Indian Institute of Technology Bombay
Department of Computer Science and Engineering
CS753 : Automatic Speech Recognition

Audio Style Transfer
Project Report

Sanchit Jain & Sudhanshu Raj
160050043 160050041

Supervisor:
Prof. Preethi Jyothi

Date: $22^{nd} November 2019$

Table of Contents

1 Introduction

"Style Transfer" is the process of combining the content of one entity and the style of another to create something new. We as humans always try to create new things, don't we?
Style transfer among images has emerged as a very active research topic, fuelled by the power of convolution neural networks (CNNs), and has become fast a very popular technology in social media. A Mobile application called prisma is an example of how much it has been accepted by the community as a whole.
In this project we try and apply the same concept of style transfer in Audio signals where we take a **reference audio signal**(aud_ref) and **target audio content** (aud_cont) as input and gives an output as a **styled audio signal** (aud_style) which has the content of aud_cont and the style of aud_ref.
By **content** we mean the actual words or phones which are spoken i.e. for example if a person says "the quick brown fox" then the content is the text "the quick brown fox".
By **style** we mean the sound texture model which extracts the statistics characterization of aud_ref. Some examples of what we can define as these characterizations is speakers identity, intonation,accent or emotion. In case of music it can be score played or rhythm or style such as jazz.

2 Prior Work

There have been great progress and success stories in the field of style transfer both computer vision(images) and speech(audio). The success is mainly catered to and more understood in the field of computer vision and we have a plethora of papers written in this field and we will be focussing on four of the ones related to our project.

- They[1] propose a flexible framework for the task, which uses a sound texture model to extract statistics characterizing the reference audio style, followed by an optimization-based audio texture synthesis to modify the target content. In contrast to mainstream optimization-based visual transfer method, the proposed process is initialized by the target content instead of random noise and the optimized loss is only about texture, not structure. These differences proved key for audio style transfer in their experiments. In order to extract features of interest, they investigate different architectures, whether pretrained on other tasks, as done in image style transfer, or engineered based on the human auditory system. The architectures include VGG, SoundNet, Shallow and McDermott out of which the latter two gives satisfactory. We borrowed the framework which they are using and the Shallow method of extraction of features where we use a wide shallow network as explained in the next section.Our approach is different from theirs in the sense that we actually use a content loss instead of just initializing it with the content. Otherwise we were having trouble in preserving the content as well.

- This [2] was the first paper which introduced style transfer and the underlying concept behind it is beautifully explained in this paper. No doubt this paper was a pioneering one and we cannot do away without mentioning it. They used the power of convolutional neural networks. We borrow the Gram matrix approach to get the style loss from this paper.

- This [3] paper is an extension to [2] where they use perceptual loss function for content and style reconstruction as oppose to per-pixel loss. Perceptual loss functions based on high-level features extracted from pretrained networks[4]. They show that their approach is three times faster than [2] and give comparative results.From this paper we adopted the perceptual loss which is combined with the framework mentioned in [1].

- We have used the implementation at [5] as our baseline code.

3 Implementation Details

- First step was to extract the spectral properties from the input content and style audio. We have used three techniques to extract the features from the audio signal. We used 'librosa' library for extracting the features.
  - **STFT**: It is simply the short time fourier transform of the input audio signals. This returns the fourier transform of the audio signal containing 1025 channels
  - **MEL**: First we take the fourier transform of the input then extract the MEL spectogram of the input audio signals. The output contains 128 channels.
  - **MFCC**: In this we take the fourier transform of the input then extract the MFC coefficients of the input audio signals. The output contains 40 channels.
- The next step was to design the Neural Network. We tried various types of hidden layers as well as different types of activation functions to analyze the performance.
  - We used wide shallow network for our final implementation. We tried various types of hidden layers like AvgPool layer and MaxPool layer but the results detoriated with introducing new such layers. So, we finally stuck with a Conv2D layer.
  - We used two types of activation function: RELU and Tanh. In our final network we stuck with Tanh function as it gave better results.
- Then, there was objective function for the Neural network. Our aim was to minimize the loss function which we calculated in two different ways.
  First way was a sum of simple L2 norms between:
  - The content spectogram and the final output spectogram(which was initialized randomaly in the beginning)
  - The gram matrix of the style spectogram and the output spectogram. Gram matrix **G** of a matrix **A** is:

$$G = A^T A$$

  Secondly, we also tried the perceptual loss method.
  Perceptual loss functions are used when comparing two different images that look similar, like the same photo but shifted by one pixel. The function is used to compare high level differences, like content and style discrepancies, between images. We used it for audio signals here.
  In this loss method we:
  - Pass the target output matrix and the content matrix through an another Neural Network (which was pretrained) to calculate the perceptual loss between those matrices.
  - This network basically calculate difference betweem these two matrices depending on high-level features extracted from a convolutional network.
  - We used a pretrained network [4] for calculating the pereptual loss.
  - Then we calculated L2 norm of the gram matrices same as above and our goal was to minimize the sum of this perceptual loss and L2 loss.
- The last step was to restore the audio signal from the calculated target spectogram matrix. So, based on the initial feature extraction method, corresponding inverse function was used. For Mel and MFCC we used two methods: Using GriffinLim phase reconstruction method and without this method. Former one gave better result so, in our final implementation we stuck with the GriffinLim method.

4 Experiment and Result

First we setup our basecode on the Jupyter Notebook but due to limited computation power we used Google Colab for rest of our project.

- So, as stated above we used three different types of feature extraction method and we got the best result from the Mel spectogram(result here) then stft(result here) and then MFCC(result here). In MFCC, there were only 40 channels and hence we may have lost a lot of relevant information of the content in MFCC and that's why it may have given poor results.
- As for hidden layers in the Neural Network, as stated we tried MaxPool and AvgPool layer (result here)but the results were better when no pool layers were used(result here). Maybe this was due to loss of spatial information introduced due to pooling layer. As we increased the width of the pooling, the representation became more key-invariant, and the distortion of the content representation increased.
- We used two different types of loss function. The L2 norm gave better result(result here) than the perceptual loss function(result here). The feature vector extracted from the embeddings of the layers in the perceptual network was very sparse and this may be the reason for perceptual loss to not give better results. Alongwith that, the pretrained network was trained on dataset completely different from ours dataset. It was trained on YouTube Dataset but we used it for MagnaTagATune dataset. So, this may also be the reason for its decreased performance.
- For activation function tanh(result here) was better than RELU(result here).
- For reconstructing the audio signals from the spectogram, GriffinLim algorithm(result here) gave better result than simple inverse Fourier Transform(result here).

  This is a sample result of the spectogram for a style, content and the targeted output.

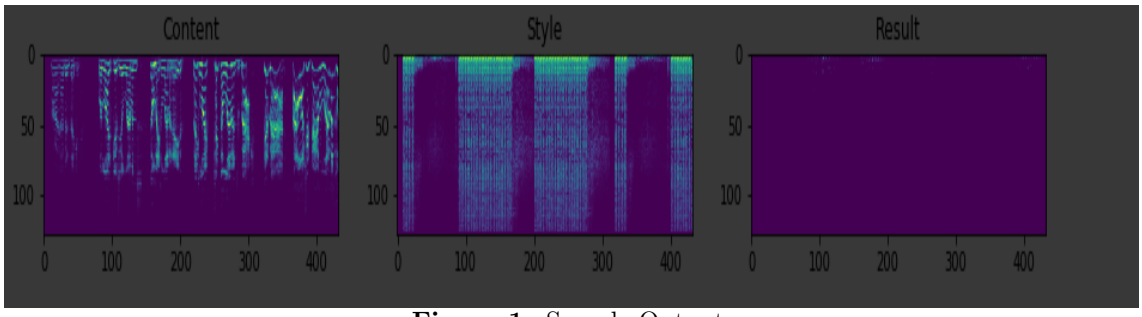

Figure 1: Sample Output

Our project is available here[6].

5 Conclusion

In this work we have successfully used a combination of approaches mentioned in several papers mentioned in prior work and borrowed some crucial aspects of our approach from them to get respectable results on the audio signals mentioned. We realise that though some features represent audio signal well it does not directly correlate to them being a good candidate to be used in the loss function. As an example we saw that perceptual loss from VGGish network did not help in the content reconstruction loss. Same thing we saw while using MFCC features. The positive correlation that we observes that more the dimensionality of our feature vector more was the pleasantness of the styled audio. This is a somewhat obvious to explain observation yet it was missing from most of the literature that we read. We hope that this work will help in motivating our juniors to take up some related audio style transfer project.

6 Problems Faced

- Integrating Perceptual Loss with the baseline code was a problem because the frameworks and libraries used was different.
- The blackbox nature of tensorflow took us two days to get into it and update the value of the variable we wanted to optimize to feed it to the VGGish network.
- tensorflow v1.15.0rc was the only one where our code was working. In one of the laptops the version was v2 and hence we faced this issue.
- After adding max pool and average pool layer it was taking a lot of time to invert the obtained spectogram after stylizing.

References

[1] Eric Grinstein, Ngoc Q. K. Duong, Alexey Ozerov, and Patrick Pérez. Audio style transfer. *CoRR*, abs/1710.11385, 2017. 1

[2] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. A neural algorithm of artistic style. *CoRR*, abs/1508.06576, 2015. 1

[3] Justin Johnson, Alexandre Alahi, and Fei-Fei Li. Perceptual losses for real-time style transfer and super-resolution. *CoRR*, abs/1603.08155, 2016. 1

[4] Google. tensorflow/models/vggish. `https://github.com/tensorflow/models/tree/master/research/audioset/vggish`. 1, 2

[5] DmitryUlyanov. neural-style-audio-tf. `https://github.com/DmitryUlyanov/neural-style-audio-tf`. 1

[6] Sudhanshu Raj and Sanchit Jain. audio_style_transfer. `https://github.com/Sudhanshuraj/audio_style_transfer`, 2019. 3