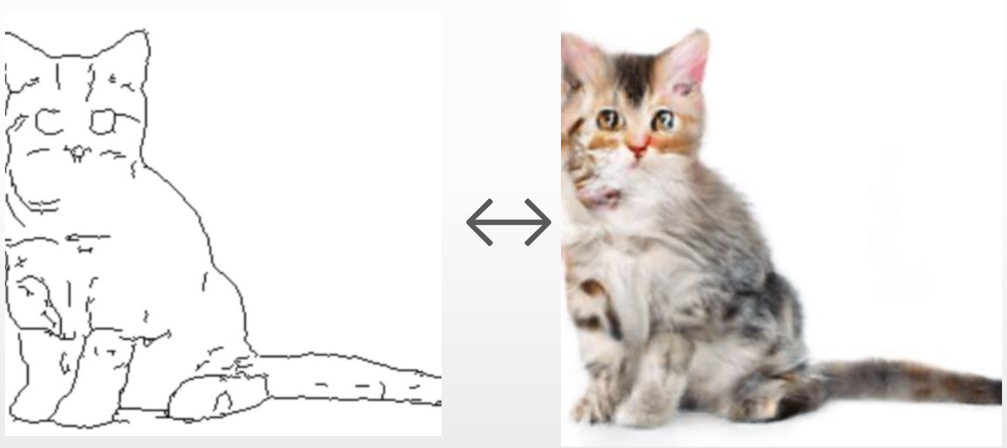
Unpaired Image-to-Image Translation Using CycleGAN

HINAL SHAH

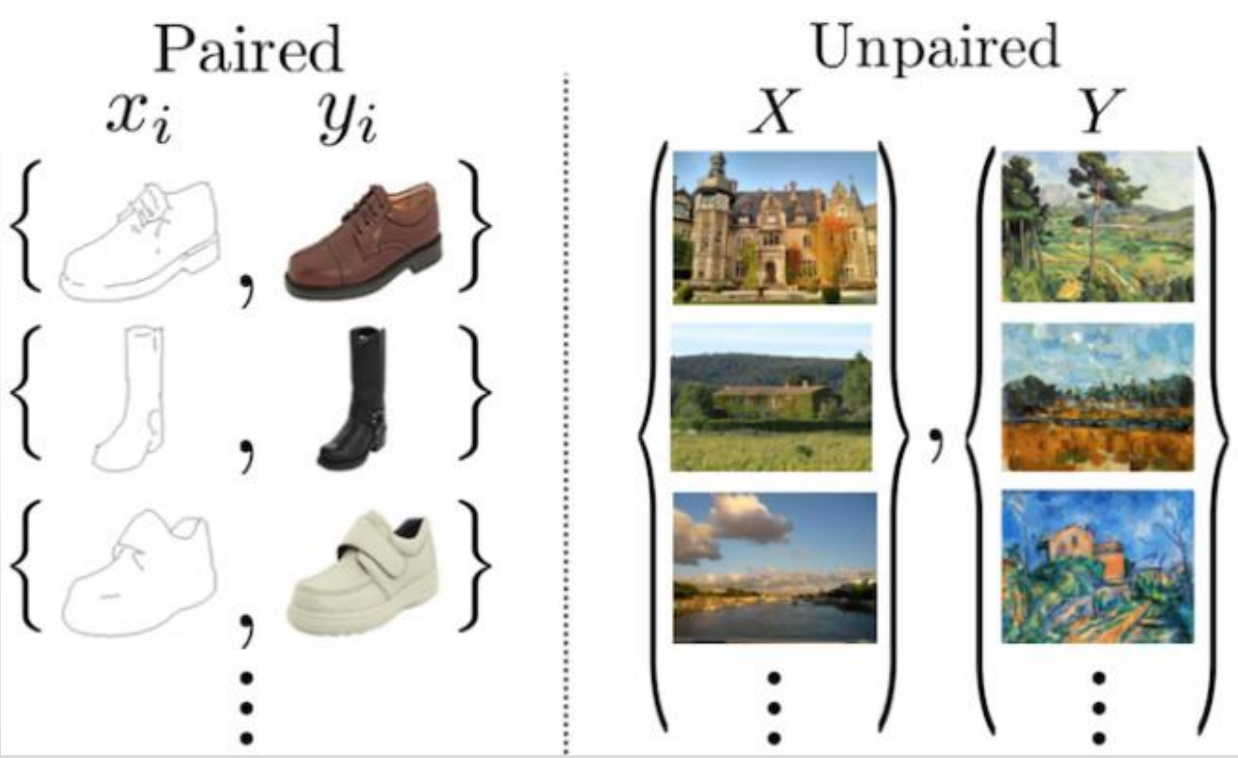
Introduction

- Image-to-image translation is the task of transforming an image from one domain to another by learning mapping between an input image and an output image



Introduction

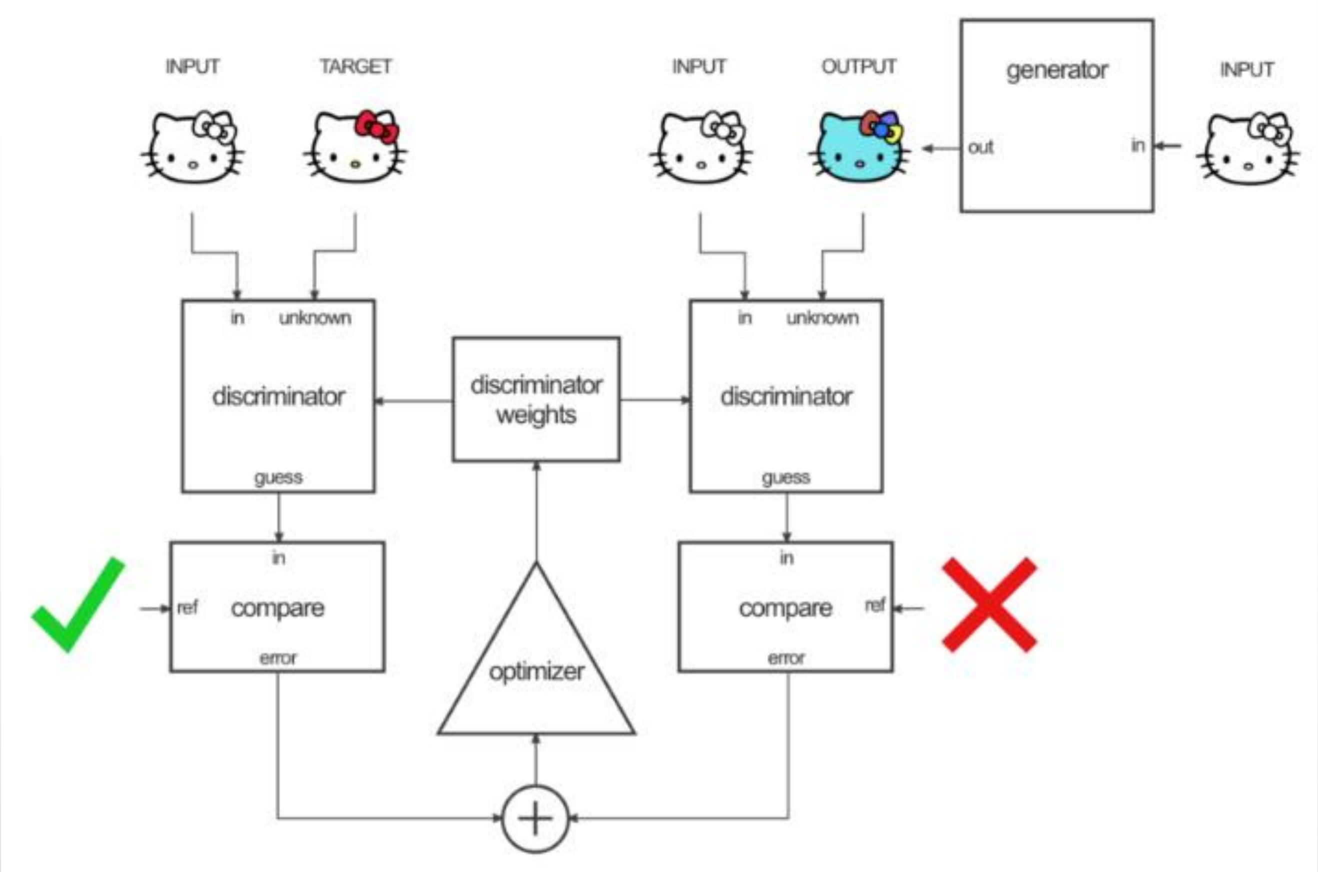
- Paired data is harder to find in most domains, and not even possible in some, so the unsupervised training capabilities of CycleGAN is useful.



Pix2Pix Overview

In paired dataset, input and output images share some common features. So there is a meaningful transformation

(Eg: Pix2Pix model)

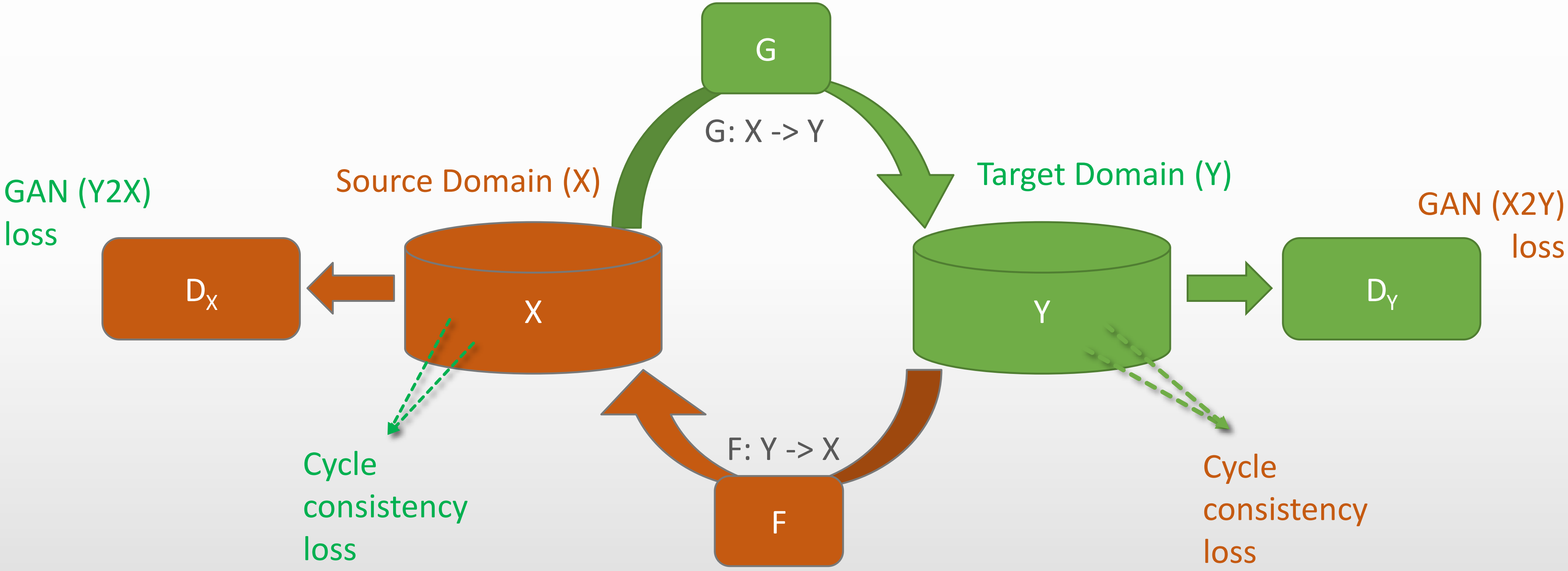


CycleGAN

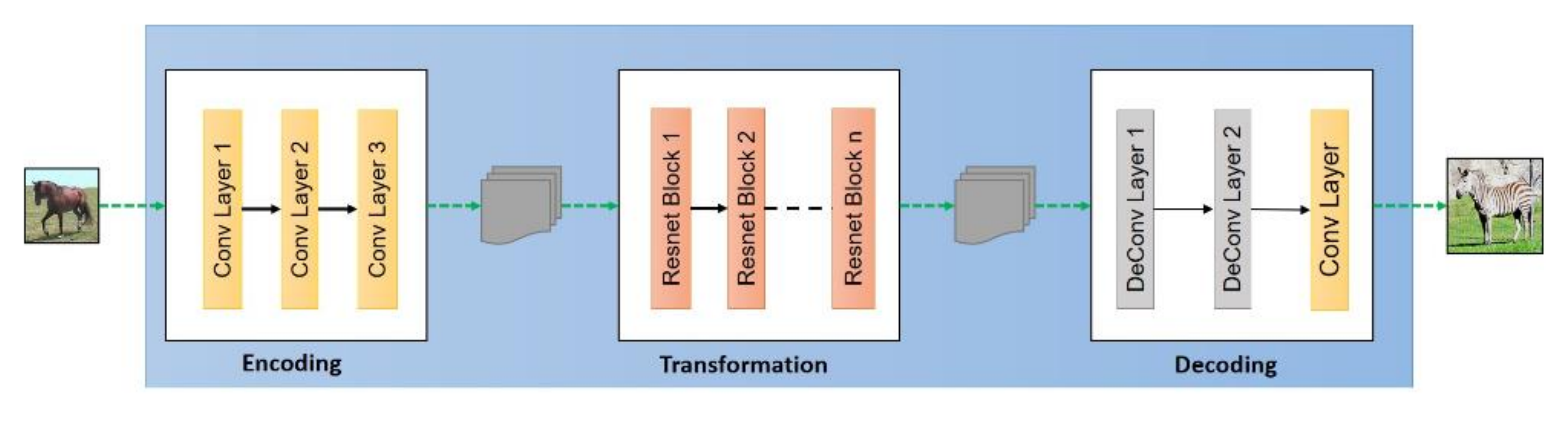
- In cycleGAN, datasets are unpaired. So, to have meaningful mapping between source and domain images, they must share some feature that can be used to map this output image back to input image
- And this can be done by using another generator that must be able to map back this output image back to original input image.
- Since mapping is done in both direction Source → Target Domain and Target → Source Domain, we have two Discriminators along with two Generators.

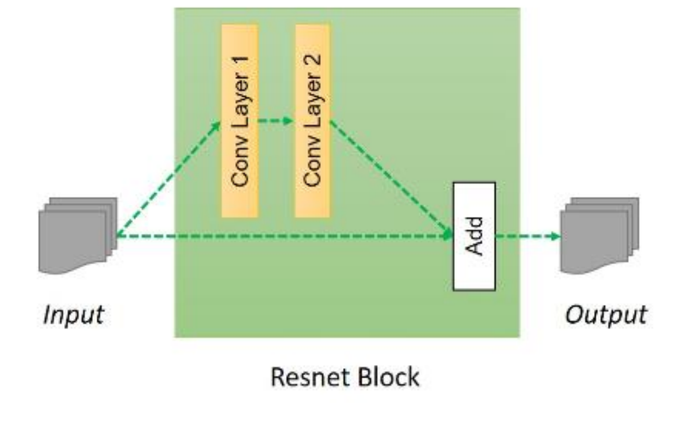
Research Paper: **Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks**
(Jun-Yan Zhu,Taesung Park, Phillip Isola, Alexei A. Efros, *last revised 15 Nov 2018)*

CycleGAN Architecture



Generator Architecture

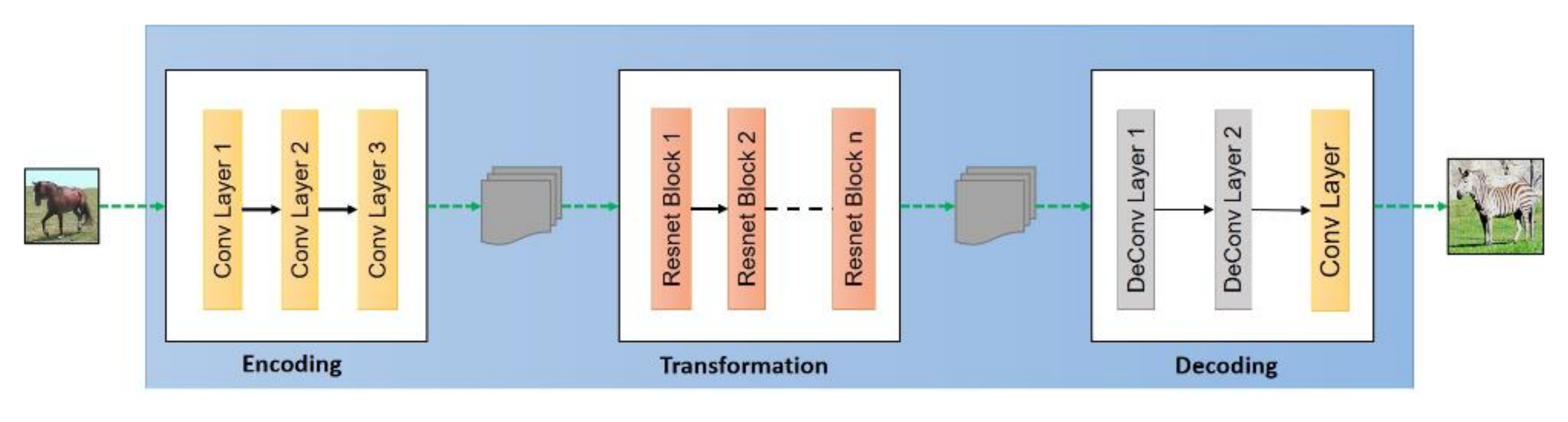


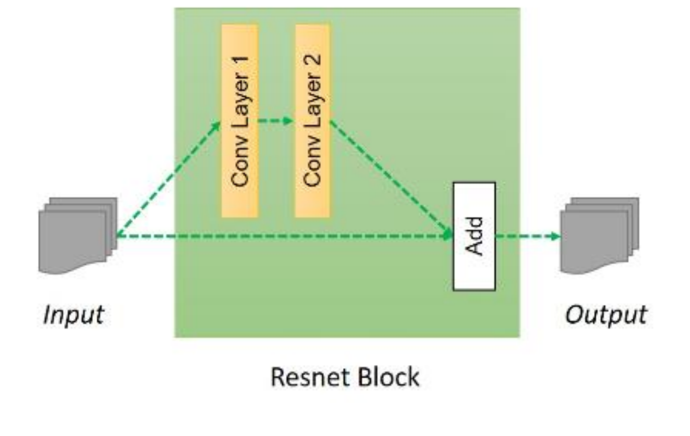


6 Resnet Blocks for **128 x 128** images

- **Conv Layers:** 32, 64, 128,
- **Resnet Layers:** R128, R128, R128, R128, R128, R128,
- **DeConv Layers:** 64, 32, 3

Generator Architecture

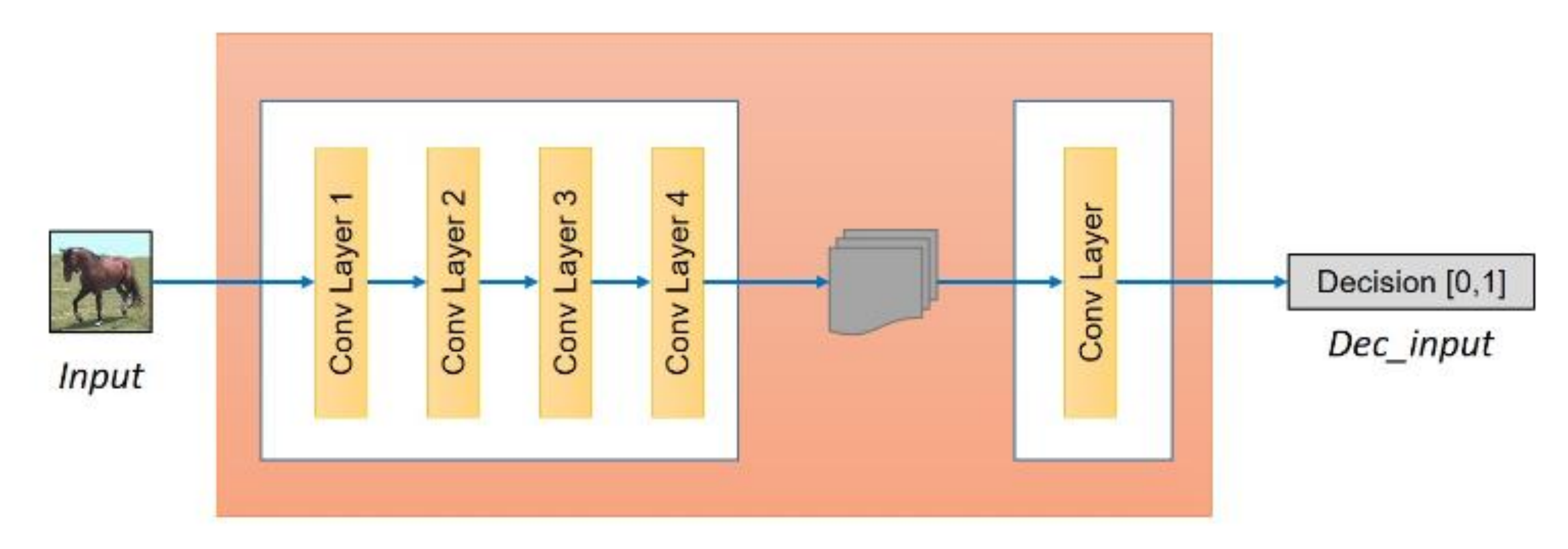




9 Resnet Blocks for **256 x 256** and higher resolution images

Conv Layers: 64, 128, 256
Resnet Layers: R256, R256, R256, R256, R256, R256, R256, R256, R256,
DeConv Layers: 128, 64, 3

Discriminator Architecture



For **128 x 128** and higher resolution images

Conv Layers: 32, 64, 128, 256

Last Conv Layer: 1 (sigmoid layer)

For **256 x 256** and higher resolution images

Conv Layers: 64, 128, 256, 512

Last Conv Layer: 1 (sigmoid layer)

Objective Function

Adversarial Loss (G: X→ Y):

$$\mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) = \mathbb{E}_{y \sim p_{\text{data}}(y)}[\log D_Y(y)] + \mathbb{E}_{x \sim p_{\text{data}}(x)}[\log(1 - D_Y(G(x))]$$

Cycle Consistency Loss:

$$\mathcal{L}_{\text{cyc}}(G, F) = \mathbb{E}_{x \sim p_{\text{data}}(x)}[\|F(G(x)) - x\|_1] + \mathbb{E}_{y \sim p_{\text{data}}(y)}[\|G(F(y)) - y\|_1].$$

Full Objective Function:

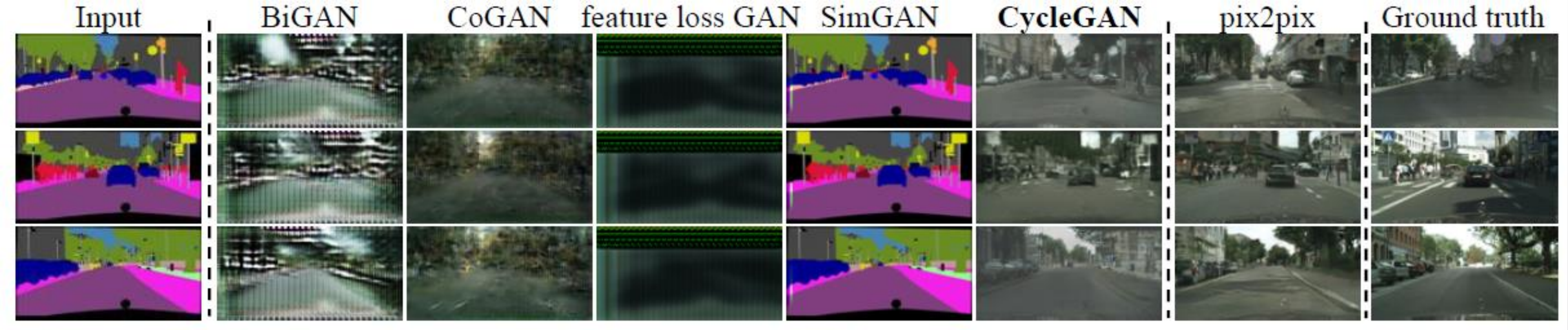
$$\begin{aligned}\mathcal{L}(G, F, D_X, D_Y) =& \mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) \\ &+ \mathcal{L}_{\text{GAN}}(F, D_X, Y, X) \\ &+ \lambda \mathcal{L}_{\text{cyc}}(G, F),\end{aligned}$$

So we are trying to solve:

$$G^*, F^* = \arg \min_{G,F} \max_{D_x, D_Y} \mathcal{L}(G, F, D_X, D_Y)$$

Evaluation

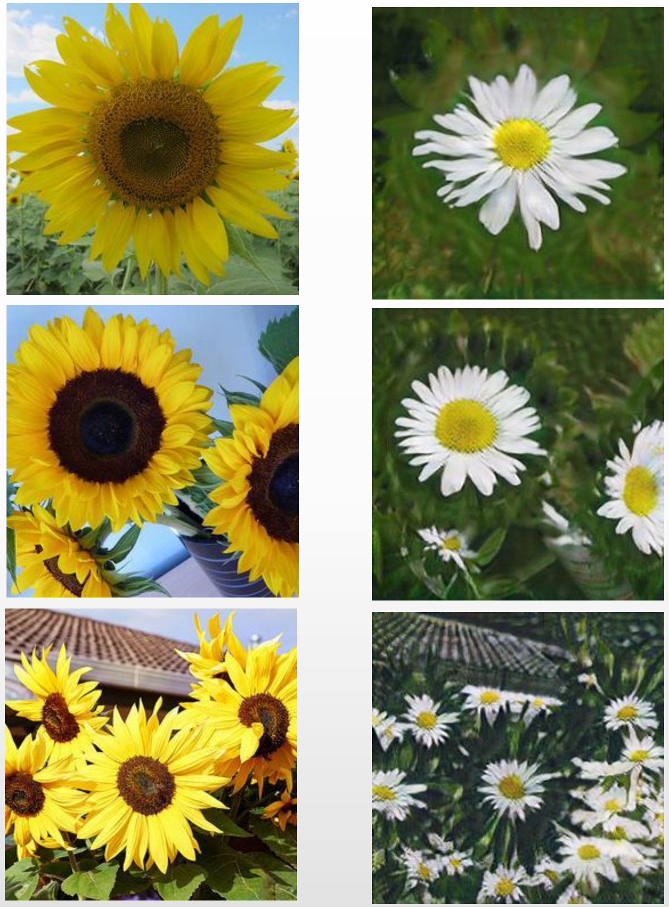
- No ground Truth values available as datasets are unpaired



Evaluation on Cityscapes Dataset

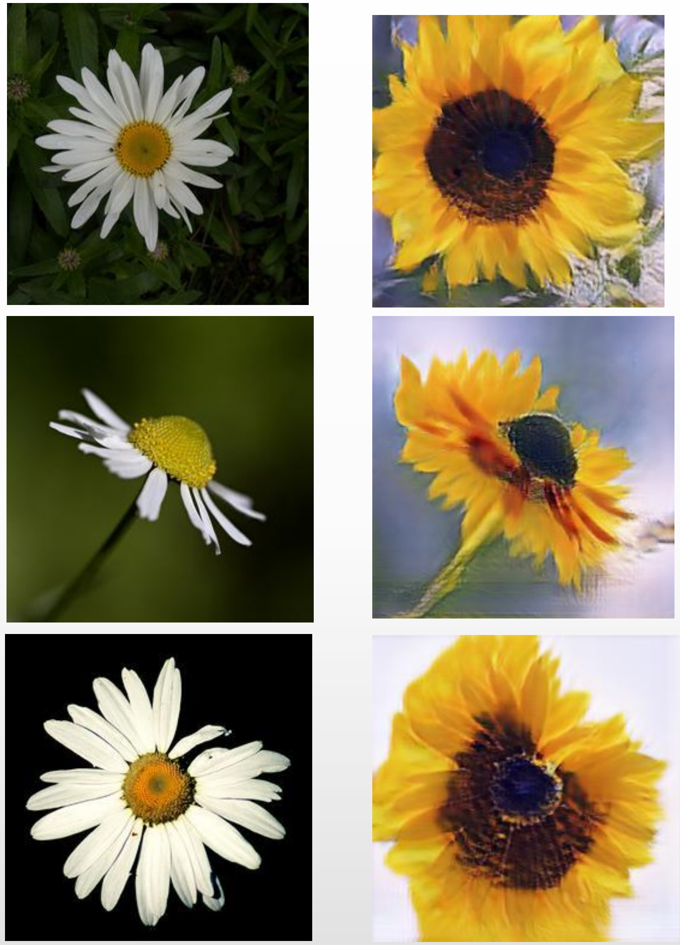
Results

Input Output



Sunflower → Daisy

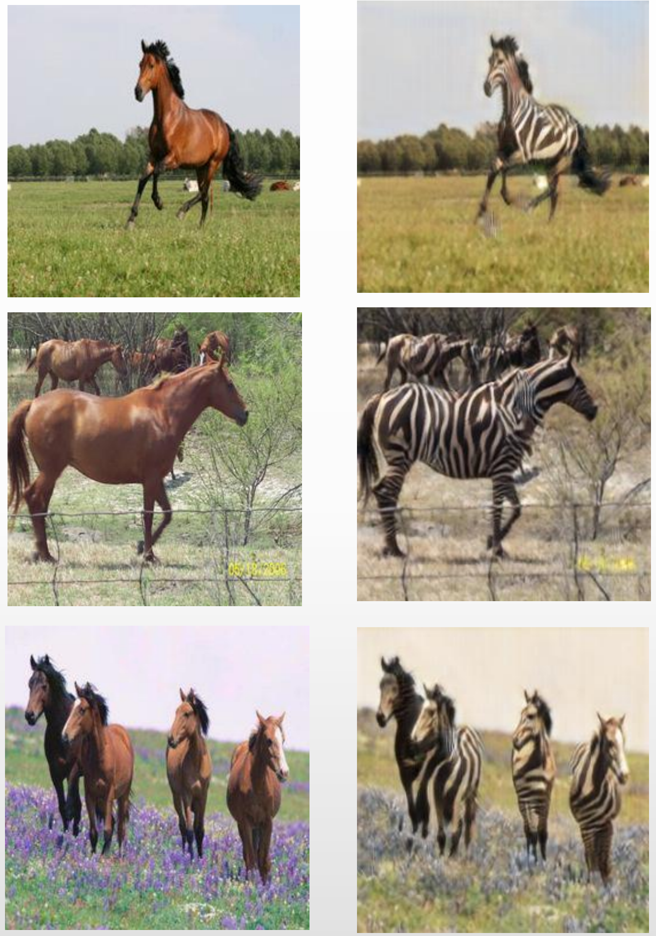
Input Output



Daisy → Sunflower

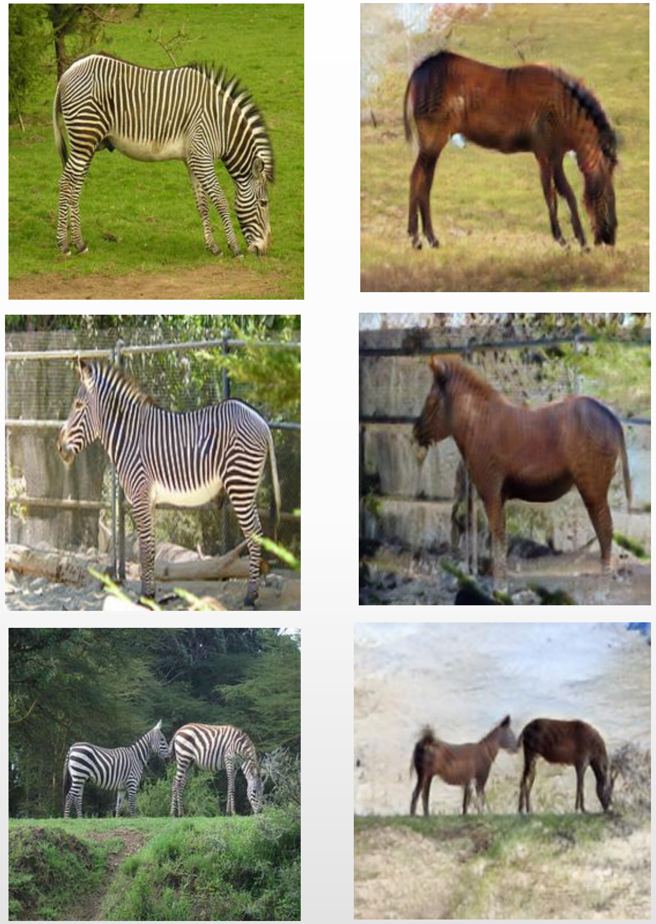
Results

Input Output



Horse → Zebra

Input Output



Zebra → Horse

Results

Input Output

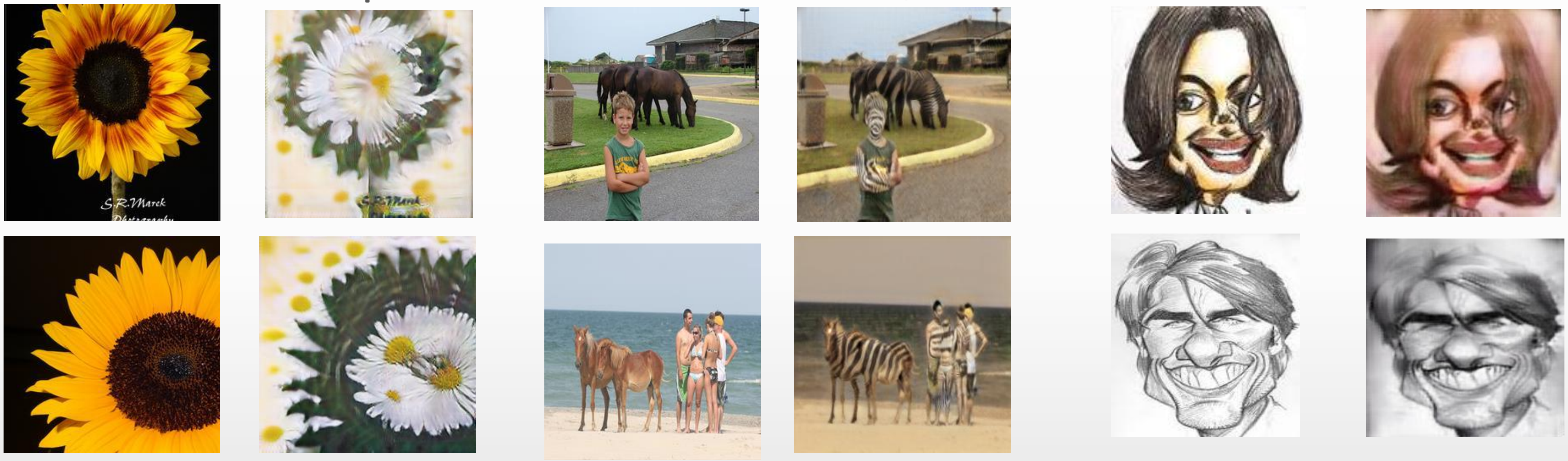
Face → Cartoon

Input Output

Cartoon → Face

Results

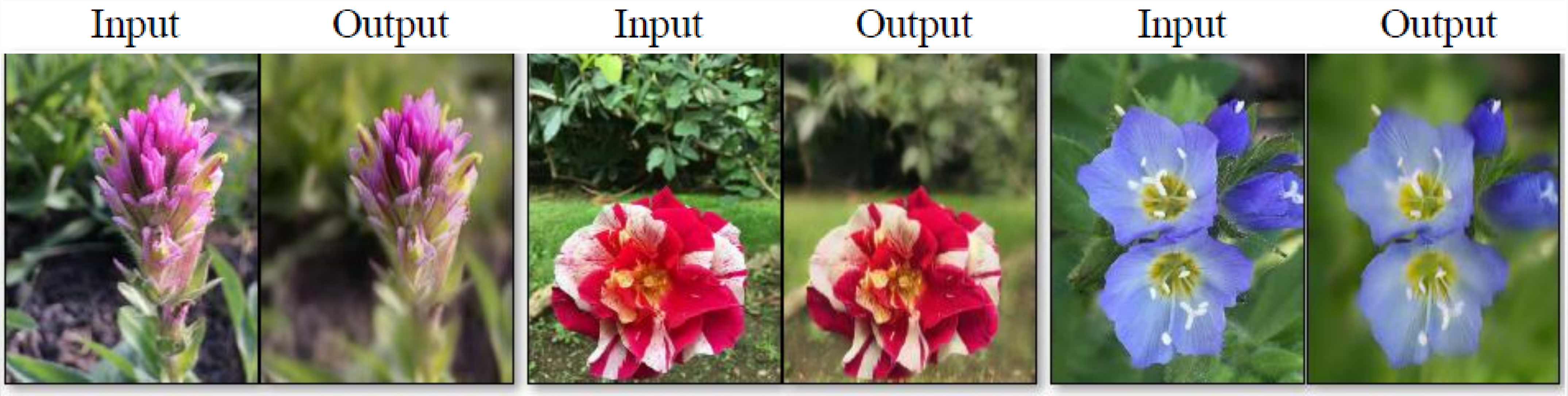
Input Output Input Output Input Output



Some Failed Cases

Conclusion

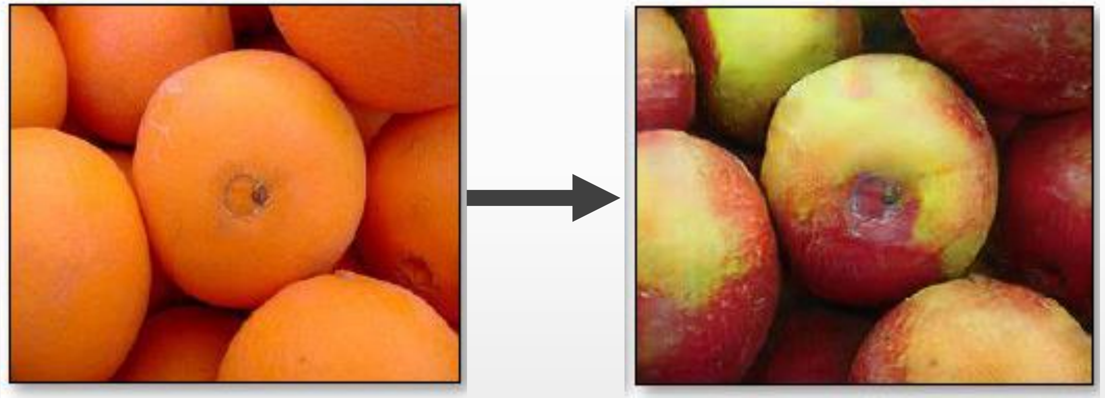
This method works well in performing tasks that involve color and texture changes like photo enhancement



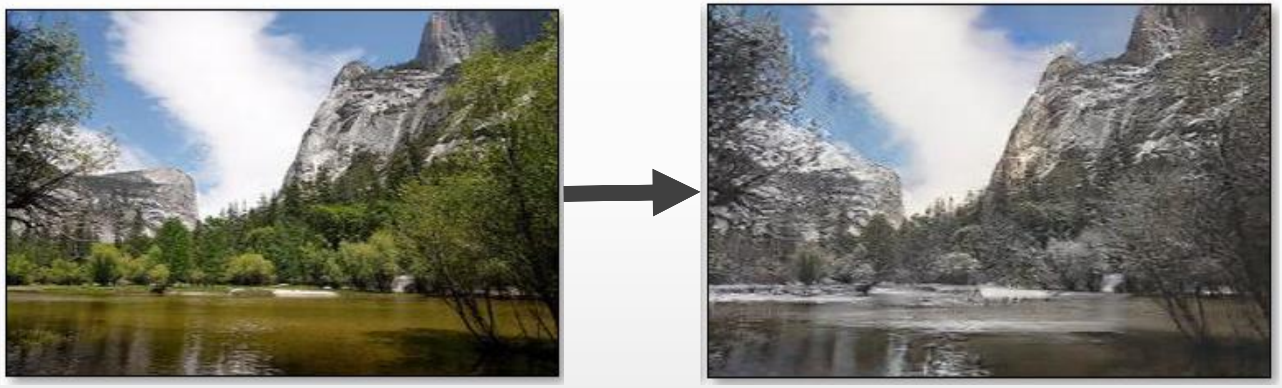
SmartPhone Snaps → DSLR Photographs

Conclusion

This method works well in performing tasks that involve color and texture changes like photo enhancement



Orange → Apple

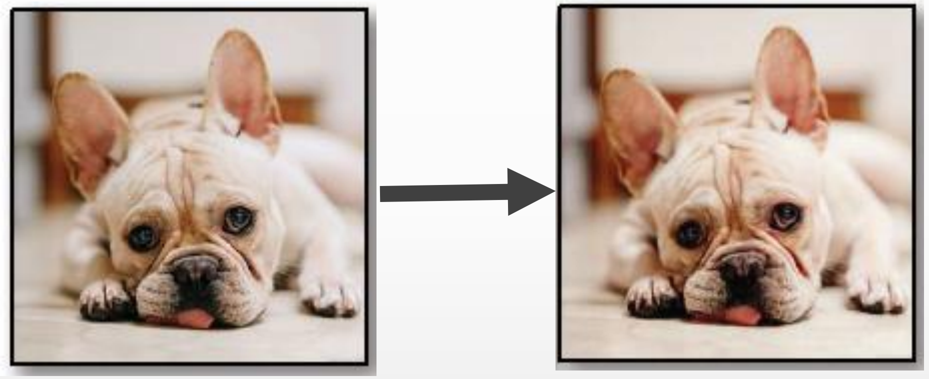


Summer Yosemite → Winter Yosemite

Conclusion

It does not work well in tasks that require geometric changes

Cat → Dog



Dog → Cat

Conclusion

- Does not perform well in translating images that has complicated backgrounds
- It does only one-to-one mapping

Horse → Zebra

Questions?