Arrange, Inpaint, and Refine: Steerable Long-term Music Audio Generation and Editing via Content-based Controls

Liwei Lin[1,2] , **Gus Xia**[1,2] , **Yixiao Zhang**[3] and **Junyan Jiang**[1,2]
[1]New York University Shanghai
[2]Mohamed bin Zayed University of Artificial Intelligence
[3]C4DM, Queen Mary University of London
{ll4270, gxia, jj2731}@nyu.edu, yixiao.zhang@qmul.ac.uk

Abstract

Controllable music generation plays a vital role in human-AI music co-creation. While Large Language Models (LLMs) have shown promise in generating high-quality music, their focus on autoregressive generation limits their utility in music editing tasks. To bridge this gap, we introduce a novel Parameter-Efficient Fine-Tuning (PEFT) method. This approach enables autoregressive language models to seamlessly address music inpainting tasks. Additionally, our PEFT method integrates frame-level content-based controls, facilitating track-conditioned music refinement and score-conditioned music arrangement. We apply this method to fine-tune MusicGen, a leading autoregressive music generation model. Our experiments demonstrate promising results across multiple music editing tasks, offering more flexible controls for future AI-driven music editing tools. A demo page[1] showcasing our work and source codes[2] are available online.

1 Introduction

In recent years, there has been a surge in the development of deep music generative models, encompassing both audio and symbolic domains [Dhariwal *et al.*, 2020; Copet *et al.*, 2023; Agostinelli *et al.*, 2023; Huang *et al.*, 2018; Huang and Yang, 2020; Wang and Xia, 2021]. Particularly, the emergence of Large Language Models (LLMs) has enabled various systems to achieve high-quality generations. Such progress offers a glimpse into the potential of constructing effective tools for facilitating human-AI co-creation.

While many audio generative models focus on autoregressive music generation [Dhariwal *et al.*, 2020; Copet *et al.*, 2023; Agostinelli *et al.*, 2023], they currently lack the capability of music editing, a common practice in music creation involving iterative refinements of arbitrary audio segments. In this context, a Masked Language Model (MLM) presents a promising alternative due to its ability to incorporate both past and future contexts, enabling non-autoregressive processing.

[1]https://kikyo-16.github.io/AIR/.
[2]https://github.com/Kikyo-16/airgen.

Although some preliminary works have demonstrated high-fidelity music inpainting, they often struggle to capture *long-term* music context dependencies [Zhou *et al.*, 2019; Marafioti *et al.*, 2020; Garcia *et al.*, 2023]. As a result, the effectiveness is limited to short-range gap filling of one or two seconds. Moreover, existing models lack the capacity to accept content-based controls such as chord progressions, piano arrangements, or drum patterns. Consequently, the overall usability and controllability of current MLMs for music editing fall short of meeting realistic demands.

To this end, we introduce AIRGen, a model capable of doing **a**rrangement, **i**npainting, and **r**efinement by applying content-based controls to fine-tune MusicGen [Copet *et al.*, 2023], a state-of-the-art autoregressive music audio language model. Our approach involves a novel Parameter-Efficient Fine-Tuning (PEFT) method, which re-purposes the autoregressive language model to seamlessly solve mask-filling tasks, thereby equipping MusicGen with music inpainting ability (Fig. 1(a)). Moreover, our PEFT method integrates frame-level audio controls into the model architecture, enabling precise content-based manipulation, including (1) drum tracks, (2) chord progression, and (3) piano covers. This enables track-conditioned music refinement (Fig. 1(b)) and score-conditioned music arrangement (Fig. 1(c)).

In summary, the contributions of this work are as follows:

Parameter-efficient heterogeneous adapter: We introduce a novel heterogeneous adapter inspired by LLaMA-Adapter [Zhang *et al.*, 2023], combined with a well-designed masking training scheme. This approach successfully transforms an autoregressive LM into a masked LM, while incorporating sequential controls through Parameter-Efficient Fine-Tuning.

Flexible long-term music editing: As shown in Figure 1, our model enables flexible editing of arbitrary segments of the audio mixture. This includes re-generating (inpainting) the segment based on its past and future contexts, refining the segment based on prescribed audio content (e.g., the drum track), and re-arranging the segment given semantic controls such as chord progressions and piano arrangements.

High-quality steerable Generation: Experimental results demonstrate that our model outperforms existing baselines at the inpainting task. Additionally, for the refinement and arrangement tasks, our model achieves comparable quality to unconditioned autoregressive generation, a capability not ob-

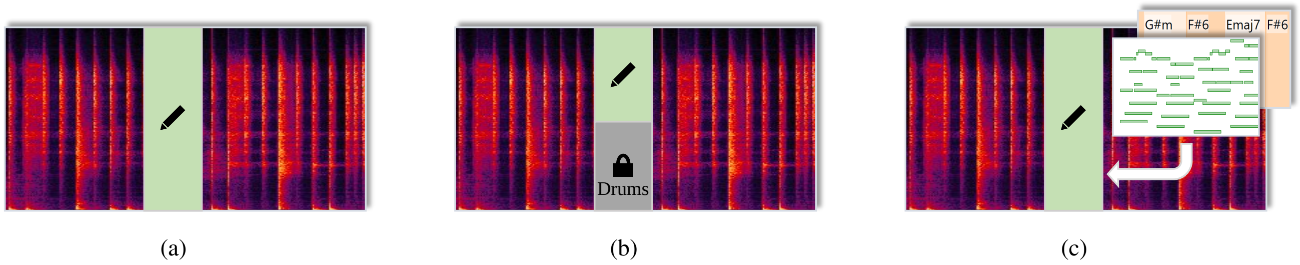


Figure 1: Different music editing tasks accomplished by the model. Light green blocks with pen icons denote the masked parts to generate. (a) To inpaint an entire segment; (b) To refine some tracks conditioned on other tracks (e.g., drums); (c) To arrange the target segment following user-provided piano cover or chord controls.

served in other models.

2 Related Work

2.1 General Audio and Music Generation Models

Music audio generation necessitates extensive contextual modeling to account for the intricate structure of musical language. Recent large-scale music audio generative models, encompassing autoregressive and diffusion-based approaches, have made remarkable strides in capturing such a long-term structure while introducing cross-modal conditions. For example, Jukebox [Dhariwal *et al.*, 2020] leverages VQ-VAE [Van Den Oord *et al.*, 2017] and transformer decoders to achieve lyric and genre-based generation; Diffusion-based Moûsai [Schneider *et al.*, 2023] adopts the pre-trained frozen T5 encoder [Raffel *et al.*, 2020] to summarize text conditions; autoregressive MusicGen [Copet *et al.*, 2023] realizes monophonic melody and text controls by assembling EnCodec [Défossez *et al.*, 2022], T5 encoder, and an acoustic transformer decoder.

2.2 Music Inpainting

Most music generation models adhere to a strict left-to-right temporal order, which limits their scope of application. In contrast, inpainting models support music generation in more diverse and interactive scenarios. To this end, various music inpainting models have been proposed.

In the music audio domain, Perraudin et al. [2018] use a similarity graph to retrieve suitable segments in the source audio to patch up the masked section. Superseding rule-based methods are deep learning methods. Marafioti et al. [2019] use a conv-deconv neural net to predict the masked spectrogram given the past and the future contexts. Both the mask and the contexts are fixed-sized. Their prediction is a point estimate instead of a distribution (trained with an L2 loss), therefore unable to capture the multi-modal nature of music content. One of the solutions is to fully characterize the masked information via info-rich conditions. For example, Zhou et al. [2019] condition the inpainting on the music performance video via a visual-audio joint feature space. However, in general we aim to model uncertainty instead of requiring full information. Therefore, GACELA [Marafioti *et al.*, 2020] uses a generator with a latent condition code as input to address the multi-modality of music content. Still, the latent variable controls the entire segment and does not form a time sequence. Advancements in NLP has shown that the standard solution to multi-modality is the proper language modeling (LM) of token distributions. Vampnet [Garcia *et al.*, 2023] adopts that methodology and uses masked acoustic token modeling to implement music inpainting. Unfortunately, in all the above mentioned works, the inpainting duration is limited to under 2 seconds, and the models are trained from scratch on limited data.

In the symbolic music domain, the inpainting duration is usually longer. Ippolito et al. [2018] achieve mask filling via autoregressive continuation by artificially reordering the music into a [prefix; suffix; masked] setup. Music sketchnet [Chen *et al.*, 2020] provides symbolic inpainting with user-specified pitch and rhythm. Polyffusion [Min *et al.*, 2023] generates symbolic music via diffusion on the piano roll, which supports multi-measure inpainting.

2.3 Parameter-Efficient Fine-tuning

Methods such as appending task-specific prefixes to input sequences [Li and Liang, 2021; Liu *et al.*, 2022] and employing low-rank adaption (LoRA) for fine-tuning pre-trained linear matrices [Hu *et al.*, 2021] have shown promise. Additionally, approaches like LLaMA-Adapter adjust attention outputs using prompt adapters and zero gate scalars, with a multi-modal conditional variant incorporating global image representations [Zhang *et al.*, 2023; Gao *et al.*, 2023]. Inspired by LLaMA-Adapter, we introduce a novel PEFT method in this study, designed to fine-tune large-scale models while accommodating external sequential conditions, transforming an autoregressive language model into a masked language model (MLM).

3 Methodology

In this section, we describe in detail the base LLM to finetune, the well-designed content-based controls, and the proposed heterogeneous adapter.

Algorithm 1 Piano reduction algorithm

```
Input: $M$ = Original MIDI tracks
Output: $P$ = Reduced piano track
 1: Let $\Gamma$ = Piano-like MIDI tracks in $M$.
 2: $\Gamma'$ = Sort all note events in $\Gamma$ by duration in a descending order.
 3: Initiate an empty piano track $P$.
 4: for all Note event $e$ in $\Gamma'$ do
 5:     $e' \leftarrow$ Set program of $e$ to 0.
 6:     if $e'$ does not overlap with any event in $P$ then
 7:         Add $e'$ to $P$.
 8:     end if
 9: end for
10: return $P$
```

3.1 MusicGen

In this study, we take the text-only version of MusicGen as the base LLM. MusicGen is an autoregressive transformer-based model, conditioned on text or melody, using quantized audio tokens from an EnCodec [Défossez *et al.*, 2023] audio tokenizer for high-fidelity music generation. It incorporates Residual Vector Quantization to handle multiple streams of discrete tokens from learned codebooks.

This configuration of MusicGen comprises three key components: a pre-trained EnCodec, a pre-trained T5 encoder, and an acoustic transformer decoder. The transformer decoder is composed of N layers, each featuring a causal self-attention block and a cross-attention block tailored to handle conditioning text prompts. Specifically, our PEFT training process solely modifies the self-attention blocks within the acoustic transformer decoder.

3.2 Content-Based Controls

We categorize content-based controls into two types: external and internal controls. Internal controls involve surface-level manipulation of acoustic tracks, requiring the generated audio to include the condition tracks. External controls, on the other hand, entail conceptual adjustments such as chord progression or piano cover. The apparent challenge is how MusicGen, an audio model, can process the symbolic conditions (e.g., notes, chords) prescribed by the external controls. Our solution is to algorithmically render all symbolic controls into the audio format before inputting them to MusicGen.

In our experiments that demonstrate internal controls, we use drum tracks as the condition, whose training data is obtained using Demucs. The external controls in our experiments include chord progression and piano cover. We derive pseudo chord and beat annotations using Madmom [Böck *et al.*, 2016] and a chord recognition model [Jiang *et al.*, 2019]. During audio synthesis, we render the current chord annotation into a block chord at each beat onset. For piano cover controls, we introduce a rule-based piano reduction algorithm outlined in Algorithm 1 to extract piano covers from provided MIDI annotations. Subsequently, we render the piano reduction to audio.

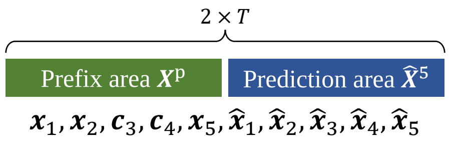

$2 \times T$
Prefix area $\boldsymbol{X}^{\mathrm{p}}$
Prediction area $\hat{\boldsymbol{X}}^5$
$\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{c}_3, \boldsymbol{c}_4, \boldsymbol{x}_5, \hat{\boldsymbol{x}}_1, \hat{\boldsymbol{x}}_2, \hat{\boldsymbol{x}}_3, \hat{\boldsymbol{x}}_4, \hat{\boldsymbol{x}}_5$

Figure 2: Design of the input sequence, with $T = 5$ as an example. The prefix area is followed by the prediction area. The prefix area spans from 1 to T. Here the unmasked locations contain the infilling contexts. At the masked locations, the ground-truth audio is masked away and replaced by the frame-level condition audio track. The prediction area spans from $T+1$ to $2\times T$, where the model is trained to output the ground-truth audio, essentially repeating the prefix area except replacing the condition audio with newly inpainted audio. In this example, the masks happen to be contiguous, while in reality an arbitrary set of frames can be masked.

3.3 Tokenization Design

We extract discrete tokens from both the target and the condition audio sequences using Encodec, operating at a resolution of 50 frames per second. Let $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_T\}$ ($\boldsymbol{X} \in \mathbb{R}^{T\times n}$) and $\boldsymbol{C} = \{\boldsymbol{c}_1, \boldsymbol{c}_2, ..., \boldsymbol{c}_T\}$ ($\boldsymbol{C} \in \mathbb{R}^{T\times n}$) represent the extracted tokens of the target and condition sequences, respectively, where T denotes the total number of frames and n denotes the dimension of a discrete token.

The acoustic transformer decoder generates Encodec tokens sequentially in an autoregressive fashion. To ensure the decoder captures full-context information when inpainting masked locations, we prepend the combination of $\boldsymbol{C}$ and masked $\boldsymbol{X}$ as the prefix of the input sequence to the decoder (Figure 2). Let the prefix $\boldsymbol{X}^{\mathrm{p}} = \{\boldsymbol{x}_1^{\mathrm{p}}, \boldsymbol{x}_2^{\mathrm{p}}, ..., \boldsymbol{x}_T^{\mathrm{p}}\}$ ($\boldsymbol{X}^{\mathrm{p}} \in \mathbb{R}^{T\times n}$) be the combination of $\boldsymbol{C}$ and masked $\boldsymbol{X}$,

$$\boldsymbol{x}_t^{\mathrm{p}} = \begin{cases} \boldsymbol{x}_t, & \text{if } t\text{-th frame is masked} \\ \mathbf{c}_t, & \text{otherwise} \end{cases}. \tag{1}$$

Let $\hat{\boldsymbol{X}}^t = \{\hat{\boldsymbol{x}}_1, \hat{\boldsymbol{x}}_2, ..., \hat{\boldsymbol{x}}_t\}$ ($\hat{\boldsymbol{X}}^t \in \mathbb{R}^{t\times n}$) be the predicted tokens up to t, then the next predicted audio token $\hat{\boldsymbol{x}}_{t+1} \in \mathbb{R}^n$ is,

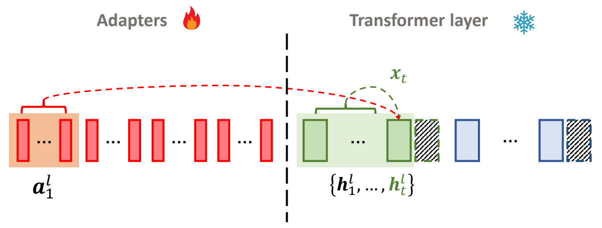
$$\hat{\boldsymbol{x}}_{t+1} = \mathrm{Decoder}([\boldsymbol{X}^{\mathrm{p}}; \hat{\boldsymbol{X}}^t]). \tag{2}$$

During network optimization, we apply cross-entropy loss to the prediction area for both masked and unmasked tokens,

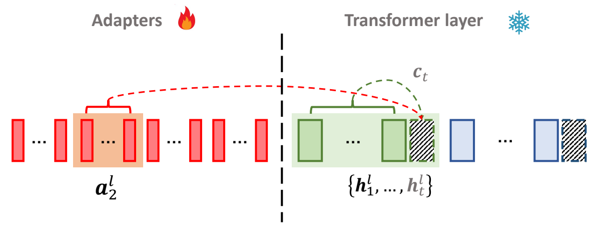
$$L = \mathrm{Cross\_Entropy}(\boldsymbol{X}, \hat{\boldsymbol{X}}^T). \tag{3}$$

3.4 Heterogeneous Adapter

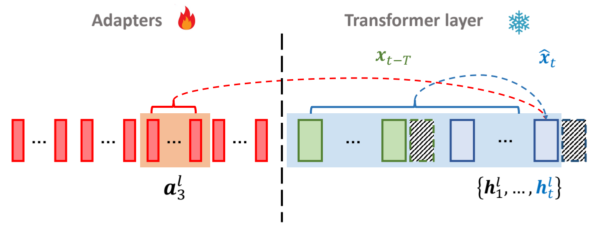
To enhance decoder fine-tuning efficiency, we introduce a heterogeneous adapter, where multiple different lightweight adapters are trained to work together in the same decoder. In vanilla PEFT with adapters, the same type of adapters is applied to all frames since the signal $\boldsymbol{X}^{\mathrm{p}}$ is homogeneous in a typical LM task. In our method, each self-attention layer incorporates four different types of trainable adapters, with each type directed at frames of a specific type, as depicted in Figure 3. This allows a type mixture in the signal $\boldsymbol{X}^{\mathrm{p}}$. The usage of heterogeneous adapters nullifies the need for section separators, in-context examples, or other parsing mechanisms. Let $\{\boldsymbol{a}_1^l, \boldsymbol{a}_2^l, \boldsymbol{a}_3^l, \boldsymbol{a}_4^l\} \in \mathbb{R}^{4\times m\times d}$ represent the adapters of the l-th self-attention layer, where m denotes the



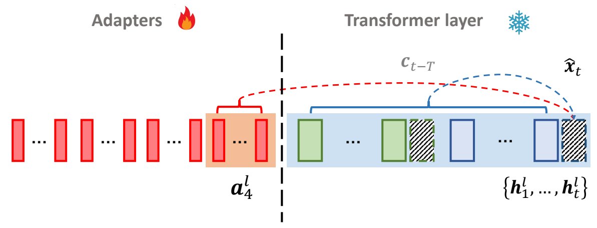
(a) To incorporate the infilling context in the prefix area, the first type of adapters is inserted into MusicGen.



(b) To incorporate the condition audio (at the masked locations) in the prefix area, the second type of adapters is inserted into MusicGen.



(c) To repeat the infilling context in the prediction area, the third type of adapters is inserted into MusicGen.



(d) To generate the inpainted result (at the masked locations) in the prediction area, the fourth type of adapters is inserted into MusicGen.

Figure 3: Heterogeneous adapters, in which we apply four types of adaptors to MusicGen: (a), (b), (c), and (d). For each type of adapter, $\boldsymbol{a}_i^l$ is applied to a specific type of frame. Dashed lines depict the computation of the attention-weighted sum within each self-attention layer of the transformer decoder. Red blocks denote learnable adapters $\boldsymbol{a}_i^l$. Line-filled blocks appear at the masked locations. In the prefix area, green blocks $\boldsymbol{h}_t^l$ correspond to the infilling context $\boldsymbol{x}_t$, and line-filled blocks with green borders $\boldsymbol{h}_t^l$ indicate masked locations containing the given condition $\boldsymbol{c}_t$. In the prediction area, blue blocks $\boldsymbol{h}_t^l$ are trained to eventually output $\hat{\boldsymbol{x}}_{t-T}$ that repeats the infilling context $\boldsymbol{x}_{t-T}$, and line-filled blocks with blue borders $\boldsymbol{h}_t^l$ represent inpainted areas $\hat{\boldsymbol{x}}_{t-T}$ given $\boldsymbol{c}_{t-T}$.

number of adapters per type and d denotes the dimension of the decoder's hidden state. We deploy $\boldsymbol{a}_1^l$ and $\boldsymbol{a}_3^l$ to incorporate the infilling context at the unmasked locations, and deploy $\boldsymbol{a}_2^l$ and $\boldsymbol{a}_4^l$ to take in condition information and perform inpainting at the masked locations.

Let $\boldsymbol{H}^l = \{\boldsymbol{h}_1^l, \boldsymbol{h}_2^l, ..., \boldsymbol{h}_T^l, ..., \boldsymbol{h}_{2T}^l\}$ denote the input representation of the l-th self-attention layer, $\boldsymbol{H}^{l,t} = \{\boldsymbol{h}_1^l, \boldsymbol{h}_2^l, ..., \boldsymbol{h}_t^l\}$ represent the prefix of $\boldsymbol{H}^l$ up to t, and $\boldsymbol{S}^l$ signify the attention output before modification by heterogeneous adapters. Let $\boldsymbol{W}_{\text{q}}^l$, $\boldsymbol{W}_{\text{k}}^l$, and $\boldsymbol{W}_{\text{v}}^l$ denote the matrices that transform the input representation into queries, keys, and values in the decoder, respectively. After being modified by the adapters, the adjusted attention output is given by:

$$\boldsymbol{S'}^l = \boldsymbol{S}^l + \boldsymbol{U}^l, \tag{4}$$

where $\boldsymbol{U}^l = \{\boldsymbol{u}_1^l, \boldsymbol{u}_2^l, ..., \boldsymbol{u}_{2T}^l\}$. Continuing from the attention mechanism in the decoder, $\boldsymbol{U}^l$ is computed as the attention-weighted sum:

$$\boldsymbol{u}_t^l = g_t^l \cdot \text{softmax}(\frac{(\boldsymbol{q}_t^l)^\intercal \boldsymbol{k}_t^l}{\sqrt{d}}), \tag{5}$$

$$\boldsymbol{q}_t^l = \boldsymbol{W}_{\text{q}}^\intercal \boldsymbol{x}_t^l, \tag{6}$$

$$\boldsymbol{k}_t^l = \boldsymbol{W}_{\text{k}}^\intercal \boldsymbol{y}_t^l, \tag{7}$$

$$\boldsymbol{v}_t^l = \boldsymbol{W}_{\text{v}}^\intercal \boldsymbol{y}_t^l, \tag{8}$$

where $\boldsymbol{y}_t^l = [\boldsymbol{H}^{l,t}; \boldsymbol{z}_t^l]$ is the concatenation of learnable adapters $\boldsymbol{z}_t^l$ and the prefix of $\boldsymbol{H}^l$ resulting from a causal mask.

Here, $\boldsymbol{z}_t^l$ is defined as follows. If $g_t^l \in \{b_1^l, b_2^l, b_3^l, b_4^l\}$ is the learnable gating factor initialized to 0 during training. The selections of $\boldsymbol{z}_t^l$ and g_t^l are based on whether the t-th frame is in the prefix area or the prediction area and whether it is masked:

$$\boldsymbol{z}_t^l, g_t^l = \begin{cases} \boldsymbol{a}_1^l, b_1^l, & \text{if } t \le T \text{ and } \boldsymbol{x}_t \in \boldsymbol{X}^{\text{P}} \\ \boldsymbol{a}_2^l, b_2^l, & \text{if } t \le T \text{ and } \boldsymbol{x}_t \notin \boldsymbol{X}^{\text{P}} \\ \boldsymbol{a}_3^l, b_3^l, & \text{if } t > T \text{ and } \boldsymbol{x}_{t-T} \in \boldsymbol{X}^{\text{P}} \\ \boldsymbol{a}_4^l, b_4^l, & \text{otherwise} \end{cases}. \tag{9}$$

Throughout the training, all parameters in the decoder remain frozen except for the adapters $\{\boldsymbol{a}_1^l, \boldsymbol{a}_2^l, \boldsymbol{a}_3^l, \boldsymbol{a}_4^l\}$ and the gates $\{b_1^l, b_2^l, b_3^l, b_4^l\}$ at each layer.

4 Experiments

4.1 Dataset

Our model is trained on the training split of Slakh2100, a synthetic music audio dataset with MIDI annotations [Manilow *et al.*, 2019]. The total training set comprises 1289 instrumental songs with silenced onsets and tails removed. Evaluation is conducted on both the test split (151 songs) of Slakh2100 and RWC-POP100, a dataset featuring 100 real-recording pop songs with MIDI, chord, and beat annotations [Goto *et al.*, 2002].

4.2 Training

We train three models: Drums-AIR, Chord-AIR, and Piano-AIR, tailored for drum-track conditioning, chord-progression

(a) Slakh2100 Test Set

	CLAP$_{\text{src}}$ ↑		**FAD**$_{\text{vgg}}$ ↓	
	Full	Prefix	Full	Prefix
Drum-AIR	0.749	0.756	1.423	1.422
Chord-AIR	0.753	0.757	**1.220**	1.222
Piano-AIR	**0.755**	**0.761**	1.290	1.282
MusicGen	0.656	0.687	1.251	**1.218**
VampNet	0.631	0.643	2.910	3.424

(b) RWC-POP-100

	CLAP$_{\text{src}}$ ↑		**FAD**$_{\text{vgg}}$ ↓	
	Full	Prefix	Full	Prefix
Drum-AIR	**0.619**	**0.627**	1.606	1.691
Chord-AIR	0.614	0.625	1.593	1.681
Piano-AIR	0.611	0.621	**1.531**	**1.623**
MusicGen	0.373	0.441	2.474	2.276
VampNet	0.613	0.618	3.689	3.910

Table 1: Evaluation of the models in terms of inpainting musicality. Adapter width $m = 50$.

conditioning, and piano-cover conditioning, respectively.

Configuration
Each model is trained using two A800's with an initial learning rate of 2e-3, a batch size of 24, and a sample duration of 15 seconds, for 10 epochs, totaling 8 hours of training time. The warm-up epoch is set to 1, and the model is updated using a cross-entropy loss.

Masking Scheme
Throughout the training process, we uniformly sample a mask ratio r from the interval $[0.4, 0.8]$ and generate a binary mask with this ratio. To maintain the continuity of audio segments, we utilize a median filter with a window size of 11 to smooth the mask before applying it to the input Encodec tokens.

4.3 Evaluation

We assess the performance of our models using two distinct datasets: the synthetic instrumental dataset (Slakh2100 Test Set) and a real-recording vocal dataset (RWC-POP-100). From each test song, we extract a 15-second audio segment for evaluation. We present three types of masks to corrupt the test audio, resulting in discontinuous masking areas: one, two, and four masking areas, respectively, referred to as Mask 1, Mask 2, and Mask 3, as shown in Figure 5. The total duration of masking areas sums up to 7.5s for each setting. The metrics we employ are as follows:

1. **Source-to-Distortion Ratio (SDR)**. For Drum-AIR, we quantify the SDR between the inpainted drum tracks (separated by Demucs) and the condition drum tracks, following newest SDR definition from the Music Demixing Challenge (MDX 2021) competition [Mitsufuji *et al.*, 2022].

(a) Slakh2100 Test Set

	Drum$_{\text{SDR}}$ ↓	**Chord**$_{\text{rec}}$ ↑	**Chroma**$_{\text{cos}}$ ↑
Drum-AIR	6.676	-	-
Chord-AIR	-	0.745	0.830
Piano-AIR	-	0.720	0.849

(b) RWC-POP-100

	Drum$_{\text{SDR}}$ ↓	**Chord**$_{\text{rec}}$ ↑	**Chroma**$_{\text{cos}}$ ↑
Drum-AIR	6.531	-	-
Chord-AIR	-	0.650	0.837
Piano-AIR	-	0.615	0.845

Table 2: Evaluation of the models in terms of controllability. Adapter width $m = 50$.

2. **Chord Accuracy**. For both Chord-AIR and Piano-AIR, we compare the inpainted chords (predicted by a chord recognition model) with the condition chord progression. We report weighted recall scores with a time resolution of 20ms for chord accuracy.

3. **Chroma Cosine Similarity**. For both Chord-AIR and Piano-AIR, we evaluate the chroma cosine similarity between the inpainted audios and the condition tracks, with a window size of 2048 and a hop size of 640 for the chromagram.

4. **CLAP Score**. For each model, we report the CLAP score [Wu *et al.*, 2023], assessing the semantic distance between the inpainted audio and the original audio.

5. **Fréchet Audio Distance (FAD)**. Within each test set, we calculate the FAD of VGG (Visual Geometry Group) features between the inpainted audio and the original audio.

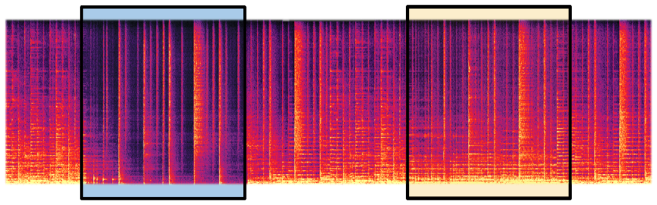
4.4 Baselines

We designate MusicGen [Copet *et al.*, 2023] and VampNet [Garcia *et al.*, 2023] as our baselines, as there are currently no other comparable content-based controllable inpainting models available for comparison. As MusicGen can only perform continuation rather than inpainting, we only evaluate MusicGen using the past context of Mask 1.

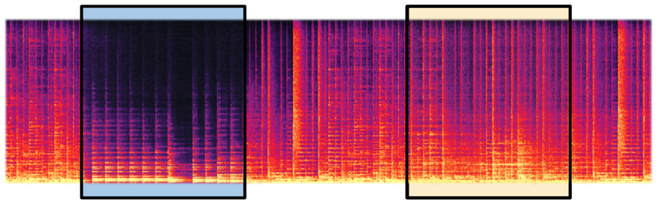
1. **MusicGen**. We reproduce the MusicGen model with the *MusicGen-large* official checkpoint.[3]

2. **VampNet**. VampNet employs a non-autoregressive, bidirectional transformer with a variable masking schedule, enabling high-fidelity music synthesis across various tasks, such as inpainting and outpainting, through flexible inference masks. We reproduce the VampNet model with the the official code[4] and checkpoint[5].

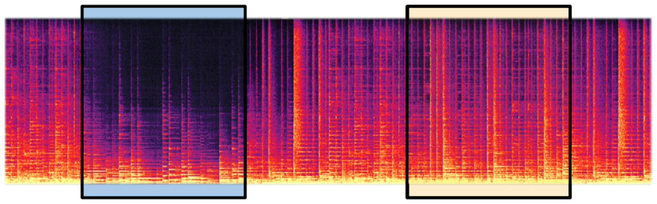
[3]https://huggingface.co/facebook/musicgen-large.
[4]https://github.com/hugofloresgarcia/vampnet.
[5]https://zenodo.org/record/8136629/.



(a) Drum track controls, where the condition is drum audio.

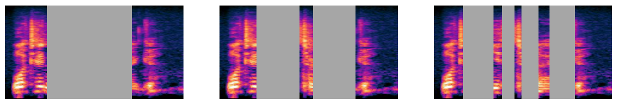


(b) Chord progression controls, where the condition is block chords audio.



(c) Arrangement and orchestration from piano cover, where the condition is piano cover audio.

Figure 4: Inpainting with various controls. The spectrogram conceptually repeats the same music segment twice, with the left half $(1, ..., T)$ providing the model with the infilling context and the controls, and the right half $(T+1, ..., 2T)$ for the model to predict. Specifically, in the left half, outside the blue rectangle is the unmasked infilling context, and inside the blue rectangle is the separated/synthesized/user-specified audio representation of the condition (e.g., (a) drum, (b) block chords, (c) piano cover). In the right half, the orange rectangle highlights the inpainted results.



(a) Mask 1. (b) Mask 2. (c) Mask 3.

Figure 5: The three types of masks used in evaluation. The number of masks is by design and the location is random.

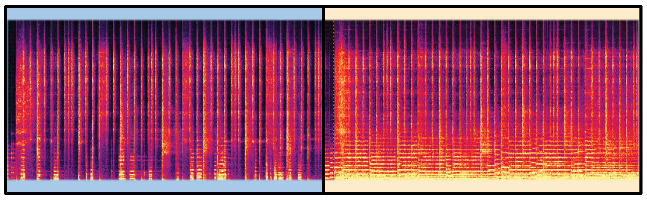


Figure 6: Music audio inpainting under a masking rate of 0.7. In this example, 70% of all frames are masked away and replaced by the drums condition. In the spectrogram above, the music audio can be identified by visible harmonics in the low-frequency region. Where those harmonics are absent there are the drum condition segments. Even under this setup, the model works well at the conditioned inpainting task.

4.5 Results

As illustrated in Table 2 and 1, our model demonstrates competitive performance compared to unconditional generation and inpainting models. Furthermore, it showcases promising steerabilty. More audio demos are available here.[6]

Long-Gap Inpainting

Table 1 presents the semantic similarity between inpainted audios and the original recordings, along with the audio qualities. In this scenario, MusicGen receives the prefix of Mask 1 (3.75s) and is tasked with generating an 11.25-second continuation. Conversely, VampNet and AIR models are conditioned with both the prefix and suffix of Mask 1. We present metrics for prefix-CLAP and prefix-FAD, evaluating the predicted (repeated) prefix and inpainted segments (11.25s), while full-CLAP and full-FAD assess both the predicted prefix, the inpainted segment, and the predicted suffix.

The comparison between prefix-CLAP and full-CLAP values in Table 0(a) indicates that as MusicGen generates longer continuations, it "strays further" from the given context. When employing our heterogeneous adapter to enable inpainting settings for MusicGen, the resulting audio exhibits promising fidelity in preserving the original semantic context during generation. Furthermore, the low FAD further indicates that audio samples inpainted by our model exhibit competitive naturalness compared to those generated by the vallina MusicGen without suffix constraints. Compared to VampNet, our model achieves superior long-gap inpainting, as evidenced by the FAD value.

Table 0(b) presents the results on the real-recording dataset with vocals, which is Out-Of-Distribution (OOD) w.r.t. the training dataset of MusicGen and our models but not w.r.t. VampNet. Comparing these results with those in Table 0(a), it is evident that our models exhibit superior generalization ability, outperforming the vanilla MusicGen.

Controllable Inpainting

As demonstrated in Table 1, our model excels in controllable inpainting across various music editing tasks. The high SDR values in Table 2 indicate our model's proficiency in internal control, effectively inpainting the masked areas while preserving the condition track. Additionally, chord, chroma, and beat metrics underscore the capability of our models to integrate controls. Furthermore, CLAP values in Table 1 highlight that applying content-based controls extracted from the original audio serve as an auxiliary mechanism for maintaining the semantics of the original audio.

Arrangement and Refinement

Figure 4 illustrates examples of the results for the score-conditioned arrangement and the track-conditioned refinement tasks, highlighting our model's capability to reconstruct missing segments of the spectrogram. This demonstrates our

[6] https://kikyo-16.github.io/AIR/#more.

m	Trainable parameters	Storage Space	Slakh2100 Test Set			RWC-POP-100		
			Drum$_{\text{SDR}}$ ↑	**CLAP**$_{\text{src}}$ ↑	**FAD**$_{\text{vgg}}$ ↓	**Drum**$_{\text{SDR}}$ ↑	**CLAP**$_{\text{src}}$ ↑	**FAD**$_{\text{vgg}}$ ↓
10	3.75M	15M	**6.475**	0.758	**1.416**	6.109	**0.652**	**1.634**
30	11.25M	45M	6.257	**0.761**	1.434	6.479	0.646	1.705
50	18.75M	75M	6.628	0.759	1.442	**6.628**	0.643	1.673

Table 3: The performance of models with different m on Mask 2.

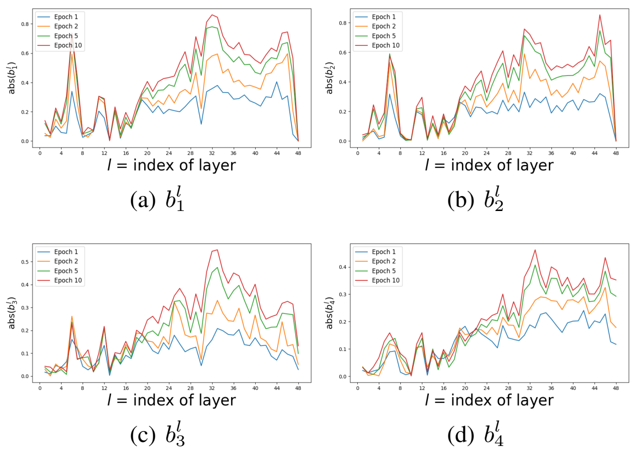


Figure 7: The variation of $|g_t^l|$ across different types of adapters during training.

approach's adaptability in performing spectrogram inpainting under various conditions.

More specifically, Figure 6 presents a case of inpainting where 70% of the spectrogram was intentionally masked prior to the inpainting process. Despite the significant loss of information, our model is able to restores the music composition, showing its robustness in handling substantial data gaps.

Low-Resource Fine-Tuning

Referring to Table 3, our heterogeneous adapters demonstrate a notable advantage in low-resource settings, with trainable parameters totaling less than 20M and a storage space requirement of less than 75M for the largest model. In comparison with the 3.3B parameters of vanilla MusicGen, our method effectively facilitates fine-tuning, converting an autoregressive music audio LLM into a steerable inpainting model.

4.6 Ablation

Impact of Different Adapter Size

A smaller m implies a reduced size of adapters, resulting in fewer trainable parameters. However, Table 3 illustrates that a small m does not significantly impair performance, highlighting the effectiveness of the proposed heterogeneous adapters. Additionally, Figure 7 displays the variation of absolute values of learnable gates g_t^l in different layers during training. It reveals that the adapter influences deep layers more than shallow layers and that the adapters $\boldsymbol{a}_{\leq 2}^l$ for the prefix area encode more information than the prediction area $\boldsymbol{a}_{>2}^l$.

(a) Slakh2100 Test Set

	Mask 1	Mask 2	Mask 3
Drum-AIR	1.423	1.442	1.467
VampNet	2.910	2.390	2.091

(b) RWC-POP-100

	Mask 1	Mask 2	Mask 3
Drum-AIR	1.606	1.673	1.759
VampNet	3.689	3.496	3.336

Table 4: The FAD values of generated audio with different masked contexts. A lower value indicates higher audio quality.

Impact of Different Masks

During the ablation study, we utilize three different masks, as illustrated in Figure 5, and report the FAD values of inpainted audios in Table 4. The context becomes more continuous, and the gap to be filled becomes shorter as we progress from Mask 1 to Mask 3. The results indicate that our model is highly robust to different mask structures and demonstrates a promising long-gap inpainting capability. Conversely, the baseline model's performance deteriorates as the masked gap lengthens.

5 Conclusions

In this paper, we propose a novel Parameter-Efficient Fine-Tuning (PEFT) method and apply it to MusicGen. It enables the model to inpaint arbitrary segments of music, while retaining the generation quality of MusicGen. Furthermore, we introduce frame-level content-based controls during fine-tuning, and demonstrate it with conditional generation tasks given the drum track, chord sequence, or piano cover. The model shows improved performance over baseline methods on several metrics. The result shows strong usability and flexibility for future music editing tools.

There are several potential improvements for the method. First, the unmasked infilling context weakens the effectiveness of the text prompt for MusicGen. A different fine-tuning setting might be required to resolve the problem. Other future works include introducing more content-based controls, as well as conditional generation based on multiple content-based controls simultaneously.

References

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

[Böck *et al.*, 2016] Sebastian Böck, Filip Korzeniowski, Jan Schlüter, Florian Krebs, and Gerhard Widmer. Madmom: A new python audio and music signal processing library. In *Proceedings of the 24th ACM international conference on Multimedia*, pages 1174–1178, 2016.

[Chen *et al.*, 2020] Ke Chen, Cheng-i Wang, Taylor Berg-Kirkpatrick, and Shlomo Dubnov. Music sketchnet: Controllable music generation via factorized representations of pitch and rhythm. In *Proceedings of the 21st Conference of the International Society for Music Information Retrieval*, pages 77–84, 2020.

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

[Défossez *et al.*, 2022] Alexandre Défossez, Jade Copet, Gabriel Synnaeve, and Yossi Adi. High fidelity neural audio compression. *arXiv preprint arXiv:2210.13438*, 2022.

[Défossez *et al.*, 2023] Alexandre Défossez, Jade Copet, Gabriel Synnaeve, and Yossi Adi. High fidelity neural audio compression. *Transactions on Machine Learning Research*, 2023. Featured Certification, Reproducibility Certification.

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

[Gao *et al.*, 2023] Peng Gao, Jiaming Han, Renrui Zhang, Ziyi Lin, Shijie Geng, Aojun Zhou, Wei Zhang, Pan Lu, Conghui He, Xiangyu Yue, et al. Llama-adapter v2: Parameter-efficient visual instruction model. *arXiv preprint arXiv:2304.15010*, 2023.

[Garcia *et al.*, 2023] Hugo F Flores Garcia, Prem Seetharaman, Rithesh Kumar, and Bryan Pardo. Vampnet: Music generation via masked acoustic token modeling. In *Ismir 2023 Hybrid Conference*, 2023.

[Goto *et al.*, 2002] Masataka Goto, Hiroki Hashiguchi, Takuichi Nishimura, and Ryuichi Oka. Rwc music database: Popular, classical and jazz music databases. In *Proceedings of the 3rd Conference of the International Society for Music Information Retrieval*, volume 2, pages 287–288, 2002.

[Hu *et al.*, 2021] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[Huang and Yang, 2020] Yu-Siang Huang and Yi-Hsuan Yang. Pop music transformer: Beat-based modeling and generation of expressive pop piano compositions. In *Proceedings of the 28th ACM international conference on multimedia*, pages 1180–1188, 2020.

[Huang *et al.*, 2018] Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Noam Shazeer, Ian Simon, Curtis Hawthorne, Andrew M Dai, Matthew D Hoffman, Monica Dinculescu, and Douglas Eck. Music transformer. *arXiv preprint arXiv:1809.04281*, 2018.

[Ippolito *et al.*, 2018] Daphne Ippolito, Anna Huang, Curtis Hawthorne, and Douglas Eck. Infilling piano performances. In *NIPS Workshop on Machine Learning for Creativity and Design*, volume 2, page 5, 2018.

[Jiang *et al.*, 2019] Junyan Jiang, Ke Chen, Wei Li, and Gus Xia. Large-vocabulary chord transcription via chord structure decomposition. In *Proceedings of the 20th Conference of the International Society for Music Information Retrieval*, pages 644–651, 2019.

[Li and Liang, 2021] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

[Liu *et al.*, 2022] Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68, 2022.

[Manilow *et al.*, 2019] Ethan Manilow, Gordon Wichern, Prem Seetharaman, and Jonathan Le 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)*, pages 45–49. IEEE, 2019.

[Marafioti *et al.*, 2019] Andrés Marafioti, Nathanaël Perraudin, Nicki Holighaus, and Piotr Majdak. A context encoder for audio inpainting. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 27(12):2362–2372, 2019.

[Marafioti *et al.*, 2020] Andres Marafioti, Piotr Majdak, Nicki Holighaus, and Nathanaël Perraudin. Gacela: A generative adversarial context encoder for long audio inpainting of music. *IEEE Journal of Selected Topics in Signal Processing*, 15(1):120–131, 2020.

[Min *et al.*, 2023] Lejun Min, Junyan Jiang, Gus Xia, and Jingwei Zhao. Polyffusion: A diffusion model for polyphonic score generation with internal and external controls. In *Proceedings of the 24th Conference of the International Society for Music Information Retrieval*, pages 231–238, 2023.

[Mitsufuji *et al.*, 2022] Yuki Mitsufuji, Giorgio Fabbro, Stefan Uhlich, Fabian-Robert Stöter, Alexandre Défossez, Minseok Kim, Woosung Choi, Chin-Yun Yu, and Kin-Wai Cheuk. Music demixing challenge 2021. *Frontiers in Signal Processing*, 1:808395, 2022.

[Perraudin *et al.*, 2018] Nathanael Perraudin, Nicki Holighaus, Piotr Majdak, and Peter Balazs. Inpainting of long audio segments with similarity graphs. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(6):1083–1094, 2018.

[Raffel *et al.*, 2020] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

[Schneider *et al.*, 2023] Flavio Schneider, Ojasv Kamal, Zhijing Jin, and Bernhard Schölkopf. Mo\^ usai: Text-to-music generation with long-context latent diffusion. *arXiv preprint arXiv:2301.11757*, 2023.

[Van Den Oord *et al.*, 2017] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

[Wang and Xia, 2021] Ziyu Wang and Gus Xia. Musebert: Pre-training music representation for music understanding and controllable generation. In *ISMIR*, pages 722–729, 2021.

[Wu *et al.*, 2023] 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 *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.

[Zhang *et al.*, 2023] Renrui Zhang, Jiaming Han, Aojun Zhou, Xiangfei Hu, Shilin Yan, Pan Lu, Hongsheng Li, Peng Gao, and Yu Qiao. Llama-adapter: Efficient fine-tuning of language models with zero-init attention. *arXiv preprint arXiv:2303.16199*, 2023.

[Zhou *et al.*, 2019] Hang Zhou, Ziwei Liu, Xudong Xu, Ping Luo, and Xiaogang Wang. Vision-infused deep audio inpainting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 283–292, 2019.