Single-to-Multi Music Track Composition Using Interactive Chaotic Evolution

Ying Kai Hung[1], Yan Pei[1]*, and Jianqiang Li[2]

[1] Graduate School of Computer Science and Engineering, University of Aizu,
Aizuwakamatsu, Fukushima, 965-8580, Japan
m5251111@u-aizu.ac.jp; peiyan@u-aizu.ac.jp
[2] Faculty of Information Technology, Beijing University of Technology
Beijing, 100124, China
lijianqiang@bjut.edu.cn

Abstract. This research presents a new music generation model and a novel MIDI data format for MIDI music generation. This innovative data format allows us to process MIDI music in a manner analogous to video analysis. Initially, the model employs Convolutional Neural Networks (C-NN) as an encoder to effectively capture local and global features within the musical data. Subsequently, we utilize a Transformer as a decoder, leveraging its self-attention mechanism to handle the long-term dependencies present in music data. In the training process, an interactive chaotic algorithm is introduced to update the model's weights, assisting the model in avoiding entrapment in local optima. This enhances the learning efficiency of the model and improves the quality of the generated output, enabling the model to generate music, including accompaniment, that aligns with human aesthetics from any given melody.

Keywords: Music Composition, MIDI, Convolutional Neural Networks, Transformer, Interactive Evolutionary Computation, Interactive Chaotic Evolution

1 Introduction

In recent years, deep learning has been extensively applied in music-related research, leading to advancements in tasks such as genre recognition, song index recommendation, and music style transformation. These tasks, which previously required significant time and music theory experience, can now be expedited through the application of deep learning, reducing the associated time costs. When it comes to creating music on a computer, the most convenient data format for creation and recording is the Musical Instrument Digital Interface (MIDI) music information. MIDI records pitch, music intensity, volume, and instrument timing as a digital signal, and deep learning models are commonly used to process MIDI data for various tasks that involve handling long sequences. However, a challenge arises due to the extensive length of the data, making it

* Dr. Yan Pei is the corresponding author, peiyan@u-aizu.ac.jp

difficult to effectively correlate all units and resulting in outcomes that do not align with human aesthetics.

In other digital signal domains, techniques such as MFCC or Mel spectrograms have been used to transform signal data into continuous images, effectively capturing audio features for tasks like voice conversion and voice recognition. Building on this knowledge, we propose a new music generation model and a novel method to modify the MIDI data format for MIDI music generation in this study. We convert each piece of MIDI format music into a piano roll, segment it according to the music beat, and stack each segment as a frame in a 3D data format. This allows us to process the data in a manner analogous to video analysis. The model employs a Convolutional Neural Network (CNN) as an encoder to effectively capture the global features of each frame. Subsequently, we utilize a Transformer as a decoder, leveraging its self-attention mechanism to handle the long-term dependencies in the data.

This design provides our model with high flexibility and adaptability, enabling it to efficiently generate complete music with complex structures, including accompaniment and melody. In addition to the model and data design, an interactive chaotic algorithm is introduced during the training process to update the model's weights. This algorithm simulates chaotic phenomena in nature, allowing the model to self-organize and self-adjust during the learning process, thereby generating more creative and aesthetically pleasing music. Moreover, the interactive chaotic algorithm helps the model avoid local optima, enhancing its learning efficiency and the quality of the generated music, resulting in compositions that better align with human aesthetics.

2 Related Works

Reference [1] is a study that explores the impact of different music input representations on the performance of Convolutional Neural Network (CNN) music classification models. In this paper, the researchers compared three common music input representations: Mel spectrograms, spectrograms, and constant-Q transforms. They found that all input representations could be effectively used by the CNN model. My research converted MIDI data into piano roll images and used a CNN as an encoder to compress features based on the data nature. Our approach shares some similarities with the method proposed in this paper, as we both aim to find an effective way of visualizing music as images and improve the model's ability to extract musical features.

Reference [2]: This research paper, written by Jean-Baptiste Alayrac and others, mainly discusses the method of using Transformer networks to handle video information. In this paper, the researchers proposed a new video understanding model based on the Transformer network that can directly handle raw video frames and capture long-term dependencies in the video. Their model uses a self-attention mechanism to comprehend the contextual information in the video and can automatically learn the dynamic and static features in the video. Inspired by this paper, I conceived a new data representation method that converts MIDI music into piano rolls and frames them into a series of frames, forming a three-dimensional data format similar to video. Next, we designed a model that uses a convolutional neural network (CNN) as an encoder to capture global features in each timestep, and a Transformer as a decoder to utilize its self-attention mechanism to handle long-term dependencies in the data and achieve the task of generating music.

Reference [3] is a research paper published in 2017 by Cheng-Zhi Anna Huang and others. The paper explores the use of Convolutional Neural Networks (C-NN) in music generation, specifically in generating counterpoint-style music. The researchers employed deep learning techniques, particularly Convolutional Neural Networks (CNNs), to construct a model capable of generating music in the style of counterpoint. Their model can learn and imitate the rules of counterpoint-style music and generate new counterpoint-style melodies. Their research demonstrates that deep learning techniques can be effectively applied to the field of music composition, resulting in artistic and innovative music. A part of my research model also utilizes a Convolutional Neural Network (CN-N) as the encoder, although there are differences in the way the input captures music features and the specific model architecture. However, both approaches aim to effectively capture global features in musical data and generate complete music with accompaniment from a single melody input.

Reference [4] is a research paper published by the Google Magenta team in 2019. This groundbreaking work introduced the Transformer architecture to the field of music generation, successfully addressing the long-term structural issues in MIDI music generation. The primary objective of Music Transformer is to handle the long-term dependency problem in music generation. To accomplish this, the researchers utilized the Transformer, a deep learning model with a powerful self-attention mechanism. Building upon this, they developed a new MIDI event representation called "Relative Global Encoding." This encoding method not only captures the rhythmic structure in music but also considers the relative timing of notes, enabling the model to generate works with longer musical structures. My research shares many similarities with the work of "Music Transformer." Firstly, it also employs a Transformer-based model and utilizes a self-attention mechanism to address the long-term dependency problem in music data. However, our research introduces an innovative approach to process MIDI data, converting each MIDI music piece into a piano roll-like format and then framing and stacking it into a 3D data structure. This method effectively captures global music features and extracts longer temporal features.

Reference [5] is a technique proposed by Yan Pei that combines chaotic dynamics and evolutionary algorithms. The main idea is to guide evolutionary algorithms in a global search within the solution space of optimization problems by leveraging the randomness generated through chaotic mapping. This approach effectively prevents the optimization process from converging to local optima and enhances the quality and diversity of optimization results.In our research, we introduce this chaotic algorithm to update the model's weights and incorporate human evaluation into the training process to enable the model to self-organize and self-adjust based on human perception, resulting in the generation of more creative and aesthetically appealing music. Our model design combines the strengths of chaos theory, the capabilities of deep learning, and human aesthetic evaluations, thereby enhancing learning effectiveness and generation quality. The outcome is music that better aligns with human aesthetics.

3 Method

This study employed several methods to generate MIDI music that aligns with human aesthetics. These methods include converting MIDI music into a 3D piano roll image, utilizing CNN as an encoder, employing Transformer as a decoder, and training the model using an interactive chaotic algorithm.

To begin, MIDI music was transformed into a 3D matrix data structure in the piano roll image format, which mimics the format used by humans when creating music scores. This approach is not limited to any specific music genre or style, making it applicable to various types of MIDI music, such as classical or pop.

The use of CNN as an encoder offers the advantage of effectively capturing both local and global features of the piano roll, thereby transforming them into an input representation suitable for the Transformer decoder. This enhances the model's learning capability and improves the quality of the generated music.

By using the Transformer as a decoder and leveraging its self-attention mechanism, it is possible to address the challenge of long-term dependencies in music and generate music that is more musically coherent.

Lastly, the utilization of an interactive chaotic algorithm during training enhances the efficiency of the model's learning process and prevents it from getting trapped in local optima. Additionally, human perception can be leveraged to optimize the model and generate music of higher quality.

Overall, these methodologies, involving the conversion of MIDI music to a 3D piano roll image, the use of CNN and Transformer, and the incorporation of an interactive chaotic algorithm, contribute to the generation of music that is aesthetically pleasing and aligns with human preferences.

Here are the detailed methods:

3.1 Data Preparation

In this research, the data preparation process plays a crucial role in efficiently handling MIDI music data for subsequent learning and generation tasks. The first step involves converting the music data into a graphical representation using the piano roll format. The resulting image has a size of (128, n), where 128 represents a fixed pitch value in MIDI music, and n corresponds to the length of the MIDI reading time, determined by the chosen sampling rate. To ensure computational efficiency, a sampling rate of 0.1 seconds is used to read the MIDI data.

Next, the rhythmic duration is extracted from the MIDI music. The music sequences are then sliced into frames, with each frame comprising four beats, aligning with a single measure to maintain the musical structure. To standardize the varying lengths of the music, each frame is padded with zeros to a duration of four seconds, resulting in a uniform duration of five minutes for all music data. These transformed frames are stacked into a matrix of size 128x40x150. Additionally, the target data is adjusted to a consistent size of 128x3000 to ensure seamless usage by the model.

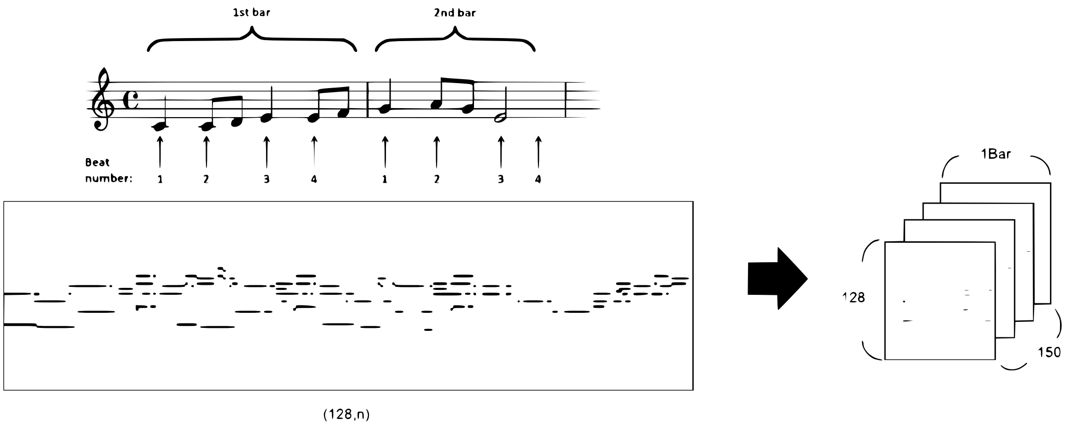

Fig. 1. Example of MIDI process

3.2 Model Architecture

The architecture of our model consists of an encoder and a decoder. The encoder utilizes a Convolutional Neural Network (CNN), while the decoder is based on the Transformer network. In the following sections, we provide a comprehensive description of each layer's structure and functionality within the model. Please refer to Fig 2 for a visual representation of the model architecture.

Encoder (Convolutional Neural Network) The encoder component of the model processes the input data through several layers, each performing specific transformations. The description of each layer is as follows:

- **Input Layer:** The model accepts an input of size (128, 40, 150), representing the transformed music data in a piano roll-like format.
- **Conv2D Layer:** The first convolutional layer applies 32 different filters to the input data, resulting in a feature map of size (128, 40, 32). Each filter focuses on detecting specific features in the input.
- **MaxPooling2D Layer:** The max-pooling layer reduces the spatial dimensions of the feature map to (64, 20, 32) while preserving important features. This step enhances computational efficiency and helps prevent overfitting.

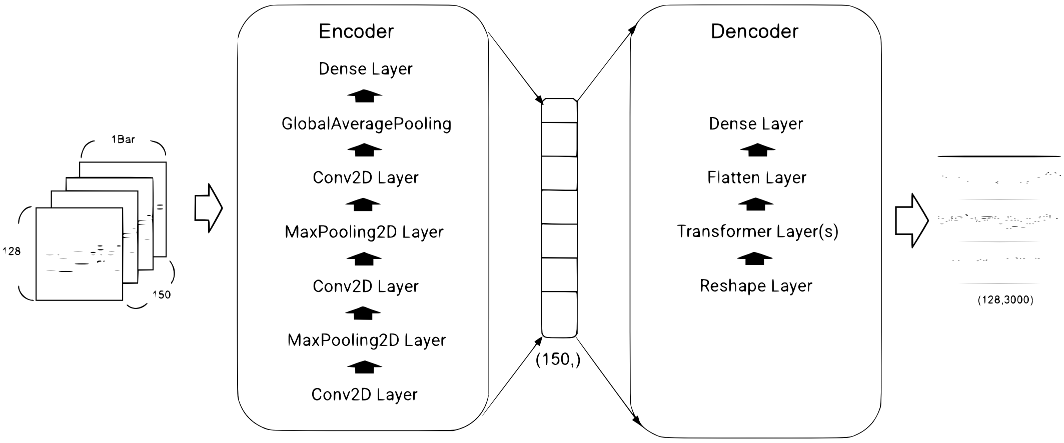

Fig. 2. Example of model architecture

- **Conv2D Layer:** The second convolutional layer applies 64 filters to the pooled feature map, producing a new feature map of size (62, 18, 64).
- **MaxPooling2D Layer:** Another max-pooling layer further reduces the s-patial dimensions of the feature map to (31, 9, 64).
- **Conv2D Layer:** The final convolutional layer applies 64 filters to the pooled feature map, generating a feature map of size (29, 7, 64).
- **GlobalAveragePooling2D Layer:** This layer computes the average value of each feature map, reducing the dimensions to (64,).
- **Dense Layer:** The dense layer (also known as a fully connected layer) takes the output from the previous layer and transforms it into a (150,) vector.

Decoder (Transformer) The decoder receives the output from the encoder and processes it through several layers:

- **Input Layer:** The decoder takes an input of size (150,).
- **Reshape Layer:** The input is reshaped into a 2D matrix of size (150, 1) to be compatible with the following Transformer layers.
- **Transformer Layer(s):** The Transformer layers take the reshaped input and transform it through a series of self-attention and feed-forward neural network layers. The output is a matrix of size (150, 64). The operations in the Transformer can be represented as follows:

$$\text{Self-Attention: Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{1}$$

$$\text{Feed-forward: FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2 \tag{2}$$

Here, Q, K, and V are the query, key, and value in the attention mechanism, respectively, and d_k is the dimension of the key. In the feed-forward network, W_1, b_1, W_2, and b_2 are the weights and biases of the two layers.

- **Flatten Layer:** The Flatten layer reshapes the 2D output matrix from the previous layer into a 1D vector of size (9600,).
- **Dense Layer:** The final dense layer transforms the flattened vector into the desired output shape of (128, 3000), which represents the generated music data in the piano roll format.

By combining Convolutional Neural Networks (used for feature extraction) and Transformers (used for sequence modeling), the architecture of this model enables it to effectively learn the features and structure of input music data, generate new music data while maintaining the input features.

3.3 Interactive Chaotic Evolution

In the process of model training, the optimizer plays a crucial role in determining the convergence rate and final performance of the model. Two key parameters of the optimizer are the learning rate and momentum, which have a significant impact on the training process. In this section, we propose an interactive chaotic evolution approach to optimize the learning rate and momentum of the ADAM optimizer using the logistic map.

The logistic map is a nonlinear dynamic system that exhibits chaotic behavior under certain conditions. It can be used to generate a sequence of pseudo-random numbers, which can then be mapped to a specific range to serve as the learning rate and momentum parameters for the ADAM optimizer. The overall algorithm for our interactive chaotic evolution approach is as follows:

1. Initialize the learning rate α and momentum β_1 and β_2 of the ADAM optimizer.
2. Train the model using the ADAM optimizer for a fixed number of iterations.
3. Check the loss function after each iteration. If the loss has not decreased significantly over the last k iterations, go to step 4. Otherwise, continue training using the current ADAM optimizer parameters.
4. Generate a sequence of pseudo-random numbers using the logistic map.
5. Map the pseudo-random numbers to a specific range to obtain the new values of the learning rate and momentum parameters for the ADAM optimizer.
6. Train the model for a fixed number of iterations using the new ADAM optimizer parameters.
7. Compare the performance of the model trained using the new ADAM optimizer parameters with that of the model trained using the previous ADAM optimizer parameters.
8. If the new model outperforms the previous one, update the ADAM optimizer parameters to the new values and continue training using the new parameters. Otherwise, continue training using the previous ADAM optimizer parameters.
9. Go back to step 3 and repeat until the model converges.

The logistic map used to generate the pseudo-random numbers is defined as follows:

$$x_{n+1} = rx_n(1 - x_n), \tag{3}$$

where r is the control parameter, x_n is the current value of the logistic map, and x_{n+1} is the next value of the logistic map. The value of r is typically set to a value between 3.6 and 4.0 to ensure that the map exhibits chaotic behavior.

The learning rate and momentum parameters for the ADAM optimizer are updated using the following equations:

$$\alpha_n = \frac{1}{1 + e^{-rx_n}}, \tag{4}$$

$$\beta_{1,n} = \frac{1}{1 + e^{-rx_{n+1}}}, \tag{5}$$

$$\beta_{2,n} = \frac{1}{1 + e^{-rx_{n+2}}}, \tag{6}$$

where α_n, $\beta_{1,n}$, and $\beta_{2,n}$ are the updated learning rate, momentum for the first moment estimate, and momentum for the second moment estimate at iteration n, respectively.

After adding chaotic algorithms, the update formulas for learning rate and momentum are as follows:

$$\alpha_{t+1} = \alpha_{\min} + (\alpha_{\max} - \alpha_{\min}) \times x_t \tag{7}$$

$$\beta_{t+1} = \beta_{\min} + (\beta_{\max} - \beta_{\min}) \times x_{t+1} \tag{8}$$

By using the interactive chaotic evolution approach, we can optimize the learning rate and momentum parameters of the ADAM optimizer in a more efficient manner, leading to better model performance and faster convergence.

4 Experiment and Evaluation

This experiment used three datasets for model training, including FreeMidi, Midi World, and the POP909 dataset curated by other researchers. There were two evaluation methods used: the first involved calculating the average distance between the model-generated audio and the original audio using the Euclidean distance method, while the second involved human evaluation to determine whether the generated audio was similar to the original audio.These evaluation methods are used to assess the performance of a model before and after the addition of interactive chaotic algorithms, to measure the expected improvements.

4.1 Dataset

This study used two self-collected MIDI music datasets, namely FreeMidi and Midi World, as well as the POP909 dataset compiled by others.

FreeMidi is a collection of 2,386 MIDI files of various genres and styles, including classical, jazz, pop, and rock. Each song is approximately 2-4 minutes long and consists of 64-80 measures. The total duration of the FreeMidi collection is approximately 112 hours.

Midi World is another collection of MIDI files, consisting of 2,857 songs in various genres such as classical, rock, jazz, and pop. Each song is also approximately 2-4 minutes long and contains 64-80 measures. The total duration of the Midi World collection is approximately 140 hours.

POP909 [6] is a collection of 909 MIDI files in the pop genre. Each song is approximately 2-5 minutes long and consists of 64-80 measures. The total duration of the POP909 collection is approximately 48 hours.Below are the detailed descriptions of the datasets used:

Table 1. datset information

Dataset	files	time(min)
FreeMIDI	2186	6218
MIDIworld	3658	8431
POP909	909	2982

4.2 Evaluation

In this evaluation, 20 music pieces generated by the model were assessed for their similarity using Euclidean distance, and their originality was evaluated by human judgment.

Table 2. The table shows the evaluation results

Dataset	Euclidean disrance	Human Evaluations
FreeMIDI	0.332	0.60
MIDIworld	0.401	0.65
POP909	0.284	0.50

Previous studies have shown that the Euclidean distance between cover songs and original songs is usually between 0.1 and 0.2. Based on our results, there is still a noticeable gap between the music generated by the model and human-created music, but these differences are not significant. It should be noted that music is subjective, and more than half of the human evaluators cannot distinguish whether the music is generated by the model or created by humans. This

Table 3. The table shows using interactive chaotic algorithms expected outcomes

Dataset	Euclidean disrance	Human Evaluations
FreeMIDI	0.30	0.65
MIDIworld	0.35	0.70
POP909	0.20	0.60

suggests that human creativity and evaluation of music still involve some degree of subjectivity. Therefore, the difference in the results of the Euclidean distance may be due to the fact that the generated songs have a different style from the original songs.

The expected outcome is a conservative estimate of a 10% improvement. This is based on the fact that interactive chaotic algorithms have a chaotic randomness and incorporate the subjective perception of humans, which allows the model to gain stronger randomness during training and optimize towards human aesthetic direction, thereby generating music that is closer to human-created works.

5 Future Work

For future research, we have outlined several directions to further enhance our MIDI music generation model's performance. Firstly, we plan to explore alternative optimization algorithms and generation models to achieve even better results. Specifically, reinforcement learning algorithms such as Deep Q-Network (DQN) and Actor-Critic (AC) will be investigated to enhance the model's ability to produce music with increased diversity and creativity.

Additionally, we aim to explore the potential of Generative Adversarial Networks (GANs) in generating more realistic and human-like music. GANs have demonstrated success in various image and audio generation tasks, and we believe they hold promise for music generation as well.

Moreover, we intend to incorporate user feedback into the model's training process to further refine the quality of the generated music. This will involve developing an interactive system that enables users to provide feedback on the generated music, which can be used to dynamically update the model's weights in real-time.

Finally, to showcase the generalizability and scalability of our proposed model, we plan to evaluate its performance on a larger and more diverse dataset. By doing so, we aim to demonstrate its applicability in various music-related domains such as music composition, sound design, and game development.

References

1. Y. M. Costa, L. S. Oliveira, and C. N. Silla Jr, "An evaluation of convolutional neural networks for music classification using spectrograms," *Applied soft computing*, vol. 52, pp. 28–38, 2017.

2. D. Neimark, O. Bar, M. Zohar, and D. Asselmann, “Video transformer network,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 3163–3172.
3. C.-Z. A. Huang, T. Cooijmans, A. Roberts, A. Courville, and D. Eck, “Counterpoint by convolution,” *arXiv preprint arXiv:1903.07227*, 2019.
4. C.-Z. A. Huang, A. Vaswani, J. Uszkoreit, N. Shazeer, I. Simon, C. Hawthorne, A. M. Dai, M. D. Hoffman, M. Dinculescu, and D. Eck, “Music transformer,” *arXiv preprint arXiv:1809.04281*, 2018.
5. Y. Pei, “Chaotic evolution: fusion of chaotic ergodicity and evolutionary iteration for optimization,” *Natural Computing*, vol. 13, pp. 79–96, 2014.
6. Z. Wang*, K. Chen*, J. Jiang, Y. Zhang, M. Xu, S. Dai, G. Bin, and G. Xia, “Pop909: A pop-song dataset for music arrangement generation,” in *Proceedings of 21st International Conference on Music Information Retrieval, ISMIR*, 2020.