General Adversarial Networks (GANs) for Music Generation

Kazuki Neo[1*] **Muhammad Haris Saleem** [1*]
[1]Information Technology and Web Sciences
Rensselaer Polytechnic Institute
{neok,saleem}@rpi.edu

GAN Music Generation - Project Github Repository Link - https://github.com/kazneo/GAN-Music

1 Introduction

In recent years, the advent of Generative Adversarial Networks (GANs) has sparked significant interest across various domains, from image synthesis to natural language processing. Among these applications, one area of particular intrigue is the generation of musical content. This project delves into the realm of GAN-based music generation, exploring its potential to revolutionize the way we conceive, compose, and experience music.

Music, as a form of artistic expression, embodies a rich tapestry of melody, harmony, and rhythm. Traditionally, the creation of music has been a deeply human endeavor, rooted in creativity, intuition, and cultural context. However, with advancements in deep learning and computational techniques, the boundaries between human and machine creativity are becoming increasingly blurred.

Some relevant work has been done in understanding of different techniques that can be utilized for Music and melody generation using LSTM, CNN and other similar architectures. A survey by Y.Zhu [1] highlights several non-neural network and neural network approaches are discussed for music generation, some of the AI based approaches include Variational Auto Encoders (VAE), Transformers, LSTM and Diffusion based models. The research couldn't conclude what algorithm performs best, but highlighted advantages and disadvantages of each approach in music generation covering flexibility and complexity of approaches. Liu Quibin [2] utilized Melody-RNN and Biaxial-RNN to generate pretty listenable music pieces but faced some computational challenges. H.-W. Dong [3] utilized General Adversarial Networks (GANs) with CNN architecture to generate additional piano tracks similar to the human input provided. Dong tackled the data sparsity presented due to few notes being played in the entire song which harms the learning of the model by merging the tracks of similar instruments and making a piano roll. Each multi-track piano-roll is then compressed into five tracks: bass, drums, guitar, piano and strings. This enriches in the data which enhances the learning of the model to generate diversified music. Neves [4] discussed use of Transformer GANs to capture human sentiment states in the music utilizing valence and arousal dimensions.

At the heart of GAN music generation lies a dynamic interplay between two neural networks: the generator and the discriminator. The generator aims to produce music that is indistinguishable from human-composed pieces, while the discriminator seeks to differentiate between generated and real music. Through iterative training, these networks engage in a game of cat and mouse, honing their abilities to generate increasingly convincing musical compositions. This paper seeks to explore the underlying mechanisms of GAN-based music generation, shedding light on the technical intricacies involved in training such models. Additionally, it aims to evaluate the efficacy of GANs in capturing the essence of musical creativity, examining the quality, diversity, and originality of generated compositions.

By unraveling the inner workings of GAN music generation, this research endeavor aims to contribute to a deeper understanding of the intersection between artificial intelligence and artistic expression. Fur-

I Can't Believe It's Not Better Workshop III at NeurIPS 2022.

thermore, it seeks to provoke critical discourse on the implications of automated music composition for the future of musical creativity, aesthetics, and cultural heritage.

In the subsequent sections, we delve into the methodology employed in GAN music generation, present our experimental findings, and discuss the broader implications of our research. Through rigorous analysis and reflection, we endeavor to elucidate the potential, challenges, and ethical considerations surrounding the integration of AI in the realm of music composition.

2 Problem Statement

Synthesizing music that remains coherent across longer timescales while still capturing the local aspects that make it sound "realistic" or "human-like" is still challenging. Therefore, there is a need for novel techniques that can effectively model and generate coherent musical sequences over extended duration's, while retaining the local characteristics that make the output sound authentic and appealing to human listeners.

3 Methodology

The GAN model comprises two fundamental components: the LSTM-based generator and the discriminator. The generator is tasked with producing realistic music sequences, while the discriminator aims to differentiate between real and generated sequences.

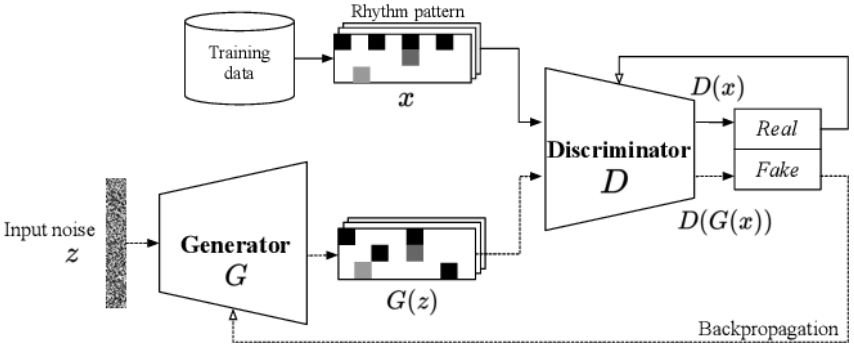

Figure 1: GANs Training

Utilizing MIDI data representation, which encapsulates musical notes, durations, and offsets, as input to both models, the generator synthesizes sequences of notes, durations, and offsets, while the discriminator learns to discern between real and generated sequences. Adversarial training is employed, where the generator seeks to deceive the discriminator, while the discriminator endeavors to distinguish between real and synthetic samples. This iterative training process progressively refines the parameters of both models to enhance their performance.

The LSTM-based generator consists of multiple layers of LSTM cells followed by a fully connected layer to generate the output sequence.

Initially, a random noise vector serves as input to the generator, undergoing transformation into a sequence of notes, durations, and offsets. Similarly, the discriminator, built upon LSTM cells, receives sequences of notes, durations, and offsets as input and generates a probability score indicating the authenticity of the input sequence. Training entails utilizing a dataset of MIDI files, preprocessed to extract musical features such as notes, durations, and offsets. The dataset is partitioned into batches, and the models are trained using mini-batch stochastic gradient descent. Throughout the training phase, both the generator and discriminator undergo iterative updates via adversarial training, persisting until convergence or completion of a predefined number of epochs.

For evaluation, Binary Cross-Entropy (BCE) served as the primary loss metric, aligning with prevalent practices in GANs. This metric facilitated the quantitative assessment of the discriminator's discernment ability and the generator's proficiency in producing authentic music sequences. Complementing the BCE loss, a comprehensive set of evaluation metrics spanning qualitative and quantitative

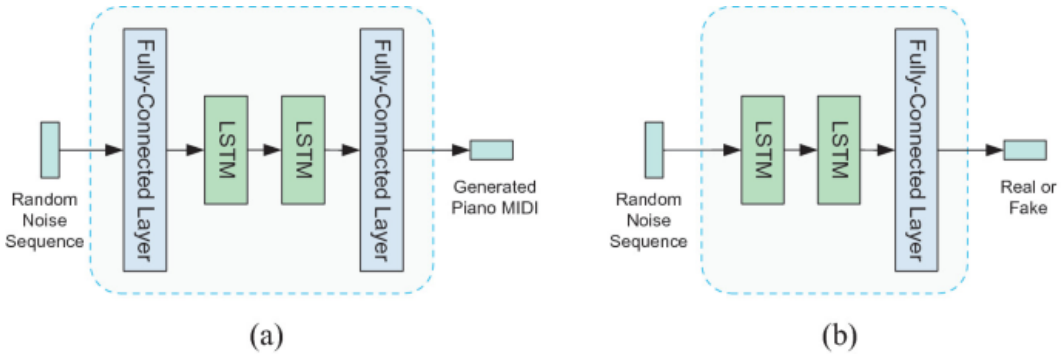

Figure 2: GANs Architecture - (a) Generator, (b) Discriminator

```python
class LSTMGenerator(nn.Module):
    def __init__(self, input_size, hidden_size, output_size, num_layers):
        super(LSTMGenerator, self).__init__()
        self.hidden_size = hidden_size
        self.num_layers = num_layers
        self.lstm = nn.LSTM(input_size, hidden_size, num_layers, batch_first=True)
        self.fc = nn.Linear(hidden_size, output_size)

    def forward(self, x):
        batch_size = x.size(0)
        h0 = torch.zeros(2, batch_size, self.hidden_size).to(x.device)
        c0 = torch.zeros(2, batch_size, self.hidden_size).to(x.device)
        out, _ = self.lstm(x.unsqueeze(1), (h0.squeeze(0), c0.squeeze(0)))
        out = self.fc(out[:, -1, :])
        return out

class LSTMDiscriminator(nn.Module):
    def __init__(self, input_size, hidden_size, num_layers):
        super(LSTMDiscriminator, self).__init__()
        self.hidden_size = hidden_size
        self.num_layers = num_layers
        self.lstm = nn.LSTM(input_size, hidden_size, num_layers, batch_first=True)
        self.fc = nn.Linear(hidden_size, 1)

    def forward(self, x):
        batch_size = x.size(0)
        h0 = torch.zeros(2, batch_size, self.hidden_size).to(x.device)
        c0 = torch.zeros(2, batch_size, self.hidden_size).to(x.device)
        out, _ = self.lstm(x.unsqueeze(1), (h0.squeeze(0), c0.squeeze(0)))
        out = self.fc(out[:, -1, :])
        return torch.sigmoid(out)
```

Listing 1: Python code for LSTM Generator and LSTM Discriminator classes.

dimensions was employed. These metrics encompassed diverse aspects such as pitch accuracy, rhythmic coherence, and harmonic progression, providing a nuanced understanding of the fidelity and musicality of the generated compositions. By adopting a multifaceted evaluation approach, the study aimed to provide a holistic appraisal of the LSTM-based GAN framework's effectiveness in generating high-quality musical outputs.

4 Experiments

4.1 Dataset

The implementation leveraged the PyTorch deep learning framework, renowned for its flexibility and efficiency, for the development and training of the GAN models. The GAN models were trained utilizing a meticulously curated dataset sourced from Kaggle, comprising a diverse collection of over 290 classical music compositions. This dataset served as a rich source of musical inspiration, encompassing a wide array of styles, tempos, and instrumentation commonly found in classical music repertoire.

4.2 Preprocessing

Prior to model training, the dataset underwent thorough preprocessing to extract essential musical features, including notes, durations, and offsets. This feature extraction was done using Python music21 and typing libraries.

This preprocessing step was essential to transform the raw MIDI music files into a structured format suitable for training the LSTM-based generator and discriminator. By extracting these musical features, the models were equipped with the necessary information to learn and generate coherent musical sequences.

4.3 Training

The training regimen spanned 100 epochs, with a carefully chosen batch size of 64 and a learning rate of 0.001 for both the generator and discriminator. These hyper-parameters were selected through iterative experimentation to strike a balance between model convergence and computational efficiency, ensuring robust and stable training dynamics.

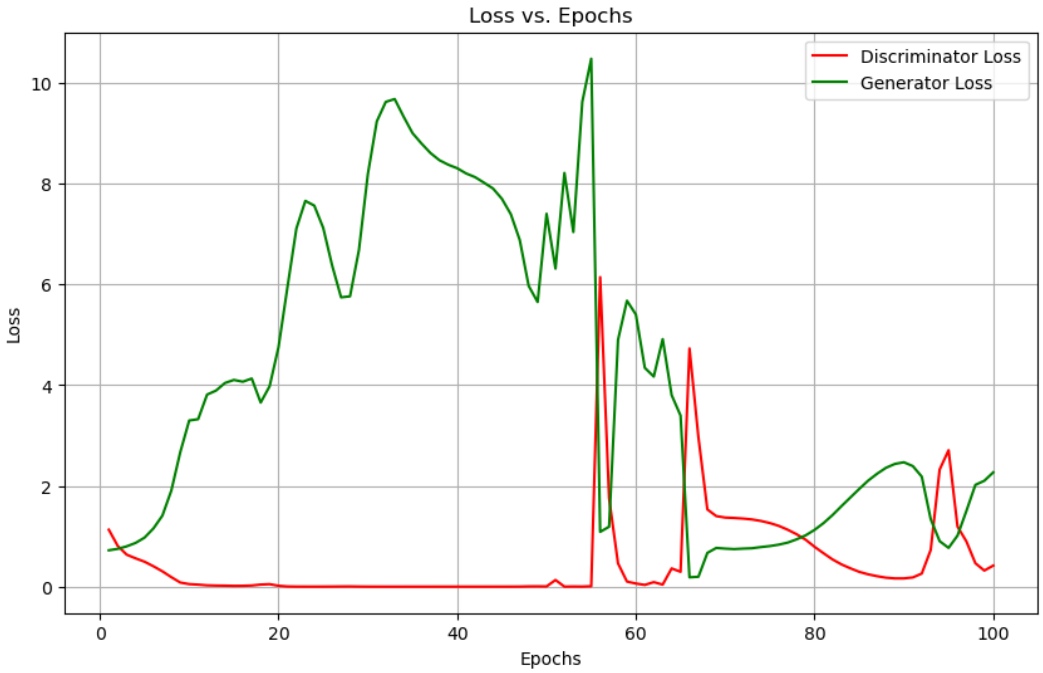

Figure 3: Model Training Losses

The loss for Generator in Figure 3 is very high at the start as the input data for generator is random noise, however the model trains itself based on back propagation from discriminator and generator. After a while there is a drop in generator loss but a spike in discriminator loss suggesting model is

unable to detect real from fake effectively. This iteration goes on for 100 epochs when both the losses seem be at lowest point and are in slow increase and decrease trend.

4.4 Results

The experimental outcomes yielded promising results, showcasing the efficacy of the LSTM-based GAN framework in generating high-quality music sequences reflective of classical compositions. The generated music exhibited fidelity to the training data, capturing the intricate nuances of classical melodies, rhythms, and harmonies. However, the experimentation process also revealed challenges such as mode collapse and training instability, highlighting potential areas for further investigation and refinement.

5 Conclusion

The study provides a nuanced exploration into the capabilities and potential of LSTM-based GANs in the domain of classical music generation. While the findings demonstrate promising results, it is important to acknowledge that the generated music, while evocative, may not yet attain the pinnacle of sonic excellence expected in professional compositions.

The generated music exhibits certain hallmarks of classical compositions, including diverse melodies, coherent rhythms, and harmonic richness. However, it is evident that challenges such as mode collapse and training instability impact the fidelity and overall quality of the generated compositions. The study sheds light on the untapped potential inherent within LSTM-based GANs for music generation. The compositions produced, while not devoid of imperfections, offer valuable insights into the intersection of artificial intelligence and artistic creativity.

Moving forward, it is imperative for future research endeavors to address the limitations identified in our study and strive towards refining model architectures, optimizing training methodologies, and exploring innovative techniques. By embracing these challenges and opportunities, we can further advance the state-of-the-art in computational music generation and unlock new avenues for artistic expression.

6 References

[1] Y. Zhu, J. Baca, B. Rekabdar, and R. Rawassizadeh, “A Survey of AI Music Generation Tools and Models,” 2023. Available: https://arxiv.org/pdf/2308.12982.pdf

[2] Q. Lou, “Music Generation Using Neural Networks.” Accessed: May 02, 2024. [Online]. Available: https://cs229.stanford.edu/proj2016/report/Lou-MusicGenerationUsingNeuralNetworks-report.pdf

[3] H.-W. Dong, W.-Y. Hsiao, L.-C. Yang, and Y.-H. Yang, “MuseGAN: Multi-track Sequential Generative Adversarial Networks for Symbolic Music Generation and Accompaniment,” Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32, no. 1, Apr. 2018, doi: https://doi.org/10.1609/aaai.v32i1.11312.

[4] P. Neves, J. Fornari, and J. Florindo, “GENERATING MUSIC WITH SENTIMENT USING TRANSFORMER-GANS.” Accessed: May 02, 2024. [Online]. Available: https://arxiv.org/pdf/2212.11134.pdf

[5] C. Hernandez-Olivan and J. Beltran, “MUSIC COMPOSITION WITH DEEP LEARNING: A REVIEW.” Available: https://arxiv.org/pdf/2108.12290.pdf