Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks

Taesung Park, UC Berkeley

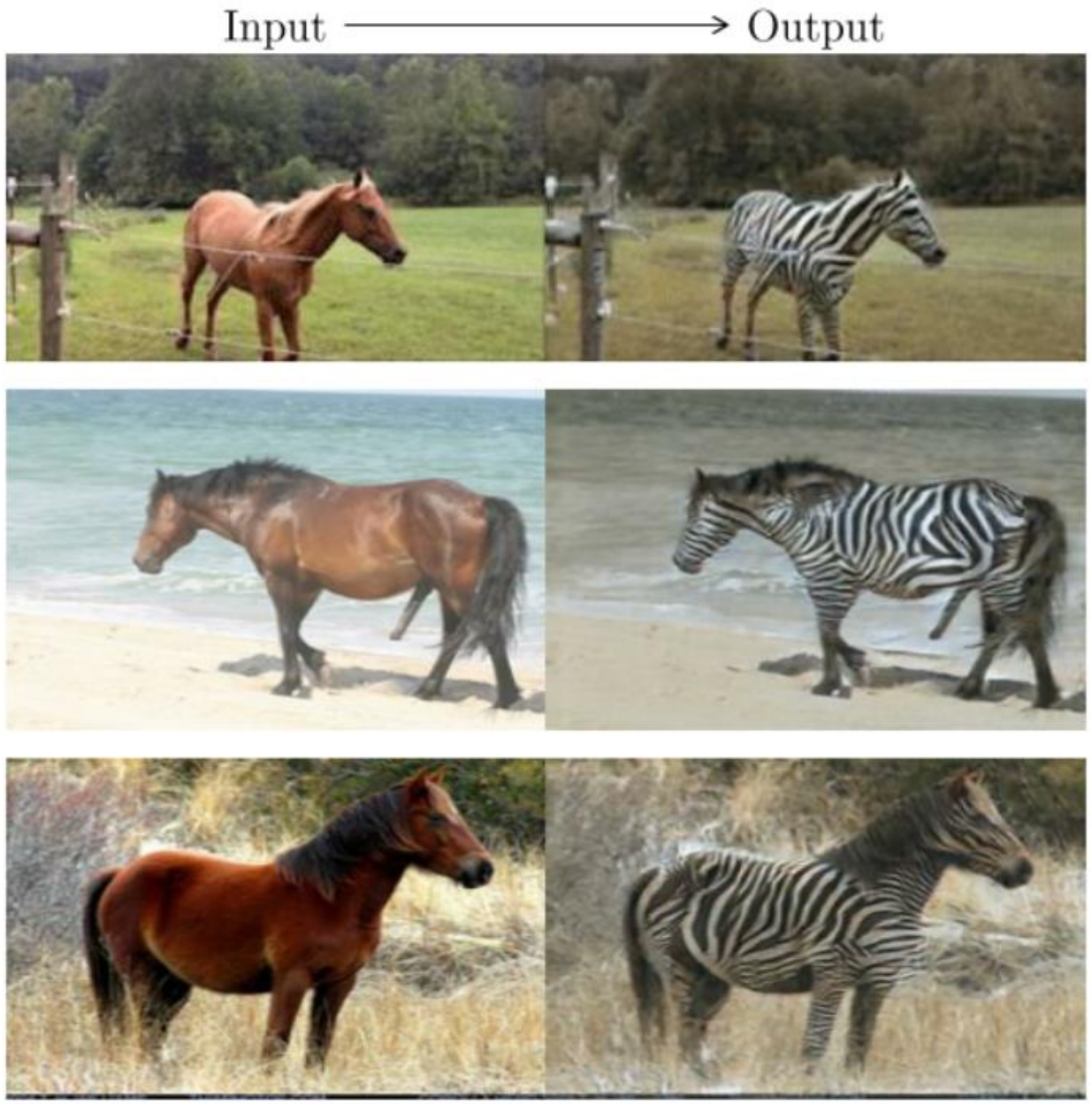
2019. 05 30

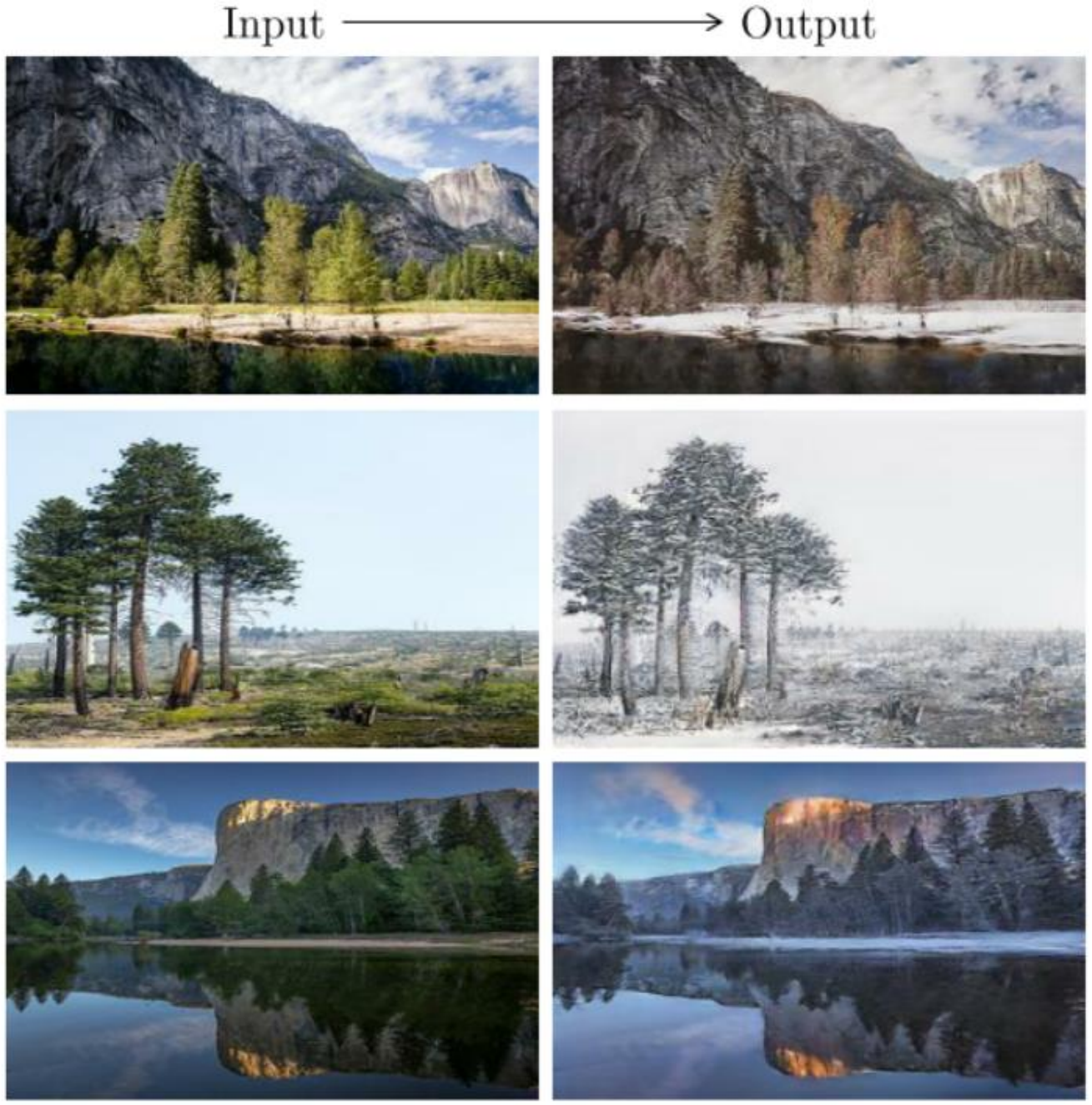
Boaz 분석 D조

문현주 신광욱 고동희 박서호

https://arxiv.org/pdf/1703.10593.pdf

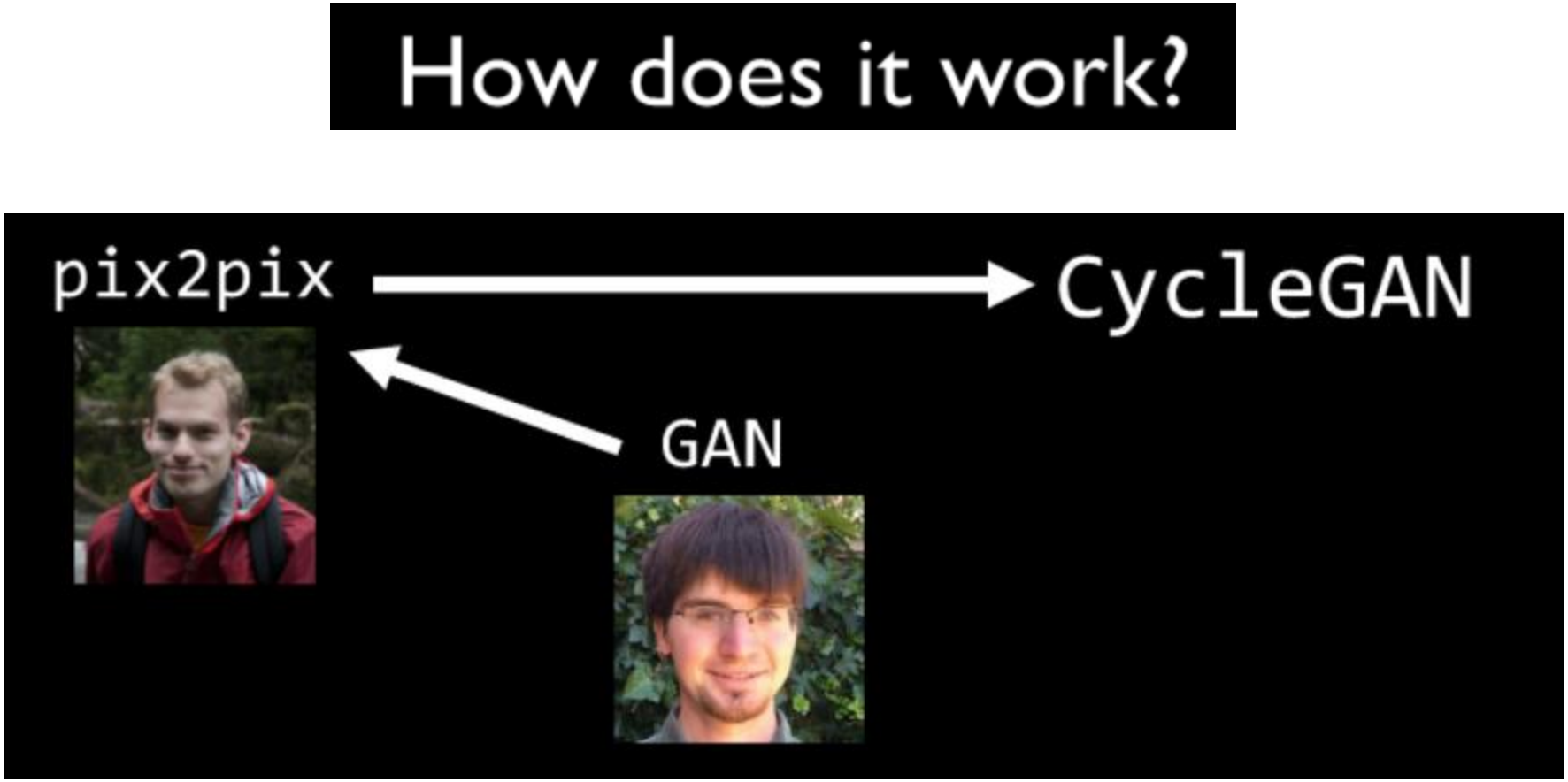
Index


Input
Output

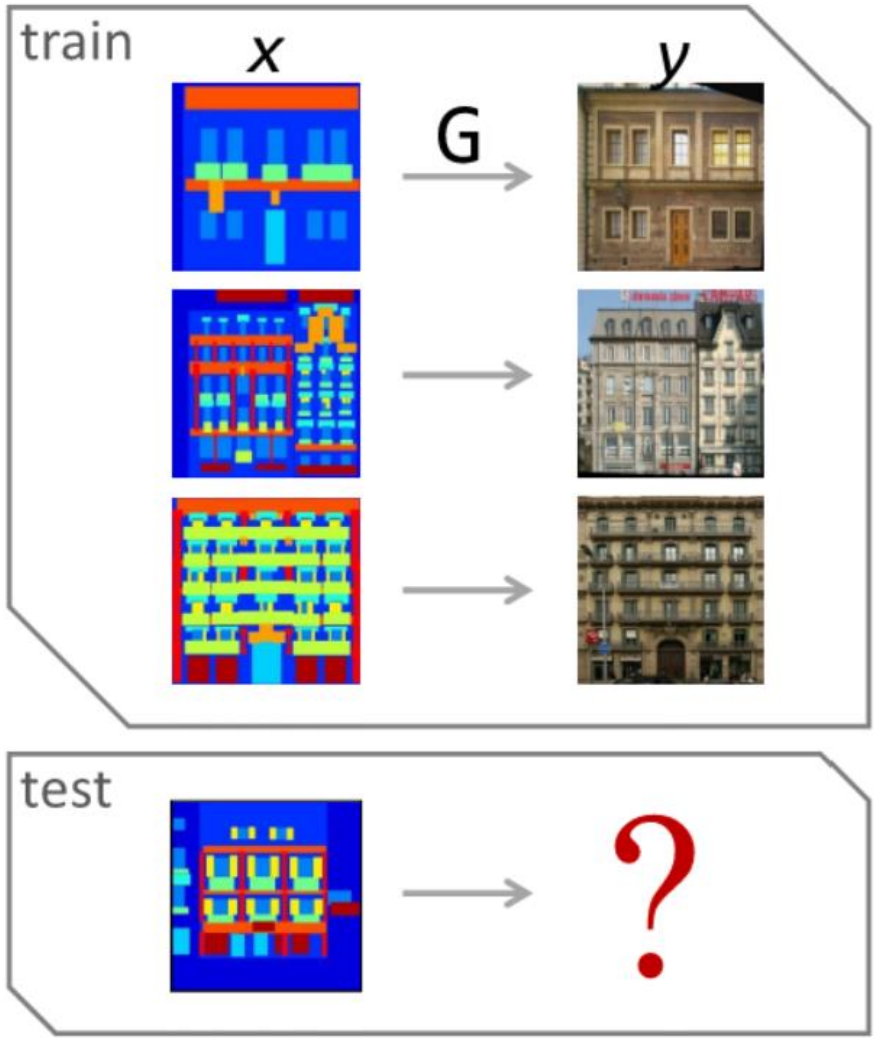

Input
Output

즉, cyclegan 은 한 이미지를 다른 dataset의 이미지로 바꿔줄 수 있는 기능을 하게 된다.

그와 동시에 반대쪽 수정 작업도 동시에 훈련시킬 수 있다.


How does it work?
pix2pix
CycleGAN
GAN

pix2pix (Isola et al., 2017) - Input 과 Output이 모두 사진이어야 한다.



- Supervised
- loss: Minimize the difference between output *G(x)* and ground truth *y*

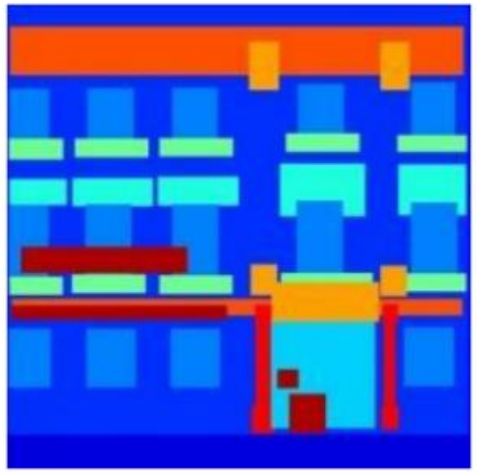
Data from [Tylecek, 2013]

X= 각 픽셀 별로 라벨을 표시한 dataset Y= 건물 앞 부분 찍은 dataset.

pix2pix (Isola et al., 2017)

Loss: Minimize the difference between output *G(x) and the ground truth y*

$$\sum_{(x,y)} \|y - G(x)\|_1$$



Input

Output

Ground Truth

pix2pix (Isola et al., 2017)

Loss: Minimize the difference between output *G(x) and the ground truth y*

$$\sum_{(x,y)} \|y - G(x)\|_1$$

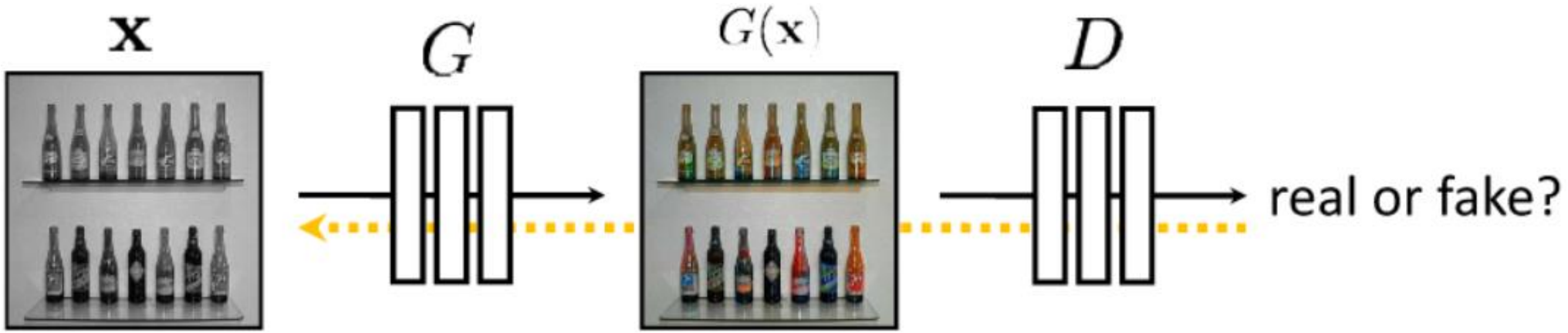
Input

Output

Ground Truth

Let another deep network point out the difference

Generative Adversarial Network (GAN) (Goodfellow et al., 2014)



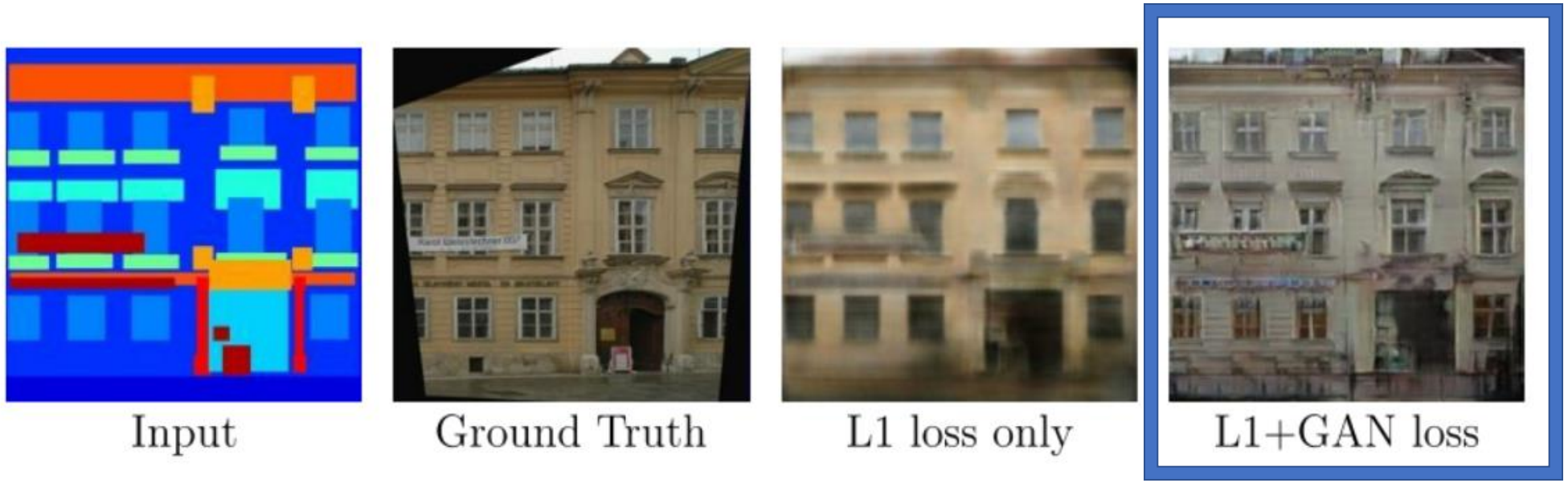
D : 뭐가 진짜고 뭐가 가짜인지 판별해내는 역할

G : D가 진짜/가짜 구별을 하기 어려울 정도로 Realistic한 이미지 생성하는 역할

Pix2pix (P. Isola et al., 2017)

Loss: Minimize the difference between and output *G(x)* and ground truth *y*

$$\sum_{(x,y)} \|y - G(x)\|_1 + L_{GAN}\,(G(x), y)$$



Input | Ground Truth | L1 loss only | L1+GAN loss

Related Work [Pix2Pix] 요약:

1. Image to Image Mapping Network에서 Photo-realistic을 추구하고 싶음

-> 그래서 GAN의 Adversarial Training을 도입

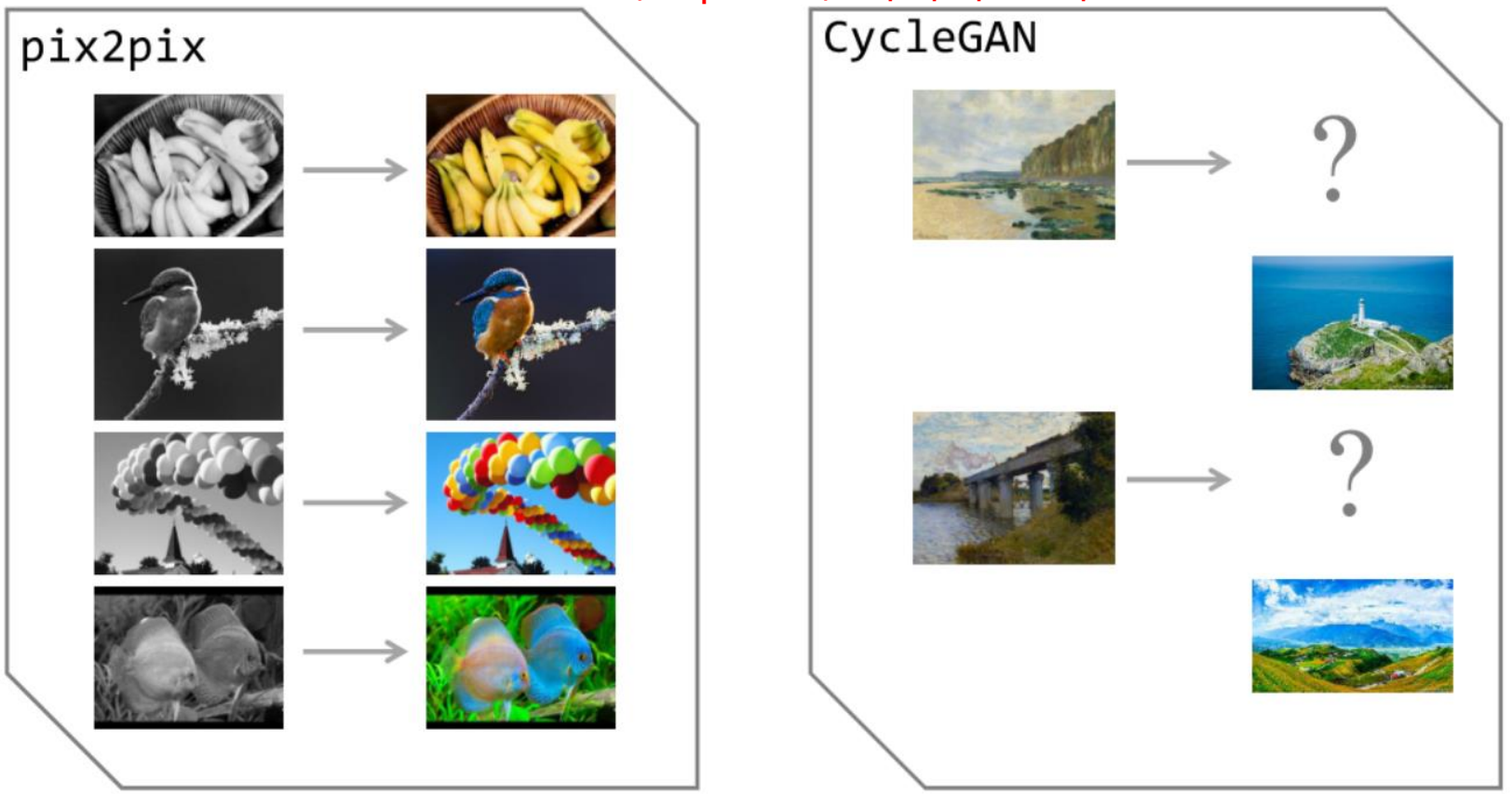
3. U-Net과 PatchGAN등을 통해서 성능 최적화

4. Training Data가 Pair로 존재해야 함 (그래서 CycleGAN, DiscoGAN이 나

실습코드 :
https://github.com/tensorflow/tensorflow/blob/r1.13/tensorflow/contrib/eager/python/examples/pix2pix/pix2pix_eager.ipynb

CycleGAN (Zhu et al. 2017)

→ '쌍을 이루는'(Paired) 데이터가 아닌, '쌍을 이루지 않는'(Unpaired) 데이터로 학습.



Pix2Pix에서나 CycleGAN의 공통된 문제 정의: **'한 도메인에 있는 이미지를 다른 도메인으로 해석한다.'**

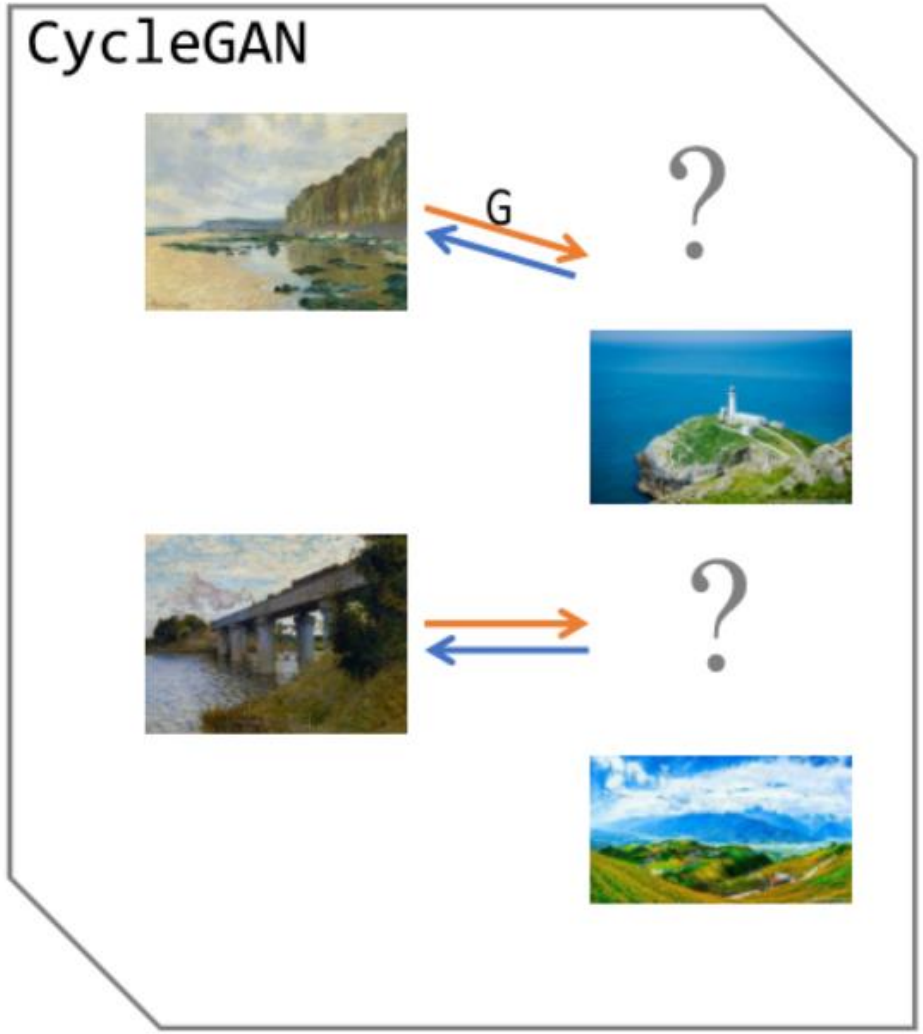
그러나 Pix2Pix 모델에서는 이 문제를 풀기 위해서는 반드시 두 도메인 양쪽에 대응되는 데이터 쌍이 존재해야 한다. CycleGAN 모델에서 제시하는 것은, *그러한 데이터 쌍이 없이도* 문제를 풀어보자는 것

CycleGAN (Zhu et al. 2017)

Loss: $L_{GAN}(G(x), y)$

G(x) should just look photorealistic

and be able to reconstruct *x*



CycleGAN (Zhu et al. 2017)

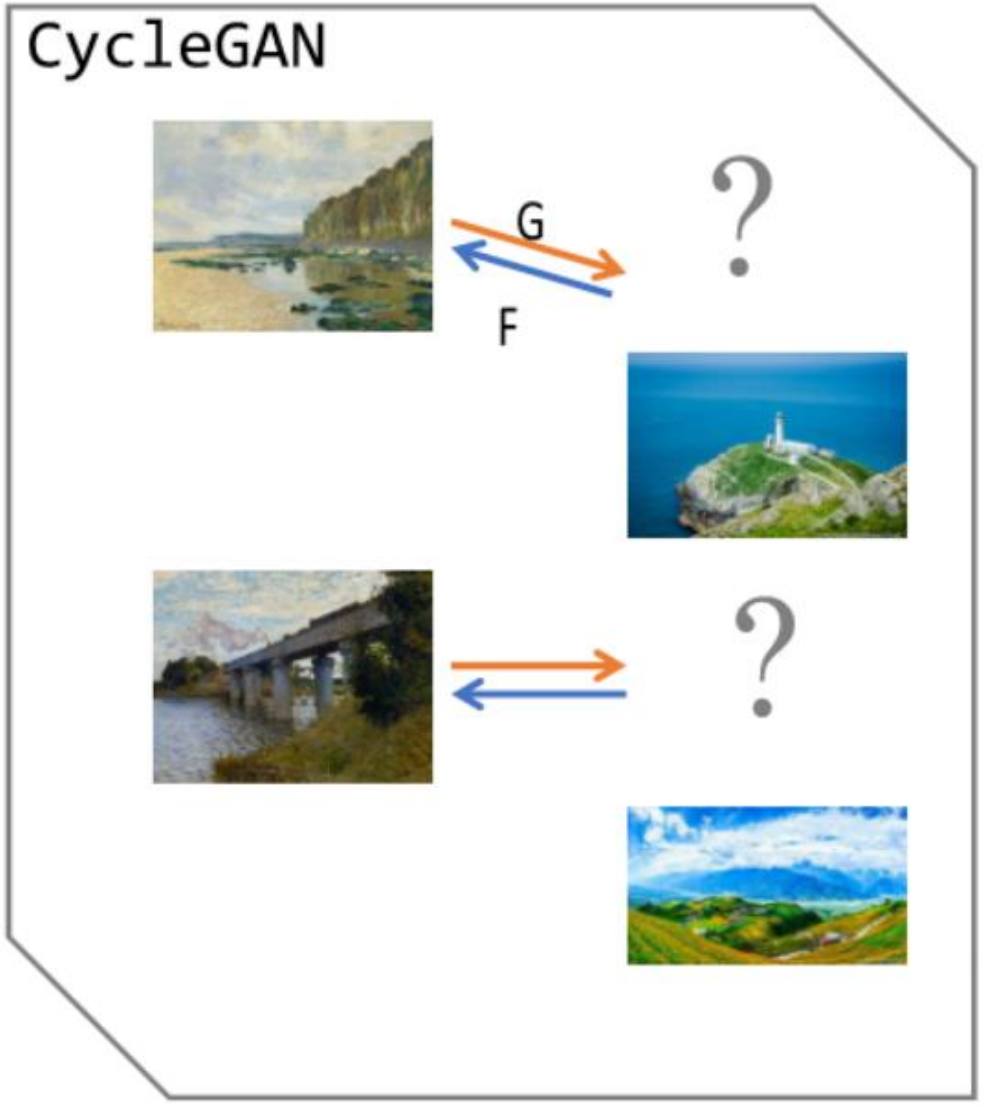
Loss

$$L_{GAN}(G(x), y) + \left\| F\left(G(x)\right) - x \right\|_1$$

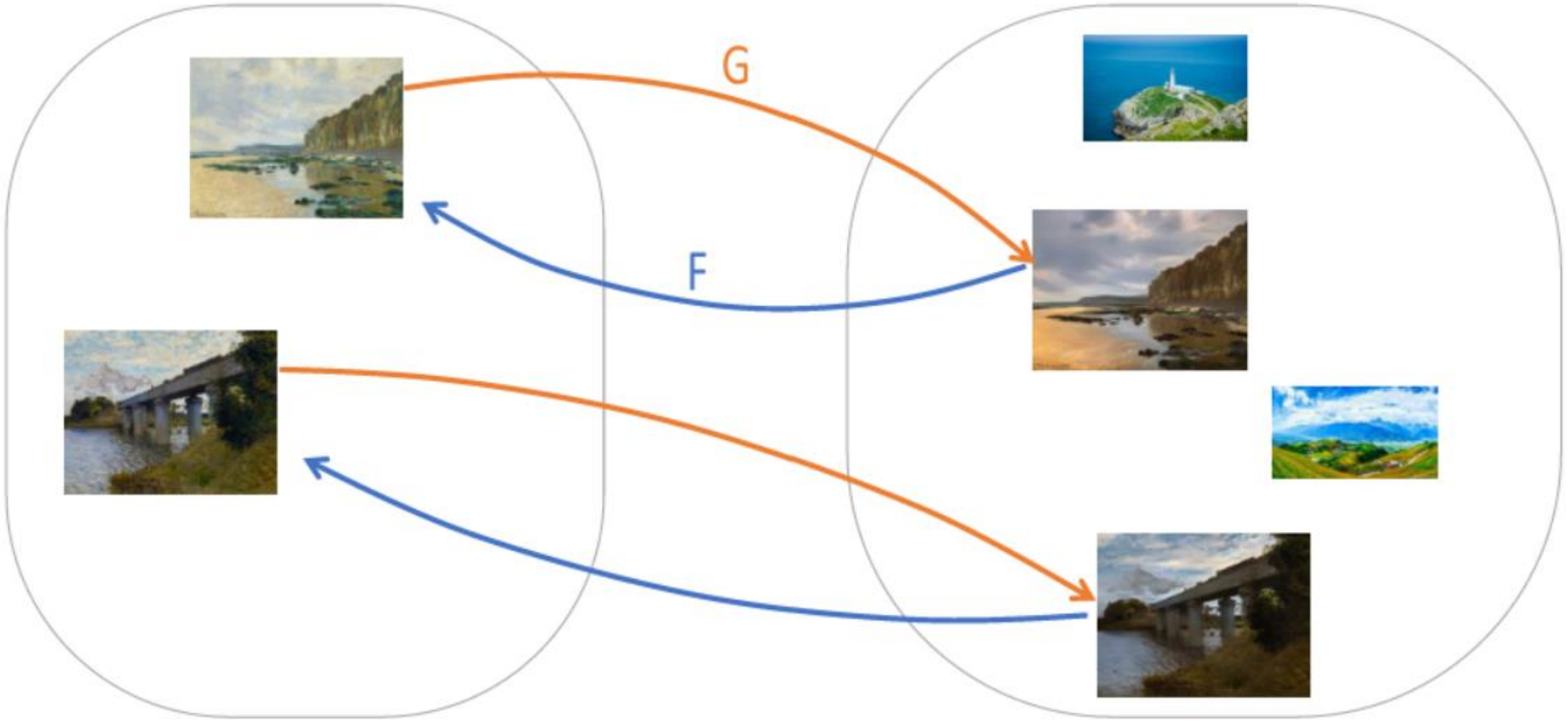
G(x) should just look photorealistic

and *F(G(x))* should be *F(G(x)) = x*,

where *F* is the inverse deep network

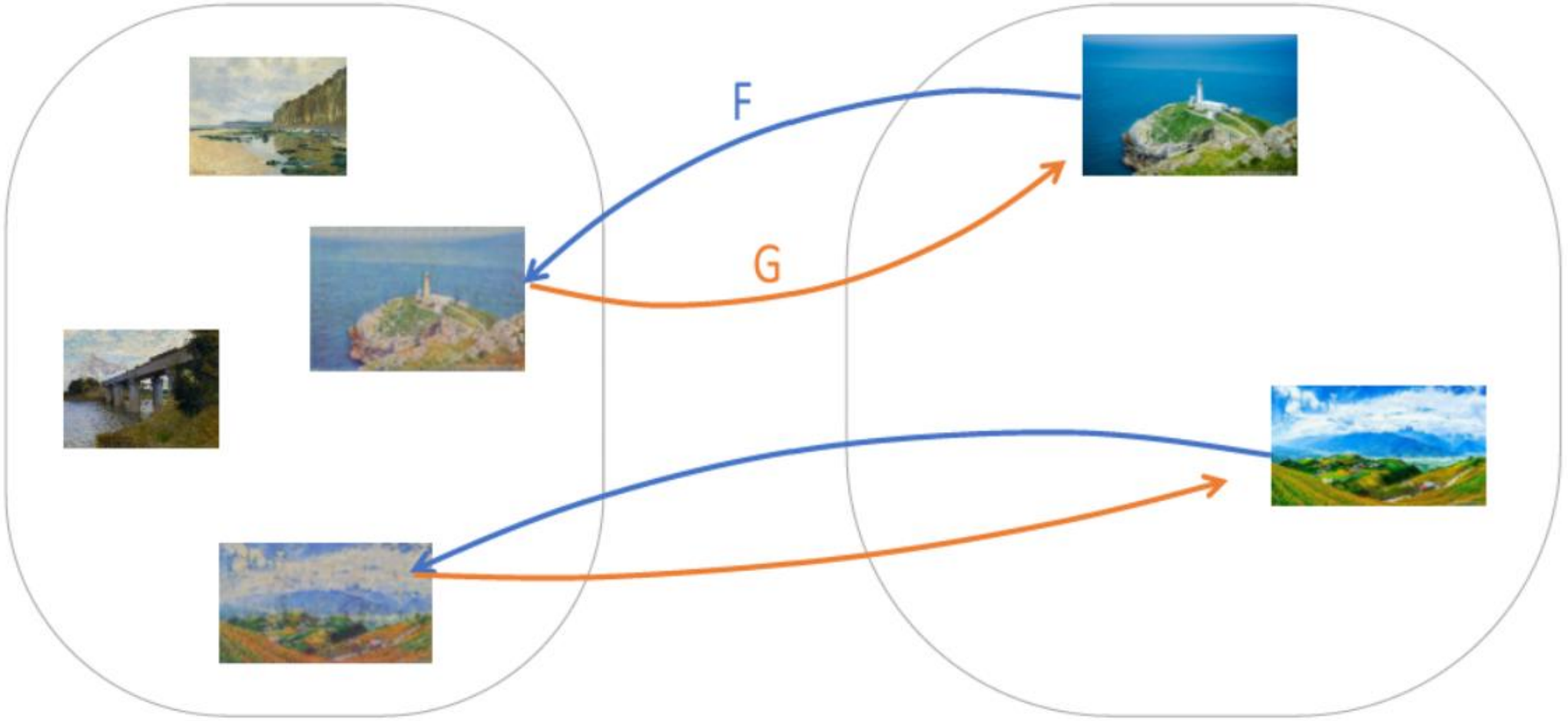


CycleGAN (Zhu et al. 2017)



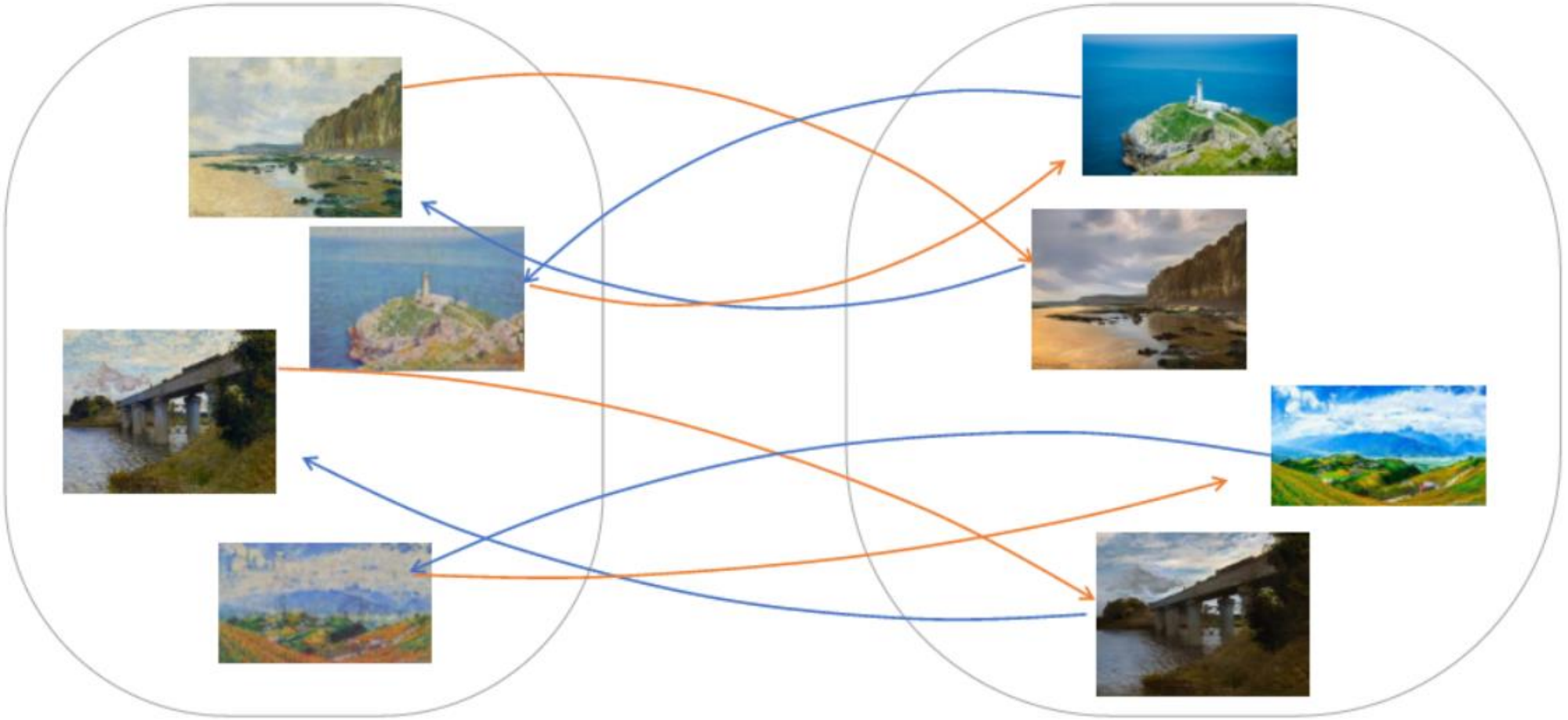
$$L_{GAN}(G(x), y) + \left\| F(G(x)) - x \right\|_1$$

CycleGAN (Zhu et al. 2017)



$$L_{GAN}(F(y), x) + \|G(F(y)) - y\|_1$$

CycleGAN (Zhu et al. 2017)



→ 4가지의 Loss term이 합쳐진것이 Cycle Gan 의 Full Loss Formulation :

$$L_{GAN}(G(x), y) + \left\| F(G(x)) - x \right\|_1 \quad + \quad L_{GAN}(F(y), x) + \left\| G(F(y)) - y \right\|_1$$

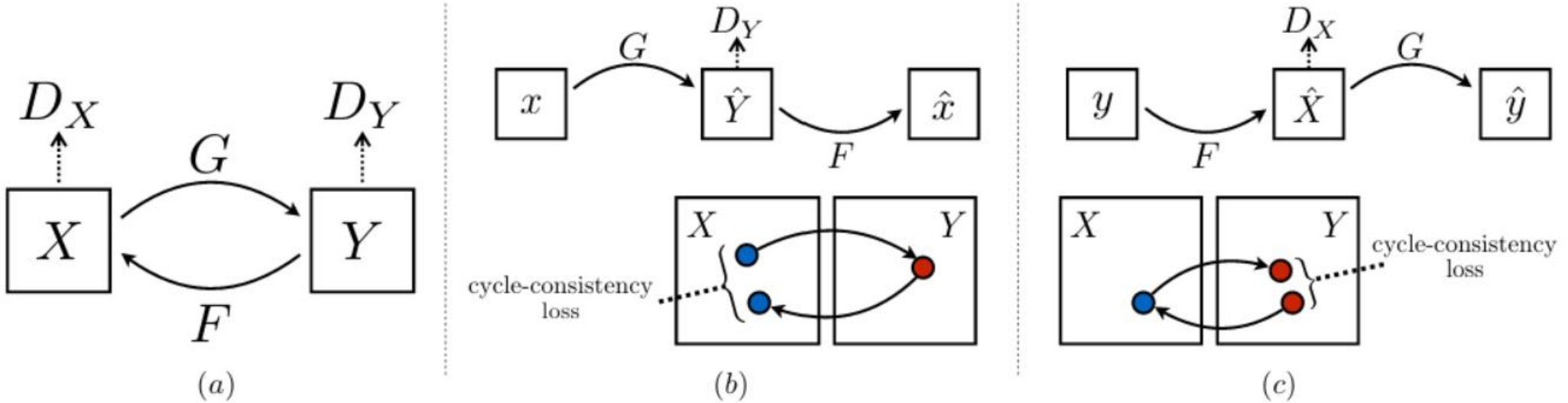
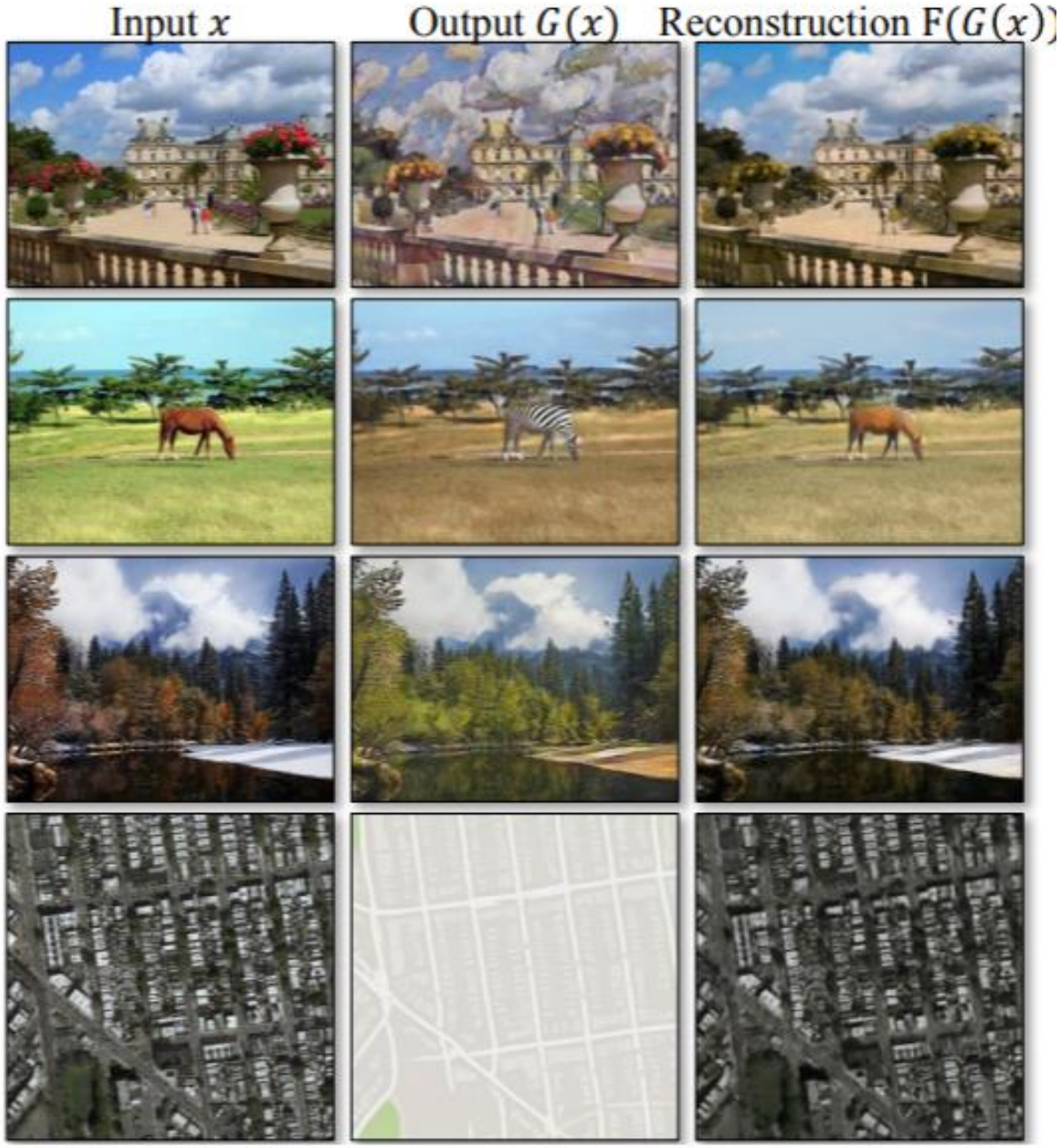
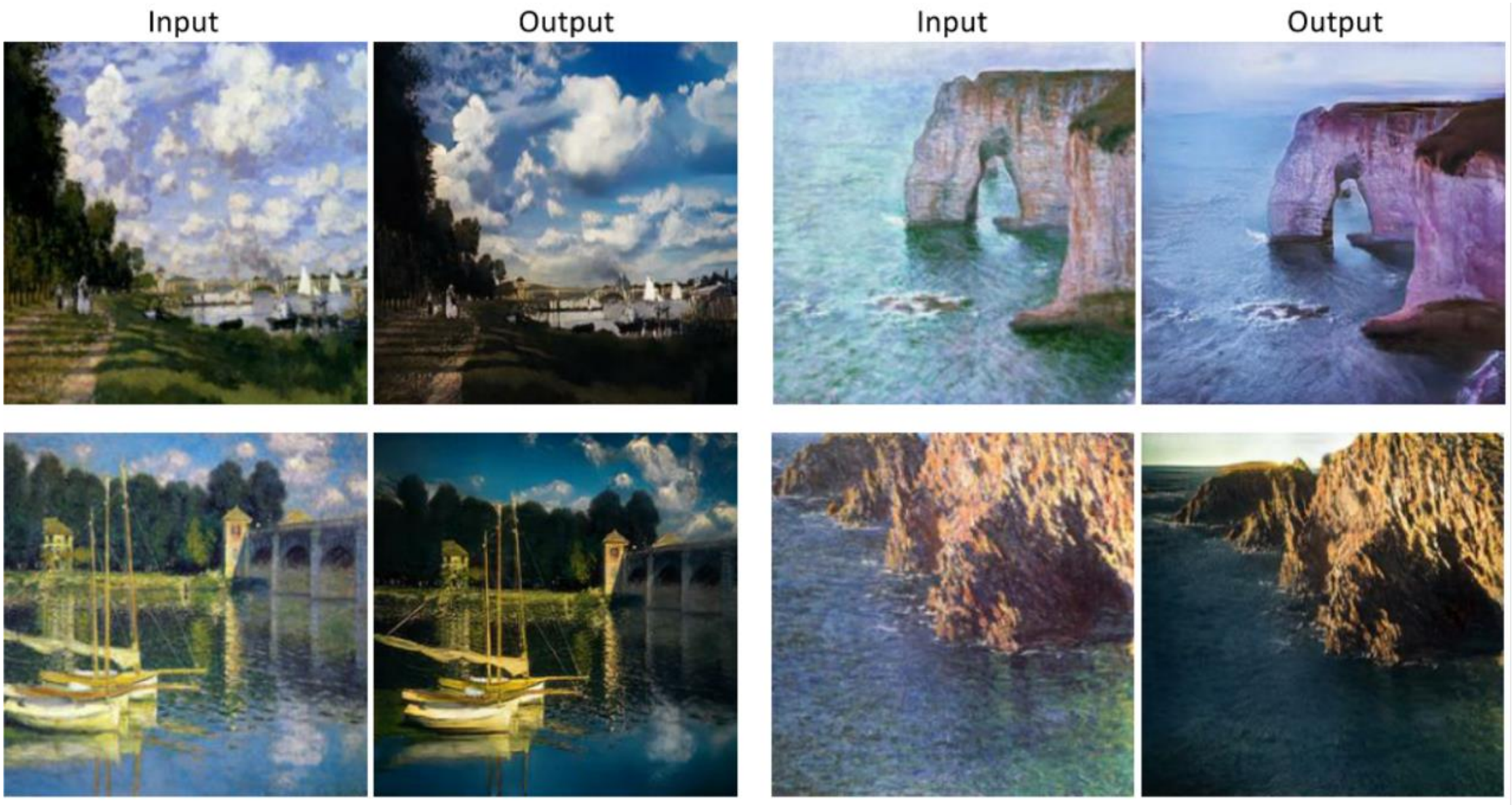


Figure 3: (a) Our model contains two mapping functions $G : X \rightarrow Y$ and $F : Y \rightarrow X$, and associated adversarial discriminators D_Y and D_X. D_Y encourages G to translate X into outputs indistinguishable from domain Y, and vice versa for D_X and F. To further regularize the mappings, we introduce two *cycle consistency losses* that capture the intuition that if we translate from one domain to the other and back again we should arrive at where we started: (b) forward cycle-consistency loss: $x \rightarrow G(x) \rightarrow F(G(x)) \approx x$, and (c) backward cycle-consistency loss: $y \rightarrow F(y) \rightarrow G(F(y)) \approx y$

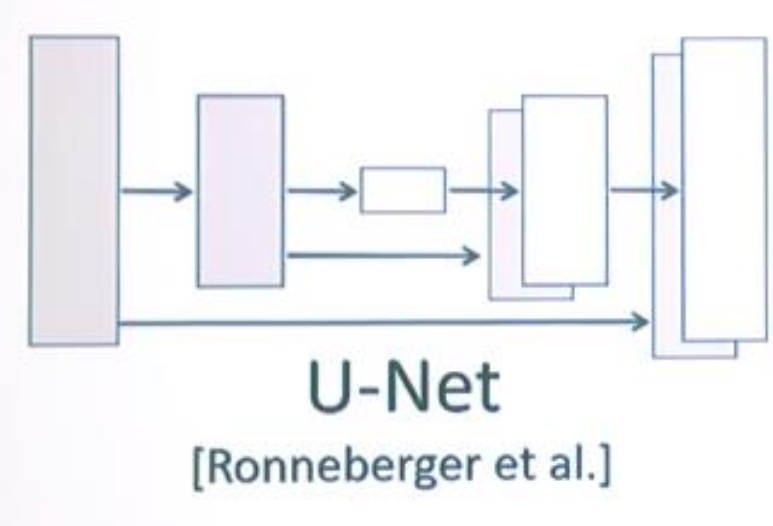
3. Formulation



3. Formulation



Network Architecture – Generator ***G***



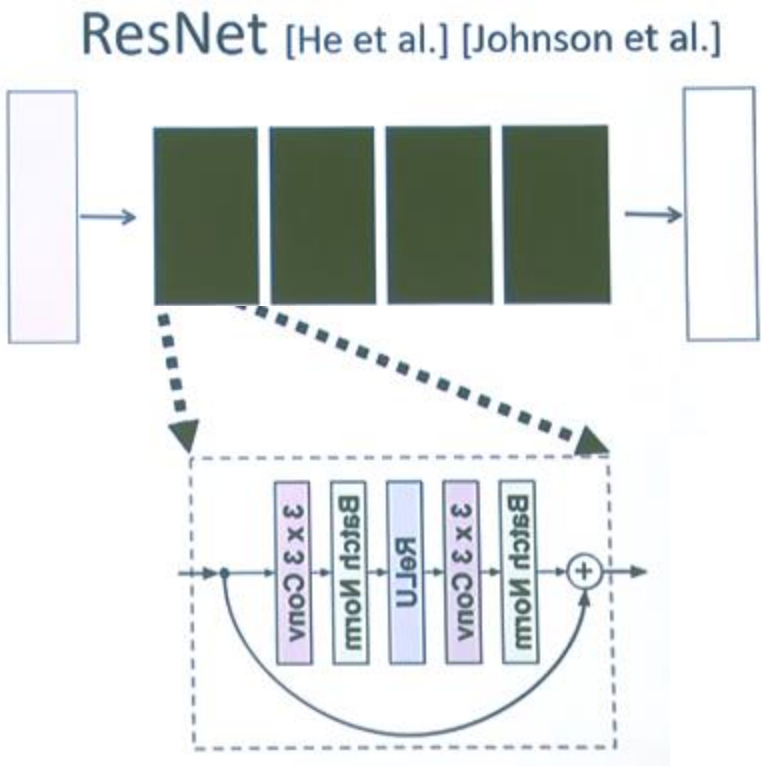
처음 시도했던 Architecture

encoder-decode에
skip-connection을 추가적으로 사용한 U-Net을 이용

skip-connection을 이용함으로써 Bottle-neck을 통과하면서 생기는 컨텐츠의 손실을 방지

하지만 depth가 거의 없어서 결과가 만족스럽지 않음

Network Architecture – Generator ***G***



최종적으로 사용한 Architecture

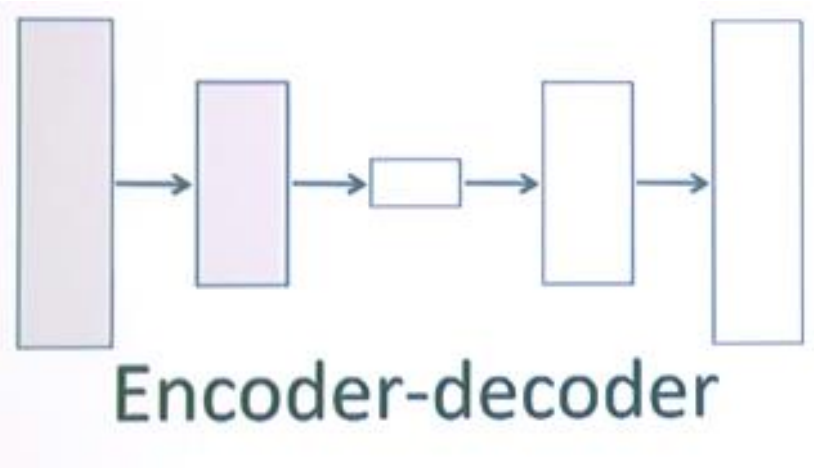
Bottle-neck이 없으므로 컨텐츠 디테일에 손실이 적음

ResNet의 특징인 residual block이 skip-connection

이미지 퀄리티 측면에서는 만족할 만한 결과

하지만 bottle-neck이 없어서 메모리를 많이 차지
그 결과, 학습할 수 있는 parameter의 수가 적어져 형태변화에 제한이 생기는 것 같음

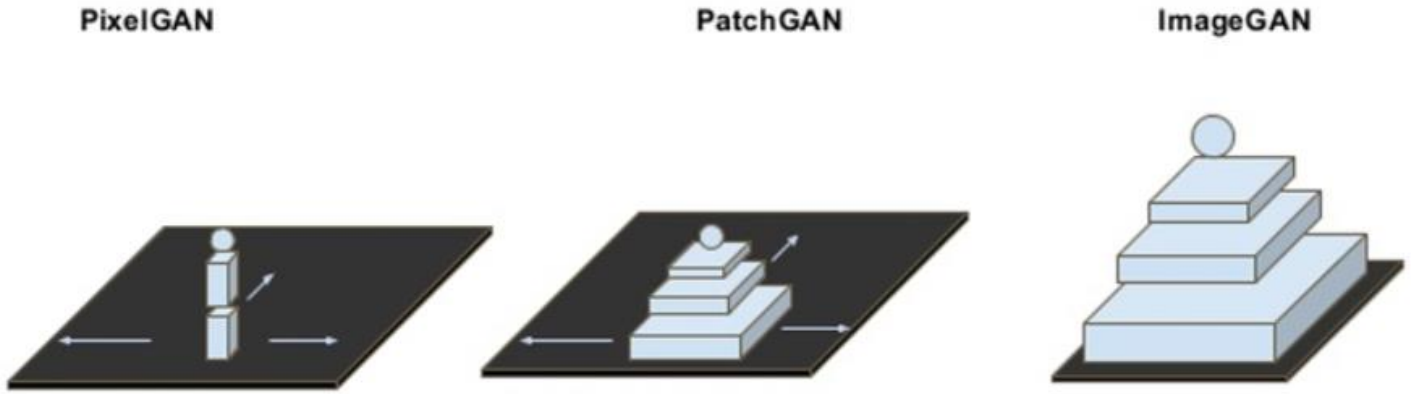
DiscoGAN과의 차이점 – Generator ***G***



Bottle-neck에서 콘텐츠의 디테일에 손실이 크지만,
보다 자유로운 변화된 이미지 생성이 가능

목적의 차이가 존재
CycleGAN의 경우 '컴퓨터 그래픽을 사진처럼 바꿔보자',
'고해상도의 이미지를 다루어 보자'라는 목적으로 시작

Network Architecture – Discriminator ***D***



$$G^* = \arg\min_G \max_D \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{L1}(G)$$

Discriminator에서 사용한 Architecture는 PatchGAN이다.

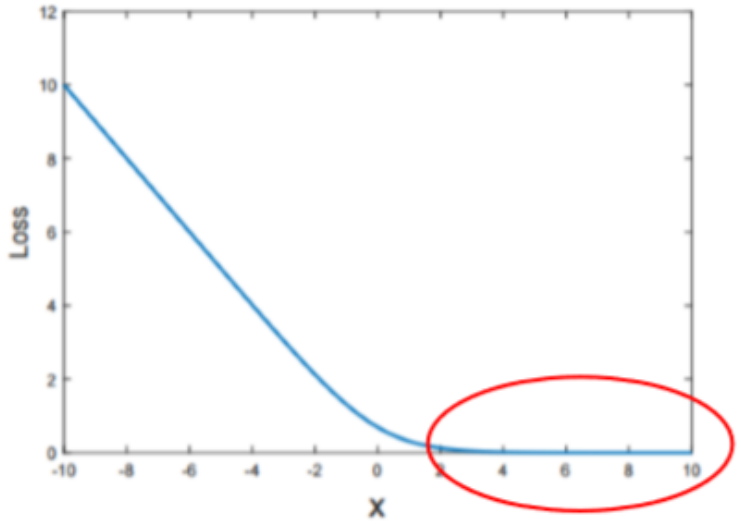
전체 이미지에 대해서 진짜/가짜를 구별 – ImageGAN
특정 크기의 patch 단위에 대하여 진짜/가짜 구별 – PatchGAN

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, Alexei A. Efros
Image-to-Image Translation with Conditional Adversarial Networks

Training details - Loss

- GANs with cross-entropy loss

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



Vanishing gradients

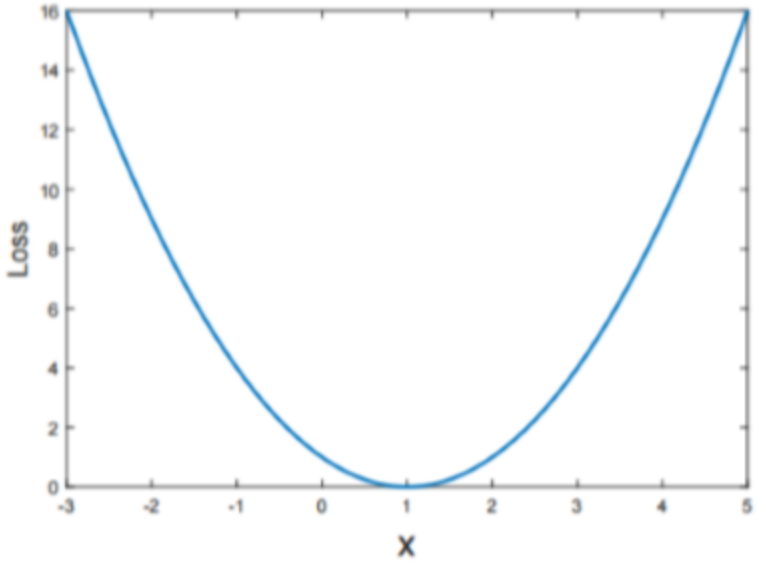
원래 GAN의 Loss는 Vanishing gradients로 인해서 학습과정에 어려움이 생김

Training details - Loss

- Least square GANs [Mao et al. 2016]

  Stable training + better results

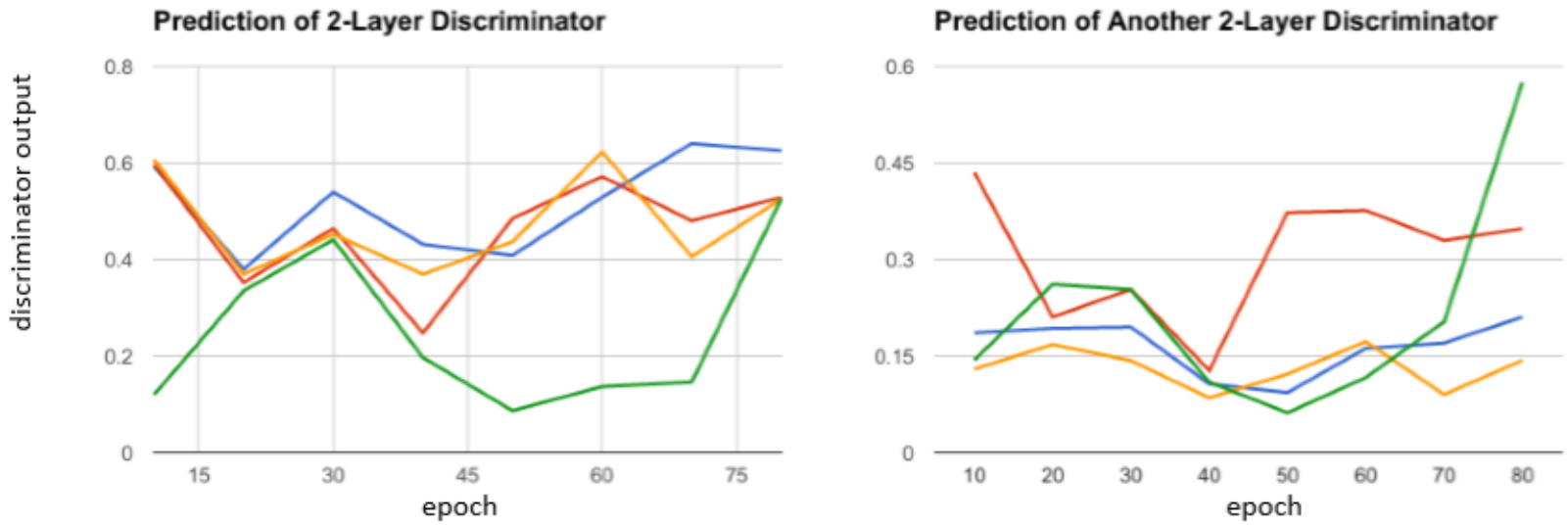
$$\mathcal{L}_{\text{LSGAN}}(G, D_Y, X, Y) = \mathbb{E}_{y \sim p_{\text{data}}(y)}[(D_Y(y) - 1)^2] + \mathbb{E}_{x \sim p_{\text{data}}(x)}[D_Y(G(x))^2],$$



여러가지 다른 형태의 loss를 가진 GAN들이 있었으나,
휴리스틱하게 선택

X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. P. Smolley.
Least squares generative adversarial networks. In CVPR. IEEE, 2017

Training details – replay buffer

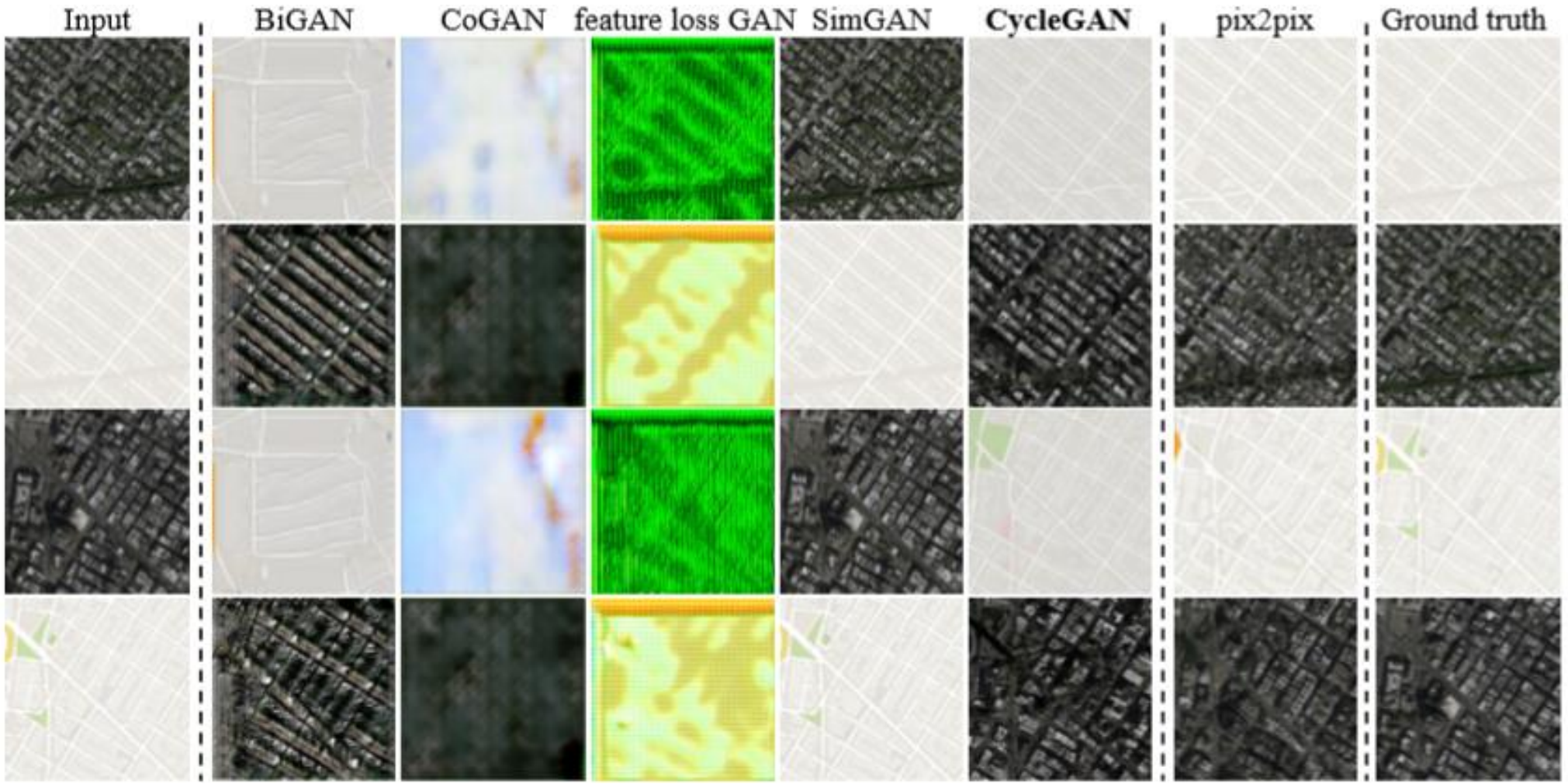


불안정성을 해결하기 위해서 두가지 솔루션

1> D를 여러 개 생성하여 평균하여 결과 제시, 메모리 소모가 커지는 단점

2> 이전에 G가 생성했던 이미지를 D에게 주기적으로 제시 -> 예전 G가 어떻게 행동했는지 D가 대응하면서 보다 안정정으로 학습이 진행

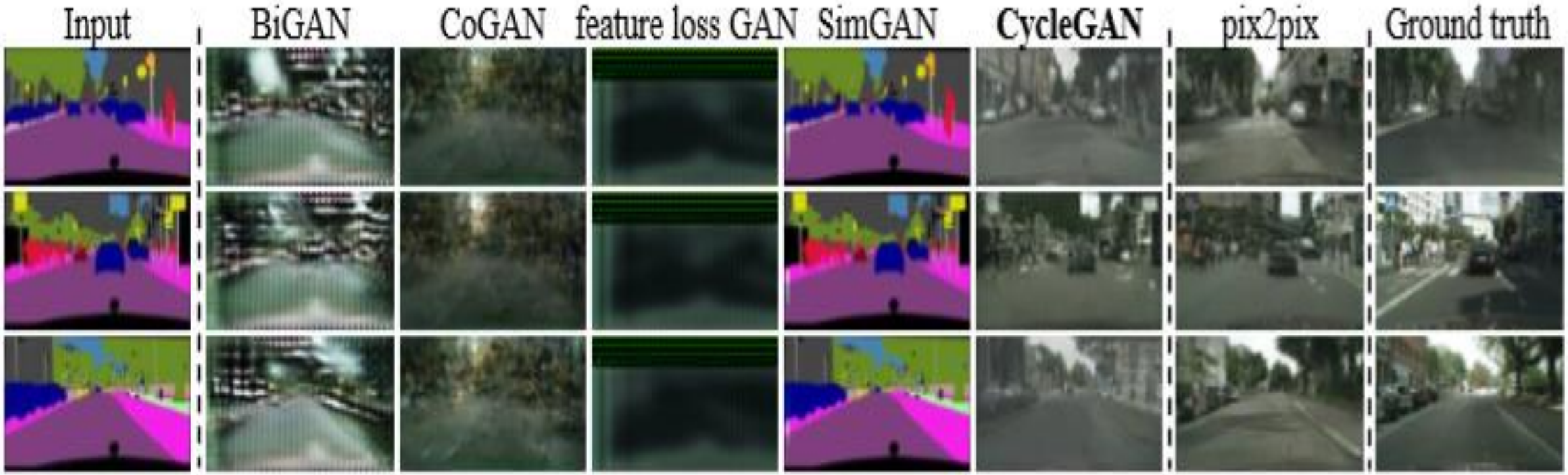
Evaluation Metrics – AMT perceptual studies



Evaluation Metrics – AMT perceptual studies

Loss	Map → Photo % Turkers labeled *real*	Photo → Map % Turkers labeled *real*
CoGAN [32]	0.6% ± 0.5%	0.9% ± 0.5%
BiGAN/ALI [9, 7]	2.1% ± 1.0%	1.9% ± 0.9%
SimGAN [46]	0.7% ± 0.5%	2.6% ± 1.1%
Feature loss + GAN	1.2% ± 0.6%	0.3% ± 0.2%
CycleGAN (ours)	**26.8% ± 2.8%**	**23.2% ± 3.4%**

Evaluation Metrics – FCN score / Semantic segmentation metrics

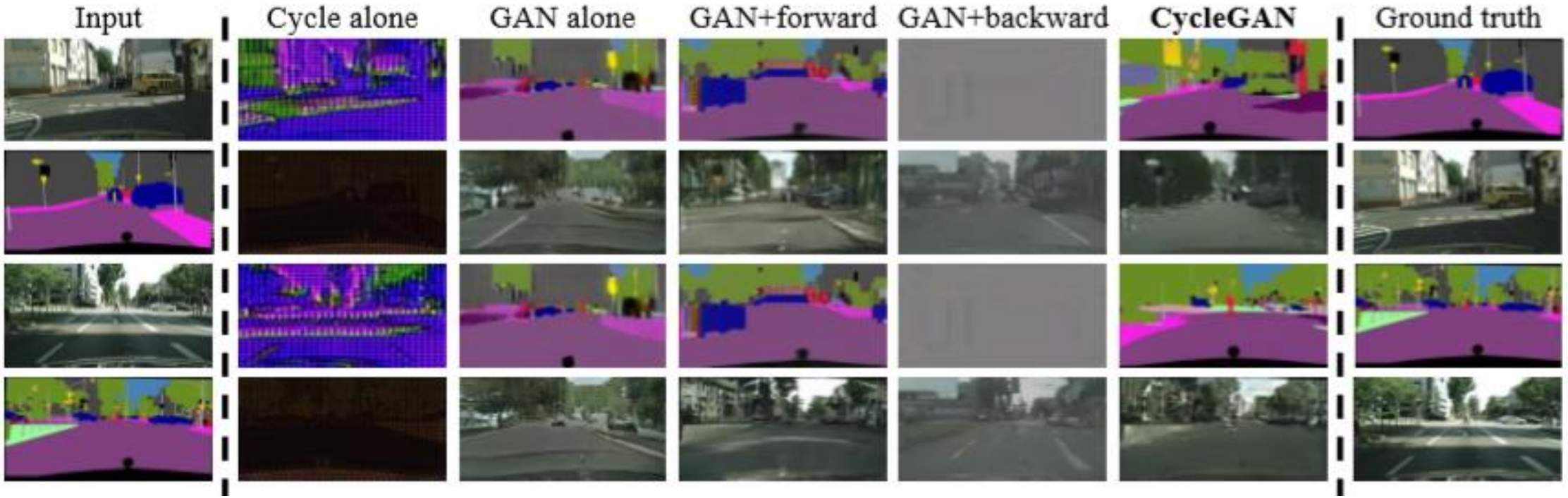


Evaluation Metrics – FCN score / Semantic segmentation metrics

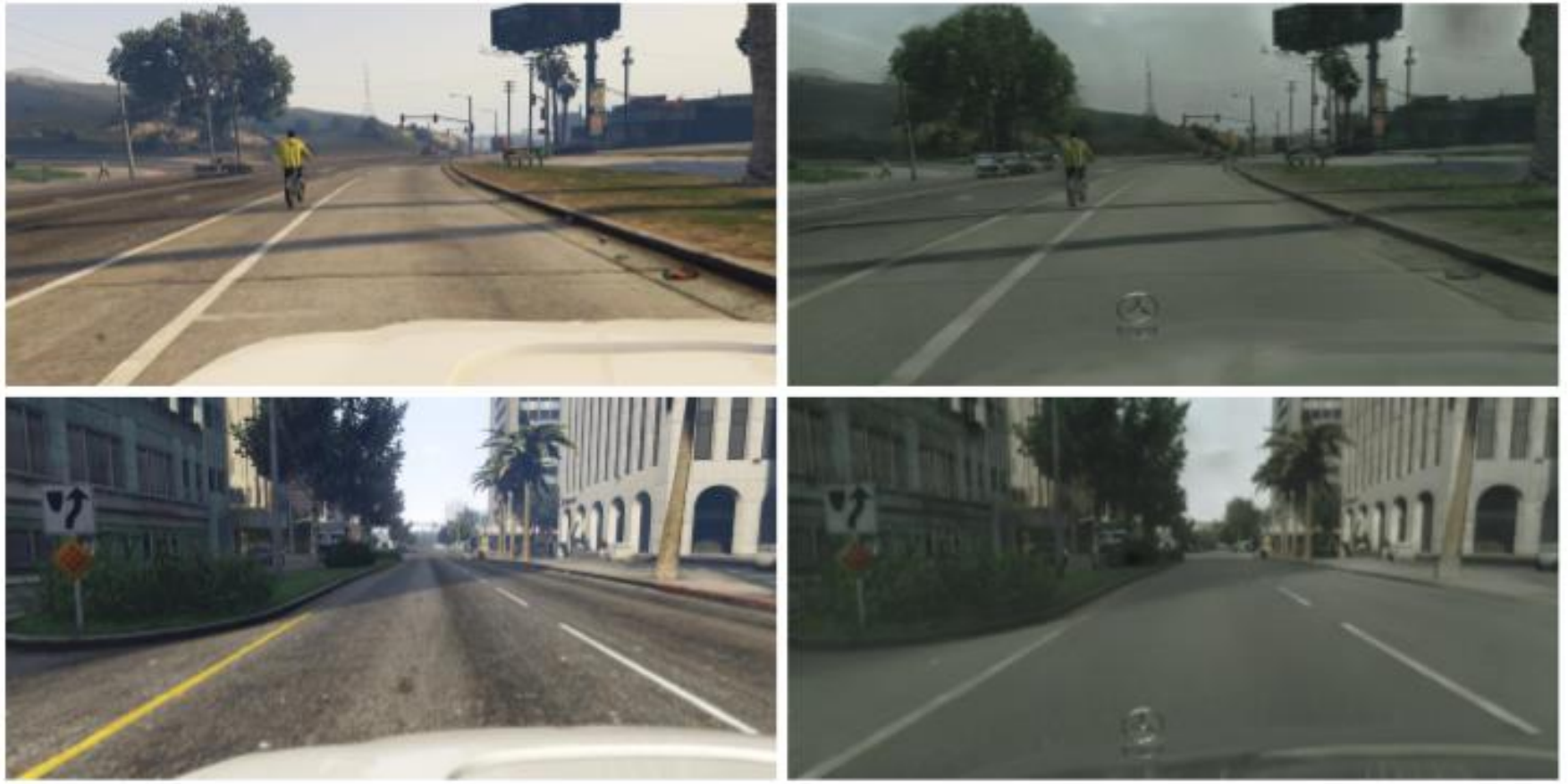
Loss	Per-pixel acc.	Per-class acc.	Class IOU
CoGAN [32]	0.40	0.10	0.06
BiGAN/ALI [9, 7]	0.19	0.06	0.02
SimGAN [46]	0.20	0.10	0.04
Feature loss + GAN	0.06	0.04	0.01
CycleGAN (ours)	**0.52**	**0.17**	**0.11**
pix2pix [22]	0.71	0.25	0.18

더 여러가지 결과가 제시 되어있지만, 간단하게 말하면 pix2pix보다는 성능이 좋지 않지만 다른 baseline들 보다는 성능이 많이 좋다.

Analysis of the loss function



Application on Domain Adaptation



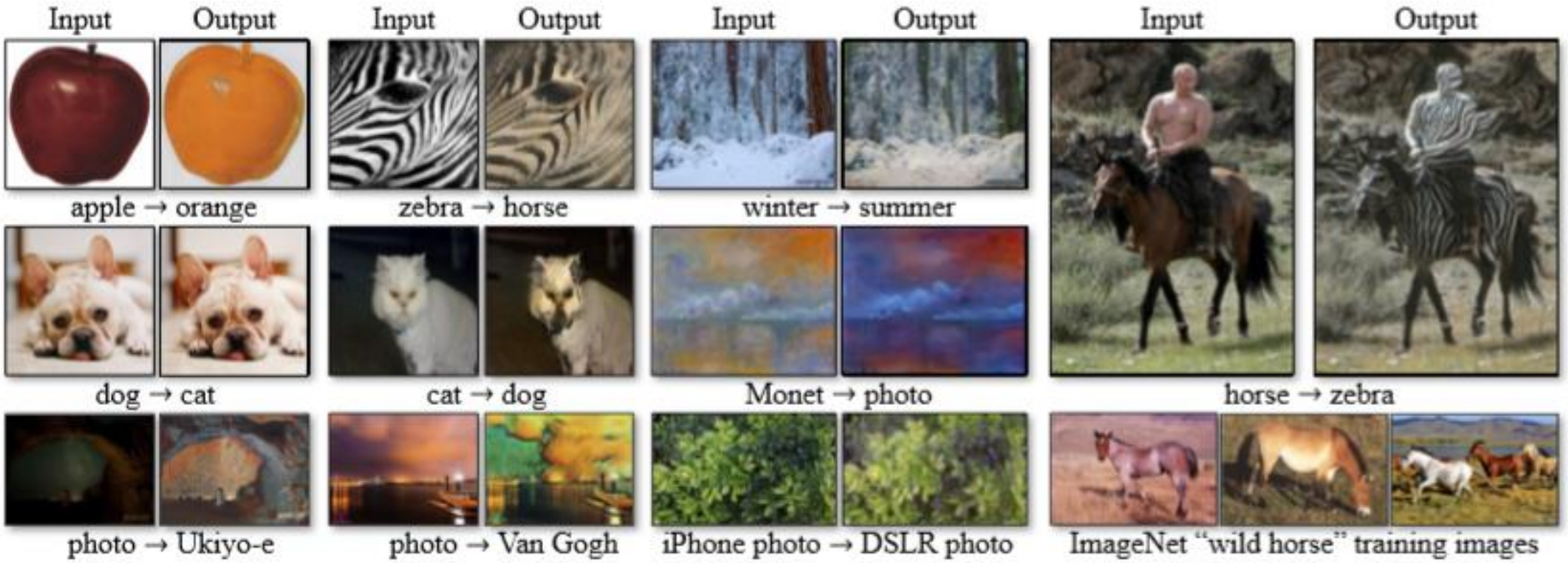
Application on Domain Adaptation

Train on CycleGAN data

Test on real images

	Per-class accuracy	Per-pixel accuracy
Oracle (Train and test on Real)	60.3	93.1
Train on CG, test on Real	17.9	54.0
FCN in the wild [Hoffman et al.]	27.1 (+6.0)	-
Train on CycleGAN, test on Real	**34.8 (+16.9)**	**82.8**

Failure cases



Q & A

구현 코드 링크: https://junyanz.github.io/CycleGAN/
(Pytorch , Torch)