CS419 Project
Exploring GANs and Application[2]

[2] https://github.com/siddhant-midha/CS419M-Project

Table of Contents

Motivation

- Throughout the course, we were exposed to a variety of algorithms and basic ML techniques.
- We were also briefly introduced to deep learning.
- Hence, we took up this project to help us appreciate some crucial aspects of deep learning all the while understanding the fundamentals.
- Out of interest, we opted for GANs.
- Out of more interest, we chose image translation – particularly, anime face colouration.

Table of Contents

What are GANs?

- TLDR: GANs implicitly estimate distributions by learning the ability to sample from them.
- Two components,
  1. Generator (G) – The artist: Tries to generate samples from the distribution.
  2. Discriminator (D) – The art critic: Tries to root out generated samples from the real samples.
- This competition enables the learning of a mapping from the latent space to the image space.

What are GANs?

Formally, we have

$$J_D(\theta_D, \theta_G) := -(\mathbb{E}_{x \sim p_{data}} log(D(x)) + \mathbb{E}_z log(1 - D(G(z))))$$

$$J_G(\theta_D, \theta_G) := -J_D(\theta_D, \theta_G)$$

Define $V(\theta_D, \theta_G) := -J_D(\theta_D, \theta_G)$. Thus we have the following game

$$\min_{\theta_G} \max_{\theta_D} V(\theta_G, \theta_D)$$

Minima achieved when G draws samples exactly from the distribution.

Table of Contents

A basic GAN

- We observed that the G and D functions incur a lot of freedom in design.
- Hence, we began with a basic design wherein both of them consist of Dense layers.
- We noted that certain up sampling is required in the G architecture.
- Also noted that down sampling is needed in the D architecture, followed by a `sigmoid`.
- We trained this basic GAN on both `MNIST` and `Fashion-MNIST` datasets.

A basic GAN – Results

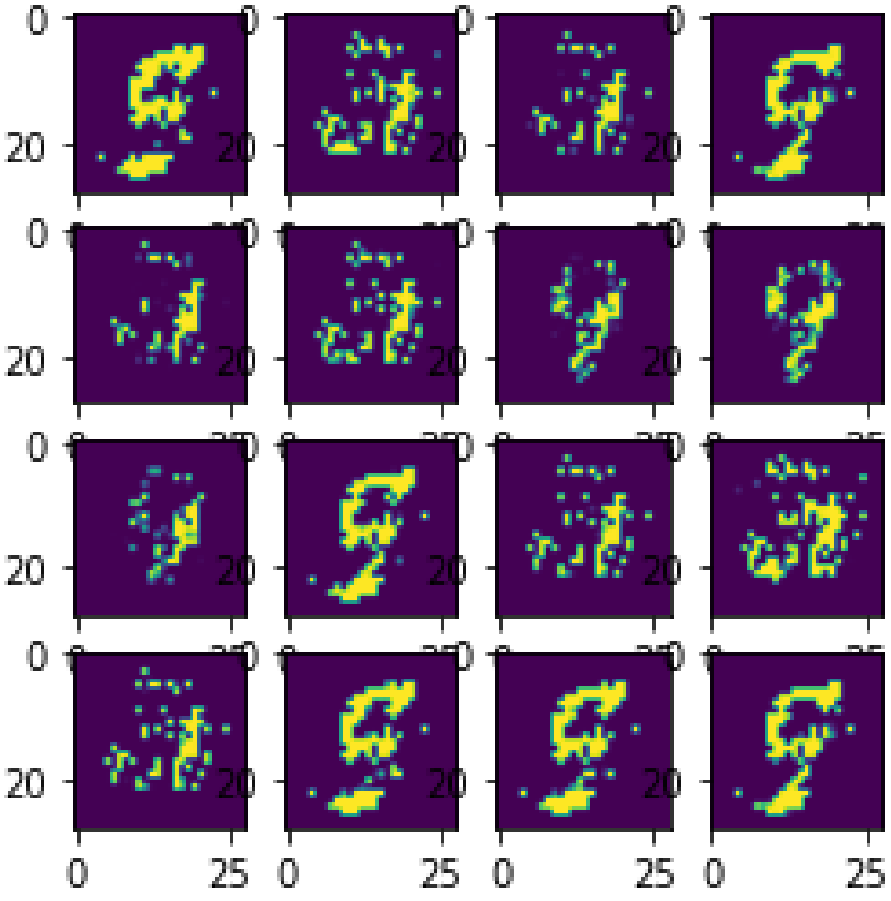

Figure: MNIST, 20 epochs

A basic GAN – Results

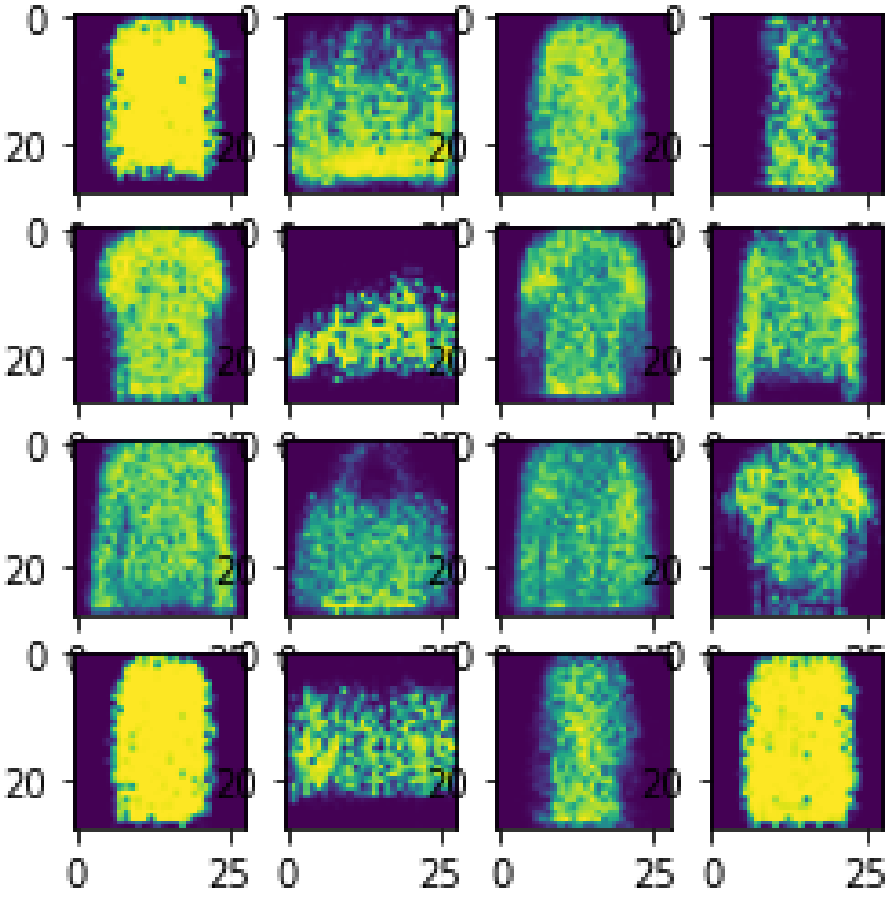

Figure: FashionMNIST, 20 epochs

A basic GAN – Observations

- We see that overall performance is better on the `Fashion-MNIST` dataset.
- Even more so, we note that additional batch normalization was needed in order for the GAN to work on the `MNIST` dataset.
- That is, initially the exact same model performed much better on the former than the latter – intricacies in the dataset.

A deeper GAN

- We note the deficiencies in the basic GAN architecture, and realize the need for convolutional layers to preserve local information in images.
- Here we implement the convolutional-GAN.
- We implement this on `MNIST` and `FashionMNIST` datasets.

A deeper GAN – Results

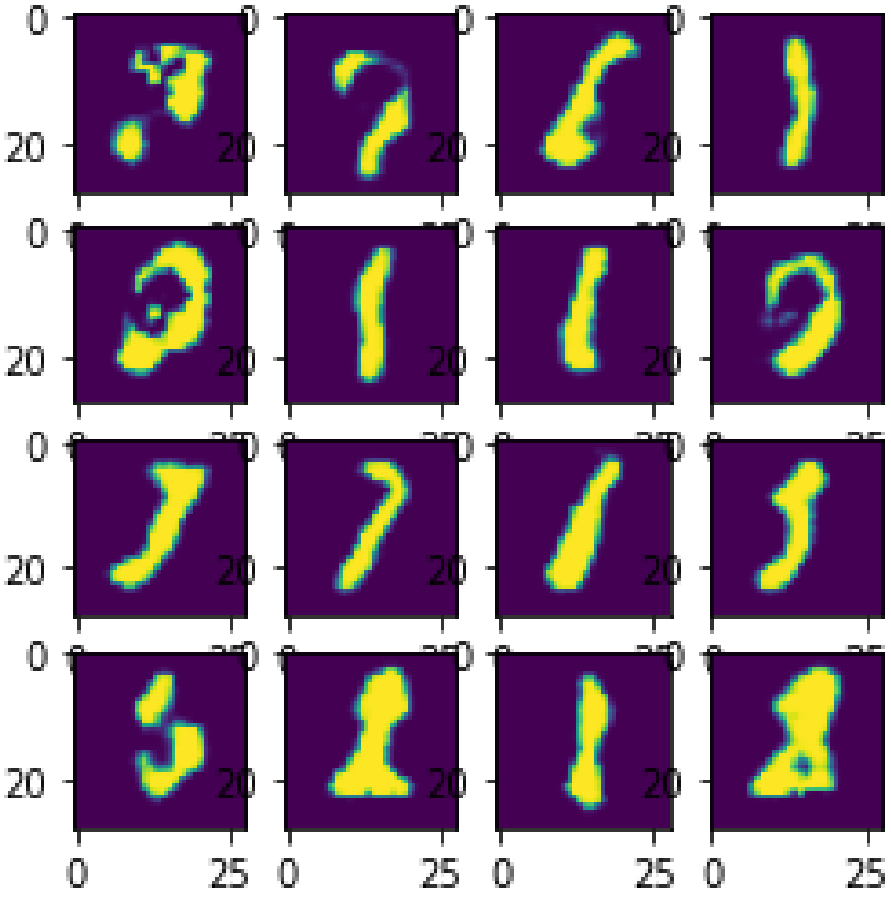

Figure: MNIST, 20 epochs

A deeper GAN – Results

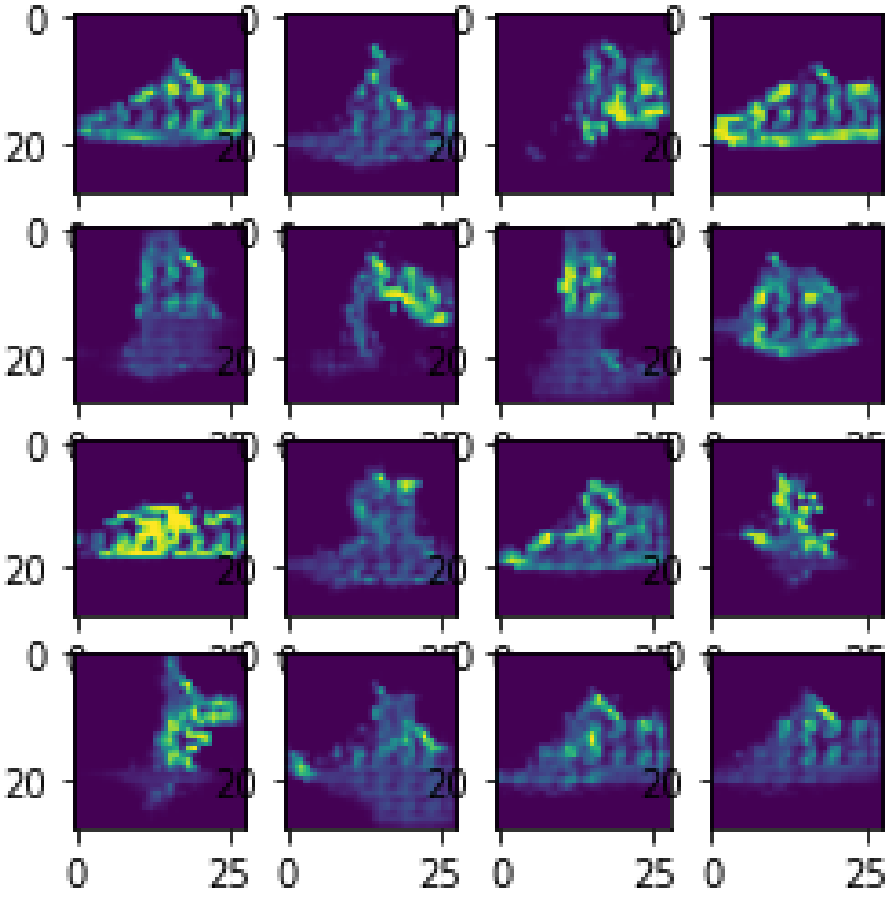

Figure: FashionMNIST, 50 epochs

A deeper GAN – observations

- Image quality improves, as expected.
- Surprisingly, here with batchnorm and 20 epochs.`MNIST` does much better than `FashionMNIST`.
- Then, we try increasing the number of epochs for the former.
- Does not help.
- Decreasing Batchnorm helps.

Conditioning the GAN

- We appreciate the use of GANs for sampling from distributions.
- We now require to sample from particular classes.
- That is, we feed the class-labels (such as the digit `7` in `MNIST`) in the training process.
- Hence getting conditional-GAN.

Conditoning the GAN – Results

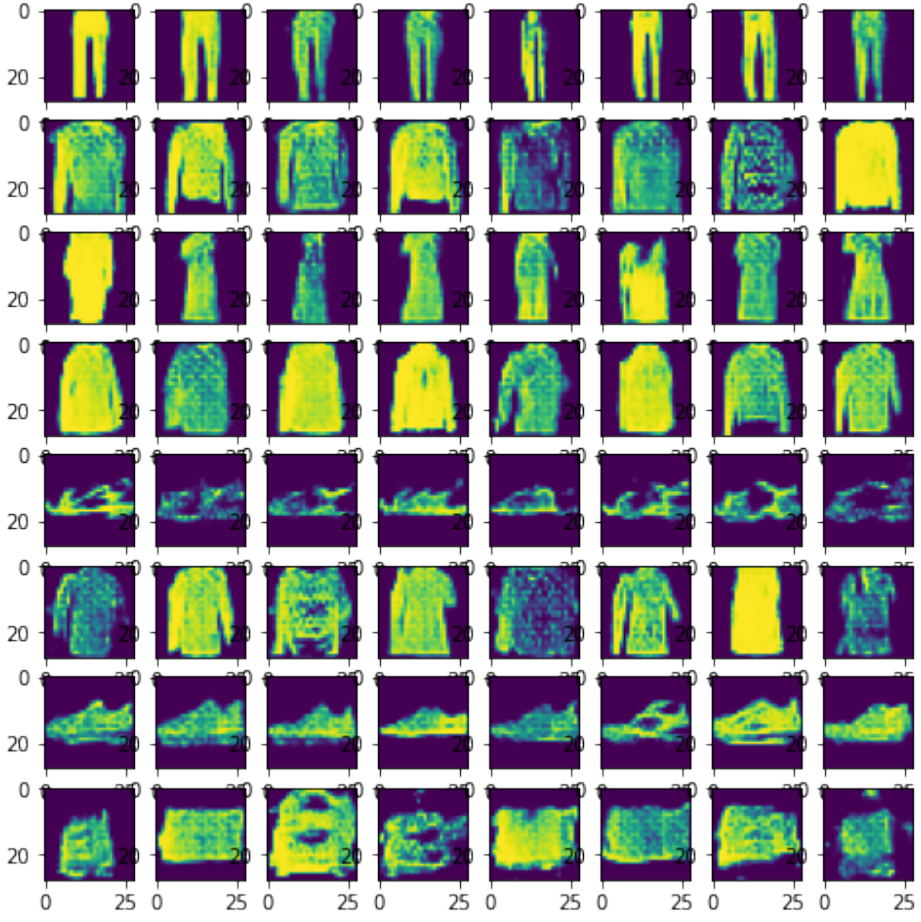

Figure: FashionMNIST, 20 epochs, each row corr. to one class query

Conditoning the GAN – Results

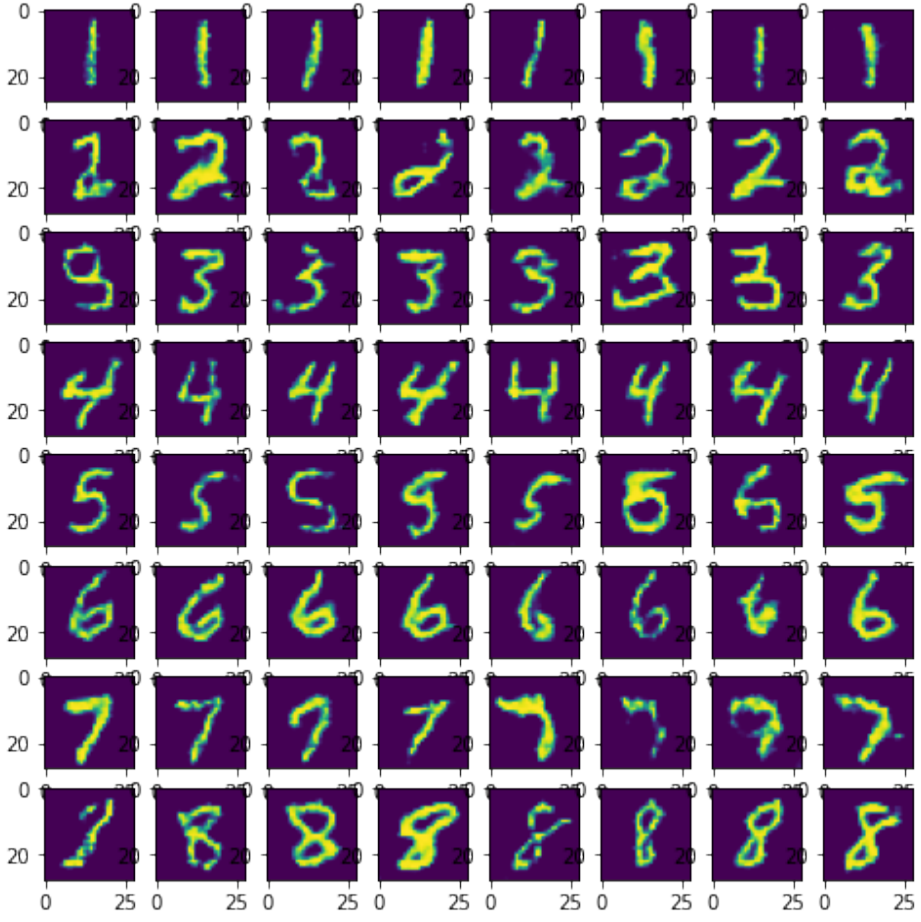

Figure: MNIST, 20 epochs, each row corr. to one class query

Conditioning the GAN – Observations

- We had to modify the architecture in code due to taking class label as input – explained in report.
- `FashionMNIST` trained faster than `MNIST`.
- We did not require additional batchnorm for `MNIST`.

Image Translation Using GANs

- After gaining basic understanding of GANs we decided to apply them in a practical scenario.
- We trained a condGAN to learn to colour anime characters given uncoloured images.
- The results on a sample image during training are shown after every 20 epochs.

Image Translation Using GANs – Results

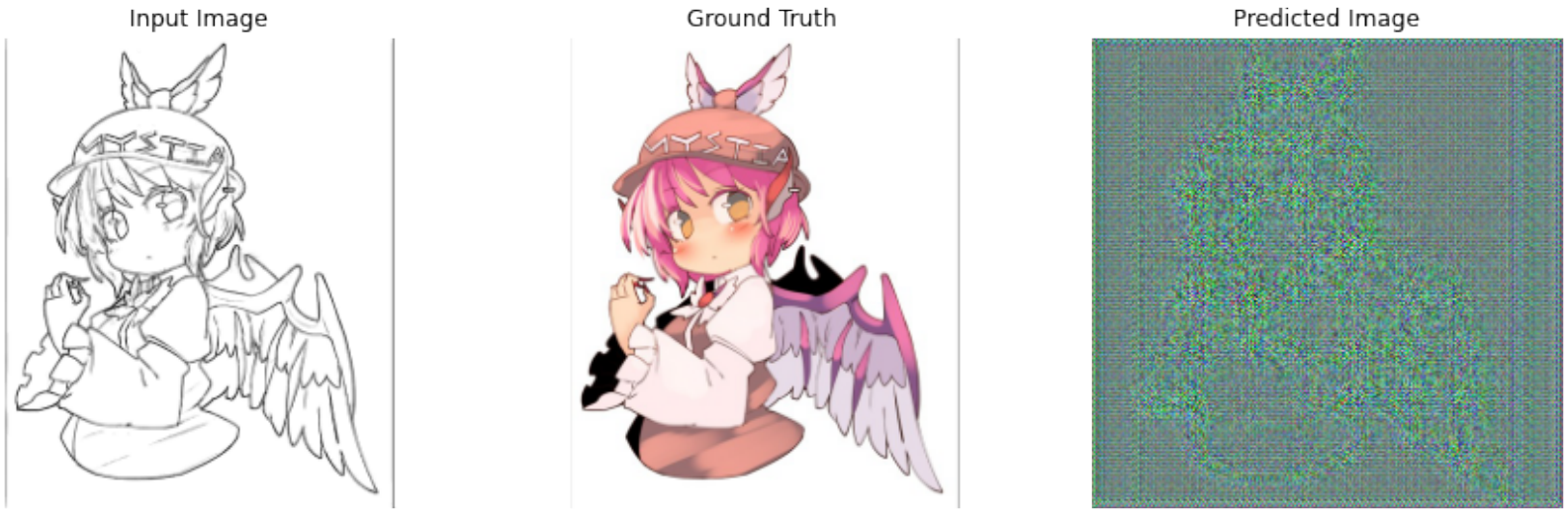

Figure: Result : 0 epochs

Image Translation Using GANs – Results

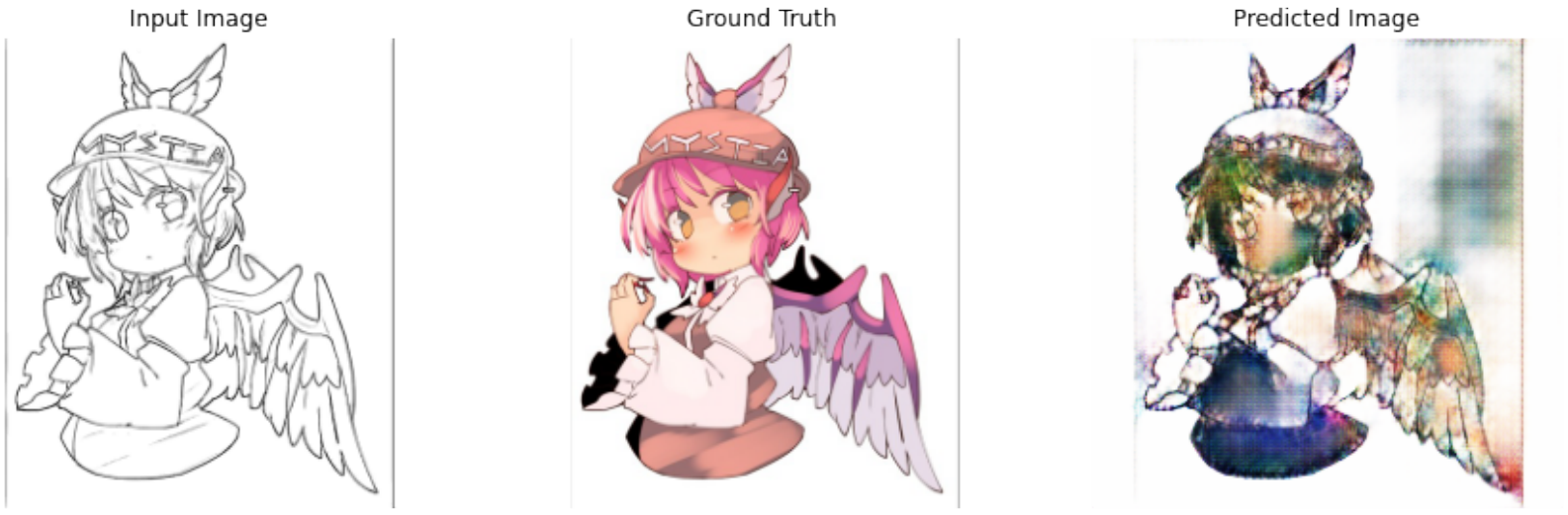

Figure: Result : 20 epochs

Image Translation Using GANs – Results

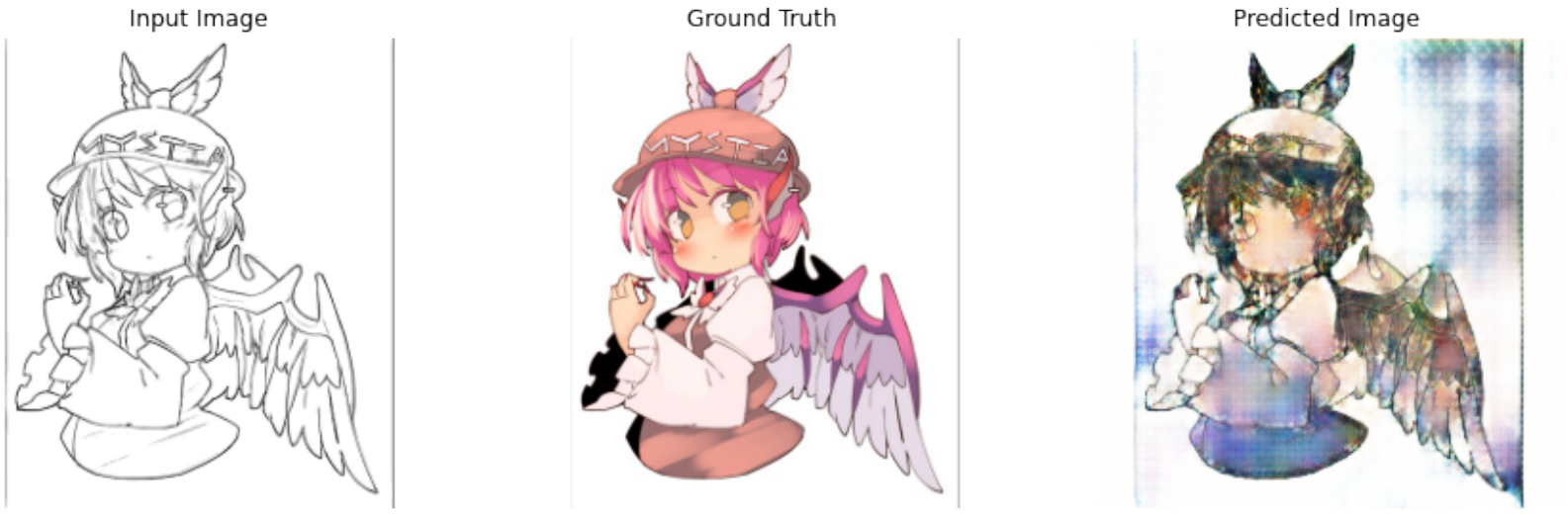

Figure: Result : 40 epochs

Image Translation Using GANs – Results

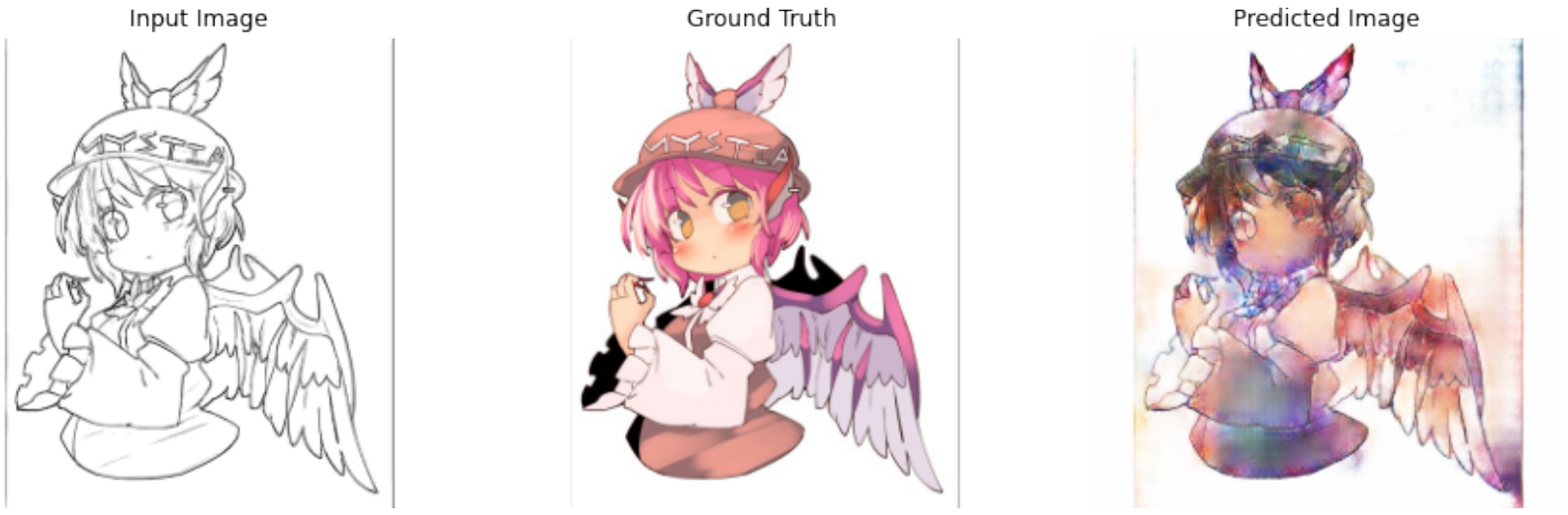

Figure: Result : 60 epochs

Table of Contents

Contribution

- Siddhant Midha (200070078)
  1. Theory part of report
  2. Presentation slides
  3. Wrote code and trained for basicGAN
  4. Wrote code and trained for convGAN
  5. Modified the convGAN code to form condGAN and trained
- Waqar Mirza (200070090)
  1. Results part of report.
  2. Trained the Anime Colourization GAN.
  3. Presentation slides - Image Translation using GANs
- Dadhichi Telwadkar (20D070083)
  1. Code for Anime Colourization GAN.
- Yuvraj Singh (200070093)
  1. Code for Anime Colourization GAN.

Code Sources

- Referred to the GAN course on coursera for basic GAN theory and code organization.
- Referred to the `Gradient Tape` and GAN tutorials by keras.
- (None of the sources above involved direct use of code.)
- Referred to this tutorial for the Anime Colourization.