Generative Adversarial Networks

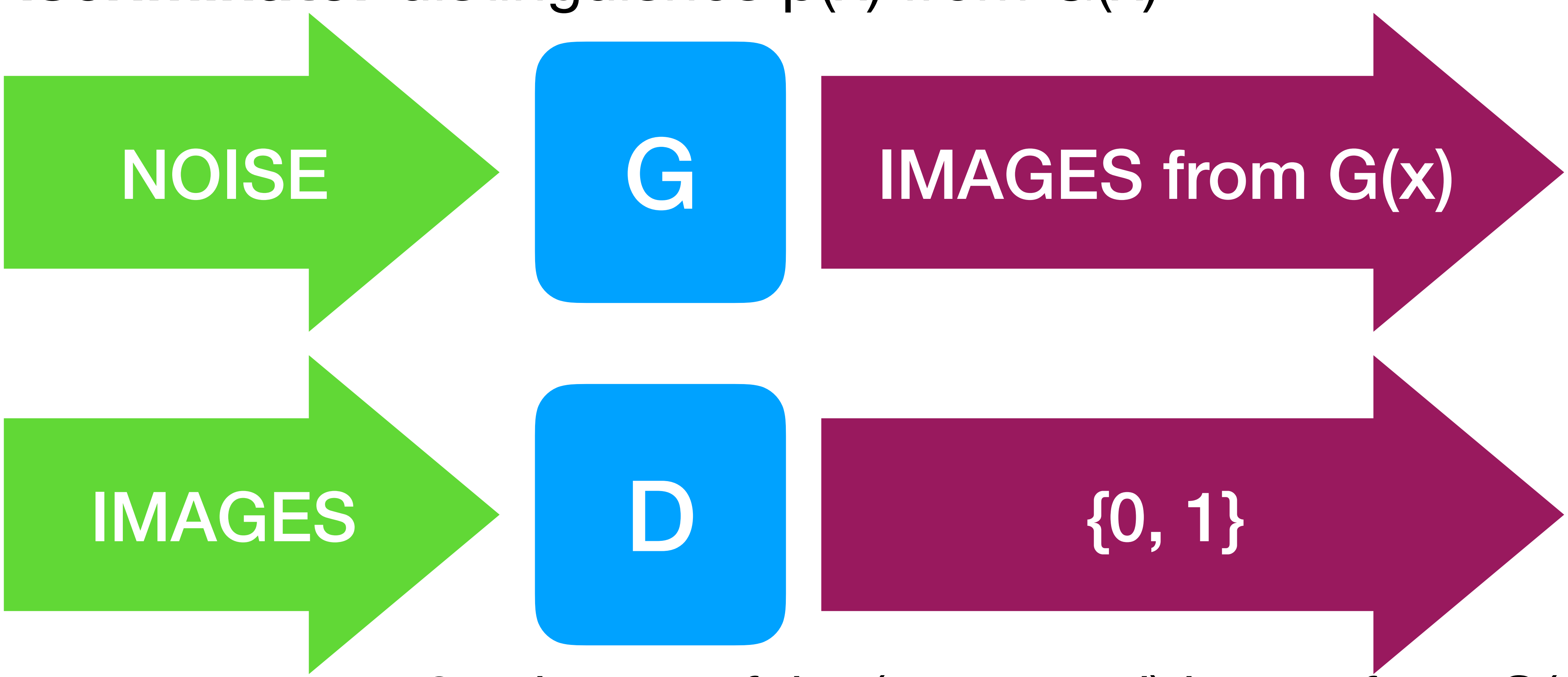
Maxwell Horton
mchorton@cs.washington.edu

Introduction

- Used to generate data that follows some target distribution
- Two Neural Networks (a **Generator** and a **Discriminator**) working together
- Naturally enforces similarity between Generator output and a training set distribution without complicated loss
- State of the Art for Image Generation

Key Idea

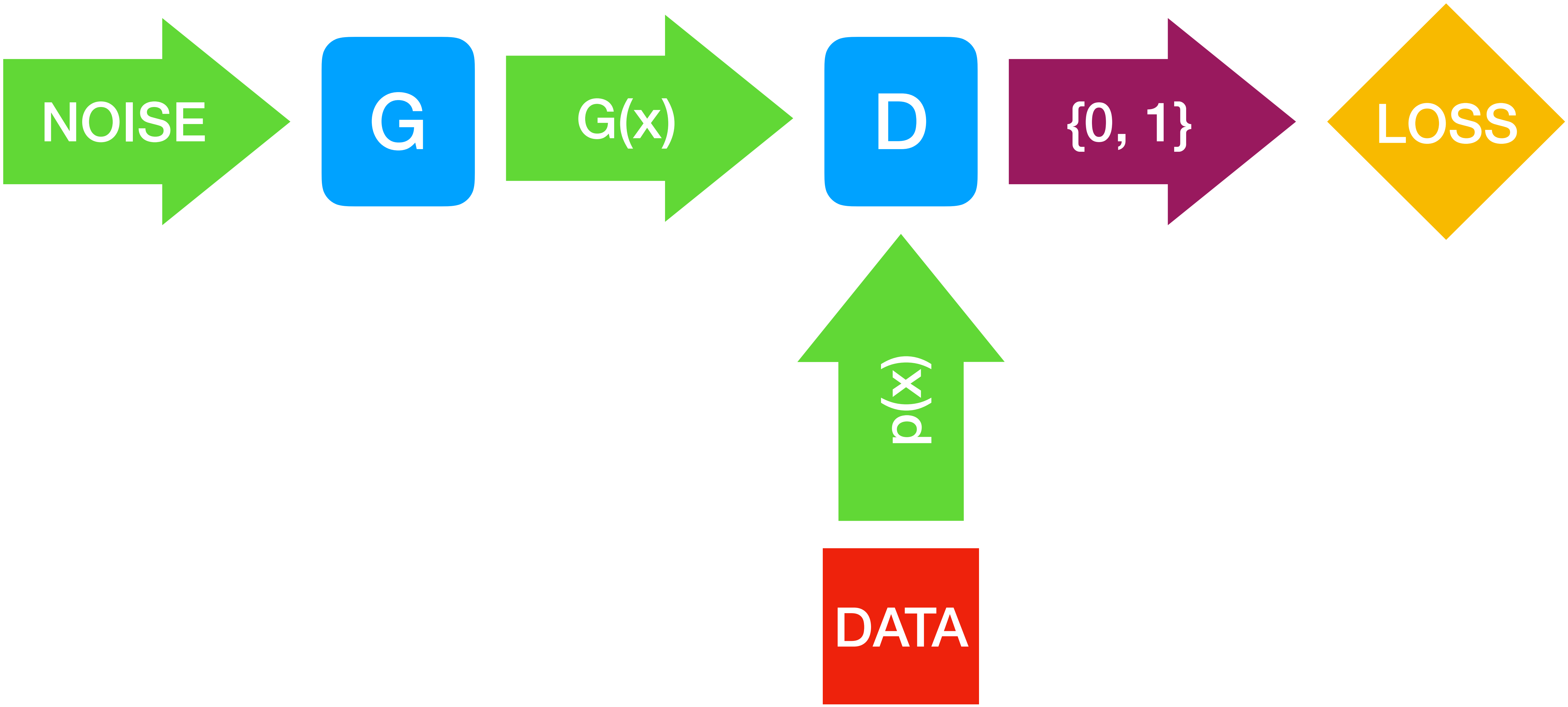
- **Generator** outputs data G(x), hoping to match some target p(x)
- **Discriminator** distinguishes p(x) from G(x)



0 - denotes fake (generated) image from G(x)
1 - denotes real image from p(x)

Key Idea

- **Generator** and **Discriminator** learn from each other



$$\min_G \max_D V(D, G) = \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z} \sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

Considerations

Pros	Cons
Naturally enforces $G(x) \sim p(x)$	Usually need additional losses
Often a Unique Optimum	Difficult to Train in Practice
State of the art for image generation	

Exploration of GANs

- **conditional GAN (cGAN):** $G(x|y) \sim p(x|y)$
- **pix2pix / CycleGAN:** image-to-image translation
- **SeGAN:** segmentation, generation of occluded patches
- **Wasserstein GAN:** training algorithm
- **DRAGAN:** training algorithm

… and over 100 more: https://github.com/hindupuravinash/the-gan-zoo

Training Tips

- Don’t give up
- Normalize image inputs to [-1, 1]
- Label Smoothing: Real label in [0.7, 1.2], Fake label in [0.0, 0.3]
- If successful, **Discriminator** loss decreases steadily with low variance
- Give the **Generator** Gaussian noise
- More tips: https://github.com/soumith/ganhacks

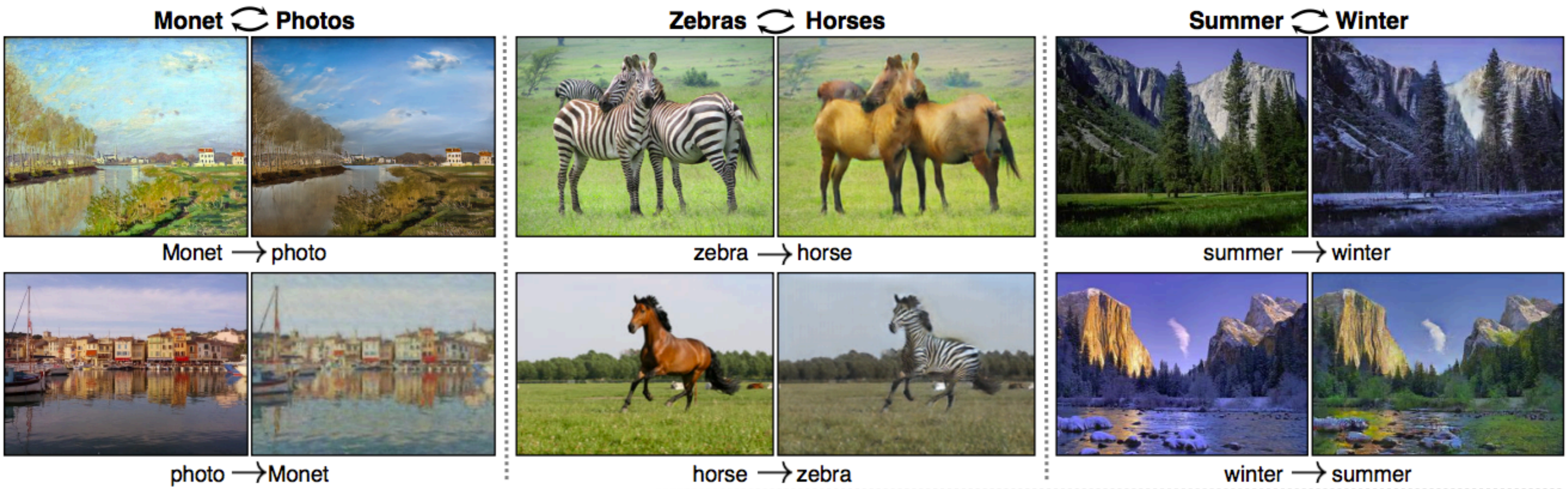
Sample Application:

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

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

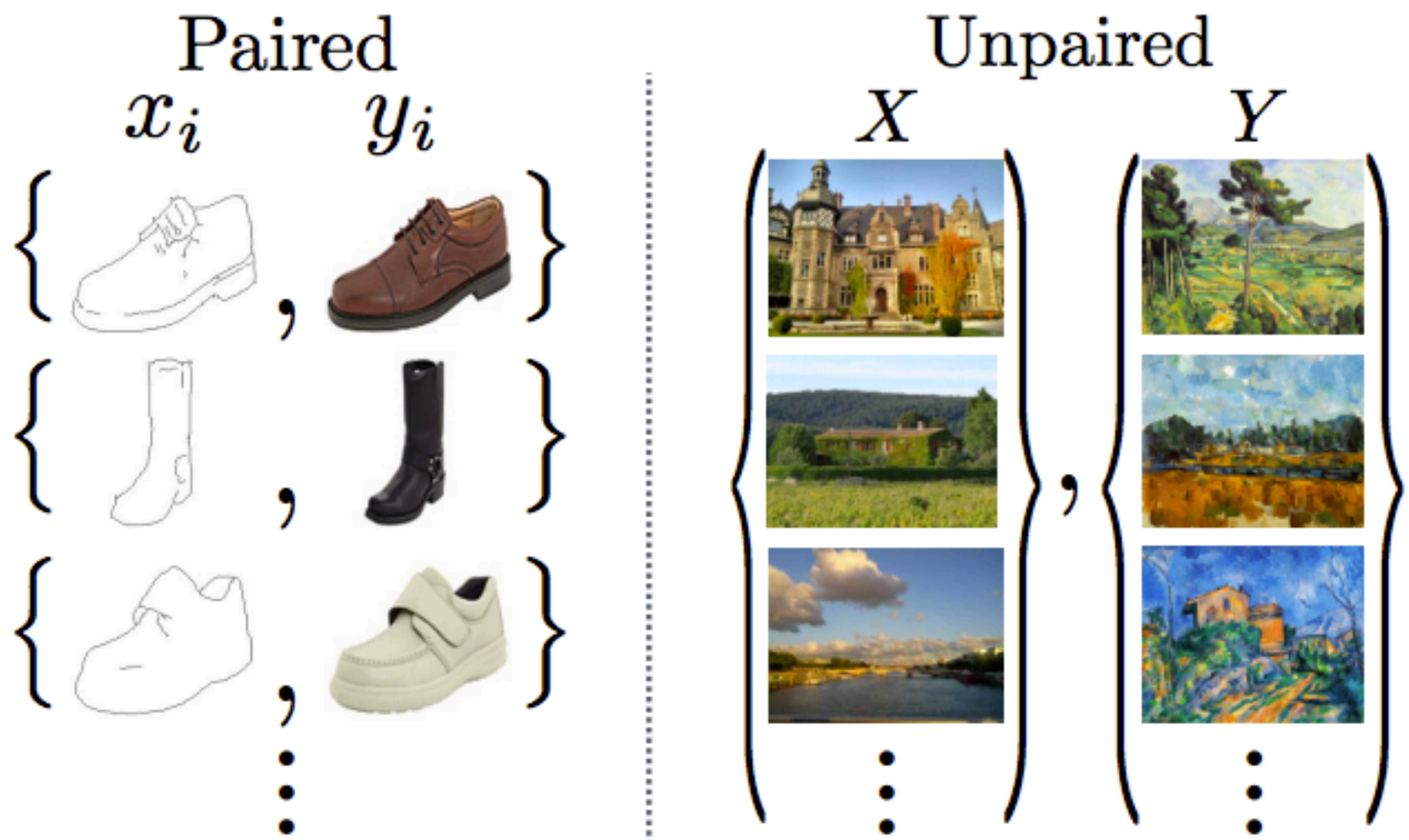
Authors: **Jun-Yan Zhu*** **Taesung Park***
Phillip Isola **Alexei A. Efros**

Image-to-Image Translation



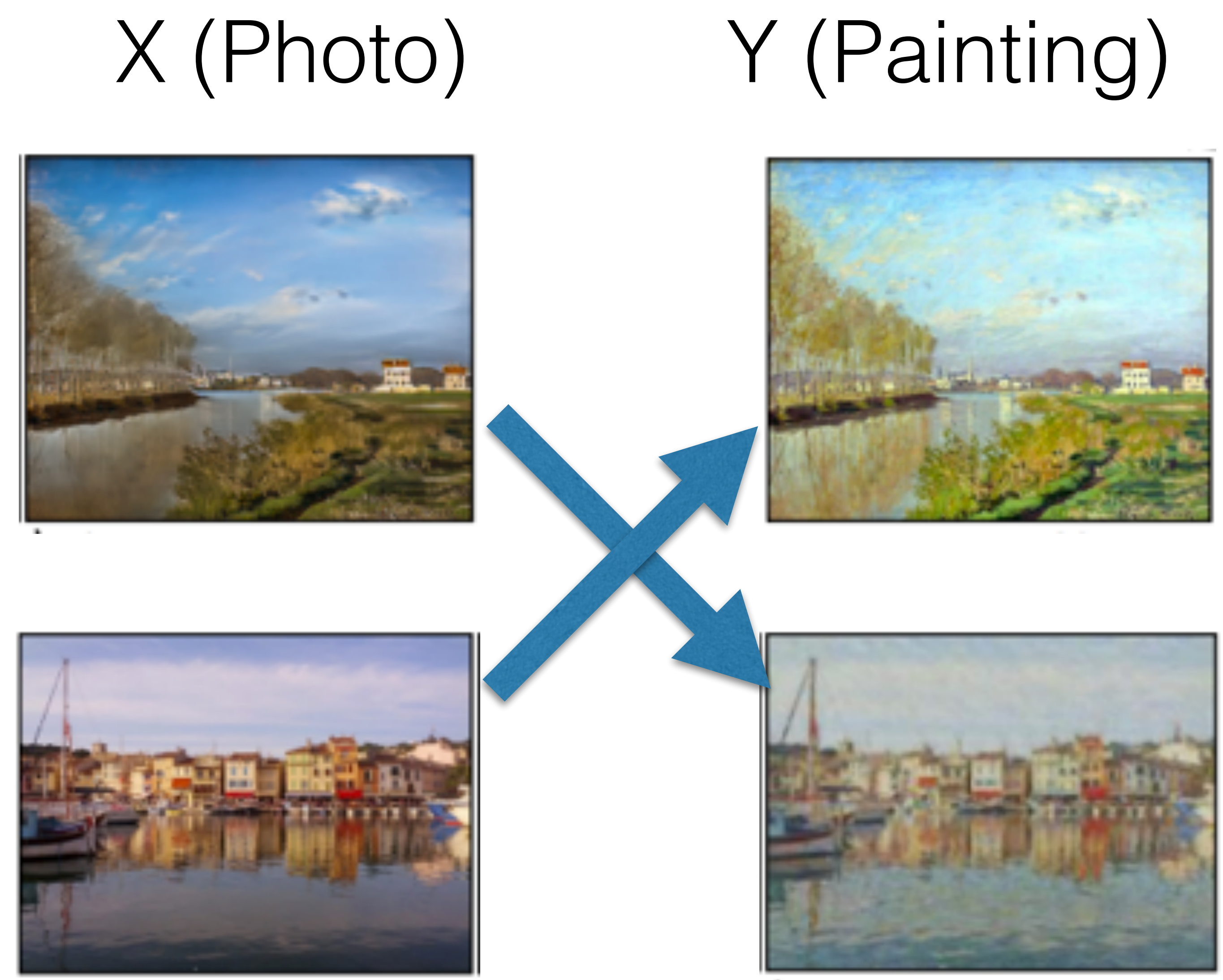
Goals

- Translate X->Y without paired examples
- Goal: learn some G:X->Y, G(X) matches Y
- Jointly learn inverse transformation F:Y->X



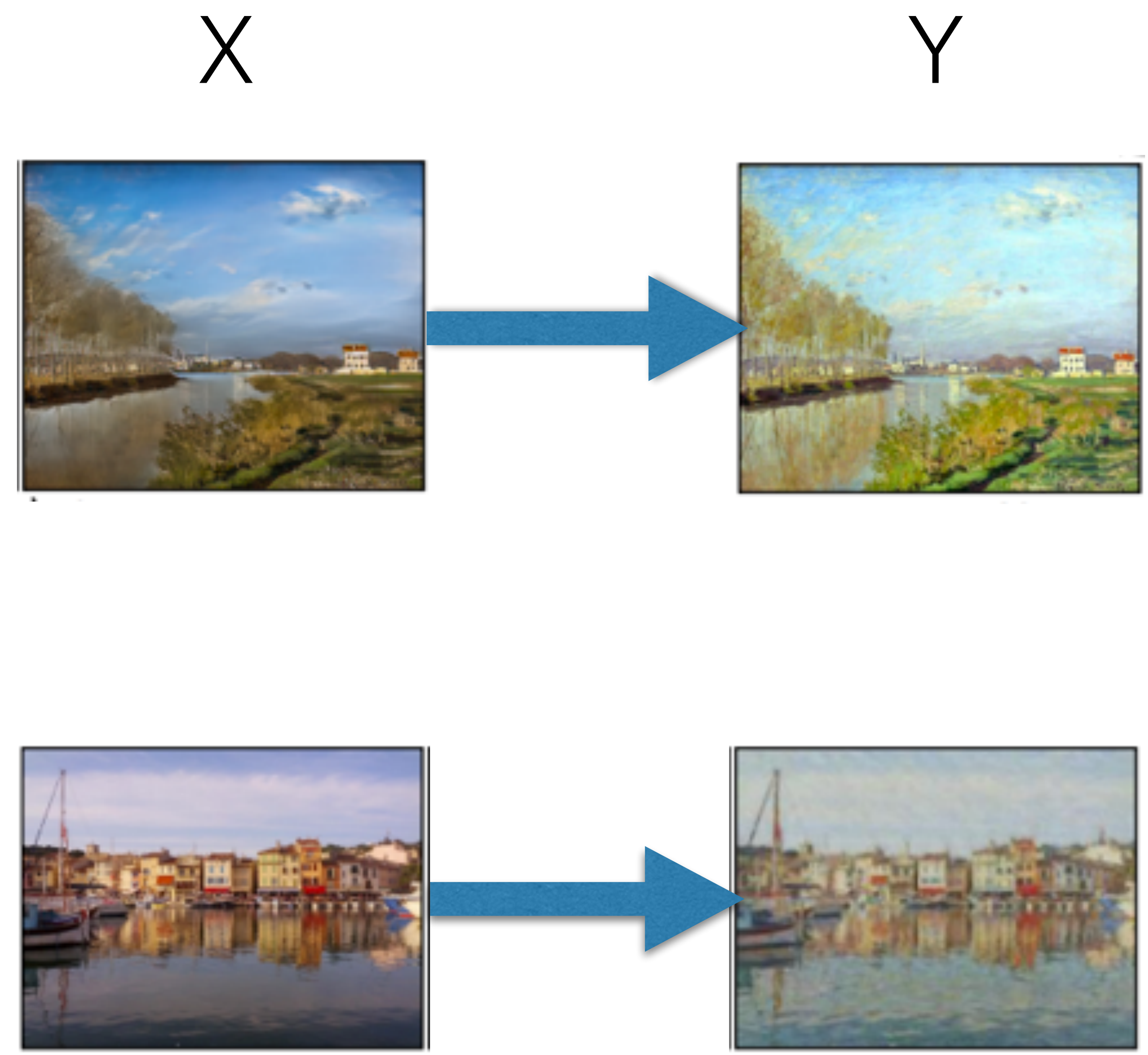
Cycle Consistency

- G: X -> Y
- What if G doesn't encode a meaningful transformation?



Cycle Consistency

- G: X -> Y, F: Y -> X
- Cycle consistency loss encourages F(G(x)) = x, G(F(y)) = y



Formulation

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

(likewise for F)

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

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

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

Evaluation

- AMT Studies; Real/Fake
- FCN Score
  - predict a label map for a generated photo
  - should detect ‘car on road’ from generated photo

AMT Results

Loss	**Map → Photo** % Turkers labeled *real*	**Photo → Map** % Turkers labeled *real*
CoGAN [27]	$0.6\% \pm 0.5\%$	$0.9\% \pm 0.5\%$
BiGAN [6, 5]	$2.1\% \pm 1.0\%$	$1.9\% \pm 0.9\%$
Pixel loss + GAN [41]	$0.7\% \pm 0.5\%$	$2.6\% \pm 1.1\%$
Feature loss + GAN	$1.2\% \pm 0.6\%$	$0.3\% \pm 0.2\%$
CycleGAN (ours)	$\mathbf{26.8\% \pm 2.8\%}$	$\mathbf{23.2\% \pm 3.4\%}$

Table 1: AMT "real vs fake" test on maps↔aerial photos.

FCN Results

Loss	Per-pixel acc.	Per-class acc.	Class IOU
CoGAN [27]	0.40	0.10	0.06
BiGAN [6, 5]	0.19	0.06	0.02
Pixel loss + GAN [41]	0.20	0.10	0.0
Feature loss + GAN	0.07	0.04	0.01
CycleGAN (ours)	**0.52**	**0.17**	**0.11**
pix2pix [18]	0.71	0.25	0.18

Table 2: FCN-scores for different methods, evaluated on Cityscapes labels→photos.

Loss	Per-pixel acc.	Per-class acc.	Class IOU
CoGAN [27]	0.45	0.11	0.08
BiGAN [6, 5]	0.41	0.13	0.07
Pixel loss + GAN [41]	0.47	0.11	0.07
Feature loss + GAN	0.50	0.10	0.06
CycleGAN (ours)	**0.58**	**0.22**	**0.16**
pix2pix [18]	0.85	0.40	0.32

Table 3: Classification performance of photo→labels for different methods on cityscapes.

Ablations

Loss	Per-pixel acc.	Per-class acc.	Class IOU
Cycle alone	0.22	0.07	0.02
GAN alone	0.52	0.11	0.08
GAN + forward cycle	**0.55**	**0.18**	**0.13**
GAN + backward cycle	0.41	0.14	0.06
CycleGAN (ours)	0.52	0.17	0.11

Table 4: Ablation study: FCN-scores for different variants of our method, evaluated on Cityscapes labels→photos.

Loss	Per-pixel acc.	Per-class acc.	Class IOU
Cycle alone	0.10	0.05	0.02
GAN alone	0.53	0.11	0.07
GAN + forward cycle	0.49	0.11	0.07
GAN + backward cycle	0.01	0.06	0.01
CycleGAN (ours)	**0.58**	**0.22**	**0.16**

Table 5: Ablation study: classification performance of photos→labels for different losses, evaluated on Cityscapes.

$$\mathbb{E}_{x\sim p_{\text{data}}(x)}[\|F(G(x))-x\|_1]$$ Forward

$$\mathbb{E}_{y\sim p_{\text{data}}(y)}[\|G(F(y))-y\|_1]$$ Backward

Results

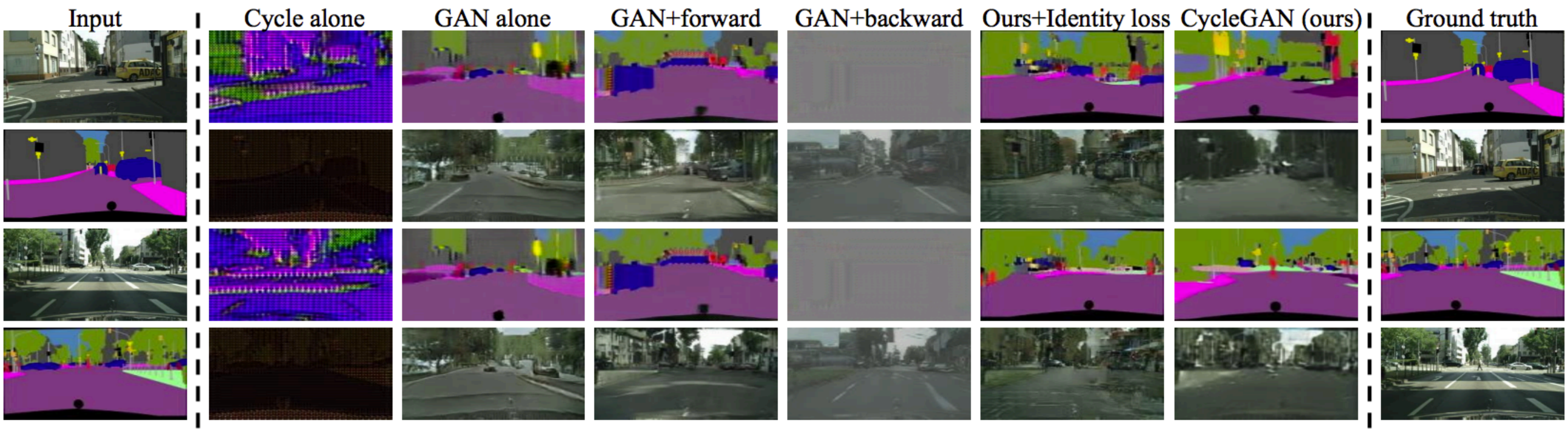
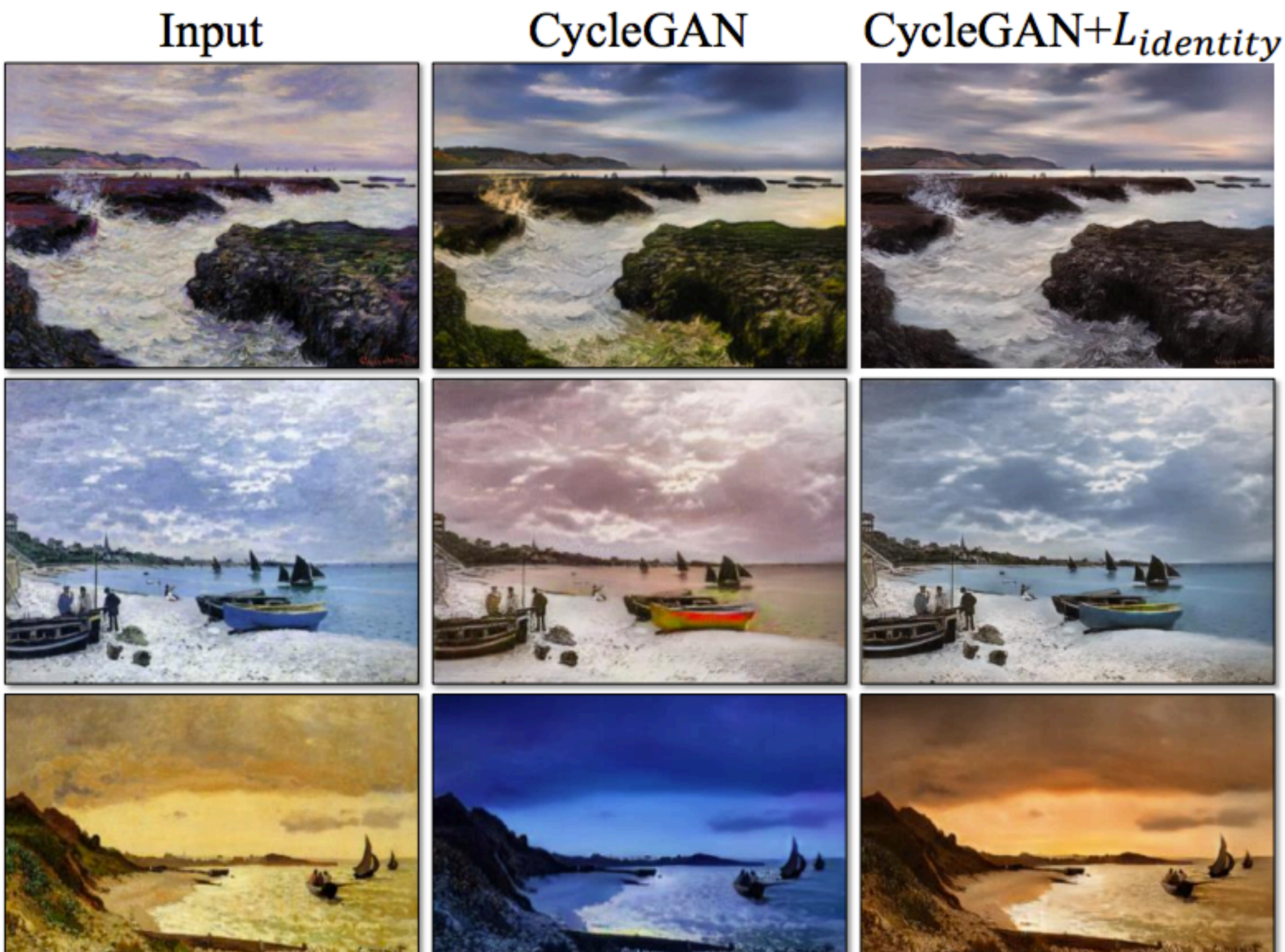


Figure 7: Different variants of our method for mapping labels↔photos trained on cityscapes. From left to right: input, cycle-consistency loss alone, adversarial loss alone, GAN + forward cycle-consistency loss ($F(G(x)) \approx x$), GAN + backward cycle-consistency loss ($G(F(y)) \approx y$), CycleGAN (our full method), and ground truth. Both *Cycle alone* and *GAN + backward* fail to produce images similar to the target domain. *GAN alone* and *GAN + forward* suffer from mode collapse, producing identical label maps regardless of the input photo.

Reconstructed Images

Input x | Generated image $G(x)$ | Reconstruction $F(G(x))$

Identity Mapping Loss



The End