Assignment Report

Submitted in partial fulfillment of the requirements for ***Deep Learning - CS F425***

Generative Audio Networks Group-11

Name	ID
Ramanathan Rajaraman	2019A7PS0115G
Vishal Sharma	2019A7PS0036G
Aditya Sharma	2019A7PS0176G

Data Preprocessing

-Vishal Sharma

The audio data was loaded using the librosa library which generated raw waveform data. Various sampling rates were tried to minimize the number of features required to be learnt without compromising noticeably on the quality of the original audio. In the end the value chosen for the **sampling rate** was **3150** (default/7).

The length of the audio files loaded was clipped at **29 seconds** to avoid the situations where the file wasn't exactly 30 seconds long.

Different representations of the audio data were tried, namely **MFCC**, **mel spectrogram**, **stft** and **raw waveform** data. Among these, MFCC data couldn't reproduce the audio files reliably again and the stft data generated had complex numbers which pytorch accepts but most of the loss functions that were used did not behave as expected leading to truncation of imaginary part of the data. Finally, mel spectrogram data and raw waveform data were used to train and test the models.

Each audio file was **divided into half** and stored as separate audio files to increase the amount of data for training.

After processing the data was stored in numpy array format and saved as npy files for further use.

The .npy file used in the included .ipynb file has also been uploaded along with this report.

Notebook*:* Link

Generative Adversarial Network

- Aditya Sharma

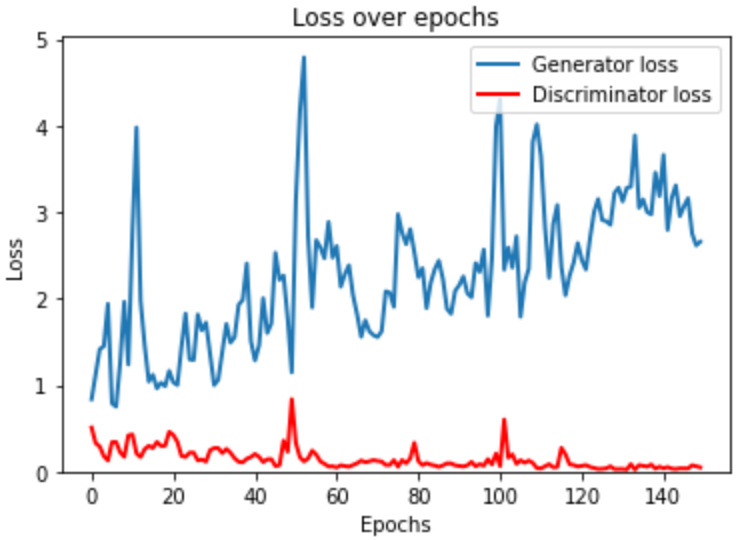
Mel Spectrogram was used in this model. A Generator class was made that took 1024 dimensional noise as input, and generated a [1025, 179] dimensional vector, which was the same shape as the input data. The generator used dense layers followed by sets of ConvInverse2d layers to generate the data. The Discriminator was a simple Dense network that classified an input as real or fake.

Loss for both Discriminator and Generator was Binary Crossentropy loss, while the optimizers for both were Adam. A weight decay of 1e-5 was used for regularization.

Notebook: Link

All 100 samples were split into batches of 10, so as to not exceed the available memory in Google Collab. Training the model for 100 epochs seemed to be the sweet spot for the model. Our GAN was not able to generate good samples because of the following potential reasons:

- Not a good representation of audio input was found.
- Memory limitations of Collab affected the complexity of the model.
- GANs are incredibly sensitive to the hyperparameters. Trying to tune the hyperparameters caused the GPU limit of collab to exceed very soon.



The audio generated seems to have some essence of classical music, but the output is noisy. The Audio file is playable in the last cell of the notebook.

Variational Autoencoder

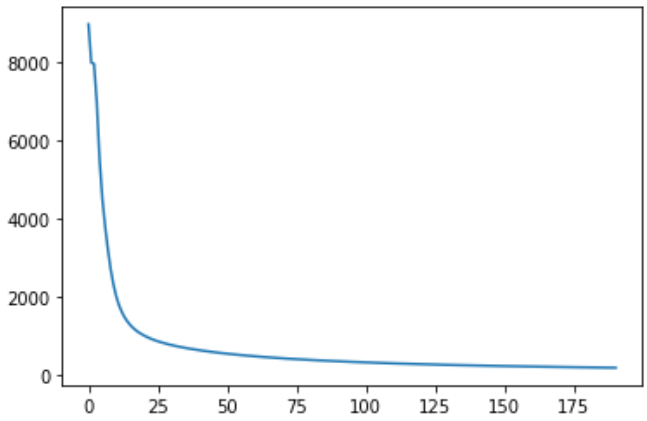
- Ramanathan Rajaraman

The **waveform** preprocessed data and used in 2 convolutional Variational Auto-Encoders. The first follows a simple 1D convolution architecture. The second follows the deep image processing network philosophy of **Conv-Pool** pairs. The reproduction loss used was **MSE.** Audio files are playable in the **last 3 cells** of both notebooks. Through trial and error we noticed that adding more genres to the train set led to crowding and noisier generation samples **hence we only used classical samples**.

Network 1: Link

The network was able to reduce KL-D loss but it struggles with reproduction. Total Loss has an initial and stark increment as can be seen in the notebook linked above. The following loss graph omits the first 8 epochs due to this irregularity. The number of **epochs trained was 200**.

Training Loss vs Epoch



Network 2: Link

This Model is our best performing network and please refer to the audio clips towards its end. It is also the file submitted along with this report. The network manages to balance the reduction of both KL-D and reproduction losses. The training loss curve behaves a lot more monotonically and also converges faster taking **only 50 epochs**. The Conv-Pool architecture was decided upon by taking into account that similar to spatial information in 2D images, temporal information in wave-form audio could be extracted using 1D Convs.

Training Loss vs Epoch