Multi-Intrumental Genre Transfer Using CycleGAN

Justin Young (justiny), Mac Broido (mbroido), Emmanuel Ayumba (eayumba) [1]

Introduction

While genre is perhaps the most common term for classification of musical sound, characterizing genre and timbre from audio waveform is notoriously difficult. This is especially so for modern music with complex combinations of musical sound. While genre is difficult to describe explicitly, it is plausible to cluster similar sounds, suggesting the feasibility of generative audio models trained on data labeled by genre. In recent years we have seen convolutional neural networks revolutionize image and video generation, and we hypothesize that similar technologies will empower advances in audio generation, manipulation, and reconstruction. We produce a generative model for genre transfer between *lo-fi hip hop, trap*, and *R&B* waveforms, using a CycleGAN with Mel-spectrogram representations to output a constructed spectrogram for the target genre. While our immediate application is a creative tool to inspire artists, significant results could foreshadow generational improvements in audio communication technologies.

Related Work

Our method is inspired by recent advances of Huang et. al (2019), who used a constant-Q transform (CQT) waveform representation, a CycleGAN, and a WaveNet synthesizer to transfer a line of music from one instrument to another (e.g. piano to harpsichord).[2] However, instead of simply monophonic instruments, we train a model to differentiate and produce different genres of music. Viere et. al (2019) used a CycleGAN on Mel-spectrograms with *liquid* and *dancefloor* drum and bass songs (which produce high-contrast spectrograms) with alignment of phase information. We use data without phase alignment or labeling, and less specifically-defined sound characteristics. Brunner et. al (2018) used a CycleGAN for symbolic musical transfer (i.e. MIDI data) with notable success. This suggests promising potential for computational musical style transfer, which we independently extend to audio waveforms. We stand on the shoulders of Zhu et. al, whose CycleGAN for various style transfers with unpaired images we use as an inspiration for our implementation.

Dataset

Data Representation

Our dataset to train the CycleGAN is collected from curated YouTube playlists of instrumental music from three distinct genres: 1) *lo-fi*, 2) *trap*, and 3) *old R&B*. Our data is separated into 3 groups (corresponding to each genre) of 800 data points each. Each data point is a 4-second .wav clip that was chopped from the original playlist of the respective genre, which in turn is then transformed into a MEL-scaled spectrogram with resolution 496×369. We obtain the MEL-scaled spectrogram by first applying a short-time Fourier transform (STFT) with a Hamming window $w(n)$ (see Results for figures). Compactly, the STFT is given by

$$X[n,k] = \sum_{m=0}^{L-1} x[m]w[n-m]e^{-i\frac{2\pi k}{n}m}$$

where $L = 2048$ is the window length, and our step size is 1024. We then square the STFT spectrogram's amplitude and convert it to a decibel scale ($20\log_{10}(|S|)$). This representation is chosen as the MEL-frequency scale offers higher resolution to lower-frequencies, which is key in drum and bass heavy music. Each group of 800 points is split into 790 training examples and 10 test examples. Note we do have a total

[1] https://github.com/eayumba/pytorch-CycleGAN-and-pix2pix

[2] We contacted the lead author and found that their code has not been made public because they had some trouble duplicating their results after several key training example audio files were lost. However, we believe their method still works, as many others have tried similar approaches of translating wave-forms to images where ML and deep-learning techniques are more robust.

of more than 1300 data points in each genre, so training or testing on more examples is definitely possible. However, given the repetitive nature of the songs, and given that our 800 data points were randomly pulled from the whole set of data points, we believe this approach is sufficient to show our results.

Each genre was chosen precisely to fulfill a set of criteria that would lead to sensible, interpretable results. First, every genre needs to be objectively distinguishable from another. This entails having genres be sufficiently different with discernible characteristics. For instance, while *trap* and *drill* are different genres, they may be so similar that most differences can be explained by idiosyncratic tastes. The same could be argued comparing *modern R&B* and *lo-fi*. Second, these three genres were chosen because we know certain well-defined elements are present in each – elements that should be seen in our CycleGAN output. For instance, knowing that *lo-fi* music is characterized by high-pass filtered drums gives us some insight into what we should expect to see in our output. In addition, stuttering hi-hats which define *trap* music will never be seen in *lo-fi*. In contrast, *old R&B* is comprised of real instrumentation and older analog recording quality. The feel is distinctly unique, and from an analytical view, the frequency spectrum should be smaller than that of modern music.

The original CycleGAN included data augmentation in the form of random cropping and jittering. We decided the former because each spectrogram contains important information in frequency along the y-dimension, and all of our examples are time aligned on the x-dimension (4 seconds). We similarly did not implement the latter because the 4-second chopping effectively is random jittering, given that no song other than 60bpm will lead to chops at regular intervals.

Data Considerations

For quality control, we note that none of these songs have "producer tags" so no unnecessary artifacts are introduced. In addition, because words are often included in *lo-fi* songs, we chose a playlist where this very infrequently occurred. To our knowledge, we have removed 4-second clips where words occur. To further improve our data, we removed 4-second clips where the the audio cuts and transitions to the next song.

We chose not to standardize the song sections in our data (e.g. pulling 4-second snippets only from choruses). Veire (2019) chooses to align both the tempo as well as the section, but we believe the nature of our genres makes this unnecessary. In particular, *lo-fi* and *trap* are highly repetitive, and choruses are often not-well defined. However, we acknowledge that the beginning and ends of songs are often different from the middle sections, but explicitly accounting for this may not provide much benefit, given that beginning sections do often repeat in *lo-fi* music. Whether this would improve our training significantly will be evaluated in the future, but for our project, we thought it would be more important to collect more data points with a small error than to have less data with less error.

Methods

We focused primarily on the translation between *lo-fi* and *trap*. As a first pass, we trained a logistic classifier to differentiate between the two genres. If the logistic classifier could not distinguish between two spectrograms of different genres, then it is unlikely that our trained GAN would produce any sensible genre transfer. Afterwards, we trained the CycleGAN. As did not discuss GANs in class, we first talk briefly about the GAN architecture and CycleGAN as an extension.

Logistic Classifier

As part of our initial research, we trained a logistic regression classifier to determine if it would be feasible to identify differences between spectrograms of clips from different genres. To do this, we trained on 1200 256x256 4-second spectrograms, with 600 each from our *lo-fi* and *trap* datasets (order randomly shuffled), denoted 0 and 1 respectively. We used the remaining 200 spectrograms from each genre as our test set. We trained the model using a Limited-memory Broyden–Fletcher–Goldfarb–Shanno solver with optimal learning rate schedule and convergence condition $\text{loss} > \text{loss}_{\text{best}} - .001$ for 5 consecutive iterations.

GAN & CycleGAN

In the original GAN introduced by Goodfellow et al. (2014), two neural networks, a generator G and a discriminator D, play a min-max game against each other, updating their respective parameters θ_G and θ_D until each player's loss is at a local minimum with respect to their own parameters. The generator pulls noise z from some prior latent distribution and generates $G(z)$, a "fake" that the discriminator attempts to tell apart from $x \in X$, a real example. The discriminator is trained like a logistic classifier. The loss function is (almost) the cross-entropy for classifying real vs. fake:

$$J_D(\theta_D, \theta_G) = \mathbb{E}_{x \sim p_{data}}[-\log D(x)] + \mathbb{E}_{z \sim p_z}[-\log(1 - D(G(z))]$$

The CycleGAN by Zhu et al. (2017b) is an extension of the GAN. We now have a pair of GANs arranged cyclically, between a source X and target Y. Two generators are now present, $G : X \to Y$, and $F : Y \to X$, with two adversarial discriminators D_X and D_Y. Instead of images being generated by random noise, now the discriminators try to distinguish between real examples and translated examples. For instance, D_X tries to distinguish between images $\{x\}$ and $\{F(y)\}$. Here, X represents the set of *lo-fi* spectrograms, while Y represents the set of *trap* (or *R&B*) spectrograms. There are now two GAN loss functions, one for each discriminator: $J_{GAN}(G, D_Y, X, Y) = \mathbb{E}_{y \sim p_{data}(y)}[\log D_Y(y)] + \mathbb{E}_{x \sim p_{data}(x)}[\log(1 - D_Y(G(x)))]$, and $J_{GAN}(G, D_X, X, Y) = \mathbb{E}_{x \sim p_{data}(x)}[\log D_X(x)] + \mathbb{E}_{y \sim p_{data}(y)}[\log(1 - D_X(F(y)))]$. In addition, by the cyclic nature, we also have a cycle-consistency or reconstruction loss, which in short penalizes the L1 norm of how different x is from $F(G(x))$ (forward loss) and how different y is from $G(F(y))$ (backward loss): $J_{cyc}(G, F) = \mathbb{E}_{x \sim p_{data}(x)}[||F(G(x)) - x||_1] + \mathbb{E}_{y \sim p_{data}}[||G(F(y)) - y||_1]$. Weighting the reconstruction cost by λ, the total loss can be expressed as

$$J(G, F, D_X, D_Y) = J_{GAN}(G, D_Y, X, Y) + J_{GAN}(F, D_X, Y, X) + \lambda L_{cyc}(G, F)$$

In addition, we also considered an identity loss J_{Id} to regularize the generator. Essentially, when fed a $y \in Y$, $G(y)$ should equal y, i.e. $G(\,\cdot\,)$ should be the identity. The same applies to x and $F(x)$.

Experiments & Results

We first discuss the logistic classifier. Then, for the CycleGAN, we focus on the mapping from *lo-fi* to *trap* and provide select results for brevity, but our GitHub has a link to all tests performed. In whole, we have 10 from *lo-fi* to *trap*, 10 from *trap* to *lo-fi*, 10 from *lo-fi* to *R&B*, 10 from *R&B* to *lo-fi*. Furthermore, we re-ran everything without the identity loss J_{Id} included, which led to overall poorer results.

Logistic Classifier

Our logistic regression classifier was 93.68% accurate on the test set, which inspired confidence in the feasibility of identification of musical differences in genre with machine learning techniques and spectrogram representations. Indeed, upon inspection of misclassified test examples (fig. 1.1), generally we either agreed that the spectrogram could have represented music from the other genre, or exhibited structural abnormalities such as the fourth spectrogram in the first row, which was produced by audio at the beginning of a track. With this result in hand, we began researching generative audio models.

CycleGAN

Implementation Specifics

We largely follow Zhu et al. (2017b) in implementation. Our discriminator is a 70×70 PatchGAN which classifies whether overlapping 70×70 pixel images are real or fake. This speeds up learning as such a discriminator has fewer parameters than a full-image discriminator. Our generator is a convolutional ResNet with 9 residual blocks adapted from He et al. (2015).

GANs are highly notorious for being unstable. One stabilizing modification comes from replacing J_{GAN} with a least-squares loss. Furthermore, as in Zhu et al. (2017b), the discriminator updates using a history of 50

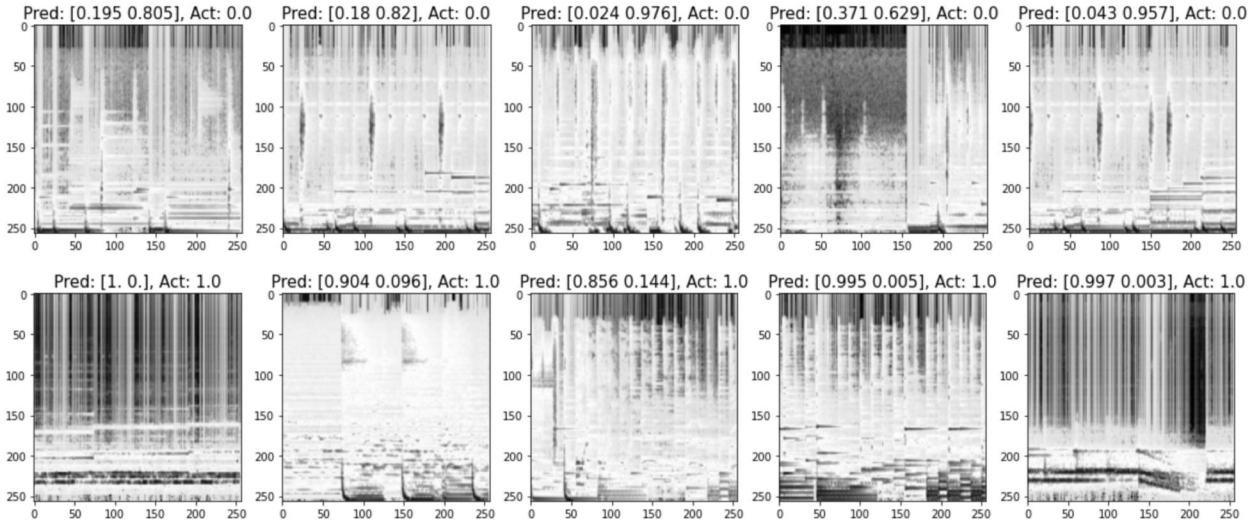

Figure 1.1: Misclassified Test Examples

images, rather than the one image produced last by the generator. To keep the discriminator from being too powerful, we divided the objective by 2 to slow down the learning speed of D. We trained our CycleGAN first for 150 epochs, and then later dialed it back to 50 epochs, as our spectrograms seemed to come out to decent results without the additional run-time. As with Zhu et al., we used an Adam solver with batch size 1 for our SGD, and a learning rate of 0.0002 which starts to linearly decreases to 0 after completion of half the epochs have been completed. Lastly, we weighted $\lambda = 10$ for the cycle-consistency loss. We ran the CycleGAN on the GPU with High-RAM on Google Colab Pro.

Metrics

Qualitatively, our metric for successful CycleGAN training will be how the images look to us given what we know. Stuttering hi-hats are characteristic to trap, and these are visible as ticks of dark red in each spectrogram. In addition, we would expect translation from X to Y or Y to X to preserve the drum structure. While no tempo was synced, each genre hovers around the same 60-80 beats per minute (*trap* can be thought of as simply double-time). Because each 4-second spectrogram in either genre has a well-defined tempo based on the Euclidean x-coordinate distance between hi-hats, snares, or kick drums (represented as a narrow flash of darker red), we expect the CycleGAN to preserve this. One last qualitative metric would be listening to the spectrogram if it were to be further translated into a .wav file via a phase-inferring process.

Quantitatively, our metric for successful CycleGAN training will seen in the loss of each discriminator, generator, cycle, and identity (if present) loss. From Zhu et al. (2017b), GAN training often does not lead to convergence, unless it is a WGAN. The losses are difficult to interpret as G and D are playing a minimax game that changes with each iteration. As long as losses do not explode and decrease with training, the CycleGAN is tentatively successful.

Results

In figures 1.2-1.8, we analyze MEL-spectrograms with a frequency y-axis ranging from 20-20000 Hz, and a time x-axis ranging from 0-4 seconds. We are only analyzing genres *lo-fi* and *trap*, but full results are can be found via the README on our GitHub.

We first begin with qualitative observations. Figure 1.2 below represents a real *lo-fi* spectrogram x, which is transformed into a fake *trap* spectrogram $G(x)$ (figure 1.3). We note that the red streaks in the top half of $G(x)$ are indicative of a potential hi-hat that has now been introduced by generator G. In the same vein, we see in figures 1.4 and 1.5 that a *trap* spectrogram y loses its hi-hat stutter after being translated by F

into the *lo -fi* domain. In addition, it appears that the drum pattern is largely untouched, which suggests that at least some structural integrity of the tempo is preserved by the CycleGAN.

On the quantitative side, we note that the losses on average do decrease with each epoch. We have included the link to the full loss results in the Google Drive link on GitHub. It appears that all losses start out relatively big and decrease as training goes on. By the metrics listed above, this is indicative that our CycleGAN has tentatively succeeded.

We note briefly here that when we ran the CycleGAN without the identity loss, the results were noticeably worse, as seen comparing figure 1.6 and figure 1.8. It thus appears that the identity loss is important as regularization as it prevents G and F from overfitting. This is further echoed by the full loss result numbers mentioned above. Looking at the loss log, it seems as if all aspects of loss are significant, and every single one plays a substantial role in training the CycleGAN.

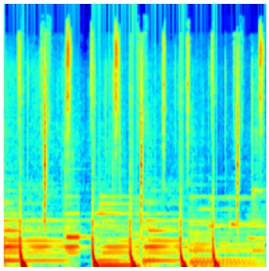

Figure 1.2: Real x

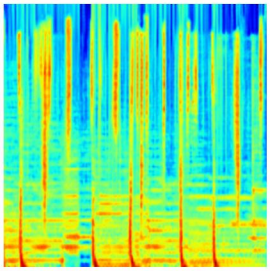

Figure 1.3: Fake $G(x)$

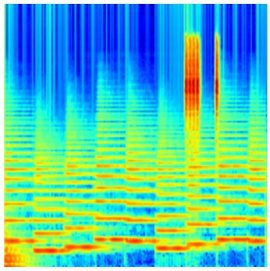

Figure 1.4: Real y

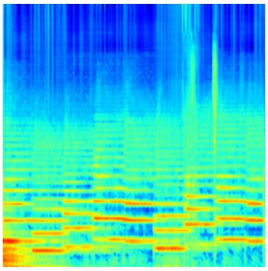

Figure 1.5: Fake $F(y)$

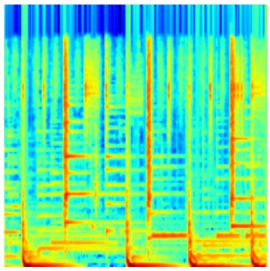

Figure 1.6: Original y

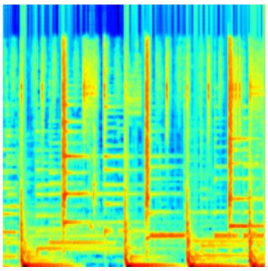

Figure 1.7: $G(F(y))$ with J_{Id}

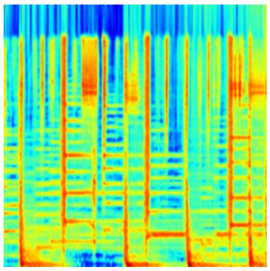

Figure 1.8: $G(F(y))$ without J_{Id}

Future Work

In our implementation, we have attempted to transfer genres first between *lo-fi* and *trap*, and second between *lo-fi* and *R&B*. We first used a MEL-spectrogram representation of audio forms and then trained a CycleGAN to translate between genres with moderate success.

With more time and resources for this project, we would naturally next implement and train a Wavenet Synthesizer to produce audio outputs from the model result spectrograms, in accordance with Huang et al. (2018). This would allow for a qualitative assessment of the genre transfer viability of our model. We spent some time looking for existing implementations of such a network and found successful implementations using arrays of MEL-spectrograms. However, we ran into issues when translating our spectrograms into array representations. Future work would first involve further exploring the implementation of such a MEL-Spectrogram-to-array representation as additional post-processing. Along with a Wavenet Vocoder, we could then use the ultimate qualitative metric: listening to the CycleGAN outputs and analyzing the quality of genre transfer.

Contributions

All group projects contributed equally. Mac and Emmanuel worked on implementation while Justin worked on understanding the theory of the GAN/CycleGAN. We all taught each other what we knew and debugged together the CycleGAN code together.

References

[1] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

[2] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, 2017.

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[4] Len Vande Veire, T. D. Bie, and J. Dambre. A cyclegan for style transfer between drum & bass subgenres. 2019.

[5] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks, 2018.

[6] Roger Wattenhofer Gino Brunner, Yuyi Wang and Sumu Zhao. Symbolic music genre transfer with cyclegan. 2018.

[7] Sicong Huang, Qiyang Li, Cem Anil, Xuchan Bao, Sageev Oore, and Roger B. Grosse. Timbretron: A wavenet(cyclegan(cqt(audio))) pipeline for musical timbre transfer. *CoRR*, abs/1811.09620, 2018.

[8] Zafar Rafii. Audio functions in python for audio signal analysis, 2021.