Adversarial Audio Generation by penalizing the Banach Space Gradients

Ambarish Gurjar, Dipam Vasani, Abhigya Agrawal
Luddy School of Informatics, Computing and Engineering
Indiana University
Bloomington, IN 47408
{agurjar,dvasani,abagra}@iu.edu

Abstract

Non-progressive audio generative architectures like WaveGAN have recently shown promising results in a raw-waveform generation of audio. The WaveGAN is capable of generating coherent speech sounds as well as drums, piano and bird sounds. The architecture of the WaveGAN uses a WGAN-GP for training. The evaluation scores of these outputs are just satisfactory. We propose clipping of weights by imposing a gradient penalty over Banach space of the discriminator rather than penalizing the norm of the gradients. This method has shown boosts performance for generating image data. We explore the use of this method over audio data to explore the effects of the choice of the norm for stable training and efficient feature learning.

1 Introduction

The Generative Adversarial Networks(GANs) are a class of unsupervised neural networks used to generate synthetic data (7). They consist of 2 neural networks, namely, the generator and the critic which compete against each other. The generator is supposed to learn how to generate samples from the given dataset while the discriminator is supposed to learn whether the given images are fake or real. Both networks get better with training thereby generating high-quality samples of data bearing features that are strikingly similar to the original dataset. The GANs have found a lot of success in generating image data from datasets like the CIFAR-10, Celeb-A, LSUN Bedroom dataset. However, audio data is a lot more complex as it involves a time dimension. A team of researchers from Google Brain has demonstrated promising results in generating audio data using the WaveGAN model.

GANs were traditionally known to be unstable and posed a difficulty in converging. To tackle this problem a Wasserstein distance metric or an Earth mover's distance was introduced (2). This way the generator would learn in a stable manner and generate data that was similarly distributed with respects to the training data. To even improve the performance of this metric, a gradient penalty was imposed on the Wasserstein metric so that the algorithm doesn't learn to model unimportant features of the data quickly.(9). The gradient penalty was imposed using a simple L^2 norm of the gradients. It was said that L^2 norm is too general and using the choice of the norm used to impose a gradient penalty can highly influence the quality of generation and can add a significant boost in evaluation scores (1).

For the scope of this project, we will be generating audio data using piano, bird chirp, and drum dataset. We will be using WaveGAN as a baseline and testing if the dual norm penalized gradients effectively increase the quality of the generated samples.

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

2 Background

2.1 Generative Adversarial Networks

In GANs, we define two different neural networks, the generator, and the discriminator. The generator takes random noise as input and tries to produce an output similar to a given data space. The discriminator then takes either the real or the generated data as input and tries to label it. The idea behind training GANs is to have these two models compete against each other in the hope that they co-train (7). Eventually, the generator will become so good that the discriminator won't be able to distinguish between generated and real data. This condition is reached when a Nash Equilibrium is achieved.

The Minimax objective between the Generator G and Discriminator D is defined as follows:

$$\min_{G} \max_{D} \mathop{\mathbb{E}}_{X \sim \mathbb{P}_r} [\log(D(X))] + \mathop{\mathbb{E}}_{Z \sim \mathbb{P}_Z} [\log(1 - D(G_\theta(Z)))] \,. \tag{1}$$

GANs have a lot of variants for different applications. Some really interesting applications include pose guided image generation, style transfer, and super-resolution. However, they are not limited to images. GANs have also been used for audio generation. A popular variant for this particular application is called WaveGAN. However, we first want to introduce the Wasserstein metric and Wasserstein GAN with a gradient penalty as they serve as important components in our proposed method.

2.2 WGAN and WGAN-GP

The generative modeling community has used the Kullback–Leibler divergence or the Jensen-Shannon divergence to estimate the likelihood of a generated distribution being similar to the original dataset. These divergences needed to be minimized to approximate a good generation function. However, there were problems like the GAN would too unstable to converge with such loss functions. A Wasserstein-1 metric was introduced to calculate the distance between the distribution of the generated sample and the distribution of the real data.

The Wasserstein-1 distance is calcualted using the Kantorovich-Rubinstein duality(12)

$$W(_{r}, _{\theta}) = \sup_{\|f\|_L \leq 1} {}_{x \sim \mathbb{P}_r}[f(x)] - {}_{x \sim \mathbb{P}_\theta} [f(x)] \tag{2}$$

Where the supremum is over all the 1-Lipschitz functions. The GAN generally tends to converge if the loss function is able to enforce the Lipschitz constraint.

For WGAN-GP, where GP stands for gradient penalty, the weights of the discriminator are clipped by using a gradient penalty that enforces the 1-Lipschitz constraint in a creative way. The norm of the gradient of critic's output with respect to its input is constrained. Thus the loss is calculated using the following objective function.

$$L = (\mathbb{E}_{X \sim \mathbb{P}_\Theta} D(X) - \mathbb{E}_{X \sim \mathbb{P}_r} D(X)) + \lambda \mathbb{E}_{\hat{X} \sim \mathbb{P}_{\hat{X}}} [\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1)^2] \tag{3}$$

Where, λ is a regularization factor that is usually set as 10. Here a soft enforcement of the Lipschitz constraint is observed. And using this method implies that an L^2 norm is chosen to calculate the underlying distance measure on the data space.

2.3 WaveGAN

WaveGAN aims to generate audio using raw waveform data (4). The authors of the paper have chosen two approaches for generative modeling of audio. Using Waveform data and using spectrogram data. However, it was suggested that the spectrograms are non-invertible thus creating problems in a generative setting (8).

The architecture of the WaveGAN is constructed by choosing the DCGAN as a base (10). Transposed convolutional layers are used to upsample low dimensional latent space to high-quality outputs. The 2-D convolutional filters are replaced with 1-D filters to effectively capture features of audio data.

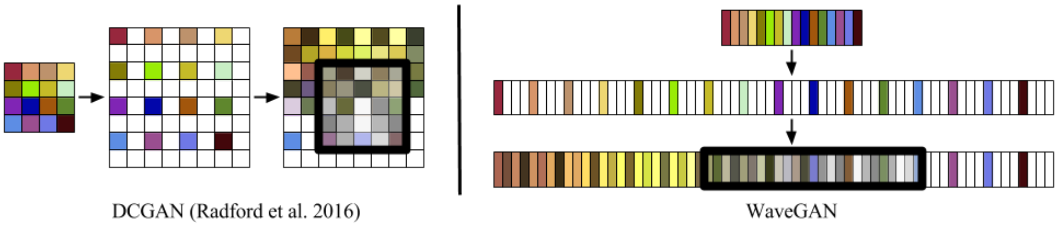

Figure 1: Illustration of how convolutional filters are flattend in WaveGAN

Thus a 5x5 filter is flattened in a 25- length 1-D filter. The WGAN-GP method is used for training. A phase shuffling method is introduced that shuffles the phase of the latent space. This makes the job of the critic tougher but it acts as an effective optimization technique. This method discourages the critic to learn trivial approximation policies. This is especially important when waveform based data is concerned.

2.4 Banach Spaces

A normed space is called a Banach space if its Cauchy Sequence converges. It is a complete normed vector space that enables us to calculate distance betweens vectors (11). Let's start by talking about L^p spaces. Let Ω be some domain The equation of a L^p space is given by

$$\|x\|_{L^p} = \left(\int_\Omega x(t)^p dt \right)^{1/p} \tag{4}$$

Sobolev Spaces are useful while calculating the edges. In a sense L^p spaces can be considered as special cases of a Sobolev Space where s=0. The equation for Sobolev spaces are given by the following equation

$$\|x\|_{W^{1,p}} = \left(\int_\Omega x(t)^p + |\nabla x(t)|^2 dt \right)^{1/p} \tag{5}$$

In this equation s is set as 1. Calculating gradients is effectively equivalent to multiplying with ξ in Fourier Space. Thus we compute a Fourier transform. Conduct the operations and then take the inverse Fourier transform (3).

$$\|x\|_{W^{s,p}} = \left(\int_\Omega \left(\mathcal{F}^{-1} \left[(1 + |\xi|^2)^{s/2} \mathcal{F}x \right] (t) \right)^p dt \right)^{1/p} \tag{6}$$

In the next section, we discuss about our proposed method to train the GAN for audio synthesis based on the concepts highlighted in this section.

3 Banach Space penalized gradients for audio generation

This section will outline our proposed method. The generator makes one update per five updates of the discriminator. The noise variables are initialized at the start of the training. The Generator and Discriminator architectures conform to the specifications of the WaveGAN model. The way the algorithm works is a latent space variable z is fed into the generator.

$$\tilde{x} \leftarrow G(z) \tag{7}$$

An interpolates variable $\hat{X}$ is calculated using real data and generated fake data. The equation to calculate the interpolates variable is very similar to the KL-divergence. Here x is the real data while $\tilde{x}$ is the fake data.

$$\hat{X} \leftarrow \epsilon x + (1 - \epsilon)\tilde{x} \tag{8}$$

The interpolates variable $\hat{X}$ is fed into the discriminator. And the gradients of the output of the discriminator are calculated.

$$gradients \leftarrow \nabla D(\hat{X}) \tag{9}$$

A dual norm of of the gradients is taken. First a Sobolev norm is taken using equation (6) that is followed by an L^p norm using the equation (4). The Sobolev norm step is especially computationally heavy as it involves taking a Fourier and inverse-Fourier transform of a high length 4-D tensor.

The loss function that we have generalized in our approach is given below. The Banach space norm or the dual norm used is denoted by $||.||_{B^*}$. The author's of the Banach Wasserstein GAN paper have already proven a λ-Lipchitz constrain is enforced if the Banach space is frechet differentiable.

$$L = \underbrace{\frac{1}{\gamma}(\mathbb{E}_{X\sim\mathbb{P}_\theta} D(X) - \mathbb{E}_{X\sim\mathbb{P}_r} D(X))}_{\text{Wasserstein loss}} + \underbrace{\lambda\mathbb{E}_{\hat{X}}\left(\frac{1}{\gamma}\|\nabla D(\hat{X})\|_{B^*} - 1\right)^2}_{\text{dual normed gradient penalty}} \tag{10}$$

4 Experimental Setup

We have used the following 3 different datasets for our experiments with a combined audio length of approximately 2 hours.

- Piano: Consists of professional pianists playing a variety of Bach compositions.
- Drums: Consists of audio files with single drum hits.
- Birds: Consists of sounds of birds in wildlife.

We start by generating a baseline on these datasets using WaveGAN with phase shuffle. For this, we used a single NVIDIA K80 GPU and a batch size of 3. The optimizer used was Adam with beta1 and beta2 values set to 0.5 and 0.9 respectively. For the piano dataset, the network converged after around 2.5 days of training and 16k epochs. We saved our model and generated outputs after every 100 epochs and in 11k epochs, the generator was capable of generating decent outputs as confirmed by human feedback.

We then trained our own BW-WaveGAN (we are open to suggestions for a better name) on these 3 datasets by optimizing for gradient penalty. The gradient penalty regularization factor lambda is set to 10. Again, we use a single NVIDIA K80 GPU per dataset for training. The batch size was reduced to 1. The model size was set to 8 and the number of input channels to our generator was 128 (model size * 16). The network took longer to train compared to WaveGAN with a training time of 4 days and converging around 12k epochs.

For this implementation, s of sobolev norm equals 0 and c equals to 5. The p of L^p norm is 2.

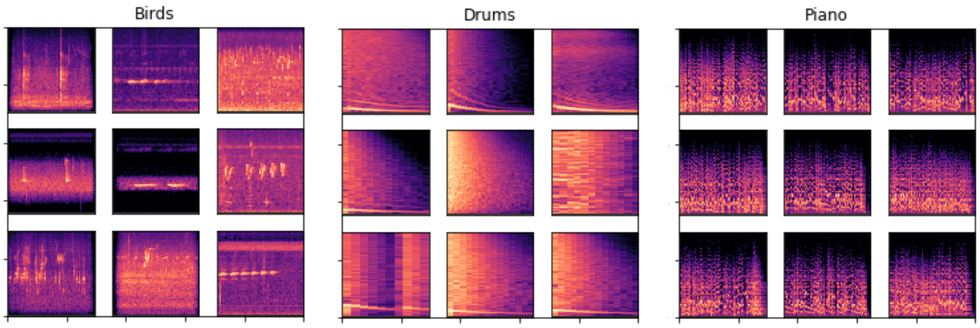

Figure 2: Spectrograms of random samples of the various training datasets

5 Discussion and Future Work

We have attached generated samples along with this report. We hope human evaluation would be an appropriate metric to determine the success of this project. The duration factor in training this model was time crucial hence we could not perform an evaluation of the generated samples. In the GANSynth paper, evaluation metrics such as Inception Score, Frechet Inception Distance (FID score), Number of Statistically Different Bins (NDB), Pitch accuracy and Pitch entropy are mentioned along with human evaluation(5). We plan to use all these metrics while submitting a mature version of this implementation in an academic publication.

To accommodate the computational resources required by the Fourier and inverse Fourier transform required to calculate the Sobolev norm, we needed to compromise on the size of the model. The largest convolutional layer of the generator had only 128 input channels. The model deployed by the authors of the Wavegan was significantly larger with 1024 input channels. Thus the outputs generated by their generator were significantly better. We ran baseline tests with various model size and the general trend depicts that a larger model produces better samples. We even tested a generator with 4096 input channels which were really nice. However, in order to draw a comparison outputs of low sized baseline and proposed method networks are included. We would want to leverage the supercomputing resources in the university for our camera-ready work.

We would also like to test various values of s and p for L^p and Sobolev norms to find whether various datasets have performance sensitivities to different underlying distances. Also, we would want to expand our datasets and also try out the speech datasets like TIMIT and SC09 (6).

We believe that this project shows promising early results and would like to contribute to its advancement in the future. Using strong evaluation metrics and appropriate computational resources we think this research would push the state of the art. To the best of our knowledge, we couldn't find any research that generated audio from waveforms where weight clipping was implemented using Banach spaces of the gradients. Most of the generative community is still trying to find creative methods to enforce the Lipschitz constraint.

Acknowledgments

We would like to thank Dr. Minje Kim. ENGR 533 was truly one of the best courses we have ever taken in our entire lives. We would also like to thank Google to be generous enough to give a 50 dollar credit and a platform like Colab that made project implementation really easy. We would also want to thank the AIs and everyone who contributed on piazza. Also a special mention to Jonas Adler from DeepMind, the author of the original Banach Wasserstein GAN paper. He provided us with some guidance regarding the Sobolev Spaces.

References

[1] Jonas Adler and Sebastian Lunz. Banach wasserstein gan, 2018. `arXiv:1806.06621`.

[2] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan, 2017. `arXiv:1701.07875`.

[3] Haim Brezis. *Functional analysis, Sobolev spaces and partial differential equations*. Springer Science & Business Media, 2010.

[4] Chris Donahue, Julian McAuley, and Miller Puckette. Adversarial audio synthesis, 2018. `arXiv:1802.04208`.

[5] Jesse Engel, Kumar Krishna Agrawal, Shuo Chen, Ishaan Gulrajani, Chris Donahue, and Adam Roberts. Gansynth: Adversarial neural audio synthesis, 2019. `arXiv:1902.08710`.

[6] John S Garofolo. Timit acoustic phonetic continuous speech corpus. *Linguistic Data Consortium, 1993*, 1993.

[7] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[8] Daniel Griffin and Jae Lim. Signal estimation from modified short-time fourier transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 32(2):236–243, 1984.

[9] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans, 2017. `arXiv:1704.00028`.

[10] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[11] Walter Rudin. Functional analysis, mcgrawhill. *Inc, New York*, 1991.

[12] Cédric Villani. Optimal transport: Old and new. 2008.