Generating Music by Fine-Tuning the MusicLM

September 26, 2023

Team Members: Dihong Huang, Akaash Dash, Saikanam Siam, Peter Zhen

1 Project Summary

Our project is to fine tune a large music transformer called MusicLM that applies to the domain of violins. The input is prompt text to describe the violin pitch and melody. The output is a short audio clip with a few seconds based on the prompt texts. The project difficulty is moderate, and it doesn't always require a large amount of training data. Low-rank fine-tuning decreases the data to be taken and increases the training speed. What's more, the project doesn't require manually labeling a large data set.

Our idea is to add a task-specific final output layer on top of pre-trained generative model for efficient parameter fine-tuning and kept the rest of original pre-trained model's layers frozen so that MusicLM improves its accuracy in genres that our feeding training music data belongs to. We also need a decoder to map the hidden representations to the generated audio output. Based on the concept of transfer learning, we can add a conditioning mechanism to the output layer since our tasks involves on conditional music generation.

2 Proposed Method

Text generation often benefits from a large language model given the past to learn the next word in a sequence. Similarly, a music model effectively generates the music notes from start to finish and predicts the next note in a sequence given the past music data. If treating text as input and music composition as output, we can leverage a model to perform hierarchical sequence-to-sequence modeling tasks such as MusicLM with a prompt text. According to the music style described in the prompt text description, MusicLM can transform whistled and hummed melodies and "customize" the music it generates.

MusicLM is built on top of a model called AudioLM which relies on hierarchical tokenization and general scheme to address the trade-off between coherence and high-quality synthesis. AudioLM utilizes two types of tokens, including semantic tokens and acoustic tokens. MusicLM is modified with three extra features: 1) The generation process is subject to descriptive language 2) conditioning can be extended to signals such as melody 3) The researchers model different types of long music sequences other than piano music. Music-text joint embedding model called MuLan plays an important role since it is trained on pairs of music clips and corresponding text annotation texts for the music.

The data preparation phase will focus on assembling a dataset consisting of violin audio clips paired with corresponding prompt texts that describe the musical content they generate from a variety of sources. We will clean and preprocess the audio clips, which may include tasks such as audio normalization, sample rate adjustment, and noise reduction. Simultaneously, we tokenize the textual prompts to make them suitable for model input.

Then we apply Low-rank Optimization for Approximate (LORA) Learning to the MusicLM model to reduce the parameter space.

We have a few Loss functions in mind:

Mean Squared Error: The difference between the spectograms of the generated and real audio clips could be compared to get the MSE between them. This would allow the model to quantifically judge the nuances between different violin sounds.
Frechet Audio Distance: This metric is similar to Frechet Image Distance [6]. This tries to compare the distributions between two different music clips, which we can use to evaluate the generative model for audio.
MuLan Cycle Consistency: this metric measures the similarity between pairs of music and text. We first derive embeddings from the textual descriptions and the generated music. Then, we determine the average cosine similarity between these embeddings.

3 Related Work

Below are the seven relevant papers we found to support our project. Transformer is the foundation model that most LLMs adapt, including the recent MusicLM and the Music Transformer. Mulan is the joint music-text model used by MusicLM to create embeddings. Frechet Audio Distance is one of our evaluation metrics for audio generation. LoRA is our technique to fine tune the model efficiently.

1. Andrea Agostinelli, Timo I. Denk, Zalan Borsos, Jesse Engel, Mauro Verzetti, Antoine Caillon, Qingqing Huang, Aren Jansen, Adam Roberts, Marco Tagliasacchi, Matt Sharifi, Neil Zeghidour, and Christian Frank. (2023). MusicLM: Generating Music From Text.

2. Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, & Weizhu Chen. (2021). LoRA: Low-Rank Adaptation of Large Language Models.

3. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, & Illia Polosukhin. (2023). Attention Is All You Need.

4. Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Noam Shazeer, Ian Simon, Curtis Hawthorne, Andrew M. Dai, Matthew D. Hoffman, Monica Dinculescu, & Douglas Eck. (2018). Music Transformer.

5. Flavio Schneider, Zhijing Jin, & Bernhard Scholkopf. (2023). Mousai: Text-to-Music Generation with Long-Context Latent Diffusion.

6. Kevin Kilgour, Mauricio Zuluaga, Dominik Roblek, & Matthew Sharifi. (2019). Frechet Audio Distance: A Metric for Evaluating Music Enhancement Algorithms.

7. Huang, Q., Jansen, A., Lee, J., Ganti, R., Li, J. Y., and Ellis, D. P. W. (2022). Mulan: A joint embedding of music audio and natural language. In International Society for Music Information Retrieval Conference (ISMIR).

4 Datasets / Environments

The following websites contain music data sets from which we can filter by genre/domain and extract useful fields. Additional pre-processing (noise reduction, converting to MIDI, etc.) may be necessary:
http://millionsongdataset.com
https://research.google.com/youtube8m/explore.html
https://www.midiworld.com/

Previous work has also been done created genre-tagged and detail oriented data sets:
https://arxiv.org/pdf/1612.01840.pdf
https://doi.org/10.7916/D8NZ8J07

Additionally, the data sets to test MusicLM can be applied here:
https://www.kaggle.com/datasets/googleai/musiccaps/data

Altogether, we can combine data sets and annotated descriptions to create a data set big enough for batched training. Normalization will be necessary, but much processing tools and documentation already exists as Python libraries (music21, audiolazy, etc.), making this task reasonable.