Music Super-resolution with Spectral Flatness Loss and HiFi-GAN Discriminators

Jozef Marus Coldenhoff, Zeng Ren
CS-433 Machine Learning (2021), EPFL, Switzerland

***Abstract*—Recent work in audio-super-resolution tasks has focused on Generative adversarial networks as a means to improve the audio quality of low-resolution signals. In this paper, we investigate the integration of the HiFi-GAN discriminator into an existing GAN network. We observed a small improvement in the time-domain loss of the network during training. We also present the novel loss function spectral flatness that attempts to mitigate issues with the traditional time-domain loss. Adding this loss to the network resulted in less audible noise when the weight was properly tuned.**

I. Introduction

In this paper, we present a contribution to the research regarding audio super-resolution. Earlier works have shown promise in creating high-resolution audio from low-resolution input. Our contribution focuses on the integration of a state-of-the-art discriminator network (HiFi-GAN) to an existing GAN framework and analyzing the resulting GAN. Moreover, we investigate a new loss function that could mitigate some of the issues with current losses within the audio-super-resolution GAN setting.

II. Theoretical Background

A. GAN

A generative adversarial network (GAN) is a machine learning framework for estimating generative models for generic data. It has demonstrated its success in the vision domain, as well as the audio domain, including tasks such as style transfer, domain to domain translation[1], and super-resolution.

On a high level, GAN models the training process as a differentiable 2 player game between a generator and a discriminator, which are neural network models for synthesizing data and classifying data respectively [2].

1) Equilibrium Condition

GAN models the training process for generator and discriminator as a finding the model parameters (θ_G^*, θ_D^*) such that they achieve the Nash Equilibrium [3].

$$L(\theta_G, \theta_D^*) < L(\theta_G^*, \theta_D^*) < L(\theta_G^*, \theta_D)$$

If we further assume the loss for both generator and discriminator is twice differentiable, the above condition can be reduced to the first-order condition

$$\|\nabla_{\theta_G} L(\theta_G^*, \theta_D^*)\| = \|\nabla_{\theta_D} L(\theta_G^*, \theta_D^*)\| = 0,$$

together with the second-order condition that $\nabla^2_{\theta_G} L(\theta_G^*, \theta_D^*)$ being positive definite and $\nabla^2_{\theta_D} L(\theta_G^*, \theta_D^*)$ being negative definite.

2) Optimal Generator and Discriminator

The optimal Generator is the one such that $p_g = p_d$, whose output distribution matches the true distribution.

The optimal Discriminator is the one achieves accuracy:

$$D^*(x) = \frac{p_d(x)}{p_d(x) + p_g(x)}$$

III. Model Architecture

In this section, we will provide an overview of the model architecture. As presented in the previous section, our model consists of two distinct networks, the discriminator, and the generator. The discriminator used in this implementation was first presented in the paper titled *"HiFi-GAN: Generative Adversarial Networks for Efficient and High Fidelity Speech Synthesis "* [4]. In their paper, they present two distinct types of discriminators, which in conjunction make up the entire discriminator network. The two presented models are the Multi-Period discriminator (MPD) and the Multi-Scale discriminator (MSD).

A. Multi-Period Discriminator

In their paper, the authors explain that human speech signals consist of sinusoidal audio signals of various periods, the same is true for natural audio signals in general. Because of the fact that audio consists of signals of multiple periods, the authors propose the MPD, this MPD consists of multiple sub-discriminators that each act on a subset of equally spaced audio signals. For each sub-discriminator, the audio is reshaped from its original length T to a data matrix of shape pxT where p is the period of the sub-discriminator. On this new reshaped data, a 2D convolution with stride 3 with kernel size 1xk where k is 5 is applied in order to only apply the convolution on one equally spaced sample of audio. Each sub-discriminator is a stack of 6 convolutions layers with input sizes $[1, 32, 128, 512, 1024, 1024]$ and output sizes $[32, 128, 512, 1024, 1]$. After each convolutional layer except the last, a leaky ReLU activation is used with slop 0.1. The complete MPD then consists of 5 sub-discriminators with periods $[2, 3, 5, 7, 11]$, note that the choice was made to use prime periods as this minimizes the overlap between the sub-discriminators. Figure 1 b shows an overview of an MPD sub-discriminator with a period of 3.

B. Multi-Scale discriminator

The second network in the discriminator is the MSD, this network, in contrast with MPD, attempts to capture the

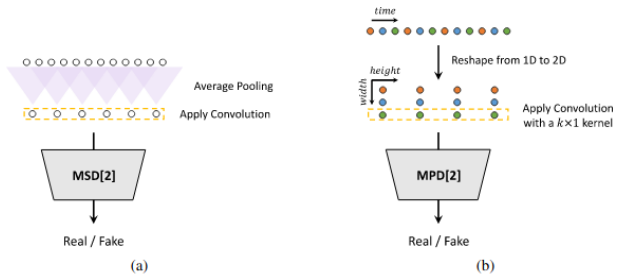

Figure 1. Sub-discriminators of MSD and MPD. On the left is a sub-discriminator of MSD with 4x average pooled audio. On the right a sub-discriminator with period 3. [4]

larger structure of the waveform, i.e. repetitive patterns and long-term dependencies. To this end, the MSD uses multiple sub-discriminators with different pooling of the input audio signal to operate on different smoothed wave-forms [4]. This MSD architecture was proposed in the MelGAN architecture [5].

In the MSD architecture, a sub-discriminator consists of a pooling layer that smoothes the raw input wave-form, followed by stacked 2D convolutional layers with leaky ReLU activation. The pooling layers used in the network are 1x, 2x, and 4x average pooling, meaning that the first sub-discriminator simply works on the raw waveform. The convolutional layers have input sizes of $[1, 128, 128, 256, 512, 1024, 1024, 1024]$ and output sizes of $[128, 128, 256, 512, 1024, 1024, 1024, 1]$.

C. Generator

The generator used in our network is the same as the implementation created by [6]. The generator network is based on the paper by [7]. The generator consists of a U-Net architecture in which multiple down and up-sampling blocks make up the network [8]. The architecture uses 8 up and down-sampling blocks combined with skip connections. In addition, the generator makes use of inception modules [9] in order to prevent the careful hand selection of kernel sizes. Moreover, the generator implements subpixel and superpixel layers[10]. These superpixel and subpixel layers were used to increase the spatial resolution, while at the same time preventing checkerboard artifacts which can be induced by large spatial filters.

IV. Audio Domain Background

A. Motivation for using spectral flatness

If we look at the spectrum of a sound that gives us a clear sense of pitch, we will find regularities in the distribution of its frequencies. In particular, the energy of the signal should be concentrated on the harmonic series of a fundamental pitch. The distribution of the frequencies in the sound at a given time should be far away from uniform.

Spectral flatness [11], similar to entropy, describes the notion of "spikiness" or "flatness" of the frequency domain distribution of a sound. Because of this reason, we are interested in incorporating spectral flatness in a loss function to guide our generative model.

We hope that the spectral flatness loss could mitigate an issue with the current time-domain loss, where a network could attempt to minimize the loss by simply adding an average noise on a specific frequency band.

V. Loss Functions

The total loss for the discriminator and the generator is the following:

$$L_{total}(D;G) = L_{adv}(D;G)$$

$$L_{total}(G;D) = \lambda_{adv}L_{adv}(G;D)+\lambda_{flat}L_{flat}(G)+L_{time}(G)$$

A. Adversarial losses

The Discriminator adversarial losses is normally of following form:

$$L_{adv}(D;G) = \mathbb{E}\left[log(D(x)) + log(1 - D(G(x)))\right]$$

We applied the L_2 version, as this loss was used in HiFi-GAN, and prevents vanishing gradients during the training process [12].

$$L_{adv}(D;G) = \mathbb{E}\left[D^2(x) + (1 - D(G(x))^2\right]$$

B. Time-domain loss

Normally one would use the Mel-spectrogram loss:

$$L_{Mel}(G) = \mathbb{E}\left[\|\phi(x_G) - \phi(x_D)\|_2^2\right]$$

where ϕ is the function that transforms audio to its Mel-spectrogram. However, since in our cause we are working with the time-domain signal directly, we use the time-domain loss:

$$L_{time}(G) = \mathbb{E}\left[\|x_G - x_D\|_2^2\right]$$

C. Spectral Flatness loss

$$L_{flat}(G) = \mathbb{E}\left[\|SF(x_G) - SF(x_D\|_2^2)\right]$$

$$SF(x) = \frac{\exp(\frac{1}{N_w}\sum_w \ln(X(w,t)))}{\frac{1}{N_w}\sum_w X(w,t)}$$

where X is the short-term Fourier transform of x. SF is a vector of the ratios between the geometric mean and the arithmetic mean of the short-term Fourier transform signal across frequency.

Librosa [13], a popular Python package for audio signals has implemented spectral flatness. We build a *PyTorch* version so that gradient could pass through the function to train the GAN model. Our *PyTorch* version was tested against *Librosa*'s implementation to ensure that they have the same result.

VI. Data Preparation

A. Using EQ modification to simulate "low-resolution" audio

Instead of down-sampling by applying noise to the audio, we silence out the high-frequency sound from the audio to achieve the effect of low-resolution audio, creating a mumbling texture that blurs the sound. Based on our listening, we feel this choice better captures the perceptual notion of low-resolution audio and it is the same effect as one covering a speaker with a cardboard box for example.

Here is an example of the Mel-spectrograms for the original and EQ modified audio sample.

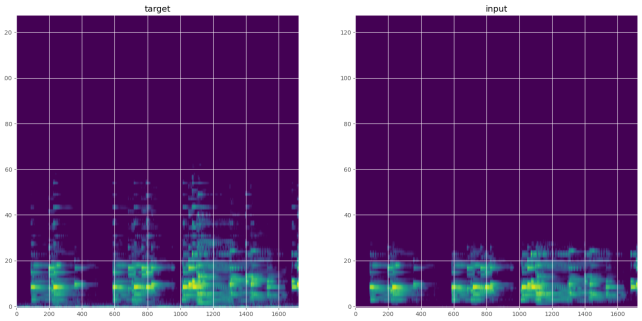

Figure 2. Down sampling with silencing the high frequency sound. Left is the original (target) sample and right is the EQ modified (input) sample

VII. Training and Evaluation

A. Hyper-parameters

We tried various values $[1, 10^{-1}, 10^{-2}, 10^{-3}]$ on two of the main hyper-parameters: λ_{adv} and λ_{flat} which are the weight for the adversarial loss and the weight for spectral flatness loss respectively.

B. Scheduler, optimizer

During the training of the models, except the original GAN, the AdamW optimizer was used with $\beta_1 = 0.8, \beta_2 = 0.99$, and a learning rate of 0.0002. We used an exponential step-size decay on the learning rate with $\gamma = 0.999$.

C. Loss curves

In the first comparison, we examine the effect of incorporating HiFi-GAN multi-discriminator on the general loss in the time-domain. We choose this loss to compare because it's the only objective criteria that are the same between the baseline and current model. In Figure 3, the dark blue curve is our baseline (without the Hifi-GAN multi-discriminator) and the light blue and grey curve is the new discriminator with spectral flatness loss untoggled/toggled. We see that the Hifi-GAN discriminator improves the time-domain generator loss.

In particular, within the new discriminator model, we compared the training processes of two runs, the first is without spectral flatness ($\lambda_{adv} = 10^{-3}, \lambda_{flat} = 0$) and the second is with spectral flatness ($\lambda_{adv} = 10^{-3}, \lambda_{flat} = 10^{-3}$). Both runs were 20 epochs and took 6 hours each to finish. From Figures 4, 3, and 5, we see that model can minimize the spectral flatness without sacrificing for worse performance in Discriminator loss and Generator time-domain loss. This shows that the spectral flatness loss, at least at our current training stage and with a low weight setting, does not cause conflict with the other loss. However, in other runs when we set λ_{flat} to be higher, we observe that the generator's time-domain loss was sacrificed for the spectral flatness loss. This means that tuning λ_{flat} is a subtle process.

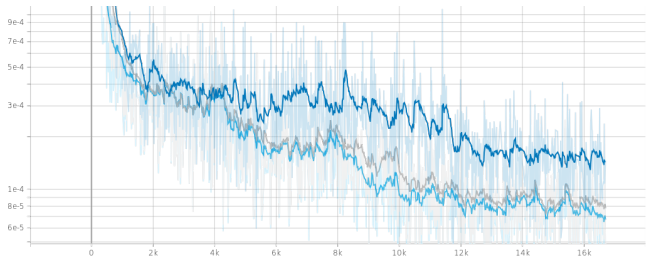

Figure 3. Generator loss on Time-domain (grey is with spectral flatness, light blue is without spectral flatness, dark blue is the baseline)

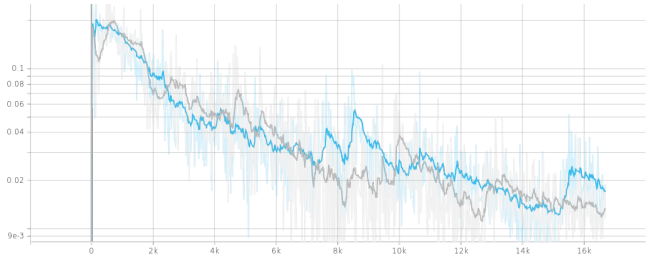

Figure 4. Discriminator loss (grey is with spectral flatness, blue is without spectral flatness)

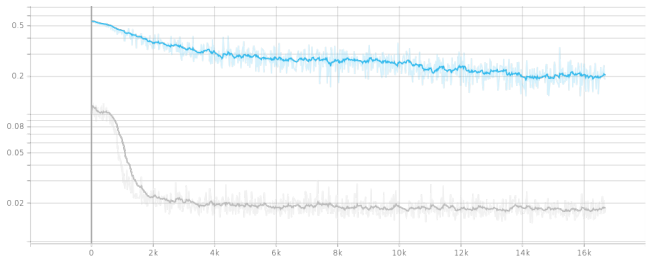

Figure 5. generator spectral flatness loss (grey is with spectral flatness, blue is without spectral flatness)

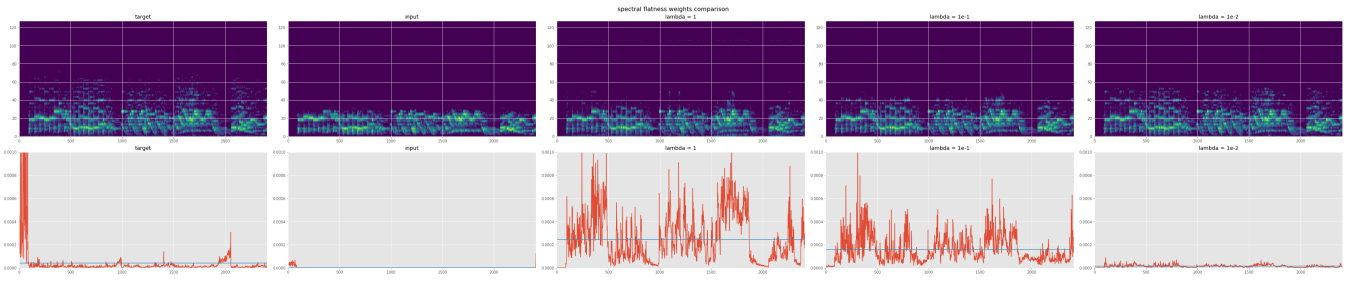

Figure 6. Spectral flatness (bottom row) curves alongside with Mel-spectrogram (upper row). For the generated samples (the 3 columns of the right side), the perceived quality increases as the spectral flatness curve decreases.

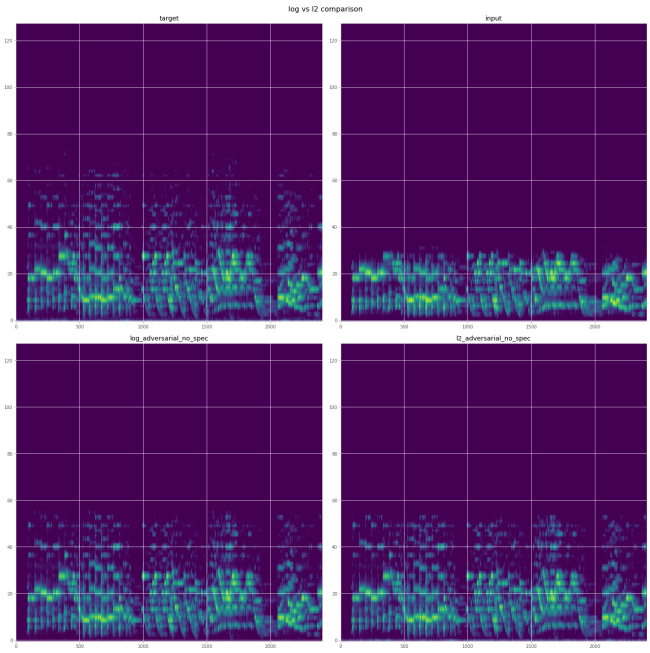

Figure 7. We observe a minimal difference in the output samples when using cross-entropy loss vs L_2 in binary classification

VIII. Discussion

A. Choice between log and L_2 on binary classification loss

Aside from preventing the vanishing gradient problem for log, another point is interest is to examine is how this choice affects the generated result. From Figure 7 We observed a minimal difference in the Mel spectrogram of the output sample. We also listened to the audio and found no particular differences.

B. Effect of Spectral flatness loss on the perceived quality of the model output

There are many unexpected results, both positive and negative, concerning the spectral flatness loss. On the positive side, across the different generated samples for the same input, the ones that sound better are the ones with low spectral flatness, this means the notion of spectral flatness is at least pointing to the right direction (see Figure 6), moreover, from close listening, it became apparent that spectral flatness suppressed the volume of noise in the generated signal. On the negative side, high weights for spectral flatness loss lead to a high-frequency noise and high spectral flatness in the generated audio, which is against our intuition.

There could be several reasons that lead to this phenomenon and could be further investigated in further research. First, since the spectral flatness loss is implemented as minimizing the $L2$ distance between the generated target sample, some outliers at some point in time, in particular silence, could tamper with the distance. Looking at the input spectral flatness curve in figure 6, we see that the overall spectral flatness is low but there is this high flatness region before the onset of the first note. We suspect that this outlier segment encourages the model to generate results with high spectral flatness. One possible solution to account for this is to weigh spectral flatness at a certain time window by the signal's amplitude.

C. Beyond spectral flatness

Spectral flatness is invariant of permutation in the frequency domain. So it only captures part of the acoustic regularities above. The rest, in particular, the information about harmonic series is outside the scope of this measure. In future works, we could design a better loss function that further focuses on the distribution of harmonics.

IX. Summary

Our contribution consists of two parts. First, by incorporating the Hifi-GAN multi-discriminator, we observed improvement in the time domain loss for the generator. Second, we found some interesting effects of the spectral flatness loss that are both positive and negative, which may provide further insights on designing a new loss function to control the quality of music audio synthesis. Our code could be found here [1].

[1] https://github.com/CS-433/ml-project-2-super_audio

REFERENCES

[1] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Computer Vision (ICCV), 2017 IEEE International Conference on*, 2017.

[2] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," 2014.

[3] J. F. Nash, "Equilibrium points in n-person games," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 36, no. 48-49, 1950.

[4] J. Kong, J. Kim, and J. Bae, "Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis," 2020.

[5] K. Kumar, R. Kumar, T. de Boissiere, L. Gestin, W. Z. Teoh, J. Sotelo, A. de Brebisson, Y. Bengio, and A. Courville, "Melgan: Generative adversarial networks for conditional waveform synthesis," 2019.

[6] P. A. Alahi, B. Sifringer, and T. Stegmüller, "Audio enhancing with wasserstein gans," 2020. [Online]. Available: https://github.com/stegmuel/audio-super-resolution/blob/master/docs/report.pdf

[7] S. Kim and V. Sathe, "Adversarial audio super-resolution with unsupervised feature losses," 2019. [Online]. Available: https://openreview.net/forum?id=H1eH4n09KX

[8] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," 2015.

[9] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," 2014.

[10] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," 2016.

[11] J. Johnston, "Transform coding of audio signals using perceptual noise criteria," *IEEE Journal on Selected Areas in Communications*, vol. 6, no. 2, pp. 314–323, 1988.

[12] X. Mao, Q. Li, H. Xie, R. Y. K. Lau, Z. Wang, and S. P. Smolley, "Least squares generative adversarial networks," 2017.

[13] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "librosa: Audio and music signal analysis in python," in *Proceedings of the 14th python in science conference*, vol. 8, 2015.