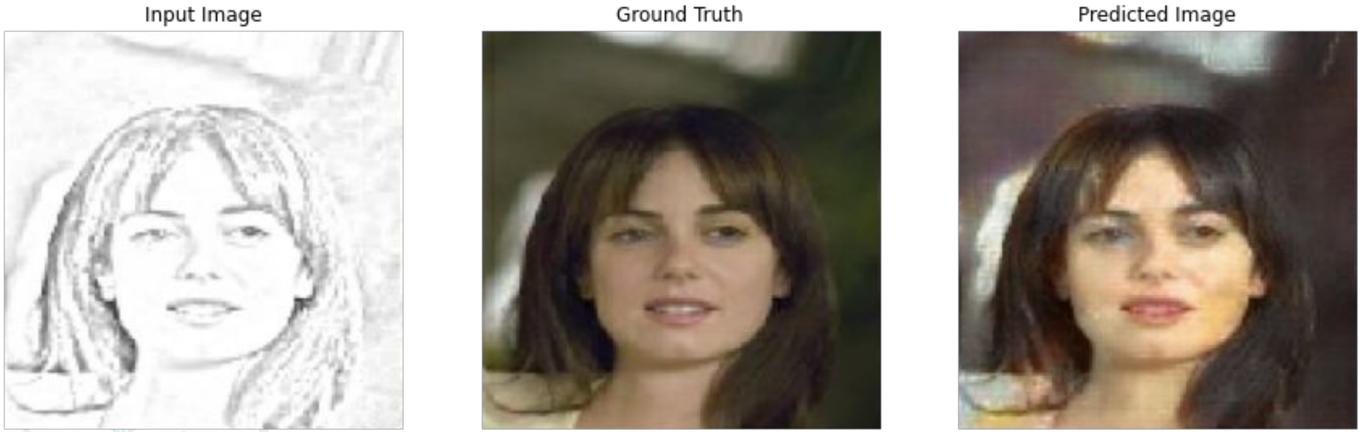
# Sketch to Face Translation

## Week 12/15 Final Presentation




Group 9: Joe Suzuki, Jason Li, Anthony Sokolov

# Problem Description

**We aim to implement a model that takes a sketch of a human face and try to generate a realistic looking face conditioned with the sketch**

- Can be used for rendering and animation tasks
- Useful to create new training data
- Useful for forensic purposes

# Roles

**Anthony** explored the performance of conditional GANs cycle consistent GANS under different learning rates and generator types. He implemented a ResNet generator and set up the scaling algorithm to be used for our resolution assisted GAN. He computed the Frechet Inception Distance (FID) to measure image quality evaluation of each GAN's output.

**Joe** implemented markovian (PatchGAN) discriminator and a U-Net encoder-decoder generator for a conditional GAN architecture. He also implemented a conditional GAN with a U-net + residual blocks + DCGAN generator, and a loss that consists of adversarial loss, identity loss, and L1 loss.

**Jason** preprocessed the dataset and used OpenCV to generate sketch images. He trained conditional GAN models on different styles of input sketches to determine the ideal dataset to use. He visualized performance and evaluated training loss by using the loss log for each model to compare results.

# Generative Adversarial Networks (GANs)

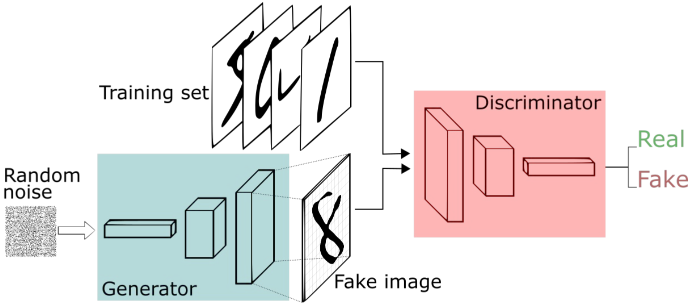
Generative Adversarial Networks (GANs) are a powerful generative model first introduced in Ian Goodfellow's 2014 paper.

GANs consist of a generator network and discriminator network

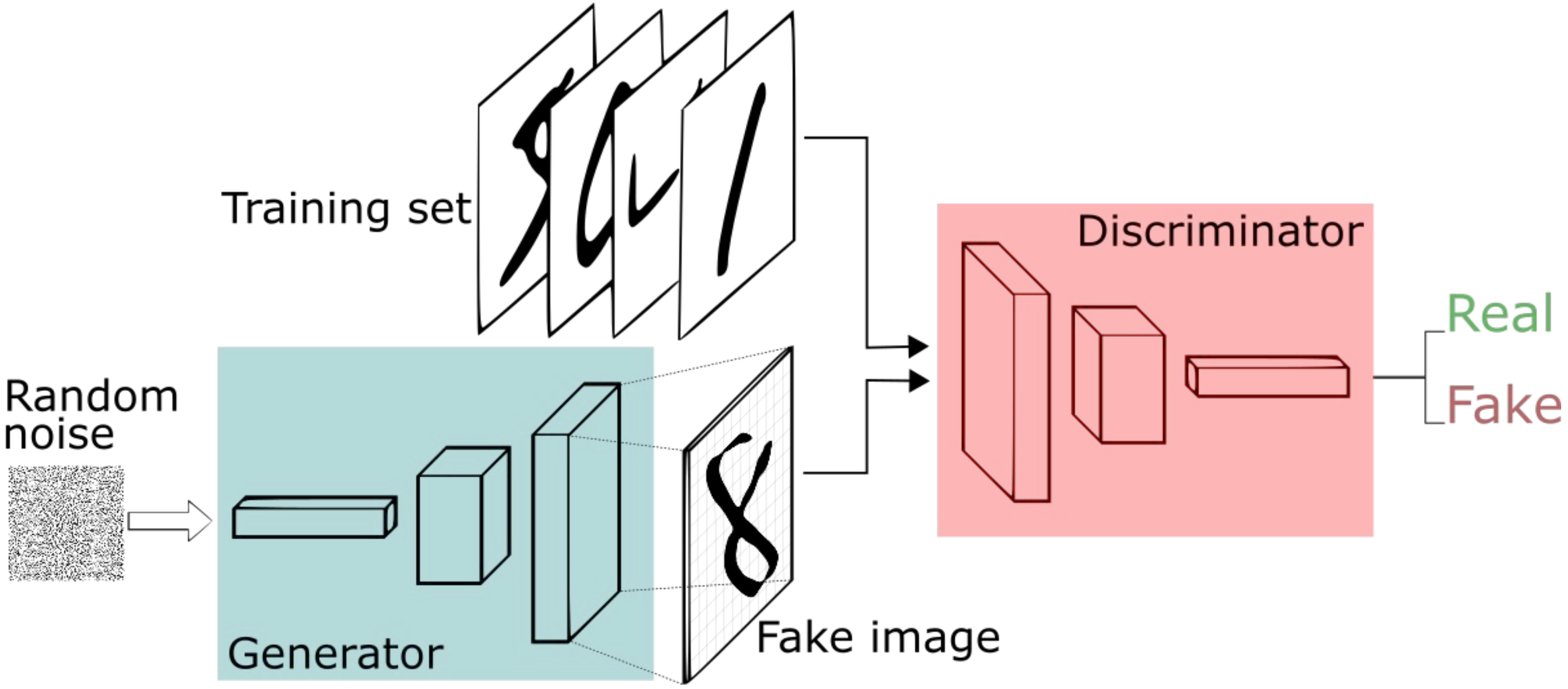
The generator tries to create images that fool the discriminator: (noise ξ, class Y | feature x)

The discriminator tries to classify whether image is real or fake:  models the P(class Y | feature X))

The two networks are trained in tandem in a game like manner



# Generative Adversarial Networks (GANs)



# How to condition a sketch in our GAN model?

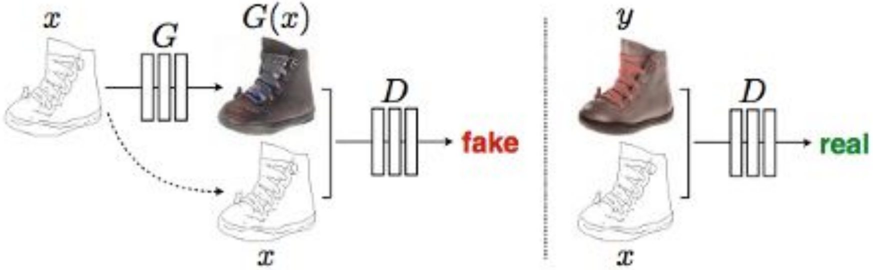
# Conditional Adversarial Networks

To build out conditional GAN we used the Pix2Pix architecture as our base

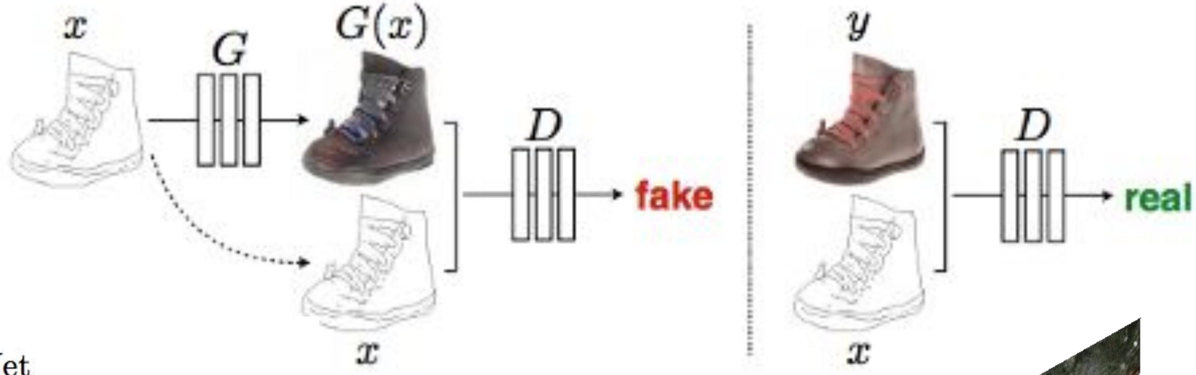
cGANs takes in an input image on which it is conditioned to make generations

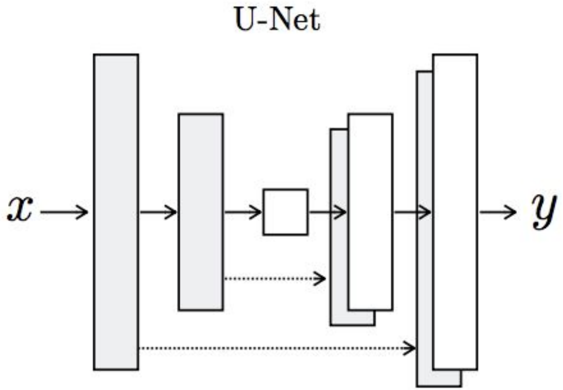
The generator of the Pix2Pix GAN uses a modified encoder-decoder network called U-Net.

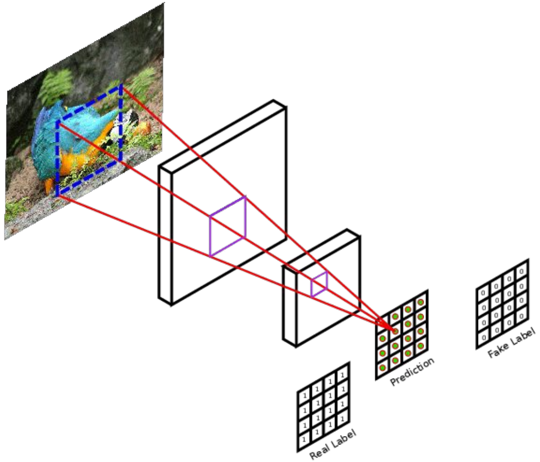
The Pix2Pix discriminator, called PatchGAN, outputs a matrix of classifications instead of a single value(real or fake).



# Conditional Adversarial Networks



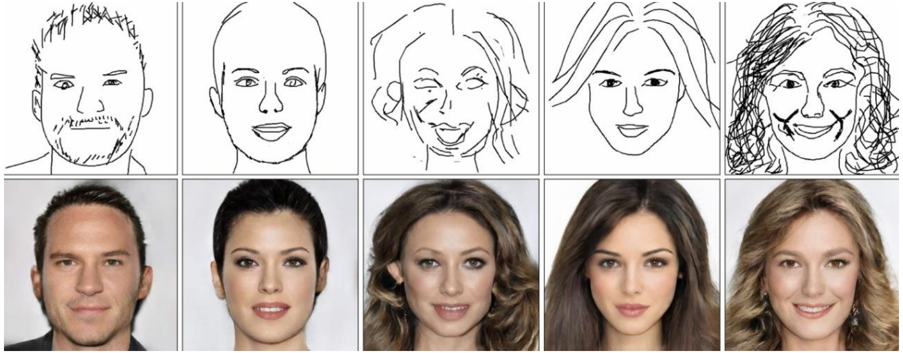



# Recent Advancements

In past years GANs have had a lot of success in image translation tasks

DeepFaceDrawing (2020)



High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs (2018)

# Setup

**Dataset:** Celebrity Faces Dataset

**Environment:** Google Colab

**Input Images:**

- Face edges using Canny edge detection algorithm
- Realistic sketches using OpenCV

# Experiments

We fit conditional GAN (paired images) and CycleGAN (unpaired images) architectures to our dataset

We explored the use of different learning rates, generators, and loss functions

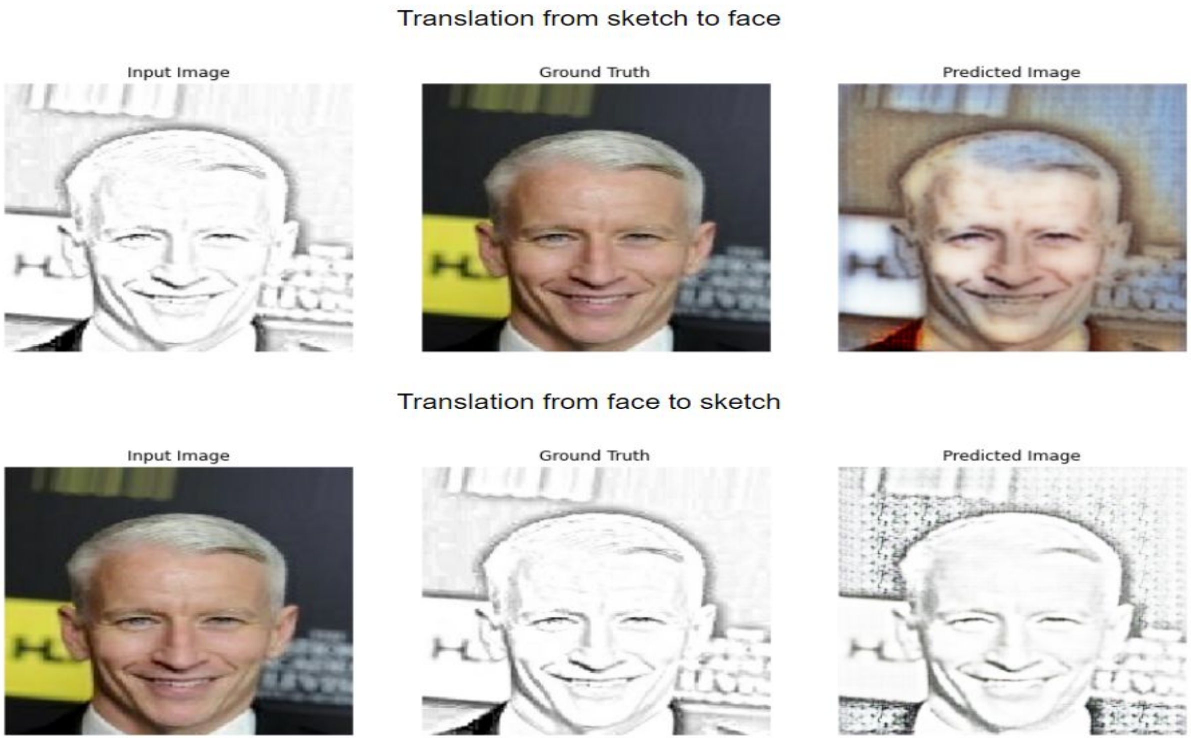
As a novel approach, we explored training lower resolution, more stable GANs and then scaling the output using deep learning

# Conditional GAN Learning Rates Comparison

Sample outputs using different learning rates

| 0.01 | 0.001 | 0.0001 | 0.00001 |
|---|---|---|---|

# CycleGAN Translations



# CycleGAN Learning Rates Comparison

Equal learning rates

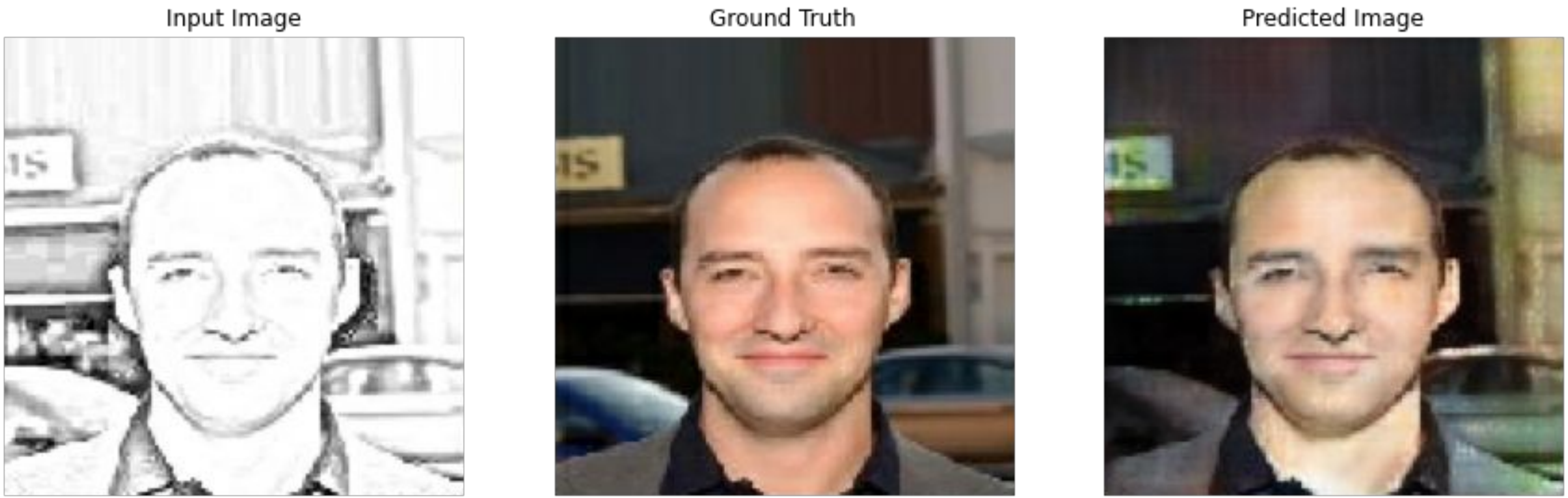
Generator learns twice as fast

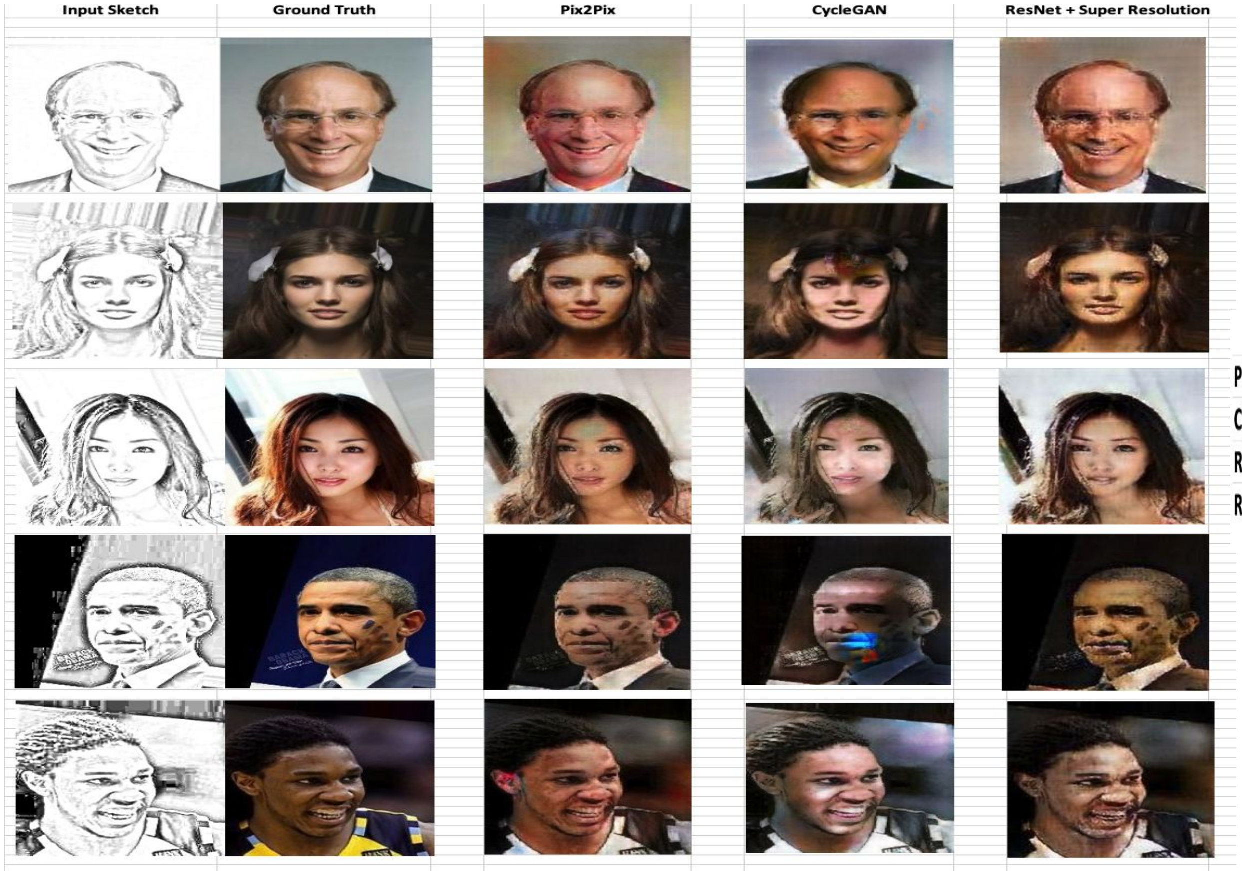
Note: the "checkerboard" appearance occurs from transposed convolutions during upsampling. Ways to avoid this is by using upsampling + convolution

# Smaller Conditional GAN + Super Resolution

We trained a smaller Conditional GAN that outputs 128x128 images

We used a pretrained CNN to scale the outputs from the GAN to 256x256






| | Frechet Inception Distance | Average Pixel Difference |
|---|---|---|
| Pix2Pix | 142.44 | 0.9 |
| CycleGAN | 143.18 | 0.96 |
| ResNet + Bicubic Interpolation | 142.05 | 0.91 |
| ResNet + Super Resolution | 139.07 | 0.91 |

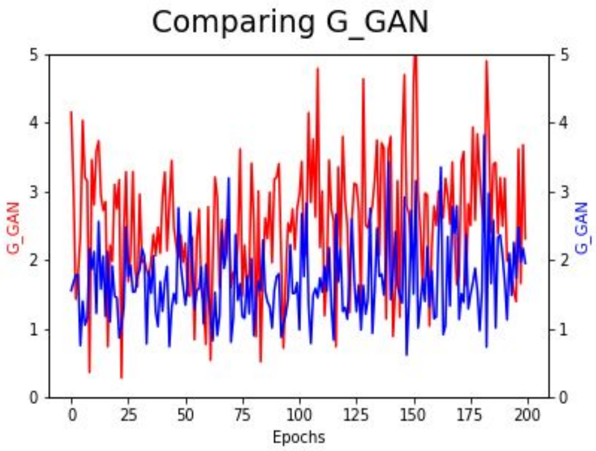
# Initial Testing

- Tested the model using face edges
- Tested the model using realistic sketches

Face Edges Input | Face Edge Output | Realistic Sketch Input | Realistic Sketch Output | Actual Image

# Evaluations

- Loss is uninformative of performance as values often oscillated which does not give us a clear indication of how well the model is performing.
- We used a distance metric between generated and real distributions as a rough evaluation metric




| | Frechet Inception Distance | Average Pixel Difference |
|---|---|---|
| Pix2Pix | 142.44 | 0.9 |
| CycleGAN | 143.18 | 0.96 |
| ResNet + Bicubic Interpolation | 142.05 | 0.91 |
| ResNet + Super Resolution | 139.07 | 0.91 |

# Lessons Learned

This project taught us a lot about neural networks

Complex models are hard to interpret

ML is still an expanding field

We learned about applications of entropy and KL divergence

# Challenges faced

- Data preprocessing
- Time constraint
- Environment Crashing

# Goals not achieved

- Used FID(Frechet Inception Distance) instead of other types of evaluation metric due to time constraint
- Test models for certain biases (such as skin color)