Music Style Transfer

Kun Fang and Xihui Wu

1. Introduction:

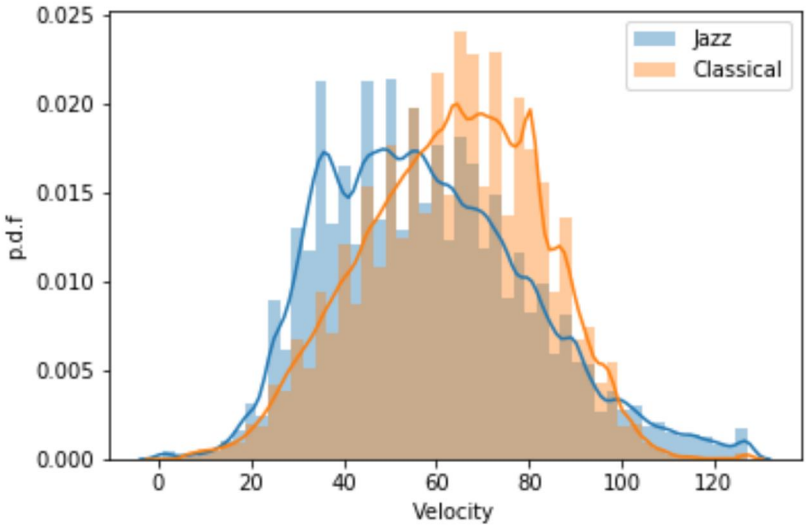
Recent success of neural style transfer on visual art inspires us to experiment it on music. The goal of our project is to build a neural network that can transfer music from one style into another style. This is achieved by training a sequential model. Music style manifests in different ways. In this study, we try to capture the essence of music style in the velocities of the music notes. The input to our algorithm is music notes without the information of velocities. The neural network structure has three layers, including encoder layer, styler layer and output layer. The output of our algorithm is music notes with velocities in a certain music style.

2. Previous work:

We were mostly inspired by the work (Malik, 2017). In their work, they investigated into the using LSTM to encode and transfer music styles. The encoder layer in their study is shared by different genres of music. The encoder layer is then connected to multiple bidirectional LSTM layers in parallel. Each of these bidirectional layers represents music style. We want to explore the possible configurations of the encoder layer and output layers. We were inspired by (Yang, 2017) to use the CNN network to extract features in the encoder layer. We used a python package that were developed by the (Dong, 2018) to process midi files and produce pianoroll plots.

3. Dataset, data structure and data visualization:

We used the same data set from the paper (Malik, 2017). It contains 349 classical tracks and



THE DISTRIBUTION OF VELOCITY

349 jazz tracks. These data are in MIDI format. A python package, pypianoroll (Dong, 2018), is

used to read and preprocess data. Each song it can represented by a [T_x, n_{pitch}] matrix M_T, where T_x is the number of time steps of the song, which differs from song to song, and n_{pitch} is the number of pitches available in the MIDI format, which is fixed at 128. Matrix entries represent note velocities, which denote the how rapidly a key is pressed. Zero velocity means the note is not pressed. The original range of velocity is [0, 127]. For convenience, we rescale velocity to the range of [0,1] as we will use sigmoid outputs to represents the predicted note velocities.

We also tried to visualize the differences between different styles. For example, we found that the velocities that used in Jazz music tend to be lower than the classical music (Fig. 1).

3. The machine learning problem:

We try to capture the essence of music style in the note velocity. The machine learning problem can be phrased as follows. Given a piece of music $M_T^{Original}$, we replace non-zero entries in M_T with ones. This erases the information about the note velocity, leaving only the information that what notes are being activated at a given time step. The new matrix, denoted by X_T, serves as the input to the model. The model will be trained to produce a piece of music in $\hat{M}_T^{style}$ with velocities in the matrix given the input matrix X_T. Note that the intention is to predict the velocities of the activated notes, not to predict what notes will be activated.

4. Methods:

4.1 Model:

The designed network has three layers, an encoder layer, a styler layer and an output layer. The encoder layer is designed to extract features from the music notes. It includes a fully connected network that is shared by different time steps. The setup of this FC layer is to extract features at each time step, for example, the line of melody and chords in a song. At each time step, activated pitches are either part of melodies or chords. The FC layer alone, however, couldn't extract temporal features. Thus a RNN sub-layer is added to the encoder layer. RNN is good at sequential data and is capable of extracting the temporal structures from the data. An alternative structure that replaces the FC layer with a CNN layer is discussed in Sec. 5.2. The encoder layer is then connected to the styler layer, which is also a RNN. The RNN works on the encoded features and try to predict the dynamics of the song. The output FC layer is connected to the styler layer to convert the encoded dynamics into actual velocities. This FC network is shared by different time steps.

4.2 Output and Loss functions:

The velocities of notes in this format are integers ranging from 0 to 127. We use sigmoid functions as the output units and round the outputs to their nearest integers. We think this is more preferable than softmax output and linear output. Softmax output needs more parameters and doesn't impose ordering among outputs whereas in this context there is ordering between note velocities; we can say velocity = 70 is louder than velocity = 40. Since the output should be within a range, linear output is not suitable either; otherwise certain cutoff process is needed following the output units.

It is natural to define a simple loss function at time step t as $L_t = \frac{1}{n_{pitch}} ||M_t - \hat{M}_t||^2$, where M_t and $\hat{M}_t$ are row vectors that represent the the ground true velocities and the predicted

velocities at t and $||\cdot||$ is the L2 norm. And the cost function is defined as $L = \sum L_t$. However, due to the sparseness in the activated pitches, we found that the led to outputs that are very close to zeros. At given a time step, only couple notes will be activated (safe to say less than 10 pitches will be activated since we have only ten fingers after all). So we design a loss function that assigns a smaller weight to the unactivated pitches,

$$L_t = \frac{1}{n_t^{activated}} ||(M_t - \hat{M}_t) * X_t||^2 + \frac{\alpha_{zero}}{n_{pitch} - n_t^{activated}} ||\hat{M}_t * (1 - X_t)||^2,$$

where $n_t^{activated}$ is the number of activated pitches at t, α_{zero} is a constant smaller than 1, in this study we set it at 0.01. By using this loss function, we want the network to focus more on predicting the velocities of the activated pitches.

	GRU + GRU (G2)	CNN + 2 * LSTM (CNN+LSTM2)
Encode Layer	A FC network with three layers with RELU activation (512, 256, 128) and a GRU layer with 256 units.	A Conv1d (kernel size = 2) and a LSTM layer with 128 units
Styler Layer	A GRU layer with 256 units	A LSTM layer with 64 untis
Output Layer	A FC network with three layers (512, 256, 128). The last layer uses sigmoid activation.	A single FC layer

5 Experiment:

5.0 Training a simple neural network:
Just for fun, we trained a 10-layer FC network. The number of variables is comparable to the other networks we report here. The input to the network is the velocities at a given time step and the output is the predicted velocities. The training cost flattens out relatively early around 0.02, which corresponds to RMS error around 18 in velocity.

5.1 Training the G2 model
We used GRU unit as the RNN unit for it is simpler (with fewer parameters) than LSTM cell and its performance is on par with LSTM. We used three layers FC network to connect to the inputs and outputs. This is to guarantee there is enough nonlinearity to transfer from and to the encoded space.
Before determining the current structure, we tested multiple structures for the encoder and output layers, for example, replacing both the encoder and output layers with simple 128-unit FC single layer. We found that the training cost couldn't drop below 0.01. Transferring this result to the physical unit, it corresponds to a RMS error in velocity of 12.8.
We used 'panda mode' to train the G2 model. Adam optimizer was used with standard configuration ($\beta_1 = 0.9, \beta_2 = 0.999$) and the learning rate was first set a 0.001 for the first 200 epochs and 0.0006 the second 200 epochs. The batch size is 128 and $T_x = 128$. The model is capable of reducing the cost at 5×10^{-5}, which corresponds to RMS error in velocity in the order of 1. This indicates that the model is complex enough to learn the music style in velocities.
This model was trained with the Jazz piano songs and we used it to transfer classical songs into the Jazz style. An obvious difference, we observed after the style transferred, is that the velocities tend to be smaller after the applying the music transfer. This agrees with the empirical study in the previous section (Fig. 1) that Jazz songs tends to use smaller velocities.

One subtle difference we found in the transferred song is the following example. In this piece of music we can tell very clearly the melody line and chords. From steps 100 to 420 (Fig. 2), the melody line is the activated pitches between C4 and C6 while the chords are between C2 and C4. The color represents the velocities of the activated pitches with dark color indicating larger velocities. In the original piece, the velocities of both melody line and the chords increase progressively. In the transferred piece, only the velocities of the melody line increase progressively, while the velocities of the chords remain at a relative steady level. Similar pattern can be found in the some training set as well. This is an indication that the network is capable of identifying the melody line and chords and assigning different velocities correspondingly. Also the fact that the velocities of the transferred melody line is increases progressively demonstrates that the network is able to produce output with temporal structures.

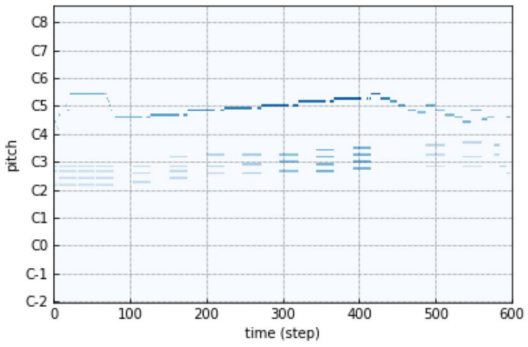


FIG 2.1 ORIGINAL CLASSICAL

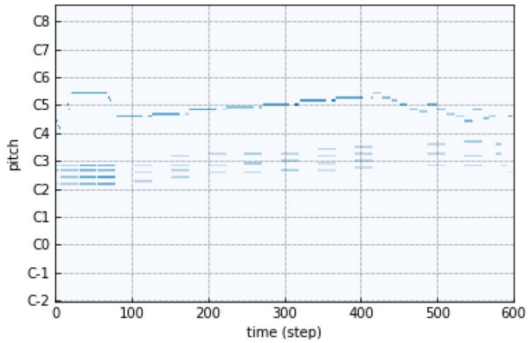


FIG 2.2 JAZZ-TRANSFERED

5.2 CNN-LSTM2:
At a given time step, an activated note can be preceded by another activated note. This kind of note is a sustained note. Alternatively this can be a newly articulated note. We think it's worthwhile to capture this relevance of a note from its preceding note. So we added a CNN sub-layer into the encoder layer. We applied an 1D CONV layer of kernel size 2 at the very first, expecting it to learn the adjacent relevance. Following it is a LSTM which can learn longer contextual relevance. There two together form the encoder layer. A second LSTM acts as style layer learning style specific features. And finally an fully connected output layer with sigmoid activation will scale velocities back to range [0, 128) before final output.

After training on jazz dataset for 1000 epochs we got the model that is able to transfer other music to jazz style. From following link you can listen a Jazz-style music transferred from a sample classical music.

https://s3.us-east-2.amazonaws.com/music-style-transfer/Jazz+-+Chelsea+Bridge+transferred.mp3

And you can compare with the original one:
https://s3.us-east-2.amazonaws.com/music-style-transfer/Classical+-+Chelsea+Bridge.mp3

Conclusion:

In this study we tried to develop a neural network to transfer music into different music styles. We investigated into different structures along the line of encoder-styler-output tri-layer structure. We found that certain level complexity in the encoder layer is necessary to convert music notes into something that can be learned in a RNN layer. The G2 model shows that multiple FC layers and RNN can serve as a good encoder. The CNN + 2 LSTM model shows that a proper CNN make a good difference to make the predicted music more smooth and nature.

For the future work, we would like to study how to build a better encoder. For example, an encoder that can separate the line of melody and chords will reduce the amount of training work and can be shared by different music styles. We also would like to investigate into the possibility to change the note duration according to the music styles. This may involves feeding the output of the t-1 step as an input to the t step.

Contribution:

Both authors worked on data collection, literature research and writing reports. Xihui Wu worked on the CNN-LSTM2 model. Kun Fang worked on the data visualization and the G2 model. Our project code is available in https://github.com/xihw/music-style-transfer.

Bibliography:

Yang, Li-Chia, Szu-Yu Chou, and Yi-Hsuan Yang. "MidiNet: A convolutional generative adversarial network for symbolic-domain music generation." *arXiv preprint arXiv: 1703.10847*(2017).

Dong, Hao-Wen, Wen-Yi Hsiao, and Yi-Hsuan Yang. "Pypianoroll: Open source Python package for handling multitrack pianoroll." *Proc. ISMIR. Late-breaking paper;*(2018).

Malik, Iman, and Carl Henrik Ek. "Neural translation of musical style." *arXiv preprint arXiv: 1708.03535* (2017).