

INSTRUCT-MUSICGEN: UNLOCKING TEXT-TO-MUSIC EDITING FOR MUSIC LANGUAGE MODELS VIA INSTRUCTION TUNING

Yixiao Zhang¹, Yukara Ikemiya², Woosung Choi², Naoki Murata², Marco A. Martínez-Ramírez², Liwei Lin³, Gus Xia³, Wei-Hsiang Liao², Yuki Mitsufuji², Simon Dixon¹

¹ C4DM, Queen Mary University of London

² Sony AI

³ Music X Lab, MBZUAI

first.last@qmul.ac.uk, first.last@sony.com, {gus.xia, 114270}@nyu.edu

ABSTRACT

The task of text-to-music editing, which employs text queries to modify music (e.g. by changing its style or adjusting instrumental components), presents unique challenges and opportunities for AI-assisted music creation. Previous approaches in this domain have been constrained by the necessity to train specific editing models from scratch, which is both resource-intensive and inefficient; other research uses large language models to predict edited music, resulting in imprecise audio reconstruction. In this paper, we introduce *Instruct-MusicGen*, a novel approach that finetunes a pretrained MusicGen model to efficiently follow editing instructions such as adding, removing, or separating stems. Our approach involves a modification of the original MusicGen architecture by incorporating a text fusion module and an audio fusion module, which allow the model to process instruction texts and audio input concurrently and yield the desired edited music. Remarkably, although Instruct-MusicGen only introduces ~8% new parameters to the original MusicGen model and only trains for 5K steps, it achieves superior performance across all tasks compared to existing baselines. This advancement not only enhances the efficiency of text-to-music editing but also broadens the applicability of music language models in dynamic music production environments. ^{1 2}

1. INTRODUCTION

The rapid advances in text-to-music generation have opened up new possibilities for AI-assisted music creation [1–5]. This paradigm shift has also sparked a growing interest in developing models that offer greater control-

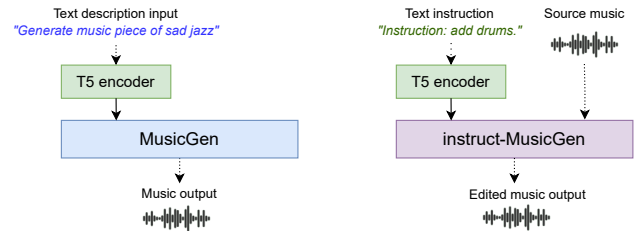


Figure 1: Comparison between MusicGen and instruct-MusicGen. Instruct-MusicGen accepts both audio input and editing instruction text as conditions.

lability [6–9] and editability [10–12] over the music generation process. In music production, a stem—a mixed group of tracks often related by instrument type (like drums or lead vocals)—is essential for mixing and mastering because it allows producers to isolate, adjust, and manipulate individual elements of a song. Following the definition in MusicMagus [11], “text-to-music editing” involves using textual queries to modify various aspects of a music recording, which can be categorised into two main types: intra-stem editing, which focuses on modifying a single stem (e.g., changing the instrument, timbre, or performance style), and inter-stem editing, which involves altering the relationships among stems (e.g., adding, removing, or separating stems). Our work mainly focuses on the problem of inter-stem editing.

Previous attempts to develop text-based music editing models have encountered several challenges. Some approaches [10, 13] have focused on training specialised editing models from scratch, which is resource-intensive and may not yield results comparable to state-of-the-art music generation models. Other work [12, 14, 15] has sought to leverage existing large language models (LLMs) and MusicGen [2], allowing the LLM to interpret editing instructions without further training the music model. Although this approach offers flexibility, it often lacks the ability to precisely reconstruct the conditional audio, leading to unreliable results. To address these limitations, an ideal solution should harness the knowledge embedded in pretrained models to ensure high-quality audio output while adapting the architecture to accommodate the specific requirements

¹ Code, model weights and demo are available at: <https://github.com/ldzhangyx/instruct-musicgen>.

² This work was done during Yixiao Zhang’s internship at Sony AI.



of music editing tasks.

In this paper, we introduce Instruct-MusicGen, a novel approach that applies an instruction-following tuning strategy to the pretrained MusicGen model, enhancing its ability to follow editing instructions effectively without finetuning all its parameters. As shown in Figure 1, by incorporating an audio fusion module based on LLaMA-Adapter [6, 16] and a text fusion module based on LoRA [17] into the original MusicGen architecture, we allow the model to process both precise audio conditions and text-based instructions simultaneously, which the original MusicGen does not do. This enables Instruct-MusicGen to perform a range of editing tasks. In this paper, we focus on a specific set of these tasks: adding, separating, and removing stems. To train Instruct-MusicGen, we synthesize an instructional dataset using the Slakh2100 dataset [18], introducing only 8% additional parameters compared to the original model, and finetune the model for only 5K steps, which is less than 1% of training a music editing model from scratch.

We evaluate Instruct-MusicGen on two datasets: the Slakh test set and the out-of-domain MoisesDB dataset [19]. Our model outperforms existing baselines and achieves performance comparable to models specifically trained for individual tasks. This demonstrates the effectiveness of our approach in leveraging pretrained models for text-to-music editing while maintaining high-quality results.

2. RELATED WORK

Text-based music editing provides a flexible approach for editing music using textual queries. This method is similar to those used in other modalities that require editing, such as image [20, 21] and video [22, 23] editing. In text-to-music editing, text is used to specify precise alterations to existing music compositions. Previous research such as AUDIT [13] and InstructME [10] developed a diffusion model trained with paired music editing data. Additionally, models like M²UGen [12], Loop Copilot [14], MusicAgent [24], ComposerX [25] and WavCraft [26] use large language models (LLMs) for reasoning and regenerate music with external music generation models. Furthermore, GMSDI [27] attempts to model a joint multi-stem distribution of music for text-based generation and separation. Certain models focus exclusively on specific tasks within music editing, such as conditioned generation [6, 7, 9] and separation [28], along with intra-stem editing tasks such as text-based timbre transfer and style transfer [11, 29–31].

The task of inter-stem music editing is closely related to stem-wise music generation. Although not directly tied to text-based controls, some research focuses on modeling stem-wise representations to enable simultaneous stem generation and separation. For instance, Jen-1 Composer [32] and MSDM [33] jointly model the distribution of music with four stems using a diffusion model. The abilities of most existing stem-wise music models are restricted to a fixed set of 4 stems, which limits flexibility but enhances controllability. Besides, StemGen [34] trains

a LLaMA-based auto-regressive model for flexible stem-wise audio generation.

Our work distinguishes itself from these existing efforts in several key ways. First, rather than developing a new model from scratch or strictly adhering to a fixed set of stems, we leverage the power of a pretrained music language model, MusicGen, and enhance it with instruction tuning. This approach not only reduces the computational cost but also retains the high audio quality of the original MusicGen model. Furthermore, our method introduces minimal additional parameters and requires significantly less training, demonstrating a more efficient and scalable solution for text-based music editing.

3. METHOD

3.1 MusicGen

The original MusicGen consists of three components: (1) the EnCodec [35] audio encoder and decoder, which compress music audio waveforms into latent codes and reconstruct them back into waveforms; (2) a multi-layer transformer architecture that models sequences of latent codes, capturing higher-level music representations and efficiently modeling internal relationships within music audio; and (3) the T5 [36] text encoder, which converts text descriptions into embeddings for text-conditioned generation.

EnCodec employs Residual Vector Quantization (RVQ) [37] to compress audio into tokens using multiple codebooks, where each quantizer encodes the quantization error from the previous one. For a reference audio $X \in \mathbb{R}^{d \cdot f_n}$, where d is the duration and f_n is the sample rate, EnCodec compresses X into $Q \in \{1, \dots, L\}^{N \times d \cdot f_s}$, where L is the RVQ codebook size, N is the number of codebooks, and f_s is the latent code sample rate ($f_s \ll f_n$). In MusicGen, $N = 4$, $f_n = 50$, $f_s = 32000$, and $L = 2048$. Finally, the transformer models the sequence relationships over latent codes³.

3.2 Instruct-MusicGen

MusicGen is a text-to-music generation model, capable of generating music audio from a given text prompt. However, MusicGen cannot edit existing music audio. To address this limitation, we introduce Instruct-MusicGen, which transforms MusicGen into a model that can follow editing instructions to modify existing music audio.

Instruct-MusicGen takes a music audio input X^{cond} and a text instruction X^{instruct} (e.g., "Add guitar") as inputs. The model then edits the music audio X^{cond} according to the instruction X^{instruct} and generates the desired edited music X^{music} . As illustrated in Figure 2, Instruct-MusicGen incorporates two additional modules into the vanilla MusicGen: an audio fusion and a text fusion module.

³ MusicGen’s encodec uses a 50Hz sample rate, which is different from the original 75Hz EnCodec model.

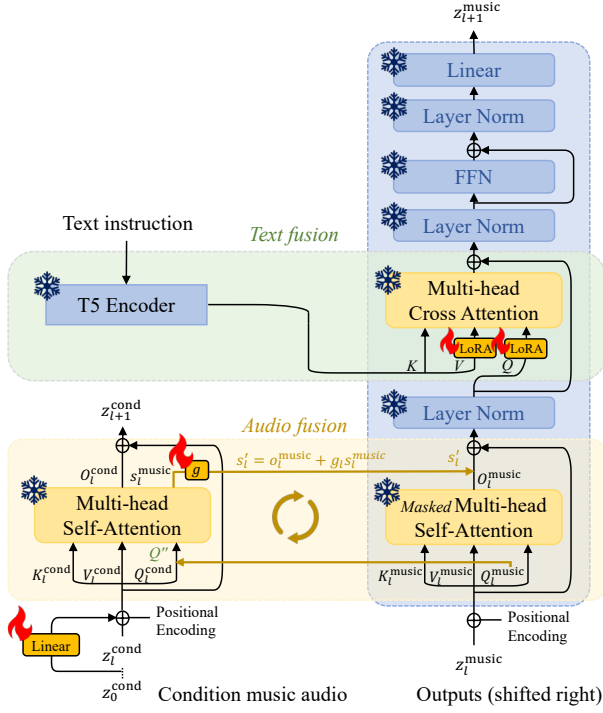


Figure 2: Illustration of the fusion mechanism inside the Transformer module of instruct-MusicGen. The audio fusion module transforms the conditional music audio into embeddings using a duplicated encoder and integrates these embeddings into the MusicGen decoder. The text fusion module modifies the cross-attention mechanism to handle text instructions by finetuning specific layers while keeping the text encoder parameters frozen.

3.2.1 Audio Fusion Module

The audio fusion module enables Instruct-MusicGen to accept external audio inputs, which is inspired by LLaMA-Adapter [16] and Coco-mulla [6]. The lower part of Figure 2 illustrates the audio fusion module. Initially, we convert X^{cond} into EnCodec tokens, followed by re-encoding these tokens into the embedding z^{cond} through the pre-trained embedding layers of MusicGen. Similarly, we transform X^{music} into the pretrained embedding z^{music} .

The module begins with duplicating self-attention modules of the pretrained MusicGen model to extract latent representations of z^{cond} . Given that MusicGen consists of M layers, we denote

$$Z^{cond} = \{z_0^{cond}, z_1^{cond}, \dots, z_M^{cond}\}, \quad (1)$$

$$Z^{music} = \{z_0^{music}, z_1^{music}, \dots, z_M^{music}\}, \quad (2)$$

which represent the hidden states of X^{cond} and X^{music} respectively. Note that we use a learnable input embedding as z_0^{cond} and initialize z_0^{music} with z^{music} .

We compute the vanilla self attention for X^{music} as follows:

$$Q_l^{music}, K_l^{music}, V_l^{music} = \text{QKV-projector}(z_l^{music}), \quad (3)$$

$$o_l^{music} = \text{SelfAttn}(Q_l^{music}, K_l^{music}, V_l^{music}). \quad (4)$$

We project z^{cond} to a high-dimension representation h via a linear layer f_l and learnable positional encoding e_l ,

$$h = f_l(z^{cond}) + e_l. \quad (5)$$

Then, we compute the $(l + 1)$ -th layer hidden states of X^{cond} as follows:

$$Q_l^{cond}, K_l^{cond}, V_l^{cond} = \text{QKV-projector}(z_l^{cond} + h), \quad (6)$$

$$z_{l+1}^{cond} = \text{SelfAttn}(Q_l^{cond}, K_l^{cond}, V_l^{cond}). \quad (7)$$

To fuse information of X^{cond} into X^{music} , we compute the cross attention between them,

$$s_l^{music} = \text{CrossAttn}(Q_l^{music} + Q_l^{cond}, K_l^{cond}, V_l^{cond}). \quad (8)$$

Finally, the attention output of X^{music} is updated as follows,

$$s'_l = o_l^{music} + g_l \cdot s_l^{music}, \quad (9)$$

$$z_{l+1}^{music} = \text{TextFusion}(s'_l, X^{instruct}), \quad (10)$$

where g is a zero-initialized learnable gating factor.

Thus, the total trainable parameters in Instruct-MusicGen include the input embedding z_0^{cond} , linear layers f_l , learnable position embeddings e_l , learnable gating factors g , and learnable parameters in the text fusion module.

3.2.2 Text Fusion Module

To replace the text description input with instruction input, we modify the behavior of the current text encoder. We achieve this by finetuning only the cross-attention module between the text embedding and the music representations while keeping the text encoder's parameters frozen.

The instruction is embedded and encoded by the T5 text encoder as $z^{instruct} = \text{T5}(X^{instruct})$. For efficient finetuning of the cross-attention module, we apply LoRA to the query and value projection layers. Thus, we expand Equation 10 as follows,

$$Q_l, K_l^{instruct}, V_l^{instruct} = \text{QKV-Lora}(s'_l, z^{instruct}), \quad (11)$$

$$z_{l+1}^{music} = \text{CrossAttn}(Q_l, K_l^{instruct}, V_l^{instruct}). \quad (12)$$

During fine-tuning, only query and value projection layers are trainable in the text fusion module.

4. EXPERIMENTS

We conduct both subjective experiments and objective experiments for evaluation, and also provide example spectrograms in Figure 3.

4.1 Objective Experiments

4.1.1 Dataset

For our objective evaluations, we utilise two distinct datasets, each serving a specific purpose in assessing both in-domain and out-of-domain performance capabilities of various models.

1. **Slakh2100 dataset** [18]. The Synthesized Lakh (Slakh) Dataset, originally derived from the Lakh MIDI Dataset v0.1, comprises audio tracks synthesised using high-quality sample-based virtual instruments. This dataset features 2100 tracks complete with corresponding MIDI files.
2. **MoisesDB dataset** [19]. The MoisesDB dataset includes 240 real audio tracks sourced from 45 diverse artists spanning twelve musical genres. Uniquely, MoisesDB organises its tracks into a detailed two-level hierarchical taxonomy of stems, offering a varied number of stems per track, each annotated with textual descriptions.

The rationale for selecting two datasets lies in their diverse configurations and common applications. While the Slakh dataset is traditionally utilised for training models tailored to a four-stem arrangement, our model, Instruct-MusicGen, although initially trained on this dataset, is designed to generalise to various stem configurations. Conversely, models such as InstructME and AUDIT are trained on private or larger, more diverse datasets. By employing both Slakh2100 and MoisesDB, we ensure a comprehensive evaluation, allowing us to fairly compare the adaptability and performance of different models under varying conditions of data familiarity and complexity.

4.1.2 Data Preprocessing

We utilised the Slakh2100 dataset to construct an instruction-based dataset for our experiments, employing the following pipeline:

- A data point was randomly selected from the Slakh training dataset.
- An instruction was chosen from a predefined set $\{add, remove, extract\}$ along with a target stem. Subsequently, n other stems were selected from the remaining stems.
- An offset was randomly determined to cut a 5-second audio clip. If the target stem contained more than 50% silence, a different offset was selected.
- The stems were mixed according to the specified instructions to create a triplet consisting of $\{instruction\text{ text}, condition\text{ audio input}, audio\text{ ground truth}\}$.

4.1.3 Experimental Setup

For the finetuning of MusicGen, we jointly trained the audio fusion module and the text fusion module. The optimisation process utilised the AdamW optimiser, with a learning rate set at 5×10^{-3} . We use L2 loss over latent tokens as the training objective. Training incorporated a Cosine Annealing scheduler with an initial warmup of 100 steps. The training regimen extended over 5,000 steps with an accumulated batch size of 32, achieved through setting the batch size to 8 and using gradient accumulation over 4 iterations. The finetuning process was executed on a single NVIDIA A100 GPU and was completed within two days.

4.1.4 Baselines

In this section, we explore two baseline models, each distinguished by their unique methodologies for handling audio data.

1. **AUDIT** [13]: AUDIT is an instruction-guided audio editing model, consisting of a variational autoencoder (VAE) for converting input audio into a latent space representation, a T5 text encoder for processing edit instructions, and a diffusion network that performs the actual audio editing in the latent space. The system accepts mel-spectrograms of input audio and edit instructions, and generates the edited audio as output.
2. **M²UGen** [12]: The M²UGen framework leverages large language models to comprehend and generate music across various modalities, integrating abilities from external models such as MusicGen [2] and AudioLDM 2 [4]. It is designed to stimulate creative outputs from diverse sources, showcasing robust performance in multi-modal music generation.

Besides, InstructME can also perform instruction-guided music editing and remixing with latent diffusion models. We exclude it from comparison because InstructME’s model weights and evaluation protocol are not publicly released.

Model	Param size	Dataset	Hours (h)	Steps
AUDIT	942M (1.5B)	Multiple	~6500	0.5M
InstructME	967M (1.7B)	Multiple	417	2M
M ² UGen	637M (~9B)	MUEdit	60.22	-
Ours	264M (3.5B)	Slakh	145	5K

Table 1: Comparison of different models, where the param size numbers are trainable parameters and total parameters respectively. Our model has the lowest parameter size, and only requires 5K training steps.

4.1.5 Metrics

The metrics to evaluate model performance are listed below.

1. **Fréchet Audio Distance (FAD)** [38]⁴ measures the similarity between two sets of audio files by comparing multivariate Gaussian distributions fitted to feature embeddings from the audio data. We use the FAD score to evaluate the overall audio quality of the predicted music.
2. **CLAP Score (CLAP)** [39]⁵ is used in our experiments to measure the correspondence between the edited music and a target text. For the removal task, the target text is generated by deleting the name of the removed instrument from the original text.

⁴ <https://github.com/gudgud96/frechet-audio-distance>.

⁵ <https://github.com/LAION-AI/CLAP>.

3. **Kullback-Leibler Divergence (KL)**⁶ assesses the difference between the probability distributions of audio features from two sources, indicating information loss when approximating one distribution with another. A low KL score indicates the predicted music shares similar features with the ground truth.
4. **Structural Similarity (SSIM)** [40] is an image quality metric that we adapt to evaluate structural similarity between predicted music and ground truth.
5. **Scale-Invariant Signal-to-Distortion Ratio (SI-SDR)** [41] quantifies audio quality, especially in source separation tasks. It is scale-invariant, useful for varying audio volumes, and measures distortion relative to a reference signal. We use SI-SDR to evaluate the signal loss of the predicted audio.
6. **Scale-Invariant Signal-to-Distortion Ratio improvement (SI-SDRi)** [42] extends SI-SDR, measuring the improvement in signal-to-distortion ratio before and after processing. It is commonly used in audio enhancement and separation contexts.

To further investigate whether the model successfully adds, removes or extracts the instrument, we propose the **P-Demucs score** to evaluate the model performance. This metric specifically focuses on detecting the presence of a newly added instrument in the generated audio. It leverages the Demucs model, a source separation model, to isolate the target instrument from the audio. After separation, the root-mean-square energy (RMSE) of the isolated track is analyzed. For example, if the instruction is to "add guitar," a non-silent guitar track is regarded as a successful edit.

4.1.6 Objective Experiment Results

Our evaluation of Instruct-MusicGen demonstrates its superior performance across various tasks compared to existing text-to-music editing baselines (AUDIT, InstructME, M²UGen). On the Slakh dataset (Table 2), Instruct-MusicGen excelled in adding, removing, and extracting stems, achieving the lowest Fréchet Audio Distance (FAD) and highest CLAP and SSIM scores. It also significantly improved the signal-to-noise ratio (SI-SDR) in the removal task, showing balanced performance across all metrics and proving its robustness in various editing scenarios. Similarly, in the MoisesDB dataset evaluations (Table 3), Instruct-MusicGen demonstrated strong performance, with the best performance on most metrics over the three tasks.

We find that all models exhibit negative SI-SDR and SI-SDRi scores, which is a common occurrence when evaluating generative models on a signal level. These metrics are typically designed for source separation tasks and are not entirely fair to generative models, as they penalise even minor discrepancies between the generated and original signals. Generative models like Instruct-MusicGen often focus on producing perceptually plausible audio rather than perfectly matching the original signal at a technical level.

⁶https://github.com/haoheliu/audioldm_eval.

4.2 Subjective Experiments

4.2.1 Experimental Setup

We conducted a subjective listening test to evaluate the model's performance.⁷ This test involved disseminating an online survey within the Music Information Retrieval (MIR) community and our broader research network, which resulted in the collection of 30 complete responses. The gender distribution of the participants was 23 males (76.7%) and 7 females (23.3%). Regarding professional musical education experience, 4 participants (13.3%) had less than 1 year of experience, 13 (43.3%) had between 1 and 5 years, and 13 participants (43.3%) had more than 5 years of experience. For the data preparation, we randomly selected a subset of data points from the objective test dataset. Specifically, 6 audio samples were chosen, comprising 2 audio samples for each subtask (add, remove, extract). Each data point included results from the baseline models, our models, and the ground truth from the dataset.

4.2.2 Metrics

1. **Instruction Adherence (IA)** assesses how accurately the generated music follows the given editing instruction. In this experiment, participants rate the generated music on a scale from 1 to 5, where 1 indicates that the instruction was not followed at all, and 5 indicates that the instruction was followed perfectly. For example, if the instruction is "Remove Drums," a rating of 1 would mean that the drums were not removed at all, while a rating of 5 would mean that the drums were completely removed.
2. **Audio Quality (AQ)** evaluates the overall audio quality of the generated music in comparison to the original music. Participants rate the audio quality on a scale from 1 to 5, where 1 represents very poor quality with significant degradation compared to the original music, and 5 represents excellent quality, as good as or better than the original music. This metric helps in understanding how the editing process affects the overall sound quality of the music.

4.2.3 Subjective Experiment Results

The results of our subjective experiments are summarised in Table 4. We conducted two paired t-tests with Bonferroni correction, setting the significance level at $\alpha = 0.05$. The results shows that our model demonstrates a significant improvement in both Instruction Adherence (IA) and Audio Quality (AQ) compared to the baseline models, AUDIT and M²UGen.⁸

5. CONCLUSION

In this paper, we introduced Instruct-MusicGen, a novel approach to text-to-music editing that fosters joint musical

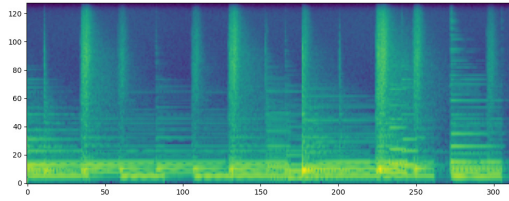
⁷ This subjective test was approved by the ethics committee of Sony.

⁸ More audio samples can be found at <https://bit.ly/instruct-musicgen>.

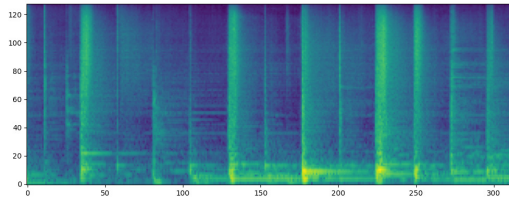
Task	Models	FAD↓	CLAP↑	KL↓	SSIM↑	P-Demucs↑	SI-SDR↑	SI-SDRi↑
Add	AUDIT	6.88	0.12	1.02	0.21	0.53	-	-
	M ² UGen	7.24	0.22	0.99	0.20	0.43	-	-
	Ours	3.75	0.23	0.67	0.26	0.80	-	-
Remove	AUDIT	15.48	0.07	2.75	0.35	0.33	-45.60	-47.28
	M ² UGen	8.26	0.09	1.59	0.23	0.70	-44.20	-46.13
	Ours	3.35	0.12	0.66	0.45	0.76	-2.09	-3.77
Extract	AUDIT	15.08	0.06	2.38	0.42	0.61	-52.90	-50.16
	M ² UGen	8.14	0.11	2.15	0.31	0.60	-46.38	-43.53
	Ours	3.24	0.12	0.54	0.52	0.75	-9.00	-6.15

Table 2: Comparison of text-based music editing models on the Slakh dataset (4 stems).

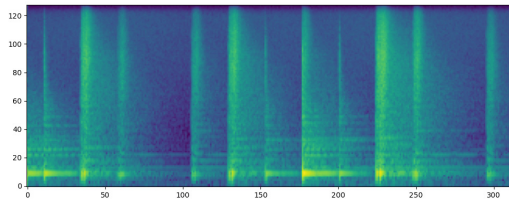
Task	Models	FAD↓	CLAP↑	KL↓	SSIM↑	P-Demucs↑	SI-SDR↑	SI-SDRi↑
Add	AUDIT	4.06	0.12	0.84	0.21	0.50	-	-
	M ² UGen	5.00	0.18	0.83	0.20	0.45	-	-
	Ours	3.79	0.18	0.35	0.35	0.77	-	-
Remove	AUDIT	10.72	0.10	2.46	0.34	0.41	-44.32	-57.10
	M ² UGen	3.75	0.13	1.27	0.19	0.72	-43.94	-56.73
	Ours	5.05	0.10	0.84	0.34	0.78	-13.70	-26.48
Extract	AUDIT	6.67	0.07	1.97	0.45	0.60	-54.53	-56.17
	M ² UGen	5.74	0.08	1.91	0.25	0.52	-42.84	-44.49
	Ours	4.96	0.11	1.36	0.40	0.78	-21.39	-23.03

Table 3: Comparison of text-based music editing models on the MoisesDB dataset.


(a) Input music.



(b) Edited music output.



(c) Ground truth.

Figure 3: Spectrograms when Instruct-MusicGen removes the drum stem.

Model	Instruction Adherence↑	Audio Quality↑
AUDIT	1.54	2.56
M ² UGen	1.70	1.92
Ours	3.85	3.55
Ground truth	4.36	4.21

Table 4: The subjective experiment results.

and textual controls. By finetuning the existing MusicGen model with instruction tuning, Instruct-MusicGen demonstrated its capability of editing music in various ways, including adding, separating and extracting a stem from music audio using textual queries, without the need for training specialised models from scratch. Also, it outperforms various baseline models that are dedicated to specific music editing tasks. Furthermore, our method uses significantly fewer resources than previous models, with a requirement of tuning only 8% of the parameters of the original MusicGen.

6. ACKNOWLEDGEMENTS

This work was done during Yixiao Zhang’s internship at Sony AI. Yixiao Zhang was a research student at the UKRI Centre for Doctoral Training in Artificial Intelligence and Music, supported jointly by the China Scholarship Council, Queen Mary University of London and Apple Inc.

7. REFERENCES

- [1] A. Agostinelli, T. I. Denk, Z. Borsos, J. H. Engel, M. Verzett, A. Caillon, Q. Huang, A. Jansen, A. Roberts, M. Tagliasacchi, M. Sharifi, N. Zeghidour, and C. H. Frank, “MusicLM: Generating music from text,” *CoRR*, vol. abs/2301.11325, 2023. [Online]. Available: <https://doi.org/10.48550/arxiv.2301.11325>
- [2] J. Copet, F. Kreuk, I. Gat, T. Remez, D. Kant, G. Synnaeve, Y. Adi, and A. Défossez, “Simple and controllable music generation,” in *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, Eds., 2023. [Online]. Available: http://papers.nips.cc/paper_files/paper/2023/hash/94b472a1842cd7c56dcb125fb2765fbd-Abstract-Conference.html
- [3] P. Li, B. Chen, Y. Yao, Y. Wang, A. Wang, and A. Wang, “JEN-1: Text-guided universal music generation with omnidirectional diffusion models,” *CoRR*, vol. abs/2308.04729, 2023. [Online]. Available: <https://doi.org/10.48550/arxiv.2308.04729>
- [4] H. Liu, Q. Tian, Y. Yuan, X. Liu, X. Mei, Q. Kong, Y. Wang, W. Wang, Y. Wang, and M. D. Plumbley, “AudioLDM 2: Learning holistic audio generation with self-supervised pretraining,” *CoRR*, vol. abs/2308.05734, 2023. [Online]. Available: <https://doi.org/10.48550/arxiv.2308.05734>
- [5] 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,” *CoRR*, vol. abs/2308.01546, 2023. [Online]. Available: <https://doi.org/10.48550/arxiv.2308.01546>
- [6] L. Lin, G. Xia, J. Jiang, and Y. Zhang, “Content-based controls for music large language modeling,” *CoRR*, vol. abs/2310.17162, 2023. [Online]. Available: <https://doi.org/10.48550/arxiv.2310.17162>
- [7] S.-L. Wu, C. Donahue, S. Watanabe, and N. J. Bryan, “Music controlnet: Multiple time-varying controls for music generation,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 32, pp. 2692–2703, 2024.
- [8] J. Melechovsky, Z. Guo, D. Ghosal, N. Majumder, D. Herremans, and S. Poria, “Mustango: Toward controllable text-to-music generation,” *arxiv preprint arxiv:2311.08355*, 2023.
- [9] L. Lin, G. Xia, Y. Zhang, and J. Jiang, “Arrange, inpaint, and refine: Steerable long-term music audio generation and editing via content-based controls,” *CoRR*, vol. abs/2402.09508, 2024. [Online]. Available: <https://doi.org/10.48550/arxiv.2402.09508>
- [10] B. Han, J. Dai, X. Song, W. Hao, X. He, D. Guo, J. Chen, Y. Wang, and Y. Qian, “InstructME: An instruction guided music edit and remix framework with latent diffusion models,” *CoRR*, vol. abs/2308.14360, 2023. [Online]. Available: <https://doi.org/10.48550/arxiv.2308.14360>
- [11] Y. Zhang, Y. Ikemiya, G. Xia, N. Murata, M. A. M. Ramírez, W. Liao, Y. Mitsufuji, and S. Dixon, “MusicMagus: Zero-shot text-to-music editing via diffusion models,” *CoRR*, vol. abs/2402.06178, 2024. [Online]. Available: <https://doi.org/10.48550/arxiv.2402.06178>
- [12] A. S. Hussain, S. Liu, C. Sun, and Y. Shan, “M²UGen: Multi-modal music understanding and generation with the power of large language models,” *CoRR*, vol. abs/2311.11255, 2023. [Online]. Available: <https://doi.org/10.48550/arxiv.2311.11255>
- [13] Y. Wang, Z. Ju, X. Tan, L. He, Z. Wu, J. Bian, and S. Zhao, “AUDIT: Audio editing by following instructions with latent diffusion models,” in *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, Eds., 2023. [Online]. Available: http://papers.nips.cc/paper_files/paper/2023/hash/e1b619a9e241606a23eb21767f16cf81-Abstract-Conference.html
- [14] Y. Zhang, A. Maezawa, G. Xia, K. Yamamoto, and S. Dixon, “Loop Copilot: Conducting AI ensembles for music generation and iterative editing,” *CoRR*, vol. abs/2310.12404, 2023. [Online]. Available: <https://doi.org/10.48550/arxiv.2310.12404>
- [15] D. Yang, J. Tian, X. Tan, R. Huang, S. Liu, X. Chang, J. Shi, S. Zhao, J. Bian, X. Wu, Z. Zhao, S. Watanabe, and H. Meng, “UniAudio: An audio foundation model toward universal audio generation,” *CoRR*, vol. abs/2310.00704, 2023. [Online]. Available: <https://doi.org/10.48550/arxiv.2310.00704>
- [16] R. Zhang, J. Han, A. Zhou, X. Hu, S. Yan, P. Lu, H. Li, P. Gao, and Y. Qiao, “LLaMA-Adapter: Efficient fine-tuning of language models with zero-init attention,” *CoRR*, vol. abs/2303.16199, 2023. [Online]. Available: <https://doi.org/10.48550/arxiv.2303.16199>
- [17] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, and W. Chen, “LoRA: Low-rank adaptation of large language models,” *CoRR*, vol. abs/2106.09685, 2021. [Online]. Available: <https://arxiv.org/abs/2106.09685>
- [18] E. Manilow, G. Wichern, P. Seetharaman, and J. L. Roux, “Cutting music source separation some Slakh: A dataset to study the impact of training data quality and quantity,” in *2019 IEEE Workshop on Applications*

- of Signal Processing to Audio and Acoustics, WASPAA 2019, New Paltz, NY, USA, October 20-23, 2019.* IEEE, 2019, pp. 45–49. [Online]. Available: <https://doi.org/10.1109/WASPAA.2019.8937170>
- [19] I. Pereira, F. Araújo, F. Korzeniowski, and R. Vogl, “MoisesDB: A dataset for source separation beyond 4-stems,” in *Proceedings of the 24th International Society for Music Information Retrieval Conference, ISMIR 2023, Milan, Italy, November 5-9, 2023*, A. Sarti, F. Antonacci, M. Sandler, P. Bestagini, S. Dixon, B. Liang, G. Richard, and J. Pauwels, Eds., 2023, pp. 619–626. [Online]. Available: <https://doi.org/10.5281/zenodo.10265363>
- [20] T. Brooks, A. Holynski, and A. A. Efros, “InstructPix2Pix: Learning to follow image editing instructions,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*. IEEE, 2023, pp. 18 392–18 402. [Online]. Available: <https://doi.org/10.1109/CVPR52729.2023.01764>
- [21] H. Liu, C. Li, Q. Wu, and Y. J. Lee, “Visual instruction tuning,” in *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, Eds., 2023. [Online]. Available: http://papers.nips.cc/paper_files/paper/2023/hash/6dcf277ea32ce3288914faf369fe6de0-Abstract-Conference.html
- [22] W. Chai, X. Guo, G. Wang, and Y. Lu, “StableVideo: Text-driven consistency-aware diffusion video editing,” in *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*. IEEE, 2023, pp. 22 983–22 993. [Online]. Available: <https://doi.org/10.1109/ICCV51070.2023.02106>
- [23] D. Ceylan, C. P. Huang, and N. J. Mitra, “Pix2Video: Video editing using image diffusion,” in *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*. IEEE, 2023, pp. 23 149–23 160. [Online]. Available: <https://doi.org/10.1109/ICCV51070.2023.02121>
- [24] D. Yu, K. Song, P. Lu, T. He, X. Tan, W. Ye, S. Zhang, and J. Bian, “MusicAgent: An AI agent for music understanding and generation with large language models,” in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023 - System Demonstrations, Singapore, December 6-10, 2023*, Y. Feng and E. Lefever, Eds. Association for Computational Linguistics, 2023, pp. 246–255. [Online]. Available: <https://doi.org/10.18653/v1/2023.emnlp-demo.21>
- [25] Q. Deng, Q. Yang, R. Yuan, Y. Huang, Y. Wang, X. Liu, Z. Tian, J. Pan, G. Zhang, H. Lin *et al.*, “ComposerX: Multi-agent symbolic music composition with LLMs,” *arxiv preprint arxiv:2404.18081*, 2024.
- [26] J. Liang, H. Zhang, H. Liu, Y. Cao, Q. Kong, X. Liu, W. Wang, M. D. Plumbley, H. Phan, and E. Benetos, “WavCraft: Audio editing and generation with large language models,” in *ICLR 2024 Workshop on Large Language Model (LLM) Agents*, 2024.
- [27] E. Postolache, G. Mariani, L. Cosmo, E. Benetos, and E. Rodolà, “Generalized multi-source inference for text conditioned music diffusion models,” *CoRR*, vol. abs/2403.11706, 2024. [Online]. Available: <https://doi.org/10.48550/arxiv.2403.11706>
- [28] X. Liu, Q. Kong, Y. Zhao, H. Liu, Y. Yuan, Y. Liu, R. Xia, Y. Wang, M. D. Plumbley, and W. Wang, “Separate anything you describe,” *CoRR*, vol. abs/2308.05037, 2023. [Online]. Available: <https://doi.org/10.48550/arxiv.2308.05037>
- [29] H. Manor and T. Michaeli, “Zero-shot unsupervised and text-based audio editing using DDPM inversion,” *CoRR*, vol. abs/2402.10009, 2024. [Online]. Available: <https://doi.org/10.48550/arxiv.2402.10009>
- [30] S. Li, Y. Zhang, F. Tang, C. Ma, W. Dong, and C. Xu, “Music style transfer with time-varying inversion of diffusion models,” in *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2024, February 20-27, 2024, Vancouver, Canada*, M. J. Wooldridge, J. G. Dy, and S. Natarajan, Eds. AAAI Press, 2024, pp. 547–555. [Online]. Available: <https://doi.org/10.1609/aaai.v38i1.27810>
- [31] F.-D. Tsai, S.-L. Wu, H. Kim, B.-Y. Chen, H.-C. Cheng, and Y.-H. Yang, “Audio prompt adapter: Unleashing music editing abilities for text-to-music with lightweight finetuning,” *arXiv preprint arXiv:2407.16564*, 2024.
- [32] Y. Yao, P. Li, B. Chen, and A. Wang, “JEN-1 Composer: A unified framework for high-fidelity multi-track music generation,” *CoRR*, vol. abs/2310.19180, 2023. [Online]. Available: <https://doi.org/10.48550/arxiv.2310.19180>
- [33] G. Mariani, I. Tallini, E. Postolache, M. Mancusi, L. Cosmo, and E. Rodolà, “Multi-source diffusion models for simultaneous music generation and separation,” *CoRR*, vol. abs/2302.02257, 2023. [Online]. Available: <https://doi.org/10.48550/arxiv.2302.02257>
- [34] J. D. Parker, J. Spijkervet, K. Kosta, F. Yesiler, B. Kuznetsov, J. Wang, M. Avent, J. Chen, and D. Le, “StemGen: A music generation model that listens,” *CoRR*, vol. abs/2312.08723, 2023. [Online]. Available: <https://doi.org/10.48550/arxiv.2312.08723>

- [35] A. Défossez, J. Copet, G. Synnaeve, and Y. Adi, “High fidelity neural audio compression,” *CoRR*, vol. abs/2210.13438, 2022. [Online]. Available: <https://doi.org/10.48550/arxiv.2210.13438>
- [36] 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, pp. 140:1–140:67, 2020. [Online]. Available: <http://jmlr.org/papers/v21/20-074.html>
- [37] N. Zeghidour, A. Luebs, A. Omran, J. Skoglund, and M. Tagliasacchi, “SoundStream: An end-to-end neural audio codec,” *IEEE ACM Trans. Audio Speech Lang. Process.*, vol. 30, pp. 495–507, 2022. [Online]. Available: <https://doi.org/10.1109/TASLP.2021.3129994>
- [38] K. Kilgour, M. Zuluaga, D. Roblek, and M. Sharifi, “Fréchet audio distance: A reference-free metric for evaluating music enhancement algorithms,” in *Interspeech 2019, 20th Annual Conference of the International Speech Communication Association, Graz, Austria, 15-19 September 2019*, G. Kubin and Z. Kacic, Eds. ISCA, 2019, pp. 2350–2354. [Online]. Available: <https://doi.org/10.21437/Interspeech.2019-2219>
- [39] 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,” in *IEEE International Conference on Acoustics, Speech and Signal Processing ICASSP 2023, Rhodes Island, Greece, June 4-10, 2023*. IEEE, 2023, pp. 1–5. [Online]. Available: <https://doi.org/10.1109/ICASSP49357.2023.10095969>
- [40] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: From error visibility to structural similarity,” *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004. [Online]. Available: <https://doi.org/10.1109/TIP.2003.819861>
- [41] J. L. Roux, S. Wisdom, H. Erdogan, and J. R. Hershey, “SDR - half-baked or well done?” in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2019, Brighton, United Kingdom, May 12-17, 2019*. IEEE, 2019, pp. 626–630. [Online]. Available: <https://doi.org/10.1109/ICASSP.2019.8683855>
- [42] Y. Z. Isik, J. L. Roux, Z. Chen, S. Watanabe, and J. R. Hershey, “Single-channel multi-speaker separation using deep clustering,” in *Interspeech 2016, 17th Annual Conference of the International Speech Communication Association, San Francisco, CA, USA, September 8-12, 2016*, N. Morgan, Ed. ISCA, 2016, pp. 545–549. [Online]. Available: <https://doi.org/10.21437/Interspeech.2016-1176>