Super Musician: Music Style Transfer Based on Autoencoder

Yining Cao, Guanru Wang , Zichen Fan, Shenghao Jiang, Fanghao Liu
{rimacyn, wguanru, zcfan, shhjiang, fanghaol}@umich.edu

Abstract

Music style transfer, which enables music professionals to work more productive and innovative, still remains under-explored areas. In this project, we summarized the related music/audio style transfer works and proposed an auto-encoder based music style transfer network - Super Musician. Inspired by AutoVC[1], Super Musician mainly contains 4 parts: style encoder, content encoder, decoder and spectrum inverter.

In this work we create a dataset of 3750 music clips of 5 different instrument for model training and evaluation. The style encoder and content encoder are trained separately with a carefully designed network structure and bottleneck. We implement a new spectrogram inversion method, which achieves better performance than the network proposed in AutoVC.

Results show that Super Musician is capable of switching music timbre between different instruments. The transformation performance varies on different instruments. The transformation between melodious tunes played by violin, clarinet and flute outperforms compact tunes played piano and guitar. We published a blog post[1] to show the results and audio demos.

1 Introduction

Domain transfer is becoming a popular field in the application of machine learning. Although different in definition between literature, domain transfer can be characterized as the operation of taking an object which belongs to one domain and map it to another domain as if it belongs to the other domain in the first place[2]. One of the significant domain transfer application, music style transfer, which enables music professionals to work more productive and innovative, still remains under-explored areas. Professional musicians often need to refer to different instruments while composing or creating cover songs. However, not all musicians know all kinds of instruments well. Inspired by this, we proposed Super Musician: a fast music style transfer algorithm, which aims at performing music domain transfer from one instrument (like flute) to another instrument (like violin). This project has its significance in the sense that using deep generative algorithms to do domain transfer can produce satisfying music [2] and at the same time greatly reduce the time needed for large scale music genre transfer. Difficulties of this project mainly arises from the synthesis of music. The capture of the relative values of timing and pitch pairs are the most important challenge for retaining the content of the music while at the same time do the domain transfer [3]. The detailed solution about how we address these challenges can be seen in the following report.

The report is structured as following: Firstly, Sec 1 and 2 introduce the background and related works. Then our proposed method and feasibility analysis are shown in Sec 3. In Sec 4, we will present our experiment method and some final results, showing the spectrum of the style-transferred music. Finally, Sec 5 concludes this report.

[1] Medium Blog: "Super Musician: Music Style Transfer Based on AutoEncoder"

2 Related Work

The concept of style transfer with deep learning algorithms is first applied to images using pre-trained convolutional neural networks[4]. In follow-up studies, deep generative model such as generative adversarial networks(GAN) and conditional variational autoencoder(CVAE) are being applied as new solutions to style transfer problems[5].

However, Effective audio synthesis remains challenging due to the difficulty in balancing global latent structure and locally coherent waveforms[6]. Most typical works in the field of automatic music generation cover Recurrent Neural Networks(RNN) and Long Short Term Memory Networks (LSTM)[2] with attention mechanism[3]. Further, a combination of convolutional and recurrent neural networks structure has been successfully implemented[7]. This project will focus on music genre transfer with deep generative model.Most recently, generative models have been successfully applied to music composition. Yu et al.[8] first applied RNN-based GAN to music generation. Brunner et al.[9] successfully used CycleGAN(CNN-based GAN) for symbolic music genre transformation.

Though GAN has been proved a solution to music synthesis by the google brain team[6], it is notoriously sophisticated to train. Qian et al.[1] recently proposed a innovative style transfer schema, AutoVC, which is constructed by merely an auto-encoder with a delicate bottleneck. Its efficacy and efficiency has been proved when applying to speech voice conversion. In this study, we will implement this schema for music genre transformation.

3 Proposed Method

3.1 Overall Architecture

As demonstrated in previous works, GAN is effective yet considerably sophisticated in training. AutoVC[1], the network that is referred in this project, solves the voice conversion problem with a more efficient auto-encoder framework. The overall architecture is shown in Figure(Fig. 1). Firstly, the pre-processed spectrum of the music pieces will be fed into content and style encoder to get the style and content embedding vectors, which separately represent the tone of the targeting instrument genre and the melody of a specific source instrument. Then, the content and style embedding will be up-sampled, concatenated and fed into the decoder to project to a new spectrum, which represents a new music piece with tone of the target instrument and melody from the source instrument. Finally, the new spectrum will be transferred to a music wave by other pre-trained conversion neural networks or external tools.

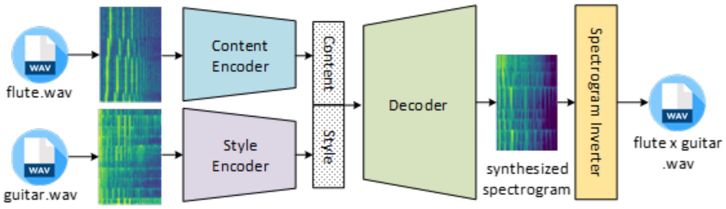


Figure 1: Super Mucisian: overall network architecture

Similar to the principle of conditional variational auto-encoder (CVAE), the whole network only needs to be trained based on the self-reconstruction loss, but we are trying to split the content and style information separately from the source and to perform a distribution matching property like GAN's. To implement this, the AutoVC paper proves a theorem to show that if we have a well-performed style encoder to extract the tone and choose the proper information bottleneck, the dimension of the content embedding vector, then the well-designed bottleneck will help the content encoder of the network to learn how to remove the style information from the source and only get the content information after down-sampling, which is the same goal of CVAE but is hard to guarantee during the CVAE training.

3.2 Feasibility Analysis

To model this problem, we need to assume that music waves and music melody are two stochastic process $W(t)$, $M(t)$ and the tone identity of the music is a random variable T. If the tone identity has a distribution of $p_T(\cdot)$ and the melody can be randomly sampled from a joint distribution $p_M(\cdot)$, the music waves can be randomly sampled from another joint distribution $p_W(\cdot|T, M)$ characterized by the distribution of the tone and the melody. Based on this model, consider two set of independent and identically distributed (i.i.d.) variables, (M_1, T_1, W_1) and (M_2, T_2, W_2). Our goal is to design a music style converter that produces the conversion output, $\hat{X}_{1\to 2}$, which preserves the melody of W_1 but using the instrument in W_2. For the components of the neural network, the content encoder and style encoder are modelled as two functions, $E_c(\cdot)$ and $E_s(\cdot)$. The decoder is modelled as a function $D(\cdot, \cdot)$.

During the conversion stage, we firstly feed the music W_1 into the style encoder and combine the output style vector S_1 with the original data W_1 together to get the metadata of W_1 and feed the metadata into the content encoder to get the content vector C_1. Then we also use the style encoder to get the style vector S_2 of music W_2. We feed both the target style vector S_2 and the source content vector C_1 into the decoder to get the the conversion output $\hat{W}_{1\to 2}$.

$$C_1 = E_c(\{W_1, E_s(W_1)\}), \quad S_2 = E_s(W_2), \quad \hat{W}_{1\to 2} = D(C_1, S_2)$$

During the training stage, we will firstly pre-train the style encoder $E_s(\cdot)$ to extract tone dependent embedding. Then we will feed the content encoder and the style encoder with the same tone but different melody music W_1 and W_1'. We hope the network can reconstruct the music W_1 itself.

$$C_1 = E_c(\{W_1, E_s(W_1)\}), \quad S_1 = E_s(W_1'), \quad \hat{W}_{1\to 1} = D(C_1, S_1)$$

Based on this reconstruction idea, the loss function is defined to minimize the weighted combination of the self-reconstruction error $L_{reconstr}$ and the content reconstruction error $L_{content}$, which is

$$\min_{E_s(\cdot), D(\cdot,\cdot)} L = L_{reconstr} + L_{content}$$

where

$$L_{reconstr} = \mathbb{E}\left\|\hat{W}_{1\to 1} - W_1\right\|_2^2$$

$$L_{content} = \mathbb{E}\left\|E_c(\{\hat{W}_{1\to 1}, E_s(\hat{W}_{1\to 1})\}) - C_1\right\|_1$$

As mention in the previous part, the AutoVC paper proves that based on the following assumptions the auto-encoder is enough to reach an ideal conversion property.[1]

Assumptions:

1. If $W_1 = W_2$, $E_s(W_1) = E_s(W_2)$.

2. If $W_1 \neq W_2$, $E_s(W_1) \neq E_s(W_2)$.

3. $X_1(t)$ has finite cardinality and is an ergodic stationary order-τ Markov process with bounded second moment, which is

$$p_{W_1(t)}(\cdot|W_1(1:t-1), T_1) = p_{W_1(t)}(\cdot|W_1(t-\tau:t-1), T_1)$$

4. Denote n as the dimension of C_1. If n^* is the optimal coding length of $p_{W_1(t)}(\cdot|T_1)^2$, n should satisfy $n = \lfloor n^* + t^{2/3} \rfloor$. n^* should be a constant if we continue to assume that each instrument produces the same amount of gross information, which is

$$H(W|T) = const$$

Then for each t, there exists a content encoder $E_c^*(\cdot; t)$ and a decoder $D^*(\cdot, \cdot; t)$, so that

$$\lim_{t\to\infty} L = 0$$

$$\lim_{t\to\infty} \frac{1}{t} \cdot KL(p_{\hat{W}_{1\to 2}}(\cdot|T_2, M_1)||p_W(\cdot|T = T_2, M = M_1)) = 0$$

where $KL(\cdot||\cdot)$ donates the KL-divergence.

From the first two assumptions, we can know that the style encoder need to be well-trained to distinguish the instruments for different musics during the style embedding. The third and forth assumptions tell us that if the length of the music t we choose is large enough, it is possible to set a properly bottleneck content vector dimension n that makes our loss function L and the KL divergence $KL(p_{\hat{W}_{1\to 2}}(\cdot|T_2, M_1)||p_W(\cdot|T = T_2, M = M_1))$ approximately equal to 0 during the training, where

$$p_{\hat{W}_{1\to 2}}(\cdot|T_2, M_1) = p_W(\cdot|T = T_2, M = M_1)$$

represents a desirable property of an ideal converter.

Based on the conclusion of this theorem, besides a well-trained style encoder, the most critical part of network training is to set proper content information bottleneck. If the bottleneck is too loose, the content embedding will be meddled with target genre; If the bottleneck is too narrow, the content information can be lost, which leads to failure in reconstruction. Only if the bottleneck was properly adjusted, the network can reconstruct the content, and the content embedding contains no source melody information.

4 Experimental Results

4.1 Dataset

To transfer the instrument from piano to another instrument using our neural network, a feature vector needs to be extracted to perform this operation. A list of the music instrument classification datasets has been investigated including the RWC, NSynth, Openmic etc.. However, none of these datasets are ideal for our case. In our case, desirable dataset would be a number of music clips with 10 seconds in length and the number of clips to be reasonably large. The length of the music clips is fixed to improve the feature extraction result. [10] A sampling rate of 16kHz is used in the data processing. According to previous works, no strict restrictions are imposed on the sampling rate of the input music clips. 16kHz[11], 22kHz [12] and 32kHz [13] are seen in previous works and worked fine. 16kHz were tested on our music clips and the sampled output preserved its original contents well.

Taking the consideration of available music pieces into account, the conversion between five instruments are investigated in our project, a dataset is constructed for this purpose. The five instruments are piano, guitar, flute, clarinet and violin. These five instruments are of quite different characteristics and if the transfer of music instrument worked, it can prove that our work is not too limited. Around 130 minutes of music clips for each type of music instrument are downloaded from various websites in the format of MPEG-3. [2] The music clips are of different length and are processed using python to form music clips of 10 seconds. As a result, in the dataset, there are 750 10-second-duration music clips for each instrument. Not all of the 130-minute clips are turned into data to be fed into the neural network because the last ten seconds of each music clip are mostly silent, so the last ten seconds of the music clips are simply discarded during the dataset processing. The link for the dataset used in this project is shown in the footnote below.[3]

4.2 The Style Encoder Net

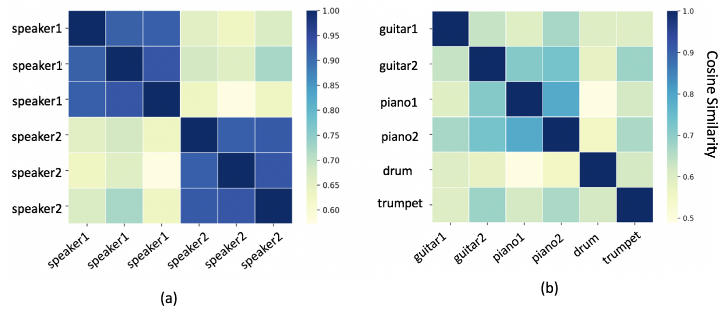
4.2.1 Performance of Existing Style Encoder Net

The style encoder net is essential for transformation as it gives a single embedding vector that represents high level features of certain instrument in a fixed dimensional space. In the original work of AutoVC[1], the style extraction network is a pre-trained model. However, in our first stage experiment, we found that this pre-trained model works specifically for speech encoder and fails to extract representative features for different instruments.

We tried 3 utterances for 2 different speakers and also different music segments for different instruments. As is shown in Fig 2, the original speaker embedding can distinguish utterance between

[2] Downloaded from : https://www.flutetunes.com
https://audionautix.com/
https://www.free-scores.com/index_uk.php

[3] https://drive.google.com/open?id=1RPOWaVrMw8ZsEDMHaJf6J3Dq4uJDdHb5



(a)

(b)

Figure 2: Experiment for style encoder net

different speakers but performs worse on distinguish different instruments.
There exist no style encoder network for music instrument, thus we decide to re-build our own style encoder network.

4.2.2 Network Structure

We train the style network based on the network structure proposed by Heigold, et al[14]. The original network use features extracted from each utterance is first fed into a Long short-term memory recurrent neural network (LSTM). A linear fully connected layer connected to the last LSTM layer to extract a single vector for style representation.
The original paper proposed a two layers LSTM for style embedding. For this specific problem, we added an additional LSTM layer to get an optimal performance. The model architecture we implemented is depicted in Fig 4.

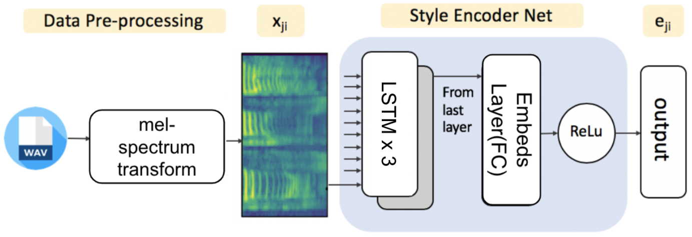


Figure 3: Style encoder net structure

The hidden cell size for the LSTM layer is 768 and the final embedding size is 256.

4.2.3 Training Method

For data pre-processing, we derive a 256 dimensional mel-spectrogram series for each wave segments based on the 16k sampling rate.
We implement similar training methods as Wan, Li, et al [15] as well as their proposed Generalized end-to-end (GETE) loss function, which gives a state-of-art performance in speech verification.
For each batch of training data, we include NxM music segments: N different instruments and M different segments for each instruments. We use x_{ji} represents the features extracted from instrument j segment i and $f(x_{ji};w)$ represents the output from the style encoder net with learnable parameters w. The final embedding is in the L2-norm form:

$$e_{ji} = \frac{f(x_{ji};w)}{\|f(x_{ji};w)\|_2}$$

For all segments for the same instruments(i.e. same j), we define the center(c_k) and similarity(S_{ji}) as below:

$$c_k = \frac{\Sigma_{m=1}^{M} e_{km}}{M}$$

$$S_{ji} = cos(e_{j,} \ , c_k) = \langle e_{j,} \ , c_k \rangle$$

We put a softmax on $S_{ji,k}$. The GETE loss for a single embedding e_{ji} is defined as:

$$L(e_{j,i}) = -S_{ji,j} + log\Sigma_{k=1}^{N} exp(S_{ji,k})$$

The total loss is simply the sum of loss over all segment embeddings in one training batch. We choose the adam optimizer and 0.001 learning rate for updating. Our training code is implemented with PyTorch with a reference to an existing github repository Resemblyzer.[4]. Detailed training code can be found in our **Esnet** branch.[5]

The Sytle Encoder net was trained on our self-prepared dataset. Considering the limitation of the dataset volume, we implemented a 10-fold cross validation method as suggested by Kuhn et al[16]. For each batch, 25 periods of music are sampled (5 in each music instruments). The training was conducted for 100 epochs with the GETE loss converges at 0.09.

4.2.4 Style Encoder Net Evaluation

The testing dataset contains 50 periods (10 for each instruments). The GETE loss for the test dataset is 0.16. To have a more intuitive sense of the model performance, we choose three samples from each instrument and plot the similarity between embeddings:

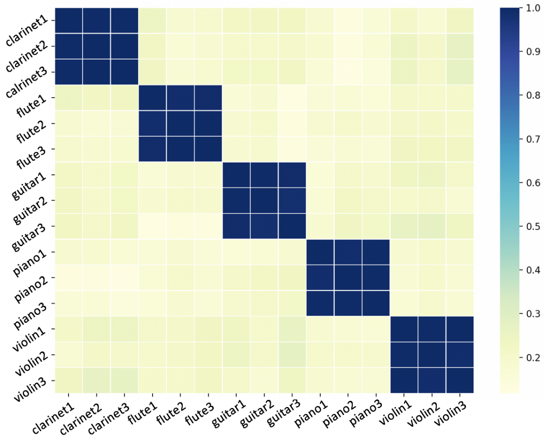


Figure 4: Results for retrained Esnet

Result shows that the retrained Style Encoder Net is able to differentiate the belch timbre of different instruments.

4.3 The Content Encoder and Decoder

We have trained the content encoder and decoder using a self-build dataset (5 instruments, flute, guitar, piano, clarinet and violin. Each instrument contains 130 minutes of solo pieces). The network structure of the content encoder and decoder we implemented is shown in the Fig. 5(similar to AutoVC). We send a mel-spectrum into the content encoder, embed it into a lower dimension vector, and then concatenate with the style embedding for each time step. The concatenated features are fed into the decoder and then recovered to the same dimension of the input mel-spectrum by up-sampling.

During the network training, because we assume that we have a perfectly trained style encoder, we only train the weights in the remaining autoencoder. We send different melodies of the same instrument into the style and content encoder, combining the self-reconstruction error with the content embedding reconstruction error to build the loss function, which have been proved enough to produce an ideal converter.

[4] Refer to : https://github.com/resemble-ai/Resemblyzer

[5] https://github.com/fairchildfzc/AutoVC_music/tree/EsNet

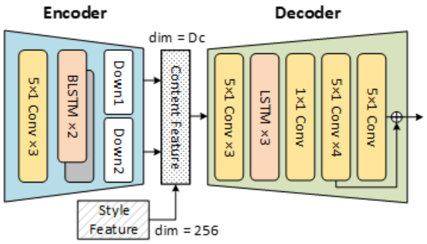


Figure 5: The content encoder and decoder structure

Because we have already had a perfect style embedding, the size selection for content embedding is the most important part of this method. For a too wide size, the content embedding will contain some style info from the original instrument. Also, for a too narrow size, we will have some loss of the target content during the conversion. So In the training, we will change the dimension of the content feature vector (Dc in Fig.5). By choosing the proper size of embedding, we can get purely content info from the target music and perfect style info from the well-trained style encoder. During the conversion test, we only need to send different music of different instruments into style and content encoder separately.

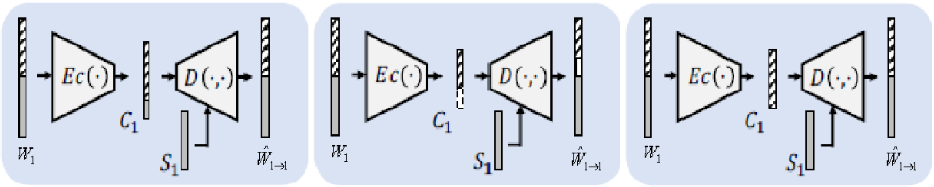


Figure 6: The content embedding dimension selection. Left: Too wide embedding, containing some style info from the source. Middle: Too narrow embedding, losing some content info from the source. Right: Proper size embedding, perfectly extracting the content info.

4.4 The Spectrogram Inverter

4.4.1 Performance of existing spectrogram inverter

The spectrogram inverter converts spectrograms to audio waves after the melspectrogram generated. In our implementation, we firstly tried the conventional method: Griffin-Lim Algorithm (GLA). GLA is used for signal estimation from short-time fourier transformation (STFT). This method is based on the redundancy of STFT. By implementing this method, we first inverse melspectrogram to STFT spectrum, then use GLA to convert spectrums to audio signals. However, we found the quality of the audio produced by GLA is low, owing to the lack of prior knowledge of the target signal. In the AutoVC paper, after the melspectrograms are generated, the WaveNet[17] vocoder is applied. WaveNet is a deep neural network for generating raw audio waveforms. We tested the pre-trained wavenet in melspectrogram-audio conversion. The result is shown below (Fig. 7 Left and Medium). The result shows that the WaveNet can generate a similiar audio as the input. However, as shown in the figure above, the melspectrograms of the vocoder generated audio contains some noises and not clear. The probable reason maybe WaveNet model we used is provided by AutoVC, which is pre-trained on human voices. Another problem is that the WaveNet contains 40 layers and the computation burden is huge. For instance, the generation time for a 10s music fragment costs nearly 30 minutes. Therefore, a more accurate and efficient spectrogram inverter should be proposed.

4.4.2 Phase gradient heap integration (PGHI) based spectrogram inverter

Phase Gradient Heap Integration (PGHI)[18] is a non-iterative algorithm for the phase reconstruction from the STFT magnitude. Since the computation is non-iterative, the algorithm is very fast and is suitable for long audio signals. As mentioned in the paper, PGHI outperforms other traditional method such as GLA, IBFGS[19] and TF-RTISI-LA[20]. TifGAN[21] used this method in their

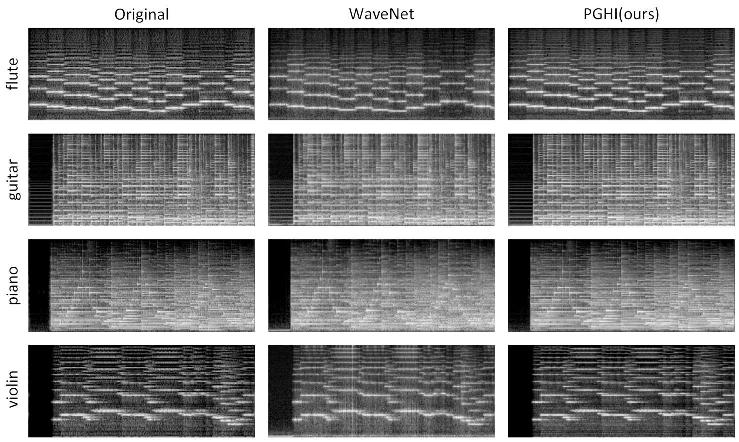


Figure 7: Melspectrograms of audio samples. Left: The original audio samples' spectrum. Middle: The spectrums of audio samples generated by WaveNet. Left: The melspectrum of the PGHI recover audio.

paper and got good results. The result of the PGHI based spectrogram inverter can be shown in Fig. 7 Right. The PGHI melspectrum shows more details and the image is more clear. Moreover, this method does not need for training, it is compatible for both instruments and human voice. The conversion speed is much faster than wavenet-based vocoder, taking less than 1s to convert one melspectrum to audio.

4.5 Music style transfer results

Fig. 8 shows the loss values during the training. Fig. 9 shows one example result. The x axis of the melspectrum is time and the y axis is frequency. As seen in the figure, the original flute has more components in low frequency domain (brighter the larger), and the violin has more details in higher frequency. The converted music attains the flute's melody and shows more features of violin in frequency domain. As for the style transfer of other instruments, we were not able to obtain satisfying results. This was mainly due to two reasons. In the first place, our dataset was not ideal. Some of the music pieces within the dataset contains multiple sound tracks, which caused some distortions in the extraction of the style and content vector. In the second place, the style of, e.g. piano and violin differs varies greatly, resulting in a great difference in the style vector. We were not able to obtain satisfying results due to these two reasons. The satisfying transfer is the flute to violin transfer.

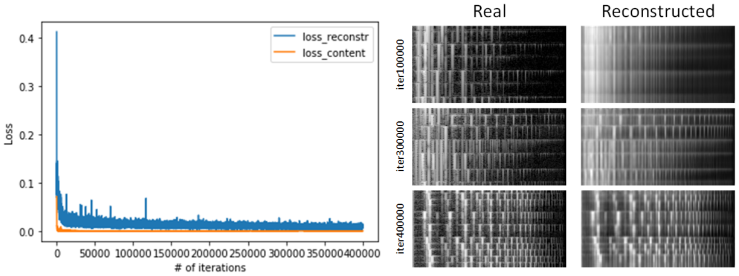


Figure 8: Training loss.

5 Conclusion

In this project we have built Music instruments Transfer Based on Autoencoder. all modules including content encoder style encoder, decoder, and spectrogram inverter have been implemented. The design fulfilled novelty that adopt Autoencoder for music genre transfer where similar task were previously

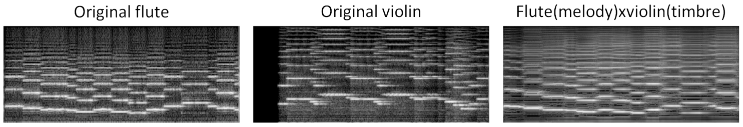


Figure 9: Music style conversion example: The converted audio attains the melody of original flute and the timbre of violin.

performed by GAN. The network produced satisfactory change in music transfer between violin and flute, however the result still can be improved by providing more training data to avoid over-fitting and single-chord pieces for more accurate training results.

Author Contributions

All five teammates firstly do literature review together. Yining Cao is responsible for style encoder training. Guanru Wang is responsible for feasible analysis and content encoder-decoder training. Zichen Fan is responsible for content encoder-decoder training and spectrogram inverter design. Shenghao Jiang and Fanghao Liu is responsible for dataset construction and system testing.

References

[1] Kaizhi Qian and et al. Zero-shot voice style transfer with only autoencoder loss. *arXiv preprint arXiv:1905.05879*, 2019.

[2] Gino Brunner and et al. Jambot: Music theory aware chord based generation of polyphonic music with lstms. In *2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 519–526. IEEE, 2017.

[3] Cheng-Zhi Anna Huang and et al. Music transformer. *arXiv preprint arXiv:1809.04281*, 2018.

[4] Leon A Gatys and et al. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2414–2423, 2016.

[5] Jun-Yan and Zhu and et al. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.

[6] Jesse Engel and et al. Gansynth: Adversarial neural audio synthesis. *arXiv preprint arXiv:1902.08710*, 2019.

[7] Daniel D Johnson. Generating polyphonic music using tied parallel networks. In *International conference on evolutionary and biologically inspired music and art*, pages 128–143. Springer, 2017.

[8] Lantao Yu and et al. Seqgan: Sequence generative adversarial nets with policy gradient. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[9] Gino Brunner and et al. Symbolic music genre transfer with cyclegan. In *2018 IEEE 30th International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 786–793. IEEE, 2018.

[10] Noris Mohd Norowi, Shyamala Doraisamy, and Rahmita Wirza. Factors affecting automatic genre classification: an investigation incorporating non-western musical forms. In *Proceedings of the International Conference on Music Information Retrieval*, pages 13–20, 2005.

[11] Alan P Schmidt Trevor KM Stone. Music classification and identification system.

[12] Daniel Piccoli, Mark Abernethy, Shri Rai, and Shamim Khan. Applications of soft computing for musical instrument classification. In *Australasian Joint Conference on Artificial Intelligence*, pages 878–889. Springer, 2003.

[13] Roisin Loughran, Jacqueline Walker, Michael O'Neill, and Marion O'Farrell. Musical instrument identification using principal component analysis and multi-layered perceptrons. In *2008 International Conference on Audio, Language and Image Processing*, pages 643–648. IEEE, 2008.

[14] Georg Heigold, Ignacio Moreno, Samy Bengio, and Noam Shazeer. End-to-end text-dependent speaker verification. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5115–5119. IEEE, 2016.

[15] Li Wan, Quan Wang, Alan Papir, and Ignacio Lopez Moreno. Generalized end-to-end loss for speaker verification. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4879–4883. IEEE, 2018.

[16] Max Kuhn and Kjell Johnson. *Applied predictive modeling*, volume 26. Springer, 2013.

[17] Aaron van den Oord and et al. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[18] Zdeněk Prša, Peter Balazs, and Peter Lempel Søndergaard. A noniterative method for reconstruction of phase from stft magnitude. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25(5):1154–1164, 2017.

[19] Rémi Decorsière, Peter L Søndergaard, Ewen N MacDonald, and Torsten Dau. Inversion of auditory spectrograms, traditional spectrograms, and other envelope representations. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(1):46–56, 2014.

[20] Jonathan Le Roux, Hirokazu Kameoka, Nobutaka Ono, and Shigeki Sagayama. Phase initialization schemes for faster spectrogram-consistency-based signal reconstruction. In *Proceedings of the Acoustical Society of Japan Autumn Meeting*, number 3-10, page 3, 2010.

[21] Andrés Marafioti, Nicki Holighaus, Nathanaël Perraudin, and Piotr Majdak. Adversarial generation of time-frequency features with application in audio synthesis. *arXiv preprint arXiv:1902.04072*, 2019.