International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 17 (2025) pp. 1-19

https://doi.org/10.70917/ijcisim-2025-0294
Article

A study on deep reinforcement learning based method for voice expression enhancement of opera singers

Laijunwa Kong *

Music and Dance School, Henan Normal University, Xinxiang, Henan, 453000, China; 67676767kk@sina.com

Abstract: Opera singer's voice expression is the soul of art. In reality, the singer's voice is susceptible to the interference of background noise, and the melody and rhythm of the voice are highly dependent on personal experience and state. In this paper, we propose a framework for enhancing the vocal performance of opera singers that integrates speech signal processing methods and deep reinforcement learning. The framework employs an improved convolutional transfer function generalized paraflap canceller algorithm (CTF-GSC) to suppress the non-coherent noise in the performance environment to achieve the noise reduction effect. A model based on two-layer unidirectional LSTM network and a model based on reinforcement learning AC algorithm are constructed to enhance the rhythm and melody of the generated music. The results show that the improved CTF-GSC algorithm effectively suppresses the correlated and non-correlated noise and enhances the vocal expression of the opera singer's voice. The generated music tunes are richer, and the probability of notes in the key reaches 98.8%. This study provides a new intelligent method for realizing the artistic performance of opera singing and opens up a new path for performance innovation in vocal art.

Keywords: deep reinforcement learning; two-layer unidirectional LSTM network; reinforcement learning AC algorithm; musical rhythmic melody generation

1. Introduction

The so-called opera, which is an organic combination of music, dance, stage art and other elements, tells the story, fully reveals the characters, and brings the audience a sense of pleasure as a mode of theater, which possesses a unique charm and is an important art form [1-4]. In opera performance, the actor's voice expression is especially important, which directly determines whether the work can be perfectly interpreted [5]. Actors in singing opera works, the skills are the most critical, but the voice expression will give the complete work to add the finishing touch of an important piece, however, the actor to show expressive voice is not an easy task [6-8]. It is firstly according to the form and connotation given by the lyricist on the score; secondly, its realization depends on the actor's correct understanding of the work, tacit cooperation, and with their own characteristics of the interpretation and processing, is the work can be visualized, and enable the audience to achieve concentration, into the mood, moved, so that the audience can be infected by the art while appreciating the art and get the enjoyment of the beauty of the art [9-13]. However, in the actual performance, from time to time, the following situation occurs, there are many opera singers themselves are very good, but it is difficult to fully display their voice expression. This is because on the one hand, the performer does not deeply understand the connotation of the work, and on the other hand, the performer lacks expressive power, and it is these two factors that lead to the failure of the opera performance [14-16].

With the development of artificial intelligence, deep reinforcement learning provides technical support to compensate for the above deficiencies. Deep reinforcement learning is a popular direction in the field of artificial intelligence, which integrates the advantages of deep learning and reinforcement learning, and is able to make decisions and learn in complex environments, and has been widely used in the fields of automatic driving, game play optimization, and intelligent recommender systems, etc.

[17-20]. In enhancing the voice performance of opera performers, deep reinforcement learning learns the optimal strategy through the interaction between the intelligent body and the environment, can analyze the noise characteristics and adjust the processing strategy in real time, continuously optimizes the enhancement effect through the reward mechanism, and directly maps from the original waveforms to the enhanced speech to reduce the intermediate error [21-24].

The environment of live opera performances is generally noisy, and the noise in the environment can seriously affect the expressiveness and infectiousness of opera singers' voices. In order to solve the above dilemma, this study is dedicated to applying advanced deep reinforcement learning techniques to opera singing, and developing a set of intelligent programs that can enhance the expressiveness of opera singers' voices. The traditional CTF-GSC algorithm is improved in a relevant way, and the variable step-size normalized least mean square algorithm is used for adaptive noise cancellation, which enhances the filtering degree of non-coherent noise. The rhythm generation part is completed by LSTM network, and the melody generation part introduces the multi-label classification technique to realize the simultaneous output of multiple notes, which improves the generation efficiency of the model and provides an innovative and efficient method for the creation of opera singers.

2. CTF-GSC based speech enhancement algorithm

2.1. Speech signal modeling

In an acoustic environment containing noise, assume that there are M microphones forming a microphone array, and the signal received by the microphone array is:

$$x_m(t) = a_m(t) * s(t) + n_m(t); m = 1, \cdots, M \tag{1}$$

where $x_m(t)$ denotes the signal received by the m th microphone, $a_m(t)$ denotes the transfer function from the pure speech signal to the m th microphone, $*$ denotes the convolution, $s(t)$ denotes the pure speech signal, and $n_m(t)$ denotes the noisy signals, including coherent and incoherent noise, received by the m th microphone. Performing a short time Fourier transform [25] on the above equation and expressing it in vector form gives:

$$X(k,l) = A(k)S(k,l) + N(k,l) \tag{2}$$

where k and l are the frequency index and time index, respectively, and $A(k)$ is the time-invariant transfer function. Taking the first microphone of the microphone array as the reference microphone and the pure speech signal received from the reference microphone as the target speech signal, the above acoustic model can be expressed by the relative transfer function model as:

$$X(k,l) = H(k)A_1(k)S(k,l) + N(k,l) \tag{3}$$

2.2. Basic CTF-GSC algorithm

The basic structural block diagram of the CTF-GSC algorithm is shown in Fig. 1, which consists of three parts: fixed beamformer (FBF), blocking matrix (BM), and adaptive noise canceler (ANC).The CTF-GSC algorithm is based on the traditional GSC algorithm, which takes into account the mutual interference between different frequencies, and uses the scaling factors of the relative transfer functions (RTFs) between each microphone for the fixed beamformer and blocking matrix for a new construction to model the array signal transfer function using the CTF approximation form in the frequency domain.

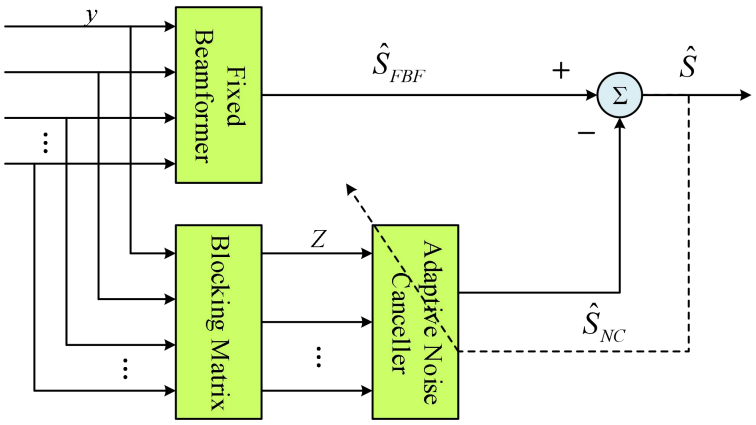


Figure 1. CTF-GSC algorithm block diagram.

In a noisy and reverberant complex environment with M microphones operating, only one target speech signal source is set up. The speech signal captured by the i th microphone is:

$$y_i(n) = a_i(n) * s(n) + u_i(n) \triangleq d_i(n) + u_i(n) \tag{4}$$

where $i = 1, 2, \cdots, M$, $*$ denotes the convolution, $s(n)$ denotes the target speech signal source, $a_i(n)$ denotes the transfer function of the path from the target speech signal source to the i th microphone, and $d_i(n)$ and $u_i(n)$ denote the speech signals and noise signals captured by the i th microphone.

In the CTF-GSC algorithm, the CTF approximation form is referenced to the blocking matrix of the traditional GSC algorithm, where the convolution matrix satisfies $H_i(k', k) = 0, \forall k' \neq k$. So the blocking matrix is designed as:

$$B = \begin{bmatrix} -H_2^H & -H_3^H & \cdots & -H_M^H \\ I & O & \cdots & O \\ O & I & \cdots & \vdots \\ \vdots & \vdots & \vdots & O \\ O & \cdots & O & I \end{bmatrix} \tag{5}$$

And there:

$$H_i = \begin{bmatrix} H_i(0,0) & 0 & \cdots & 0 \\ 0 & H_i(1,1) & & \vdots \\ \vdots & \vdots & \vdots & 0 \\ 0 & \cdots & 0 & H_i(N-1, N-1) \end{bmatrix} \tag{6}$$

$$K(k) = \left[\sum_{i=1}^{M} H_i^H(k,k) H_i(k,k) \right]^{-1} \tag{7}$$

Then the BM module output is:

$$z_i(k) = y_i(k) - H_i(k,k) y_1(k) \tag{8}$$

where $i = 2, \cdots, M$.

The FBF module output is:

$$\{\hat{S}_{FBF}(k)\}_{CTF} = K(k) \sum_{i=1}^{M} H_i^H(k,k) y_i(k) \tag{9}$$

The filter coefficients are updated iteratively using the NLMS algorithm in the ANC module:

$$\tilde{G}_i(p+1,k) = G_i(p,k) + \mu \frac{z_i(e^{j\omega}) y_1(p,k)}{P_{est}(p,k)} \tag{10}$$

$$G_i(p+1,k) \overset{FIR}{\leftarrow} \tilde{G}_i(p+1,k) \tag{11}$$

Among them:

$$P_{est}(p,k) = \rho P_{est}(p-1,k) + (1-\rho) \sum_{i=1}^{M} \left| y_i(p,k) \right|^2 \tag{12}$$

In the CTF-GSC algorithm, the most important thing is to estimate the RTFs between the speech signals captured by each microphone, i.e., $H_i(i=1,2,\cdots,M)$. Let the noisy speech signal captured by the i th microphone be $y_i(k)$ and the noisy speech signal captured by the first microphone be $y_1(k)$. So there is:

$$y_1(k) = d_1(k) + u_1(k) \tag{13}$$

According to the vector transformation of STFT, the speech signal captured by each microphone is:

$$y_i(k) = y_1(k) h_i(k',k) + v_i(k) \tag{14}$$

where $v_i(k)$ denotes the noise signal captured by the i th microphone. When $k' \neq k$, $h_i(k',k) = 0$, $h_i(k,k)$ denotes the relative impulse response of the speech signals captured by each microphone at the same frequency point, and $h_i(k) = [h_i(0,k), h_i(1,k), \cdots, h_i(P-1,k)]^T$.

Also:

$$v_i(k) = u_i(k) - h_i(k,k) U_1(k) \tag{15}$$

where $U_1(k)$ denotes the Toeplitz matrix consisting of the STFT coefficients of $u_1(k)$.

Then the mutual power spectrum is solved simultaneously for the k th frame of the speech signal at both ends of the equal sign of Eq. (13) and Eq. (14), and we have:

$$\Phi_{i,1}^{y}(k) = \Psi_{1,1}^{y}(k) h_i(k,k) + \Phi_{i,1}^{v}(k) \tag{16}$$

$$\Phi_{i,1}^{v}(k) = \Phi_{i,1}^{u}(k) - \Psi_{1,1}^{u}(k) h_i(k,k) \tag{17}$$

Since the speech signal $s(n)$ and the noise signal $u(n)$ are uncorrelated, $\Psi_{1,1}^{y}(k) = \Psi_{1,1}^{s}(k) + \Psi_{1,1}^{u}(k)$, according to Eqs. (16) and (17), then there is:

$$\Psi_{1,1}^{y}(k) = \Psi_{1,1}^{s}(k) h_i(k,k) + \Psi_{1,1}^{u}(k) \tag{18}$$

Based on the power spectral density (PSD) estimation, Eq. (18) is rewritten as:

$$\widehat{\Phi}(k) = \widehat{\Psi}(k) h_i(k,k) + e(k) \tag{19}$$

where $e(k)$ denotes the PSD estimation error:

$$\widehat{\Phi}(k) \triangleq \widehat{\Phi}_{i,1}^{y}(k) - \widehat{\Phi}_{i,1}^{u}(k) \tag{20}$$

$$\widehat{\Psi}(k) \triangleq \widehat{\Psi}_{1,1}^{s}(k) = \widehat{\Psi}_{1,1}^{y}(k) + \widehat{\Psi}_{1,1}^{u}(k) \tag{21}$$

Applying the weighted least squares method to Eq. (19) yields:

$$\hat{h}_i(k,k) = \left[\widehat{\Psi}^H(k) \Gamma(k) \widehat{\Psi}(k) \right]^{-1} \widehat{\Psi}^H(k) \Gamma(k) \widehat{\Psi}(k) \tag{22}$$

where $\Gamma(k)$ denotes the weight matrix, $\Phi_{i,1}^{y}(k)$ and $\Psi_{1,1}^{y}(k)$ can be computed from the captured speech signals, $\Psi_{1,1}^{u}(k)$ and $\Phi_{i,1}^{u}(k)$ can be obtained from the silent segment.

2.3. Improved CTF-GSC algorithm

In the ANC module of the CTF-GSC algorithm, the basic fixed-step NLMS algorithm is used to iteratively update the filter coefficients. The μ denotes the step size factor, which controls the adaptive speed and stability. The fixed-step-size NLMS adaptive algorithm is contradictory between convergence rate, tracking rate and steady state misalignment noise; when μ takes a small value, the steady state misalignment noise is small and the accuracy is high, but the convergence and tracking speed are slow. In order to overcome this drawback, the variable step size normalized least mean square (VSS-NLMS) algorithm is used in this paper to improve the step factor μ.

$$\mu(k) = \theta\mu(k-1) + (1-\theta)\frac{\hat{\sigma}_e^2(k)}{\lambda\hat{\sigma}_v^2(k)} \tag{23}$$

where θ denotes the forgetting factor, λ denotes the positive parameter with added design degrees of freedom, $\hat{\sigma}_e^2(k)$ denotes the mean square error, and $\hat{\sigma}_v^2(k)$ denotes the system noise power.

$$\hat{\sigma}_e^2(k) = \theta\hat{\sigma}_e^2(k-1) + (1-\theta)y^2(k) \tag{24}$$

$$\hat{\sigma}_v^2(k) = \hat{\sigma}_e^2(k) - \frac{1}{\hat{\sigma}_y^2(k)}\hat{r}_{ey}(k)^T\hat{r}_{ey}(k) \tag{25}$$

In Eq. (25), $\hat{r}_{ey}(k)$ denotes the correlation between $y(k)$ and $e(k)$, and $\hat{\sigma}_y^2(k)$ denotes the input signal power. They can be estimated by the following formula:

$$\hat{\sigma}_y^2(k) = \theta\hat{\sigma}_y^2(k-1) + (1-\theta)y^2(k) \tag{26}$$

$$\hat{r}_{ey}(k) = \theta\hat{r}_{ey}(k-1) + (1-\theta)y(k)e(k) \tag{27}$$

The variable step size NLMS algorithm is invoked in the ANC module to iteratively update the filter coefficients. The improved variable step-size NLMS algorithm can further reduce the influence of the input speech signal on the noise estimation, so that the ANC system has faster convergence, higher stability, and better suppression of interference, and minimizes the mean-square error between the output signal of the filter and the desired output signal, so that the system outputs are the best estimation of the useful signal. Thus, the CTF-GSC algorithm is more capable of suppressing noise.

3. Reinforcement Learning Based Enhancement Method for Voice Generation of Opera Performers

3.1. Melodic and rhythmic generation problems

3.1.1. Description of the problem

Melody consists of notes, and the problem of melody generation is actually the problem of note sequence generation. The essence of melody generation is to predict the subsequent notes through the previous known notes. The music generation problem based on deep neural networks is to learn the knowledge related to melodic progression and so on in the music samples through deep neural networks, and then based on the knowledge learned by the network, predict the next note by giving the initial note or randomly initialized notes, and then continue to predict the subsequent notes based on the data of the note sequences that have already been generated to generate the complete note sequences. In this paper, the method of generating rhythms and notes separately is used.

3.1.2. Problem definition

Formal definition of the music generation problem: In melodic and rhythmic generation, let the sequence of notes $S_n = \{n_1, n_2, ..., n_L\}$, and a sequence of rhythms $S_r = \{r_1, r_2, ..., r_L\}$, where $n_i = \{s_1, s_2, ..., s_N\}$, $(i = 1, 2, ...L)$, L is the length of the note sequence, and N is the number of notes contained at a given moment in the note sequence. When $N = 1$ it means that the output is a single note, and when $N > 1$ it means that the output is a complex note. The note sequence S_n and the rhythmic sequence S_r are first encoded accordingly and converted into data that can be input to the

model as follows:

$$S_n^E = Encoder\left(S_n\right) = MultiHot\left(S_n\right) = \left\{n_1^{mh}, n_2^{mh}, ..., n_L^{mh}\right\}$$
$$S_r^E = Encoder\left(S_r\right) = OneHot\left(S_r\right) = \left\{r_1^{oh}, r_2^{oh}, ..., r_L^{oh}\right\} \tag{28}$$

Let the final note sequence generated by the model be $S_n^g = \{n_1^g, n_2^g, ..., n_L^g\}$, and the rhythmic sequence is $S_r^g = \{r_1^g, r_2^g, ..., r_L^g\}$, and finally the rhythmic sequence S_r^g and the note sequence S_n^g are combined to get the complete melody $S_m^g = \left\{\left\{r_1^g, n_1^g\right\}, \left\{r_2^g, n_2^g\right\}, ..., \left\{r_L^g, n_L^g\right\}\right\}$.

3.2. Rhythmic and melodic generation models

For melody generation, this paper proposes ACMGM, a melody generation model based on reinforcement learning AC algorithm, which consists of two main parts: data processing and model network. The data processing module transforms the original score data into data that can be input to the model and transforms the model output into a score file. The model network is based on the LSTM network [26]. In this paper, the note data extracted from the original music files are used to pre-train the LSTM network and serve as the return network of the ACMGM model, so as to provide feedback corresponding to the value return for the actions taken by the Actor.

The final note sequences are generated by the Actor network in the ACMGM model and integrated with the rhythmic and note sequences into a complete MIDI score through the data processing module.

The rhythmic sequences are generated by a separate rhythm generation model, which is implemented through an LSTM network and trained with rhythmic data extracted from the original music data files. The rhythm data is floating-point duration data in seconds, and the floating-point form of the duration data is mapped to an integer data range by means of the base duration unit in order to facilitate data processing.

3.2.1. Data processing

The processing of musical note data in this paper is mainly realized based on the music data processing module in MagentA framework. However, since this paper needs to perform multi-hot encoding for note data, and the music data processing module in MagentA does not have the mechanism of multi-hot encoding for notes, so this paper extends the function of the music data processing module in MagentA and adds the mechanism of multi-hot encoding for notes. In addition, the music data processing module in MagentA does not have a separate mechanism for processing rhythm data, so this paper extends the rhythm data processing function on the basis of MagentA data processing at the same time.

After the original MIDI format score is preprocessed by the extended music data processing module, the note data is finally encoded as multi-hot data that can be used as model input, and the rhythm data is encoded as one-hot form data.

3.2.2. Rhythm generation

The network structure of the rhythm generation model is a two-layer unidirectional LSTM network, and in order to facilitate the use, the rhythm generation and model training are run separately.

The rhythmic data used in the model training process is processed by the music data processing module of MagentA framework. Since there is no mechanism in MagentA for the time being that is specifically used to extract the rhythmic data in the score file, this paper extends the functionality based on the data processing framework of MagentA and adds the function of rhythmic data extraction, and in order to facilitate the unified processing of note data and rhythmic data, this paper extracts the rhythmic data at the same time as the note data, and the rhythmic data is extracted from the note data. In order to facilitate the unification of note data and rhythm data, this paper extracts the rhythm data at the same time as the note data, and stores the note data and rhythm data into different lists. Since the generation of rhythm is not like the generation of melody which needs to consider the case of polyphonic melody, and does not involve the scenario of generating multiple rhythmic timings at the same time, the rhythmic timings only need to be encoded as one-hot data.

The rhythm generation process starts with initializing the model by reading the trained model parameter file to initialize the model and given an initial duration data r_{init} or an initial rhythm sequence $S_r = \left\{r_1, r_2, ..., r_N\right\}$ or use randomly selected duration data r_{random} as the initialized duration data for

the rhythmic sequence and initialize the length L of the generated rhythmic sequence. The initial note r_{init} is computed by LSTM network and output O_{lstm}, O_{lstm} is transformed by Linear layer to get O_{linear}, and O_{linear} is then entered into the $soft\max$ module to output rhythmic time-value's probability distribution, and finally randomly select the corresponding rhythmic time value according to the probability distribution. The flow of rhythm generation calculation is shown in Eq. (29):

$$\begin{aligned} O_{linear} &= O_{lstm} * w^T + b,\ \ w \in \mathbb{R}^{D_r * D_n}, b \in \mathbb{R}^{D_r} \\ O_{soft\max} &= soft\max\left(O_{linear}\right) \end{aligned} \tag{29}$$

The model generates a rhythmic sequence $S_r^g = \{r_1, r_2, ..., r_L\}$ after L cycle, and the final data processing module converts the rhythmic sequence S_r^g into a MIDI format music file for output.

3.2.3. Melody Generation

For the generation of melody, this paper proposes the note sequence generation model ACMGM based on the reinforcement learning AC algorithm.Among them, for the generation of melody with polyphony, this paper realizes the simultaneous output of multiple notes by the method of multi-label classification technique.

In this paper, the LSTM network is firstly trained with the note dataset, then the pre-trained network is set as the reward network in reinforcement learning, and finally the corresponding reward value r is obtained by feeding the notes into this reward network.

Before training the Actor and Critic networks of the ACMGM model, the payoff network RewardNet first needs to be pretrained. The model structure of RewardNet is similar to that of the rhythmic generation model, but in order to increase the focus on the important notes in the note sequences and to fully learn them, the Attention module is added to the RewardNet model compared to the rhythmic generation model. In addition, the activation function in the rhythmic generation model is changed from $soft\max$ to $sigmoid$ in order to support the generation of polyphonic melodies. The loss function of the model was changed from $soft\max$ to $sigmoid$ cross-entropy loss function. The payoff network model is trained by the melody data extracted from the original music files, and the network parameters are saved locally after the payoff network training is completed. The model training calculation process is as follows:

$$\begin{aligned} O_{linear} &= O_{lstm} * w^T + b,\ \ w \in \mathbb{R}^{D_m * D_n}, b \in \mathbb{R}^{D_m} loss \\ &= sigmoid\_cross\_entropy\left(O_{linear}\right) \\ &= cross\_entropy\left(sigmoid\left(O_{linear}\right)\right) \end{aligned} \tag{30}$$

where D_m is the number of note categories and the $sigmoid$ function is defined as $f(x) = \dfrac{1}{1+\exp(-x)}$.

Taking the binary classification model as an example, its $sigmoid$ cross-entropy loss function is calculated as follows:

$$\begin{aligned} loss &= -\left[y * \log\left(\frac{1}{1+\exp(x)}\right) + (1-y) * \log\left(1 - \frac{1}{1+\exp(x)}\right)\right] \\ &= y * \log\left(1+\exp(x)\right) + (1-y) * \log\left(x + \log\left(1+\exp(-x)\right)\right) \\ &= (1-y) * x + \log\left(1+\exp(-x)\right) = x - x * y + \log\left(1+\exp(-x)\right) \end{aligned} \tag{31}$$

where x is the output of the model before the activation function and y is the corresponding label. When $x < 0$ in order to avoid x being too small resulting in $\exp(-x)$ overflow, Eq. (31) is converted to the following equivalent form, the

$$\begin{aligned} loss &= x - x^* y + \log(1+\exp(-x)) \\ &= -x^* y + \log(1+\exp(x)) \end{aligned} \tag{32}$$

And in reality, to ensure the stability of training and to avoid overflow, the equivalent formula of the above equation is used:

$$loss = \max(x,0) - x^* y + \log(1+\exp(-abs(x))) \tag{33}$$

The return value problem is solved by the return network, and then the LSTM network is used as the basis to construct the Actor and Citric networks in the reinforcement learning AC algorithm.

4. Simulation and Case Study on Enhancement of Opera Performer's Voice Expression

4.1. Denoising results of improved CTF-GSC algorithm

The original signal waveform is shown in Figure 2.

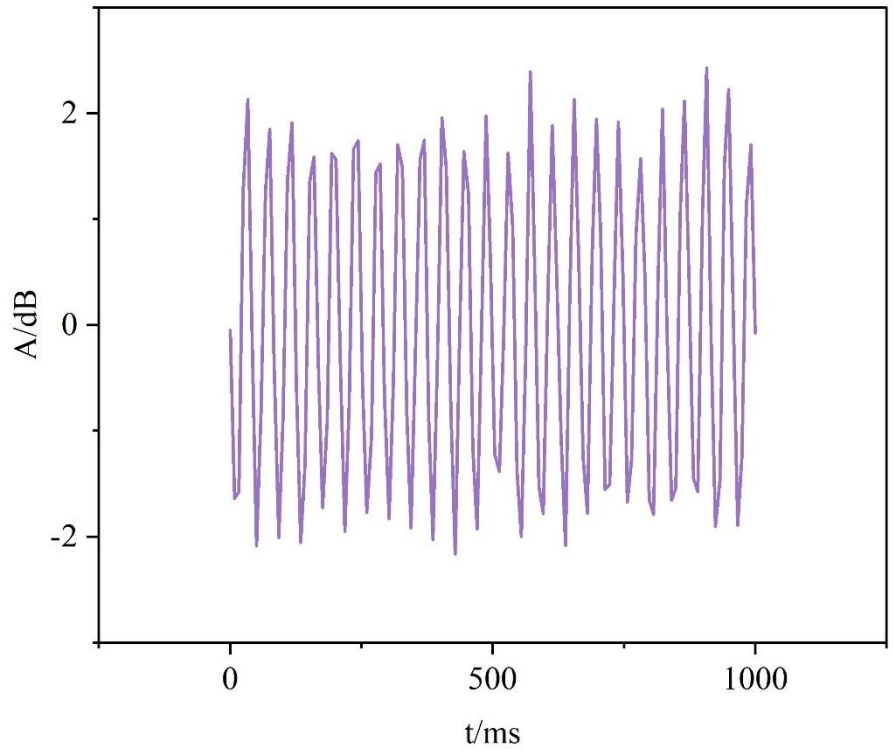


Figure 2. Original signal waveform.

Three microphones, spaced 1 cm apart, are uniformly distributed in a straight line to form a microphone array, with the angle between the incoming wave direction and the array at 30° and the incidence direction of the speech at 45°. The received signal is used for the simulation of the algorithm. The simulation assumes that the order of the filter is W=7, and the number of sampling points is 1000. After adding Gaussian white noise with a signal-to-noise ratio of 3 dB and mean value of zero to the original signal, the resulting waveform of the input signal is shown in Fig. 3.

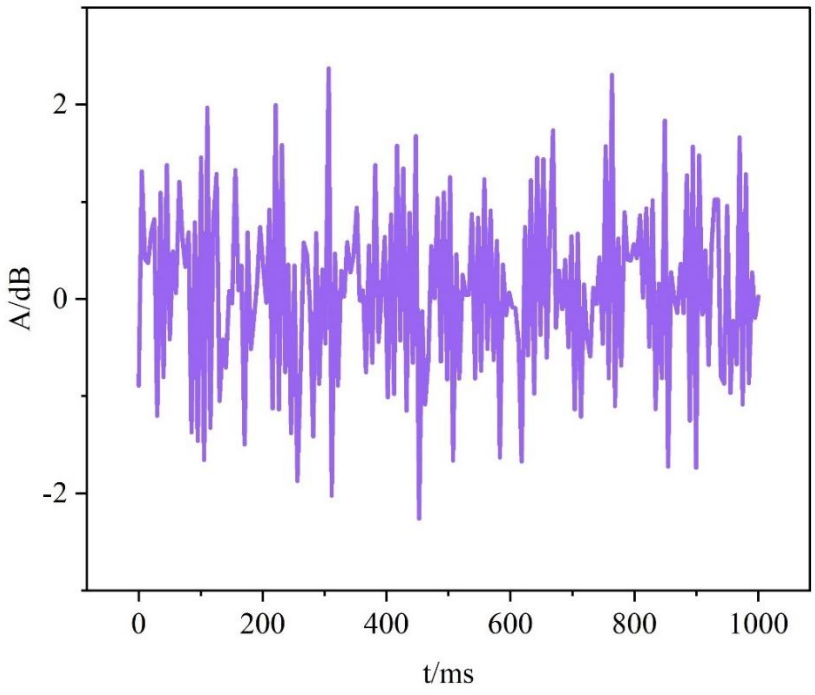


Figure 3. Input signal waveform with noise added.

After adding noise, the output signal waveform of the input signal after Wiener filtering is shown in Fig. 4. Through simulation, it is found that Wiener filtering is more effective in suppressing noise, and can suppress most of the doped noise, but not completely suppress all the noise. The limitation of Wiener filtering lies in the fact that it needs to know many parameter information of the original signal. This condition is often not satisfied in practical applications.

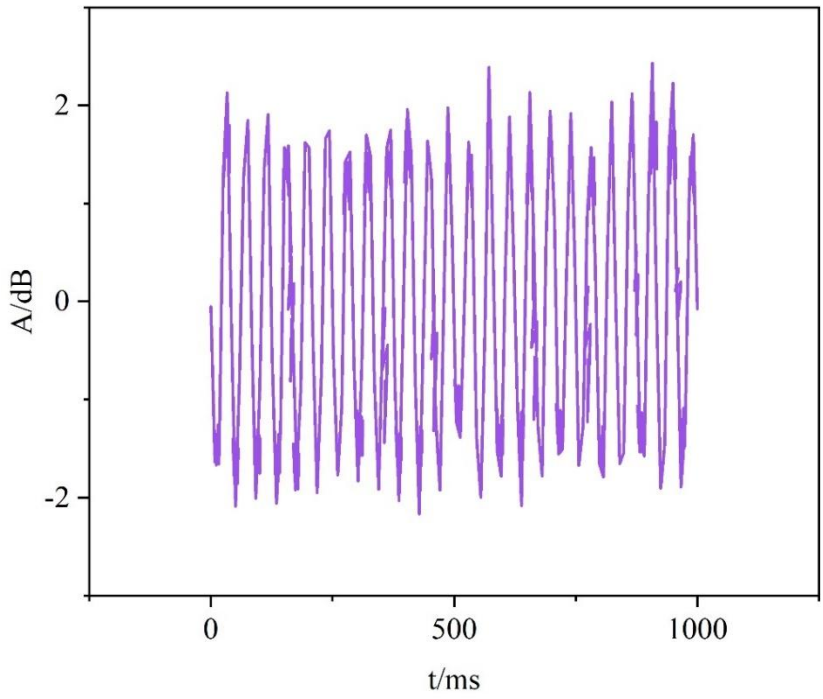


Figure 4. Output signal waveform after Wiener filtering.

After adding the noise, the waveform of the output signal after the input signal is adaptively filtered

by the GSC is shown in Fig. 5. As can be seen in Fig. 5, the denoising effect is good and most of the noise is suppressed because the adaptive filtering tracks the signal by adjusting the changes in its own filter parameters to find the minimum time difference of the output signal.

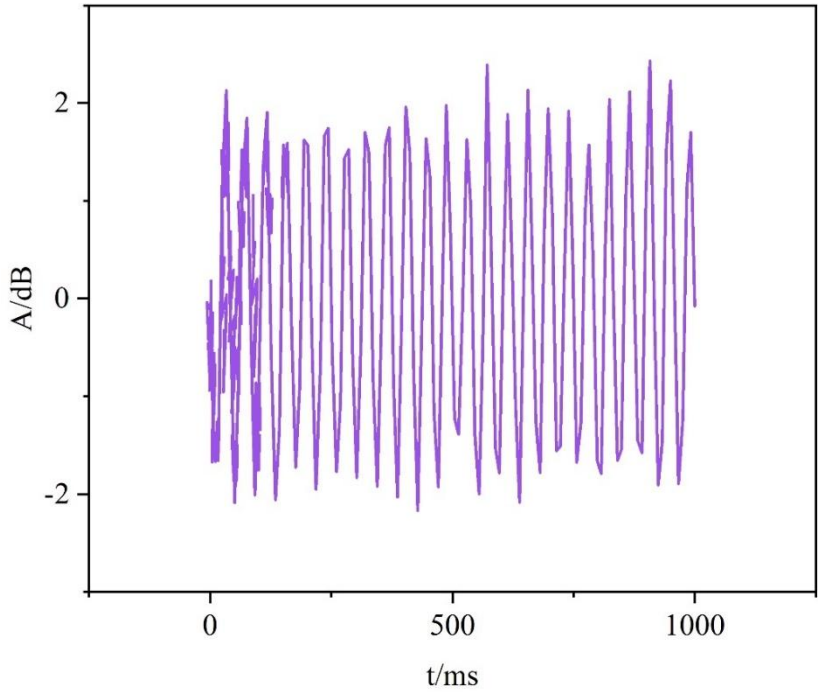


Figure 5. Output signal waveform after adaptive filtering.

The output signal waveform based on the CTF-GSC algorithm is shown in Fig. 6. From Fig. 6, it can be seen that the improved algorithm suppresses the noise well, both in the coherent noise environment and in the non-coherent noise environment, and basically can keep the same with the original signal waveform in Fig. 2.

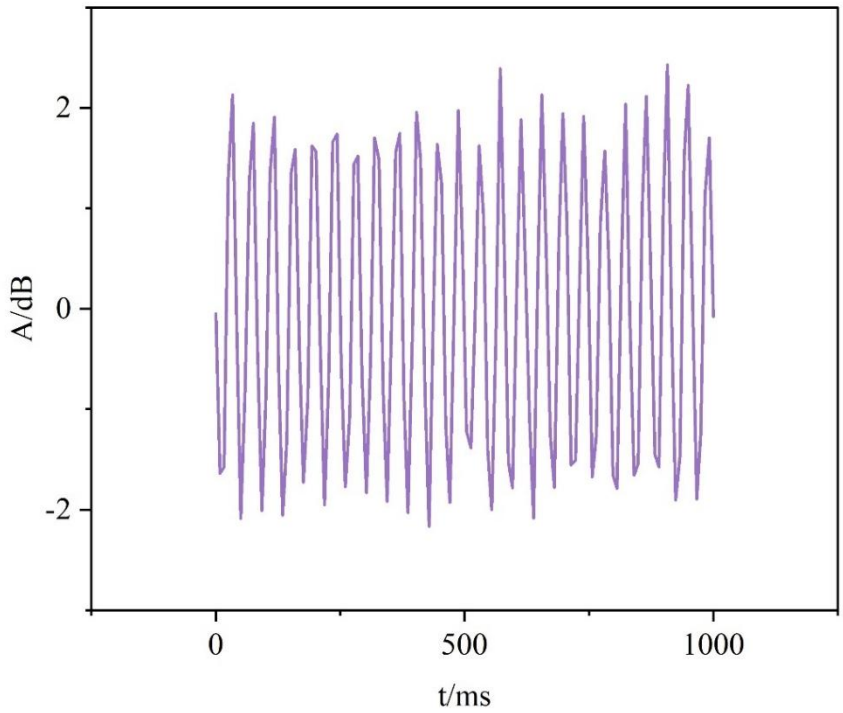


Figure 6. Output signal waveform after algorithm fusion.

4.2. Sound Expression Enhanced Melodic and Rhythmic Generation Results

4.2.1. Effect of different parameters on experimental results

In this paper, four sets of comparison experiments were done to test the effect of different numbers of neurons in the hidden layer on the experimental results, and the effect results are shown in Fig. 7. As can be seen from Fig. 7, the more the number of neurons, the stronger the network's ability to learn the nature of the dataset and abstract the features of the data, and the more effective it is to reduce the error between the predicted value and the target value, and the number of neurons is 512, the value of the loss is only 0.65. The size of the dimension of the hidden layer has a very important impact on the quality of the generated music, but the use of a deeper and wider neural network in the training phase requires more computational effort.

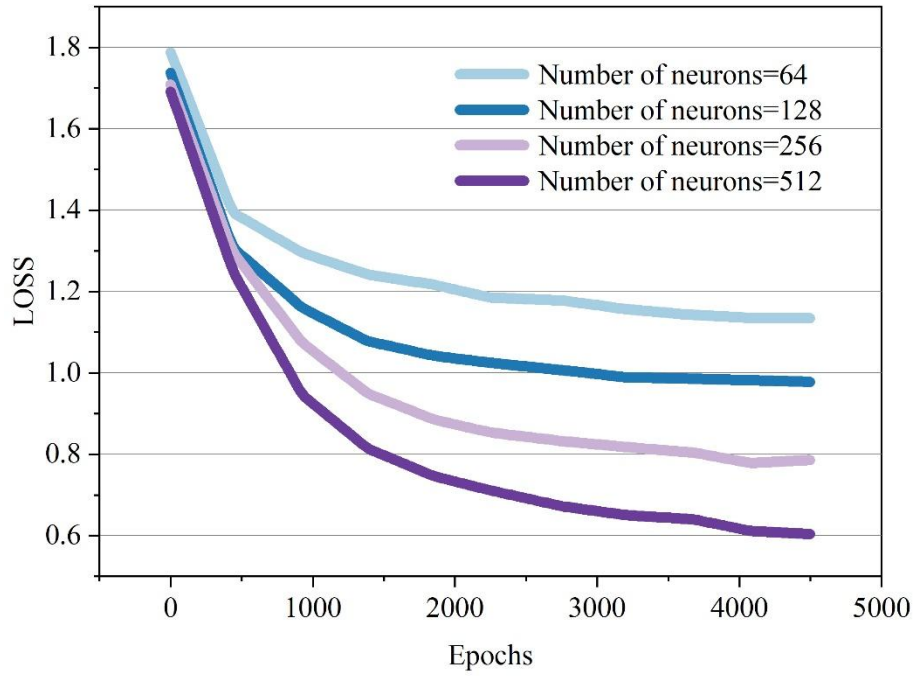
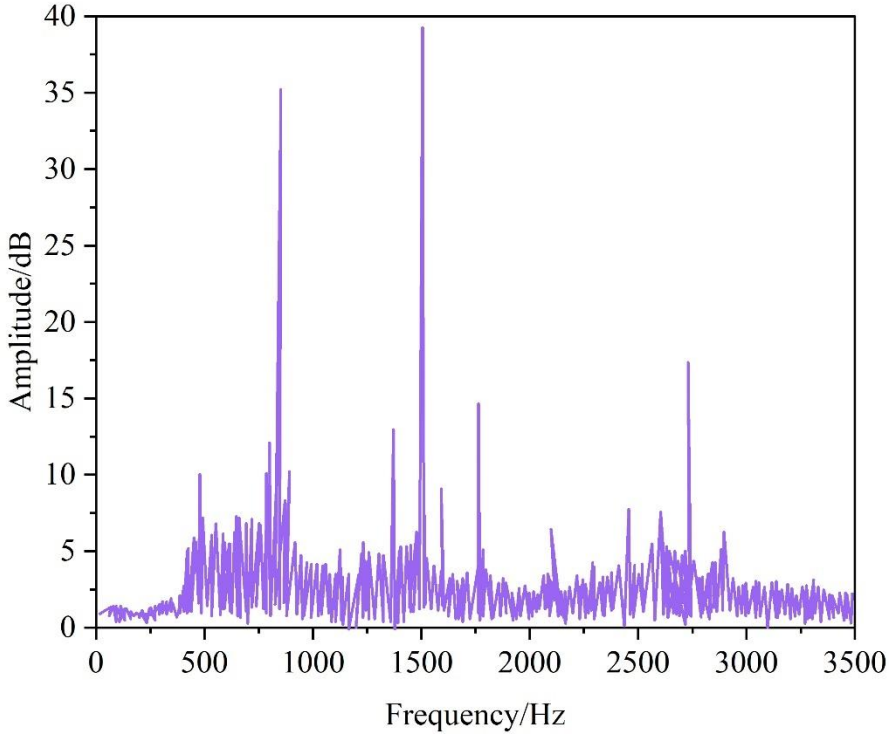
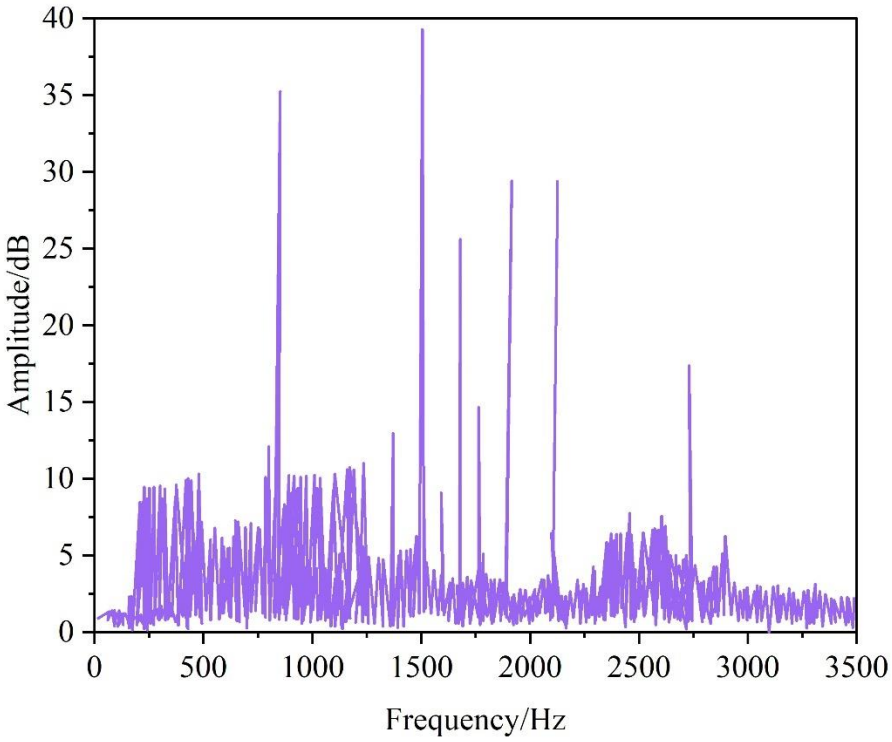


Figure 7. The effect of different neurons on the training effect.

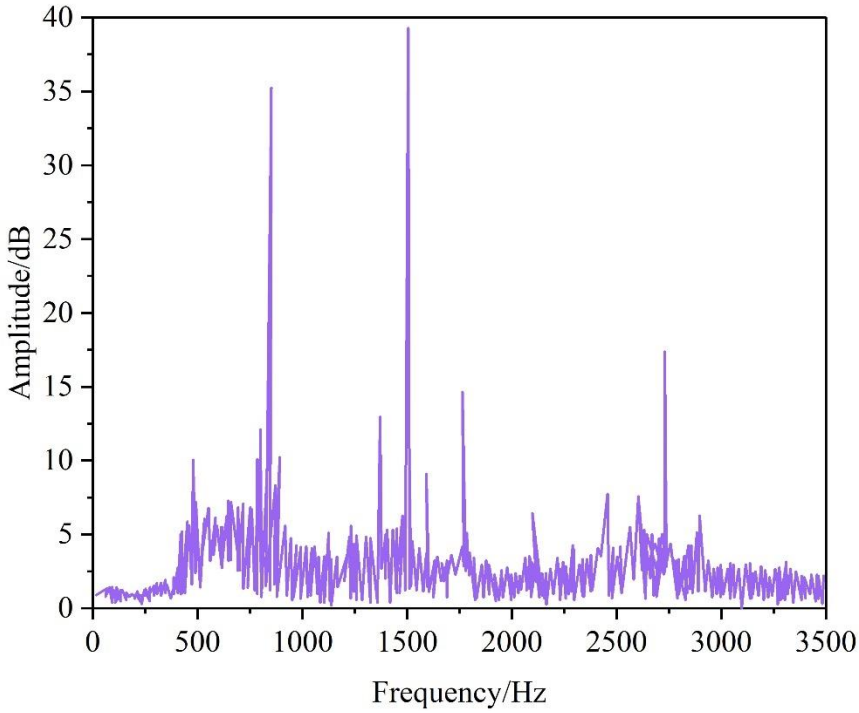
For the music files generated by different iterations, this paper does spectrum analysis for the generated music and sample music, and the results of spectrum analysis are shown in Fig. 8, Figs. (a)~(d) show the original sample music spectrum and the generated music spectrum with 1500, 2500 and 4500 iterations, respectively. It can be seen that the music generated at 1500 iterations has a lot of frequencies that the sample music does not have, so the generated music is also disorganized. At 2500 iterations, the frequency spectrum of the generated music starts to show the frequencies of the sample music, but the frequencies of the generated music are missing some frequencies in the sample music. When the iteration reaches 4500 times, the frequency distribution of the generated music sequence generally converges with the sample frequency distribution. It can be proved that the larger the number of iterations, the more the weighting parameters are learned and adjusted, which will improve the accuracy of the model to some extent.



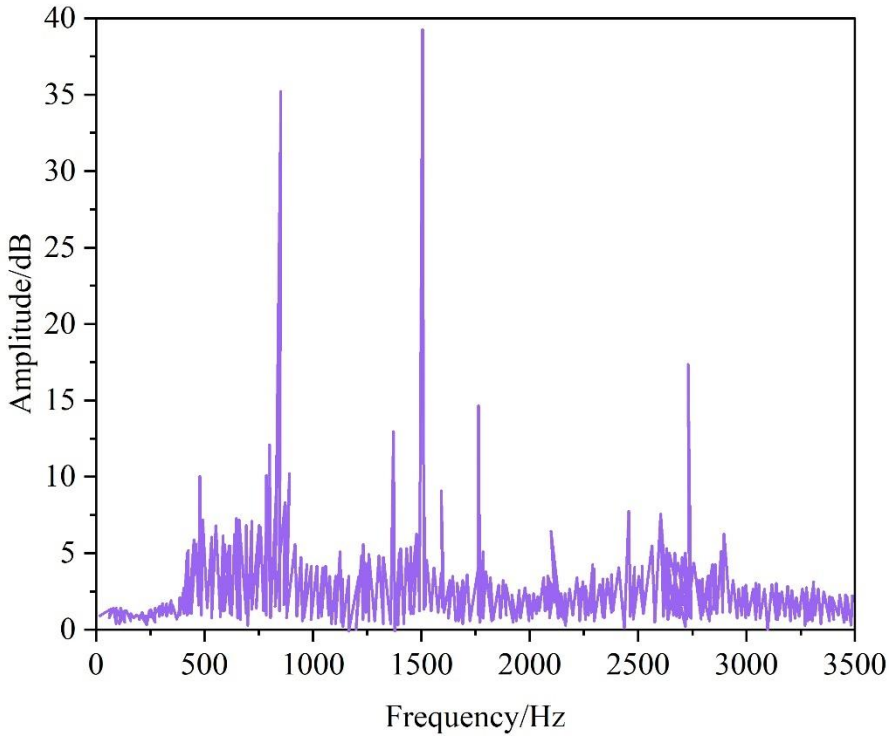
(a) Original sample music spectrum



(b) Iteration of 1500 music spectrum



(c) Iteration of 2500 music spectrum



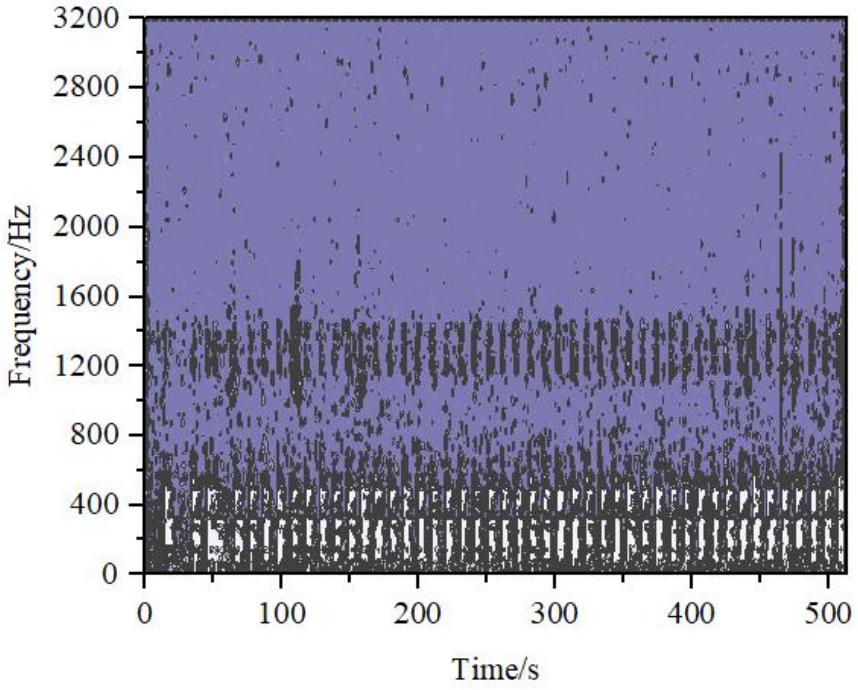
(d) Iteration of 4500 music spectrum

Figure 8. Spectral analysis.

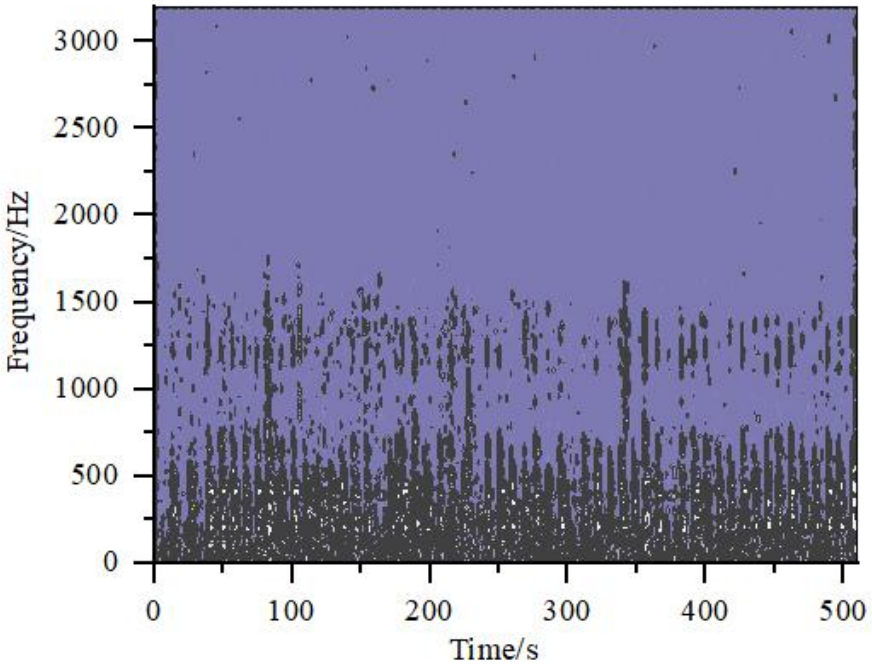
4.2.2. Generating music sound spectrograms for analysis

In order to show the melodic characteristics of the music generated by this paper's method more intuitively, the acoustic spectrogram analysis of the generated music and the sample music is added, and the results of the acoustic spectrogram analysis are shown in Fig. 9, and Figs. (a) to (c) show the acoustic

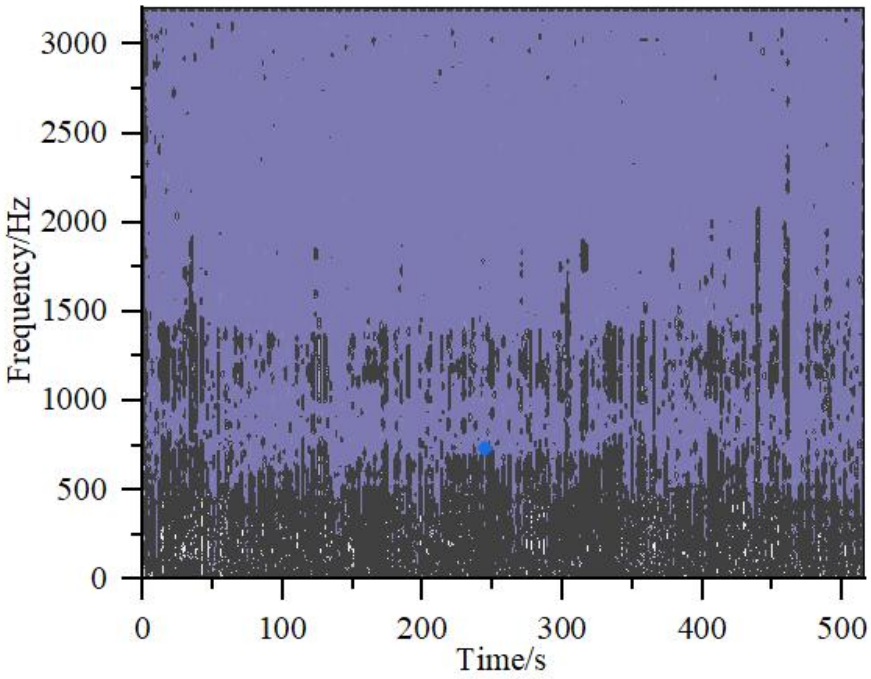
spectrogram of the original music, the acoustic spectrogram of the generated music, and the difference map between the original music and the generated music, respectively. It can be seen that the acoustic spectrograms of the music generated by this paper's method and the acoustic spectrograms of the sample music are very close to each other in the overall distribution, so it can be verified that the music generated by this paper's method has similar melodic characteristics as the sample music.



(a) Primitive music spectrum



(b) The music is generated



(c) Generate music and sample music differences

Figure 9. Sound spectrum analysis results.

4.2.3. Comparative analysis of the characteristics of generated music

In order to show the effectiveness of this paper's method, two sets of experiments are done in this paper, and the comparison network is a conventionally trained character-level LSTM-based music generation network. This network and this paper's method are set up by the same experimental environment, and each generates 400 pieces of music as test samples.

Twelve equal temperament is a method of musical law, which divides a pure octave into twelve equal parts, each of which is called a semitone, and is the predominant tuning method. The number of times the twelve equal-tempered notes are used in the two test samples are extracted and their statistical distributions are calculated, and the results are shown in Fig. 10.

The test samples generated by the conventionally trained LSTM have a high number of occurrences of the notes C, D, E, G, A, and B, and the remaining notes are almost absent, while the proportion of each note in the music generated by this paper's method is not much different. The comparison results can prove that this paper's method selects a larger variety of notes when generating music, i.e., the generated music tunes are richer.

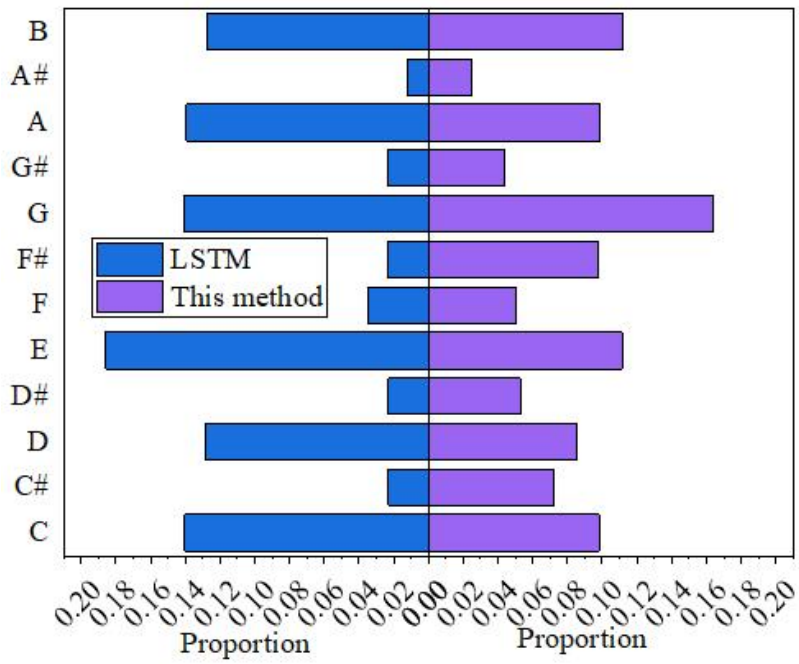


Figure 10. The statistical distribution of the two averages.

In order to show the effectiveness of the method of this paper even further, the representation of music theory rules was quantified. Seven effective feature information are extracted from the generated test sample music, compared with the known music rules, and the statistical data are calculated, and the results are shown in Table 1.

It can be seen that the music generated by this paper's method effectively avoids the phenomena of excessive note repetition and interval spanning, and the probability of a note being out of key is only 1.2%, and the generated music has a very obvious improvement in classical style compared to the conventionally trained LSTM, which is more in line with the music theory rules.

Table 1. Comparison of the rules of music theory.

Feature	Regular training LSTM/%	This method/%
Overrepetition	63.5	21.8
Average self-correlation	14.8	2.1
The note is no longer tuned	10.2	1.2
The interval is less than eight degrees	75.5	91.8
The biggest note of unique	68.4	58.4
Unique smallest note	58.1	57.4
Classical style	47.6	78.9

4.3. Results of musical analysis of vocal expressiveness of opera singers

In order to verify the artistic rationality and application value of the methodology proposed in this paper, Farewell, My Dreams of the Past is selected as the object of analysis, which provides an excellent model for studying how different actors use voice techniques to portray characters.

Farewell, sweet dreams of the past is one of the most central arias in the opera, expressing a terminally ill daughter's longing for love after receiving a letter from her father. The aria is sung in a single three-part sectional song with dynamic reproduction. The oboe's introduction foreshadows the central material of the aria - a motif that suggests a sense of longing. Violetta then sings two repetitions of this motif: “Farewell, the happy dreams of old, the rosy color in the cheeks is gone”. After a middle section and a recapitulation of gradually detached memories and longings, the coda again features a solo

oboe that brings the music back to a bleak reality, and Violetta[27] struggles with the motive “to stay” with the final result, as stated in the lyrics: "It's all over now! It's all over now".

Figures 11 and 12 show the IOI deviation and intensity analysis of the songs of two different opera singers, Callas and Muzio. By adding the IOI deviation analysis of the intensity presentation, the end of Callas's 1958 London version of Farewell to Dreams of the Past is compared with the recording left by Claudia Muzio, an Italian singer whom she greatly admired, in 1935. It can be seen that there are some similarities between the two in terms of intensity and deviation. It is believed that Callas had listened to Muzio's recordings on a regular basis and may have been more or less influenced by them. But the differences between the two are also obvious: Muzio expresses Violetta's physical and mental state at this point in time mainly through extra-musical means, such as short breaths, whereas Callas realizes the score's “broadening and fading” and “weak wisps of sound” more accurately and precisely through an extremely demanding control of the voice. The “wisp of a voice” is a requirement that Callas realizes more precisely through a very demanding modulation of the voice. Whereas Callas slows down at the end and the last note is airy, Muzio adopts a strong, albeit no less dramatic, finish that is a bit pushy.

The results of these analyses suggest that the ultimate goal of expressive sound enhancement methods is not to produce a standardized sound, but rather, like the artist, to provide a set of individualized, quantifiable aids to artistic expression.

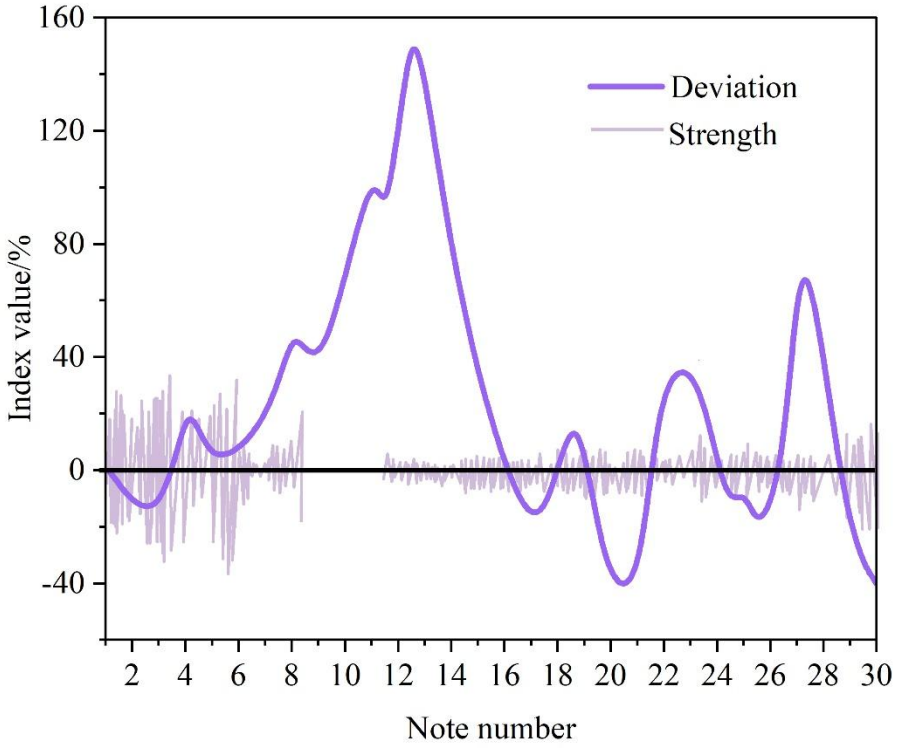


Figure 11. The ioi deviation and strength of the karas song.

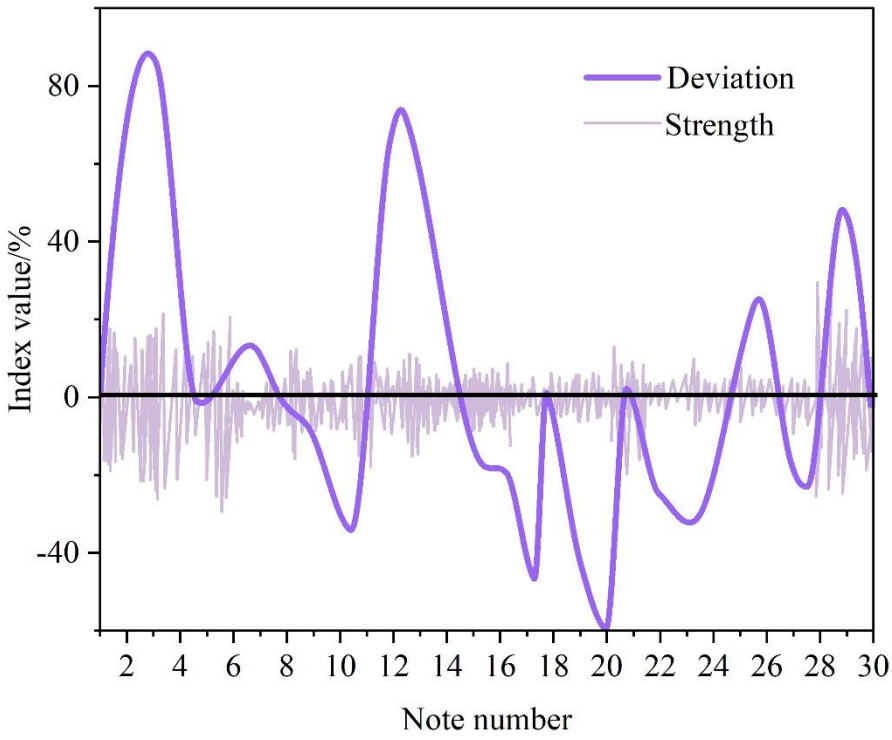


Figure 12. The ioi deviation and strength of muzio songs.

5. Conclusion

With the continuous development of deep reinforcement learning technology, many deep reinforcement learning techniques have now been applied to artificial intelligence music generation. In this paper, a comprehensive sound performance enhancement method based on deep reinforcement learning is successfully constructed.

Experimental results show that the algorithm proposed in this paper has a good ability to suppress both coherent and incoherent noise in the opera performance environment, and the sound signal waveform output by the algorithm can be consistent with the original sound signal waveform. It effectively reduces the distortion of the opera singer's voice in the noisy environment and improves the quality and expressiveness of the voice. And based on the two-layer unidirectional LSTM network and the reinforcement learning AC algorithm generates music that selects a larger variety of notes and a more balanced number of notes, and generates a more stable music structure, more personal style, which can enhance the expressive power of the sound.

The results of this paper confirm the potential of deep reinforcement learning technology in optimizing and generating works of art, which can be used in the daily training of opera singers and singing assistance during stage performances, injecting the vitality of modern science and technology into the classical art.

However, due to the complexity of music, the music generated by the method in this paper does not fully meet the actual needs of people in various neighborhoods, so further research is needed to explore more styles and different scenarios to improve the quality of music generation.

About the Author

Laijunwa Kong (1987.02-), female, Han, Zhengzhou, Henan Province, is a 2012 graduate of the Master's program at the Conservatory of Music in Ferrara, Italy. She currently serves as the director of the Paul Korne Opera Art Center at the School of Music and Dance, Henan Normal University, with her primary research focus on opera performance.

References

1. Carter, T. (2014). What is opera?. The Oxford handbook of opera, 16-16.
2. Wechsberg, J. (2023). The opera. Plunkett Lake Press.
3. Ardelean, I. (2018). Opera buffa, definition, origins and becoming until WA Mozart. Învăţământ, Cercetare, Creaţie, 4(1), 21-25.
4. Goehr, L. (2014). The concept of opera. Chapter, 5, 92-133.
5. HEBEISEN-MOȘUC, E. (2024). The Vocal Technique as an Instrument of Expression on Stage. Învăţământ, Cercetare, Creaţie, 10(1), 149-170.
6. Kostyuk, A. A., & Alekseeva, G. V. (2023). Emotions as a Phenomenon of Vocal and Opera Music. Russian Musicology, (1), 168-177.
7. Wilén, S. (2019). The play of vocal actors. Expanding the Space for Improvisation Pedagogy in Music: A Transdisciplinary Approach, 82.
8. Kruglova, M., Shcherbakova, A., Gareeva, A., & Vasilenko, A. (2021). Vocalist or actor: which one is better prepared to perform in the musical theatre genre?. Rast Müzikoloji Dergisi, 9(3), 3085-3104.
9. Sundberg, J., Salomão, G. L., & Scherer, K. R. (2024). Emotional expressivity in singing. Assessing physiological and acoustic indicators of two opera singers' voice characteristics. The Journal of the Acoustical Society of America, 155(1), 18-28.
10. Scherer, K. R., Sundberg, J., Fantini, B., Trznadel, S., & Eyben, F. (2017). The expression of emotion in the singing voice: Acoustic patterns in vocal performance. The Journal of the Acoustical Society of America, 142(4), 1805-1815.
11. Scherer, K. R., Sundberg, J., Tamarit, L., & Salomão, G. L. (2015). Comparing the acoustic expression of emotion in the speaking and the singing voice. Computer Speech & Language, 29(1), 218-235.
12. Xiao, C. (2021). Melody as a factor of vocal and performing interpretation of the opera image. Scientific Journal of Polonia University, 46(3), 117-122.
13. Nane, D. (2021). Emotional Expressivity in the Interpretation of Lyrical Characters. The Role of the Dramatic Training of the Opera Performer. Doctoral Horizons, 2(1), 41-49.
14. Williams, S. (2014). Opera and the Absence of Acting. The Oxford Handbook of Opera, 442.
15. Lisecka, M. (2018). "Offence to the Eye": Farinelli as an actor on Opera stage. Case from anthropology of theatre. Bulletin of the Transilvania University of Braşov, Series VIII: Performing Arts, 11(2), 73-86.
16. Radu-Giurgiu, C. (2022). A perspective on the opera audience of the 21st century. Studia Universitatis Babes-Bolyai-Musica, 67(1), 119-137.
17. Nguyen, T. T., Nguyen, N. D., & Nahavandi, S. (2020). Deep reinforcement learning for multiagent systems: A review of challenges, solutions, and applications. IEEE transactions on cybernetics, 50(9), 3826-3839.
18. Osband, I., Aslanides, J., & Cassirer, A. (2018). Randomized prior functions for deep reinforcement learning. Advances in neural information processing systems, 31.
19. Fenjiro, Y., & Benbrahim, H. (2018). Deep reinforcement learning overview of the state of the art. Journal of Automation Mobile Robotics and Intelligent Systems, 12(3), 20-39.
20. Viquerat, J., Meliga, P., Larcher, A., & Hachem, E. (2022). A review on deep reinforcement learning for fluid mechanics: An update. Physics of Fluids, 34(11).
21. Latif, S., Cuayáhuitl, H., Pervez, F., Shamshad, F., Ali, H. S., & Cambria, E. (2023). A survey on deep reinforcement learning for audio-based applications. Artificial Intelligence Review, 56(3), 2193-2240.
22. Xiao, P. (2025). Enhancing emotional expression in algorithmic music composition systems using reinforcement learning. Journal of Computational Methods in Sciences and Engineering, 14727978251352150.
23. Zhang, L., & Tian, Z. (2022). Research on music emotional expression based on reinforcement learning and multimodal information. Mobile Information Systems, 2022(1), 2616220.
24. Bratan, C. A., Tocila-Matasel, C. L. A. U. D. I. A., Andrei, A. G., Tebeanu, A. V., Franti, E., Dascalu, M. O. N. I. C. A., ... & Iorgulescu, G. (2024). The Observation of Actors' Vocal Emotion Exercises with Deep Learning and Spectral Analysis. WSEAS Trans. Inf. Sci. Appl, 21, 153-159.
25. A Utami, F Febriani, Z Irayani, C N Dewi, E A Ratnasari & F Nuraeni. (2024). Fast Fourier Transform (FFT) Application For Short-Term Earthquake Precursor Analysis Using Geomagnetic Data (Case Study of Kupang Geomagnetic Station, East Nusa Tenggara, Indonesia). Journal of Physics: Conference Series,2866(1),012059-012059.https://doi.org/10.1088/1742-6596/2866/1/012059.
26. Gers F A,Schmidhuber J & Cummins F. (2000). Learning to forget: continual prediction with LSTM.. Neural computation,12(10),2451-71.https://doi.org/10.1162/089976600300015015.
27. Emma Dillon. (2016). Violetta, Historian Verdi, 'Sempre libera' (Violetta), La traviata , Act I (1853). Cambridge Opera Journal,28(2),191-197.https://doi.org/10.1017/S0954586716000239.