EECS 545 Project Progress Report: Tempo-Lite: Drum Beats Generated from Other Instruments

Congni Shi, Li Ye, Muye Jia, Jerry Cheng
{congni, yeliqd ,muyej, chengjry}@umich.edu

March 10, 2022

Abstract

Comparing to Vision, Audio generation has always been a field that is less studied with deep learning. In ths course project, a lightened version of a temporal music generator MuseGAN [?], Tempo-Lite, is proposed. The Tempo-Lite model takes the existing tracks of musical instruments other than drum and generate complementary drum tracks for the existing tracks. As a milestone of this objective, the group has also provided classification methods and model for the Pianoroll representation of music signals. Although the project is still in progress, the classification models has shown some promising results that can be used in following stages. The group has also finished the Generator and will soon be tested once the Discriminator is finished.

1 Introduction

2 Proposed Method

2.1 Classification for Global Induction

2.1.1 VGG-based classifier

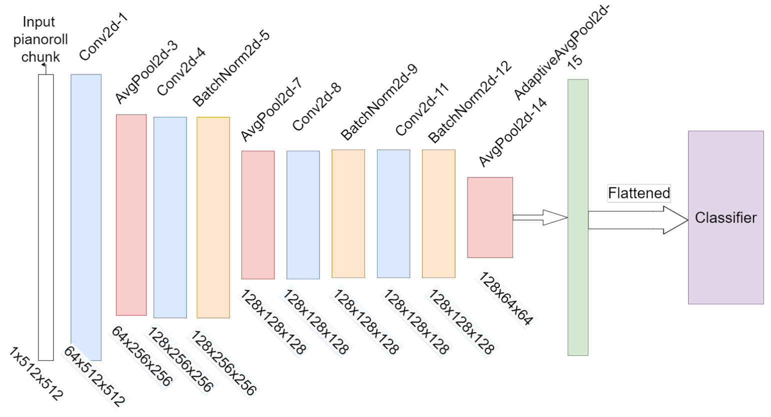

Figure 1: The feature extraction architecture of the CNN classifier

The first classification method the group proposed is a convolutional neural network (CNN) based classifier. The CNN classifier is based on the VGG-16 architecture, featuring convolution layers with small 3×3 filters. However to adapt the binary nature of the pianoroll, the team decided to replace the max pooling layers into average pooling layers to introduce more structural properties of the data as induction for the model. A more specific architecture will be discussed in the next several paragraphs.

Figure 1 shows the feature extraction architecture of the CNN classifier. The feature extraction is done by convolutional layers stride of 2 and padding of 1. We decided not to pad the input too much to maintain the structure of the binary data. The average pooling layers all have a kernel size of 2 and stride of 2 with no padding. The non-linear layer used in the feature extractor was ReLU.

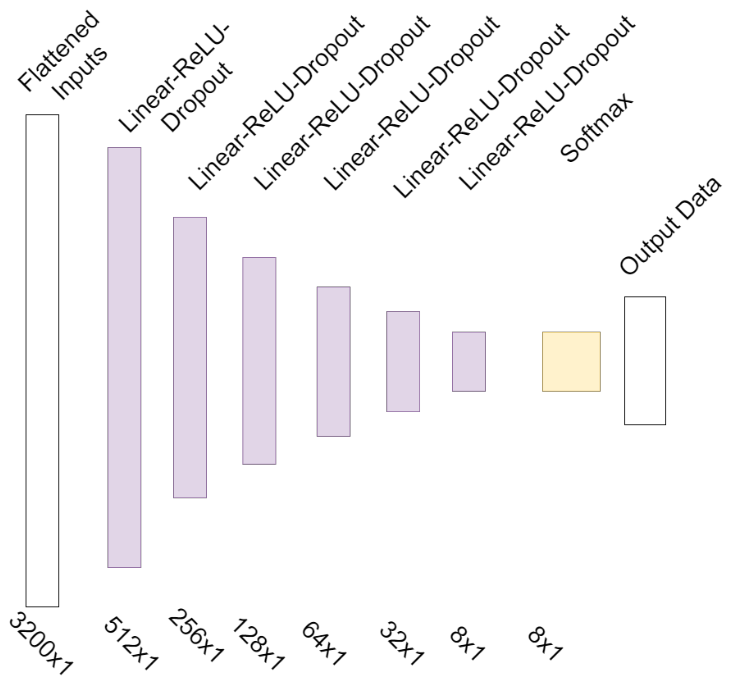

Figure 2: Linear Classifier of the CNN classifier

The output of the convolutional layers are pooled into a single vector. The output of the pooling layer is fed into a linear classifier, which is shown in Figure 2. Dropout layers in the linear classifier shares a probability of 0.3. Since we believe that can be used to prevent overfitting due to the small training set. In the end, the output of the linear-ReLU-Dropout layers was fed into a softmax layer for classification.

2.1.2 SVD classifier

2.1.3 Perceiver Classifier

2.2 Generator

2.3 Discriminator

3 Related Work

4 Experimental Results

4.1 Classification

4.1.1 CNN Classifier Results

The VGG-based CNN classifier was able to provide some sensible results for the classification task after training of 50 epochs. The network was able to give around 76% of accuracy upon the testing set and about 81% on the training set on average of 3 to 4 training attempt. The model was trained on the LPD-5 cleansed dataset on an Nvidia GTX 1070 first, with parallel training with different hyperparameters on a different machine with an Nvidia Tesla T4. Training of 50 epochs usually takes about 7 hours on either GPU with no significant difference. However, the Tesla T4 was able to train the model with a larger batch size with the additional graphical memories, which usually leads to a slightly better training results.

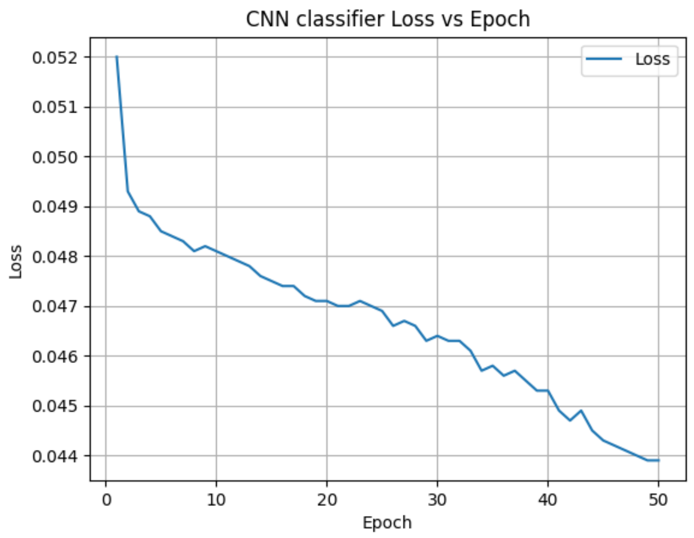

Figure 3: Loss Histogram of the CNN training

Figure 3 shows the loss histogram of training the classifier model. According to the plot, the loss of the training is not plateaued at the ending training epoch, signifying that the model is still learning, yet at the end of 50 epochs, the model was able to give an output of around 80% accuracy. Thus the group believe that with longer training time and better hardware accelerator, the model will be able to achieve better results. The group would try to investigate on further training with the next half of the project.

4.1.2 SVD Classifier

4.2 Generator

5 Future Milestone

6 Conclusion

Author Contributions

For this course project, Jerry Cheng has proposed the network architecture and their respective ablations, collected the data and constructed the customized dataloader. Jerry also finished the VGG-based CNN classifier. Congni Shi finished the naive SVD classifier, and incorporated the MFCC features once it was finished by Muye Jia. Muye wrote and tested the Generator and the MFCC feature extractor for the SVD classifier. Jerry oversees the cloud computing platforms and is in charge of the deployment of all deep learning models. All member contributed equally on this report.