

Generative Adversarial Networks

Wang Chao, Group of DL

Talk of 2017Spring CV

2017-6-1

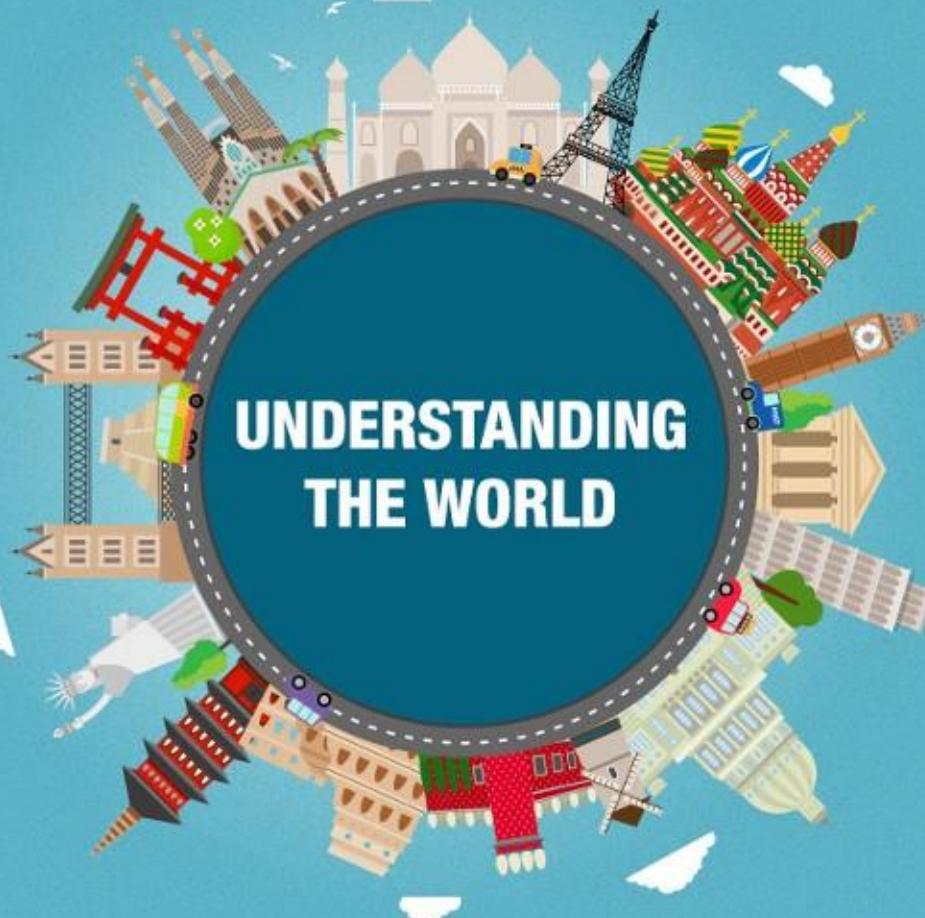
Vision@OUC

Overview

- Introduction
- Why learn GAN
- GANs in details
- Research Fronts
- Q&A

Introduction

Understand the world



Visual system

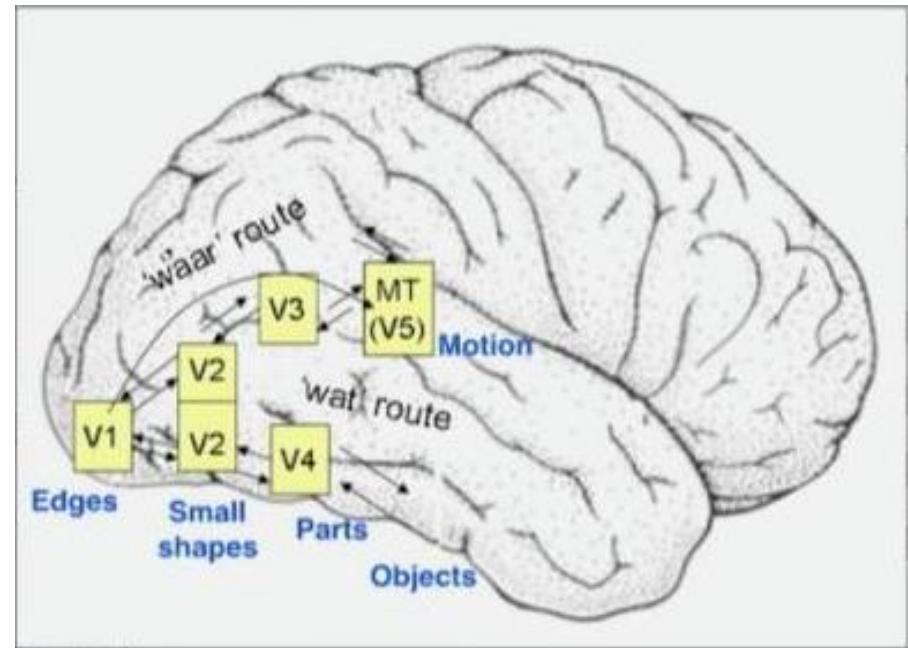
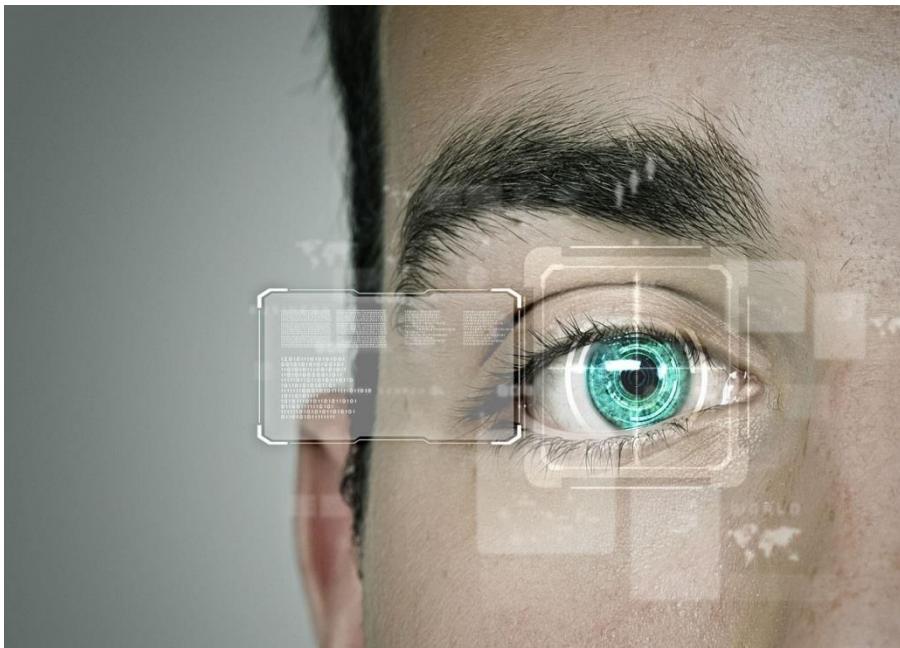
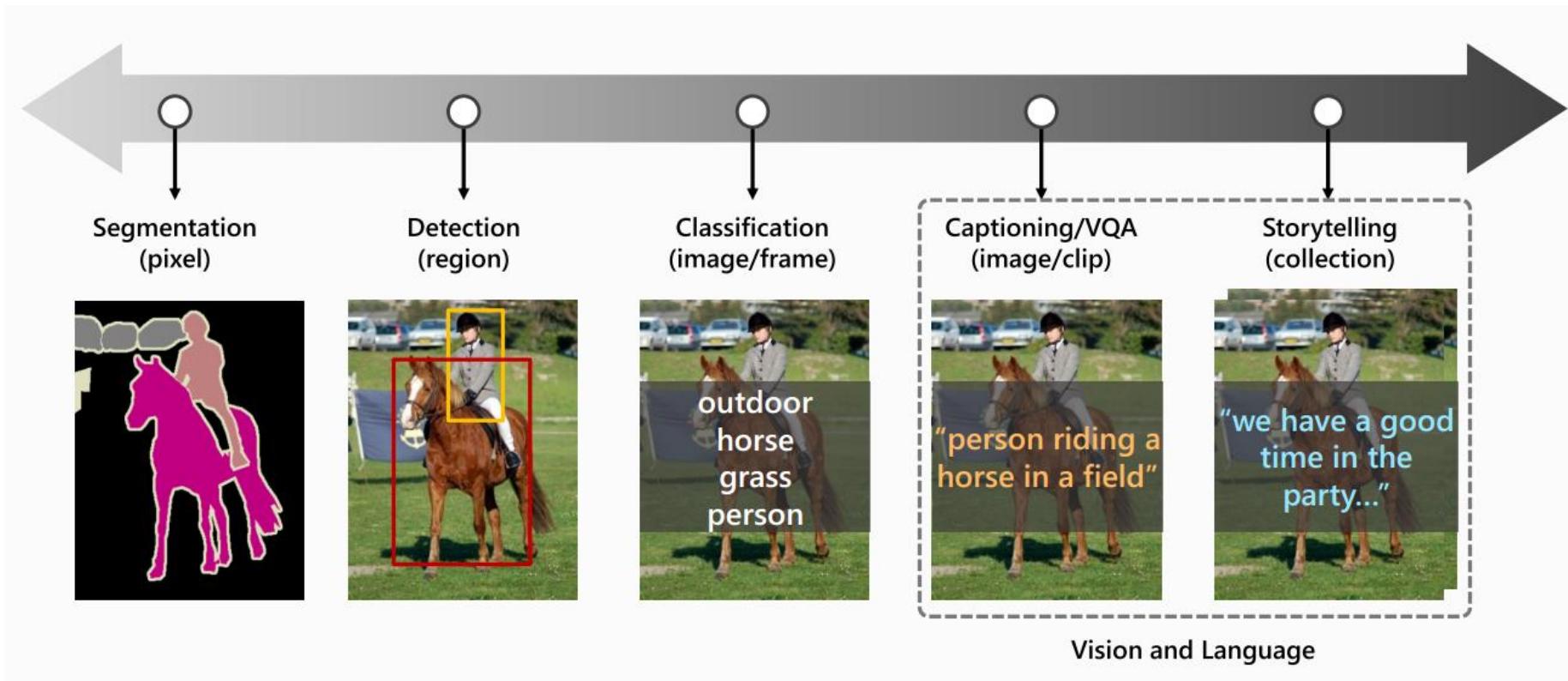


Image and video understanding

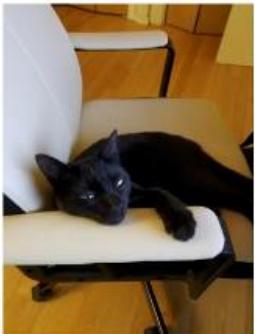


*“What I cannot create, I do
not understand.”*

--- *Richard Feynman*

Representation learning

Things...

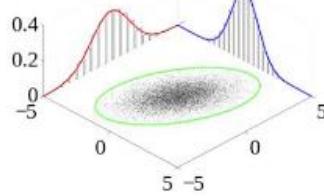
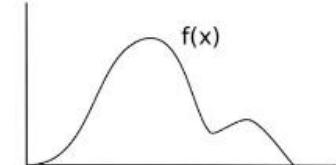
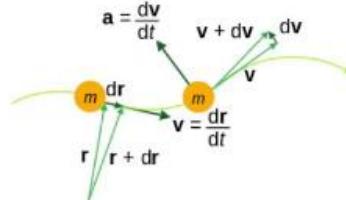


My heart beats as if the world is dropping,
you may not feel the love but i do its a heart
breaking moment of your life. enjoy the times
that we have, it might not sound good but
one thing it rhymes it might not be romantic
but i think it is great, the best rhyme i've ever
heard.



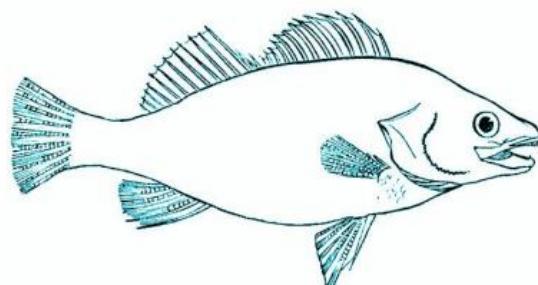
Representation

Engineering Knowledge...

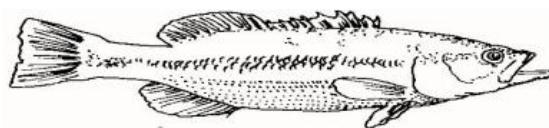


$\alpha^2 + \beta^2 = c^2, c = \sqrt{\alpha^2 + \beta^2}$
 $c^2 - a^2 = \beta^2, \alpha^2 - \beta^2 = a^2$
 $\frac{a}{c} = \frac{HB}{c} \text{ and } \frac{\beta}{c} = \frac{AH}{c}$
 $\beta d = \frac{\sin \alpha}{\cos \alpha}$
 $\alpha^2 = c \times HB \text{ and } \beta^2 = c \times AH$
 $\alpha^2 + \beta^2 = c^2 = c \times HB + c \times AH = c \times (HB + AH) = c^2$
 $\alpha^2 + \beta^2 = c^2, \sin \alpha = \frac{\beta}{c}; \cos \alpha = \frac{\alpha}{c}$
 $\operatorname{ctg} \alpha = \frac{\beta}{\alpha}; \operatorname{tg} \alpha = \frac{\alpha}{\beta}; \operatorname{ctg} \alpha = \frac{\cos \alpha}{\sin \alpha}$

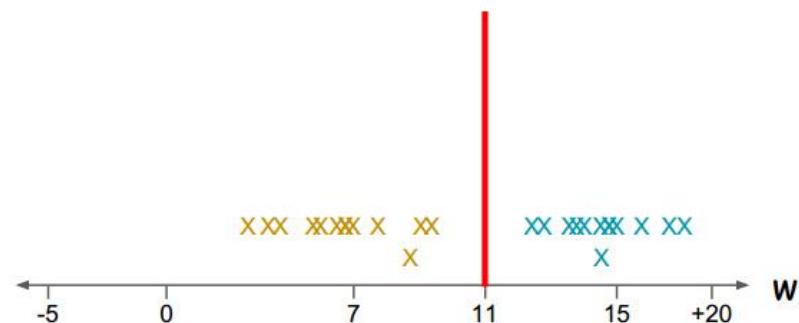
Representation Learning



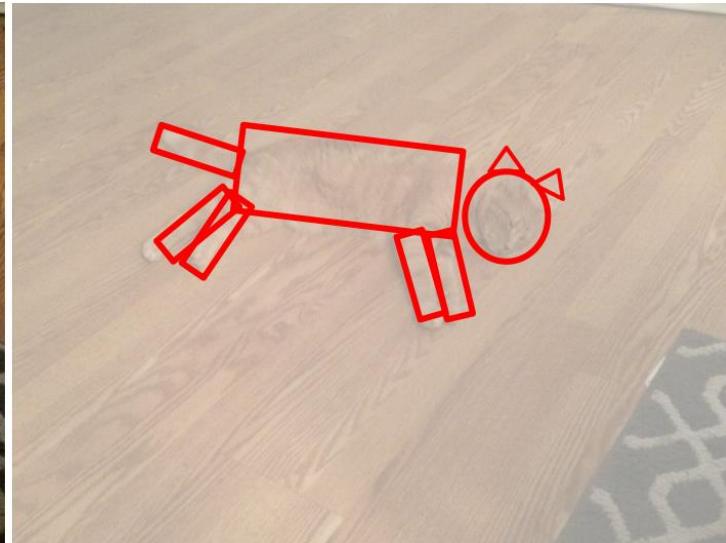
~12 lbs



~8 lbs



Representation Learning



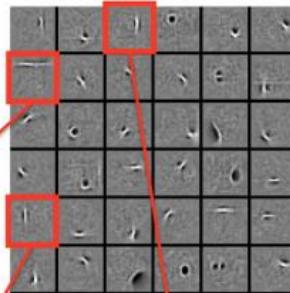
Represent the cat for a cat detector!

Representation Learning

Observed Data
Subset of 25,000 characters



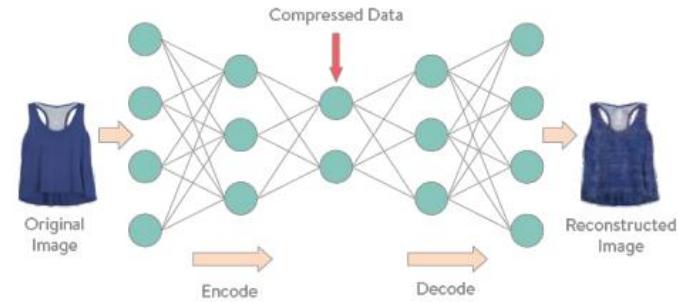
Subset of 1000 features



Sparse representations

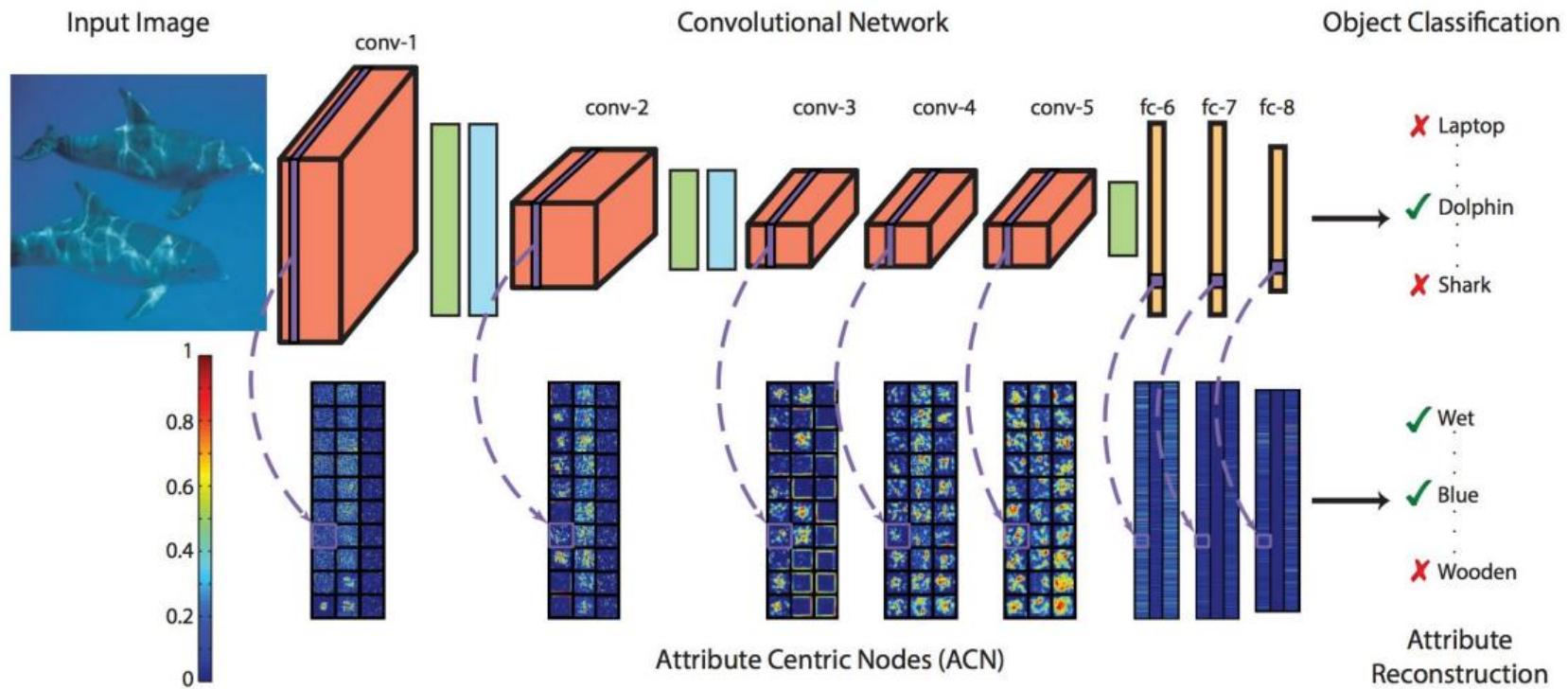
New Image:

$$\text{Digit} = 0.99 \times \text{Feature 1} + 0.97 \times \text{Feature 2} + 0.82 \times \text{Feature 3} + \dots$$



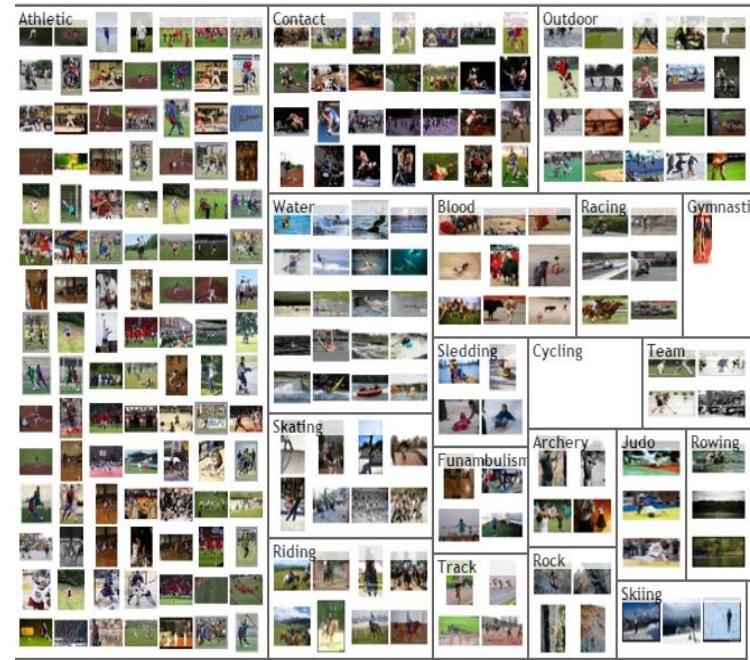
Unsupervised learning

Representation Learning



Supervised learning

GAN



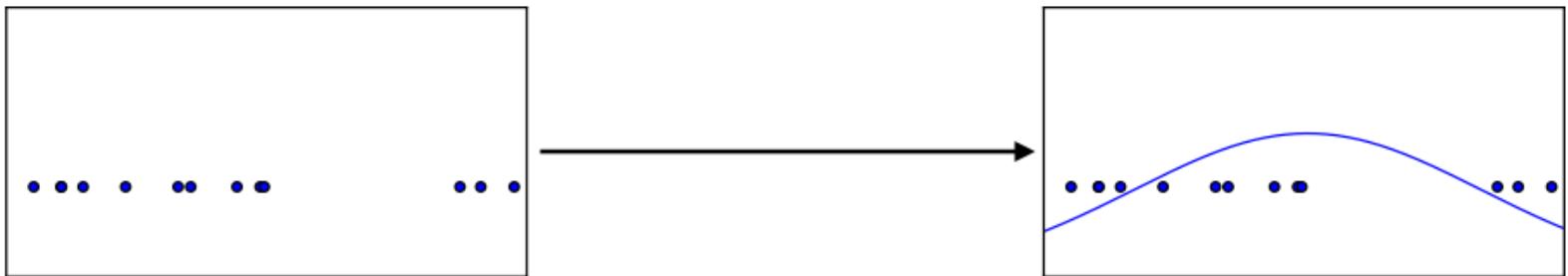
GAN model
Total 8 layers
Only 145MB

VS

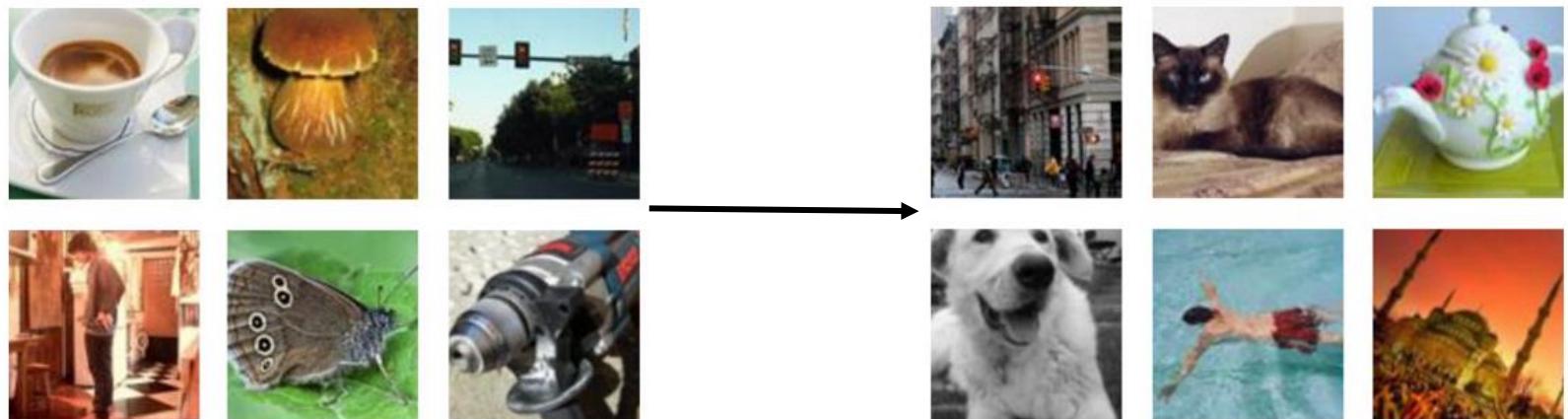
ImageNet
1.3 million images
200GB

Generative Modeling

- Probability **density** estimation



- Sample **generation**

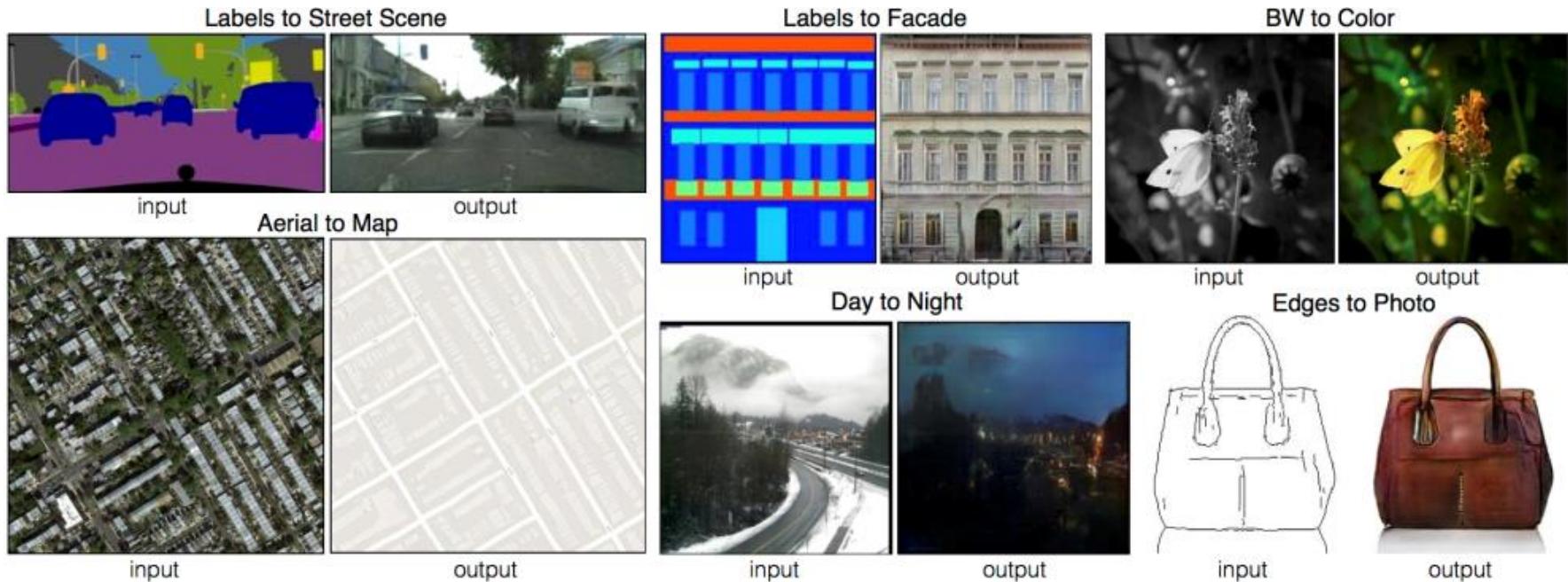


Why Generative Model?

- Test of our ability to use high-dimensional, complicated data
- RL
- Missing data
- Any situation need “generative model”

Applications

Image to image translation

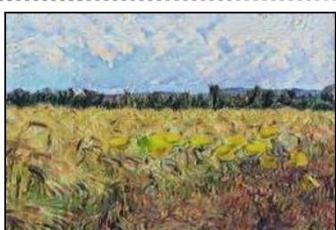


Monet \leftrightarrow PhotosMonet \rightarrow photoZebras \leftrightarrow Horseszebra \rightarrow horseSummer \leftrightarrow Wintersummer \rightarrow winterphoto \rightarrow Monethorse \rightarrow zebrawinter \rightarrow summer

Photograph



Monet



Van Gogh



Cezanne



Ukiyo-e

Text to image

**Text descriptions
(content)**

The bird has a **yellow breast** with grey features and a small beak.

This is a large **white** bird with **black wings** and a **red head**.

A small bird with a **black head and wings** and features grey wings.

This bird has a **white breast**, brown and white coloring on its head and wings, and a thin pointy beak.

A small bird with **white base** and **black stripes** throughout its belly, head, and feathers.

A small sized bird that has a cream belly and a short pointed bill.

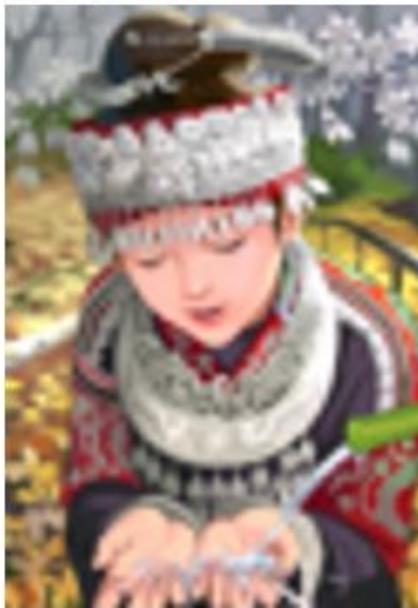
This bird is **completely red**.

**Images
(style)**

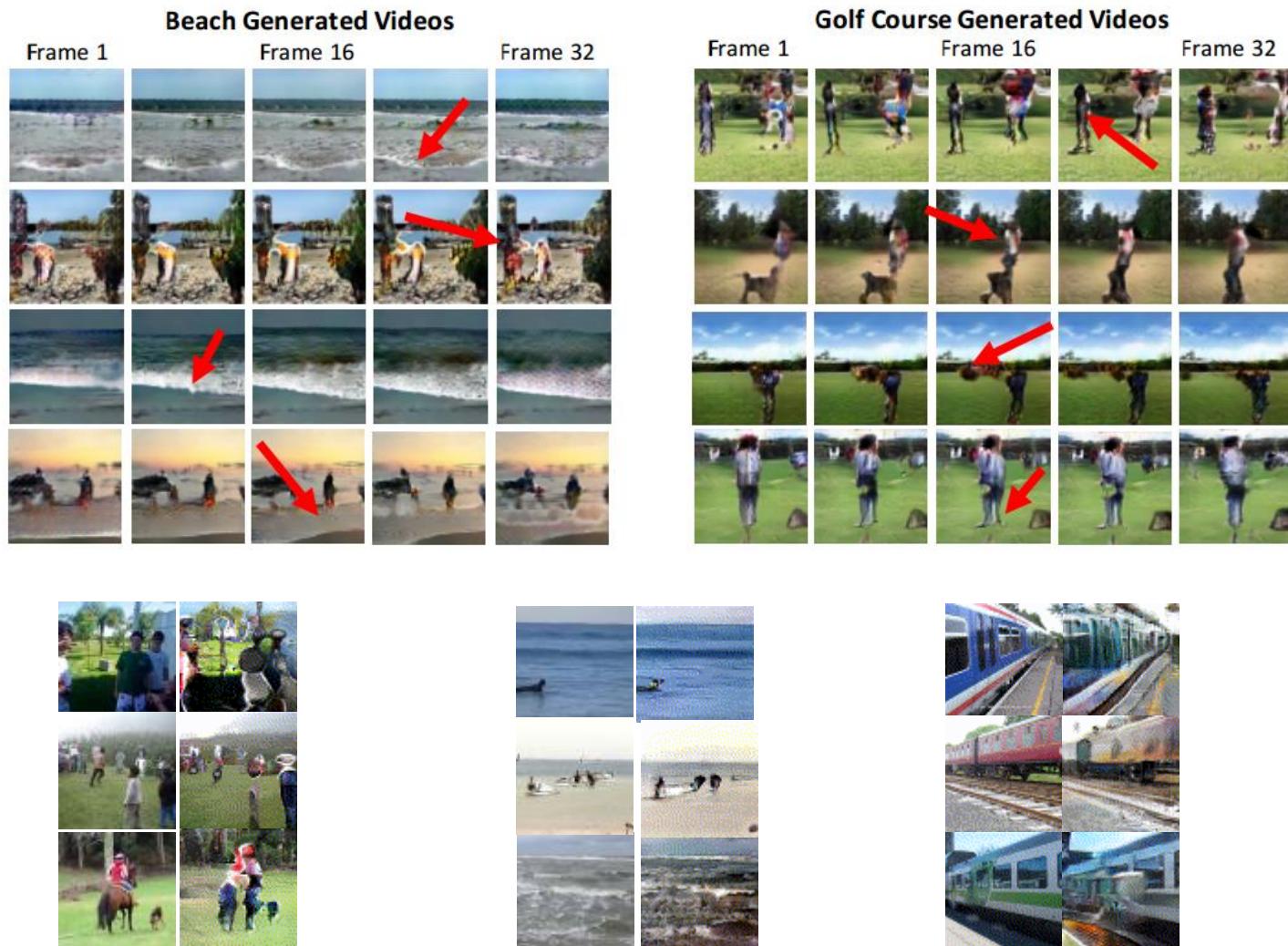


Super resolution

original

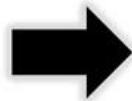
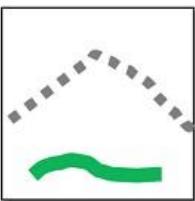
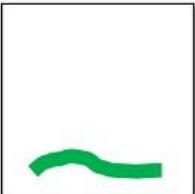
bicubic
(21.59dB/0.6423)SRResNet
(23.44dB/0.7777)SRGAN
(20.34dB/0.6562)

Video Prediction



iGAN

User edits



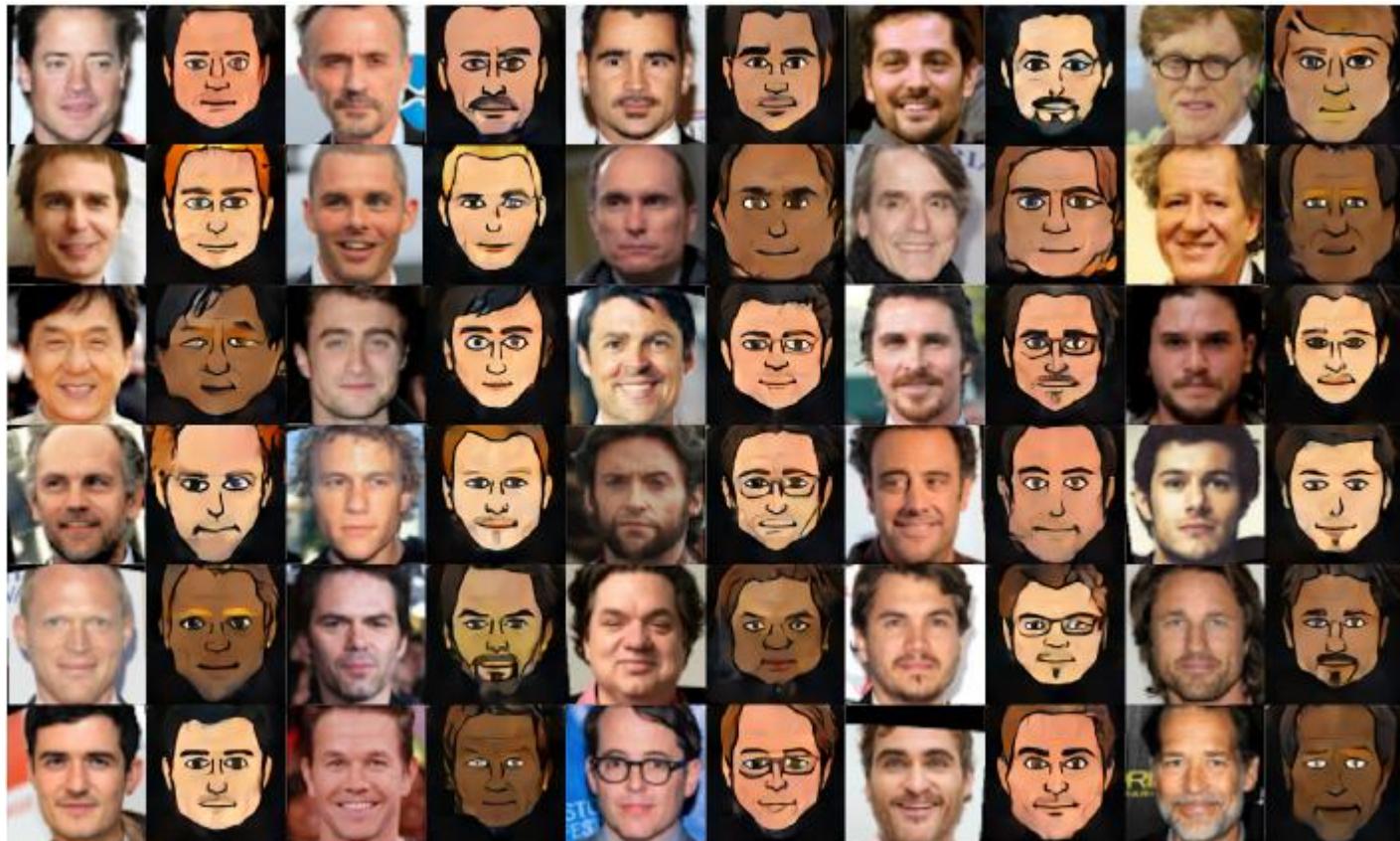
Generated images



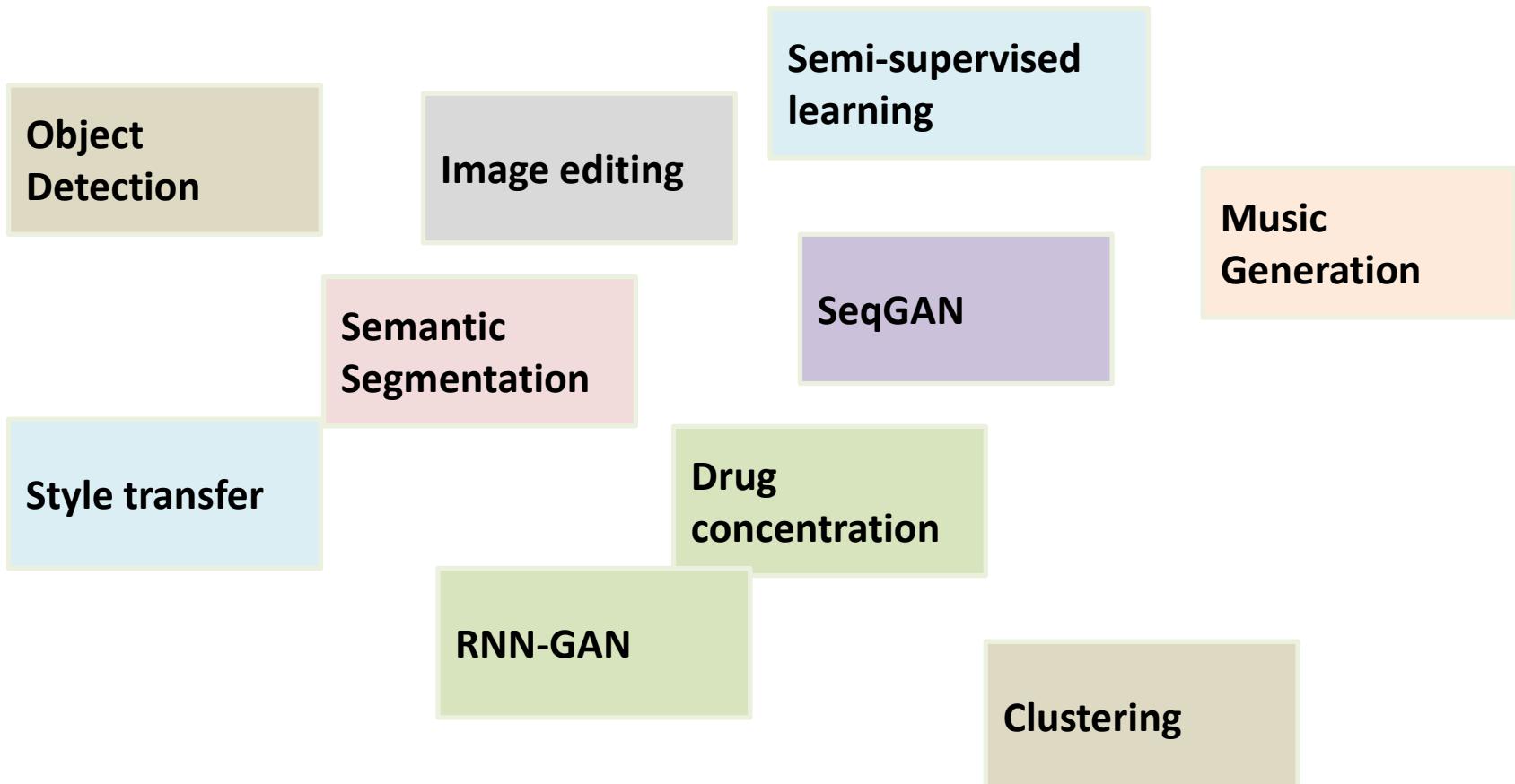
— Color

— Sketch

Unsupervised Cross-Domain Image Generation

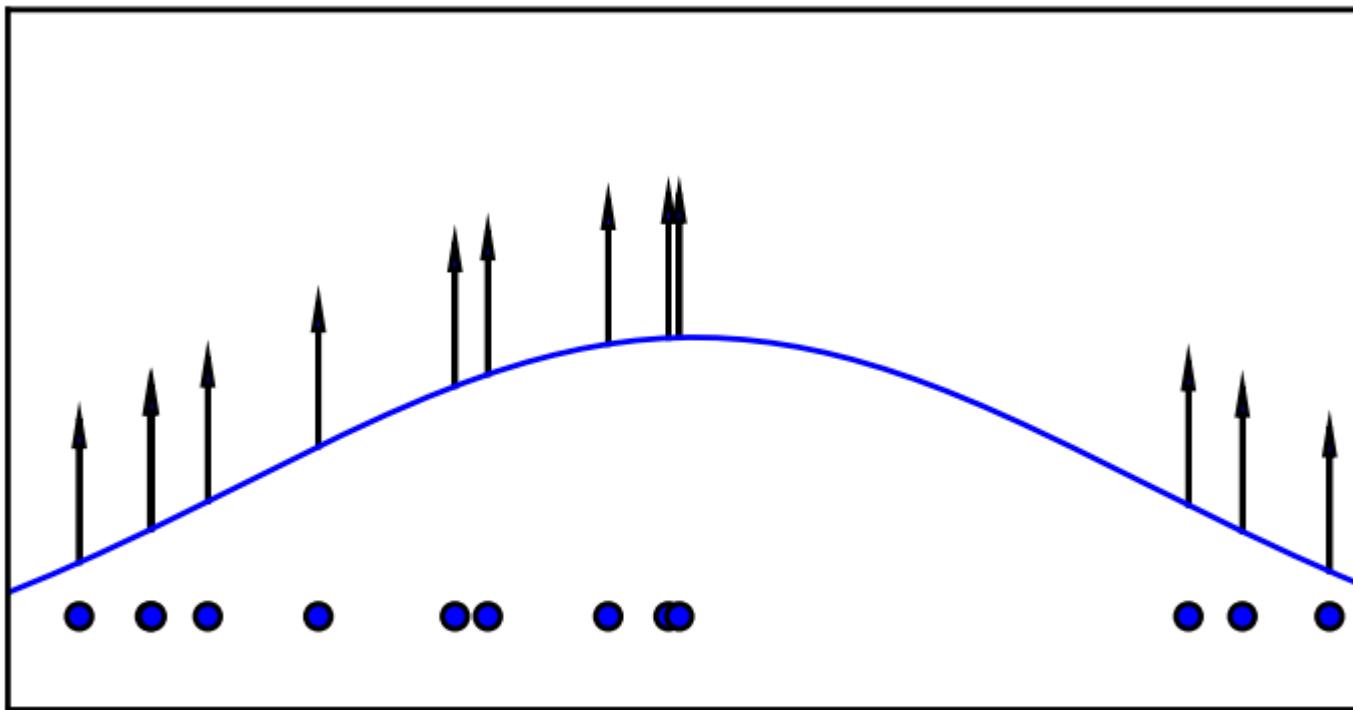


GAN application zoo



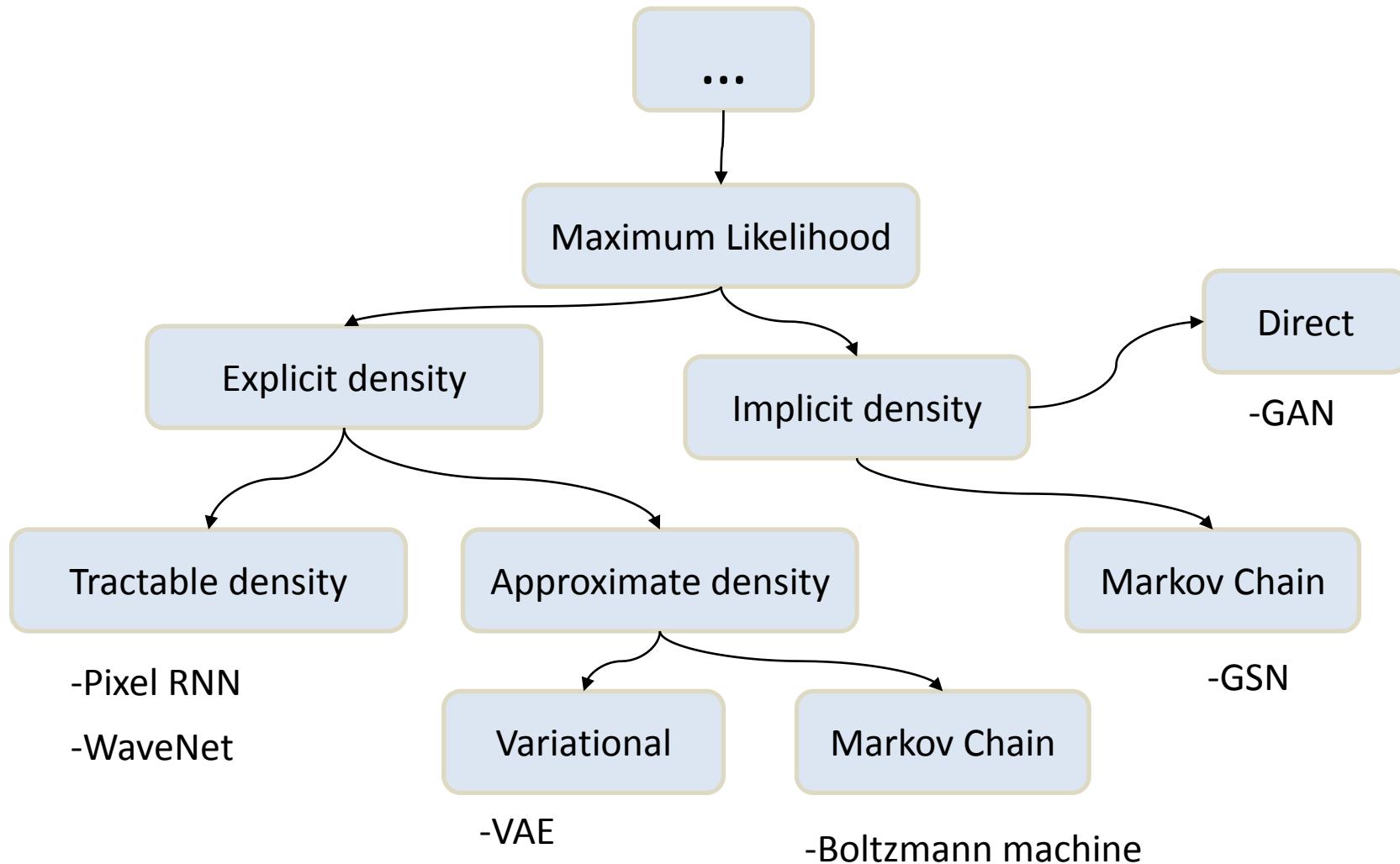
How Generative model works?

Maximum Likelihood



$$\theta^* = \arg \max_{\theta} \mathbb{E}_{x \sim p_{\text{data}}} \log p_{\text{model}}(\mathbf{x} \mid \theta)$$

Generative Models



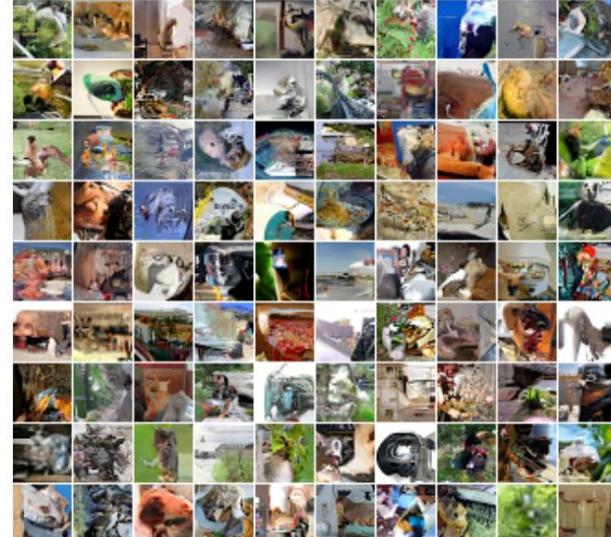
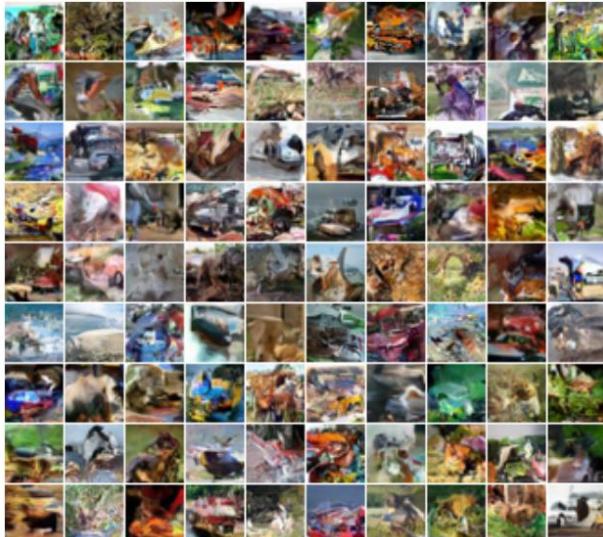
Pixel RNN

- Explicit formula based on chain rule:

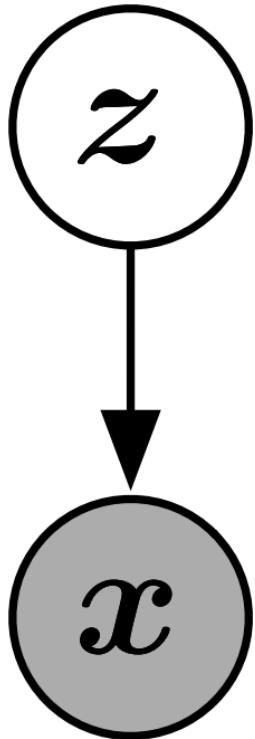
$$P_{model}(x) = P_{model}(x_1) \prod_{i=2}^n P_{model}(x_i|x_1, \dots, x_{i-1})$$

Disadvantages:

- $O(n)$ sample generation cost
- Generation not controlled by a latent code



Variational Autoencoder(VAE)



- $\log p(x) \geq \log p(x) - D_{KL}(q(z)||p(z|x))$
= $\mathbb{E}_{z \sim q} \log p(x, z) + H(q)$
- Disadvantages:
 - Not asymptotically consistent unless q is perfect
 - Samples tend to have lower quality



CIFAR10 samples

Adversarial training

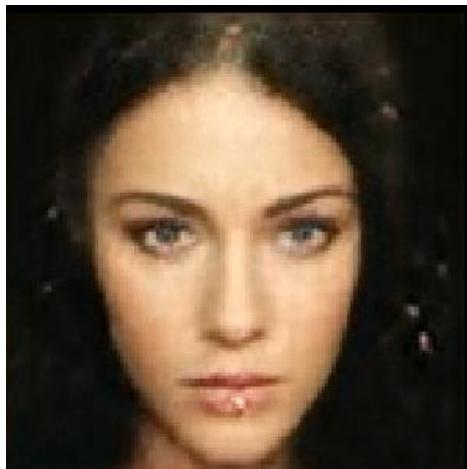
- A phrase whose usage is in flux; a new term that applies to both new and old ideas.
- **Examples:**
- An agent playing against a copy of itself in a board game
- Robust optimization / robust control
- Training neural networks on adversarial examples

GANs in details

- Architecture
- Object function
- Theory(Minimax Game)
- Problems

GAN

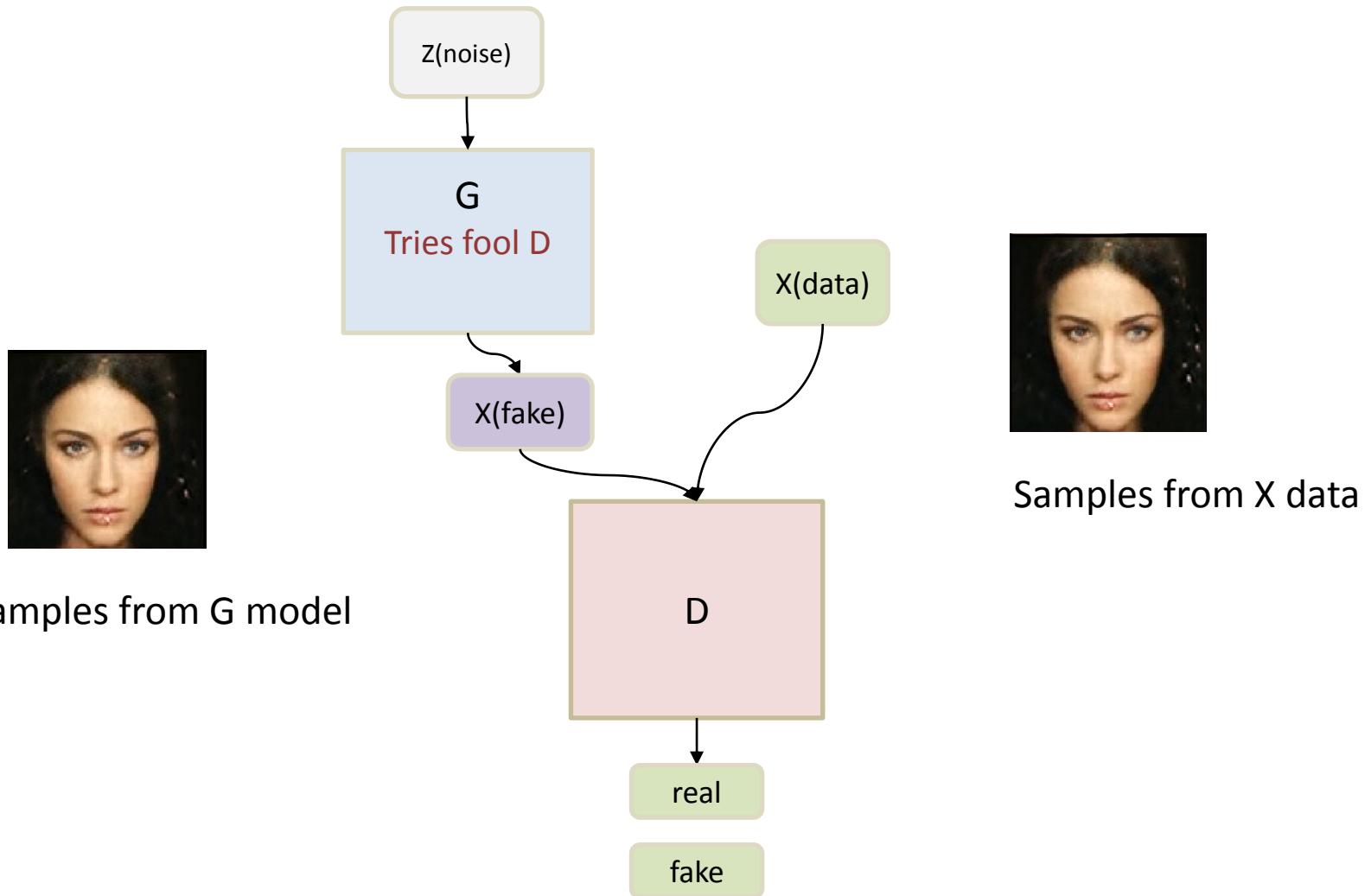
[0.1,0.5,1.2,1.1,0.6..]



Games

- A **counterfeiter-police game** between two components: a generator **G** and a discriminator **D**
- **G**: counterfeiter -> trying to fool police with fake currency
- **D**: policy, trying to detect the counterfeit currency

GAN Framework



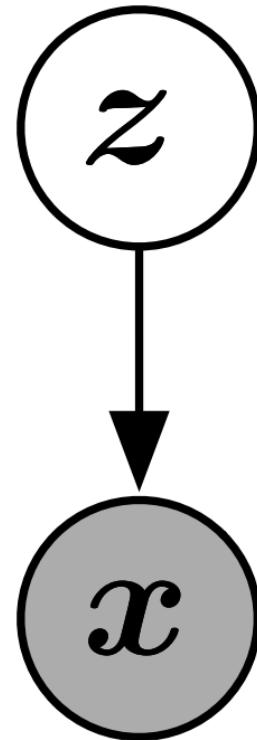
Object Function

$$\min_G \max_D V(D, G) = E_{x \sim p_{data}(x)}[\log D(x)] + E_{z \sim p_z(z)}[\log(1 - D(G(z)))]$$

- $D(x)$: predict that real data's label is “1”
- $D(G(z))$: predict that fake data's label is “0”

Generator Network

- Use a latent code
- Unlike variational method
- No Markov chains needed
- Regards as producing best results



Training algrothim

- Algorithm: minibatch SGD training.

- For iterations in training:

- For **K steps** do:

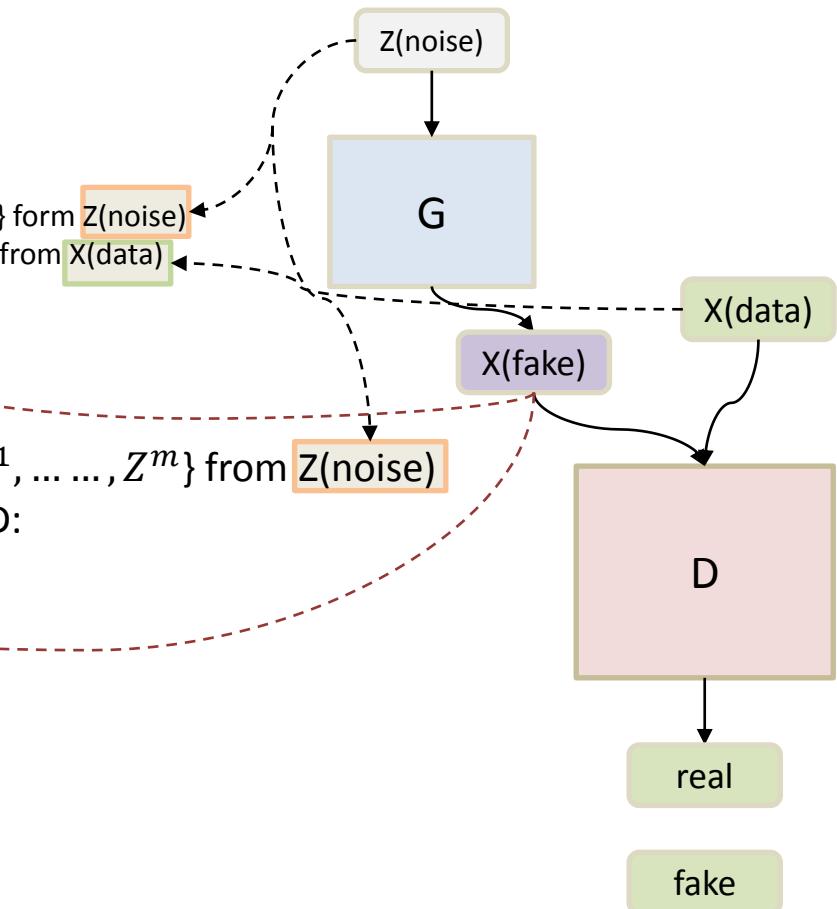
- Samples minibatch of **m** noise from $\{Z^1, \dots, Z^m\}$ form $Z(\text{noise})$
 - Samples minibatch of **m** examples $\{X^1, \dots, X^m\}$ from $X(\text{data})$
 - Update the **discriminator** by ascending its SD:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^m [\log D(x^i) + \log(1 - D(1 - D(G(z^i)))]$$

- End for
 - Sample minibatch of **m** noise samples $\{Z^1, \dots, Z^m\}$ from $Z(\text{noise})$
 - Update the **generator** by descending its SD:

$$\cdot \nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^m (1 - D(G(z^i)))$$

- End for



Minimax Game

- $J^d = -\frac{1}{2} \mathbb{E}_{x \sim P_{data}(x)} [\log D(x)] - \frac{1}{2} \mathbb{E}_z \log(1 - D(G(z)))$
- $J^g = -J^d$
- **Zero-sum game**
- **D's goal:** maximize $V(D, G)$
- **G's goal:** minimize $V(D, G)$

Discriminator

- *Global Optimality:* $P_{\text{data}} = P_g$
- *Fix G:* $D(x) = \frac{P_{\text{data}}(x)}{P_{\text{data}}(x) + P_{\text{model}}(x)}$
- *Maximize:* $V(G, D)$

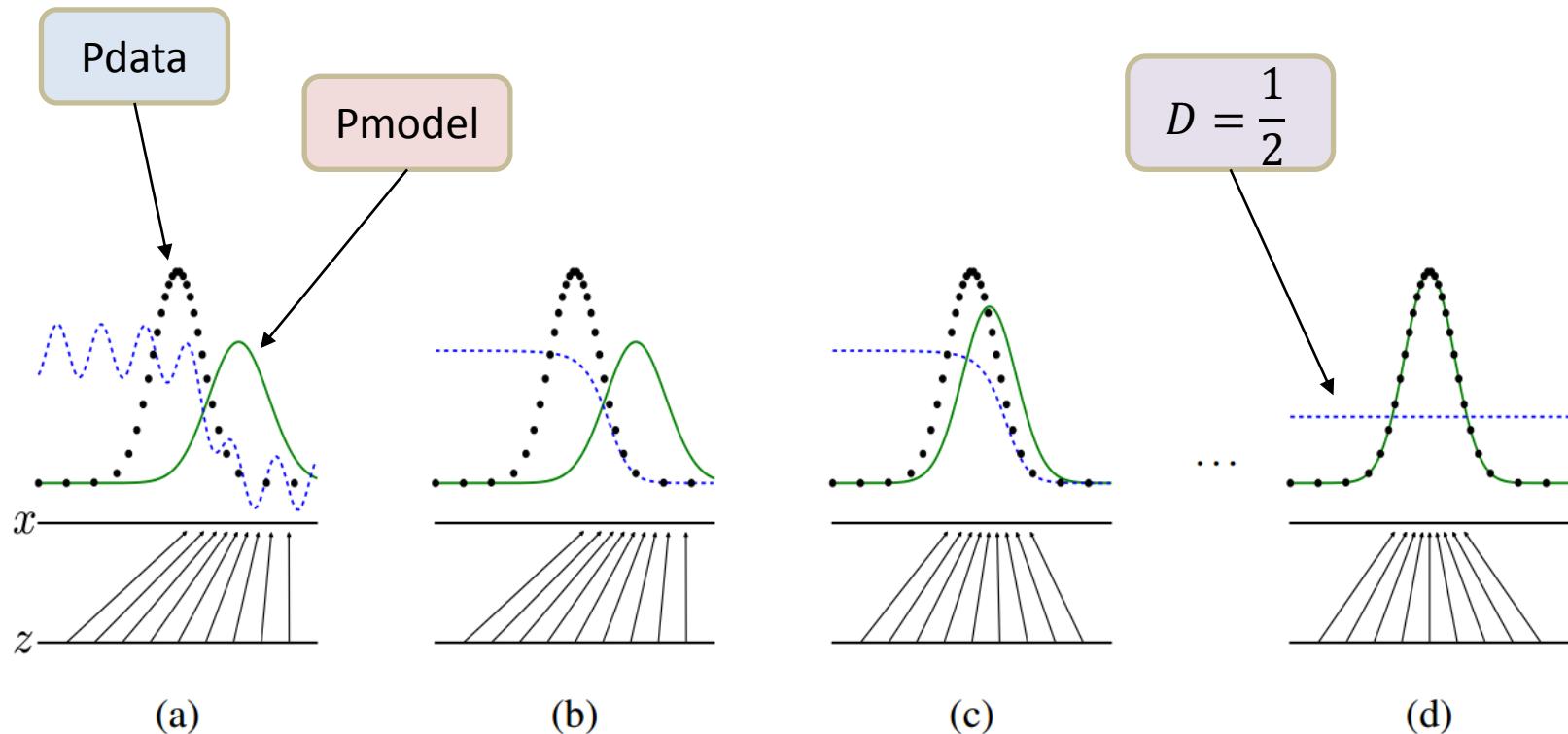
$$\begin{aligned} V(G, D) &= \int_{\mathbf{x}} p_{\text{data}}(\mathbf{x}) \log(D(\mathbf{x})) d\mathbf{x} + \int_z p_{\mathbf{z}}(\mathbf{z}) \log(1 - D(g(\mathbf{z}))) dz \\ &= \int_{\mathbf{x}} p_{\text{data}}(\mathbf{x}) \log(D(\mathbf{x})) + p_g(\mathbf{x}) \log(1 - D(\mathbf{x})) d\mathbf{x} \end{aligned}$$

$$y \rightarrow a \log(y) + b \log(1 - y)$$



$$y = \frac{a}{a + b}$$

Discriminator Strategy



Non-Saturating Game

- $J^d = -\frac{1}{2} \mathbb{E}_{x \sim P_{data}(x)} [\log D(x)] - \frac{1}{2} \mathbb{E}_z \log(1 - D(G(z)))$
- $J^g = -\frac{1}{2} \mathbb{E}_z \log D(G(z))$
- No longer a single loss
- G model maximizes the log-probability of D being mistaken
- Heuristically, G can still learn even though D rejects all G samples

Examples



a)



b)



c)



d)

“There’s no free lunch.”

original



generated



(李嫣然，深入浅出GAN原理篇.)

Problems

- No convergence
- Mode collapse
- Mode missing
- Unstabilitly

Non convergence

- Optimization algorithms often approach a saddle point or local minimum rather than a global minimum
- Game solving algorithms may not approach an equilibrium at all

Objective Function

- D function:

$$\log(D, g_\theta) = E_{x \sim p_{data}(x)}[\log D(x)] + E_{z \sim p_z(z)}[\log(1 - D(G(z)))]$$

- G function:

- Original: $E_{z \sim p_z(z)}[\log(1 - D(G(z)))]$

- Alternative: $E_{z \sim p_z(z)}[-\log D(g_\theta(z))]$

- Zero-sum Game -> non-saturating game

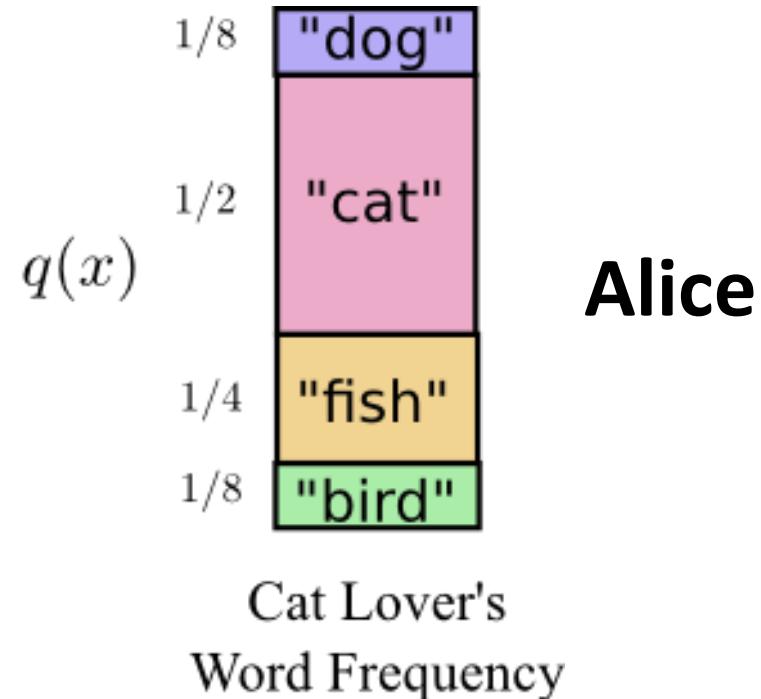
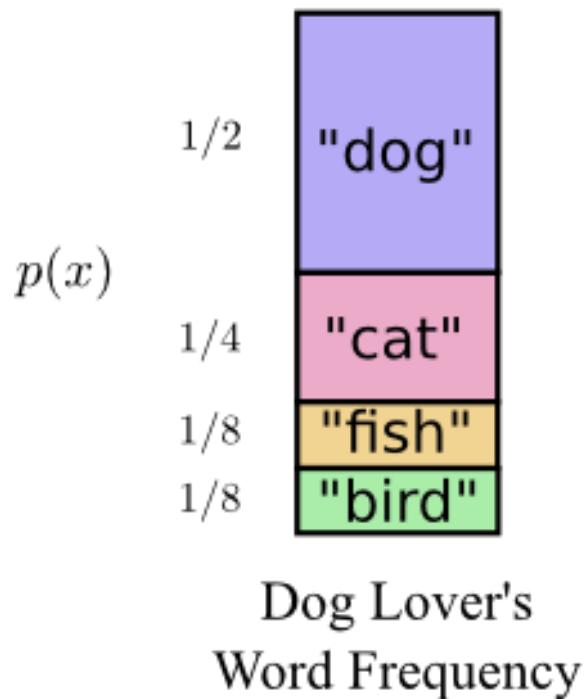


Non convergence

- G original: $E_{x \sim p_{data}(x)}[\log D(x)] + E_{z \sim p_z(z)}[\log(1 - D(G(z)))]$
- D optimal: $D^*(x) = \frac{P_r(x)}{P_r(x) + P_g(x)}$
- G become: $E_{x \sim p_{data}(x)}[\log \frac{P_r(x)}{\frac{1}{2}[P_r(x) + P_g(x)]}] + E_{z \sim p_z(z)}[\log \frac{P_g(x)}{\frac{1}{2}[P_r(x) + P_g(x)]}] - 2 \log 2$

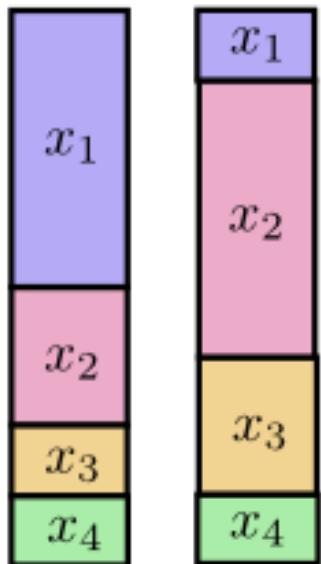
$$L(D^*, g_\theta) = 2JSD[P_r \parallel P_g] - 2 \log 2$$

Understanding KL&JS

Bob

Understanding KL&JS

$$p(x) \quad q(x)$$

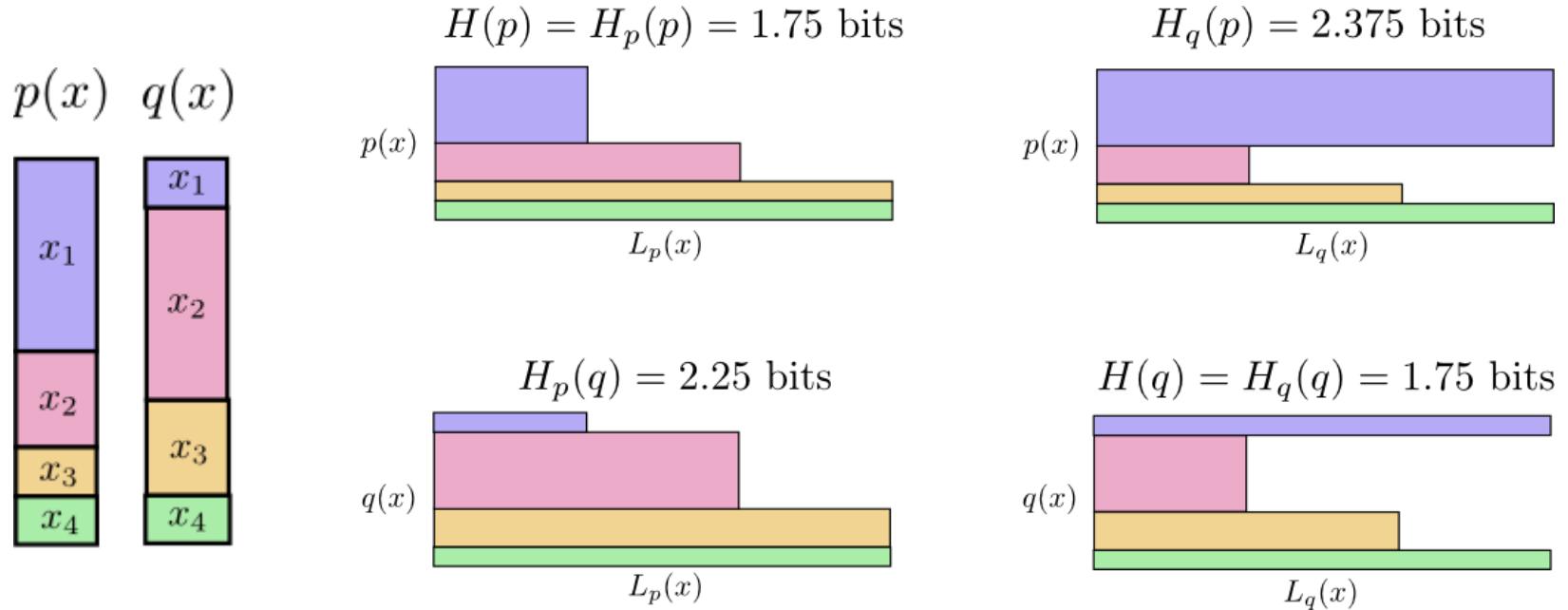


Cross-Entropy: $H_p(q)$

Average Length
of message from $q(x)$
using code for $p(x)$.

$$H_p(q) = \sum_x q(x) \log_2 \left(\frac{1}{p(x)} \right)$$

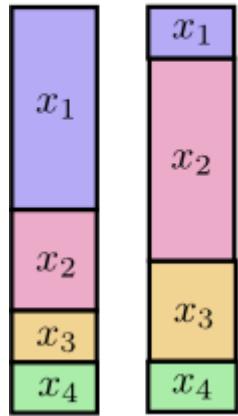
Understanding KL&JS



$$H_p(q) \neq H_q(p)$$

Understanding KL&JS

$p(x)$ $q(x)$



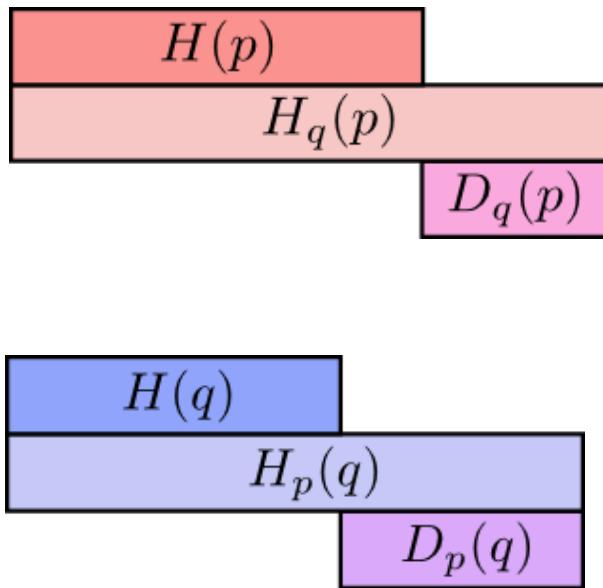
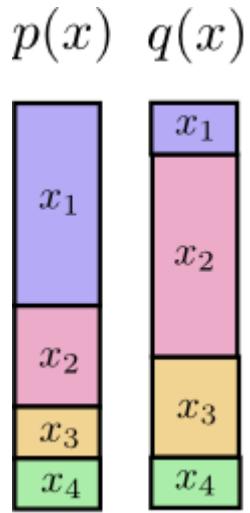
JS Divergence:

$$D_{JS}(p \mid q) = D_{JS}(q \mid p) = \frac{1}{2} D_{KL}(p \mid r) + \frac{1}{2} D_{KL}(q \mid r)$$

$$r = \frac{1}{2}(p + q)$$

KL Divergence: $D_q(p) = H_q(p) - H(p)$

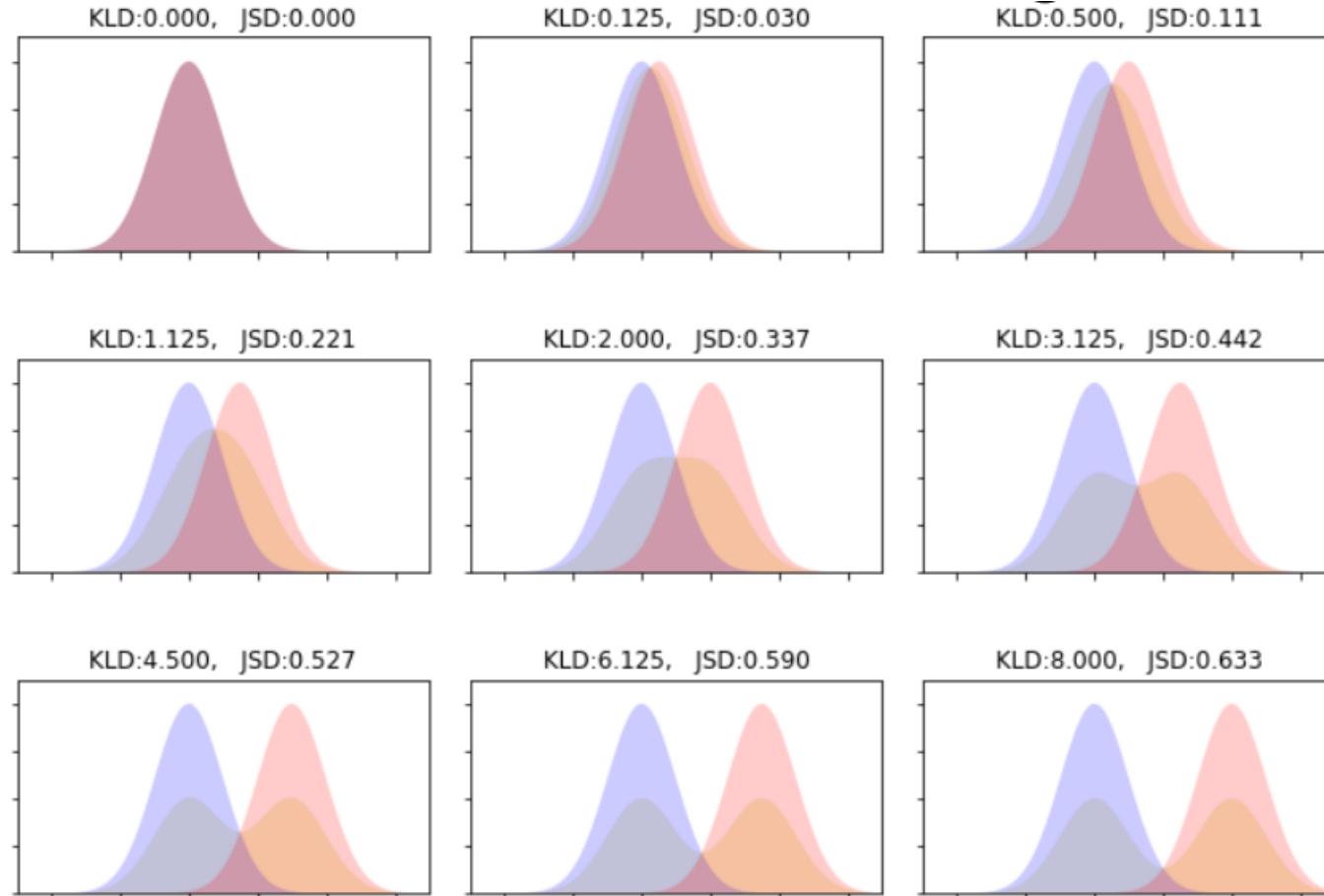
Understanding KL&JS



$$H_p(q) \neq H_q(p)$$

KL Divergence: $D_q(p) = H_q(p) - H(p)$

Understanding KL&JS



Non convergence(original G function)

$$L(D^*, g_\theta) = 2JSD[P_r \parallel P_g] - 2\log 2$$



$$P_r(x) = 0 \text{ 且 } P_g(x) = 0$$

$$P_r(x) \neq 0 \text{ 且 } P_g(x) \neq 0$$

$$P_r(x) = 0 \text{ 且 } P_g(x) \neq 0$$

$$P_r(x) \neq 0 \text{ 且 } P_g(x) = 0$$



$$2\log 2$$

Unstable & Mode missing

- G alternative: $E_{z \sim p_z(z)}[-\log D(G(z))]$
- D optimal: $D^*(x) = \frac{P_r(x)}{P_r(x) + P_g(x)}$
- G become: $KL(P_g \parallel P_r) - 2JS(P_r \parallel P_g)$

MIN

MAX

Unstable problem

- Minimizing the KL divergence while
Maximizing the JS divergence is **crazy**:

$$\text{KL}(P_g \parallel P_r) - 2\text{JS}(P_r \parallel P_g)$$

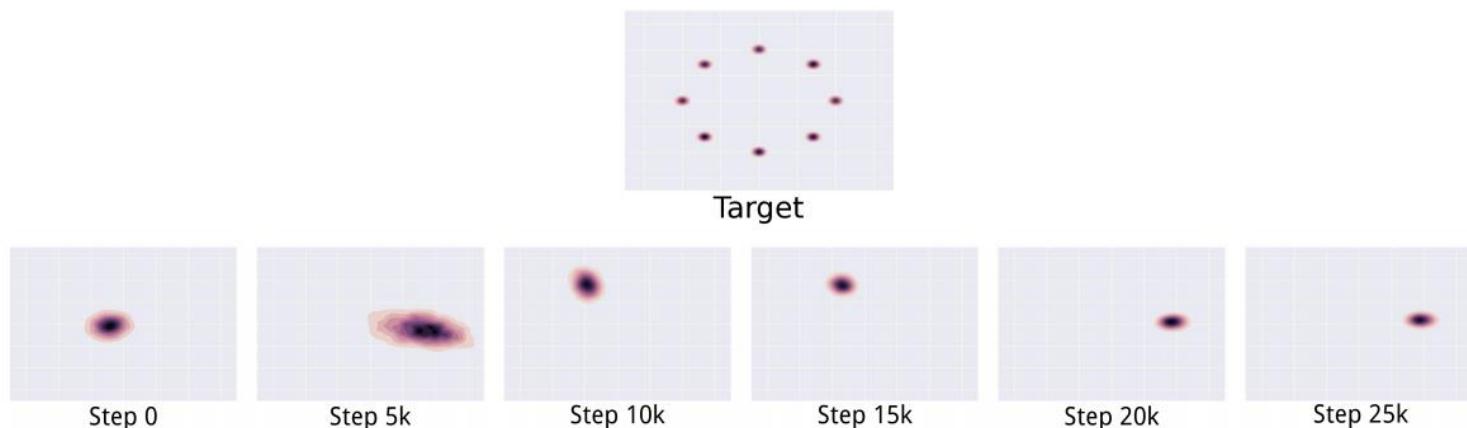
- Result in gradient unstable

Mode collapse

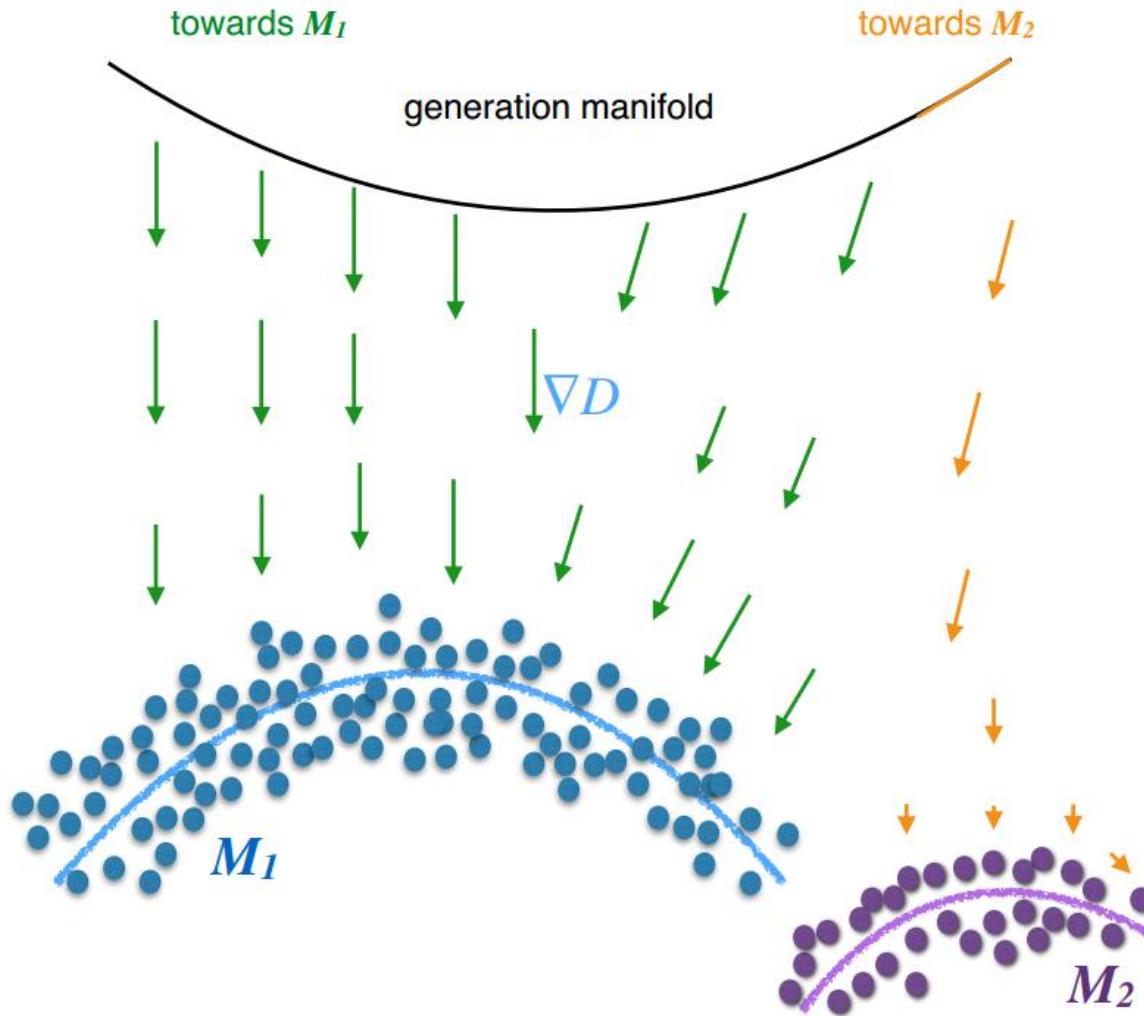
- Minimizing KL divergence only:

When $P_g(x) \rightarrow 0$, $P_r(x) \rightarrow 1$; $P_g(x)\log\frac{P_g(x)}{P_r(x)} \rightarrow 0$, $\text{KL}(P_g \parallel P_r) \rightarrow 0$

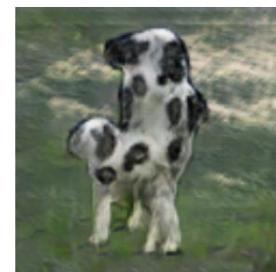
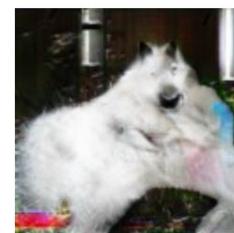
When $P_g(x) \rightarrow 1$, $P_r(x) \rightarrow 0$; $P_g(x)\log\frac{P_g(x)}{P_r(x)} \rightarrow +\infty$, $\text{KL}(P_g \parallel P_r) \rightarrow +\infty$



Mode missing



Other problems

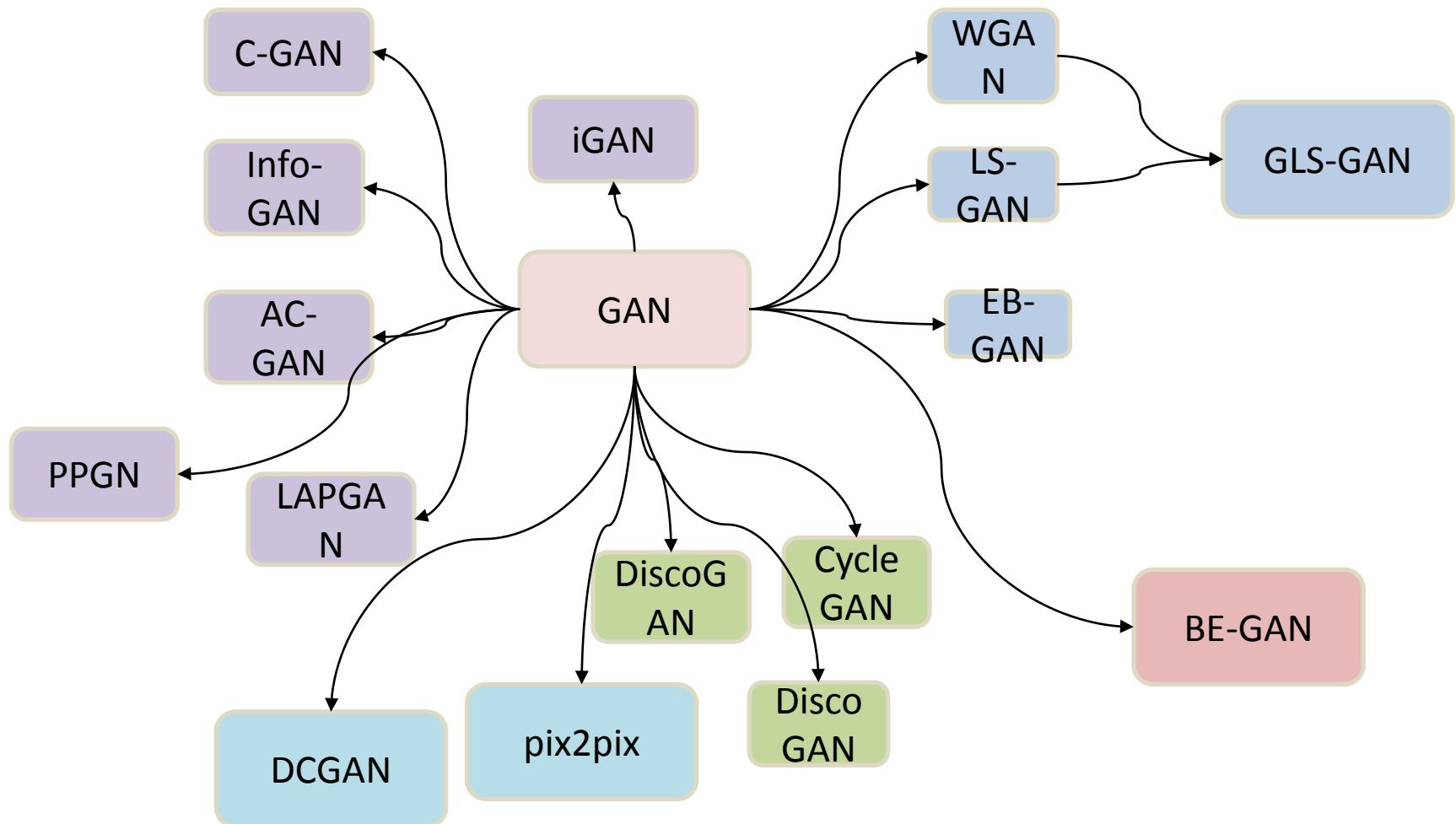


(Goodfellow 2016)

Counting

Global structure

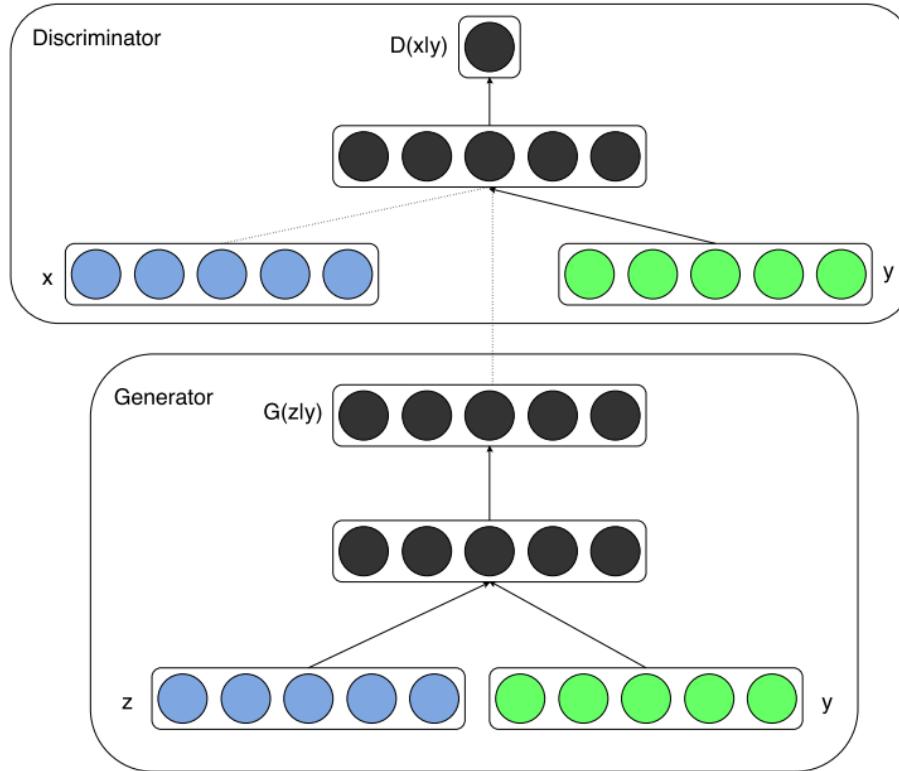
Research



Add More infomation

- CGAN
- InfoGAN
- iGAN

Conditional GAN



$$\min_G \max_D V(D, G) = E_{x \sim p_{data}(x)} [\log D(x | y)] + E_{x \sim p_z(x)} [\log(1 - D(G(z | y)))]$$

InfoGAN

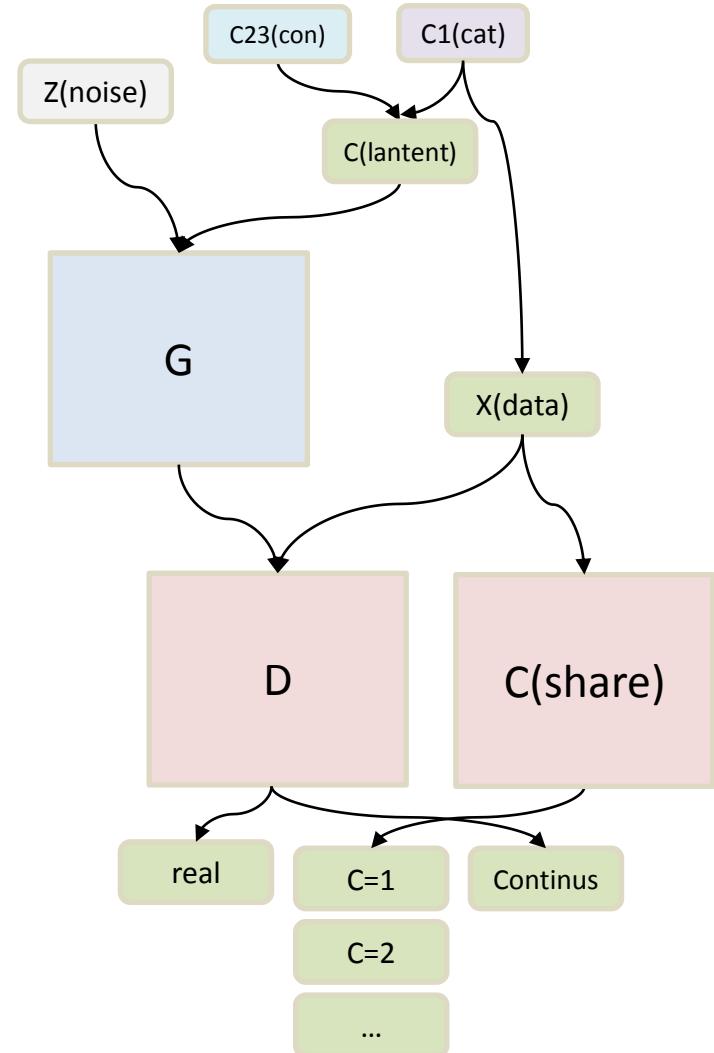
- Information theory

Mutual information:

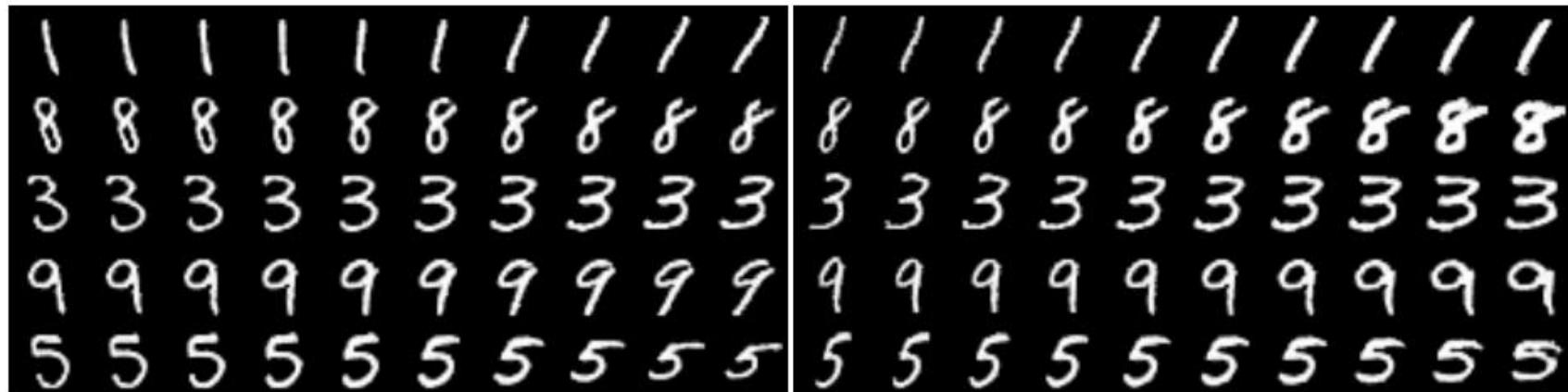
$$I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)$$

Objective function:

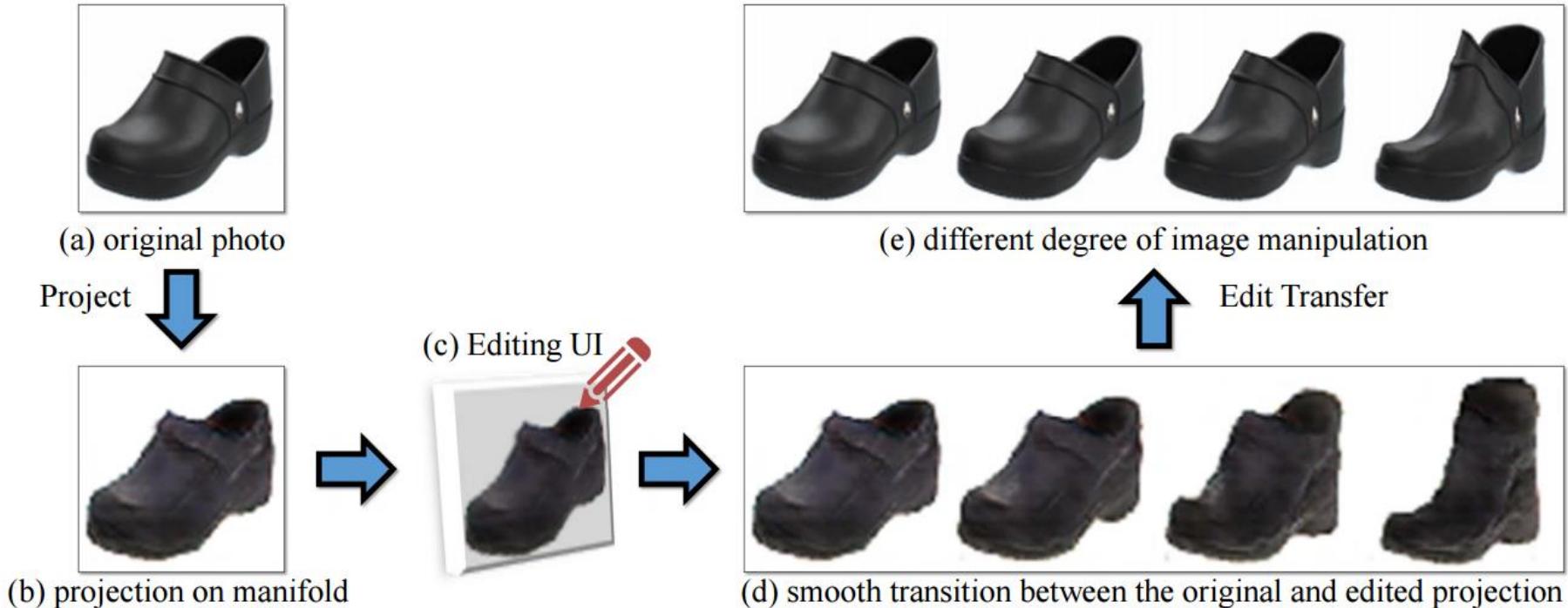
$$\min_G \max_D V_1(D, G) = V(D, G) - \lambda I(c; G(z, c))$$



InfoGAN



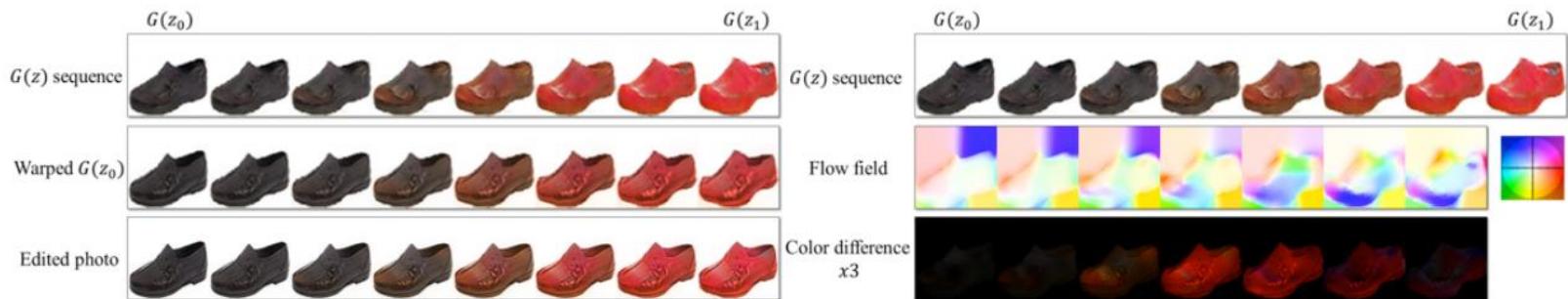
iGAN



iGAN

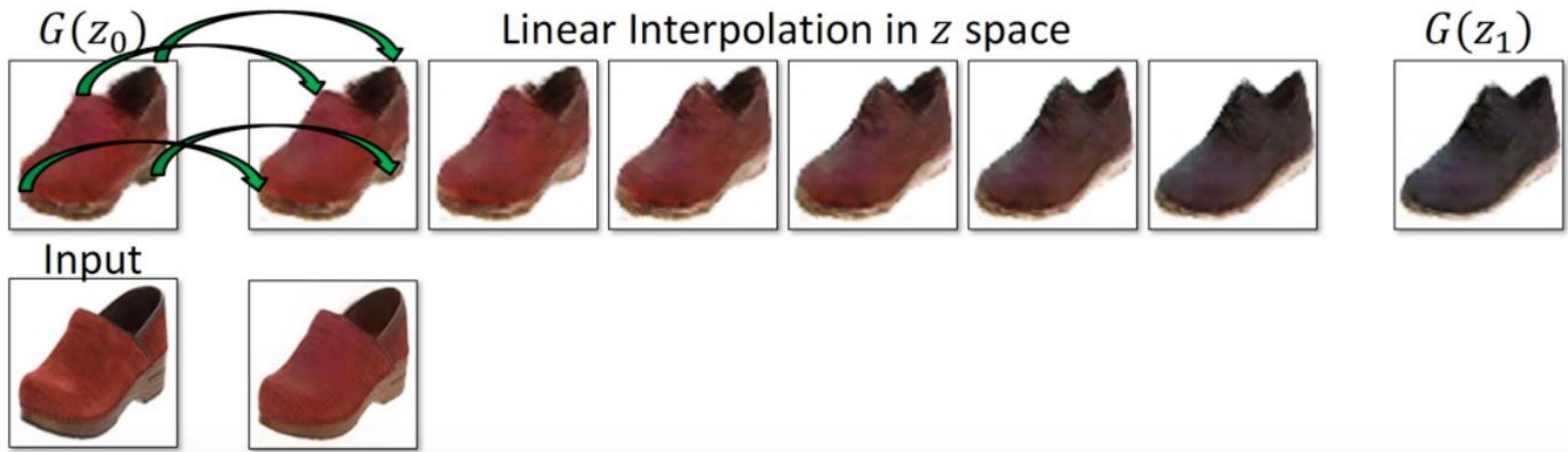
(a) User constraints v_g at different update steps $G(z_0)$

(b) Updated images according to user edits

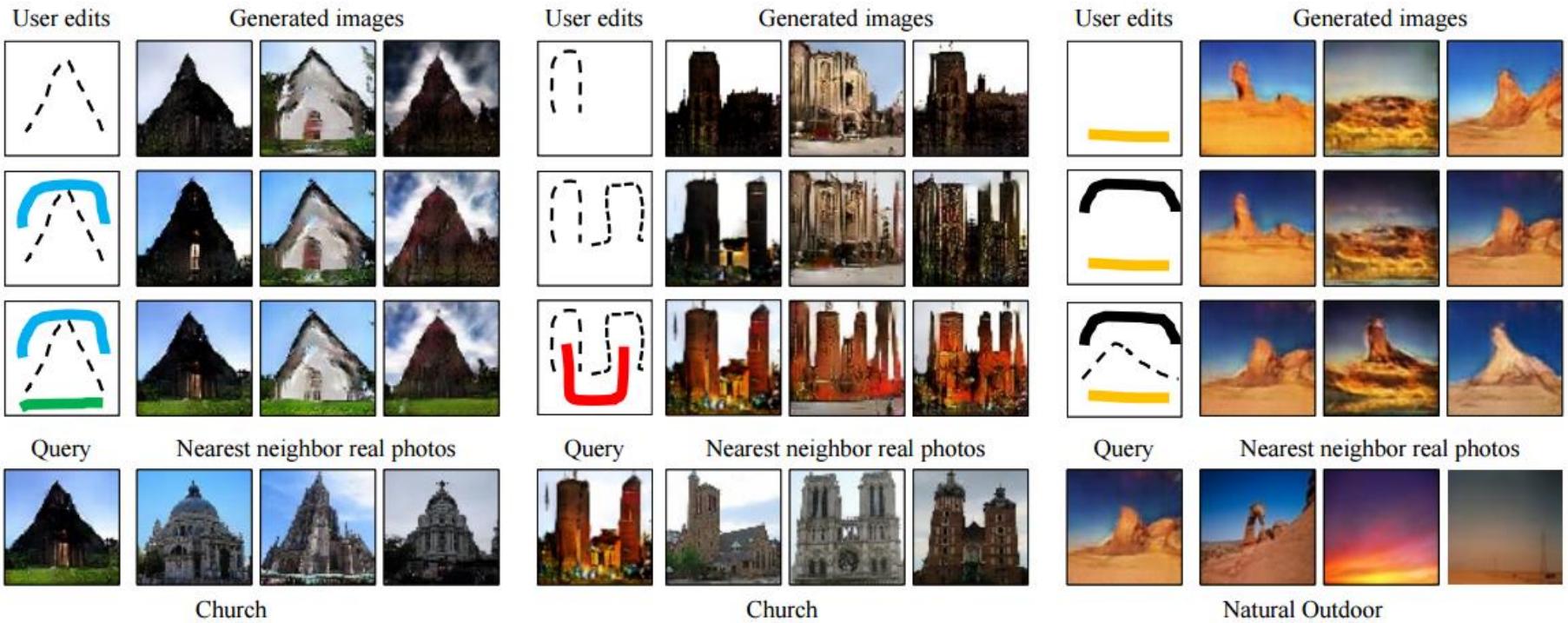
 $G(z_1)$ 

iGAN

$$\iint \underbrace{\|I(x, y, t) - A \cdot I(x+u, y+v, t+1)\|^2}_{\text{data term}} + \underbrace{\sigma_s (\|\nabla u\|^2 + \|\nabla v\|^2)}_{\text{spatial reg}} + \underbrace{\sigma_c \|\nabla A\|^2}_{\text{color reg}} dx dy$$



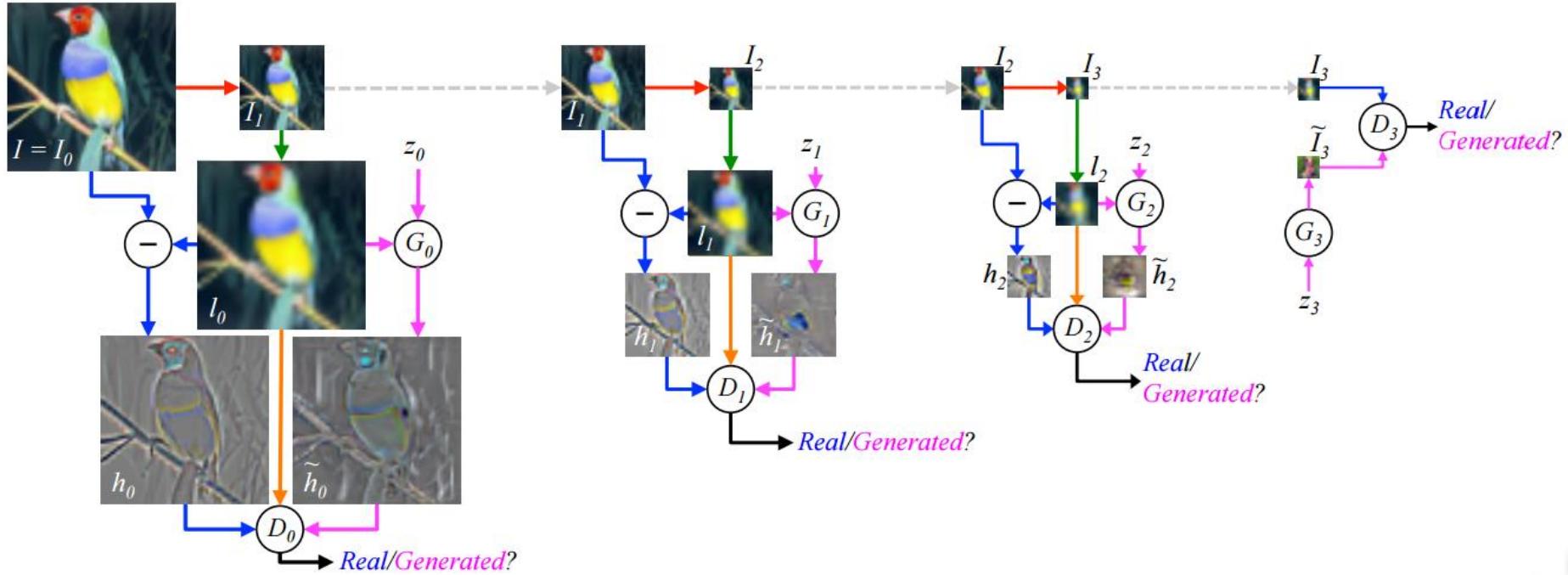
iGAN



Fine-grained Guidance

- LAPGAN
- Matching-aware Discriminator
- PPGN

LAPGAN



$$\min_G \max_D \mathbb{E}_{h,l \sim p_d(h,l)} [\log D(h,l)] + \mathbb{E}_{z \sim p_{noise}(z), l \sim p_l(l)} [\log(1 - D(G(z,l),l))]$$

Text to image

- Matching-aware Discriminator
 - Implicitly separate two source of error:unrealistic
 - Images, and realistic images of the wrong class that mismatch the conditioning
- $\mathbf{x} \leftarrow G(\mathbf{z}, h)$ {Forward through generator}
- $s_r \leftarrow D(x, h)$ {right image, right text}
- $s_w \leftarrow D(x, h)$ {real image, wrong text}
- $s_f \leftarrow D(x, h)$ {fake image, right text}

$$L_D \leftarrow \log(s_r) + (\log(1 - s_w) + \log(1 - s_f)) / 2$$

Text to image

Text descriptions
(content)

Images
(style)

The bird has a **yellow breast** with grey features and a small beak.

This is a large **white** bird with **black wings** and a **red head**.

A small bird with a **black head and wings** and features grey wings.

This bird has a **white breast**, brown and white coloring on its head and wings, and a thin pointy beak.

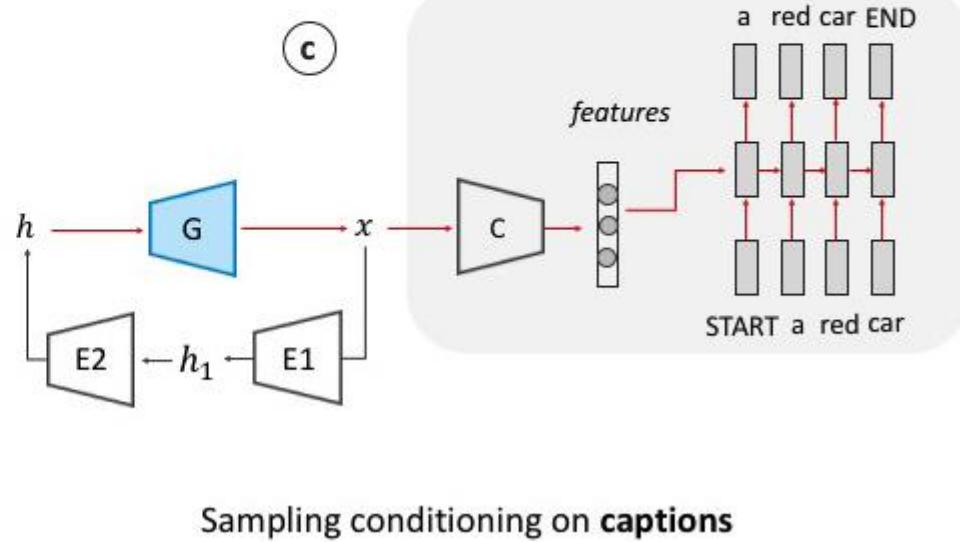
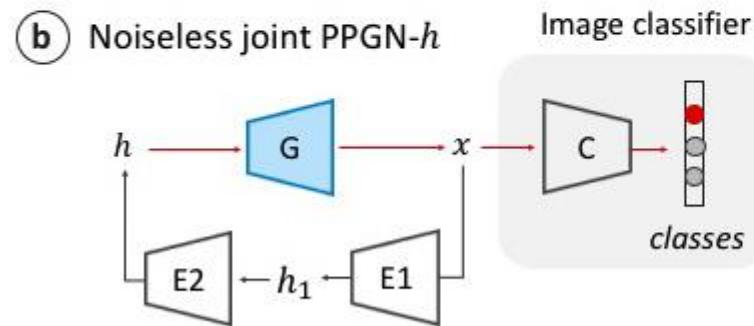
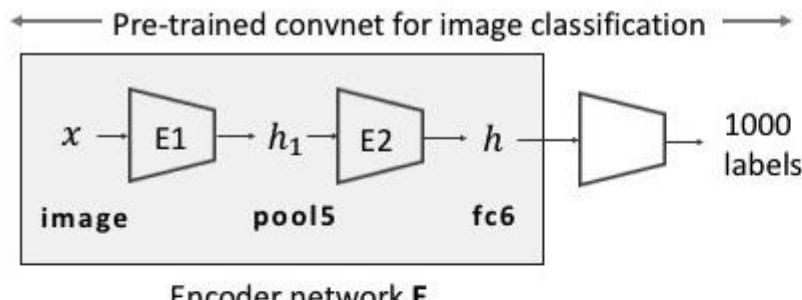
A small bird with **white base** and **black stripes** throughout its belly, head, and feathers.



PPGN



PPGN



PPGN



cardoon



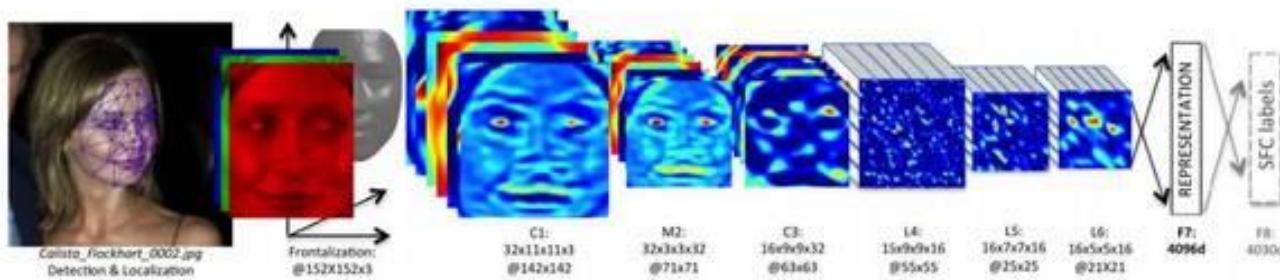
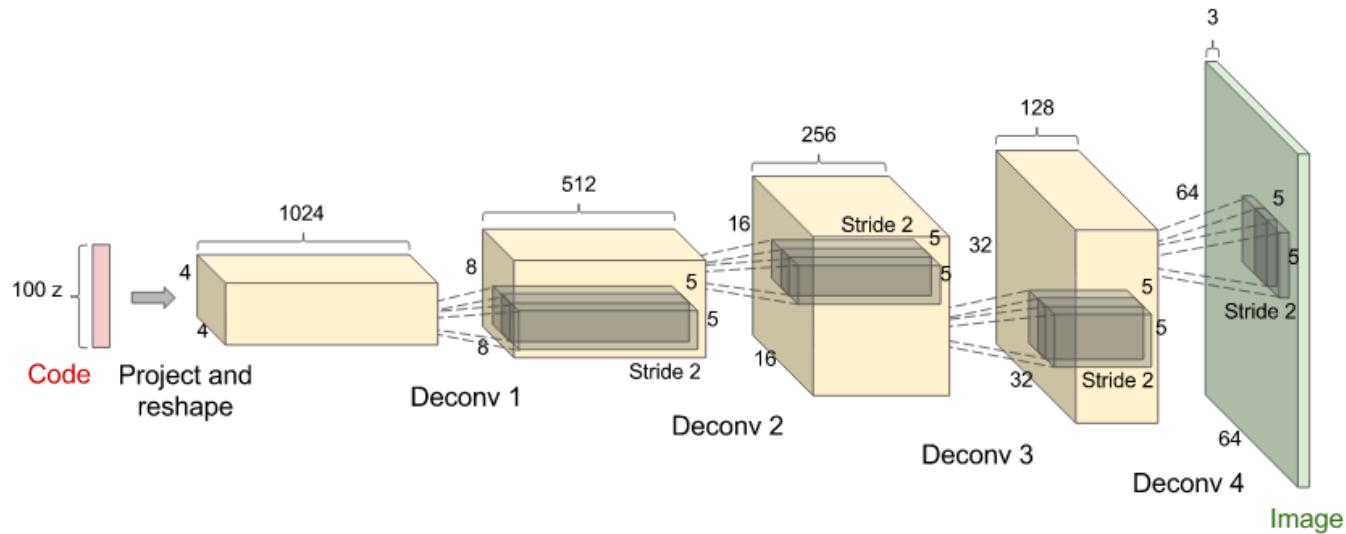
cardoon



cardoon

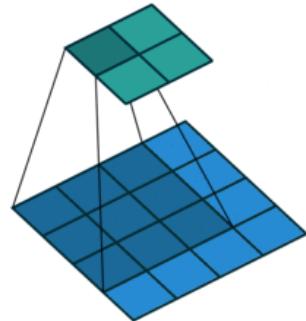
Architectures

DCGAN&tricks

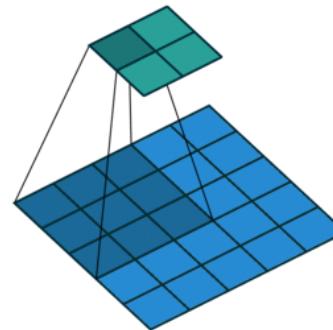


[Taigman et al. 2014]

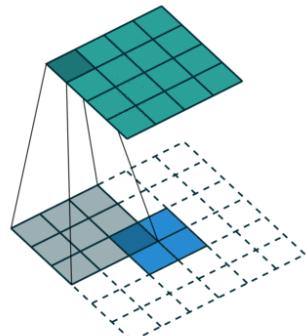
Unstranding ‘Deconvolution’



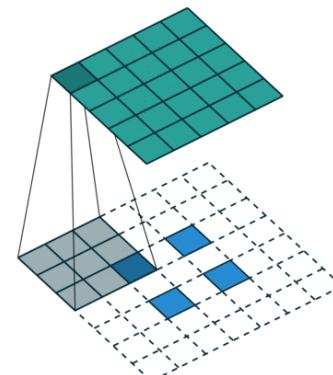
No padding, no stride



No padding, stride



No padding, stride
transposed



No padding, stride
transposed

In details

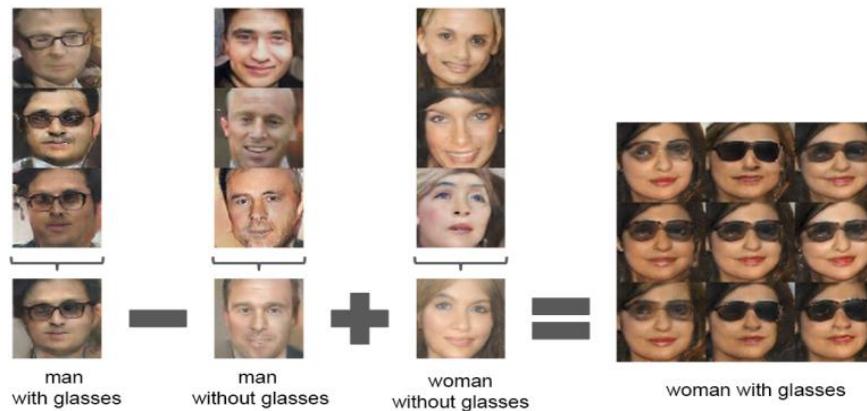
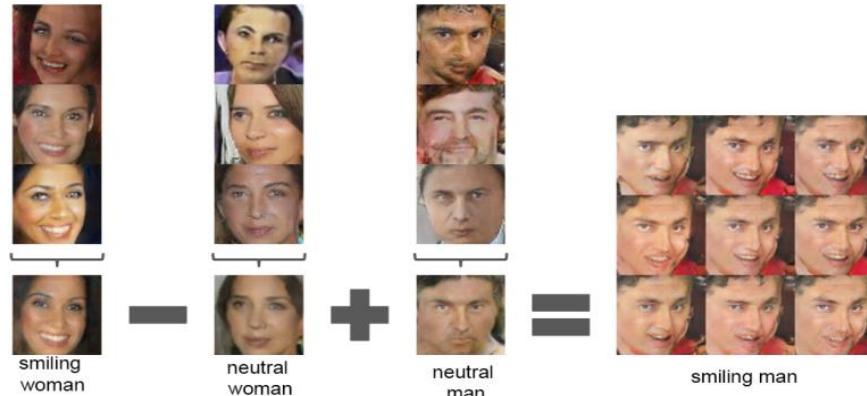
- For stable Deep Convolutional GANs
 - Replace any pooling with Convolutions (strided; fractional-strided)
 - Use batchnorm
 - Remove fully connected hidden layers
 - Use ReLu activation in G
 - Use leakyReLu activation in D

Results of DCGAN



On Lsun bedroom

Results of DCGAN

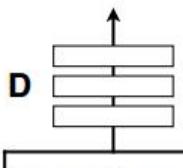


Face editing

Pix2pix

Positive examples

Real or fake pair?

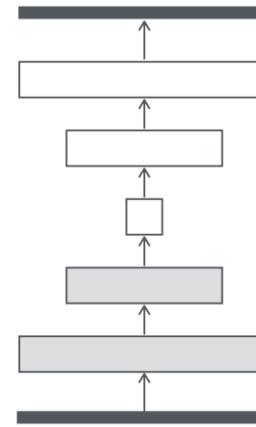
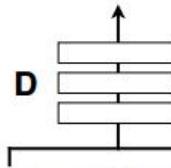


G tries to synthesize fake images that fool **D**

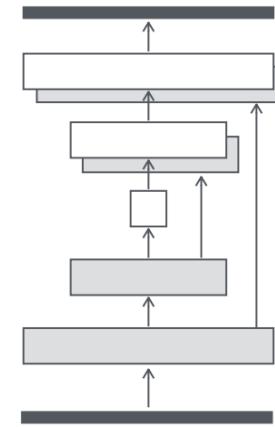
D tries to identify the fakes

Negative examples

Real or fake pair?

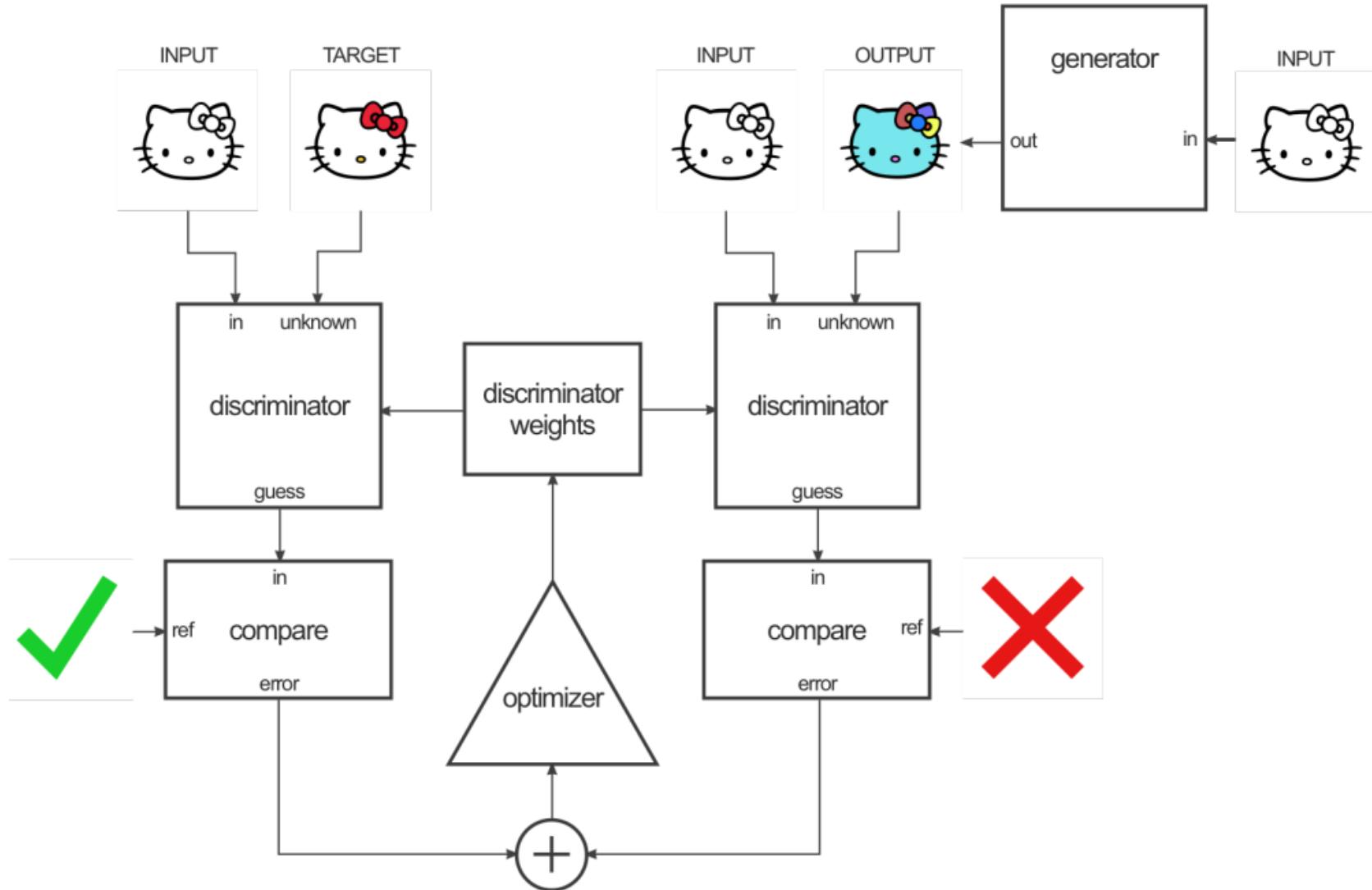


Encoder-decoder

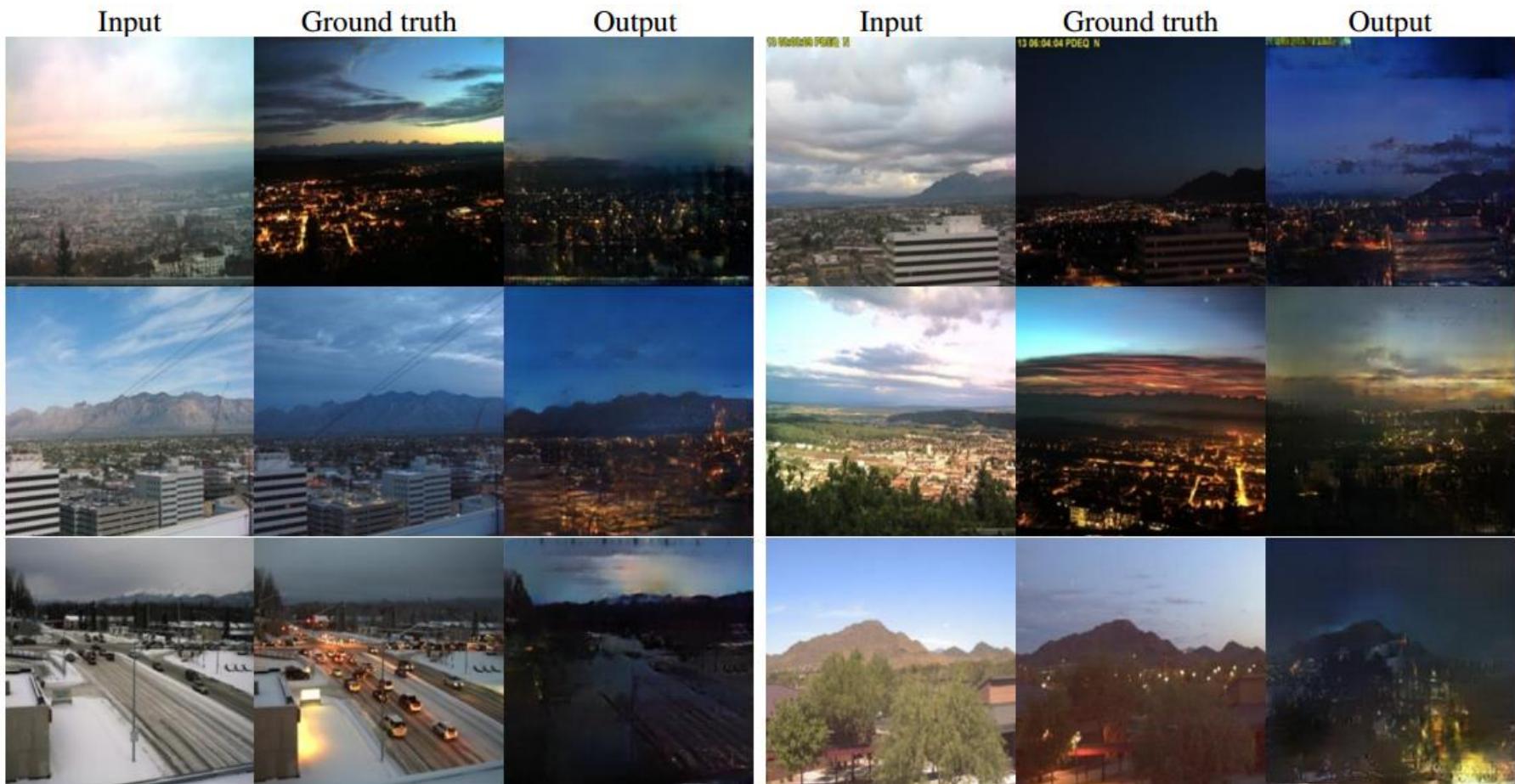


U-Net

Pix2pix



Pix2pix



Pix2pix



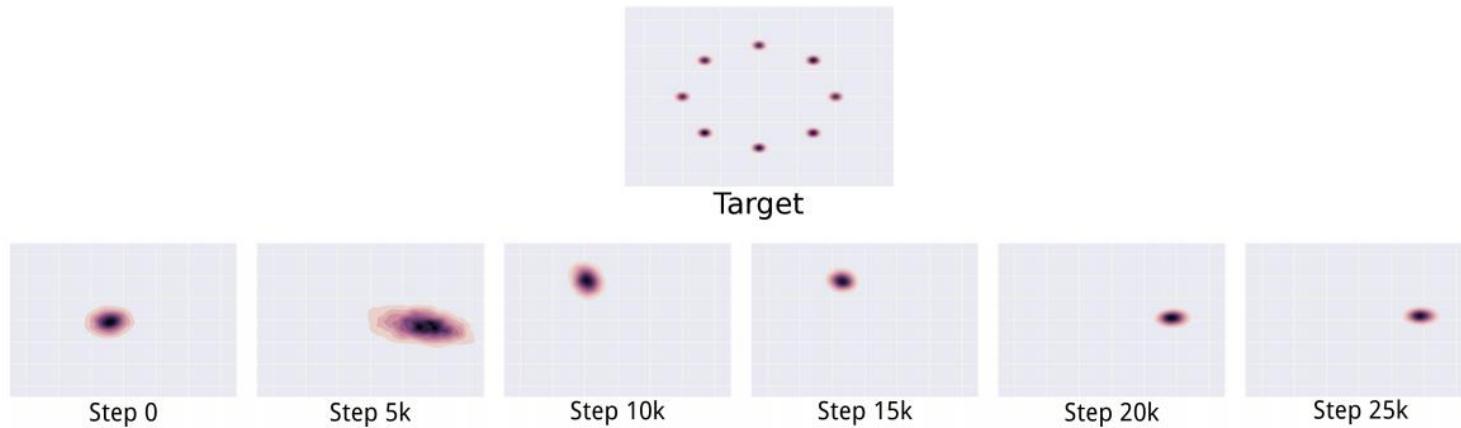
Encode effectives

- MRGAN
- EBGAN
- BEGAN

Review mode collapse

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

- D in inner loop: convergence to correct distribution
- G in inner loop: place all mass on most likely point



Mode Regularized GANs

- Regularized GANs

- encoder E:

$$\mathbb{E}_{x \sim p_d} [\lambda_1 d(x, G \circ E(x)) + \lambda_2 \log D(G \circ E(x))]$$

- generator G:

$$-\mathbb{E}_z [\log D(G(z))] + \mathbb{E}_{x \sim p_d} [\lambda_1 d(x, G \circ E(x)) + \lambda_2 \log D(G \circ E(x))]$$

- discriminator D: same as vanilla GAN

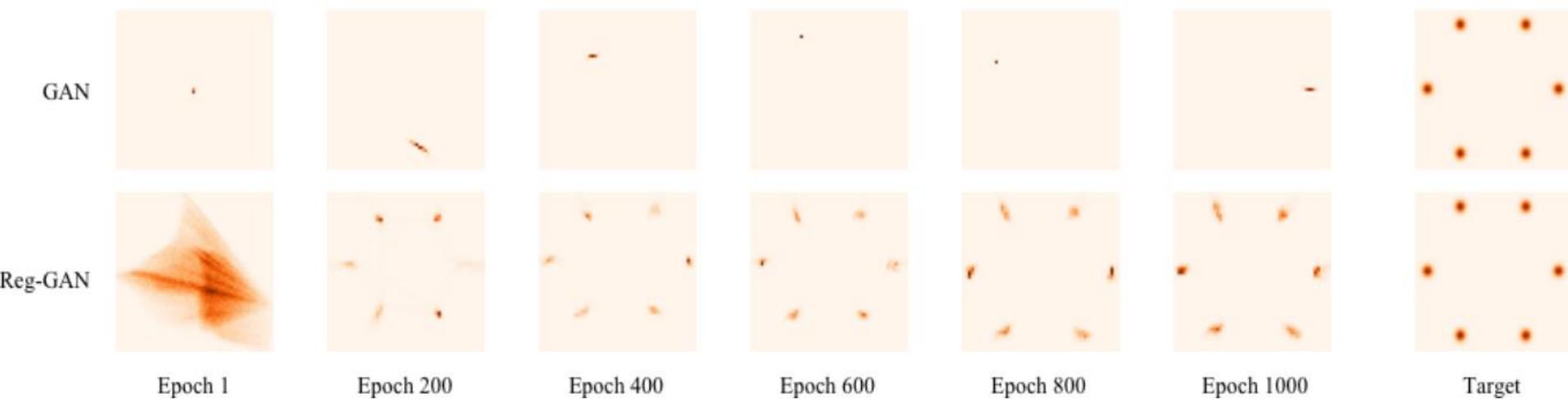
- Still Suffer from gradient vanishing

Mode Regularized GANs

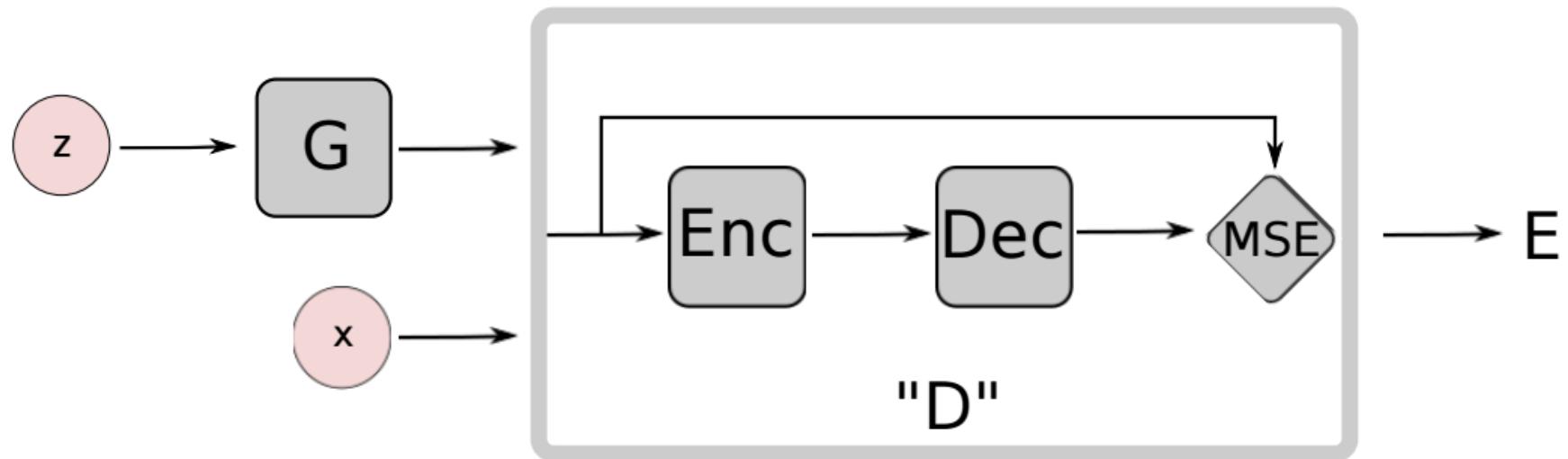
Manifold-Diffusion GANs:

- Manifold-step:
 - Match generation manifold and real data manifold
- Diffusion step:
 - Distribute the probability mass on the generation manifold fairly
- D: firstly comparing between real data and the encoded data

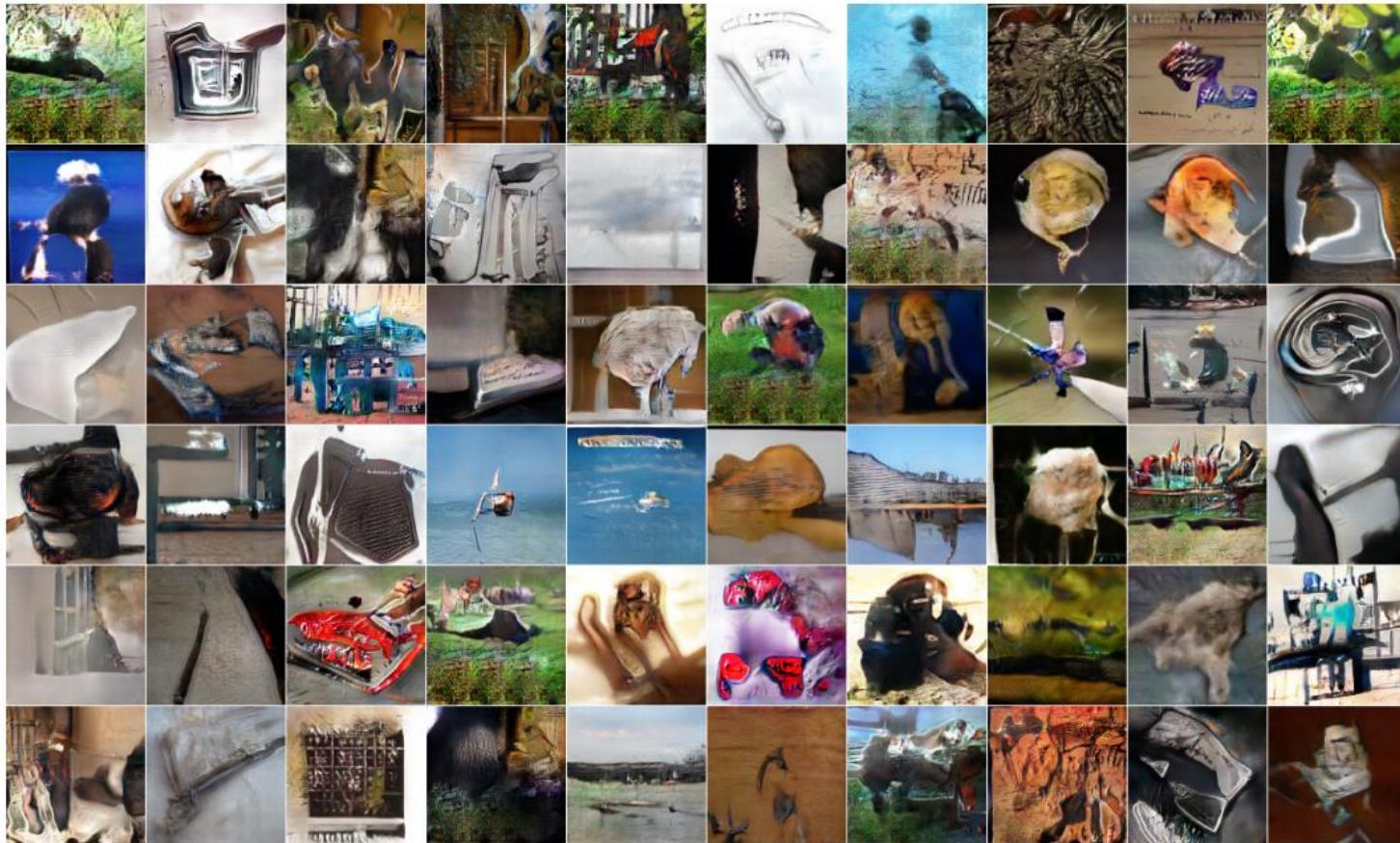
Mode Regularized GANs



EBGAN

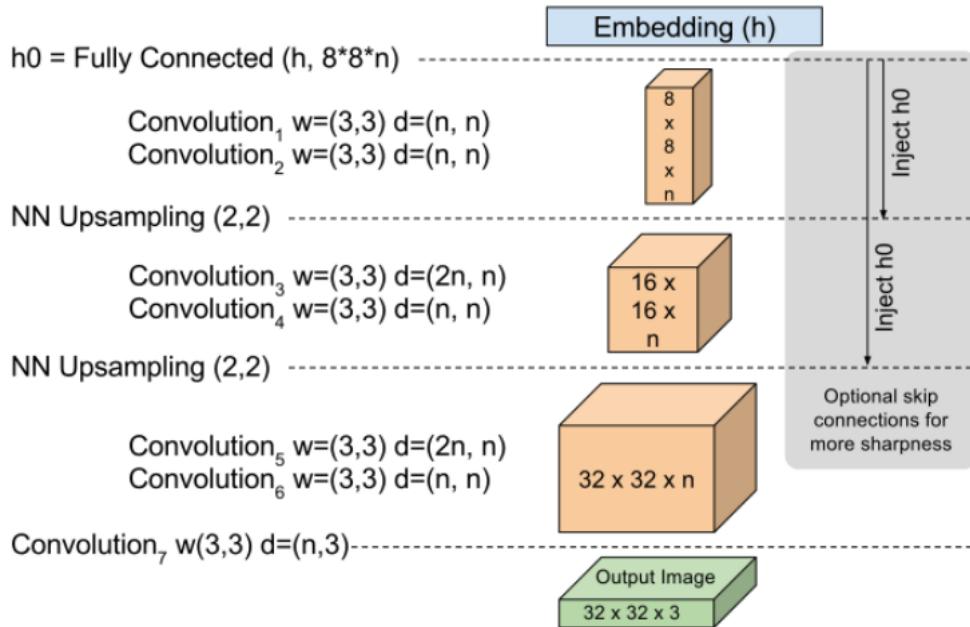


EBGAN

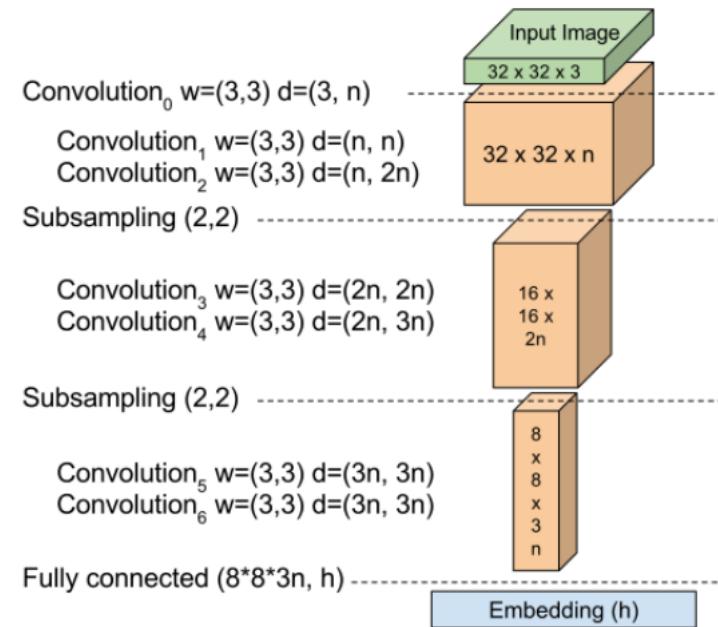


On ImageNet

BEGAN



(a) Generator/Decoder



(b) Encoder

BEGAN

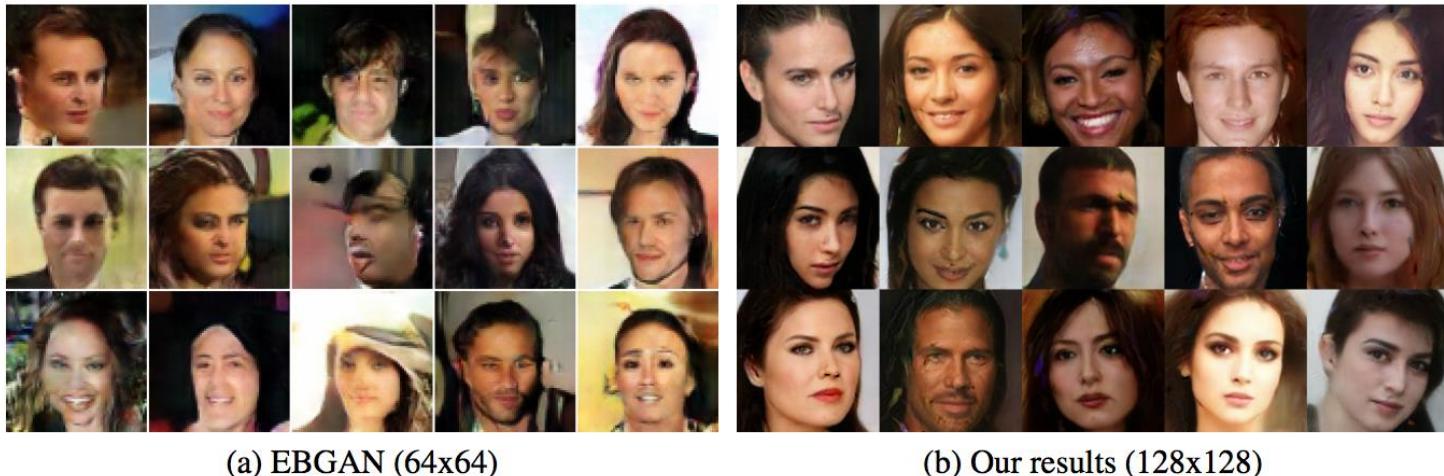


Figure 2: Random samples comparison



Figure 3: Random 64x64 samples at varying diversities $\gamma \in \{0.3, 0.5, 0.7\}$

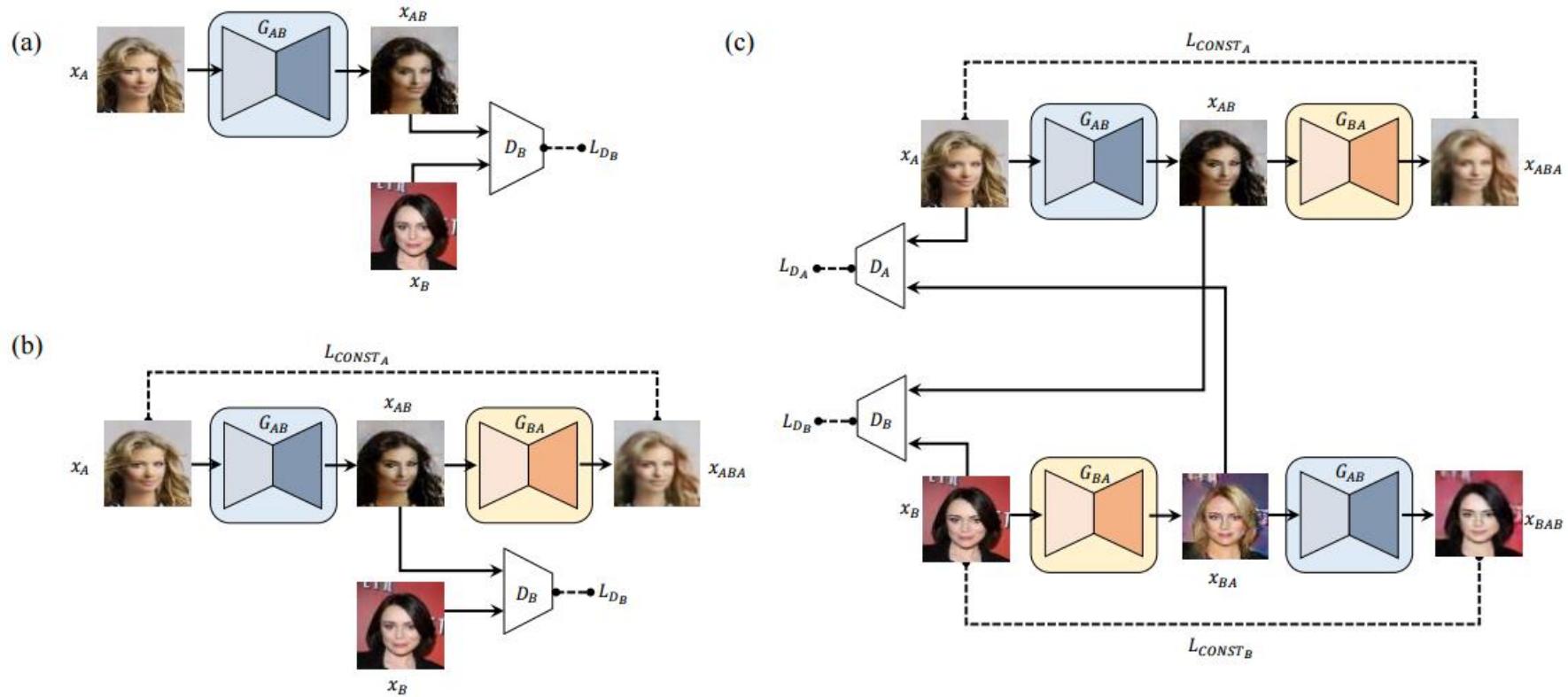
BEGAN



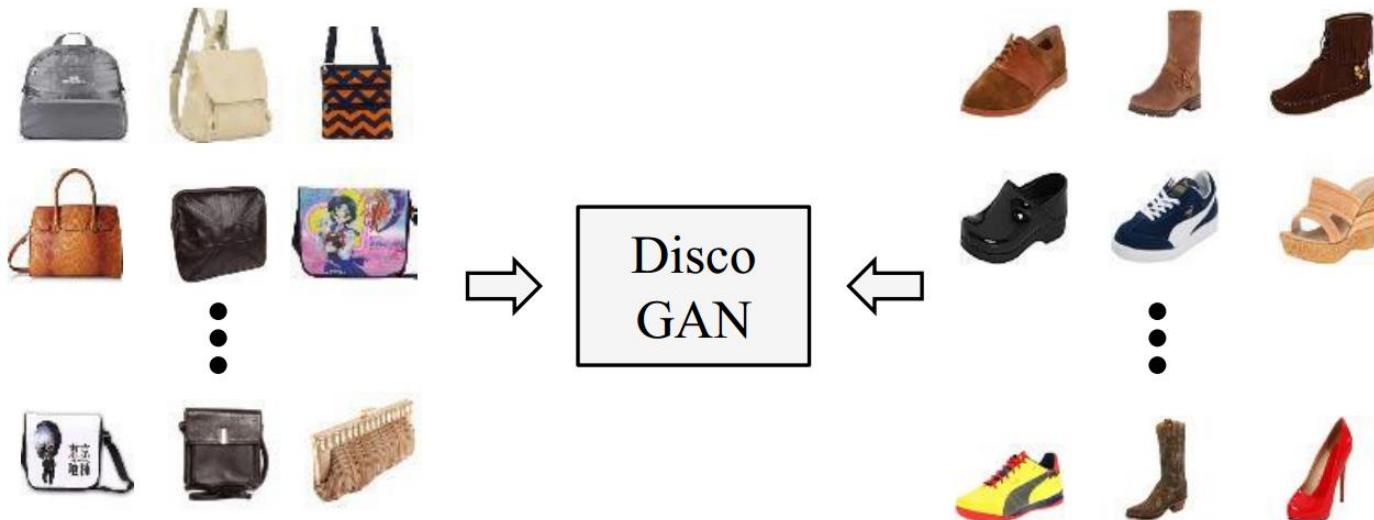
GAN “三兄弟”

- CycleGAN
- DiscoGAN
- DualGAN

DiscoGAN



Learning to discover cross-domain relations

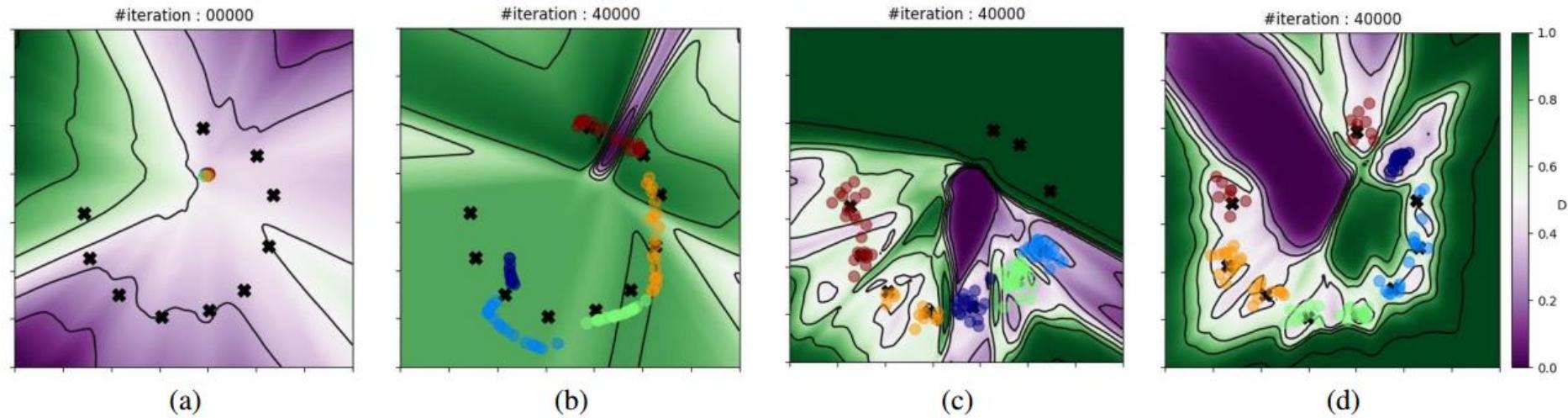


(a) Learning cross-domain relations **without any extra label**

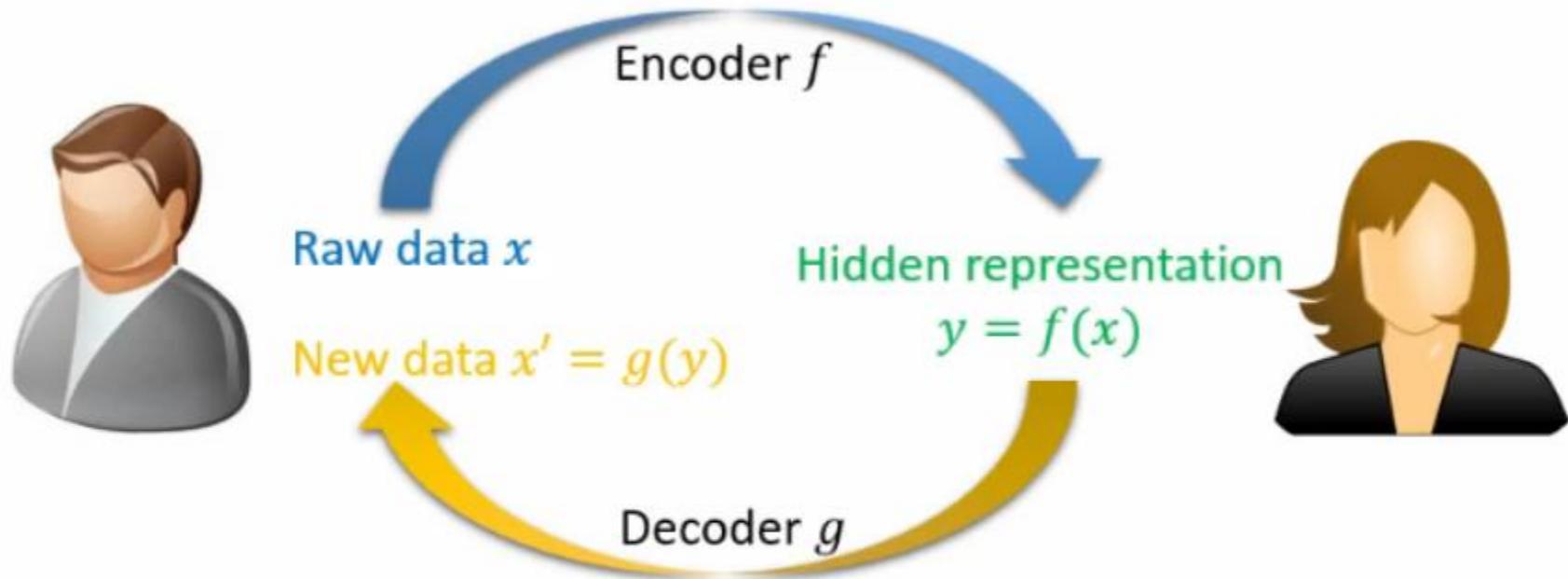


(b) Handbag images (input) & **Generated** shoe images (output)

Mode missing



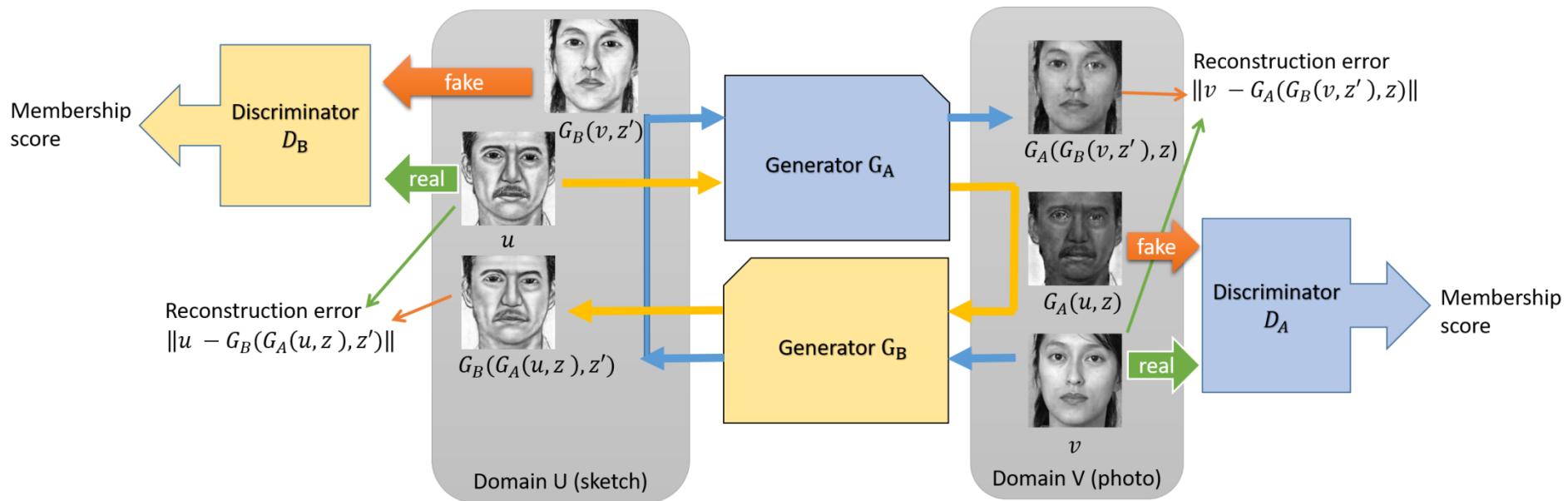
Dual Learning

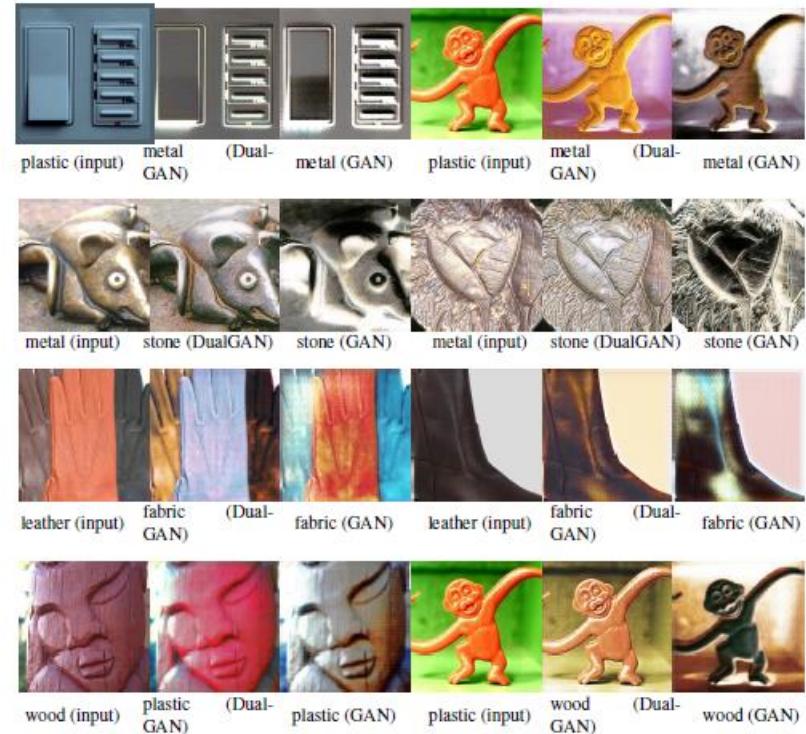
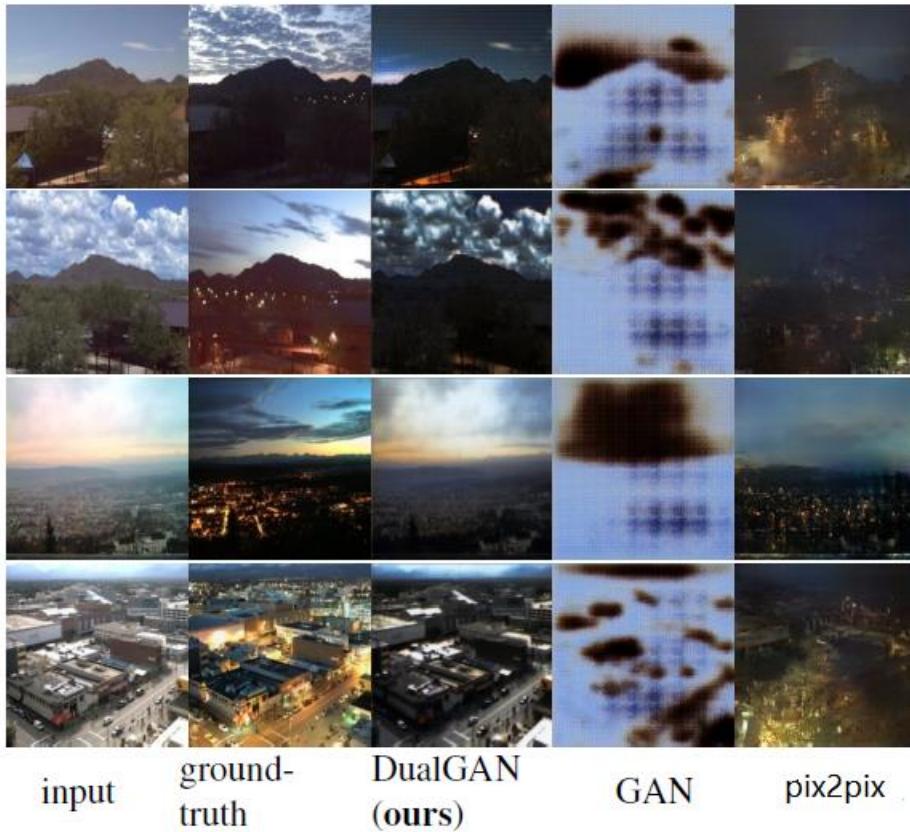


Feedback signals during the loop:

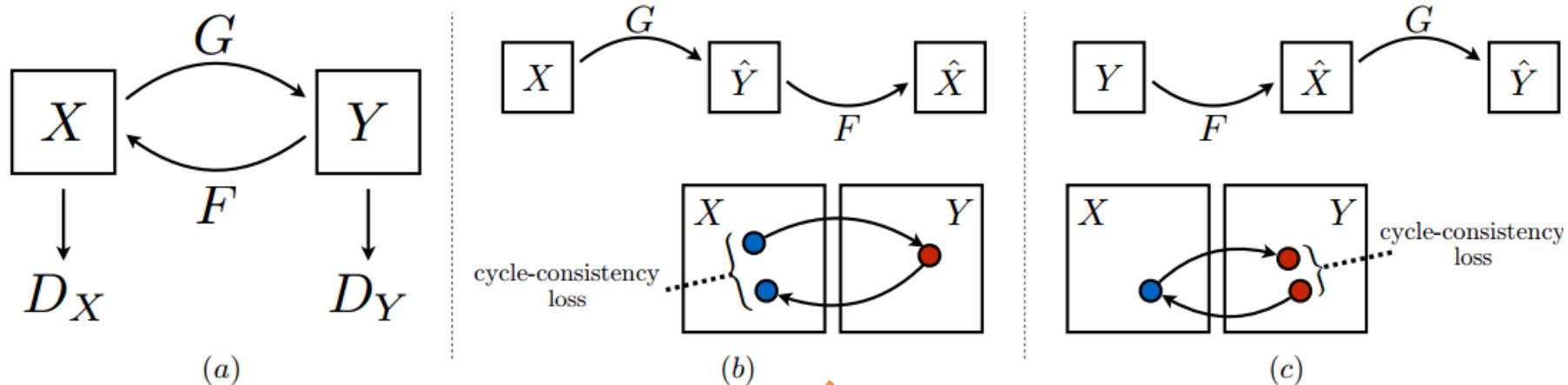
- $R(x, f, g) = s(x, x')$: similarity between x and x'

DualGAN





CycleGAN

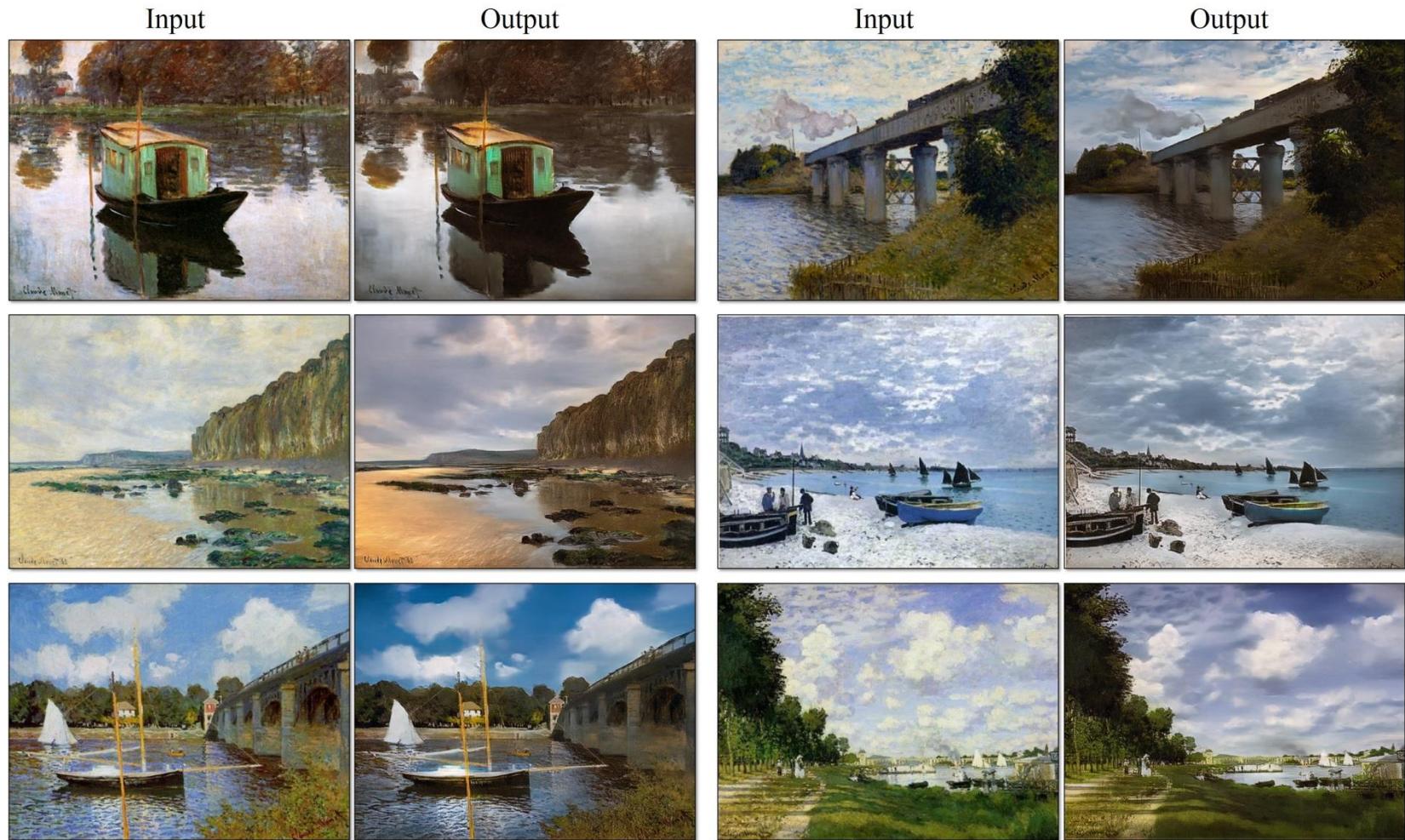


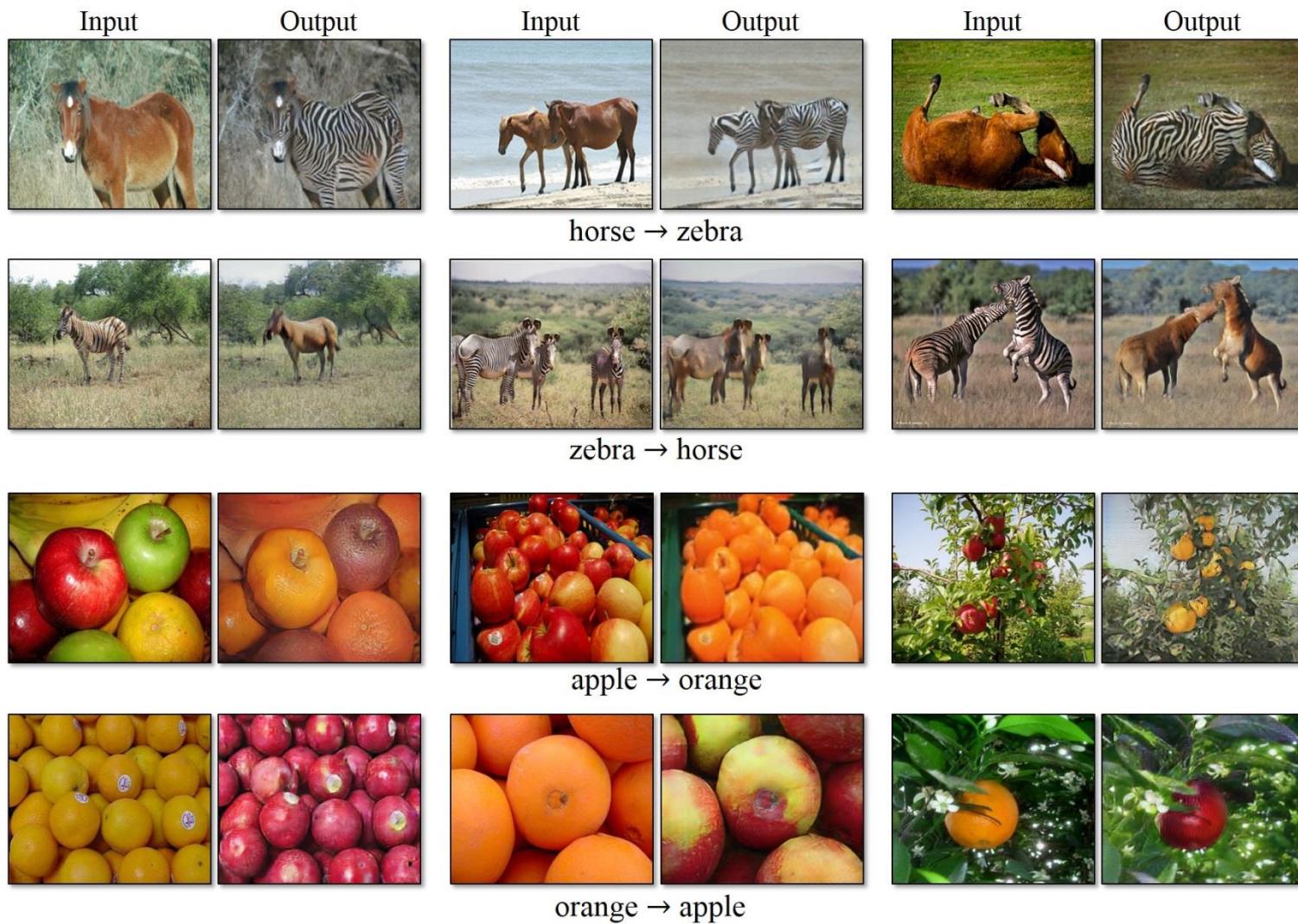
(a)

(b)

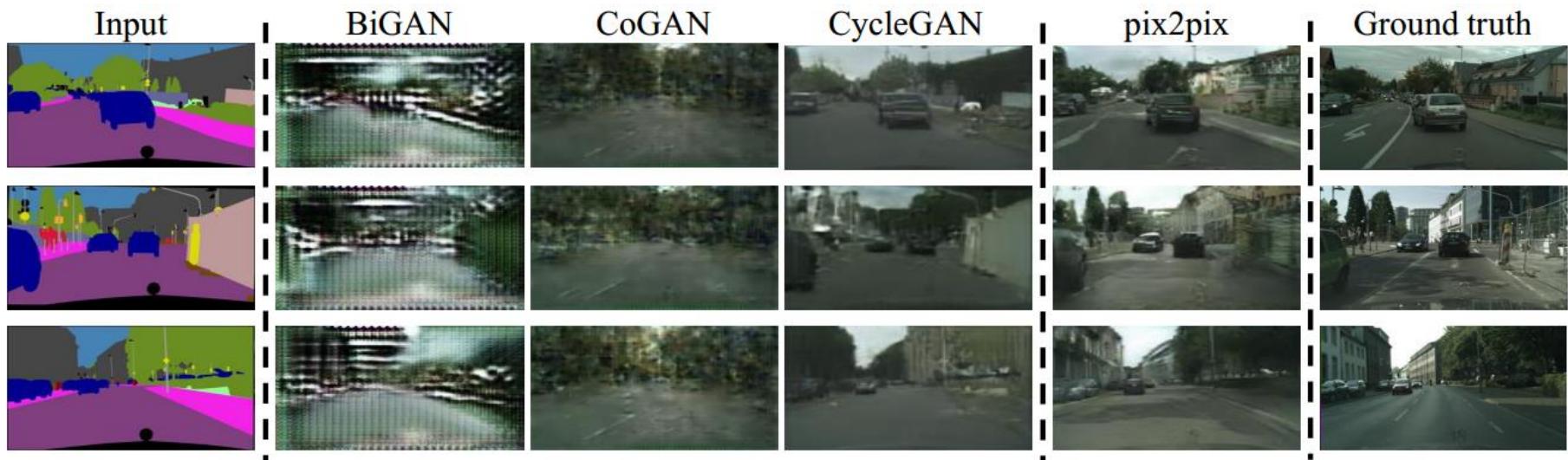
(c)

- $$L_{cyc}(G, F) = E_{x \sim p_{data(x)}} [| | F(G(x)) - x | |] + E_{y \sim p_{data(y)}} [| | G(F(y)) - y | |]$$



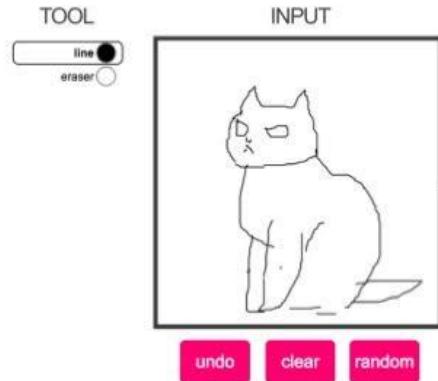


Encoders-incorporated

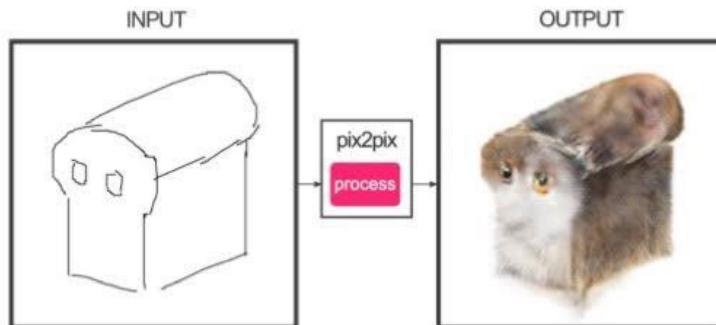


edges2cats

User Interface



@gods_tail



Ivy Tasi @ivymyt

Results



Vitaly Vidmirov @vvid



@ka92

- <https://affinelayer.com/pixsrv/>

WGAN

- Wasserstein Distance:

$$W(P_r, P_g) = \inf_{\gamma \in \Pi(P_r, P_g)} E_{(x,y)}[\|x - y\|]$$

- Why is it superior to KL & JS divergence?

Wasserstein Distance

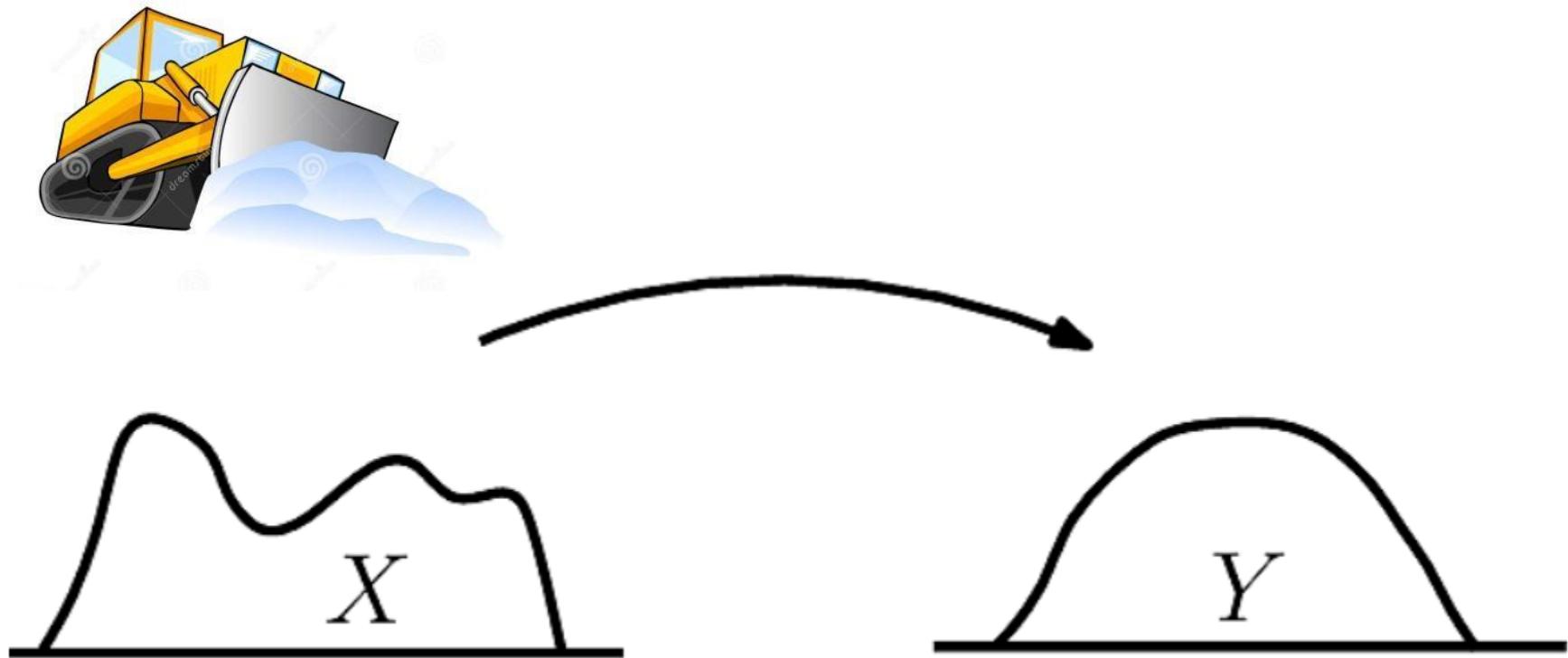
- Wasserstein Distance:

$$W(P_r, P_g) = \inf_{\gamma \in \Pi(P_r, P_g)} E_{(x,y)}[\|x - y\|]$$

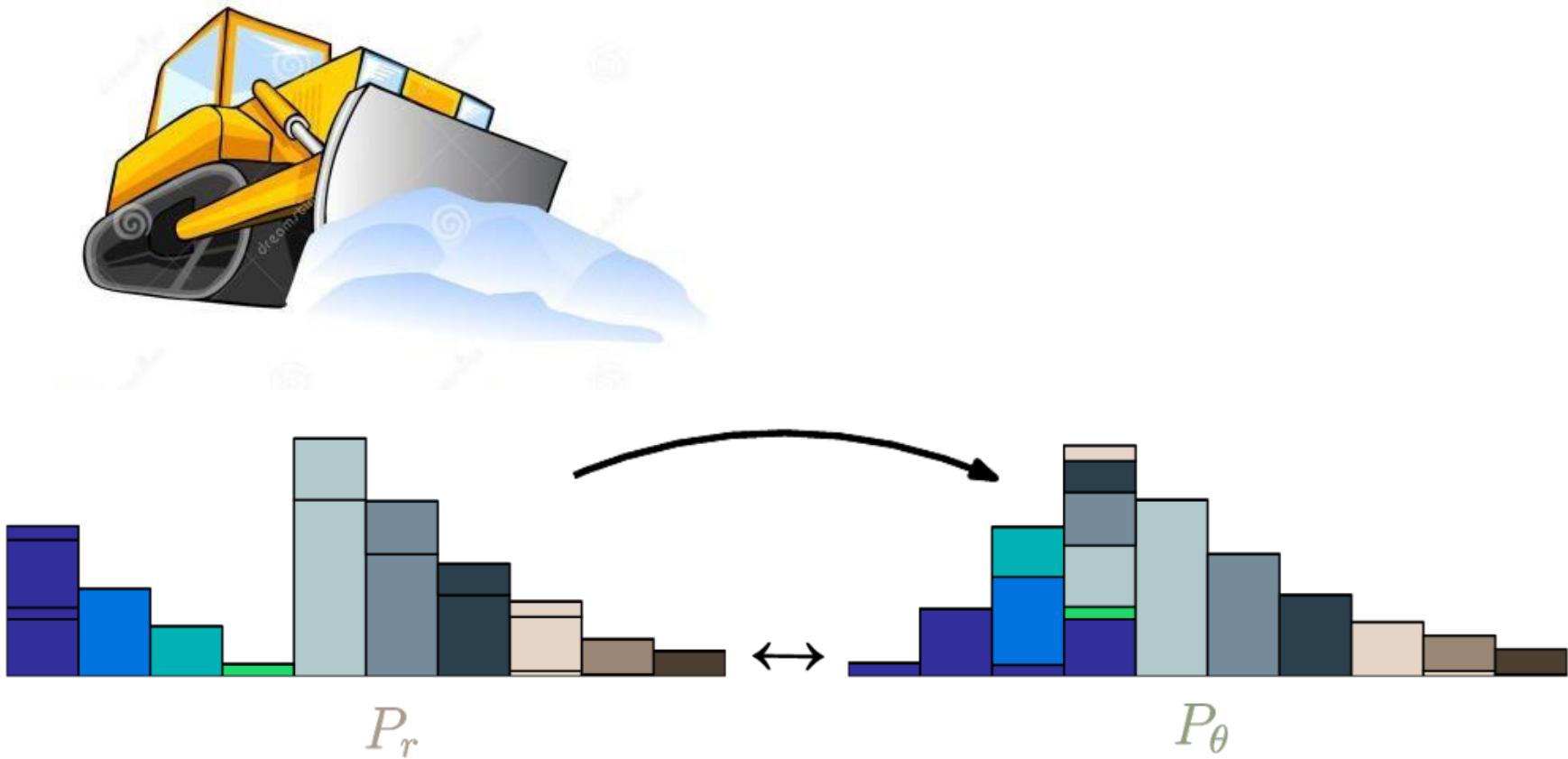
- $\Pi(P_r, P_g)$ denotes the set of all joint distribution $\gamma(x, y)$ whose marginals are respectively P_r and P_g .
- $\gamma(x, y)$ indicates how much “mass” must be transported from x to y in order to transform the distribution P_r to P_g .
- EM distance is the cost of the optimal transport plan

Wasserstein distance

- “吊打” KL & JS



Wasserstein Distance



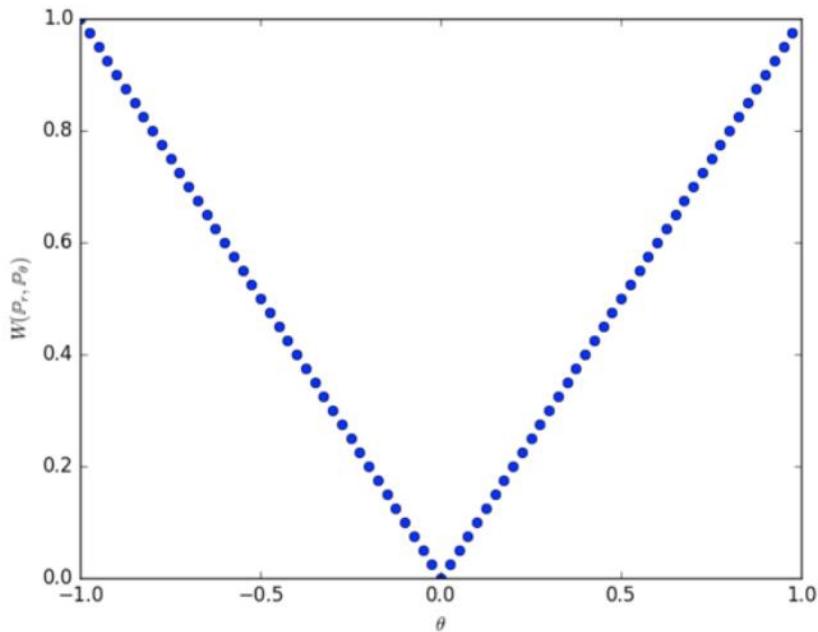
Wasserstein Distance

- Wasserstein Distance:

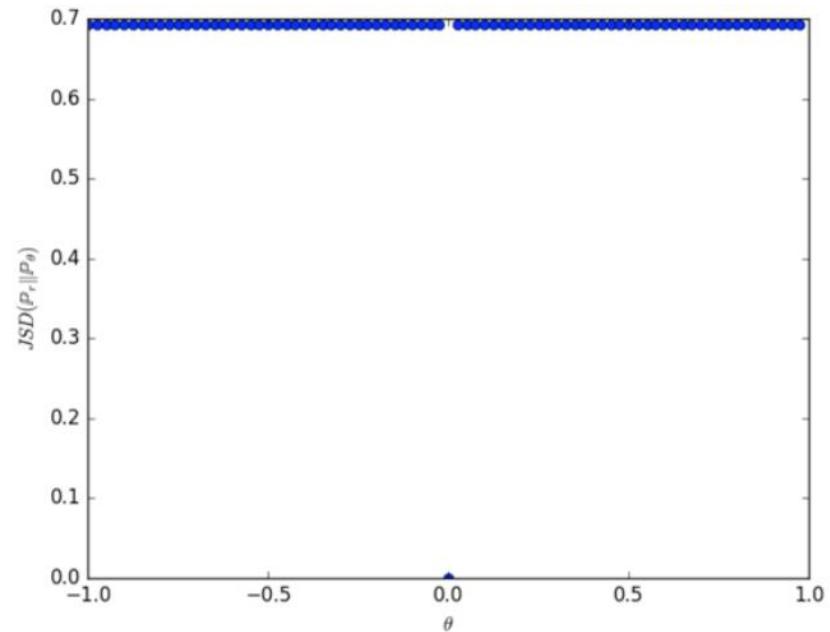
$$W(P_r, P_g) = \inf_{\gamma \in \Pi(P_r, P_g)} E_{(x,y)}[\|x - y\|]$$

- The distance is shown to have the desirable property
- Continuous everywhere
- Differentiable almost everywhere
- Most importantly:
 - It can reflect the distance of two distributions even if they do not overlap, and thus can provide meaningful gradient.

Wasserstein Distance



Wasserstein distance



JS Divergence

From Wasserstein Distance to WGAN

- Wasserstein Distance:

$$W(P_r, P_g) = \inf_{\gamma \in \Pi(P_r, P_g)} E_{(x,y)}[\|x - y\|]$$

- By applying the kantorovich-Rubinstein duality it becomes:

$$\min_G \max_{D \in D} E_{x \sim P_r}[D(x)] - \tilde{E}_{x \sim P_g}[D(x)]$$

Lipschitz Continuity

- Real-value function: $f: R \rightarrow R$

- Positive constant: K

$$|f(x_1) - f(x_2)| \leq K|x_1 - x_2|$$

- In other words, a **Lipschitz continuous** function has bounded first derivative. Intuitively, the slope of a KK Lipschitz function never exceeds KK.

$$d_Y(f(x_1), f(x_2)) \leq K d_X(x_1, x_2)$$

From Wasserstein Distance to WGAN

- Wasserstein Distance:

$$\min_G \max_{D \in \mathcal{D}} E_{x \sim P_r} [D(x)] - E_{\tilde{x} \sim P_g} [\tilde{D}(\tilde{x})]$$

- To satisfy the requirement, WGAN apply
weight clipping $[-c, c]$

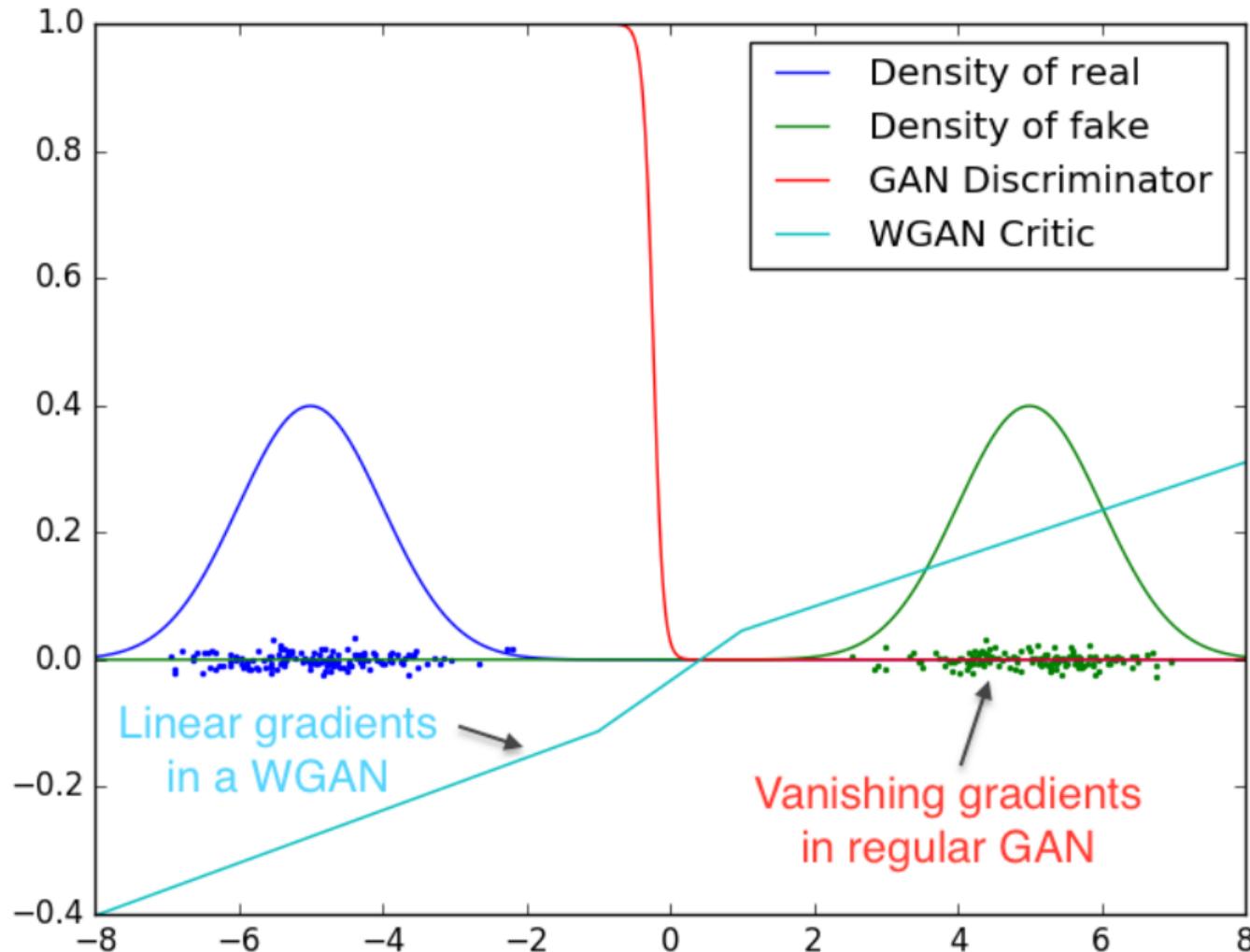
From Wasserstein Distance to WGAN

- Wasserstein Distance:

$$\min_G \max_{D \in D} E_{x \sim P_r} [D(x)] - E_{\tilde{x} \sim P_g} [\tilde{D}(\tilde{x})]$$

- WGAN removes the sigmoid layer in D because by using Wasserstein distance, D is doing regression rather than classification.

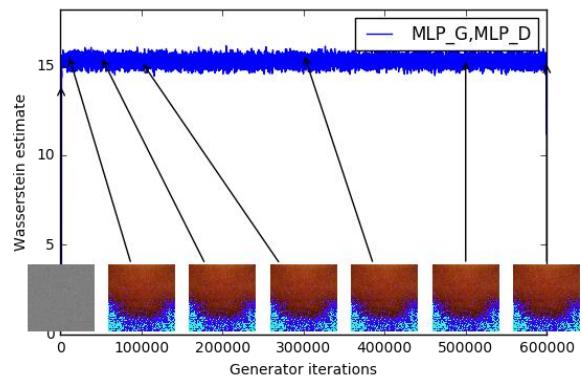
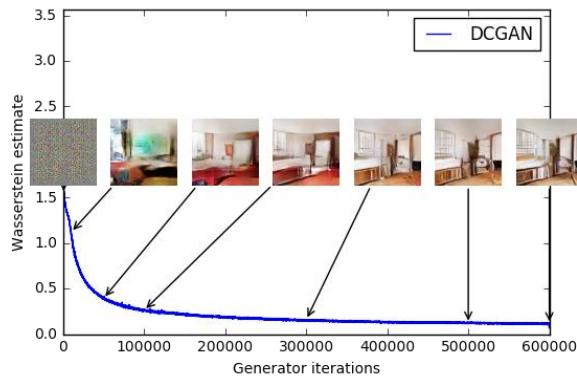
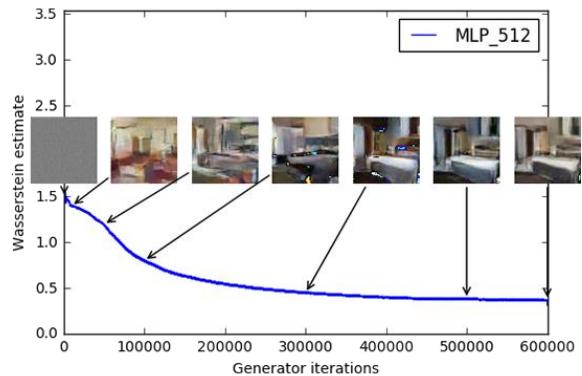
From Wasserstein Distance to WGAN



Efficient of WGAN

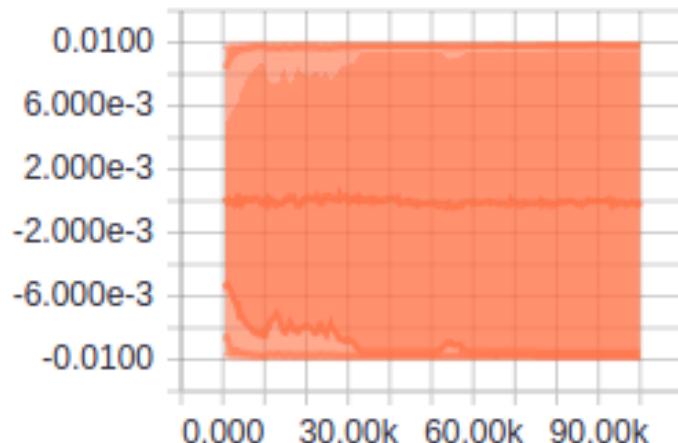
- GAN loss **corresponds** with image quality
- **Converge**, so you can tune the hyper-parameters
- **Stable** training; even without BN
- Way **less mode collapse**
- Theory about why it works and JS is terrible

Efficient of WGAN

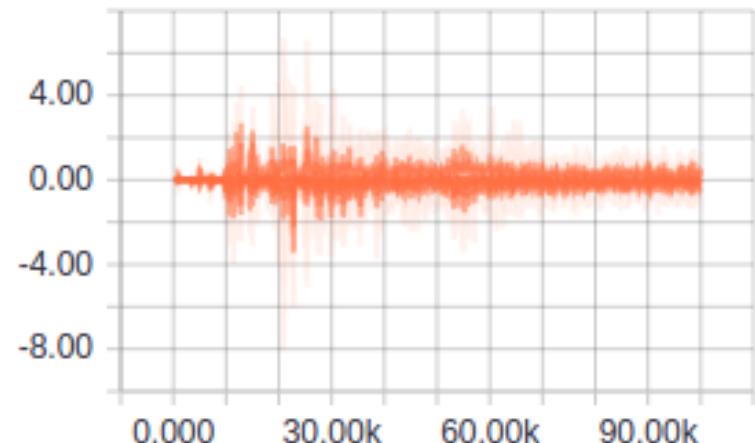


Efficient of WGAN

discriminator/W_0

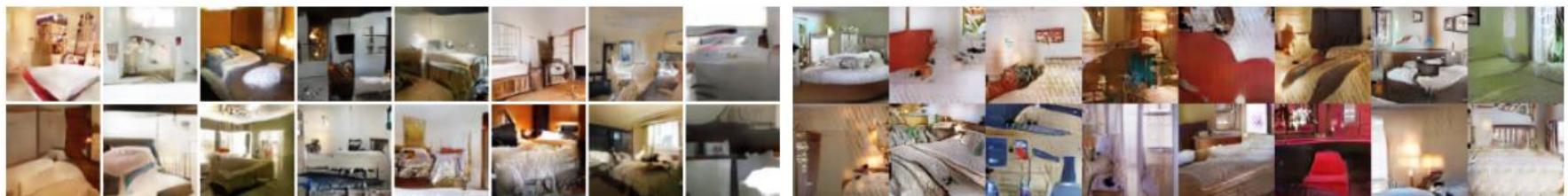


discriminator/W_0/gradient



Training WGAN

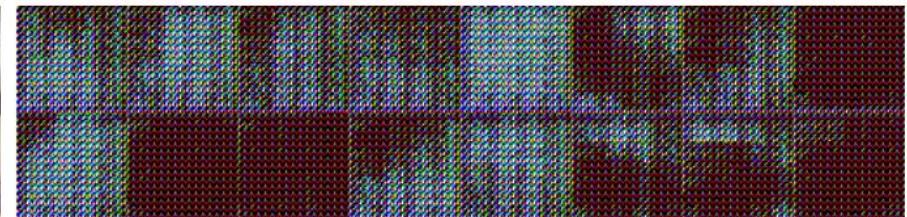
GAN & DCGAN



WGAN

DCGAN

Without BN

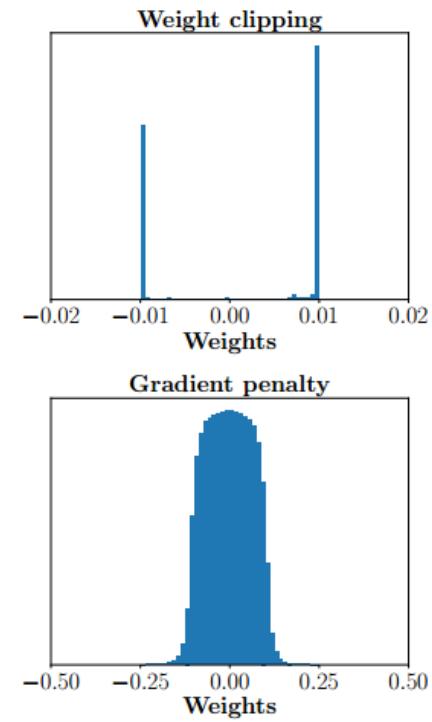
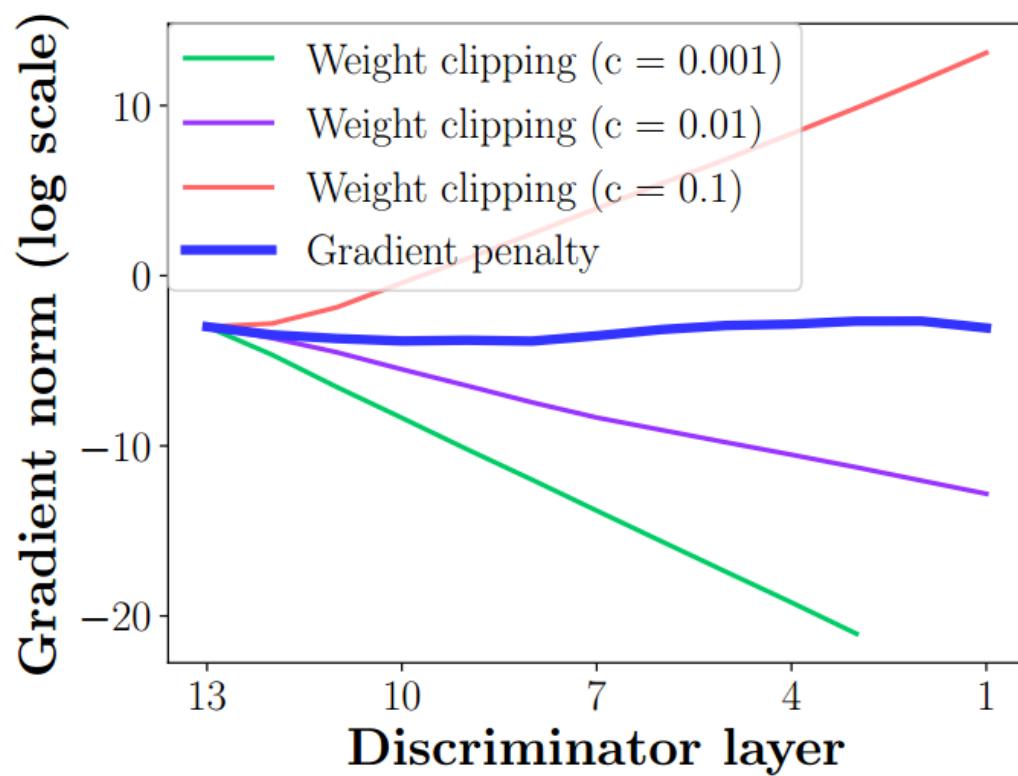


WGAN

DCGAN

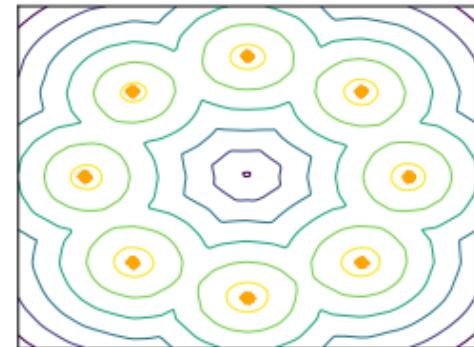
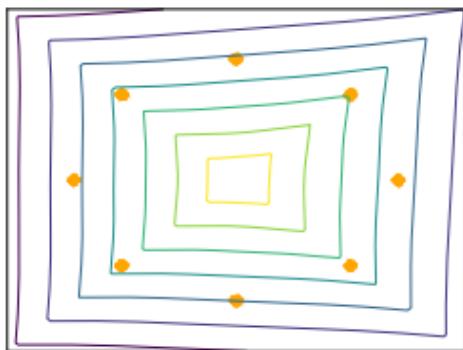
Improved WGAN

- Weight clipping

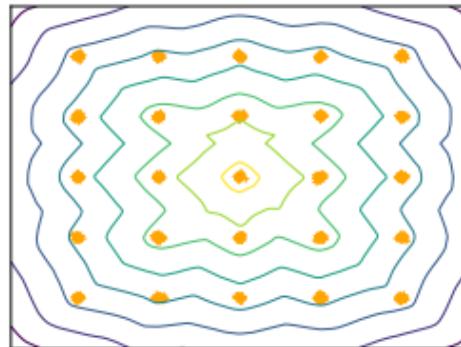
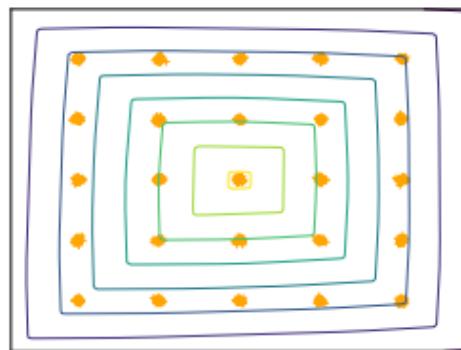


Drawback of weight clipping

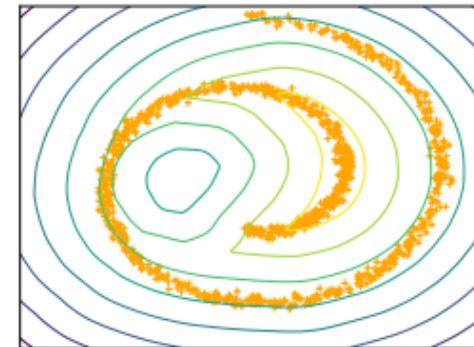
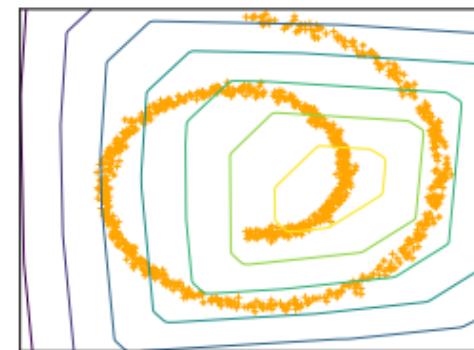
8 Gaussians



25 Gaussians



Swiss Roll

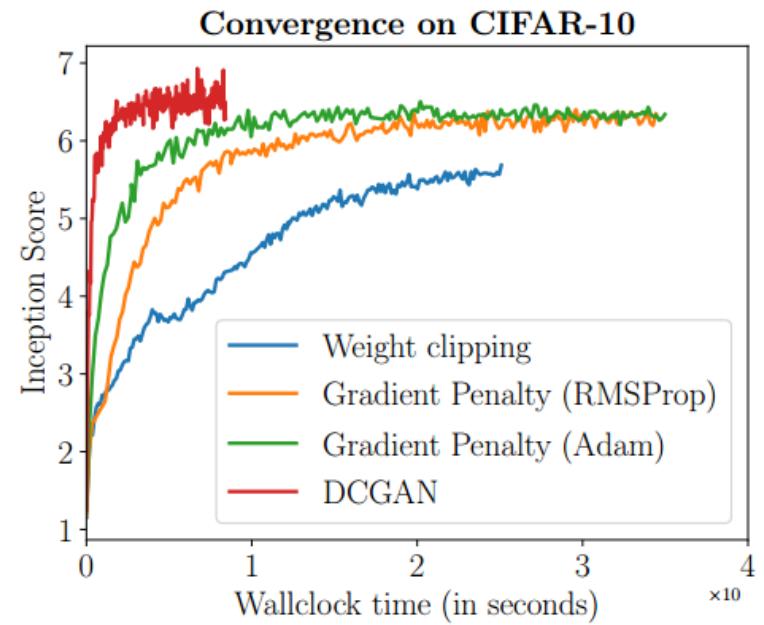
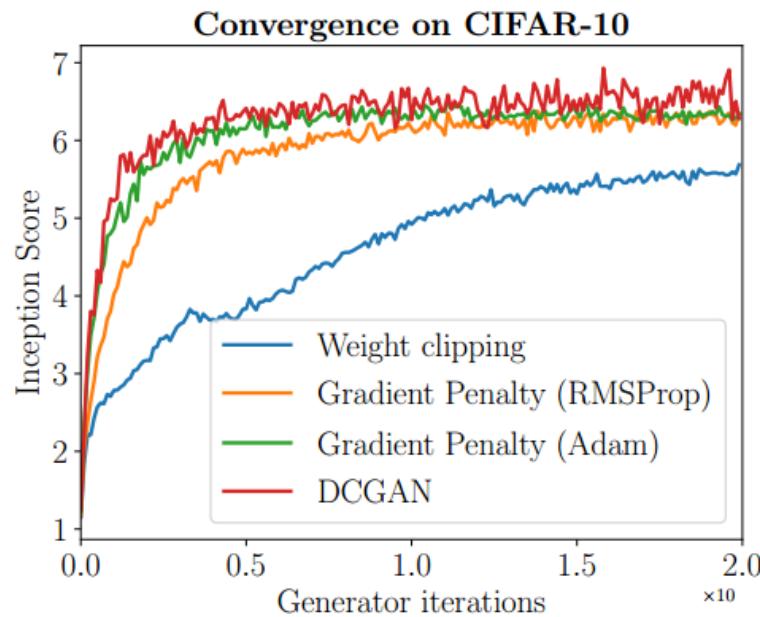


Improved WGAN

- D of WGAN
 - Has gradients with norm 1 almost everywhere under P_r and P_g
- The objective of improved WGAN

$$L = \underbrace{\mathbb{E}_{\tilde{x} \sim \mathbb{P}_g} [D(\tilde{x})] - \mathbb{E}_{x \sim \mathbb{P}_r} [D(x)]}_{\text{Original critic loss}} + \underbrace{\lambda \mathbb{E}_{\hat{x} \sim \mathbb{P}_{\hat{x}}} [(\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1)^2]}_{\text{Our gradient penalty}}.$$

Improved WGAN



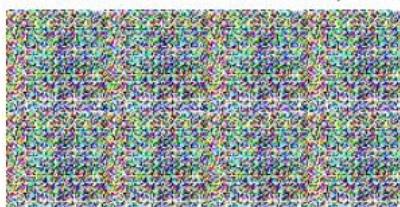
Improved WGAN

DCGAN**LSGAN****WGAN (clipping)****WGAN-GP (ours)**

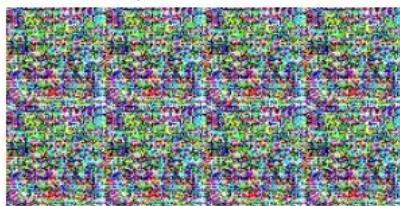
Baseline (G : DCGAN, D : DCGAN)



G : No BN and a constant number of filters, D : DCGAN



G : 4-layer 512-dim ReLU MLP, D : DCGAN



Improved WGAN

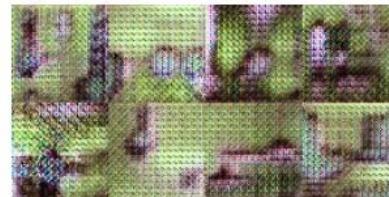
Gated multiplicative nonlinearities everywhere in G and D



$tanh$ nonlinearities everywhere in G and D



101-layer ResNet G and D



What's **Next?**

See **you** two weeks later