Generating images For Concept Design using Generative Adversarial Networks

Justin Huang
https://jvhuang1786.github.io
https://www.linkedin.com/in/justinvhuang

Problem Statement

Input of Old Assets created by studio

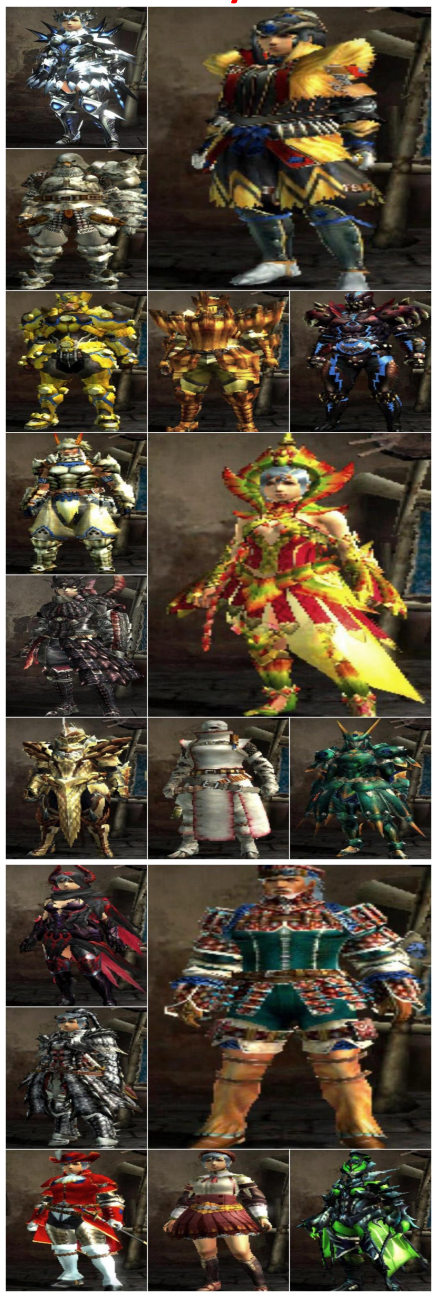
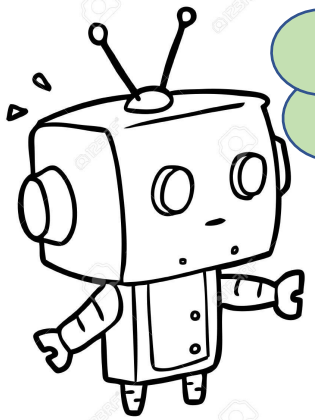
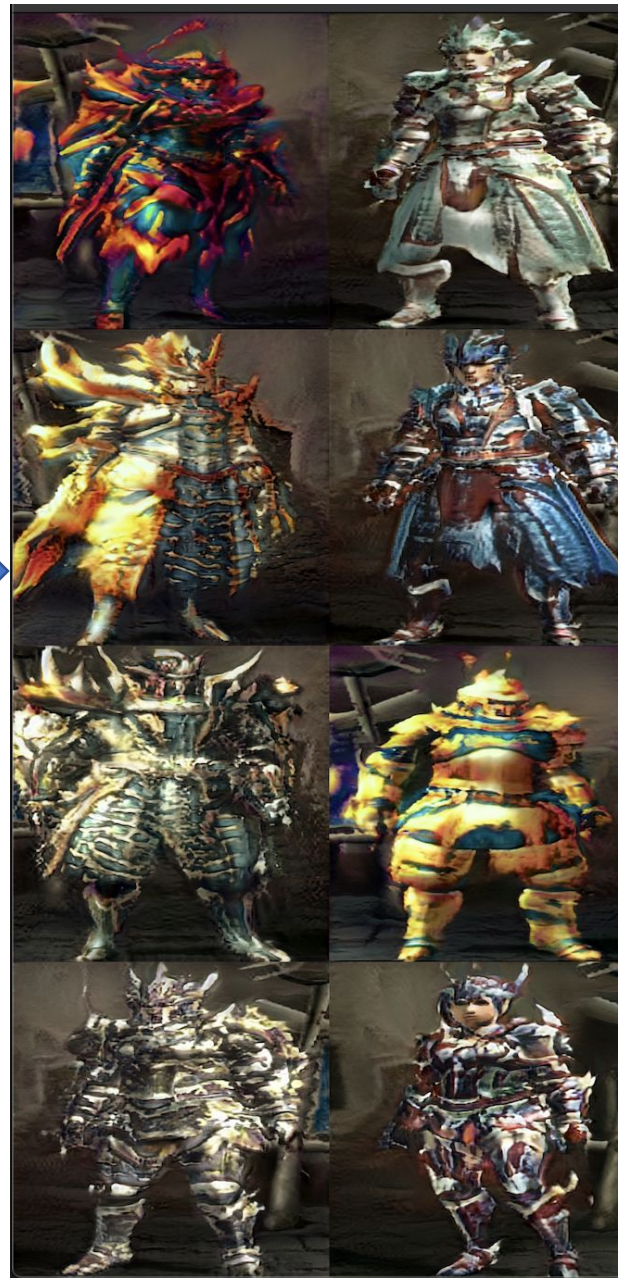


Image Generator and model

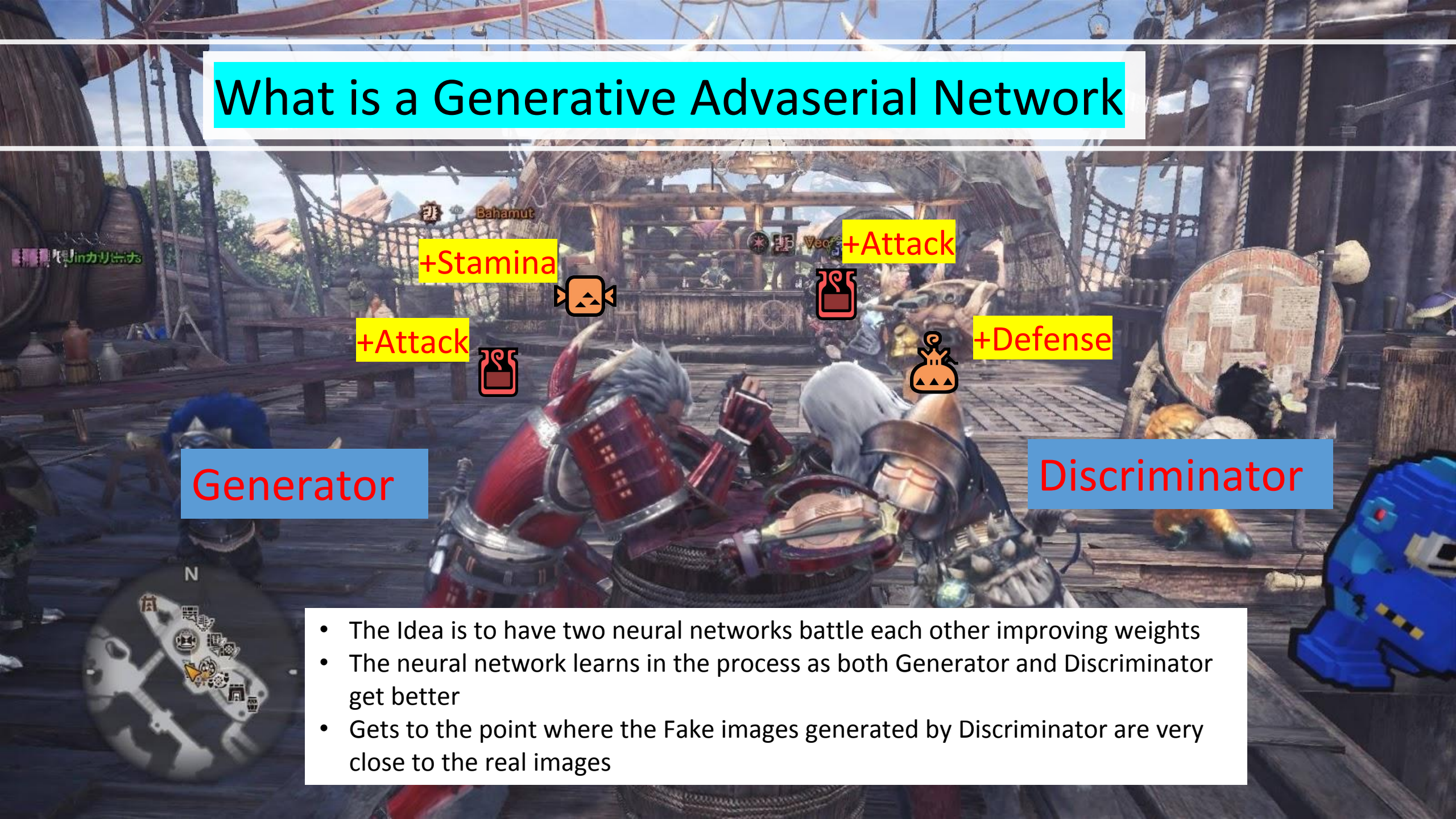


Learning to draw beboo bebop 0101010

Generate new Images for Concept design

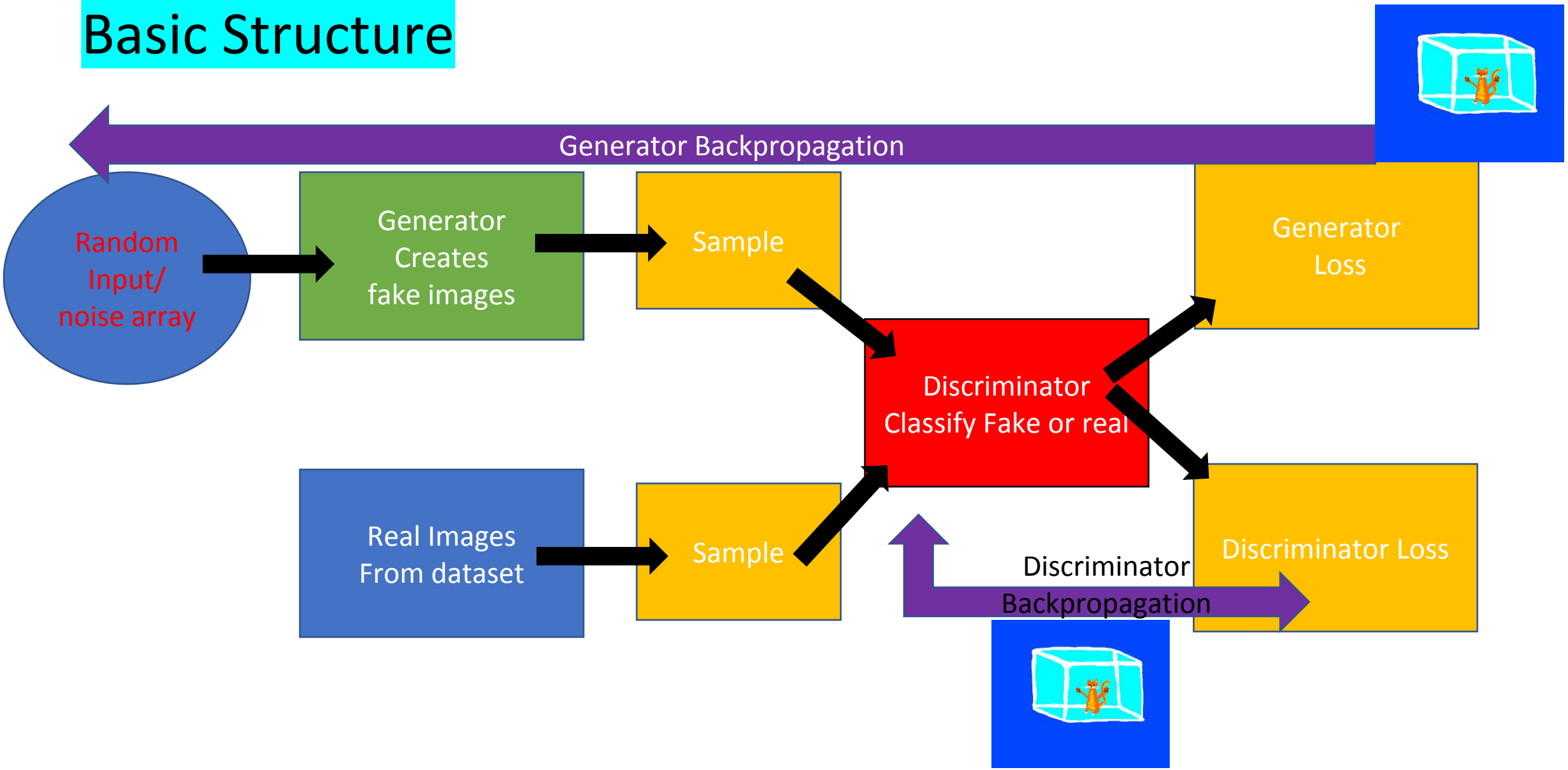


What is a Generative Advaserial Network



- The Idea is to have two neural networks battle each other improving weights
- The neural network learns in the process as both Generator and Discriminator get better
- Gets to the point where the Fake images generated by Discriminator are very close to the real images

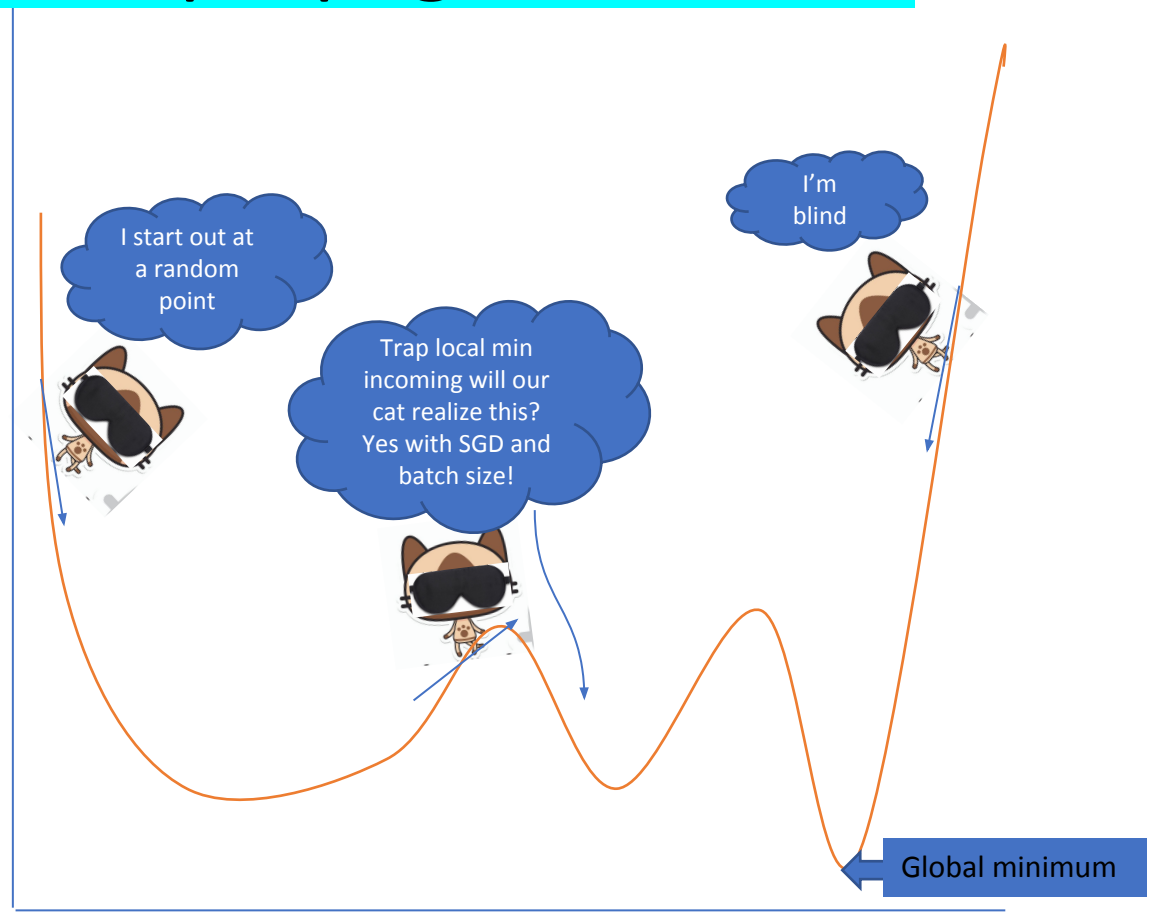
Vanilla Generative Adversarial Network Basic Structure



Gradient Descent and Backpropagation Basics

Cost

- The goal is to descend the gradient to reach the bottom where the cost function is the lowest.
- However, our cat here is blind. It can only slowly traverse downwards to see if the cost decreases. It would take repeat this step of taking steps forwards until it finds the lowest-possible point.
- Step size is dependent on the learning rate which is a hyperparameter to the model. Too small and it takes forever, too big and we might overshoot the minimum.



Parameter

- This would be either the weight or bias normally but could be billions of parameters.
- This would have a billion-dimensional space compared to our 1-dimensional space here.

Data Augmentation 1083 to 19437

Mirror our Images

Brightness random sampling 0.7 to 1.2

Resolution increase with PS

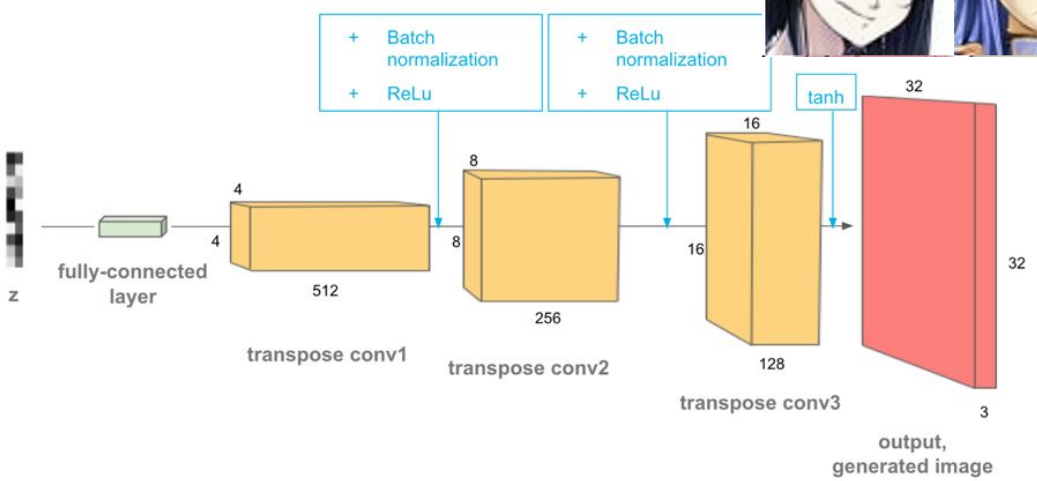
Before Photoshop was 600, 320, 3

After Photoshop increased to 1500,800,3

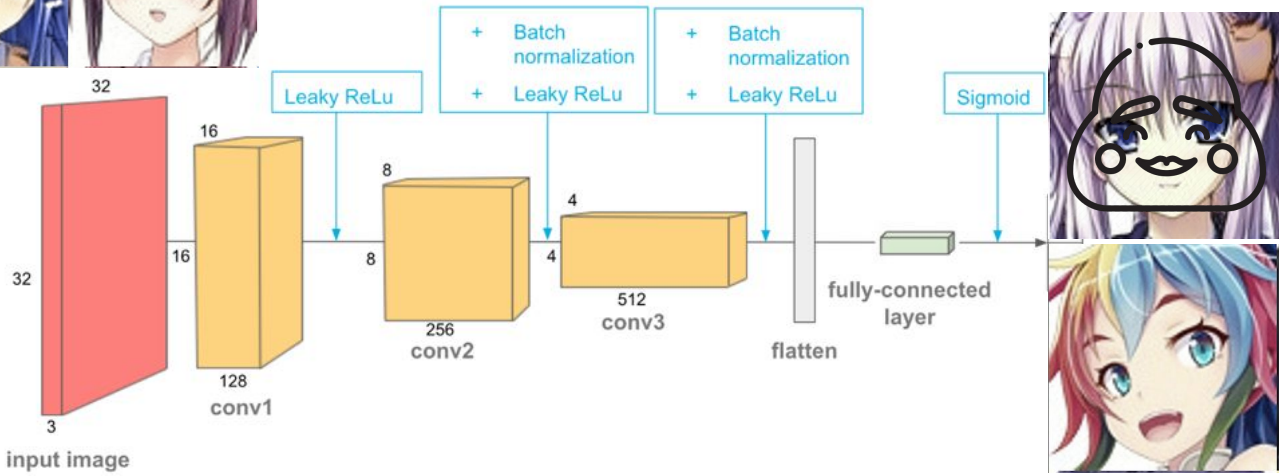
Deep Convolutional GAN Structure

Feed in Fake

Generator

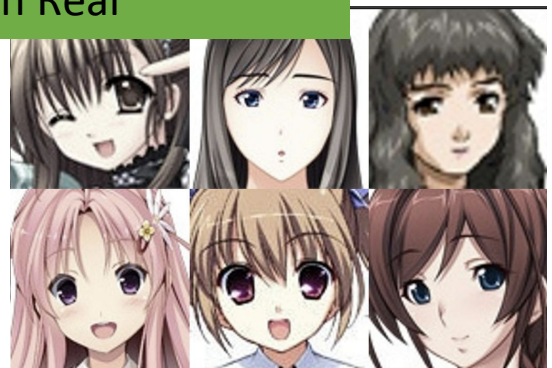


Discriminator



Fake

Feed in Real

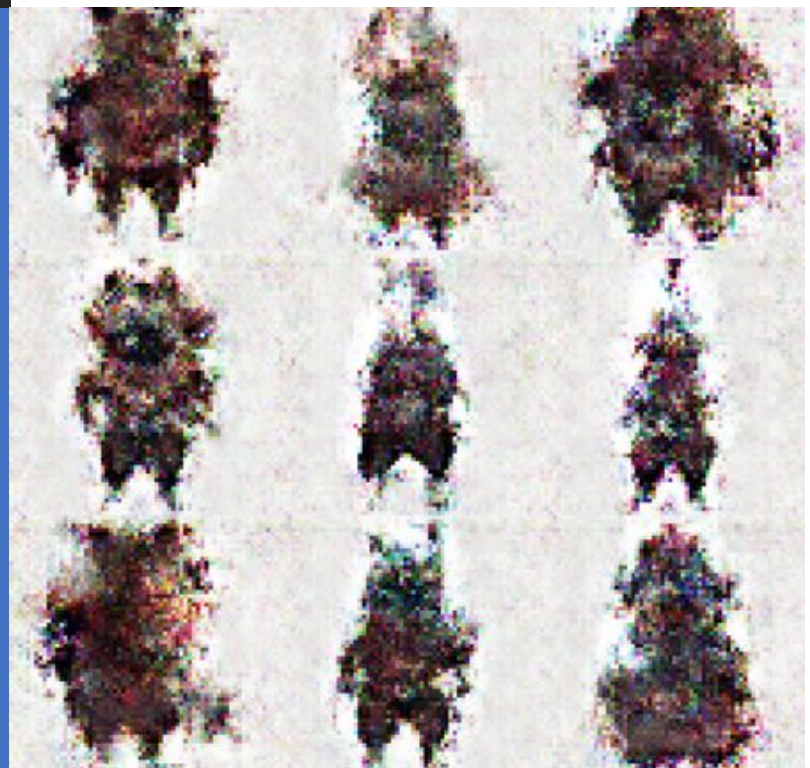


Real

DCGAN Quick VIZ

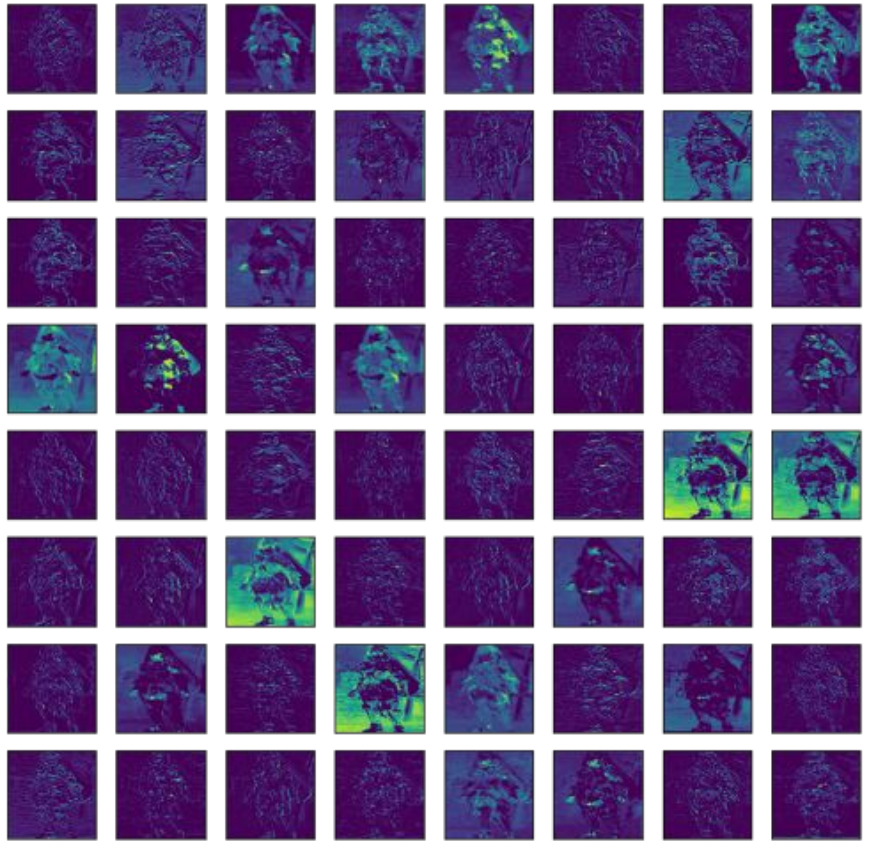
- 2000 steps only took 5 hours to be able to visualize some images already
- Style Gan will take a few days to train
- Is an improvement than vanilla GAN.

    - Batch normalization separates the fake images and real images instead of putting them together
    - Uses Leaky ReLU so we don’t have any dead neurons

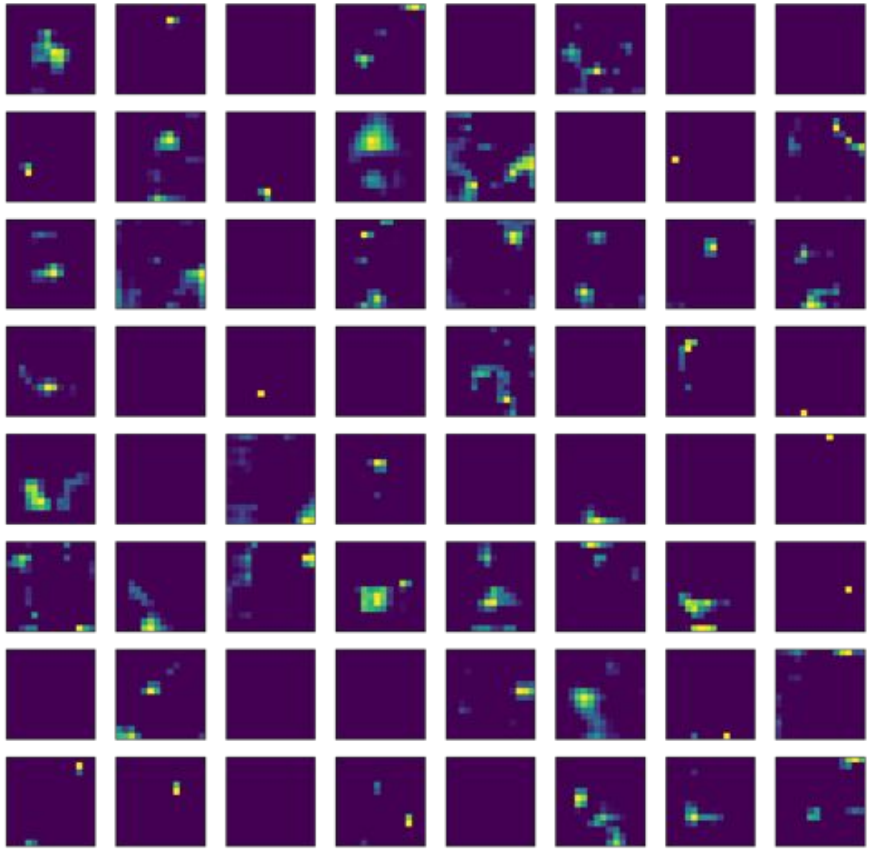


VGG Model Visualize Feature Map CNN

Closest to input



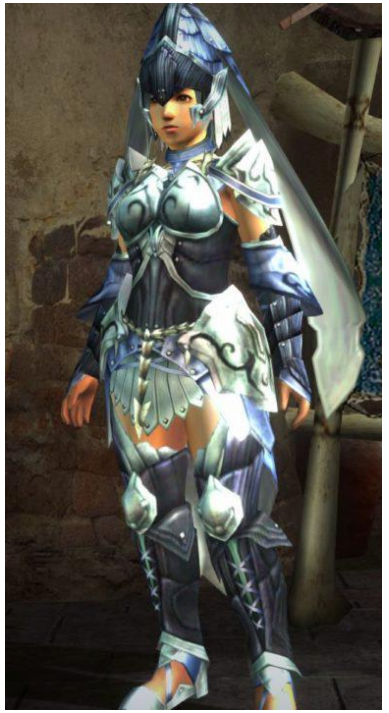
Furthest from input



Activation map/Feature map

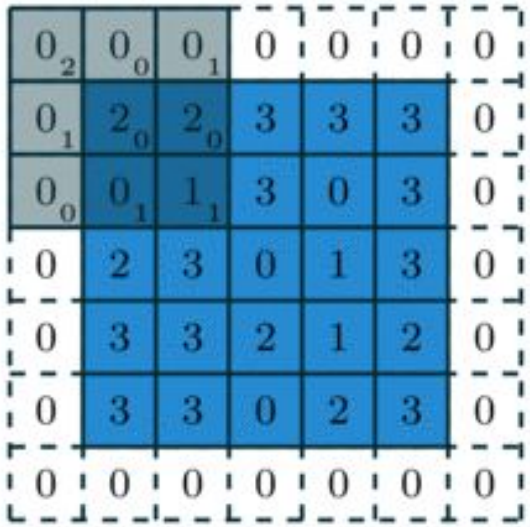
- Capture result of applying filter to input image or another feature map.
- The goal is to understand what features detected or kept

- Closer to input have more detail while further is going to have more general features.



Hyper Parameters

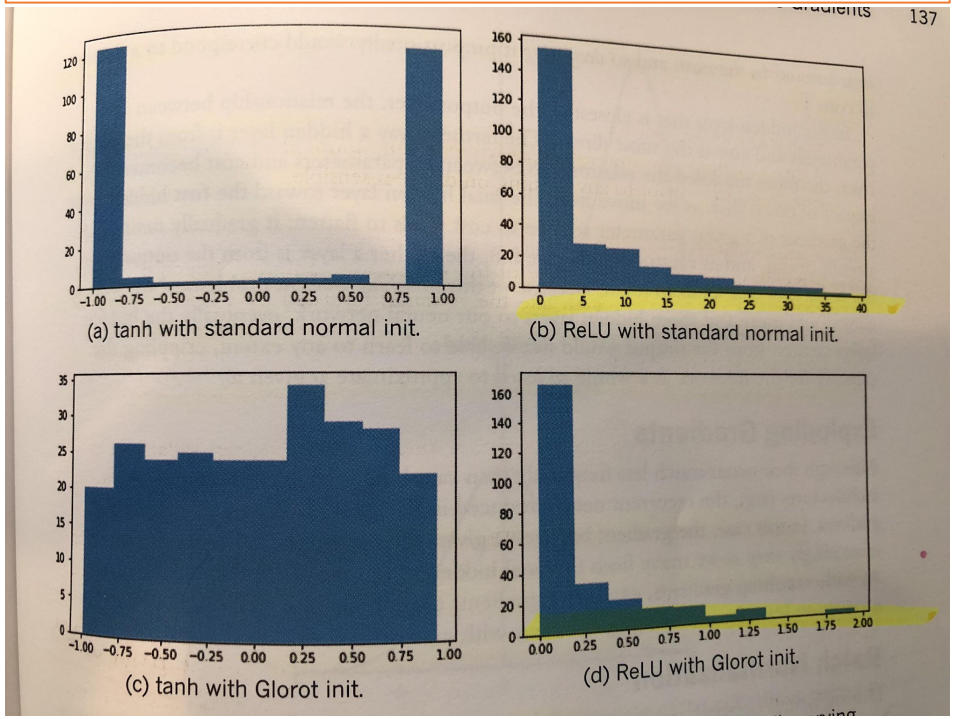
Filters	Was Conv2D for discriminator down sample from 64 to 512, Generator up sample using Conv2dTranspose
Kernel Size	Was set to 4
Strides	Size of the step the kernel takes was set to 2
Padding	Allows no loss of information to include padding of 0 set to padding = ‘same’
Kernel initializer	Set to glorot uniform to avoid neurons getting saturated or having too strong opinion before training



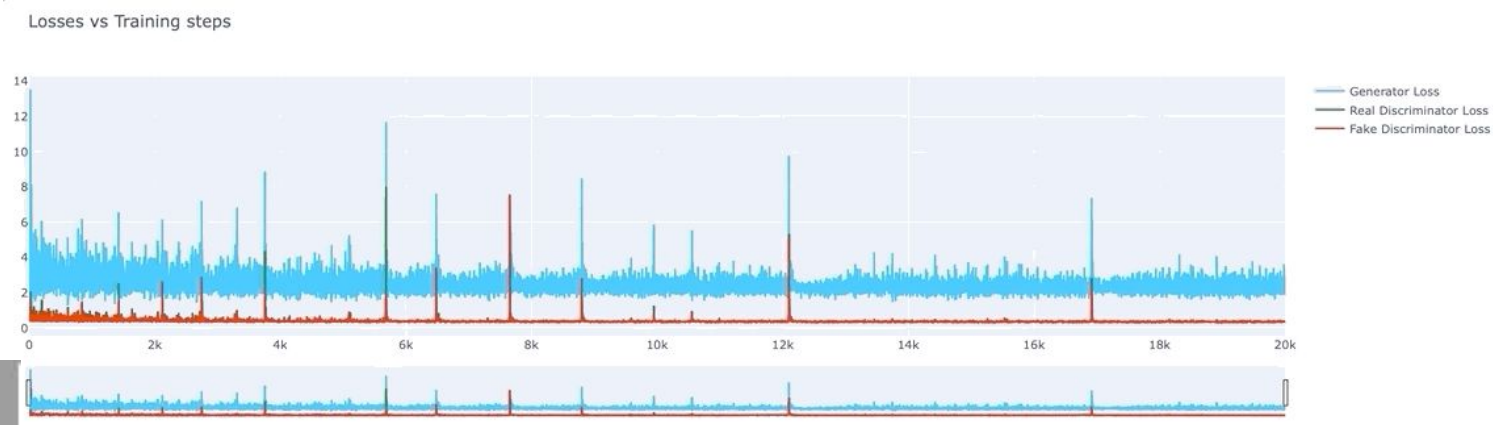
Here’s a model from deeplearning.net

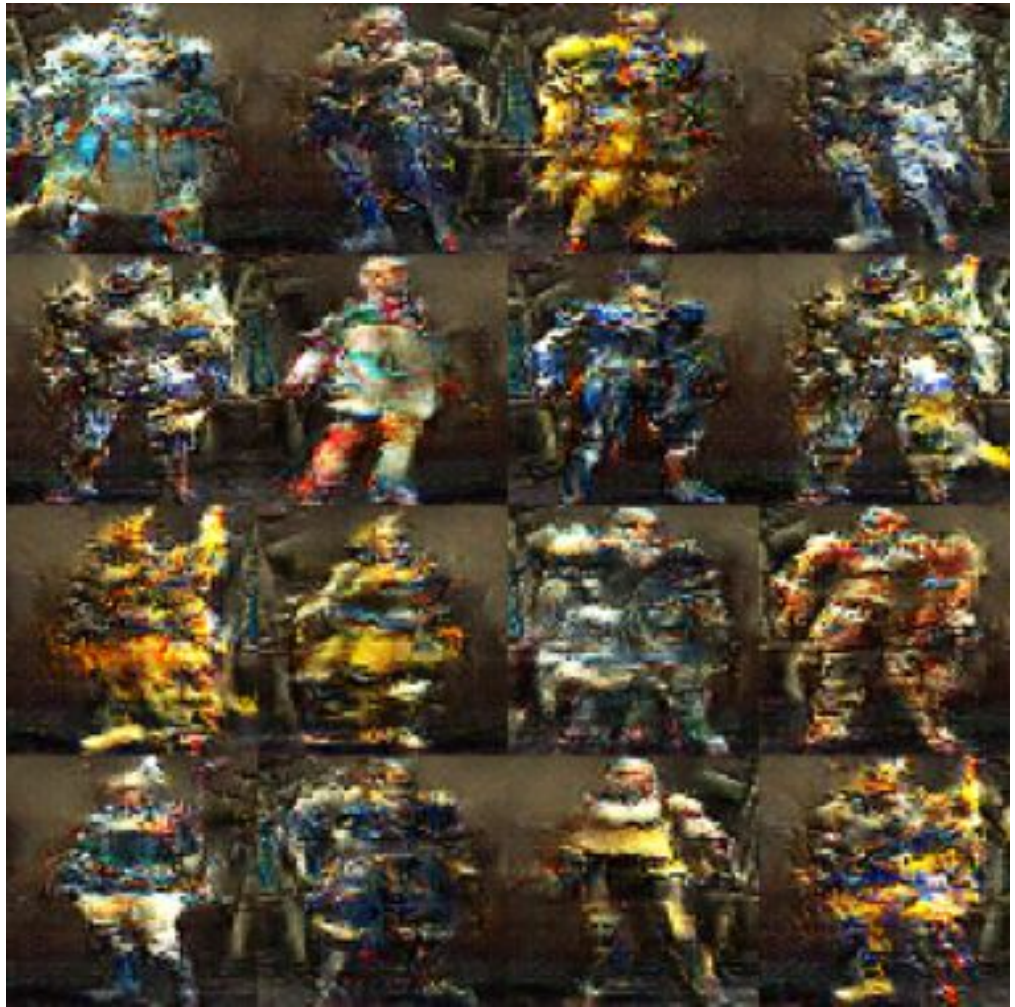
- Has 3x3 kernel size
- 2x2 Stride
- 5x5 input pad
- With 1x1 padding

Normal Initialization vs Glorot Intialization



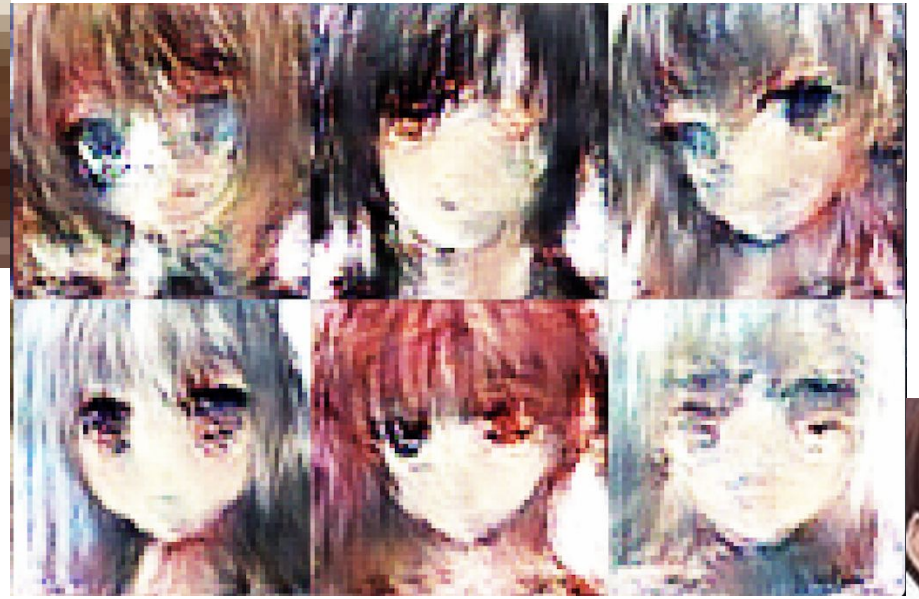
Results after 20000 Steps





GAN Measure

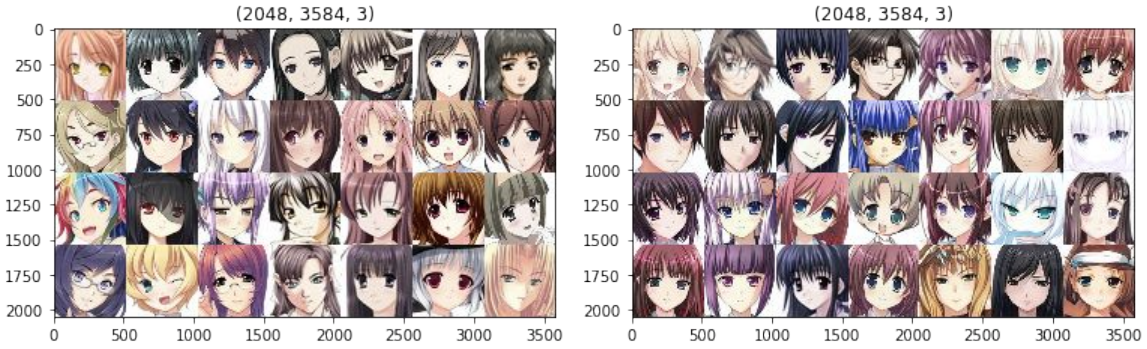
Usually the most common way we can tell our GAN is doing well is just looking at the comparison of the fake images generated by generator



Getting better!

Getting better!

Quantitative Inception Score & Frechet Inception Distance



IS uses two criteria in measuring the performance of GAN

- Quality of generated Images
- Diversity
- Higher the score the better indicating we can tell what the image is and is separated into the right number of classes with conditional probability

FID uses the same network as IS but:

- Model data distribution using Gaussian distribution
- use mean and covariance
- Finds the score between the real images and generated images.
- Lower the score the better the image quality and better diversity

In [56]:

```python
# prepare the inception v3 model
model = InceptionV3(include_top=False, pooling='avg', input_shape=(299,299,3))
# define the fake and real image collection
images1 = img12
images2 = img13
print('Prepared', images1.shape, images2.shape)
# convert integer to floating point values
images1 = images1.astype('float32')
images2 = images2.astype('float32')
# resize images
images1 = scale_images(images1, (299,299,3))
images2 = scale_images(images2, (299,299,3))
print('Scaled', images1.shape, images2.shape)
# pre-process images
images1 = preprocess_input(images1)
images2 = preprocess_input(images2)
```

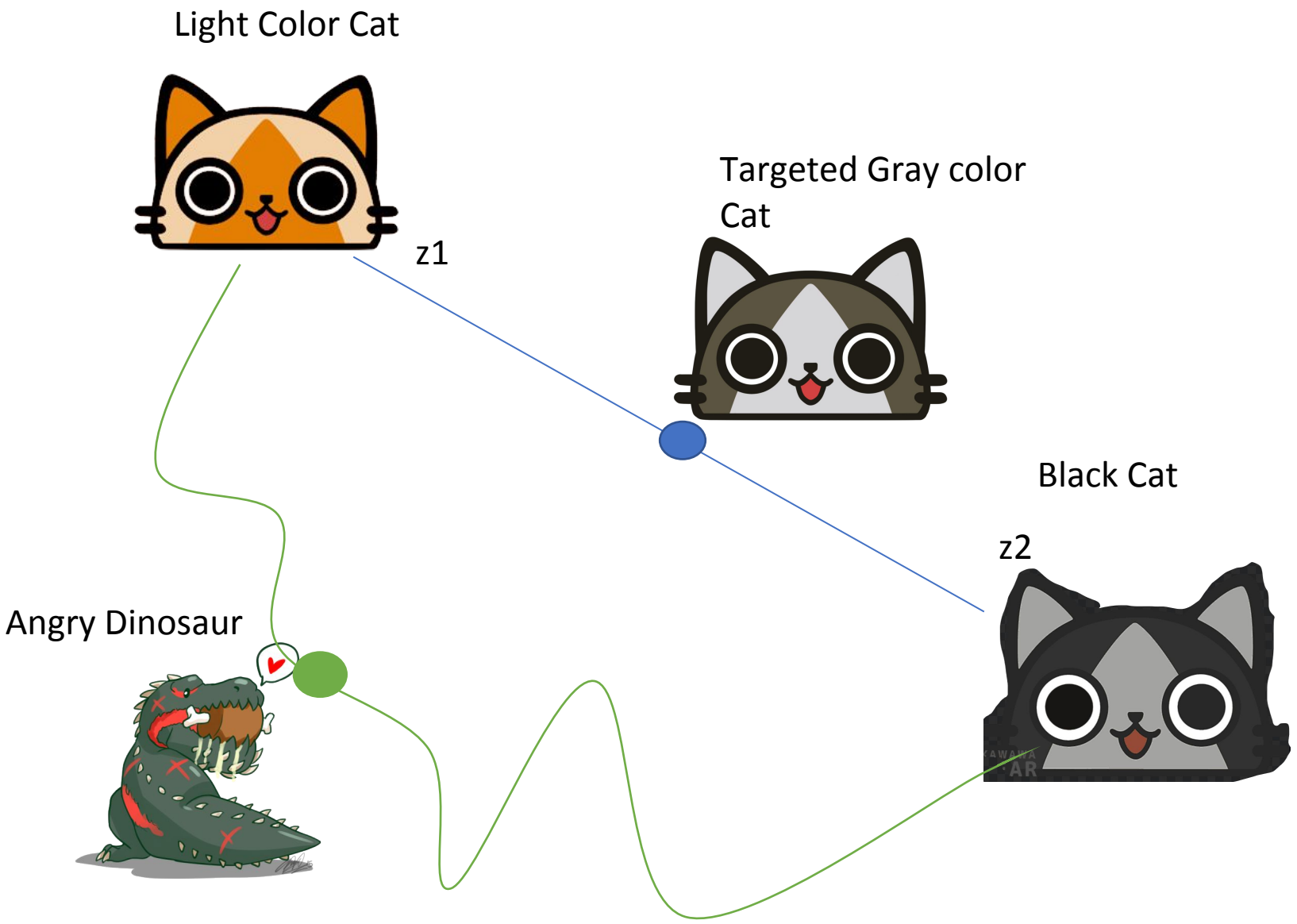
```
Prepared (128, 128, 3) (128, 128, 3)
Scaled (128, 299, 299, 3) (128, 299, 299, 3)
```

In [57]:

```python
# fid between images1 and images1
fid = calculate_fid(model, images1, images1)
print('FID (same): %.3f' % fid)
# fid between images1 and images2
fid = calculate_fid(model, images1, images2)
print('FID (different): %.3f' % fid)
```

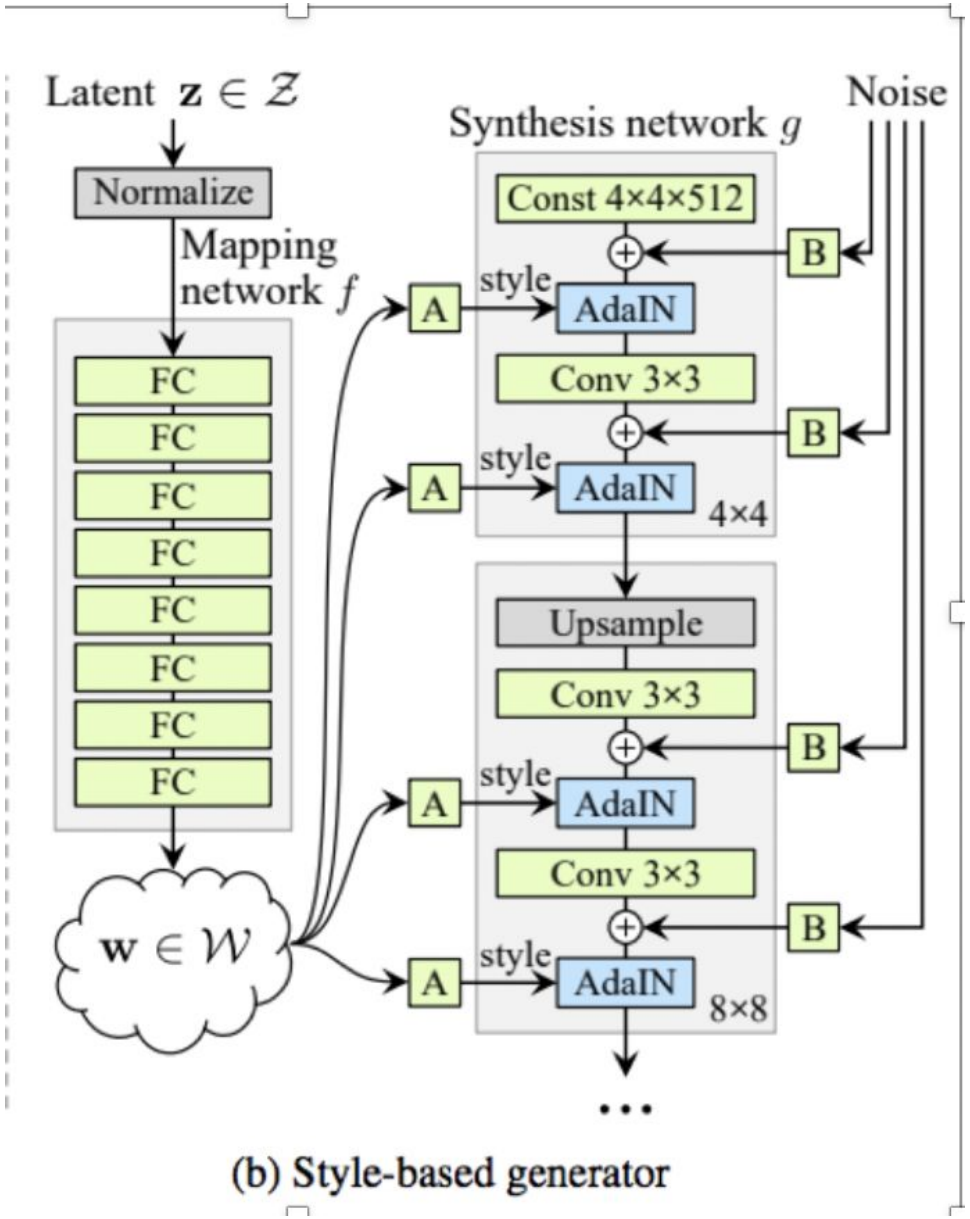
```
FID (same): -0.000
FID (different): 7.314
```

Quantitative Perceptual Path Length



- The idea here is to choose the “perceptually” shortest distance.
- For example, using a latent variable z1 produces a brownish light colored palico cat and a latent variable z2 produces a black palico cat.
- Since only color is being changed the ideal path is the blue line where the middle ground is a gray palico cat which is “perceptually” the shortest distance.
- The longest distance “perceptually” is the dinosaur as it needs to change shape and color
- The lower the PPL the better.

Style Gan Improvement



(b) Style-based generator

- Use Progressive Growing to gradually increase the resolution.
  - Synthesis network g
- Generate images from fixed value tensor, not generating images from stochastically generated latent variables as in conventional GANs.
  - StyleGAN generates image from fixed 4x4x512 tensor.
  - And stochastic latent variables are used as style vectors in StyleGAN.
- The stochastically generated latent variables are used as style vectors through Adaptive instance normalization at each resolution after nonlinearly transformed by an 8-layer neural network.

Style GAN- Anime Faces

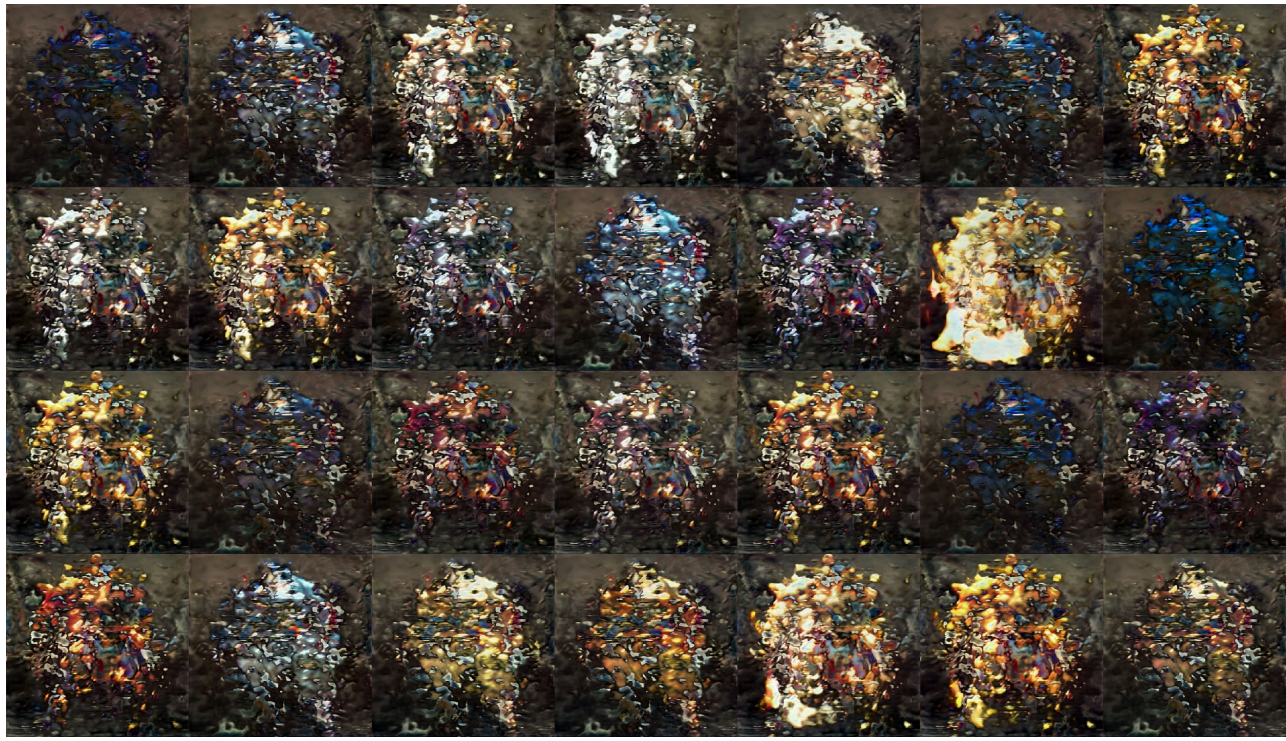
Anime Faces

- Started with 85000 images
- 512 x 512 dimension using tfrecords
- LR was 0.003
- Minibatch repeat was set to 5
- Trained for 7 days

Style GAN- MHXX Armor

Hoping it was as easy here's 2 days of training

Mode Collapse at 4 days of training



What is mode collapse

- Mode collapse when generator only capable of generating one or a small subset of different outcomes, or modes. Like our exploding blurred armor on the right at 4 days of training

Possible reasons for mode collapse

- Not enough data vs our face dataset. Only 19437 we might be overfitting the data with augmentation
- Generator has collapsed to a single point believing what it is drawing is real, then discriminator learns from that single point ang gradient descent unable to separate identical outputs

Transfer Learning using anime faces

Hyperparameter changes

- While learning rate is one of the more important thing to set in Stylegan 1 for some reason you can only set learning rate the same for D and G in Stylegan 2; decreased LR to 0.001
- set minibatch repeat to 1 instead of 5. Perhaps having too many minibatches to give to discriminator caused the model to be unstable

Transfer Learning take a model trained on a large dataset and **transfer** its knowledge to a smaller dataset.

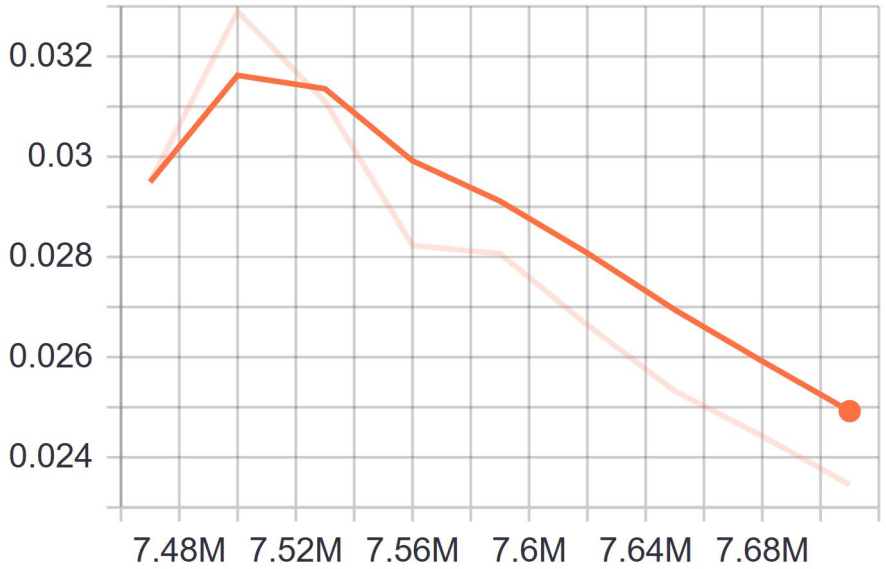
- Set resume_kimg to the pickled face model at 7440
- This allows us to use the advantage that the anime face model had a bigger data set

Transfer learning Results in 16 hours finally armor and face detail!

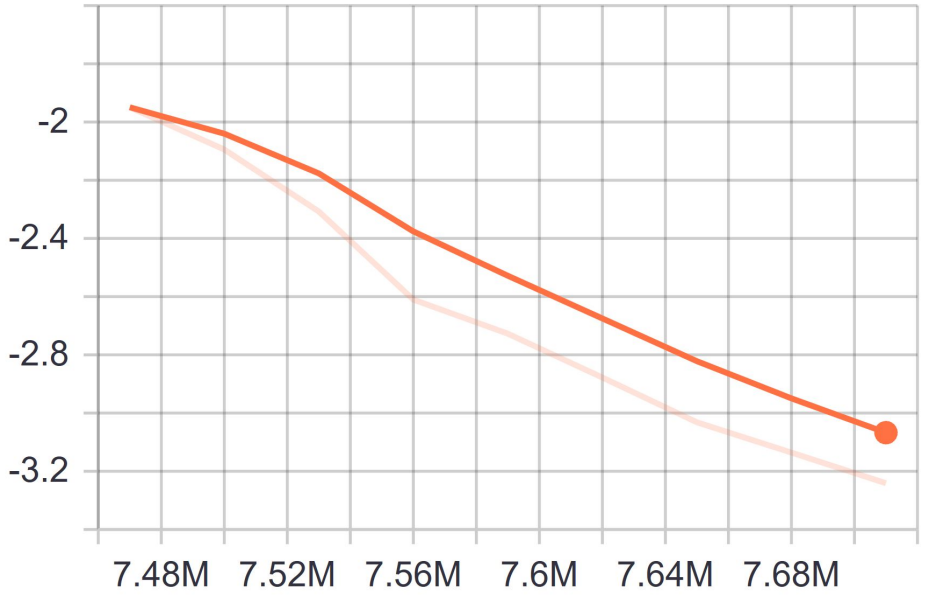
Armor Morphing we can choose a concept Art!

TensorBoard

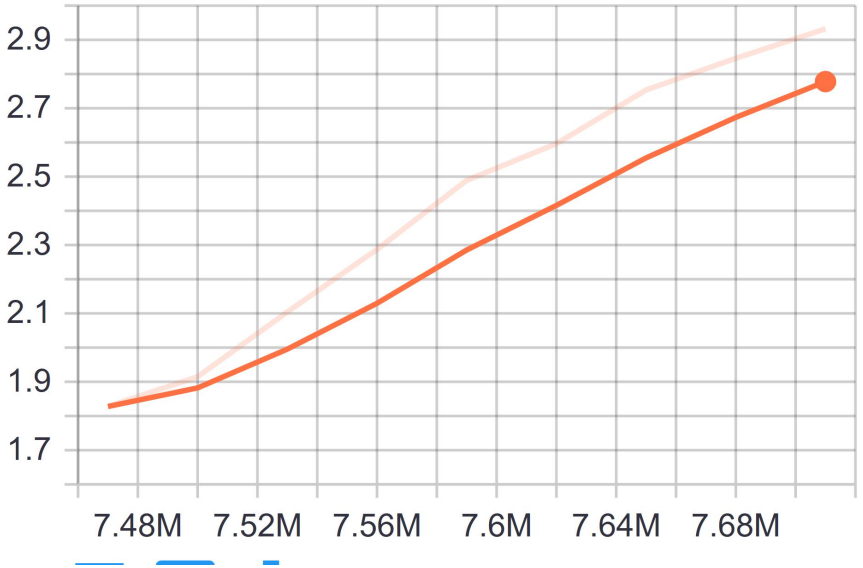
r1_penalty
tag: Loss/r1_penalty



scores/fake
tag: Loss/scores/fake



scores/real
tag: Loss/scores/real

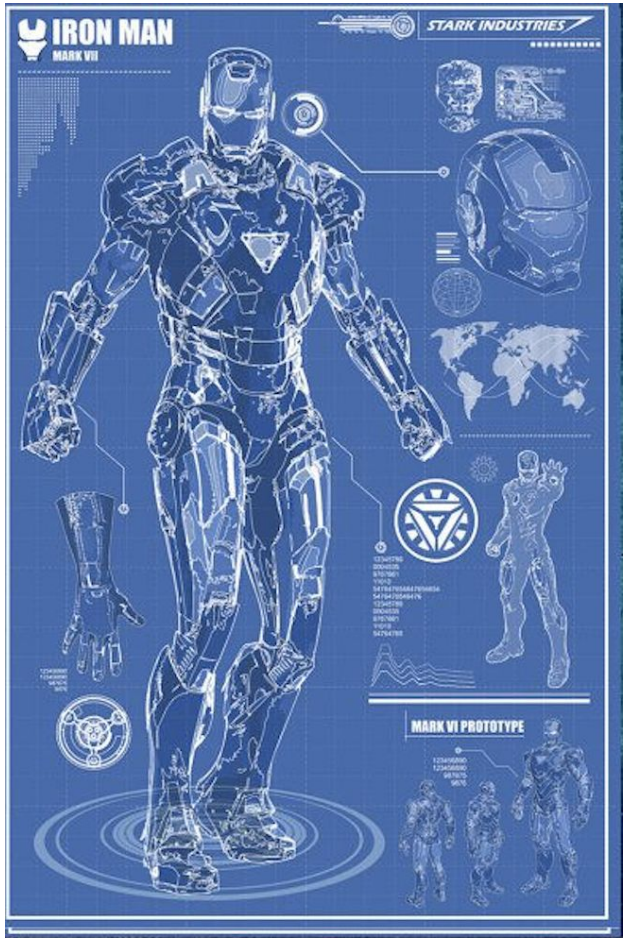


- r1_penalty-Generator loss is going down so producing better images

- Fake_score- Discriminator Fake Loss – Getting better at classifying fake images

- Real_score – Discriminator Real Loss – Getting higher and is getting worse at classifying real images
- Might be due to data size or artifact blurring

Conclusion and Improvements



Improvements to try
Mixed Armor from MH community
Ability to manipulate the vector to change features on hunter

Additions for future
GPT2 for armor combinations and hunter name
Add in more variation such as weapons and the newer MHW

Future Work to explore
Generation ideas are infinite as GAN improve
Background images and character concept illustrations