



OPEN

# Automatic composition of Guzheng (Chinese Zither) music using long short-term memory network (LSTM) and reinforcement learning (RL)

Shuling Chen<sup>1,2</sup>, Yong Zhong<sup>1,2</sup>✉ & Ruxu Du<sup>1</sup>

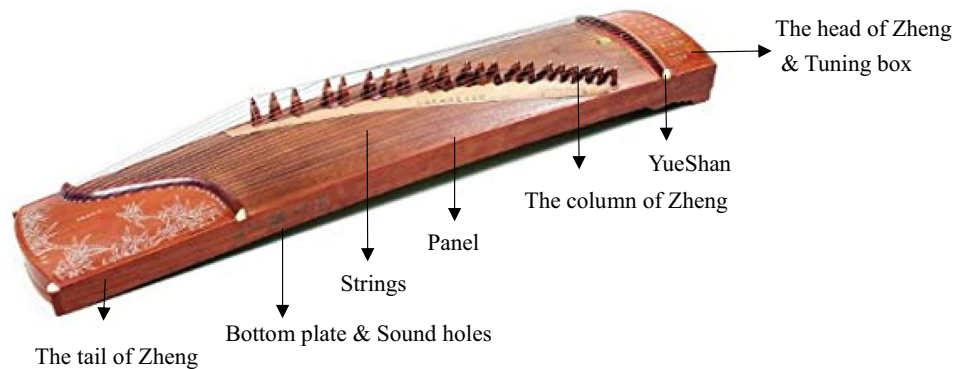
In recent years, with the advance of Artificial Intelligence, automatic music composition has been demonstrated. However, there are many music genres and music instruments. For a same piece of music, different music instruments would produce different effects. Invented some 2500 years ago, Guzheng is one of the oldest music instruments in China and the world. It has distinct timbres and patterns that cannot be duplicated by other music instruments. Therefore, it is interesting to see whether AI can compose Guzheng music or alike. In this paper we present a method that can automatically compose and play Guzheng music. First, we collect a number of existing Guzheng music pieces and convert them into Music Instrument Digital Interface format. Second, we use these data to train a Long Short-Term Memory (LSTM) network and use the trained network to generate new Guzheng music pieces. Next, we use the Reinforcement Learning to optimize the LSTM network by adding special Guzheng playing techniques. Comparing to the existing AI methods, such as LSTM and Generative Adversary Network, our new method is more effective in capturing the characteristics of Guzheng music. According to the evaluations from skilled Guzheng players and general audiences, our Guzheng music is very close to the real Guzheng music. The presented method can also be used to automatically compose the music of other Chinese music instruments.

Music is a universal art enjoyed by all the people around the world. It is well known that music has different genres and can be played using different music instruments. Music composition has always been considered as creative art that only talented musicians can master. In recent years, the technology of Artificial Intelligence (AI) has been greatly advanced. There have been a number of successful attempts to use AI for music composition. However, the success is rather limited as only a few kinds of music, such as piano music and guitar music, can be generated. Particularly, there has been no reported success in composing traditional Chinese music.

Among various traditional Chinese musical instruments, Guzheng (Chinese Zither) (古筝) is perhaps ranked at the top in popularity. Invented some 2500 years ago<sup>1</sup>, through years of inheritance and innovation, development and improvement, it has its unique music language, playing style, as well as special identity unlike any others. In fact, just hearing Guzheng music, many people around the world would think of China.

Figure 1 shows the picture of a classical Guzheng. Its body is a hollow box made of sycamore wood. Its strings are placed on the top face while the sound holes are at the bottom (not shown in the figure). From the structure point of view, Guzheng consists of 5 parts: (1) the head of the Guzheng and tuning box, (2) Yue Shan (岳山) on which the strings are mounted, (3) the column of Guzheng that acted as the bridge for the strings, (4) the panel, on which the strings are crossed, and (5) the tail of Guzheng. According to the historical records, the original Guzheng had only 5 strings. In Three-Kingdom period (220–280), it had 12 strings. In Tang Dynasty (618–907), it added to 13 strings, and in Ming Dynasty (1368–1644) and Qing Dynasty (1636–1912) it had 16 strings. In the modern era, it was sublimated to modern musical instrument making method and gradually increased to 18 strings, 21 strings and even 25 strings. Presently, the design of Guzheng is standardized with 21 strings and

<sup>1</sup>S. M. Wu School of Intelligent Engineering, South China University of Technology, No. 777 Xingye Road East, Panyu District, Guangzhou 511442, China. <sup>2</sup>These authors contributed equally: Shuling Chen and Yong Zhong. ✉email: zhongyong@scut.edu.cn



**Figure 1.** Guzheng, a traditional Chinese music instrument (Free of copyright).

a fixed length of 1.63 m long. Its tone is in pentatonic scale with a range of three octaves. The mode is mostly G or D. Also, players need to wear tortoise's bill armor pieces (i.e., false fingernail) to play.

There are a number of special skills for playing Guzheng. The right hand uses the thumb, index finger, middle finger and ring finger to pluck the strings. The fingering methods include Gou (勾), Tuo (托), Pi (劈) and etc. The left hand presses the strings and the fingering techniques include Press (按), Slide (滑), Chatter (揉) and so on. In recent years, new Guzheng playing techniques are also added such as playing the left and right hand alternatively and playing multi-part together. It creates very rich and vibrant music<sup>2–4</sup>.

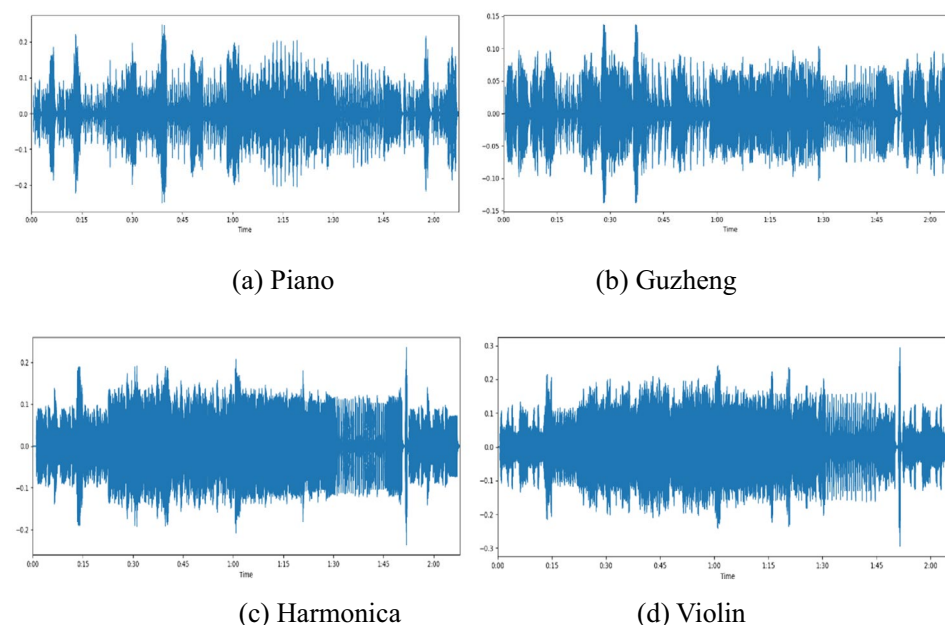
There are at least 200,000 people are playing Guzheng with different skill levels. In 2019, there were over 17,000 people participating the national exam, making it the most popular traditional Chinese music instrument. Indeed, many people view Guzheng as important inheritance and have special cultural value. In recent years, although the effort of Guzheng music composition also increases, few music pieces could not reach the height of the traditional Guzheng music pieces. Therefore, it is wondrous if we can use AI to compose good Guzheng music.

AI based music composition, also referred to as computer music generation, refers to generate a specific sequence of music notes with minimum human intervention. Based on literature survey, the first attempt of computer music generation can be dated back to 1950s<sup>5</sup>. The early pioneers proposed to use Markov model to generate music, but the experiments showed that only fragment musical notes were generated. Later research found that the Markov chain model offers only a stitching process and cannot be called music composition. Consequently, Hidden Markov Model (HMM) were proposed which was able to generate polyphony music for the first time<sup>6</sup>. However, HMM has some limitations, such as its state evolution is linear, it is difficult to learn long-term dependence, and it is incapable of dealing with large data sets.

Neural network is another commonly used method for computer music generation<sup>7</sup>. For example, in<sup>8</sup>, Schuster and Paliwal used bidirectional Recurrent Neural Network (RNN) to generate computer music. However, because of the gradient disappearance of RNN, it is difficult to learn the long-term correlation within a music piece, resulting in fragmented note cycles. In<sup>9</sup>, Hang and Urtasun used Long Short-Term Memory network (LSTM) to generate more complex music. LSTM can memorize long-term information and avoid the aforementioned gradient disappearance problem. Consequently, it makes the generated music pieces more coherent. In<sup>10</sup>, a new LSTM with Gated Recurrent Unit (GRU) was used obtaining improved results. Subsequently, LSTM is widely used in computer music generation<sup>11–13</sup>, so we pick LSTM to generate Guzheng melody. In<sup>14</sup>, in order to solve the problem of melodic level of music generated by LSTM, an algorithm called Music VAE (Variational Autoencoder) is proposed. In<sup>15</sup>, VAE is used to improve the quality of audio synthesis. The emerging of Generative Adversarial Networks (GAN) algorithm<sup>16</sup> provides a new method for AI based music composition. GAN consists of two networks: the generating network G (Generator) and the discriminant network D (Discriminator), the training achieves a Nash equilibrium, making G capable of generating data with the same distribution as that of the training data and D capable of detecting false data, which is ideal for music composing. The used of GAN for music composition is reported in<sup>17,18</sup>, though, it seems the generated music pieces are still full of random notes.

Reinforcement Learning (RL) is another common method for AI music generation. In<sup>19</sup>, Kotecha used a two-layer bi-axial LSTM with deep reinforcement learning (DQN) aimed for better global coherence. Though, the generated music pieces are rather short and assessed by only one music professional with vague comments. In<sup>20</sup> Jiang and et al. used RL for interactive music generation. Again, the generated music pieces are short and lack of fluency. In<sup>21</sup> Karbasi and et al. used RL to model musical rhythms, however, the rhythms are all quite simple, and the agent seems to be blind to the dynamical behavior of the environment. In<sup>22</sup> Jaques et al. developed a new RL method for fine-tuning the generation models with KL (Kullback Leibler)-control. They used music generation as an application example. Unfortunately, the generated music pieces contain many repeated notes.

In short, there are a number of common problems for automatic music composition, First, it is clear that the brute searching and matching of the existing AI methods could only result in poor imitation. Second, there is a lack of subjective and objective evaluation method for generated music. Given the fact that most of the generated music pieces are rather short and lack of rhythm and fluency, there is still a long way to go.



**Figure 2.** The sound waves of different music instruments when playing a same piece of song with the same volume.

Moreover, no one has tried to use AI to generate Guzheng music. This may be attributed to several challenges. Firstly, it is known that the modern digital music format is MIDI. Currently, the MIDI library contains the timbres of many music instruments, such as piano, violin, guitar etc., but not Guzheng. Consequently, it becomes difficult to generate Guzheng music. One may of course use MP3 format, though, it requires much more training samples and computation loads. Yet, it cannot guarantee the success. Secondly, Guzheng has many unique playing skills, which cannot be readily expressed in MIDI format. Third, simply apply the AI methods for Guzheng music generation may not work. This is because Guzheng music has distinct characteristics created by its unique playing techniques. These playing techniques cannot be further decomposed and hence, mock up. Also, there are few people who know both AI and Guzheng.

In this paper, we present a new method for composing Guzheng music using LSTM and Reinforcement Learning (RL). It consists of three steps: preprocessing, training and music generation, as well as post processing. Accordingly, the rest of the paper is organized as follows. “Pre-processing” section describes the preprocessing steps including a brief review of music, the conversion of the Guzheng music to MIDI format, as well as the extraction of chorus. “Training using LSTM and RL” section discusses the core of our method: training using LSTM and Reinforcement Learning (RL). “Post-processing and experiment results” section shows the post-processing step as well as the experiment results. Finally, “Conclusions and future work” section contains conclusions and future work.

## Pre-processing

**A brief review on the elements of music.** There are several different views on the elements of music. From the acoustic point of view, music is made of sounds which can be characterized by its time and frequency characteristics. From the music point of view, though, the definitions vary. The commonly accepted definition is that the basic elements of music include pitch, duration, loudness, timbre, sonic texture and spatial location. Though, the most important elements of music are timbre and melody.

Timbre is the perceived sound quality of a musical note, a sound or a tone. It distinguishes a sound from the others. Timbre can be characterized by the following attributes: the tone (the arrangement of pitch and/or chord), the spectral envelop, the time envelop, the changes of the frequency and spectral envelop, as well as the onset of the sound<sup>23</sup>. Guzheng, like the other music instruments, has its unique timbre. Figure 2 shows the sound wave from Piano, Guzheng, Harmonica, and Violin when playing a same piece of song with the same volume. As expected, Guzheng has unique timbre different to all others.

Melody, also called tune, is a timely succession of music tones. It is a combination of pitch and rhythm. Melody often consists of one or more musical phrases and are usually repeated within the music piece. Guzheng music belongs to traditional Chinese music and has its own melody styles and patterns. In particular, its emphases on artistic conception and cavity rhyme. As shown in <sup>24</sup>, traditional Chinese music uses five tones: C, D, E, G, and H. Additionally, it usually uses single instrument, single rhythm. Though, modern Chinese music is more diversify and uses the standard seven tones. However, its artistic conception and cavity rhyme are difficult to describe.

In order to generate Guzheng music, it is necessary to sort out the timbre of the Guzheng as well as the melodies of the Guzheng music first.



**Figure 3.** Music format conversion.

**Guzheng music data pre-processing.** Music data can be represented in two kinds of format: the audio format (MP3 or AVI) and the MIDI format. In this paper, we use the MIDI format as it is concise and easy to edit. MIDI files contain musical notes, timing, and playing definitions of up to 16 channels. For each channel, it includes channel number, length, volume, strength, etc. Because a MIDI file is a series of instructions, it requires little disk space. More importantly, one can easily add, delete or edit the attributes of a note.

We pick Guzheng music pieces from the popular Guzheng repertoires in the Guzheng training textbooks. These music pieces are in the form of numbered notation. Next, we convert these music pieces to form of staff notation. Then, we use a freeware MuseScore to convert them to MIDI format. This process is illustrated in Fig. 3.

A total of 31 pieces of Guzheng music are used for training. These music pieces range from 30 s to 4 min in length. It shall be pointed out that MIDI has the play definitions of many music instruments, such as piano, violin, guitar and etc., but not Guzheng. From the mechanics point of view, Guzheng is similar to Guitar, which generate sounds with one hand pressing the strings while the other hand toggling the strings. However, Guzheng has 21 strings and can play register from D2–D6 (International phonetic alphabet) while Guitar can only play register from E2–C6. Therefore, we use piano to animate Guzheng, as it can play 88 notes, for music generation. The generated music will be converted to Guzheng music by converting to the Guzheng timbre in post-processing, which will be shown in Sect. 4.

**Extraction of the melodies.** As pointed out in earlier, music is characterized by its melody. Thus, we extract the melodies of the training samples use them to enhance the training. In general, melody may appear in two ways: (1) For simple and short music, its melody is the chorus which replicates itself with minor variations. (2) For complicated and long music, its melody is characterized by its tensed motion and cadence.

For capturing the melodies, we use the similarity matrix. In the MIDI format, the music is scanned at a rate of 10 frames per second, or one frame every 0.1 s. Each frame is represented by a 128-dimensional vector giving 128 pitches in total. The value in each dimension represents the strength of the note in the corresponding pitch. Thus, each frame can be described by a vector,  $v = [x_1, x_2, \dots, x_{128}]$ . Moreover, a piece of music can be described by a matrix  $X = [v_1, v_2, \dots, v_N]^T$  where,  $N$  number of frames. Accordingly, the similarity matrix is a  $(N-1) \times (N-1)$  matrix, whose element is as follows:

$$S_{ij} = \sqrt{(v_i - v_{i+1})^2 + (v_j - v_{j+1})^2} \quad (1)$$

where  $i, j = 1, 2, \dots, N-1$ .

Figure 4 shows the similarity matrix of a piece of music. In the figure, different levels of brightness represent different levels of similarity. It is seen that the ones marked by yellow boxes have a similar pattern and hence, are correspondent to the melody.

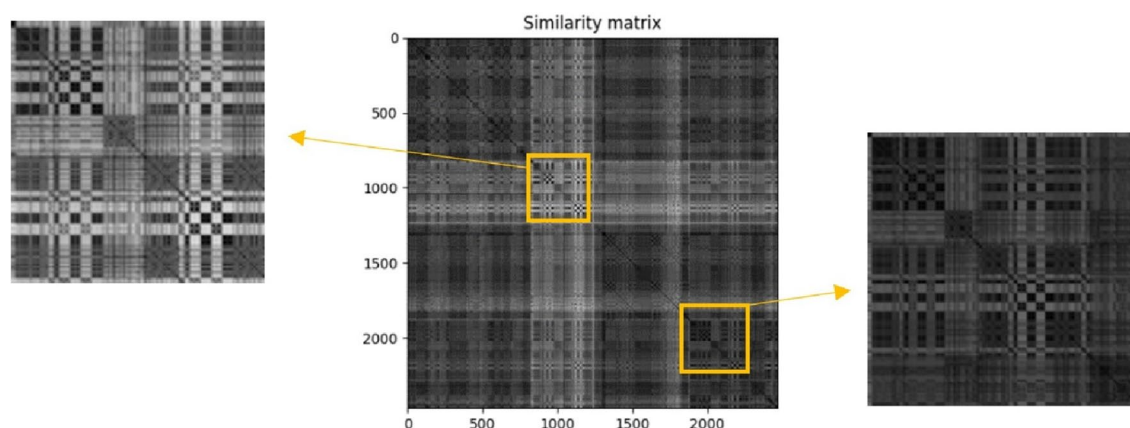
The similarity matrix works fine for finding repeats with minor variations. For repeats with relatively large variations, we can use the similar index defined below:

$$S = 1 - \frac{\|X_i - X_{i+1}\|}{\sqrt{12}} \quad (2)$$

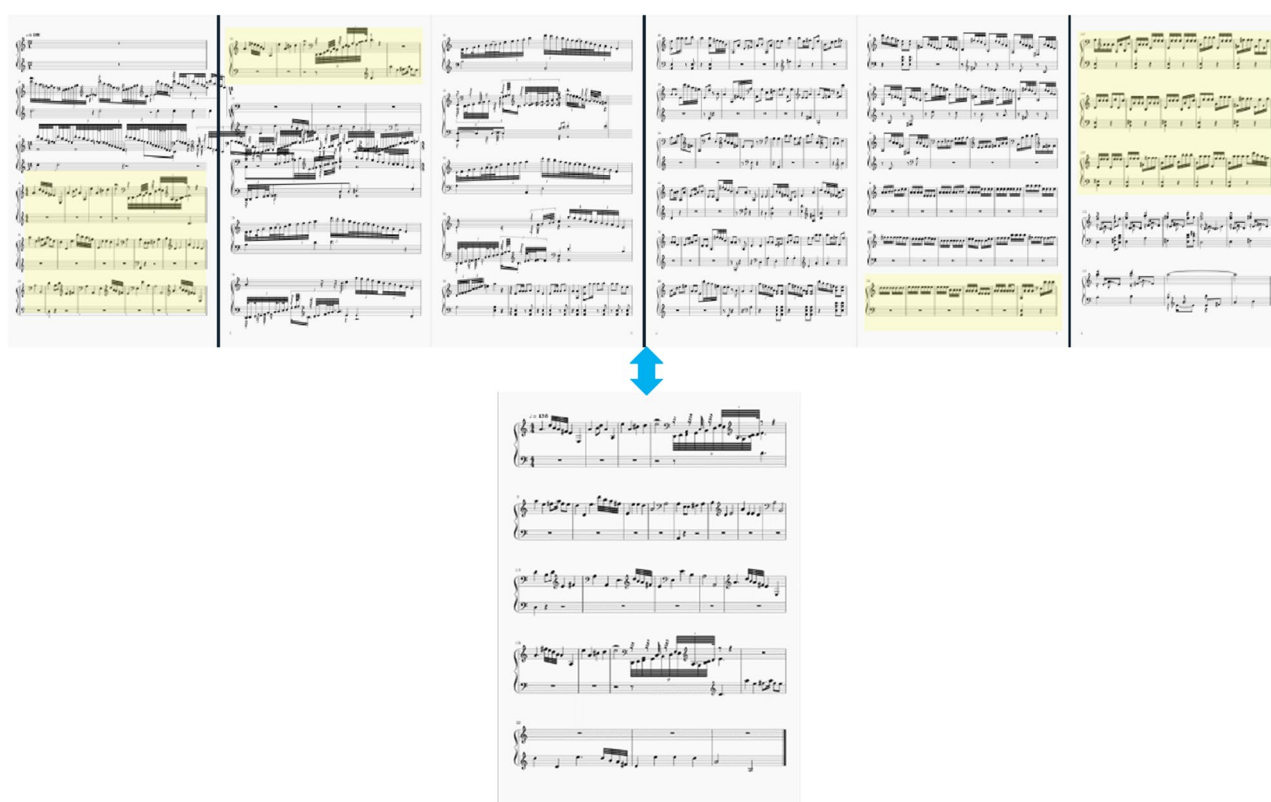
where 12 corresponds to the number of basic notes,  $X_i$  and  $X_{i+1}$  are the matrices corresponding to the two segments in the music piece. If the value of  $S$  is high, then the two segments are similar.

As an example, Fig. 5 illustrates the extraction of a melody.





**Figure 4.** The similarity matrix of a sample music.



**Figure 5.** An example of melody extraction, the melody is marked by yellow.

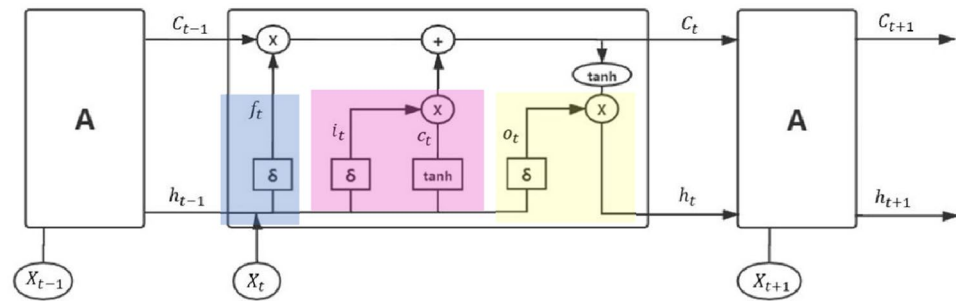
There are Guzheng music pieces that do not have repetitions, such as “Gao shan liu shui (高山流水)” and “Yu zhou chang wan (渔舟唱晚)”. In this case, we will select the segment with high intense and cadence.

### Training using LSTM and RL

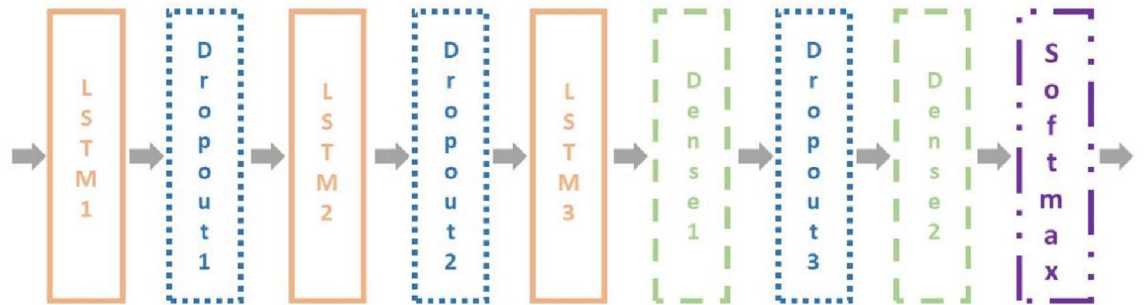
**A brief introduction of LSTM network.** In this paper, we use LSTM network for Guzheng music generation. Figure 6 shows a LSTM unit. It has three inputs: the data input (note vectors),  $X_t$ , as well as the outputs from the previous unit,  $h_{t-1}$  and  $C_{t-1}$ . Here,  $h_{t-1}$  is the output while  $C_{t-1}$  represents how well the output shall be retained. As shown in the figure, the LSTM unit is made of three gates: the forgetting gate (in blue), the input gate (in pink) and the output gate (in yellow).

The forgetting gate is used to control how much information shall be “forgot” from the previous unit and is computed as follows:

$$f_t = \delta(W_f * [h_{t-1}, X_t] + b_f) \quad (3)$$



**Figure 6.** A LSTM unit.



**Figure 7.** Model demonstration.

where  $h_{t-1}$  is the output of the previous unit,  $X_t$  is the input,  $W_f$  is the weight,  $b_f$  is the bias, and  $\delta$  is the Sigmoid function.  $f_t$  times  $C_{t-1}$  determines how much information from the previous unit will retain.

The input gate is used to determine the effect of the inputs. It contains two parts:

$$\begin{aligned} i_t &= \delta(W_i * [h_{t-1}, X_t] + b_i) \\ c_t &= \tanh(W_c * [h_{t-1}, X_t] + b_c) \end{aligned} \quad (4)$$

where  $W_f$  and  $W_i$  are the weights,  $b_i$  and  $b_c$  are the biases.

The output gate is used to determine the information output. Just like the previous unit, it gives two outputs:  $h_t$  is the output and  $C_t$  represents how well  $h_t$  shall be retain. The output  $h_t$  is computed as follows:

$$o_t = \delta(W_o * [h_{t-1}, X_t] + b_o) \quad (5)$$

$$h_t = o_t * \tanh(C_t) \quad (6)$$

where  $W_0$  is the weight and  $b_o$  is the bias. The control  $C_t$  is computed as follows:

$$C_t = f_t * C_{t-1} + i_t * c_t \quad (7)$$

It is interesting to note that when  $f_t \rightarrow 1$  and  $i_t \rightarrow 0$ ,  $C_t \approx C_{t-1}$ , which retains all the information from the previous unit. On the other hand, when  $f_t \rightarrow 0$  and  $i_t \rightarrow 1$ ,  $C_t \approx c_t$ , which retains no information from the previous unit.

In this paper we use 3 LSTM, 2 Dropouts, 2 Denses and 1 Softmax as shown in Fig. 7. In the setting, the LSTM is described by three parameters: the sequence length, the characteristic number of the hidden layer, and the number of weights in the network. The Dropout is used to reduce overfitting by hiding 30% of the neurons. In the setting, it is described by two parameters: the sequence length, the characteristic number of the hidden layer. In the setting, it can be described by three parameters just like LSTM. The Softmax function is used to generate music notes. Given data sets,  $\{h_{j,t}, j = 1, 2, \dots, M, t = 1, 2, \dots, N\}$ , it first computes the probability:

$$P(h_{j,t}) = \frac{e^{h_{j,t}}}{\sum_{j=1}^M e^{h_{j,t}}}, j = 1, 2, \dots, M, t = 1, 2, \dots, N \quad (8)$$

Then, the ones with the highest probability is selected as the generated note:

$$y_t = \arg \max_j P(h_{j,t}), t = 1, 2, \dots, N \quad (9)$$

The training of LSTM is based on the minimization of the cross-entropy function:

Name	Setting parameters
LSTM1	The sequence length = 100; The characteristic number of the hidden layer = 512; The number of weights in the network = 1,052,672;
Dropout1	The sequence length = 100; The characteristic number of the hidden layer = 512;
LSTM2	The sequence length = 100; The characteristic number of the hidden layer = 512; The number of weights in the network = 2,099,200;
Dropout2	The sequence length = 100; The characteristic number of the hidden layer = 512;
LSTM3	The sequence length = 100; The characteristic number of the hidden layer = 512; The number of weights in the network = 2,099,200;
Dense1	The sequence length = 100; The characteristic number of the hidden layer = 256; The number of weights in the network = 131,328
Dropout3	The sequence length = 100; The characteristic number of the hidden layer = 256;
Dense2	The sequence length = 100; The characteristic number of the hidden layer = 138; The number of weights in the network = 35,466;
Softmax	The sequence length = 100; The characteristic number of the hidden layer = 138;

**Table 1.** The network parameters.

$$L_1 = -\frac{1}{M} \sum_{j=1}^M (Y_j \log(Y_j) + (1 - Y_j) \log(1 - Y_j)) \quad (10)$$

where  $Y_j = \frac{1}{128} \sum_{t=1}^{128} (X_{j,t} - Y_{j,t})^2$ .

During the training, we set the learning rate as  $\alpha = 0.4 \times 10^{-3}$ . Also, considering the lack of contextual information in the first 6 notes of each musical piece, these notes are not be used.

In the setting, it can be described by the sequence length and the characteristic number of the hidden layer. In summary, Table 1 shows the network parameters.

**An introduction of Guzheng playing techniques.** Guzheng is a 2000 year old music instrument. Throughout the history, many playing techniques were added and modified. Currently, there are many complicated playing techniques, each has its own characteristics. It is impossible to list all the playing techniques, as new ones are continuously being invented and old ones are being modified. We therefore focus on the typical playing techniques instead. In general, the playing techniques can be divided into two categories: the left-hand playing techniques and the right-hand playing techniques.

The left-hand playing techniques imply the use of left-hand on the left side of the bridge. It changes the tension of the string, mainly by pressing the string, with the middle finger, ring finger or index finger. Its basic task is to decorate the play of the right hand, so that the tone is richer. Table 2 shows three typical left-hand playing techniques with their playing methods, usual location in a music piece and a sample timbre.

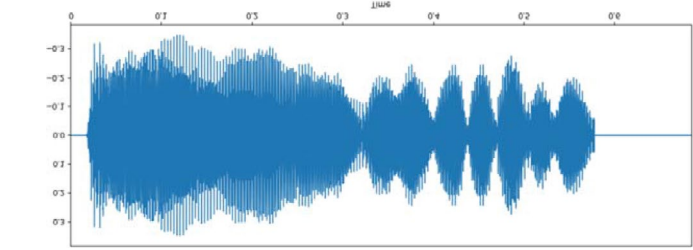
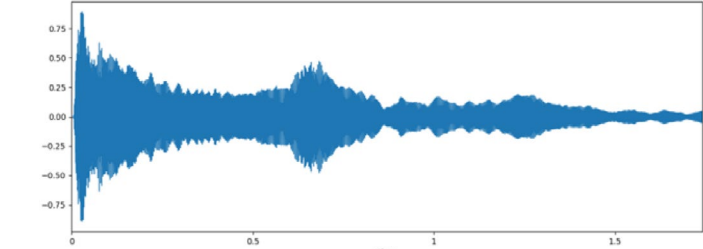
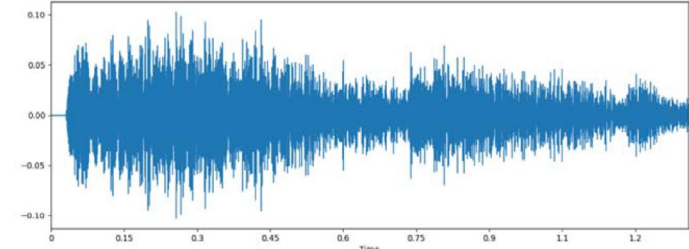

The right-hand playing techniques imply the use of right hand playing. Right hand playing is mainly to pluck the strings, and the position of the playing is mainly on the right side of the bridge. Its basic task is to play the tone. Depending on the music note, the player can use the index finger, the big finger, ring finger and/or middle finger to play. The key is to control the duration, loudness and sonic texture. Also, it is not unusual to play multi-tones at a same time. Table 3 presents four typical right-hand playing techniques with their playing methods, usual location in a music piece and a sample timbre.

**Adding the Guzheng playing techniques using RL.** While LSTM network learns the dependence among notes from the dataset, it may not catch the characteristics of Guzheng music. Thus, we use Reinforcement Learning (RL) to add the Guzheng playing techniques to the music generated by the LSTM network.

The theory of RL has been well developed<sup>25,26</sup>. Figure 8 shows the agent-environment interaction in the RL process. The minibatch is a collection of random samples from LSTM network, the main network produces possible actions and the target network produces a stable target used to compute the loss of the selected action. At time  $t$ , following a policy,  $\pi$ , and the given the current state,  $s_t$ , the agent generates an action,  $y_t$ . According to the action, a reward,  $r_{t+1}$ , and a new state,  $s_{t+1}$ , will then be generated through the environment, which in turn will be input to the agent again in the next round. This process continues until satisfactory result is obtained.

We use Deep Q-learning Network (DQN) for RL. It is a model-free algorithm based on Q-learning and its goal is to select appropriate policies to optimize the sum of rewards at the end of each epoch. We use Deep Neural Network (DNN) for the main network, i.e.:

$$y_t = \pi(s_t) = DNN(s_t) \quad (11)$$

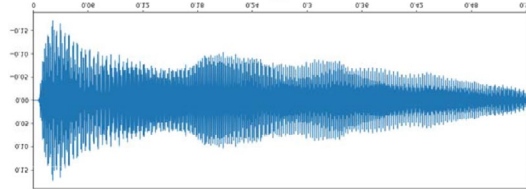
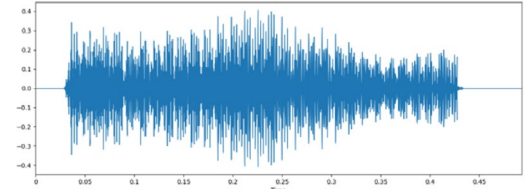
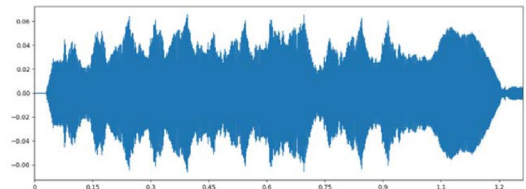
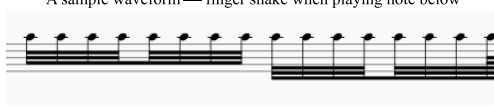
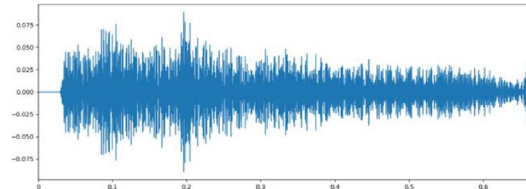

Name	Explanation	Usual location in a music piece
Press (按弦, 滑音)	Use index finger, middle finger and/or ring finger to press a string to play a note outside the pentatonic scale. For example, by pressing, F changed to G or #G	Some transitional notes, or 4, 7 notes.
	 <p>A sample waveform — press when playing D4</p>	
Vibrato (颤音)	Rapidly alternate the pressing pressure on the original position to change the amplitude and strength of a note in time.	The last note of each complete sentence.
	 <p>A sample waveform — vibrato when playing D4</p>	
Glide (滑弦)	Slide pressing through several strings. It can be either down slide or up slide. Variations also include the speed, the pressure, number of times, etc.	The turning of a section.
	 <p>A sample waveform — glide when playing the notes below</p> 	

**Table 2.** The left-hand playing techniques and their corresponding timbre example.

The reward policy is designed to catch the characteristics of Guzheng music. It is made of a number of rules. Each rule has a positive reward and a negative punishment. The rule that has the strongest impact will be set to +1 or −1, and the remaining rules will be compared to the strongest rule to set the reward values.

As an example, let us consider Arpeggio (refer to Table 3). It is a technique usually consisting of seven notes. Many even-beat positions in Guzheng music end with arpeggio. Thus, assuming that at time  $t$ , the input state is  $s_t = \{A_t^1, A_t^2, A_t^3, A_t^4, A_t^5, A_t^6, A_t^7\}$ , where  $A_t^1, A_t^2, \dots, A_t^7$  represents the seven notes incur at  $t$ , then, the reward policy is defined below:

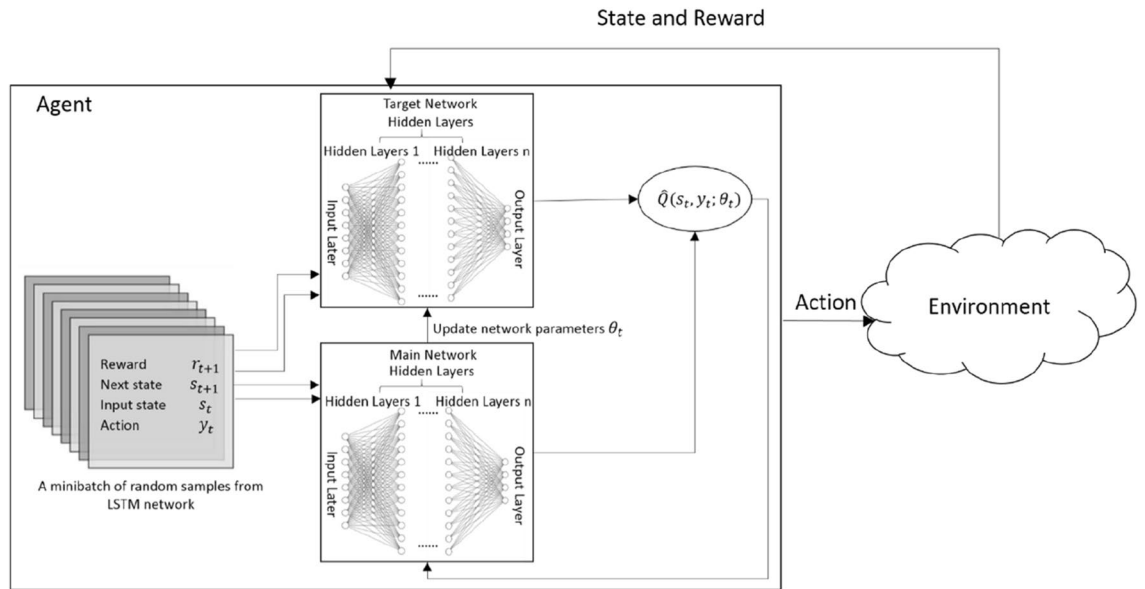


Name	Explanation	Usual location in the music piece
Big pinch (大撮)	Thumb rest, middle finger hook plucking on a relative full octave at the same time.	The beginning of a pieces.
	 <p>A sample waveform—big pinch when playing D3&amp;D4</p>	
Small pinch (小撮)	Thumb rest and index finger plucking on two strings at the same time.	Allegro
	 <p>A sample waveform—small pinch when playing D4&amp;F4</p>	
Finger shake (摇指)	Quick in and out plucking	Adagio, transition section or cooperation with the glide and/or arpeggio
	 <p>A sample waveform—finger shake when playing note below</p> 	
Arpeggio (琶音)	Use ring finger, middle finger, index finger and thumb to roll from bass to treble.	Adagio, cooperate to shake
	 <p>A sample waveform—arpeggio when playing notes below</p> 	

**Table 3.** The right-hand playing skills and their corresponding timbre example.

$$r_t^1(s_t, y_t, t) = \begin{cases} 0.2 & \text{if } y_t \in \{A_1^t, A_2^t, \dots, A_7^t\} \text{ and } (t = \text{evenbeat}) \\ -0.2 & \text{otherwise} \end{cases} \quad (12)$$

The other example is finger shake technique. We consider that when 4 same notes occur, it is the finger shake. However, if there are more than six same notes, it is considered that the notes generated are not vivid and smooth, giving people a bad experience. The reward policy,  $r_t^2$ , is to reduce the velocity by half to make it sounds more like finger shake:



**Figure 8.** The RL process of agent-environment interaction in DQN.

$$r_t^2(s_t, y_t, t) = \begin{cases} 0.3 & \text{if } y_t \in \{A_t^1 = A_{t-1}^1 = A_{t-2}^1 = A_{t-3}^1\} \\ 0 & \text{if } y_t \neq y_{t-1} \\ -1 & \text{if } y_t \in \{A_t^1 = A_{t-1}^1 = A_{t-2}^1 = A_{t-3}^1 = A_{t-4}^1 = A_{t-5}^1\} \end{cases} \quad (13)$$

As for big pinch and small pinch, they both contain two notes. Big pinch usually appears in downbeat while small pinch usually appears in upbeat. Thus, the reward policy is

$$r_t^3(s_t, y_t, t) = \begin{cases} 0.7 & \text{if } y_t \in \{A_1^t, A_2^t\} \text{ and } (t = \text{oddbeat}) \\ -0.3 & \text{otherwise} \end{cases} \quad (14)$$

In general, the total reward is:

$$R_t = \sum_{j=1}^n r_t^j(s_t, y_t, t) \quad (15)$$

where  $n$  is the number of Guzheng characteristics. Assuming that the previous total return is:

$$R_{total} = R_t + \gamma R_{t-1} + \gamma^2 R_{t-2} + \dots + \gamma^{n-t} R_1 \quad (16)$$

where  $\gamma \in [1, 0]$  is a discount value. Then, the average rate of return,  $\bar{R}_t$ , is:

$$\bar{R}_t = \frac{1}{t} \sum_{i=1}^t \frac{R_i}{R_{total}} \quad (17)$$

Depending on the action,  $y_t$ , the value of the action is as follows:

$$Q(s_t, y_t; \theta_t) = \max_{y_t} (s_t, y_t) \quad (18)$$

where  $\theta_t$  is the weights of the DNN at time  $t$ .

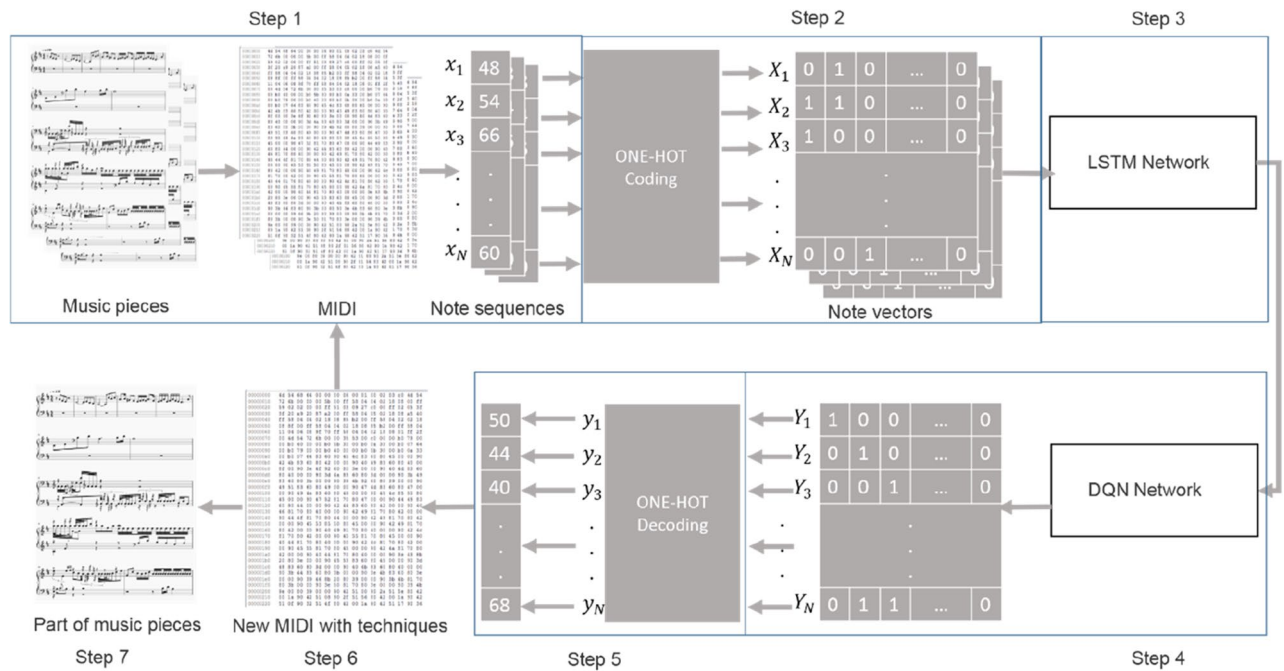
Moreover, the target Q-value,  $\hat{Q}(s_t, y_t; \theta_t)$ , can be calculated by continuously updating the right-hand side of the formula below:

$$\hat{Q}(s_t, y_t; \theta_t) \leftarrow (1 - \alpha) Q(s_t, y_t; \theta_t) + \alpha [R_t + \gamma \max_{y_t} Q(s_{t+1}, y_t; \theta_t)] \quad (19)$$

where  $\alpha \in [1, 0]$  and  $\gamma \in [1, 0]$  are the parameters for controlling the effects of the previous and current actions, and  $s_{t+1}$  is the new state. In this study, we choose  $\alpha = 0.15$  and  $\gamma = 0.8$ .

In each iteration, a data set  $(s_t, R_t, y_t)$  is generated and stored in an experience pool,  $D$ . The objectives of the RL is to train the DNN to minimize the following loss function:

$$L = E_D \left[ (R_t + \gamma \max_{y_t} Q(s_t, y_t; \theta_t^-) - Q(s_t, y_t; \theta_t))^2 \right] \quad (20)$$



**Figure 9.** The training processes.

where  $E[]$  is the mathematical expectation,  $\theta_t^-$  is the parameter of the target DNN. This optimization can be solved using gradient descent method. It converges when  $Q(s_t, y_t; \theta_t)$  reaches  $\left(R_t + \gamma \max_{y_t} (s_t, y_t; \theta_t^-)\right)$ .

The process of RL consist of the following steps:

- Step 1: Initialize the action value function  $y_t$ , initialize the experience pool  $D$ , initialize the DNN parameter  $\theta_i$ , set the value of the action  $Q$  as a random number, and set  $\theta_i^- = \theta_i$ ;
- Step 2: Enter the notes  $y_t$  generated by LSTM (refer to Eq. (9)) as the initial value of  $y_t$ ;
- Step 3: Compute the reward,  $R_t$ , using Eq. (15) and the average rate of return,  $\bar{R}_t$ , using Eq. (17);
- Step 4: Compute the value of action,  $Q(s_t, y_t; \theta_t)$ , using Eq. (18) and the target Q-value,  $\bar{Q}(s_t, y_t; \theta_t)$ , using Eq. (19);
- Step 5: Solve the loss function defined in Eq. (20) to get  $(s_t, R_t; \theta_t)$ , and put it into the experience pool  $D$ ;
- Step 6: If the target is reached ( $\bar{R}_t = 1$  and  $L \rightarrow 0$ ), stop; else update  $\theta_t^- = \theta_t$ ,  $t = t + 1$ , goto Step 3.

### Summary of the training process

Figure 9 shows the training process of our new method. It consists of 7 steps:

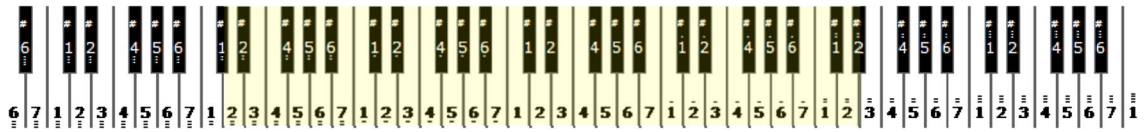
- Step 1: Convert the music notes to MIDI format, through which a piece of music can be represented as a matrix:  $\mathbf{x} = \{x_{t,j}, t = 1, 2, \dots, N, j = 1, 2, \dots, 128\}$ ;
- Step 2: Encode the data using One-Hot coding, which gives the input matrix:  $\mathbf{X} = \{X_{t,j}, t = 1, 2, \dots, N, j = 1, 2, \dots, 128\}$ ;
- Step 3: Feed the data to the LSTM network described in Fig. 7;
- Step 4: Feed the data to the DQN network described in Fig. 8, which gives the output matrix:  $\mathbf{Y} = \{Y_{t,j}, t = 1, 2, \dots, M, j = 1, 2, \dots, 128\}$ ;
- Step 5: Decode the data, which gives  $\mathbf{y} = \{y_{t,j}, t = 1, 2, \dots, M, j = 1, 2, \dots, 128\}$ ;
- Step 6: Convert back to MIDI format, when necessary it can be fed into Step 1 for training again;
- Step 7: Convert to the music note.

The computer program is written in Python using the following tools in the Python toolkit: TensorFlow V1.5, Keras V2.1.5, Music21 V5.2.0, Torch V1.8.0, and Mido V1.2.0.

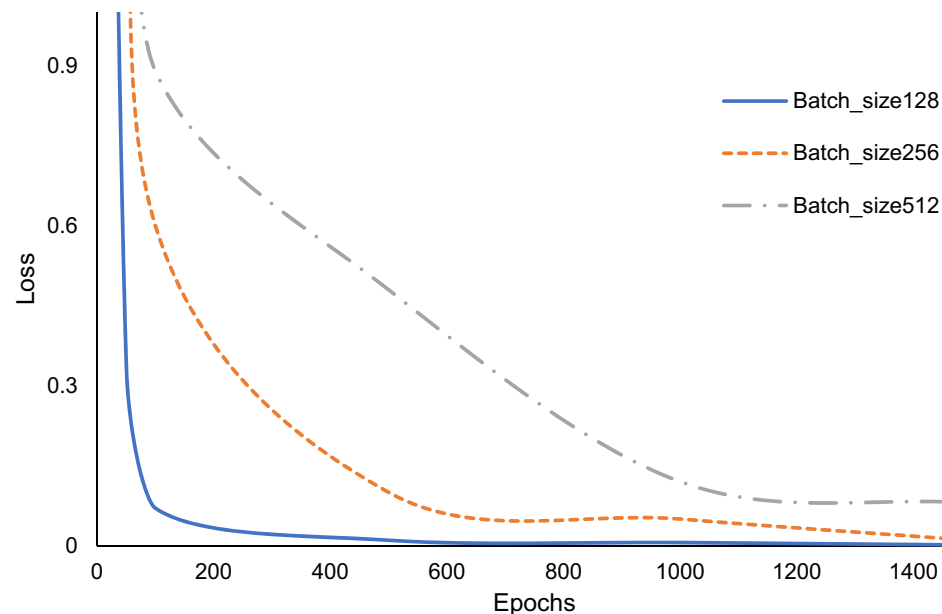
### Post-processing and experiment results

**Converting the music with Guzheng timbre.** As shown in the previous section, we use LSTM to generate Guzheng music and use RL to enhance the generated Guzheng music. However, the music is in piano timbre. Therefore, a post-processing step is necessary.

It is known that MIDI has many built-in timbre settings for different music instruments and sounds, but not for Guzheng. As discussed in Sect. 2, we generate the Guzheng music using piano timbre. To convert the piano timbre to Guzheng timbre we used a freeware “MIDI Cai Hong Gang Qi (MIDI彩虹钢琴)” for the conversion (<https://www.cnblogs.com/qingjun1991/p/4971514.html>).



**Figure 10.** The correspondents between piano and Guzheng (in yellow).



**Figure 11.** The relationship between the Loss and the Batch\_size of LSTM.

It shall be noted that piano and Guzheng have different range of sound. Figure 10 shows the correspondents between the piano keyboard and Guzheng strings, marked in yellow. The 21 basic Guzheng sounds corresponds the 21 tones on the piano with the central C key of the piano corresponds to the 12th string of Guzheng. As we use the piano MIDI to generate the music, it is possible that some music notes are out of the range of the Guzheng. In this case, these music notes will be simply deleted.

**Experiment results.** Using the aforementioned method, new music pieces are generated. First, we decompose the MIDI files of the 31 Guzheng music into segments. Each segment has 1000 lines, covering 10 s. Then, we randomly pick these segments as the input of the LSTM. Upon receiving the input, the LSTM will generate an output, also lasts 10 s. The outputs are evaluated using the loss function defined in Eq. (10). The number of inputs is referred to as the Batch size. The larger the Batch size, the better the LSTM can learn, but the more the computation load. Figure 11 shows the loss function against the batch sizes of 128, 256 and 512. From the figure, it is seen that the larger Batch size require longer time to converge.

The generated music pieces are usually lack of melody and Guzheng characteristics. Thus, as discussed in “Extraction of the melodies” section, we extract the melodies from the training samples pieces and mixed them with the original 31 Guzheng music pieces as the training samples and run the LSTM again as illustrated in Fig. 12.

Then, we use RL to add the Guzheng characteristics. The convergence of the RL is evaluated using the average rate of return (the average ratio of the total revenue and the optimal revenue in the current training session) and the loss function defined by Eq. (20). As shown in Fig. 13, its coverages after 1400 epochs.

Figure 14 shows one of the completed Guzheng music piece in which the Guzheng characteristics are marked: blue represents finger shake, yellow represents arpeggio and pink represents big & small pinch.

A number of music pieces have been generated. Three typical examples are upload on the webpage: <https://www.bilibili.com/audio/am33421435?type=7>. For comparison, a number of music pieces generated using LSTM alone and using GAN are also generated and uploaded on <https://www.bilibili.com/audio/am33419632?type=7>.

**Experiment result evaluation.** It is known that the quality of a music piece is rather difficult to evaluate. Thus, we use both the mathematical method and the subjective method to evaluate.

The mathematical method is the note accuracy method<sup>27</sup>: given a music piece, deleting the last  $N$  notes, re-inputting into the trained model and regenerating the last  $N$  notes, then comparing the newly generated notes to the original notes. If the newly generated notes are similar to the original notes, then the trained model is



**Figure 12.** A sample music generated by LSTM with extracted melodies.

considered good. Assuming that the generated notes are  $\bar{y}_{t,j}$  and the original notes is  $y_{t,j}$ , the note accuracy,  $A$ , can be expressed as follows:

$$A = \frac{\sum_{j=1}^M \sum_{t=1}^N P(y_{t,j}, \bar{y}_{t,j})}{MN} \quad (21)$$



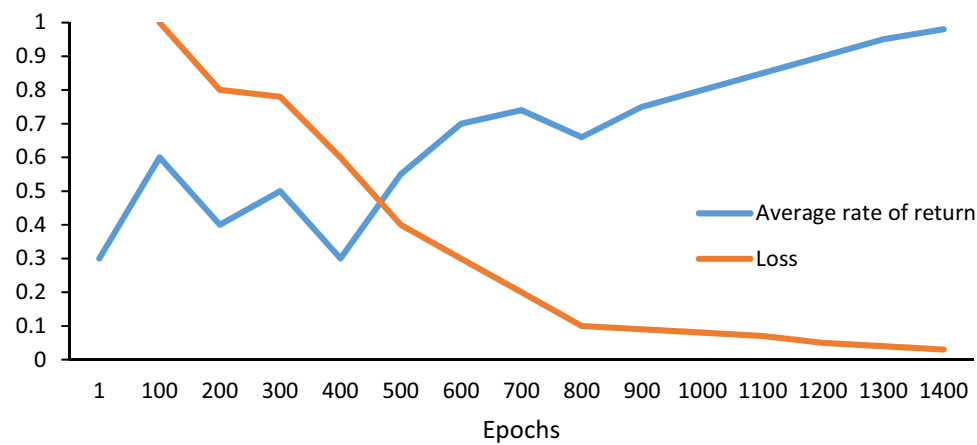


Figure 13. The relationship between the loss and the average rate of return of RL.

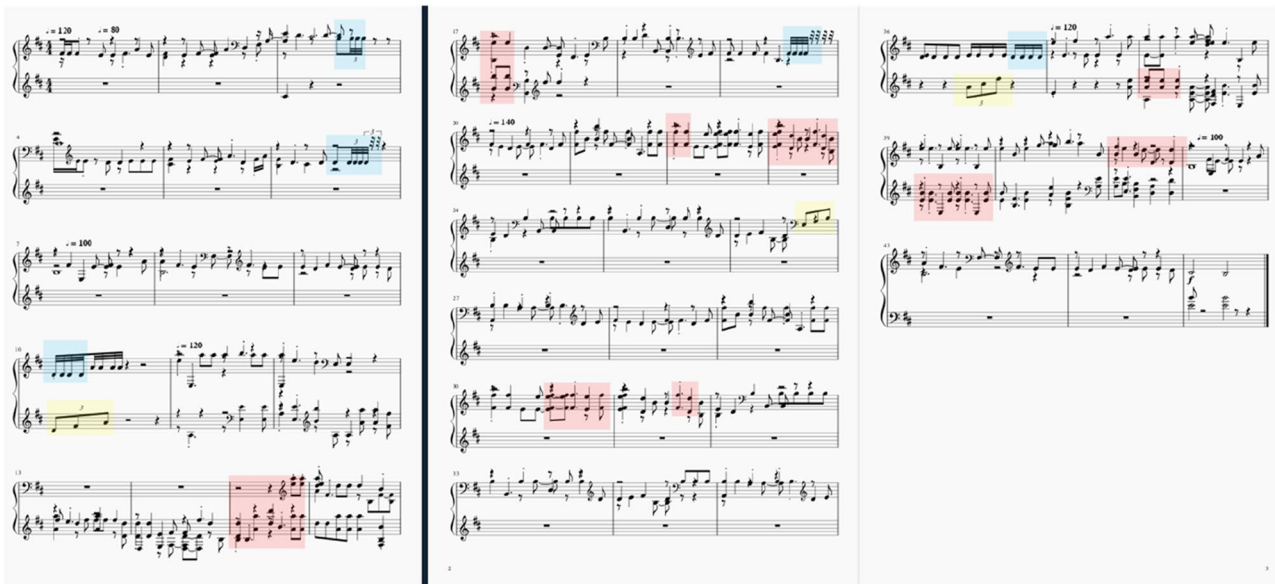


Figure 14. Generated music process.

N	Accuracy (×100%)
10	80%
20	75%
25	68%

Table 4. The note accuracy under different lengthen of nodes.

where  $j = 1, 2, \dots, M$  denote test samples,  $t$  denotes the notes in the test sample,  $N$  is the number of notes and is set to 10, 20 and 25 respectively, and

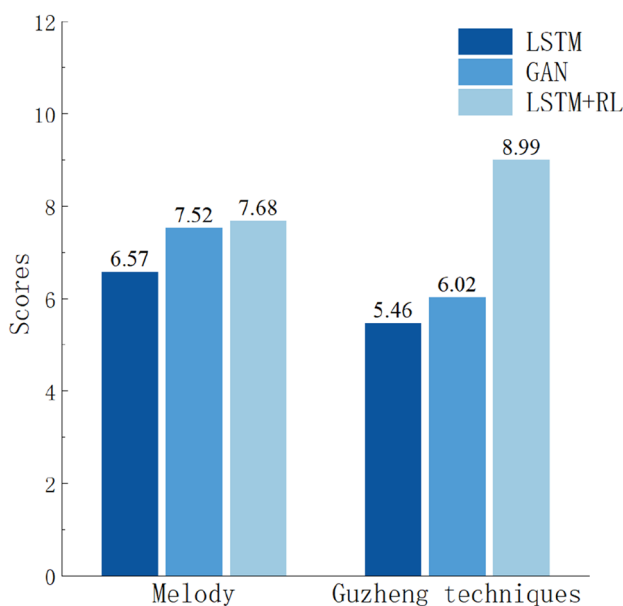
$$p(y_{t,j}, \bar{y}_{t,j}) = \begin{cases} 1, & y_{t,j} = \bar{y}_{t,j} \\ 0, & y_{t,j} \neq \bar{y}_{t,j} \end{cases}$$

We used  $M = 5$  generated music pieces to test our LSTM + RL model and the results are shown in Table 4. From the table, it is seen that our model has a good note accuracy.

Besides, we use the same method to test the effectiveness of RL. This time, instead of the last  $N$  notes, we search for the Guzheng playing techniques appeared in the music piece. Table 5 shows the experiment results

Characteristic	LSTM ( $\times 100\%$ )	LSTM + RL ( $\times 100\%$ )
Arpeggio	0.1%	6%
Finger shake	0.1%	8%
Big pinch & Small pinch	0.1%	17%
Repeating notes	11%	3%

**Table 5.** The accuracy comparison between LSTM and LSTM + RL.



**Figure 15.** The subjective evaluation scores in melody and Guzheng playing techniques.

on 10 generated music pieces. From the table, it is seen that using LSTM + RL gets much better results than that of using LSTM alone.

The subjective evaluation is based on the interview of 10 experienced Guzheng players<sup>28,29</sup>. We prepared a set of Guzheng music pieces, including three pieces generated by LSTM, three pieces generated by LSTM + RL model and three pieces generated by GAN. The interviewees are asked to rank both the melody and the Guzheng playing techniques on the scale of 10. The evaluation results are summarized in Fig. 15. As shown in the figure, compared to LSTM and GAN, the LSTM + RL has the best performance in melody. This implies that the music generated by LSTM + RL is better in resembling the traditional Guzheng music. Also, LSTM + RL generates much more Guzheng playing techniques. Consequently, it is said that our Guzheng music is indeed close to the real Guzheng music.

## Conclusions and future work

This paper presents a new method for composing Guzheng music using LSTM and RL. Based on the discussions above, following conclusions can be drawn:

- (1) The presented LSTM + RL method is effective. According to the evaluation by note accuracy method and by 10 skilled Guzheng players, the generated music pieces is closely resembling the classic Guzheng music.
- (2) The key to the success is to extract the melody from the training samples using LSTM as well as adding Guzheng characteristics using RL. This LSTM + RL combination makes the generated music more like a Guzheng music instead of mere imitation.
- (3) The presented method is efficient. Using the MIDI format, the training and music generation take only a few hours in a PC computer.

We expect that the same method can be applied to compose music for other traditional Chinese music instruments, such as Pipa (琵琶), Guqin (古琴), Lusheng (芦笙) and etc.

For the future work, two issues can be pursued.

- (1) In the training, the sampled music piece is divided into multiple segments, each has 10 s in length. While this is simple and effective, it may miss the long-term correlation in a music piece. The use of longer segments (e.g., 30 s or longer) may be beneficial for slow mode music, though, the training will be longer.
- (2) The emotional features of the music pieces can be extracted, classified and labeled in the training process to further optimize the generated music.

## Data availability

The datasets used and/or analysed during the current study available from the corresponding author on reasonable request.

Received: 17 January 2022; Accepted: 5 September 2022

Published online: 22 September 2022

## References

1. Wang, P. K. *Study on the Composition Technique of Chen Zhe Guzhang Works: Take five works for example* (Henan university Press, 2020).
2. Li, B. The historical inheritance and artistic charm of Guzhang are briefly introduced. *Song Yellow River* **4**, 9 (2020).
3. Zhou, X. Y. Research on the development and popularity of Guzhang music. *Sound Yellow River* **9**, 1 (2018).
4. Li, M. Basic course of Guzhang. *Beijing: Int. Cult. Publ. Co.* **1**, 52 (2020).
5. Westergaard, P. et al. Experimental music. Composition with an electronic computer. *J. Music Theor.* **3**, 302 (1959).
6. Andries, V. & Schulze, W. Music generation with Markov models. *IEEE Multimed.* **18**, 78–85 (2011).
7. Hadjeres, G. et al. DeepBach: A steerable model for bach chorales generation. In *34th International Conference on Machine Learning*, vol. 3, 2187–2196 (2016).
8. Schuster, M. P. et al. Bidirectional recurrent neural networks. *IEEE Trans. Sign. Process.* **45**, 2673–2681 (1997).
9. Hang, C. et al. Song from Pi: A Musically Plausible Network for Pop Music Generation. arXiv preprint <https://arxiv.org/abs/1611.03477> (2016).
10. Chung, J. et al. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. arXiv preprint <https://arxiv.org/abs/1412.3555> (2014).
11. Lambert, A. J. et al. Perceiving and predicting expressive rhythm with recurrent neural networks. In *Proceedings of the 12th International Conference in Sound and Music Computing*, 265–272 (2015).
12. Choi, K. et al. Convolutional recurrent neural networks for music classification. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2392–2396 (2017).
13. Kang, S. I. et al. Improvement of speech/music classification for 3GPP EVS Based on LSTM. *Symmetry* **10**, 8 (2018).
14. Tang, C. P. et al. Music genre classification using a hierarchical long short-term memory (LSTM) model. In *Proceedings of SPIE: The International Society for Optical Engineering*, vol. 10828, 108281B–108281B-7 (2018).
15. Tatar, K. et al. Latent timbre synthesis: Audio-based variational auto-encoders for music composition and sound design applications. *Neural Comput. Appl.* **33**, 67–84 (2020).
16. Goodfellow, I. et al. Generative adversarial nets. *Adv. Neural. Inf. Process. Syst.* **27**, 2672–2680 (2014).
17. Guan, F. et al. A GAN model with self-attention mechanism to generate multi-instruments symbolic music. In *International Joint Conference on Neural Networks*, 1–6 (2019).
18. Li, S. et al. INCO-GAN: Variable-length music generation method based on inception model-based conditional GAN. *Mathematics (Basel)* **9**, 387 (2021).
19. Kotecha, N. *Bach2Bach: Generating Music Using A Deep Reinforcement Learning Approach* (Columbia University Press, 2018).
20. Jiang, N. et al. RL-duet: Online music accompaniment generation using deep reinforcement learning. arXiv preprint <https://arxiv.org/pdf/2002.03082.pdf>. (2020).
21. Karbasi, S. M. et al. A Generative Model for Creating Musical Rhythms with Deep Reinforcement Learning. arXiv preprint [https://aimc2021.iem.at/wpcontent/uploads/2021/06/AIMC\\_2021\\_Karbasi\\_et\\_al.pdf](https://aimc2021.iem.at/wpcontent/uploads/2021/06/AIMC_2021_Karbasi_et_al.pdf) (2021).
22. Jaques, N. et al. *Tuning Recurrent Neural Networks with Reinforcement Learning* (Workshop Track: ICLR, 2016).
23. Meng, Z. Research on timbre classification based on BP neural network and MFCC. *J. Phys: Conf. Ser.* **1856**, 012006 (2021).
24. Zhu, D. *Research on Inheritance and Innovation of Chinese Guzhang Artistic Performance Techniques* (Hebei Normal University Press, 2011).
25. Li, J. et al. Deep reinforcement learning for dialogue generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. (2016).
26. Ye, H. et al. Deep reinforcement learning based resource allocation for V2V communications. *IEEE Trans. Veh. Technol.* **68**(4), 3163–3173 (2019).
27. Bai, Y. *Research and Implementation of Music Generation Based on Deep Reinforcement LEARNING* (Zhengzhou University Press, 2020).
28. Ji, S., Luo, J. et al. A Comprehensive Survey on Deep Music Generation: Multi-Level Representations, Algorithms, Evaluations, and Future Directions. arXiv preprint <https://arxiv.org/abs/2011.06801> (2020).
29. Dong, H. W. et al. MuseGAN: Symbolic-Domain Music Generation and Accompaniment with Multi-Track Sequential Generative Adversarial Networks. arXiv preprint <https://arxiv.org/abs/1709.06298v1> (2017).

## Acknowledgements

We wish to thank for Prof. Wu Li and Miss Yang Xiaoming of Xinghai Conservatory of Music for their comments and suggestions in Guzhang music.

## Author contributions

S.C. collected data, analyzed algorithm and conducted RL modelling, and wrote the first draft. Y.Z. and R.X. supervised the study, advised on algorithm analyses, and revised the paper. All authors interpreted results and approved the final version for submission.

## Funding

We acknowledge the financial support of the Natural Science Foundation of Guangdong Province (Grant No. 2020A1515010621) and the Strategic Priority Research Program of the Chinese Academy of Sciences (class A) (Grant No. XDA22040203).

## Competing interests

The authors declare no competing interests.

## Additional information

**Correspondence** and requests for materials should be addressed to Y.Z.

**Reprints and permissions information** is available at [www.nature.com/reprints](http://www.nature.com/reprints).

**Publisher's note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

© The Author(s) 2022