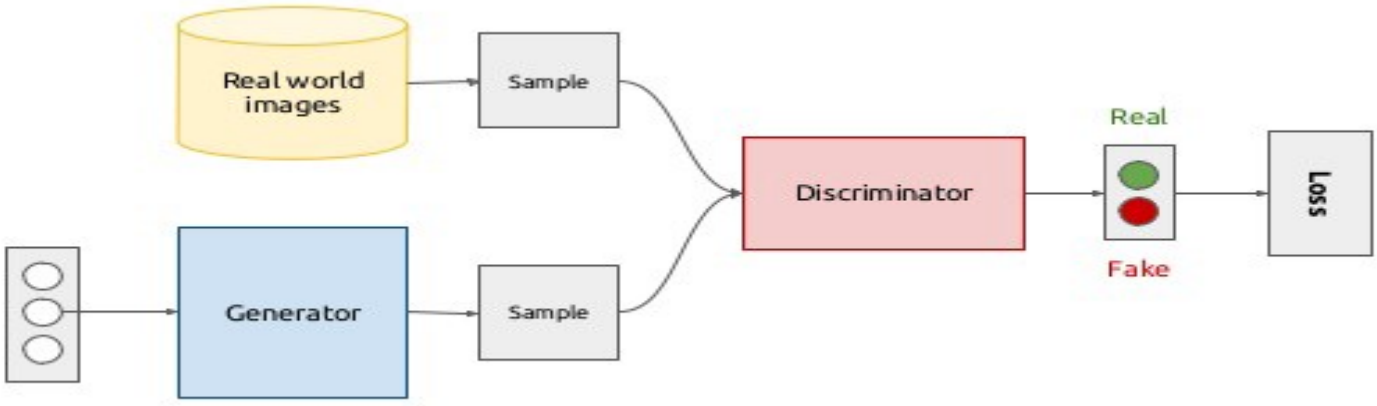
Objective

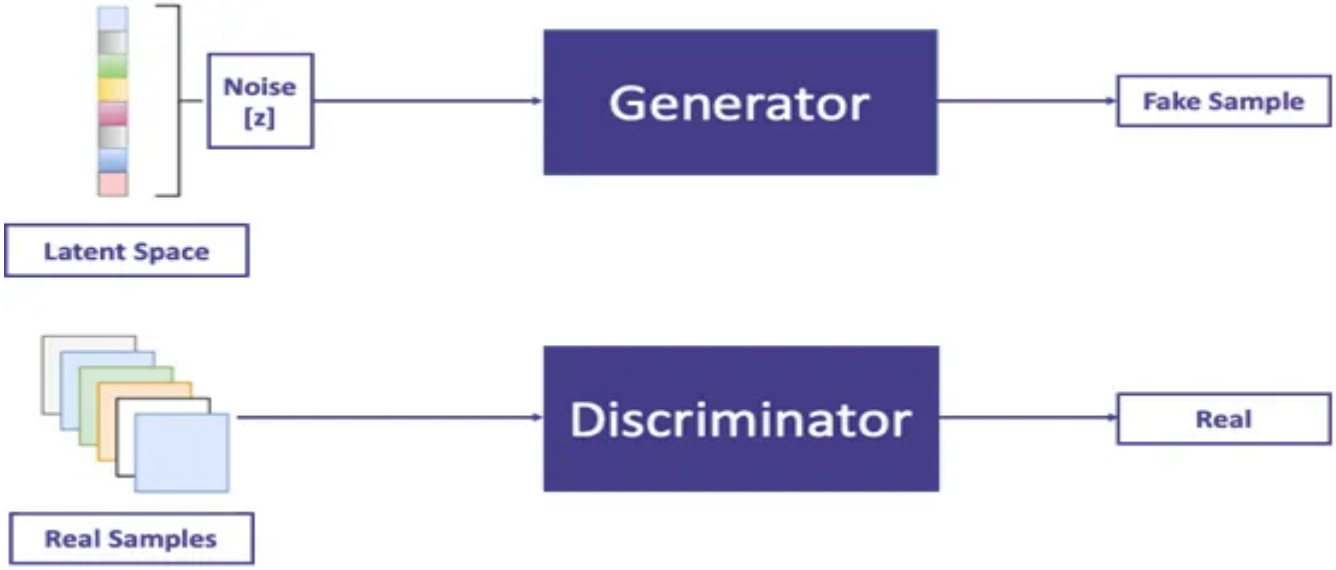
To train a generative adversarial network to generate images using the CIFAR10 dataset.

Basic Architecture for general Overview

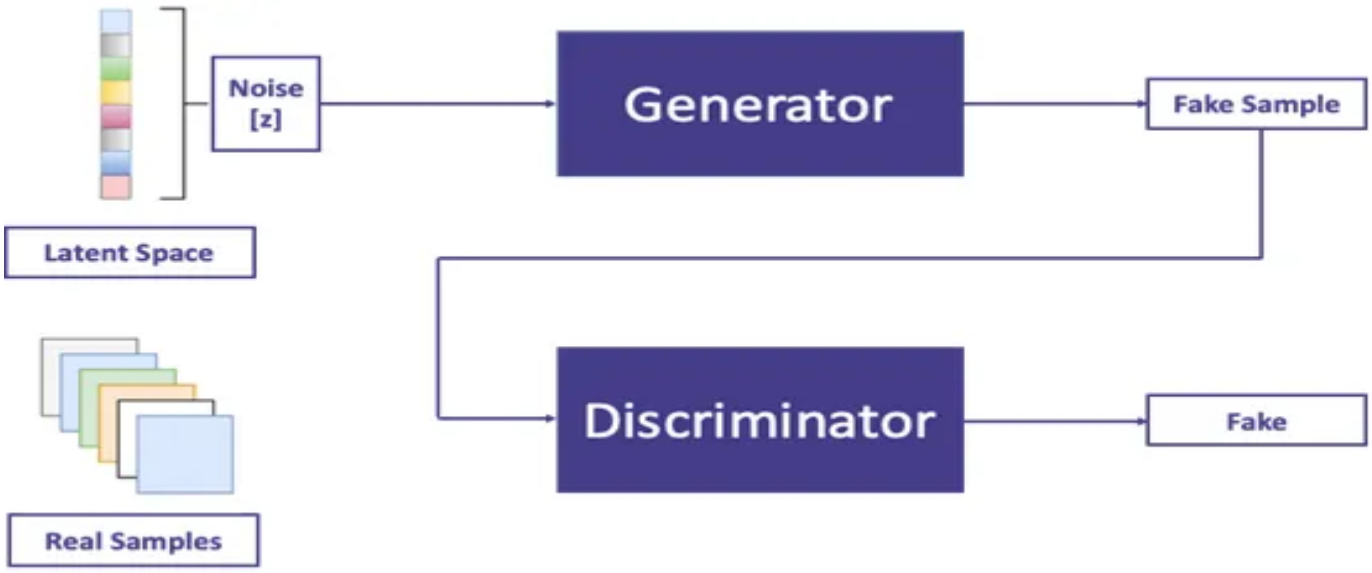
1. **During Training**



- For fitting real samples during training

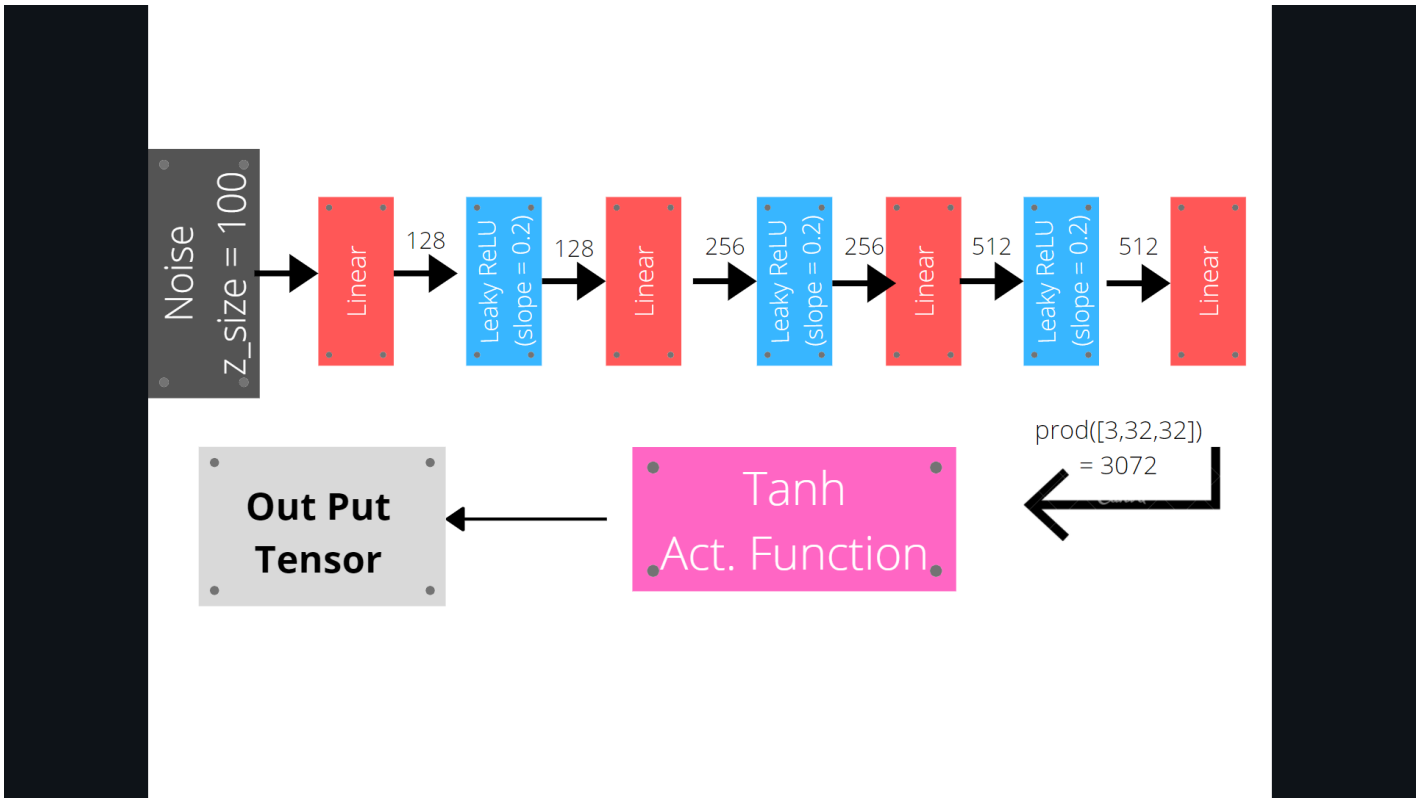


- For fitting fake samples during training

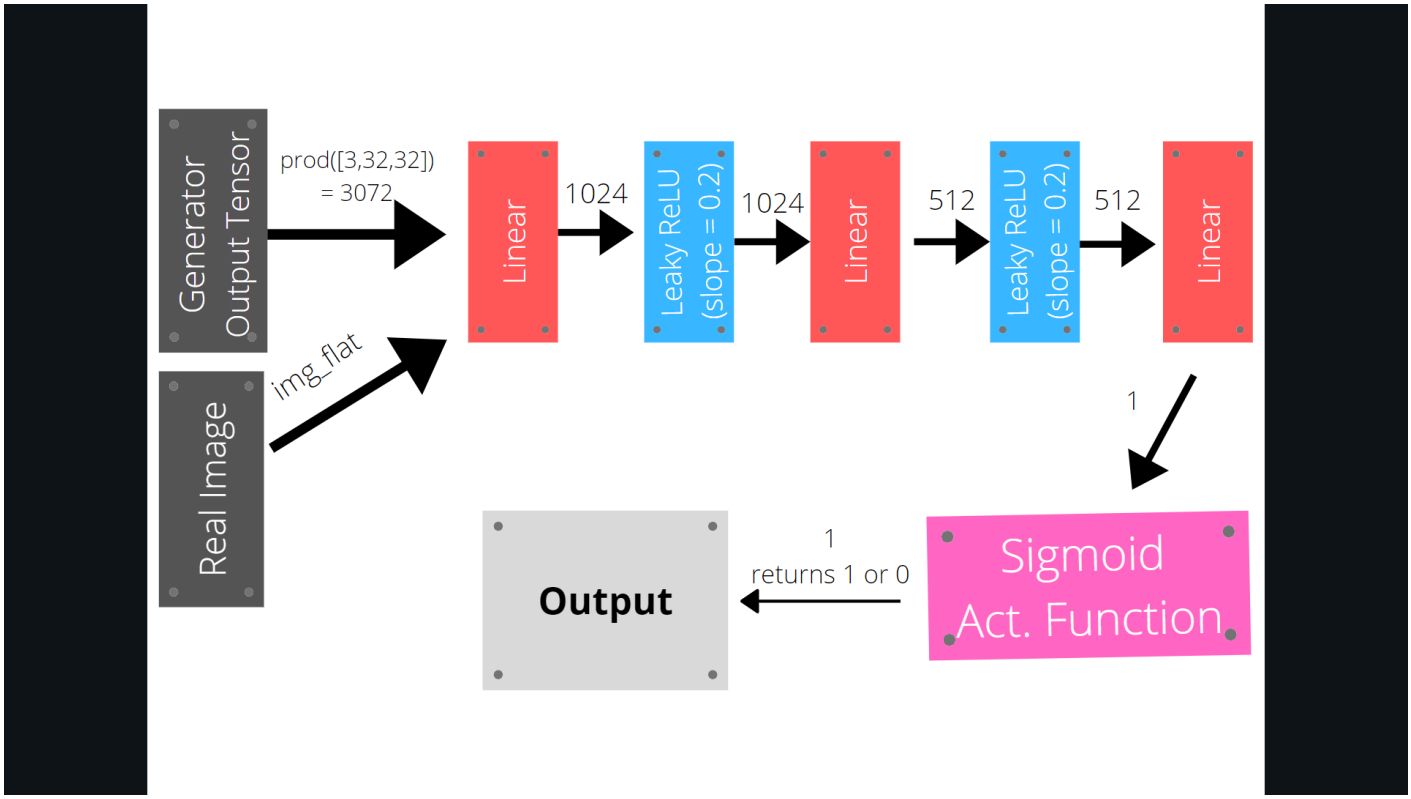


Architecture of Baseline Code

1. **Generator Model**



2. **Discriminator Model**



3. **Optimisers**
   - For Generator **SGD** optimiser is used
   - For Discriminator Model **RMSprop** is used
4. **Hyperparameters**
   - For updating the parameters **1e-4** is used as the initial learning rate.
5. **Batch Size**
   - In baseline code for training **256** was taken as batch size

Problem Faced

1. No clear image was formed for many epochs.
2. The model was very slow in terms of training i.e taking more epochs to form images.
3. It was taking large epochs to form an image having some indentation of some creature.
4. For each epoch it was taking more time to complete the same.
5. The pictures were not that effective in showing the clear different features of the same.

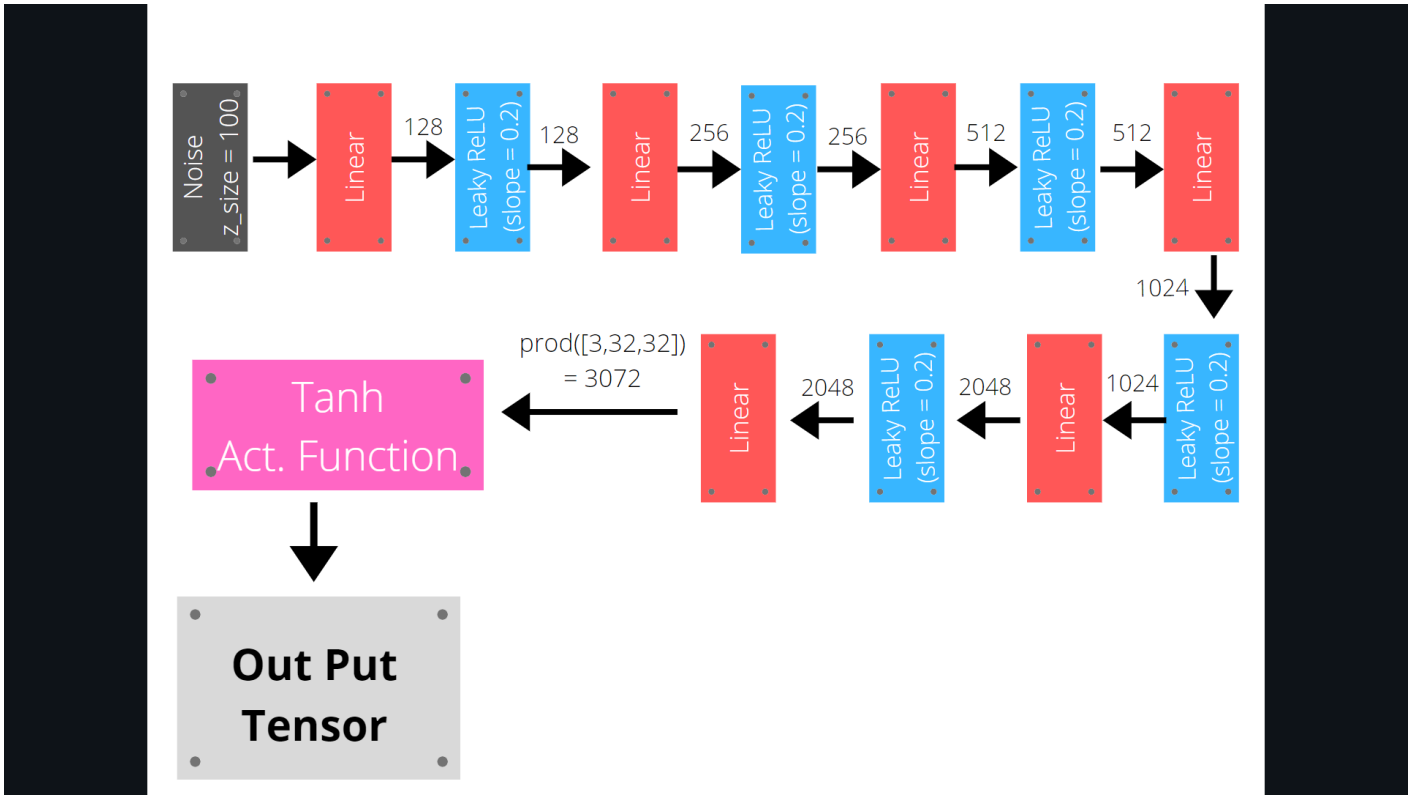
How you overcame them

I have increased the learning rate a bit more. For making GPU's speed more I decreased the batch size a bit so that the GPU can load the batch with more efficiency. So I fixed the same to 128. As the pictures came out were not that effective in showing the clear features of the object I increased the number of layers in both models.

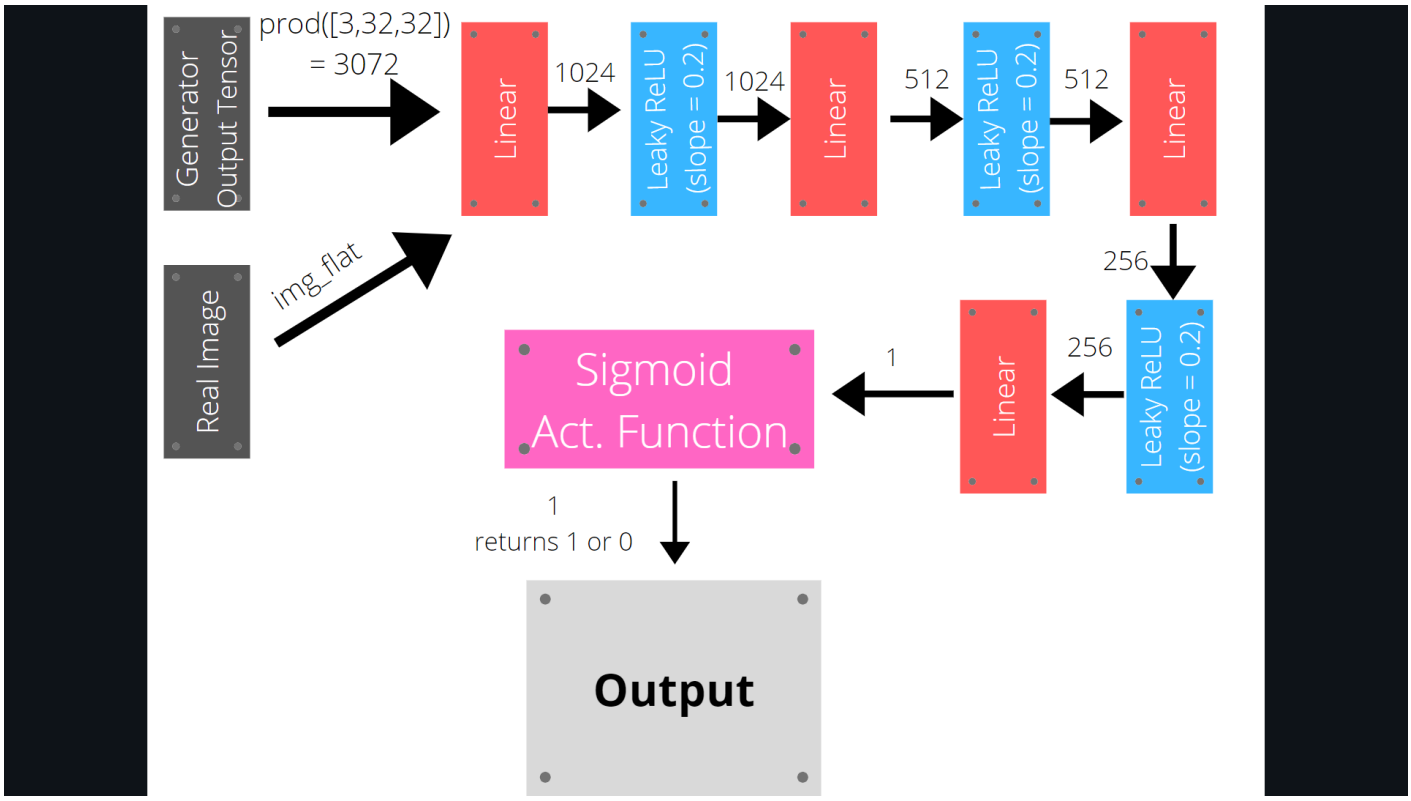
With experimenting with different optimizers finally Adam was the best one for both of my models. Hyperparameters like learning rate had a major impact during training the same, but after shifting from SGD to Adam the training was not that sensitive towards the learning rate. So finally I fixed the learning rate to 0.0002.

Architecture for Final GAN Code

1. Generator Model

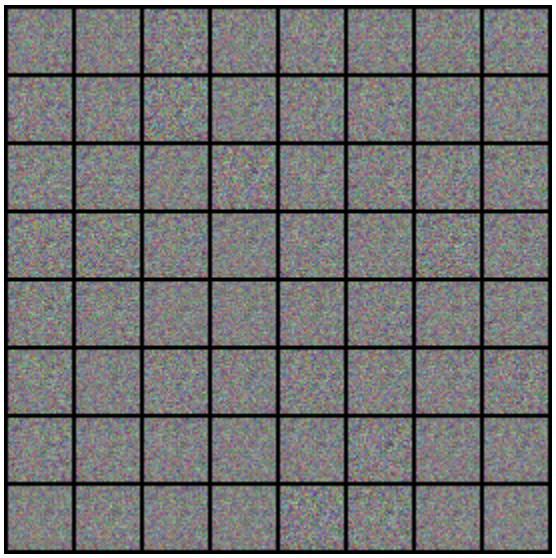
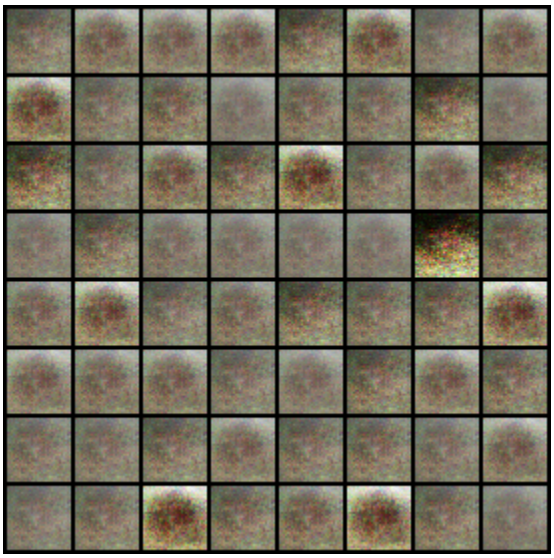
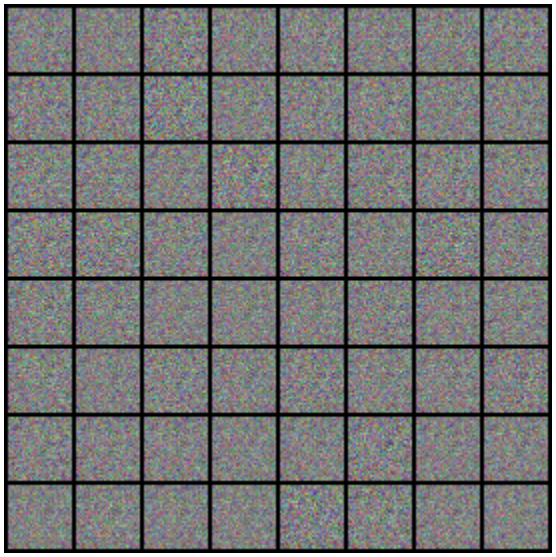
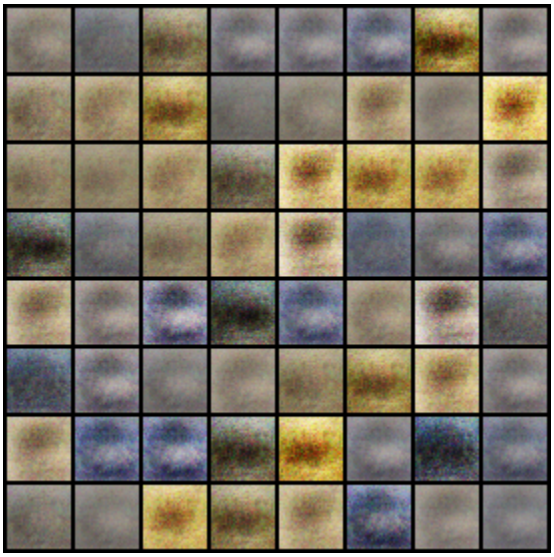


2. Discriminator Model



3. **Optimisers**
    - For Generator **Adam** optimiser is used
    - For Discriminator Model **Adam** is used
4. **Hyperparameters**
    - For updating the parameters **0.0002** is used as the learning rate.
5. **Batch Size**
    - In final GAN code for training **128** was taken as batch size

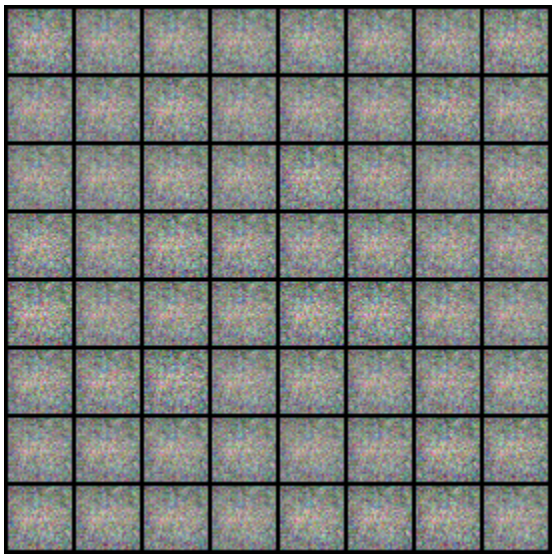
Comparison/Improvement

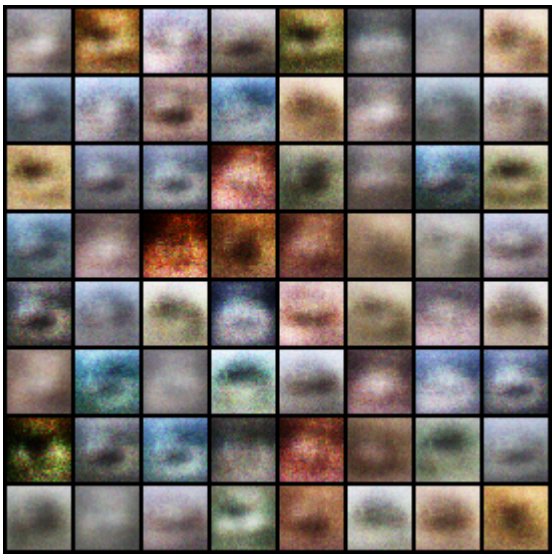
Baseline Model	Final Model
Epoch = 5	
	
Epoch = 10	
	

Baseline Model

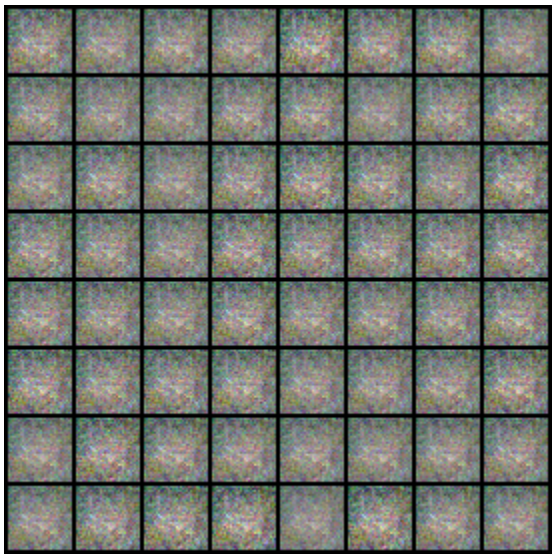
Final Model

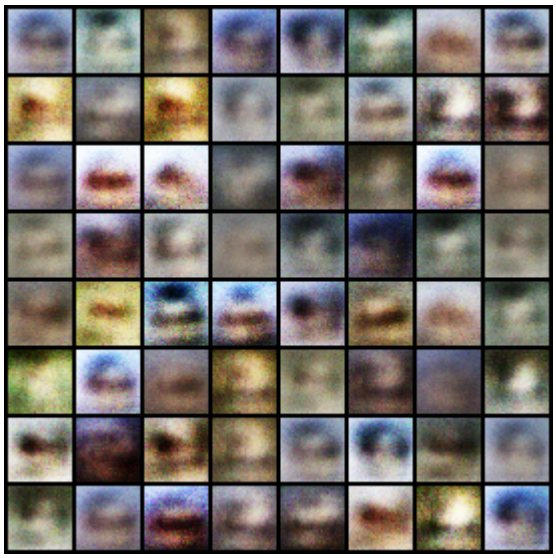
Epoch = 20



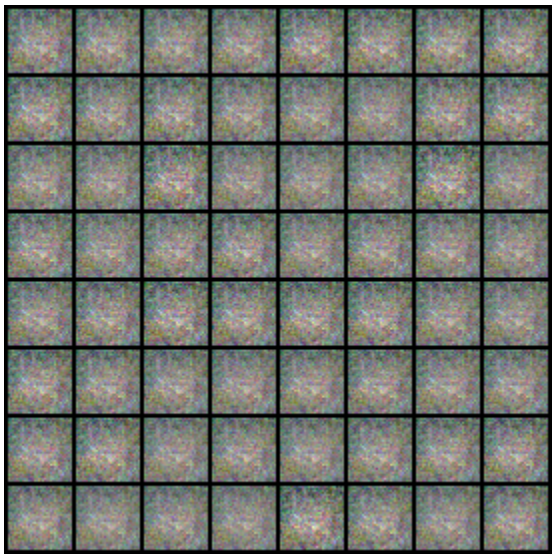


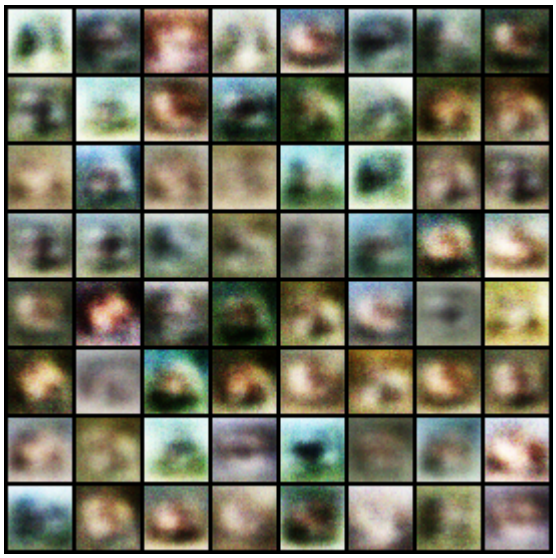
Epoch = 30





Epoch = 40

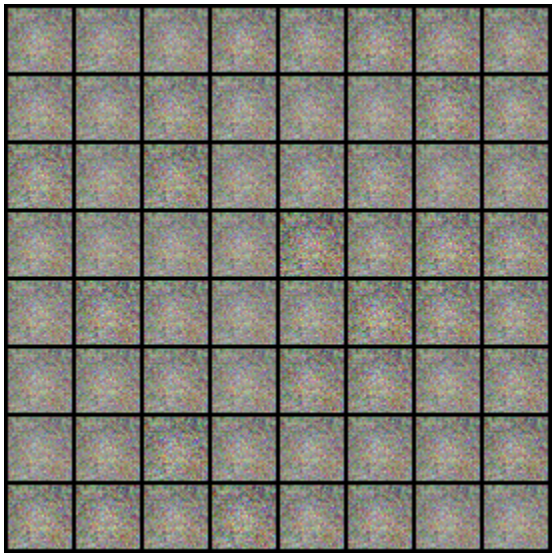


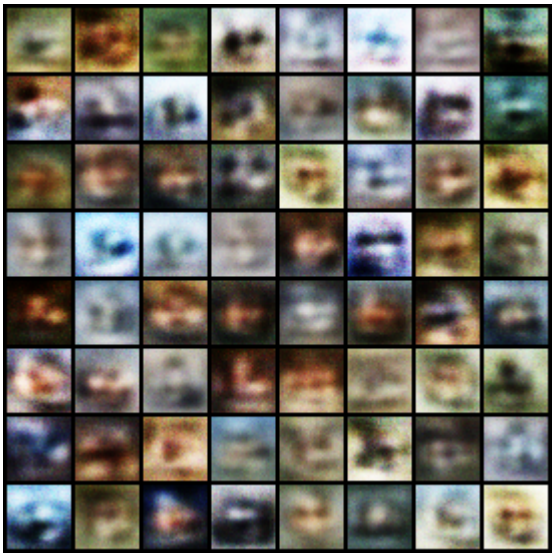


Baseline Model

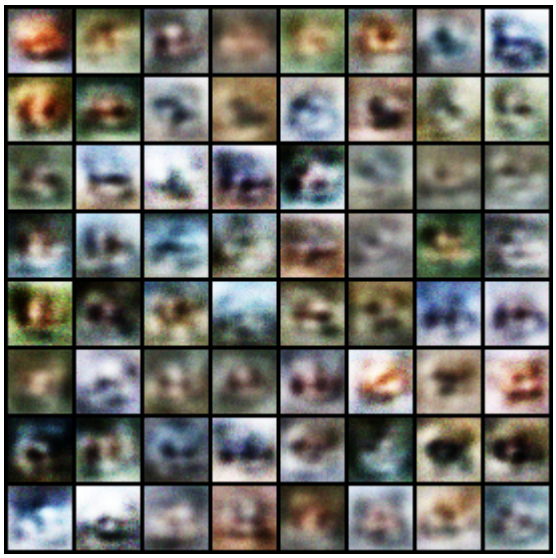
Final Model

Epoch = 80



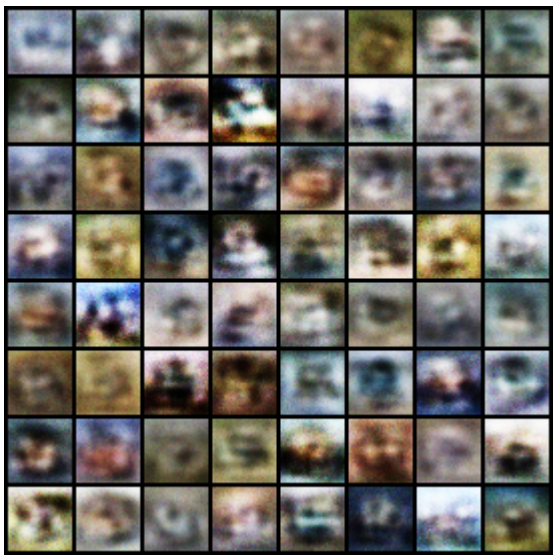


Epoch = 105



...

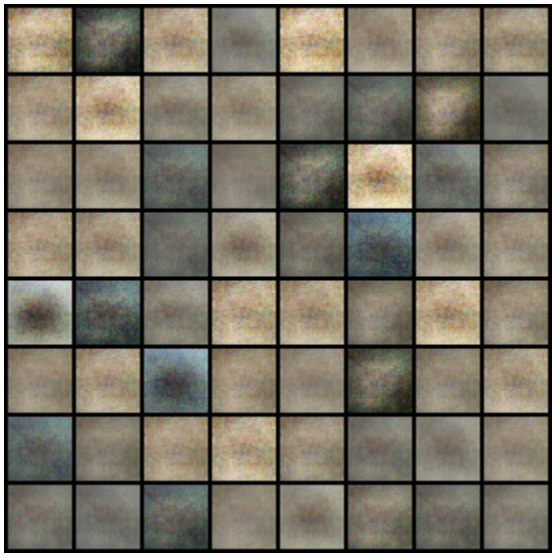
Epoch = 145



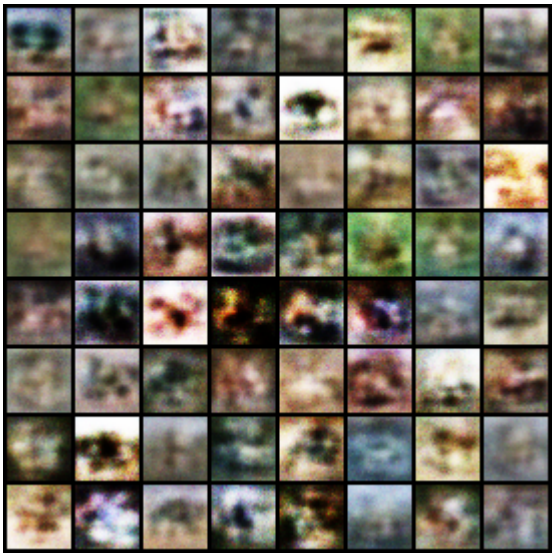
...

Baseline Model

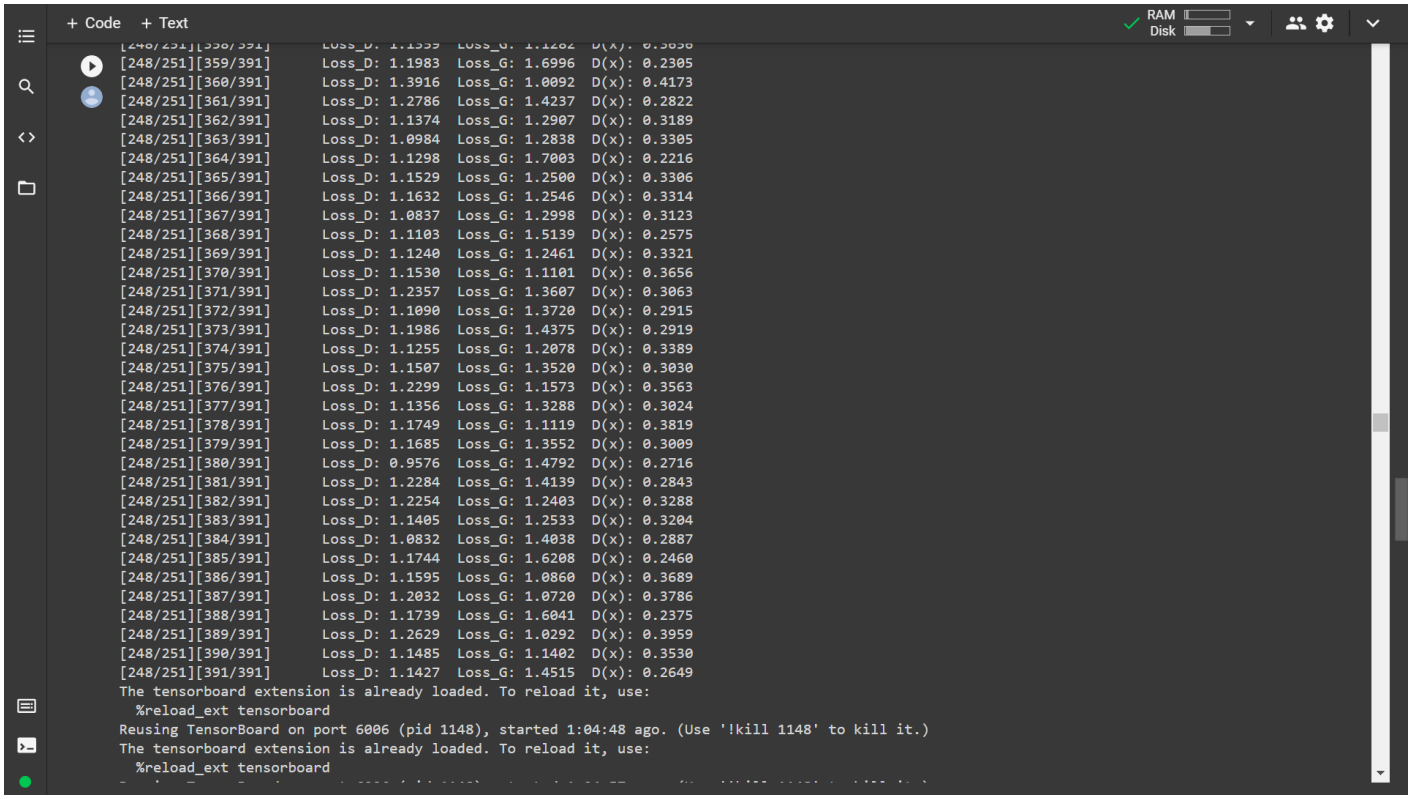
Epoch = 185



Final Model



Here is the loss values I got after 248 epochs



```
+ Code  + Text                                                    RAM  Disk
[248/251][359/391]      Loss_D: 1.1983  Loss_G: 1.6996  D(x): 0.2305
[248/251][360/391]      Loss_D: 1.3916  Loss_G: 1.0092  D(x): 0.4173
[248/251][361/391]      Loss_D: 1.2786  Loss_G: 1.4237  D(x): 0.2822
[248/251][362/391]      Loss_D: 1.1374  Loss_G: 1.2907  D(x): 0.3189
[248/251][363/391]      Loss_D: 1.0984  Loss_G: 1.2838  D(x): 0.3305
[248/251][364/391]      Loss_D: 1.1298  Loss_G: 1.7003  D(x): 0.2216
[248/251][365/391]      Loss_D: 1.1529  Loss_G: 1.2500  D(x): 0.3306
[248/251][366/391]      Loss_D: 1.1632  Loss_G: 1.2546  D(x): 0.3314
[248/251][367/391]      Loss_D: 1.0837  Loss_G: 1.2998  D(x): 0.3123
[248/251][368/391]      Loss_D: 1.1103  Loss_G: 1.5139  D(x): 0.2575
[248/251][369/391]      Loss_D: 1.1240  Loss_G: 1.2461  D(x): 0.3321
[248/251][370/391]      Loss_D: 1.1530  Loss_G: 1.1101  D(x): 0.3656
[248/251][371/391]      Loss_D: 1.2357  Loss_G: 1.3607  D(x): 0.3063
[248/251][372/391]      Loss_D: 1.1090  Loss_G: 1.3720  D(x): 0.2915
[248/251][373/391]      Loss_D: 1.1986  Loss_G: 1.4375  D(x): 0.2919
[248/251][374/391]      Loss_D: 1.1255  Loss_G: 1.2078  D(x): 0.3389
[248/251][375/391]      Loss_D: 1.1507  Loss_G: 1.3520  D(x): 0.3030
[248/251][376/391]      Loss_D: 1.2299  Loss_G: 1.1573  D(x): 0.3563
[248/251][377/391]      Loss_D: 1.1356  Loss_G: 1.3288  D(x): 0.3024
[248/251][378/391]      Loss_D: 1.1749  Loss_G: 1.1119  D(x): 0.3819
[248/251][379/391]      Loss_D: 1.1685  Loss_G: 1.3552  D(x): 0.3009
[248/251][380/391]      Loss_D: 0.9576  Loss_G: 1.4792  D(x): 0.2716
[248/251][381/391]      Loss_D: 1.2284  Loss_G: 1.4139  D(x): 0.2843
[248/251][382/391]      Loss_D: 1.2254  Loss_G: 1.2403  D(x): 0.3288
[248/251][383/391]      Loss_D: 1.1405  Loss_G: 1.2533  D(x): 0.3204
[248/251][384/391]      Loss_D: 1.0832  Loss_G: 1.4038  D(x): 0.2887
[248/251][385/391]      Loss_D: 1.1744  Loss_G: 1.6208  D(x): 0.2460
[248/251][386/391]      Loss_D: 1.1595  Loss_G: 1.0860  D(x): 0.3689
[248/251][387/391]      Loss_D: 1.2032  Loss_G: 1.0720  D(x): 0.3786
[248/251][388/391]      Loss_D: 1.1739  Loss_G: 1.6041  D(x): 0.2375
[248/251][389/391]      Loss_D: 1.2629  Loss_G: 1.0292  D(x): 0.3959
[248/251][390/391]      Loss_D: 1.1485  Loss_G: 1.1402  D(x): 0.3530
[248/251][391/391]      Loss_D: 1.1427  Loss_G: 1.4515  D(x): 0.2649
The tensorboard extension is already loaded. To reload it, use:
  %reload_ext tensorboard
Reusing TensorBoard on port 6006 (pid 1148), started 1:04:48 ago. (Use '!kill 1148' to kill it.)
The tensorboard extension is already loaded. To reload it, use:
  %reload_ext tensorboard
```

Insights

After about 200 epochs I have got about to clear images. The GAN algorithm really takes large epochs to be trained. Adjusting hyperparameter & choosing optimizer plays a key role in the efficiency of the model training.

Future Work

The model I made can be made more advanced. There are also GAN algorithms that generate clear images taking much less time in training. I have tried my level best to train my model. There are still some issues which can be checked in my model, So now I am working on that. The GAN models for generating human faces are more complicated, next I'll start learning & trying to make an algorithm for the same.

You can find my code for

- Baseline Model here
- Final Model here

Thank you

Omm Prakash Sahoo
Ops#0595
EEE