Evaluation of Vocal Audio Style Transfer

Ben Heller
bheller2@stanford.edu

Abraham Ryzhik
aryzhik@stanford.edu

Zarah Tesfai
zaraht@stanford.edu

*https://github.com/zarahtesfai/cs229projectcod*e

Abstract

We are focusing on the problem of musical style transfer, more specifically changing a song so that it sounds as if it were sung by another artist. It is relatively straightforward for humans to distinguish between different artists' singing styles, and many people also enjoy creating covers of songs in their own personal style. This project focuses on evaluating singing style transfer algorithms to generate song covers in the style of a specific artist. Our main tool to do this is using a classifier that classifies existing songs by artist, particularly classifying the singing component by the singer. This classifier could then be applied to the output of singing style transfer algorithms to judge their efficacy.

1 Introduction

We are focusing on the problem of musical style transfer, more specifically changing a song so that it sounds as if it were sung by another artist. It is relatively straightforward for humans to distinguish between different artists' singing styles, and many people also enjoy creating covers of songs in their own personal style. This project focuses on musical style transfer and generating song covers in the style of a specific artist, and also classifies songs based on artist. While groups like Facebook (1) have been successful in changing the genre of a song, or switching genders of vocals (2), we aim to recreate the sound of a specific singer, and thus expanding an artist's discography to any song we want.

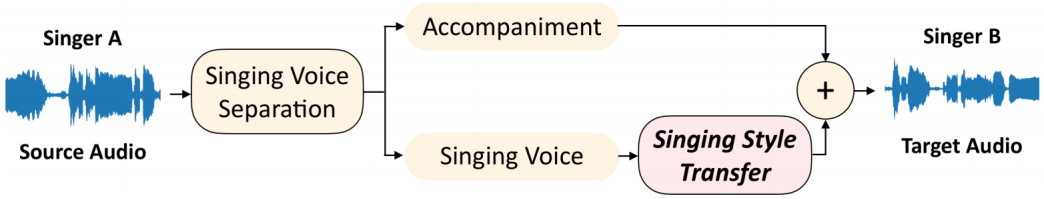
2 Related Work

Much of the inspiration for the methods we used to do audio style transfer came from methods developed for image style transfer. This is a field where style transfer has already been developed very thoroughly and neural networks have been shown to be very successful.

For artist classification using traditional machine learning techniques, we drew inspiration from papers we had seen on music genre classification and applied a similar approach to classify artists.

3 Dataset and Features

For the neural network, we obtained vocal stems (acapella versions of songs that contain only the singing, and no instrumental parts) from *voclr.it*. We used 13 samples each of the artists *Beyonce*and *The Weeknd*. These vocal stems were at $34,100Hz$ and were in $.wav$ files. We used a window of 5 seconds in the songs in order to restrict the number of features.



To preprocess our data, we constructed spectrograms, or *time vs frequency* graphs, which we then took the image of (not the raw data). The spectrograms gave a much more useful form of the audio for us to run the various models on. An example of the spectrogram can be found below in the *Experiments, Results, and Discussion* section.

4 Methods

There are two components to our project: classification and generation. Are goal is to generate music in the style of artist A from the content of artist B so that it will be classified as the music from artist A.

For our classifier, we began by using SVM to see how well one of the traditional machine learning classifiers would perform on the dataset. Since the spectrograms are very high-dimensional, it would not be ideal to pump these directly into the SVM. We used PCA to reduce the dimension of our data to 16 so that training the SVM to classify the data would be faster. PCA maximizes the value of

$$\sum_{i=1}^{n}(u^T x^{(i)})^2$$

over unit vectors u to find the directions such that the variance of the $x^{(i)}$ is maximized after the $x^{(i)}$ are projected onto those directions.

For the generator, we looked at three different models. The inspiration between each of these models was to translate successful methods from image style transfer to the problem of audio style transfer, where the spectrograms take the place of the images. The first method was a simple 2D Convolutional Neural Network along with Gram matrices to compute the differences between the filter responses. The second method was another 2D Convolutional Neural Network but where the Gram matrices were transformed to be calculated over the time axis. The Gram matrices represent the inner product between the feature maps of a layer. For a layer l and features i and j we define it concretely as:

$$G_{ij}^{\ell} = \sum_{k} F_{ik}^{\ell} F_{jk}^{\ell}$$

The goal is to minimize the mean-squared distance of the gram matrices from the style image and the generated image. The loss for each layer is:

$$\mathcal{L}_{\ell} = \frac{1}{4N_{\ell}^2 M_{\ell}^2} \sum_{i,j} (A_{ij}^{\ell} - G_{ij}^{\ell})^2$$

where N_ℓ is the number of feature maps and M_ℓ is the size of each feature map for that layer. Then the total content loss is given by:

$$\mathcal{L}_{style} = \sum_{\ell} w_{\ll} \mathcal{L}_{\ell}$$

The content loss is given by the the differnce of the feature maps for each layer (F and P are the feature maps for the content and generated image):

$$\mathcal{L}_{content,\ell} = \frac{1}{2} \sum_{i,j} (F_{ij}^{\ell} - P_{ij}^{\ell})^2$$

The total loss is then given by $\alpha \mathcal{L}_{content} + \beta \mathcal{L}_{style}$ where α and β are hyperparameters determining the contribution of the style and content to the total loss.

The last method was to use a CycleBEGAN, a certain type of Generative Adversarial Network. In addition to the usual loss function

$$L_{GAN}(G, D_Y, X, Y) + L_{GAN}(F, D_X, Y, X)$$

for GANs, this model ensures cycle-consistency by adding

$$\lambda(E[F(G(x)) - x] + E[G(F(y)) - y])$$

to the loss (7). The motivation behind this is that if we first convert from style A to style B, then convert back from style B to style A, we would very much like the doubly converted audio to resemble the original audio. Cycle consistency helps restrict the space of possible style transfer algorithms to those that make sense.

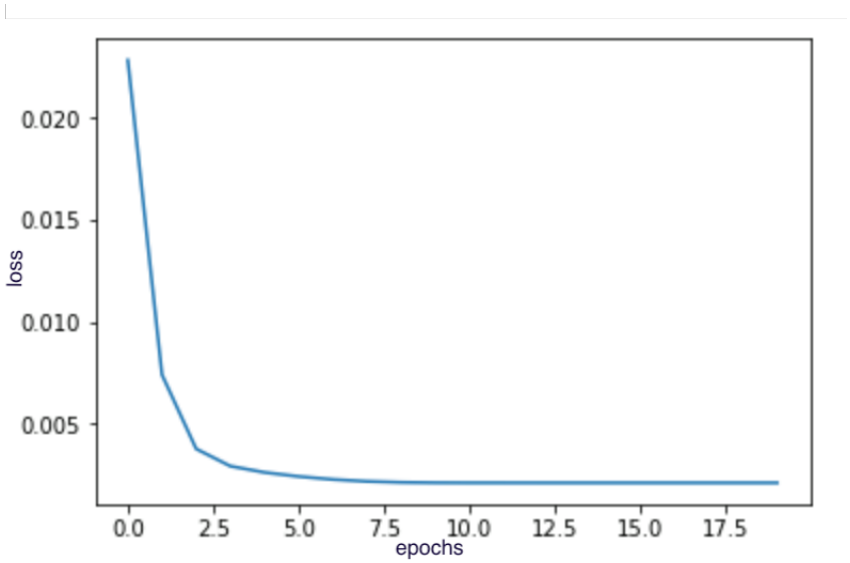
5 Experiments, Results, and Discussion

For the neural networks, the experiments we ran were to actually run the networks and see what the quality of the output was. For both 2D CNN models, the input was one style song and one content song, similar to what we often do for image style transfer.

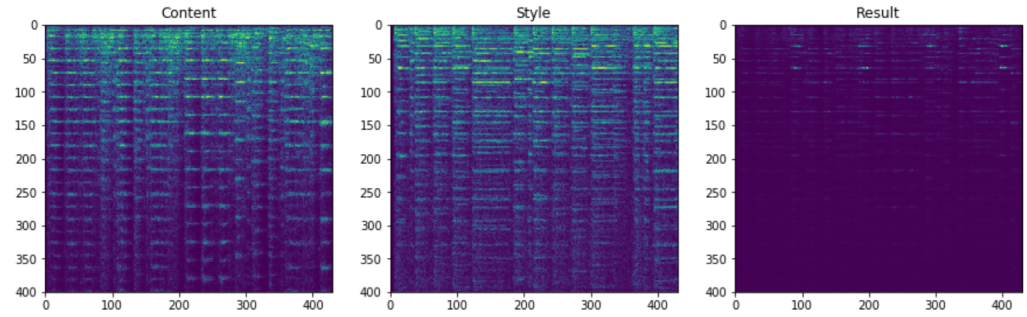
We ran the first CNN first on an instrumental to serve as a baseline. This is because the image transfer of non-vocal style is easier since the result does not have to intelligible. For this baseline test, this model performs decently. In a transfer from content B to style A, the most notable differences were the change of instruments to sound more like those from A. On vocal transfer, however, the model does not perform very well. The result sounded just as if we had mixed the two audio sources together, with the amount of each source included depending on the hyperparameters of the ratio of style loss to content loss that was included in the total loss. Training for this model was very fast, it only took about 5 minutes on a regular computer. After 300 iterations, we obtained a final loss value of 568.6033.

We ran the second CNN on the same vocal samples as the first case. In our initial tests, the results were much better, but still not fully satisfactory. In this case it was clear that the lyrics and melody purely came from the content source, but the influence of the style source was not very significant. There was a change in how fast the words were pronounced to match the style source, but the voice still sounded like the content source in the first round of testing. Since this model was based on a similar methodology as the previous one but produced much more encouraging results, we chose to do more testing on this one alone. This model took around 90 minutes to run a single GPU.

Here is a plot of the training for this method, showing how the loss converged over the epochs of training:



This is what the spectrograms look like, which is the data that we ran our models on. We have included here both the input spectrograms and the output spectrogram for the audio transfer:



For the CycleBEGAN method, one big difference hear was that we trained on a group of songs for the style and content instead of just one song for each. In the same initial experiments, this model immediately showed a much bigger change of the content audio to match the vocal style of the style audio. Changes were present throughout the entire recording rather in just several specific moments. This model took much longer to run than the previous models (the preprocessing step alone took as long as running the second model in its entirety). As a result, it was not feasible to test many songs with this method for this project.

Although our initial tests were not fully satisfactory, we decided to try more tests with the second model because it had showed some promise and each iteration did not take very long to run. We found that while this model was not very consistent, it was possible to get pretty good results. The best vocal style transfer we achieved was with the content audio being ”Can’t Feel My Face” by The Weeknd and the style audio being ”Pretty Hurts” by Beyonce. For this example, the audio clearly sounded like it was being sung by the style artist while the lyrics and melody clearly came from the content artist. We believe that the large difference in style because one artist is male while the other is female helped make the difference more pronounced and make the output audio recognizable as coming from the style artist. However, even with this example, the audio sounded very computer-generated and the sound quality was not the best.

With our SVM model, we attained 68% accuracy on training data and 56% accuracy on testing data. These results are very disappointing given that flipping a coin gives 50% accuracy, so applying the SVM model to the output of existing singing style transfer algorithms is not currently very enlightening.

6 Conclusion and Future Work

Of course, we want to improve the performance of our singer classifier to more accurately determine the singer of existing songs. However, the main goal in the future is to develop better models to perform singing style transfer, since all existing models are either very specific or largely ineffective. Judging the efficacy of style transfer algorithms can help guide the development of these algorithms in the future.

One immediate first step would be to run the CycleBEGAN model on a large collection of songs (this would probably take several days) and see what is the upper bound on possible quality of this model. This model showed the most promise in our initial tests, but do to the long runtime we were not able to test it as much as we wanted.

For this class we chose a classifier based on an SVM because it fit the curriculum of this class better, but we think that developing a CNN would make a classifier that we could rely on much better.

Lastly, if we want to make this task complete, the output audio needs to be cleaned up even in a successful audio transfer. There are two possibilities here to explore: improve the GAN neural network so that the output is cleaner, or develop a separate neural network to handle this problem alone by taking in the output audio from the style transfer neural network and removing sounds that we would view as artifacts of computer generation.

7 Contributions

Our team discussed various topics before settling on musical style transfer, which we were all excited about, and we all researched widely around the topic and shared ideas on possible methods, before focusing on more specific tasks. We worked together to write the project proposal and milestone.
Abraham researched different neural network methods to be used as baselines for testing, then implemented them for our preliminary experiments. Zarah researched about using standard machine learning methods to classify audio, and also about analyzing Fourier transforms and signals using ML to distinguish between musical characteristics. Ben researched examples of other work in the field of musical style transfer and also worked on data collection by finding vocal stems for us to train and test our models on.
During most of our testing process, we met up and collaborated together so we could constantly update each other on the progress. While Zarah set up a poster and final paper template, we all contributed to the content. Ben worked hard on the SVM and CycleBEGAN code, Abraham focused on the first two CNN models, and Zarah assisted with both and analysed the results.

References

[1] Noam Mor, Lior Wolf, Adam Polyak, Yaniv Taigman (Facebook AI Research). *A Universal Music Translation Network.*
https://arxiv.org/pdf/1805.07848.pdf

[2] Cheng-Wei Wu, Jen-Yu Liu, Yi-Hsuan Yang, Jyh-Shing R. Jang.
Singing Style Transfer Using Cycle-Consistent Boundary Equilibrium Generative Adversarial Networks.
https://arxiv.org/abs/1807.02254

[3] Dmitry Ulyanov and Vadim Lebedev. *Audio texture synthesis and style transfer.*
https://dmitryulyanov.github.io/audio-texture-synthesis-and-style-transfer/

[4] Toya Akira. *Voice style transfer with random CNN.*
https://github.com/mazzzystar/randomCNN-voice-transfer

[5] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. *A Neural Algorithm of Artistic Style.*
https://arxiv.org/pdf/1508.06576.pdf

[6] Jeremy F. Alm and James S. Walker. *Time-Frequency Analysis of Musical Instruments.*
https://epubs.siam.org/doi/pdf/10.1137/S00361445003822

[7] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. *Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks.*
https://arxiv.org/abs/1703.10593