PERSONALIZED TIMBRE GENERATION

Fang Duo Tsai
National Taiwan University
r12942150@ntu.edu.tw

Hung-Jui Chen
National Taiwan University
r12942144@ntu.edu.tw

ABSTRACT

In this project, our goal is to finetune the text-to-audio model AudioLDM2 to generate personalized timbre while retaining the concepts it learned before. We mainly use the Dreambooth method, utilizing prior preservation loss for the concept we wish to retain. We also use the AudioMAE as an audio encoder to facilitate the personalization process. To learn personalized timbre, we chose Chinese instruments as our targets since current text-to-audio models are not familiar with them.

1. INTRODUCTION

There has been impressive progress in personalization research in the computer vision domain, generating images with personalized concepts such as a specific dog, a person's face, etc. The advantage of the personalization method is that it only requires a small amount of data and can learn something hard to describe via text. While there have been several works focusing on controlling chords, melody, dynamics, and rhythm of generated music, our work specializes our model to generate specific timbre. Our contributions include:

- 1. We use teacher forcing to fine-tune GPT2 with the ground truth LOA from AudioMAE before using Dreambooth, significantly accelerating the personalization process.
- 2. We use an extension of Dreambooth to prevent the model from ignoring other prompts and focusing solely on the fine-tuned instrument.

2. RELATED WORK

2.1 AudioLDM2

AudioLDM2 is a text-to-audio diffusion model inspired by stable diffusion, with a few differences. While stable diffusion leverages the shared embedding space of image text, AudioLDM2 uses the latent space of the AudioMAE encoder. This latent space is called the Language of Audio (LOA). Training AudioLDM2 requires both ground truth

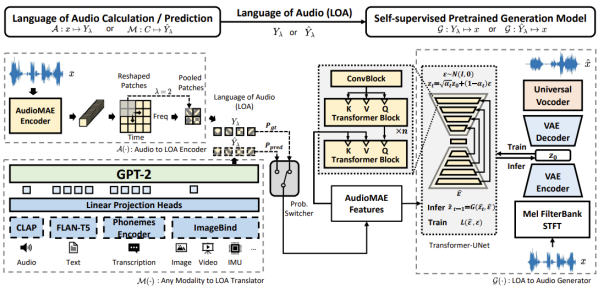

Figure 1. AudioLDM2 structure

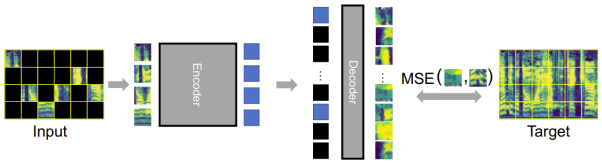

Figure 2. AudioMAE structure

LOA and predicted LOA when audio is not available. The predicted LOA is generated by a text encoder and a language model, GPT2. The language model is fine-tuned using teacher forcing with the ground truth LOA generated by the AudioMAE. After fine-tuning GPT2, joint fine-tuning with Unet is performed, with a probability of 0.25 using ground truth LOA and 0.75 using the predicted LOA.

2.2 AudioMAE

AudioMAE (Audio Masked Encoder) is similar to the image-based Masked Autoencoders, both used for self-supervised representation learning. AudioMAE takes audio spectrograms, while MAE takes images. AudioMAE first encodes audio spectrogram patches with a high masking ratio, feeding only the non-masked tokens through encoder layers. The decoder then re-orders and decodes the encoded context padded with mask tokens to reconstruct the input spectrogram. The intermediate representation between the encoder and decoder is very disentangled, as disclosed in the AudioLDM2 paper, involving fine-grained timbre information of the audio, making it suitable for our work as a condition embedding.

2.3 Dreambooth

Large text-to-image models can synthesize high-quality images from a given text prompt, but these models fall back on mimicking the appearance of subjects in a given reference set and generating the same subject in another background. Dreambooth takes a few images as input and uses a unique, rarely-seen string as an identifier for the subject in the picture. During fine-tuning, the model learns that the identifier refers to the specific subject. However, to avoid the model mistakenly learning that all subjects refer to the specific subject, we use prior preservation loss to ensure the model still knows the generalized subject. For example, to learn a specific dog's appearance, we use the prompt "a photo of a sks dog" to teach the model that "sks" refers to the target concept, while prior preservation calculates the loss between the output of "a photo of a dog" before and after fine-tuning to ensure the model doesn't generalize "sks dog" to mean all kinds of dogs. In the audio domain, we use a method similar to Dreambooth, with slight differences explained in Section 3.

3. METHODS

3.1 Variation of Dreambooth

We use "sks flute" to represent the Chinese flute during the fine-tuning process. We discovered that after fine-tuning the model with "a recording of a sks flute," the model could not generate "a recording of a sks flute played with piano" while still generating "a recording of a piano." This means the fine-tuning process misleads the model to perform only the fine-tuned instrument "sks flute" and ignore other instruments. Our solution is to use another prompt "a recording of a sks flute with piano," where the corresponding audio is made by directly mixing the training concept audios of the sks flute with piano audios generated by the original model. This may slightly affect the compatibility of the two instruments in the same generated audio, but it prevents the model from ignoring the second instrument in the prompt. We use the following prompts for personalization:

class prompt: "A recording of a sks flute solo."

validation prompt: "A recording of a piano with sks flute."

3.2 AudioMAE Teacher Forcing

Since AudioLDM2 is trained to be conditioned with LOA, we simply connect the AudioMAE back with the same pretrain weight. In the pre-trained AudioLDM2 pipeline, GPT2 outputs predicted LOA with a shape of (1,8,768), a small latent space that can be fine-tuned quickly. We use teacher forcing to optimize the MSE loss between the ground truth LOA and the predicted LOA output by GPT2 in the AudioLDM2 pipeline. After only 50 epochs, which takes about 1 minute, GPT2 starts from a better initial state for personalization. There are a few drawbacks, such as when the pre-trained AudioMAE hasn't seen the timbre of the target concept audio before, it outputs an embedding not the same as the target but close to the target. We will show quantitative results in Section 5.

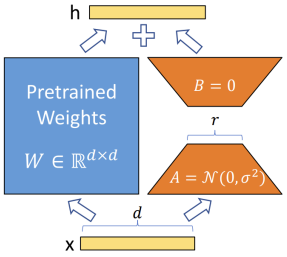

Figure 3. Lora adaptor structure

3.3 Lora Fine-Tuning

Lora is often used to fine-tune large language models since it significantly reduces the memory usage of the GPU. LoRA proposes to freeze pre-trained model weights and inject trainable layers (rank-decomposition matrices) in each transformer block, reducing the number of trainable parameters. In the paper "Encoder-based Domain Tuning for Fast Personalization of Text-to-Image Models," they measure the importance score of the Unet in the diffusion model and discover that the cross-attention layers undergo drastic changes compared to other layers. Based on their findings, we decided to use Lora adaptors connected to the cross-attention layers and freeze all other parts to fine-tune the model while restricting the fine-tuning process to prevent forgetting or ignoring concepts afterward.

4. EVALUATION

We evaluate the personalized timbre with two different metrics. The first one is Frechet Distance (FAD), calculated between the embeddings of a pre-trained VGGish audio classifier of the training set and the generated music clips. The second evaluation metric is the confidence score of the classifier we trained.

4.1 Classifier

To assess the performance of the music generator, we address two crucial questions:

1. How closely does the generated music emulate the desired instruments?

2. How effectively does the generated music avoid resembling other undesired instruments?

We employ a classification model to meet these specific requirements.

4.1.1 Dataset

For the Chinese music component, we acquire full-song audio from Chinese music albums, encompassing two distinct types of Chinese instruments: the Chinese flute and the Chinese lute. Unfortunately, most songs in these albums are not solo performances, meaning the dataset is

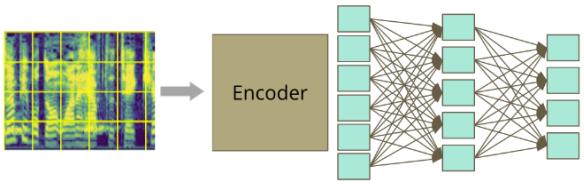

Figure 4. Classifier Model Structure

not "clean" as it includes music that does not solely feature the desired instruments. To remedy this, we use Basic Pitch[1], a lightweight audio-to-MIDI transformer. After transforming songs into MIDI files, we determine whether a song is solo or not by calculating the ratio of the count of all pitches to the total duration of pitches. This ratio reflects the proportion of time with multiple pitches in a song. If the ratio is sufficiently low, we consider the song solo, qualifying it as "clean" and suitable for inclusion in our dataset. After purifying the dataset, the next steps involve segmenting the songs and splitting the dataset at the song level, creating distinct training, validation, and testing sets.

In addition to the Chinese music dataset, we use the NSynth Dataset to train the classification model for non-Chinese music. This dataset provides audio samples for 10 different instruments, including bass, brass, flute, guitar, keyboard, mallet, organ, reed, string, and vocal. The dataset's extensive volume and well-defined partitions help overcome challenges such as overfitting, ensuring a high level of accuracy.

4.1.2 Model Structure

In constructing the classifier model, we leverage a pre-trained AudioMAE[2] as a feature extractor. The extracted features serve as the basis for classification. We connect multiple fully connected layers, integrating dropout layers and ReLU functions between them. These layers work together to calculate the score for each type of instrument the audio may belong to. The model structure is illustrated in Figure 4. While theoretically, adding more fully connected layers may enhance classifier accuracy, post-training analysis reveals that the network with two fully connected layers following the AudioMAE encoder achieves the highest accuracy. Consequently, we select this configuration as our final model.

4.1.3 Additional Training Parameters

To evaluate the retention of prior instrument generation parameters by the generator, we task it with creating audios featuring both a non-Chinese and a Chinese musical instrument. Consequently, the classifier constructed is designed as a multi-class multi-label classifier. The chosen loss function is Binary Cross Entropy, with a sigmoid function serving as the activation function before it.

Other training hyperparameters include a learning rate set to 0.001, a batch size of 64, and 120 training epochs. The parameters are shown in Table 1.

Training Parameter	Value
Classifier Type	Multi-class Multi-label
Loss Function	Binary Cross Entropy
Activation Function	Sigmoid
Learning Rate	0.001
Batch Size	64
Training Epochs	120

Table 1. Training Parameters for Audio Generation

4.1.4 Classification Results

Following 120 epochs of training, the model achieves a testing accuracy of 0.7350, as illustrated in Figure 5 by the accompanying confusion matrix. Several noteworthy points arise from these results. Firstly, it's essential to acknowledge that "synth" is an instrument exclusively present in the training dataset of NSynth. Therefore, an expected outcome is that the testing results for "synth" are all zeros. Secondly, despite an overall accuracy of 0.7350, the model exhibits exceptional proficiency in classifying the Chinese flute, the primary focus of our analysis. Lastly, the model appears to struggle in effectively classifying the Chinese lute. Notably, close to half of the Chinese lute instances are misclassified as Chinese flute. This discrepancy may be attributed to the training set containing only about two-thirds of the data available for the Chinese flute, leading to misclassification of Chinese lute instances as Chinese flute.

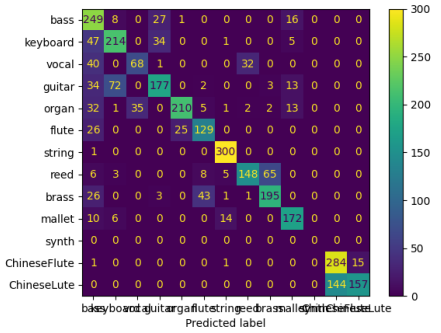

Figure 5. Confusion Matrix

5. RESULTS

We will show the results for different settings.

5.1 Teacher Forcing

Model	300 steps	25 steps	0 steps
large-gpt2	7.17	8.67	20.01
large-gpt2-teacher	6.8	8.09	13.64

Table 2. FAD results for version 2

We evaluate two sets of training audio. Version 1 has higher tones, which sound a bit like the original flute, while

Model	25 steps	0 steps
large-gpt2	23.04	45.84
large-gpt2-teacher	5.55	3.56

Table 3. FAD results for version 1

version 2 includes lower tones with a special timbre. As we can see, version 1 (Table 3) shows a significant change after teacher forcing since the timbre is more generalized. In contrast, version 2 (Table 2) shows a smaller difference after teacher forcing as the AudioMAE is not familiar with this version.

5.2 Prior Preservation for Accompaniment

The purpose of the prior preservation loss is to prevent the model from ignoring an instrument other than the training concept. We leverage the confidence output by the softmax layer of the classifier to see if, when given the prompt "A recording of a piano with sks flute," the piano will get a higher confidence score after using prior preservation loss for accompaniment. We sort the confidence scores and record the ranking of the piano. Without using prior preservation loss, we can barely hear the piano. When using prior preservation loss, we can hear a clear piano accompaniment, thus the confidence score is only smaller than the class Chinese flute.

Model	Ranking
large-gpt2	7
large-gpt2-prior	2
large-gpt2-Lora	4

Table 4. Ranking of the piano class

The result shows that the prior preservation loss works, while Lora doesn't perform as well but is still better than direct fine-tuning. Furthermore, we need to ensure that although restricting the model by prior preservation, the model can still learn the target concept well. See the comparison in Table 5. We can see that we didn't sacrifice the quality of the target concept while preserving the ability to perform the duet.

Model	FAD
large-gpt2	7.17
large-gpt2-prior	7.39

Table 5. FAD results

5.3 Best Result

Our best result combines teacher forcing and the variation of prior preservation loss. The first one, "large-gpt2-teacher-mixclass-prior," refers to first fine-tuning GPT2 and then training with the mixed audio of piano and the Chinese flute. The second one retains the settings from the first one, but the prior-preservation audio is simply generated by using the prompt "a recording of a piano with flute." Since the first setting uses mixed audio, we are concerned it might affect the synchronization of the duet, while the second setting might not produce the exact same timbre of the target concept when played with piano. Despite these concerns, the results show both settings perform well.

Model	FAD
large-gpt2-teacher-mixclass-prior	5.71
large-gpt2-teacher-chineseclass-prior	7.26

Table 6. FAD results

Model	Ranking
large-gpt2-teacher-mixclass-prior	2
large-gpt2-teacher-chineseclass-prior	2

Table 7. Ranking of the piano class

6. CONCLUSION

In this work, we successfully accelerated the personalization process, prevented the model from ignoring the second instrument after fine-tuning, and generated better results despite the restrictions mentioned above. In the future, we will research other adapters for faster and more precise fine-tuning.