Multi-Genre Symbolic Music Generation using Deep Convolutional Generative Adversarial Network

Chris Chauhan[1]*, *Bhavesh* Tanawala[1] and *Mosin* Hasan[1]

[1]BVM Engineering College, Dept. of Computer Engineering, V.V. Nagar, 388120, India

Abstract. Music is an art that uses sound to convey emotions and ideas. It is a universal language that transcends cultural boundaries and can move and inspire individuals of all ages and cultures. As with every art form, the generation of music is a complex and challenging task. Despite the challenges, music generation has made considerable strides in recent years, owing to the application of artificial intelligence and machine learning. However, most research was focused on the generation of only one genre of music, i.e., classical, jazz, etc., while there are more than 40 genres of music, each with sub-genres. This paper proposes a model for multi-genre music generation using Generative Adversarial Networks (GAN). Considering symbolic music, MIDI tracks were converted into piano-roll form after the extraction of musical information. Subsequently, a GAN based model was trained to learn the distribution of training data, and it generates new data using the learned parameters. Generated music was evaluated based on a survey of musicians and professionals. The survey results validate the GAN's ability to generate music of multiple genres.

1 Introduction

Music generation is the process of making new music using computational techniques such as machine learning and algorithms. This field has a long history, and it has been developed for more than 60 years. During this period, a large number of researchers utilised different methods, including but not limited to grammar rules, probability models, evolutionary computation, and neural networks, and carried out numerous researches on various datasets aiming at different generation tasks [1]. Deep learning, a subfield of machine learning, provides methods that are able to recognise the internal patterns and correlations of data very well; furthermore, the algorithms are able to produce valuable insights without being explicitly programmed. Due to these properties, out of all the methods, deep learning has caught the attention of most researchers, and in the past decade, music generation using deep learning has made rapid progress.

*Corresponding author: chrischauhan77@gmail.com

It is worth noting that previous research on MIDI music generation has primarily focused on well-known genres such as classical, jazz, and hip-hop [2, 3]. As a result, the potential of MIDI music generation for creating music across a broader range of genres has not been fully explored or realized. In music, a genre refers to a category or style of music that shares common characteristics, such as musical form, instrumentation, rhythm, melody, and lyrics. According to the Music Genre List, there are 41 primary genres of music, and within those primary categories, there are still subcategories of music [4].

This paper is about designing a MIDI music generation system that generates five genres of music, i.e., Trap, Rhythm & Blue, Slap house, Trance & Techno and Latin. To make the generated music more expressive and less mechanical. Velocity, also known as music dynamics, is incorporated in the data [5]. The data is depicted in the form of a piano-roll representation that involves a dual-axis system. One axis corresponds to the passage of time, while the other indicates the pitch of the notes being played over time. This allows for a visual representation of the music data that can be easily analysed and manipulated by software programs and scripts. GAN has multiple variants, like Deep Convolutional Generative Adversarial Network (DCGAN), Progressive GAN (ProGAN), StyleGAN, and many more [6-10]. We consider DCGAN as our GAN architecture so that the data representation, pre-processing, and post-processing remain simple. DCGAN has the capability of capturing and understanding the underlying patterns and structures of a given dataset, allowing it to learn the distribution of the data. Once it has learned this distribution, it can then generate new data samples that are similar in style or structure to the original dataset. That implies that the algorithm can generate new music that is consistent with the patterns and structures present in the existing music dataset. After generating new music using the proposed model, these samples were evaluated and assessed through a user survey.

2 Related Work in Music Generation

2.1 Research status

Various deep learning architectures, including simple feed-forward neural networks and more complex approaches like generative adversarial networks, have been used in music generation systems. These architectures differ in their design and complexity and offer different benefits and drawbacks depending on the specific needs and goals of the music generation system.

Sequence models are a class of machine learning models that are designed to analyse and generate sequences of data, such as text, music, or speech. These models are often used in natural language processing, speech recognition, and music generation tasks where the input and output data are sequential in nature. Due to their sequential nature, models like Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM) are used in this task, though they are often combined to overcome the shortcomings of solely RNN based architectures [11, 12].

Authors in [13] proposed an RNN-LSTM-based architecture to generate music; however, the shortcomings of this method are that the data is represented in ABC notation, which is a text based musical notation that lacks audio data and cannot be played directly without conversion to another format. It has limited expressive capabilities as it cannot represent change in tempo, dynamics, or articulation. The model generates monophonic music without velocity or harmony, which results in dull and mechanical sounding music.

Authors in [14] gave a model based on Progressive GAN. ProGAN was originally designed to generate high-resolution images with high visual fidelity by gradually increasing the resolution of the generated images over the course of training. The progressive growth approach used here allows the generator to focus on learning coarse details in the early stages of training and then gradually refine those details as the resolution of the generated samples increases. Training data was down sampled so it could work with ProGAN architecture, though down sampling crippled the network's creativity to a certain extent. The generated results were sometimes categorised as human-made by participants in the user survey, but music lacks dynamics and is limited to a single genre.

Recently, Google Research proposed MusicLM, a model for generating high-fidelity music from text descriptions [15]. It approaches the task of music generation as a hierarchical sequence to sequence modelling tasks, and it uses three pre-trained models. The pre-trained models are: Soundstream a neural audio codec, i.e., an audio compressor, which takes a waveform and compresses it at a lower bitrate while still maintaining high reconstruction quality up to 24 kHz [16]. W2v-BERT a transformer based BERT framework for audio that takes care of semantic information and models long term structure [17]. MuLan a model developed by Google Research that does music-text joint embedding [18]. Collectively, once all three models are trained, text input is passed to MuLan, and the resulting tokens are decoded through Soundstream’s decoder, which then synthesises the output music. The perception of the generated music was similar to live performance music, but the long term structure of the music was not convincing.

2.2 Introduction to GAN

GAN is a deep learning model based on generative modelling. The GAN architecture consists of two neural networks: a generator and a discriminator. The task of the generator is to create new data samples, while the discriminator works as a binary classifier and tries to distinguish between the generated samples and real samples from the training dataset. Through an iterative process of playing a zero-sum game, the generator learns to create more realistic data samples, while the discriminator becomes better at distinguishing between generated and real data. GANs are being used for a variety of applications, including image synthesis, natural language processing, and music generation.

3 Methodology

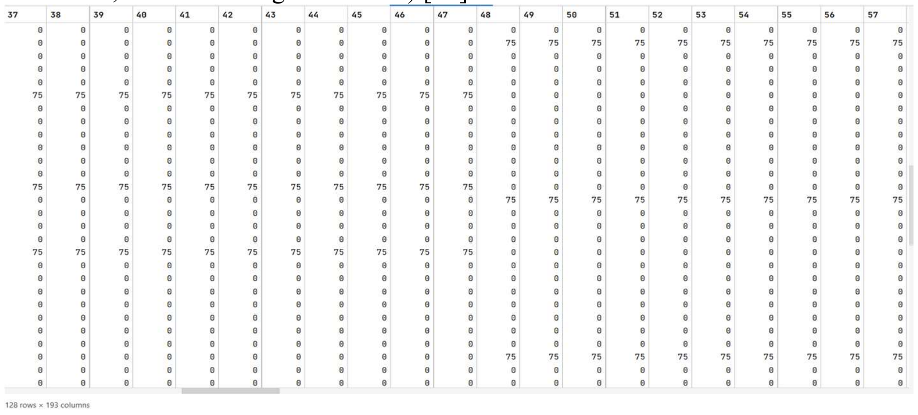
In this paper, we present a music generation system that utilizes DCGAN to produce polyphonic MIDI music in multiple genres. The proposed system leverages the ability of DCGAN to learn from the training data distribution and generate novel data based on the learned parameters. To evaluate the system’s effectiveness in generating music, a survey was conducted to demonstrate and validate the model’s capability to produce music across five genres.

3.1 Dataset

The Niko Ultimate Dance MIDI Pack and the Cymatics Pandora MIDI Collection both contain MIDI tracks of various genres [19, 20]. We consider five genres for our model: Trap, R&B, Slap house, Trance & Techno and Latin. For each genre we collected 100 MIDI tracks. Collectively, the entire training dataset consists of 500 MIDI tracks.

3.2. Data representation and visualization

We used piano-roll representation for data representation. Piano-roll is used to represent the data in matrix form, which can be easily interpreted by the model. The height of the matrix represents pitch, while the width represents time. Fig. 1 shows the matrix of shape (128, 192), where 128 represents pitch from 0-127 and 192 is time in beats. The values in the matrix represent the velocity of a note. Velocity indicates how hard the key was struck when the note was played, which usually corresponds to the note's loudness (the MIDI velocity range is from 0-127, with 127 being the loudest) [21].



37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	75	75	75	75	75	75	75	75	75	75
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
75	75	75	75	75	75	75	75	75	75	75	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
75	75	75	75	75	75	75	75	75	75	75	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	75	75	75	75	75	75	75	75	75	75
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
75	75	75	75	75	75	75	75	75	75	75	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	75	75	75	75	75	75	75	75	75	75
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

128 rows × 193 columns

Fig. 1. Piano-roll representation

For data manipulation, visualisation, and analysis, we used an open-source Python library for music generation called MusPy [22]. Fig. 2 shows the visualised piano roll using the library method; instead of a pitch range of 0-127, it shows C2 to C8 octaves. Blue marks represent the presence of a particular note at a given time.

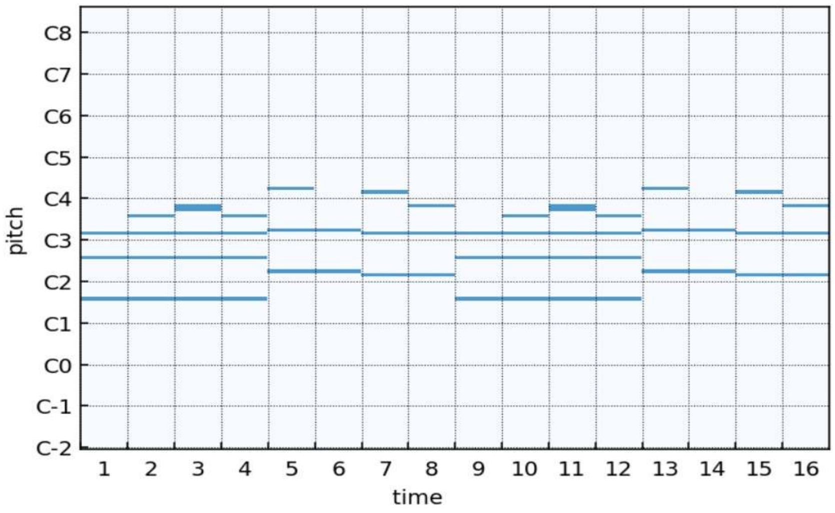


Fig. 2. Piano-roll visualization

4 Implementation

4.1 Pre-processing of data

MIDI files were first loaded into the program with the help of the MusPy library and subsequently converted into music objects. The music objects underwent pre-processing as described below:

- For optimum representation of music in the temporal dimension, we set the temporal resolution to 12 time steps per beat.
- Velocity values are constrained to at least 64, as below that, we found notes are inaudible.
- To generalise the data, we transpose all MIDI files to C major or its relative A minor key. Due to this, the notes are confined to a particular range of pitch.
- As all MIDI files don't have the same length, we truncated the music length to 8 seconds.
- Given the range of 0-127 of pitch in MIDI, not all pitches will be useful in our music data, so we only consider 4 octaves of pitch, i.e., 48 notes. Using the CV2 library, we reshape the data into the shape of (M, 48, 48, 1), i.e., where M is the number of training examples.

We applied the above process to each genre of music and then stacked the training examples of five genres of music together on axis 3, which results in a data shape of (M, 48, 48, 5). Fig. 3 shows a snapshot of a single training example having the shape of (48, 48, 5).

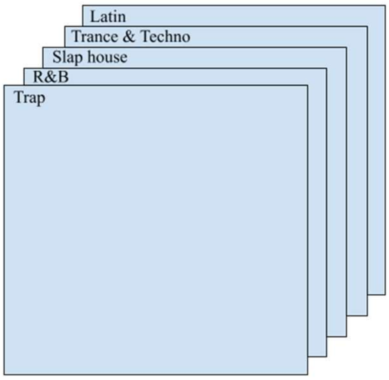


Fig. 3. Single training example

4.2 Generator architecture

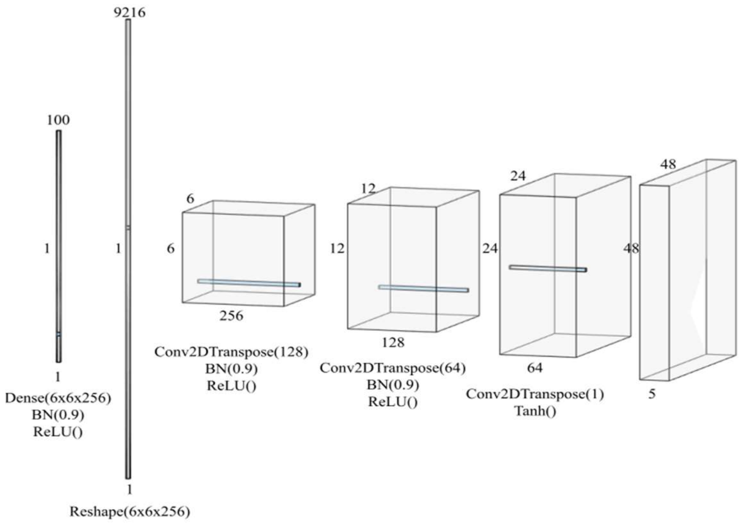


Fig. 4. Generator architecture

As shown in Fig. 4, the generator takes a noise vector of shape (100,) as input and passes it through a Dense layer having 9216 hidden units. Output of this is reshaped into shape (6, 6, 256). The model uses 3x3 kernels to perform the transpose convolution operations and thereby control the channels across the generator. The intermediate weights of the network are normalised using Batch Normalization with a momentum rate of 0.9, followed by the ReLU activation function to ensure non-linearity. The final layer uses the Tanh function and outputs a tensor of shape (48, 48, 5).

4.3 Discriminator architecture

The discriminator's network arrangement is opposite that of the generator's. It takes an input of a piano-roll data sample, i.e., a shape of (48, 48, 5), and passes it through the network; it uses kernels of 3x3 for convolution operation followed by LeakyReLU and Dropout layer for regularization. Towards the end, the network is flattened and undergoes three Dense layers. It then uses a Sigmoid activation function to generate the final predictions, i.e., fake or real data. These predictions are considered feedback for the generator to improve, and collectively, the model will learn to improve itself.

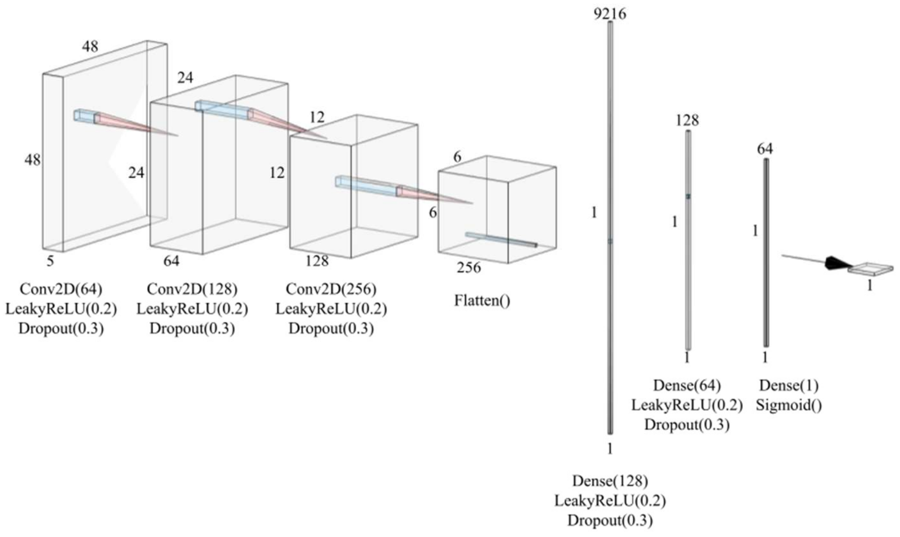


Fig. 5. Discriminator architecture

4.4 Training

We used Kaggle's online notebook with Nvidia Tesla T4 x2 GPUs as accelerators for training the model [23]. The Adam optimizer was used for training both the generator and discriminator, with a learning rate of 5×10^{-6}. The generator's task was more challenging than the discriminator's task, so in order to achieve Nash equilibrium, the generator was updated five times more frequently than the discriminator. The model was trained for 200,000 epochs.

4.5 Data post-processing and MIDI generation

The generated output from the generator is first padded to shape (128, 48, 5) before being converted into a music object using the MusPy library. Subsequently, the resulting piano roll is written into five MIDI tracks, each corresponding to a different genre.

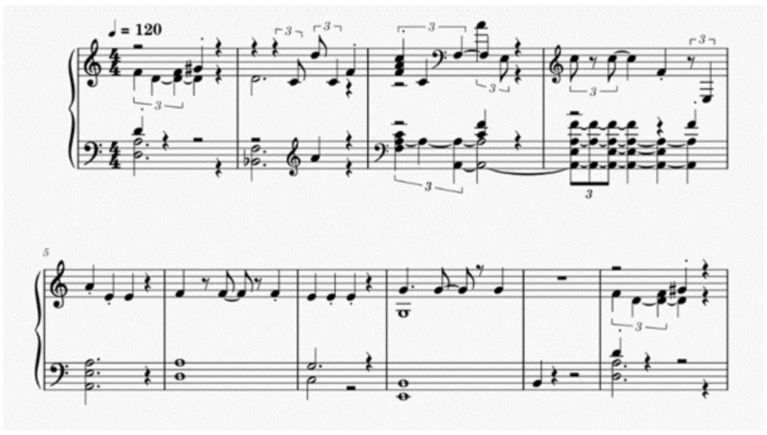


Fig. 6. Generated music score

5 Evaluation

Music evaluation is a subjective process that is influenced by personal taste, cultural background, and musical understanding, and it varies from listener to listener. Hence, to evaluate, five samples were generated, one for each genre by the model, and then, using MuseScore 3, we synthesised the generated samples. A user survey was carried out to verify if the generated samples from the model have the expressive elements required for a genre.

Table 1. Generated Sample Links

Generated Samples	Link
Slap House	https://tinyurl.com/fdu356zf
Trap	https://tinyurl.com/fzt46dzp
Rhythm & Blues	https://tinyurl.com/y38mm6yc
Latin	https://tinyurl.com/bt87k37s
Trance & Techno	https://tinyurl.com/yeynfvx3

The survey was designed to evaluate the musical and emotional aspects of the generated music and to assess the participant's ability to recognize the specific musical elements associated with a particular genre. Participants were all music professionals, including a music teacher, composer, mixing and mastering engineer, and music director. 77.8% of participants had perfect pitch, while the remaining 22.2% had relative pitch. Also, all the participants had experience playing an instrument. The survey was organised online using Google Forms, where participants were provided with generated samples and asked the questions given in Table 2 for each sample.

Table 2. User survey questions

Sr. no	Questions
1	Was the piece emotionally engaging or expressive?
2	On a scale of 1-5, how would you rate the musicality of this piece?
3	On a scale of 1-5, how would you categorize the musical genre of this piece as particular genre?
4	On a scale of 1-5, how melodious would you rate this piece?
5	On a scale of 1-5, how harmonious would you rate this piece?
6	On a scale of 1-5, how confident are you that this piece was composed and performed by a human?
7	What do you like or dislike about this piece?

Question 2 to 6 were linear scale questions, scaled from 1 to 5, with 1 being very bad and 5 being excellent. In Question 3, for every sample we provide its genre and ask the participants, Does this sample align with the given genre? E.g., for Sample 1, participants would be asked, on a scale of 1 to 5, how would he/she categorise the musical genre of the piece as Slap House?

Samples rated as being of a particular genre are shown Fig. 7, we found that the average rating for each sample belonging to a given genre is about 3.7 out of 5. This suggests that the samples have elements of the required genres and that the model is able to produce the multi-genre music with an interesting progression and melodic sequence.

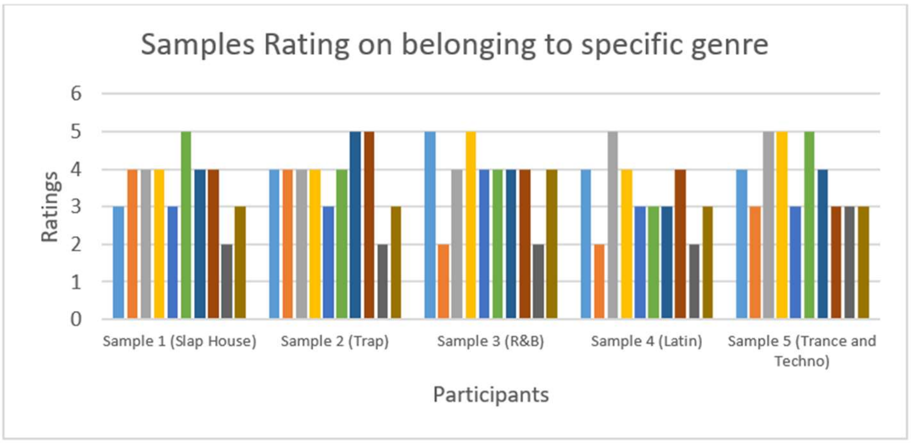


Fig. 7. User survey results

Similarly, most of the samples were engaging and expressive in nature for their respective genres, as shown in Fig. 8, and this also led to the conclusion that the generated music was very similar to human-performed music. However, Sample 4 (Latin) received lower scores in terms of expressiveness from the survey participants.

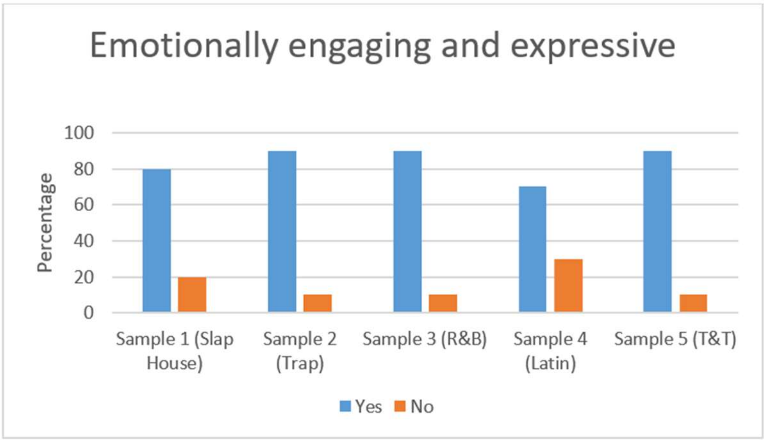


Fig. 8. User survey expressiveness results

User comments on the samples are found in Table 3. They suggest that samples had good melody structure and rhythm; we also noted that participants described some samples like 3 and 4 as ‘soft and melodic, as required in the genre. Though the progression is good, the flow seems to be inconsistent in samples 2 and 4. Overall, the responses collected from the survey suggest that the proposed music generation system has the ability to generate multi-genre music that is both engaging and expressive.

Table 3. User comments

Sample	Like	Dislike
1	"While the progressions seems interesting, the performance gets in the way"	"Single handed performance was good, but sequence of lower notes would add up to harmony"
2	"I like the harmony in this piece"	"Flow of the piece is inconsistent"
3	"Melody and harmony goes really well, great use of softer notes"	"It sounds like a student trying to learn piano"
4	"Melody is expressive and feels surreal"	"Doesn't flow, wrong notes played in the melody line against the chords"
5	"Interesting use of staccato notes"	"Simplistic progression and while the flow is better, still doesn't sound natural or well-rehearsed"

6 Conclusion

In this paper, a multi-genre music generation system was implemented using the DCGAN architecture. The generated samples were analysed, and a user survey was conducted to validate the results. The evaluation of the generated samples proves that the proposed system can generate multiple genres of expressive and engaging music. Overall, the proposed multi-genre music generation system presents a promising direction for the development of artificial intelligence based music creation tools that can assist and inspire musicians and composers in their creative processes. However, work is still required in areas such as arbitrary-length music generation and improvement in the harmony and flow of the music. In the future, we would like to work on a transformer-based architecture for solving the problem of music generation of different genres.

References

1. S. Ji, J. Luo and X. Yang, *A Comprehensive Survey on Deep Music Generation: Multi-level Representations, Algorithms, Evaluations, and Future Directions*, arXiv preprint arXiv:2011.06801 (2020)
2. A. Maduskar, A. Ladukar, S. Gore and N. Patwari, *Music Generation using Deep Generative Modelling*, IEEE International Conference on Convergence to Digital World - Quo Vadis, ICCDW, 1-4 (2020)
3. P. Yadav, S. Khan, Y. Singh, P. Garg and R. Singh, *A Lightweight Deep Learning-Based Approach for Jazz Music Generation in MIDI Format*, Computational Intelligence and Neuroscience (2022)
4. "Music Genres List" https://www.musicgenreslist.com/ (2022)
5. R. K. H. Toh and A. Sourin, *Generation of Music With Dynamics Using Deep Convolutional Generative Adversarial Network*, 2021 International Conference on Cyberworlds (CW), 137-140 (2021)
6. I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville and Y. Bengio, *Generative Adversarial Networks*, Communications of the ACM, **63**(11), 139-144 (2020)

7. A. Radford, L. Metz and S. Chintala, *Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks*, arXiv preprint arXiv:1511.06434 (2015)
8. T. Karras, T. Aila, S. Laine and J. Lehtinen, *Progressive Growing of GANs for Improved Quality, Stability, and Variation*, arXiv preprint arXiv:1710.10196 (2017)
9. T. Karras, S. Laine and T. Aila, *A Style-Based Generator Architecture for Generative Adversarial Networks*, Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 4401-4410 (2019)
10. M. Arjovsky, S. Chintala and L. Bottou, *Wasserstein GAN, International conference on machine learning*, International conference on machine learning, PMLR, 214-223 (2017)
11. L. Zhang, S. Wang and B. Liu, *Deep Learning for Sentiment Analysis: A Survey*, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, **8**(4), e1253 (2018)
12. J. Dai, S. Liang, W. Xue, C. Ni and W.-J. Liu, *Long short-term memory recurrent neural network based segment features for music genre classification*, 10th International Symposium on Chinese Spoken Language Processing (ISCSLP), IEEE, 1-5 (2016)
13. S. Sajad, S. Dharshika and M. Meleet, *Music Generation for Novices Using Recurrent Neural Network*, 2021 International Conference on Innovative Computing, Intelligent Communcation and Smart Electrical Systems (ICSES), IEEE, 1-6 (2021)
14. S. Walter, G. Mougeot, Y. Sun, L. Jiang, K.-M. Chao and H. Cai, *MidiPGAN: A Progressive GAN Approach to MIDI Generation*, IEEE 24th International Conference on Computer Supported Cooperative Work in Design (CSCWD), IEEE, 1166-1171 (2021)
15. A. Agostinelli, T. Denk, Z. Borsos, J. Engel, M. Verzett, A. C. Q. Huang, A. Jansen, A. Roberts, M. Tagliasacchi, M. Sharifi, N. Zeghidour and C. Frank, *MusicLM: Generating Music from Text*, arXiv preprint arXiv: 2301.11325 (2023)
16. N. Zeghidour, A. Luebs, A. Omran, J. Skoglund and M. Tagliasacchi, *SoundStream: An End-to-End Neural Audio Codec*, IEEE/ACM Transactions on Audio, Speech, and Language Processing, **30**, 495-507 (2021)
17. Y.-A. Chung, Y. Zhang, W. Han, C.-C. Chiu, J. Qin, R. Pang and Y. Wu, *W2v-BERT: Combining Contrastive Learning and Masked Language Modeling for Self-Supervised Speech Pre-Training*, IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), IEEE, 244-250 (2021)
18. Q. Huang, A. Jansen, J. Lee, R. Ganti, J. Y. Li and D. P. W. Ellis, *MuLan: A Joint Embedding of Music Audio and Natural Language*, arXiv preprint arXiv: 2208.12415 (2022)
19. "Pianoforproducers" https://www.pianoforproducers.com/ (2022)
20. "Cymatics," https://cymatics.fm/ (2022)
21. "Logic Pro User Guide" https://support.apple.com/enin/guide/logicpro/lgcpa8fee137/mac (2023)
22. H.-W. Dong, K. Chen, J. McAuley and T. Berg-Kirkpatrick, MusPy: A Toolkit for Symbolic Music Generation, Proceedings of the 21st International Society for Music Information Retrieval Conference (ISMIR) (2020)
23. "Kaggle" https://www.kaggle.com/ (2023)