# Do GAN's always have Nash Equilibria?

Shivam Sharma* Walstan Baptista †

May 3, 2022

**Abstract**

General Adversarial Networks (GANs) were first introduced in [1]. Since than they have been successfully used in many applications, such as image generation, speech synthesis, and text generation. GANs are a type of zero-sum game that can be used to generate new data from existing data. GANs are consist of two neural networks, namely generator and discriminator. In this paper we will discuss a noval approach to solve the games with no Nash Equilibria. We have taken the approach discussed in [2] and implemented proximal operator to generate an objective function for original objective. The solution we got from the proximal operator is called proximal equilibrium. In GANs generator first generates the data and than discriminator tries to classify the data, which can be studied by proximal equilibrium.

## 1 Introduction

As we know, GANs are a type of zero-sum game, betweeen the players, the generator and the discriminator. This game is generally written as minimax optimization problem.

$$\min_{G\epsilon\mathcal{G}} \max_{D\epsilon\mathcal{D}} V(G, D) \tag{1}$$

In 1 we have generator, $\mathcal{G}$ Which will generate input data, discriminator, $\mathcal{D}$ which will classify the data. In other words, $\mathcal{D}$ will try to find out the difference between data generated by $\mathcal{G}$ and real training data. Here $\mathcal{G}$ and $\mathcal{D}$ are two deep neural networks. And $\mathcal{V}$ is minimax objective functon with $\mathcal{G}$ and $\mathcal{D}$ as input players.

Although the problem lies in the form of minimax optimization, which is a general problem, we will try to solve it with a new approach. GAN training is well-known as a difficult optimization job which include many hyper-Parameters to be tuned. Generally, such game can be solved by Nash Equilibrium. Nash Equilibrium is a solution of the game in which none of the players can increase their gain by playing different strategy. We will try to find out the Nash Equilibrium of the game. For this game let us consider $(G^*, D^*)$ are the Nash Equilibria for the 1 problem. Then it should satisfy following condition.

$$V(G^*, D) \leq V(G^*, D^*) \leq V(G, D^*) \tag{2}$$

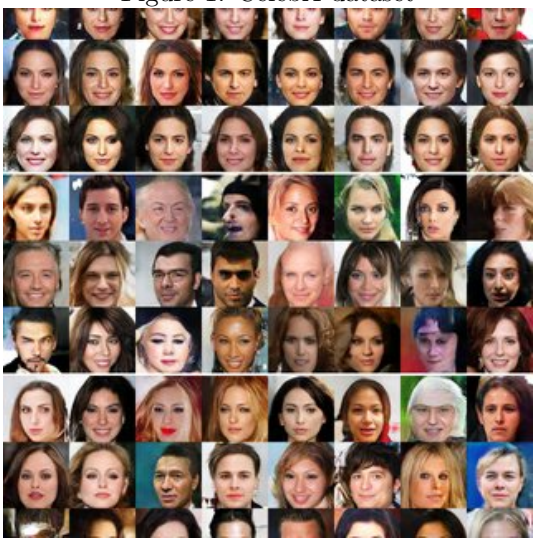
But, we can not find out the Nash Equilibrium of every GAN game. In this report we will try to find out the Nash Equilibrium of some famous GAN games, like vanilla GAN, Wasserstein GAN, and 2-Wasserstein GANs. As a result, we will define new notion for the equilibrium of GANs defined in [2]. This notion is called proximal equilibrium. As given in [3], this game resembles the sequential games and hence we can implement subgame perfect equilibrium (SPE) in the game to find equilibrium.

$$V^{prox}(G, D) := \max_{\tilde{D}\epsilon\mathcal{D}} V(G, \tilde{D}) - ||\tilde{D} - D||^2 \tag{3}$$

$$\min_{G\epsilon\mathcal{G}} \max_{D\epsilon\mathcal{D}} V^{prox}(G, D) \tag{4}$$

* RAS Master's 1st year
†RAS Master's 1st year

Figure 1: CelebA dataset



Author proposes proximal training in [2], and take 3 as the proximal objective and solve 4 by alternating gradient methods.

## 2 Problem Setup

As we have discussed in previous section, not for all games, Nash Equilibrium exist. In this section we will try to find out the Nash Equilibrium of some famous GAN games. We will study Nash Equilibrium on three standard GAN architectures. First being Wasserstein GAN with weight clipping (WGAN-WC) [4] , second being improved Wasserstein GAN with gradient penalty [5] , and third being Spectrally-normalized vanilla GAN [6].

We have used MNIST dataset [7] and CelebA dataset [8]. MNIST stands for Modified NAtional Institute of Standards and Technology datasets, and it contains 60, 0000 images of 28 x 28 pixels of each handwritten digits from 0 to 9. On the other hand, CelebA dataset contains images of famous celebs. It contains more than 200, 000 images of each celebrity face with 40 attributes.

Now, in order to achieve Nash Equilibrium, we will run GAN training for 200,000 steps to achieve $G_{\theta_{final}}, D_{W_{final}}$. To examine weather it is a Nash Equilibrium, we will use the approach defined in [2]. We will take trained discriminator and optimize generator and use first-order optimizer as follows:

$$\min_{\theta} V(G_\theta, D_{W_{final}}) \tag{5}$$

General expectation from above equation is to provide a local saddle point for the optimization problem , if $G_{\theta_{final}}, D_{w_{final}}$ is Nash Equilibrium. Additionally image generated by generator are supposed to improve or preserve same quality. But in practical, we will not be able to achieve improved or the same quality.

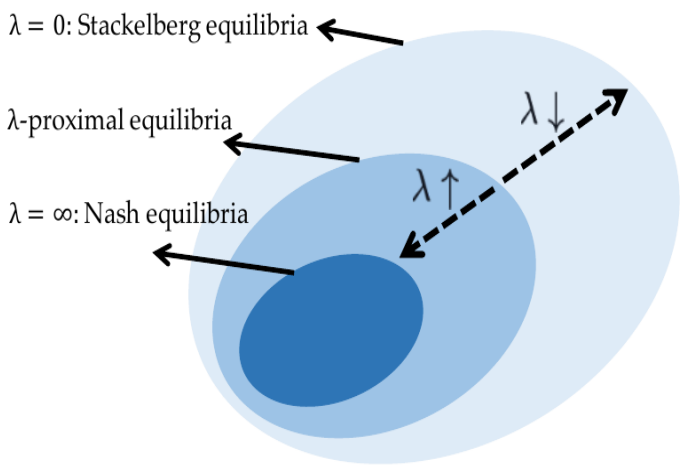
GANs were first introduced in [1], and proposed vanilla GAN whose minimax problem can be formulated as:

$$\min_{G \epsilon \mathcal{G}} \max_{D \epsilon \mathcal{D}} E[\log(D(X))] + E[\log(1 - D(G(Z)))] \tag{6}$$

In the 6 G, and D represent Generator and Discriminator function. Its work is extended in [10] to a general f-divergence. And the corresponding f-GAN problem is as follows:

$$\min_{G \epsilon \mathcal{G}} \max_{D \epsilon \mathcal{D}} E[D(X)] - E[f^* D(G(Z))]] \tag{7}$$

Figure 2: Proximal Equilibrium




Another study to help in stability issues is done in [9] which minimizes optimal transport cost to solve GAN problem.That problem can be formulated as:

$$\min_{G \epsilon \mathcal{G}} \max_{D1-Lipschitz} E[D(X)] - E[D(G(Z))]] \tag{8}$$

This paper proposes for WGAN problem if we can solve 8 for all x and $x^{'}$, than WGAN problem can be generalized as follows:

$$\min_{G \epsilon \mathcal{G}} \max_{Dc-concave} E[D(X)] - E[D^{c}(G(Z))] \tag{9}$$

In the 9 , c-transform is:

$$D^{c}(x) = sup_{x'} D(x^{'}) - x(x, x^{'}) \tag{10}$$

# 3 Proximal Equilibrium

We propose a new concept of equilibrium, proximal equilibrium, that allows us to investigate the spectrum of Nash and Stackelberg equilibria as can be seen from following figure.

Proximal Equilibrium can be formulated as follows:

$$V_{\lambda}^{prox}(G, D) := \max_{\tilde{D} \epsilon \mathcal{D}} V(G, \tilde{D}) - \lambda ||\tilde{D} - D||^{2} \tag{11}$$

In the proximal optimization, we penalize the distance between the two functions to maintain the D function variable close to D and distance is measured according to norm function in discriminator. [2] proposes some propositions to consider Nash Equilibria of V from 11 as proximal equilibria of V.

**Proposition 1**

It proposes that for every $G\epsilon\mathcal{G}$ and $D\epsilon\mathcal{D}$, if $(G^*, D^*)$ is a Nash Equilibrium for $V_\lambda^{prox}(G, D)$, it is a $\lambda$-proximal equilibrium for V(G, D). It can be formulated as follows:

$$V_\lambda^{prox}(G^*, D) \leq V_\lambda^{prox}(G^*, D^*) \leq V_\lambda^{prox}(G, D^*) \tag{12}$$

**Proposition 2**

$$V(G^*, D) \leq V(G^*, D^*) \leq \max_{\tilde{D}\epsilon D} V(G, \tilde{D}) - \lambda||\tilde{D} - D^*||^2. \tag{13}$$

If for every G $\epsilon\mathcal{G}$ and $D\epsilon\mathcal{D}$, 13 satisfies, than $(G^*, D^*)$ is a $\lambda$-proximal equilibrium.

**Proposition 3**

Consider for the V(G, D), we have a set of $\lambda$-proximal equilibria namely $PE_\lambda(V)$. It proposes if $\lambda_1 \leq \lambda_2$ satisfies, then it will also satisfy following condition:

$$PE_{\lambda 2}(V) \subseteq PE_{\lambda 1}(V) \tag{14}$$

**Proposition 4**

According to the this proposition, using a first-order optimization approach, one may efficiently compute the best solution to the proximal maximization under the given assumptions.

[2] proposes that generator and discriminator in 11 have parameters $\theta, w$ and taking it as maximization problem and assumes following consitions:

First is,

$$||\nabla_w||D_w - D^2 - \nabla_w||D_w^{'} - D||^2||_2 \geq \eta_1||w - w^{'}||_2 \tag{15}$$

In 15 for any $w, w^{'}, D$, discriminator norm like $||.||, ||D_w - d||^2$ is $\eta$-1 string convex in w for the function D. and other assumption is that GAN minimax objective which can be defined by $V(G_\theta, D_w)$ is $\eta_2 - smooth$ in w, which can be formulated as follows:

$$||\nabla_w V(G_\theta, D_w) - \nabla_w V(G_\theta, D_w)||_2 \leq \eta_2||w - w^{'}||_2 \tag{16}$$

It proposes that under above assumptions if $\eta_2 \geq \lambda\eta_1$, then the maximization objective in equation of proximal objective i.e. 11 will be $\lambda\eta_1 - \eta_2$ is concave and hence it has a unique solution, $w^*$. Additionally it shows that as the value of $\lambda$ will increase, complexity for solving proximal optimization will reduce.

# 4 Literature Review

There are various research done in game theory, like [11] discuss convergence of learning dynamics in Stackelberg games. It discusses Nash and Stackelberg equilibrium concepts and worked on a gradient-based update for the leader, while the follower uses a best response approach in zero-sum games, ensuring that each stable critical point is a Stackelberg equilibrium. This paper also proposes learning dynamics to train GANs, and give good knowledge about optimization in GANs.

Moreover, in [12] WGAN-GP is trained with competitive gradient descent(CGD) can improve Inception Score (IS) without any explicit regularization. It discuss opponent-aware generator and discriminator modeling, such as that used in competitive gradient descent (CGD), may dramatically improve ICR and hence stabilize GAN training without the need for explicit regularization. This paper have identified a fundamental weakness in the static minimax method to comprehending GANs. Moreover this paper used game-theoretic interpretation of ICR to identify algorithms such as CGD that can lead to stronger ICR.

Figure 3: MNIST Dataset

## 5 Simulation Results

I have trained Wasserstein GANs, to get the results which can be helpful to compare with the results of proximal training and other trainings. I have taken MNIST database which contains 28x28 pixels images of handwritten digits. Following figure contains the images of MNIST dataset.

The generator and discriminator are trained with the following hyperparameters:

- Number of training epochs - 100
- Size of the batch - 64
- learning rate - 0.00005
- Number of CPU threads to use during batch generation - 8
- Dimensionality of the latent space - 100
- size of each image - 28x28
- Number of image channels - 1
- Number of training steps for discriminator per iteration - 5
- Lower and upper clipping value for the discriminator weights - 0.01
- Interval between image samples - 400

I have defined a function to generate layers for the generator neural networks. It will return linear layers with activation function, Leaky rectified Linear Unit, or Leaky ReLU. LeakyReLU is based on ReLU but for negative values it reduces slowly instead of giving 0.

```
def block(in_feat, out_feat, normalize=True):
    layers = [nn.Linear(in_feat, out_feat)]
    if normalize:
        layers.append(nn.BatchNorm1d(out_feat, 0.8))
    layers.append(nn.LeakyReLU(0.2, inplace=True))
    return layers
```

Generator model has 3 hidden layers and one input and one output layer in sequential layer of torch.nn module.

```
self.model = nn.Sequential(
    *block(cfg.latent_dim, 128, normalize=False),
    *block(128, 256),
    *block(256, 512),
    *block(512, 1024),
    nn.Linear(1024, int(np.prod(img_shape))),
    nn.Tanh()
)
```

While, on the other hand discriminator model has 1 hidden layers and one input and one output layer. Following figure has the discriminator model.

```
def __init__(self):
    super(Disc, self).__init__()

    self.model = nn.Sequential(
        nn.Linear(int(np.prod(img_shape)), 512),
        nn.LeakyReLU(0.2, inplace=True),
        nn.Linear(512, 256),
        nn.LeakyReLU(0.2, inplace=True),
        nn.Linear(256, 1),
    )
```

I have used RMSProp optimizer for both the models, Generator ans Discriminator. I have used "optim" library of PyTorch to train the models.

```
# optimizers
optimizer_G = torch.optim.RMSprop(
    gen.parameters(), lr=cfg.lr
)
optimizer_D = torch.optim.RMSprop(
    disc.parameters(), lr=cfg.lr
)
```

I have taken adversarial loss for both generator and discriminator.

```
d_loss = (
    - torch.mean(disc(real_imgs))
    + torch.mean(disc(fake_imgs))
)
```

```
g_loss = - torch.mean(disc(gen_imgs))
```

After extensive training of the models, I have generated loss of discriminator and generator at each steps. As can be seen in the figure below, while both the discriminator and generator loss decreases in first few each step, in the last few epochs, loss of discriminator and generator reach the local minima.

Figure 4: Results

```
Epoch 0/100 | Batch 0/938 | D loss: 0.046213 | G loss: -0.009124
Epoch 0/100 | Batch 5/938 | D loss: -0.102064 | G loss: -0.010120
Epoch 0/100 | Batch 10/938 | D loss: -0.346080 | G loss: -0.014046
Epoch 0/100 | Batch 15/938 | D loss: -0.738695 | G loss: -0.026156
Epoch 0/100 | Batch 20/938 | D loss: -1.214182 | G loss: -0.051584
Epoch 0/100 | Batch 25/938 | D loss: -1.804343 | G loss: -0.093570
Epoch 0/100 | Batch 30/938 | D loss: -2.345008 | G loss: -0.147479
Epoch 0/100 | Batch 35/938 | D loss: -2.981106 | G loss: -0.226094
Epoch 0/100 | Batch 40/938 | D loss: -3.609321 | G loss: -0.321787
Epoch 0/100 | Batch 45/938 | D loss: -4.161057 | G loss: -0.453844
Epoch 0/100 | Batch 50/938 | D loss: -4.779850 | G loss: -0.583230
Epoch 0/100 | Batch 55/938 | D loss: -5.293278 | G loss: -0.746233
Epoch 0/100 | Batch 60/938 | D loss: -5.812374 | G loss: -0.967649
Epoch 0/100 | Batch 65/938 | D loss: -6.526595 | G loss: -1.123207
Epoch 0/100 | Batch 70/938 | D loss: -6.925036 | G loss: -1.415795
Epoch 0/100 | Batch 75/938 | D loss: -7.473508 | G loss: -1.668587
Epoch 0/100 | Batch 80/938 | D loss: -7.765137 | G loss: -1.970413
Epoch 0/100 | Batch 85/938 | D loss: -8.152660 | G loss: -2.297588
Epoch 0/100 | Batch 90/938 | D loss: -8.098282 | G loss: -2.681222
Epoch 0/100 | Batch 95/938 | D loss: -8.967533 | G loss: -3.033512
Epoch 0/100 | Batch 100/938 | D loss: -9.077085 | G loss: -3.302726
Epoch 0/100 | Batch 105/938 | D loss: -9.260449 | G loss: -3.744550
Epoch 0/100 | Batch 110/938 | D loss: -9.162416 | G loss: -4.229778
Epoch 0/100 | Batch 115/938 | D loss: -9.328634 | G loss: -4.895798
Epoch 0/100 | Batch 120/938 | D loss: -9.060573 | G loss: -5.800825
...
Epoch 99/100 | Batch 915/938 | D loss: -0.135681 | G loss: -0.353083
Epoch 99/100 | Batch 920/938 | D loss: -0.124274 | G loss: -0.372417
Epoch 99/100 | Batch 925/938 | D loss: -0.121565 | G loss: -0.351036
Epoch 99/100 | Batch 930/938 | D loss: -0.127001 | G loss: -0.322832
Epoch 99/100 | Batch 935/938 | D loss: -0.147979 | G loss: -0.318113
```

# 6 Conclusions

We have studied Wasserstein GANs, and WGAN-GP and from the results, The paper have concluded that proximal training is a good method to train GANs. It has been seen that quality of the generated images is also good. In the paper they have also observed that sometime Wasserstein and Lipschitz GANs does not always converge to local Nash Equilibrium but their solution can be Nash Equilibrium. The paper proposed proximal training method to train GANs in which Nash equilibrium does not exist. In the paper Inception scores of the generated samples with the CIFAR-10 dataset is used to compare the proximal and regular GAN training approaches. It also proves improves inception scores of proximal training are better than the ones obtained with the normal training.

While the paper has also proposed and shown in the results that proximal training has very good scores for WGAN-WC with dimension parameter of 128. This paper proves that proximal equilibria exist in GANs and proximal training proves effective, as trianing GAN is very computationally expensive task. Along with that, the paper also shown that many GAN problems exist with no Nash Equilibrium. In such cases Proximal Equilibrium can be used.

# 7 Acknowledgements

For this project, both the candidates, Shivam and Walstan have cotributed equally. My work included training wasserstein GAN models to get the result which can be compared with the results of proximal training and other trainings. While my teammate Walstan Baptista worked on WGAN-GP with MNIST and celebA dataset. I have studied propositions for proximal training and discussed deeply in this report. While we both worked equally on this project, we both have contributed whatever we had to work on this wonderful project. Most of the code is contribution of our joint efforts.

# References


[1] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In Advances in neural information processing systems, pp. 2672–2680, 2014.

[2] Farnia, Farzan, and Asuman Ozdaglar. "Do GANs always have Nash equilibria?." International Conference on Machine Learning. PMLR, 2020.

[3] Jin, C., Netrapalli, P., and Jordan, M. I. Minmax optimiza- tion: Stable limit points of gradient descent ascent are locally optimal. arXiv preprint arXiv:1902.00618, 2019.

[4] Arjovsky, M. and Bottou, L. Towards principled methods for training generative adversarial networks. arXiv preprint arXiv:1701.04862, 2017.

[5] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C. Improved training of wasserstein gans. In Advances in neural information processing systems, pp. 5767–5777, 2017.

[6] Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. Spec- tral normalization for generative adversarial networks. arXiv preprint arXiv:1802.05957, 2018.

[7] LeCun, Y. The mnist database of handwritten digits. http://yann. lecun. com/exdb/mnist/, 1998.

[8] Liu, Z., Luo, P., Wang, X., and Tang, X. Deep learning face attributes in the wild. In Proceedings of International Conference on Computer Vision (ICCV), December 2015

[9] Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein gan. arXiv preprint arXiv:1701.07875, 2017.

[10] Nowozin, S., Cseke, B., and Tomioka, R. f-gan: Training generative neural samplers using variational divergence minimization. In Advances in neural information process- ing systems, pp. 271–279, 2016.

[11] Fiez, T., Chasnov, B., and Ratliff, L. J. Convergence of learning dynamics in stackelberg games. arXiv preprint arXiv:1906.01217, 2019.

[12] Schäfer, F., Zheng, H., and Anandkumar, A. Implicit competitive regularization in gans. arXiv preprint arXiv:1910.05852, 2019.