Ghiblify! Cartoonizing Images Using GANs

Loh Chuan Hui

General Assembly

DSI 24

Content

01 Introduction

Anime artists painstakingly recreate real-life scene through hand-drawn sketches for full-feature anime film. Could artificial intelligence become part of the anime creation workflow?

Project Objective

- Explore how GANs can assist in cartoonizing real-life images for anime artistes and serve as a fun application for social network apps

- Understand how to optimize GAN hyperparameters and architecture for the best results for cartoon style transfer

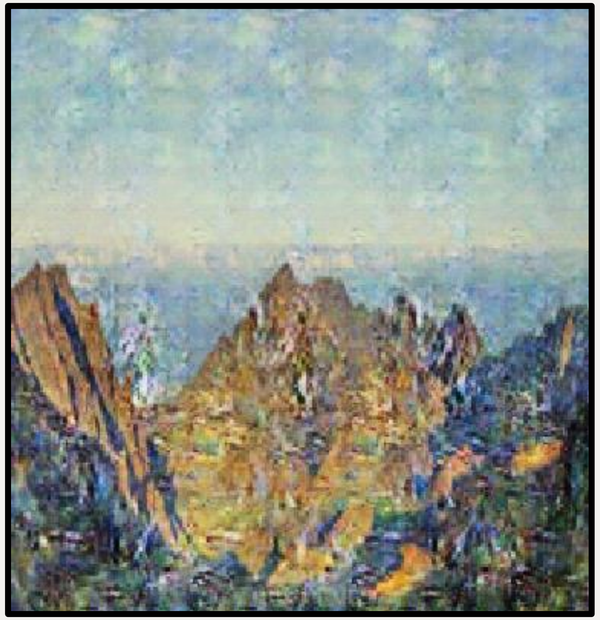
Style Transfer

Content
Mountain

+

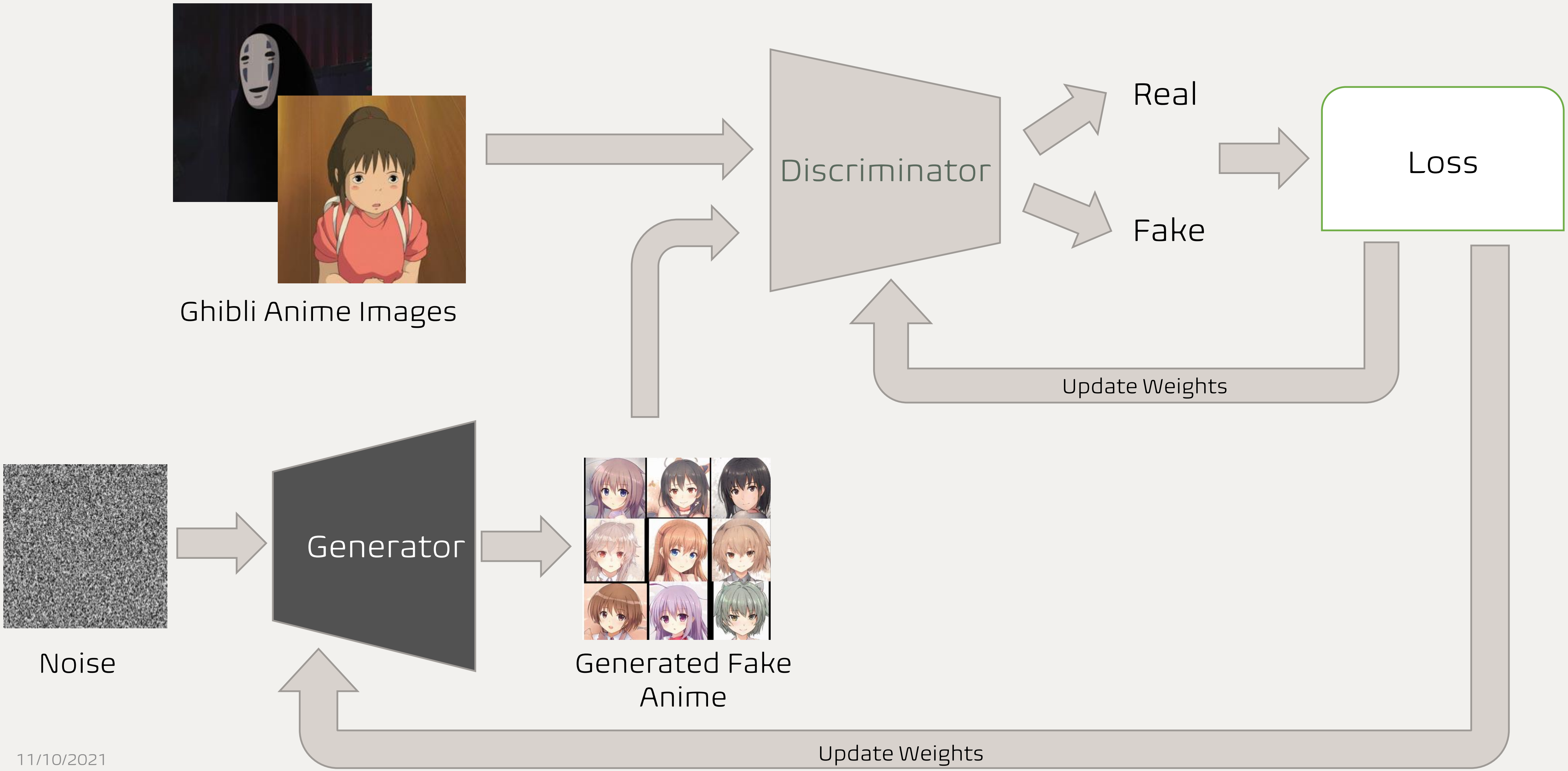
Style
Monet
Painting

=



Target

Generative Adversarial Networks



02 Analysis Methodology

Image Curation

- Real-Life Images
- Anime Images
- Images Themes

Image Pre-processing

- Image Size
- Image Aspect Ratio

Train GAN Models

- Basic GAN
- CycleGAN
- Unsupervised

Select Best Models

- Tune Hyperparameter
- Adjust architecture
- Qualitative review of generated anime

03 Images Data

Image Pre-processing

- All images have been cropped to 1:1 aspect ratio and are either 256x256 or 512x512 in size

Sample Images – Real-Life

6387 images
Theme: Mixed
Source: Unsplash

1976 images
Theme: Architecture
Source: Kaggle

Sample Images – Real-Life

3318 images
Theme: Asian Female Portraits
Source: Kaggle

11 images
Theme: Asian Female Portraits
Source: generated.photos

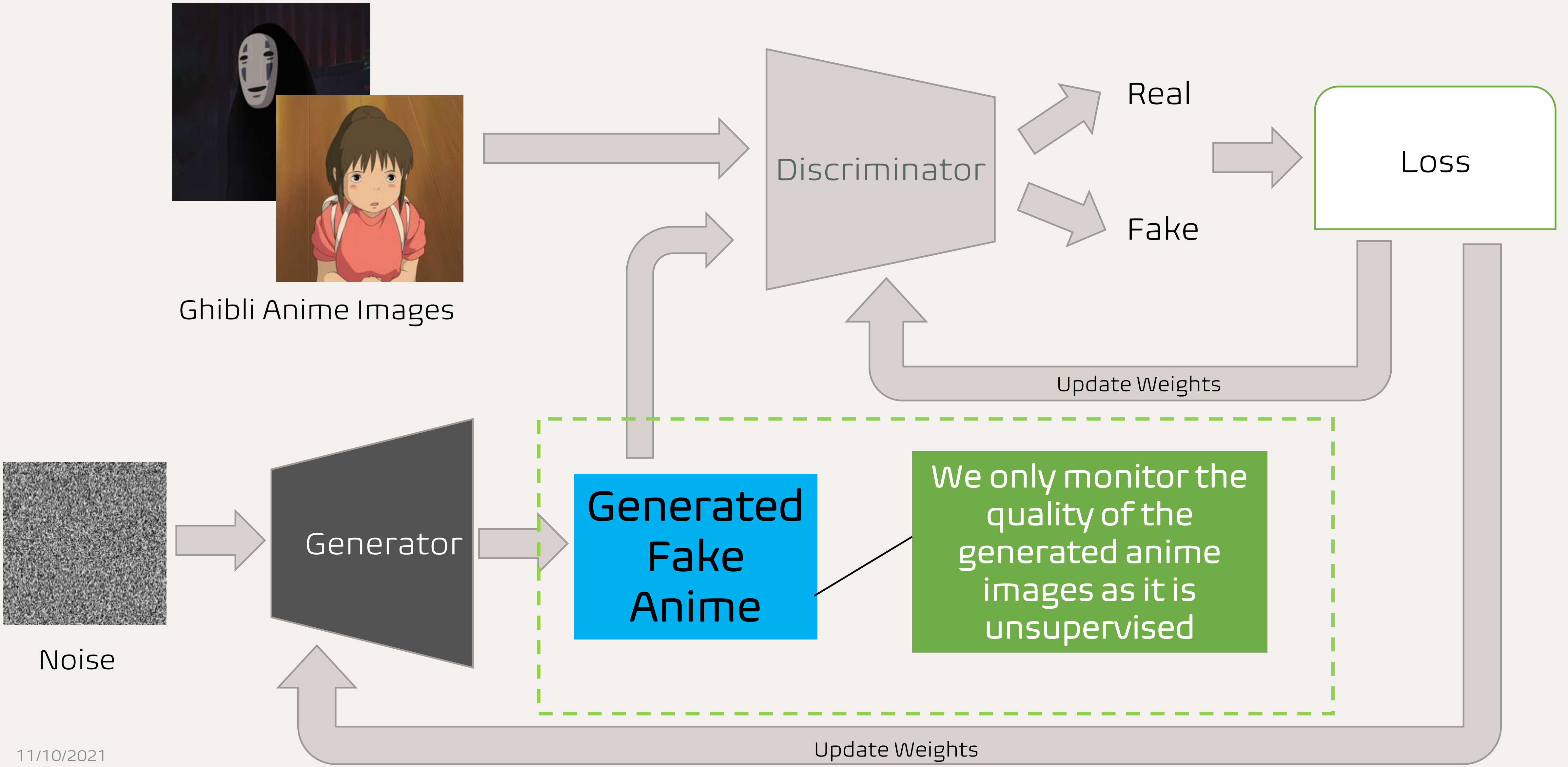
Sample Images – Anime

6079 images
Theme: Mixed, Ghibli
Source: Netflix

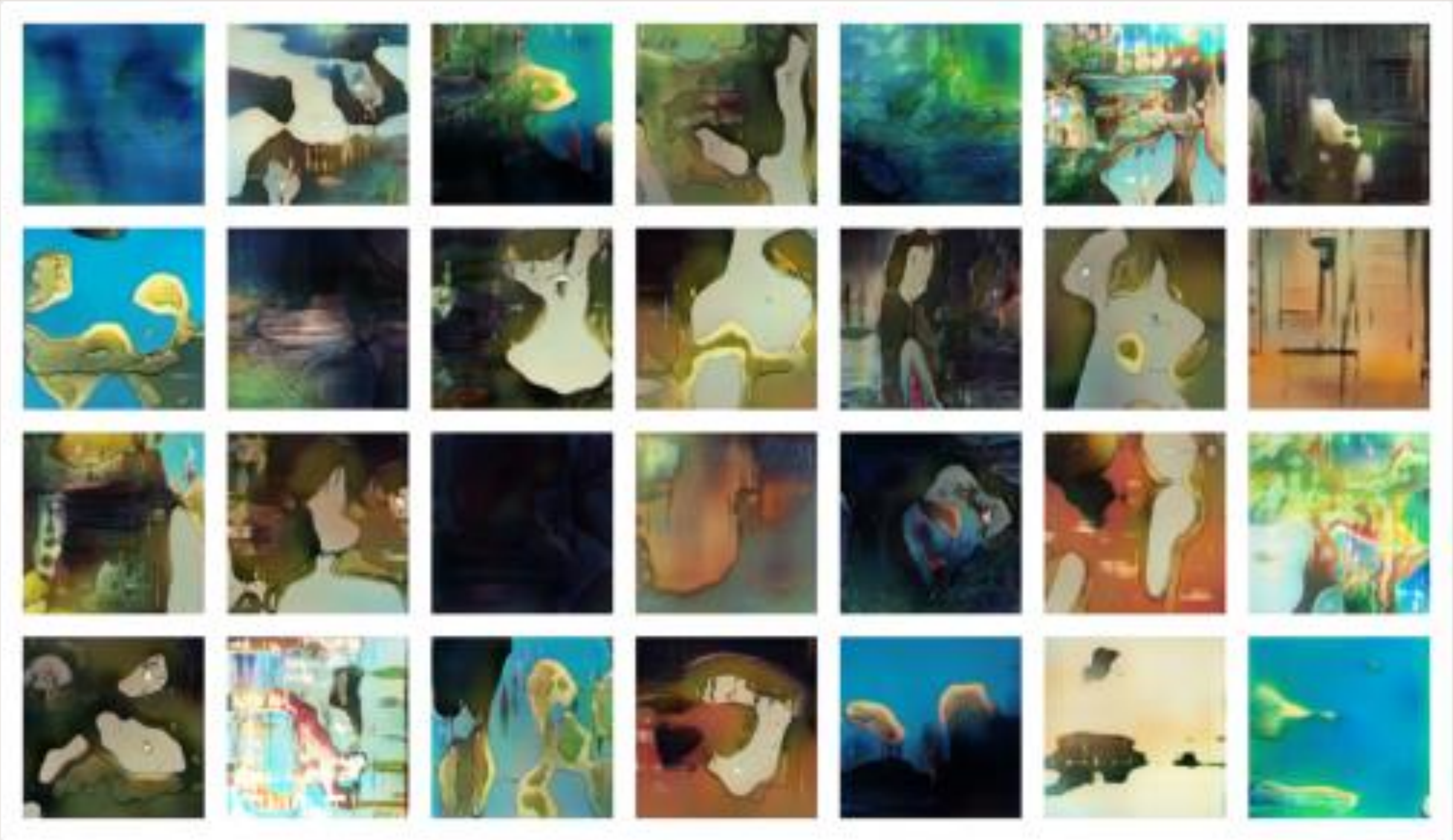
3400 images
Theme: Female Portrait
Source: Kaggle

03 Basic GAN

Basic GAN with Noise Input



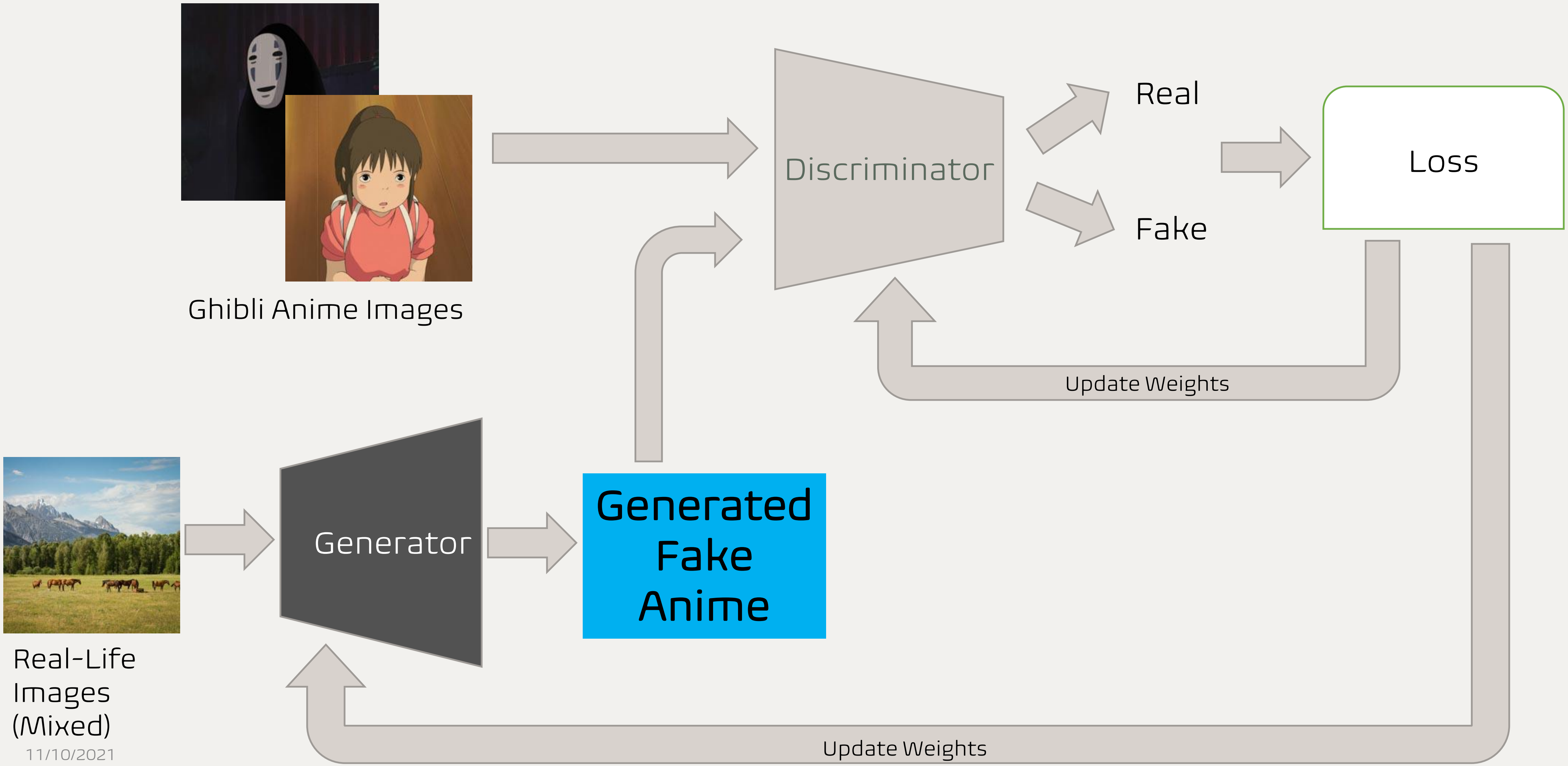
Generated Anime



Basic GAN

- Generated Anime images has the Anime style but with **no clear content**

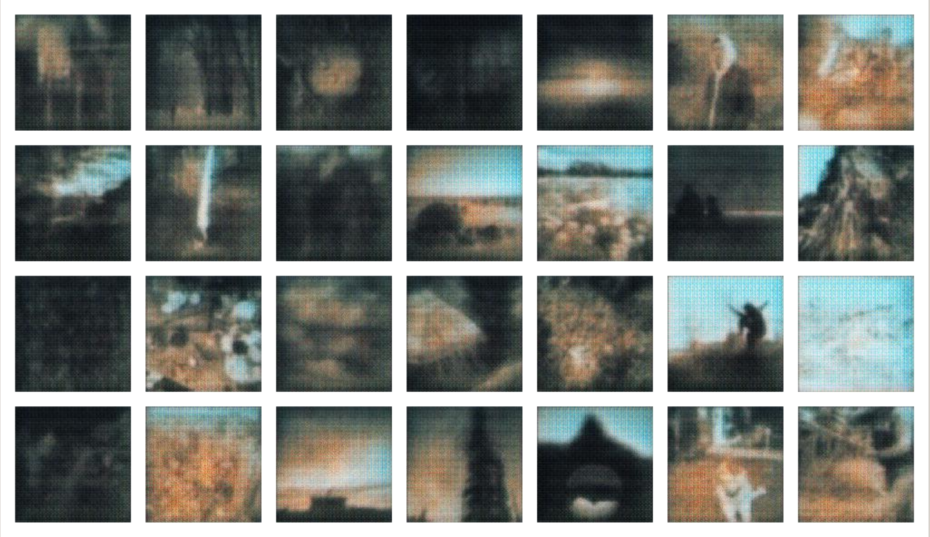
Basic GAN with Real Image Input



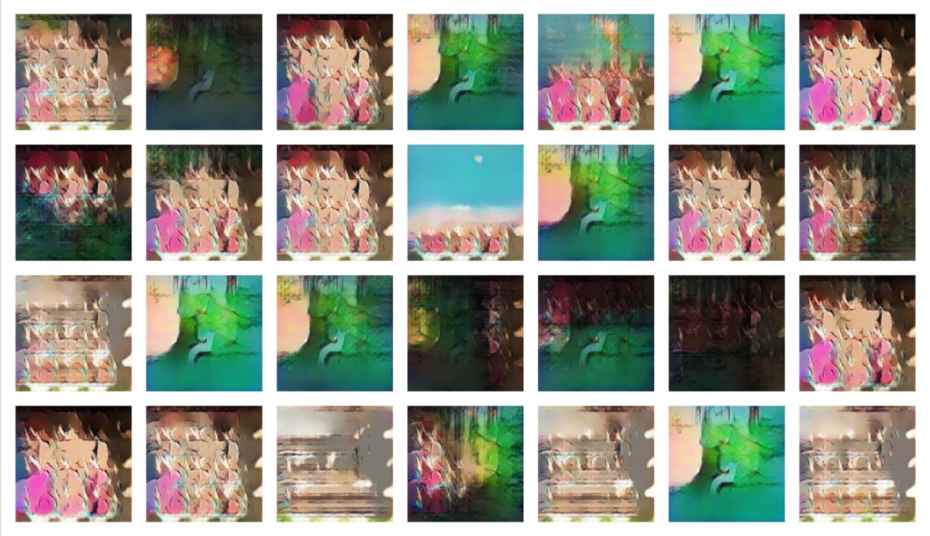
Generated Anime

Basic GAN with Real-Life Image Inputs

- Content clearly visible in early epochs but were lost after 200 epochs
- Basic GAN does not have loss function to retain content
- Another GAN model required which transfer style and retains content



Generated Anime at 1st Epoch

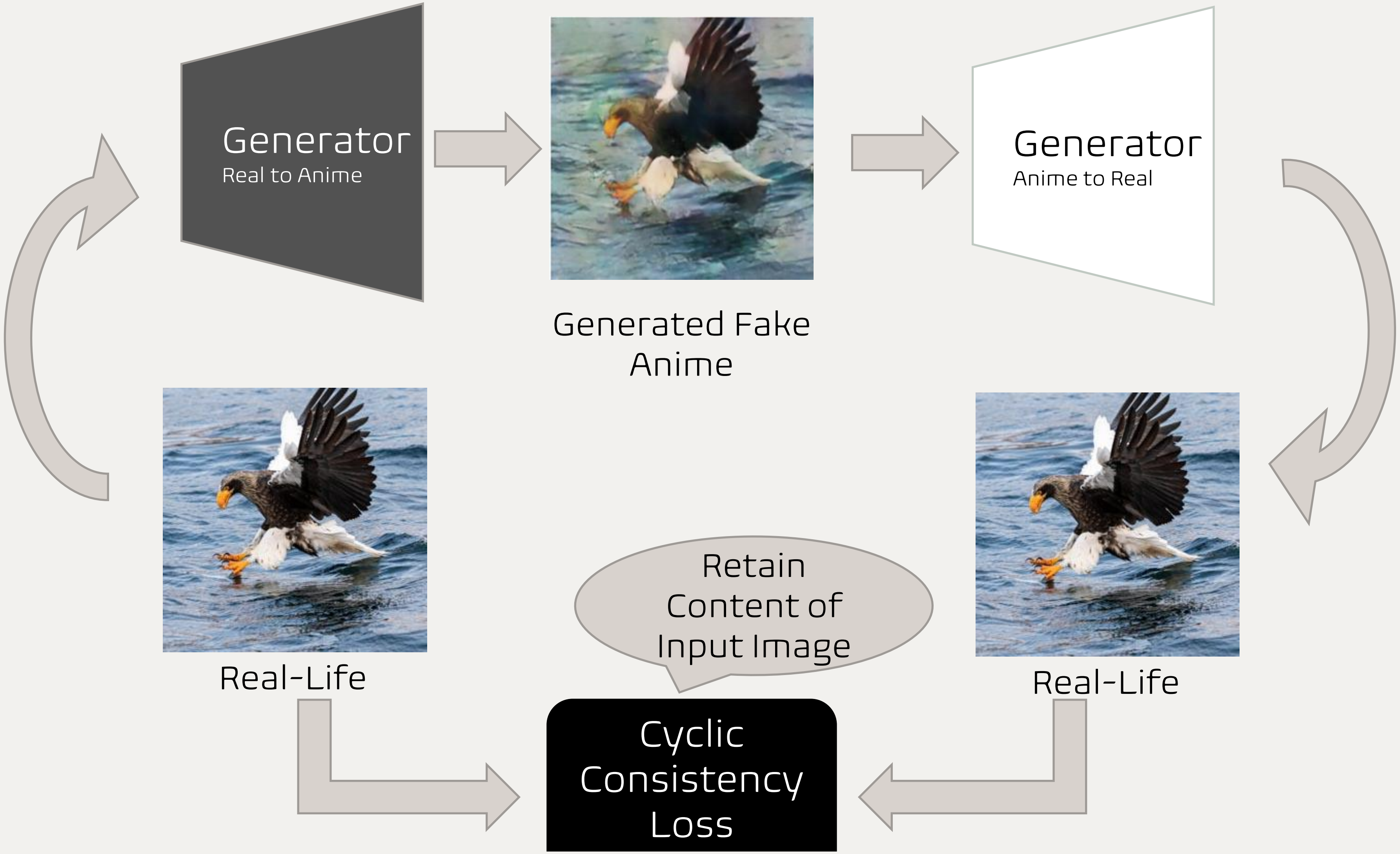


Generated Anime at 200th Epoch

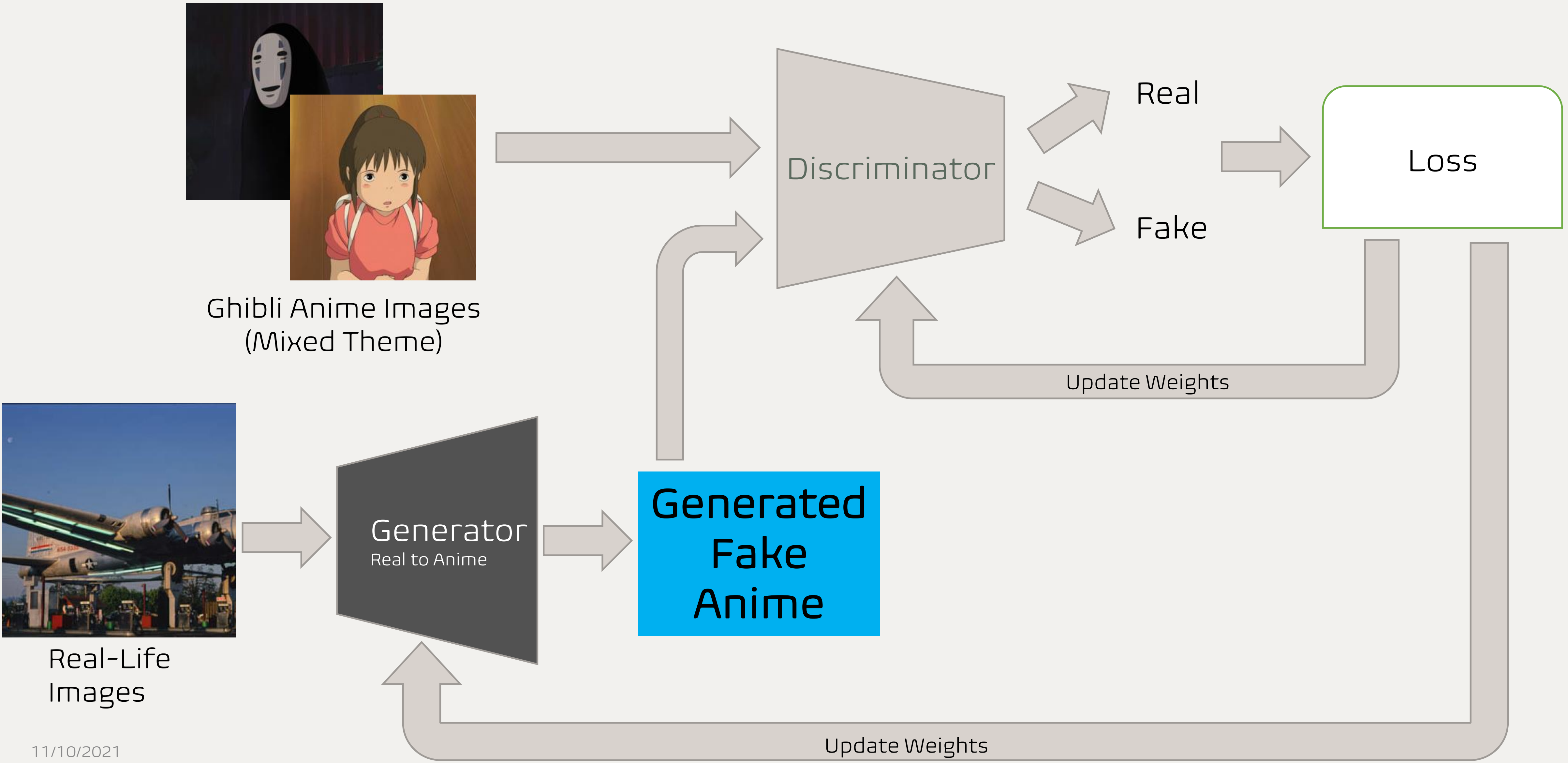
04

CycleGAN

Introduction to CycleGAN



CycleGAN with Mixed Theme Anime

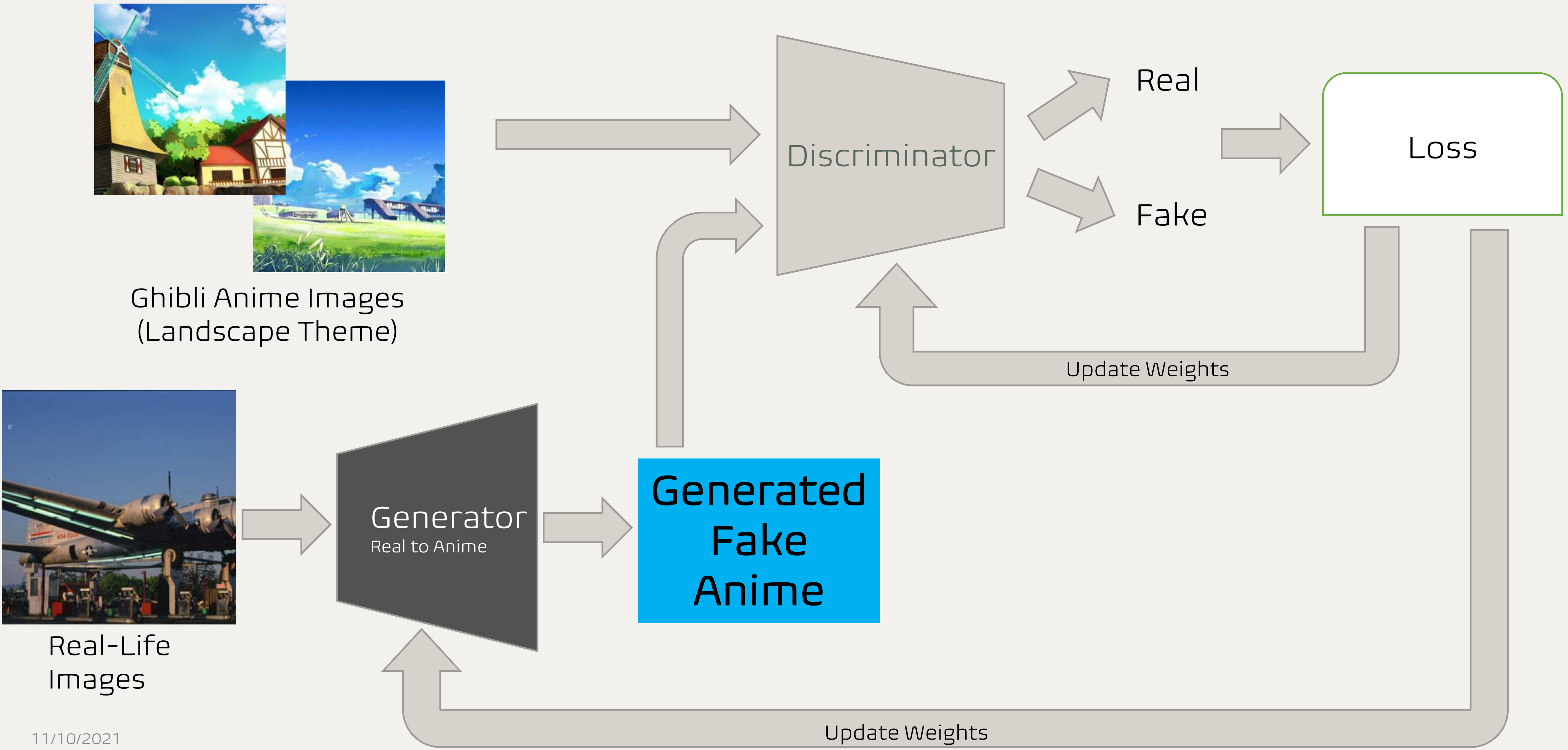


Generated Anime

CycleGAN with Mixed Theme Anime

- Anime style successfully transferred on most images
- The model generated incoherent patterns on some images with blue skies
- **Anime training data set needs to be clean and of common theme**

CycleGAN with Curated Theme Anime



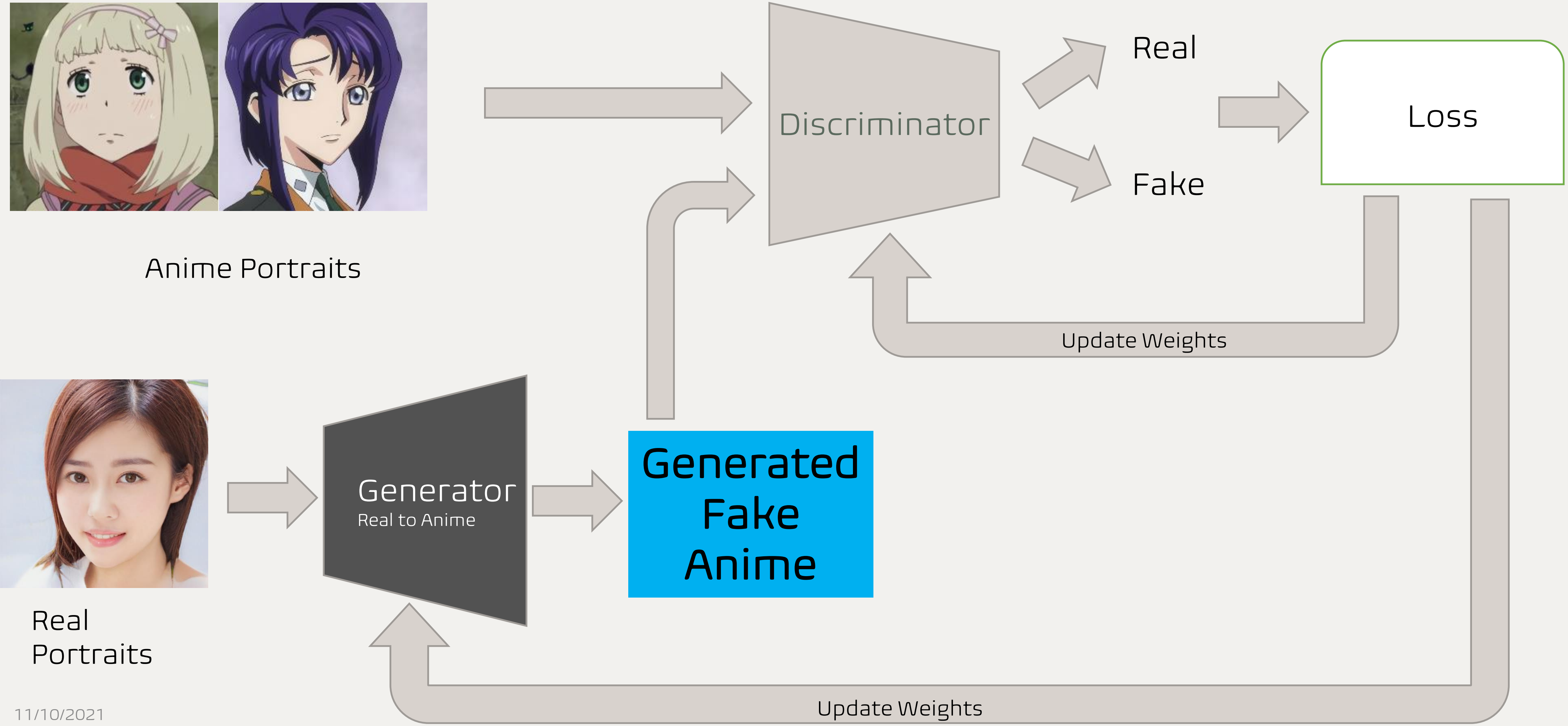
Generated Anime

CycleGAN with Curated Theme Anime

- Model do better on blue skies and may even fill clear blue skies with clouds
- Less incoherent details in generated images
- CycleGAN does not need paired images, but matching themes for input and target domains improve performance



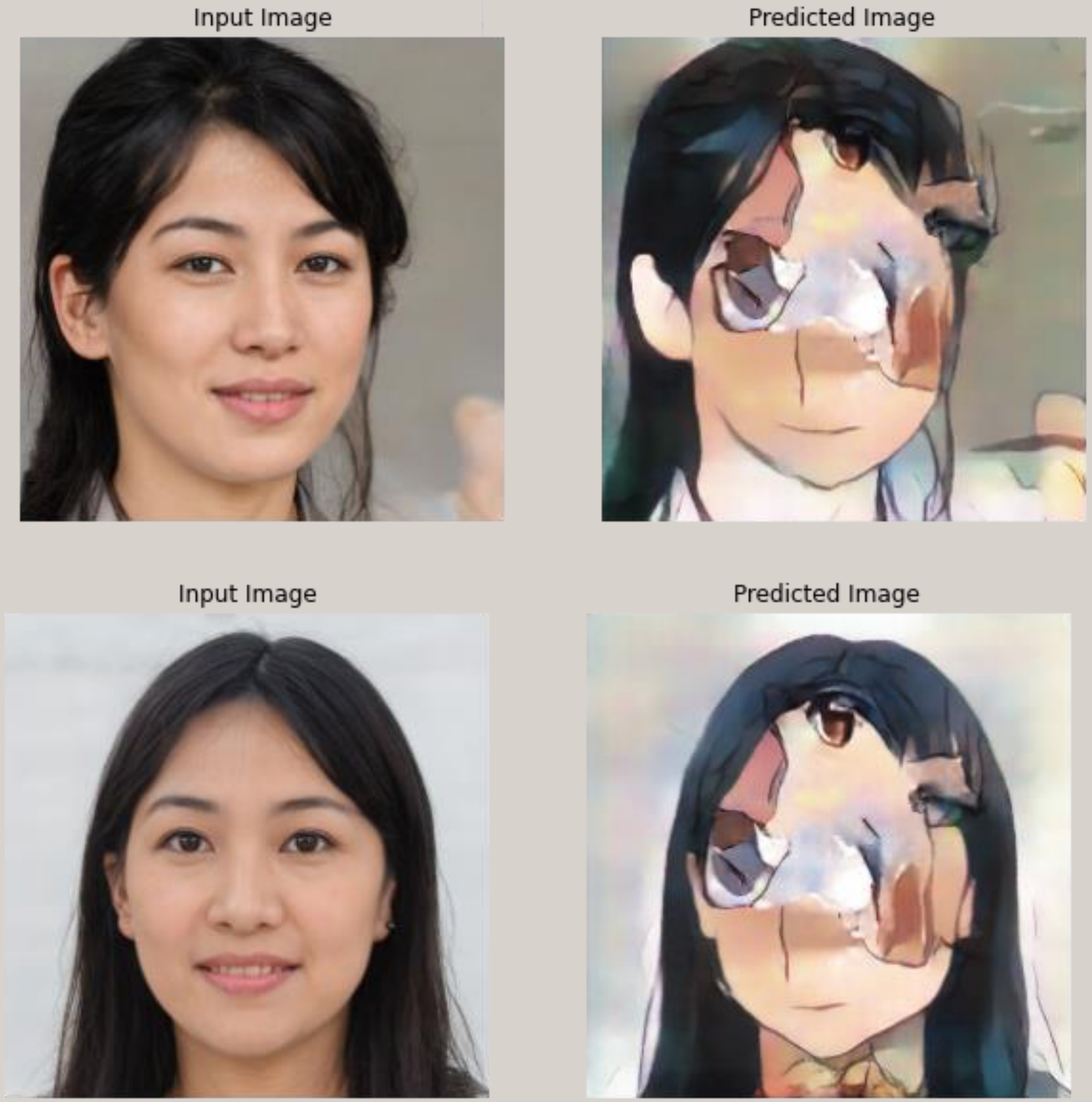

CycleGAN with Anime Portraits



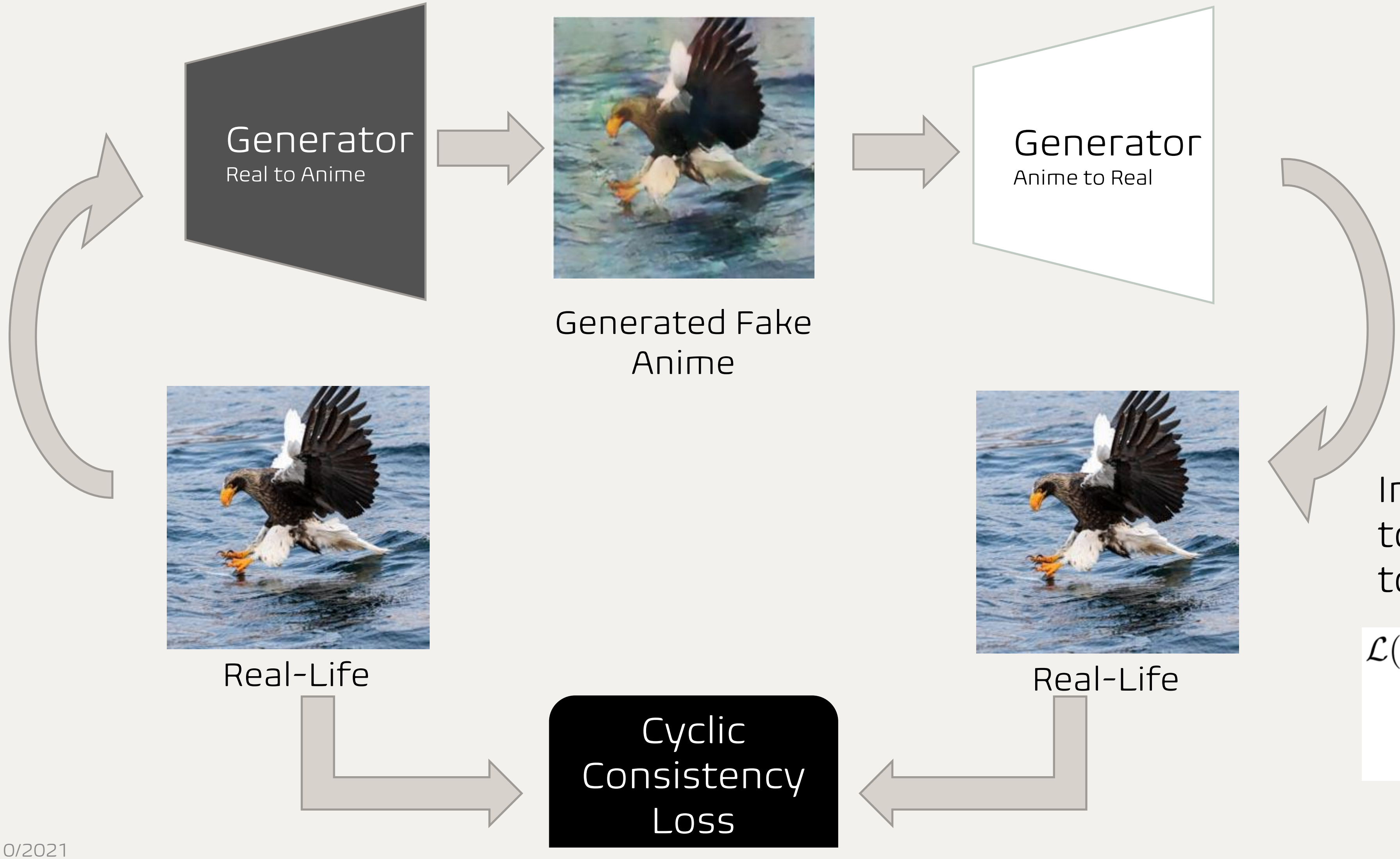
Generated Anime

CycleGAN with Anime Portraits

- Model did well with hairstyle but got progressively worse with facial features. Training was stopped early at 60^{th} epoch
- Increase the value of LAMBDA as possible solution so that cyclic consistency loss is given higher importance
- Theoretically guide the model to prioritize the position of the facial features and reduce distortion



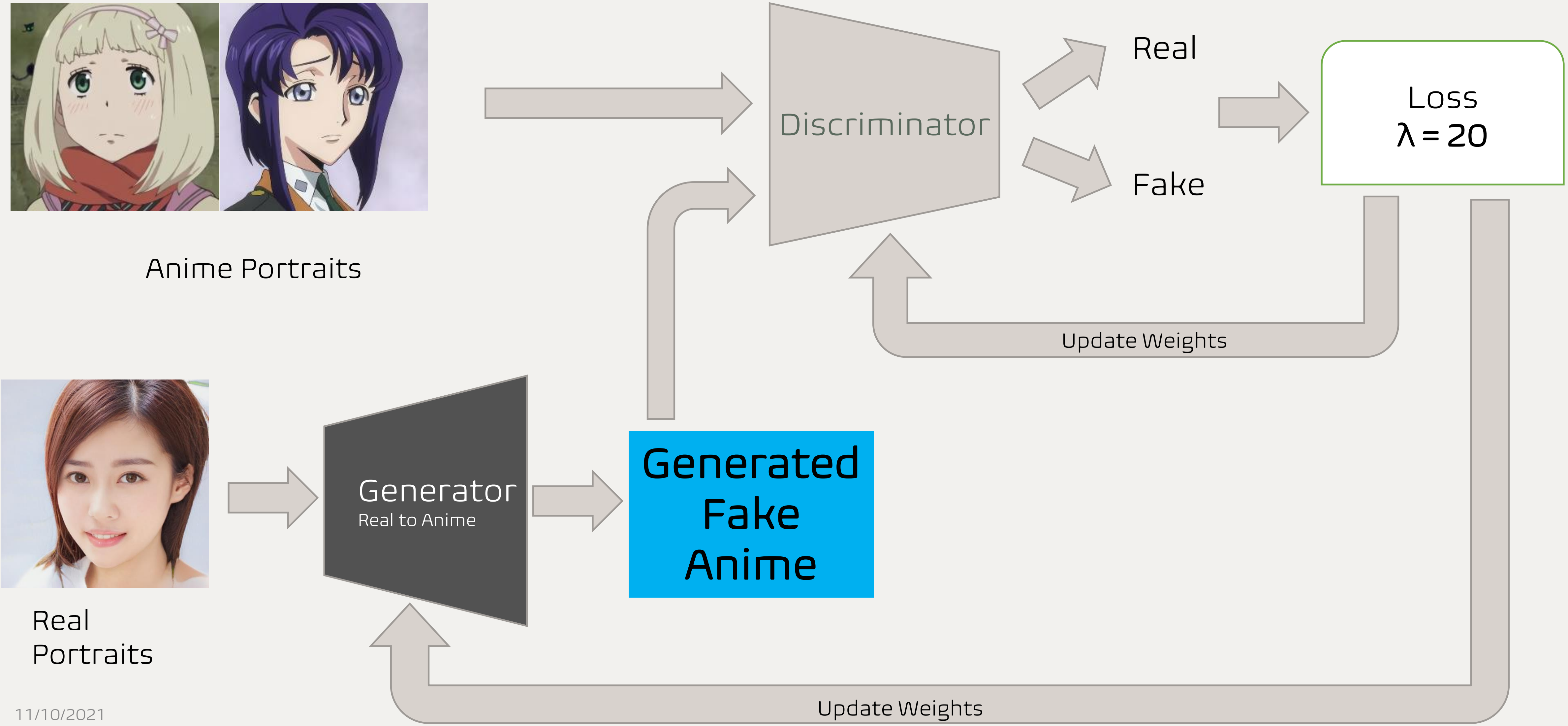
CycleGAN with LAMBDA = 20



Increasing LAMBDA from 10 to 20 gives more importance to cyclic consistency

$$\begin{aligned}\mathcal{L}(G, F, D_X, D_Y) = &\mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) \\ &+ \mathcal{L}_{\text{GAN}}(F, D_X, Y, X) \\ &+ \lambda \mathcal{L}_{\text{cyc}}(G, F)\end{aligned}$$

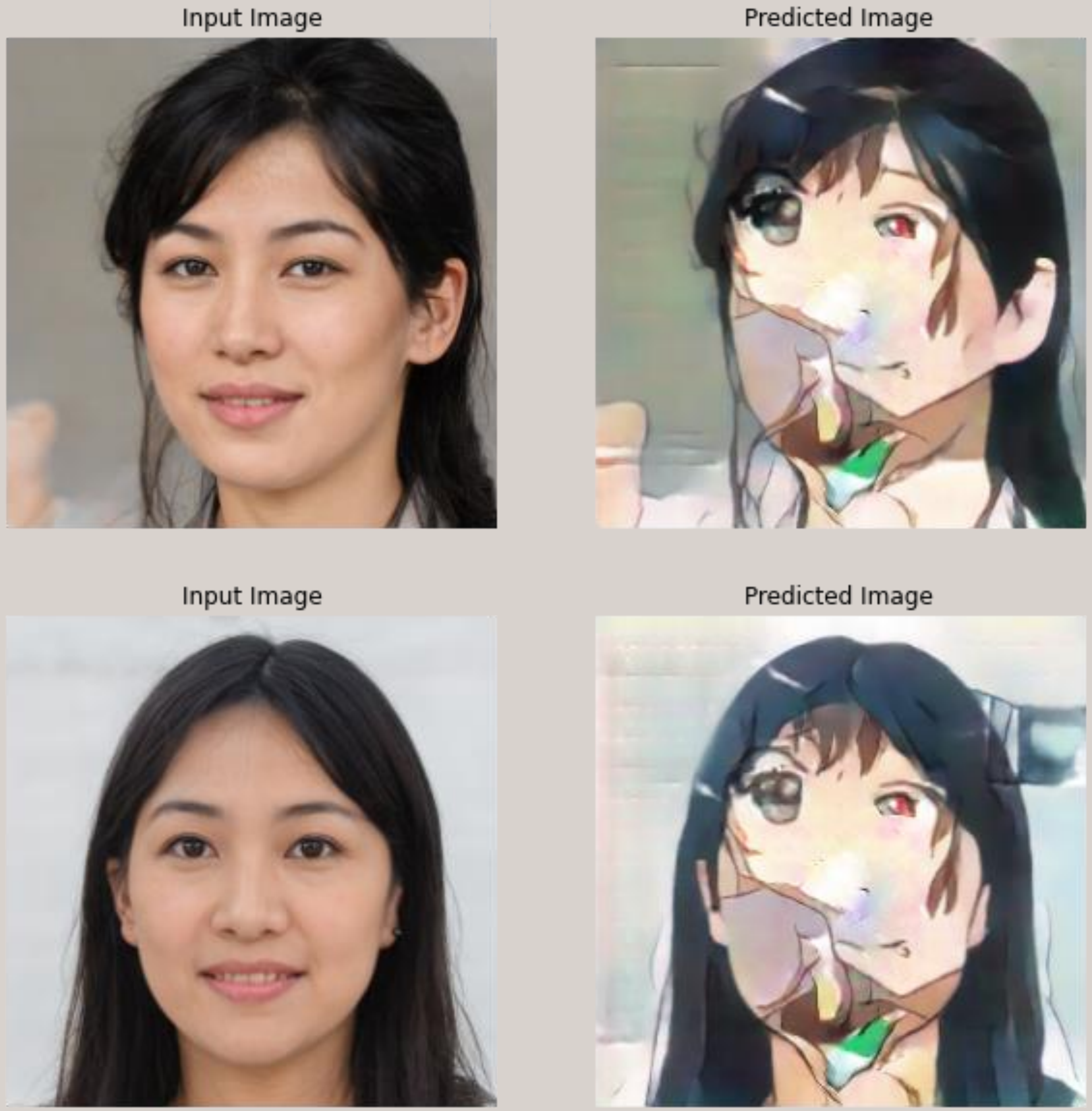
CycleGAN with LAMBDA = 20



Generated Anime

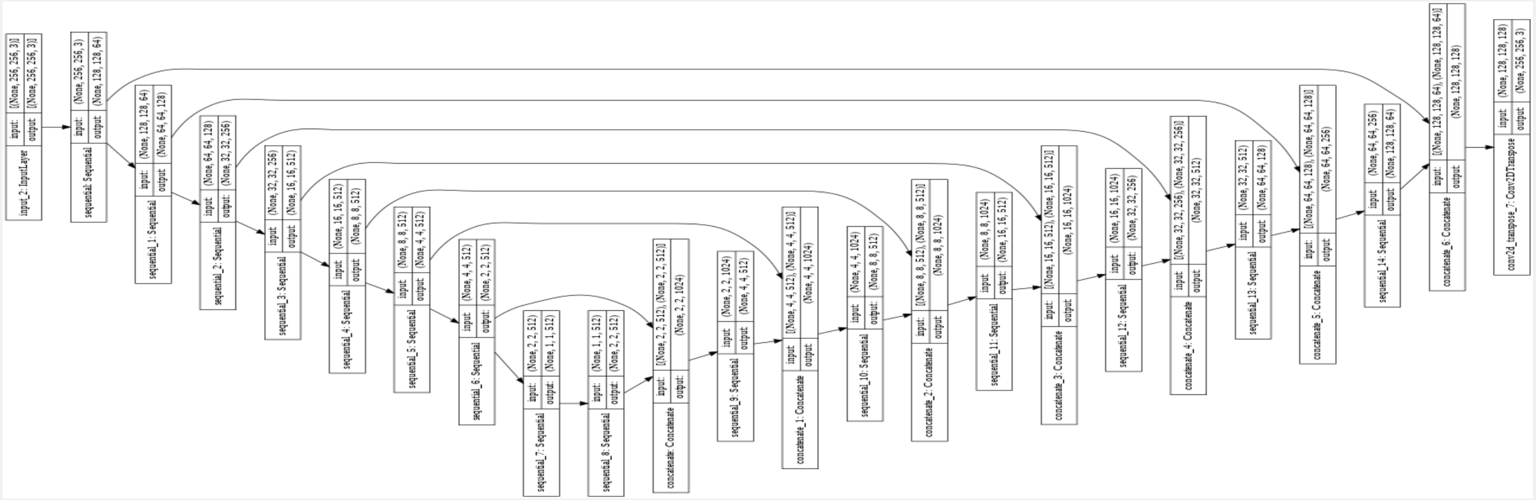
CycleGAN with LAMBDA = 20

- Model did badly for facial features right from the beginning.
- Training was ended early at 58th epoch
- **Problem may be with the CycleGAN architecture**



CycleGAN Generator Architecture

U-Net

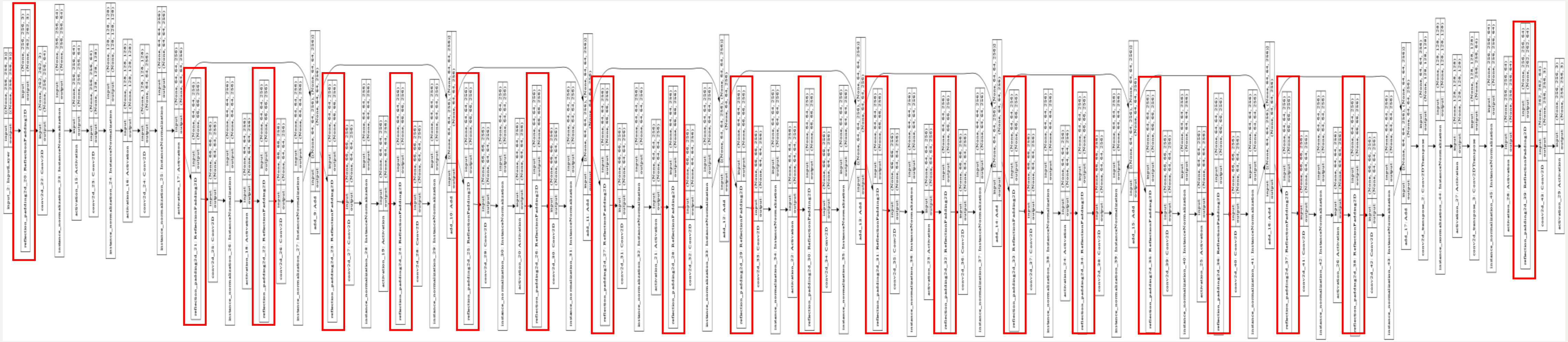


Model Summary

```
__________________________________________________________________________________________________
concatenate_4 (Concatenate)     (None, 32, 32, 512)  0           sequential_12[0][0]
                                                                 sequential_2[0][0]
__________________________________________________________________________________________________
sequential_13 (Sequential)      (None, 64, 64, 128)  590208      concatenate_4[0][0]
__________________________________________________________________________________________________
concatenate_5 (Concatenate)     (None, 64, 64, 256)  0           sequential_13[0][0]
                                                                 sequential_1[0][0]
__________________________________________________________________________________________________
sequential_14 (Sequential)      (None, 128, 128, 64) 147648      concatenate_5[0][0]
__________________________________________________________________________________________________
concatenate_6 (Concatenate)     (None, 128, 128, 128 0           sequential_14[0][0]
                                                                 sequential[0][0]
__________________________________________________________________________________________________
conv2d_transpose_7 (Conv2DTrans (None, 256, 256, 3)  3459        concatenate_6[0][0]
==================================================================================================
Total params: 30,618,691
Trainable params: 30,618,691
Non-trainable params: 0
__________________________________________________________________________________________________
```

CycleGAN Generator Architecture

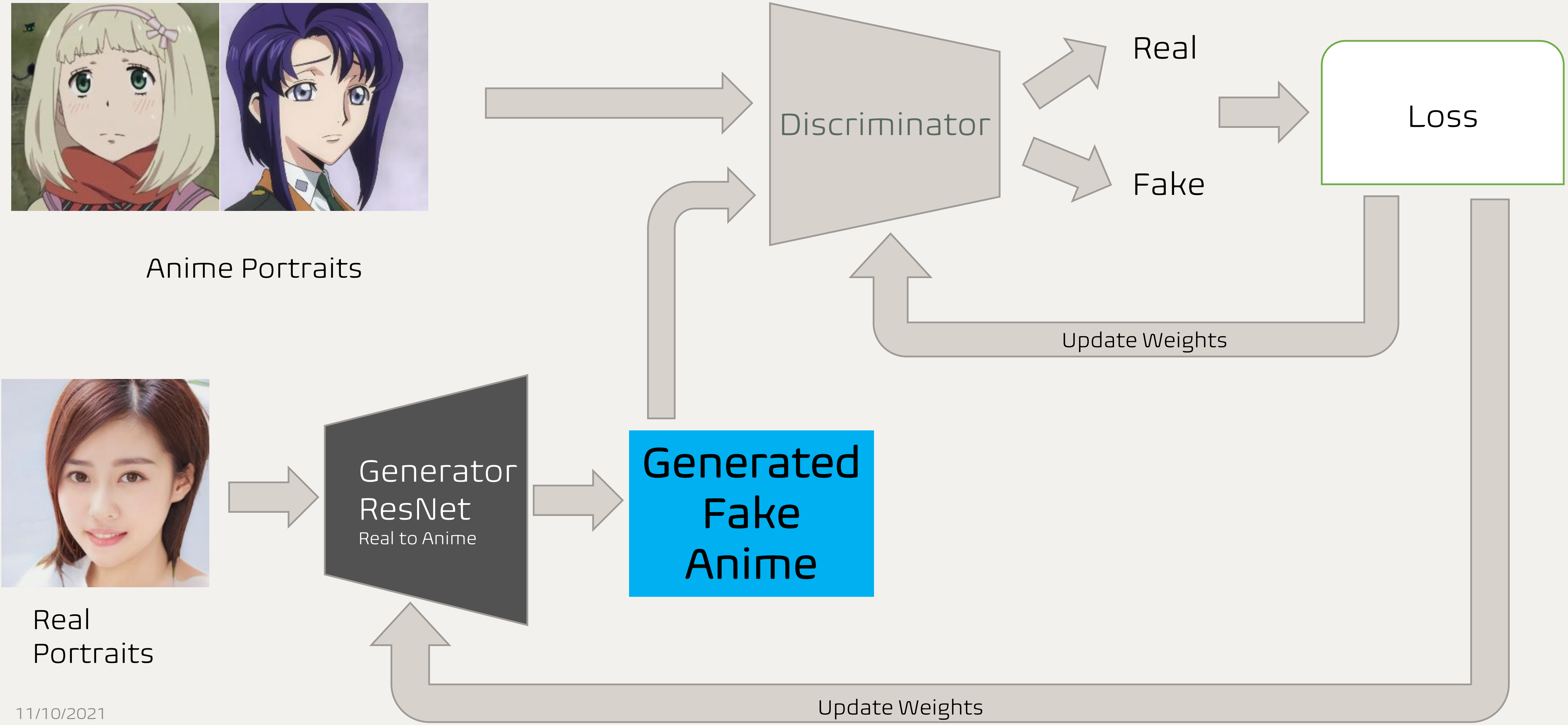
ResNet with ReflectionPadding2D



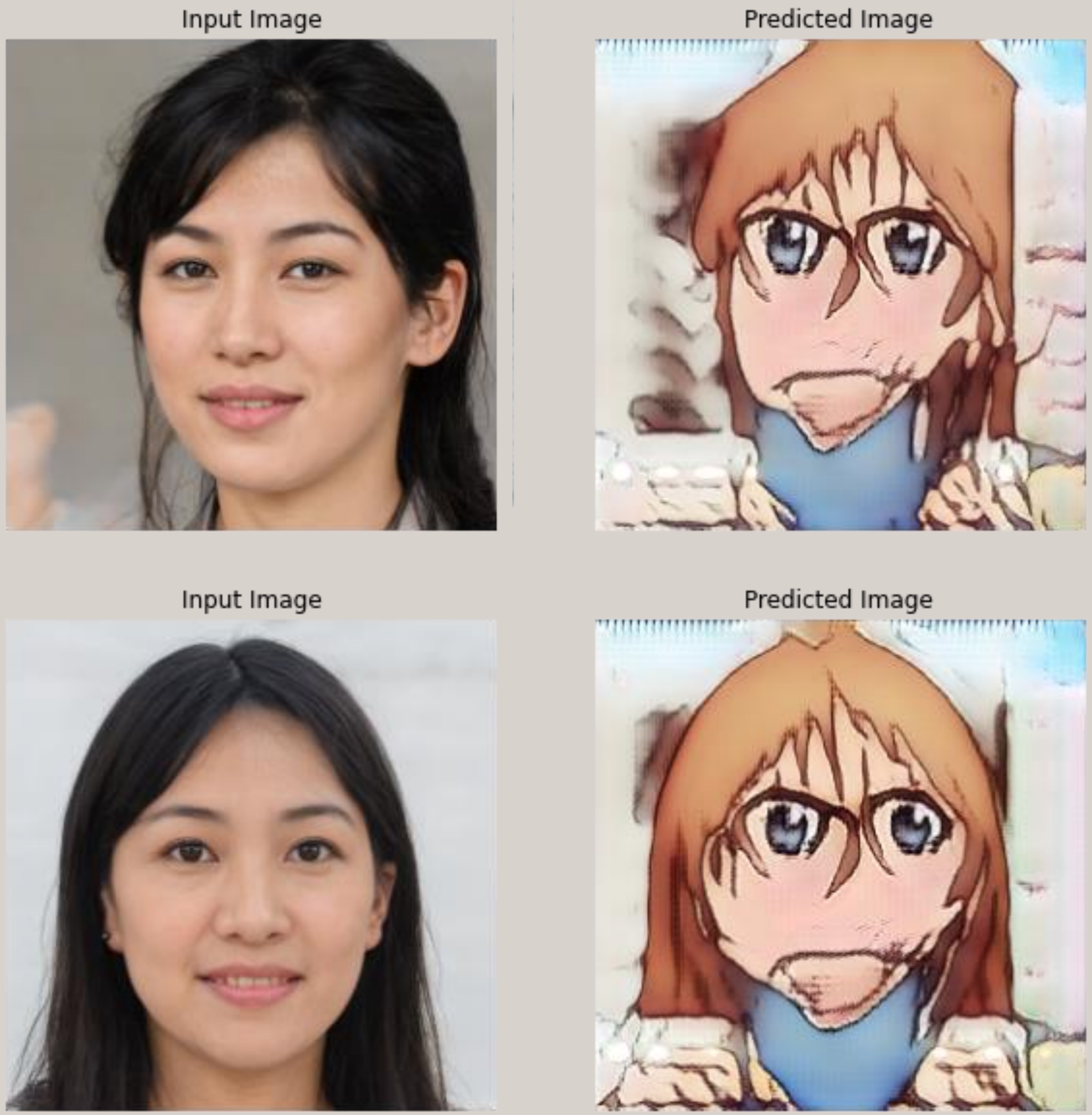
Model Summary

```
activation_28 (Activation)       (None, 256, 256, 64) 0          instance_normalization_45[0][0]
________________________________________________________________________________________________
reflection_padding2d_39 (Reflec (None, 262, 262, 64) 0          activation_28[0][0]
________________________________________________________________________________________________
conv2d_43 (Conv2D)              (None, 256, 256, 3)  9411       reflection_padding2d_39[0][0]
________________________________________________________________________________________________
activation_29 (Activation)      (None, 256, 256, 3)  0          conv2d_43[0][0]
================================================================================================
Total params: 11,388,675
Trainable params: 11,388,675
Non-trainable params: 0
```

CycleGAN with ResNet Generator



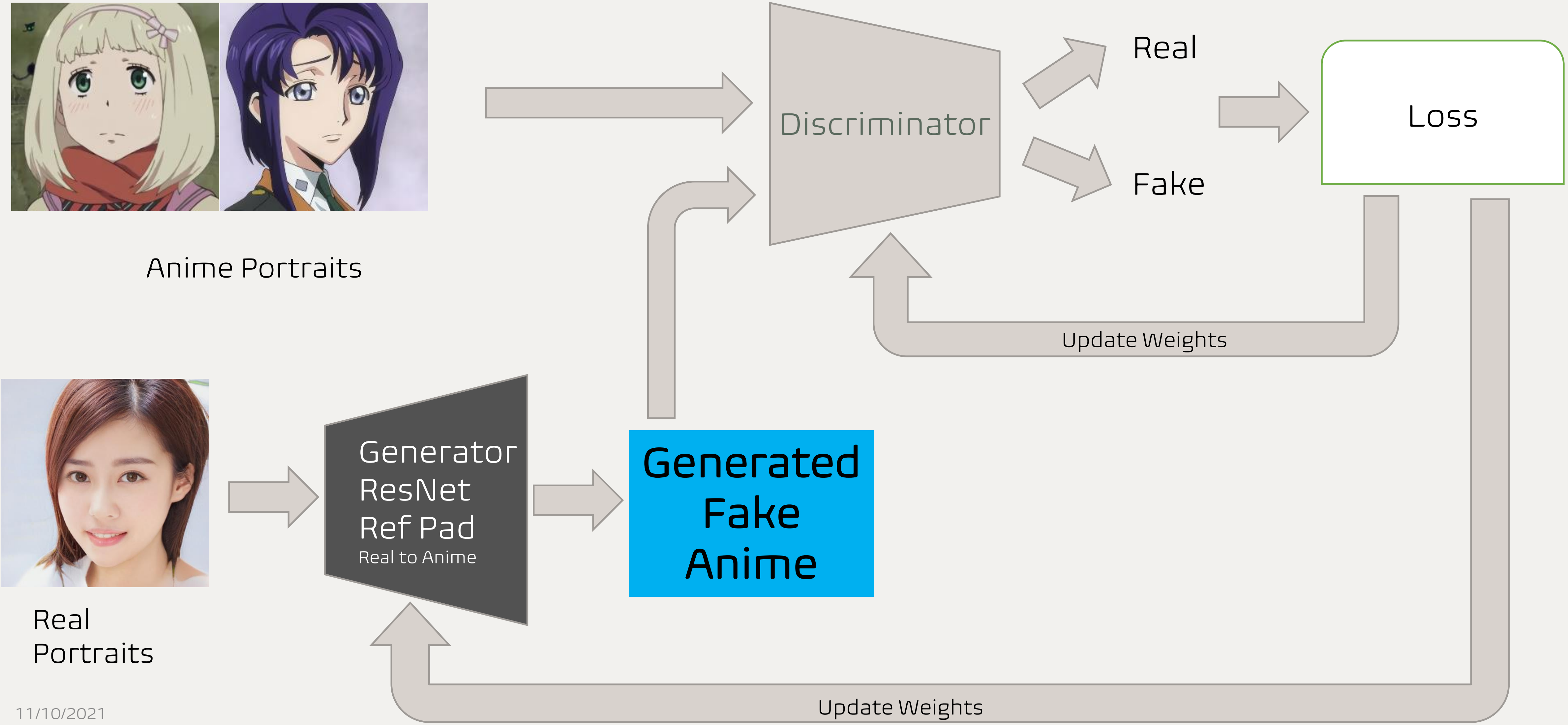
Generated Anime



CycleGAN with ResNet

- Model did well in facial features position
- Training was ended early at 58th epoch
- Quality is still lacking. Further adjustment of architecture required

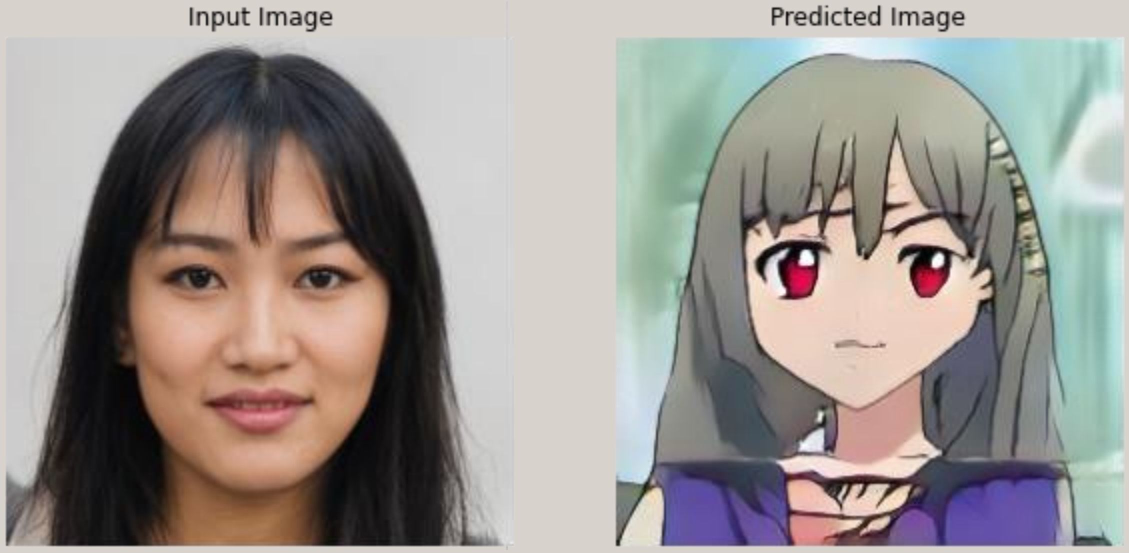
CycleGAN with ResNet Gen and Reflection Padding

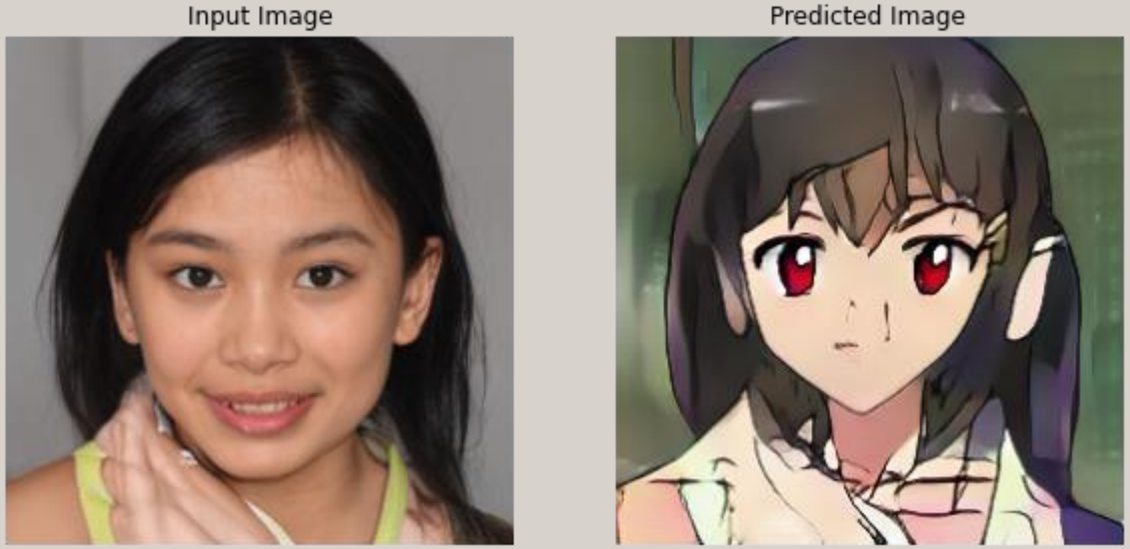


Generated Anime

CycleGAN with ResNet Gen and Reflection Padding

- Inclusion of ReflectionPadding2D improved quality of image tremendously
- Training was ended early at 100th epoch. Best result at 60th epoch
- Both content of input and anime style was successful transferred





Generated Examples

Input Image

Predicted Image

Input Image

Predicted Image

Input Image

Predicted Image

Input Image

Predicted Image

05 Key Takeaways

Landscape and scenery images can be cartoonized using CycleGAN with a U-Net Generator. Requires matching themes and clean real-life and anime datasets

Cartoonization of portraits is sensitive to position of facial features. ResNet may have performed better than U-Net due to fewer parameters and downsampling, which is better for color and style transfer

Cyclic Consistency and the CycleGAN architecture allowed input image content retention without paired data

ReflectionPadding2D helps to keep data distribution of input image intact vs Zero Padding, which may have helped performance

THANK YOU!

github.com/chuanhuiloh