Subject: Algorithms and Optimization for Big Data
Final Project Report

Topic: Regret Minimization in Non-Convex Games using Generative Adversarial Networks(GANs)

Team: Anushree Rankawat, Himol Shah, Meet Patel, Harsh Shah, Divya Dass

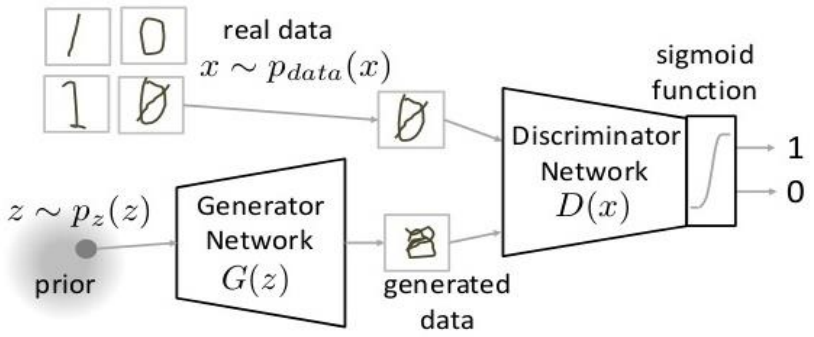
1. Problem Statement

Our project focuses on implementing regret minimization by considering window of past losses occurred using the Generative Adversarial Networks. We have used MNIST dataset for showing optimal results obtained by regret minimization. We would like to reach the equilibrium faster where Generator generates images that are indistinguishable to Adversary. In order to do so, we adopt the strategy of risk averse players and perform regret minimization in our generator.

Our work aims at enhancing the sharpness, accuracy and speed of these generator models. We have implemented the inclusion of Regret minimization in GANs in Tensorflow platform.

2. Introduction to Generative Adversarial Networks (GANs)

The main idea behind a Generative Adversarial Network is to have two competing neural network models. One takes noise as input and generates samples (and so is called the generator). The other model (called the discriminator) receives samples from both the generator and the training data, and has to be able to distinguish between the two sources. These two networks play a continuous game, where the generator is learning to produce more and more realistic samples, and the discriminator is learning to get better and better at distinguishing generated data from real data. These two networks are trained simultaneously, and the hope is that the competition will drive the generated samples to be indistinguishable from real data.



But, GANs face certain issues as well. While training a GAN, it essentially tries to find the Nash equilibrium of the game that is being played between the generator and the adversary. Gradient Descent can achieve this sometimes, but there is no guarantee of this strategy to work every single time. This raises the question that GANs might be more unstable than their other generator counterparts.

So, to handle this problem, we have tried to include regret minimization which achieves local equilibrium introduced by Hazan et al. [ref] The paper argues that the local equilibrium achieved is similar to the Nash equilibrium and can hence be used for solving our issue.

3. Regret Minimization

As per the approach proposed in the paper, we are introduced to an equilibrium state (local minima) which is achieved faster with the help of Regret minimization. By this state, we propose that our generator will be able to learn to produce images closer to real images faster.

We achieve regret minimization by introducing a window over which we take mean of our previous losses, and we then compute gradient over this loss. We update weights of generator by using window loss and supplying this loss over to Adam optimizer.

Regret quantifies the objective of prediction points with small gradients on average. In other words, regret is loss of the generator i.e., the difference between generated output and real input. Regret Minimization is to reduce the time taken by the generator to produce the images as similar as the real images.

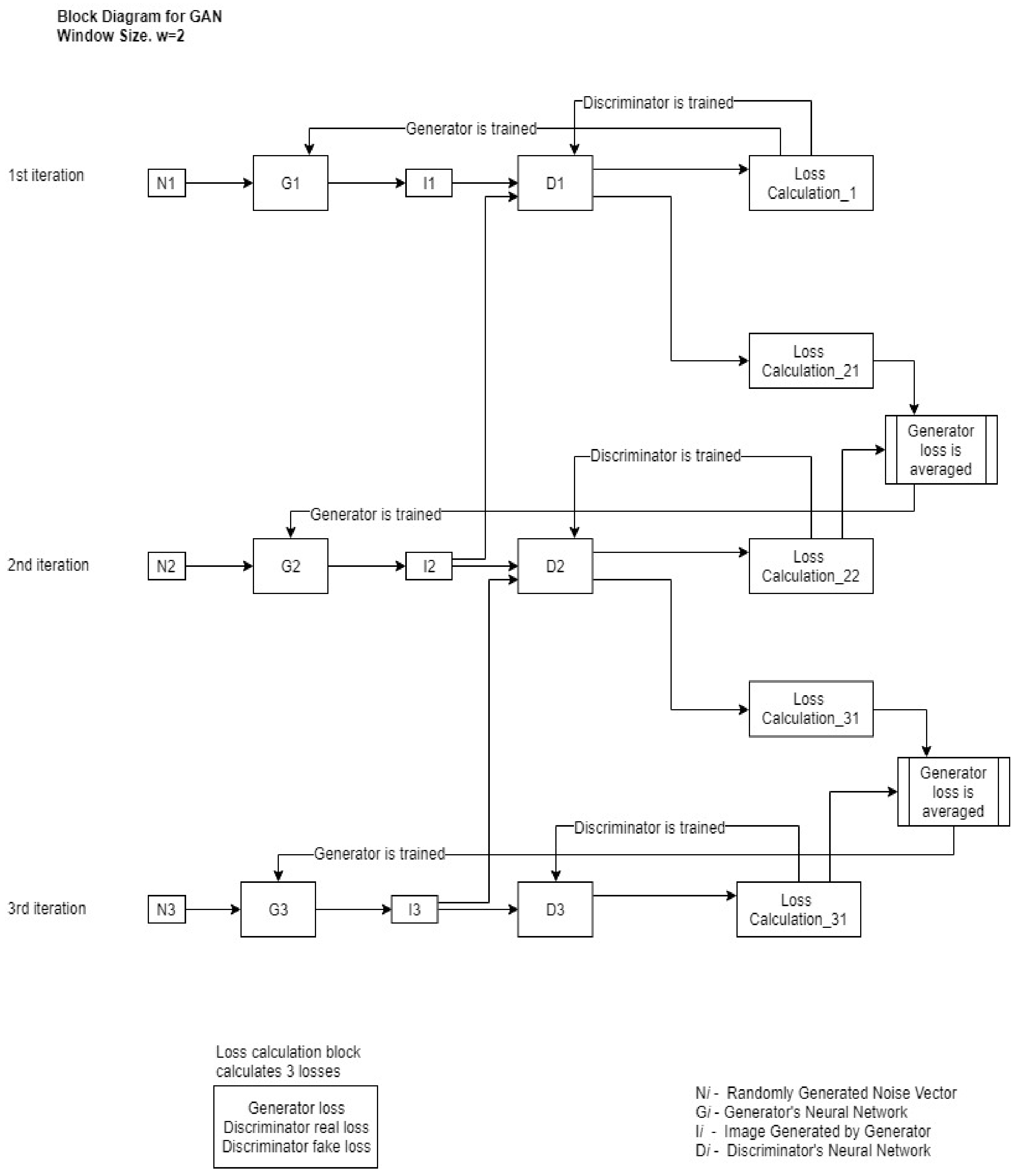
The generator weights are updated by using Adam's Optimizer that considers the computed regret along with the adaptive learning rates for all nodes and updates the generator weights in such a way that the generated output is as similar as the real inputs. This process is repeated for certain number of times to achieve the minimum possible generator loss. At the end when generator is totally trained, the probability of the discriminator to predict any frame as real or fake would settle at 0.5.

4. Inclusion of Regret Minimization in Vanilla GANs with MNIST dataset

Here, we introduced window into the loss calculation on simple vanilla GANs. We have updated weights by first finding average of losses over the window size, and then we optimized our weights by using this mean loss by using Adam optimizer.

Approach for including sliding window for regret minimization:

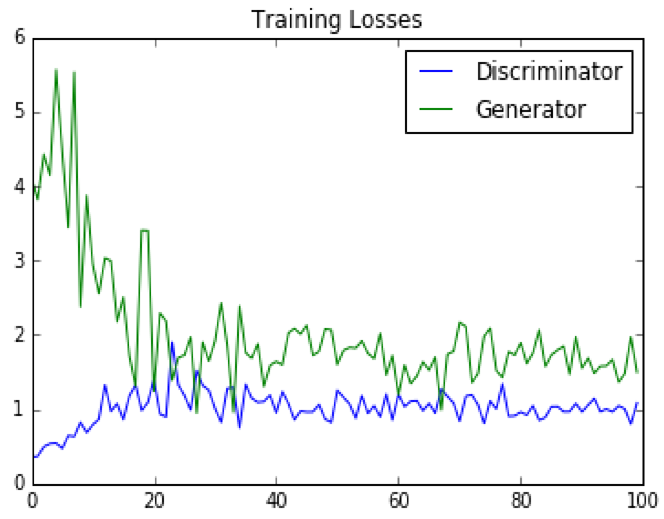
System Architecture:



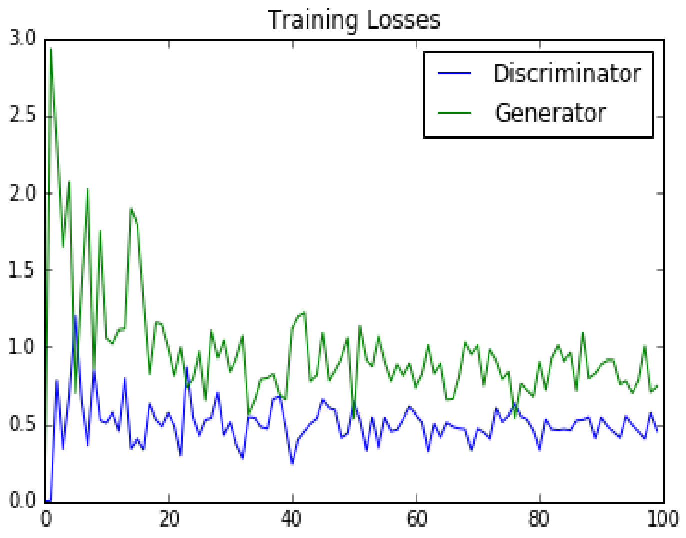
5. **Results of GANS with Regret Implementation with specific window size (w).**

   Loss Graph with MNIST dataset for 100 epochs:

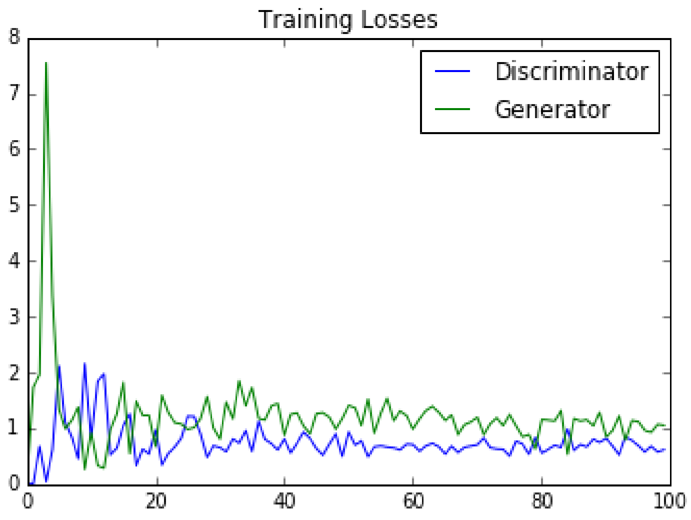
1. Loss for Generator and Discriminator in non-regret implementation (w = 1)



2. Loss for Generator and Discriminator with regret minimization (w = 2)



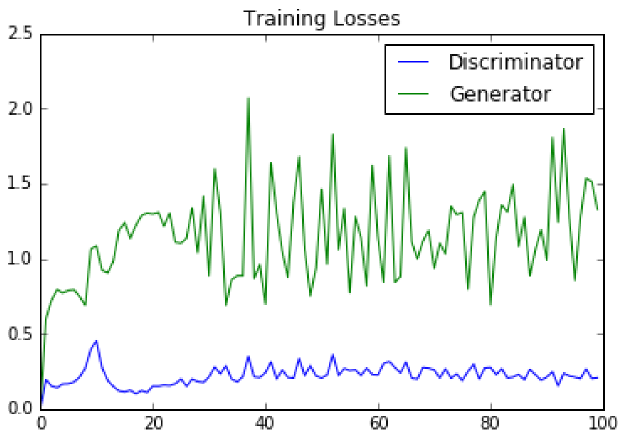
3. Loss for Generator and Discriminator with regret minimization (w = 3)

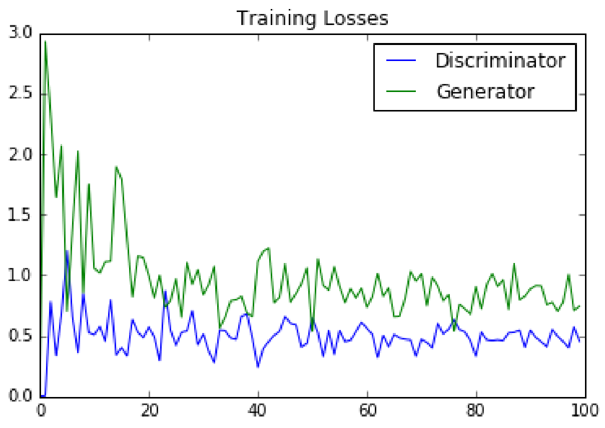


4. Gradient Descent Optimizer vs Adam Optimizer for window size = 2

Gradient Descent

Adam Optimizer





6. Conclusions

Our algorithm works well for finding Local Equilibrium (similar to Nash Equilibrium) for Generative Adversarial Networks with the inclusion of Regret Minimization. You can see the convergence rate increases as we increase window size from 1 to 2 and then from 2 to 3. It helps us reduce the time taken for convergence and the accuracy of the images generated drastically.

7. Reference

1. Hazan, Elad, Karan Singh, and Cyril Zhang. "Efficient Regret Minimization in Non-Convex Games." *arXiv preprint arXiv:1708.00075* (2017).