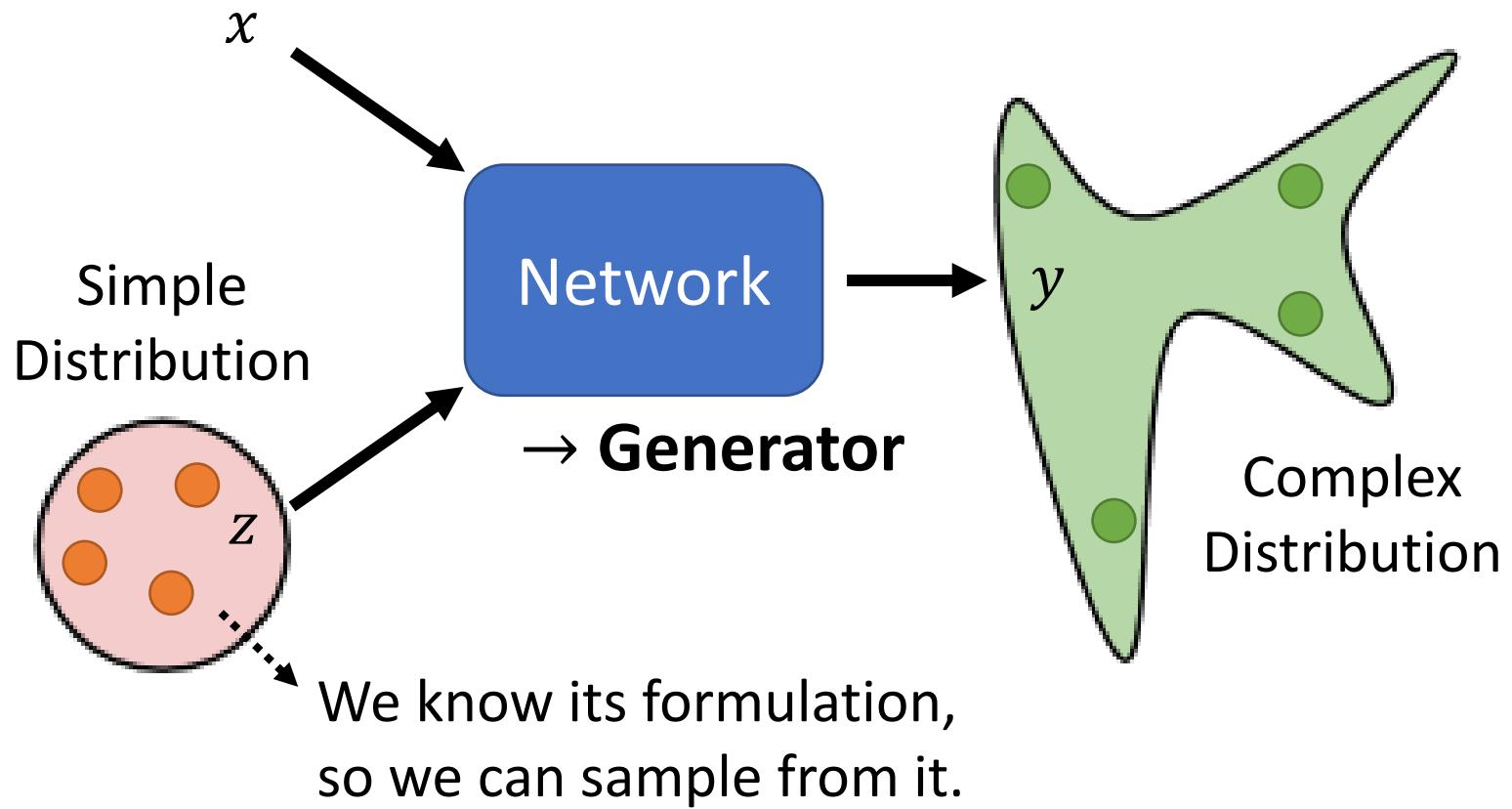


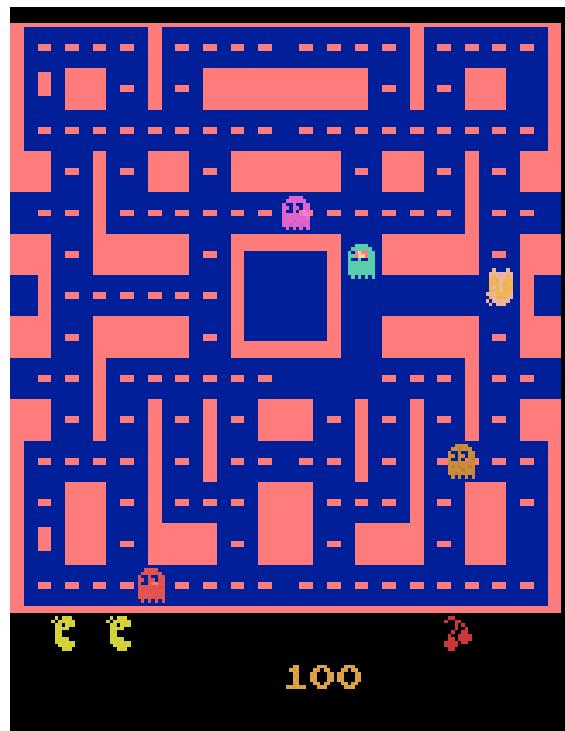
Generation

Hung-yi Lee 李宏毅

Network as Generator

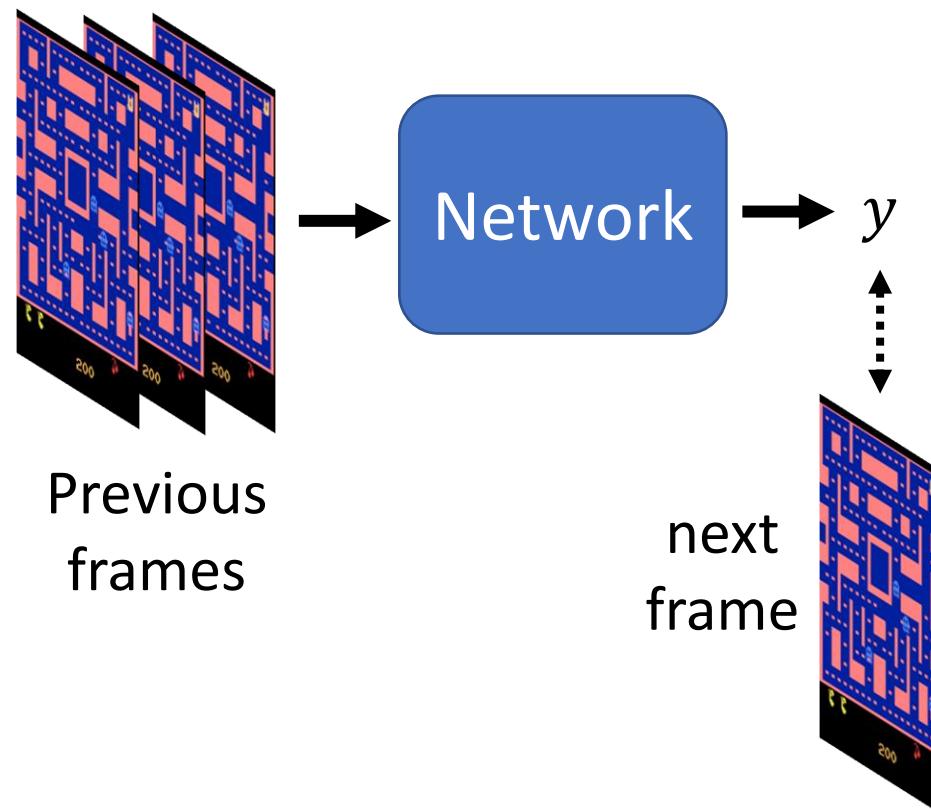


Why distribution?



Real Video

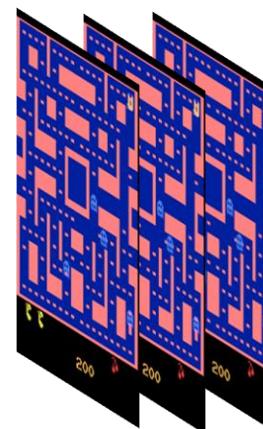
Video Prediction



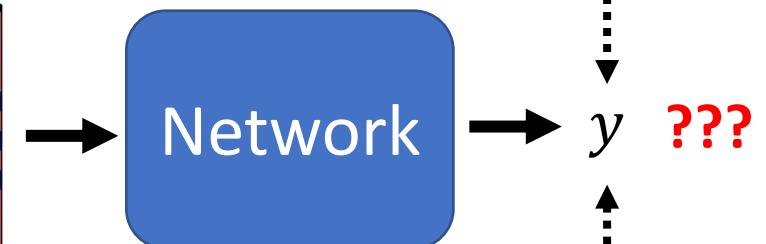
Why distribution?



Video Prediction



Previous
frames



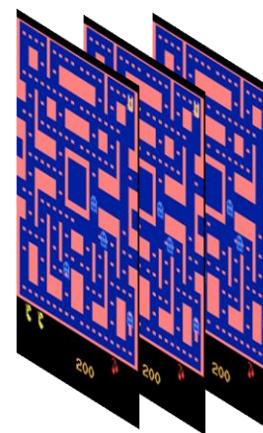
4

Why distribution?

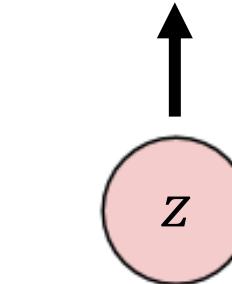
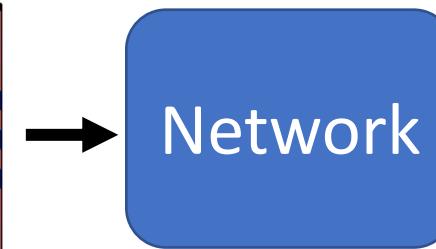


Prediction

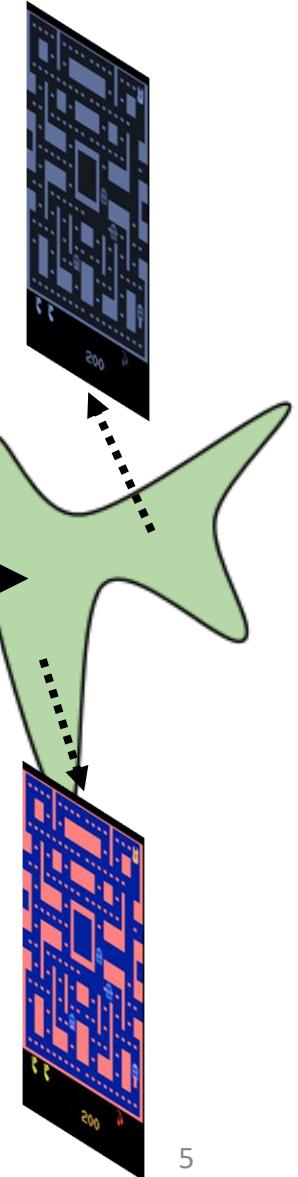
Video Prediction



Previous frames



Simple
Distribution



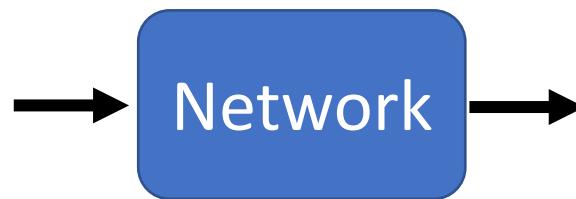
Why distribution?

(The same input has different outputs.)

- Especially for the tasks needs “*creativity*”

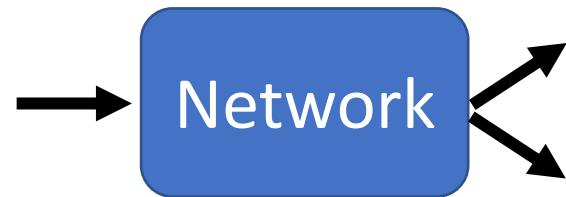
Drawing

Character
with red eyes



Chatbot

你知道輝夜是
誰嗎？



她是秀知院學生會 ...

她開創了忍者時代 ...

Generative Adversarial Network (GAN)

GAN

- How to pronounce “GAN”?



Google 小姐

All Kinds of GAN ...

<https://github.com/hindupuravinash/the-gan-zoo>

GAN

ACGAN

BGAN

CGAN

DCGAN

EBGAN

fGAN

GoGAN

⋮

- SeUDA - Semantic-Aware Generative Adversarial Nets for Unsupervised Domain Adaptation Segmentation
- SG-GAN - Semantic-aware Grad-GAN for Virtual-to-Real Urban Scene Adaption ([github](https://github.com/zhongyuan-zhou/SG-GAN))
- SG-GAN - Sparsely Grouped Multi-task Generative Adversarial Networks for Facial Attribut
- SGAN - Texture Synthesis with Spatial Generative Adversarial Networks
- SGAN - Stacked Generative Adversarial Networks ([github](https://github.com/taozi123456789/SGAN))
- SGAN - Steganographic Generative Adversarial Networks
- SGAN - SGAN: An Alternative Training of Generative Adversarial Networks
- SGAN - CT Image Enhancement Using Stacked Generative Adversarial Networks and Tissue Segmentation Improvement
- sGAN - Generative Adversarial Training for MRA Image Synthesis Using Multi-Contrast
- SiftingGAN - SiftingGAN: Generating and Sifting Labeled Samples to Improve the Remote Sensing Classification Baseline in vitro
- SiGAN - SiGAN: Siamese Generative Adversarial Network for Identity-Preserving Face H
- SimGAN - Learning from Simulated and Unsupervised Images through Adversarial Trai
- SisGAN - Semantic Image Synthesis via Adversarial Learning

Mihaela Rosca, Balaji Lakshminarayanan, David Warde-Farley, Shakir Mohamed, "Variational Approaches for Auto-Encoding Generative Adversarial Networks", arXiv, 2017

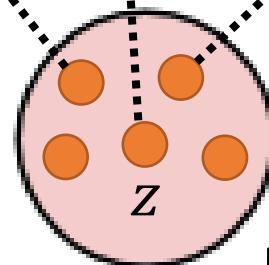
²We use the Greek α prefix for α -GAN, as AEGAN and most other Latin prefixes seem to have been taken <https://deephunt.in/the-gan-zoo-79597dc8c347>.

Anime Face Generation

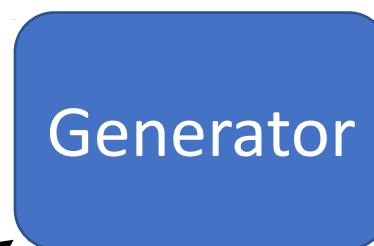
- Unconditional generation

**Low-dim
vector**

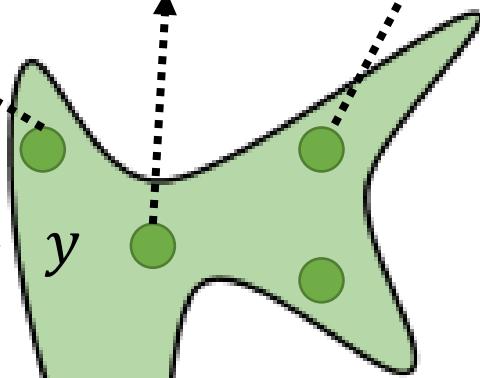
$$\begin{bmatrix} 0.3 \\ -0.1 \\ \vdots \\ -0.7 \end{bmatrix} \begin{bmatrix} 0.1 \\ -0.1 \\ \vdots \\ 0.7 \end{bmatrix} \begin{bmatrix} -0.3 \\ 0.1 \\ \vdots \\ 0.9 \end{bmatrix}$$



Normal
Distribution



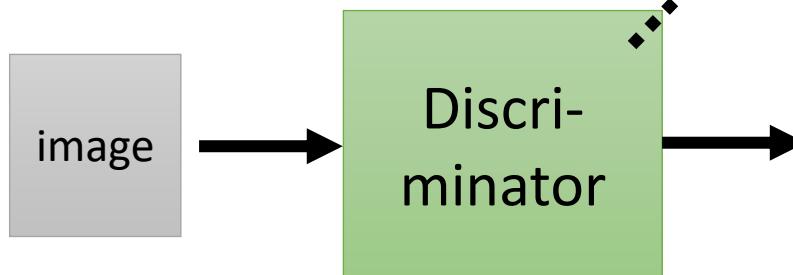
**high-dim
vector**



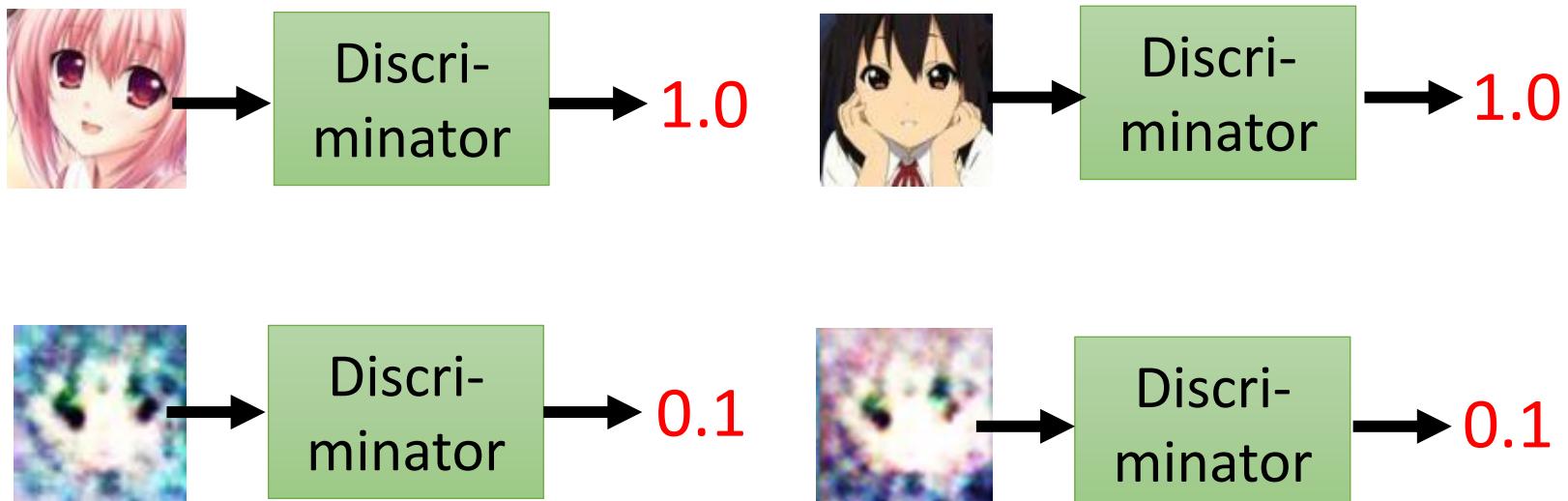
Complex
Distribution

Discriminator

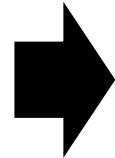
It is a neural network
(that is, a function).



Scalar: Larger means real,
smaller value fake.



Basic Idea of GAN



Brown



veins

Generator

Butterflies are
not brown



Butterflies do
not have veins



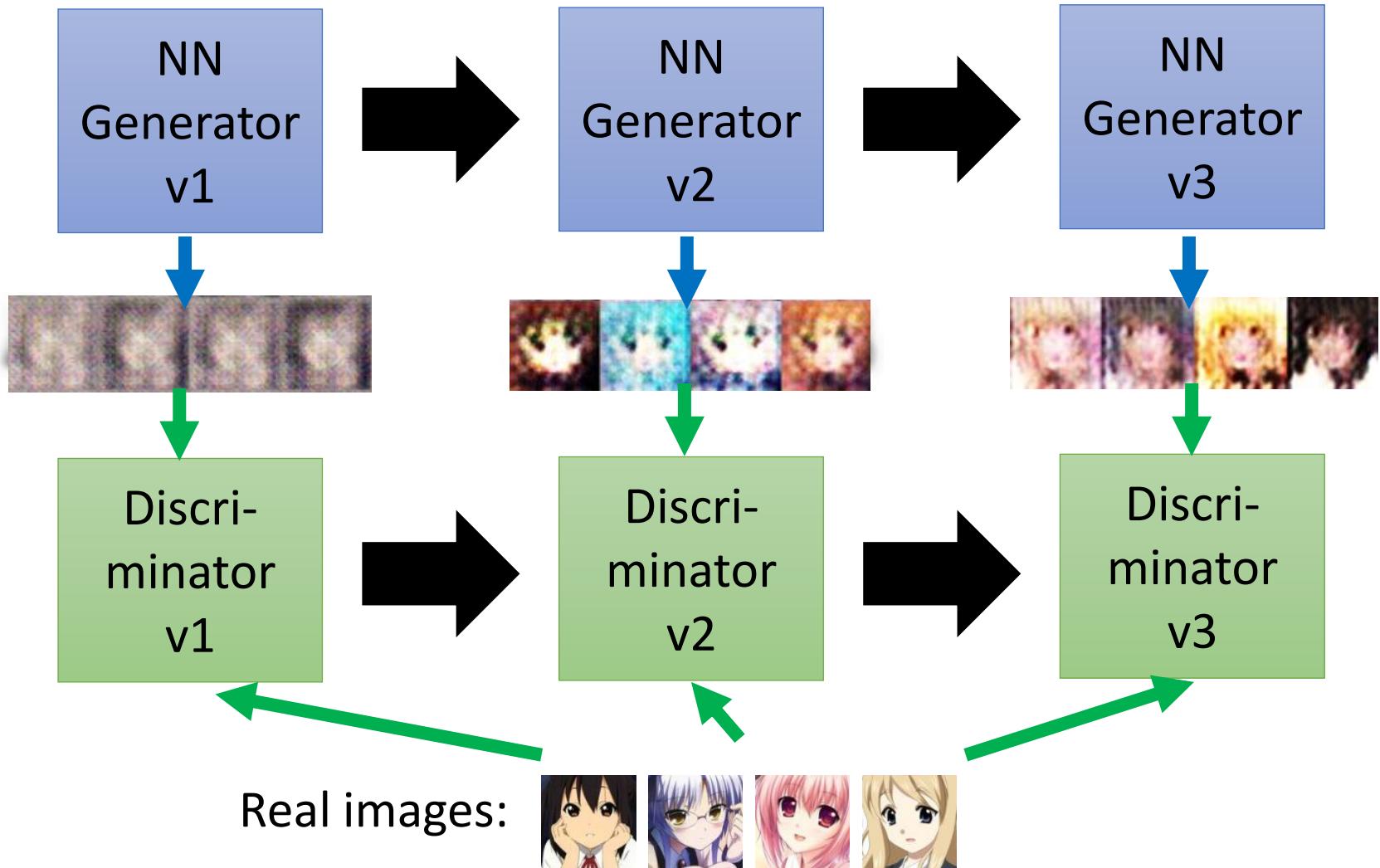
.....



Discriminator

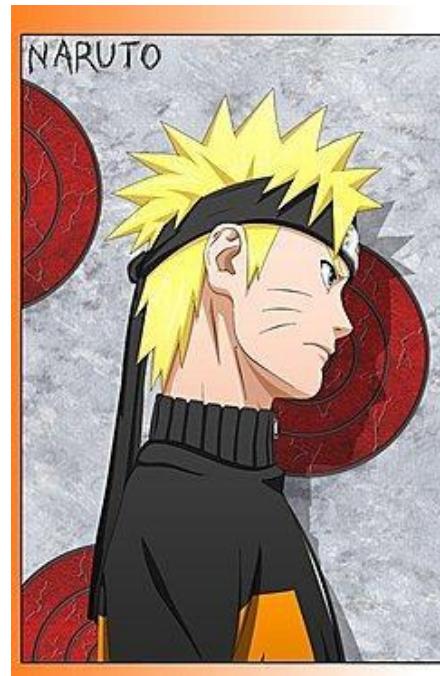
Basic Idea of GAN

This is where the term
“adversarial” comes from.



Basic Idea of GAN

- 寫作敵人，唸做朋友

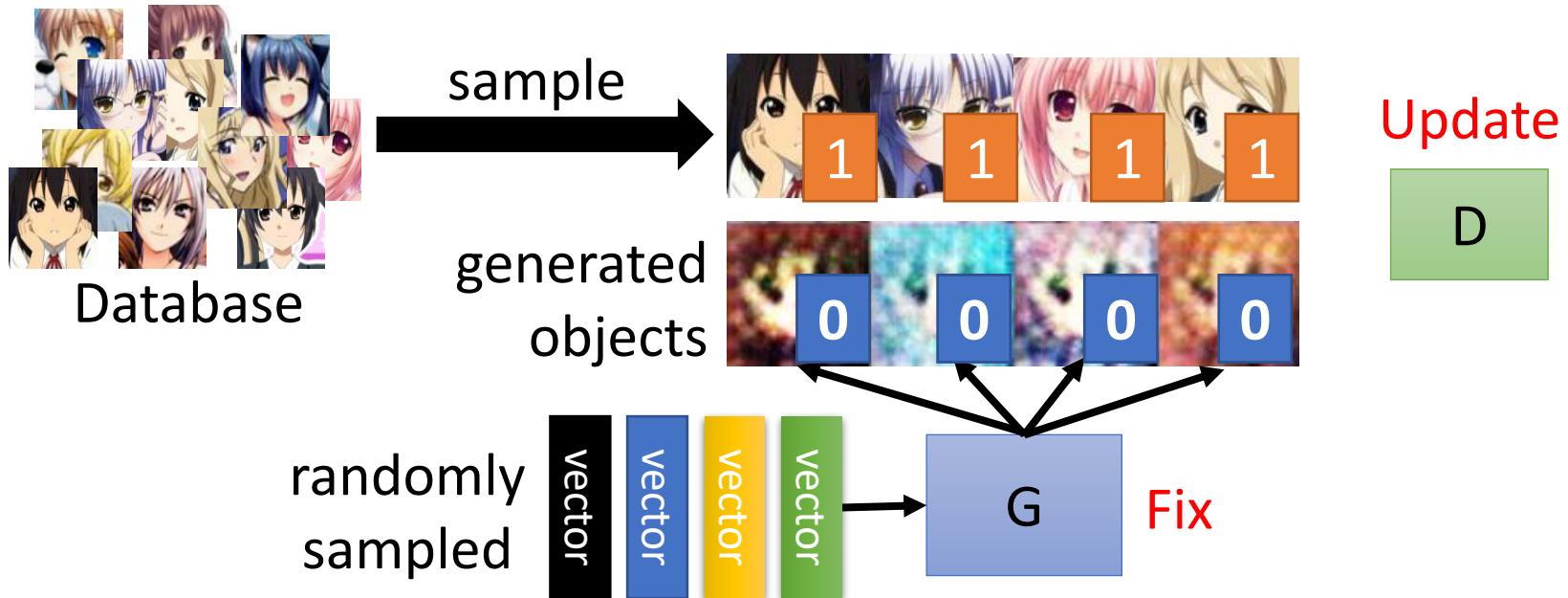


Algorithm

- Initialize generator and discriminator
- In each training iteration:



Step 1: Fix generator G, and update discriminator D



Discriminator learns to assign high scores to real objects and low scores to generated objects.

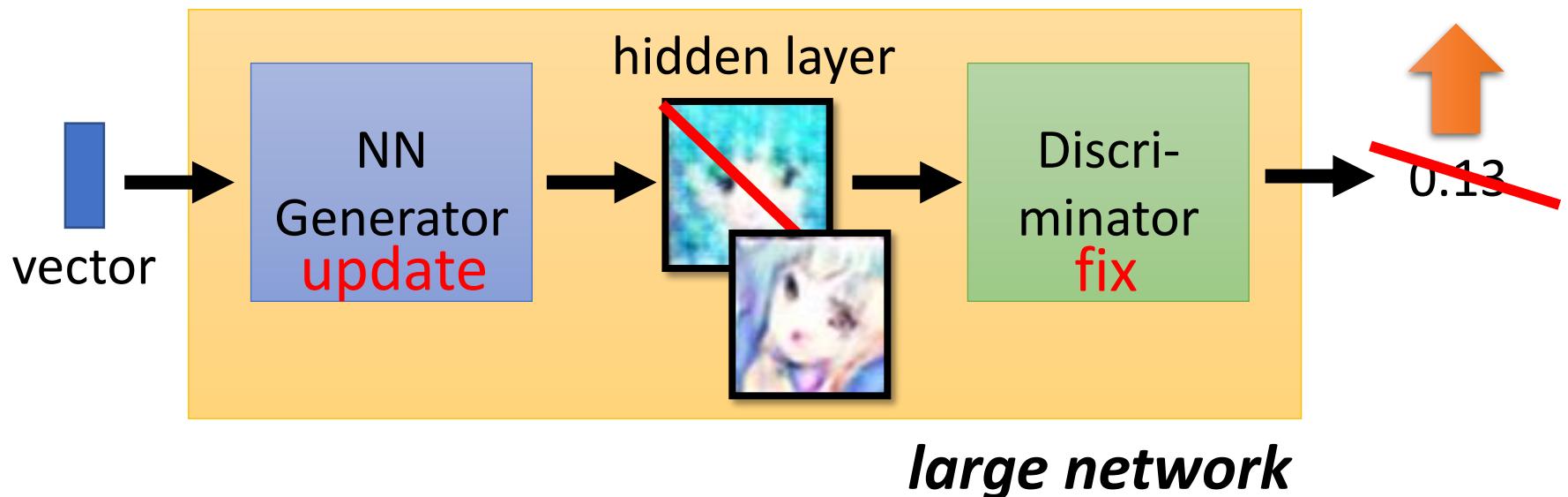
Algorithm

- Initialize generator and discriminator
- In each training iteration:



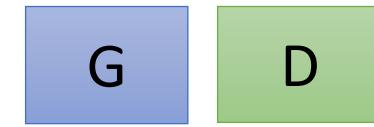
Step 2: Fix discriminator D, and update generator G

Generator learns to “fool” the discriminator



Algorithm

- Initialize generator and discriminator
- In each training iteration:



Learning
D

Sample some
real objects:



Update
D

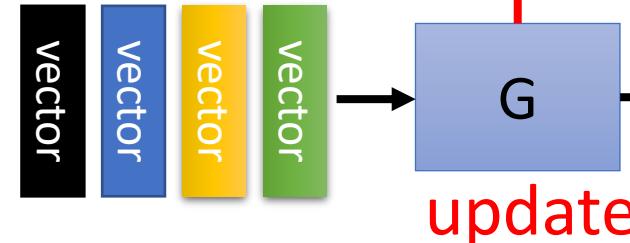
Generate some
fake objects:



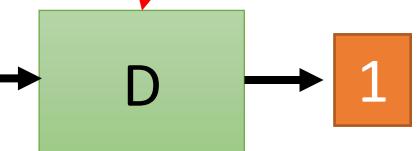
fix

D

Learning
G

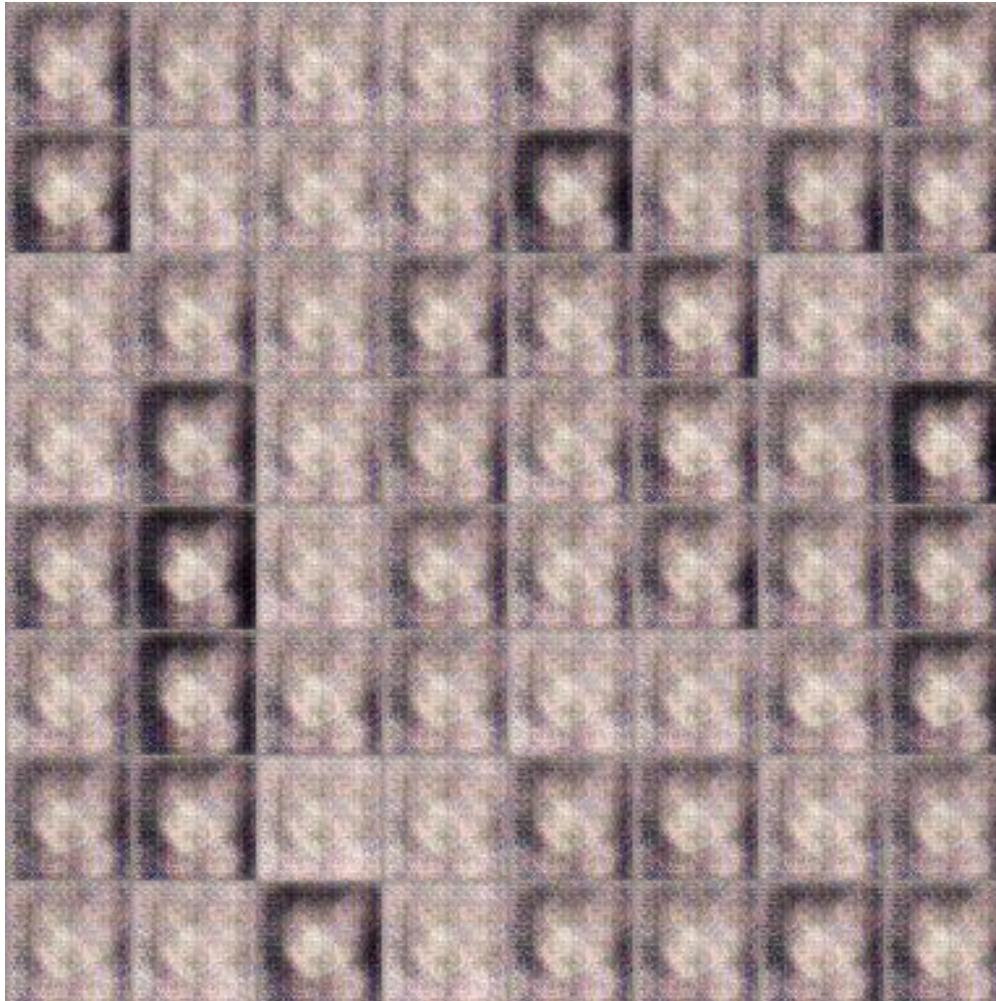


update



fix

Anime Face Generation



100 updates

Source of training data: <https://zhuanlan.zhihu.com/p/24767059>

Anime Face Generation



1000 updates

Anime Face Generation

2000 updates



Anime Face Generation

5000 updates



Anime Face Generation



10,000 updates

Anime Face Generation



20,000 updates

Anime Face Generation



50,000 updates



The faces
generated by
machine.

圖片生成：
吳宗翰、謝濬丞、
陳延昊、錢柏均

In 2019, with StyleGAN



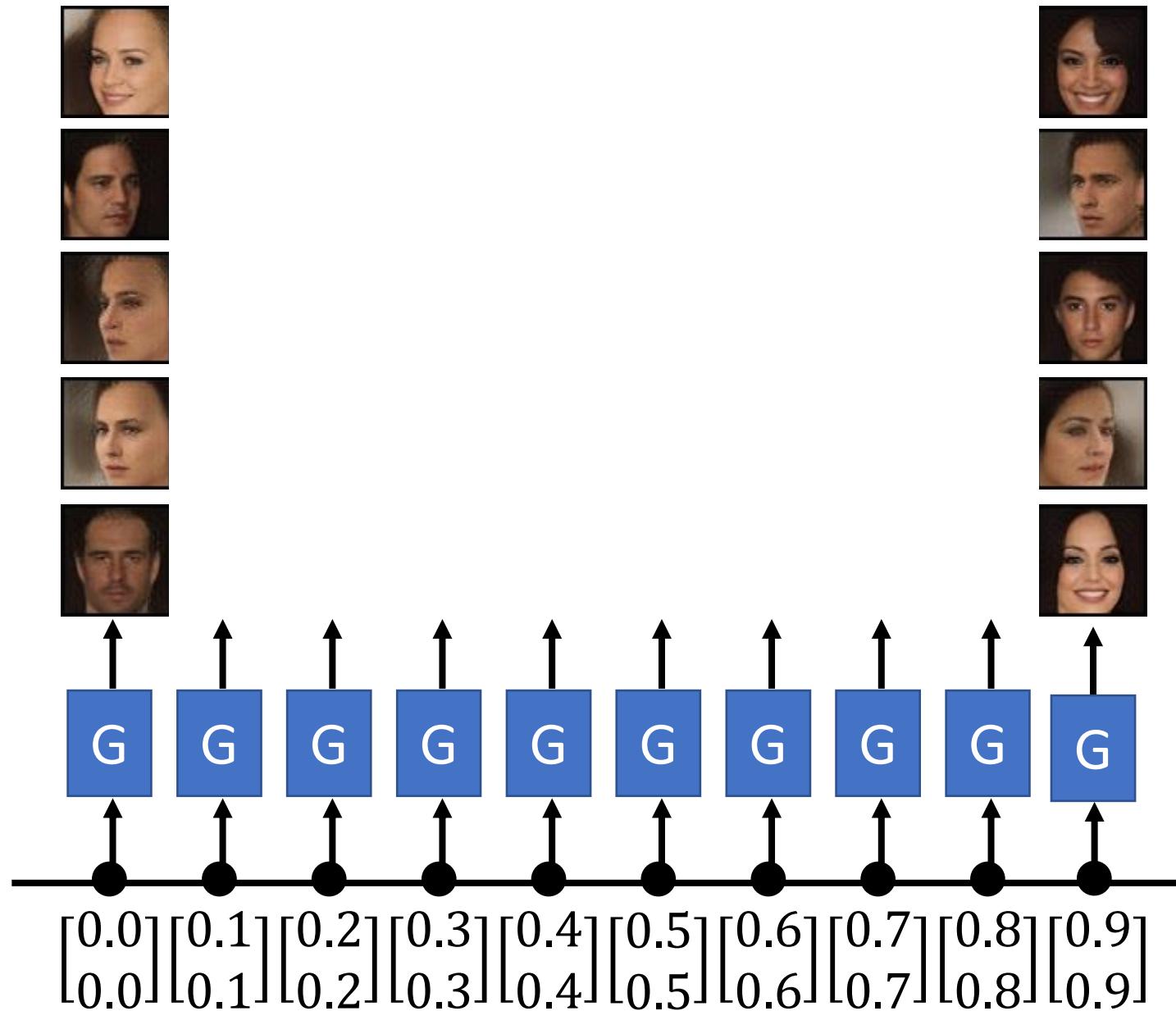
Source of video:

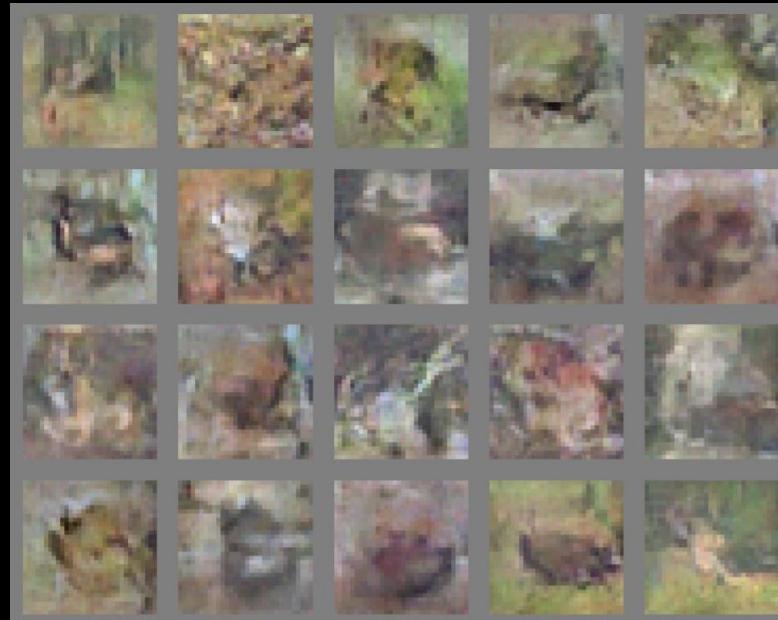
<https://www.gwern.net/Faces>



Progressive GAN |

<https://arxiv.org/abs/1710.10196>
27





The first GAN |

<https://arxiv.org/abs/1406.2661> (Ian J. Goodfellow)



Today BigGAN |

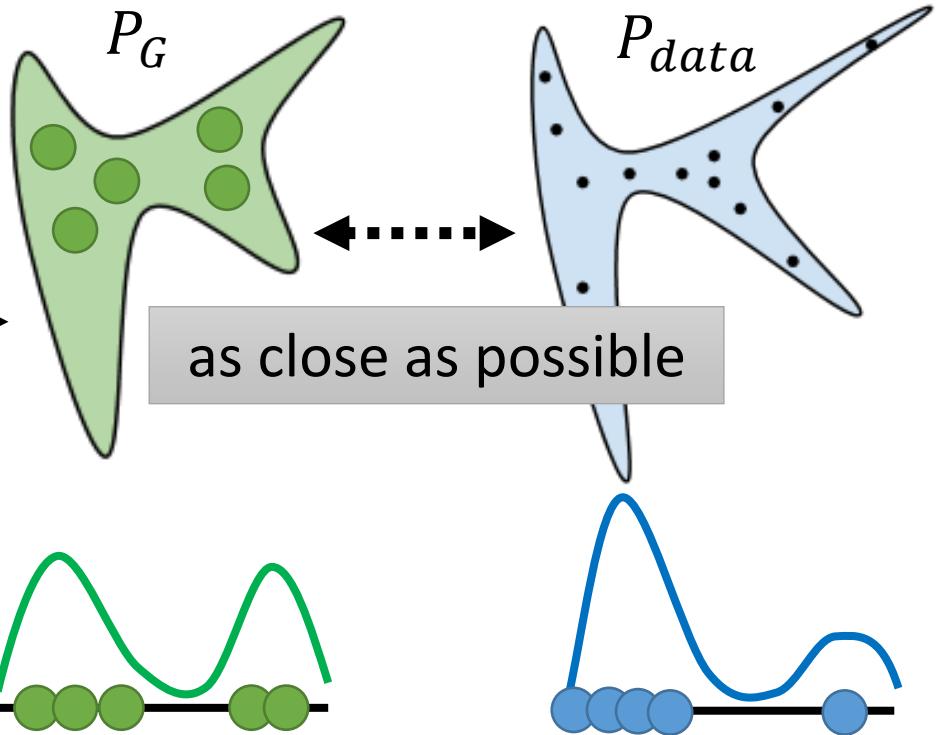
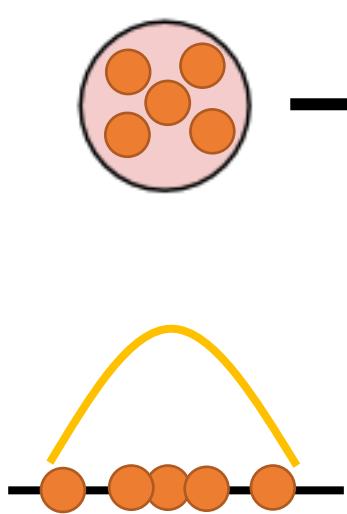
<https://arxiv.org/abs/1809.11096>

Theory behind GAN

c.f. $w^*, b^* = \arg \min_{w,b} L$

Our Objective

Normal
Distribution



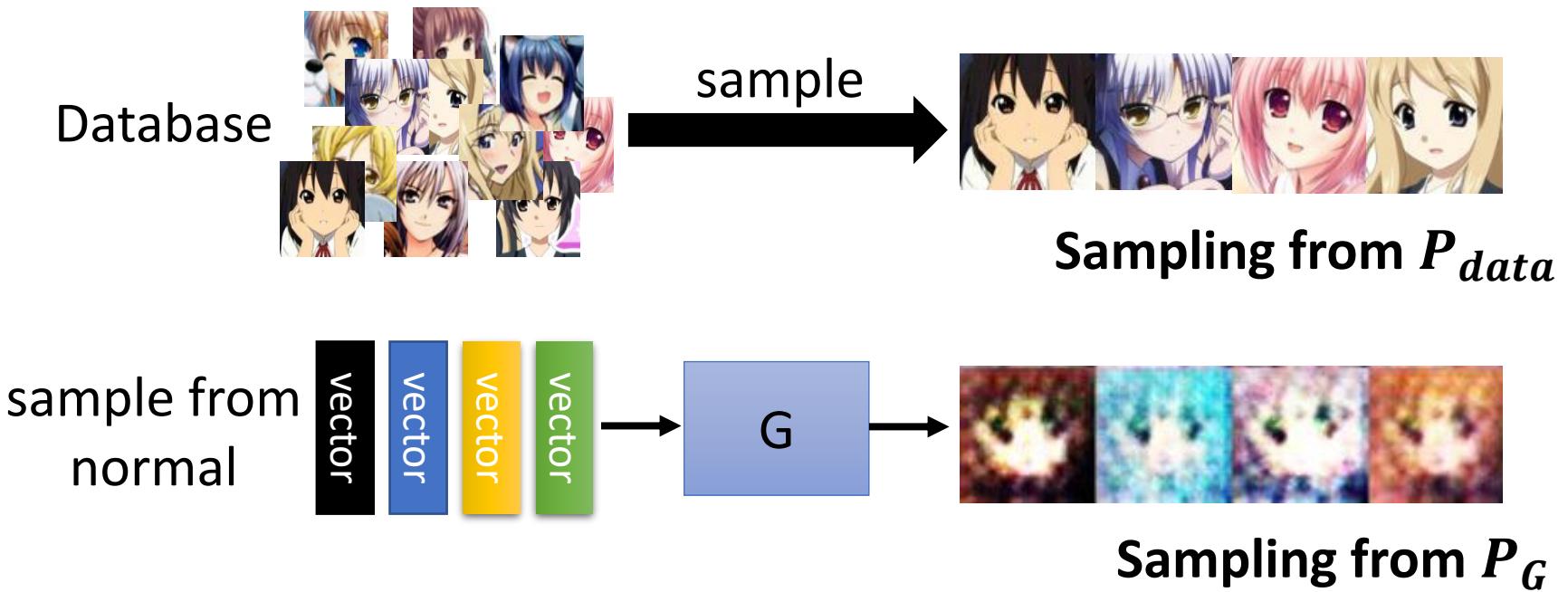
$$G^* = \arg \min_G \underline{Div}(P_G, P_{data})$$

Divergence between distributions P_G and P_{data}
How to compute the divergence?

Sampling is good enough

$$G^* = \arg \min_G \text{Div}(P_G, P_{\text{data}})$$

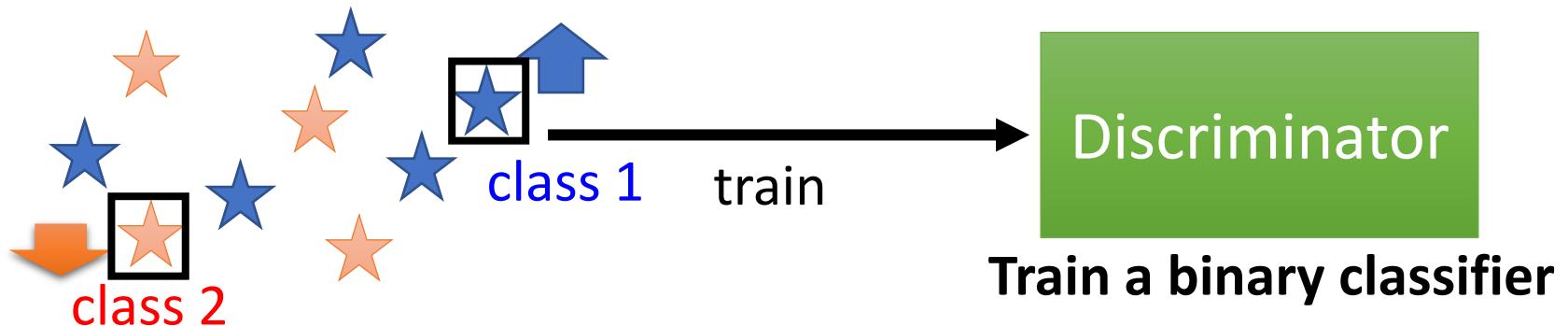
Although we do not know the distributions of P_G and P_{data} , we can sample from them.



Discriminator

$$G^* = \arg \min_G \text{Div}(P_G, P_{data})$$

★ : data sampled from P_{data} ★ : data sampled from P_G



Training: $D^* = \arg \max_D V(D, G)$

The value is related to JS divergence.

Objective Function for D

$$V(G, D) = E_{y \sim P_{data}} [\log D(y)] + E_{y \sim P_G} [\log(1 - D(y))]$$

$D^* = \arg \max_D V(D, G)$
negative cross entropy

= Training classifier:
minimize cross entropy

Discriminator

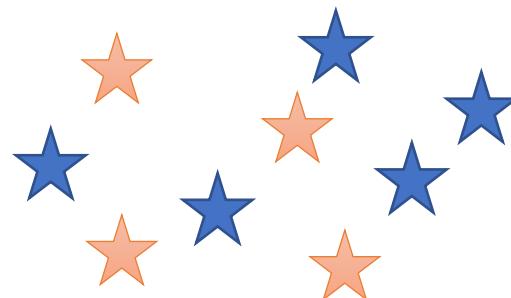
$$G^* = \arg \min_G \text{Div}(P_G, P_{data})$$

★ : data sampled from P_{data}

☆ : data sampled from P_G

Training:

$$D^* = \arg \max_D V(D, G)$$



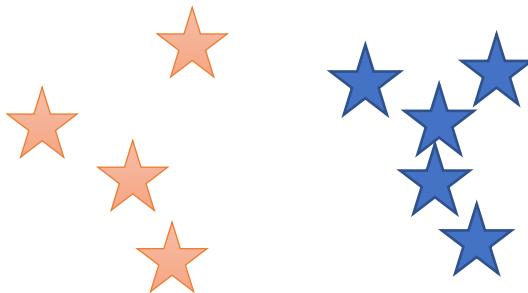
small divergence

train

Discriminator

hard to discriminate

$$\text{Small } \max_D V(D, G)$$



large divergence

train

Discriminator

easy to discriminate

$$G^* = \arg \min_G \max_D V(G, D)$$

$$D^* = \arg \max_D V(D, G)$$

The maximum objective value is related to JS divergence.

- Initialize generator and discriminator
- In each training iteration:

Step 1: Fix generator G, and update discriminator D

Step 2: Fix discriminator D, and update generator G

Can we use other divergence?

Name	$D_f(P\ Q)$	Generator $f(u)$
Total variation	$\frac{1}{2} \int p(x) - q(x) dx$	$\frac{1}{2} u - 1 $
Kullback-Leibler	$\int p(x) \log \frac{p(x)}{q(x)} dx$	$u \log u$
Reverse Kullback-Leibler	$\int q(x) \log \frac{q(x)}{p(x)} dx$	$-\log u$
Pearson χ^2	$\int \frac{(q(x)-p(x))^2}{p(x)} dx$	$(u - 1)^2$
Neyman χ^2	$\int \frac{(p(x)-q(x))^2}{q(x)} dx$	$\frac{(1-u)^2}{u}$
Squared Hellinger	$\int \left(\sqrt{p(x)} - \sqrt{q(x)} \right)^2 dx$	$(\sqrt{u} - 1)^2$
Jeffrey	$\int (p(x) - q(x)) \log \left(\frac{p(x)}{q(x)} \right) dx$	$(u - 1) \log u$
Jensen-Shannon	$\frac{1}{2} \int p(x) \log \frac{2p(x)}{p(x)+q(x)} + q(x) \log \frac{2q(x)}{p(x)+q(x)} dx$	$-(u + 1) \log \frac{1+u}{2} + u \log u$
Jensen-Shannon-weighted	$\int p(x)\pi \log \frac{p(x)}{\pi p(x)+(1-\pi)q(x)} + (1-\pi)q(x) \log \frac{q(x)}{\pi p(x)+(1-\pi)q(x)} dx$	$\pi u \log u - (1 - \pi + \pi u) \log(1 - \pi + \pi u)$
GAN	$\int p(x) \log \frac{2p(x)}{p(x)+q(x)} + q(x) \log \frac{2q(x)}{p(x)+q(x)} dx - \log(4)$	$u \log u - (u + 1) \log(u + 1)$

Name	Conjugate $f^*(t)$
Total variation	t
Kullback-Leibler (KL)	$\exp(t - 1)$
Reverse KL	$-1 - \log(-t)$
Pearson χ^2	$\frac{1}{4}t^2 + t$
Neyman χ^2	$2 - 2\sqrt{1-t}$
Squared Hellinger	$\frac{t}{1-t}$
Jeffrey	$W(e^{1-t}) + \frac{1}{W(e^{1-t})} + t - 2$
Jensen-Shannon	$-\log(2 - \exp(t))$
Jensen-Shannon-weighted	$(1 - \pi) \log \frac{1-\pi}{1-\pi e^{t/\pi}}$
GAN	$-\log(1 - \exp(t))$

Using the divergence
you like ☺

<https://arxiv.org/abs/1606.00709>

GAN is difficult to train

NO PAIN
NO GAN

(I found this joke from 陳柏文's facebook.)

Tips for GAN

JS divergence is not suitable

- In most cases, P_G and P_{data} are not overlapped.
- 1. The nature of data

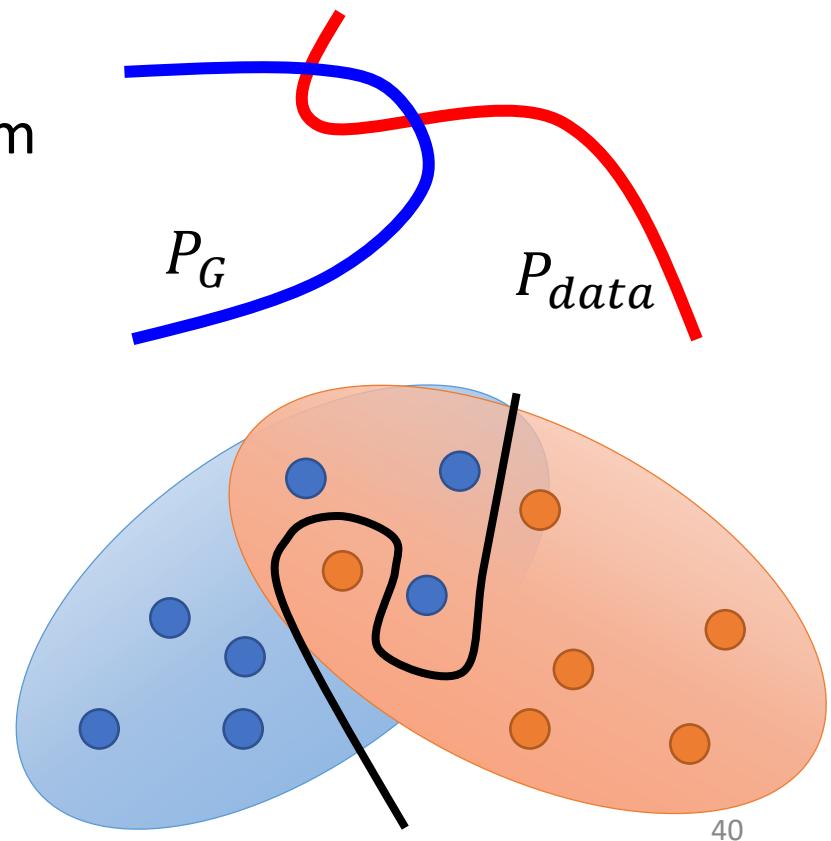
Both P_{data} and P_G are low-dim manifold in high-dim space.

The overlap can be ignored.

- 2. Sampling

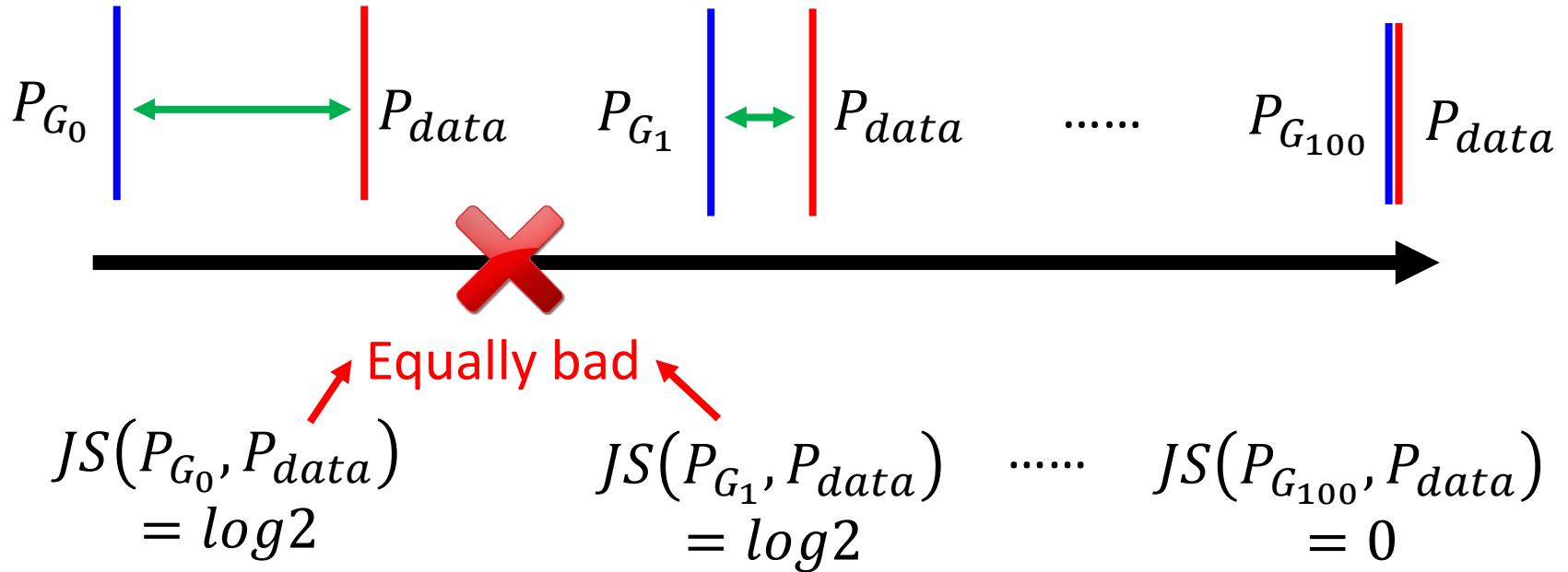
Even though P_{data} and P_G have overlap.

If you do not have enough sampling



What is the problem of JS divergence?

JS divergence is always $\log 2$ if two distributions do not overlap.

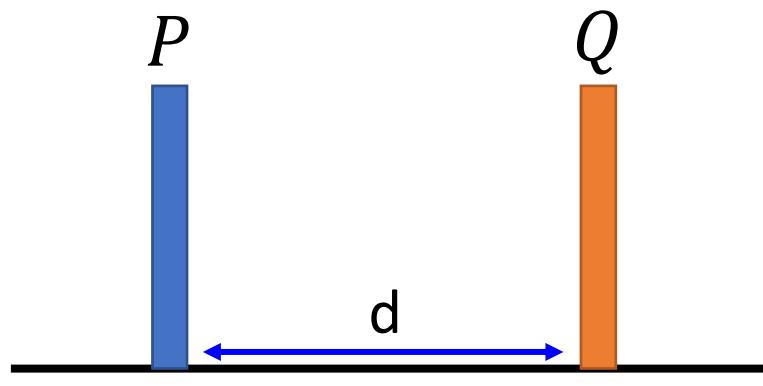


Intuition: If two distributions do not overlap, binary classifier achieves 100% accuracy.

The accuracy (or loss) means nothing during GAN training.

Wasserstein distance

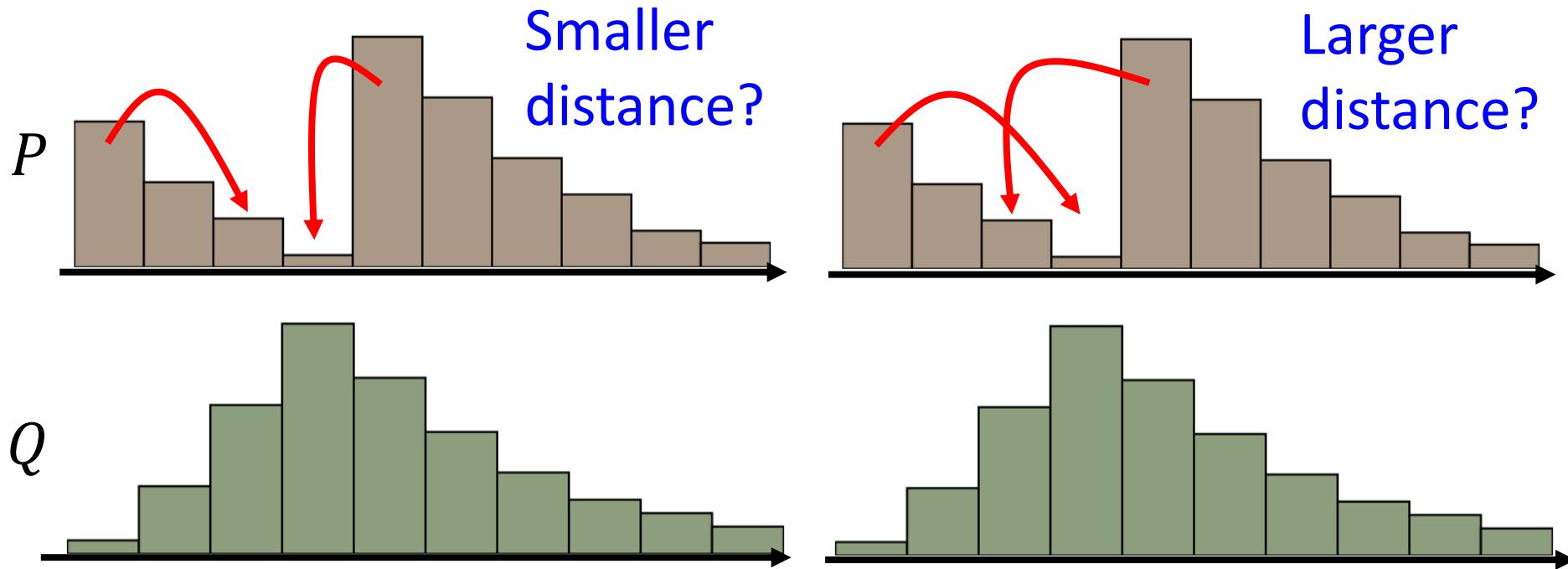
- Considering one distribution P as a pile of earth, and another distribution Q as the target
- The average distance the earth mover has to move the earth.



$$W(P, Q) = d$$



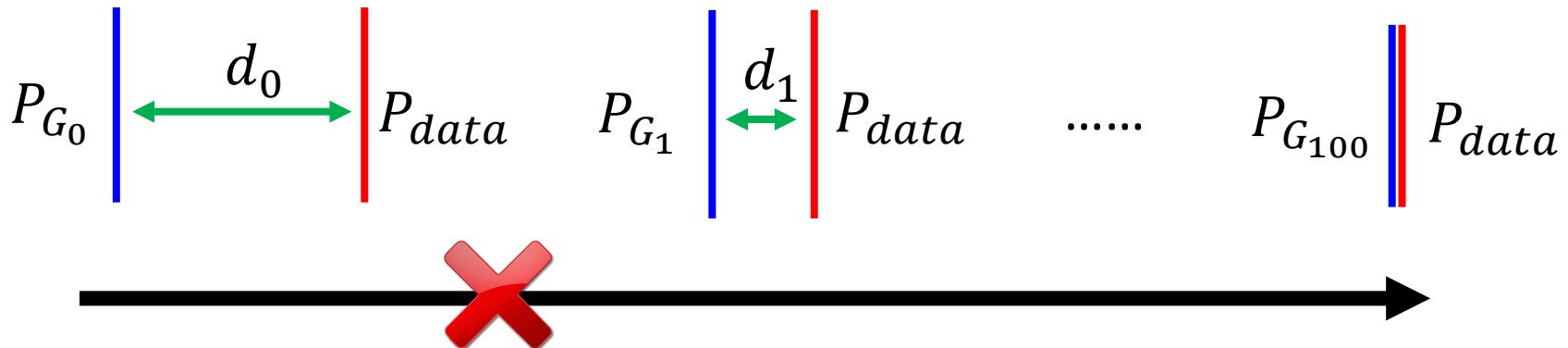
Wasserstein distance



There are many possible “moving plans”.

Using the “moving plan” with the smallest average distance to define the Wasserstein distance.

What is the problem of JS divergence?



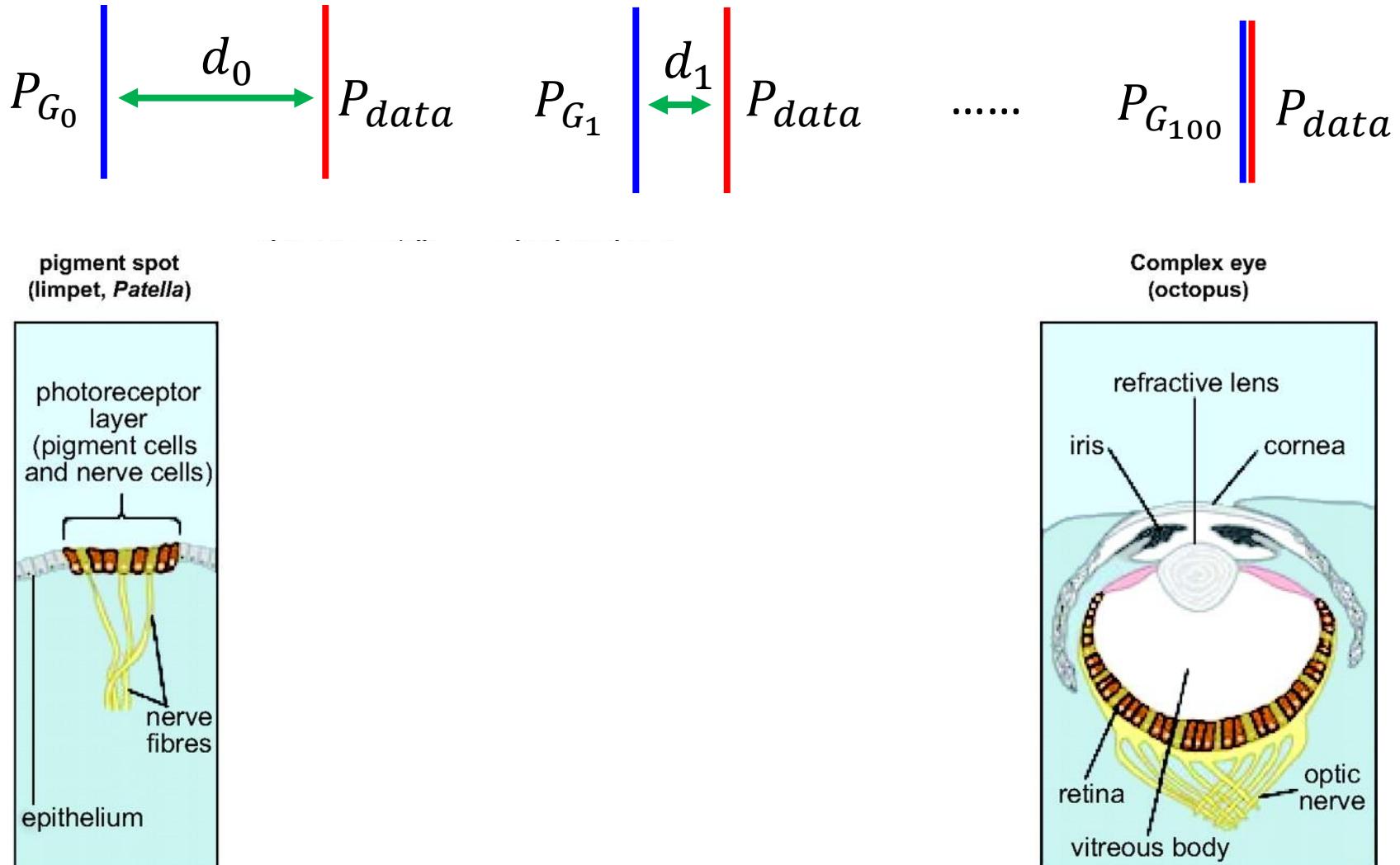
$$JS(P_{G_0}, P_{data}) = \log 2$$
$$JS(P_{G_1}, P_{data}) = \log 2$$
$$\dots$$
$$JS(P_{G_{100}}, P_{data}) = 0$$

$$W(P_{G_0}, P_{data}) = d_0$$
$$W(P_{G_1}, P_{data}) = d_1$$
$$\dots$$
$$W(P_{G_{100}}, P_{data}) = 0$$

Better!



What is the problem of JS divergence?



WGAN

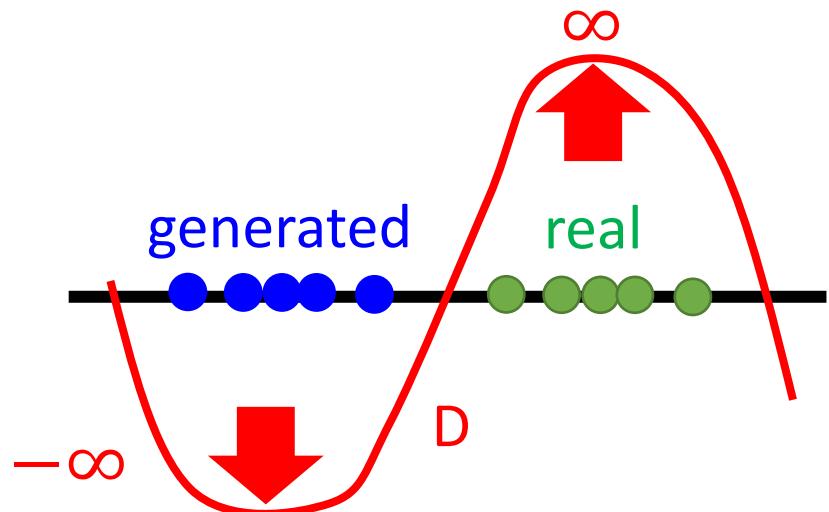
Evaluate Wasserstein distance between P_{data} and P_G

$$\max_{\substack{D \in 1-\text{Lipschitz}}} \left\{ E_{y \sim P_{data}}[D(y)] - E_{y \sim P_G}[D(y)] \right\}$$

D has to be smooth enough. How to fulfill this constraint?

Without the constraint, the training of D will not converge.

Keeping the D smooth forces $D(y)$ become ∞ and $-\infty$



$$\max_{D \in 1\text{-Lipschitz}} \{E_{y \sim P_{data}}[D(y)] - E_{y \sim P_G}[D(y)]\}$$

- Original WGAN → Weight

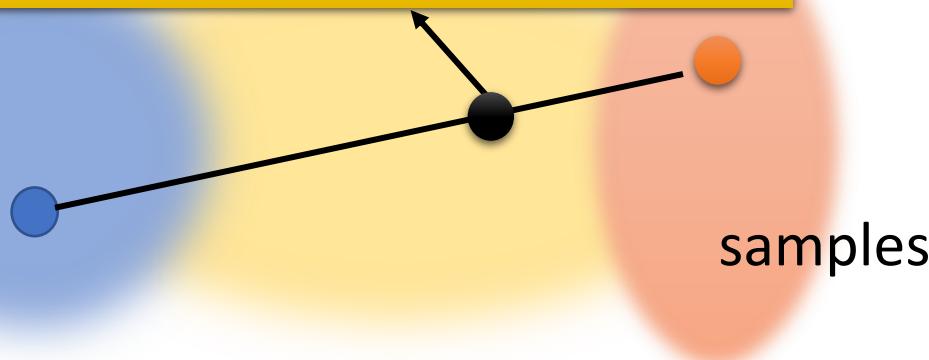
Force the parameters w between c and -c

After parameter update, if $w > c$, $w = c$; if $w < -c$, $w = -c$

- Improved WGAN → Gradient Penalty

Keep the gradient close to 1

<https://arxiv.org/abs/1704.00028>



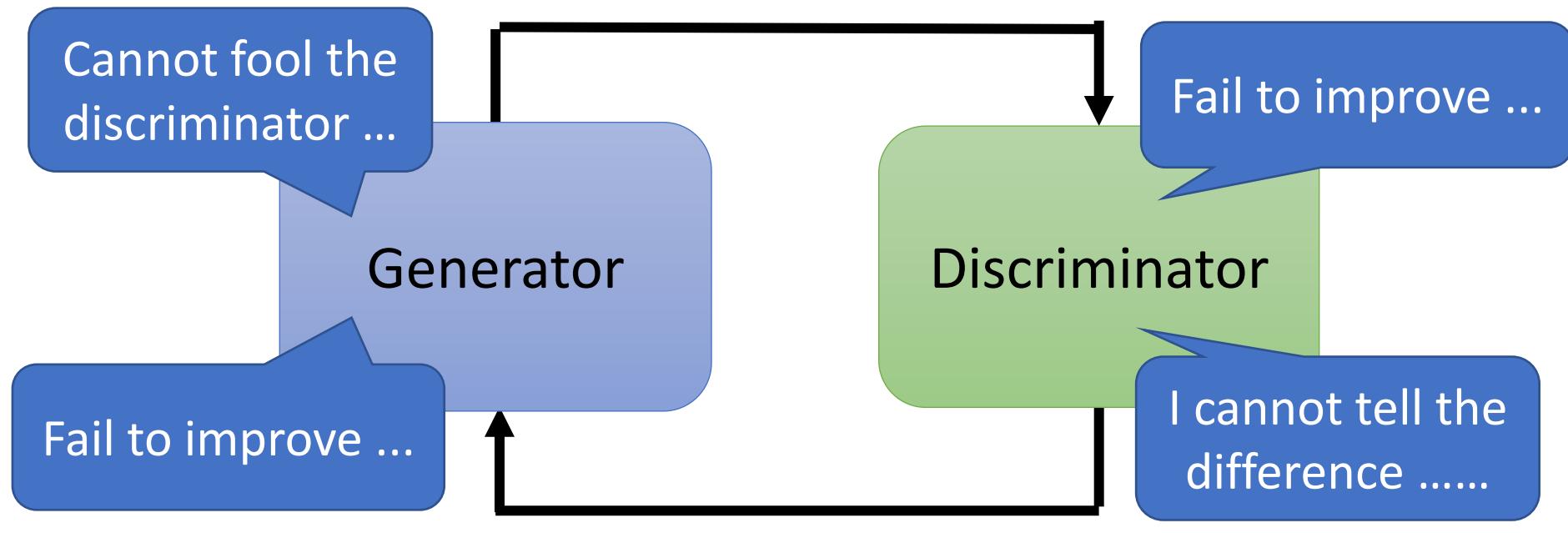
- Spectral Normalization → Keep gradient norm smaller than 1 everywhere

<https://arxiv.org/abs/1802.05957>

GAN is still challenging ...

- Generator and Discriminator needs to match each other (棋逢敵手)

Generate fake images to fool discriminator

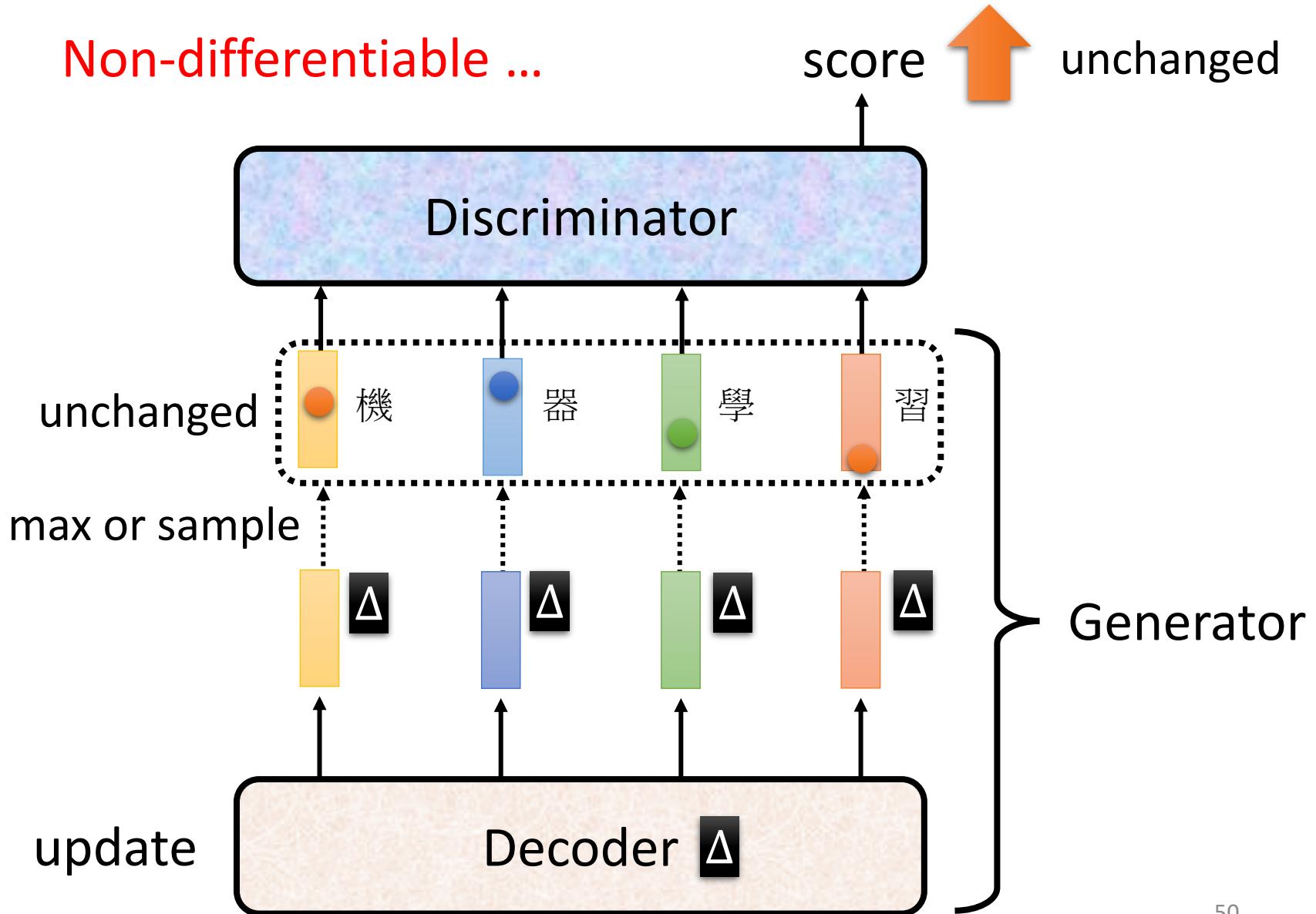


Tell the difference between real and fake

More Tips

- Tips from Soumith
 - <https://github.com/soumith/ganhacks>
- Tips in DCGAN: Guideline for network architecture design for image generation
 - <https://arxiv.org/abs/1511.06434>
- Improved techniques for training GANs
 - <https://arxiv.org/abs/1606.03498>
- Tips from BigGAN
 - <https://arxiv.org/abs/1809.11096>

GAN for Sequence Generation

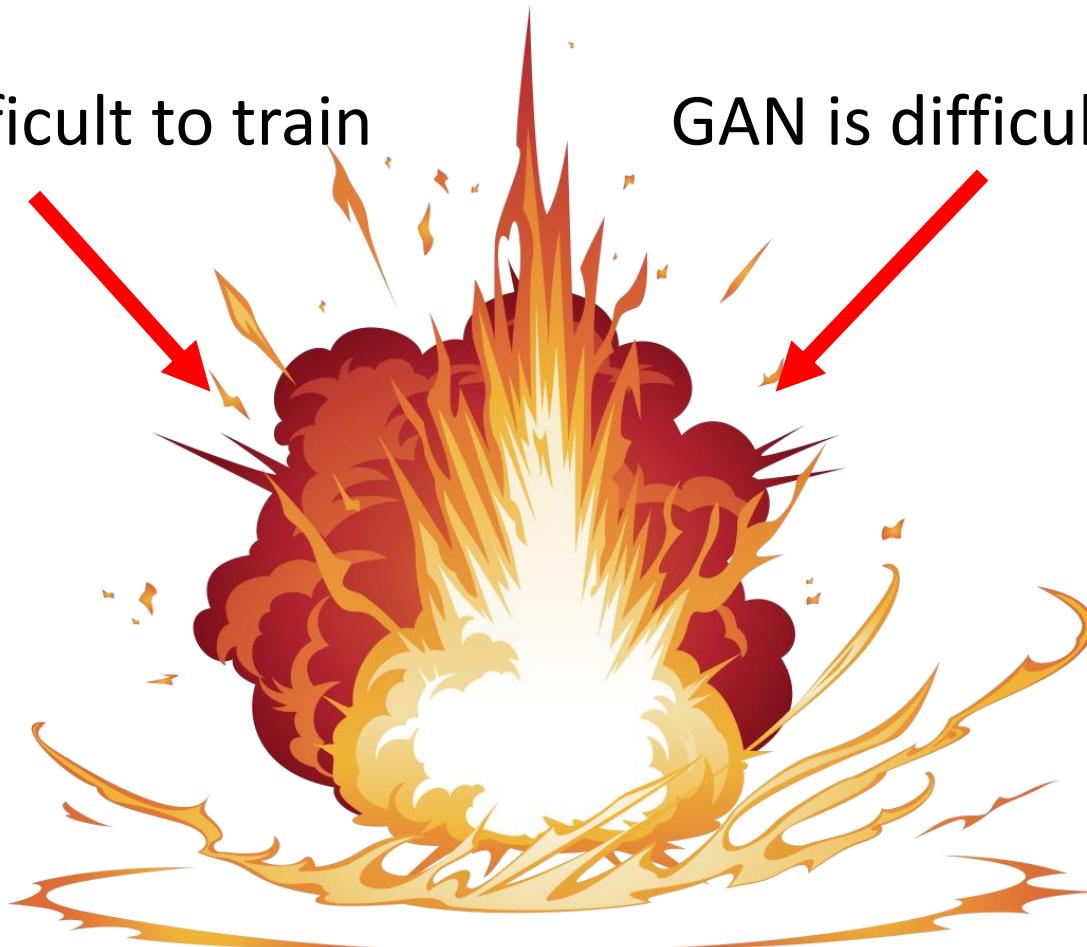


GAN for Sequence Generation

Reinforcement learning (RL) is involved

'RL is difficult to train

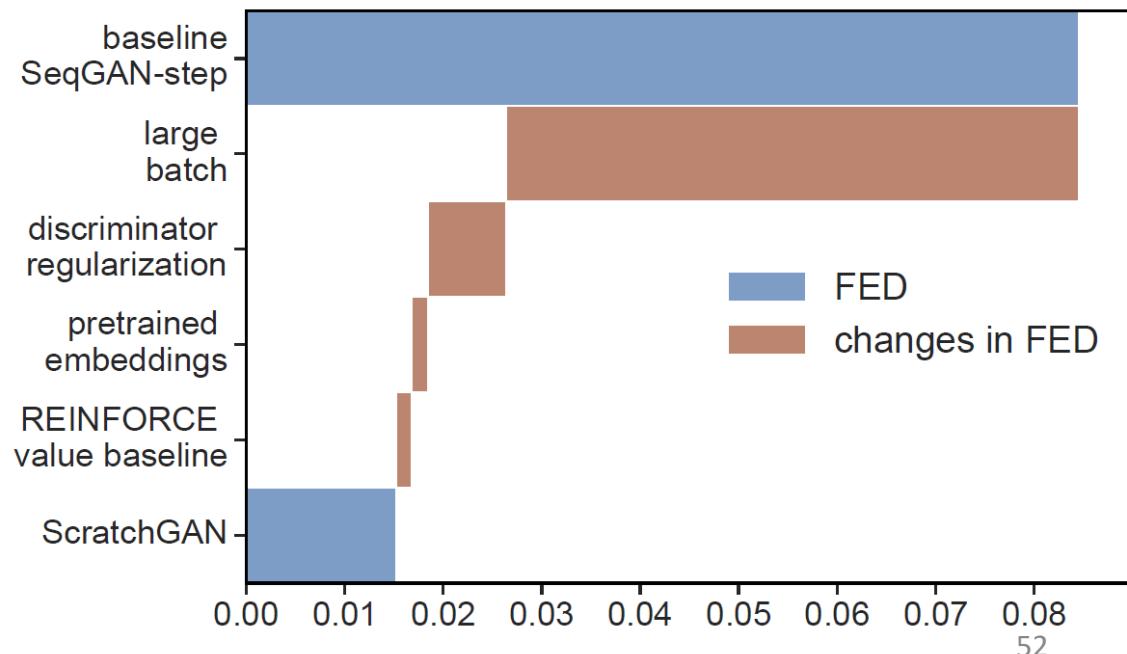
GAN is difficult to train



Sequence Generation GAN (RL+GAN)

GAN for Sequence Generation

- Usually, the generator are fine-tuned from a model learned by other approaches.
- However, with enough hyperparameter-tuning and tips, ScarchGAN can train from scratch.



Training language
GANs from Scratch

[https://arxiv.org/abs/
1905.09922](https://arxiv.org/abs/1905.09922)

Generative Models

- This lecture: Generative Adversarial Network (GAN)



Full version

https://www.youtube.com/playlist?list=PLJV_el3uVTsMq6JEFPW35BCiOQTsoqwNw

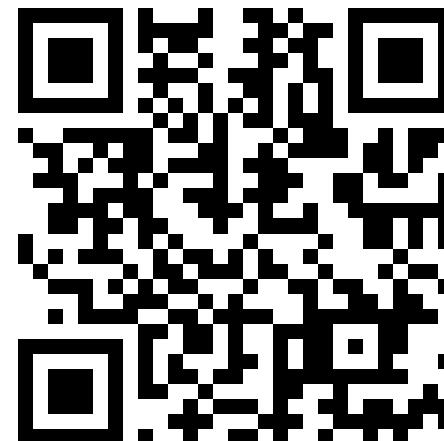
More Generative Models

Variational
Autoencoder (VAE)



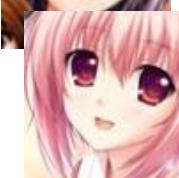
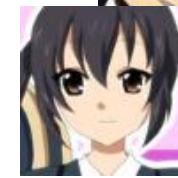
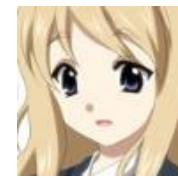
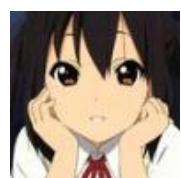
<https://youtu.be/8zomhgKrsmQ>

FLOW-based
Model



<https://youtu.be/uXY18nzdSsM>

Possible Solution?



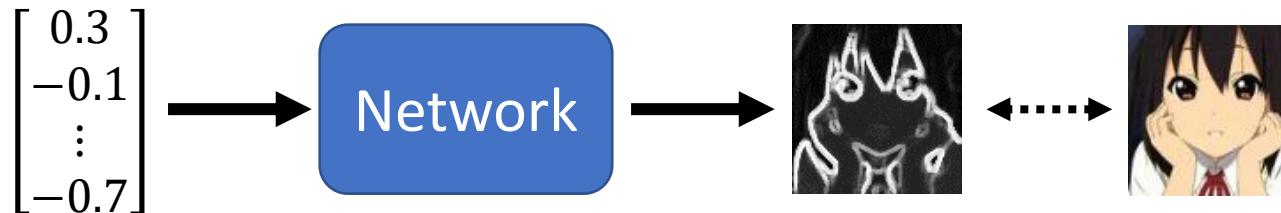
$$\begin{bmatrix} 0.3 \\ -0.1 \\ \vdots \\ -0.7 \end{bmatrix}$$

$$\begin{bmatrix} 0.1 \\ -0.1 \\ \vdots \\ 0.7 \end{bmatrix}$$

$$\begin{bmatrix} -0.3 \\ 0.1 \\ \vdots \\ 0.9 \end{bmatrix}$$

$$\begin{bmatrix} 0.7 \\ 0.1 \\ \vdots \\ -0.9 \end{bmatrix}$$

$$\begin{bmatrix} -0.1 \\ 0.8 \\ \vdots \\ 0.8 \end{bmatrix}$$

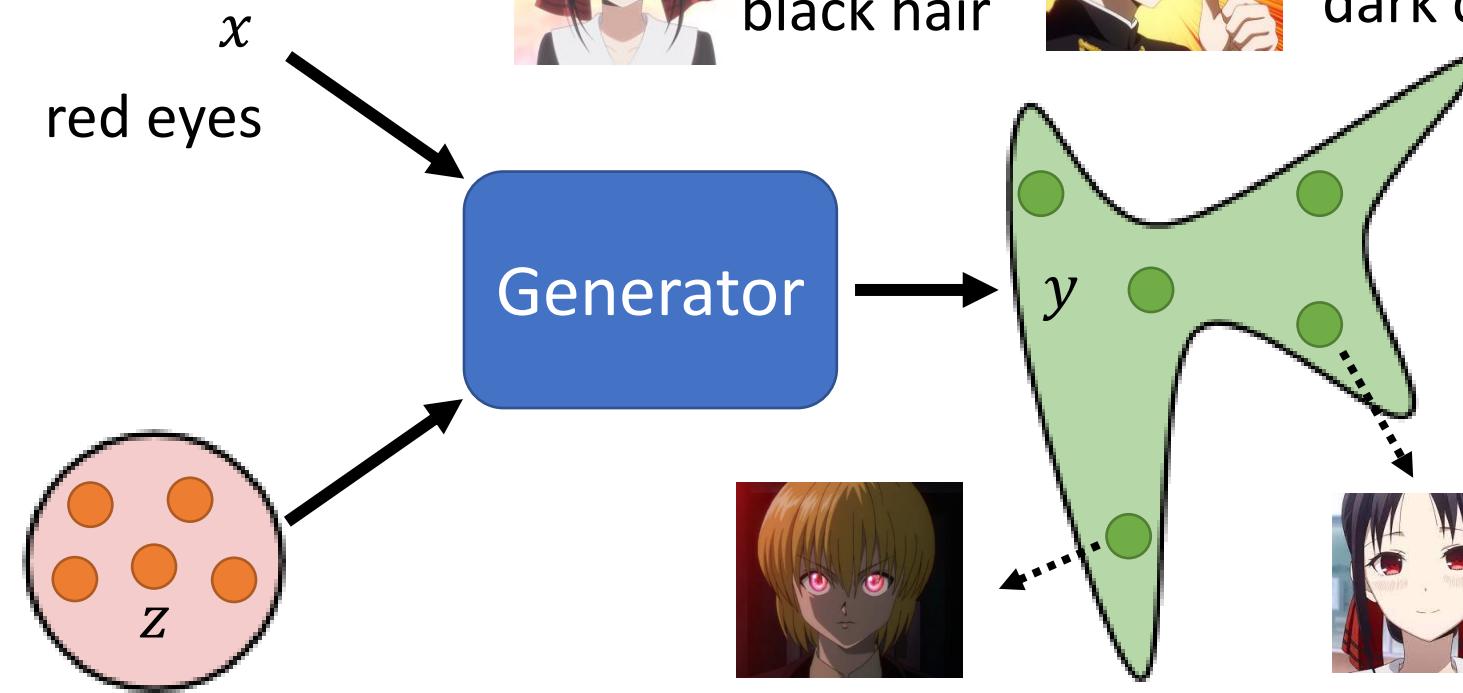


Using typical learning approaches?

Generative Latent Optimization (GLO), <https://arxiv.org/abs/1707.05776>
Gradient Origin Networks, <https://arxiv.org/abs/2007.02798>

Conditional Generation

Text-to-image



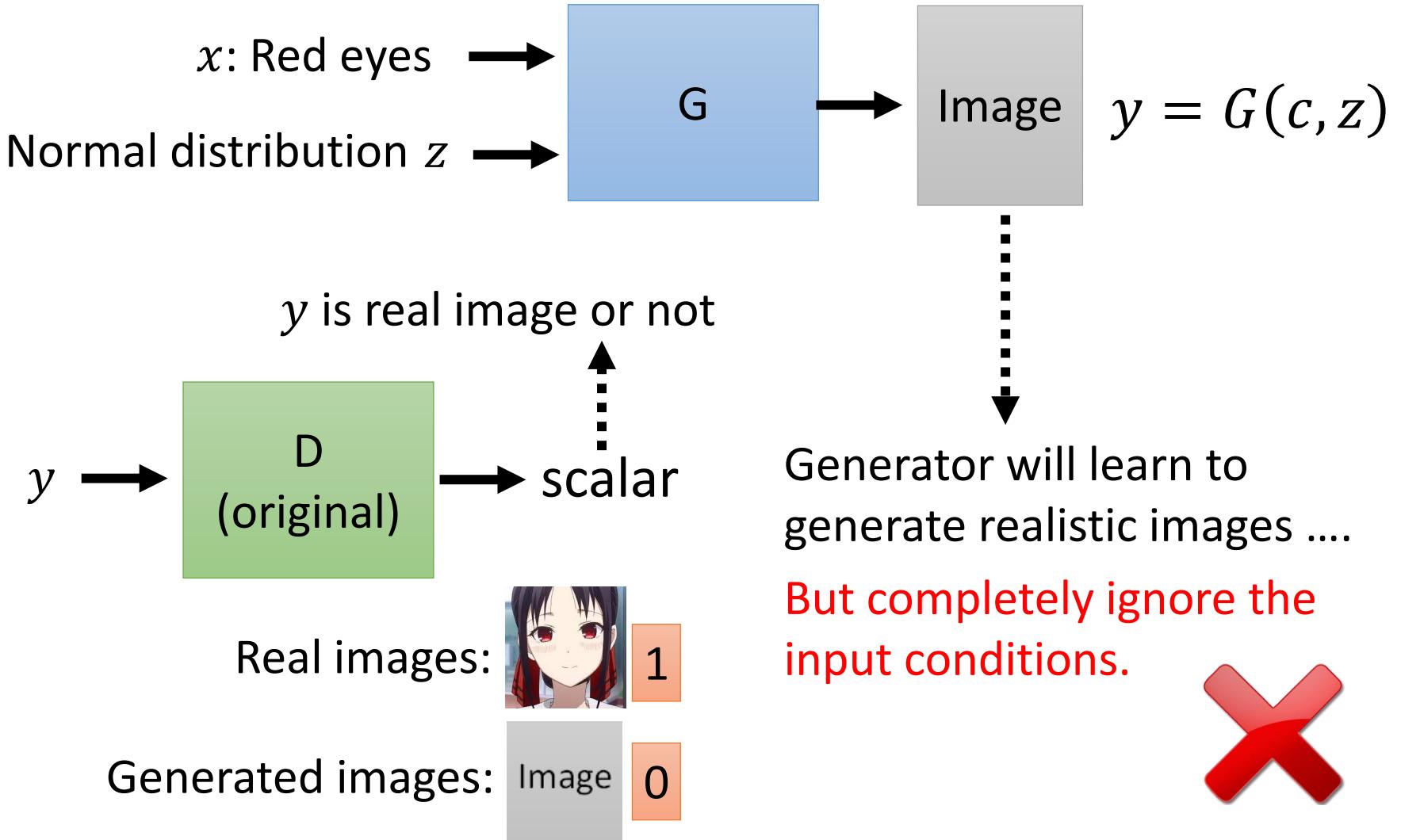
red hair,
green eyes



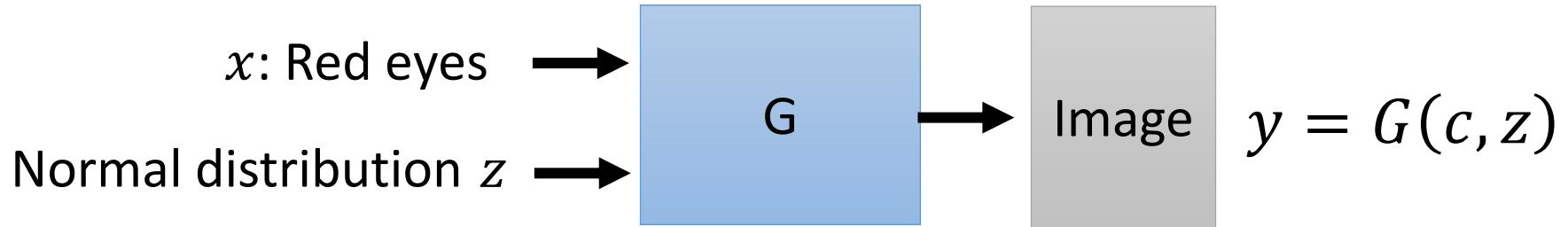
blue hair,
red eyes



Conditional GAN



Conditional GAN



True text-image pairs: (red eyes,



) 1

(red eyes,)

0

(red eyes,) 0

Conditional GAN

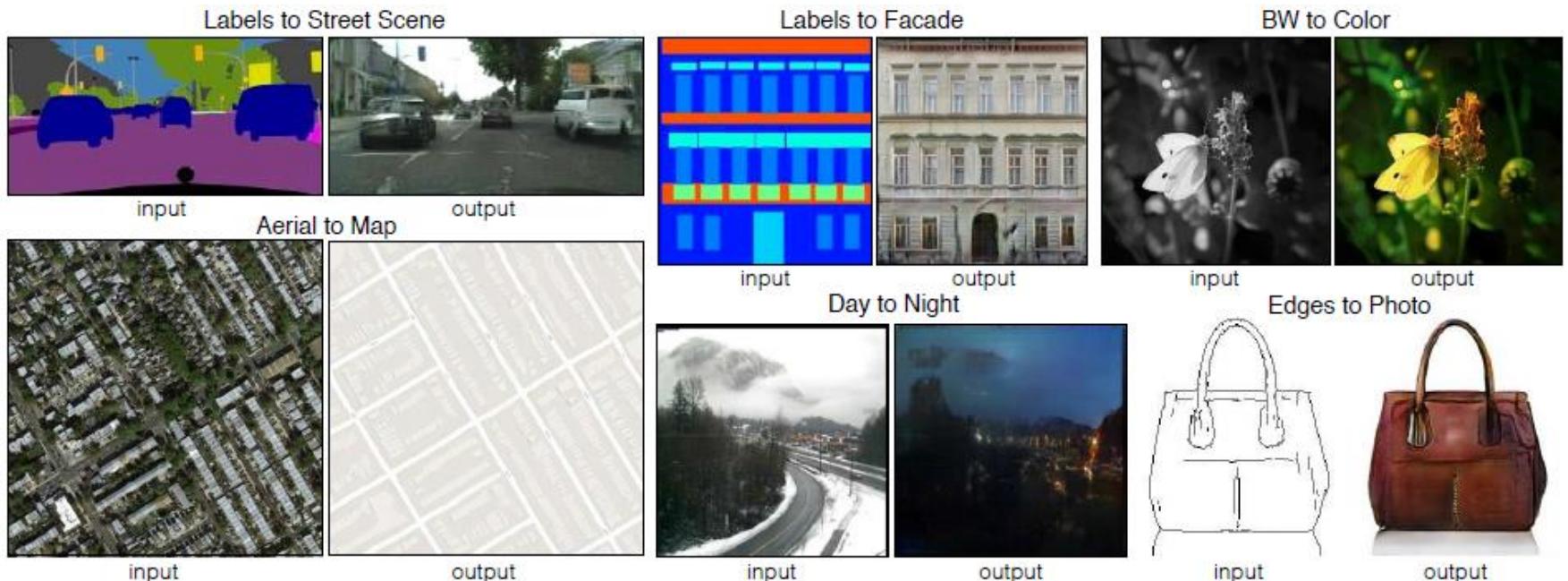
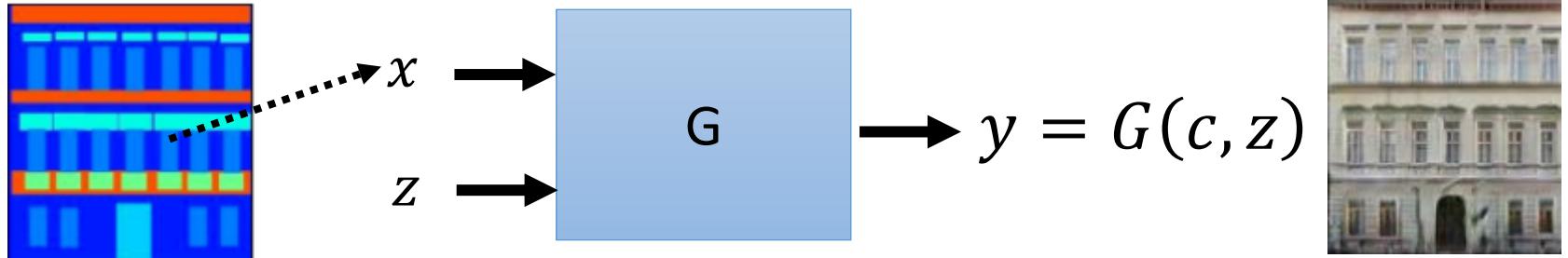
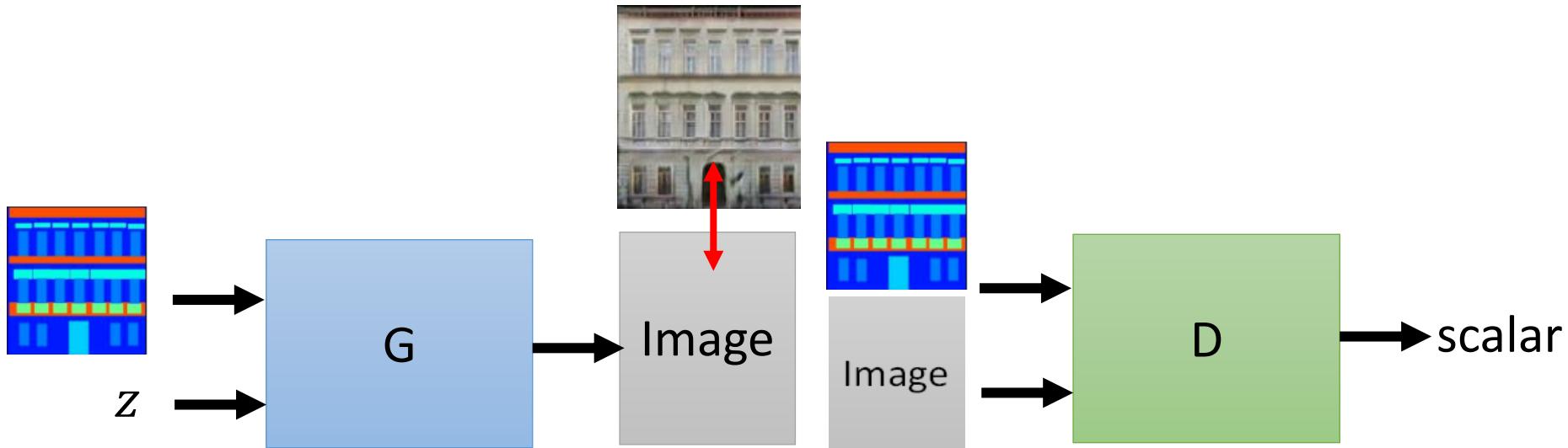
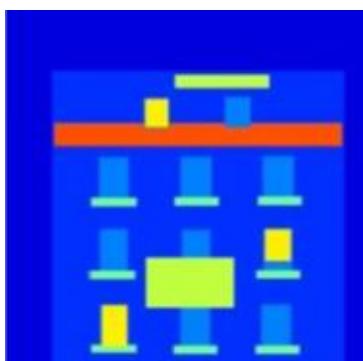


Image translation, or **pix2pix**

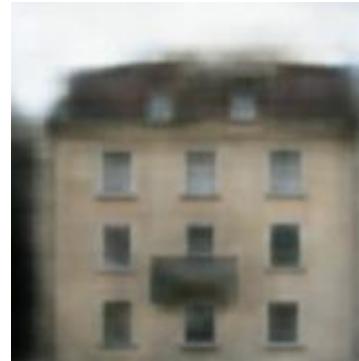
Conditional GAN



Testing:



input



supervised

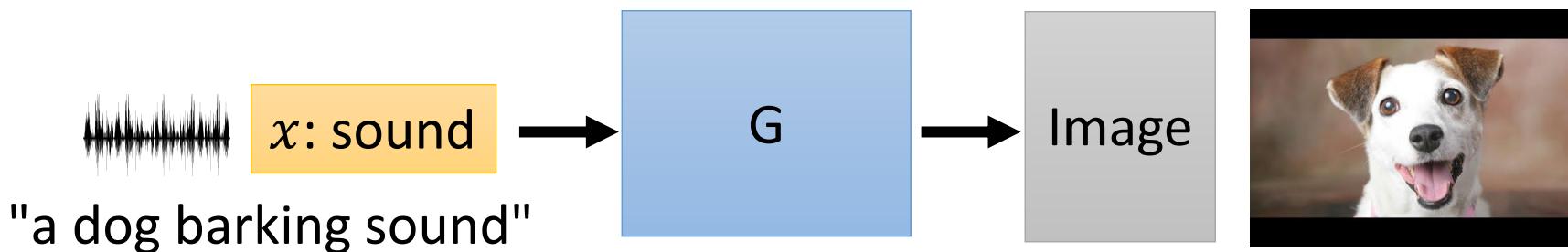


GAN

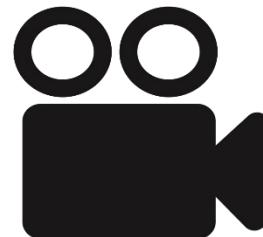


GAN + supervised

Conditional GAN



Training Data Collection



video

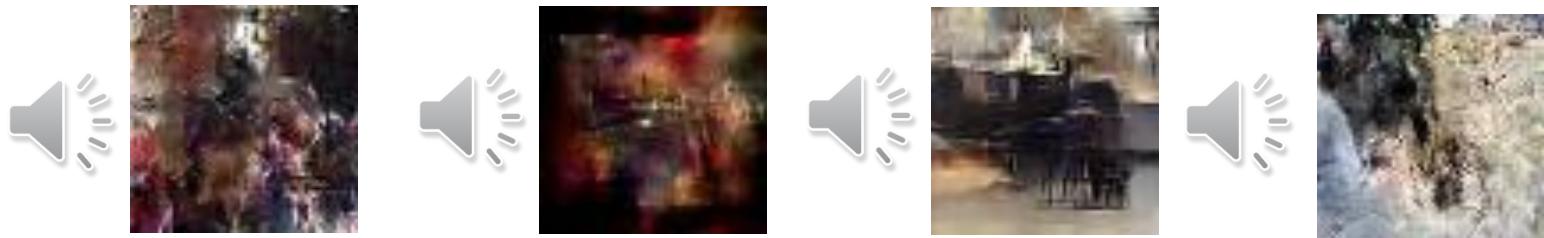
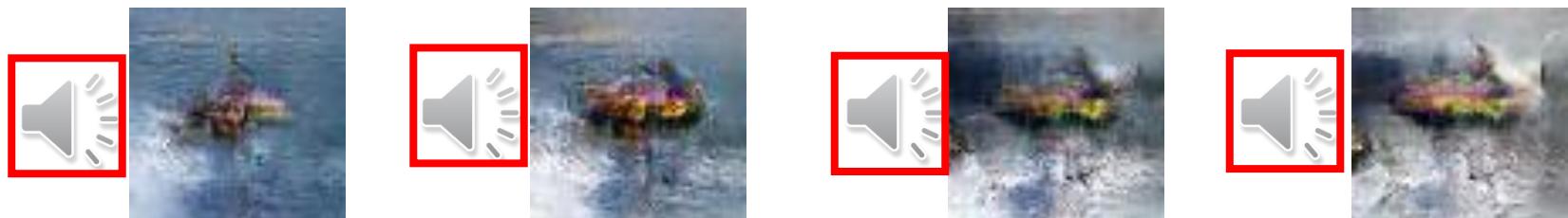


Conditional GAN

The images are generated by Chia-Hung Wan and Shun-Po Chuang.
[https://wjohn1483.github.io/
audio_to_scene/index.html](https://wjohn1483.github.io/audio_to_scene/index.html)

- Sound-to-image

Louder



Conditional GAN

Talking Head Generation



<https://arxiv.org/abs/1905.08233>

Conditional GAN

Multi-label Image Classifier = Conditional Generator

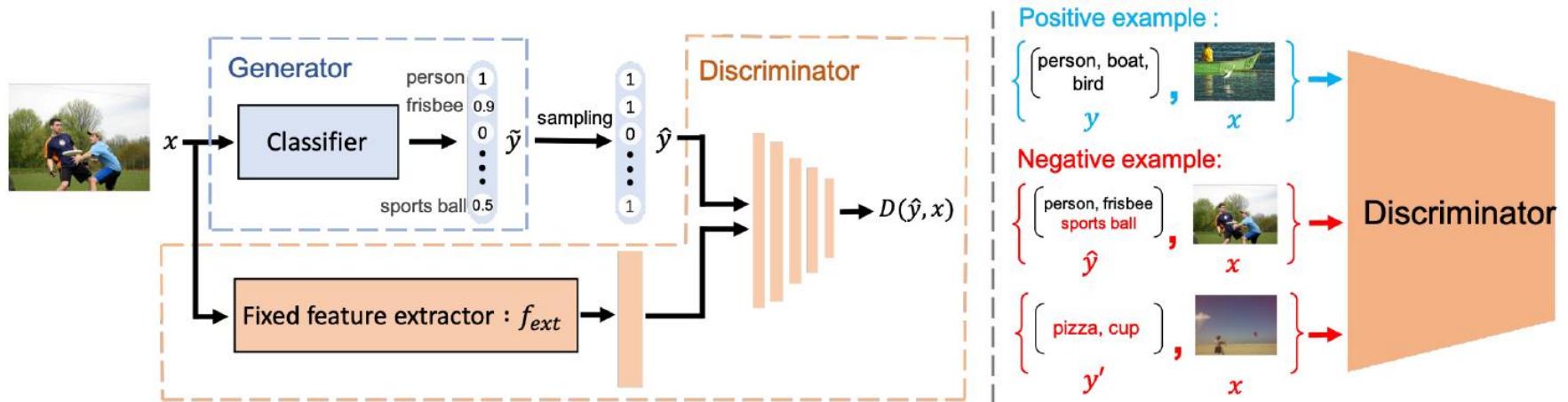


Input condition

person, sports ball,
baseball bat, baseball glove



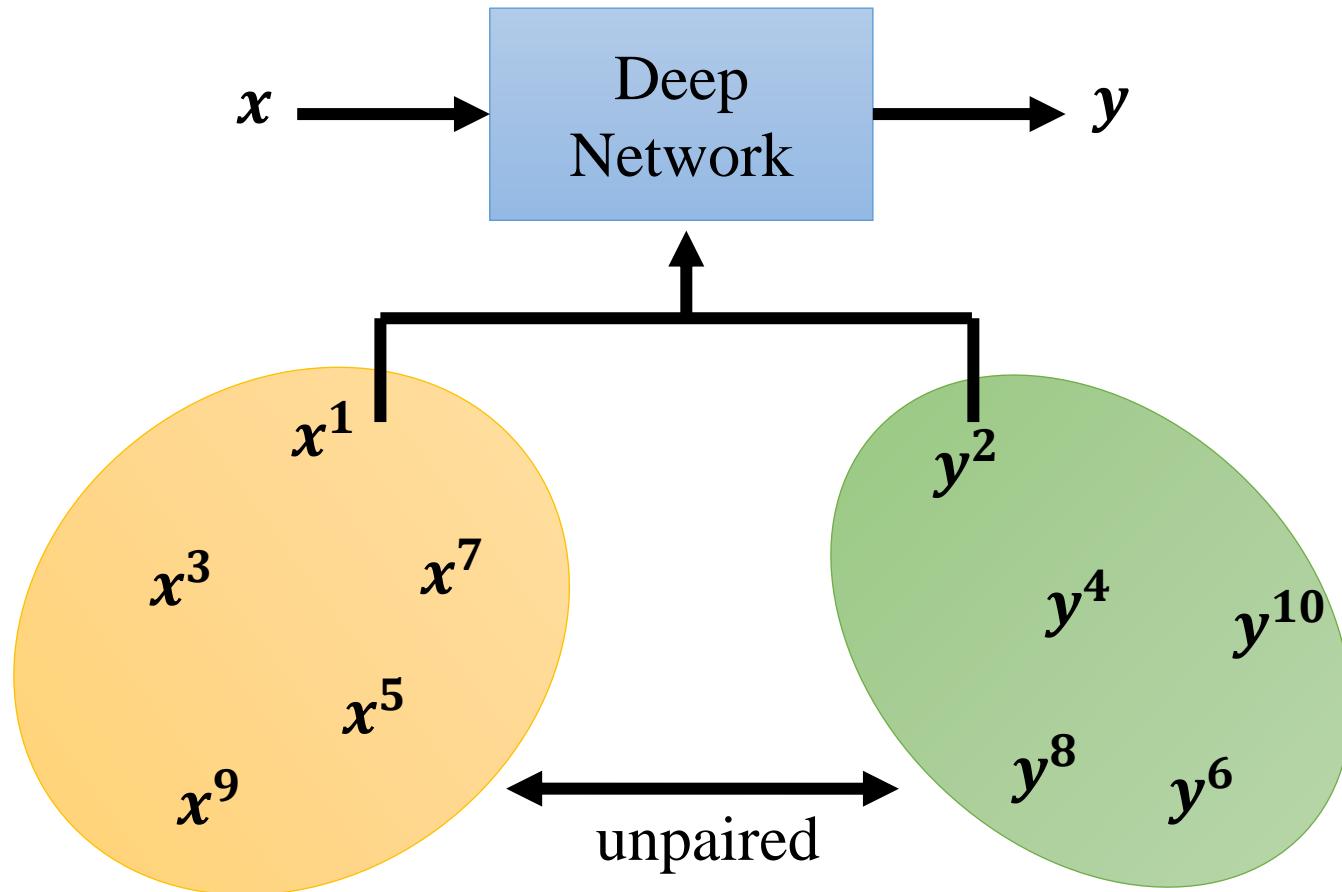
Generated output



Q&A

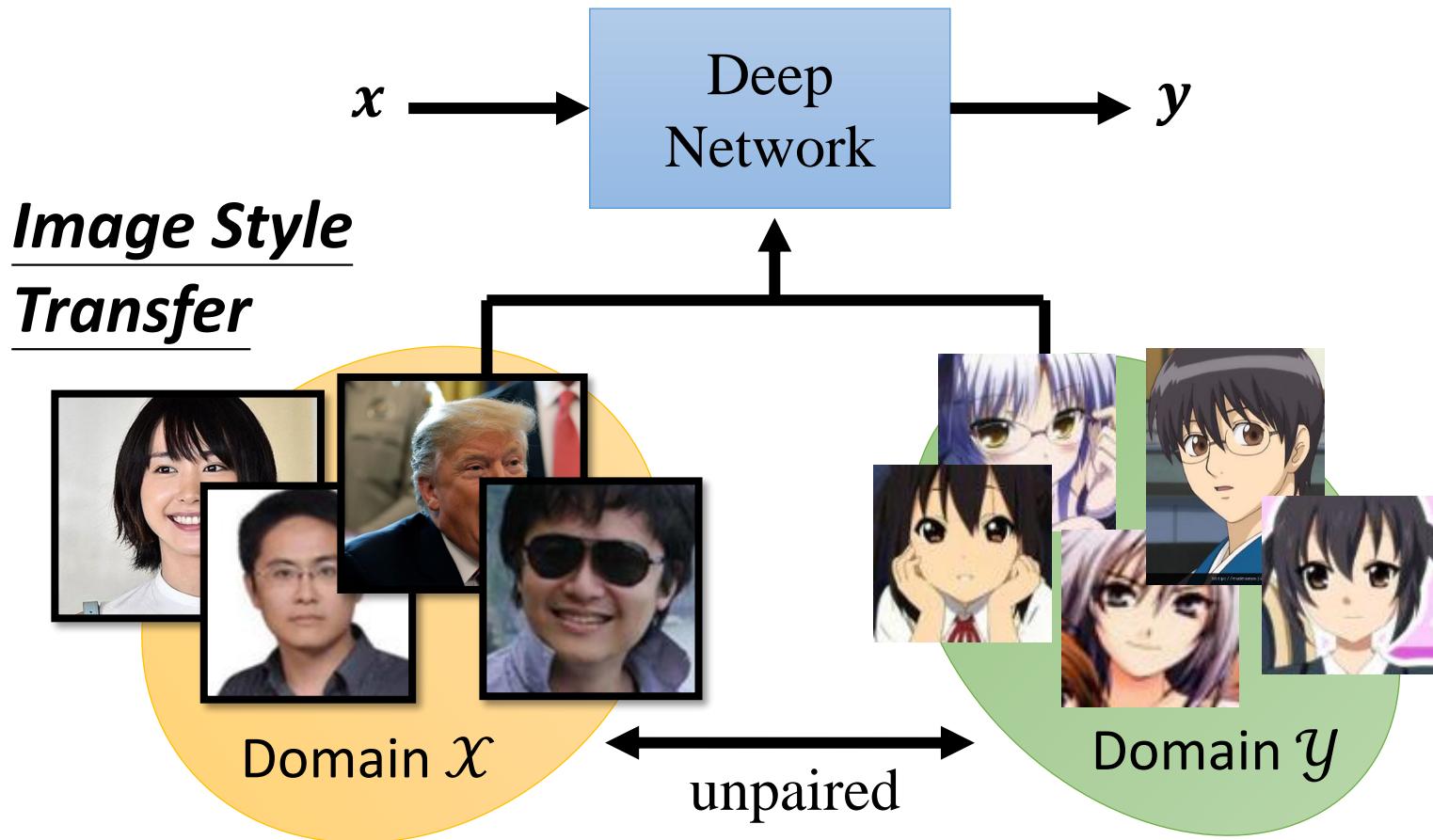
Learning from Unpaired Data

Learning from Unpaired Data



HW3: pseudo labeling
HW5: back translation } Still need **some**
paired data

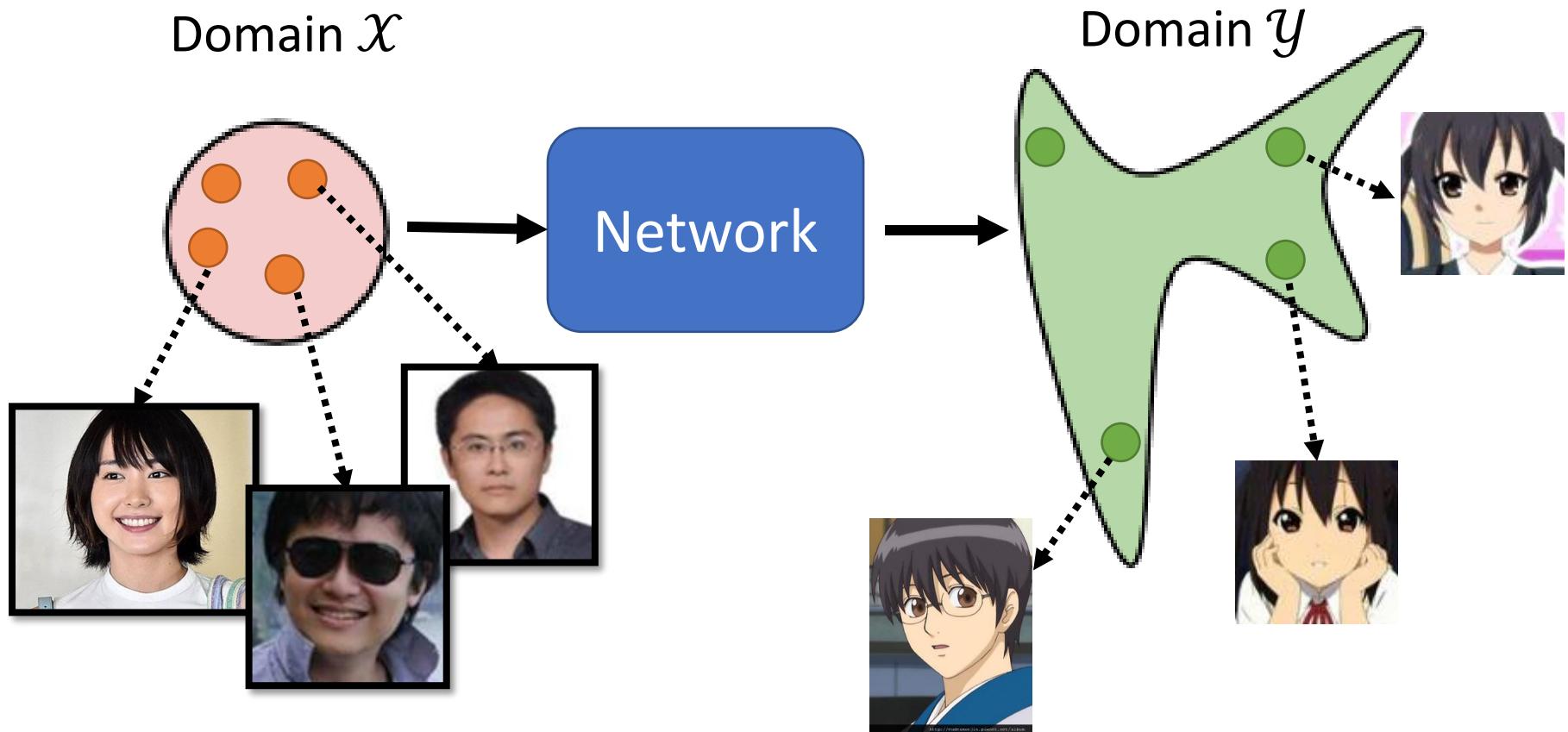
Learning from Unpaired Data



Can we learn the mapping without any paired data?

Unsupervised Conditional Generation

Learning from Unpaired Data



Cycle GAN

Domain \mathcal{X}



Domain \mathcal{Y}



Domain \mathcal{X}



$G_{\mathcal{X} \rightarrow \mathcal{Y}}$



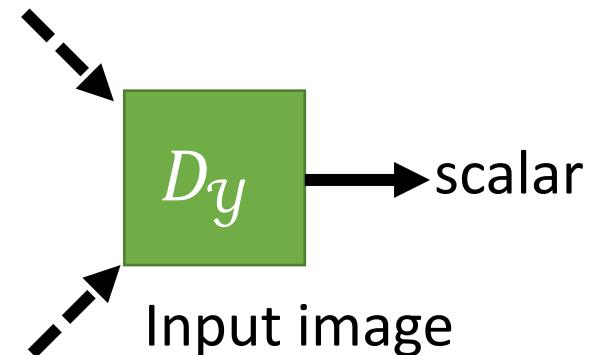
Become similar
to domain \mathcal{Y}



Domain \mathcal{Y}

D_y

scalar



Input image
belongs to
domain \mathcal{Y} or not

Cycle GAN

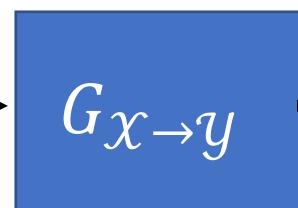
Domain \mathcal{X}



Domain \mathcal{Y}



Domain \mathcal{X}



ignore input

Become similar
to domain \mathcal{Y}



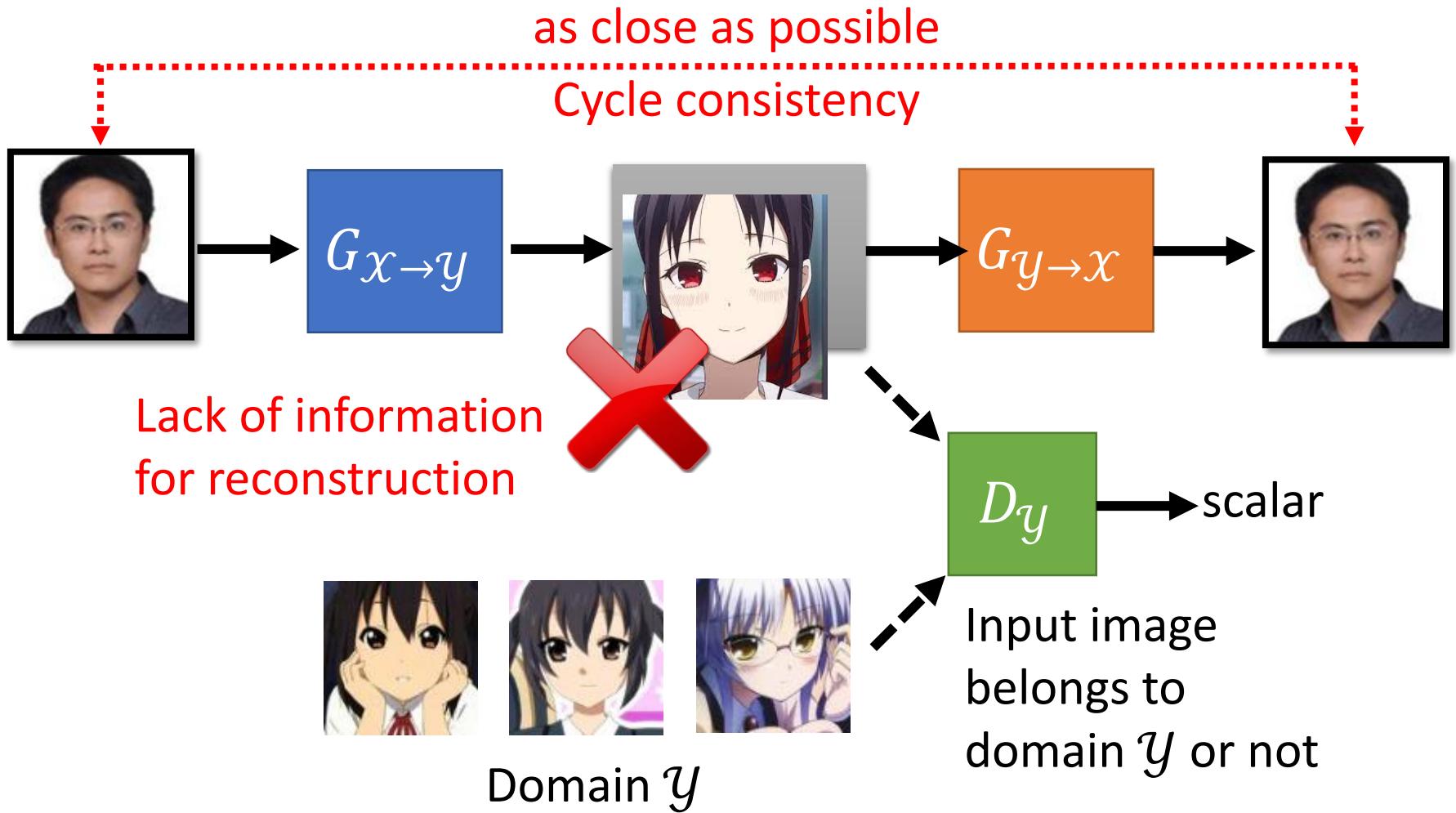
scalar



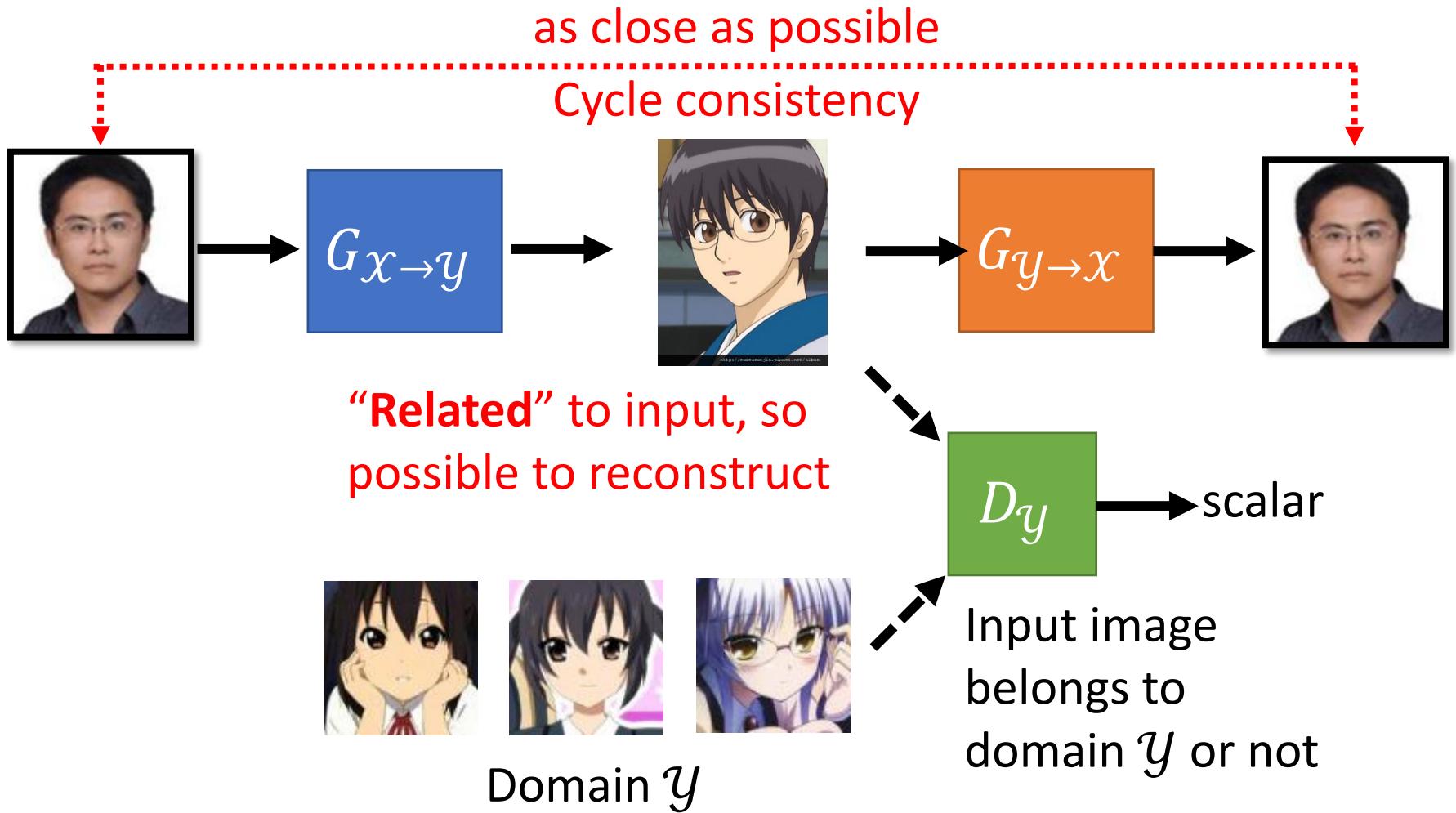
Domain \mathcal{Y}

Input image
belongs to
domain \mathcal{Y} or not

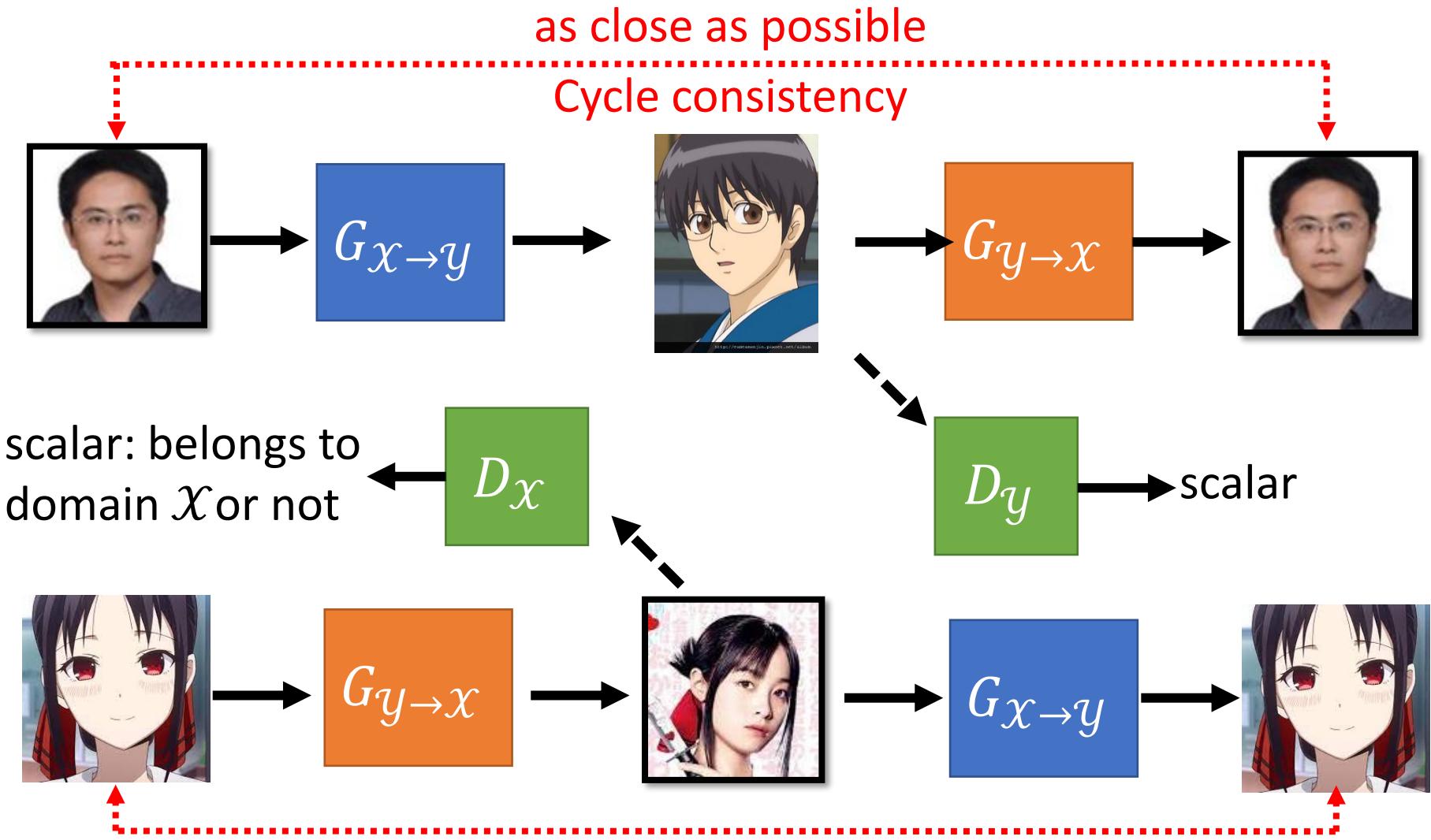
Cycle GAN



Cycle GAN

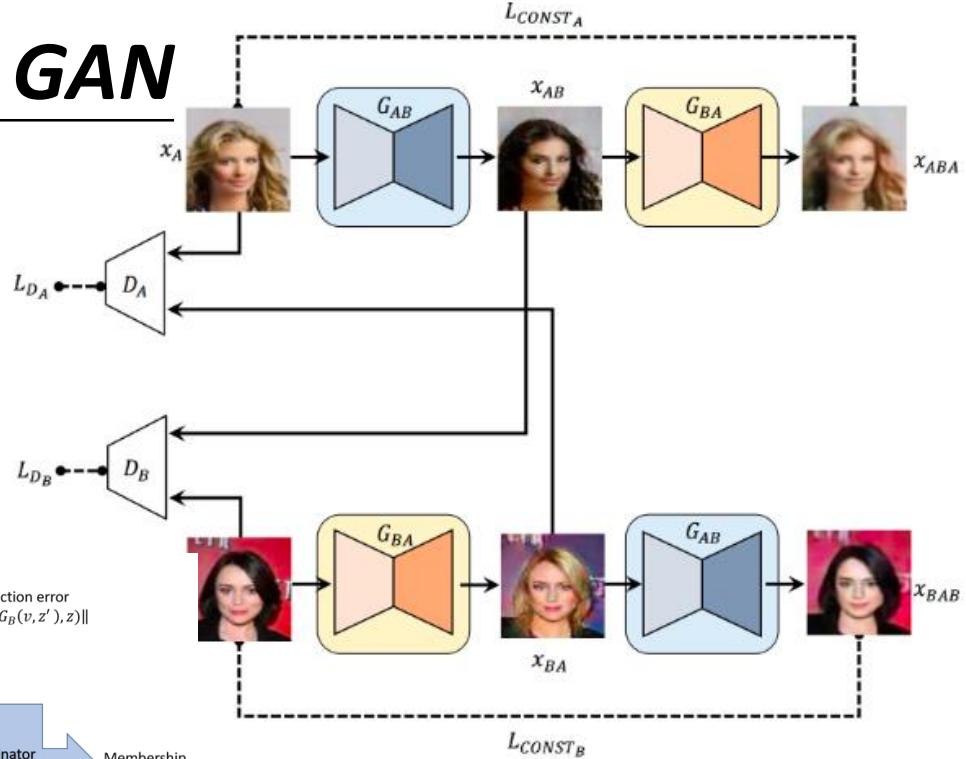


Cycle GAN



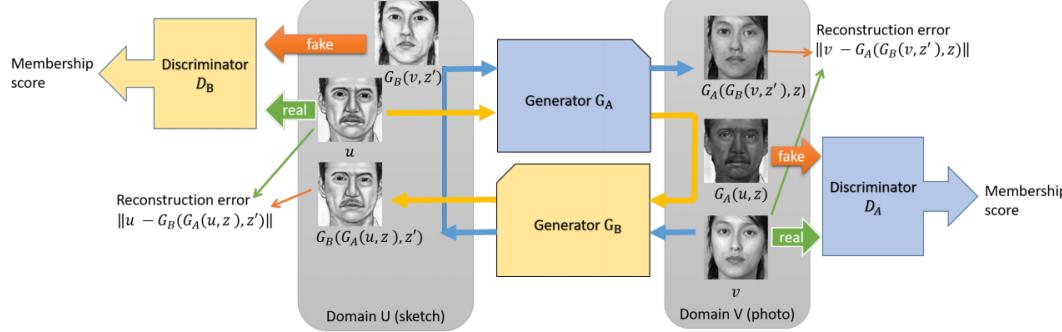
Disco GAN

<https://arxiv.org/abs/1703.05192>



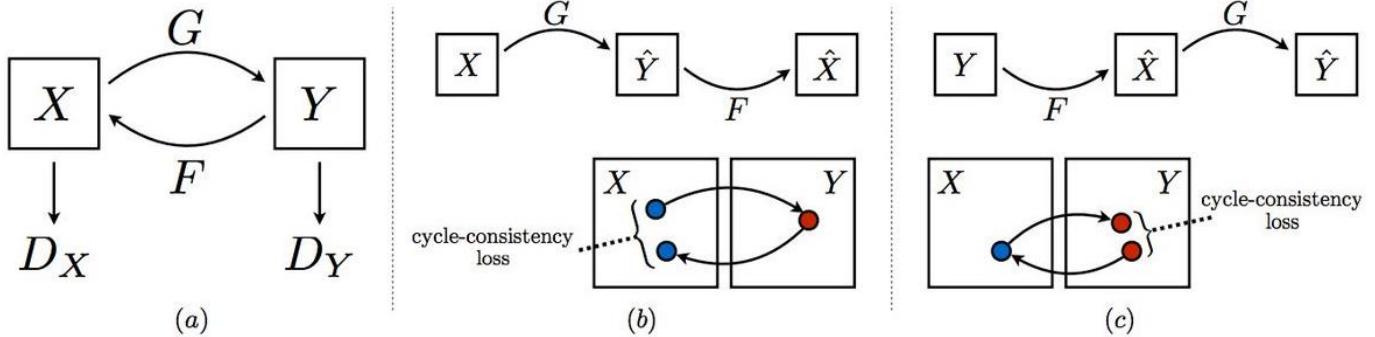
Dual GAN

<https://arxiv.org/abs/1704.02510>



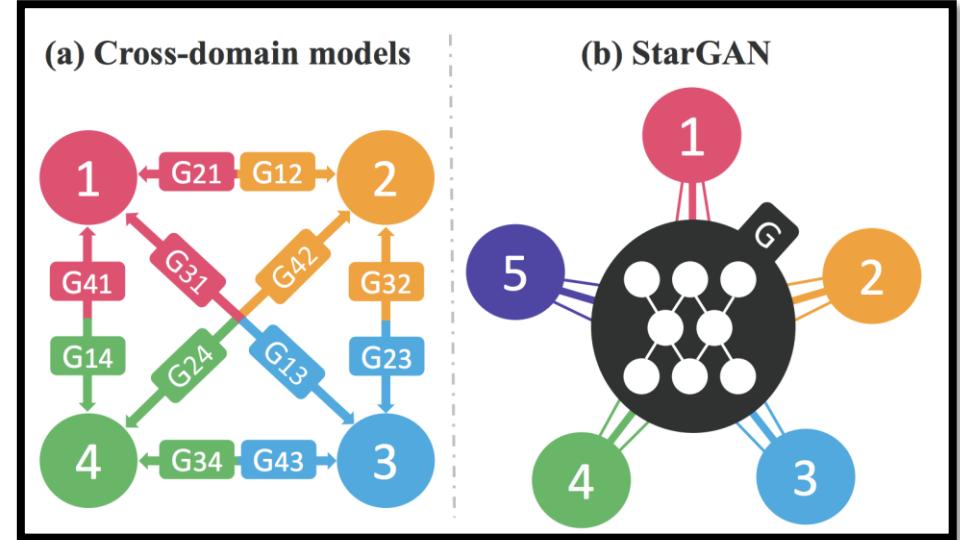
Cycle GAN

<https://arxiv.org/abs/1703.10593>

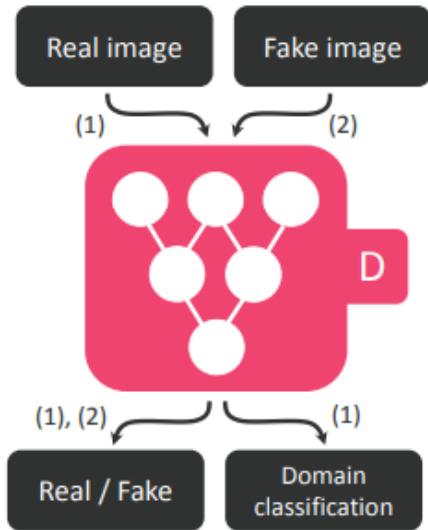


StarGAN

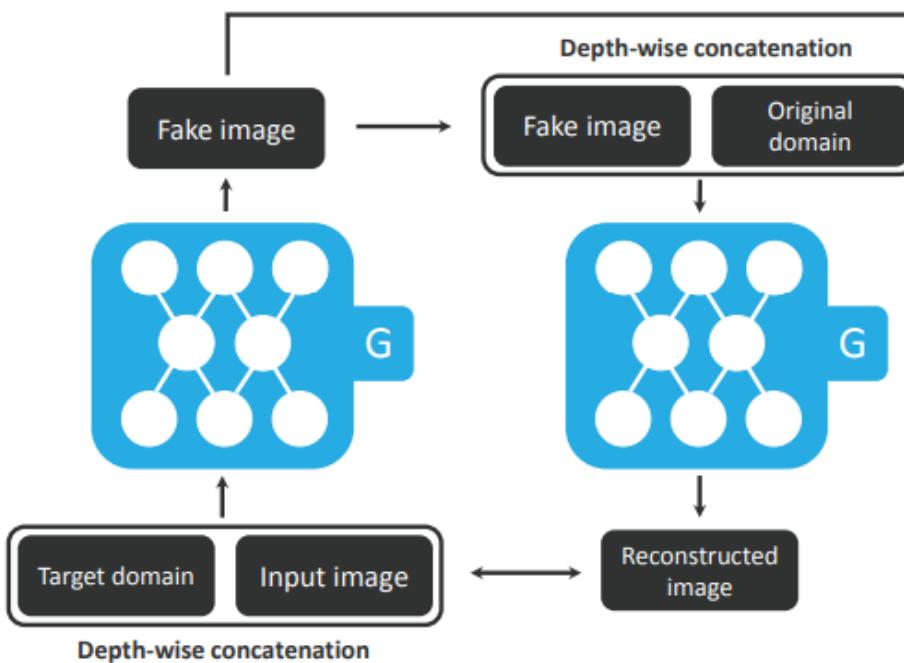
<https://arxiv.org/abs/1711.09020>



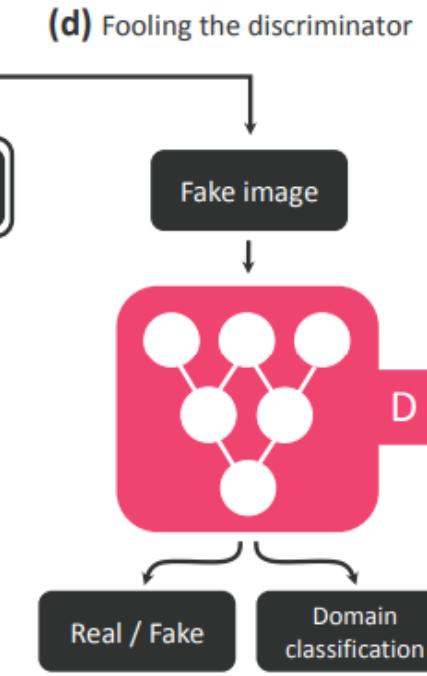
(a) Training the discriminator



(b) Original-to-target domain



(c) Target-to-original domain

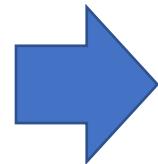
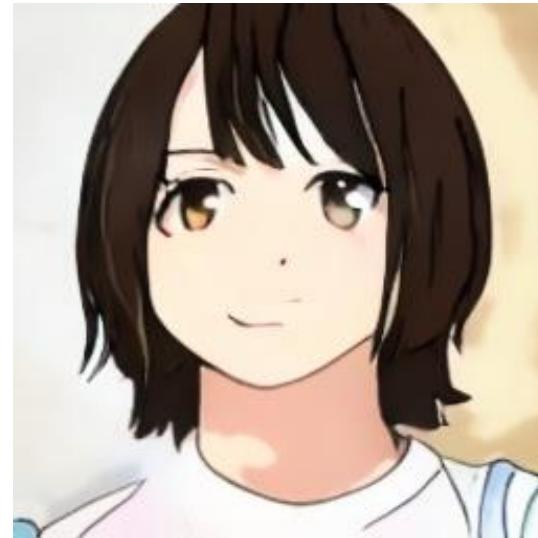


(d) Fooling the discriminator

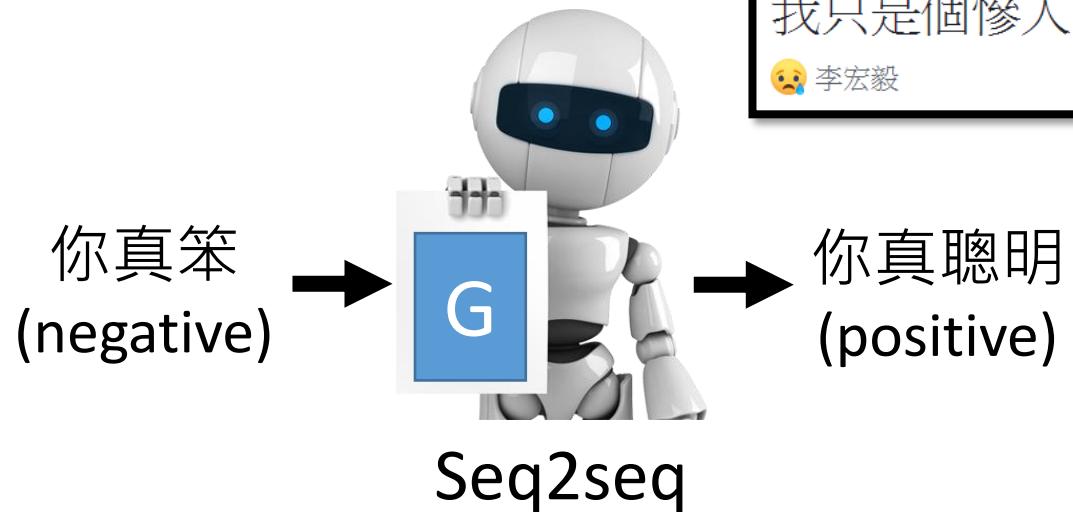
Q&A

SELFIE2ANIME

<https://selfie2anime.com/>
<https://arxiv.org/abs/1907.10830>

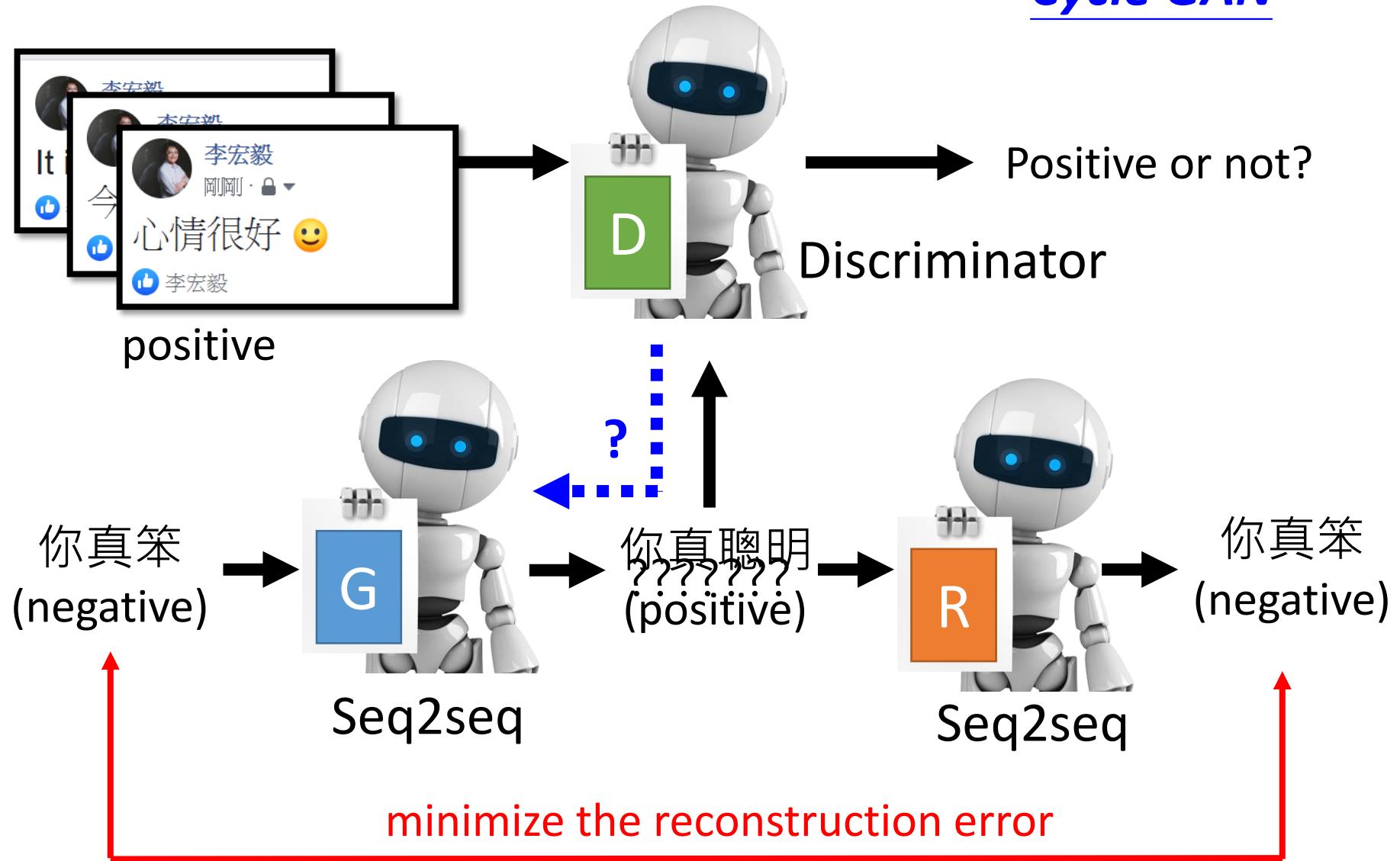


Text Style Transfer

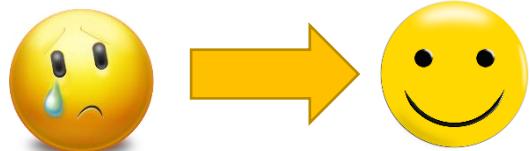


Text Style Transfer

Cycle GAN



Text Style Transfer



- From **negative sentence** to **positive** one

胃疼，沒睡醒，各種不舒服

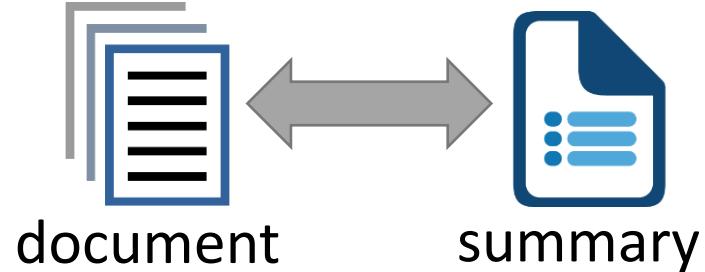
我都想去上班了，真夠賤的！

暈死了，吃燒烤、竟然遇到個變態狂

我肚子痛的厲害

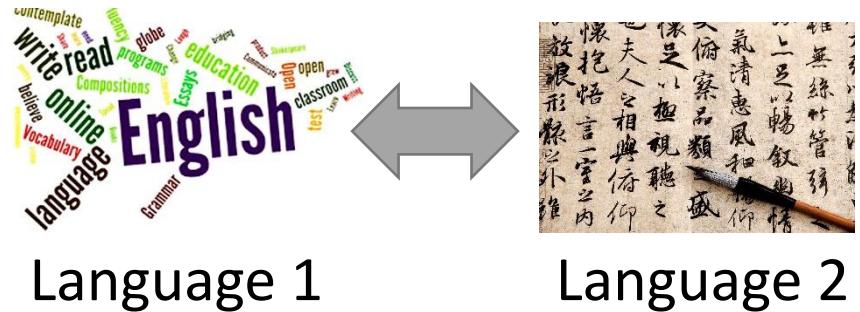
Unsupervised Abstractive Summarization

<https://arxiv.org/abs/1810.02851>



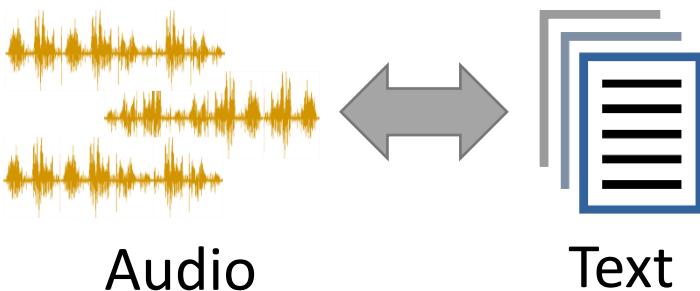
Unsupervised Translation

<https://arxiv.org/abs/1710.04087>
<https://arxiv.org/abs/1710.11041>



Unsupervised ASR

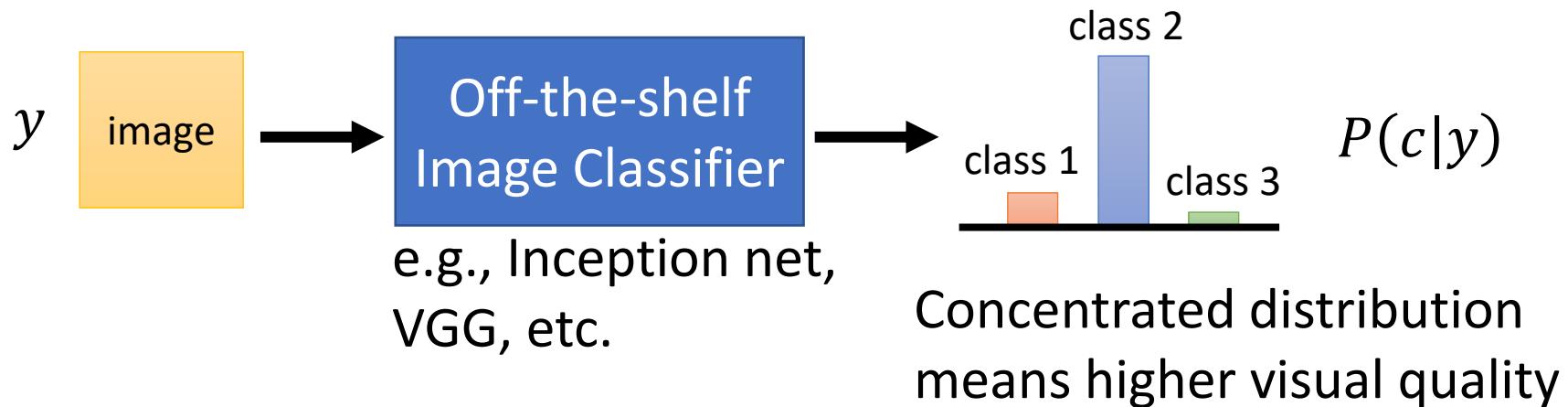
<https://arxiv.org/abs/1804.00316>
<https://arxiv.org/abs/1812.09323>
<https://arxiv.org/abs/1904.04100>



Evaluation of Generation

Quality of Image

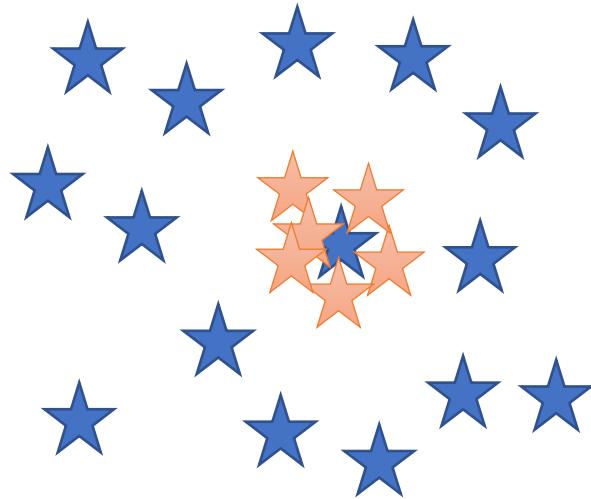
- Human evaluation is expensive (and sometimes unfair/unstable).
- How to evaluate the quality of the generated images automatically?



Diversity - Mode Collapse

★ : real data

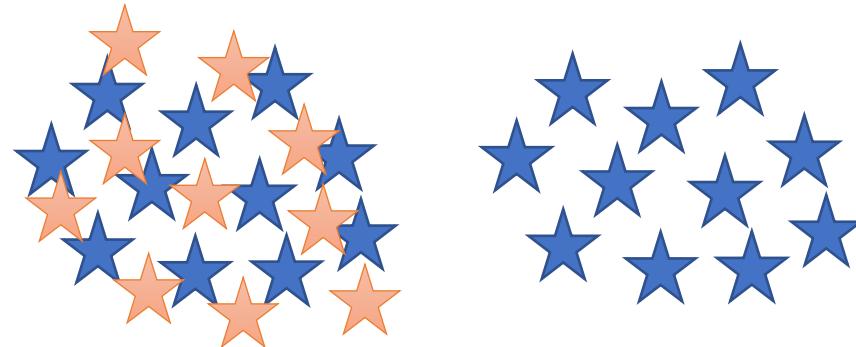
★ : generated data



Diversity - Mode Dropping

★ : real data

★ : generated data



Generator
at iteration t

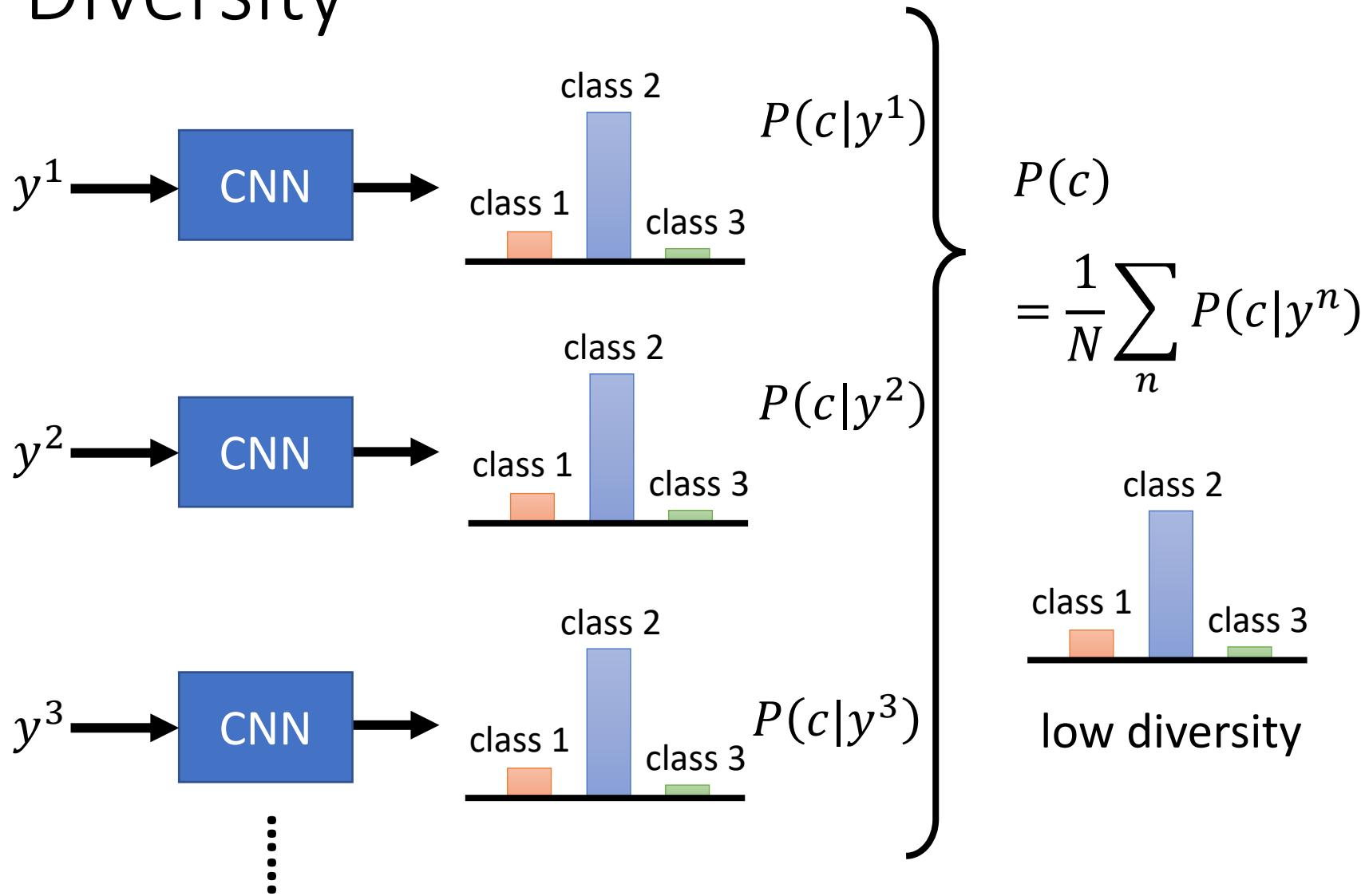


Generator
at iteration t+1

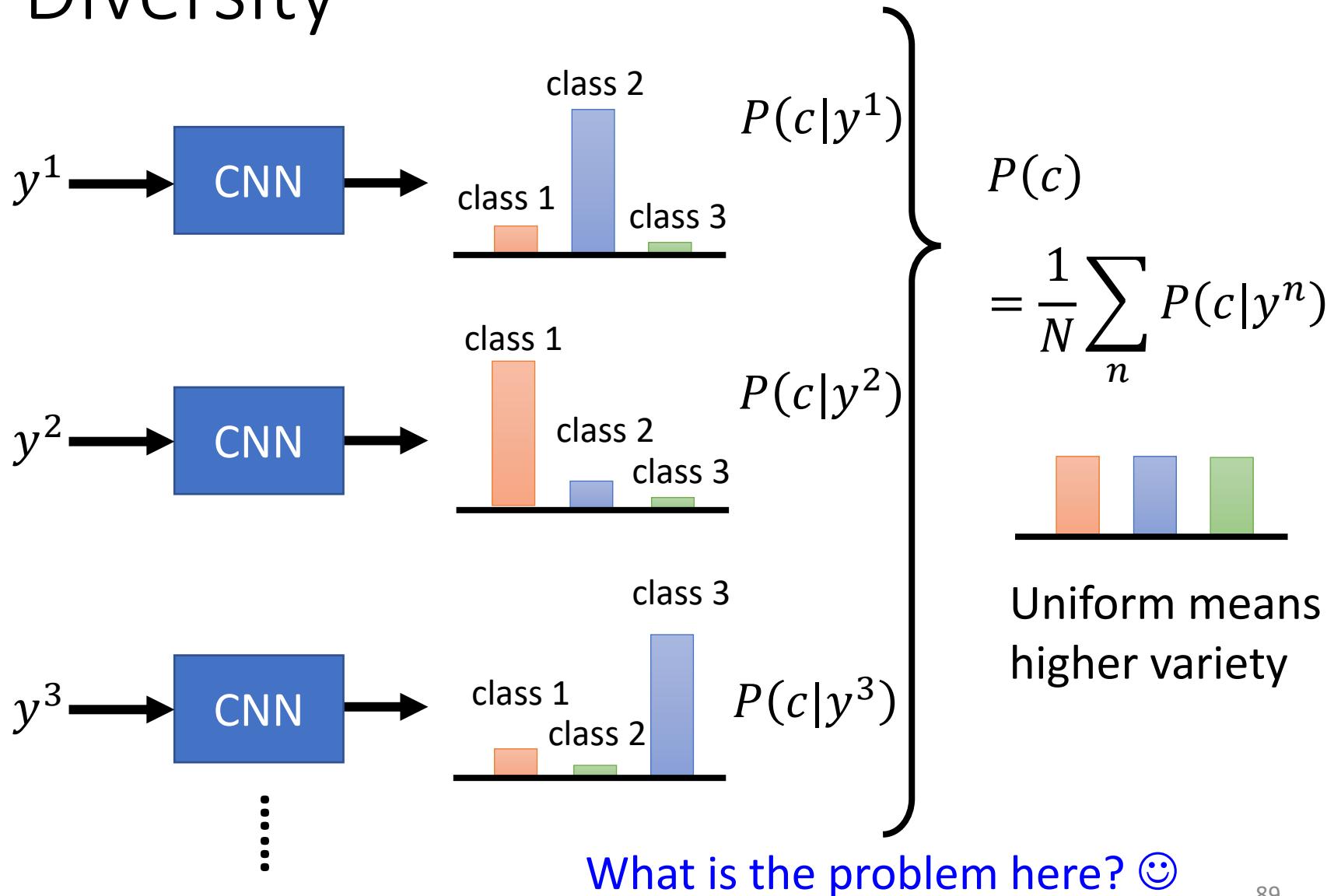


(BEGAN on CelebA)

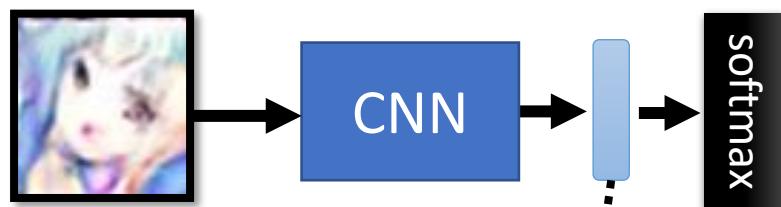
Diversity



Diversity

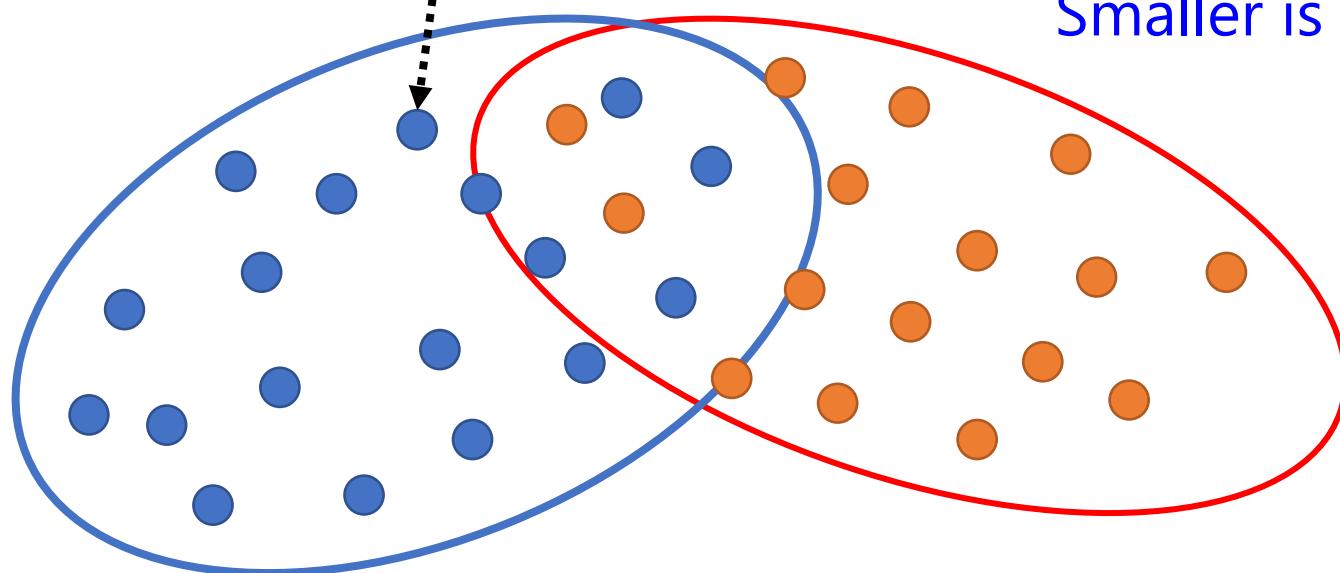


Fréchet Inception Distance (FID)

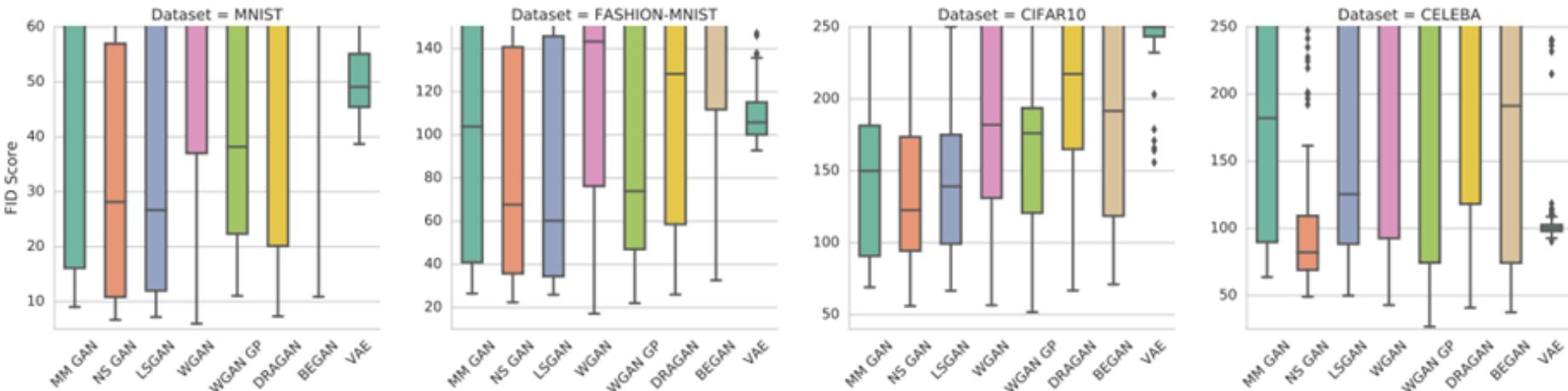


red points: real images
blue points: generated images
FID = Fréchet distance
between the two **Gaussians** ???

Smaller is better



GAN	DISCRIMINATOR LOSS	GENERATOR LOSS
MM GAN	$\mathcal{L}_D^{GAN} = -\mathbb{E}_{x \sim p_d} [\log(D(x))] + \mathbb{E}_{\hat{x} \sim p_g} [\log(1 - D(\hat{x}))]$	$\mathcal{L}_G^{GAN} = -\mathcal{L}_D^{GAN}$
NS GAN	$\mathcal{L}_D^{NSGAN} = \mathcal{L}_D^{GAN}$	$\mathcal{L}_G^{NSGAN} = \mathbb{E}_{\hat{x} \sim p_g} [\log(D(\hat{x}))]$
WGAN	$\mathcal{L}_D^{WGAN} = -\mathbb{E}_{x \sim p_d} [D(x)] + \mathbb{E}_{\hat{x} \sim p_g} [D(\hat{x})]$	$\mathcal{L}_G^{WGAN} = -\mathcal{L}_D^{WGAN}$
WGAN GP	$\mathcal{L}_D^{WGAN} = \mathcal{L}_D^{WGAN} + \lambda \mathbb{E}_{\hat{x} \sim p_g} [(\nabla D(\alpha x + (1 - \alpha)\hat{x}) _2 - 1)^2]$	$\mathcal{L}_G^{WGAN} = -\mathbb{E}_{\hat{x} \sim p_g} [D(\hat{x})]$
LS GAN	$\mathcal{L}_D^{LSGAN} = -\mathbb{E}_{x \sim p_d} [(D(x) - 1)^2] + \mathbb{E}_{\hat{x} \sim p_g} [D(\hat{x})^2]$	$\mathcal{L}_G^{LSGAN} = -\mathbb{E}_{\hat{x} \sim p_g} [(D(\hat{x}) - 1)^2]$
DRAGAN	$\mathcal{L}_D^{DRAGAN} = \mathcal{L}_D^{GAN} + \lambda \mathbb{E}_{\hat{x} \sim p_d + \mathcal{N}(0, c)} [(\nabla D(\hat{x}) _2 - 1)^2]$	$\mathcal{L}_G^{DRAGAN} = -\mathcal{L}_D^{NSGAN}$
BEGAN	$\mathcal{L}_D^{BEGAN} = \mathbb{E}_{x \sim p_d} [x - AE(x) _1] - k_t \mathbb{E}_{\hat{x} \sim p_g} [\hat{x} - AE(\hat{x}) _1]$	$\mathcal{L}_G^{BEGAN} = \mathbb{E}_{\hat{x} \sim p_g} [\hat{x} - AE(\hat{x}) _1]$

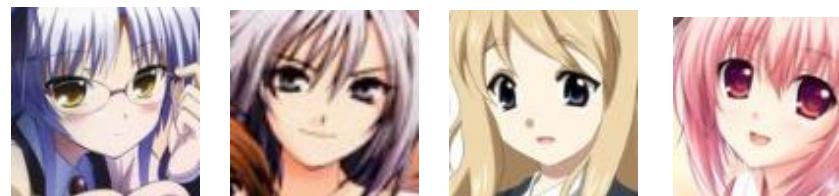


FIT: Smaller is better

Are GANs Created Equal? A Large-Scale Study
<https://arxiv.org/abs/1711.10337>

We don't want memory GAN.

Real Data



Generated
Data



Same as real data ...

Generated
Data



Simply flip real data ...

To learn more about evaluation ...

	Measure	Description
Quantitative	1. Average Log-likelihood [18, 22]	<ul style="list-style-type: none"> • Log likelihood of explaining realworld held out/test data using a density estimated from the generated data (e.g. using KDE or Parzen window estimation). $L = \frac{1}{N} \sum_i \log P_{model}(\mathbf{x}_i)$
	2. Coverage Metric [33]	<ul style="list-style-type: none"> • The probability mass of the true data "covered" by the model distribution $C := P_{Data}(dP_{model} > t)$ with t such that $P_{model}(dP_{model} > t) = 0.95$
	3. Inception Score (IS) [3]	<ul style="list-style-type: none"> • KLD between conditional and marginal label distributions over generated data. $\exp(\mathbb{E}_{\mathbf{x}} [\text{KL}(p(y \mathbf{x}) \ p(y))])$
	4. Modified Inception Score (m-IS) [34]	<ul style="list-style-type: none"> • Encourages diversity within images sampled from a particular category. $\exp(\mathbb{E}_{\mathbf{x}_i} [\mathbb{E}_{\mathbf{x}_j} ([\text{KL}(P(y \mathbf{x}_i) P(y \mathbf{x}_j))])])$
	5. Mode Score (MS) [35]	<ul style="list-style-type: none"> • Similar to IS but also takes into account the prior distribution of the labels over real data. $\exp(\mathbb{E}_{\mathbf{x}} [\text{KL}(p(y \mathbf{x}) \ p(y^{train}))] - \text{KL}(p(y) \ p(y^{train})))$
	6. AM Score [36]	<ul style="list-style-type: none"> • Takes into account the KLD between distributions of training labels vs. predicted labels, as well as the entropy of predictions. $\text{KL}(p(y^{train}) \ p(y)) + \mathbb{E}_{\mathbf{x}} [H(y \mathbf{x})]$
	7. Fréchet Inception Distance (FID) [37]	<ul style="list-style-type: none"> • Wasserstein-2 distance between multi-variate Gaussians fitted to data embedded into a feature space $FID(r, g) = \ \mu_r - \mu_g\ _2^2 + Tr(\Sigma_r + \Sigma_g - 2(\Sigma_r \Sigma_g)^{\frac{1}{2}})$
	8. Maximum Mean Discrepancy (MMD) [38]	<ul style="list-style-type: none"> • Measures the dissimilarity between two probability distributions P_r and P_g using samples drawn independently from each distribution. $M_k(P_r, P_g) = \mathbb{E}_{\mathbf{x}, \mathbf{x}' \sim P_r} [k(\mathbf{x}, \mathbf{x}')] - 2\mathbb{E}_{\mathbf{x} \sim P_r, \mathbf{y} \sim P_g} [k(\mathbf{x}, \mathbf{y})] + \mathbb{E}_{\mathbf{y}, \mathbf{y}' \sim P_g} [k(\mathbf{y}, \mathbf{y}')]$
	9. The Wasserstein Critic [39]	<ul style="list-style-type: none"> • The critic (e.g. an NN) is trained to produce high values at real samples and low values at generated samples $\hat{W}(\mathbf{x}_{test}, \mathbf{x}_g) = \frac{1}{N} \sum_{i=1}^N \hat{f}(\mathbf{x}_{test}[i]) - \frac{1}{N} \sum_{i=1}^N \hat{f}(\mathbf{x}_g[i])$
	10. Birthday Paradox Test [27]	<ul style="list-style-type: none"> • Measures the support size of a discrete (continuous) distribution by counting the duplicates (near duplicates)
	11. Classifier Two Sample Test (C2ST) [40]	<ul style="list-style-type: none"> • Answers whether two samples are drawn from the same distribution (e.g. by training a binary classifier)
	12. Classification Performance [1, 15]	<ul style="list-style-type: none"> • An indirect technique for evaluating the quality of unsupervised representations (e.g. feature extraction; FCN score). See also the GAN Quality Index (GQI) [41].
	13. Boundary Distortion [42]	<ul style="list-style-type: none"> • Measures diversity of generated samples and covariate shift using classification methods.
	14. Number of Statistically-Different Bins (NDB) [43]	<ul style="list-style-type: none"> • Given two sets of samples from the same distribution, the number of samples that fall into a given bin should be the same up to sampling noise
	15. Image Retrieval Performance [44]	<ul style="list-style-type: none"> • Measures the distributions of distances to the nearest neighbors of some query images (i.e. diversity)
	16. Generative Adversarial Metric (GAM) [31]	<ul style="list-style-type: none"> • Compares two GANs by having them engaged in a battle against each other by swapping discriminators or generators. $p(\mathbf{x} y=1; M_1)/p(\mathbf{x} y=1; M_2) = (p(y=1 \mathbf{x}; D_1)p(\mathbf{x}; G_2))/(p(y=1 \mathbf{x}; D_2)p(\mathbf{x}; G_1))$
	17. Tournament Win Rate and Skill Rating [45]	<ul style="list-style-type: none"> • Implements a tournament in which a player is either a discriminator that attempts to distinguish between real and fake data or a generator that attempts to fool the discriminators into accepting fake data as real.
	18. Normalized Relative Discriminative Score (NRDS) [32]	<ul style="list-style-type: none"> • Compares n GANs based on the idea that if the generated samples are closer to real ones, more epochs would be needed to distinguish them from real samples.
	19. Adversarial Accuracy and Divergence [16]	<ul style="list-style-type: none"> • Adversarial Accuracy: Computes the classification accuracies achieved by the two classifiers, one trained on real data and another on generated data, on a labeled validation set to approximate $P_g(y \mathbf{x})$ and $P_r(y \mathbf{x})$. Adversarial Divergence: Computes $\text{KL}(P_g(y \mathbf{x}), P_r(y \mathbf{x}))$
	20. Geometry Score [47]	<ul style="list-style-type: none"> • Compares geometrical properties of the underlying data manifold between real and generated data.
	21. Reconstruction Error [48]	<ul style="list-style-type: none"> • Measures the reconstruction error (e.g. L_2 norm) between a test image and its closest generated image by optimizing for z (i.e. $\min_z \ G(\mathbf{z}) - \mathbf{x}^{(test)}\ ^2$)
	22. Image Quality Measures [49, 50, 51]	<ul style="list-style-type: none"> • Evaluates the quality of generated images using measures such as SSIM, PSNR, and sharpness difference
	23. Low-level Image Statistics [52, 53]	<ul style="list-style-type: none"> • Evaluates how similar low-level statistics of generated images are to those of natural scenes in terms of mean power spectrum, distribution of random filter responses, contrast distribution, etc.
	24. Precision, Recall and F_1 score [23]	<ul style="list-style-type: none"> • These measures are used to quantify the degree of overfitting in GANs, often over toy datasets.
Qualitative	1. Nearest Neighbors	<ul style="list-style-type: none"> • To detect overfitting, generated samples are shown next to their nearest neighbors in the training set
	2. Rapid Scene Categorization [18]	<ul style="list-style-type: none"> • In these experiments, participants are asked to distinguish generated samples from real images in a short presentation time (e.g. 100 ms); i.e. real v.s fake
	3. Preference Judgment [54, 55, 56, 57]	<ul style="list-style-type: none"> • Participants are asked to rank models in terms of the fidelity of their generated images (e.g. pairs, triples)
	4. Mode Drop and Collapse [58, 59]	<ul style="list-style-type: none"> • Over datasets with known modes (e.g. a GMM or a labeled dataset), modes are computed as by measuring the distances of generated data to mode centers
	5. Network Internals [1, 60, 61, 62, 63, 64]	<ul style="list-style-type: none"> • Regards exploring and illustrating the internal representation and dynamics of models (e.g. space continuity) as well as visualizing learned features

Pros and cons of GAN evaluation measures

<https://arxiv.org/abs/1802.03446>

Concluding Remarks

Introduction of Generative Models

Generative Adversarial Network (GAN)

Theory behind GAN

Tips for GAN

Conditional Generation

Learning from unpaired data

Evaluation of Generative Models

Q&A