Multimedia Tools and Applications (2022) 81:29865–29885
https://doi.org/10.1007/s11042-022-12925-w

Polyphonic music generation generative adversarial network with Markov decision process

Wenkai Huang[1] · Yihao Xue[2] · Zefeng Xu[2] · Guanglong Peng[2] · Yu Wu[3]

Received: 7 December 2020 / Revised: 17 February 2021 / Accepted: 9 March 2022 /
Published online: 5 April 2022

Abstract
In the process of polyphonic music creation, it is important to combine two or more independent melodies through technical treatment. However, due to the diversity of polyphonic music sequences and the limitations of neural networks, it is difficult to create chords or melodies beyond the training data. As the music sequence increases, the probability of the generator producing the same note will increase, which will destroy the coherence of the music. Therefore, this paper proposes a novel polyphonic music creation model, combining the ideas of the Markov decision process (MDP) and Monte Carlo tree search (MCTS) and improving the Wasserstein Generative Adversarial Network (WGAN) theory. Through the zero-sum game and conditional constraints between generator and discriminator, the model in this study is closer to the unconstrained creation of music, and the growth of music sequence will not affect music coherence. Experimental results show that the algorithm proposed here has a better effect on polyphonic music generation than the latest methods.

Keywords Polyphonic music generation · Markov decision process (MDP) · Monte Carlo tree search (MCTS) · Deep learning · Wasserstein Generative Adversarial Network (WGAN)

1 Introduction

Producing various independent melodies and combining them harmoniously through technical processing are key to creating polyphonic music. For composers, this is a very heavy task, so there is hope that polyphonic music may be generated through neural networks. However, the

✉ Wenkai Huang
16796796@qq.com

[1] Center for Research on Leading Technology of Special Equipment, School of Mechanical and Electrical Engineering, Guangzhou University, Guangzhou 510006, China

[2] School of Mechanical and Electrical Engineering, Guangzhou University, 510006 Guangzhou, China

[3] Laboratory Center, Guangzhou University, Guangzhou 510006, People's Republic of China

classic Generative Adversarial Network (GAN) [7, 13] is limited for polyphonic music creation. For example, it is difficult for a classic GAN to create a new melody outside the training dataset, and it is difficult to break through the shackles of melody and tone in that dataset. As a polyphonic music sequence grows, the probability of the generated sequence samples continuously playing the same note increases, which would destroy music coherence. Therefore, the present research team designed a GAN model based on the Markov decision process (MDP) [21] and Monte Carlo tree search (MCTS) [4] to generate polyphonic music. This model will effectively improve composers' work efficiency and greatly reduce their burden.

Prior to this paper, other music generation and music sequence processing algorithms have been developed based on deep learning. Dean et al. [8] have constructed algorithm and improvisation libraries, which are medium-sized post-tone and post-rhythm keyboard music symbol libraries. After training, both libraries can acquire models with sufficient generalization ability, thus generating music. Goienetxea et al. [12] have extended the method for generating coherent melodies by using the coherent melody structure in template fragments to generate bertso melodies. Polo et al. [28] have proposed a music vision system, which can convert images into music (lasting seconds or minutes) to customize music rendering results according to application requirements. Whorley et al. [31] have studied the problem of extracting samples from music statistical models and have evaluated an improved iterated random walk technique using a four-part, multi-view system combined with a set of small general coordination rules. This technology can effectively find low cross entropy (high probability) solutions and improve the quality of music generation. Herremans et al. [15] have also proposed a Morpheus music generation system, which can generate polyphonic music at given tension profiles, with long-term and short-term repetition pattern structures. Hadjeres et al. [14] have solved the shortcomings of the traditional recurrent neural network (RNN), which is neither interactive nor creative. They propose a new structure—an anticipation-RNN—for interactive music generation, and they have proven its efficiency at generating melodies satisfying the unitary constraints of the soprano style in a J. S. Bach chorus. There are many similar music generation methods. For example, Conklin et al. [6] proposed a method for chord sequence generation. Mo et al. [25] proposed a music automatic generation model based on simulated annealing and genetic algorithm. Agarwal et al. [1] introduced some effective data set preprocessing and reconstruction techniques, and used LSTM to generate grammatically correct music scores. Mao et al. [24] proposed an end-to-end music generation model, which can create music according to a specific composer's style combination.

Based on the above research, Goodfellow et al. have drawn lessons from a novel pair theory (GAN) for the network construction of image generation and learning. For example, Dong et al. [9] proposed a GAN model which can have a convolution, which needs to add an additional refiner network to the generator. The network uses binary neurons in the output layer, and it can generate binary piano rolls directly from binary neurons. The experimental results show that better results can be obtained in many objective measurements with binary neurons. The team has also proposed two versions of music generation models MuseGAN V1 [10] and MuseGAN V2 [11]. MuseGAN V1 uses convolution in the generator and discriminator, which can generate multi-channel pop/rock music from scratch or can accompany a track provided by the user. MuseGAN V2 uses three symbolic, multi-channel, music generation models in a GAN framework: the interference model, the composer model, and the hybrid model. Experiments show that, given a specific track composed by humans, MuseGAN V2 can generate four additional accompaniment tracks. However, since GAN was proposed,

problems have hindered its implementation, including training difficulties, loss of generators (as discriminators cannot indicate the training process), and a lack of diversity among the generated samples. Many studies, including those concerning MuseGAN V1 and MuseGAN V2, are trying to solve this problem, but the effect is not yet satisfactory. For example, some researchers [22, 32–34] have improved algorithms in the original GAN. These algorithms rely on the experimental enumeration of the discriminator/generator architectures to find better network architecture settings, but they do not completely solve the training difficulties and the lack of diversity among the generated samples. To thoroughly solve the above problems, Martin et al.[2] propose using the Wasserstein Generative Adversarial Network (WGAN), based on the work of Goodfellow et al. [13], to remedy the instability of GAN training and collapse mode. This method removes the sigmoid function of the last layer of the discriminator, changes the loss function of the generator and discriminator, and truncates their absolute values to no more than a fixed constant, C, after updating the discriminator parameters. This renders the network more stable and makes it converge faster. However, in polyphonic music generation, as the length of the music sequence generated by WGAN increases, sequence coherence is broken, and the discriminator in the WGAN model finds it difficult to evaluate the incomplete sequence. That is, the discriminator is unable to evaluate the generation status of the music sequence in real time and provide feedback for the generator. Therefore, aiming at the problems of WGAN polyphonic music generation, this paper improves the structure of the generator and discriminator in the WGAN model and adds a policy gradient algorithm [29] to update the generator parameters. On the other hand, MDP [3, 21, 27, 35] is a mathematical model of sequential decision, which is used to simulate the stochastic strategies and rewards agents can achieve in the environment via the Markov property. As a deep learning method, the goal of MDP is to enable software agents to maximize return in interactive learning. For polyphonic music generation, the generator in the GAN model supervises learning needs to be informed of the appropriate notes required for each small segment sequence. However, there are two problems in the existing GAN model; (1) the generator finds it difficult to transfer gradient update, and (2) the discriminator finds it difficult to evaluate the incomplete sequence. Therefore, the present research team has the introduced MDP mechanism into the GAN model to generate polyphonic music. MDP has been widely used in interactive learning, which can help neural networks continuously gain knowledge according to the reward or punishment obtained. This effectively solves the problem of the GAN generator finding it difficult to transfer the gradient update, and MDP is suitable for the generation model construction in this paper.

However, it is still difficult for the discriminator to evaluate no holonomic sequences. Based on MDP theory, this paper introduces the idea of Monte Carlo tree search (MCTS)4. MCTS is a general term for a class of tree search algorithms. It mainly describes a search algorithm based on a tree data structure, which can balance exploration and utilization and remains effective in a huge search space. Since the birth of MCTS, it has mainly been used to solve the problem of large exploration space. Among other studies in this area, Chen et al. [5] have developed a deep MCTS control method for visual autonomous driving. The team uses two deep neural networks (DNNs) to, respectively, predict the action state transition and action selection probability. Deep MCTS then takes this prediction information from the DNNs and reconstructs multiple possible trajectories to predict driving maneuvers. Based on the current road conditions and the predicted driving operation, deep MCTS can select the best track. This method can, thus, predict driving maneuvers and improve the stability and performance of driving control. Sironi et al. [30] have applied MCTS to general game playing and propose a

self-adaptive MCTS strategy (SA-MCTS), which automatically adjusts the search control parameters of each game in the search process. SA-MCTS is a successful strategy for domains in which parameter adjustment is needed for every problem. SA-MCTS is also an effective alternative for domains in which offline parameter adjustment is expensive or infeasible. In this paper, MCTS is introduced to take advantage of its efficient search strategy for exploring polyphonic music sequences and completing sequences from the generator, thus solving the problem of the discriminator being unable to evaluate incomplete sequences. In this way, the model can be applied to any incomplete polyphonic music sequence at any time.

This paper combines WGAN and MDP theory, integrates MCTS, and proposes a polyphonic music generation method based on MDP and WGAN, as shown in Fig. 1. The generator in the proposed network is a long short-term memory (LSTM) unit, while the discriminator is a convolution unit. Therefore, similar to typical neural networks, the generator and discriminator in this model can automatically update the internal convolution check parameters through iterative learning. At the same time, to solve the problem in which the discrete data gradient cannot be transmitted back to the generator, the research team has combined the generator in the model with MDP theory, and added MCTS to transform generator optimization into maximized rewards; this optimizes the generator by using the policy gradient [29]. Again, MCTS can also effectively solve the problem in which the discriminator from the existing.

GAN model is unable to evaluate incomplete music sequences. This will improve the diversity of the music from the generator and the quality of the music generated by the network as a whole. By learning the characteristics of and mapping between various independent melody sequences in a polyphonic music dataset, this model can generate polyphonic music that is closer to real-world music, effectively resolving the issues concerning the continuity of polyphonic music sequences and ensuring the diversity of generated samples—making the model more powerful for unsupervised music sequence processing.

This method has the following advantages. (1) The model is built based on MDP and WGAN. Adding the policy gradient mechanism to the generator can transform generator optimization into maximized rewards, thus optimizing the generator. This makes the network more quickly converge in the direction of ideal polyphonic music. Thus, the generated polyphonic music will not be destroyed with the growth of the sequence; better quality music will be generated; the shackles of the dataset will be broken; and melodies outside the dataset can be created. At the same time, the model, after learning, will be able to generate polyphonic

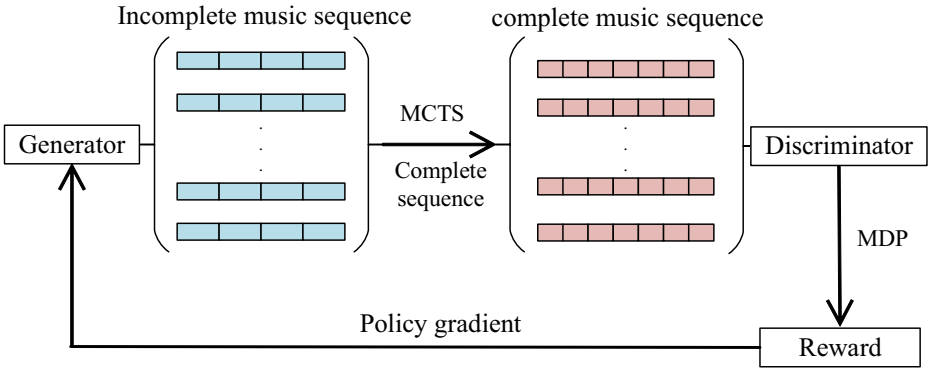

Fig. 1 Flowchart of polyphonic music generation in this study

music that is closer to real-world music styles. (2) The construction of this model draws lessons from WGAN principles and adds Wasserstein distance [2], which stabilizes model training by resolving the issue in which, the better the discriminator is trained in the existing GAN, the more seriously the generator gradient disappears. (3) After training, the generator in the GAN model can be used independently of the discriminator. The model has no requirements for the length of the input polyphonic music sequence and can automatically generate excellent music corresponding to the input sequence. (4) This paper is the first to integrate MCTS into WGAN for polyphonic music generation, removing the issues in which it is difficult for the discriminator, in the existing GAN model, to evaluate incomplete sequences. This proposed model, thus, improves the ability of neural networks to generate diverse polyphonic music. This study, therefore, is certainly innovative. Figure 2 visualizes the original music and the music generated by the model.

2 Technical details

2.1 Network construction

In this paper, the researchers have built a neural network based on WGAN and MDP to learn and generate polyphonic music. When learning a polyphonic music sequence, the generator randomly generates the polyphonic music sequence, and then inputs the music generated by the model into the discriminator with the original music for classification. The output of the discriminator represents the true or false judgment for each piece music. After finding the error between the judgment result and the label, the parameters of the generator model are adjusted according to the gradient descent method to achieve the purpose of the model's autonomous learning. This model was constructed by referring to the WGAN model presented by Martin et al. [2]. During the generation process, the original learning sequence and the polyphony of the LSTM are reconstructed. The discriminator model includes a convolution layer and a fully-connected layer, which are used to accurately identify the real polyphonic music and the polyphonic music generated by the model in the iterative learning process. To improve the stability of the model for polyphonic music generation, the research team has added batch normalization to all layers except the input convolution layer of the discriminator [16]. Figure 3 depicts the polyphonic music generation model of this paper, including the generator and discriminator models.

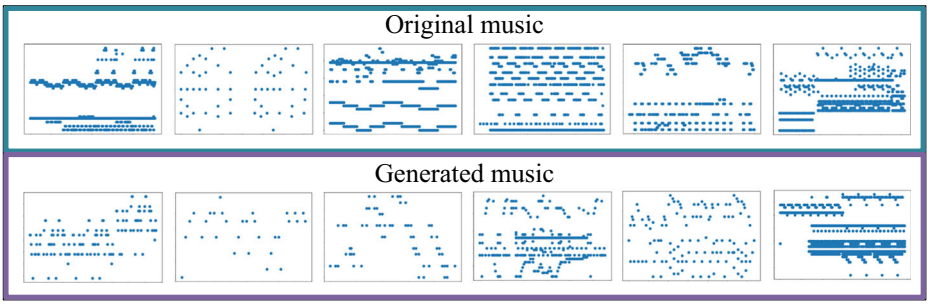

Fig. 2 Visualization of original and generated music

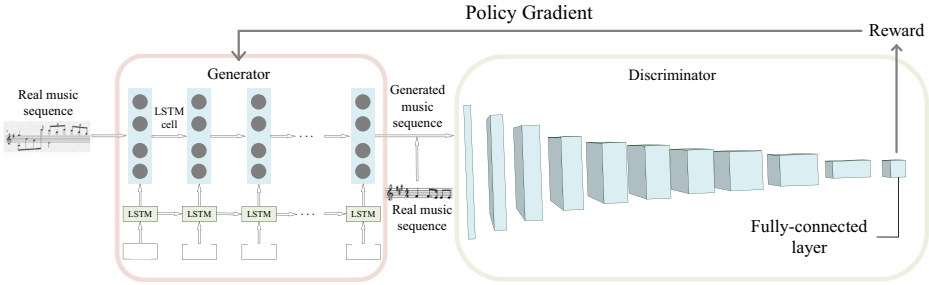

Fig. 3 Schematic diagram of the polyphonic music generation model in this study

During the training process, the generator model learns to generate polyphonic music similar to real polyphonic music, while the discriminator model learns to more accurately distinguish between the real and the model generated polyphonic music. This study combines WGAN and MDP to generate polyphonic music. Discrete music sequences differ from continuous image processing, which is not conducive for training typical GAN. Moreover, GAN can only evaluate the whole sequence, rather than the local sequence, which leads to the generator producing a discrete output and makes it difficult for the discriminator to return a gradient to update the generator. Therefore, the current research team has used the optimization algorithm for reference, keeping the model based on the idea of confrontation. The discriminator must be asked to score the complete polyphonic music sequence. Therefore, this study uses MCTS to complete the various possibilities of each sequence. The discriminator scores these complete sequences and generates a reward, which is then sent back to update the generator.

As mentioned above, the generator network has been built with reference to the WGAN structure proposed by Martin et al. [2]. However, the current team has applied an LSTM layer instead of the standard convolution layer. The main feature of LSTM is that it is helpful for learning the internal memory of sequence information, receiving important input information and accurately predicting the next input based on it. In the process of polyphonic music generation, the prediction of the next music sequence depends on the previous one. Therefore, the generator network designed by the present team is composed of LSTM units. When input samples are input into an LSTM unit, the unit will not hide the activation from the beginning to replace the whole activation but will, instead, keep the state unaffected by the previous activation. To improve the overall stability of the network, the team has added a batch normalization layer to all LSTM layers. In addition, to increase the processing capacity of the generator network for more complex and longer music sequences, the team has increased the number of LSTM units. There are 512 LSTM units in the network, and each LSTM unit has 400 feature maps.

The discriminator network of this polyphonic music generation model is composed of a convolution layer and a fully-connected layer. It can, therefore, extract and classify input music sequence features. In addition, the discriminator network must also create a reward for the complete sequence generated by the generator network and send it back to update the generator network. As described above, the network in this paper is divided into two parts. The generator network is composed of LSTM units, while the discriminator network is composed of a convolution layer and a fully-connected layer. Batch normalization layers have been added to each network unit, except for the output unit of generator, the input convolution unit, and the fully-connected layer of the discriminator. Additionally, a Rectified Linear Unit

(ReLU) layer has been added after the convolution unit of the discriminator, and the Tanh activation function is used in the output layer of the generator network. After training, the generator network can be used alone to generate polyphonic music. Table 1 shows the concrete structure of the discriminator.

In this model, batch normalization is used in all layers except the output unit of the generator, the input convolution unit, and the fully-connected layer of the discriminator. After calculating the mean and variance in each mini batch, the output signal of each layer is normalized. Batch normalization can reduce the influence of the weight initialization value distribution, while improving the stability of the model and accelerating its convergence speed.

2.2 Network strategy gradient

When designing the strategy gradient for the polyphonic music generation model, it was necessary for the authors to introduce the relevant GAN theory proposed by Goodfellow et al. [13]. To learn the target generator distribution from standard data, the GAN input is defined as $P_z(z)$, and the generation model is defined as $G_\theta(y_t|Y_{1:t-1})$. G_θ is the model parameter; y_t is the output at time t; and $Y_{1:t-1}$ is the output before time t. $G(z)$represents the music sequence generated by the generator. The output, $D(x)$, of the discriminator represents the ratio at which the corresponding input music sequence is identified as true. The process of confrontation between the generator and the discriminator is shown in Eq. (1).

$$\min_G \max_D V(D,G) = E_{x\mathbb{P}_{data}(x)}[log(D(x))] + E_{z\mathbb{P}_z(z)}[log(1-D(G(z)))] \tag{1}$$

However, classic GAN presents a large problem. The better the discriminator is trained, the more seriously the generator gradient disappears. Therefore, Martin et al. [2] propose WGAN, based on Goodfellow et al. [13], and replace Jensen-Shannon divergence with Wasserstein distance, which effectively solves the problem of instability in the classic GAN training process. Therefore, the team introduced WGAN into the polyphonic music generation model.

Wasserstein distance, also known as Earth–Mover distance, is defined as shown in Eq. (2).

$$W(P_r, P_g) = \inf_{\gamma\sim\Pi(P_r,P_g)}\mathbb{E}_{(x,y)\sim\gamma}[||x-y||] \tag{2}$$

P_r is the sample distribution of a real polyphonic music sequence; P_g is the sample distribution of a polyphonic music sequence produced by the generator; and $\Pi(P_r, P_g)$ is the set of all

Table 1 The structure of the discriminator

Numbers	Layer type	Convolution kernel	Steps	Depth
1	Convolution	3×1	1×1	32
2	Convolution	3×1	1×1	32
3	Convolution	3×1	1×1	32
4	Convolution	5×1	2×1	64
5	Convolution	5×1	2×1	64
6	Convolution	5×1	2×1	64
7	Convolution	5×1	2×1	128
8	Convolution	5×1	2×1	128
9	Convolution	5×1	2×1	128
10	Convolution	5×1	2×1	256
11	Fully-connected	1×1	1×1	2

possible joint distributions combined by P_r and P_g. In other words, the marginal distribution of each distribution in $\Pi(P_r, P_g)$ is P_r and P_g. For each possible joint distribution, γ, $(x, y) \sim \gamma$ may be sampled to obtain a real music sequence sample, x, and a model generated music sequence sample, y, and calculate the distance $||x - y||$ of the pair of samples. In this way, the expected value $\mathbb{E}_{(x,y)\sim\gamma}[||x - y||]$ of the sample pair distance under joint distribution γ may be calculated. In all possible joint distributions, the lower bound $\gamma \sim \Pi(P_r, P_g)\inf\mathbb{E}_{(x,y)\sim\gamma}[||x - y||]$, is defined as Wasserstein distance.

To define Wasserstein distance as the loss function of the generator and to generate a meaningful gradient for updating the generator, so that the generated distribution would be closer to the real distribution, Martin et al. [2] transformed it into the following form.

$$W(P_r, P_g) = \frac{1}{K}\| f \|_L \leq K\sup\mathbb{E}_{x\sim P_r}[f(x)] - \mathbb{E}_{x\sim P_g}[f(x)] \tag{3}$$

It should be noted that the concept of Lipschitz continuity is used in Eq. (3). In fact, an additional restriction is imposed on a continuous function f, which requires a constant $K \geq 0$, so that any two elements, x_1 and x_2, in the definition domain satisfy Eq. (4).

$$| f(x_1) - f(x_2) | \leq K | x_1 - x_2 | \tag{4}$$

The Lipschitz constant of the function, f, is called K. Thus, Eq. (4) means that, if the Lipschitz constant $|| f ||_L$ of function f does not exceed K, the upper bound of $\mathbb{E}_{x\sim P_r}[f(x)] - \mathbb{E}_{x\sim P_g}[f(x)]$ is taken for all f that may satisfy the condition and then divided by K. A set of parameters, ω, may be used to define a series of possible functions, f_ω. At this time, the solution to Eq. (4) can be approximately changed into solving Eq. (5).

$$K \cdot W(P_r, P_g) \approx \omega : | f_\omega |_L \leq K\max\mathbb{E}_{x\sim P_r}[f_\omega(x)] - \mathbb{E}_{x\sim P_g}[f_\omega(x)] \tag{5}$$

Equation (5) needs to satisfy the limit $|| f_\omega ||_L \leq K$, K can only enlarge the gradient of neural network by K times. As long as $K \neq +\infty$, the direction of gradient will not be affected. Moreover, it is necessary to limit all parameters ω_i of neural network f_θ not to exceed a certain range $[-c, c]$. At this time, the derivative $\frac{\partial f_\omega}{\partial x}$ of the input sequence sample x will not exceed a certain range, so there is a constant K so that the local variation of f_ω will not exceed the range.

Therefore, the research team can construct a discriminator network, f_ω, with parameter, ω, and the last layer not being a nonlinear activation layer. Under the condition that ω does not exceed a certain range, Eq. (6) can be maximized as much as possible.

$$L = \mathbb{E}_{x\sim P_r}[f_\omega(x)] - \mathbb{E}_{x\sim P_g}[f_\omega(x)] \tag{6}$$

When Eq. (6) is as large as possible, L approximates the Wasserstein distance between the real distribution and the generated distribution. The discriminator in the original GAN accomplishes the task of true and false dichotomy, so the sigmoid function is used in the last layer. However, the discriminator, f_ω, in WGAN does approximate fitting Wasserstein distance, which belongs to the regression task, so it is necessary to remove the sigmoid function of the last layer.

Next, the generator should approximately minimize Wasserstein distance, so L can be minimized. Because of the excellent properties of Wasserstein distance, the generator gradient does not disappear. Considering that the first term of L is independent of the generator, two WGAN losses can be obtained, as shown in Eqs. (7) and (8).

$$- \mathbb{E}_{x\sim P_g}[f_\omega(x)] \tag{7}$$

$$\mathbb{E}_{x\sim P_g}[f_\omega(x)] - \mathbb{E}_{x\sim P_r}[f_\omega(x)] \tag{8}$$

Equation (7) is the loss function for the generator in this model, and Eq. (8) is the loss function for the discriminator.

Because this study introduces MDP into the GAN model, the goal of the generator is to generate sequences to maximize the expectation of return, as shown in Eq. (9).

$$J(\theta) = \mathbb{E}[R_T|s_0,\theta] = \sum_{\mathcal{Y}_1 \epsilon y} G_\theta(\mathcal{Y}_1|s_0)\cdot Q_{D_\phi}^{G_\theta}(s_0,\mathcal{Y}_1) \tag{9}$$

The expectation of a return that produces a complete sequence, under the conditions of s_0 and θ, is expressed as $J(\theta)$. G_θ represents the generator model, and $Q_{D_\phi}^{G_\theta}$ is the action-value function of the sequence. The calculated $J(\theta)$ is the function the polyphonic music generation model seeks to maximize. For the Q value in the above formula, this study uses the reinforcement algorithm and takes the Q value as the return value of the discriminator, as shown in Eq. (10).

$$Q_{D_\phi}^{G_\theta}(a=\mathcal{Y}_T, s=Y_{1:T-1}) = D_\phi(Y_{1:T}) \tag{10}$$

Because the reward generated by an incomplete music sequence has no practical significance, the Q value of Y_T cannot be calculated directly after Y_T generation in the case of original Y_1 to Y_{T-1}, unless Y_T is the last value in the whole sequence. Therefore, the present research team has used MCTS to complete the contents after Y_T. In this study, all possible sequences after the same Y_T with MCTS[4] are calculated as forward, and then the average value is calculated, as shown in Eq. (11).

$$Q_{D_\phi}^{G_\theta}(s=Y_{1:T-1}, a=\mathcal{Y}_T) = \left\{ arraycl \frac{1}{N}\sum_{n=1}^{N} D_{\left(Y_{1:T}^n\right)}, Y_{1:T}^n \in MC^{G_\beta}(Y_{1:T};N) for\ t \right. \tag{11}$$

After generating a complete realistic music sequence, the discriminator is trained as shown in Eq. (12).

$$\min - \mathbb{E}_{Y\sim p_{data}}[\log \mathrm{D}(\mathrm{Y})] - \mathbb{E}_{Y\sim G_\theta}[\log(1-\mathrm{D}(\mathrm{Y}))] \tag{12}$$

In the above formula, the model must maximize D, to judge whether the real music sequence is true, and minimize the opposite number of D, to judge that the music sequence generated by the model is false. After one or more rounds of training, the discriminator obtains a better D. Then, it must update G with D. This update of G is a gradient descent

$$\theta \leftarrow \theta + \alpha_h \nabla_\theta J(\theta) \tag{13}$$

where α_h represents the learning rate. The derivation of the gradient for the generator objective function is shown in Eq. (14).

$$\nabla_\theta J(\theta) = \mathbb{E}_{Y_{1:t-1}\sim G_\theta}\left[\sum_{\mathcal{Y}_t \in y} \nabla_\theta G_\theta(\mathcal{Y}_t \mid Y_{1:t-1})\cdot Q_D^{G_\theta}(Y_{1:t-1},\mathcal{Y}_t)\right] \tag{14}$$

In this study, the research team has used MCTS [4]. When evaluating polyphonic music sequences at any time, they have considered the possible long-term rewards [20], as well as the overall situation for each sequence. The MCTS algorithm belongs to a kind of Monte Carlo method, and MCTS is a heuristic search algorithm based on statistical simulation. Therefore, it has been used here for the decision-making process of polyphonic music sequence generation. Adding the MCTS algorithm resolves the problem in which the discriminator cannot evaluate incomplete music sequences. It also solves the problem in which the generator's music results worsen as the length of the music sequence increases (Fig. 4).

3 Experimental results and analysis

3.1 Dataset composition and preprocessing

The research team have constructed 6,947 original datasets composed of polyphonic music [19] to serve as the dataset for the polyphonic music generation model. The file format for the dataset is music instrument digital interface (MIDI) [17], which is usually an alternating sequence of notes and chords. MIDI is the most widely used, standard music format in the field of music generation. MIDI files record music with digital control signals of notes, and a complete MIDI file can contain dozens of music tracks. MIDIs do not transmit sound signals, but notes, control parameters, and other instructions, which is very suitable for music generation research. After the team loaded a MIDI file in the dataset, each note in the file was parsed into a list of start time, duration, tone, and speed. Then the team inserted virtual notes into the file to make the length of notes and chord sequence the same and marked each time step with different lengths. The marked music sequence was used as the input sequence for the generation model. After training with the model presented in this paper, the resulting music sequence is converted into two musical symbols, one note and one chord by the opposite process of pre-processing. These symbols are added to the melody and chord sequences of the file. After all the markers have been processed, the melody and chord sequences are merged into a MIDI file. The complete process is shown in Fig. 5.

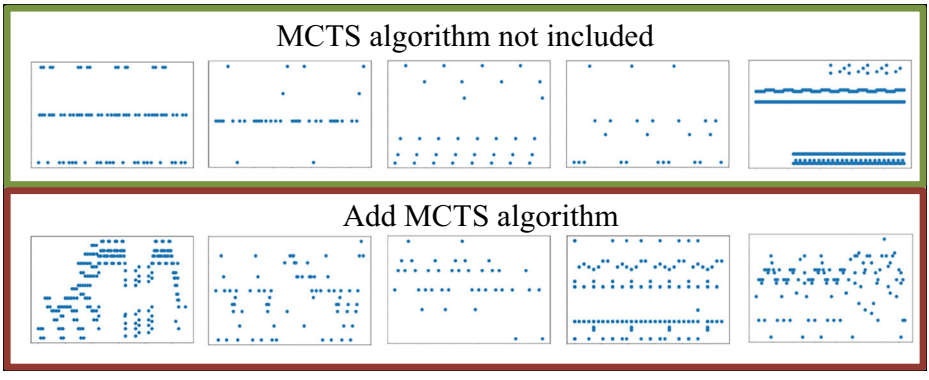

Fig. 4 Visual graph of adding the MCTS algorithm and not adding the MCTS algorithm to the model

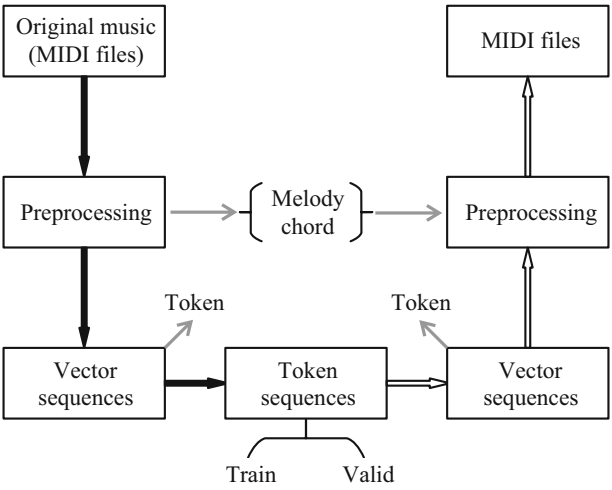

Fig. 5 Complete processing flow chart of original music dataset

3.2 Training process

The experiment has employed three models to generate music results: the WGAN model, the WGAN and MDP model, and the proposed model for comparative training (Fig. 6). At the same time, the research team has compared the effects of the model in this study with the MuseGAN v1 model [10], the MuseGAN v2 model [11], and the LSTM music generation model [23] for polyphonic music generation.

According to Fig. 6, the model provided in this paper has relatively good music generation effects, regardless of the length of the music sequence generated. In the training process, when using WGAN for music generation, the research team has employed a convolution network as the generator. However, convolution networks find it difficult to effectively process and extract the context features of music sequences, and they cannot retain the location information in the sequence. As the generated music sequence grows, music generation worsens. Then, the team used WGAN + MDP to generate music, and this has improved the original WGAN by replacing the convolution layer of the generator with an LSTM layer to better extract the characteristics of the original music sequence. The MDP algorithm also makes the generated music more fluent, and, as the generated music sequence grows, the model will not always generate single melody music, thereby giving the GAN model a certain creative ability. Subsequently, the team used the study's proposed method to generate music. The MCTS algorithm can balance the exploration and use of a polyphonic music sequence. As the input sequence grows, the algorithm can still maintain its effectiveness, as shown in Fig. 6. In the first stage, the strategy used during music sequence sampling in the MCTS can be continuously improved throughout the sampling process; in the second stage, if the current state of the music sequence is not in the MCTS, the default strategy is used to complete the sampling of the whole state sequence, and the current sampling state of the music sequence is included in the search tree. As shown in Fig. 6, with the addition of the MCTS algorithm, the proposed model further improves the effect of music generation. Regardless of the length of the music sequence to be generated, this model is more fluent.

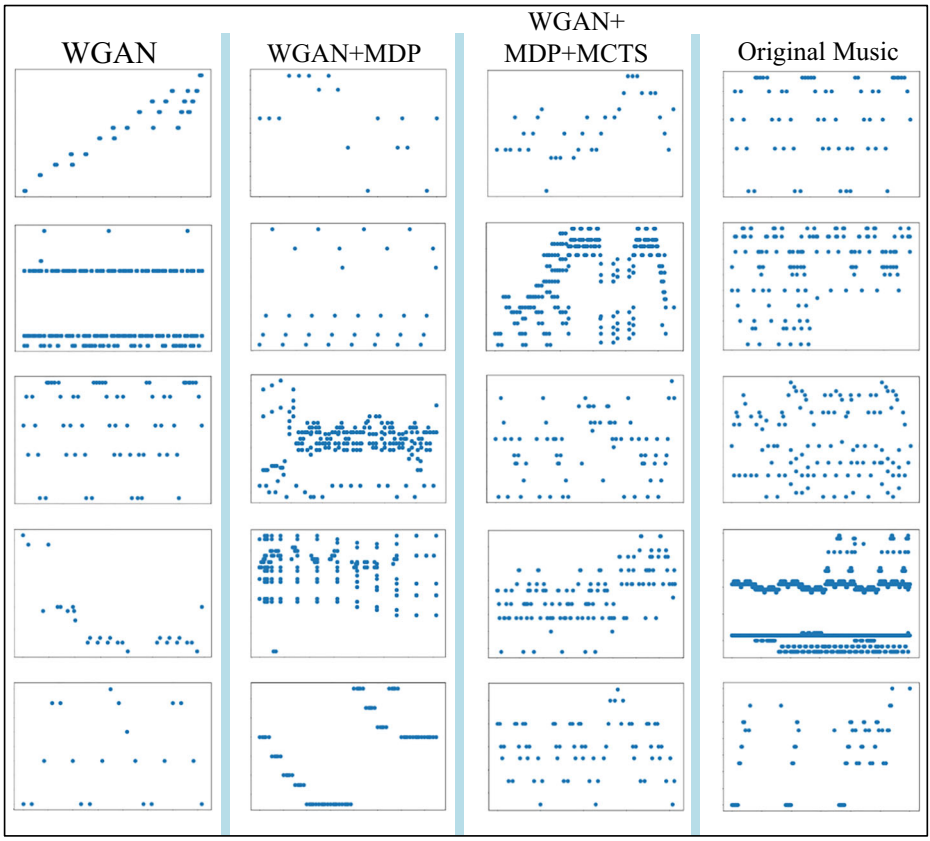

Fig. 6 The music generated by the proposed model, the music generated by the WGAN model, and the music contrast map generated by the WGAN and MDP model

Figure 7 shows the change of loss value with iteration times for the WGAN model, the WGAN and MDP model, and the proposed model.

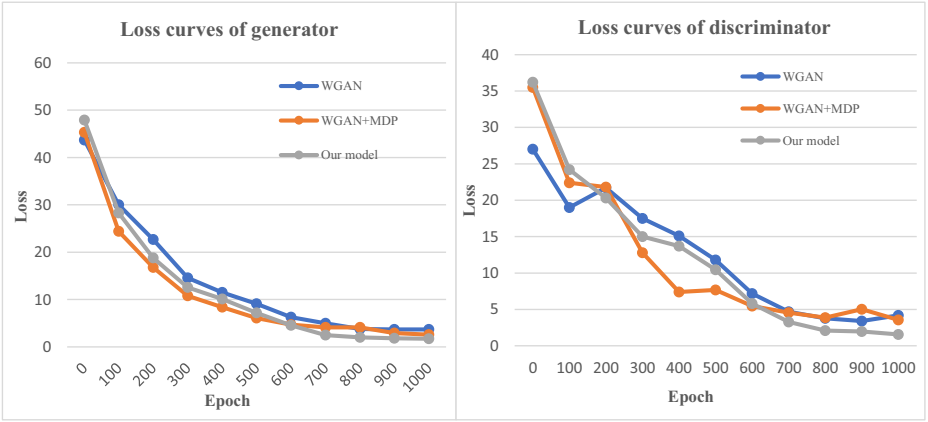

Fig. 7 Curve of functions changing with iterations

According to Fig. 7, the proposed model has a faster convergence speed and reaches a more stable state as soon as possible. During the training process, it is difficult for the convolution layer of the generator in the WGAN model to learn complex music sequence features. It is also difficult for the convolution layer to take into account the correlation before and after the sequence when processing a music sequence, which cannot effectively solve the problem of long-term dependence, resulting in slow convergence speed and poor quality music generation. When comparing the WGAN and MDP model with the proposed model, it is evident that the generator with the MCTS mechanism has the fastest convergence effect. Compared with image processing, the neural network for sequence processing does not require a large number of iterations to achieve a stable state. When the overall number of iterations in the network is 700, the errors of the generator and discriminator become stable, and the model generates polyphonic music well. The discriminator can also provide a stable residual for the generator, which helps both the generator and the discriminator with internal parameter fine adjustments. Figure 8 compares the visualization pictures output by the proposed model during the learning process, as well as their corresponding iteration numbers.

3.3 Experimental result

By studying the existing music generation methods and analyzing the WGAN model, through a large number of experiments, it has been proven that the model proposed in this paper is more suitable for polyphonic music generation than both the WGAN model and the WGAN and MDP model. In the experiments, Python 3.7 and Tensorflow 2.2.0 have been used to build the model, and the training has been carried out on the Windows platform with a NVIDIA GTX 1080 TI GPU. It should be noted that the input of this model does not require any preprocessing, and, after training, the generated network can output music sequences of any size, as shown in Fig. 9.

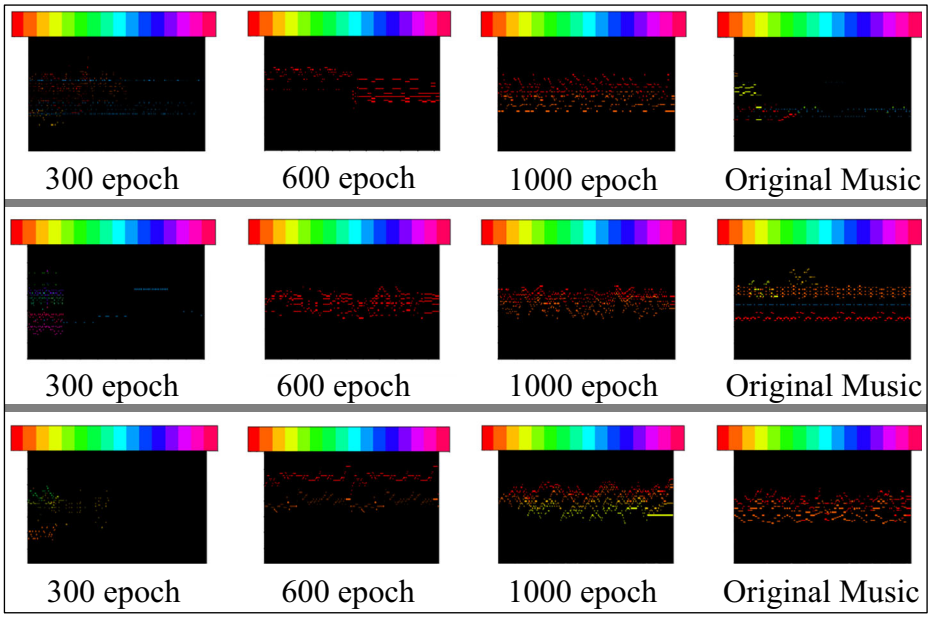

Fig. 8 Music visualization images obtained by different iterations of this model

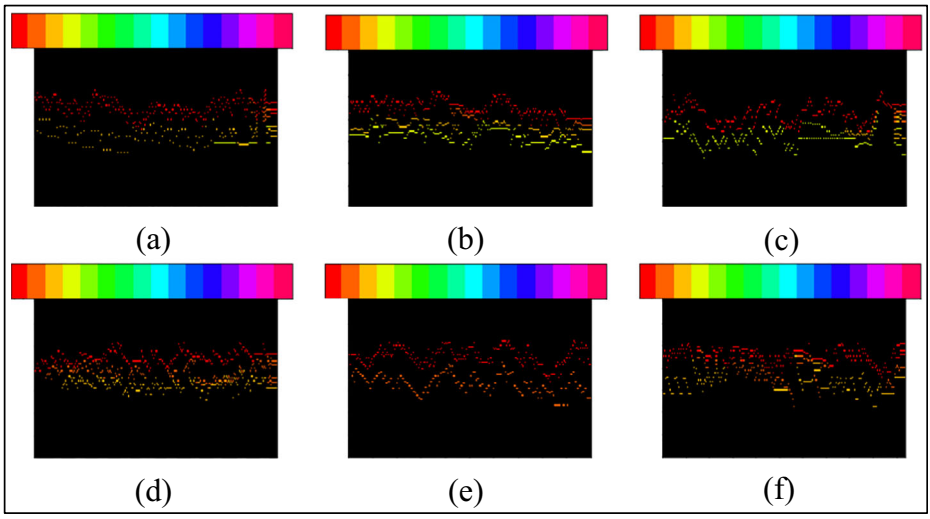

Fig. 9 Model-generated visual graphs for music sequences of different lengths

In Fig. 9(a)–(f) represent the music visualization map generated by the proposed model, and the length of the music sequences increases gradually from (a) to (f). As seen in Fig. 9, when the music sequence is long, the generated music obtains more details; when the music sequence is short, the generated music details are relatively simplified. However, regardless of the length of the output music sequence, the proposed music generation model can generate excellent corresponding music sequences, which can be directly used by composers and effectively help/inspire them to create music. Figure 10 shows the various styles of polyphonic music output by the model using the dataset created for this paper.

Figure 10 indicates that the network in this paper can obtain better generation results for various styles of music generation tasks.

3.4 Comparison of existing technologies

This paper uses the test music sequence set in the algorithm verification stage, and uses the relevant principles of macro precision (macro-P), macro recall (macro-R) and corresponding Macro-F1, and makes a detailed calculation and comparative analysis between the model proposed in this paper and the model proposed by Dong et al. [11]. The results show that, compared with the music generation model proposed by Dong et al., the polyphonic music generation model in this paper has better polyphonic music sequence generation ability.

The proposed model can be regarded as a binary classifier. For the binary classification problem, precision and recall are suitable performance measures suitable. Usually, both are used to measure a model's output. The examples can be divided into four cases according to the combination of real class and classifier prediction category: true positive (TP), false positive (FP), true negative (TN), and false negative (FN). The definitions of precision and recall are shown in Eqs. (15) and (16), respectively.

$$P = \frac{TP}{TP + FP} \tag{15}$$

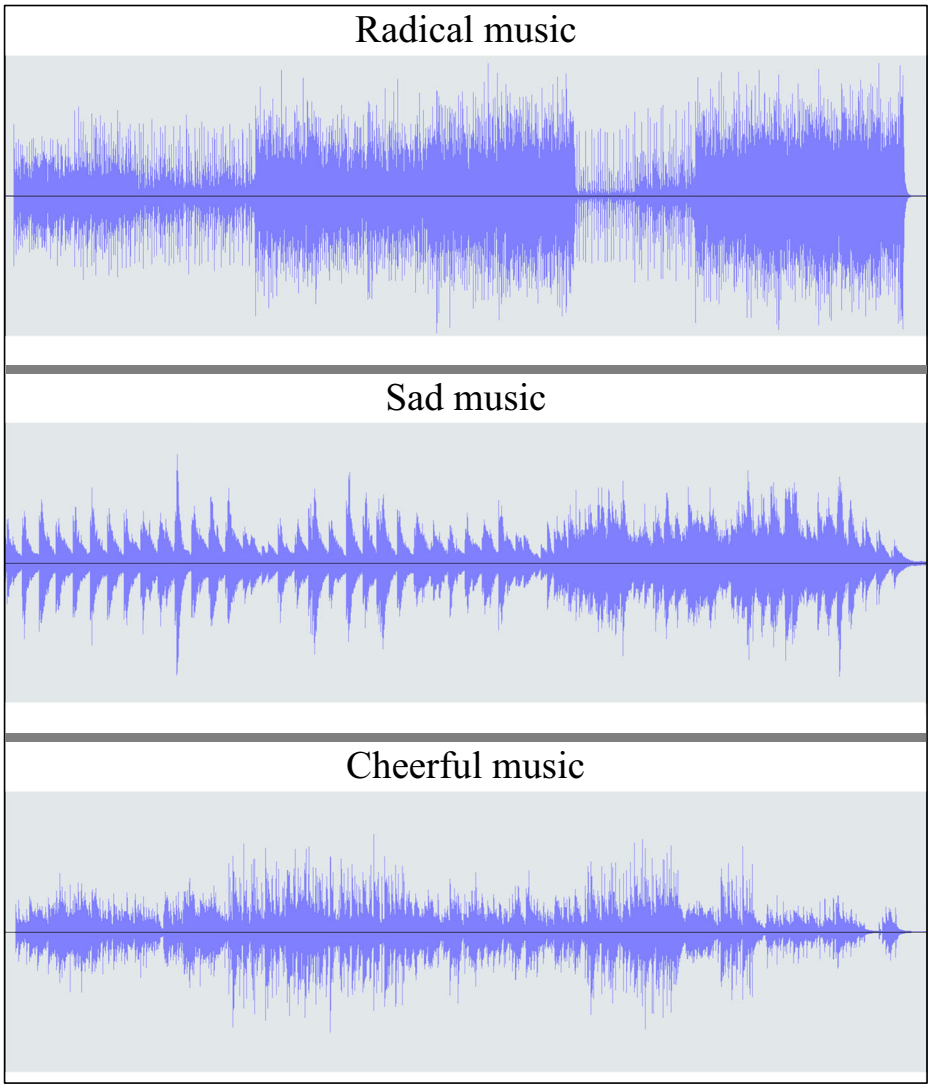

Fig. 10 Visualization for different styles of generated music

$$R = \frac{TP}{TP + FN} \tag{16}$$

F1 score is an index used to measure the accuracy of binary classification models and unbalanced data in statistics. It considers both the precision and recall of a classification model. F1 score is a weighted average of precision and recall; its maximum value is 1, and its minimum value is 0. In the multi-classification problem, when calculating the F1 score of a model, there are two calculation methods: micro-F1 and macro-F1. The use of the two methods has always been controversial. However, Lewis et al. [18] point out that macro-F1 is the average value of F1 score in all classes, and they believe that it is, therefore, the correct calculation method. Opitz et al. [26] have analyzed both methods and proven that micro-F1 is not sensitive to the distribution of errors; they also recommend macro-F1. Therefore, this paper

uses macro-P and macro-R to calculate the corresponding macro-F1 and compares that value with the latest music generation methods. The formulas of macro-P, macro-R, and macro-F1 are as follows.

$$macro-\mathrm{P} = \frac{1}{\mathrm{n}} \sum_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{P_i} \tag{17}$$

$$macro-\mathrm{R} = \frac{1}{\mathrm{n}} \sum_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{R_i} \tag{18}$$

$$macro-\mathrm{F1} = \frac{2 \times \mathrm{macro-P} \times \mathrm{macro-R}}{\mathrm{macro-P} + \mathrm{macro-R}} \tag{19}$$

In this paper, all polyphonic music sequences in the original dataset are "true" samples, while noise sequences are "false" samples. At the same time, in the polyphonic music generated by the corresponding model, the regular music sequence is defined as the "true" sample predicted by the classifier, and the generated noise sequence is the sample predicted as "false." It should be noted that, in this experiment, the research team has classified music sequences that repeat the same tone as noise sequences. Similarly, the result of the example can also be used as the classification result of the music generation model presented by Dong et al. [11].

During the experiments, the research team input the original polyphonic music dataset into the proposed music generation model to obtain the polyphonic music generated by the algorithm in this paper. Simultaneously, the researchers input the same polyphonic music dataset into the model from Dong et al. [11] to obtain the polyphonic music generated in that study [11]. After completing the above steps, the team used the polyphonic music sequence in the original dataset and the formulas for precision and recall to obtain the comparison test results of the two models, as shown in Table 2.

It should be noted that, after the neural network, the output music sequence will produce a variety of new notes, so it is not possible to determine the one-to-one correspondence between the music sequence in the original dataset and the generated music sequence. Therefore, a certain calculation range must be set. To determine the calculation range, the research team conducted experiments concerning the influence of the range on the $macro-\mathrm{F1}$ and $macro-\mathrm{F1_D}$ test results (Fig. 11).

The $macro-\mathrm{F1}$ test results for the proposed model, as well as the $macro-\mathrm{F1_D}$results for the Dong et al. model's [11] output music, stably improve as the range increases. In addition, when the range is 1 notes × 100 notes, the maximum difference between $macro-\mathrm{F1}$ and $macro-\mathrm{F1_D}$ is 0.0267. When the range is less than 1 notes × 100 notes, because the model's note

Table 2 Test results of macro-P, macro-R, and macro-F1 for the output music sequences given by the Dong et al. model [11] and the proposed model

Test items	Results of Dong et al. model	Results of proposed model
macro-P (The range is 1 notes × 200 notes)	0.2540	0.2912
macro-R (The range is 1 notes × 200 notes)	0.2796	0.2723
macro-F1 (The range is 1 notes × 200 notes)	0.2661	0.2815

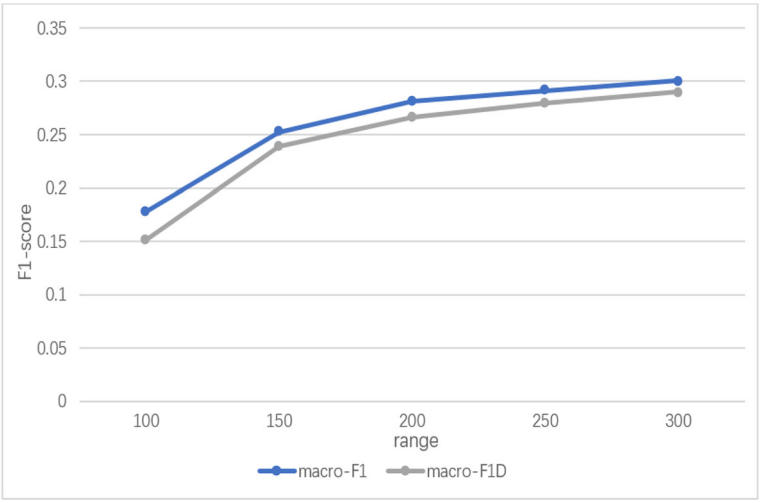

Fig. 11 Experiment concerning the influence of range change on $macro-F1$ and $macro-F1_D$

displacement impacts both $macro-\mathrm{F1}$ and $macro-\mathrm{F1_D}$, the difference between them decreases. When the range is larger than 1 notes × 100 notes, the tolerance condition is further relaxed due to the increase of contrast note range, and the reliability of test results also decreases. Although the both F1 scores have been improved, the test range is too large, making the results unconvincing. Therefore, the research team chose the values of $macro-\mathrm{F1}$ and $macro-\mathrm{F1_D}$ in the range of 1 note × 200 notes as the results of the comparative test in this experiment. It should be noted that, with the change of test range, the value obtained for $macro-\mathrm{F1}$ is always higher than that obtained for $macro-\mathrm{F1_D}$. Thus, it is proven that the polyphonic music generation model based on WGAN theory, which integrates the ideas of MDP and MCTS, is better than the MuseGAN model proposed by Dong et al. [11].

To more effectively verify the effectiveness of the model proposed in this paper, the research team has compared this model with the latest music generation models, including the MuseGAN v1 model [10], the MuseGAN v2 model [11], and the LSTM model [23]. To avoid generator collapse in the MuseGAN model and to optimize the training speed, the WGAN gradient penalty (WGAN-GP) network is used. Mangal et al. [23] propose an algorithm to generate notes using RNN, mainly via an LSTM network. The model proposed in this study seeks to learn polyphonic note sequences on a single-layer LSTM network. Compared with the above research, the model in this paper combines MDP theory and MCTS, based on WGAN. When training polyphonic music sequences in this model, there is no problem of gradient explosion or gradient disappearance. Therefore, the research team did not add a gradient penalty term to the original WGAN. In this paper, GAN is used instead of RNN to generate music, and LSTM units are used in the generator. Compared with convolution units, LSTM units have better music sequence generation ability and produce higher quality, more detailed music. The music visualization diagrams generated by the proposed model, the MuseGAN v1 model, the MuseGAN v2 model, and the LSTM models are shown in Fig. 12.

According to Fig. 12, the model proposed in this paper performs better than the other latest algorithms [10, 11, 23]. As music sequences grow, the proposed model will not always produce the same melody. Combining GAN and MDP theory can also effectively enhance

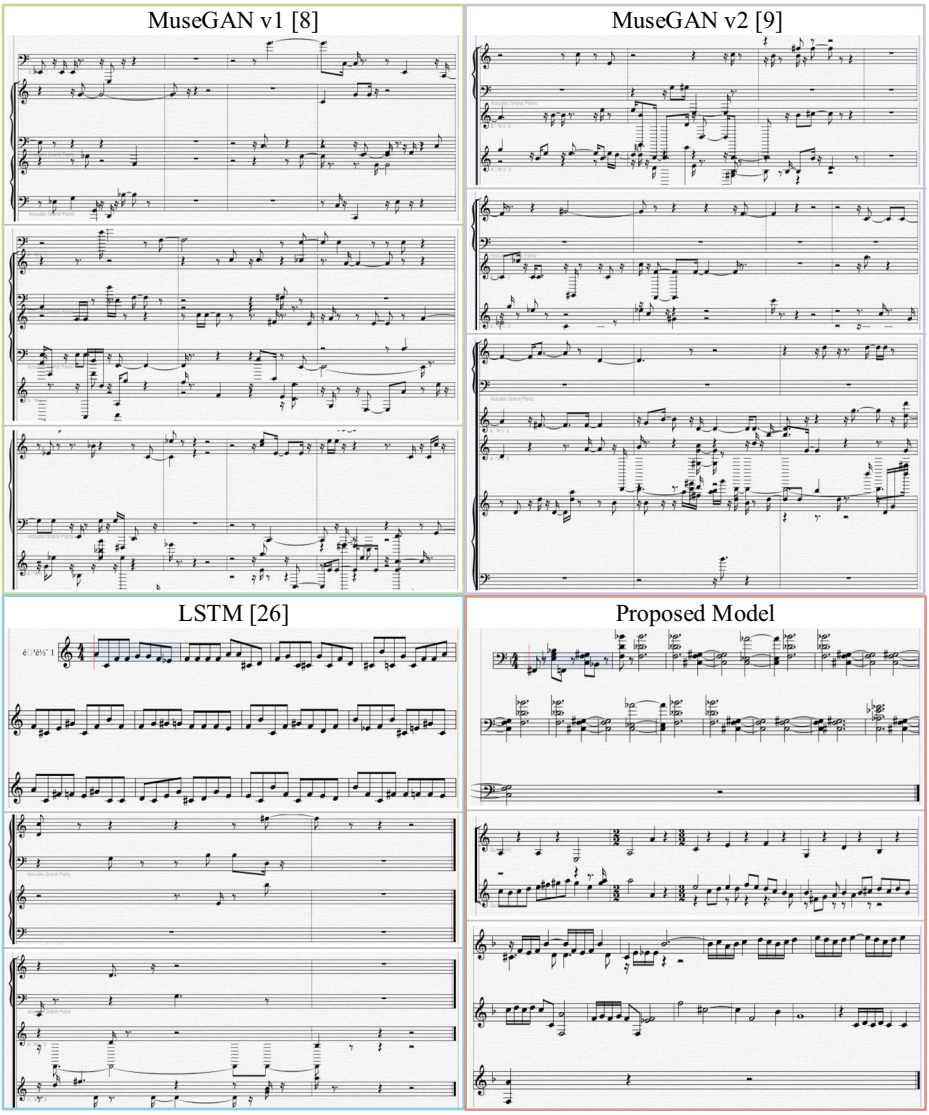

Fig. 12 Visual contrast chart of music generated by the proposed model and other models

the innovation of neural networks for music generation and will not be limited to the melody or tone in the original music dataset.

3.5 Subjective evaluation

On the basis of the above objective evaluations, this study also conducts a survey to evaluate the quality of the model for music production. In this study, 40 subjects were invited to take a subjective test (20 music majors and 20 non-music majors) in which each participant listened to 20 pieces of music (a total of four models, each producing five pieces of music). After listening to music, subjects rated them on a 10-point scale ranging from 1 (very low) to 10

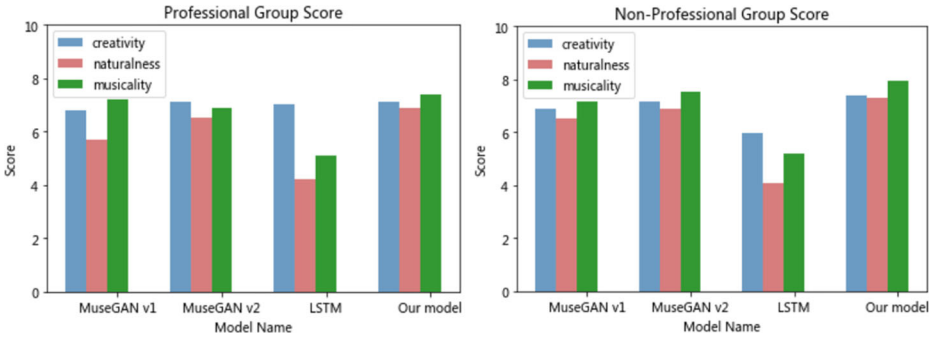

Fig. 13 The average score of the four models in professional group and non-professional group

(very high), based on three criteria: creativity, naturalness and musicality. Figure 13 shows the average score of the four models for the three criteria.

As shown in Fig. 13, the scores of creativity of the four models in the professional group and the non-professional group are very close. In terms of naturalness and musicality, this model has achieved good results, and the average level is higher than the other three models. In the professional group, the average score of this model is 0.26 higher than that of the second, while in the non-professional group, the average score of this model is 0.35 higher than that of the second.

3.6 Limitations

The polyphonic music generation model proposed in this paper has more hidden layers and higher complexity because it contains a generator and a discriminator, adds the MDP mechanism, and integrates MCTS ideas. Therefore, this model requires more training time in exchange for generating quality music. The current research has also only focused on generating relatively short phrases or paragraphs. As the sequence continues to grow beyond what is considered here, the generated music will still become more monotonous and random. This study also relies on changing the dataset of the input model for the style change of music production. In future studies, the team will incorporate the style migration module to improve this issue. In addition, with the increase of training iterations, it is difficult to guarantee that the generated results will not fit the existing music in the training set. This kind of infringement risk is a problem that must be solved in the future.

4 Conclusions

Based on the WGAN model, a novel polyphonic music generation method using MDP [21] and MCTS [4] is proposed to produce higher quality generated music. This model effectively solves the problem in which the generator produces the same note as the music sequence increases, which destroys music continuity. Adding MDP and MCTS to the model prevents the generated music from being constrained to the melody and tone in the original dataset, making the generated music more original. At the same time, the research team has built a new dataset, in which all data files are in MIDI format. In addition, the polyphonic music model uses a GPU for training, and the team plans to collect more types of polyphonic music of various styles so that the model can generate better music in subsequent training.

Acknowledgements This work was supported by the Guangdong Provincial Key Platform and Major Scientific Research Projects under Grant 2018GXJK138.

Author contributions Conceptualization, Huang, W.K. and Xue, Y.H.; methodology, Xue, Y.H. and Xu, Z.F; software, Xue, Y.H.; validation, Huang, W.K. and Xue, Y.H.; formal analysis, Xue, Y.H. and Xu, Z.F; investigation, Huang, W.K. and Xue, Y.H.; resources, Huang, W.K. and Xue, Y.H.; data curation, Huang, W.K. and Xue, Y.H.; writing—original draft preparation, Xue, Y.H. and Xu, Z.F; writing—review and editing, Huang, W.K. and Xu, Z.F.; visualization, Xue, Y.H. and Peng, G.L.; supervision, Huang, W.K. and Xu, Z.F.; project administration, Huang, W.K. and Wu, Y.; funding acquisition, Huang, W.K. All authors have read and agreed to the published version of the manuscript.

Declarations

Conflict of interest The authors declare no competing interests.

References

1. Agarwal S, Saxena V, Singal V, Aggarwal S (2018) LSTM based music generation with dataset preprocessing and reconstruction techniques
2. Arjovsky M, Chintala S, Bottou L (2017) Wasserstein GAN. arXiv, 1701.07875. [Online]. Available: https://arxiv.org/abs/1701.07875. Accessed 6 Dec 2017
3. Bi CK et al (2019) Evacuation route recommendation using auto-encoder and Markov decision process. Appl Soft Comput 84. https://doi.org/10.1016/j.asoc.2019.105741
4. Browne CB et al (2012) A survey of Monte Carlo tree search methods. IEEE Trans Comput Intell AI Games 4:1–43. https://doi.org/10.1109/tciaig.2012.2186810
5. Chen JN, Zhang C, Luo JT, Xie JF, Wan Y (2020) Driving maneuvers prediction based autonomous driving control by deep monte carlo tree search. IEEE Trans Veh Technol 69:7146–7158. https://doi.org/10.1109/tvt.2020.2991584
6. Conklin D, Gasser M, Oertl S (2018) Creative chord sequence generation for electronic dance music. Appl Sci-Basel 8. https://doi.org/10.3390/app8091704
7. Creswell A et al (2018) Generative adversarial networks an overview. IEEE Signal Process Mag 35:53–65. https://doi.org/10.1109/msp.2017.2765202
8. Dean RT, Forth J (2020) Towards a deep improviser: a prototype deep learning post-tonal free music generator. Neural Comput Appl 32:969–979. https://doi.org/10.1007/s00521-018-3765-x
9. Dong HW, Yang YH (2018) Convolutional generative adversarial networks with binary neurons for polyphonic music generation. Presented at the 19th International Society for Music Information Retrieval Conference (ISMIR), Paris, France, Sept. 23–27
10. Dong HW, Hsiao WY, Yang LC (2017) MuseGAN: demonstration of a convolutional GAN based model for generating multi-track piano-rolls. Presented at the 18th International Society for Music Information Retrieval Conference (ISMIR), Suzhou, China, 23–27
11. Dong HW, Hsiao WY, Yang LC (2017) MuseGAN: multi-track sequential Generative Adversarial Networks for symbolic music generation and accompaniment. arXiv, 1709.06298. [Online]. Available: https://arxiv.org/abs/1709.06298. Accessed 24 Nov 2017
12. Goienetxea I, Mendialdua I, Rodriguez I, Sierra B (2019)Statistics-based music generation approach considering both rhythm and melody coherence. IEEE Access 7:183365–183382. https://doi.org/10.1109/access.2019.2959696
13. Goodfellow IJ et al (2014) Generative adversarial networks. Adv Neural Inf Process Syst 3:2672–2680
14. Hadjeres G, Nielsen F (2020) Anticipation-RNN: enforcing unary constraints in sequence generation, with application to interactive music generation. Neural Comput Appl 32:995–1005. https://doi.org/10.1007/s00521-018-3868-4
15. Herremans D, Chew E, Morpheu S (2019) Generating structured music with constrained patterns and tension. IEEE Trans Affect Comput 10:510–523. https://doi.org/10.1109/taffc.2017.2737984
16. Ioffe S, Szegedy C (2015) Batch normalization: accelerating deep network training by reducing internal covariate sift. arXiv, 1502.03167. [Online]. Available: https://arxiv.org/abs/1502.03167. Accessed 2 Mar 2015

17. Juan LI, Mingquan Z (2011) Music database construction based on MIDI melody feature extraction. Com Eng App 47(26):124–128
18. Lewis D, Schapire R, Callan J (1996) Training algorithms for linear text classifiers. ACM SIGIR-96, pp 298–306
19. Link: https://pan.baidu.com/s/13VGjGvYCZ8gid0KnVXOZEA, Extraction code:50zj
20. Liu JS, Chen R (1998) Sequential Monte Carlo methods for dynamic systems. J Am Stat Assoc 93:1032–1044. https://doi.org/10.2307/2669847
21. Lovejoy WS (1991) A survey of algorithmic methods for partially, observed Markov decision processes. Ann Oper Res 28:47–65. https://doi.org/10.1007/bf02055574
22. Ma JY, Yu W, Liang PW, Li C, Jiang JJ (2019) FusionGAN: A generative adversarial network for infrared and visible image fusion. Inform Fusion 48:11–26. https://doi.org/10.1016/j.inffus.2018.09.004
23. Mangal S, Modak R, Joshi P (2019) LSTM based music generation system. arXiv, 1908.01080. [Online]. Available: https://arxiv.org/abs/1908.01080. Accessed 2 Aug 2019
24. Mao HH, Shin T, Cottrell GW, IEEE (2018) In IEEE 12th International Conference on Semantic Computing IEEE International Conference on Semantic Computing, 377–382
25. Mo F, Wang X, Li S, Qian H (2018) A music generation model for robotic composers. IEEE International Conference on Robotics and Biomimetics 18511690. https://doi.org/10.1109/ROBIO.2018.8665078
26. Opitz J, Burst S (2021) Macro F1 and micro F1. arXiv, 1911.03347. [Online]. Available: https://arxiv.org/abs/1911.03347. Accessed 8 Feb 2021
27. Parras J, Zazo S (2019) Learning attack mechanisms in Wireless Sensor Networks using Markov Decision Processes. Expert Syst Appl 122:376–387. https://doi.org/10.1016/j.eswa.2019.01.023
28. Polo A, Sevillano X (2019) Musical vision: an interactive bio-inspired sonification tool to convert images into music. J Multimodal User Interfaces 13:231–243. https://doi.org/10.1007/s12193-018-0280-4
29. Sehnke F et al (2010)Parameter-exploring policy gradients. Neural Netw 23:551–559. https://doi.org/10.1016/j.neunet.2009.12.004
30. Sironi CF, Liu JL, Winands MHM (2020)Self-adaptive Monte Carlo tree search in general game playing. IEEE Trans Games 12:132–144. https://doi.org/10.1109/tg.2018.2884768
31. Whorley RP, Conklin D (2016) Music generation from statistical models of harmony. J New Music Res 45:160–183. https://doi.org/10.1080/09298215.2016.1173708
32. Xie Y, Franz E, Chu MY, Thuerey N (2018) tempoGAN: A temporally coherent, volumetric GAN for super-resolution fluid flow. ACM Trans Graphics 37. https://doi.org/10.1145/3197517.3201304
33. Yang G et al (2018) Deep de-aliasing generative adversarial networks for fast compressed sensing MRI reconstruction. IEEE Trans Med Imaging 37:1310–1321. https://doi.org/10.1109/tmi.2017.2785879
34. Zhang H et al (2019) StackGAN plus plus: Realistic image synthesis with stacked generative adversarial networks. IEEE Trans Pattern Anal Mach Intell 41:1947–1962. https://doi.org/10.1109/tpami.2018.2856256
35. Zhang YZ, Yao KJ, Zhang JD, Jiang F, Warren MA (2020) New Markov decision process based behavioral prediction system for airborne crews. IEEE Access 8:28021–28032. https://doi.org/10.1109/access.2019.2961239