TRANSFERRING THE STYLE OF HOMOPHONIC MUSIC USING RECURRENT NEURAL NETWORKS AND AUTOREGRESSIVE MODELS

Wei-Tsung Lu and Li Su
Institute of Information Science, Academia Sinica, Taiwan
s603122001@gmail.com, lisu@iis.sinica.edu.tw

ABSTRACT

Utilizing deep learning techniques to generate musical contents has caught wide attention in recent years. Within this context, this paper investigates a specific problem related to music generation, music style transfer. This practical problem aims to alter the style of a given music piece from one to another while preserving the essence of that piece, such as melody and chord progression. In particular, we discuss the style transfer of homophonic music, composed of a predominant melody part and an accompaniment part, where the latter is modified through Gibbs sampling on a generative model combining recurrent neural networks and autoregressive models. Both objective and subjective test experiment are performed to assess the performance of transferring the style of an arbitrary music piece having a homophonic texture into two different distinct styles, Bachs chorales and Jazz.

1. INTRODUCTION

Automatic music generation is gaining traction in the music industry because of its potential in mass producing music according to a user-assigned *style*, such as genre or mood. For example, the artificial intelligence (AI) music composition service, Jukedeck, supports four genre options (i.e., folk, rock, electronic, and ambient) and allows users to choose how the music feels (i.e., ambient, sparse, meditate, and sci-fi) [1]. Most of the existing advanced techniques employ deep learning techniques to perform end-to-end generative modeling of a music style based on a symbolic music format such as the musical instrument digital interface (MIDI) [3]. Various kinds of model configurations were explored in this fashion, such as the encoder-decoder framework [16], generative adversarial networks (GAN) [6], autoregressive models [13], variational autoencoders (VAE) [8], long-short-term memory (LSTM) networks [7, 12], recurrent Boltzmann machines (RBM) [2] and tied parallel networks [10]. These models are designed for two slightly different scenarios of music generation: one is to simply generate music by taking noise as input [16], while the other is to generate adapted accompaniment or voices for a given melody or a lead sheet, also known as *reharmonization* [7] or *reorchestration* [14].

In this paper, we discuss the *music style transfer* problem. More specifically, we aim at *rearranging* the elements that highly affects style of a given music piece (e.g., rhythm patterns in the accompaniment) while preserving the essence (e.g., melody and chord progression) of that piece. This problem has been of interest for a long time; previous related studies include the use of genetic algorithm (GA) [15] and optimization approaches [19]. In contrast to reharmonization, the style transfer problem in this paper uses the entirety of a music piece, including all voices and accompaniment, as the input of a model. In other words, we aim to obtain a system that can automatically determine which part of an input music is to be preserved and which part is to be adapted to another style.

Besides, by leveraging the end-to-end modeling capability of deep learning, we investigate the potential of a single neural network to model two or more distinct styles based on training data of each style. To solve the style transfer problem, there are two main challenges, model complexity and the diversity over various musical genres. For model complexity, since the DeepBach model is designed for generating *polyphonic* music that only has a fixed number of voices (i.e., number of polyphony) such as Bach's four-part chorales instead of music having varying numbers of polyphony, such as Jazz music, the number of voices in the DeepBach model needs to be increased to accommodate the maximum number of voices. For example, the DeepBach model can be extended to 10 voices or more, but doing so also increases the model size and complexity considerably. For the second challenge, the diversity of various musical genres means that different music styles correspond to different preferable model setting, making it virtually impossible to develop a universal framework applicable to all kinds of music. For example, [14] proposed a solution to generate music having different styles but had to adopt different models for those styles.

This paper proposes a solution of music style transfer with one single generalized model. It assumes that input music is *homophonic* music decomposable into a *predominant melody* and an *accompaniment* part. Therefore, we only need to consider style transfer of the accompaniment; the melody, while being unaltered in the output, can be used as a condition of the network. We employ a DeepBach-based model to model temporal in-

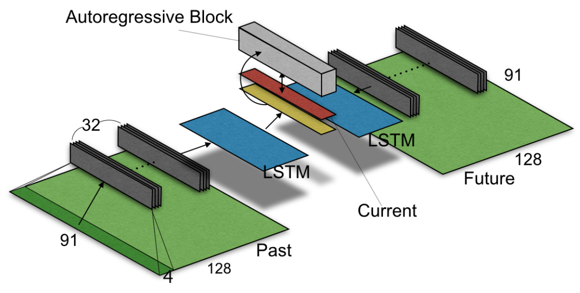


Figure 1: The proposed model for music style transfer. The past and future part of the input data (in green) is first transferred to another feature mapping (in dark gray) through a shared fully-connected network. The LSTM network then takes this feature mapping as input and outputs a 150-by-2 matrix (in yellow), which acts as the condition of the autoregressive model (in light gray). Finally, the autoregressive model predicts the activation of the interested note given the current part of the input data (in red).

formation, and combine this model with an autoregressive model (Wavenet [13]) to model the pitch information without restricting the number of note co-occurrence in a single frame. Experiment results of transferring the style of an arbitrary homophonic music piece to Bachs chorale and Jazz styles are provided in this paper since these two styles are arguably the two most extensively investigated music styles in the literature of automatic composition. The source code and listening samples are available on-line. [1]

2. MODEL

Our proposed model of an input music piece is shown in Fig. 1. The model contains two LSTM networks, with one taking data preceding a reference time as input and another taking data following the reference time as input. In addition, a specific autoregressive model, WaveNet, is used to process the data of the reference time. Details of the proposed model are discussed below.

2.1 Data representation

We represent a music score as a *piano-roll* matrix $S \in \mathbb{R}_+^{I\times J}$ and a *metadata* matrix $M \in \mathbb{R}_+^{3\times J}$. The concatenated matrix $[S; M]$ is then used as the input of the system. The element at the i-th row and the j-th column of S is denoted as S_{ij}, representing the pitch activation at pitch i and at time j, where $i \in [1, I]$, $j \in [1, J]$, $I = 88$, and J is the number of time steps of the music piece. $S_{:j} \in \mathbb{R}^I$ is then the j-th column of S, representing the pitch profile at time j. To represent homophonic music data, a note activation, S_{ij} for $i \in [1, 88]$, is represented as a Bernoulli random variable, and $S_{ij} = 1$ if there is a note activation at pitch i and time j, and $S_{ij} = 0$ if no note activation occurs at time j. Since the number of polyphony of homophonic music varies with time, $S_{:j}$ becomes a multi-hot vector, where its number of non-zero elements varies with j.

The metadata matrix M describes the time grids, the start and the end symbol of the music piece, thereby forming a 3-by-J matrix. The time grids used in this paper are the same as the ones in DeepBach: each beat interval is divided into four subdivisions, and are indexed by 1, 2, 3, and 4 respectively. The starting time and ending time are denoted as 1 and others as 0. As a result, the dimension of the input, $[S_{:j}; M_{:j}]$, is 91.

To facilitate our discussion, we simplify the input data in the following two ways. First, sustained note are considered as repeating notes with the same pitch . Secondly, any two notes with the same time and pitch are considered as one note.

2.2 Model Architecture

The proposed model with parameterization θ is obtained by the following optimization problem:

$$\max_{\theta} \sum_{ij} \log p\left(S_{ij} = 1 | S_{\setminus ij}, M, \theta\right) . \qquad (1)$$

The formulation (1) can be viewed as a generalized version of the original DeepBach network discussed in [7], where the number of voices is fixed at 4 and the pitch of each voice is modeled individually:

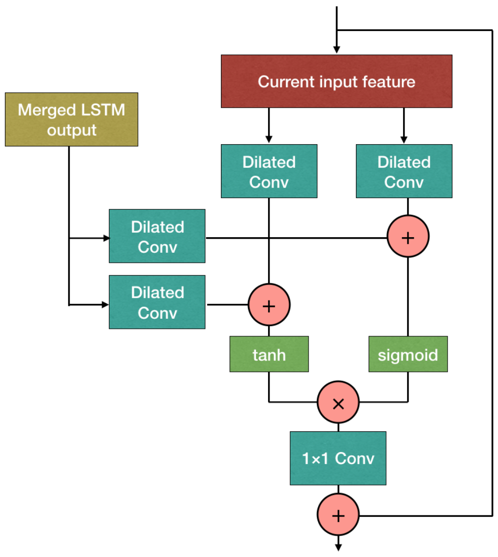
$$\max_{\theta_i} \sum_{j} \log p_i\left(V_{ij} | V_{\setminus ij}, M, \theta_i\right) , \text{ for } i \in [1, 4] . \qquad (2)$$

The data representation $V_{ij} \in \mathbb{R}^{4\times J}$ in (2) is different from S_{ij}; V_{ij} is the pitch number of the i-th voice (i.e., 1 for soprano, 2 for alto, 3 for tenor, and 4 for bass) at time j. The four networks in DeepBach have the same structure, each processing only one part of the four-part Bach's chorales and producing an output limited to monophonic music. By using the four networks together, the conditional probability of notes occurred simultaneously in different parts can be covered.

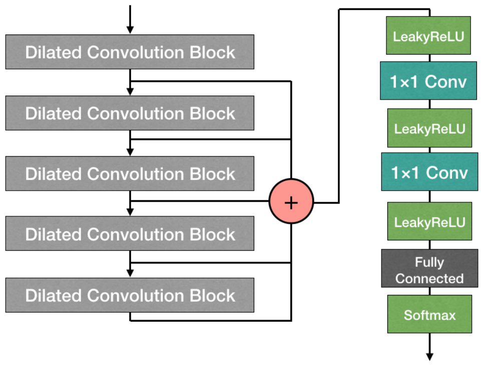
We tackle our task on the basis of DeepBach because accessing both past and future parts of a score mitigates the problem of transition modeling [18], a major obstacle for music modeling. Besides, the temporal feature of music can be well captured with this model (shown in Section 3.4). Although DeepBach succeeds in handling Bachs 4-part chorales, it is not readily suitable for our problem scenario. To adapt the original DeepBach to accommodate more music types, especially for the music having a homophonic texture with varying numbers of polyphony, we remove the restriction of voicing and abandon the original four-network structure, and adopt one single network to process the multi-hot piano-roll representation.

Reducing the number of networks to one causes our generalized DeepBach to lose the ability to model the joint distribution of notes articulated simultaneously at a given time step. Besides, [7] indicated that using the piano-roll representation causes the generated result to be trapped in isolated regions during the Gibbs sampling process (see

[1] https://github.com/s603122001/Music-Style-Transfer



(a) Dilated convolutional block


Dilated Convolution Block
Dilated Convolution Block
Dilated Convolution Block
Dilated Convolution Block
Dilated Convolution Block
LeakyReLU
1×1 Conv
LeakyReLU
1×1 Conv
LeakyReLU
Fully Connected
Softmax

(b) Autoregressive block

Figure 2: Illustration of the autoregressive block and dilated convolutional block of a conditional Wavenet.

Section 2.4). In order to overcome these issues, we employ an autoregressive model, Wavenet [13], to control the relationship among the 88 possible pitch activations at a given time step j. The joint distribution of $S_{:j} = \{S_{1j}....S_{88j}\}$ can be written as the product of conditional probabilities of all pitches:

$$p(S_{:j}) = \prod_{i=1}^{88} P(S_{ij}|S_{1j},...,S_{(i-1)j}, S\backslash S_{:j})\,. \qquad (3)$$

The Wavenet models the conditional probabilities by stacking dilated causal convolution layers [13]. We then employ the output of the generalized DeepBach model as a constraint. This is implemented by a dilated convolutional block with constraint (see Fig. 2a) and an autoregressive block to predict the output by running from $i = 1$ to $i = 88$ (see Fig. 2b). Implementation details of them can be seen in [13].

In summary, the output of the generalized DeepBach represents the temporal context of music, and it constrains the Wavenet model to ensure that the output is musically reasonable in terms of the harmony progression and other contextual structures.

2.3 Implementation details

The model is implemented using the Keras [4] library with tensorflow as the back end. First, input data is divided, such that each unit of the input data contains a segment of four time steps, i.e., a 91×4 matrix, and the segments do not overlap (see the dark green part in Fig. 1). Every segment is first flattened, and the flattened segment is embedded into a 150-D vector with a shared fully connected layer for dimension reduction (see the dark gray part in Fig. 1), so as to incorporate information over a larger temporal range with a smaller model capacity. Similar to the original DeepBach model, two LSTM models are employed, one dealing with the past embedded feature mappings and the other dealing with the future ones. Both LSTM networks take a series of embedded features with 32 time steps, i.e., a 150-by-32 matrix, as the input. Both networks contain 4 LSTM layers, each having 150 hidden units (see the blue block in Fig. 1). The outputs of the two networks are concatenated and then transformed to an 88-D vector with another fully-connected layer. This merged LSTM output is then used as the condition of the Wavenet that employs the original input feature map at the current time step, as illustrated in Fig. 2a. The Wavenet consists of five dilated convolution layers as shown in Fig. 2b, where only the top two of the five layers are conditioned by the merged LSTM output and the other three are not conditioned. A dropout rate of 30% is adopted for each layer, and batch normalization is added after the activation of each convolutional layer. The model is optimized using ADAM [11].

2.4 Style transfer

The algorithm of style transfer is shown in Algorithm 1, and the number of pitch range p is 88 in this paper. Inspired by the idea in [7], the style transfer is conducted by using Gibbs sampling to sample the model and then performing an iterative update on the elements of the input score matrix. In contrast to those models using noise as input [7,16], the input of our model is the music score to be transferred, and this initialization enables the resulting musical structure to follow the original one. In every iteration of the optimization process, all the elements at the same time step in the input matrix (including both note activation and silence) are visited and updated iteratively. It is important to point out that all notes at the same time step are updated together for the chosen target time step. While doing this partly violates the original concept of Gibbs sampling, it produces stable results in the experiments.

Sampling from an autoregressive model is inefficient due to the sequential property that every output is conditioned on all the previous ones. To speed up, we adopt the strategy of independent Gibbs sampling [9, 17]. Independent Gibbs sampling uses an *annealed* masking probability α that controls the percentage of the elements in the matrix that are to be updated independently, making the input approach a stable condition in a short time [9,17]. In the n-th

Algorithm 1 Music Style Transfer

```
Input: $I$ by $J$ score matrix $S$, number of pitch range $P$, maximum number of iteration $N$, maximum and minimum annealed masking probability [$\alpha_{max}$ $\alpha_{min}$], annealed masking ratio $\eta$
 1: $\alpha \leftarrow \alpha_{max}$, $\hat{S} \leftarrow S$, $c \leftarrow 0$
 2: for n from 1 to N do
 3:     Choose time index $j$ in the range of $J$
 4:     if $\alpha > \alpha_{min}$ then
 5:         Update $\{S_{ij}\}_{i=1}^{P}$ by $p(S_{ij}|\hat{S}_{1j}, \cdots, \hat{S}_{(i-1)j}, \hat{S}\backslash\hat{S}_{:j})$
 6:         $c \leftarrow c + P$
 7:         if $c > (\alpha \cdot P \cdot J)$ then
 8:             $c \leftarrow 0$
 9:             $\hat{S} \leftarrow S$
10:         end if
11:         $\alpha \leftarrow \alpha - \frac{\alpha_{max}-\alpha_{min}}{\eta \cdot N}$
12:     else
13:         Update $\{S_{ij}\}_{i=1}^{P}$ by $p(S_{ij}|S_{1j}, \cdots, S_{(i-1)j}, S\backslash S_{:j})$
14:     end if
15: end for
Output: Transferred score matrix $S$
```

iteration of such a sampling process, and for some maximal and minimal probabilities $\alpha_{\min}$ and $\alpha_{\min}$, α is updated by the following formula:

$$\alpha_n = \max\left(\alpha_{\min}, \alpha_{\max} - \frac{n(\alpha_{\max} - \alpha_{\min})}{\eta N}\right), \quad (4)$$

where N and η represent the total number of Gibbs steps and the annealed masking ratio controlling the required time for α approaching $\alpha_{\min}$. As α is reduced to the minimum, the procedure approximates the standard Gibbs sampling, which updates only one element at one time and compensates the poor result produced in the independent phase. The advantage of using independent Gibbs sampling is its efficiency. Besides, independent Gibbs sampling gives more stable outcomes, especially when transferring to challenging genres like Jazz.

This research also discovers that during the transfer process, the melody in the original music tends to vanish during the iteration. As a result, we currently apply a constraint on the melody part to address this issue, and leave the style transfer of the melody part to future work.

3. EXPERIMENTS AND DISCUSSION

3.1 Datasets

Two datasets, Bach's four-part chorales and Jazz music, are employed as the training data. The Bach dataset contains 357 Bach four-part chorales included in the music21 toolkit [5]. The Jazz dataset contains 487 songs either collected manually on-line or generated on our own according to the scores in the well-known Real Book. To simplify the experiment, we did not distinguish among the sub-genres of Jazz, and all of the Jazz pieces are played in Jazz trio, containing one piano, one double bass and one drum. The drummer part is ignored since we consider only the harmonic part of music in this paper. In the training process, we perform data augmentation, by pitch-shifting each song in the two datasets up and down by at most 6 semitones in order to cover all possible keys. As a result, we have 4858 pieces and 6331 pieces in the Bach dataset and the Jazz dataset, respectively. In addition, the two datasets are compiled in different time resolutions. In the Bach dataset, a sixteenth note is defined as one time step, while in the Jazz dataset, a thirty-second note is defined as one time step, as the latter one contains faster note groups.

Four songs with different styles were selected as the testing data: *Rocky Raccoon* by Beatles, *Paranoid Android* by Radiohead, *Live and Let Die* by Paul McCartney, and Beethoven's *Moonlight Sonata*, Op. 27, No. 2. Each of the song was cropped to 30 seconds long. These four testing songs will be used in both the objective evaluation and the subjective test.

3.2 Experiment settings

Experiments are conducted to demonstrate the effect of our solution to the task of transferring the style of an input music piece to Bach or Jazz style, and we employ the proposed models trained from the afore-mentioned datasets of Bachs chorales and Jazz, respectively. To verify the capability of the network in modeling signals with a varying number of voices, we employed two different versions of the network, one without the autoregressive model (denoted as "LSTM only"), and the other incorporating the autoregressive model (denoted as "LSTM-WN"). We compare the following three models:

1. LSTM-to-Bach: transfer the style of input music to Bach's chorale using the LSTM network only.
2. LSTM-WN-to-Bach: transfer the style of input music to Bach's chorale using the LSTM combined with the Wavenet.
3. LSTM-WN-to-Jazz: transfer the style of input music to Jazz using the LSTM combined with the Wavenet.

LSTM-to-Jazz is excluded from the experiment results because our pilot study showed that without the autoregressive model, the generated outputs appear to be composed of random note groups due to the wide diversity of the Jazz dataset. Since a subjective test is hard to be performed with such output samples, we eliminate this case in the following experiments. For further comparison, we also compare our model with the original DeepBach model, under the scenario of reharmonizing given melodies using a pretrained DeepBach model.

3.3 Metrics for objective evaluation

To analyze the performance of a style transfer system and to compare the songs before and after a transferring process, we divide the style transfer task into 3 sub-parts and evaluate them with different metrics.

1. *Content preservation of the original style.* According to our definition of music style transfer, a good music style transfer system should preserve the backbone of an input song. This means that the overall structure of a song, such as chord progression, should not be changed. Therefore, the content similarity between the original song and the transformed one should be considered. To evaluate the content similarity between two music pieces, we first transfer the MIDI representation (in piano roll) of every time step into a chroma vector with the 12 pitch classes, then compute the moving average of the chroma vector within a frame size of half bar, which is 8 time steps in the Bach's dataset and 16 time steps in the Jazz dataset. Finally, the cosine similarity between every time step in the two score matrices is calculated to measure the content preservation degree of the transfer process.

2. *Harmony structure similarity to the transferal target style.* Common harmony sets (i.e., combination of notes) vary across different music styles. For example, chords with an extension note like 9th and 11th are more often used in Jazz music than in Bach's chorales. This characteristic is utilized to visualize how such distribution changes after a style trans-fer. First the transfer target styles of interest in this paper are represented by collecting all existing harmony combinations in the two datasets, and then mapping them to a 2-D plane by using the principal component analysis (PCA) and then the t-distributed stochastic neighbor embedding (t-SNE) method. By visualizing such 2-D features of the original and transferred songs on the plane, we could observe the difference of locations between them, and how the transferred song moves toward the target dataset.

3. *Temporal structure imitating the transferred style.* Rhythmic patterns are an important characteristic in distinguish different music styles. To model this property, we utilize the fact that rhythm has a strong correlation with the timing of harmony changes. For example, it is common that a chord change coincides with a strong beat. Therefore, we define the harmonic transition point of a song to be a time step where at least three notes change in comparison to the previous time step, and then we plot the distribution of these points within every bar with the temporal unit being an eighth note. The result is a 32-D vectors representing the major pattern of rhythm and harmonic transitions. We examine how similar such pattern of the transferred music pieces is to the pattern of a target dataset.

3.4 Objective evaluation

Table 1 shows the performance index of content preservation, the average cosine similarity, of the four testing songs before and after style transfer. We list the results of the proposed models and the original DeepBach [7], being used as a baseline. We do not use the original DeepBach model for style transfer to Jazz because its number of voice is fixed at 4. Notably, the original DeepBach, which uses only the main melody to perform reharmonization, is less effective in following the structure of the original pieces than the proposed models designed for homophonic music.

	To Bach	To Jazz
DeepBach	0.39	N/A
LSTM-to-Bach	0.86	N/A
LSTM-WN-to-Bach	0.76	N/A
LSTM-WN-to-Jazz	N/A	0.56

Table 1: Evaluation results of Content Preservation in terms of cosine similarity.

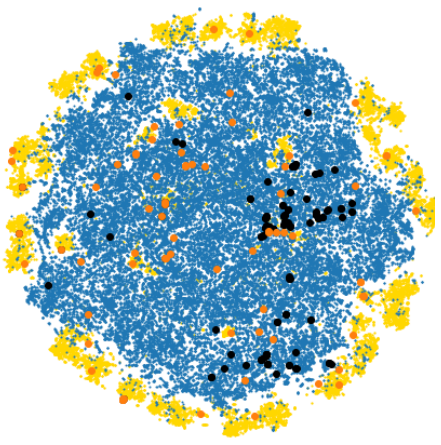


Figure 3: Result of the Harmony Distribution. Data points are the piano roll features mapped to a 2-D space through PCA and tSNE. Blue: Jazz dataset. Yellow: Bach dataset. Black: testing clips of Jazz music. Orange: testing clips transferred to Bach's style.

Fig. 3 illustrates the harmony distribution of the cropped segments of five pieces in the Jazz dataset, before and after a style transfer using the LSTM-WN-to-Bach model. Here we illustrate the result of Jazz data instead of the testing songs because this is a genre-to-genre comparison. As shown in Fig. 3, the resulting harmony distributions indicate that all data points of the Jazz data are originally within the distribution of the Jazz dataset. After style transfer, most of the data points move toward the distribution of the Bach dataset, and some of the transferred data points are even located within the Bach distribution.

Fig. 4 shows the results of temporal structure similarity of the four testing songs before and after style transfer. We used the same songs and models in the content preservation part. Results show that for the transfer-to-Bach case, all models fit fairly well to the pattern representing the Bach dataset, with the original DeepBach model producing a few extra transitions unseen in the Bach dataset. For the transfer-to-Jazz task, the proposed model also fits the pattern of the target dataset well.

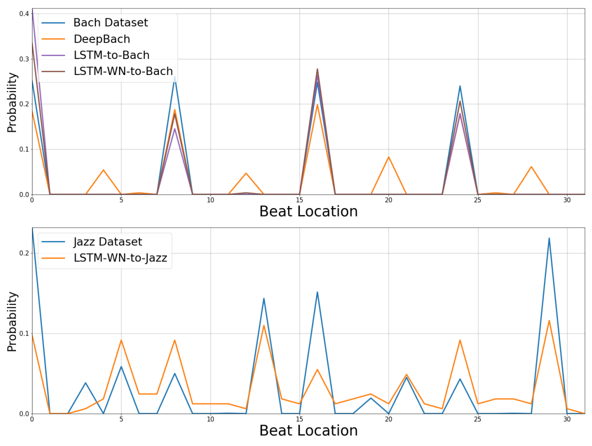


Figure 4: Evaluation results on temporal structures.

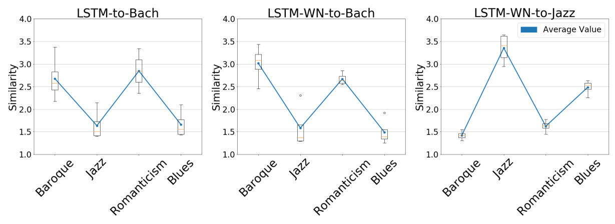


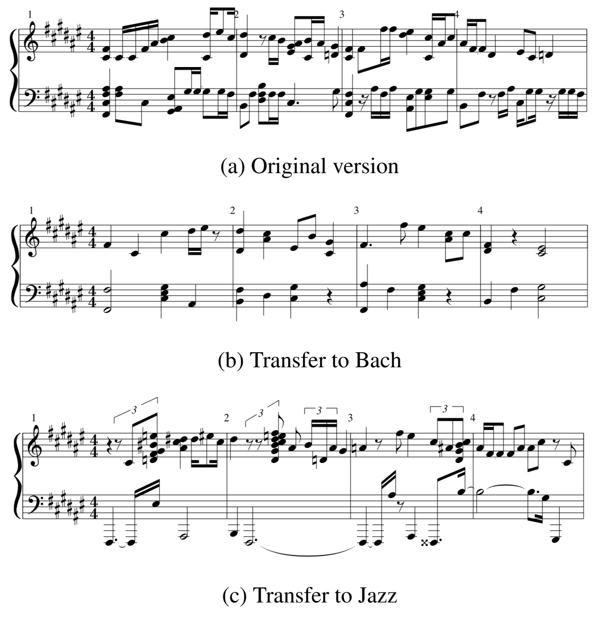
Figure 5: Result of the subjective test. The scores represent the subjects' evaluation on how similar the style of music to the genre listed in the questionnaire.

3.5 Subjective tests

To evaluate the performance of our model from a human perception perspective, a listening test was conducted with 63 participants. Among the participants, 51 of them have the experience of being a music performer, and 17 of those 51 participants receive formal music education or have work experience in related fields.

Each of the four testing songs was transferred using the three afore-mentioned models: LSTM-to-Bach, LSTM-WN-to-Bach, and LSTM-WN-to-Jazz, respectively. As a result, three different versions were produced for each song, and every participant evaluated a total of 12 songs. For each song to be transferred, the participants were asked to determine the style of the main melody from 4 music styles: Baroque music (i.e., Bachs), Jazz, Romanticism, and Blues. This question aims to direct their attention to both melody and accompaniment parts. Note that Romanticism and Blues are extra options added to avoid a possible bias in the questionnaire. After answering the question, the participants then evaluated the degree of similarity between the transferred songs and the 4 music styles above, based on their music knowledge and personal perception. The evaluation was in the scale from 1 (low) to 5 (high).

The results of the subjective test are shown in Fig. 5. The results are averaged for different models. It can be seen that, for the cases of transferring to Bach's style, the rated degrees of similarity of the transferred songs to *Baroque* and *Romanticism* are both high. According to a participant who is a major in music, this phenomenon is related to the main melodies and the original songs we pick. However, the model with Wavenet produced transferred songs rated with the highest similarity degree to *Baroque*, demonstrating the necessity of the Wavenet component in our model. For the case of transferring to Jazz style, the degree of similarity to *Jazz* surpasses other types of music.



(a) Original version

(b) Transfer to Bach

(c) Transfer to Jazz

Figure 6: Transfer results of *Live and Let Die* using the proposed model. The score is output by LogicPro.

One test sample used in the subjective test is outputted as music scores illustrated in Fig. 6 to give us some insights into the capability of the proposed models. In Fig. 6(b), the harmony and music contents are simplified with respect to the original version since the contents of Bach's 4-part chorales are usually not complicated. Apart from this, the difference between the temporal structure of the two scores is a clear example that our model has learned the temporal feature of the music style. In Fig. 6(c), we find many non-chord notes and some taste of syncopated rhythm, both marking the characteristics of Jazz music.

4. CONCLUSION AND FUTURE WORK

We have demonstrated the capability of our model in transferring arbitrary homophonic music scores into the styles of Bach's 4-part chorales and Jazz, and both objective and subjective tests are conducted. The advantage of our method is that it does not pose any restrictions on input music scores, and thus it can be easily applied in other scenarios. Besides, different styles of music can be modeled using the same framework, simplifying the process when we want to expand the collection of target music genres. Future work will focus on the style transfer of a melody, which is not considered in this paper, as well as further investigation into complicated music styles and extensive applications based on the proposed model.

5. ACKNOWLEDGEMENT

We would like to thank the anonymous reviewers for their valuable comments and suggestions to improve the quality of the paper. This work is partially supported by MOST Taiwan, under the contract MOST 106-2218-E-001-003-MY3.

6. REFERENCES

[1] Jukedeck. `https://www.jukedeck.com`. [Online; accessed 26-Nov-2017].

[2] Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent. Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription. In *Proc. International Conference on Machine Learning (ICML)*, 2012.

[3] Jean-Pierre Briot, Gaëtan Hadjeres, and François Pachet. Deep learning techniques for music generation-a survey. *arXiv preprint arXiv:1709.01620*, 2017.

[4] François Chollet et al. Keras. `https://github.com/fchollet/keras`, 2015.

[5] Cuthbert, Michael Scott, and Christopher Ariza. music21: A toolkit for computer-aided musicology and symbolic music data. 2010.

[6] Hao-Wen Dong, Wen-Yi Hsiao, Li-Chia Yang, and Yi-Hsuan Yang. Musegan: Symbolic-domain music generation and accompaniment with multi-track sequential generative adversarial networks. *arXiv preprint arXiv:1709.06298*, 2017.

[7] Gaëtan Hadjeres and François Pachet. Deepbach: a steerable model for bach chorales generation. In *Proc. International Conference on Machine Learning (ICML)*, 2017.

[8] Jay A Hennig, Akash Umakantha, and Ryan C Williamson. A classifying variational autoencoder with application to polyphonic music generation. *arXiv preprint arXiv:1711.07050*, 2017.

[9] Cheng-Zhi Anna Huang, Tim Cooijmans, Adam Roberts, Aaron Courville, and Douglas Eck. Counterpoint by convolution. In *ISMIR*, 2017.

[10] 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.

[11] Kingma, Diederik, and Jimmy Ba. Adam: A method for stochastic optimization. 2014.

[12] Feynman Liang, Mark Gotham, Matthew Johnson, and Jamie Shotton. Automatic stylistic composition of bach chorales with deep lstm. In *ISMIR*, 2017.

[13] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[14] François Pachet. A joyful ode to automatic orchestration. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 8(2):18, 2016.

[15] Dimitrios Tzimeas and Eleni Mangina. Jazz sebastian bach: A ga system for music style modification. In *Systems and Networks Communications, 2006. ICSNC'06. International Conference on*. IEEE, 2006.

[16] Li-Chia Yang, Szu-Yu Chou, and Yi-Hsuan Yang. Midinet: A convolutional generative adversarial network for symbolic-domain music generation. In *ISMIR*, 2017.

[17] Li Yao, Sherjil Ozair, Kyunghyun Cho, and Yoshua Bengio. On the equivalence between deep nade and generative stochastic networks. *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 322–336, 2014.

[18] Adrien Ycart and Emmanouil Benetos. A study on lstm networks for polyphonic music sequence modelling. In *ISMIR*, 2017.

[19] Frank Zalkow, Stephan Brand, and Bejamin Graf. Musical style modification as an optimization problem. In *Proc. Int Conf. Computer Music (ICMC)*, 2016.