

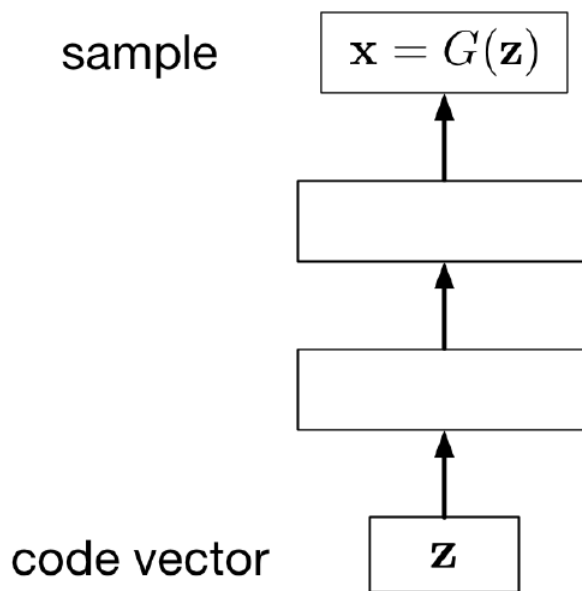


Lecture 20: Deep Generative Models IV: Variants of GANs

Xuming He
SIST, ShanghaiTech
Fall, 2019

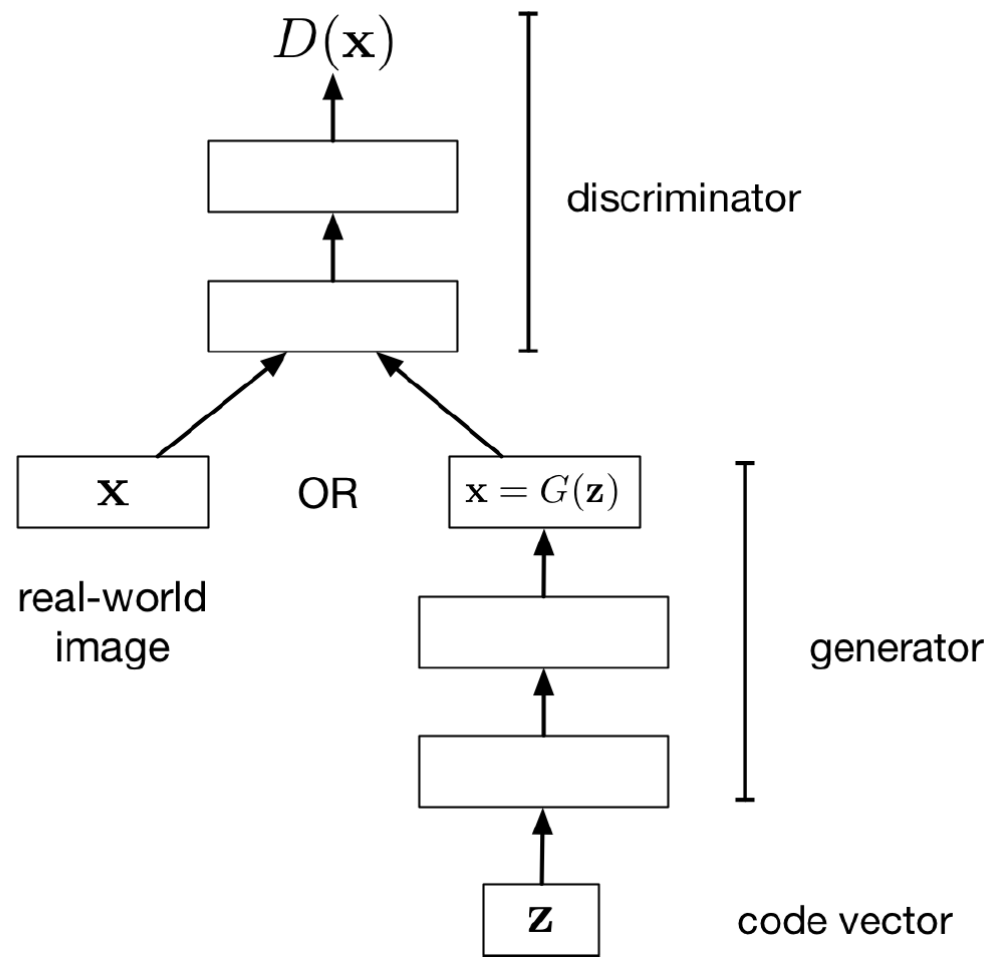
Review: Implicit Generative Models

- Implicitly define a probability distribution
- Start by sampling the code vector z from a fixed, simple distribution
- A generator network computes a differentiable function G mapping z to an x in data space



Review: Adversarial Learning

- Adversarial loss



Review: Learning procedure

■ Minimax objective function

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

Alternate between:

1. **Gradient ascent** on discriminator

$$\max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

2. **Gradient descent** on generator

$$\min_{\theta_g} \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z)))$$

Why are GANs different

- GAN optimization is fundamentally different from other neural networks
 - Gradient descent is relatively well established
 - Loss functions don't change much
 - Most deep learning research has focused on new components to use within the standard single-player framework (dropout, batchnorm, relu, etc.)
- GANs are an area of research where the objectives and descent methods are still in flux

Potential causes of instability

- Several theories on why GANs are hard to train.
- Main contributing factors:
 - Adversarial optimization is a more general, harder problem than single-player optimization.
 - Two player games do not always converge using gradient descent.
 - There is a stationary point but no guarantee of reaching it.
 - Simultaneous updates require a careful balance between the two players.
 - Generated points tend to "herd" to probable regions, causing "mode collapse".
 - Discriminator is highly nonlinear, gradient tends to be noisy or non-informative.

Common failures

- There are several common types of GAN failures that provide intuition into ways to make GANs better.
 - "Mode collapse" GAN generates a subspace really well but doesn't cover the entire real distribution. For example, train on MNIST and it only generates threes and eights.
<https://www.youtube.com/watch?v=ktxhiKhWoEE>
 - Sometimes GANs enter into clear cycles. They seem to generate a single digit relatively well, then start generating a different digit, etc. Looks like "mode collapse" on a rotating set of samples, but it does not differentiate.
 - Sometimes hard to describe failures, but videos like this are relatively typical.
<https://www.youtube.com/watch?v=D5akt32hsCQ>

Outline

- Improving GAN training
 - WGANs
- Conditional GANs
 - Text-to-image: StackGANs
 - Image-to-image translation
- CycleGAN
 - Image-to-image translation with unpaired data

Acknowledgement: CMU, UofT, Stanford notes

Optimization techniques

- The main problem with GANs is that they are tricky to train
- There are many tricks to train them better
- Not every trick works all the time or in combination with other tricks
- Most papers claim to have the golden bullet
- Best current solution is really a combination of techniques

<https://github.com/soumith/ganhacks>

Wasserstein GAN

■ Recall the GAN's formulation

- Real data distribution P_r ;

Generator's distribution P_g , implemented as $x = G(z), z \sim P(z)$

$$\min_G \max_D V(D, G)$$

- Discriminator

$$-\mathbb{E}_{x \sim P_r} [\log D(x)] - \mathbb{E}_{x \sim P_g} [\log(1 - D(x))] \quad (1)$$

$D(x)$: the probability that x from the real data rather than generator.

- Generator

$$\mathbb{E}_{x \sim P_g} [\log(1 - D(x))] \quad \text{GAN}_0 \quad (2)$$

$$\mathbb{E}_{x \sim P_g} [-\log(D(x))] \quad \text{GAN}_1 \quad (3)$$

Wasserstein GAN

■ Let's focus on the generator training

- Generator

$$\mathbb{E}_{x \sim P_g} [\log(1 - D(x))] \quad \text{GAN}_0 \quad (2)$$

$$\mathbb{E}_{x \sim P_g} [-\log(D(x))] \quad \text{GAN}_1 \quad (3)$$

Problems [Goodfellow et al., 2014]:

- P1: “In practice, GAN_0 may not provide sufficient gradient for G to learn well”, GAN_1 is used instead. (log D trick)
- P2: “ G collapses too many values of z to the same value of x ” (Mode collapse in GAN_1)

Wasserstein GAN

■ P1:

In GAN₀, better discriminator leads to worse vanishing gradient in its generator

□ Reason:

$$D^*(x) = \frac{P_r(x)}{P_r(x) + P_g(x)}$$

$$L = \mathbb{E}_{x \sim P_r} [\log D(x)] + \mathbb{E}_{x \sim P_g} [\log(1 - D(x))]$$

$$\longrightarrow 2JS(P_r || P_g) - 2 \log 2$$

- If the supports of P_r and P_g almost have no overlap, then the JS divergence is 0 and there is no gradient info
- The probability that the support of P_r and P_g almost have no overlap is 1

Wasserstein GAN

■ P2:

GAN₁ is a conflicting/asymmetric objective, thus (1)unstable gradient (2) mode callapse

- Reason: GAN₁ equals to optimize

$$KL(P_g||P_r) - 2JS(P_g||P_r)$$

- Opposite signs for KL and JS
- Mode dropping KL divergence

$KL(P_g||P_r)$ assigns an high cost to generating fake looking samples, and an low cost on mode dropping;

$KL(P_r||P_g)$ assigns an high cost to not covering parts of the data, and an low cost on generating fake looking samples;

Wasserstein GAN

■ Re-think objective

① KL

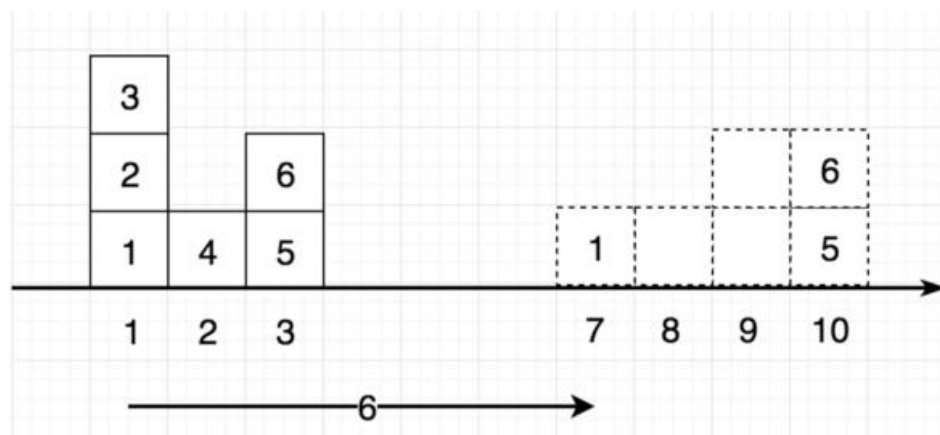
$$KL(P||Q) = \mathbb{E}_P \log \frac{P}{Q}$$

② JS

$$JS(P||Q) = \frac{1}{2}KL(P||\frac{P+Q}{2}) + \frac{1}{2}KL(Q||\frac{P+Q}{2})$$

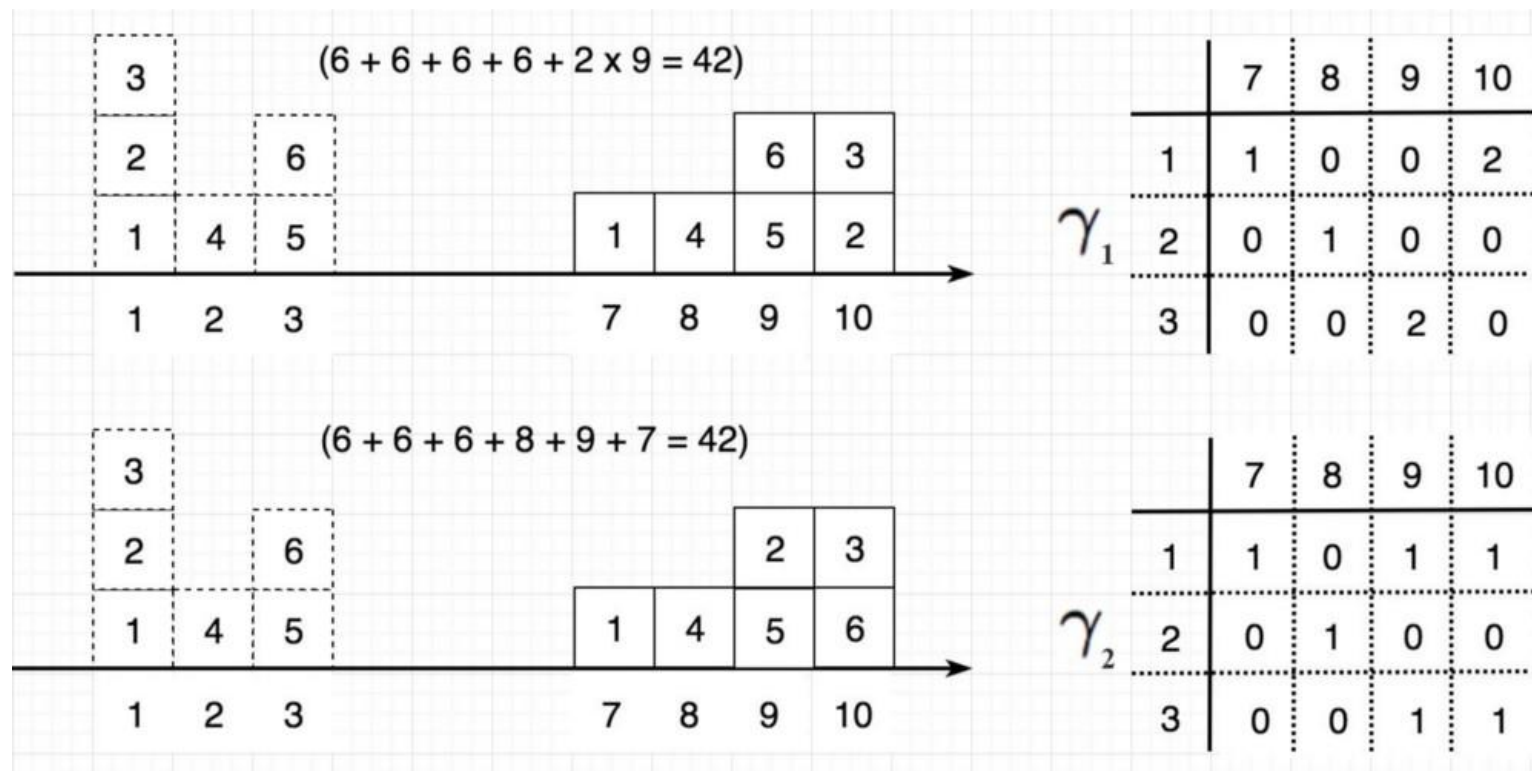
■ Let's use a different distance between two distributions

□ Earth-mover distance/Wasserstein metric



Wasserstein GAN

- Earth-mover distance
 - Different transportation plan



- The cost of the cheapest transportation plan

Wasserstein GAN

■ Formal definition

Wasserstein

$$W(P||Q) = \inf_{\gamma \in \Pi(P,Q)} \mathbb{E}_{(x,y) \sim \gamma} [||x - y||]$$

- $\Pi(P, Q)$ denotes the set of all joint distributions $\gamma(x, y)$ whose marginals are P and Q , respectively
- $\gamma(x, y)$ indicates a plan to transport “mass” from x to y , when deforming P into Q .

The Wasserstein (or Earth-Mover) distance is then the “cost” of the **optimal** transport plan

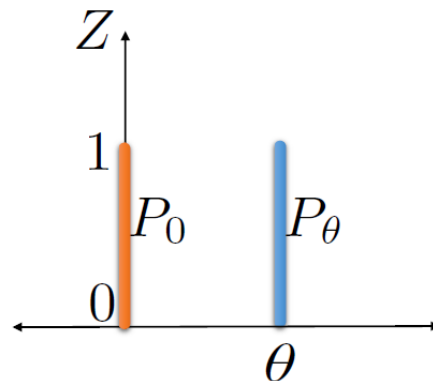
Wasserstein GAN

■ Examples of W-distance

P_0 : distribution of $(0, Z)$, where $Z \sim U[0, 1]$

P_θ : distribution of (θ, Z) , where θ is a single real parameter

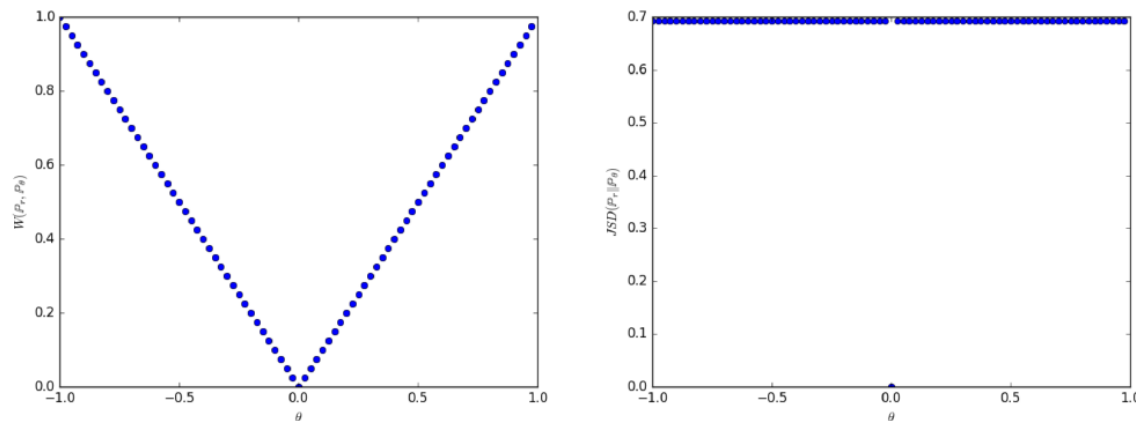
- $KL(P_0||P_\theta) = KL(P_\theta||P_0) = \begin{cases} +\infty & \text{if } \theta \neq 0 \\ 0 & \text{if } \theta = 0 \end{cases}$
- $JS(P_0||P_\theta) = \begin{cases} \log 2 & \text{if } \theta \neq 0 \\ 0 & \text{if } \theta = 0 \end{cases}$
- $W(P_0||P_\theta) = |\theta|$



(a) Distributions

Wasserstein GAN

■ Examples of W-distance



(b) Output of W and JS

① Observations

When the distributions are supported by low dimensional manifolds (such as P_r and P_g in GANs)

- KL or JS are binary, no meaningful gradient
- W is continuous and differentiable, hence always sensible

Wasserstein GAN

■ Use W-distance in GAN

- The infimum is highly intractable
- Wasserstein distance has a duality form

$$\begin{aligned} W(P_r, P_g) &= \sup_{\|f\|_L \leq 1} \mathbb{E}_{x \sim P_r}[f(x)] - \mathbb{E}_{x \sim P_g}[f(x)] \\ &= \frac{1}{K} \sup_{\|f\|_L \leq K} \mathbb{E}_{x \sim P_r}[f(x)] - \mathbb{E}_{x \sim P_g}[f(x)] \end{aligned}$$

where supremum is over all the K -Lipschitz functions

- Consider a w -parameterized family of functions $\{f_w\}_{w \in W}$ that are all K -Lipschitz

$$W(P_r, P_g) = \max_{w \in W} \mathbb{E}_{x \sim P_r}[f_w(x)] - \mathbb{E}_{x \sim P_g}[f_w(x)]$$

For example, $W = [-c, c]^l$

Wasserstein GAN

- Use W-distance in GAN
 - Loss for the discriminator

$$\mathbb{E}_{x \sim P_r} [f_w(x)] - \mathbb{E}_{x \sim P_g} [f_w(x)]$$

- Loss for the generator

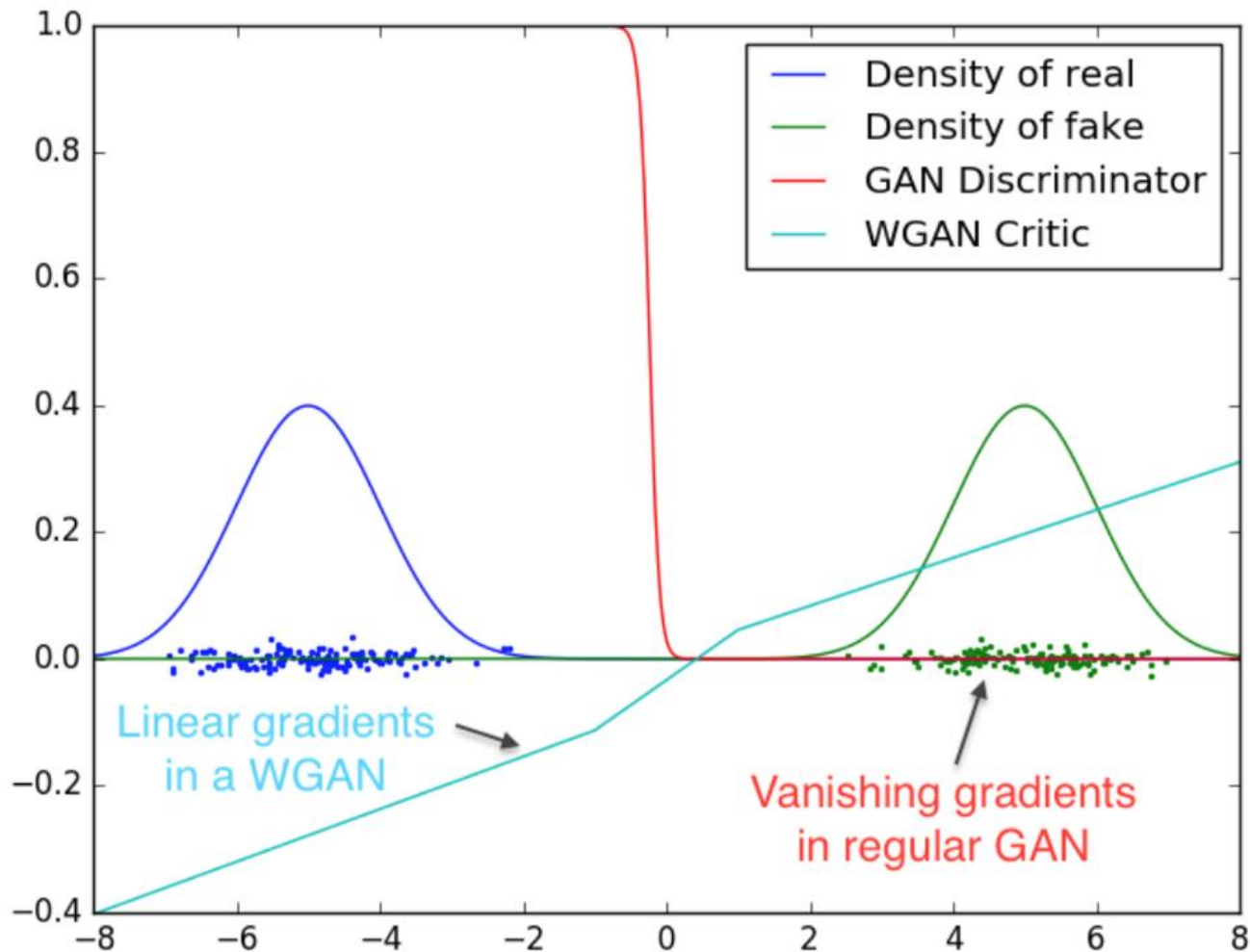
$$-\mathbb{E}_{x \sim P_g} [f_w(x)] = -\mathbb{E}_{z \sim p(z)} [f_w(g_\theta(z))]$$

- Main difference

- Remove the sigmoid of the last layer in D
 - Remove the log in the loss of D and G.
 - Clip the parameters of D in an interval centered at 0.

Wasserstein GAN

■ Benefits



Wasserstein GAN

■ Benefits

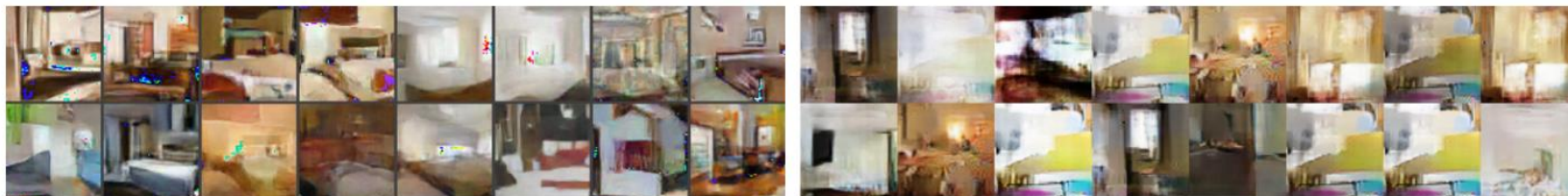
- A meaningful loss metric that correlates with the generator's convergence and sample quality. WGAN algorithm attempts to train the critic relatively well before each generator update, the loss function at this point is an estimate of the EM distance.
- It allows us to train the critic till optimality, and thus no longer need to balance generator and discriminator's capacity properly

A generator without batch normalization in DCGAN



- In no experiment did the authors see evidence of mode collapse

A generator constructed with MLP



Outline

- Improving GAN training
 - WGANs
- Conditional GANs
 - Text-to-image: StackGANs
 - Image-to-image translation
- CycleGAN
 - Image-to-image translation with unpaired data

Acknowledgement: CMU, UofT, Stanford notes

Conditional GANs

- Conditional GANs include a label and learn $P(X|Y)$
 - Add conditional variable y into G and D
 - Objective function

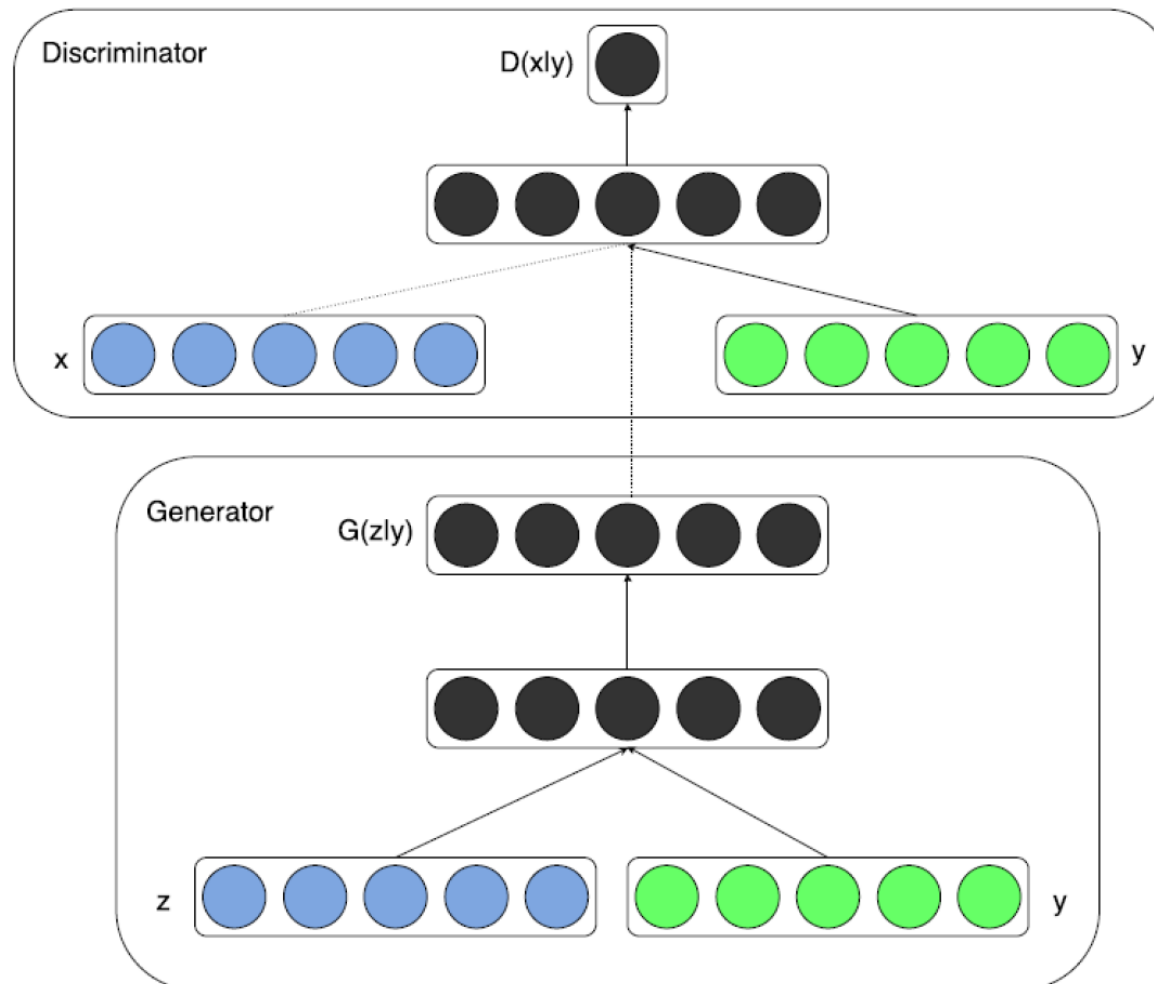
$$\min_G \max_D V(D, G) = \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}(\mathbf{x})} [\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z} \sim p_z(\mathbf{z})} [\log(1 - D(G(\mathbf{z})))].$$



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

Conditional GANs

- Model architecture



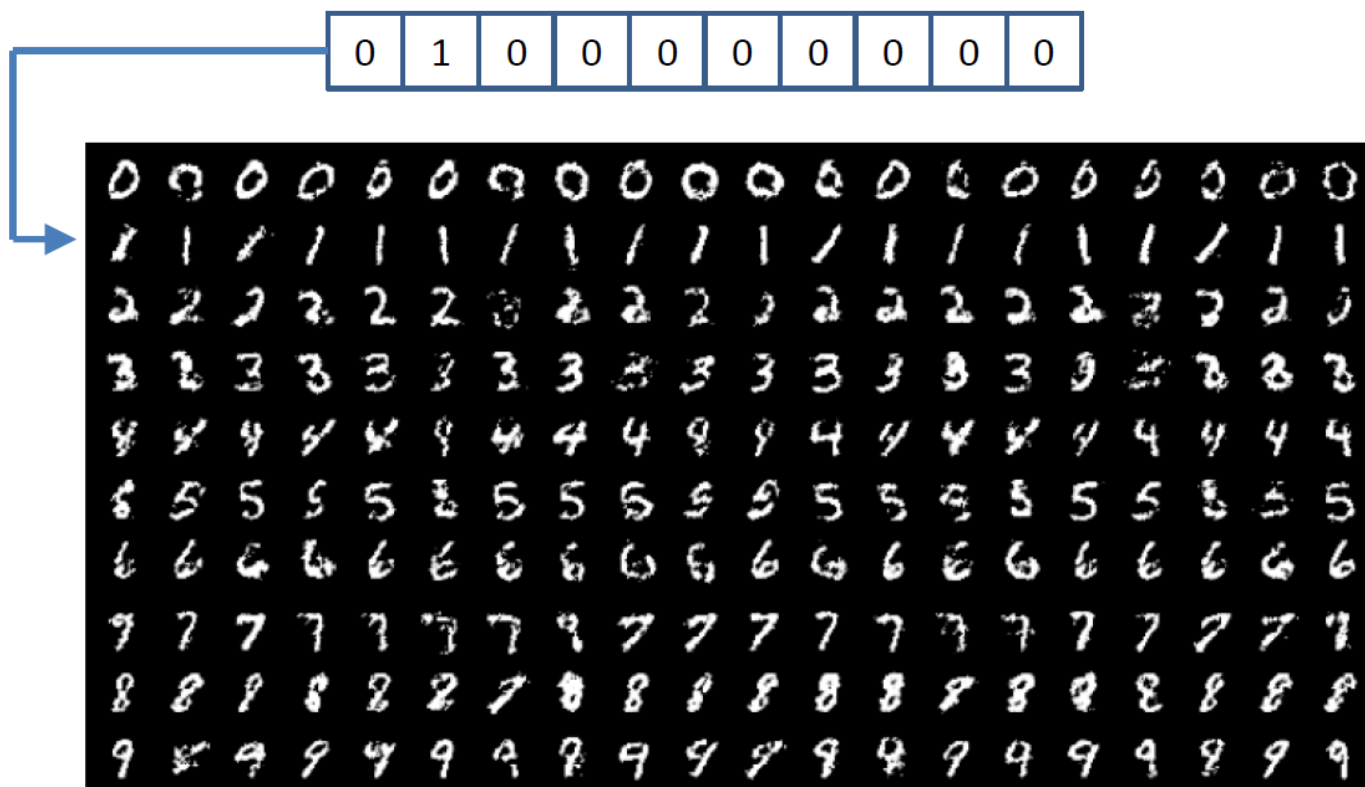
Conditional GANs

- Positive samples for D
 - True data + corresponding conditioning variable
- Negative samples for D
 - Synthesized data + corresponding conditioning variable
 - *True data + non-corresponding conditioning variable*

Conditional GANs

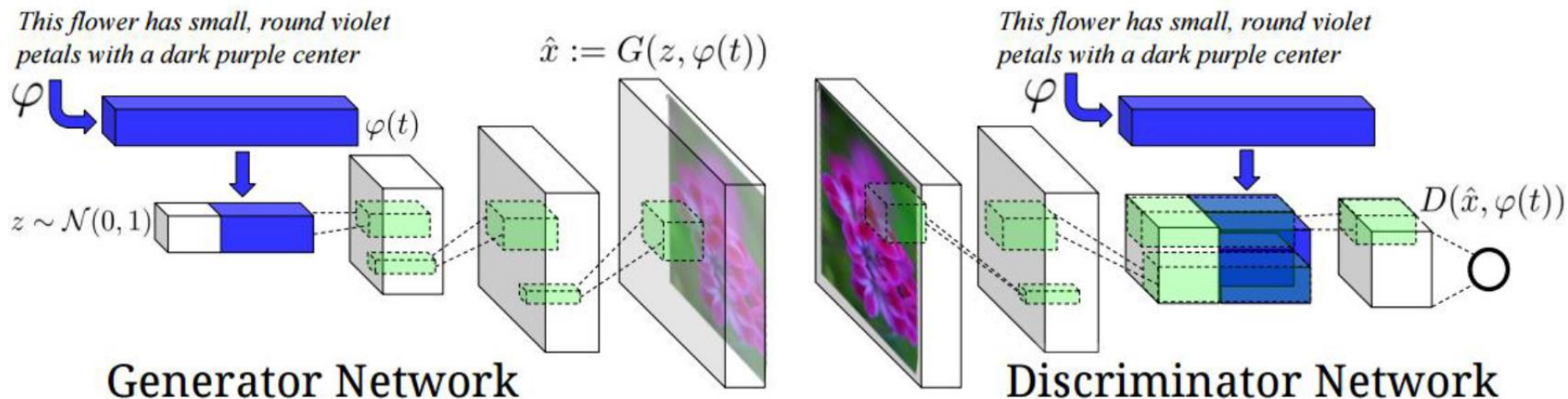
■ MNIST example

- Each row is conditioned on a different label.
- A single neural network to generate all 10 digits



Mirza and Osindero 2016

Text-to-Image synthesis



this small bird has a pink breast and crown, and black primaries and secondaries.



this magnificent fellow is almost all black with a red crest, and white cheek patch.



Reed et al 2015

StackGAN

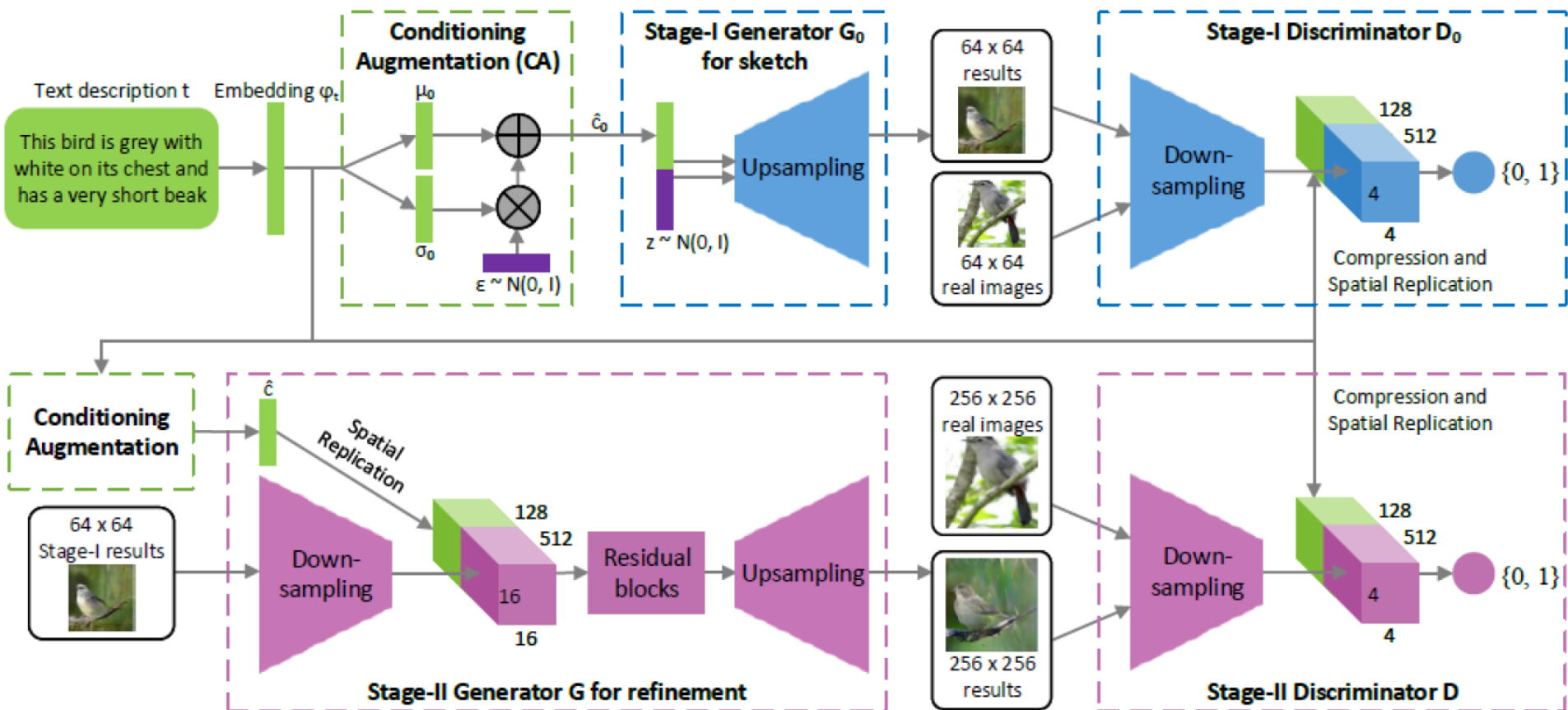
- A coarse-to-fine manner

Text description	This bird is blue with white and has a very short beak	This bird has wings that are brown and has a yellow belly	A white bird with a black crown and yellow beak	This bird is white, black, and brown in color, with a brown beak	The bird has small beak, with reddish brown crown and gray belly	This is a small, black bird with a white breast and white on the wingbars.	This bird is white black and yellow in color, with a short black beak
Stage-I images							
Stage-II images							

Zhang et al. 2016

StackGAN

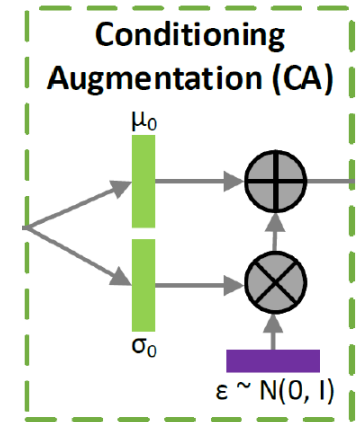
- Use stacked GAN structure









StackGAN

■ Model design

- Conditioning augmentation
 - Encoder with sampling step
- No random noise vector for Stage-2
- Conditioning both stages on text
- Spatial replication for the text conditional variable
- Negative samples for D
 - True images + non-corresponding texts
 - Synthetic images + corresponding texts



More StackGAN results

Text description	This flower is pink, white, and yellow in color, and has petals that are striped	This flower has a lot of small purple petals in a dome-like configuration	This flower is white and yellow in color, with petals that are wavy and smooth	This flower has petals that are dark pink with white edges and pink stamen
64x64 GAN-INT-CLS				
256x256 StackGAN				

Conditional image synthesis

■ Problem formulation

- Input: original image (low-res or partially observed)
- Output: target image (high-res or full image)
- Often formulated as a regression problem

$$\tilde{Y} = F_{net}(X; W)$$

$$\min_W E_X [L(Y, \tilde{Y})]$$

- However, conventional loss function usually leads to unsatisfactory results.

■ Solution: Adding adversarial loss terms

- Mapping to a distribution instead of a single image
- Structural loss (distribution-wise) instead of point-wise loss

Image-to-image translation

- One-to-many or many-to-one mapping [Isola et al., 2016]

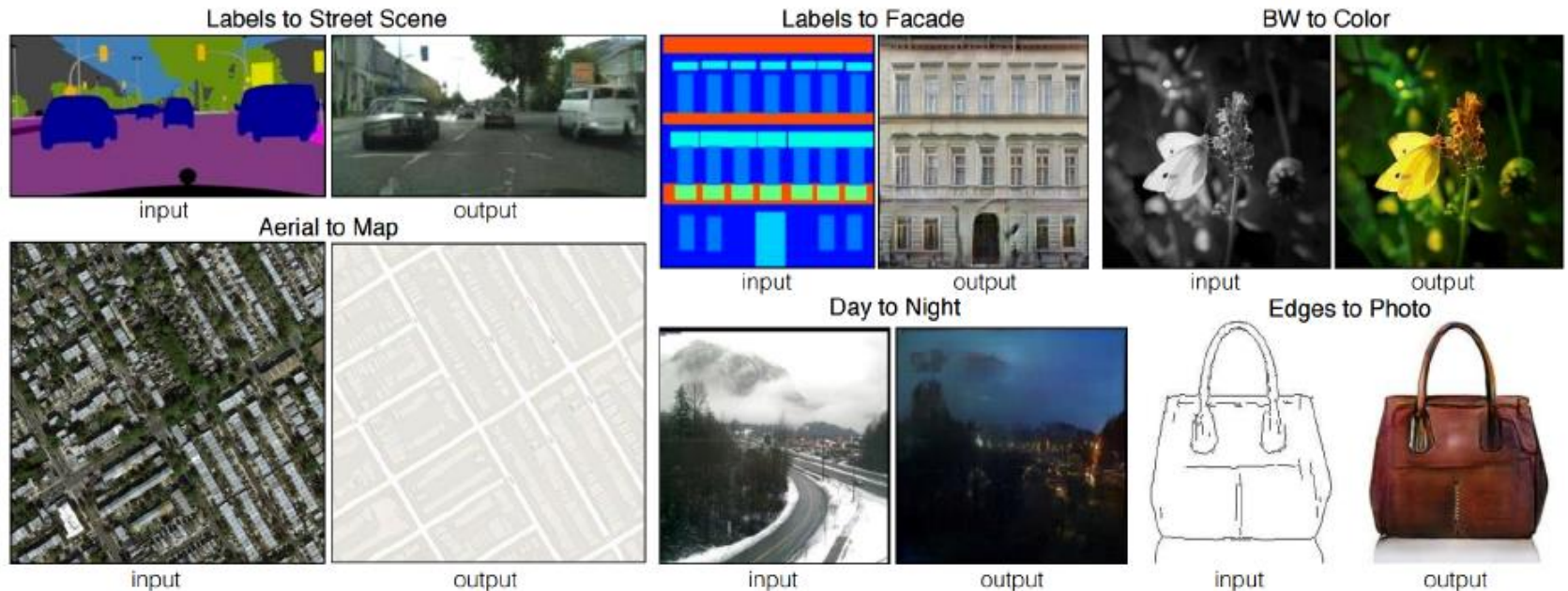


Image-to-image translation

- Combine the CGAN objective function with the L1 loss

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y \sim p_{data}(x,y), z \sim p_z(z)} [\|y - G(x, z)\|_1].$$

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

- Use the U-net structure for the generator

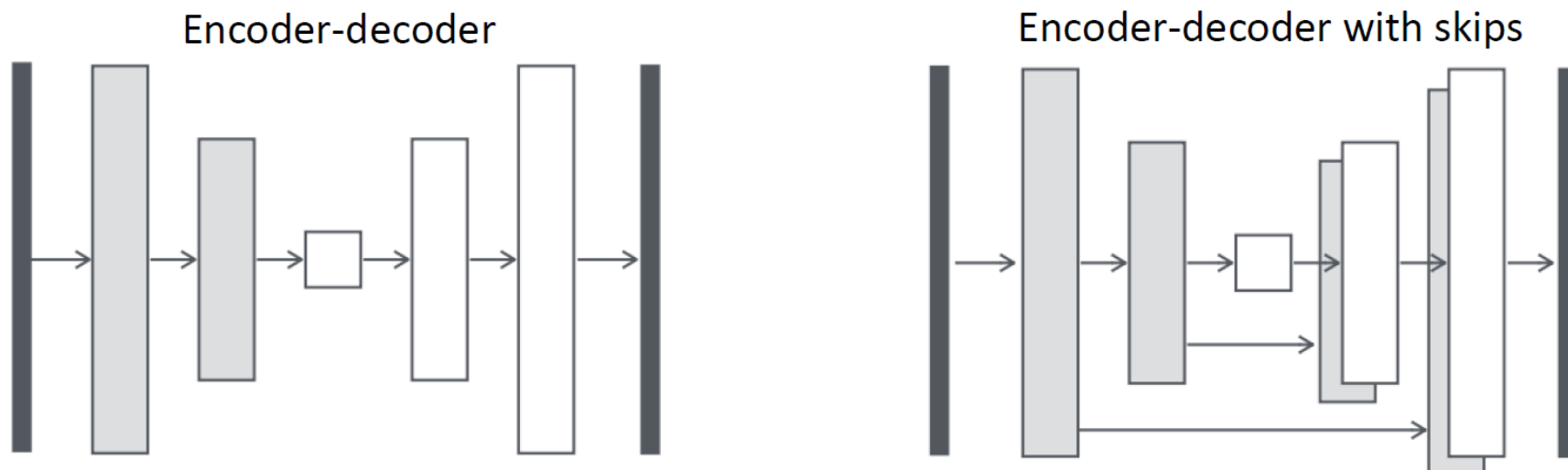


Image-to-image translation

- Patch-based discriminator
 - Separate each image into $N \times N$ patches
 - Train a patch-based discriminator

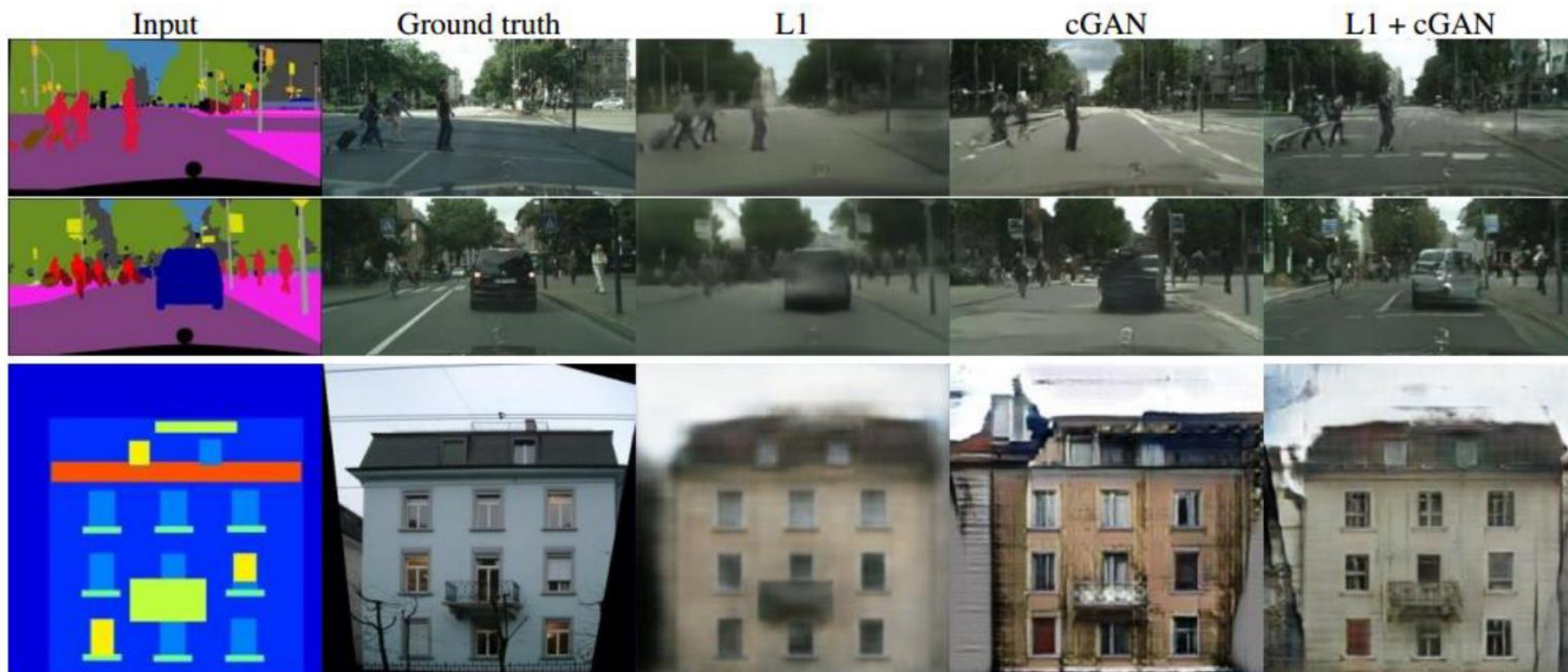


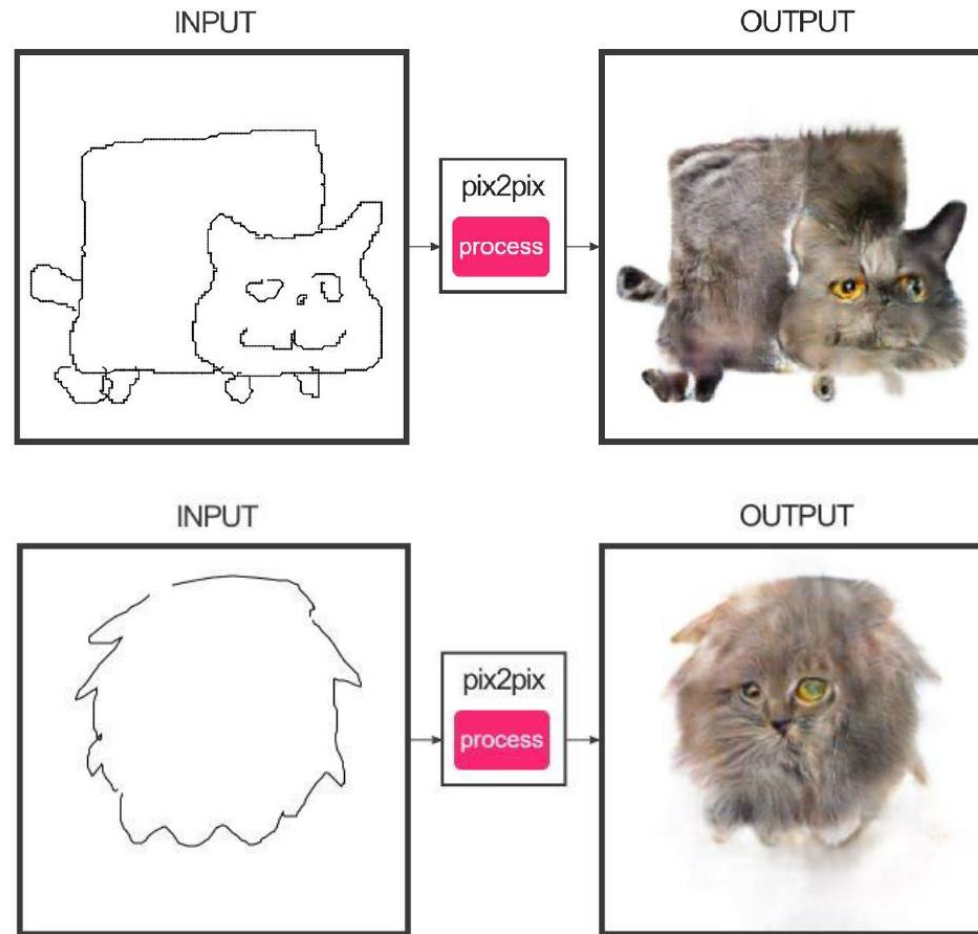
Image-to-image translation

■ More results



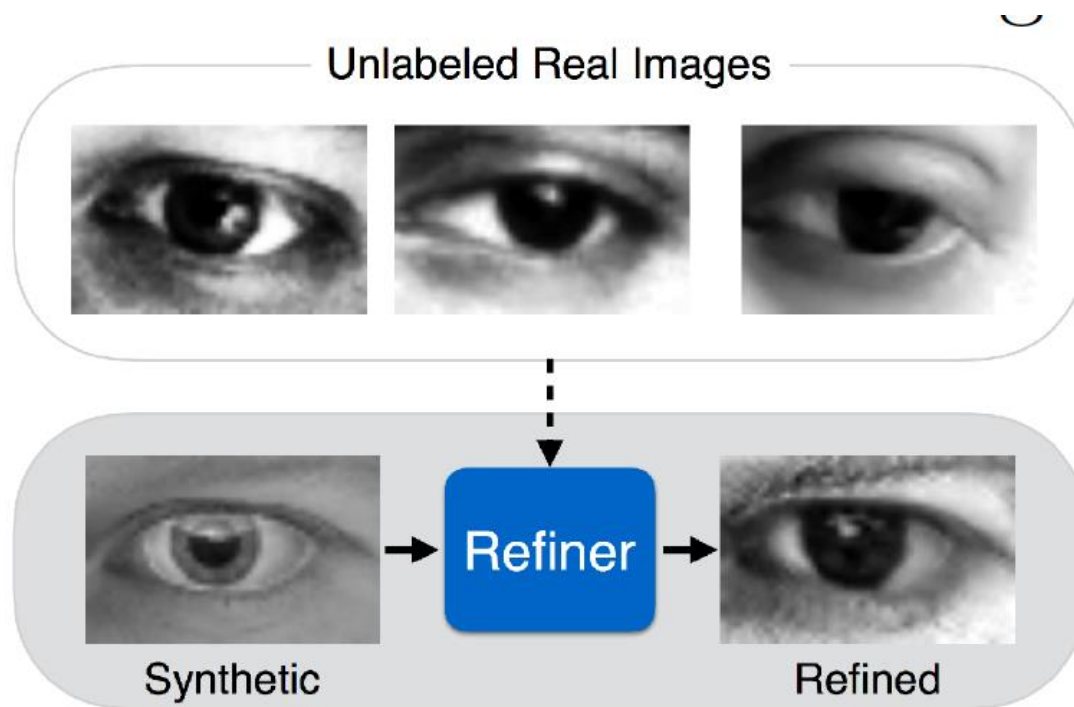
Image-to-image translation

- More results



Other im2im translation

- CGANs for simulated training data



(Shrivastava et al., 2016)

Sim-to-real synthesis

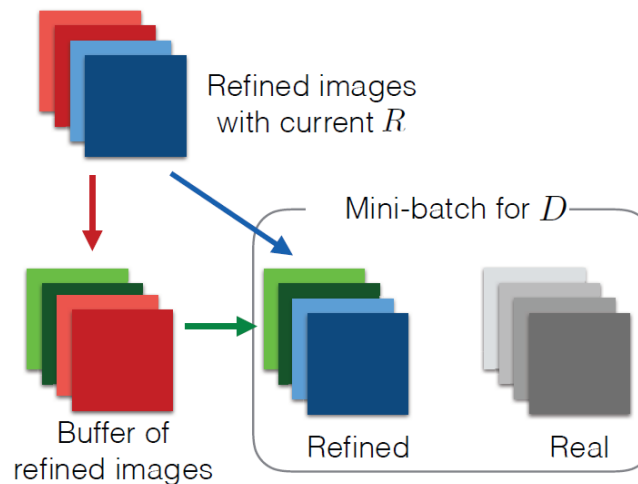
- Refiner network

$$\tilde{\mathbf{x}} = R_{\theta}(\mathbf{x})$$

- Learning objective

$$L_D(\phi) = - \sum_i \log(D_{\phi}(\tilde{\mathbf{x}}_i)) - \sum_j \log(1 - D_{\phi}(\mathbf{y}_j))$$

$$L_R(\theta) = - \sum_i \underbrace{\log(1 - D_{\phi}(R_{\theta}(\mathbf{x}_i)))}_{\text{Realistic style}} + \underbrace{\lambda \|\psi(R_{\theta}(\mathbf{x}_i)) - \psi(\mathbf{x}_i)\|_1}_{\text{Label information (content)}}$$



Outline

- Improving GAN training
 - WGANs
- Conditional GANs
 - Text-to-image: StackGANs
 - Image-to-image translation
- CycleGAN
 - Image-to-image translation with unpaired data

Acknowledgement: CMU, UofT, Stanford notes

CycleGAN

- Image-to-image translation without paired data

Monet ↔ Photos



Monet → photo

Zebras ↔ Horses



zebra → horse

Summer ↔ Winter



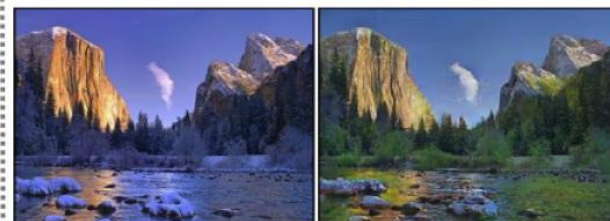
summer → winter



photo → Monet



horse → zebra



winter → summer



Photograph



Monet



Van Gogh



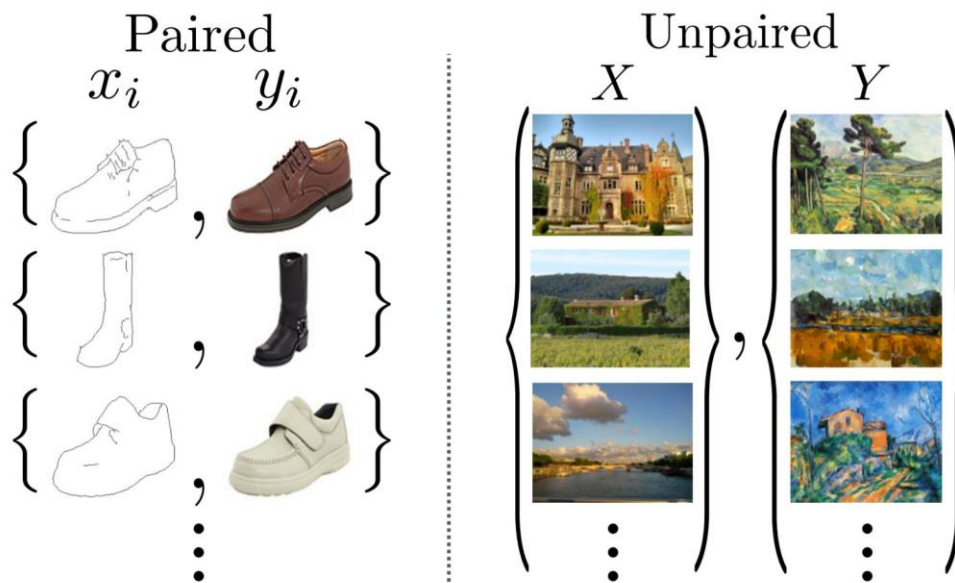
Cezanne



Ukiyo-e

CycleGAN

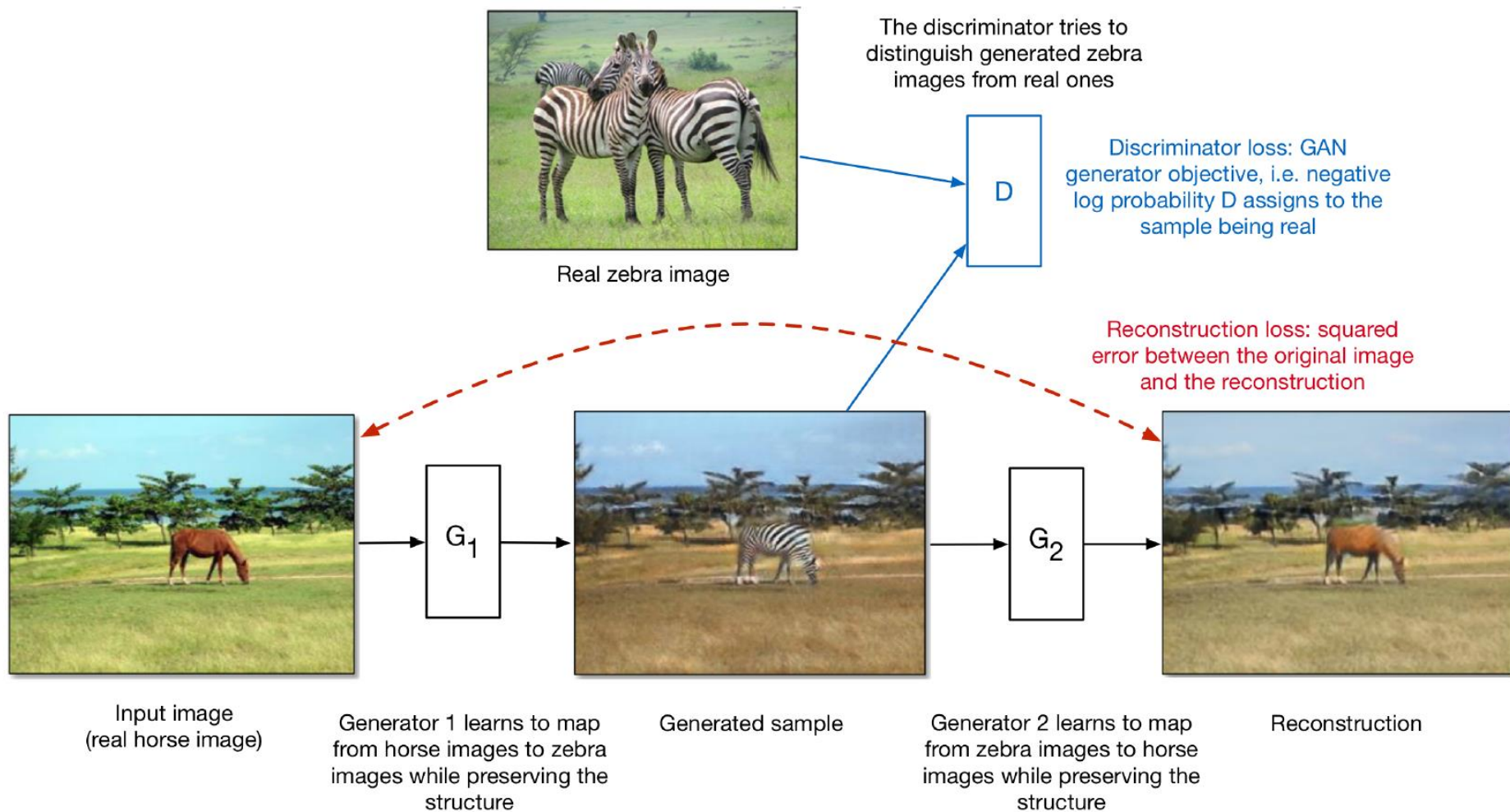
- If we had paired data (same content in both styles), this would be a supervised learning problem. But this is hard to find.



CycleGAN

- If we had paired data (same content in both styles), this would be a supervised learning problem. But this is hard to find.
- The CycleGAN architecture learns to do it from unpaired data.
 - Train two different generator nets to go from style 1 to style 2, and vice versa.
 - Make sure the generated samples of style 2 are indistinguishable from real images by a discriminator net.
 - Make sure the generators are **cycle-consistent**: mapping from style 1 to style 2 and back again should give you almost the original image.

CycleGAN

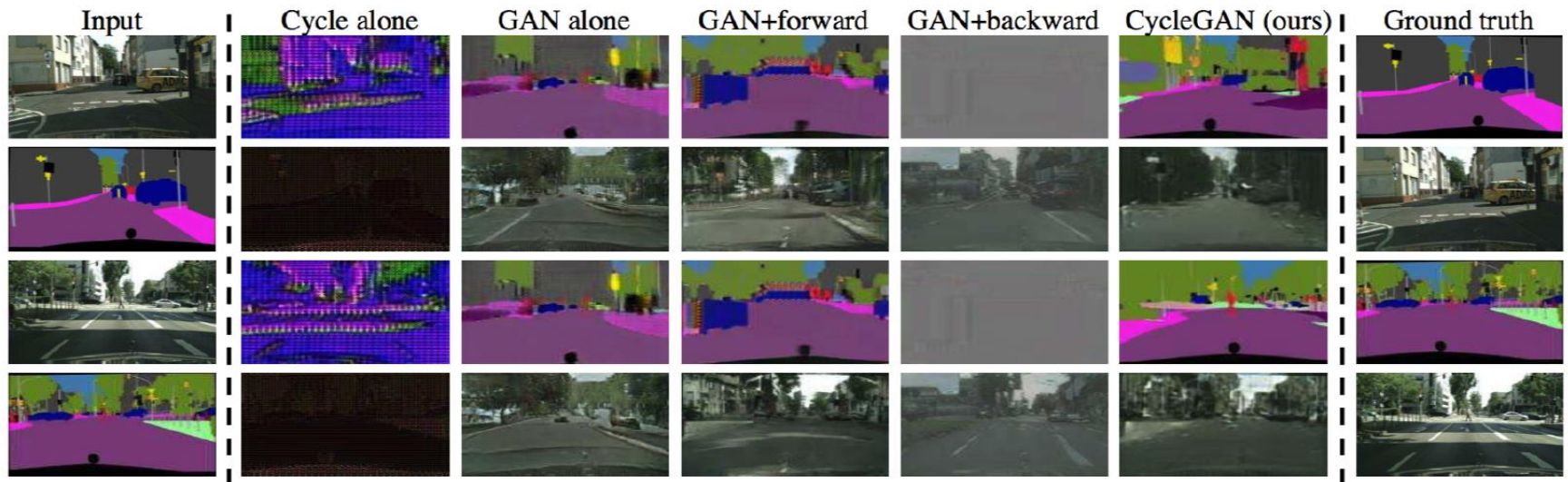


$$\text{Total loss} = \text{discriminator loss} + \text{reconstruction loss}$$

CycleGAN

■ Results

Style transfer between road scenes and semantic segmentations (labels of every pixel in an image by object category):



CycleGAN

- Results



- More details

<https://hardikbansal.github.io/CycleGANBlog/>

Summary

- Variants of GANs
 - Improving GANs
 - Conditional GANs: Conditional image synthesis
 - CycleGAN: Image-to-image translation with unpaired data
- Next time
 - Deep reinforcement learning