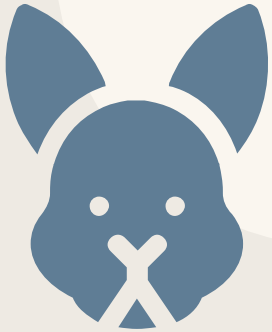







Cycle GAN

Nathalia Morales
Yuri Kaffaty
Fausto Holcombe
Christian Castillo

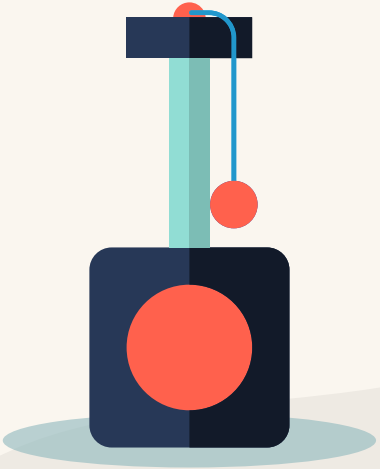
TABLE OF CONTENTS

INTRODUCTION

CycleGAN is a GAN of image-to-image translation.
Image-to-image translation is a **class of vision** and **graphics problems** where the goal is to l**earn the mapping** between an input image and an output image using a training set of **aligned image pairs.**

Learning to translate an image from a source domain X to a target domain Y in the absence of paired examples.

Our goal is to learn a mapping G: X → Y such that the distribution of images from the result is indistinguishable from the original image using an **adversarial loss.**



USE CASE AND WHY WE CHOSE IT

HOW DOES IT WORK?

WHY THIS DATASET?

It can be so **interesting** to convert a real life image into something else, such like zebras into horses and summer images to winter images, also the use of using paintings to real life images. We did have some **trouble** with our previous dataset but the selection of this model is so interesting to be able to **apply a certain style** to an image and make it seem so **realistic**.

WHAT IS THE DATA INPUT?

Pictures of horses, as long as the format of the picture is permitted, that can be the output. The algorithm later resizes the image to it's advantage.

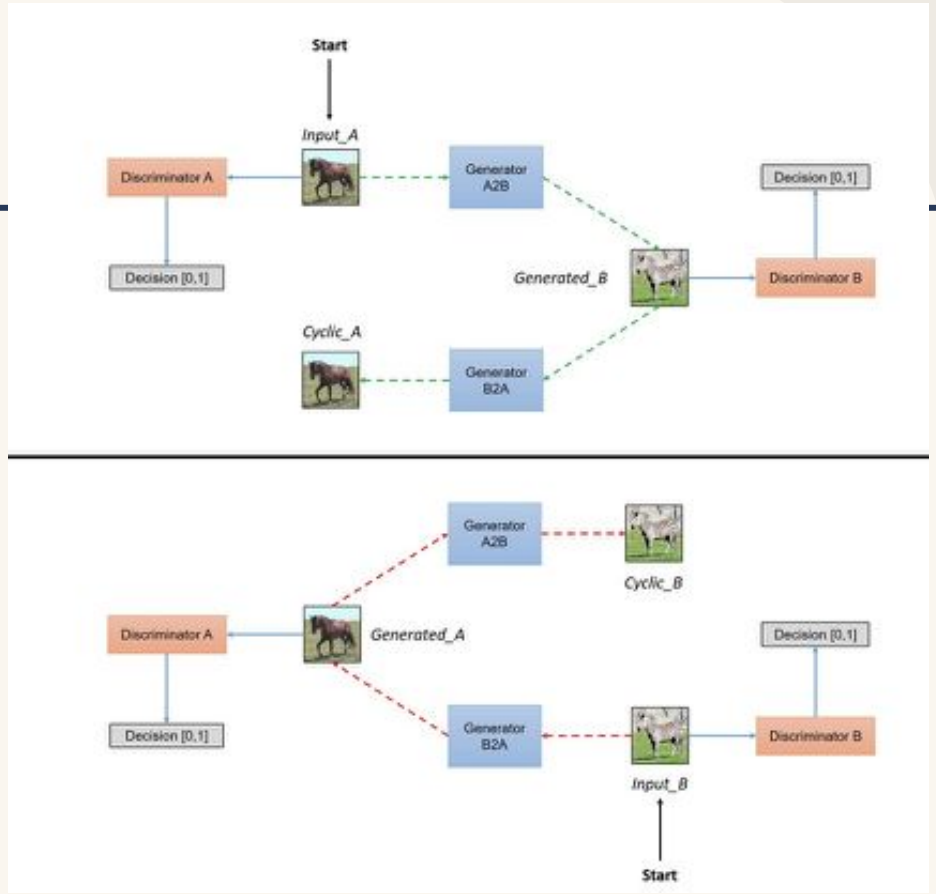
THE ARCHITECTURE

- We use **6 resnet blocks** for 128 × 128 images and **9 resnet blocks** for 256 × 256 and higher resolution training images.

- Each **resnet block** consists of a triple layer block: a convolutional layer, a normalization layer and a non-linear layer (ReLU).

- We use **instance normalization**. For the **discriminator networks** we use 70 × 70 **PatchGANs** [22, 30, 29], which aim to classify whether 70 × 70 overlapping image patches are **real or fake**. Such a patch-level discriminator architecture has **fewer parameters** than a full-image discriminator and can work on arbitrarily-sized images in a convolutional way.

- The **model** is here: https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix/blob/master/models/networks.py


Start
Input_A
Discriminator A
Generator A2B
Decision [0,1]
Decision [0,1]
Generated_B
Discriminator B
Cyclic_A
Generator B2A
Generator A2B
Cyclic_B
Discriminator A
Generated_A
Decision [0,1]
Decision [0,1]
Generator B2A
Discriminator B
Input_B
Start

How good is the data generated?

Does it make sense?

Can you tell it's from the dataset you used?

How so?

WELL LET'S SEE!

Imagen Generada

Imagen Original

Imagen Generada

Imagen Original

Imagen Generada

Imagen Original

Imagen Generada

Imagen Original

Imagen Generada

Imagen Original

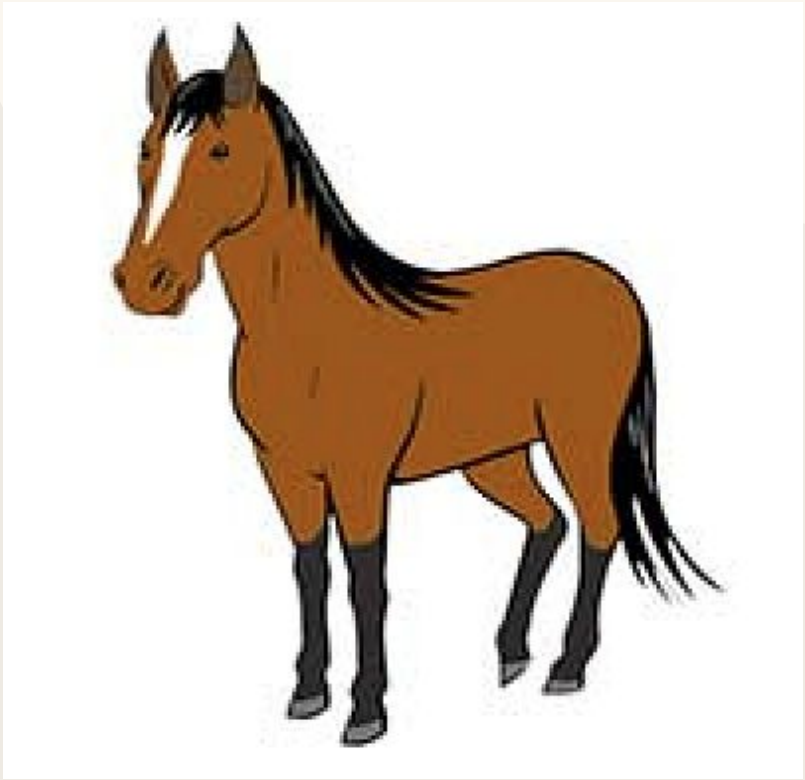


Imagen Generada

Imagen Generada

Imagen Original

Imagen Generada

Imagen Original

CONCLUSIONES

Link al demo: https://ai-ufm.anvil.app/


THANKS!