GUITAR AUDIO GENERATION VIA MUSICGEN: FINE-TUNING APPROACH FOR HIGH-QUALITY GUITAR SYNTHESIS

Jih-Ming Pai r11725041@ntu.edu.tw
Ting-An Yin b10901086@ntu.edu.tw
I-Pei Lee r13921013@ntu.edu.tw

ABSTRACT

This study explores fine-tuning the MusicGen model to generate high-quality guitar music. Using the GuitarSet dataset, we applied LoRA technology and data augmentation strategies to optimize model training. Two training strategies were tested: incorporating special tokens and enriching text prompts. Our results demonstrate that fine-tuning significantly improves the generated audio, with descriptive text prompts playing a crucial role in guiding the model to produce better guitar music. Although the fine-tuned model captures distinct guitar timbre and style, challenges remain in generating realistic melodies. This research offers valuable insights into AI-driven guitar music generation and suggests that refining training strategies and expanding datasets could further enhance model performance, providing a foundation for future advancements in this area.

1. INTRODUCTION

Text-to-audio generation models represent a groundbreaking intersection of artificial intelligence and audio synthesis, enabling the creation of soundscapes, music, and even instrumental performances from textual descriptions, which offers new tools for musicians, sound designers, and educators. Among these innovations, MusicGen, a state-of-the-art text-to-audio model developed by Meta, stands out for its ability to generate coherent and high-quality audio outputs. Its versatility make it an ideal candidate for exploring specialized audio generation tasks.

Despite significant advancements in generative audio, challenges persist when it comes to specific instruments like the guitar. Capturing the nuanced timbre, intricate dynamics, and expressive techniques unique to guitar performances is a complex task. Existing models often struggle to reproduce these subtleties, limiting their utility for applications requiring realistic and detailed guitar sounds. Addressing these challenges is crucial for bridging the gap between general audio generation and instrument-specific expertise.

Thus, this project focuses on fine-tuning MusicGen to specialize in generating high-quality guitar sounds and melodies. By leveraging the GuitarSet dataset—-a rich repository of annotated guitar recordings that includes diverse playing styles and techniques—this research aims to enhance the model's ability to capture the distinct characteristics of guitar music. The ultimate objective is to produce a system capable of generating realistic and expressive guitar audio from textual prompts.

The structure of this paper is as follows: Section 1 introduces the background, objectives of the project. Section 2 reviews related work in text-to-audio generation and fine-tuning approaches. Section 3 details the methodology, including the rationale for selecting MusicGen, LoRA implementation, and data augmentation strategies. Section 4 outlines the experimental setup, covering dataset preparation, evaluation metrics, and implementation details. Next, Section 5 presents the results and findings, including performance analysis, insights from text prompt experiments, and hyperparameter studies. Finally, Section 6 concludes with a summary of the project's contributions, discusses its limitations, and proposes directions for future research.

2. RELATED WORK

In this section, we review existing methodologies in guitar audio generation and examine state-of-the-art text-to-music models, highlighting their relevance to our study on fine-tuning MusicGen [1] for guitar audio generation.

2.1 Guitar Audio Generation Methodology

Deep learning has significantly advanced audio generation, with various models developed to synthesize realistic guitar sounds. A comprehensive survey by [2] discusses typical techniques in deep learning audio generation, including audio representations, model architectures, and training strategies.

In the realm of guitar-specific audio generation, several approaches have been explored. For instance, GuitarLSTM [3], real-time guitar amplifier emulation using deep learning, investigates the use of deep neural networks for modeling audio distortion circuits, such as guitar amplifiers and distortion pedals.

Additionally, research on expressive acoustic guitar sound synthesis [4] with an instrument-specific input representation, termed guitarroll, proposes a model tailored to the unique characteristics of guitar sounds, enhancing the expressiveness and realism of synthesized guitar audio.

2.2 State-of-the-Art Text-to-Music Models

The field of text-to-music generation has seen significant advancements, with models capable of producing high-fidelity music from textual descriptions. OpenAI's Jukebox [5] is a notable model that generates music, including rudimentary singing, as raw audio in a variety of genres and artist styles. It accepts a genre, artist, and a snippet of lyrics and outputs song samples. MuseNet [6], also developed by OpenAI, is a deep neural network capable of generating 4-minute musical compositions with up to 10 different instruments. It can combine a wide range of styles, from country to Mozart to the Beatles, and was trained to predict subsequent musical notes in MIDI files, learning patterns of harmony, rhythm, and style. RAVE [7], which stands for Real-time Audio Variational Auto Encoder, is a model designed for real-time audio generation, focusing on high-fidelity and controllable audio synthesis. It utilizes variational auto-encoders to model audio distributions, enabling the generation of diverse audio samples.

In recent studies, MusicLM [8], developed by Facebook Research, is a text-to-music model that generates high-quality music samples conditioned on text descriptions or audio prompts. Unlike existing methods, MusicLM does not require a self-supervised semantic representation and generates all codebooks in one pass. MusicGen [1], developed by Facebook, is a powerful text-to-music model capable of generating high-quality music samples conditioned on text descriptions or audio prompts. MusicGen is a single-stage, autoregressive Transformer model that generates all codebooks in one pass, offering high control over the generated content without requiring a self-supervised semantic representation.

These advancements in text-to-music models provide a solid foundation for our study, where we focus on fine-tuning MusicGen to enhance its ability to generate realistic guitar audio from textual prompts.

3. METHODOLOGY

3.1 Model Selection: Why MusicGen?

In this study, we selected **MusicGen** [1] as the foundation model for fine-tuning guitar music generation. Developed by Meta, MusicGen is a state-of-the-art (SOTA) model that aligns with our objectives due to the following key reasons:

1. **Pre-trained Model Strength:** MusicGen is pre-trained on a large dataset of diverse music tracks, enabling a robust understanding of musical patterns, styles, and instrumentation, including guitar-specific features. This makes it an ideal starting point for fine-tuning tasks.

2. **Versatile Conditioning Mechanisms:**
   The model supports conditioning on textual descriptions and melodies, allowing us to guide the generation process effectively. This flexibility is particularly valuable for producing coherent guitar music tailored to specific prompts or styles.

3. **Open-Source Accessibility:**
   MusicGen is open-source, providing us the freedom to customize and fine-tune it with domain-specific datasets (e.g., guitar music) and task-specific constraints. Other models, such as OpenAI's Jukebox, often lack this level of accessibility or are constrained by computational and licensing limitations.

3.2 Comparison with Other SOTA Models

Several other models in music generation were evaluated, but they presented limitations for our project:

- **Jukebox (OpenAI) [5]:**
  Strengths: High-quality, diverse music generation capabilities.
  Limitations: Extremely large and computationally expensive, making fine-tuning impractical. Its focus on full song generation, including lyrics, is less suited to instrumental guitar music.

- **RAVE (Realtime Audio Variational autoEncoder) [7]:**
  Strengths: Lightweight and supports real-time audio generation.
  Limitations: Primarily designed for timbre transformations, limiting its ability to generate intricate guitar compositions.

- **MuseNet (OpenAI) [6]:**
  Strengths: Generates multi-instrument compositions across various styles.
  Limitations: Lacks the flexibility and specificity needed for conditioning or fine-tuning for guitar-focused tasks.

3.3 Low-Rank Adaptation (LoRA)

In this study, we use the MusicGen Melody 1.5B model. To efficiently train under hardware resource constraints, we applied Low-Rank Adaptation (LoRA) [9] for fine-tuning. LoRA reduces the number of trainable parameters, allowing for more efficient control and saving VRAM memory used by the optimizer state during the fine-tuning process. We preliminaryly examined the relationship between rank and VRAM usage in our dataset (see Table 1). Adjusting the rank significantly impacts the consumption of VRAM.

3.4 Data Augmentation

To enhance the diversity and robustness of the dataset used for fine-tuning MusicGen, data augmentation techniques are applied to the comping section of the GuitarSet dataset [10]. This is achieved using the *dasp-pytorch* repository [11], a toolset that facilitates audio processing with reverberation, distortion, dynamic range processing, equalization and stereo manipulation.

Reverberation is introduced to simulate natural acoustic environments and add depth to the audio. The reverb parameters are set to a decay time of 0.5 seconds and a

LoRA Param	GPU Memory Usage
rank=16	15 GB
rank=128	18.2 GB
rank=512	25.2 GB

Table 1. Relationship between LoRA rank and GPU VRAM usage

wet/dry mix ratio of 0.5, balancing the original signal with the processed one. The number of samples is configured at 32,768, ensuring a consistent reverb effect across the augmented samples. Additionally, distortion is applied to emulate the saturated and overdriven tones characteristic of certain guitar styles with parameter drive set to 32.

By incorporating these augmentations, we attempt to improve the model's ability to generate realistic and expressive guitar sounds during training.

3.5 Different Dataset

To enrich the diversity of guitar sounds and include music featuring guitars with other instruments, we attempted to create a custom guitar dataset by sourcing suitable music from YouTube videos.

We broadly constrained our search to relevant instruments , musical styles, and performance types, collecting an initial set of 100 tracks. To refine the dataset, we applied two measures: manually filtering out unsuitable audio files through random sampling and using source separation to remove vocals, ensuring the focus remained on instrumental audio.

4. EXPERIMENTAL SETUP

4.1 Experimental Strategy

We designed two different training strategies and experimented with the model performance of both strategies.

4.1.1 Strategy 1 - Add Special Token

In our application, to initially enhance control over the ability of the model to generate guitar sounds - and to allow further control of rhythm and playing styles using audio input in later stages - our goal is to design a controllable mechanism that ensures that the model generates only guitar-related music. Inspired by the training approach of text-to-image, DreamBooth [12], we sought to strengthen the association between a special token and guitar audio. By increasing the frequency of a special token's occurrence in the training data and its corresponding audio patterns, we aim to improve the model's controllability over guitar sound generation linked to this special token.

To meet the need for a controllable mechanism, the special tokens in our application differ from those commonly referenced in traditional Natural Language Processing (NLP). Here, these tokens do not need to be inherently "special" but must have a clear and consistent correspondence with the audio data patterns. Based on the characteristics of GuitarSet data, our primary special tokens are **Guitar**, **<solo>**, and **<comping>**. All guitar solo audio files are paired with the text prompt input "Guitar, <solo>," while guitar comping audio files follow a similar pattern. Using this approach as a foundation, we further experimented with other variables to optimize the model.

4.1.2 Strategy 2 - Enrich Text Prompt

In this experiment, we first designed five distinct text prompts to guide the MusicGen model in generating high-quality guitar music. The purpose of this step was to evaluate which prompt yields the best results when generating guitar music that aligns with our goals. After selecting the most effective prompt, we will proceed with further training using the augmented dataset, followed by training on a larger dataset consisting of YouTube music.

The five initial prompts used in the experiment are as follows.

- **Prompt 1:** Guitar, <solo>
- **Prompt 2:** High-quality acoustic guitar solo, virtuosic performance
- **Prompt 3:** High-quality acoustic solo guitar instrumental, intricate fingerpicking, rich harmonic textures, clear articulation, gentle emotional depth, natural recording ambiance, no percussion, pure guitar melody
- **Prompt 4:** High-quality acoustic guitar solo, virtuosic performance, pure guitar melody but gentle emotional depth with dynamic techniques
- **Prompt 5:** High-quality acoustic guitar solo, virtuosic performance, clear articulation

Each prompt was chosen to guide the model in generating music with varying degrees of complexity and style. After analyzing the results from these prompts, the best-performing prompt will be used for further training to enhance the model's ability to generate high-quality guitar music. This iterative process of refining the prompt and training the model on specialized datasets is intended to improve the overall quality and expressiveness of the generated music.

4.2 Dataset

4.2.1 GuitarSet

GuitarSet [10] is a comprehensive dataset of high-quality guitar recordings paired with detailed annotations. It comprises 360 excerpts, each 30 seconds long, designed to capture a diverse range of guitar performances. These excerpts are generated from combinations of six players,

two playing styles (comping and soloing), five musical styles (Rock, Singer-Songwriter, Bossa Nova, Jazz, and Funk), three harmonic progressions (12 Bar Blues, Autumn Leaves, and Pachelbel Canon), and two tempi (slow and fast).

In addition to the audio recordings, GuitarSet includes basic annotations such as pitch contour and MIDI note data (one per string), as well as chord, beat, and tempo annotations. While these annotations provide useful context, they are not heavily utilized in our fine-tuning process, which primarily focuses on leveraging the guitar audio itself to generate realistic guitar music. Nevertheless, they offer some additional reference that could be helpful for further model improvements in the future.

4.2.2 YouTube Dataset

GuitarSet is a dataset of pure acoustic guitar. To enrich the diversity of guitar sounds and include music featuring guitars with other instruments, we attempted to create a custom guitar dataset by sourcing suitable music from YouTube videos.

We used five sets of keywords to search for videos: 1) "guitar jamming session", 2) "jazz guitar trio", 3) "guitar instrumental live record", 4) "best acoustic guitar rock songs", 5) "guitar live performance old songs."

These keywords provided broad constraints on instruments, musical styles, and playing styles. Initially, we collected a total of 100 tracks. However, keyword-based searches can only roughly narrow down the scope and cannot control specific details of the audio files (e.g., the presence of guitar effects, the prominence of guitar sounds, or the inclusion of vocals). To address this issue, we applied two measures:

- **Manual Filtering:** We randomly sampled audio files and manually filtered out those unsuitable for guitar generation training.
- **Source Separation:** Using the source separation implementation from the 'musicgen-dreamboothing' repository, we removed vocals to ensure the model was trained on instrumental audio files.

Ultimately, we retained 70 tracks. Each track was randomly trimmed into up to 10 segments of 30 seconds each, resulting in a total of 596 wav files. For these audio files, we used the text prompt format "Music with Guitar - SEARCH KEYWORD" to create input-output pairs for subsequent experiments.

4.2.3 Fine-tune on Datasets

Based on different experimental strategies, we conducted fine-tuning training using various datasets. Due to hardware and team collaboration constraints, we adopted a continue training approach as our primary training method.

The datasets can be broadly categorized into five types: 1) GuitarSet solo, 2) GuitarSet comping, 3) GuitarSet solo (augmented), incorporating data augmentation techniques mentioned in Section 3.4, 4) GuitarSet comping (augmented), similar to the above, 5) YouTube dataset.

We believe that audio files sourced from GuitarSet exhibit higher quality and purer guitar sounds. Therefore, in the fine-tuning experiments, we prioritized datasets 1–4 and tested fine-tuning with the YouTube dataset only at the final stage.

4.3 Evaluation Metrics

In order to assess the quality of the generated guitar music, we chose CLAP (Contrastive Language-Audio Pretraining) [13] as our primary evaluation metric. CLAP is a state-of-the-art model designed to evaluate the relationship between textual prompts and audio outputs by learning to align audio and textual representations in a shared space. This alignment allows effective and robust evaluation of music generation models, particularly in tasks where the quality and relevance of the generated audio need to be judged against a textual description.

CLAP works by measuring how well the generated audio matches the semantic meaning of the provided text prompt. It does this through contrastive learning, where the model contrasts paired (text-audio) examples with unpaired examples, ensuring that the audio generated from a given text prompt is aligned with the intended meaning described in the prompt. This makes CLAP particularly well-suited for tasks like music generation, where the text description provides important details about the desired musical qualities.

We chose CLAP for the following reasons:

- **Text-Audio Alignment:** CLAP directly aligns the generated audio with the provided text, making it an effective metric for assessing how well the generated music corresponds to the prompt. This is crucial to make sure that the generated guitar music matches the desired characteristics described in the prompt.
- **State-of-the-Art Performance:** Compared to traditional metrics such as FID (Fréchet Inception Distance) [14] or IS (Inception Score) [15], CLAP has demonstrated superior performance in evaluating audio-visual tasks because it is specifically designed to handle multimodal data, capturing both the auditory and semantic aspects of the generated content.
- **Fine-Grained Evaluation:** CLAP allows for more nuanced evaluations, such as assessing emotional depth, clarity, and style in generated music, which are important for tasks like generating guitar solos with emotional expression and technical virtuosity.
- **Consistency and Reliability:** Given its pre-training on a large-scale dataset of both text and audio, CLAP provides a reliable evaluation framework that is consistent across different audio generation models and tasks.

In summary, CLAP provides a comprehensive and precise method for evaluating the quality of music generation

models. Its ability to evaluate the alignment between audio and textual prompts makes it an ideal choice for this project, where the goal is to generate guitar music that accurately reflects specific textual descriptions.

4.4 Implementation Details

The implementation of fine-tuning is conducted using the open-source repository MusicGen-Dreamboothing [16], which provides an efficient framework for adapting MusicGen models to specialized tasks. This repository supports the integration of LoRA (Low-Rank Adaptation) [9], allowing for parameter-efficient fine-tuning. Adjustments are made to the LoRA parameters(rank), instance prompt, and various hyperparameters to optimize the training process.

MusicGen-Dreamboothing also includes an interface for connecting to Weights & Biases (W&B) [17], a powerful tool for experiment tracking and visualization. This enables real-time monitoring of key metrics, including training loss, evaluation loss, and CLAP scores. It provides a clear view of the model performance.

Additionally, since MusicGen requires a large GPU VRAM, our training is conducted on Google Colab using an A100 GPU and on a local machine with an RTX 4070 GPU. MusicGen-Dreamboothing is designed to leverage GPU acceleration, ensuring efficiency in handling the computational demands of text-to-audio model fine-tuning.

5. RESULTS

5.1 Performance Result

In all model training and inference, prompt 4 is chosen with slight variation in different configurations. The baseline model used for comparison is the `facebook/musicgen-melody:1.5B` model from Hugging Face [18]. The models we trained include `model_s`, `model_sca`, and `model_sca_yt`, each trained on the GuitarSet dataset or YouTube music.

- **Baseline:** The baseline model generates nylon guitar sounds with a recognizable melody line, but it suffers from high repetition. The overall timbre resembles that of nylon strings, and the melodic structure is coherent, though somewhat repetitive.
- **model_s:** This model, trained on the solo section of GuitarSet for 100 epochs, generates melodies with an acoustic guitar timbre, which is closer to GuitarSet's acoustic guitar section. The melody is less clear than the baseline, but it sounds richer and more varied.
- **model_sca:** Trained on the augmented comping section of GuitarSet for 100 epochs after `model_s`, this model produces a more rhythmic output but at the cost of losing the melodic structure learned in `model_s`. The result is less melodic but captures the rhythmic complexity typical of GuitarSet's comping section.
- **model_sca_yt:** This model was fine-tuned on YouTube Music for 100 epochs after `model_sca`. It produces electronic music noise and sounds significantly different from the previous models. The timbre is quite distinct, and the melody is absent or highly unclear, with a noticeable departure from the guitar sounds.

The results clearly demonstrate that fine-tuning MusicGen has a significant impact on the generated audio, with the outcome largely influenced by factors such as the dataset, the training sequence, and the relationship between the text prompt and the audio. Although the sound quality of the fine-tuned models does not surpass that of the baseline, the generated audio distinctly reflects the characteristics of the corresponding datasets. This indicates that fine-tuning MusicGen on domain-specific data, such as GuitarSet, can lead to a model that better captures the timbre and stylistic elements of the instrument. The next section will discuss the potential variables contributing to these outcomes, including dataset choice, training progression, and the effectiveness of prompt-audio correspondence on model performance.

5.2 Prompt Engineering

To evaluate the effectiveness of our prompts, we first measured the CLAP scores for each of the five text prompts. The results are shown in the table 2 below:

From these scores, it is evident that the fourth prompt, which included "gentle emotional depth" and "dynamic techniques," yielded the highest CLAP score of 0.26. This suggested that it was the most aligned with the intended musical qualities and produced the most effective results in terms of generating high-quality guitar music.

Initially, we believed that using a special token such as `Guitar <solo>` would provide a simple way to generate the desired music. However, this approach did not work as expected. Instead of producing guitar music, the output often resembled a trumpet-like sound without any melody. As a result, we abandoned the special token strategy and turned to more descriptive prompts.

After testing several more descriptive prompts, we found that adding adjectives and specific details such as "high-quality acoustic solo guitar" and "intricate fingerpicking" improved the generated music. However, while the music was more aligned with our expectations, it still lacked the emotional depth and pleasant qualities we were aiming for.

We then experimented with adding more adjectives to the prompt in an attempt to guide the model more precisely. While this approach seemed promising at first, we found that overly complicated prompts hindered the model's learning ability, making it difficult for the model to generate high-quality music. This overcomplication resulted in poor performance.

After analyzing the results from the previous iterations, we selected the fourth prompt for further testing. The key elements in this prompt were "gentle emotional depth" and

Prompt	CLAP Score
Guitar, <solo>	0.18
High-quality acoustic guitar solo, virtuosic performance	0.09
High-quality acoustic solo guitar instrumental, intricate fingerpicking, rich harmonic textures, clear articulation, gentle emotional depth, natural recording ambiance, no percussion, pure guitar melody	0.10
High-quality acoustic guitar solo, virtuosic performance, pure guitar melody but gentle emotional depth with dynamic techniques	0.26
High-quality acoustic guitar solo, virtuosic performance, clear articulation	0.15

Table 2. CLAP scores for each prompt

"dynamic techniques." The term "gentle emotional depth" was used to guide the model towards generating music that was emotionally rich, while "dynamic techniques" was included to ensure that the music had a clear and engaging melody. The output from this prompt was significantly better than the earlier ones.

Finally, we made a small adjustment by replacing "dynamic techniques" with "clear articulation" to emphasize clarity in the generated music. The ultimate goal was not to produce highly complex techniques, but to ensure that the music was enjoyable to listen to and had a clear melodic structure. The results of this refined prompt were the most promising.

The final CLAP scores support our decision, with the fourth prompt achieving the highest score of 0.26, indicating the best alignment with our expectations for good, high-quality guitar music.

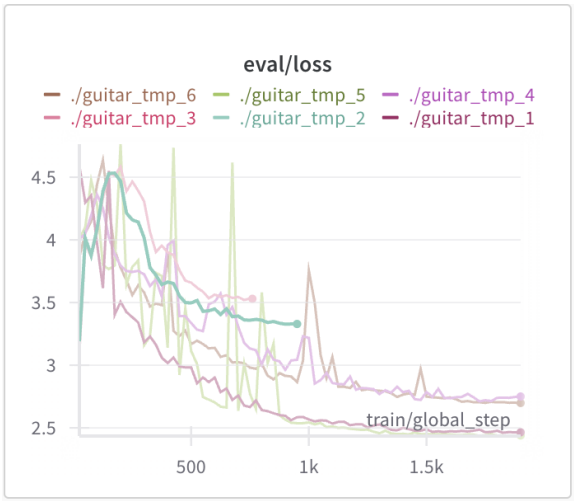

Figure 1. Fine-tuning process evaluation loss over epochs

5.3 Hyperparameter Study

In this section, we analyze the effect of various hyperparameters on the fine-tuning process of the model. Three key observations were made during the study, and the results are illustrated in the following figures.

First, as shown in Figure 1, we found that the fine-tuning process requires at least 100 epochs to converge. While the evaluation loss continuously decreases over time, we observed that it does not align with the CLAP score, as shown in Figure 2. Specifically, although the evaluation loss keeps decreasing, the CLAP score does not consistently improve. This indicates that while the loss metric suggests the model is improving, it doesn't necessarily lead to better alignment between the generated music and the textual prompts.

Next, we analyzed the impact of changing the LoRA configuration, specifically comparing `guitar_tmp_5` and `guitar_tmp_6`, where the only change was increasing the LoRA configuration from 16 to 32. As shown in Figure 1, we observed that the evaluation loss for `guitar_tmp_6` (with LoRA config 32) was much lower than the others, which used the LoRA config of 16. However, the CLAP score remained similar between `guitar_tmp_5` and `guitar_tmp_6`, suggesting that while the LoRA configuration influenced the loss, it did not have a significant impact on the alignment of the music with the prompt. Therefore, we conclude that the CLAP score is more closely related to the training prompt used rather than the LoRA configuration, which primarily affects the evaluation loss but does not necessarily lead to better music generation.

Finally, as shown in Figure 3, we discovered that prompts that started with higher CLAP scores were more likely to result in higher CLAP scores after further training. This suggests that even though the training process may take a considerable amount of time, we can still improve the model's output by periodically checking the CLAP score for a few epochs to determine the direction of improvement. Thus, the CLAP score can be a useful indicator of model performance, helping guide decisions during the training process.

In summary, the hyperparameter study reveals that the evaluation loss is closely related to the LoRA configuration, while the CLAP score is more influenced by the training prompt. Additionally, monitoring the CLAP score throughout training can provide valuable insight into the potential improvement of the model, even if the fine-tuning process takes a long time.

6. CONCLUSION

This project successfully demonstrated the potential of fine-tuning MusicGen to specialize in generating high-quality guitar music from textual prompts. By leveraging the GuitarSet dataset and experimenting with various fine-tuning strategies, we were able to capture distinct charac-

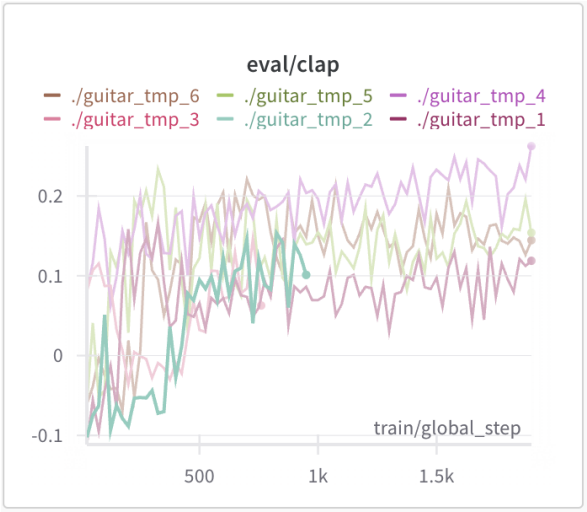

Figure 2. Fine-tuning process evaluation CLAP score over epochs

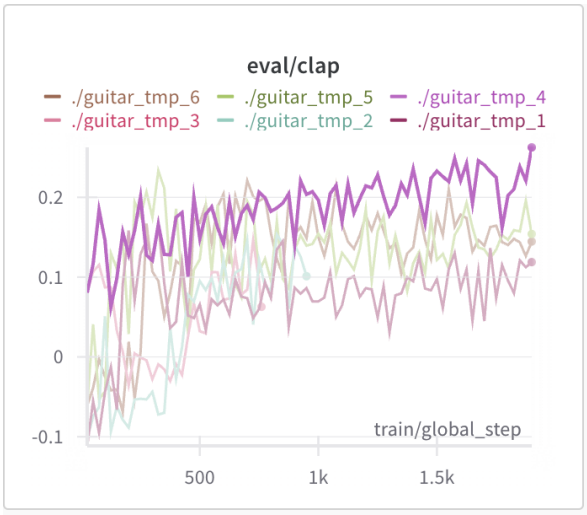

Figure 3. Correlation between initial and final CLAP scores for different prompts

teristics of guitar timbre and style. Our key contributions and findings can be summarized as follows:

- We established the effectiveness of descriptive prompts in guiding MusicGen to produce better-aligned guitar music. Among the tested prompts, the fourth prompt, which emphasized "gentle emotional depth" and "dynamic techniques," achieved the highest CLAP score of 0.26, demonstrating its alignment with the intended musical qualities.
- Through hyperparameter studies, we discovered that the LoRA configuration significantly impacts evaluation loss but has limited influence on the CLAP score, which is more strongly tied to the training prompt. This underscores the importance of prompt design in text-to-audio models.
- Our analysis showed that prompts with higher initial CLAP scores tended to maintain their performance through training, suggesting that careful prompt selection early in the process is critical for achieving optimal results.
- While the fine-tuned models captured unique aspects of the GuitarSet dataset, such as timbre and rhythmic complexity, challenges remain in consistently generating realistic and expressive melodies comparable to human performance. The results highlighted the trade-offs between dataset specialization and general audio generation capabilities.

6.1 Limitations and Future Work

Despite the progress made, this work also encountered several limitations that open opportunities for future research:

- **Training Strategy:** In Strategy 2, where we continued training on different datasets but used similar prompts, the CLAP score decreased with each step. This behavior might stem from difficulties in training caused by the proximity of prompts in the text embedding space. Future work could refine the training progression to mitigate this issue and improve performance.
- **Dataset Diversity:** The datasets used in this study presented challenges in achieving diverse and high-quality results. The GuitarSet dataset, while detailed, is relatively monotonic, and the music sourced from YouTube is noisy and lacks consistency. Future research should focus on acquiring pure instrumental music with high-resolution audio files and developing strategies to generate more diverse and compatible text-audio pairings for training.

6.2 Concluding Remarks

In conclusion, this study contributes valuable insights into the fine-tuning of text-to-audio models for instrument-specific tasks. It highlights the critical role of prompt design, dataset selection, and training strategies in achieving high-quality audio generation. While challenges remain, this work lays a foundation for future advancements in generative audio systems, offering tools for musicians, educators, and creators seeking realistic and expressive guitar audio generation.

7. REFERENCES

[1] J. Copet, F. Kreuk, I. Gat, T. Remez, D. Kant, G. Synnaeve, Y. Adi, and A. Défossez, "Simple and controllable music generation," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[2] M. Božić and M. Horvat, "A survey of deep learning audio generation methods," *arXiv preprint arXiv:2406.00146*, 2024.

[3] A. Wright, E.-P. Damskägg, L. Juvela, and V. Välimäki, "Real-time guitar amplifier emulation with deep learning," *Applied Sciences*, vol. 10, no. 3, p. 766, 2020.

[4] H. Kim, S. Choi, and J. Nam, "Expressive acoustic guitar sound synthesis with an instrument-specific input representation and diffusion outpainting," in *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2024, pp. 7620–7624.

[5] P. Dhariwal, H. Jun, C. Payne, J. W. Kim, A. Radford, and I. Sutskever, "Jukebox: A generative model for music," *arXiv preprint arXiv:2005.00341*, 2020.

[6] A. Pal, S. Saha, and R. Anita, "Musenet: Music generation using abstractive and generative methods," *International Journal of Innovative Technology and Exploring Engineering*, vol. 9, no. 6, pp. 784–788, 2020.

[7] A. Caillon and P. Esling, "Rave: A variational autoencoder for fast and high-quality neural audio synthesis," *arXiv preprint arXiv:2111.05011*, 2021.

[8] A. Agostinelli, T. I. Denk, Z. Borsos, J. Engel, M. Verzetti, A. Caillon, Q. Huang, A. Jansen, A. Roberts, M. Tagliasacchi *et al.*, "Musiclm: Generating music from text," *arXiv preprint arXiv:2301.11325*, 2023.

[9] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "Lora: Low-rank adaptation of large language models," *arXiv preprint arXiv:2106.09685*, 2021.

[10] Q. Xi, R. M. Bittner, J. Pauwels, X. Ye, and J. P. Bello, "Guitarset: A dataset for guitar transcription." in *ISMIR*, 2018, pp. 453–460.

[11] C. Steinmetz, "Dasp-pytorch: Differentiable audio signal processing," https://github.com/csteinmetz1/dasp-pytorch, 2024, accessed: 2024-12-26.

[12] N. Ruiz, Y. Li, V. Jampani, Y. Pritch, M. Rubinstein, and K. Aberman, "Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 22 500–22 510.

[13] B. Elizalde, S. Deshmukh, M. Al Ismail, and H. Wang, "Clap learning audio concepts from natural language supervision," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5.

[14] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," *Advances in neural information processing systems*, vol. 30, 2017.

[15] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," *Advances in neural information processing systems*, vol. 29, 2016.

[16] Y. Lacombe, "musicgen-dreamboothing: Fine-tuning musicgen," 2024, gitHub repository. [Online]. Available: https://github.com/ylacombe/musicgen-dreamboothing

[17] W. . Biases, "Weights & biases: Machine learning experiment tracking," https://wandb.ai/site/, 2024, accessed: 2024-12-26.

[18] H. Face, "Hugging face: The ai community building the future," https://huggingface.co/, 2024, accessed: 2024-12-26.

A. WORK DIVISION

The contributions of each team member are detailed in the table 3.

Name	**Contribution**
Jih-Ming Pai	Data collection, model tuning and analysis
Ting-An Yin	Data augmentation, model tuning and analysis
I-Pei Lee	Prompt engineering, model tuning and analysis

Table 3. Work Division Among Team Members