MUSIC GENERATION USING AI

MINI PROJECT REPORT

Prepared By :

Shubham Golwal | 2020300015
Krutik Shah | 2020300062
Vatsal Shah | 2020300065

CERTIFICATION

This is to certify that the work on the mini-project titled “Music Generation Using AI” has been carried out by the following students, who are students of Sardar Patel Institute of Technology, Mumbai.

1. Shubham Golwal (2020300015)
2. Krutik Shah (2020300062)
3. Vatsal Shah (2020300065)

Project Guide: Prof. Anand Godbole

ACKNOWLEDGEMENTS

SEM VI | Music Generation using AI Project Report

We feel great pleasure in presenting the report of our mini project titled "Music Generation using AI". We have channelized our best efforts towards a systematic approach to the project, keeping in mind the aim we need to achieve.

We are highly grateful to our **project Guide** Prof. Anand Godbole, Sardar Patel Institute of Technology (SPIT) for constant encouragement, effort, and guidance. He has always been involved in discussing our topic at each phase to make sure that our approach was designed and carried out in an appropriate manner and that our conclusions were appropriate, given our results.

The **writers** behind the impact report includes Shubham Golwal, Vatsal Shah & Krutik Shah.

TABLE OF CONTENTS

PROBLEM STATEMENT

In today's world, it's a myth that you need to be a music expert to generate music. Even a person who likes music can produce good quality music. We all like to listen to music and if it is possible to generate music automatically then it will prove to be a new revolution in the world of music industry.

Until very recently, all music generation was done manually by means of analogue signals. In recent years this has become possible with the help of AI and Deep Learning.

Overall, the goal of the music generation system is to enable musicians, composers, and music enthusiasts to easily generate original music pieces, while also pushing the boundaries of what is possible with artificial intelligence and music composition.

LITERATURE SURVEY

- **“Fine-Tuning Music Generation with Reinforcement Learning Based on Transformer”-**The research paper proposes a method to improve the quality and diversity of music generated by transformer-based models using reinforcement learning. The approach fine-tunes the model by providing rewards based on musical quality and diversity, which encourages the model to generate musically pleasing and diverse sequences that adhere to certain musical constraints. The paper demonstrates that the proposed method outperforms existing state-of-the-art methods in terms of musical quality and diversity. This approach has significant implications for music generation applications in various domains.

- "The research paper presents a transformer-based model for generating expressive pop piano compositions with beat-based modeling”- The model is trained using a dataset of pop piano compositions and incorporates beat information into the generation process to produce more rhythmically interesting and expressive compositions. The authors demonstrate that their approach outperforms existing methods in terms of musical quality and expressiveness, which has potential applications in music production, education, and entertainment.

LITERATURE SURVEY

- The research paper "**Music Transformer: Generating Music with Long-Term Structure**" proposes a transformer-based model that generates music with coherent and meaningful musical phrases by training on a large dataset of symbolic music. The model uses a novel technique called relative self-attention that allows it to capture long-term musical structure and outperforms existing state-of-the-art methods in generating musically coherent and meaningful phrases. This approach has potential applications in various domains, including music composition, education, and entertainment.

- The research paper "**A Study on the Application of Deep Learning Algorithm and Computer Neural Networks in Automatic Generation According to User Style**" proposes a method that combines deep learning algorithms and neural networks to generate music that adheres to user-defined musical styles. The approach creates a user model that represents the user's preferred musical style and uses it to generate music. The authors demonstrate that their approach is capable of generating music that adheres to user-defined styles, which has potential applications in music production, education, and entertainment.

- The research paper "**Music Deep Learning: A Survey on Deep Learning Methods for Music Processing**" provides an overview of the current state of deep learning methods for music processing. The paper covers different types of data representations and deep learning architectures used for music processing, as well as their applications in various tasks such as classification, transcription, generation, and recommendation. The paper provides a comprehensive survey of the potential applications of deep learning methods in music production, education, and entertainment.

Constraints

1. Data Availability: One of the primary constraints is the availability of data. Music generation using AI requires a large amount of data to learn from. You may have to gather data from various sources such as audio recordings, MIDI files, or music notation sheets.
2. Musical Knowledge: The AI algorithms used for music generation require a significant amount of musical knowledge. You need to have a good understanding of music theory and composition, as well as experience in music production.
3. Evaluation Metrics: Music generation is subjective, and it can be challenging to evaluate the quality of the generated music objectively. You may need to develop metrics for evaluating the quality of the music generated by your AI model.
4. Copyright Issues: You need to ensure that the music generated by your AI model does not infringe on any copyright laws. You may need to be aware of the legal implications of using copyrighted music and seek appropriate permissions or licenses.
5. Training Time: Training AI models for music generation can be time-consuming. You may need to spend a considerable amount of time training your model on the available data to generate high-quality music.

SCOPE OF WORK

Assumptions

1.The pre-processing module can accurately extract relevant features from the MIDI files, such as note duration, pitch, and velocity.
2.The transformer model can learn the patterns and structures of the input music data and generate new, original music based on those patterns.
3.The MIDI files used for training and testing the transformer model are representative of the target music genre or style.

Limitations

- 1.Lack of emotion
- 2.Limited musical styles
- 3.Ethical concerns

SCOPE OF WORK

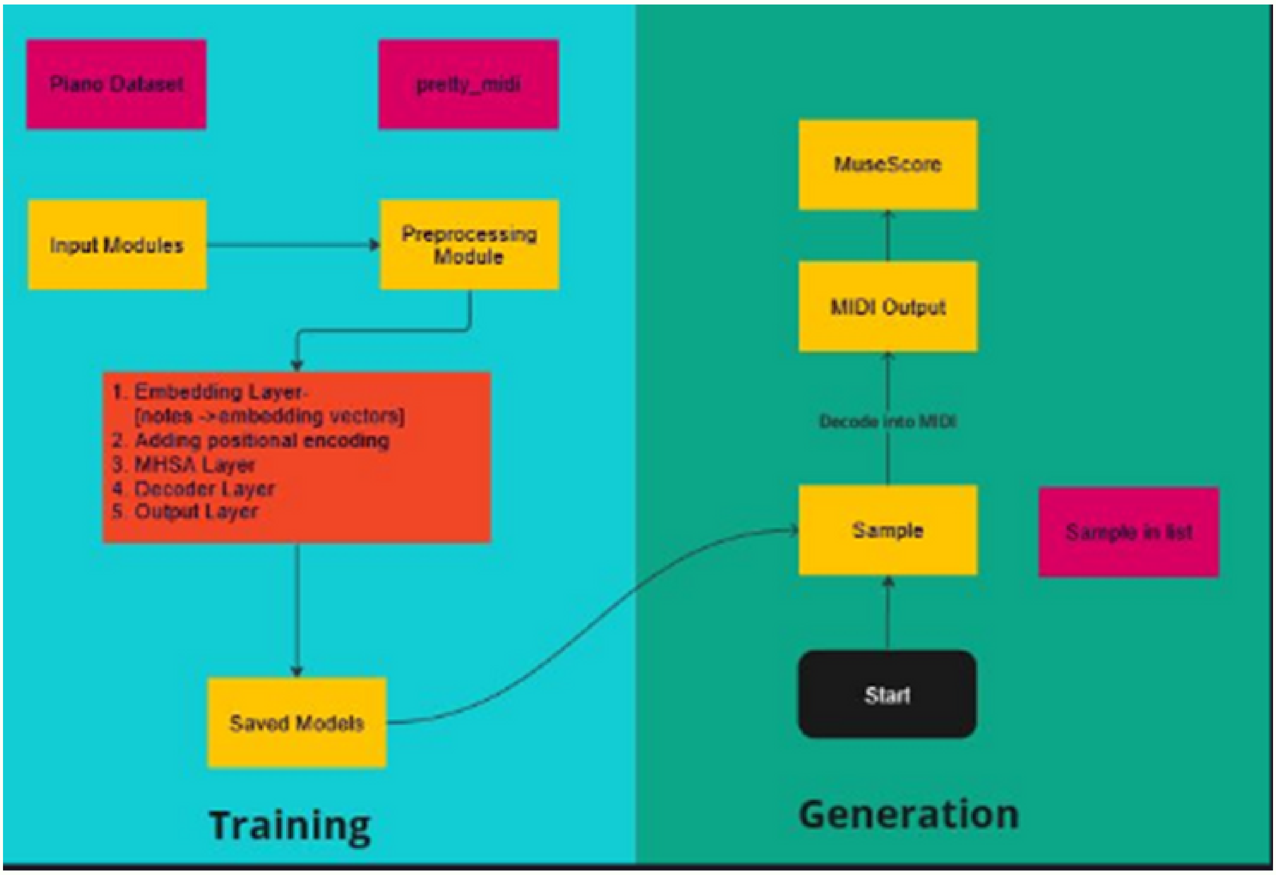
KNOW OUR OBJECTIVES!

1 Train an transformer model on a dataset of MIDI files to generate new, original music.

2 Evaluate the quality of the generated music using quantitative measures

3 Conduct user studies to assess the appeal and perceived creativity of the generated music among different audiences, such as musicians, music critics, or casual listeners.

ARCHITECTURE



00

Timeline for our Project

For Year 2022

01

FEB/MARCH

Week 1-2 : Explore deep learning.

Week 3-4: Literature Surveys of various research papers on topic.

Week 5-8 : Model Implementation sprint 1.

02

APRIL

Week 1-2 : Model building implementation sprint 2.

Week 3-4: Rigorous Testing +Sprint for additional Features, Documentation and cleaning the code.

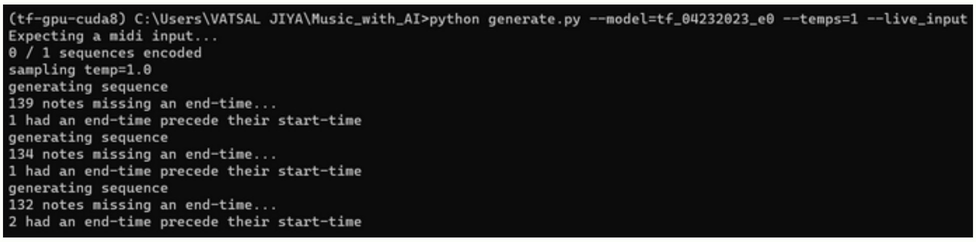
03

MAY

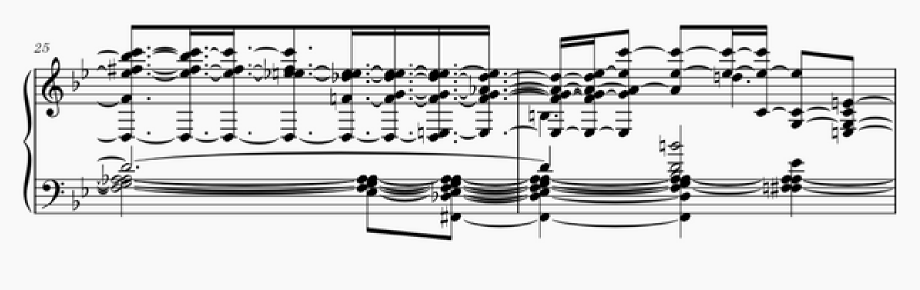
Week 1: Buffer

IMPLEMENTATION DETAILS

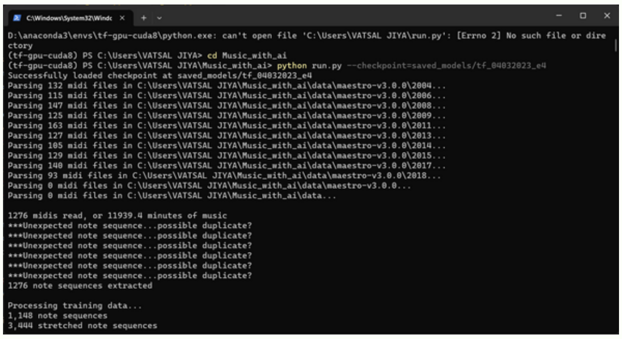
Output & Observation



```
(tf-gpu-cuda8) C:\Users\VATSAL JIYA\Music_with_AI>python generate.py --model=tf_04232023_e0 --temps=1 --live_input
Expecting a midi input...
0 / 1 sequences encoded
sampling temp=1.0
generating sequence
139 notes missing an end-time...
1 had an end-time precede their start-time
generating sequence
134 notes missing an end-time...
1 had an end-time precede their start-time
generating sequence
132 notes missing an end-time...
2 had an end-time precede their start-time
```



Sample generated with temperature as 1 and primed with input midi



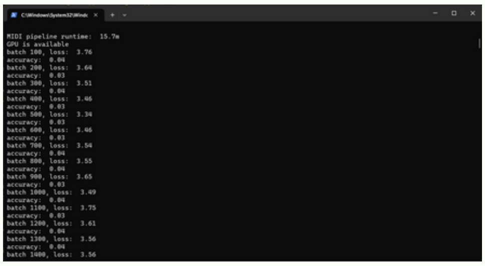
```
D:\anaconda3\envs\tf-gpu-cuda8\python.exe: can't open file 'C:\Users\VATSAL JIYA\run.py': [Errno 2] No such file or dire
ctory
(tf-gpu-cuda8) PS C:\Users\VATSAL JIYA> cd Music_with_ai
(tf-gpu-cuda8) PS C:\Users\VATSAL JIYA\Music_with_ai> python run.py --checkpoint=saved_models/tf_04032023_e4
Successfully loaded checkpoint at saved_models/tf_04032023_e4
Parsing 132 midi files in C:\Users\VATSAL JIYA\Music_with_ai\data\maestro-v3.0.0\2004...
Parsing 115 midi files in C:\Users\VATSAL JIYA\Music_with_ai\data\maestro-v3.0.0\2006...
Parsing 147 midi files in C:\Users\VATSAL JIYA\Music_with_ai\data\maestro-v3.0.0\2008...
Parsing 125 midi files in C:\Users\VATSAL JIYA\Music_with_ai\data\maestro-v3.0.0\2009...
Parsing 163 midi files in C:\Users\VATSAL JIYA\Music_with_ai\data\maestro-v3.0.0\2011...
Parsing 127 midi files in C:\Users\VATSAL JIYA\Music_with_ai\data\maestro-v3.0.0\2013...
Parsing 105 midi files in C:\Users\VATSAL JIYA\Music_with_ai\data\maestro-v3.0.0\2014...
Parsing 129 midi files in C:\Users\VATSAL JIYA\Music_with_ai\data\maestro-v3.0.0\2015...
Parsing 140 midi files in C:\Users\VATSAL JIYA\Music_with_ai\data\maestro-v3.0.0\2017...
Parsing 93 midi files in C:\Users\VATSAL JIYA\Music_with_ai\data\maestro-v3.0.0\2018...
Parsing 0 midi files in C:\Users\VATSAL JIYA\Music_with_ai\data\maestro-v3.0.0...
Parsing 0 midi files in C:\Users\VATSAL JIYA\Music_with_ai\data...

1276 midis read, or 11939.4 minutes of music
***Unexpected note sequence...possible duplicate?
***Unexpected note sequence...possible duplicate?
***Unexpected note sequence...possible duplicate?
***Unexpected note sequence...possible duplicate?
***Unexpected note sequence...possible duplicate?
***Unexpected note sequence...possible duplicate?
1276 note sequences extracted

Processing training data...
1,148 note sequences
3,444 stretched note sequences
```

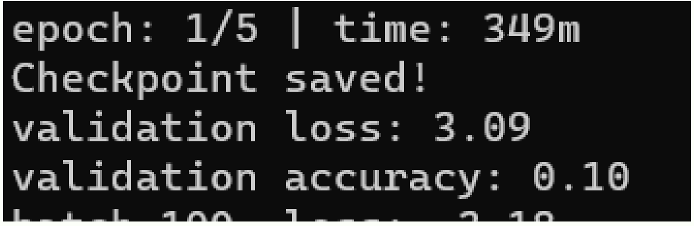
Parsing Training Dataset

Training in batches of size 4



```
MIDI pipeline runtime:  15.7m
GPU is available
batch 100, loss:  3.76
accuracy:  0.04
batch 200, loss:  3.64
accuracy:  0.03
batch 300, loss:  3.51
accuracy:  0.04
batch 400, loss:  3.46
accuracy:  0.03
batch 500, loss:  3.34
accuracy:  0.03
batch 600, loss:  3.46
accuracy:  0.03
batch 700, loss:  3.54
accuracy:  0.04
batch 800, loss:  3.55
accuracy:  0.04
batch 900, loss:  3.65
accuracy:  0.03
batch 1000, loss:  3.49
accuracy:  0.04
batch 1100, loss:  3.75
accuracy:  0.03
batch 1200, loss:  3.61
accuracy:  0.04
batch 1300, loss:  3.56
accuracy:  0.04
batch 1400, loss:  3.56
```

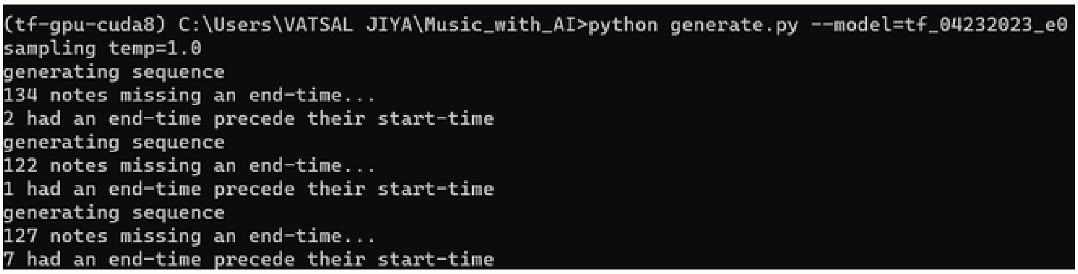
Training time taken, and model checkpoint



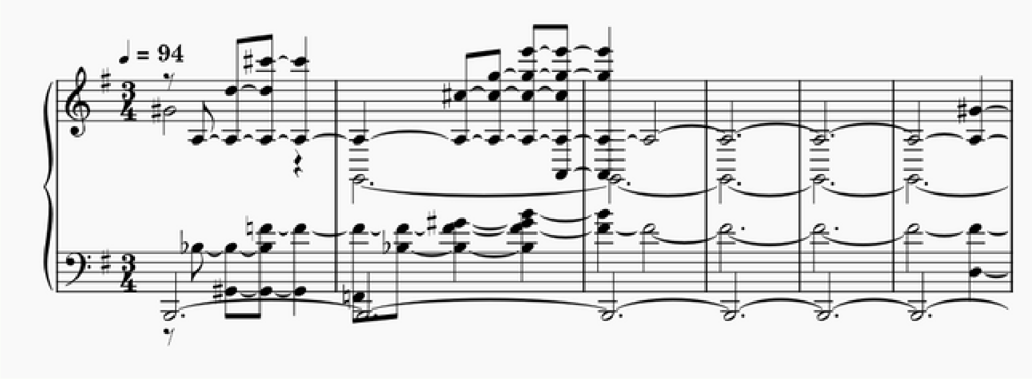
```
epoch: 1/5 | time: 349m
Checkpoint saved!
validation loss: 3.09
validation accuracy: 0.10
batch 100, loss: 3.10
```

IMPLEMENTATION DETAILS

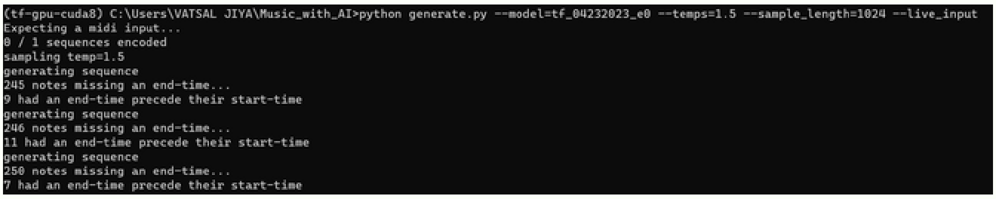
Sample with temperature 1 and not primed



```
(tf-gpu-cuda8) C:\Users\VATSAL JIYA\Music_with_AI>python generate.py --model=tf_04232023_e0
sampling temp=1.0
generating sequence
134 notes missing an end-time...
2 had an end-time precede their start-time
generating sequence
122 notes missing an end-time...
1 had an end-time precede their start-time
generating sequence
127 notes missing an end-time...
7 had an end-time precede their start-time
```

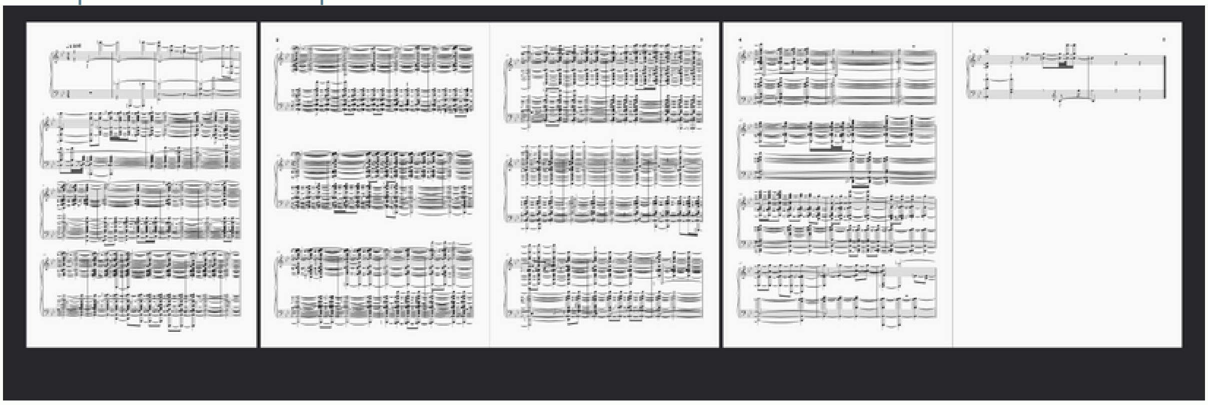


Sampling with larger music length and higher temp



```
(tf-gpu-cuda8) C:\Users\VATSAL JIYA\Music_with_AI>python generate.py --model=tf_04232023_e0 --temps=1.5 --sample_length=1024 --live_input
Expecting a midi input...
0 / 1 sequences encoded
sampling temp=1.5
generating sequence
245 notes missing an end-time...
9 had an end-time precede their start-time
generating sequence
246 notes missing an end-time...
11 had an end-time precede their start-time
generating sequence
250 notes missing an end-time...
7 had an end-time precede their start-time
```

Complete Music output



00

TECH STACK
API USED

pretty_midi 0.2.10

Pretty MIDI is a Python library used for handling and manipulating MIDI files. It allows you to parse, modify, and create MIDI files with Python code.

python™

To code the model

PyTorch

PyTorch is a popular deep learning framework used for implementing machine learning algorithms, including those for music generation using AI.

musescore

Musescore is a music notation software used for composing and arranging music.

CONCLUSION & FURTHER ENHANCEMENT

Music generation using AI is an exciting and challenging research area that has attracted significant attention in recent years. There are several constraints and challenges that need to be considered when developing music generation models, including data availability, computational resources, musical knowledge, evaluation metrics, copyright issues, user interface, training time, model capacity, overfitting, evaluation metrics, and interpretability.

However, by leveraging powerful tools such as PrettyMIDI, MuseScore, and PyTorch, it is possible to overcome these challenges and develop high-quality music generation models that can create new and inspiring musical compositions. The transformer model, in particular, has shown great promise in this field and has been used to generate music in a variety of styles and genres.

Music generation using AI has the potential to revolutionize the music industry by enabling musicians and composers to explore new creative possibilities and generate novel and unique musical compositions. It is an exciting field that is still in its early stages, and there is much room for innovation and experimentation. With continued research and development, we can expect to see even more impressive results in the future.

FURTHER ENHANCEMENT

Here are some potential avenues for further work on the problem statement and scope of work:

1. Style Transfer: You can explore the possibility of transferring the style of a given musical piece to generate new pieces with the same style but different melodies or rhythms. This can be achieved by training a style transfer model using a combination of music generation and style transfer techniques.
2. Multi-Instrument Music Generation: You can extend your project to generate multi-instrument music by incorporating multiple instrument tracks into the model. This can be achieved by using a variant of the transformer model, such as the Music Transformer-XL, that is designed for generating multi-track music.
3. User Interface: You can develop a user interface that allows users to interact with your music generation model and customize the generated music according to their preferences. This can involve developing a web application or mobile application that provides a simple and intuitive interface for users to input their preferences and generate music in real-time.
4. Reinforcement Learning: You can explore the use of reinforcement learning to train your music generation model. Reinforcement learning can enable your model to learn from feedback received from the environment, which can help it to generate more realistic and musically pleasing sequences.
5. Domain Adaptation: You can investigate the use of domain adaptation techniques to improve the generalization performance of your music generation model. Domain adaptation can help your model to generalize to different styles or genres of music, which can be useful for generating music in novel or unexplored styles.

REFERENCES

[1] Fine-Tuning Music Generation with Reinforcement Learning Based on Transformer(2022) [link]

[2] Pop Music Transformer: Beat-based Modeling and Generation of Expressive Pop Piano Compositions (2020) [link]

[3] MUSIC TRANSFORMER: GENERATING MUSIC WITH LONG-TERM STRUCTURE(2018) [link]

[4] A Study on the Application of Deep Learning Algorithm and Computer Neural Networks in Automatic Generation According to User Style (2021) DOI: 10.1109/ICCASIT53235.2021.9633546[link]

[5] Music Deep Learning: A Survey on Deep Learning Methods for Music Processing.(2022) DOI: 10.1109/MOCAST54814.2022.9837541[link]

[6] Towards Music Generation With Deep Learning Algorithms (2018) [link]