ISSN 2616-5775 Vol. 5, Issue 4: 79-86, DOI: 10.25236/AJCIS.2022.050415

Research and Implementation of Automatic Composition System Based on ACMN

Han Chunlei, Yu Jinming

School of Information Science and Technology, Donghua University, Shanghai, China
Email:hcleh11@163.com

Abstract: *The use of artificial neural network to compose music is a new computer composition method proposed in recent years. The advantage of this method is that it can process large-scale data in a relatively short time, greatly reducing the preparation work before computer composition and improving composition efficiency. However, there are still many problems that have not made obvious breakthroughs. For example, the yield is low, and a considerable proportion of the generated works have no appreciation value. They are completely random patchwork of notes and chords in the time dimension; the music produced is not pleasant, the melody is too simple and so on. In order to make the songs generated by the composition system more in line with the rules of music theory, this paper proposes a music generation model ACMN (Actor-Critic Music Network) based on the policy gradient method, which is characterized by the combination of reinforcement learning technology and neural network. Experiments show that, compared with the composition model without reinforcement learning technology, the ACMN model has more outstanding performance in terms of repetition rate and pleasantness.*

Keywords: *Artificial Neural Networks, Policy Gradients, Reinforcement Learning, ACMN*

1. Introduction

Automatic composition (or algorithmic composition) is an attempt to use a formal process to minimize human involvement in the use of computers to create music [1]. One of the meanings of automatic composition is to minimize human participation, lower the threshold and cost of music creation, and make it possible for us ordinary people to be called "composers". Before the rise of artificial neural networks, computer composition was mainly realized by Markov chain method [2], genetic algorithm [3] and other methods, and they have achieved excellent results in practice, but there are also obvious shortcomings: when the number of samples is relatively large and the structure of the music required to be generated is relatively complex, these methods are invalid.

Compared with other composition methods, the advantage of the neural network method is that it can process large-scale data in a short time, and can also greatly reduce the cost of human and material resources. Using neural network to compose music can allow composers to avoid spending a lot of time on preparatory work [4]. For example, it is necessary to manually make an extremely complex state transition table before composing music with Markov chain method. This has also become a job that is almost impossible for humans to do when the number of samples is large. The composition model ACMN proposed in this paper is obtained by a substantial improvement on the basis of the dual LSTM (Long Short-Term Memory) network model. The policy gradient method is one of the important means to solve reinforcement learning problems [5]. The uniqueness of this method is that it regards the policy as a function containing a series of parameters, which is called the policy function. The policy function uses gradients to update parameters until it finds a set of optimal or near-optimal parameters, the optimal policy. The policy gradient method has many advantages that other methods do not have. It is suitable for large-scale reinforcement learning problems, especially when artificial neural networks are widely used. Many traditional reinforcement learning methods cannot be combined with artificial neural networks, but the policy gradient method can do it. And the combination of the two can give full play to their respective advantages and solve problems quickly and efficiently.

2. ACMN model

When the scale of the action space is relatively large or the agent action is continuous, the traditional reinforcement learning method is no longer applicable, because the past learning method is not only inefficient but also difficult to obtain a satisfactory result. This experiment has about 500 songs used to train the model, and the sum of the corresponding states and actions has exceeded 100,000, so the policy gradient method is a very suitable choice. Figure 1 is the general workflow of the ACMN model.

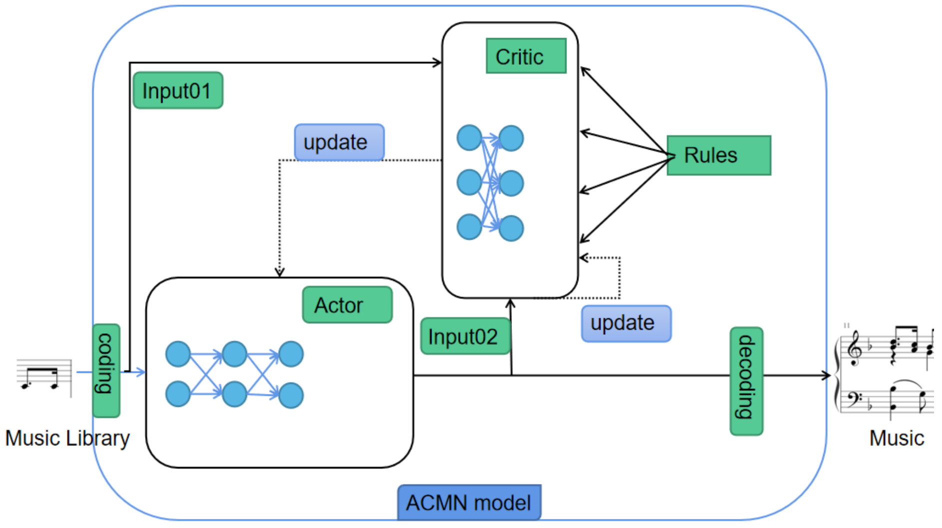

Figure 1: The workflow of the ACMN model.

It can be seen from the figure that the model is mainly composed of two networks, Actor (policy network) and Critic (evaluation network). The reason why the model includes these two networks is related to the use of the AC algorithm to update the parameters [6]. In this model, the input music segment is equivalent to the state of the agent, and the output note or chord is equivalent to the agent's action.

The biggest difference between the policy gradient method and other methods is that the policy in this method is a function (network) containing parameters. Before the policy network can function formally, we need to train its parameters. The strategy in other methods, without parameters, is a probability distribution based on state and action, and its probability distribution needs to be updated according to the value of the action, and the update efficiency is low. This is an important reason for choosing the gradient strategy method.

3. Actor and Critic Design

3.1 Actor Network

The hidden layer of Actor is a double-layer LSTM network, which plays the role of generating notes and chords in this model. Its specific composition is shown in Figure 2.

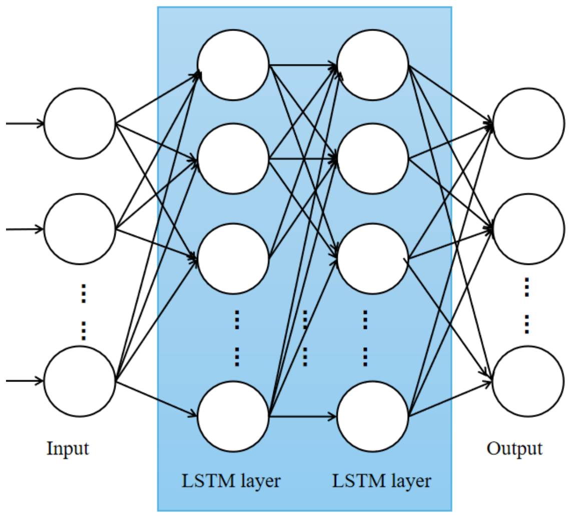

Figure 2: The construction of Actor Network.

LSTM is one of the variants of RNN, which has better memory ability than the latter. It is good at dealing with long-term dependent time series and can effectively avoid the problem of gradient disappearance [7]. The hidden layer of Actor contains 256 neurons in each layer, and the number of neurons in the input layer is equal to the length of the artificially set input music clip. This experiment uses a music clip composed of 64 notes and chords as the input of the model and a note or chord as the output.

Actor networks are equivalent to the policy π in reinforcement learning. According to the definition of policy, π should be the probability distribution based on the set of actions in a certain state. Although the state transition probability mentioned here does not directly indicate the specific probability like the state transition table shows, it exists implicitly. When a certain state is input, the probability that the system will select a certain action is determined by the parameters of the Actor network. Unlike other methods, the policy gradient method does not require complex and labor-intensive state transition tables. Here, the neural network is used to replace the state transition table, and the problem will be greatly simplified.

3.2 Critic Network

The Critic network is composed of different scales of Full Connect Neural Network (FCNN), which plays an evaluation role and is used to evaluate the quality of the policy network. Its composition is shown in Figure 3.

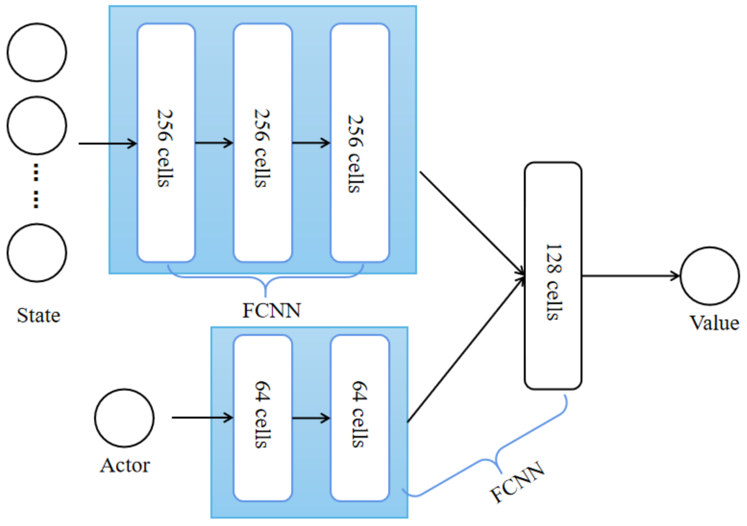

Figure 3: The composition of Critic Network.

The Critic network mainly consists of two sub-networks: the network that receives the state and the network that receives the action. The input of the network receiving the state is the same as the input of the policy network, and its hidden layer consists of three layers of fully connected network, each layer contains 256 neurons. The input of the network that receives the action is the output of the policy network. Since the Actor network only generates one note or chord at a time, the number of inputs to the network that receives the action is 1. The hidden layer of the network that receives the action is a two-layer fully connected neural network with 64 neurons in each layer.

The Critic network receives the encoded notes and chords, but does not generate notes or chords. Its input has two sources, namely the input of the Actor network and its output. The output of Critic is a numerical value that represents the value of the action, and its magnitude reflects the performance of the Actor network. The larger the value, the better the strategy is in line with the designer's expectations, and the less in line with the contrary.

4. Dynamics of the environment

The environment corresponding to this experiment only provides agents with immediate rewards after implementing a certain action, and no other information is provided. The reward mechanism of the environment is as follows.

To create a piece of music, you must first determine its range. If it exceeds a certain range, the sweetness of the piece will decrease. The corresponding limit can be expressed as:

$$R_1(a \mid s_{t-1}) = \begin{cases} 0.2, & a_t \in [A_{\min}, A_{\max}] \\ -0.5, & \text{other} \end{cases} \quad (1)$$

The above formula shows that if the pitch of the note is within the specified range, the environment will reward the agent, and if it is not within the range, the environment will punish it. In order to make the melody of the generated song proceed relatively slowly and avoid the phenomenon of large oscillation of the melody, the following restrictions are imposed on the trend of pitch:

$$R_2(a \mid s_{t-1}) = \begin{cases} -0.6, & 24 \le \sum_{i=t-2}^{t} |a_i - a_{i-1}| \\ -0.5, & 16 \le \sum_{i=t-1}^{t} |a_i - a_{i-1}| \\ -0.3, & |a_t - a_{t-1}| \ge 8 \\ 0.3, & \text{other} \end{cases} \quad (2)$$

Equation (2) shows that if the interval between any two adjacent notes in the tune generated by the system exceeds 7 whole tones, the environment will penalize the policy network. In addition, the policy network is also penalized when the sum of the intervals between adjacent notes in a measure exceeds a specified value, that is, when the melody has a continuous rise or fall or a large up and down oscillation. Then, in order to avoid the phenomenon that the system continuously generates the same note or chord, which seriously affects the quality of the song, the model is restricted as follows:

$$R_3(a \mid s_{t-1}) = \begin{cases} -0.8, & a_t = a_{t-1} = a_{t-2} = a_{t-3} \\ 0.1, & \text{other} \end{cases} \quad (3)$$

The above equation shows that the policy network is penalized for generating four identical notes or chords in a row. Finally, in the actual music creation process, composers often set a median line according to the approximate pitch distribution of the new song [8], which is beneficial to improve the efficiency of music creation. Usually, the tone in the chord is selected as the midline, and here, the difference between the midline and the midline is selected within 7 whole-tone range to arrange the music. Let the median line at time t be Mt (Mt is randomly one of m1, m2, m3, m4), the corresponding expression is:

$$R_4(a \mid s) = \begin{cases} 0.4, & |a_t - M_t| \le 7 \\ -0.1, & \text{other} \end{cases} \quad (4)$$

5. Parameter update algorithm

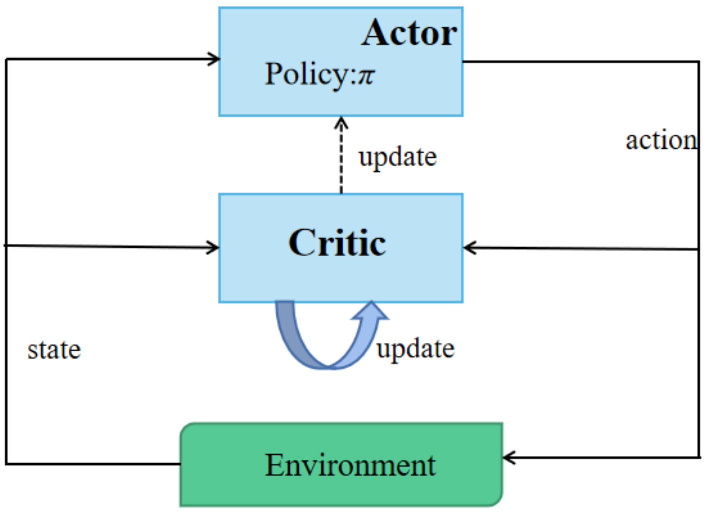

Figure 4: The working principle of the AC algorithm.

When the ACMN model updates its own parameters, it uses the Actor-Critic algorithm (AC algorithm for short). The reason why it is called the AC algorithm is that the framework corresponding to the algorithm includes two parts, Critic and Actor. The functions correspond to their respective names with Actor acting as "executor" and Critic acting as "judge". Figure 4 shows the working principle of the AC algorithm.

Actor networks are equivalent to policy functions that guide the action of agents. Critic network is equivalent to state-action value function, which is used to evaluate the actor's quality and guide its subsequent action [9]. If the Actor is given a lower score, the Critic will guide the Actor's network parameters to update in the direction of a higher score, and the training process will not end until the score converges to a higher value.

The AC algorithm includes two basic processes, one is data sampling, and the other is policy update. The sampling of the data is critical because it determines the performance of the algorithm. This experiment uses the sub-algorithm TD (0) of the Temporal-Difference (TD) sampling method, and the corresponding AC algorithm is TD (0)-AC. The sampling characteristics of the AC-TD (0) algorithm are: in addition to the immediate reward of the current state, the value of updating the current state only needs to know the value of the next state of the current state, and does not need to know the value of the state in the second step and beyond. In addition, there may be multiple states to choose from after a certain state, but only the value of one of the states needs to be obtained when sampling.

The objective function of reinforcement learning involved in this experiment is the expectation of cumulative reward, and its expression is:

$$J(\theta) = \mathrm{E}[r] = \sum_{s\in S} d(s) \sum_{a\in A} \pi(s,a) R \quad (5)$$

Where d(s) is the static probability distribution based on the Markov chain corresponding to the policy π, and R is the immediate reward for an agent to perform an action. In general, the neural network updates the parameters through the gradient descent method, but here, we want the objective function to be as big as possible, so the gradient ascent method is used to update the parameters.

$$\nabla_\theta J(\theta) = \mathrm{E}[\nabla_\theta \log \pi(s,a) Q_w(s,a)] \quad (6)$$

$$\Delta\theta = \alpha \nabla_\theta J(\theta) \quad (7)$$

Equation (6) is the gradient of the objective function, where *Qw(s, a)* is the state-action value function, and w is the parameter of evaluating the network Critic. The parameter θ of the policy network is updated by formula (7). In addition, in order to make the estimated value of the gradient closer to the true value, here we use $\delta t = Q1 - V(s,w)$ to replace *Qw(s,a)*. *Q1* is the currently estimated state-action value function, and *V(s,w)* is the parameterized state value function. During the experiment, the parameters are updated every time a sample is taken. The parameter update rules of the strategy network and the evaluation network are as follows:

$$\theta \leftarrow \theta + \alpha^\theta \delta_t \nabla_\theta \log \pi_\theta(s,a) \quad (8)$$

$$w \leftarrow w + \alpha^w \delta_t \nabla_w (s,w) \quad (9)$$

The loss function of the evaluation network is:

$$loss = \left[Q_1 - V(s,w)\right]^2 \quad (10)$$

6. Experimental results and analysis

In this experiment, nearly 500 classical songs with similar melody were collected as samples for training the model. The songs in the sample library and the songs generated by the system were all in 4/4 time. After training, the composition model will not compose music immediately, but also need to manually deliver an initial piece of music. After that, no manual intervention is required, and the system will automatically input and output. The model designed in this paper is to predict (generate) a note or chord by taking a musical fragment composed of 64 notes and chords as input. When the next note or chord is generated, the original piece of music discards the first note or chord, and the remaining pieces and the next note or chord form a new piece of music as a new input, and so on until the end. The final piece where all the outputs are stitched together is the complete song generated by one trial.

In this experiment, ACMN is compared with three other composition models in multiple aspects, which are the dual LSTM model, the VAE-GAN model [10] and the Melody_LSTM model. In order to make the experimental results more objective, the irrelevant variables should be guaranteed to be the same as much as possible during the experimental process, otherwise the experimental results will be unconvincing. It should also be noted that when training the network, we should appropriately adjust the parameters of the model and the number of training times to prevent overfitting, because overfitting means duplicating the original piece of music, which is contrary to our original intention. We hope that the model can generate a large number of songs with the same style but different melodies, otherwise the automatic composition will lose its meaning.

6.1 Experimental results

To make experimental conclusions more reliable and general, it is necessary to repeat the same experiment. Because only in this way can the contingency in the practice process be eliminated, and the final conclusion can better reflect the objective facts. Therefore, this experiment had each model compose a hundred times, recording the results they produced for each trial separately. Through the statistics and comparison of these results, the conclusion of this experiment is finally drawn.

Since there are many songs generated by the ACMN model in this experiment, only the results of three experiments are randomly selected here, and their staves are shown in Figures 5, 6, and 7.

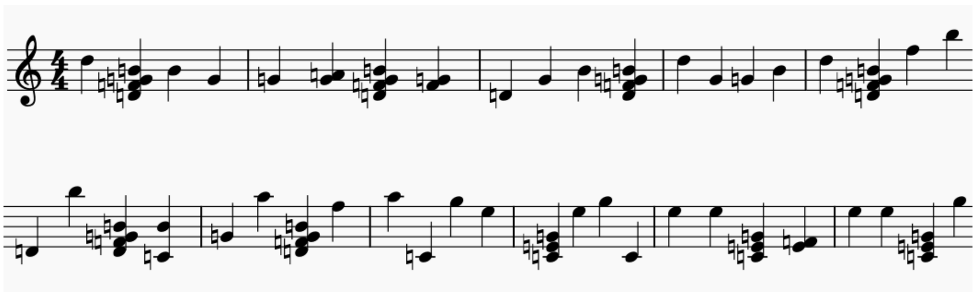

Figure 5: The staves of the tunes generated by ACMN (a)

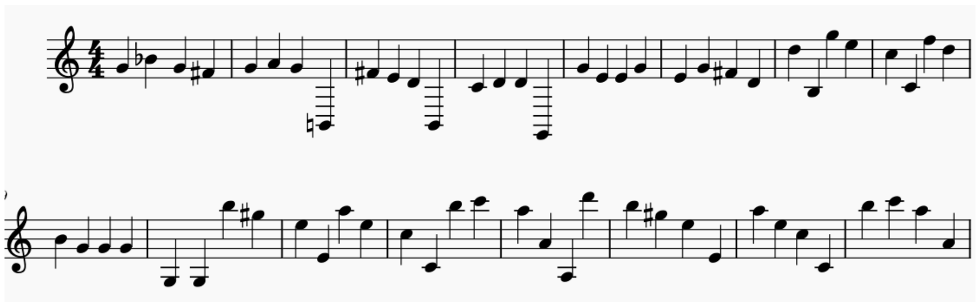

Figure 6: The staves of the tunes generated by ACMN (b)

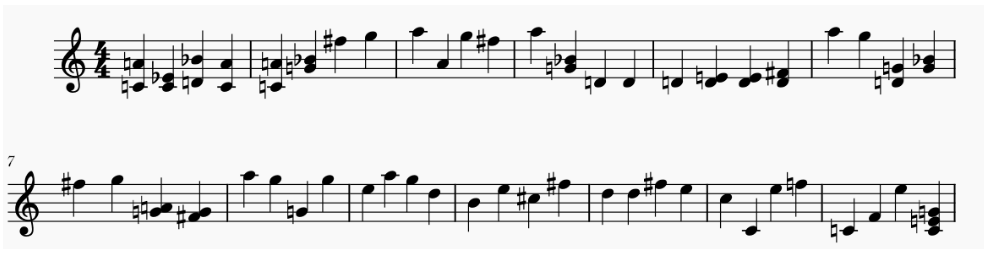

Figure 7: The staves of the tunes generated by ACMN (c)

6.2 Objective assessment

On the basis of one hundred trials, this paper compares ACMN with dual LSTM, VAE-GAN and Melody_LSTM in multiple aspects. The comparison results are shown in Table 1.

Table 1: Performance comparison of each model.

Model	Similarity	Repetition Rate	Yield
Acmn	18.76%	42.23%	43%
Vae-Gan	12.53%	47.35%	47%
Dual Lstm	8.67%	63.34%	36%
Melody_LSTM	9.26%	65.37%	36%

The three items of data shown in Table 1 are the averages of the 100-time composition results of each model. It should be noted that continuous repetition refers to the situation where four or more identical notes or chords are continuously generated; repetition rate refers to the proportion of continuously repeated notes or chords in a certain piece of music generated, in this case is the first hundred time nodes. We stipulate that the composition is successful when the total number of non-consecutively repeated notes and chords in the music segment generated by the model is not less than twenty. Regarding the calculation of the repetition rate, it is defined as the average of the repetition rate of all the songs generated under the premise of successful composition (removing one highest value and one lowest value). It can be seen from the table that the repetition rate of the songs generated by the ACMN model is the lowest, while the repetition rate of the songs generated by the Melody_LSTM model is higher. Similarity refers to the degree of similarity between two kinds of audio, here specifically refers to the degree of similarity between the generated tune and the original sample, and its size is measured by a software called "Sound-Similar Free". It can be seen from the table that the ACMN model has the strongest "inheritance" ability to classical music, followed by VAE-GAN. The percentage of successful composition times in the total composition times is called the yield. In one hundred compositions, the ACMN model succeeded 44 times, while the VAE-GAN model succeeded 48 times, so the yield of ACMN was slightly lower than that of VAE-GAN. The yields for the other two models were relatively low at 35% and 33%, respectively.

6.3 Subjective assessment

The quality of a piece of music ultimately depends on the audience's evaluation of it, and the computer currently does not have the ability to evaluate the deep emotion and art contained in the music. In order to make the results of the manual evaluation relatively objective, we randomly selected three of the works generated by these four models, and then 20 people were invited to score them (out of 10 points), and the final score was the average of the scores scored by the 20 people. The results of the evaluation are shown in Table 2.

Table 2: Manual evaluation results.

Model	Sense of rhythm	Fluency	Pleasantness
ACMN	**7.46**	**6.87**	**7.14**
VAE-GAN	7.23	6.54	6.98
Dual LSTM	6.87	6.33	6.56
Melody_LSTM	6.88	6.55	6.42

It can be seen from Table 2 that the songs generated by the ACMN model outperform the other three models in terms of rhythm, fluency and pleasantness.

7. Conclusion

This paper proposes a music generation model ACMN based on policy gradient method. The characteristics of this model apply reinforcement learning technology to artificial neural network. The ACMN is compared with three other composition models in several aspects through experiments and human evaluation. The results show that compared with the composition model without reinforcement learning technology, although the performance of ACMN is mediocre in yield, outperforms other composition models in most indicators, such as similarity and repetition rate.

References

[1] Alpen A. Techniques for algorithmic composition of music. 1995.
[2] Basset BA, Neto JJ. A stochastic musical composer based on adaptive algorithms. 1999.
[3] Zhang Yingli, Liu Hong, Ma Jingang. Research on Genetic Algorithm Composition System [J].

Information Technology and Informatization, 2005(5):106-108.

[4] Tian Mei, Huang Zhixing, Zhang Yougang. Artificial Intelligence Technology in Algorithmic Composition [J]. Journal of Sichuan Institute of Education, 2006, 22(z2): 165-166, 168.

[5] Gao Ting. Research and implementation of music melody generation algorithm based on deep learning [D]. Beijing University of Posts and Telecommunications, 2021.

[6] Jing Li. Research on spectrum allocation and AoI of Internet of Vehicles based on reinforcement learning [D]. Chongqing: Chongqing University, 2020.

[7] Hochreiter S and Schmidhuber J. Long short-term memory. [J]. Neural computation, 1997, 9(8): 1735-80.

[8] Bai Yong, Tie Yun, Jin Cong, et al. Research on intelligent composition based on reinforcement learning [J]. Artificial Intelligence, 2020(2):47-56.

[9] Ivo Grondman et al. A Survey of Actor-Critic Reinforcement Learning: Standard and Natural Policy Gradients [J]. IEEE Transactions on Human-Machine Systems, 2012, 42(6): 1291-1307.

[10] Mohammad Akbari, Jie Liang. Semi-Recurrent CNN-Based VAE-GAN for Sequential Data Generation. IEEE International Conference on Acoustics, Speech and Signal Processing, 2018.2321-2325.