Hifi-GAN vocoder on music data to ensure high-quality transforms from mel-spectrograms to music waveforms. Second, in the original AudioLDM, the model is only fed with audio embeddings as the condition during the training process, i.e., $\boldsymbol{E}_x = \boldsymbol{E}_x^a$; and it is fed with text embeddings to perform the text-to-audio generation, i.e., $\boldsymbol{E}_x = \boldsymbol{E}_x^t$. Although CLAP is trained to learn joint embeddings for text and audio, it does not explicitly enforce the embeddings to be distributed similarly in the latent space, which can make it challenging for the model to generate coherent text-to-audio outputs solely with audio-to-audio training.

To further investigate this task, we introduce two additional training approaches for comparison:

1. Train the MusicLDM directly using the text embedding as the condition, i.e., $\boldsymbol{\epsilon}_\theta(\boldsymbol{z}_n, n, \boldsymbol{E}_x^t)$.
2. Train the MusicLDM first with the audio embedding, then fine-tune it with text embedding, i.e.,$\boldsymbol{\epsilon}_\theta(\boldsymbol{z}_n, n, \boldsymbol{E}_x^a) \rightarrow \boldsymbol{\epsilon}_\theta(\boldsymbol{z}_n, n, \boldsymbol{E}_x^t)$.

The first approach follows the original target of text-to-audio, serving as a comparison with audio-to-audio training. The second approach is proposed as an improvement on audio-to-audio generation as we shift the condition distribution from the audio embedding back to the text embedding during the training of the diffusion model.

2.2. Beat-Synchronous Mixup

As shown in Figure 1, we propose two mixup strategies to augment the data during the MusicLDM training: Beat-Synchronous Audio Mixup (BAM) and Beat-Synchronous Latent Mixup (BLM).

2.2.1. Beat-tracking via Beat Transformer

Musical compositions are structured according to several musical principles, such as chord progressions, arranging instruments by timbres, creating rhythmic patterns and more. Among all of these, beats play a crucial role in alignment of simultaneous voices. In audio retrieval tasks, mixup is a popular technique that is used to augment the training data by randomly mixing different audio clips with matching tempos and beats. To achieve this, we use a SoTA beat tracking model, Beat Transformer [25], extracting the tempo and downbeat map of each music track, as shown in the left of Figure 1. We categorize each music track into different tempo groups and mix tracks only within the same group by allowing small tempo differences. Furthermore, we align the tracks according to their downbeat maps by selecting a certain downbeat to serve as the starting position for the mixup track, resulting in mixup tracks that are neatly ordered in terms of tempo and matching downbeats.

2.2.2. Beat-Synchronous Audio Mixup

As depicted in the upper part of Figure 1, once we select two aligned music tracks $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, we mix them by randomly selecting a mixing ratio from the beta distribution $\lambda \sim \mathcal{B}(5, 5)$, referred by the original work [15]:

$$\boldsymbol{x} = \lambda \boldsymbol{x}_1 + (1 - \lambda)\boldsymbol{x}_2 \tag{2}$$

We then use the mixed data $\boldsymbol{x}$ to obtain the CLAP embedding $\boldsymbol{E}_x$ and the audio latent variable $\boldsymbol{y}$. We train the latent diffusion model using the standard pipeline. This beat-synchronous audio mixup strategy is referred to as BAM.

Table 1: The evaluation of generation quality among MusicLDMs and baselines. AA-Train. and TA-Train. refer to the audio-audio training scheme and the text-audio traning scheme. MusicGen and MusicLDM are works in the same period.

Model	Training Data Size	AA-Train.	TA-Train.	$\text{FD}_{pann}\downarrow$	$\text{FD}_{vgg}\downarrow$	Inception Score ↑	KL Div. ↓
Riffusion [11]	—	✗	✓	68.95	10.77	1.34	5.00
MuBERT [8]	—	—	—	31.70	19.04	1.51	4.69
AudioLDM (w/. original CLAP) [6]	455 hours	✓	✗	38.92	3.08	1.67	3.65
Moûsai [24]	2500 hours	✗	✓	30.73	10.59	1.50	3.88
MusicGen* [10]	20000 hours	✗	✓	**25.19**	**2.17**	**1.82**	**3.10**
MusicLDM		✓	✗	26.67	2.40	**1.81**	3.80
MusicLDM (Only TA-Training)		✗	✓	32.40	2.51	1.49	3.96
MusicLDM w/. mixup	455 hours	✓	✗	30.15	2.84	1.51	3.74
MusicLDM w/. BAM		✓	✗	28.54	**2.26**	1.56	3.50
MusicLDM w/. BLM		✓	✗	**24.95**	2.31	1.79	**3.40**
MusicLDM w/. Text-Finetune		✓	✓	27.81	1.75	1.76	3.60
MusicLDM w/. BAM & Text-Finetune	455 hours	✓	✓	28.22	1.81	1.61	3.61
MusicLDM w/. BLM & Text-Finetune		✓	✓	**26.34**	**1.68**	**1.82**	**3.47**

2.2.3. Beat-Synchronous Latent Mixup

As depicted in the lower part of Figure 1, in the latent diffusion model, the mixup process can also be applied on the latent variables, referred as beat-synchronous latent mixup (BLM). After selecting two aligned music tracks $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, we feed them into the VAE encoder to obtain the latent variables $\boldsymbol{y}_1$ and $\boldsymbol{y}_2$. We then apply the mixup operation to the latent variables:

$$\boldsymbol{y} = \lambda \boldsymbol{y}_1 + (1-\lambda)\boldsymbol{y}_2 \tag{3}$$

In contrast to BAM, BLM applies the mixup operation to the latent space of audio, where we cannot ensure that the mixture of the latent variables corresponds to the actual mixture of the music features. Therefore, we first generate a mixed mel-spectrogram $\boldsymbol{x}_{mel}$ by feeding the mixed latent variable $\boldsymbol{y}$ into the VAE decoder. Then, we feed $\boldsymbol{x}_{mel}$ to the Hifi-GAN vocoder to obtain the mixed audio $\boldsymbol{x}$ as the input music. With $\boldsymbol{x}$ and $\boldsymbol{y}$, we follow the pipeline to train the MusicLDM.

2.2.4. What are BAM and BLM doing?

In the right of Figure 1, we demonstrate the interpolation between the feature space of audio when using BAM and BLM. In the feature space of audio signals, the "•" represents the feature point of music data, while the "△" denotes the feature point of other audio signals, such as natural sound, audio activity, and noise. During the pretraining process of VAE, a latent space is constructed for encoding and decoding the music data by transforming the original feature space into a lower-dimensional music manifold, on which any feature point is considered to be a valid representation of music.

BAM and BLM are concerned with augmenting the data at different levels of feature space. BAM linearly combines two points in audio space to form a new point on the red line. BLM, represented by the blue line, performs a similar operation but result in a new point in the VAE-transformed latent space, which will be decoded back onto the music manifold of audio space.

Both BAM and BLM encourage the model to learn a more continuous distribution over audio feature space, or implicitly from the latent space to the audio space, which can improve the model's generalization performance and mitigate overfitting. These mixup strategies have the potential to mitigate the limitations of data size and help avoid music plagiarism issues. By introducing small variations through mixup, the model can touch a more rich space of music data and generate music samples that are related to, but show differences with, the original training data. In Section 3.2, we evaluated whether these strategies mitigate the data limitation and plagiarism issues.

3. EXPERIMENTS

In this section, we conducted three experiments on our proposed method. First, we trained MusicLDM with different mixup strategies and compared them with available baselines. Second, we evaluated MusicLDM in terms of text-music relevance, novelty and plagiarism risk via metrics based on CLAP scores. Finally, we conducted a subjective listening test to give an additional evaluation.

3.1. Experimental Setup

In this work, we finetuned the pretrained CLAP model on music datasets in addition to its original training data, allowing it to better understand the relation between music and textual descriptions. The new CLAP model is trained on dataset of 2.8 Million text-audio pairs, including extra music data in this link[2], with an approximate total duration of 20 000 hours. For MusicLDM, we used the Audiostock dataset [16] for training, which provides correct text descriptions for correspoinding music tracks. Specifically, the Audiostock dataset contains 9000 music tracks for training and 1000 tracks for testing with the total duration of 455.6 hours. We trained all MusicLDM modules with music clips of 10.24 seconds at 16 kHz sampling rate. In both VAE and diffusion model, music clips are represented as mel-spectrograms with T=1024 frames and F=128 mel-bins. Unlike AudioLDM, MusicLDM's VAE utilizes a downsampling rate of P=8 and a latent dimension of C=16. The architecture and training process of MusicLDM's latent diffusion model follow those of AudioLDM. We present full hyperparameters and training details on the appendix page[3].

3.2. Generation Quality

We used the **audioldm_eval** library[4] to adopt frechet distance (FD), inception score (IS), and kullback-leibler (KL) divergence to evaluate the quality of generated musical audio outputs. We used two standard audio embedding models: VGGish [26] and PANN [27]. The resulting distances are denoted as FD_{vgg} and FD_{pann}. We compared the groundtruth audio from the Audiostock 1000-track test set with the 1000 tracks of music generated by each system from the corresponding textual descriptions.

Table 1 presents the results for our models in comparison with baselines. We sent textual descriptions from the test set to the official APIs of Riffusion, MuBERT, Moûsai, and MusicGen to generate corresponding audio results. Both Riffusion and MuBERT were

[2]https://github.com/LAION-AI/audio-dataset/blob/main/data_collection

[3]appendix: https://musicldm.github.io/appendix

[4]https://github.com/haoheliu/audioldm_eval

Model	Objective Metrics			Subjective Listening Test		
	T-A Similarity	SIM_{AA}@90	SIM_{AA}@95	Quality	Relevance	Musicality
MusicLDM	**0.281**	0.430	0.047	1.98	2.17	**2.19**
MusicLDM-mixup	0.234	**0.391**	0.028	—	—	—
MusicLDM-BAM	0.266	0.402	0.027	2.04	2.21	2.01
MusicLDM-BLM	0.268	0.401	**0.020**	**2.13**	**2.31**	2.07

Table 2: The objective metrics of relevance and plagiarism risk and the subjective test of MusicLDM variants.

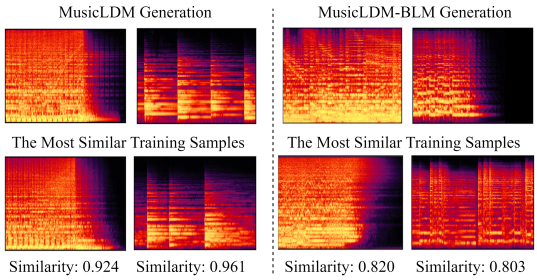

Fig. 2: The generation examples and their most similar tracks in the Audiostock training set.

unable to achieve results comparable to the remaining models. The sub-optimal performance of Riffusion resulted from poor music generation quality, while MuBERT generated high-quality pieces from real sample libraries but fell short in replicating the distribution of Audiostock dataset. Moûsai and MusicGen yielded much better generation quality by leveraging advanced model architectures as well as the large scale of internal training data. We also retrained the original AudioLDM model on the Audiostock dataset but rely on the original CLAP models for condition embeddings. Throughout all models, we observed that MusicLDM variants, trained on only 455 hours music tracks, are able to achieve competitive FD_{pann}, FD_{vgg}, and IS scores with only a slightly inferior on the KL divergence score to MusicGen. This underscores the efficacy of CLAP model pretrained for music, providing more suitable music embeddings as conditioning information.

We also observe the inferior results of "MusicLDM (Only TA-Training)" in comparison to audio-to-audio training variants. This suggests that a gap exists between distributions of text and audio embeddings, making it challenging to generate high-quality audio solely from text embedding. This hypothesis is further supported by the results of combining audio-to-audio training and text-to-audio fine-tuning. We observe a significant decrease in FD_{vgg} with small changes in FD_{pann} and IS, indicating a substantial improvement in generation quality, driven by leveraging both audio and text embeddings during training. Last, we compared MusicLDM with different mixup strategies (mixup, BAM, BLM). The comparison reveals the negative impact of the simple mixup on all metrics. This degradation lies in the inability of the simple mixup strategy to guarantee musicality in its mixture. Similar observations are evident in the BAM results, indicated by the drops in FD_{pann} and IS. However, the tempo and downbeat alignments of BAM counterbalance this defect, enhancing the model generalization ability and improving certain metrics. BLM aligns with our hypothesis that latent space mixup yields audio closely resembling music, allowing us to largely bypass the potential confusion issues tied to audio mixing, thus capitalizing on the ability of mixup to drive generalization and prevent copying. Besides, incorporating text-finetuning results in a further improvement, solidifying BLM as the most effective strategy.

3.3. Text-Audio Relevance, Novelty and Plagiarism

We proposed two metric groups, **text-audio similarity** and **nearest-neighbor audio similarity ratio** to assess text-audio relevance, novelty, and plagiarism risk in various models. Text-audio similarity measures the relevance between the text and the audio. It is defined as the cosine similarity between the groundtruth text embedding $\boldsymbol{E}_{gd}^{t}$ from the test set and the audio embedding $\boldsymbol{E}^{a}$ from generated music. Also, to measure if models are directly copying training samples, we compute the cosine similarity between the audio embedding of each generated track to all audio embeddings from the training set and obtain the maximum (i.e., the similarity of the nearest-neighbor in the training set). Then, we compute the fraction of generated outputs whose nearest-neighbors are above a threshold similarity. We refer this as SIM_{AA}@90 where the threshold is 0.9, and SIM_{AA}@95 with 0.95. **The lower this fraction, the lower the risk of plagiarism.**

Table 2 presents the results of these metrics on the 1000 tracks in the Audiostock test set and the generated music from MusicLDM variants. We did not include other baseline models because they are not trained with the Audiostock dataset, making the plagiarism detection on them pointless. We can observe that the original MusicLDM without mixup achieves the highest text-audio relevance with an average score of 0.281, but also the highest (worst) nearest-neighbor audio similarity ratio. MusicLDM with the simple mixup strategy achieves the lowest SIM_{AA}@90 ratio while sacrificing a lot in the relevance of the generation. The MusicLDM with BAM and BLM achieve a balance between the audio similarity ratios and the text-to-audio similarity. We further conduct the subjective listening test on three models to evaluate the actual hearing experience of generations. We excluded MusicLDM-mixup because of the low perceptual quality of its generations as characterized by instrumental interference and noises. 15 subjects were invited to listen to 6 groups of the generations and rated the music in terms of quality, relevance, and musicality. We observe that the samples of MusicLDM with BAM or BLM mixup achieve a better relevance and quality than those of the original MusicLDM, with an inferior on musicality but still maintaining above an acceptable threshold. In Figure 2, we also give examples of how BLM helps MusicLDM prevent the plagiarism risk and generate novel music samples with a lower T-A similarity to the groundtruth than the original model. More examples and the detail of the subjective test can be found at the appendix page.

Above all, we can conclude that BLM is the best mixup strategy in terms of quality, relevance and novelty of the generated audio.

4. CONCLUSION

We introduce MusicLDM, a text-to-music generation model that incorporates CLAP, VAE, Hifi-GAN, and latent diffusion models. We enhance MusicLDM by proposing two efficient mixup strategies: beat-synchronous audio mixup (BAM) and beat-synchronous latent mixup (BLM) during the training process. We conduct comprehensive evaluations on different variants of MusicLDM using objective and subjective metrics, assessing quality, text-music relevance, and novelty. The experimental results demonstrate the effectiveness of BLM as an effective mixup strategy for text-to-music generation. It is noted that beat tracking, as well as other MIR algorithms, is not perfect in the current stage, thus there must be some error propagation when the beats and downbeats are poorly estimated. In the future, we plan to investigate how to mitigate such errors via semi-supervised learning and incorporate more controls into the text-to-music task.

5. REFERENCES

[1] Jean-Pierre Briot, Gaëtan Hadjeres, and François-David Pachet, *Deep learning techniques for music generation*, vol. 1, Springer, 2020.

[2] Rebecca Fiebrink and Baptiste Caramiaux, "The machine learning algorithm as creative musical tool," *arXiv preprint:1611.00379*, 2016.

[3] Kristin Yim and Hema Manickavasagam, "Turn ideas into music with musiclm," https://blog.google/technology/ai/musiclm-google-ai-test-kitchen/.

[4] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen, "Hierarchical text-conditional image generation with clip latents," *arXiv preprint:2204.06125*, 2022.

[5] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer, "High-resolution image synthesis with latent diffusion models," in *Proc. CVPR*, 2022, pp. 10684–10695.

[6] 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," *Proc. ICML*, 2023.

[7] 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:2301.12661*, 2023.

[8] MubertAI, "Mubert: A simple notebook demonstrating prompt-based music generation," .

[9] 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:2301.11325*, 2023.

[10] Jade Copet, Felix Kreuk, Itai Gat, Tal Remez, David Kant, Gabriel Synnaeve, Yossi Adi, and Alexandre Défossez, "Simple and controllable music generation," *arXiv preprint:2306.05284*, 2023.

[11] Seth Forsgren and Hayk Martiros, "Riffusion - Stable diffusion for real-time music generation," 2022.

[12] 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:2302.03917*, 2023.

[13] Gowthami Somepalli, Vasu Singla, Micah Goldblum, Jonas Geiping, and Tom Goldstein, "Diffusion art or digital forgery? investigating data replication in diffusion models," *arXiv preprint:2212.03860*, 2022.

[14] Nicholas Carlini, Jamie Hayes, Milad Nasr, Matthew Jagielski, Vikash Sehwag, Florian Tramer, Borja Balle, Daphne Ippolito, and Eric Wallace, "Extracting training data from diffusion models," *arXiv preprint:2301.13188*, 2023.

[15] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz, "Mixup: Beyond empirical risk minimization," *Proc. ICLR*, 2017.

[16] Yusong Wu*, Ke Chen*, Tianyu Zhang*, Yuchen Hui*, Taylor Berg-Kirkpatrick, and Shlomo Dubnov, "Large-scale contrastive language-audio pretraining with feature fusion and keyword-to-caption augmentation," in *Proc. ICASSP*. IEEE, 2023, pp. 1–5.

[17] Diederik P Kingma and Max Welling, "Auto-encoding variational bayes," *Proc. ICLR*, 2013.

[18] Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae, "HifiGAN: Generative adversarial networks for efficient and high fidelity speech synthesis," *Proc. NeurIPS*, vol. 33, pp. 17022–17033, 2020.

[19] Ke Chen, Xingjian Du, Bilei Zhu, Zejun Ma, Taylor Berg-Kirkpatrick, and Shlomo Dubnov, "Hts-at: A hierarchical token-semantic audio transformer for sound classification and detection," in *Proc. ICASSP*. IEEE, 2022, pp. 646–650.

[20] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint:1907.11692*, 2019.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Proc. CVPR*, 2016, pp. 770–778.

[22] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Proc. NeurIPS*, 2020.

[23] Z. Kong, W. Ping, J. Huang, K. Zhao, and B. Catanzaro, "Diffwave: A versatile diffusion model for audio synthesis," in *Proc. ICLR*, 2021.

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

[25] Jingwei Zhao, Gus Xia, and Ye Wang, "Beat transformer: Demixed beat and downbeat tracking with dilated self-attention," *Proc. ISMIR*, 2022.

[26] 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 *Proc. ICASSP*. IEEE, 2017, pp. 131–135.

[27] Qiuqiang Kong, Yin Cao, and Turab Iqbal et al., "Panns: Large-scale pretrained audio neural networks for audio pattern recognition," *IEEE Trans. Audio, Speech, Lang. Process.*, 2020.