A Datasets and Data Processing

Table 8 presents some details about the datasets we used and generated. All audio data have a sampling rate of 16KHz. For AudioSet, AudioCaps, and AudioSet96, we pad or truncate all audio to 10s. For FSD50K, we pad audio shorter than 5s to 5s, and truncate or pad audio longer than 5s to 10s. All generated audio data have a length of 10 seconds. We convert 10s of audio into mel-spectrograms with a size of 80×624, using a hop size of 256, a window size of 1024, and mel-bins of size 80, covering frequencies from 0 to 8000. The autoencoder compresses the mel-spectrograms to a latent representation of size $4 \times 10 \times 78$.

Table 8: Details about audio-text datasets we use and our generated audio editing datasets

Dataset	Hours	Number	Text
AudioSet	5800	2M	label
AudioSet96	414	149K	label
AudioCaps	122	44K	caption
FSD50K	108	51K	label
FSD50K-S	31	22K	label
FSD50K-L	53	19K	label
ESC50	3	2K	label
Task	**Datasets**	**Number**	**Text**
Generation	AudioCaps, AudioSet96, FSD50K, ESC50	243K	label or caption
Adding	AudioCaps, AudioSet96, FSD50K-S,ESC50	71K	Instruction
Dropping	AudioCaps, AudioSet96, FSD50K-S,ESC50	71K	Instruction
Replacement	AudioSet96, FSD50K-S, ESC50	50K	Instruction
Inpainting	AudioSet96, AudioCaps	193K	Instruction
Super-resolution	AudioSet96, AudioCaps	193K	Instruction

B Model Details

Table 9 shows more details about our audio editing and audio generative models. We train our autoencoder model with a batch size of 32 (8 per device) on 8 NVIDIA V100 GPUs for a total of 50000 steps with a learning rate of $7.5e-5$. For both audio editing and U-Net audio generative diffusion, we train with a batch size of 8 on 8 NVIDIA V100 GPUs for a total of 500000 steps with a learning rate of $5e-5$. Both the autoencoder and diffusion models use AdamW[29] as the optimizer with $(\beta_1, \beta_2) = (0.9, 0.999)$ and weight decay of $1e-2$. We follow the official repository to train the HiFi-GAN vocoder.

C Classifier-free Guidance

[16] proposed using classifier-free guidance to trade off diversity and sample quality. The classifier-free guidance strategy has been widely used in conditional diffusion models. Our model has two additional conditions, c_{text} and $\boldsymbol{z}_{in}$. However, during training, we only consider helping the model learn the marginal distribution of $p(\boldsymbol{z}_t|\boldsymbol{z}_{in})$ (without explicitly learning $p(\boldsymbol{z}_t|c_{text})$). Therefore, during training, we mask the text with a certain probability (replacing it with an empty text $\emptyset$) to learn the no-text condition score $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{z}_t, t, \boldsymbol{z}_{in})$. Then, according to Bayes' formula $p(\boldsymbol{z}_t|c_{text}) \propto \frac{p(\boldsymbol{z}_t|c_{text},\boldsymbol{z}_{in})}{p(\boldsymbol{z}_{in}|\boldsymbol{z}_t,c_{text})}$, we can derive the score relationship in Equation 1, which corresponds to the classifier-free guidance Equation 2. Here, the parameter $s \geq 1$ is the guidance coefficient used to balance the diversity and quality of the samples.

$$\nabla_{\boldsymbol{z}_t} \log p(\boldsymbol{z}_{in}|\boldsymbol{z}_t, c_{text}) = \nabla_{\boldsymbol{z}_t} \log p(\boldsymbol{z}_t|c_{text}, \boldsymbol{z}_{in}) - \nabla_{\boldsymbol{z}_t} \log p(\boldsymbol{z}_t|\boldsymbol{z}_{in}) \tag{1}$$

$$\tilde{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_t, t, \boldsymbol{z}_{in}, c_{text}) = \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{z}_t, t, \boldsymbol{z}_{in}, \emptyset) + s \cdot (\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{z}_t, t, \boldsymbol{z}_{in}, c_{text}) - \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{z}_t, t, \boldsymbol{z}_{in}, \emptyset)) \tag{2}$$

Table 9: Details about our audio editing and audio generative models

Model	Configuration	
Autoencoder	Number of Parameters	83M
	In/Out Channels	1
	Latent Channels	4
	Number of Down/Up Blocks	4
	Block Out Channels	(128, 256, 512, 512)
	Activate Function	SiLU
T5 Text Encoder	Number of Parameters	109M
	Output Channels	768
	Max Length	300
Editing Diffusion U-Net	Number of Parameters	859M
	In Channels	8
	Out Channels	4
	Number of Down/Up Blocks	4
	Block Out Channels	(320, 640, 1280, 1280)
	Attention Heads	8
	Cross Attention Dimension	768
	Activate Function	SiLU
Generative Diffusion U-Net	Number of Parameters	859M
	In Channels	4
	Out Channels	4
	Number of Down/Up Blocks	4
	Block Out Channels	(320, 640, 1280, 1280)
	Attention Heads	8
	Cross Attention Dimension	768
	Activate Function	SiLU
HiFi-GAN	Sampling Rate	16000
	Number of Mels	80
	Hop Size	256
	Window Size	1024

D Human Evaluation

For each audio editing task, our human evaluation set comprises ten samples randomly selected from the test set. Ten raters score each sample according to two metrics, Quality and Relevance, using a scale of 1 to 100.

E Extending Instructions with ChatGPT

Some examples of instruction templates designed by ourselves:

"Add {} in the beginning"

"Add {} at the beginning"

"Add {} in the end"

"Add {} in the middle"

"Add {} in the background"

"Drop {}"

"Remove {}"

"Replace {} to {}"

"Replace {} with {}"

"Inpaint"

"Inpainting"

"Inpaint {}"

"Inpaint: {}"

"Increase resolution"

"Increase resolution: {}"

"Perform super-resolution"

"Perform super-resolution: {}"

We use ChatGPT to extend editing templates, specifically, we submit the editing instruction templates we designed to ChatGPT and let it generate more instruction templates with the same semantic information. Below are some examples.

"Mix {} into the background"

"Blend {} with existing audio"

"Incorporate {} as a new element at the end of the audio"

"Place {} in the foreground"

"Erase {} from the track"

"Subtract {} from the audio"

"Take out {} from the foreground"

"Exchange {} for {} in the mix"

"Use {} to replace {} in the audio"

"Interchange {} and {} in the track"

"Replace missing audio with synthesized sound"

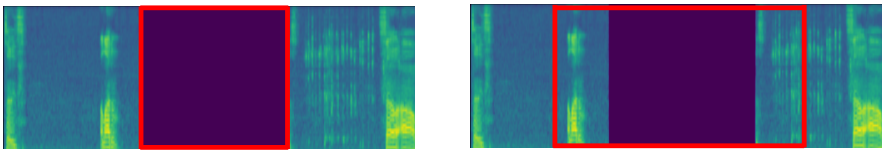
"Fill in the gaps in track {}"

"Upscale audio to higher resolution"

"Apply super-resolution to the audio to improve clarity"

F Baseline Methods

For the inpainting task, only a part of the input audio (the part that is masked) needs to be edited, we call the part that does not need to be edited the "observable" part, and we call the masked part the "unobservable" part. In each step of the denoising process, we can replace the "observable" part with the ground truth in the latent space. The difference between SDEdit-Rough and SDEdit-Precise is that in SDEdit-Rough, the "unobservable" part is a rough region, while in SDEdit-Precise, the "unobservable" part is a precise region. Figure 4 gives an example.



The "unobservable" part in SDEdit-Precise.

The "unobservable" part in SDEdit-Rough.

Figure 4: The difference between SDEdit-Rough and SDEdit-Precise.

G Case Study

We show case studies. Figure 5 shows a case for the adding task. The caption of the input audio is *"The sound of machine gun"*, and the editing target is adding a bell ringing in the beginning. AUDIT performs audio editing accurately in the correct region without modifying audio segments that do not need to be edited. Figure 6 shows a case for the dropping task. The caption of the input audio is *"A bird whistles continuously, while a duck quacking in water in the background"*, and the editing target is dropping the sound of the duck quacking. Our method successfully removes the background sound and preserves the sound of the bird whistling, but the sound of the bird whistling in the SDEdit editing result is incorrectly modified. It shows that our method can better ensure that the audio segments that do not need to be edited are not modified. Figure 7 shows a case for the inpainting task. The caption of the input audio is *"A person is typing computer"*. While both AUDIT and the baseline method generate semantically correct results, the result generated by AUDIT is more natural and contextual.

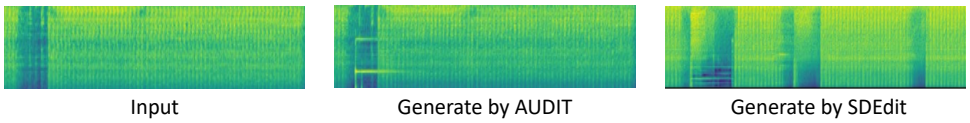


Figure 5: A case for the adding task. The caption of the input audio is *"The sound of machine gun"*, and the editing target is adding a bell ringing in the beginning.

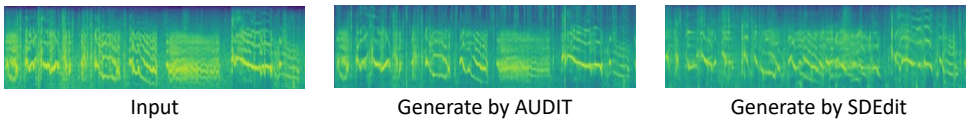


Figure 6: A case for the dropping task. The caption of the input audio is *"A bird whistles continuously, while a duck quacking in water in the background"*, and the editing target is dropping the sound of the duck quacking.

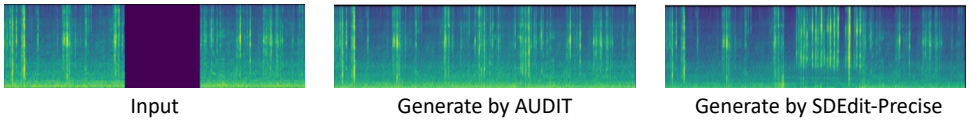


Figure 7: A case for the inpainting task. the caption of the input audio is *"A person is typing computer"*.

H Limitations and Broader Impacts

Our work still has some limitations. For example, the sampling efficiency is low since our model is a diffusion-based model. In the future, we will try to improve the generation efficiency of our model using efficient strategies like consistency models [47]. In addition, we will also explore the use of more data and more diverse editing instructions to achieve more kinds of editing tasks. AUDIT can edit existing audio based on natural language instructions, which may be used inappropriately, such as synthesizing fake audio for fraud. Therefore, we urge everyone not to abuse this technology and develop synthetic audio detection tools.