Generative capabilities of Gans

By Vivian Sedov

Final Year : Viva

The agenda

Project Overview
Aims and objective
Examples and information about the project
The theory
Pros and Cons
Ending

Project Overview

What is this project

- Core study on generative abilities of Gans on Facial Data (Anime).
- Principle to overcome fundamental problem with datasets and understanding how to avoid DS that lack of labels.
- A Comparision between different gans and their varients.

- Non-convergence
- Mode collapse
- Diminished gradient

Aims / Core Goals

Aims:

- Produced faces must be those of female and male
- Each face must have unique traits that are sufficiently distinct from the initital training batch
- The quality of the generated faces must be high
- Using Active Label Synthesis to create valid labels
- A comparision between different models and data

Objectives

Objectives :

- Compare between different Datasets | Initally (Cats Dogs) (Celeb Faces)
- Optimisation of current model and implementation of how different networks compare towards each other.
- Using the initial data gathered from the cats and dogs and celeb faces dataset to be the foundation for the animated face generator.
- Foundation for me to understand certain tools and libraries
  - Torch
  - Wand.ai
- Expand project scope for an understanding of label systhsis
- How pre data process on data is more important than the model it self.

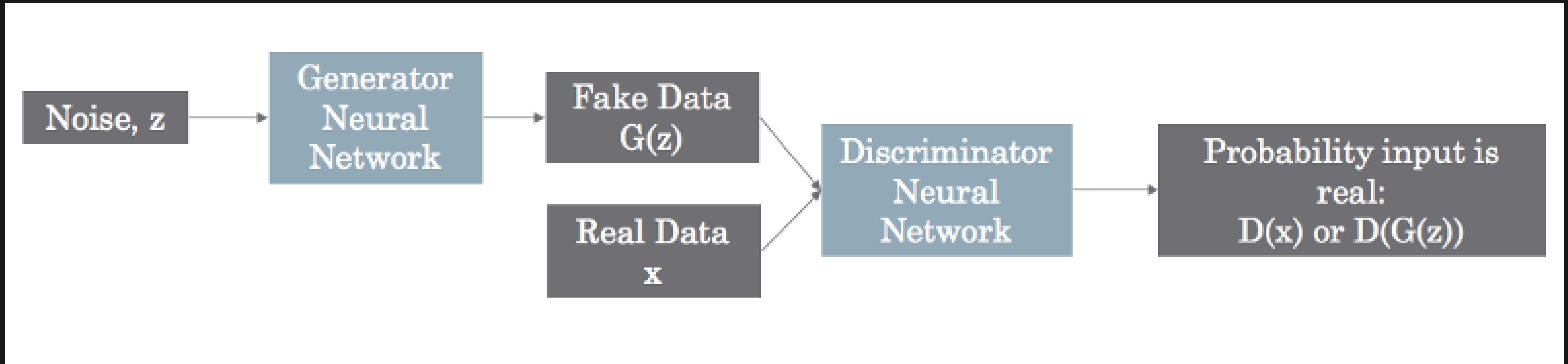
The Theory

What are Gans (Generative Adverserial Network), Main Idea ?

- 2 Networks, Generator, and Discriminator
- Two Player Adverserial Game
- Learns from the training set, Normal distribution, to generate new images, passed through 2 networks through backpropogation.

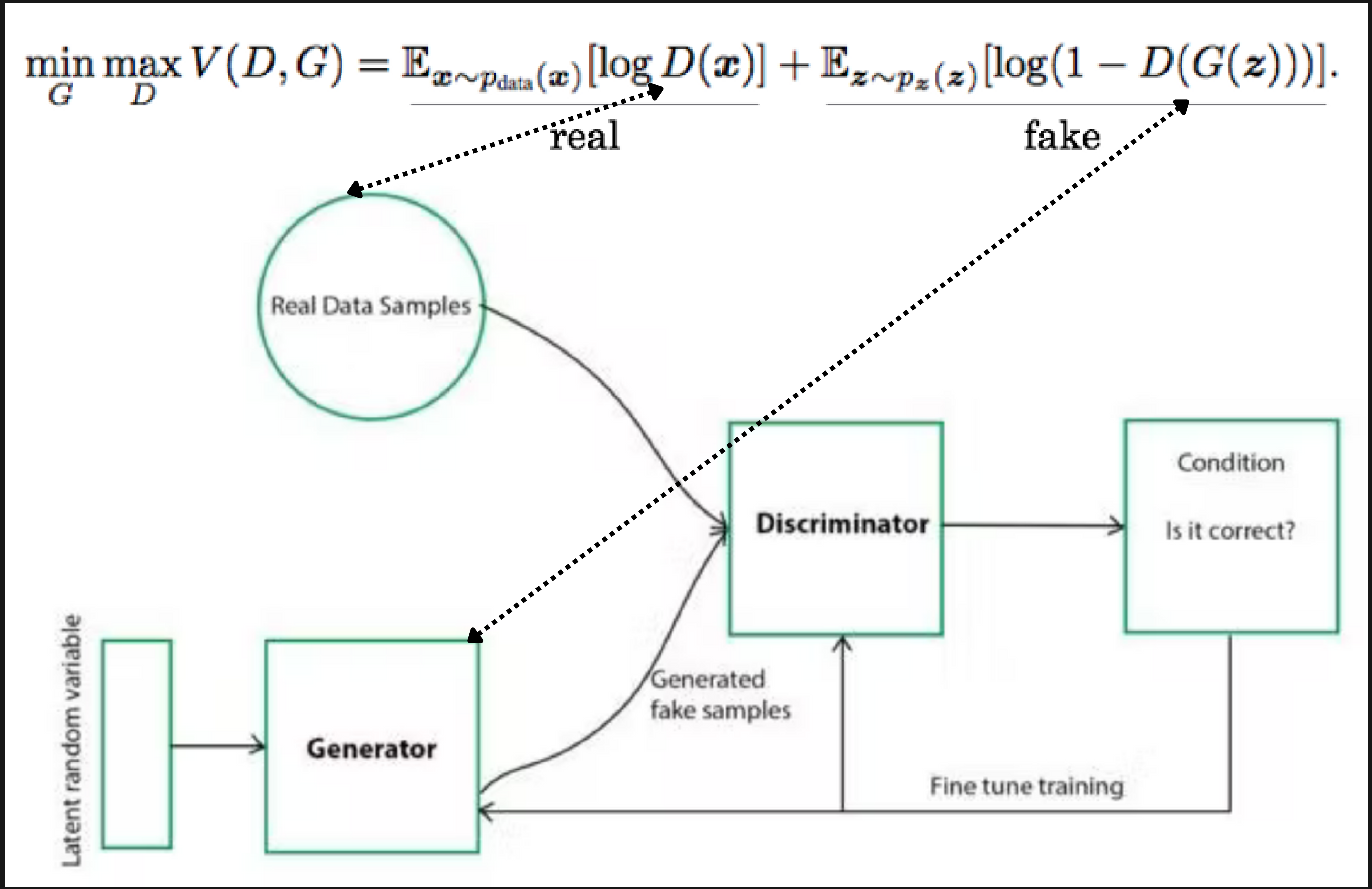
Discriminator: Learns to become better are identifying real from generated images.
Generator: Learns to generate better images to fool the discriminator

High Level View of a GAN



- For each mistake on a fake image, the discriminator gets punished and the generator gets a reward.
- The discriminator also gets punished or rewarded based on how well it sorts the real images.
- This is why they are called adversarial: the loss of the discriminator is the gain of the generator.
- Over time, the competition makes everyone better.

The Loss Function



- Real Samples are parsed to the D(x) in order to train the discriminator.
- Discriminator uses Binary cross entropy to determine whether a value is 0 or 1.
- Generator Does not need to know about the real data. Only the Back popogated data
- Initial Latent Variable can allow for better training.
- Using a scaler of a factor of 2 for the LR can provide fruitful results.

Algorithm Used:

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**
  for k steps **do**
    • Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \dots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
    • Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
    • Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

  end for
  • Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \dots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  • Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

end for
The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

- The Ascending Gradient is created to increase its understanding on the real and fake images.

- Vanishing Gradient problems can occour here when we deal with the discriminator : It also occurs when we train the generator

Diminished gradient

Instead of gradient descent with

$$\nabla_{\mathbf{w}_G} \frac{1}{n} \sum_{i=1}^{n} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right)$$

Do gradient ascent with

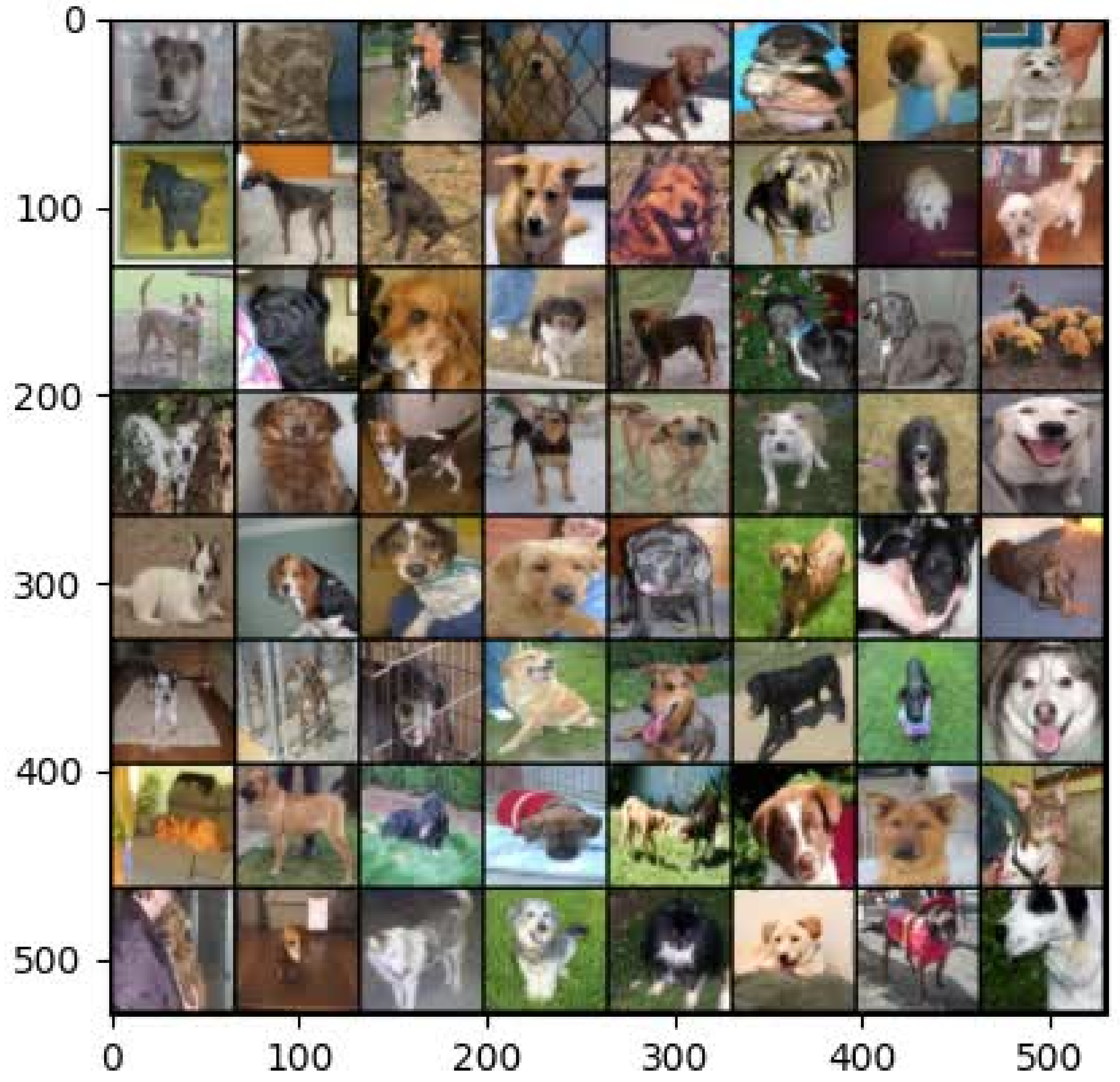
$$\nabla_{\mathbf{w}_G} \frac{1}{n} \sum_{i=1}^{n} \log\left(D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right)$$

So this occours when Discriminator is too strong and our gradient for the generator just vanishes as nothing gets parsed backwards.

Improvement Factors

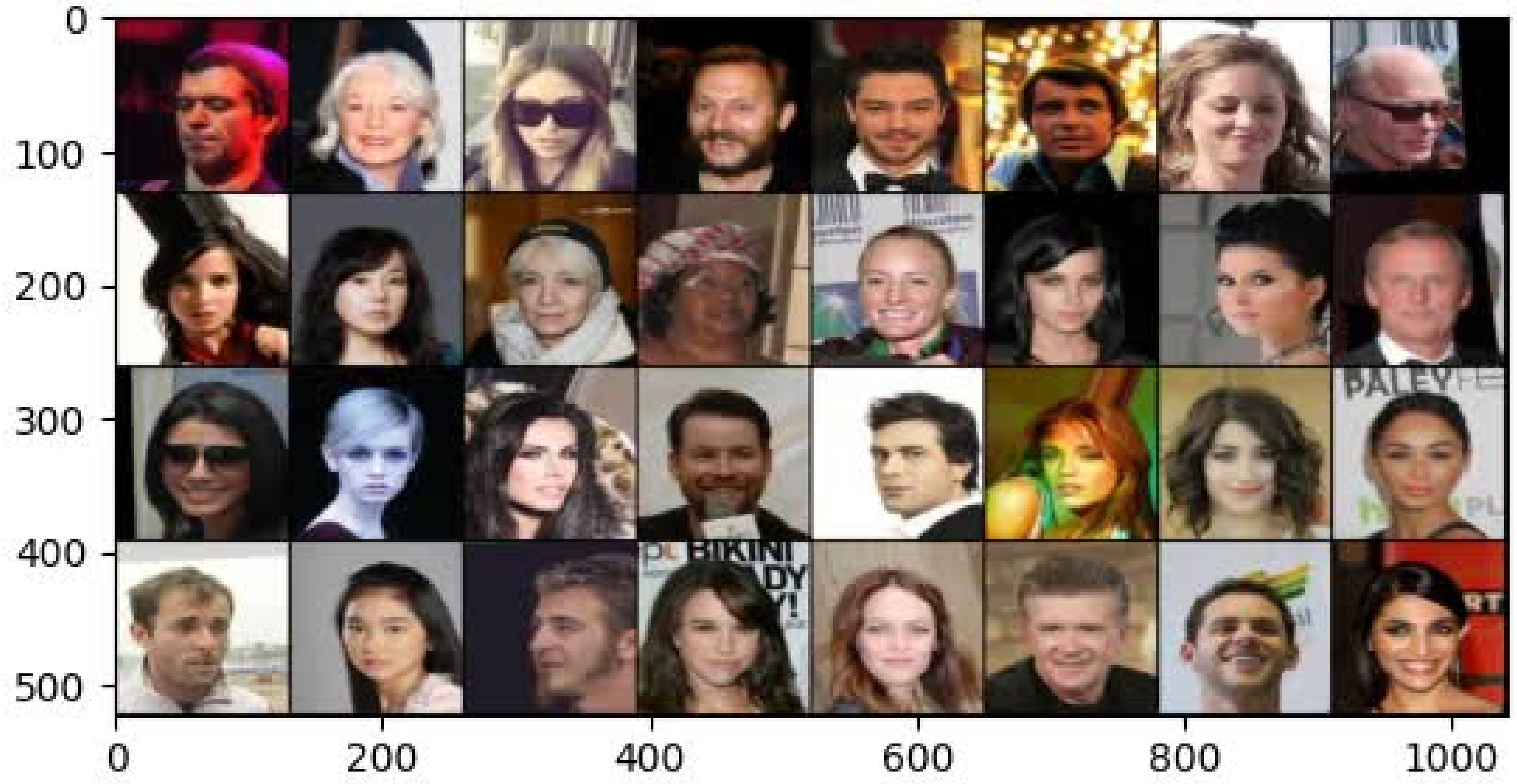
1. Using Adam optimiser [provides better results with a magnitude of 2 with respect to generator and discriminator over SGD.
2. Being able to track errors early: noticing vanishing gradient, tracking the D loss etc.
3. Using DCGan and different variations of networks [Convolutional networks provide better results with strides over].
4. Core issue of avoiding sparse gradients is to use Leaky ReLu.
    a. Leaky ReLu : G and D + for down sampling we use average pooling, and conv 2d with strides.
5. Batch normalistion : Different mini batches for real and fake: improves standard deviation for the D.

Mode collapse and Convergence issues

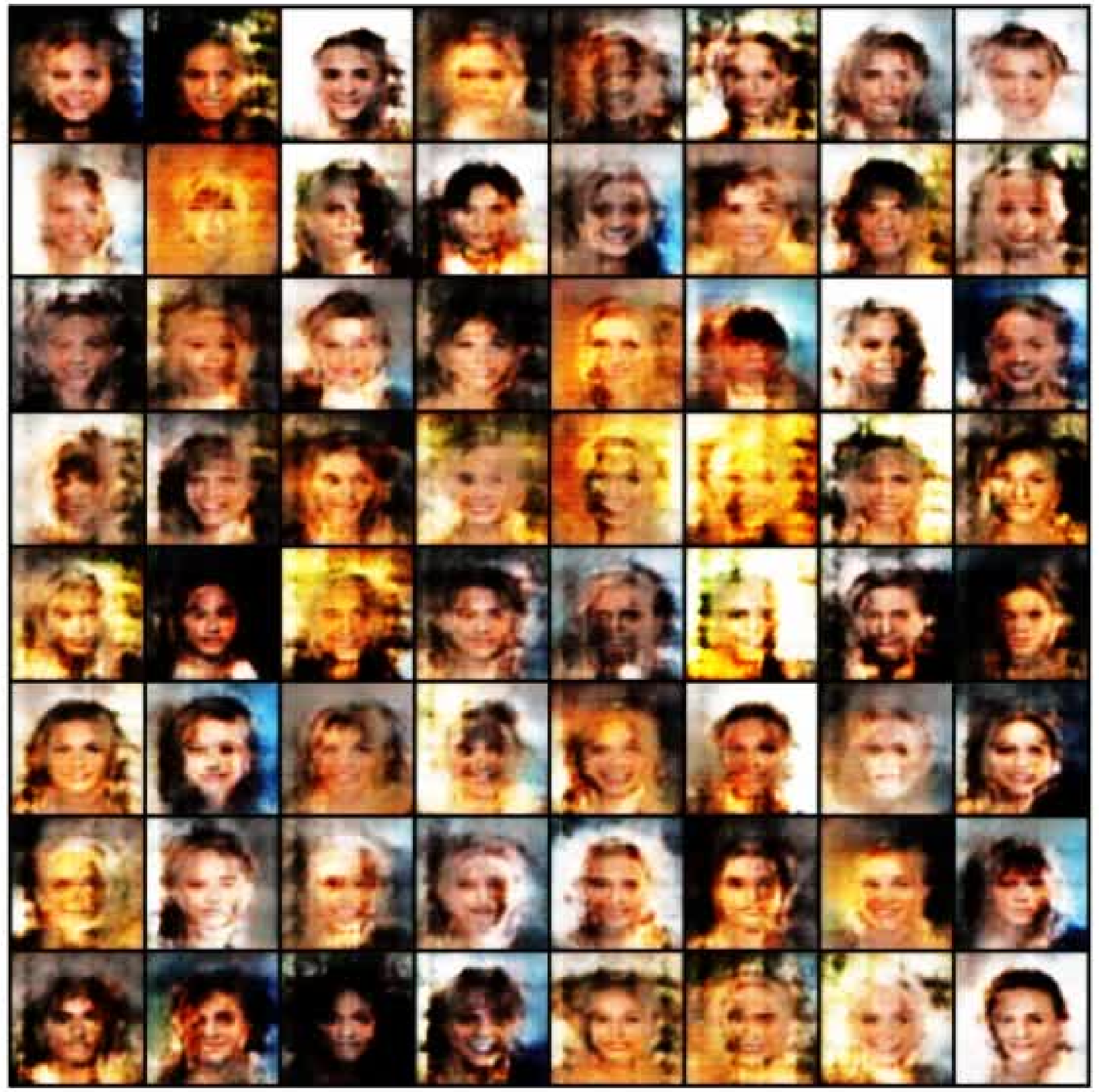

0
100
200
300
400
500
0
100
200
300
400
500


0
100
200
300
400
500
0
200
400
600
800
1000

Examples Poor Data


0
100
200
300
400
500
0
200
400
600
800
1000

Examples Decent Data



Notice something interesting ?

Here when I preproccesed the data, we have decent convergence, for a very simple Model.

This can be further processed using active label synthesis to hyper-tune an output by separating black hair, blonde hair, etc.

TimeLine

Start the pre proc for Animated Face data set

Create a NN for Active Label Synthesis

Create Generative Model

Look at other GAN Varients

Train Models and repeat for different Gans Varients

Create a report on findings

Thank you

Any Questions ?