Introduction of Generative Adversarial Network (GAN)

李宏毅

Hung-yi Lee

Generative Adversarial Network (GAN)

- How to pronounce “GAN”?

Google 小姐

Yann LeCun's comment

What are some recent and potentially upcoming breakthroughs in unsupervised learning?

Yann LeCun, Director of AI Research at Facebook and Professor at NYU
Written Jul 29 · Upvoted by Joaquin Quiñonero Candela, Director Applied Machine Learning at Facebook and Huang Xiao

Adversarial training is the coolest thing since sliced bread.

I've listed a bunch of relevant papers in a previous answer.

Expect more impressive results with this technique in the coming years.

What's missing at the moment is a good understanding of it so we can make it work reliably. It's very finicky. Sort of like ConvNet were in the 1990s, when I had the reputation of being the only person who could make them work (which wasn't true).

https://www.quora.com/What-are-some-recent-and-potentially-upcoming-breakthroughs-in-unsupervised-learning

Yann LeCun's comment

What are some recent and potentially upcoming breakthroughs in deep learning?

Yann LeCun, Director of AI Research at Facebook and Professor at NYU

Written Jul 29 · Upvoted by Joaquin Quiñonero Candela, Director Applied Machine Learning at Facebook and Nikhil Garg, I lead a team of Quora engineers working on ML/NLP problems

.....

The most important one, in my opinion, is adversarial training (also called GAN for Generative Adversarial Networks). This is an idea that was originally proposed by Ian Goodfellow when he was a student with Yoshua Bengio at the University of Montreal (he since moved to Google Brain and recently to OpenAI).

This, and the variations that are now being proposed is the most interesting idea in the last 10 years in ML, in my opinion.

https://www.quora.com/What-are-some-recent-and-potentially-upcoming-breakthroughs-in-deep-learning

All Kinds of GAN …

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

GAN

ACGAN

BGAN

CGAN

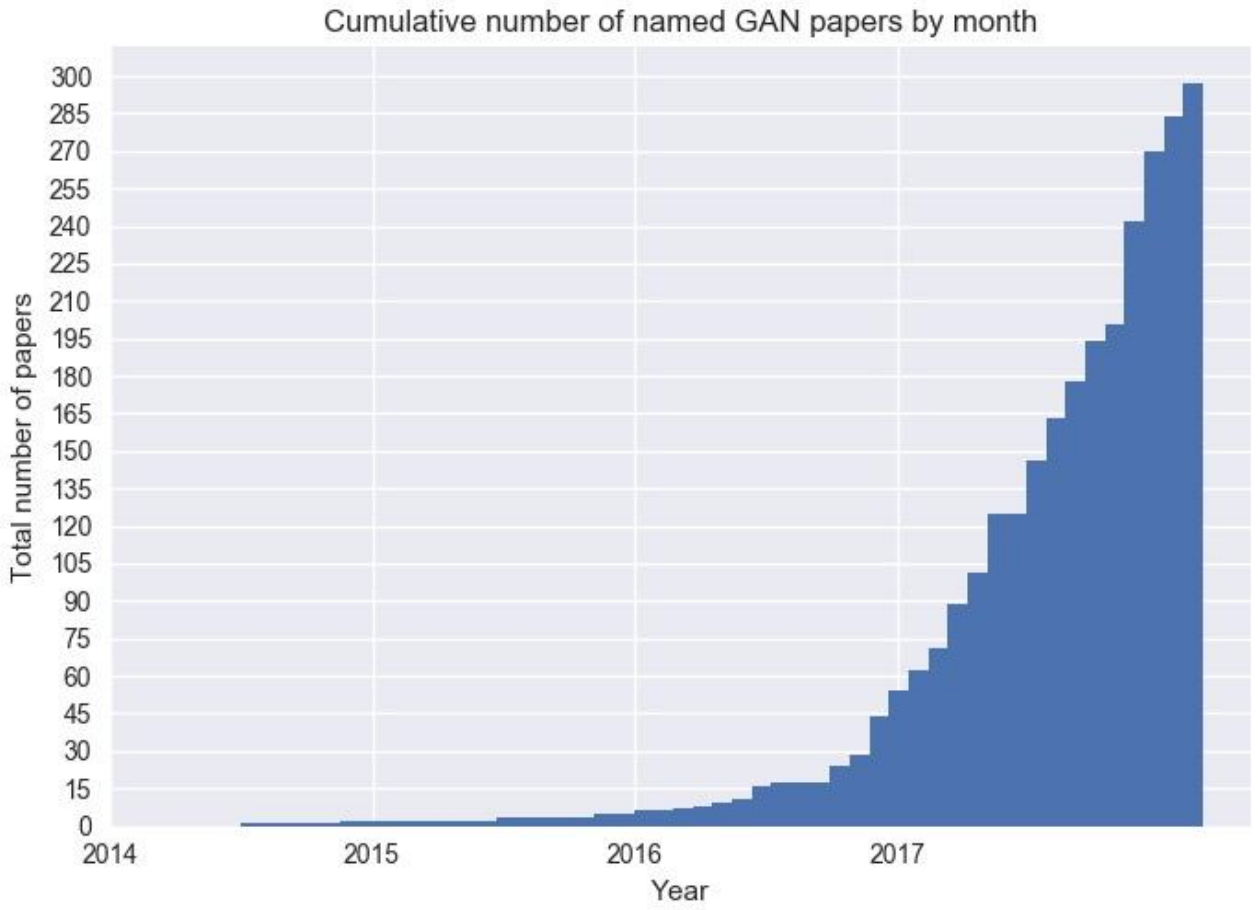
DCGAN

EBGAN

fGAN

GoGAN

⋮

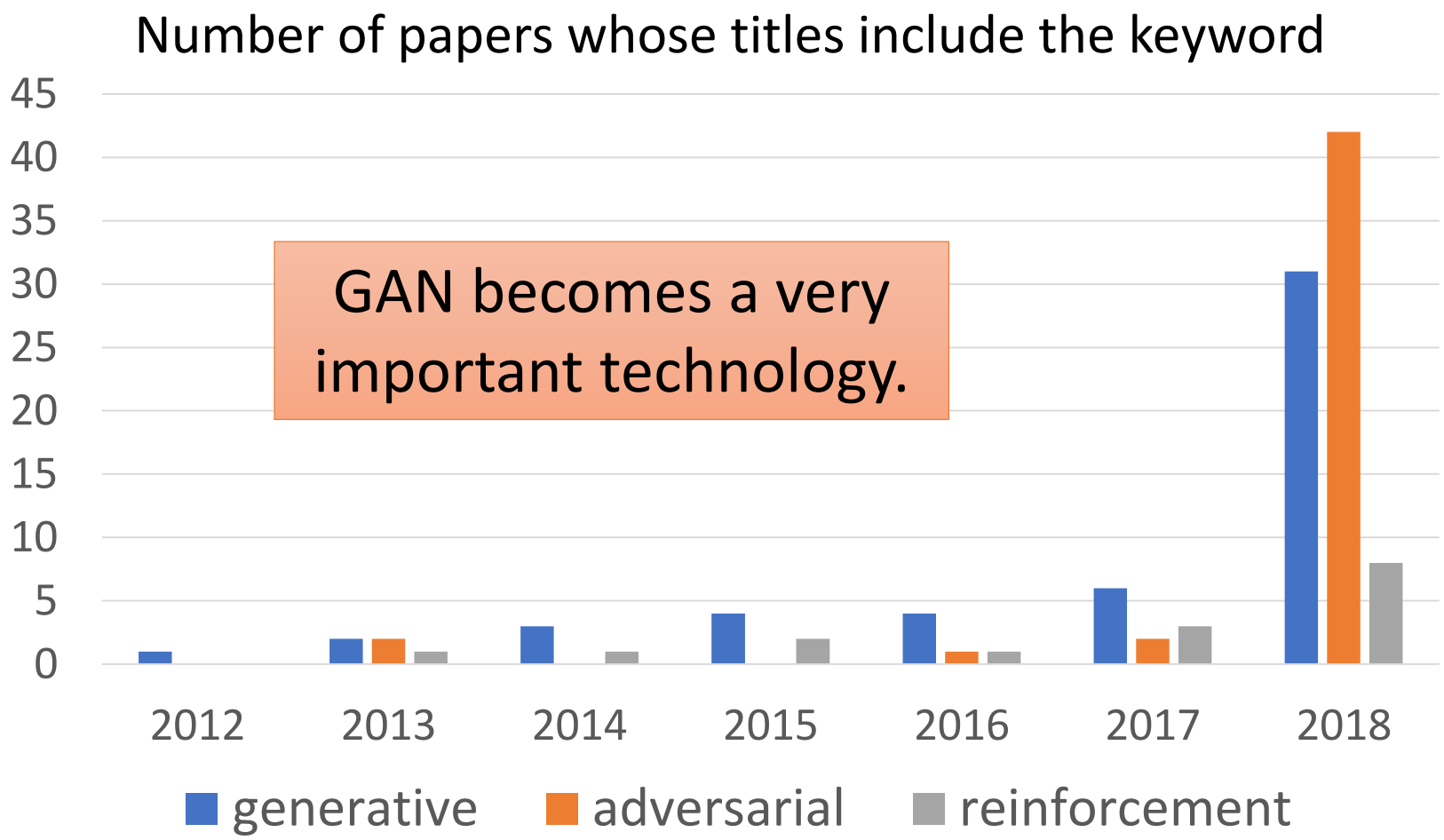


Mihaela Rosca, Balaji Lakshminarayanan, David Warde-Farley, Shakir Mohamed, "Variational Approaches for Auto-Encoding Generative Adversarial Networks", arXiv, 2017

[2]We 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.

ICASSP

Keyword search on session index page, so session names are included.



Outline

Basic Idea of GAN

GAN as structured learning

Can Generator learn by itself?

Can Discriminator generate?

A little bit theory

Generation

We will control what to generate latter. → Conditional Generation

Image Generation

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

In a specific range

→ NN Generator →

Sentence Generation

$\begin{bmatrix} 0.3 \\ -0.1 \\ \vdots \\ -0.7 \end{bmatrix} \begin{bmatrix} 0.1 \\ -0.1 \\ \vdots \\ 0.2 \end{bmatrix} \begin{bmatrix} -0.3 \\ 0.1 \\ \vdots \\ 0.5 \end{bmatrix}$

→ NN Generator →

How are you?

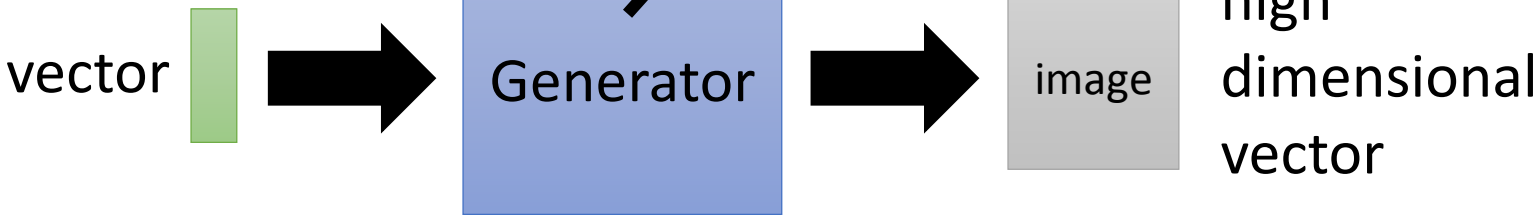
Good morning.

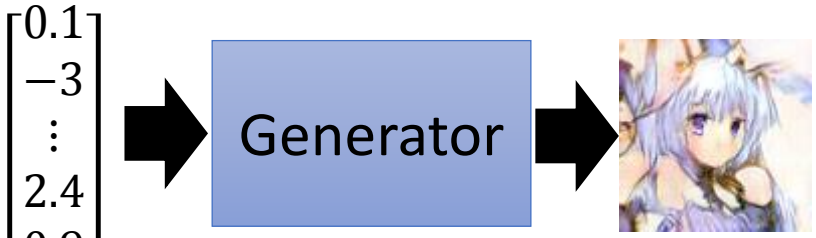
Good afternoon.

Powered by: http://mattya.github.io/chainer-DCGAN/

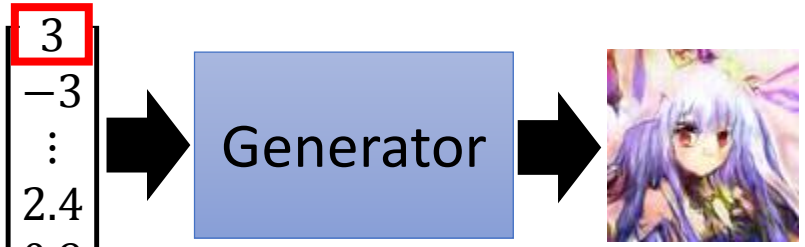
Basic Idea of GAN

It is a neural network (NN), or a function.

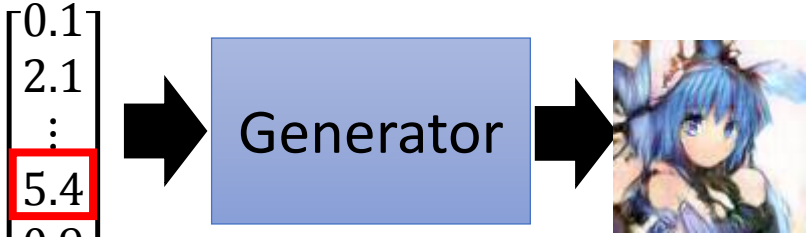




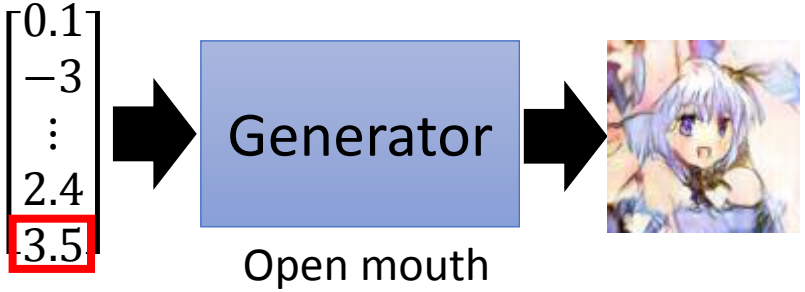
Each dimension of input vector represents some characteristics.



Longer hair



blue hair



Open mouth

Basic Idea of GAN

It is a neural network (NN), or a function.

image → Discri-minator → scalar

Larger value means real, smaller value means fake.

 → Discri-minator → 1.0

 → Discri-minator → 1.0

 → Discri-minator → 0.1

 → Discri-minator → 0.1

Basic Idea of GAN

Brown

Generator

veins

Butterflies are not brown

Butterflies do not have veins

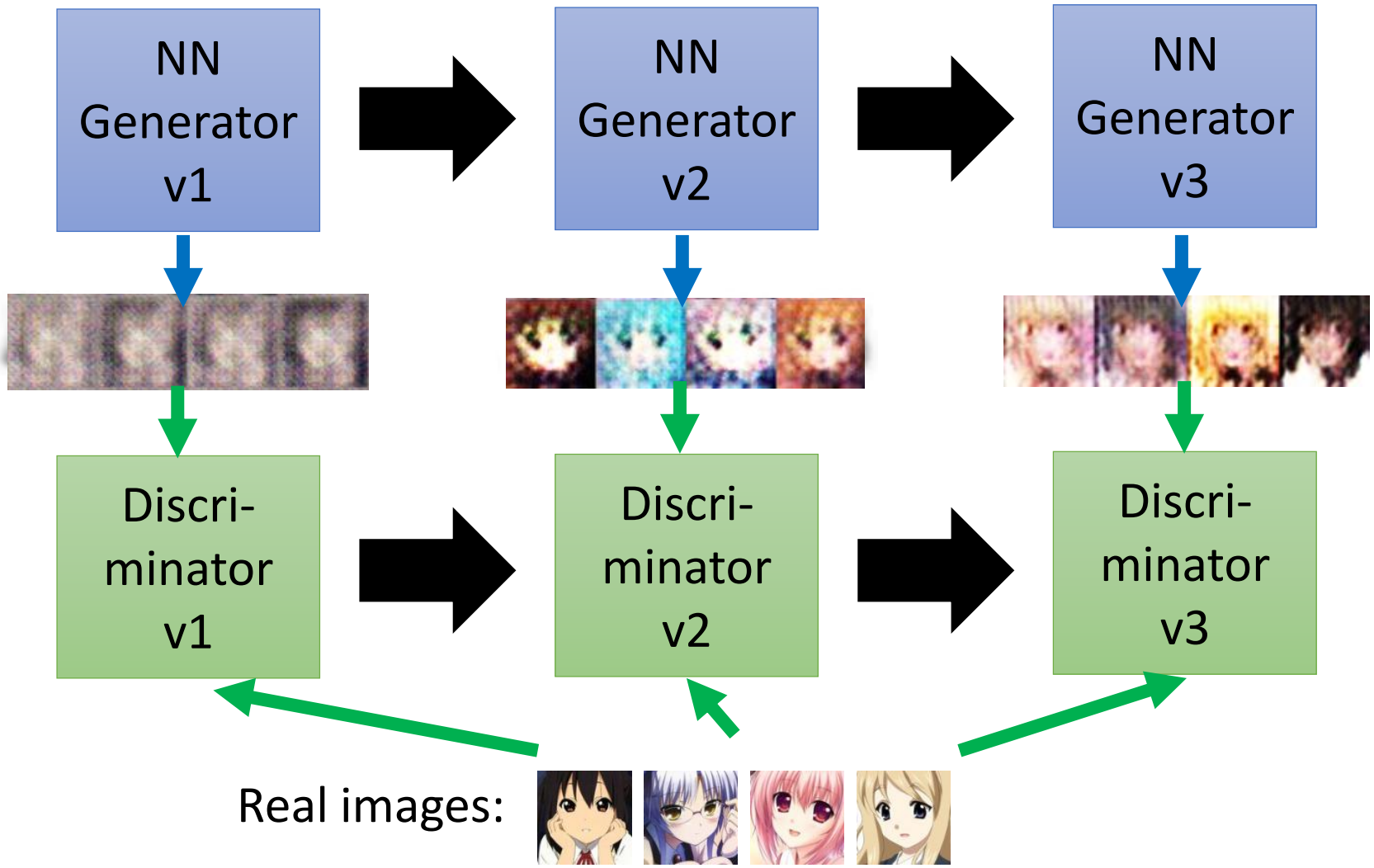
.....

Discriminator

Basic Idea of GAN

This is where the term "***adversarial***" comes from.

You can explain the process in different ways…….

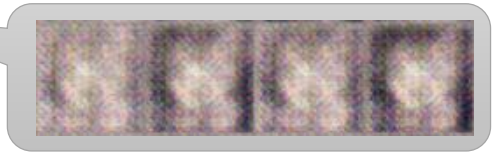


Basic Idea of GAN (和平的比喻)

Generator (student)

Discriminator (teacher)

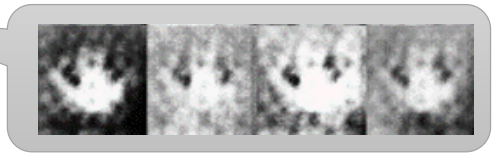
Generator v1



Discriminator v1

沒有兩個圈

Generator v2



Discriminator v2

沒有彩色

Generator v3

為什麼不自己學？

為什麼不自己做？

Generator v.s. Discriminator

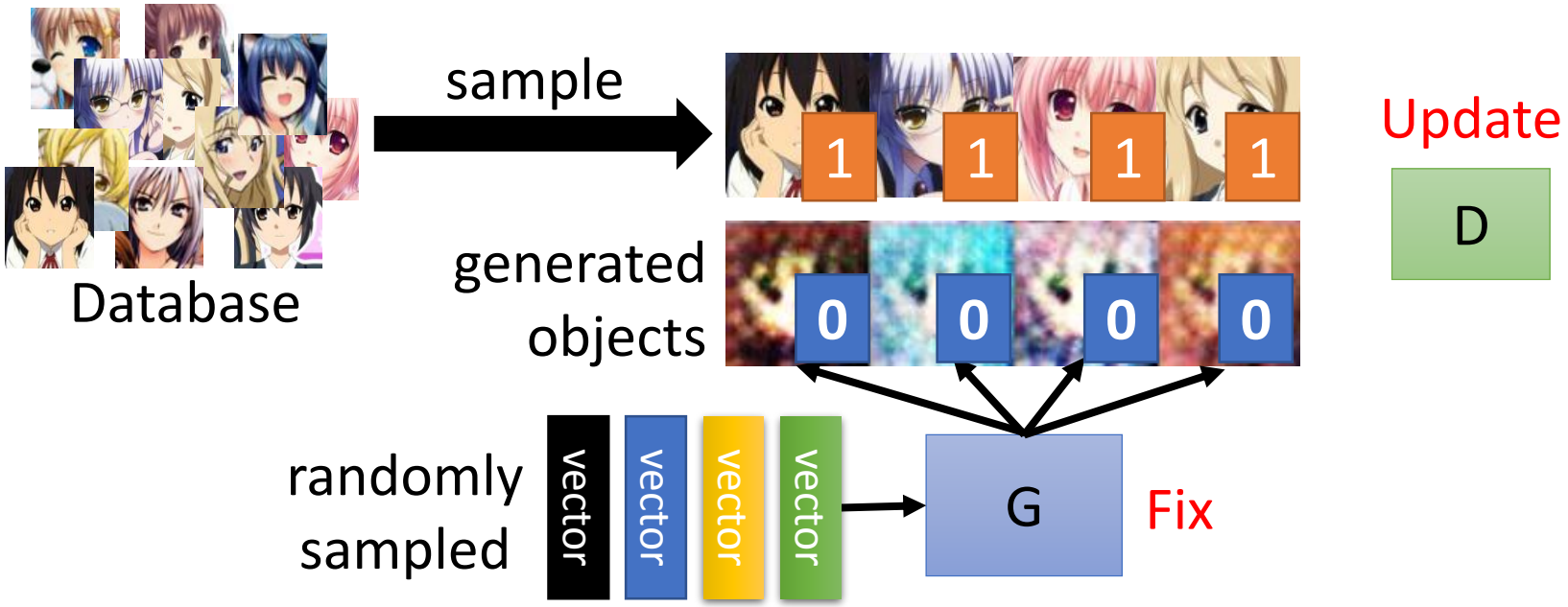
- 寫作敵人，唸做朋友

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.

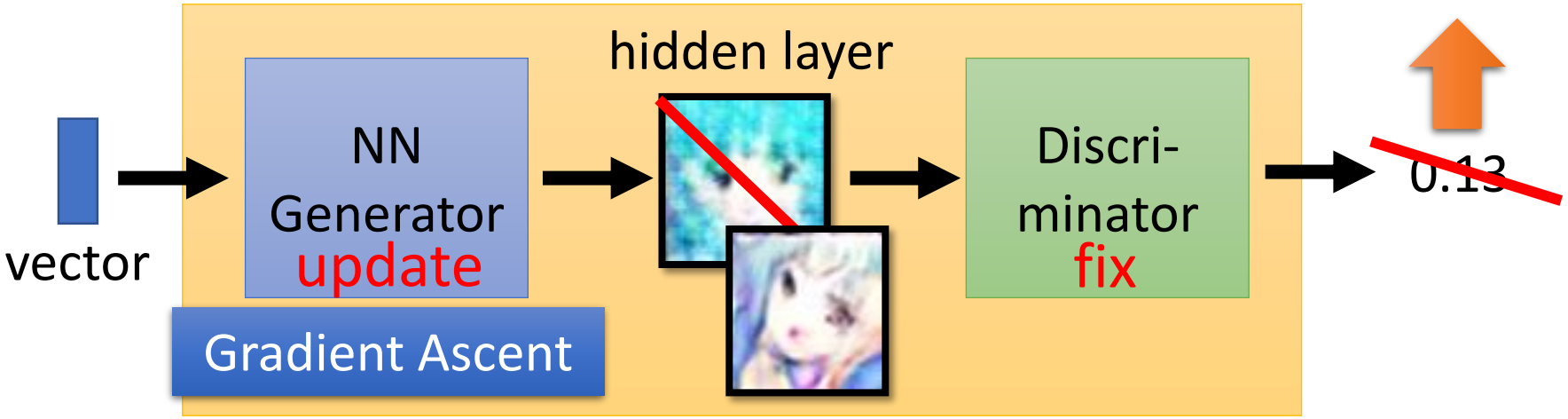
Algorithm

- Initialize generator and discriminator
- In each training iteration:

G D

Step 2: Fix discriminator D, and update generator G

Generator learns to "fool" the discriminator



large network

Algorithm Initialize θ_d for D and θ_g for G

- In each training iteration:

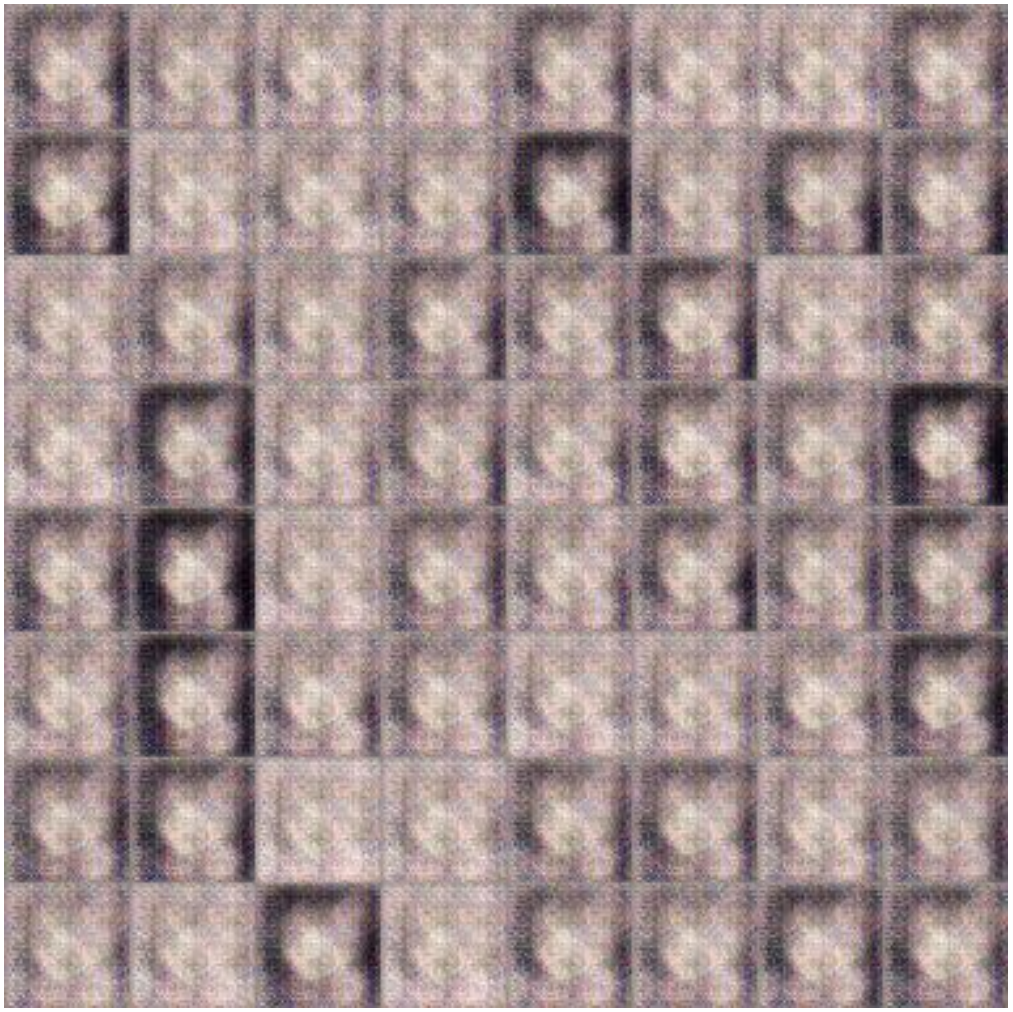
Learning D

- Sample m examples $\{x^1, x^2, \dots, x^m\}$ from database
- Sample m noise samples $\{z^1, z^2, \dots, z^m\}$ from a distribution
- Obtaining generated data $\{\tilde{x}^1, \tilde{x}^2, \dots, \tilde{x}^m\}$, $\tilde{x}^i = G(z^i)$
- Update discriminator parameters θ_d to maximize
  - $\tilde{V} = \frac{1}{m}\sum_{i=1}^{m} log D(x^i) + \frac{1}{m}\sum_{i=1}^{m} log\left(1 - D(\tilde{x}^i)\right)$
  - $\theta_d \leftarrow \theta_d + \eta \nabla \tilde{V}(\theta_d)$

Learning G

- Sample m noise samples$\{z^1, z^2, \dots, z^m\}$ from a distribution
- Update generator parameters θ_g to maximize
  - $\tilde{V} = \frac{1}{m}\sum_{i=1}^{m} log\left(D\left(G(z^i)\right)\right)$
  - $\theta_g \leftarrow \theta_g - \eta \nabla \tilde{V}(\theta_g)$

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

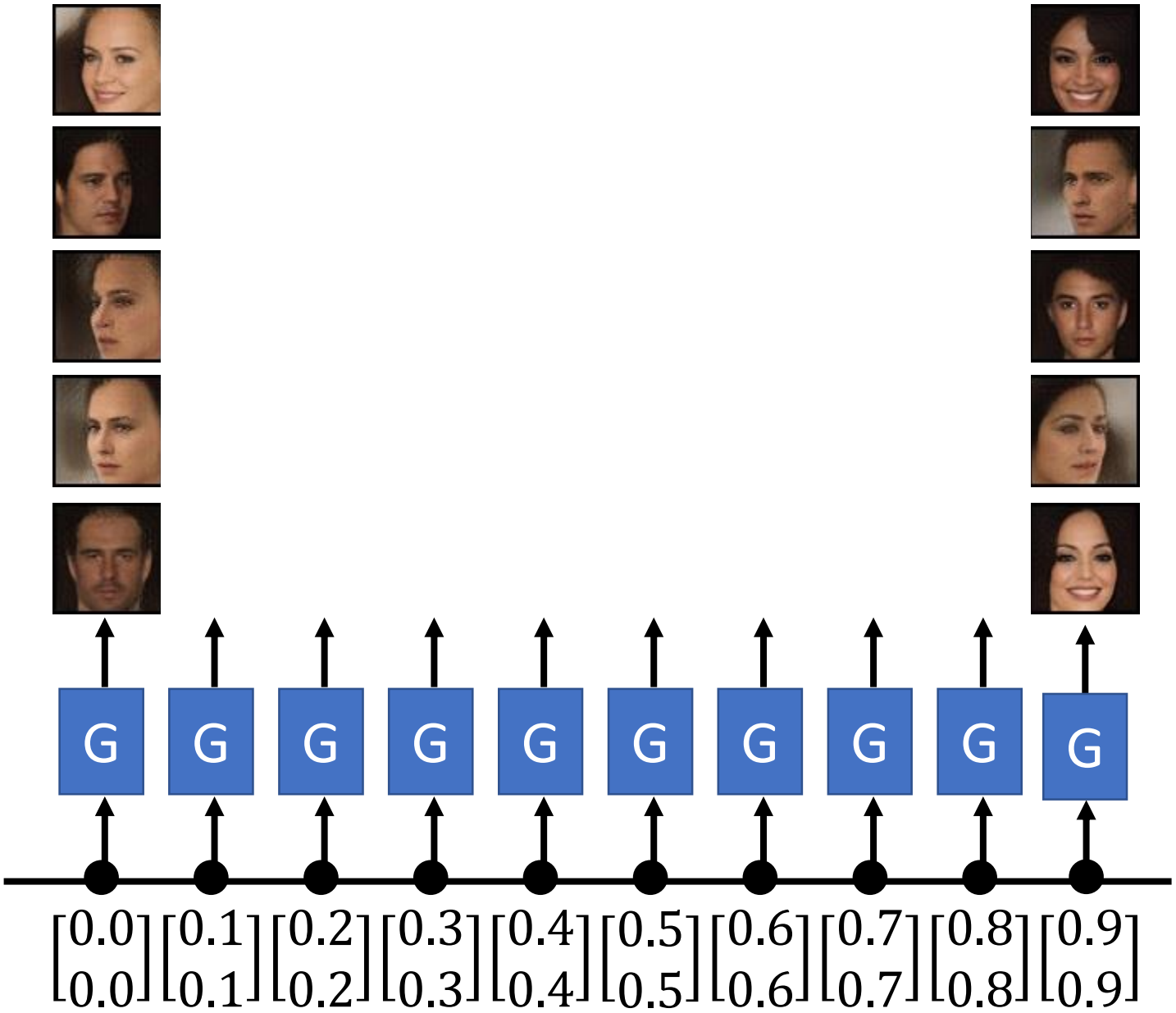
20,000 updates

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



感謝陳柏文同學提供實驗結果

Outline

Basic Idea of GAN

GAN as structured learning

Can Generator learn by itself?

Can Discriminator generate?

A little bit theory

Structured Learning

Machine learning is to find a function f

$$f : X \rightarrow Y$$

Regression: output a scalar

Classification: output a "class" (one-hot vector)

Class 1 Class 2 Class 3

Structured Learning/Prediction: output a sequence, a matrix, a graph, a tree ……

Output is composed of components with dependency

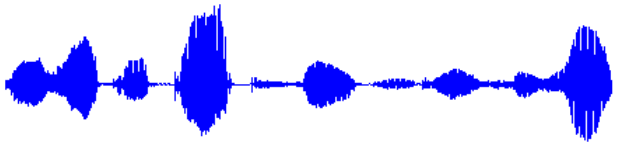
Output Sequence $f: X \rightarrow Y$

Machine Translation

X：“機器學習及其深層與結構化”
(sentence of language 1)

Y：“Machine learning and having it deep and structured”
(sentence of language 2)

Speech Recognition

X： 
(speech)

Y：感謝大家來上課”
(transcription)

Chat-bot

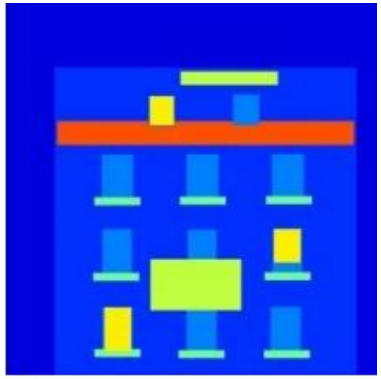
X： “How are you?”
(what a user says)

Y： “I’m fine.”
(response of machine)

Output Matrix

$$f : X \rightarrow Y$$

Image to Image

X : 

Y : 

Colorization:

Ref: https://arxiv.org/pdf/1611.07004v1.pdf

Text to Image

X : "this white and yellow flower have thin white petals and a round yellow stamen"

Y : 

ref: https://arxiv.org/pdf/1605.05396.pdf

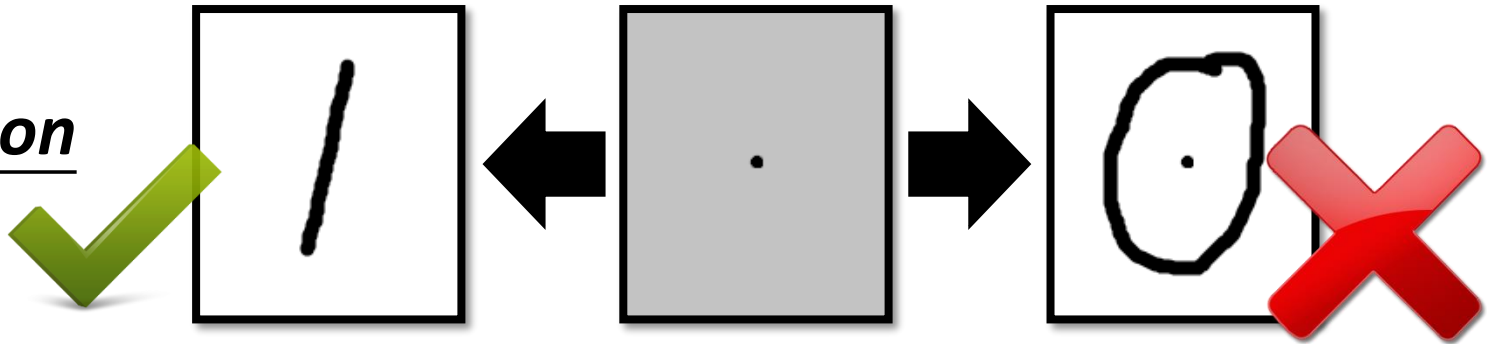
Why Structured Learning Challenging?

- **One-shot/Zero-shot Learning**:
  - In classification, each class has some examples.
  - In structured learning,
    - If you consider each possible output as a "class" ……
    - Since the output space is huge, most "classes" do not have any training data.
    - Machine has to create new stuff during testing.
    - Need more intelligence

Why Structured Learning Challenging?

- Machine has to learn to do ***planning***
  - Machine generates objects component-by-component, but it should have a big picture in its mind.
  - Because the output components have dependency, they should be considered globally.

Image Generation



Sentence Generation

這個婆娘不是人

九天玄女下凡塵

Structured Learning Approach

Generator

Learn to generate the object at the component level

Discriminator

Evaluating the whole object, and find the best one

Outline

Basic Idea of GAN

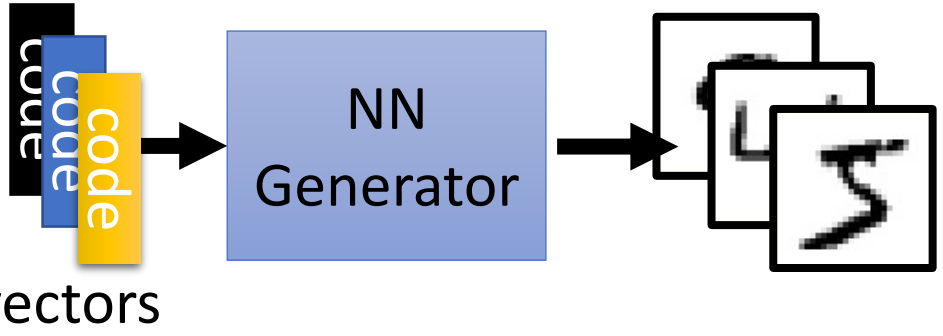
GAN as structured learning

Can Generator learn by itself?

Can Discriminator generate?

A little bit theory

Generator

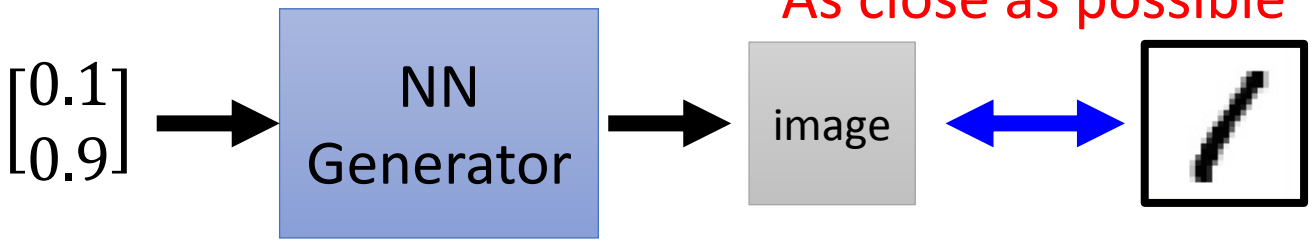


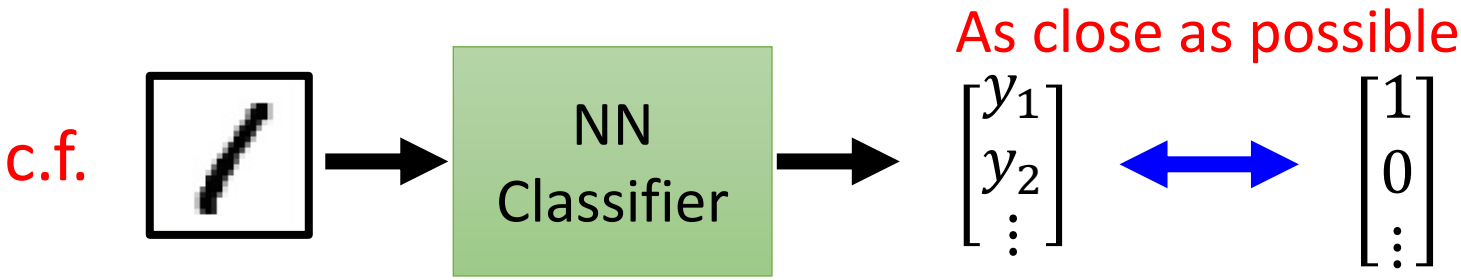
vectors

code: $\begin{bmatrix} 0.1 \\ -0.5 \end{bmatrix}$ $\begin{bmatrix} 0.1 \\ 0.9 \end{bmatrix}$ $\begin{bmatrix} 0.2 \\ -0.1 \end{bmatrix}$ $\begin{bmatrix} 0.3 \\ 0.2 \end{bmatrix}$

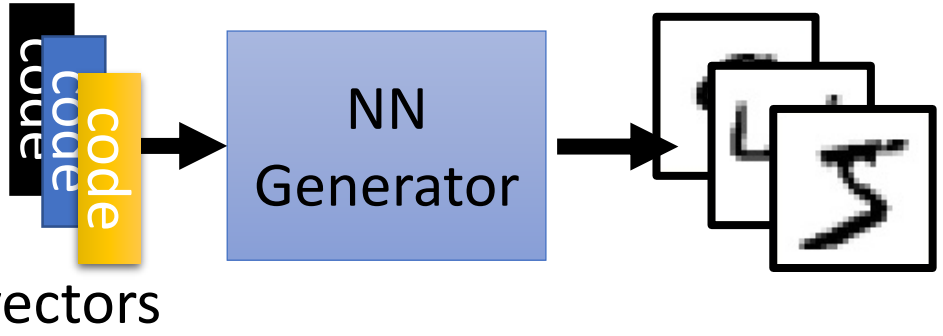
(where does they come from?)

Image:





Generator



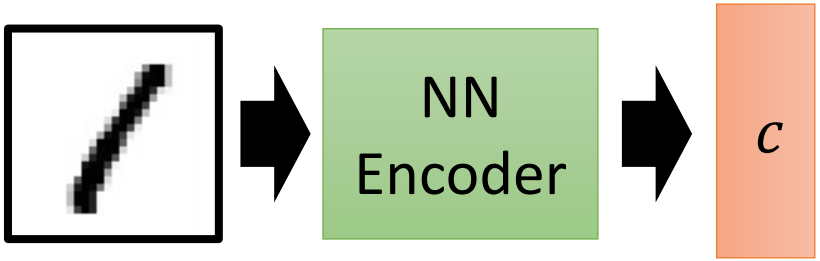
vectors

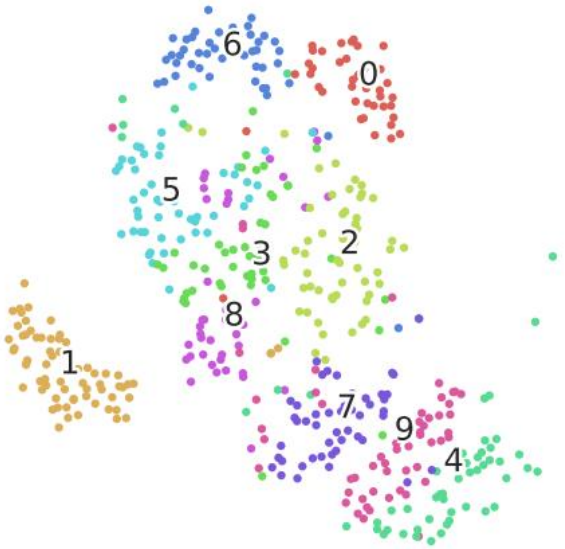
code:
(where does they come from?)

$\begin{bmatrix} 0.1 \\ -0.5 \end{bmatrix}$ $\begin{bmatrix} 0.1 \\ 0.9 \end{bmatrix}$ $\begin{bmatrix} 0.2 \\ -0.1 \end{bmatrix}$ $\begin{bmatrix} 0.3 \\ 0.2 \end{bmatrix}$

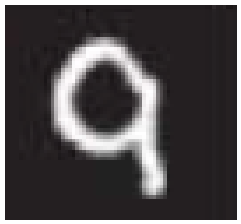
Image:

Encoder in auto-encoder provides the code ☺





Auto-encoder



28 X 28 = 784

NN Encoder

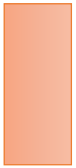
Low dimension

code

Compact representation of the input object

Learn together

code



NN Decoder

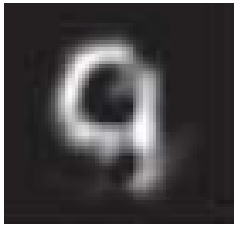
Can reconstruct the original object

Trainable

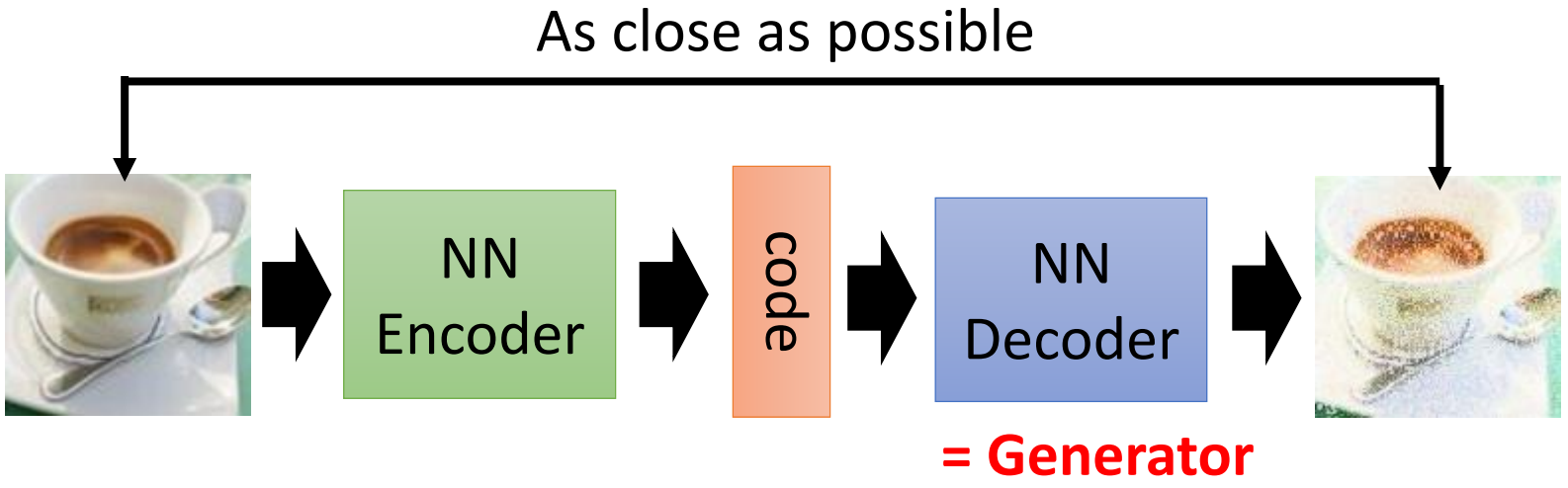
NN Encoder

c

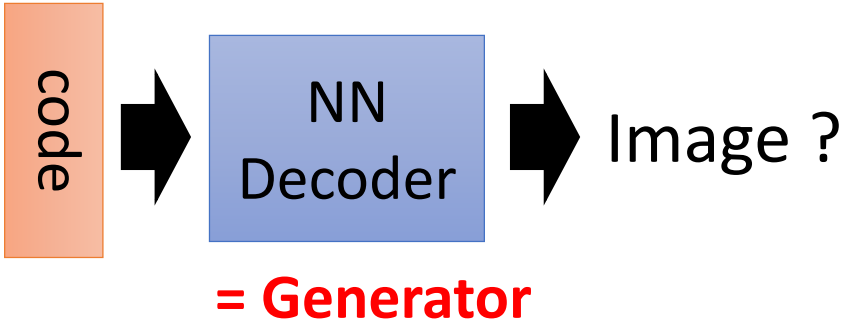
NN Decoder



Auto-encoder

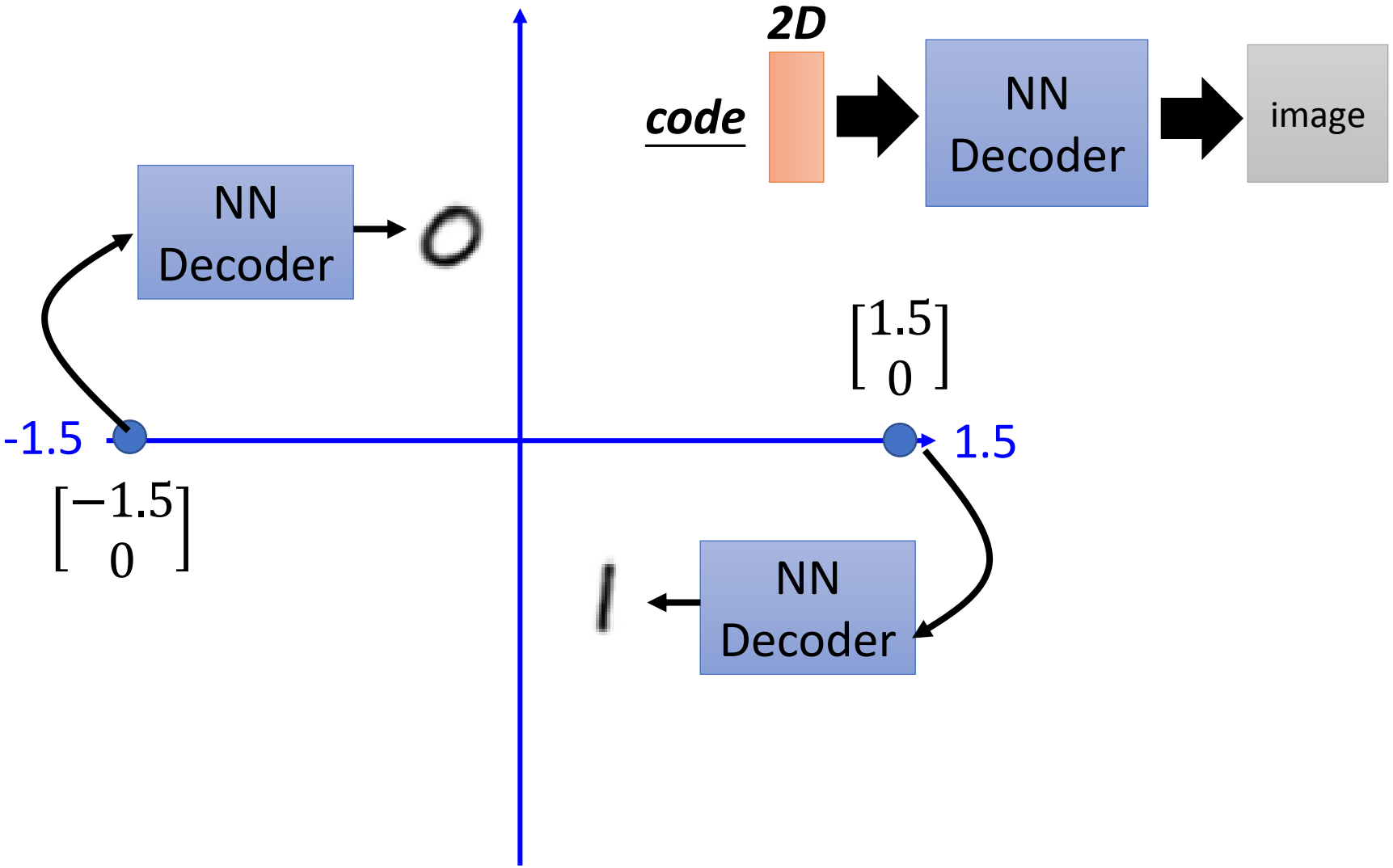


Randomly generate a vector as code



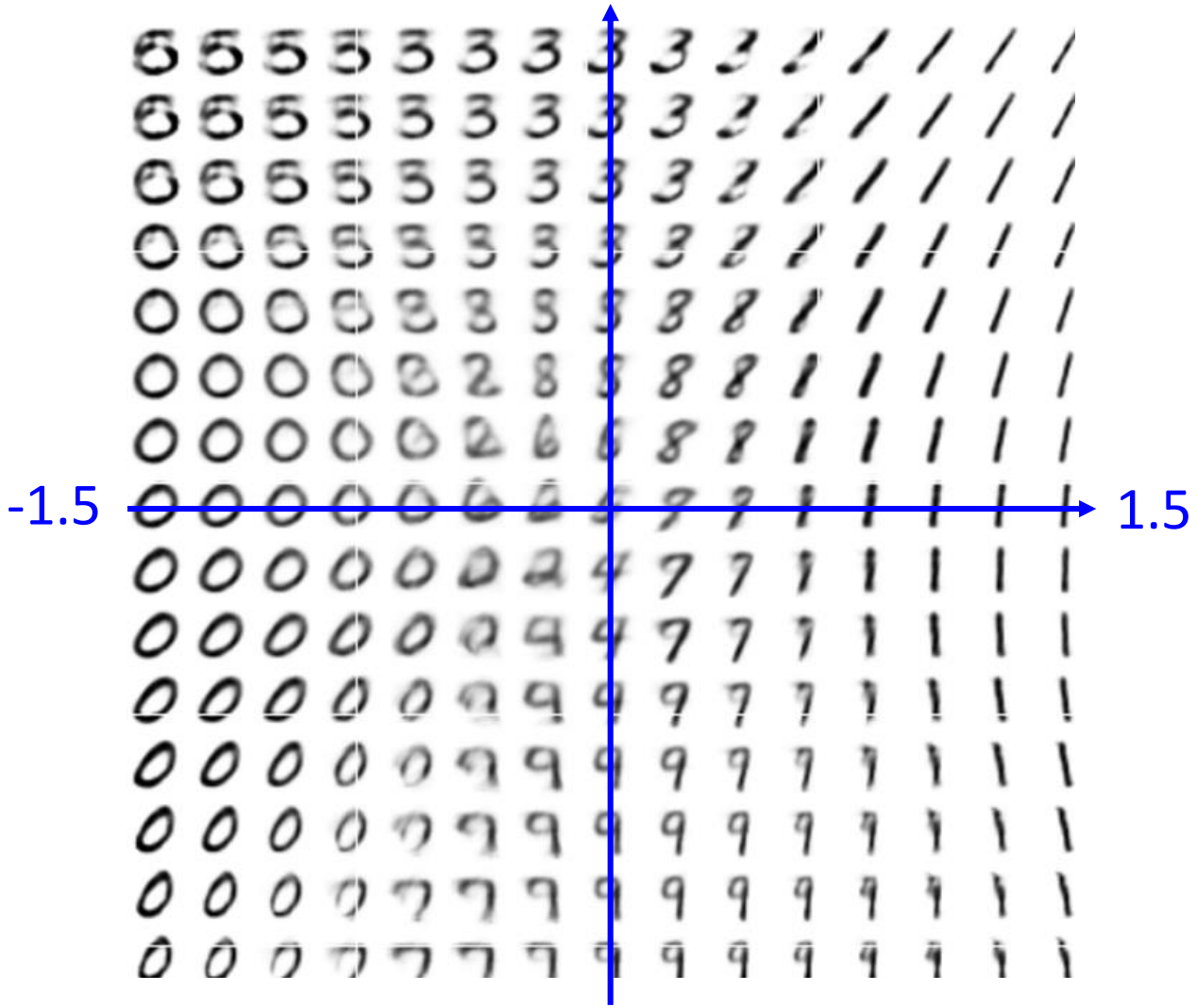
(real examples)

Auto-encoder

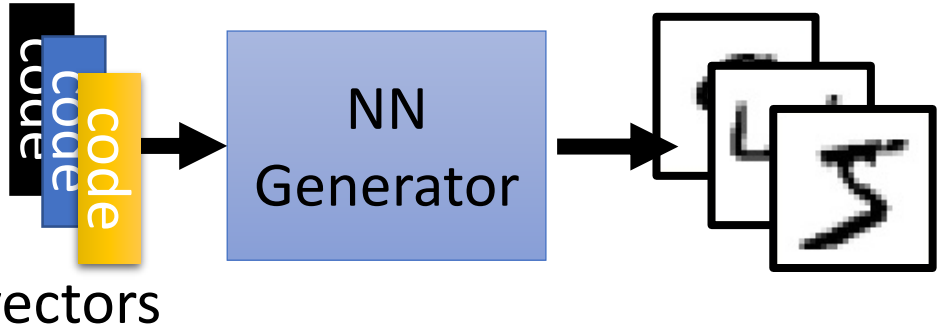


(real examples)

Auto-encoder



Auto-encoder

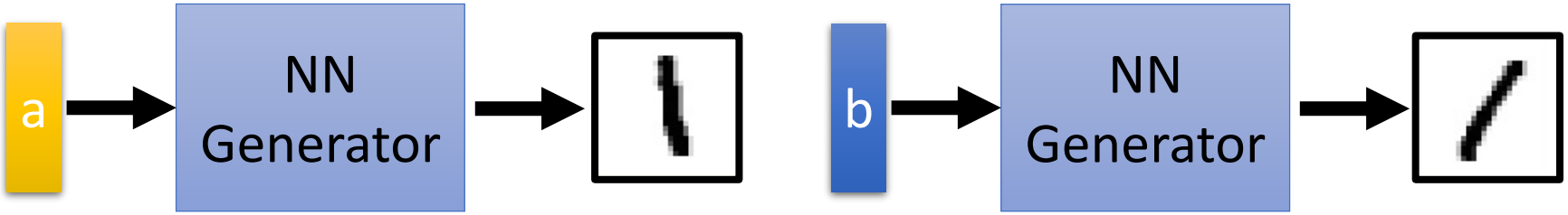


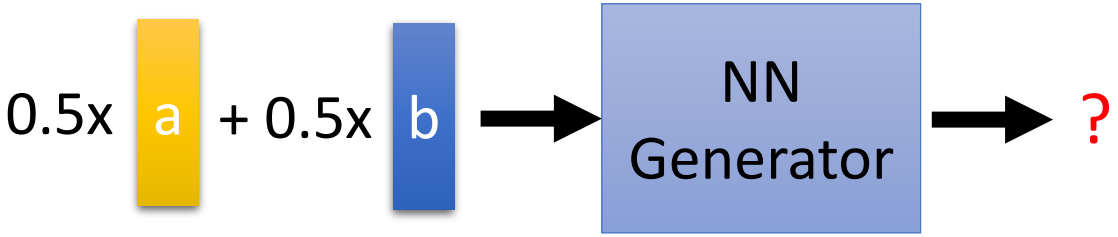
vectors

code:
(where does them come from?)

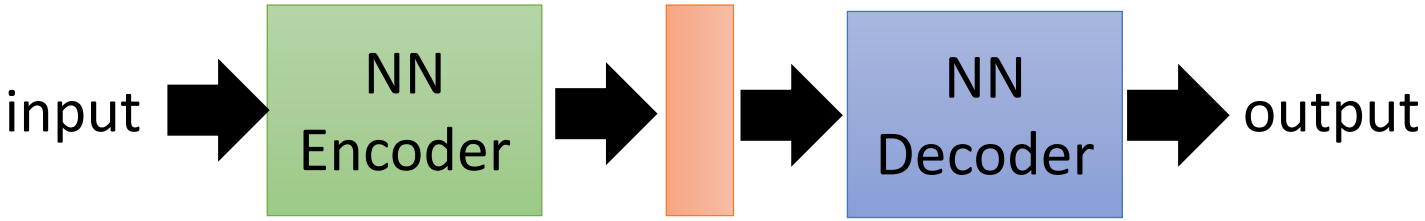
$\begin{bmatrix} 0.1 \\ -0.5 \end{bmatrix}$ $\begin{bmatrix} 0.1 \\ 0.9 \end{bmatrix}$ $\begin{bmatrix} 0.2 \\ -0.1 \end{bmatrix}$ $\begin{bmatrix} 0.3 \\ 0.2 \end{bmatrix}$

Image:

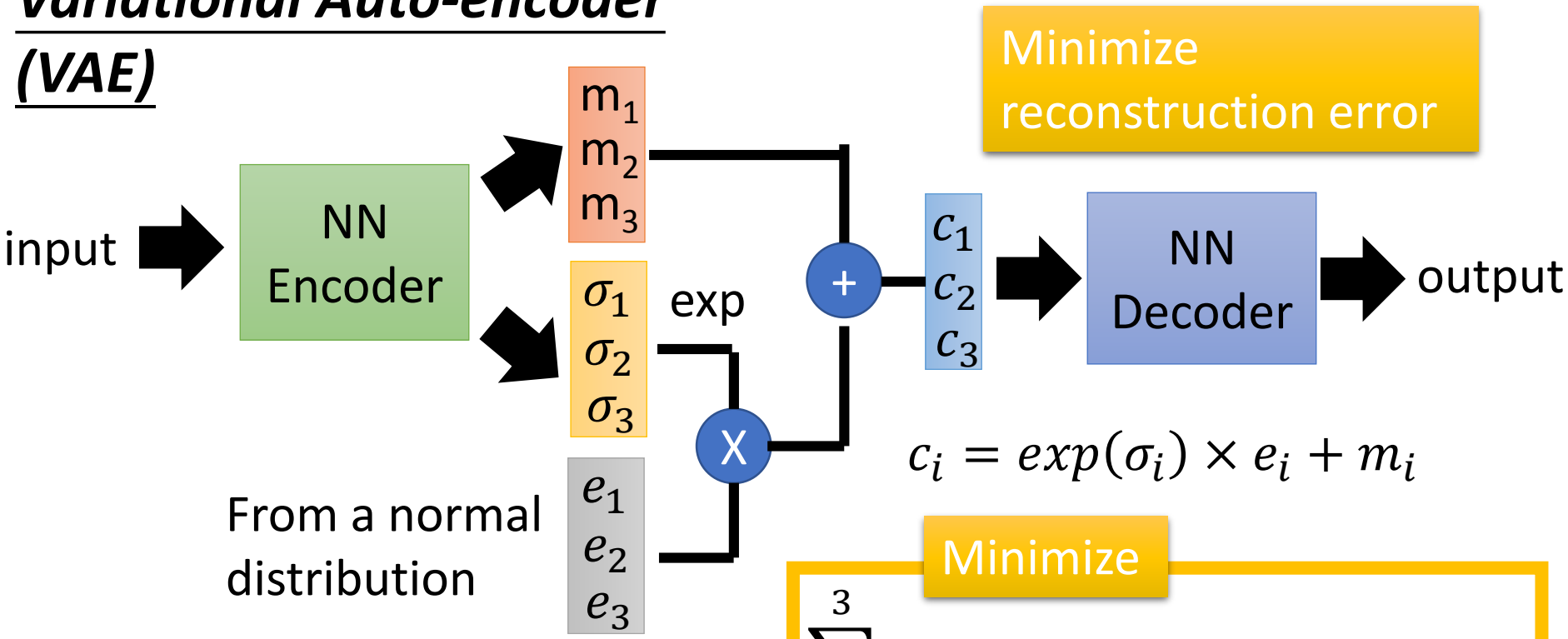




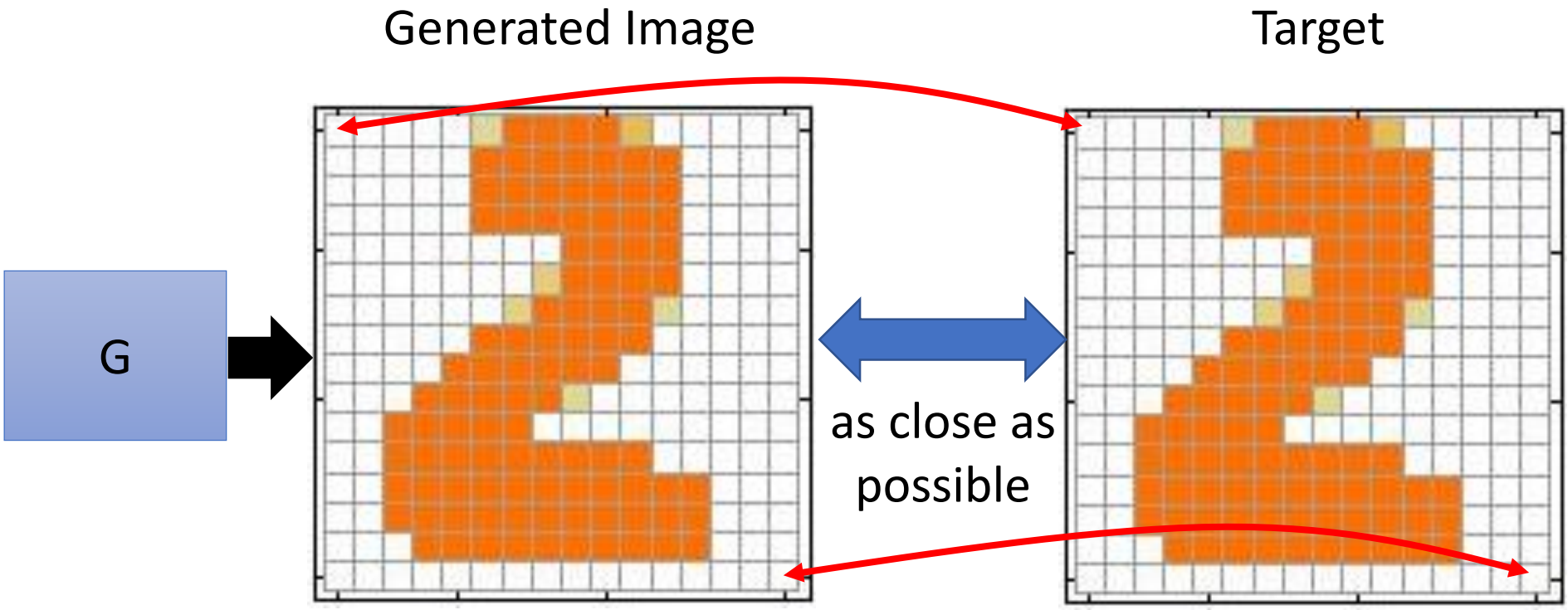
Auto-encoder



Variational Auto-encoder (VAE)



What do we miss?



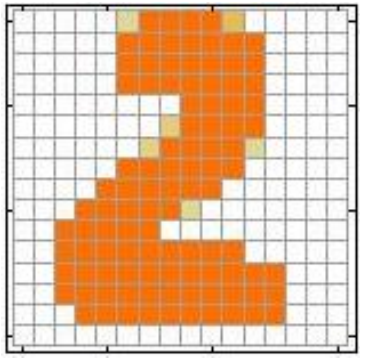
It will be fine if the generator can truly copy the target image.

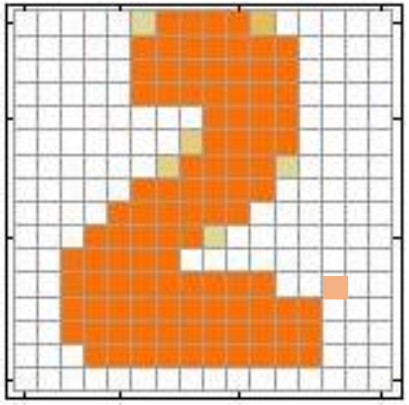
What if the generator makes some mistakes …….

Some mistakes are serious, while some are fine.

What do we miss?

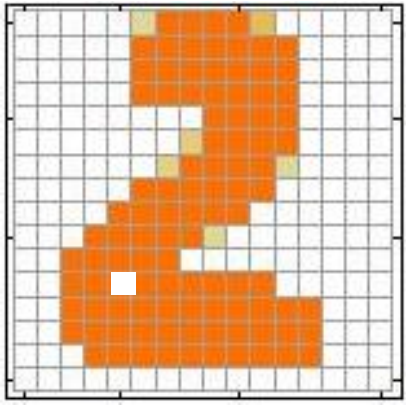
Target





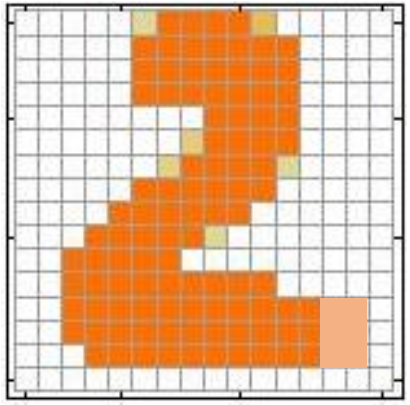
1 pixel error

我覺得不行



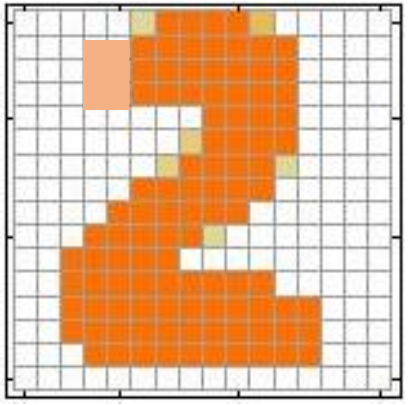
1 pixel error

我覺得不行



6 pixel errors

我覺得其實可以

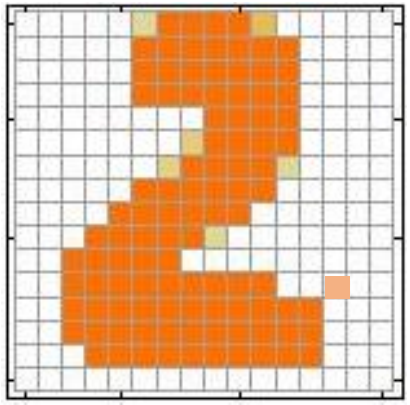


6 pixel errors

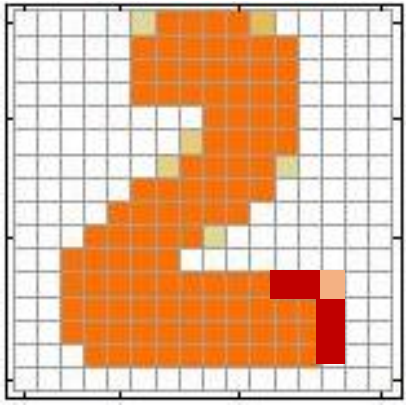
我覺得其實可以

What do we miss?

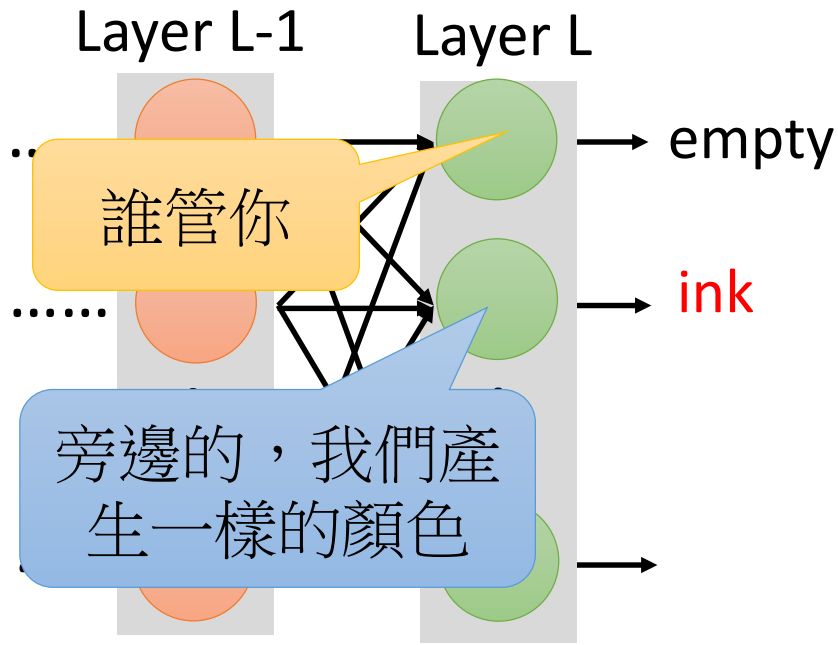
Each neural in output layer corresponds to a pixel.



我覺得不行



我覺得其實可以



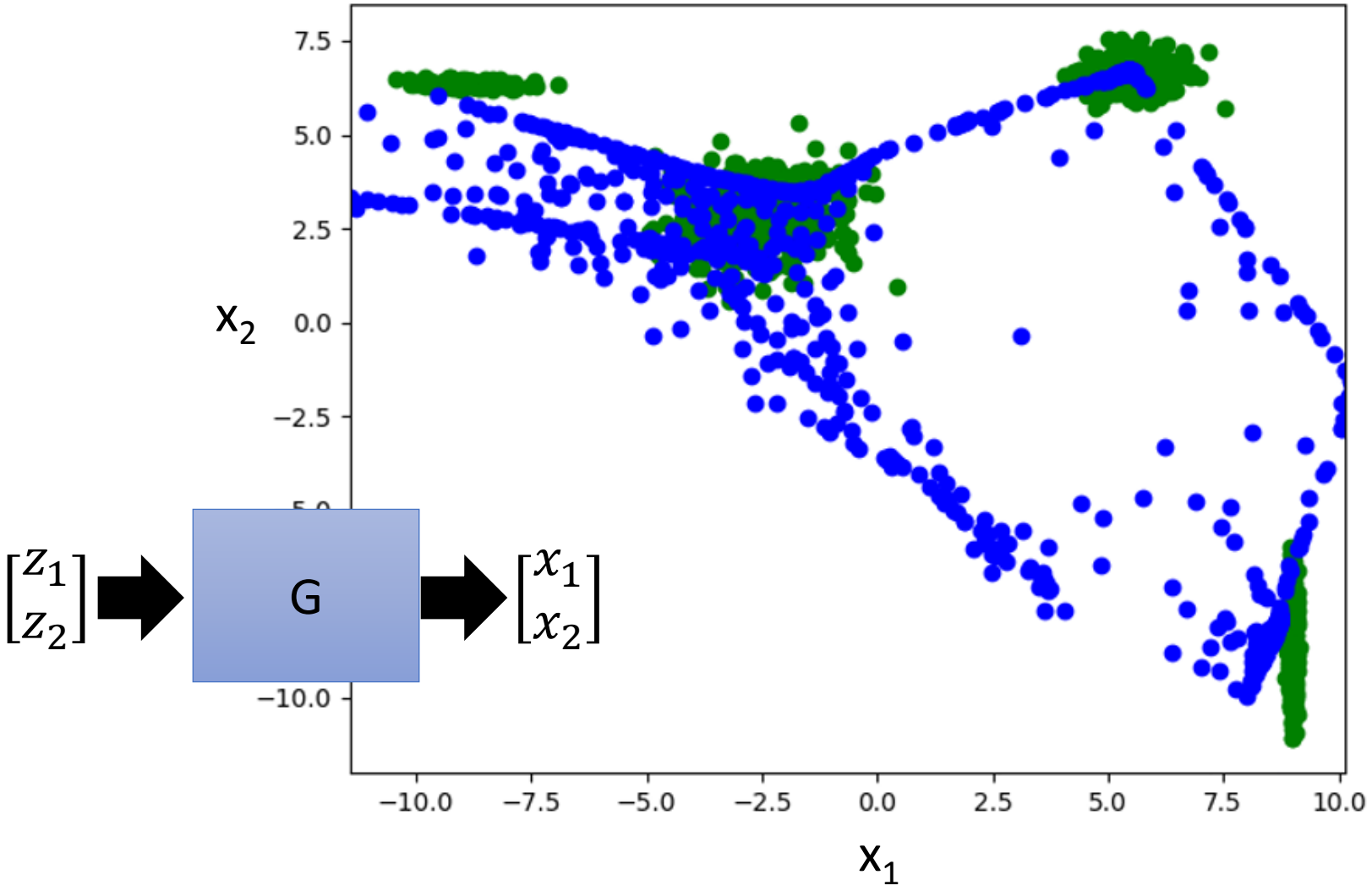
The relation between the components are critical.

Although highly correlated, they cannot influence each other.

Need deep structure to catch the relation between components.

感謝 黃淞楓 同學提供結果

(Variational) Auto-encoder



Outline

Basic Idea of GAN

GAN as structured learning

Can Generator learn by itself?

Can Discriminator generate?

A little bit theory

Discriminator

Evaluation function, Potential Function, Energy Function …

- Discriminator is a function D (network, can deep)

$$\mathrm{D}: X \rightarrow \mathrm{R}$$

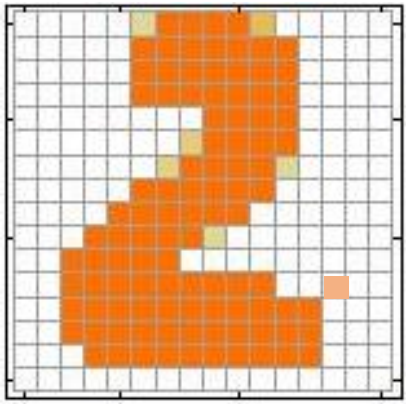
  - Input x: an object x (e.g. an image)
  - Output D(x): scalar which represents how "good" an object x is

Can we use the discriminator to generate objects?

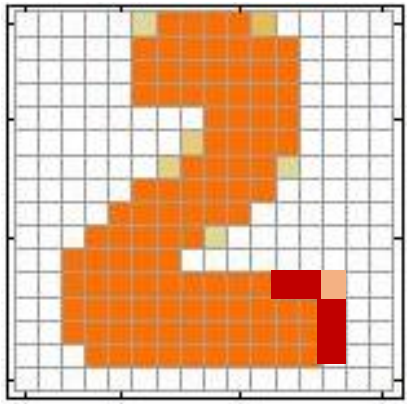
Yes.

Discriminator

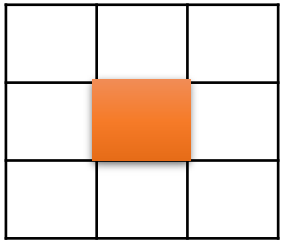
- It is easier to catch the relation between the components by top-down evaluation.



我覺得不行



我覺得其實 OK



This CNN filter is good enough.

Discriminator

- Suppose we already have a good discriminator D(x) …

Inference

- Generate object $\tilde{x}$ that

$$\tilde{x} = \arg\max_{x \in X} D(x)$$

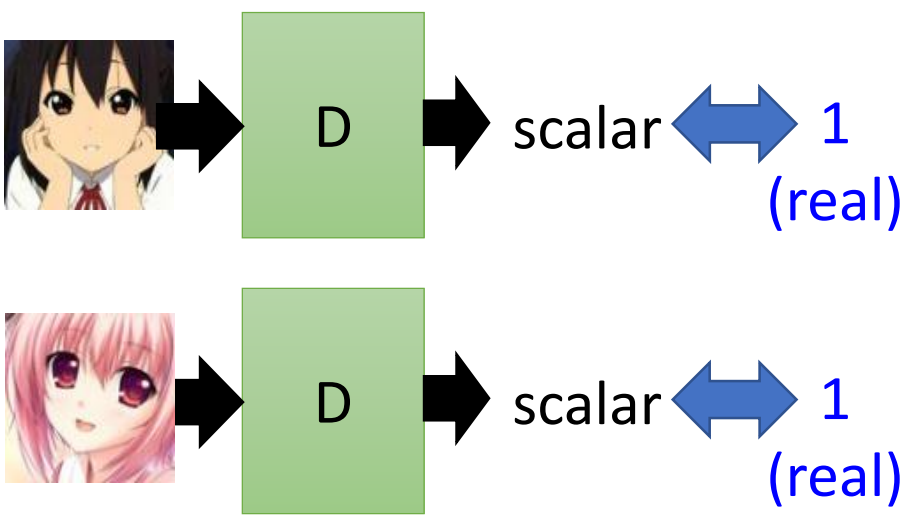
Enumerate all possible x !!!

It is feasible ???

How to learn the discriminator?

Discriminator - Training

- I have some real images

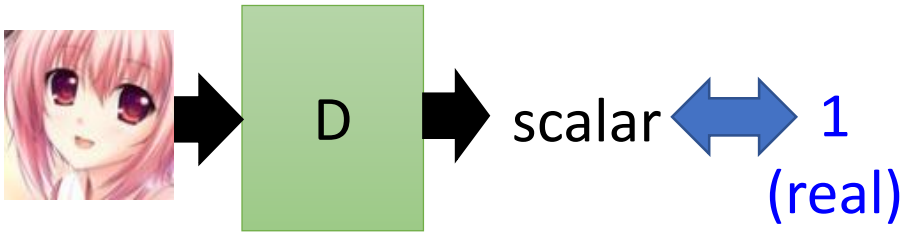


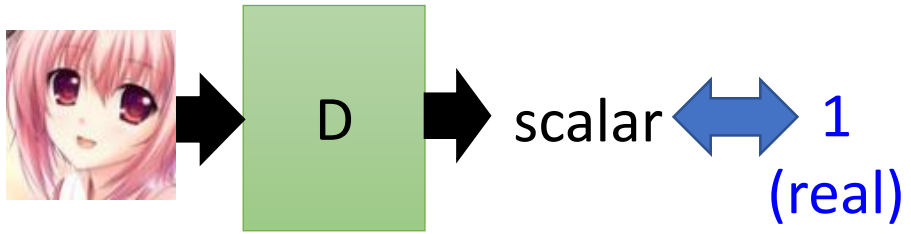
Discriminator only learns to output “1” (real).

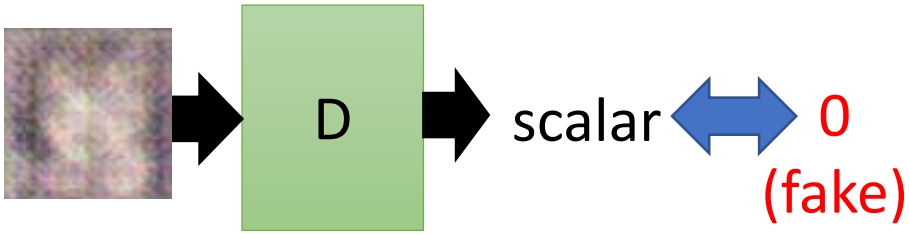
Discriminator training needs some negative examples.

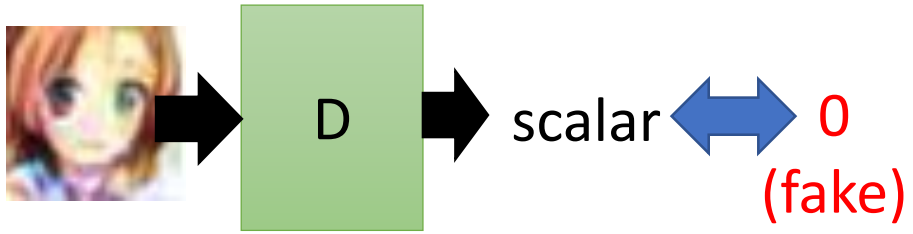
Discriminator - Training

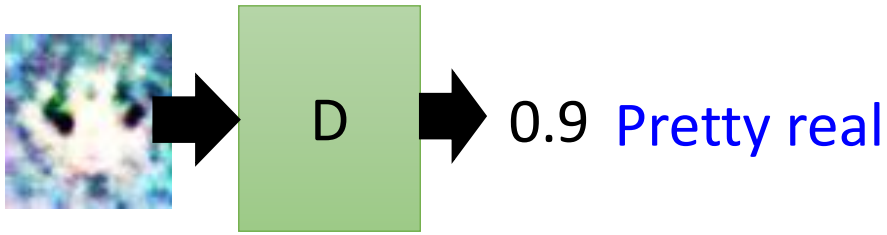
- Negative examples are critical.











How to generate realistic negative examples?

Discriminator - Training

- **General Algorithm**

  - Given a set of positive examples, randomly generate a set of negative examples.

  - In each iteration
    - Learn a discriminator D that can discriminate positive and negative examples.

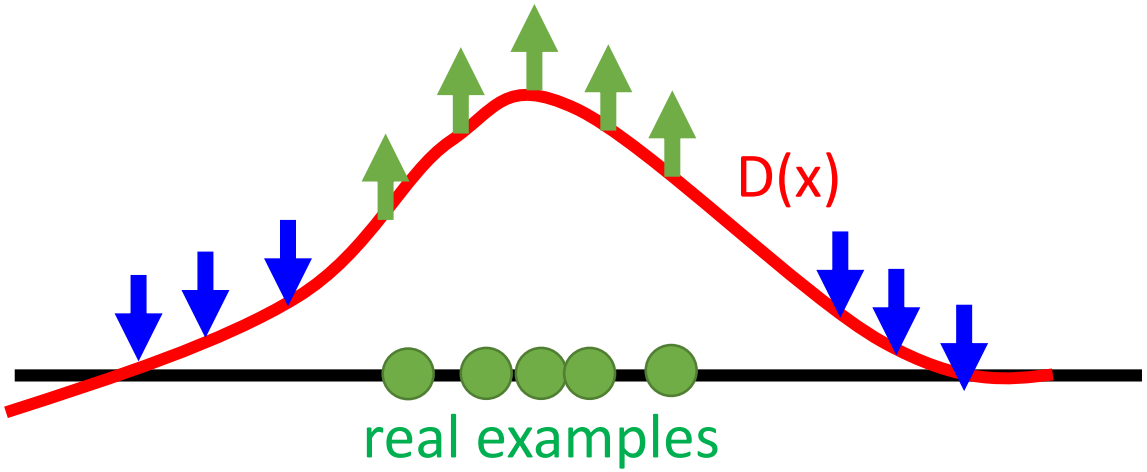

v.s.


D

    - Generate negative examples by discriminator D

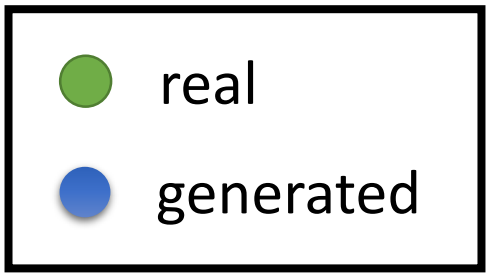
$$\tilde{x} = \arg\max_{x \in X} D(x)$$

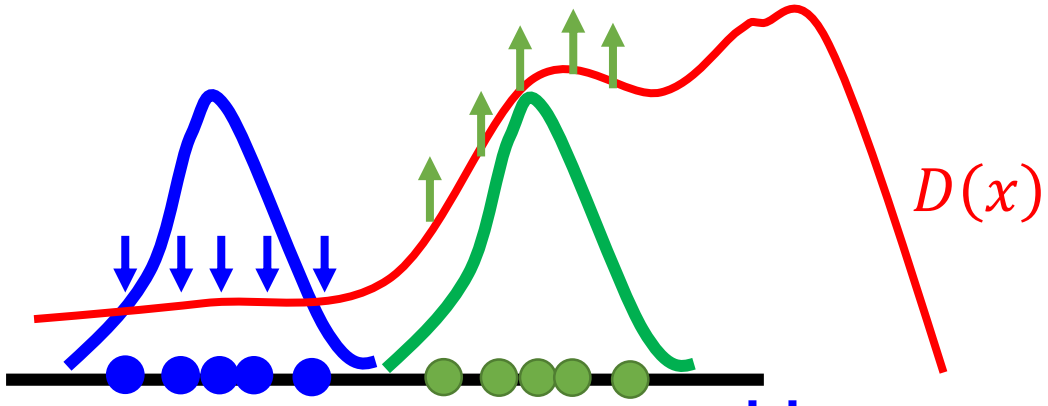
Discriminator - Training

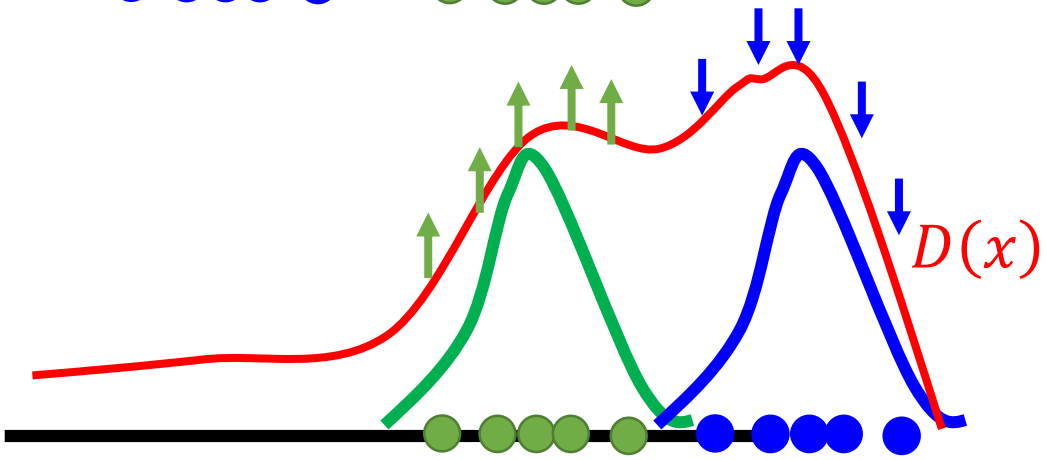


In practice, you cannot decrease all the x other than real examples.

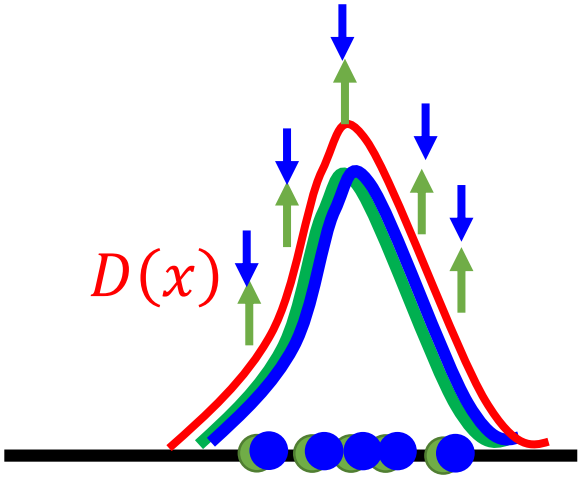
Discriminator - Training

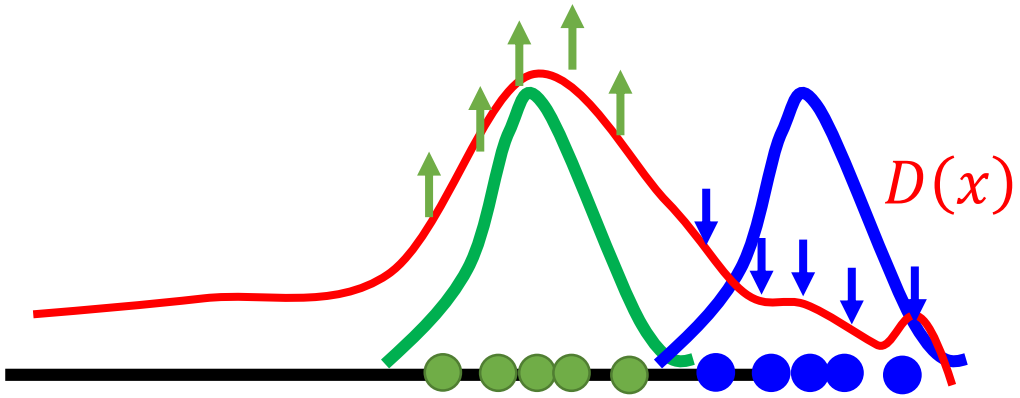


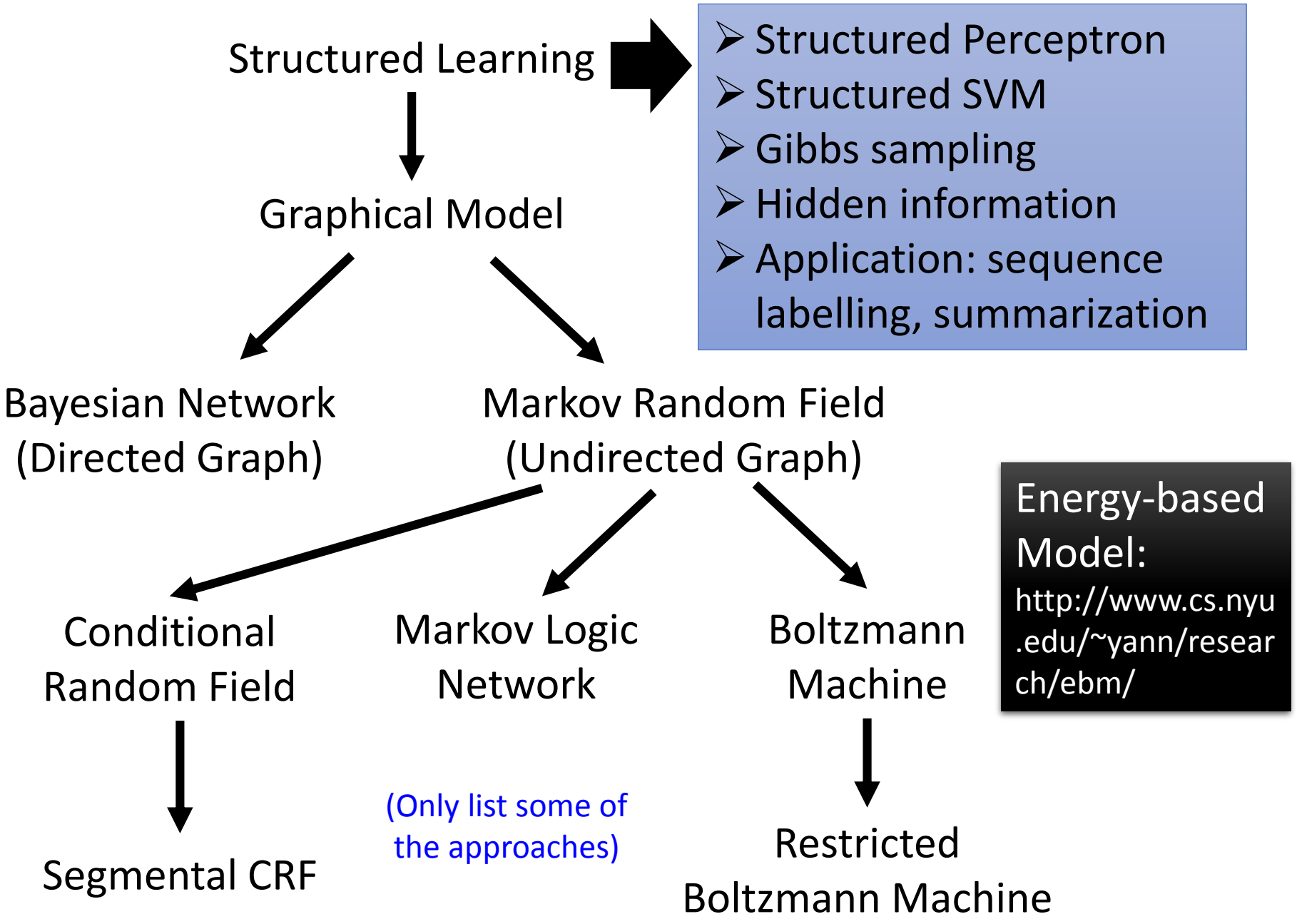




In the end ……






Structured Learning
Graphical Model
Structured Perceptron
Structured SVM
Gibbs sampling
Hidden information
Application: sequence labelling, summarization
Bayesian Network (Directed Graph)
Markov Random Field (Undirected Graph)
Conditional Random Field
Markov Logic Network
Boltzmann Machine
Energy-based Model: http://www.cs.nyu.edu/~yann/research/ebm/
Segmental CRF
(Only list some of the approaches)
Restricted Boltzmann Machine

Generator v.s. Discriminator

- ***Generator***

- Pros:
  - Easy to generate even with deep model
- Cons:
  - Imitate the appearance
  - Hard to learn the correlation between components

- ***Discriminator***

- Pros:
  - Considering the big picture
- Cons:
  - Generation is not always feasible
    - Especially when your model is deep
  - How to do negative sampling?

Generator + Discriminator

- **General Algorithm**

  - Given a set of positive examples, randomly generate a set of negative examples.

  - In each iteration
    - Learn a discriminator D that can discriminate positive and negative examples.

v.s.

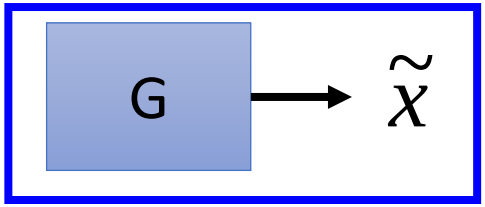
D

    - Generate negative examples by discriminator D

G $\rightarrow \tilde{x}$ = $\tilde{x} = \arg\max_{x \in X} D(x)$

Benefit of GAN

- From Discriminator's point of view
  - Using generator to generate negative samples



=

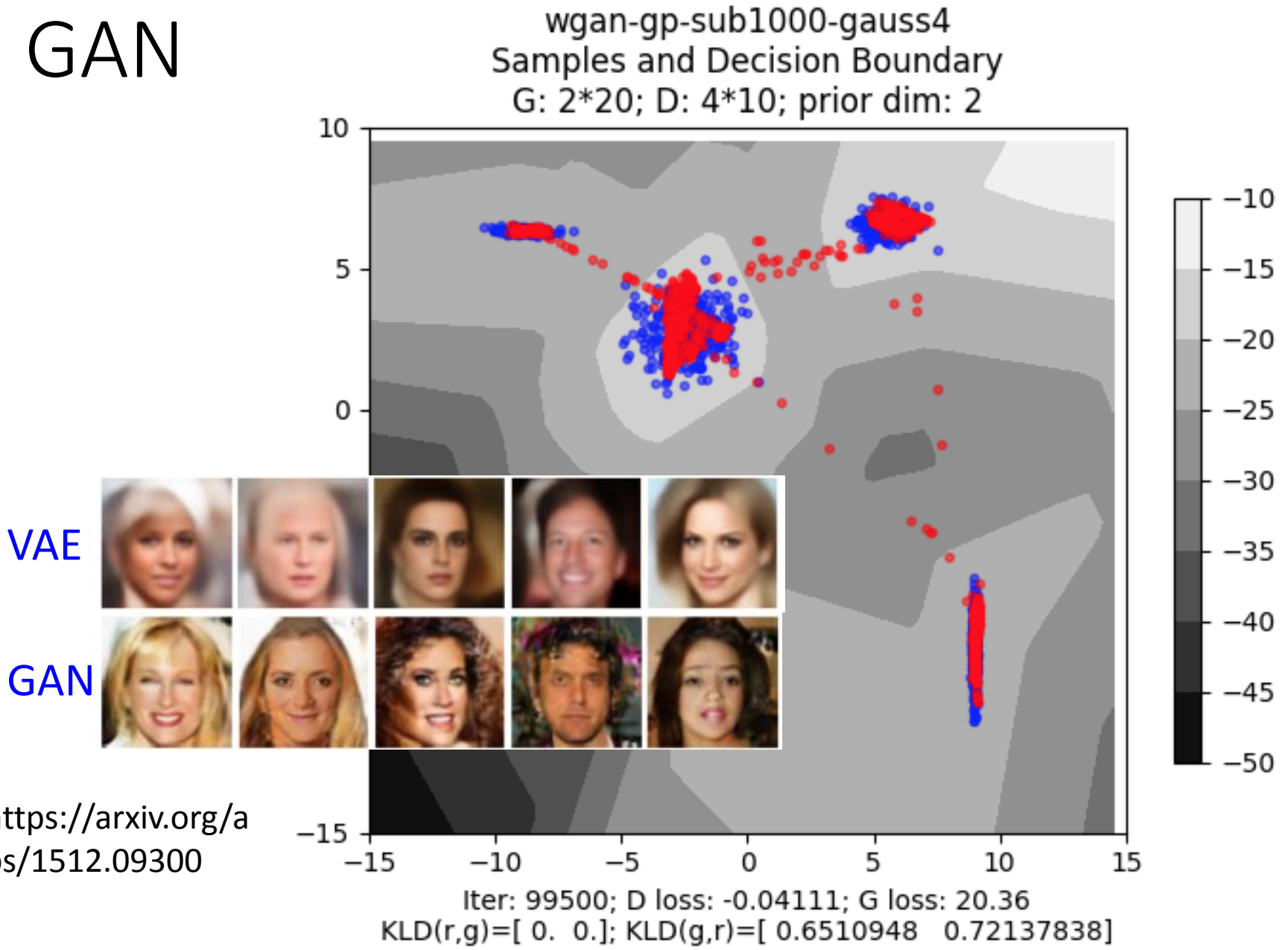
$$\tilde{x} = \arg\max_{x \in X} D(x)$$

efficient

- From Generator's point of view
  - Still generate the object component-by-component
  - But it is learned from the discriminator with global view.

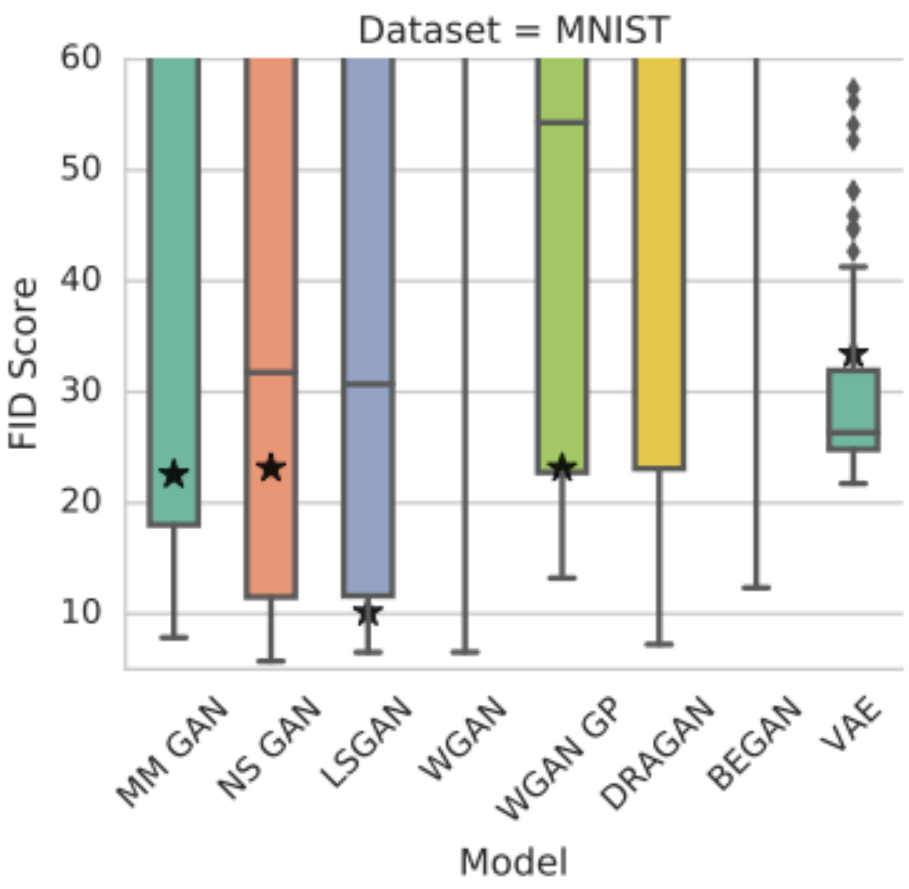
感謝 段逸林 同學提供結果

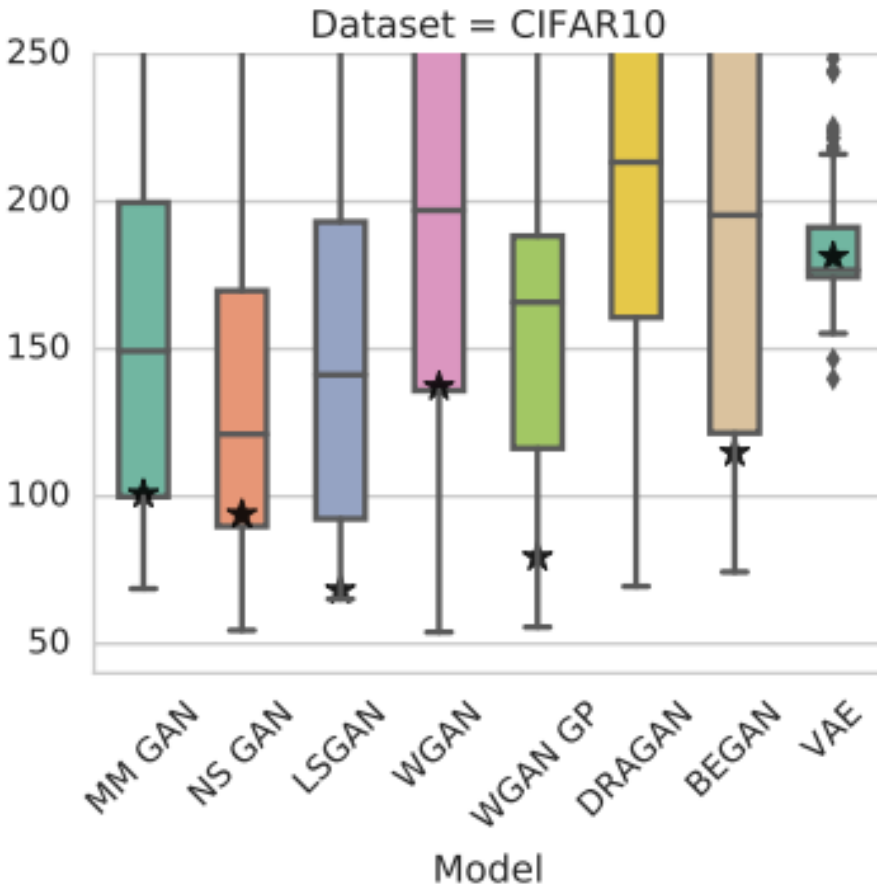
GAN



https://arxiv.org/abs/1512.09300

[Mario Lucic, et al. arXiv, 2017]





FID[Martin Heusel, et al., NIPS, 2017]: Smaller is better

Next Time

- Preview
  - https://youtu.be/0CKeqXl5IY0
  - https://youtu.be/KSN4QYgAtao

Outline

Basic Idea of GAN

GAN as structured learning

Can Generator learn by itself?

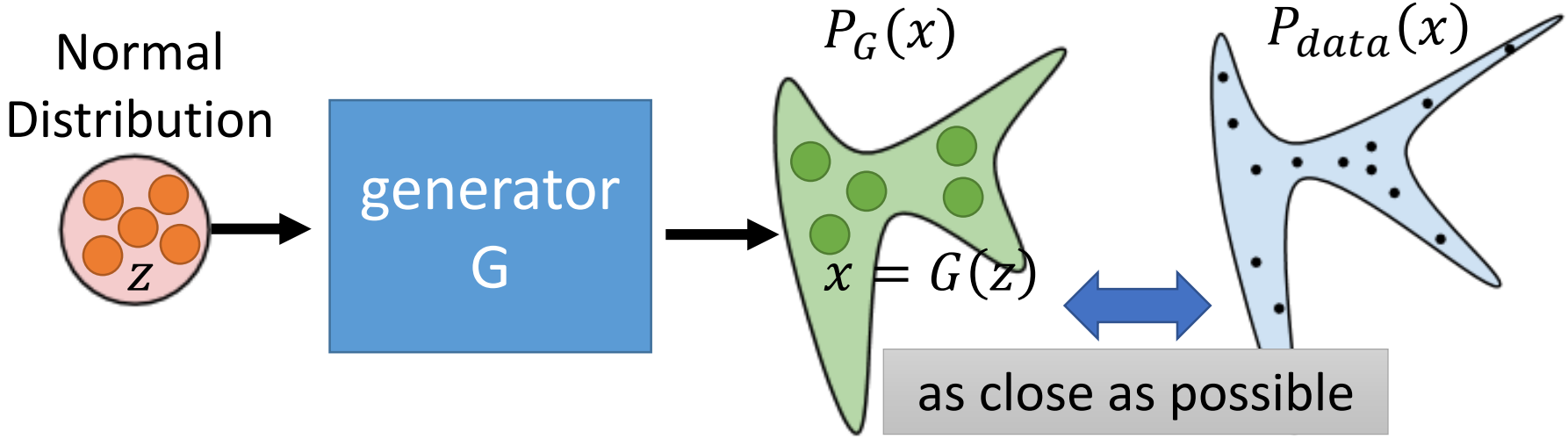
Can Discriminator generate?

A little bit theory

Generator

x: an image (a high-dimensional vector)

- A generator G is a network. The network defines a probability distribution P_G



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

Divergence between distributions P_G and P_{data}

How to compute the divergence?

Discriminator

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

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

Database

sample

Sampling from P_{data}

sample from normal

G

Sampling from P_G

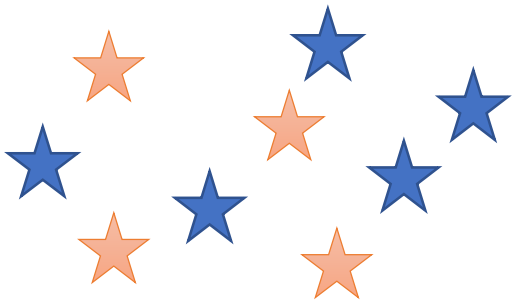
Discriminator

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

: data sampled from P_{data}

: data sampled from P_G

Using the example objective function is exactly the same as training a binary classifier.



train

Discriminator

Example Objective Function for D

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

(G is fixed)

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

The maximum objective value is related to JS divergence.

Discriminator

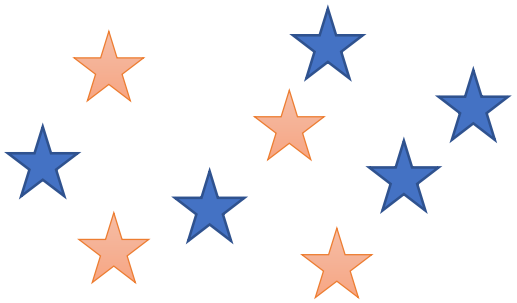
$$G^* = arg \min_G Div(P_G, P_{data})$$

: data sampled from P_{data}

: data sampled from P_G

Training:

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

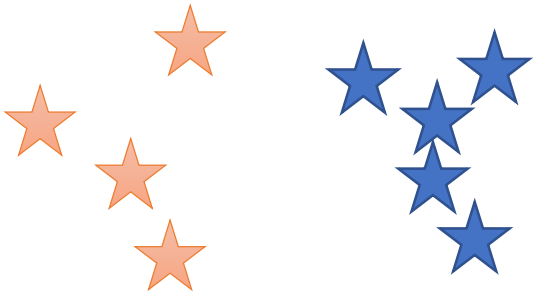


small divergence

train

Discriminator

hard to discriminate
(cannot make objective large)



large divergence

train

Discriminator

easy to discriminate

[Goodfellow, et al., NIPS, 2014]

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

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

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	$\frac{1}{2}\int \lvert p(x)-q(x)\rvert \,\mathrm{d}x$	$\frac{1}{2}\lvert u-1\rvert$
Kullback-Leibler	$\int p(x)\log\frac{p(x)}{q(x)}\,\mathrm{d}x$	$u\log u$
Reverse Kullback-Leibler	$\int q(x)\log\frac{q(x)}{p(x)}\,\mathrm{d}x$	$-\log u$
Pearson χ^2	$\int \frac{(q(x)-p(x))^2}{p(x)}\,\mathrm{d}x$	$(u-1)^2$
Neyman χ^2	$\int \frac{(p(x)-q(x))^2}{q(x)}\,\mathrm{d}x$	$\frac{(1-u)^2}{u}$
Squared Hellinger	$\int \left(\sqrt{p(x)}-\sqrt{q(x)}\right)^2\,\mathrm{d}x$	$\left(\sqrt{u}-1\right)^2$
Jeffrey	$\int (p(x)-q(x))\log\left(\frac{p(x)}{q(x)}\right)\,\mathrm{d}x$	$(u-1)\log u$
Jensen-Shannon	$\frac{1}{2}\int p(x)\log\frac{2p(x)}{p(x)+q(x)}+q(x)\log\frac{2q(x)}{p(x)+q(x)}\,\mathrm{d}x$	$-(u+1)\log\frac{1+u}{2}+u\log u$
Jensen-Shannon-weighted	$\int p(x)\pi\log\frac{p(x)}{\pi p(x)+(1-\pi)q(x)}+(1-\pi)q(x)\log\frac{q(x)}{\pi p(x)+(1-\pi)q(x)}\,\mathrm{d}x$	$\pi u\log u-(1-\pi+\pi u)\log(1-\pi+\pi u)$
GAN	$\int p(x)\log\frac{2p(x)}{p(x)+q(x)}+q(x)\log\frac{2q(x)}{p(x)+q(x)}\,\mathrm{d}x-\log(4)$	$u\log u-(u+1)\log(u+1)$

Using the divergence you like ☺

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	$-\log(1-\exp(t))$

Conditional Generation by GAN

李宏毅

Hung-yi Lee

Text-to-Image

a dog is running 

a bird is flying 

- **Traditional supervised approach**

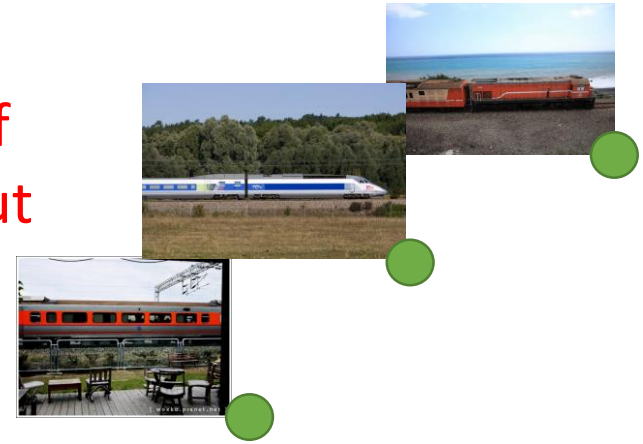
c^1: a dog is running → NN → Image ↔ 

as close as possible

Text: "train"

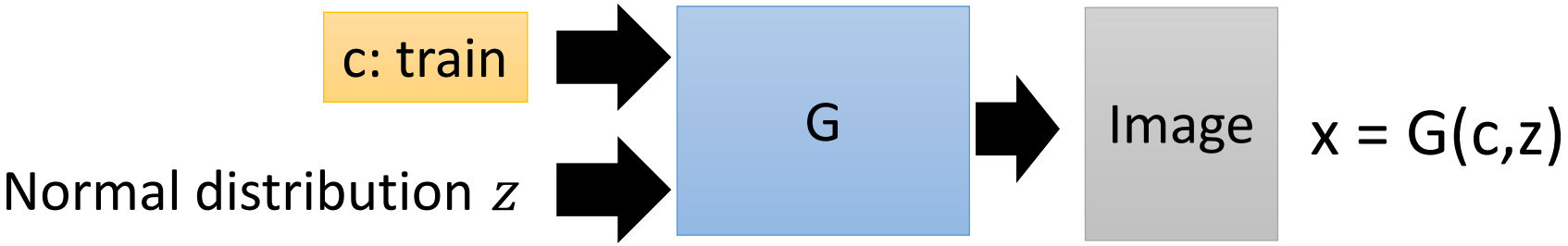
Target of NN output

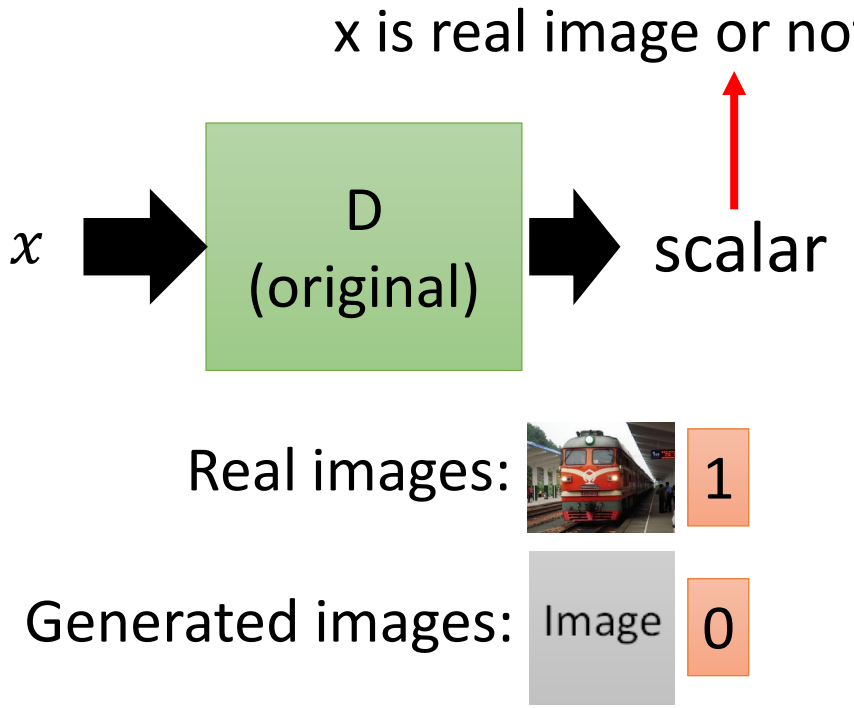
A blurry image!



[Scott Reed, et al, ICML, 2016]

Conditional GAN



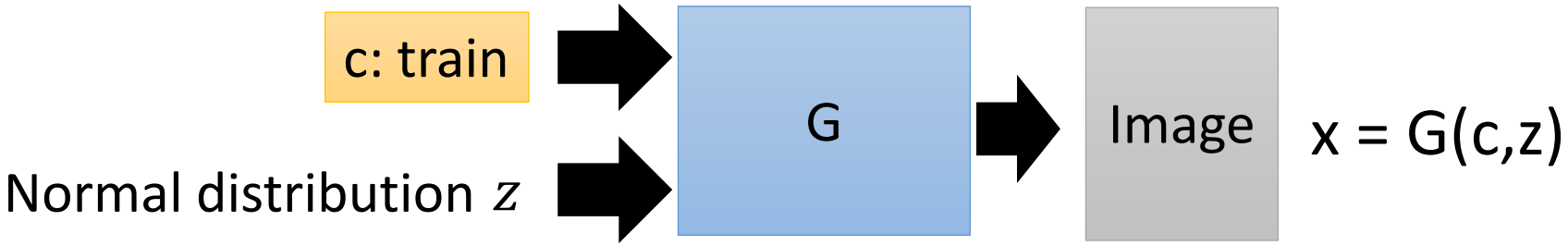


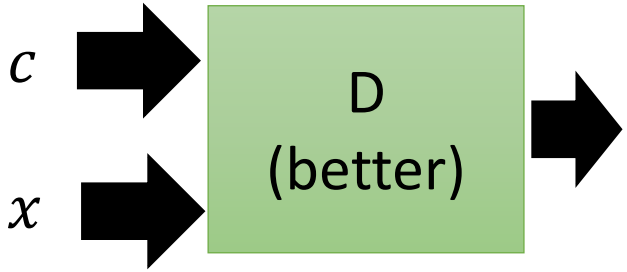
Generator will learn to generate realistic images ….

But completely ignore the input conditions.

[Scott Reed, et al, ICML, 2016]

Conditional GAN





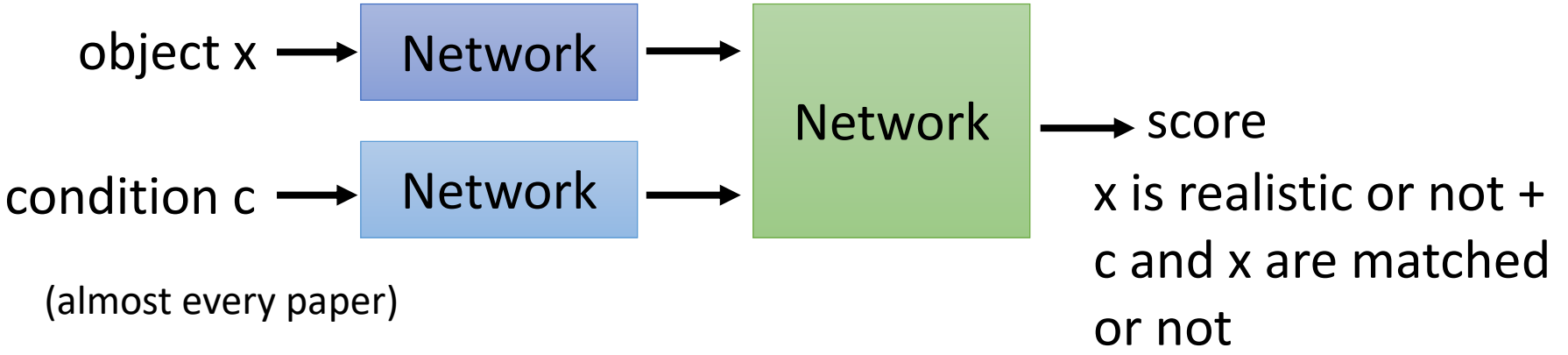
x is realistic or not +
c and x are matched or not

True text-image pairs: (train , ) 1

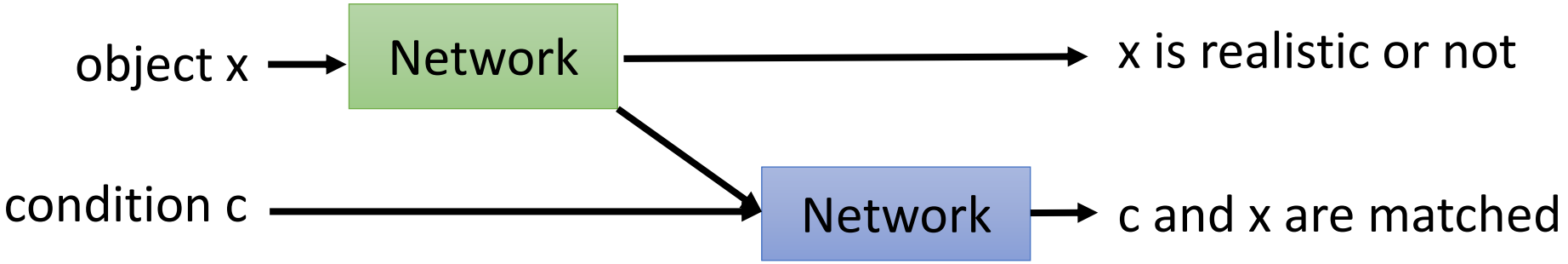
(cat , ) 0

(train , Image) 0

Conditional GAN - Discriminator



(almost every paper)



[Augustus Odena et al., ICML, 2017]
[Takeru Miyato, et al., ICLR, 2018]
[Han Zhang, et al., arXiv, 2017]

Conditional GAN

The images are generated by Yen-Hao Chen, Po-Chun Chien, Jun-Chen Xie, Tsung-Han Wu.

paired data

blue eyes
red hair
short hair

Collecting anime faces and the description of its characteristics

red hair, green eyes

blue hair, red eyes

Stack GAN

Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, Dimitris Metaxas, "StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks", ICCV, 2017

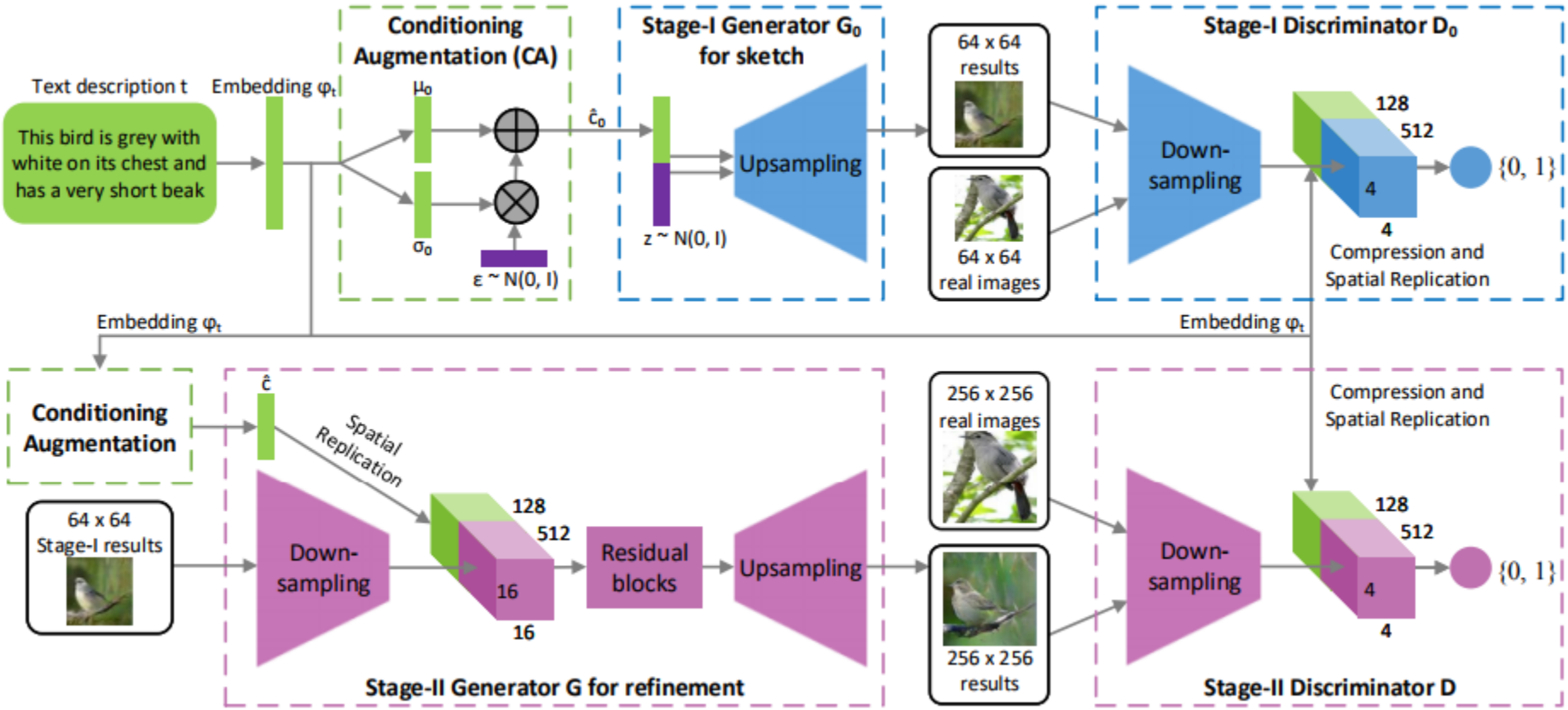
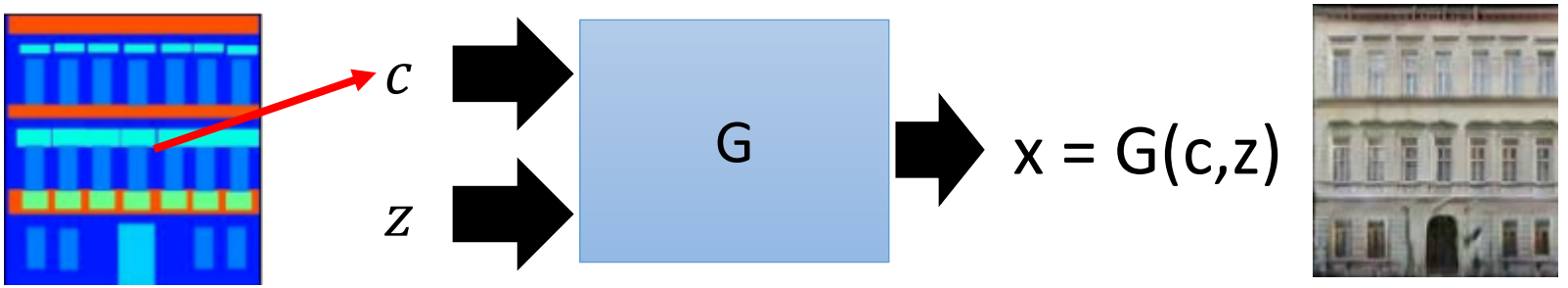
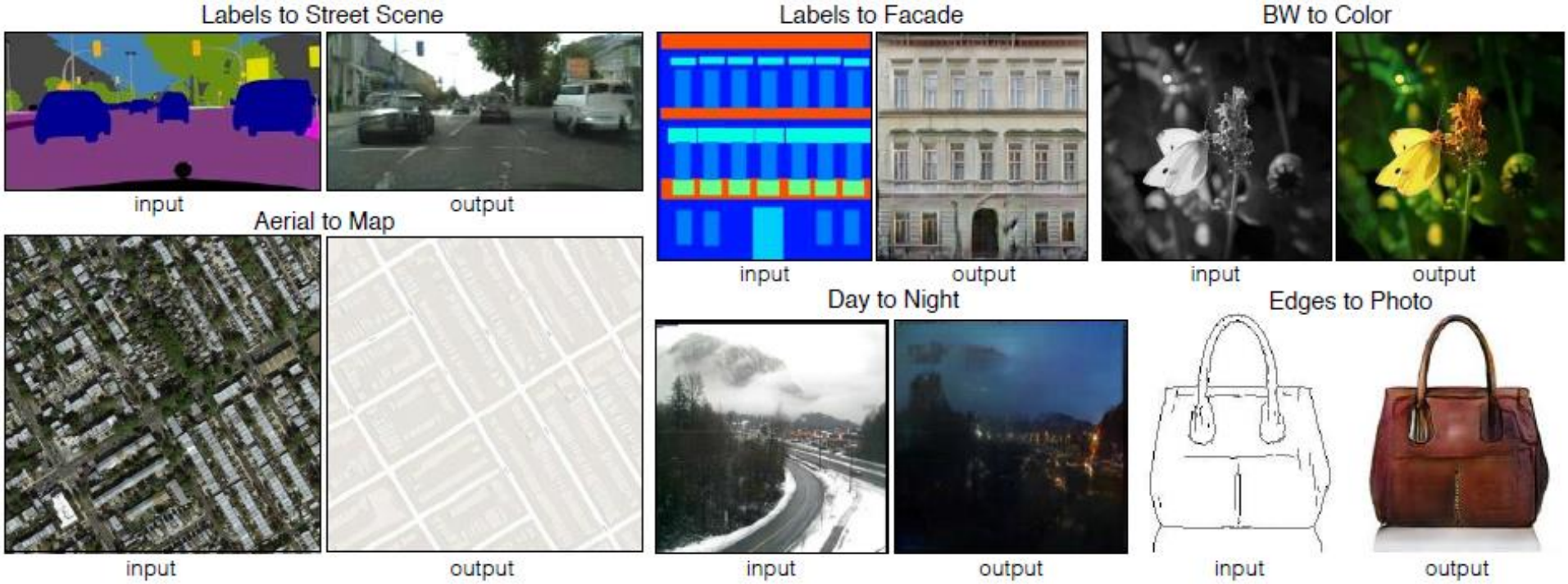


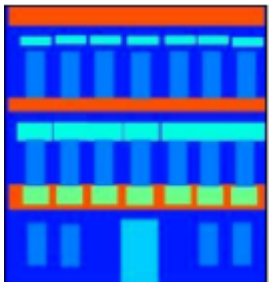
Image-to-image



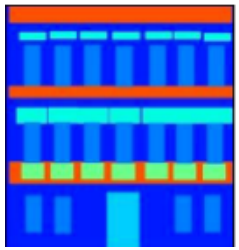


https://arxiv.org/pdf/1611.07004

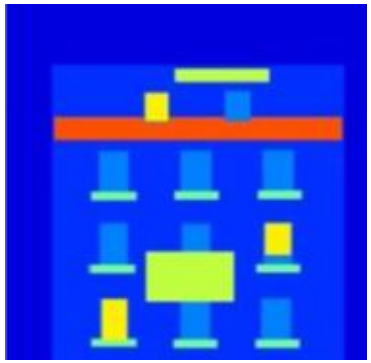
Image-to-image




- Traditional supervised approach



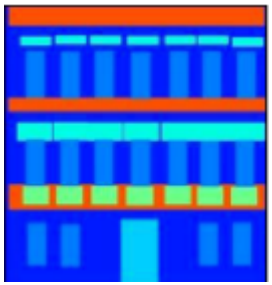
Testing:


input

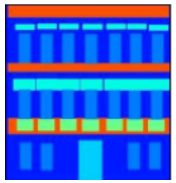

close

It is blurry because it is the average of several images.

Image-to-image



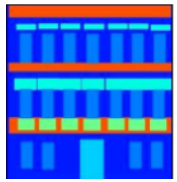
- Experimental results



z

G

Image

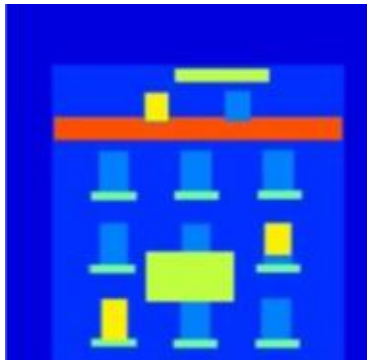


Image

D

scalar

Testing:



input

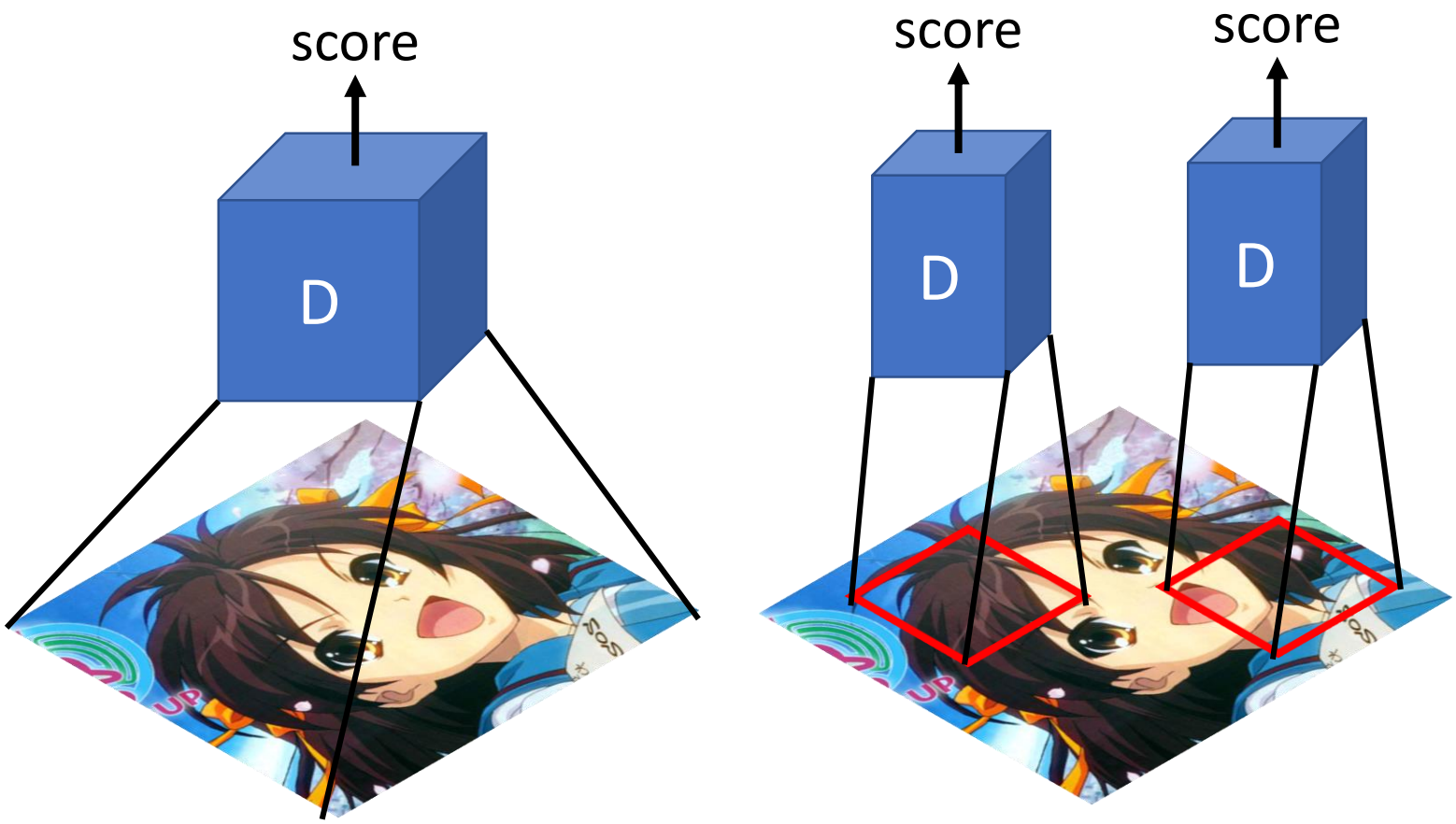
close

GAN

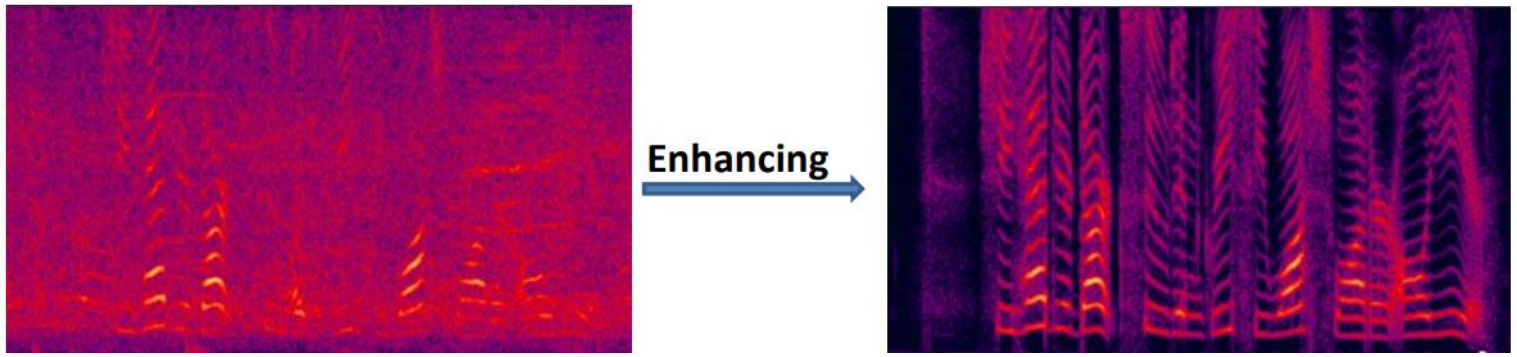
GAN + close

Patch GAN

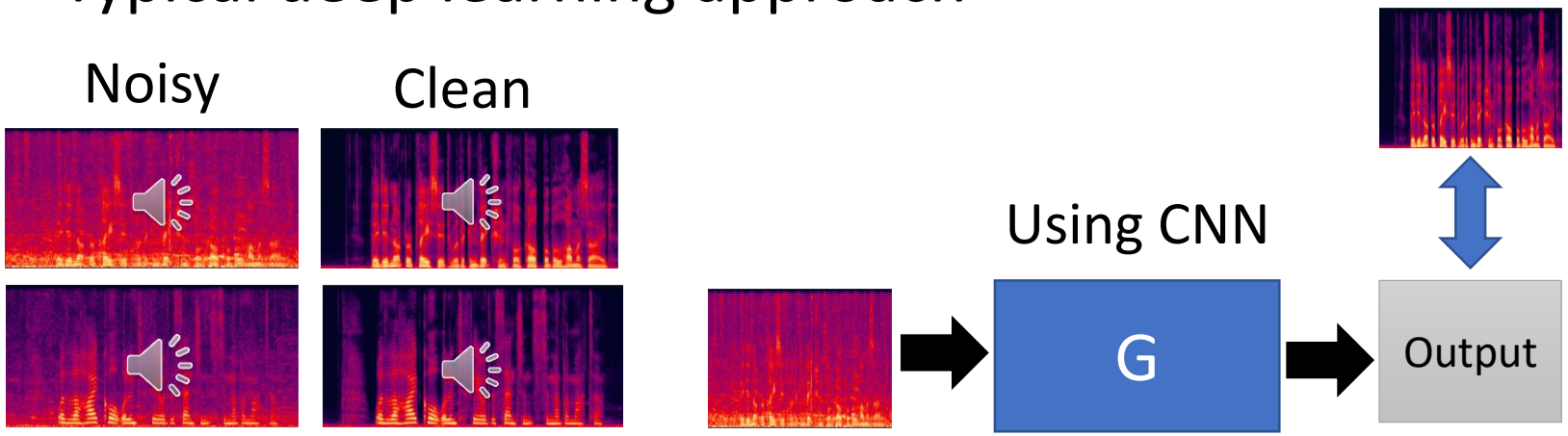
https://arxiv.org/pdf/1611.07004.pdf



Speech Enhancement

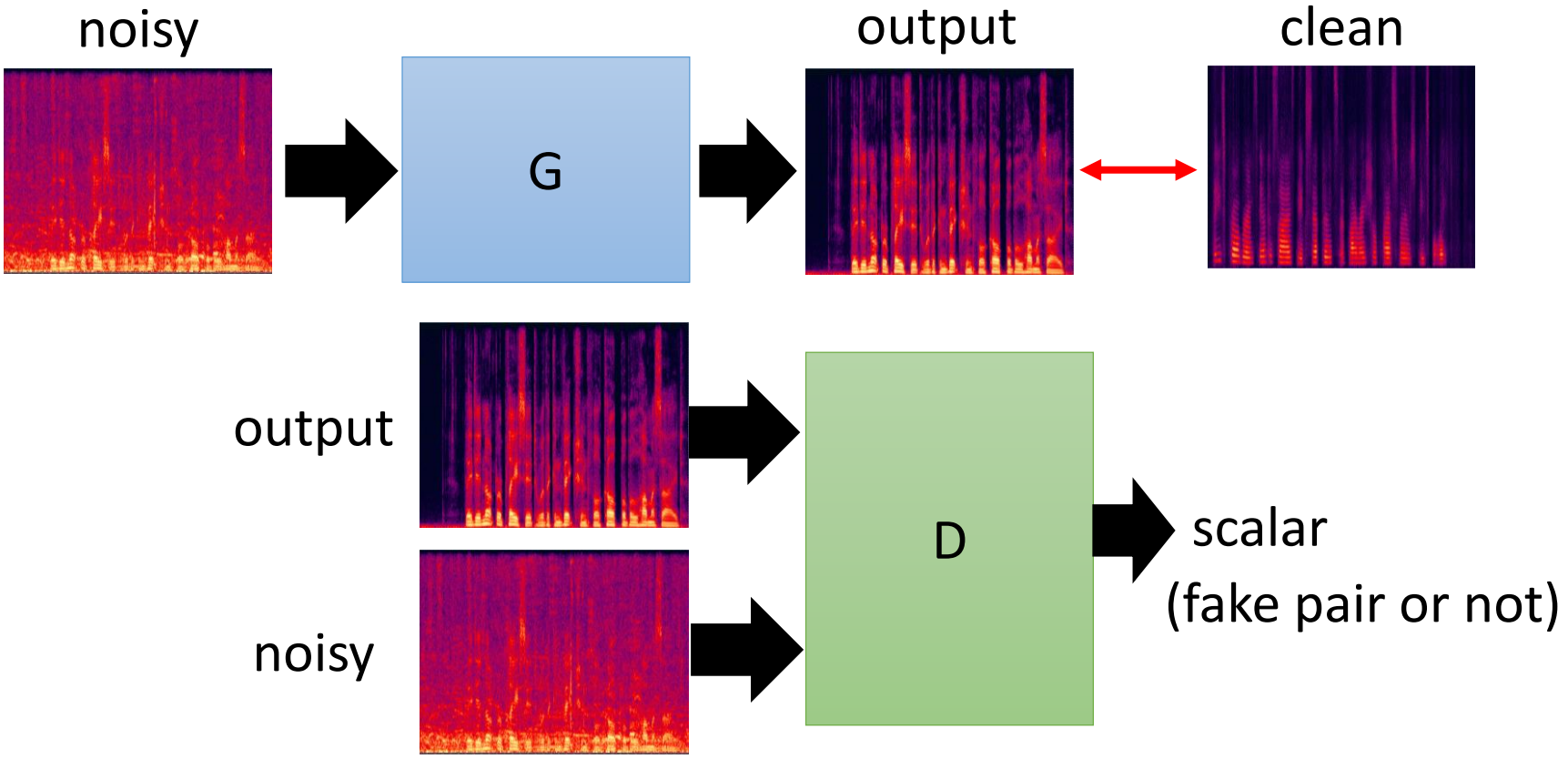


- Typical deep learning approach

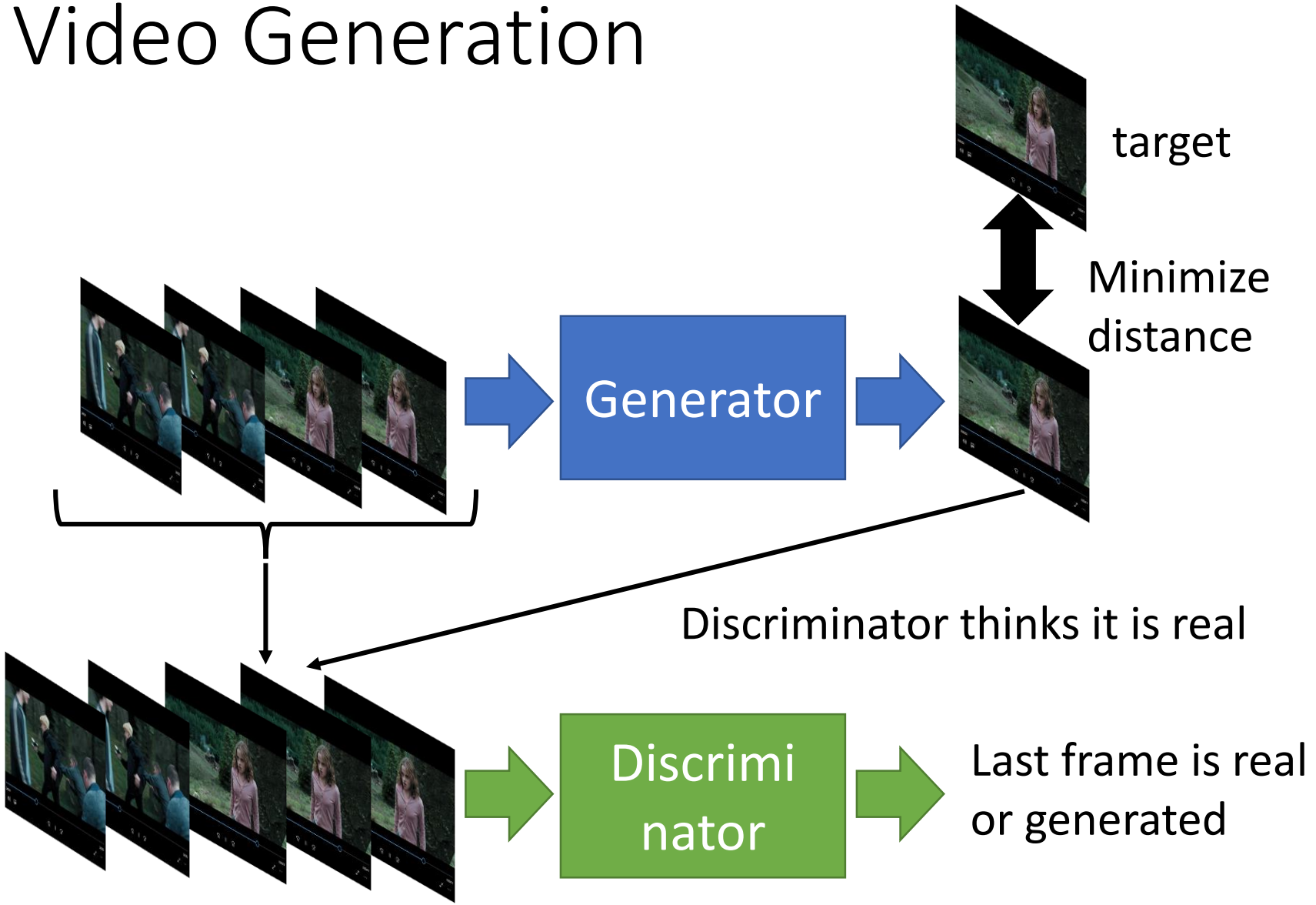


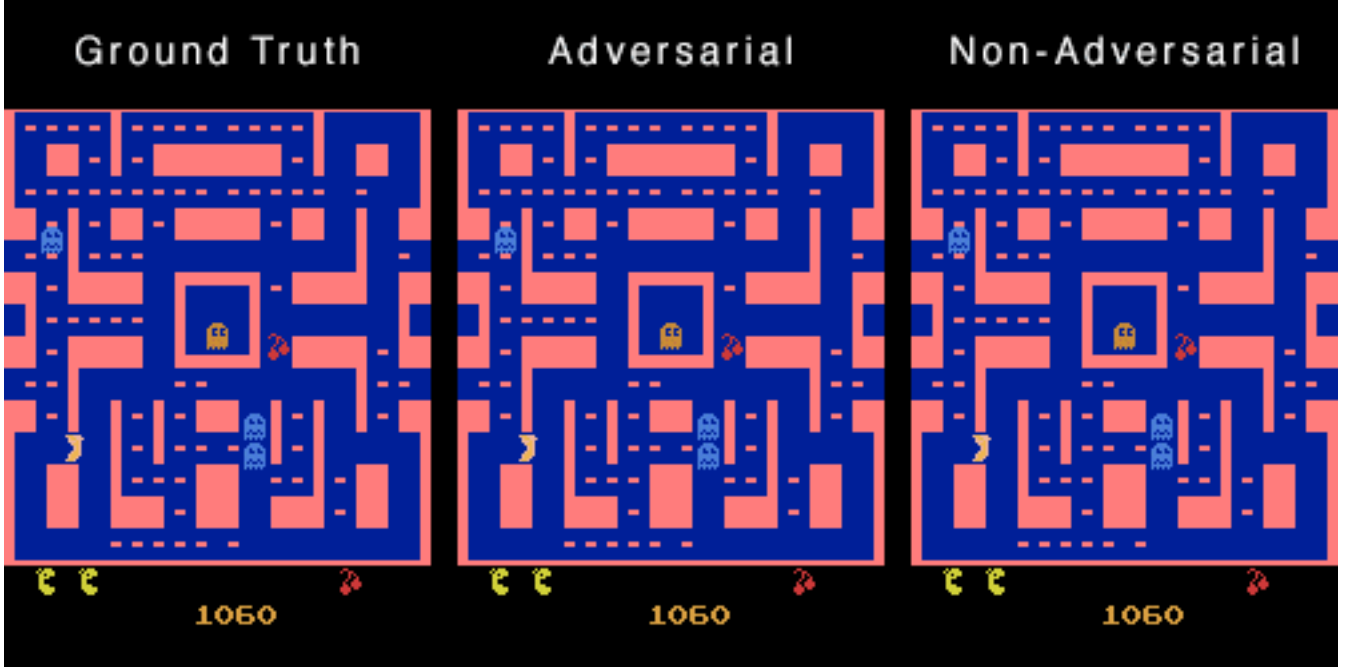
Speech Enhancement

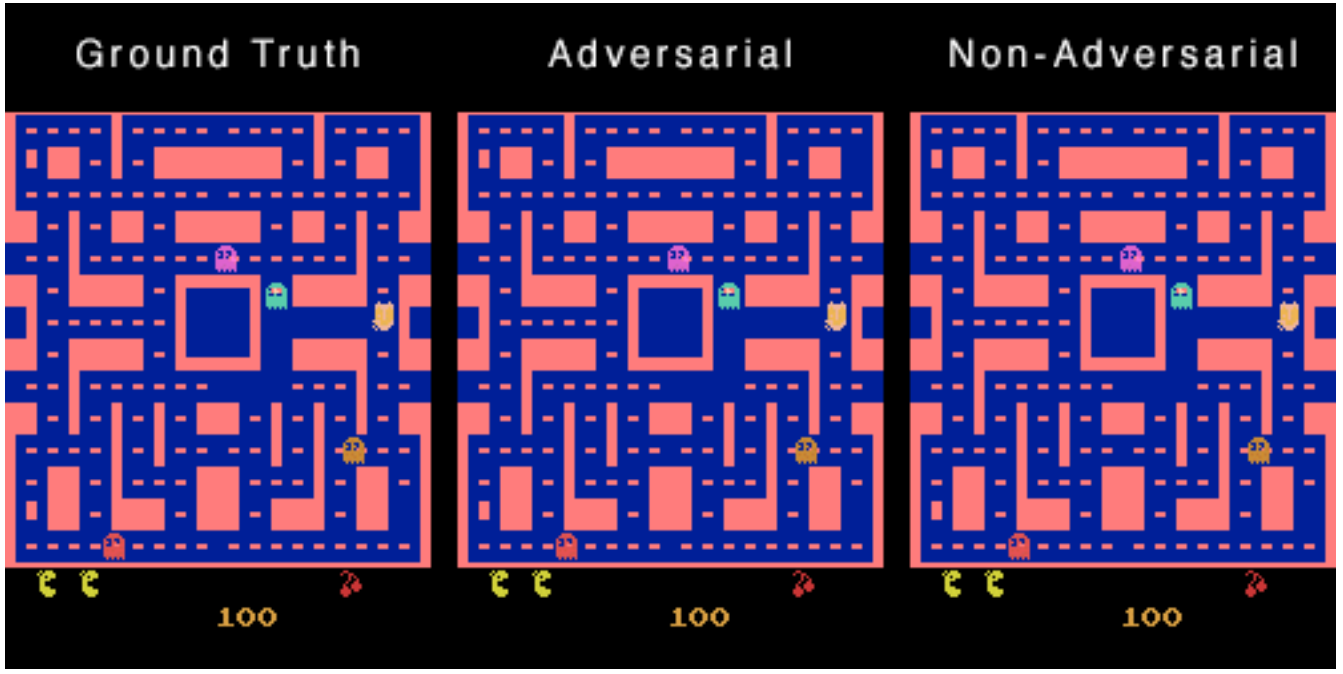
- Conditional GAN



Video Generation







https://github.com/dyelax/Adversarial_Video_Generation

Unsupervised Conditional Generation

Unsupervised Conditional Generation

Domain X  G  Domain Y

Transform an object from one domain to another ***without paired data*** (e.g. style transfer)

Domain X

male

Domain Y

female

It is good.

It's a good day.

I love you.

It is bad.

It's a bad day.

I don't love you.

Unsupervised Conditional Generation

- Approach 1: Direct Transformation

Domain X

$G_{X \to Y}$

Domain Y

For texture or color change

- Approach 2: Projection to Common Space

Domain X

EN_X

Encoder of domain X

Face Attribute

DE_Y

Decoder of domain Y

Domain Y

Larger change, only keep the semantics

Direct Transformation

Domain X

Domain Y

Domain X

$G_{X \to Y}$

Become similar to domain Y

D_Y

scalar

Input image belongs to domain Y or not

Domain Y

Direct Transformation

Domain X

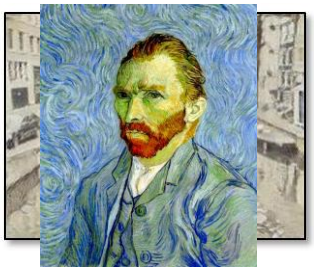
Domain Y

Domain X

ignore input

$G_{X \to Y}$

Become similar to domain Y



Not what we want!

D_Y

scalar

Input image belongs to domain Y or not

Domain Y

Direct Transformation

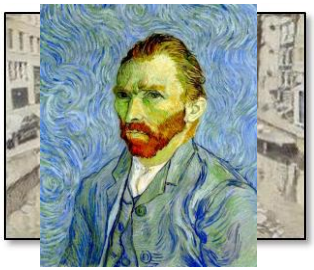
Domain X

Domain Y

Domain X

$G_{X \to Y}$

Become similar to domain Y



Not what we want!

ignore input

D_Y → scalar

Input image belongs to domain Y or not

The issue can be avoided by network design.

Simpler generator makes the input and output more closely related.

[Tomer Galanti, et al. ICLR, 2018]

Direct Transformation

Domain X

Domain Y

Domain X

$G_{X \to Y}$

Become similar to domain Y

D_Y → scalar

Input image belongs to domain Y or not

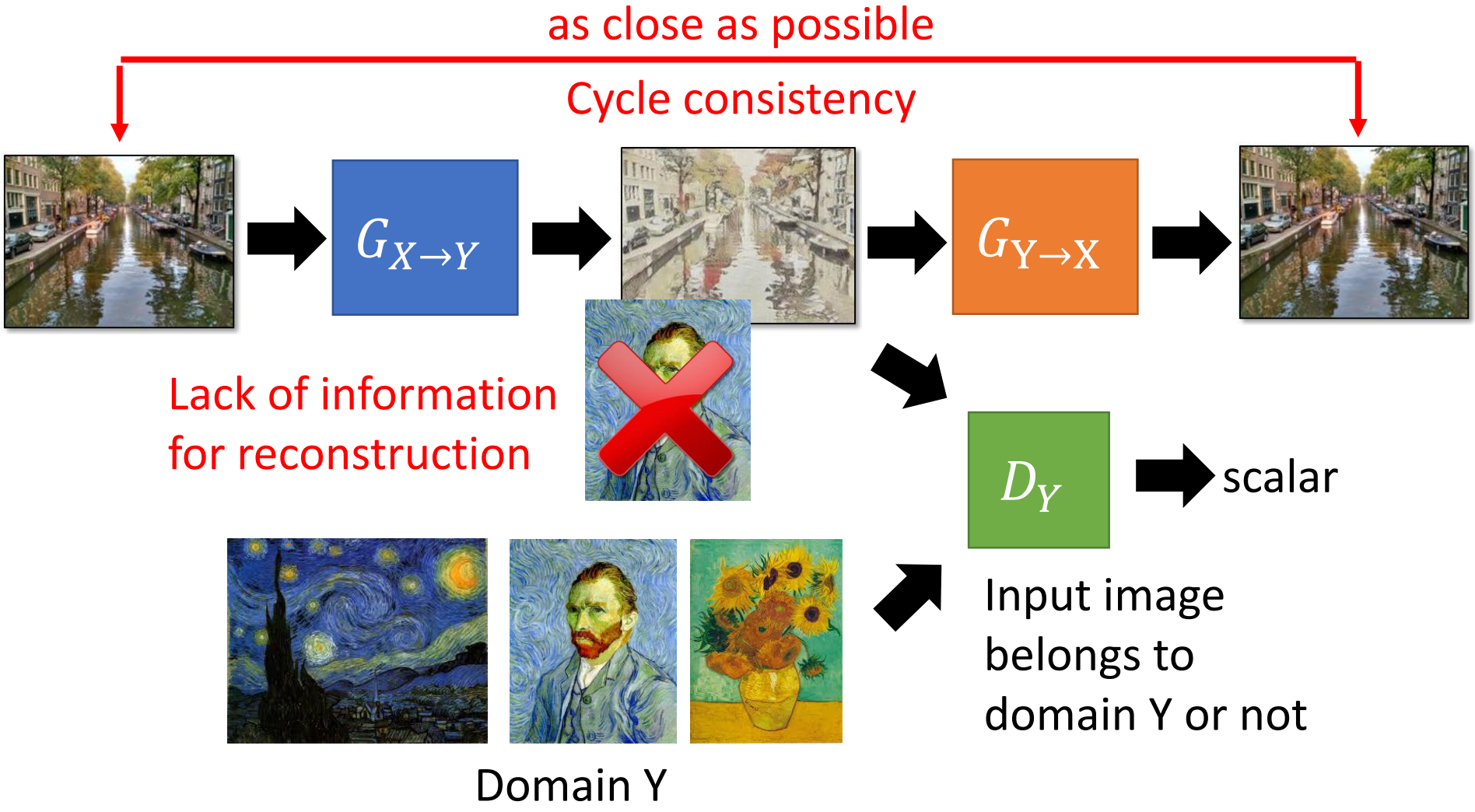
Encoder Network ← pre-trained → Encoder Network

as close as possible

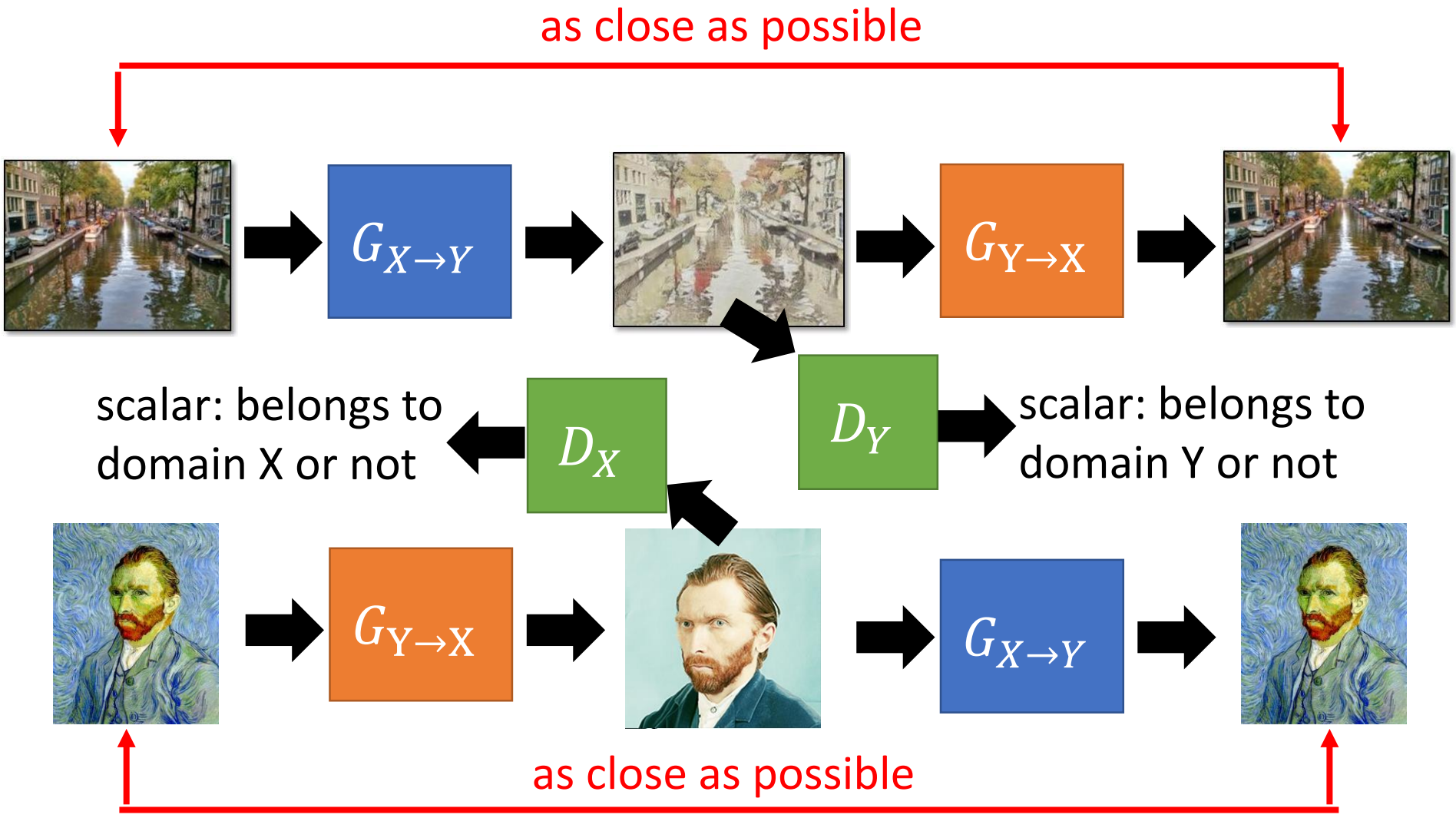
Baseline of DTN [Yaniv Taigman, et al., ICLR, 2017]

[Jun-Yan Zhu, et al., ICCV, 2017]

Direct Transformation

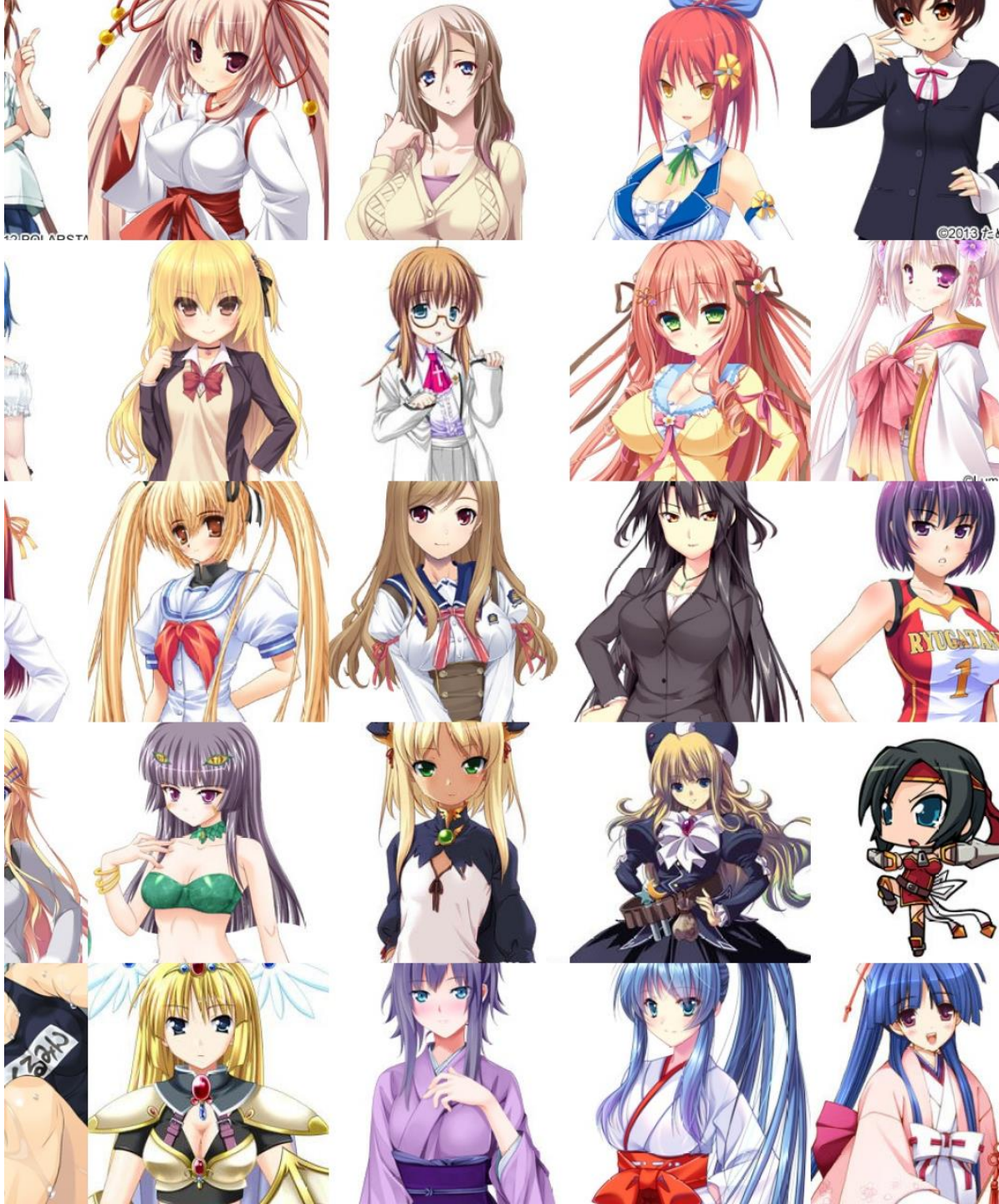


Direct Transformation



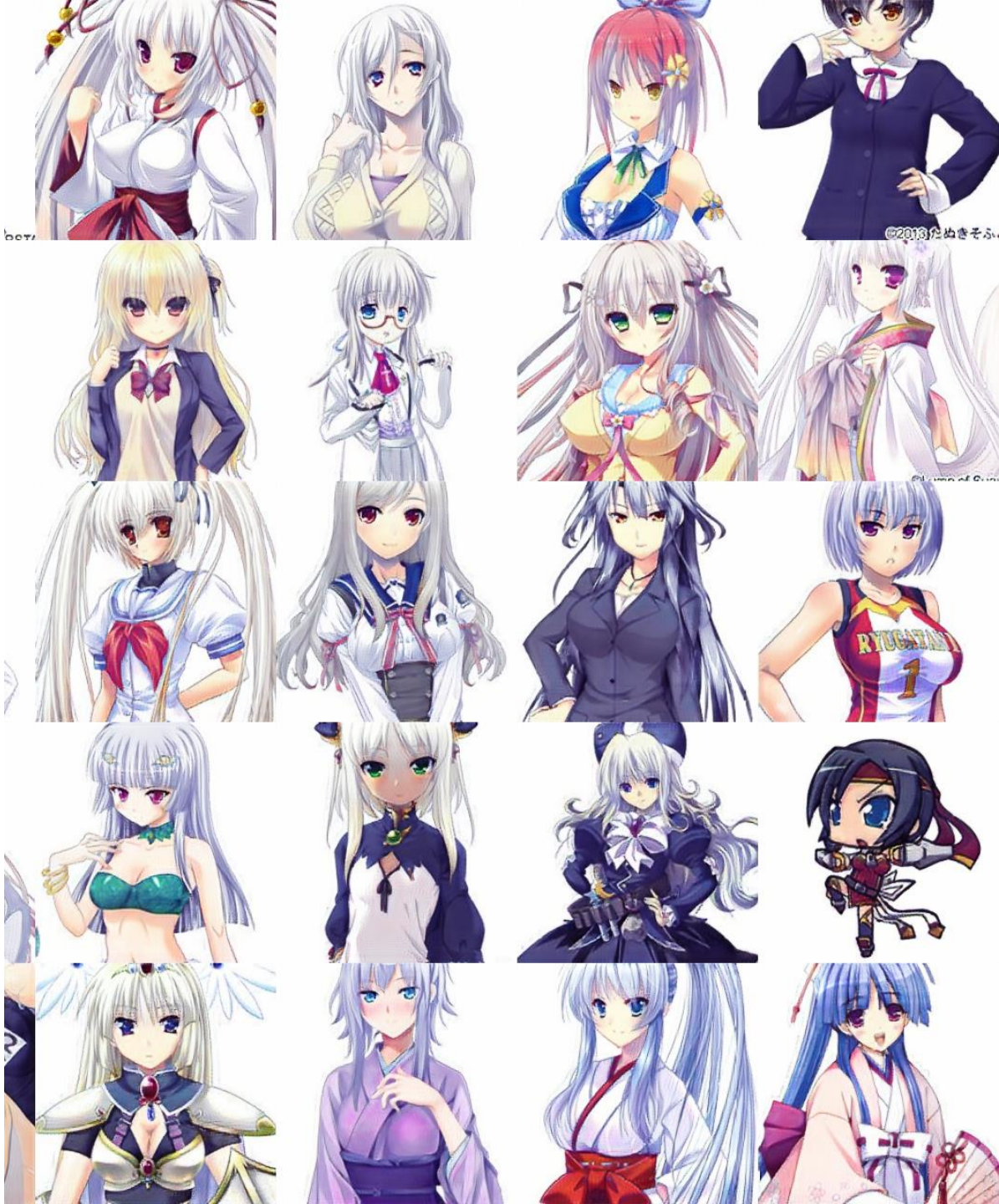
Cycle GAN – Silver Hair

- https://github.com/Aixile/chainer-cyclegan



Cycle GAN – Silver Hair

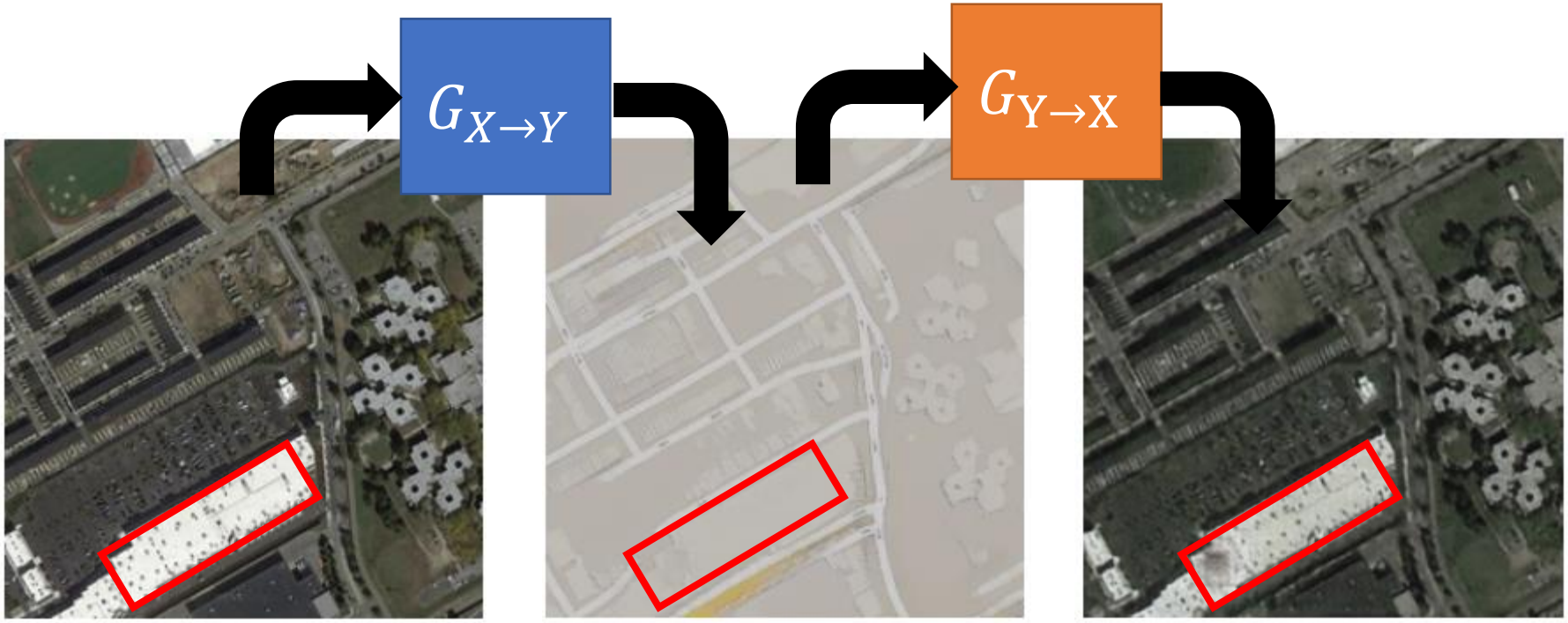
- https://github.com/Aixile/chainer-cyclegan



Issue of Cycle Consistency

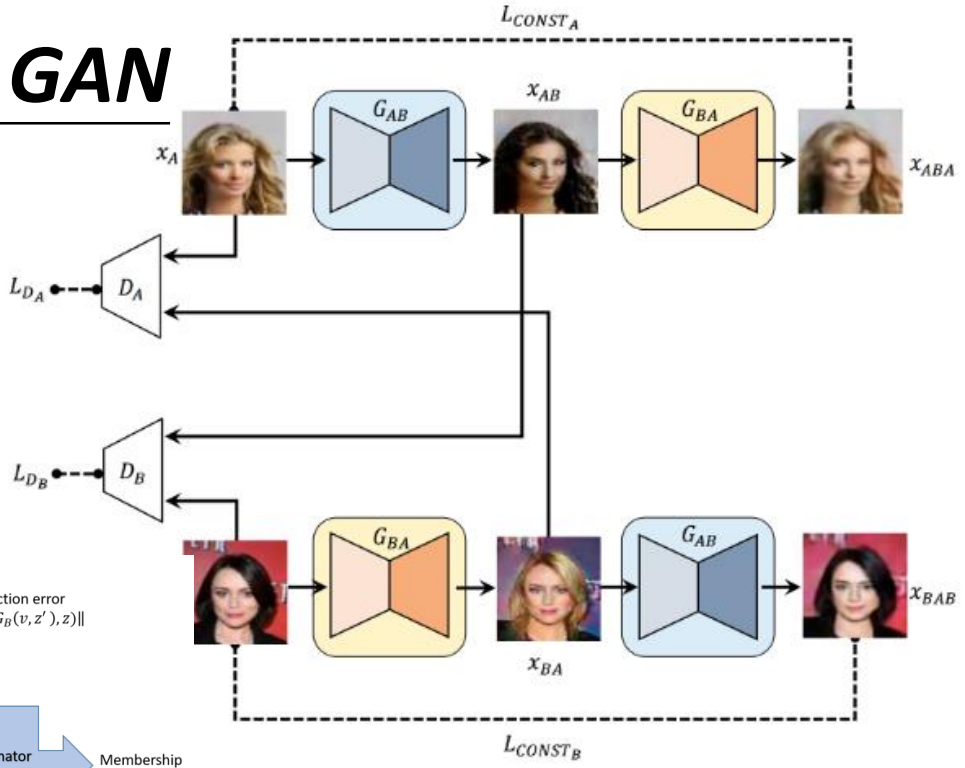
- **CycleGAN: a Master of Steganography (隱寫術)**

[Casey Chu, et al., NIPS workshop, 2017]



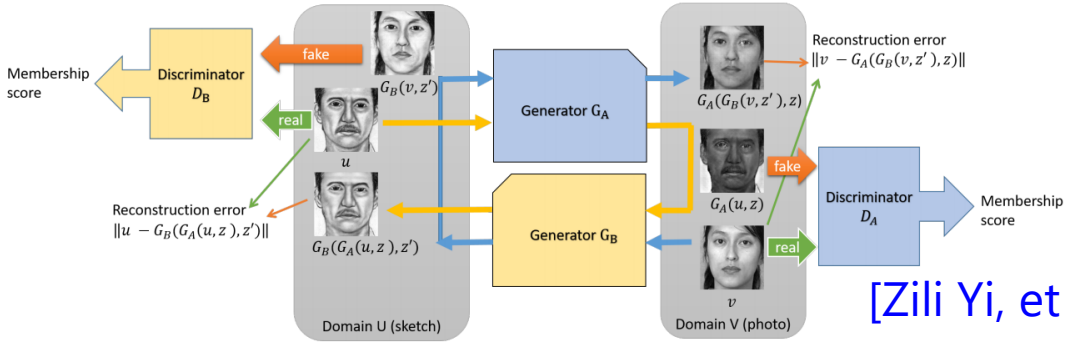
The information is hidden.

Disco GAN



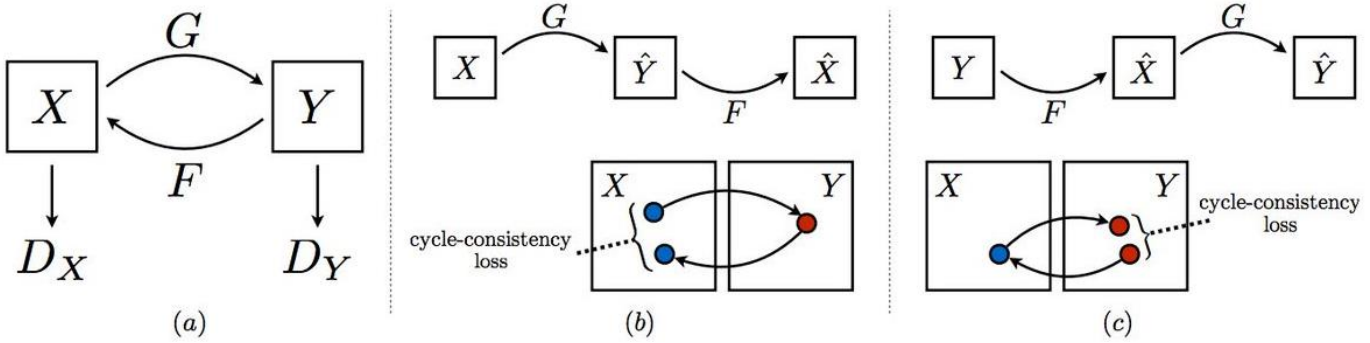
[Taeksoo Kim, et al., ICML, 2017]

Dual GAN



[Zili Yi, et al., ICCV, 2017]

Cycle GAN

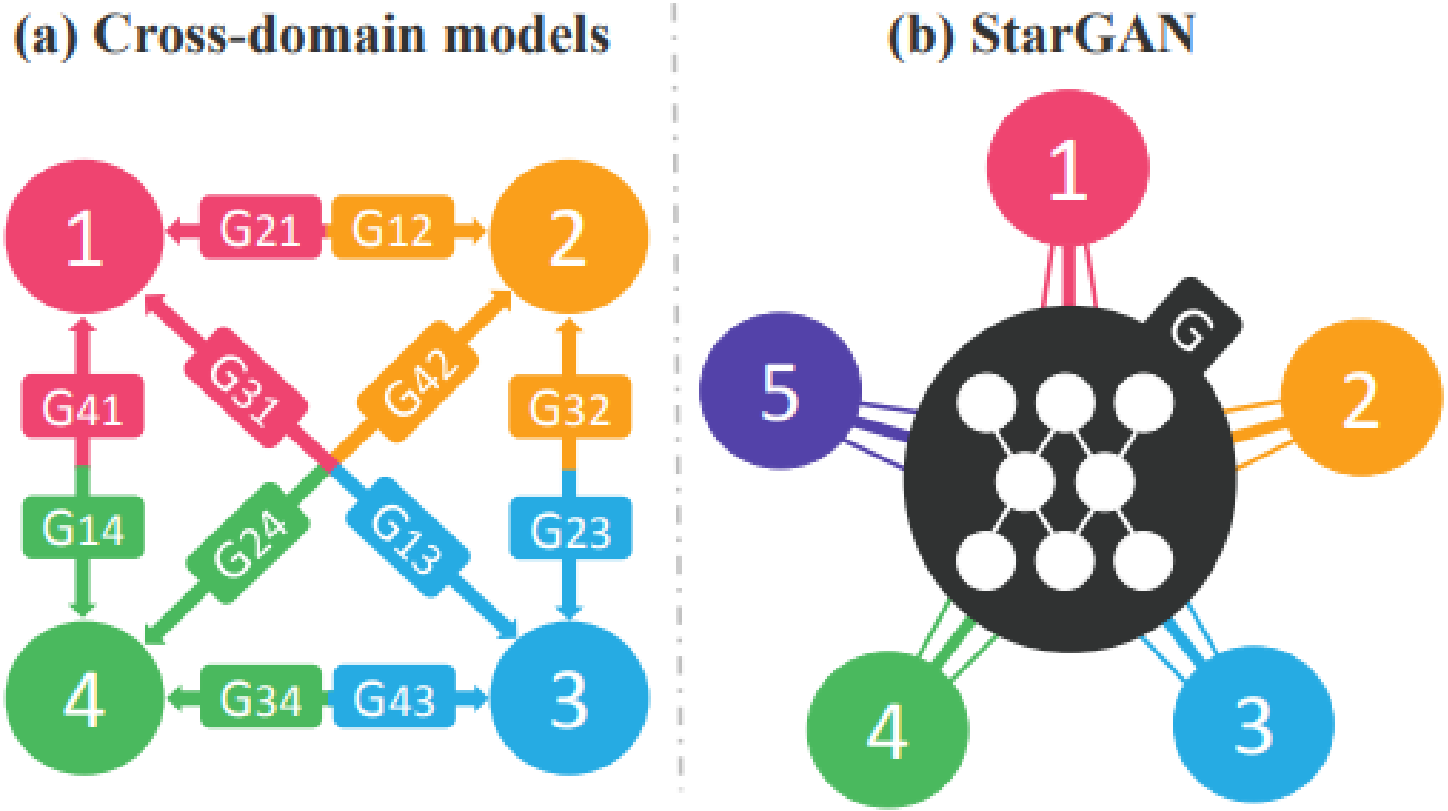


[Jun-Yan Zhu, et al., ICCV, 2017]

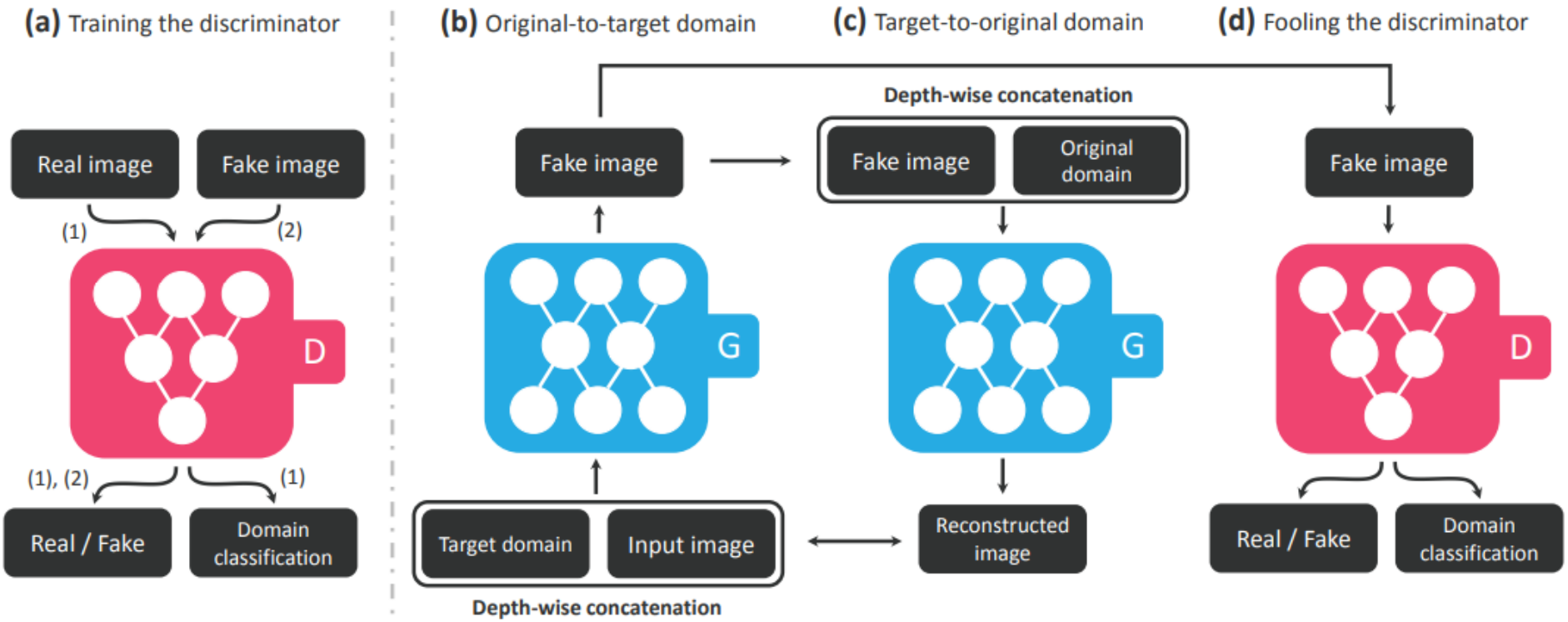
StarGAN

For multiple domains, considering starGAN

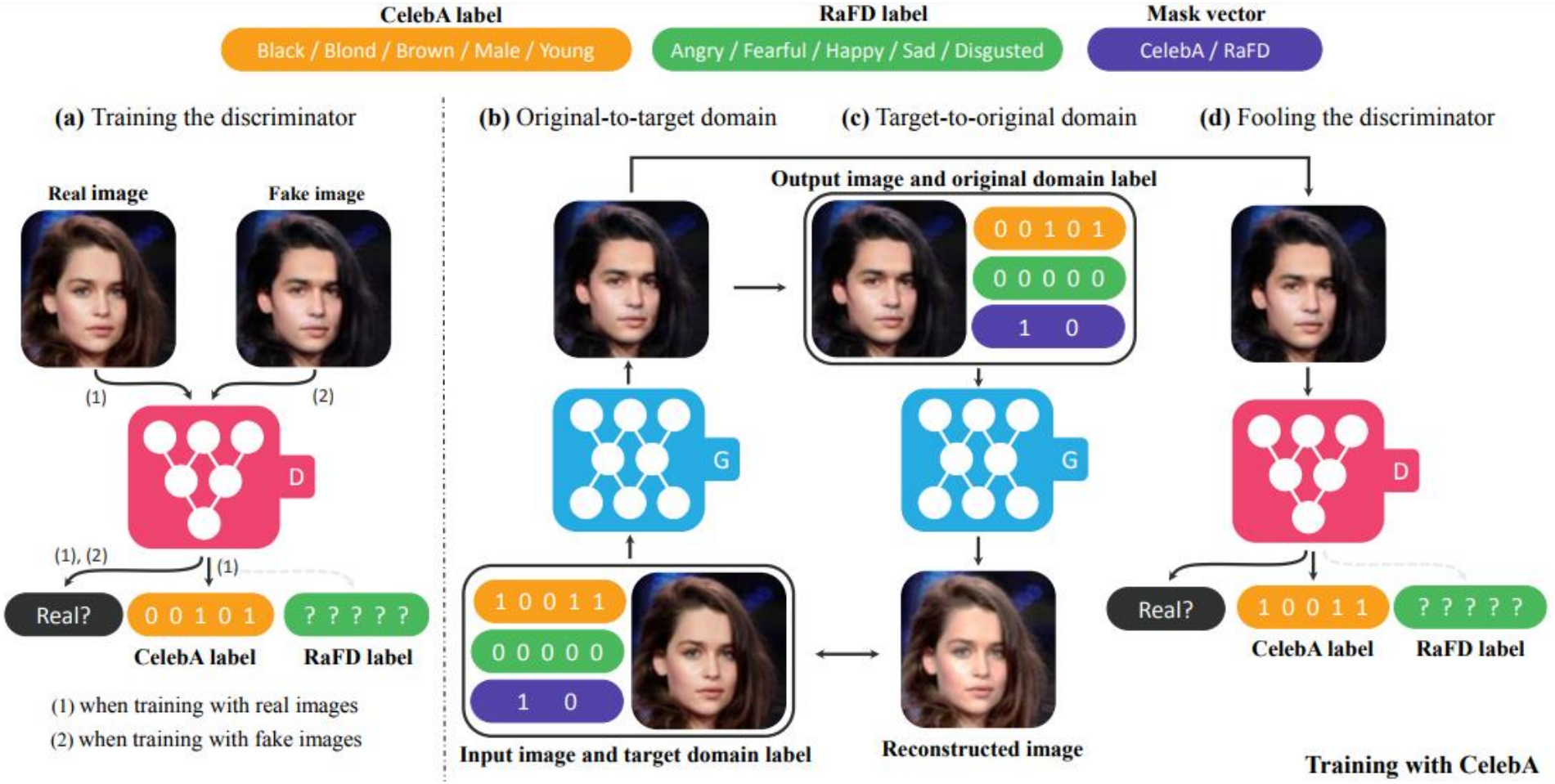
[Yunjey Choi, arXiv, 2017]



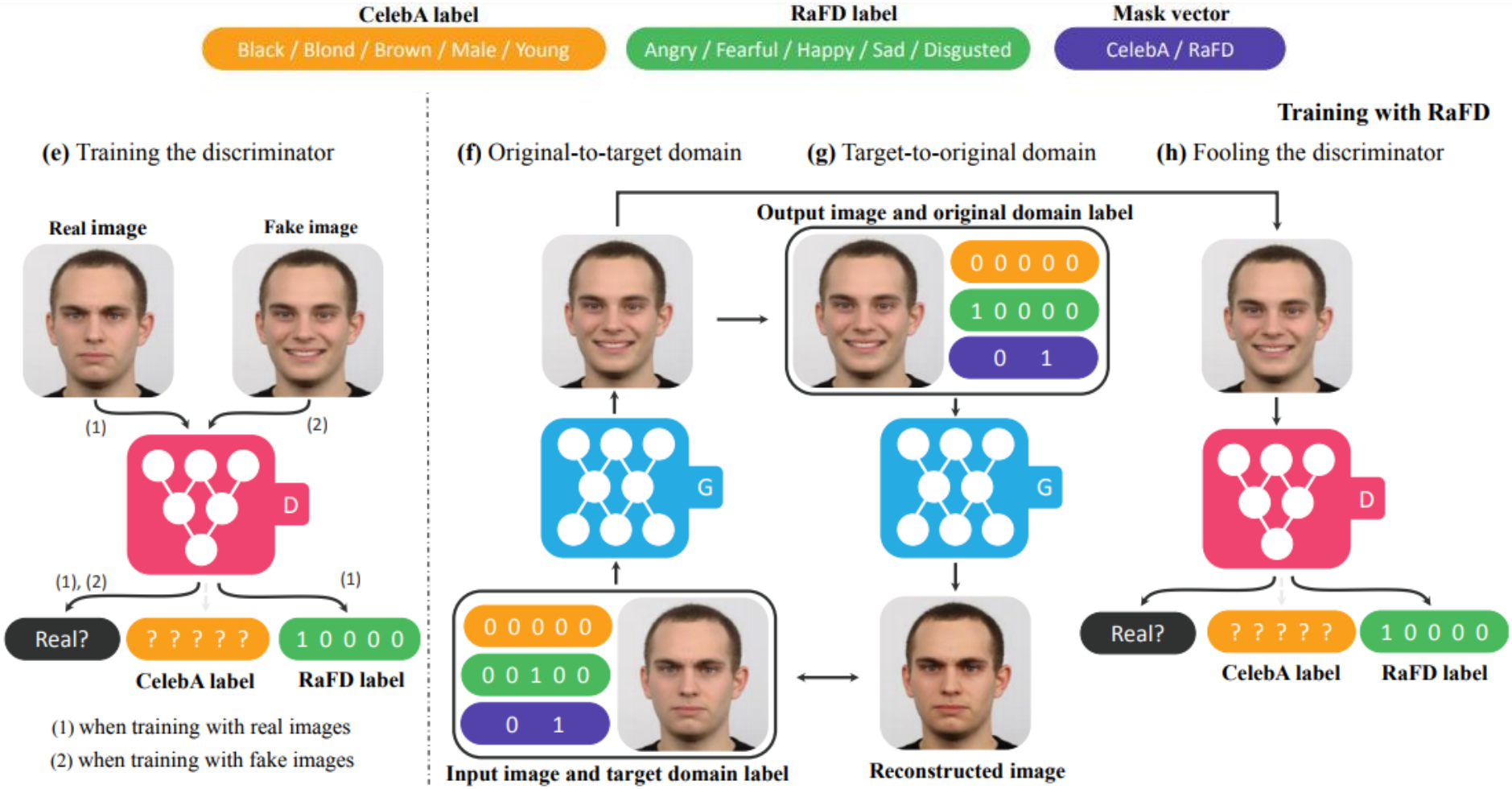
StarGAN


(a) Training the discriminator
(b) Original-to-target domain
(c) Target-to-original domain
(d) Fooling the discriminator
Real image
Fake image
(1)
(2)
D
(1), (2)
(1)
Real / Fake
Domain classification
Depth-wise concatenation
Fake image
G
Target domain
Input image
Depth-wise concatenation
Fake image
Original domain
G
Reconstructed image
Fake image
D
Real / Fake
Domain classification

StarGAN


CelebA label
Black / Blond / Brown / Male / Young
RaFD label
Angry / Fearful / Happy / Sad / Disgusted
Mask vector
CelebA / RaFD
(a) Training the discriminator
(b) Original-to-target domain
(c) Target-to-original domain
(d) Fooling the discriminator
Real image
Fake image
(1)
(2)
D
(1), (2)
(1)
Real?
0 0 1 0 1
? ? ? ? ?
CelebA label
RaFD label
(1) when training with real images
(2) when training with fake images
Output image and original domain label
0 0 1 0 1
0 0 0 0 0
1 0
G
G
1 0 0 1 1
0 0 0 0 0
1 0
Input image and target domain label
Reconstructed image
D
Real?
1 0 0 1 1
? ? ? ? ?
CelebA label
RaFD label
Training with CelebA

StarGAN


CelebA label
Black / Blond / Brown / Male / Young
RaFD label
Angry / Fearful / Happy / Sad / Disgusted
Mask vector
CelebA / RaFD
Training with RaFD
(e) Training the discriminator
(f) Original-to-target domain
(g) Target-to-original domain
(h) Fooling the discriminator
Real image
Fake image
(1)
(2)
D
(1), (2)
(1)
Real?
? ? ? ? ?
1 0 0 0 0
CelebA label
RaFD label
(1) when training with real images
(2) when training with fake images
Output image and original domain label
0 0 0 0 0
1 0 0 0 0
0 1
G
G
0 0 0 0 0
0 0 1 0 0
0 1
Input image and target domain label
Reconstructed image
D
Real?
? ? ? ? ?
1 0 0 0 0
CelebA label
RaFD label

Unsupervised Conditional Generation

- Approach 1: Direct Transformation

Domain X

$G_{X \to Y}$

Domain Y

For texture or color change

- Approach 2: Projection to Common Space

Domain X

EN_X

Encoder of domain X

Face Attribute

DE_Y

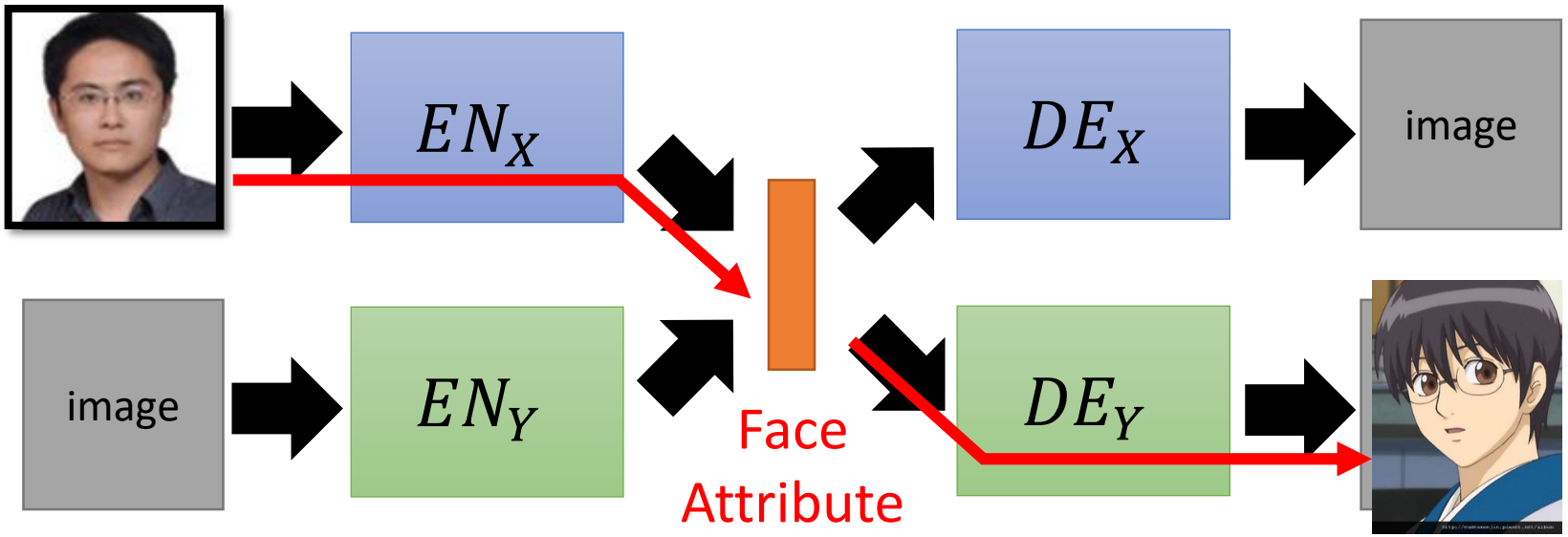
Decoder of domain Y

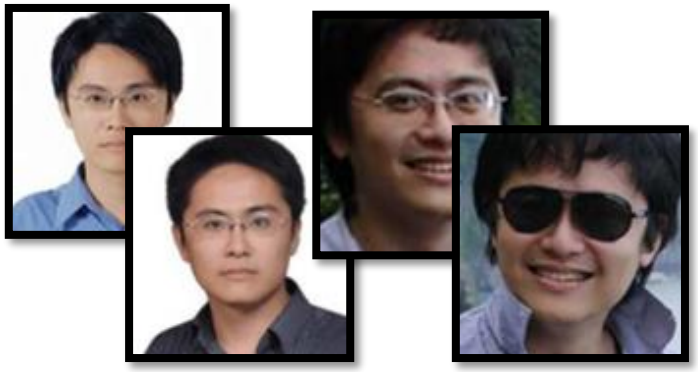
Domain Y

Larger change, only keep the semantics

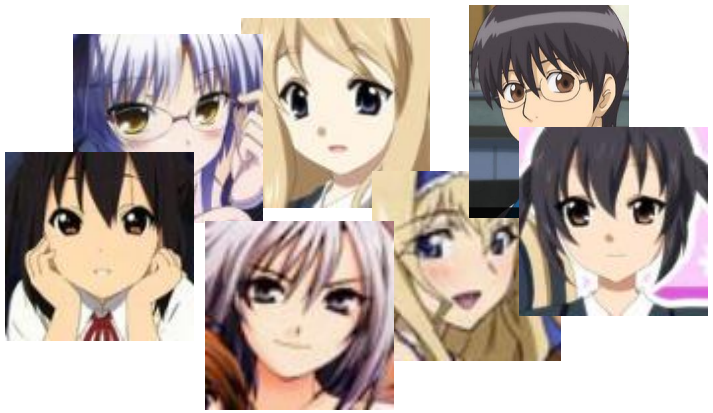
Projection to Common Space

Target





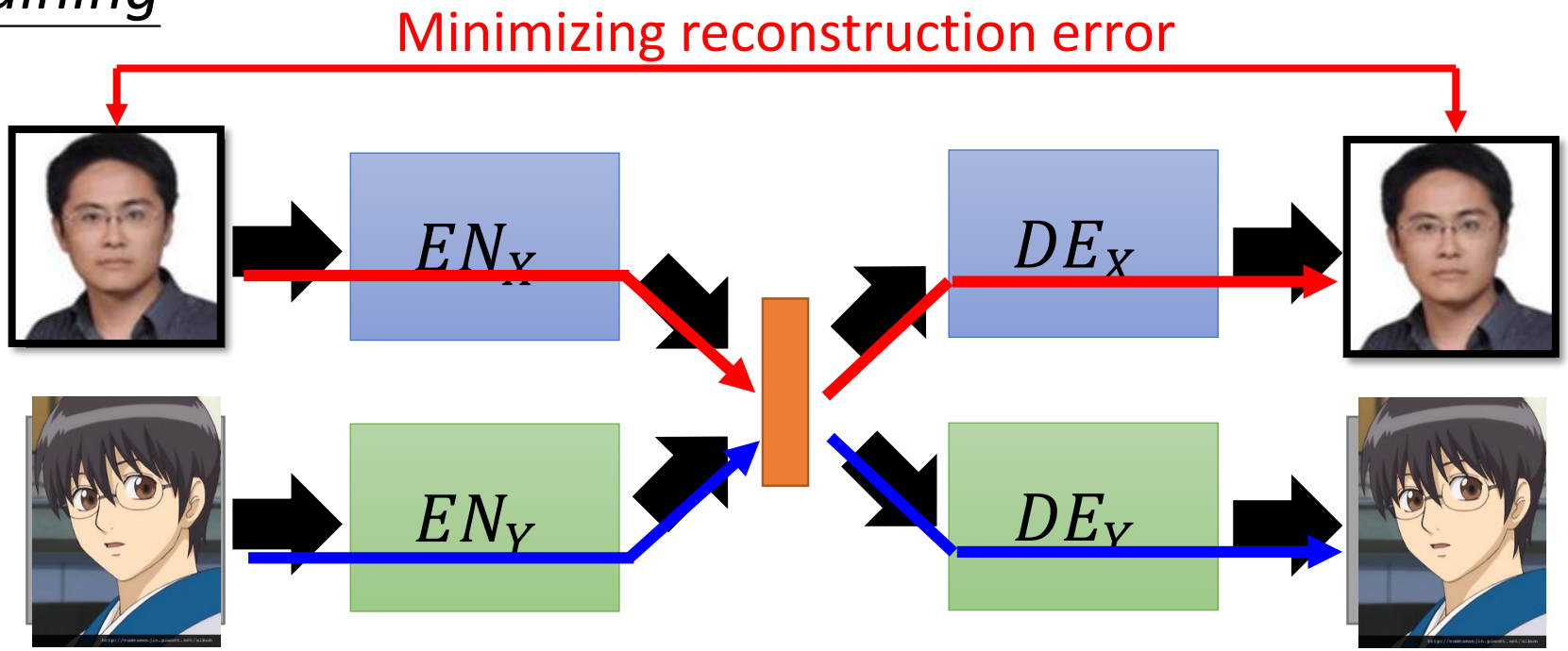
Domain X

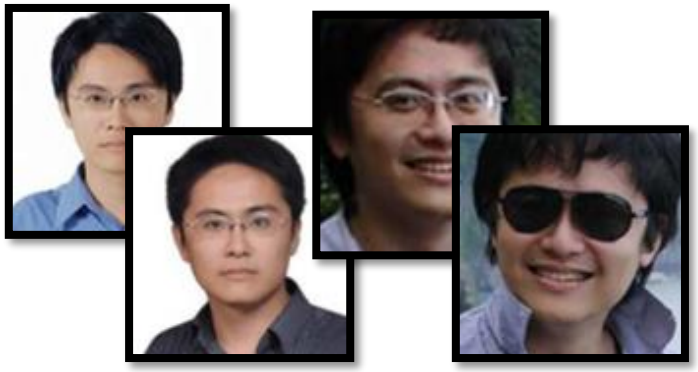


Domain Y

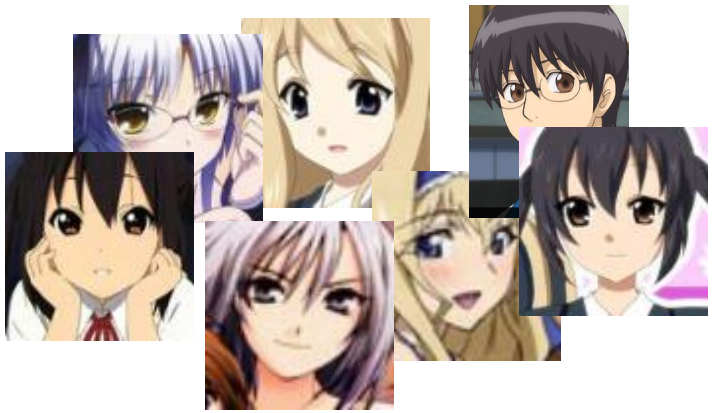
Projection to Common Space

Training





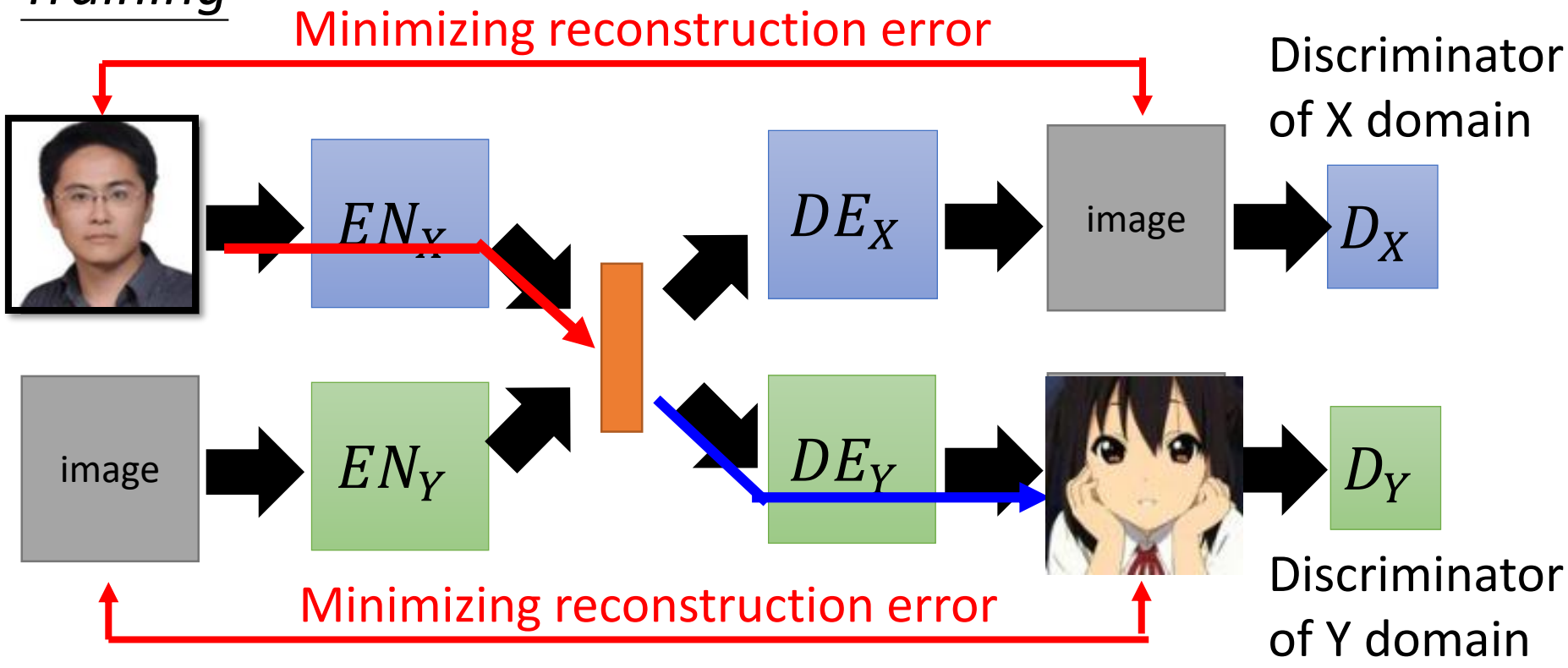
Domain X



Domain Y

Projection to Common Space

Training

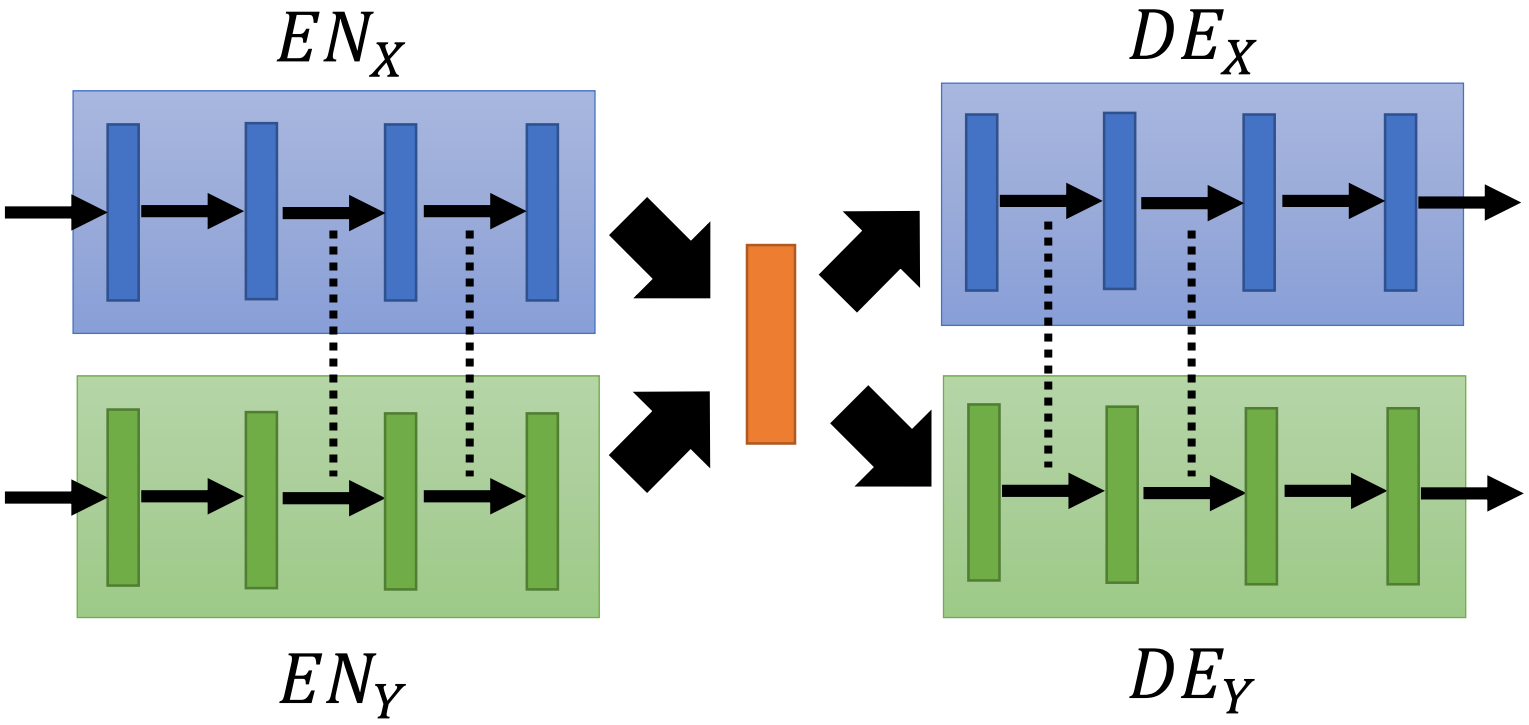


Because we train two auto-encoders separately …

The images with the same attribute may not project to the same position in the latent space.

Projection to Common Space

Training



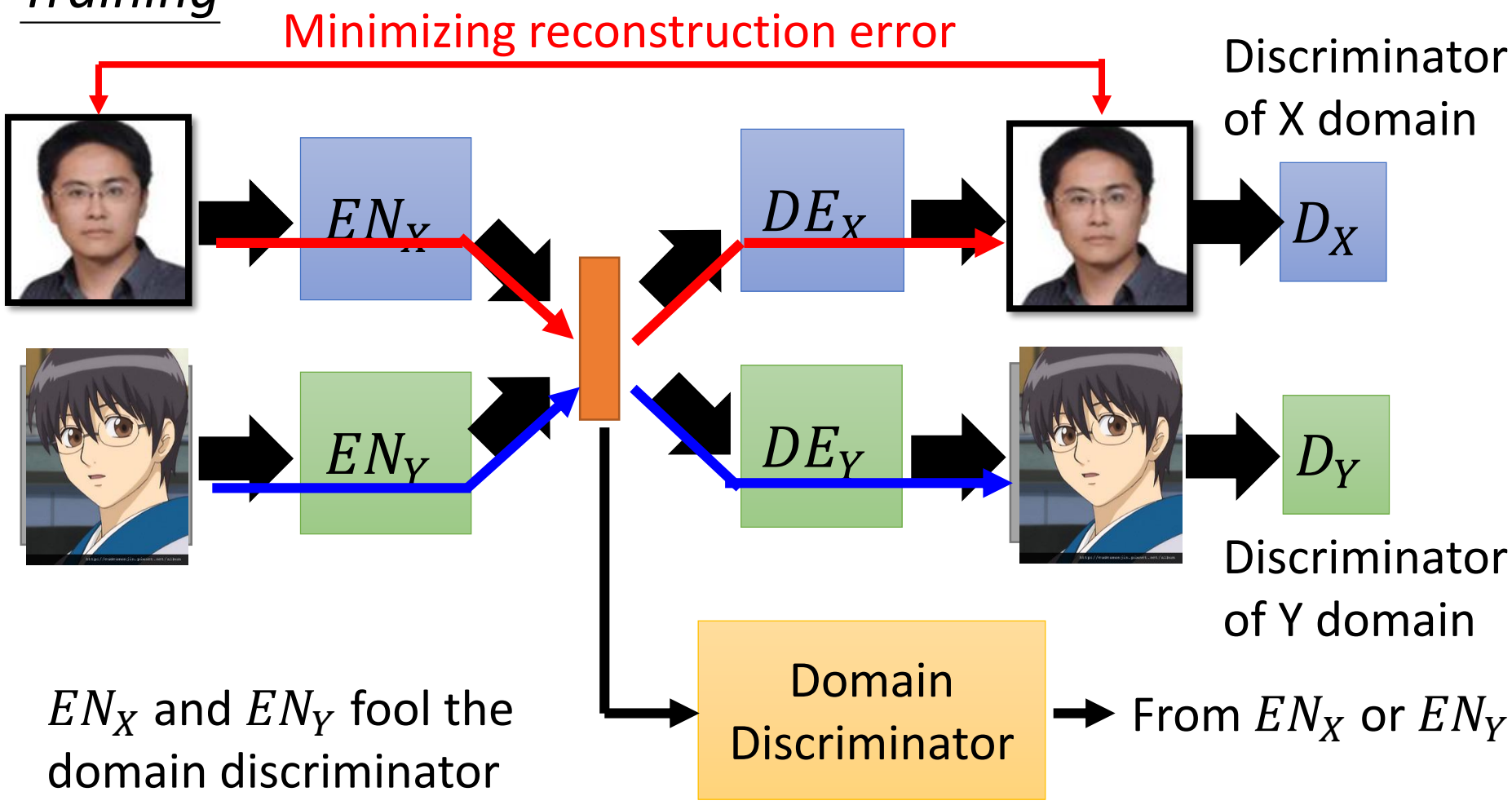
Sharing the parameters of encoders and decoders

Couple GAN[Ming-Yu Liu, et al., NIPS, 2016]

UNIT[Ming-Yu Liu, et al., NIPS, 2017]

Projection to Common Space

Training

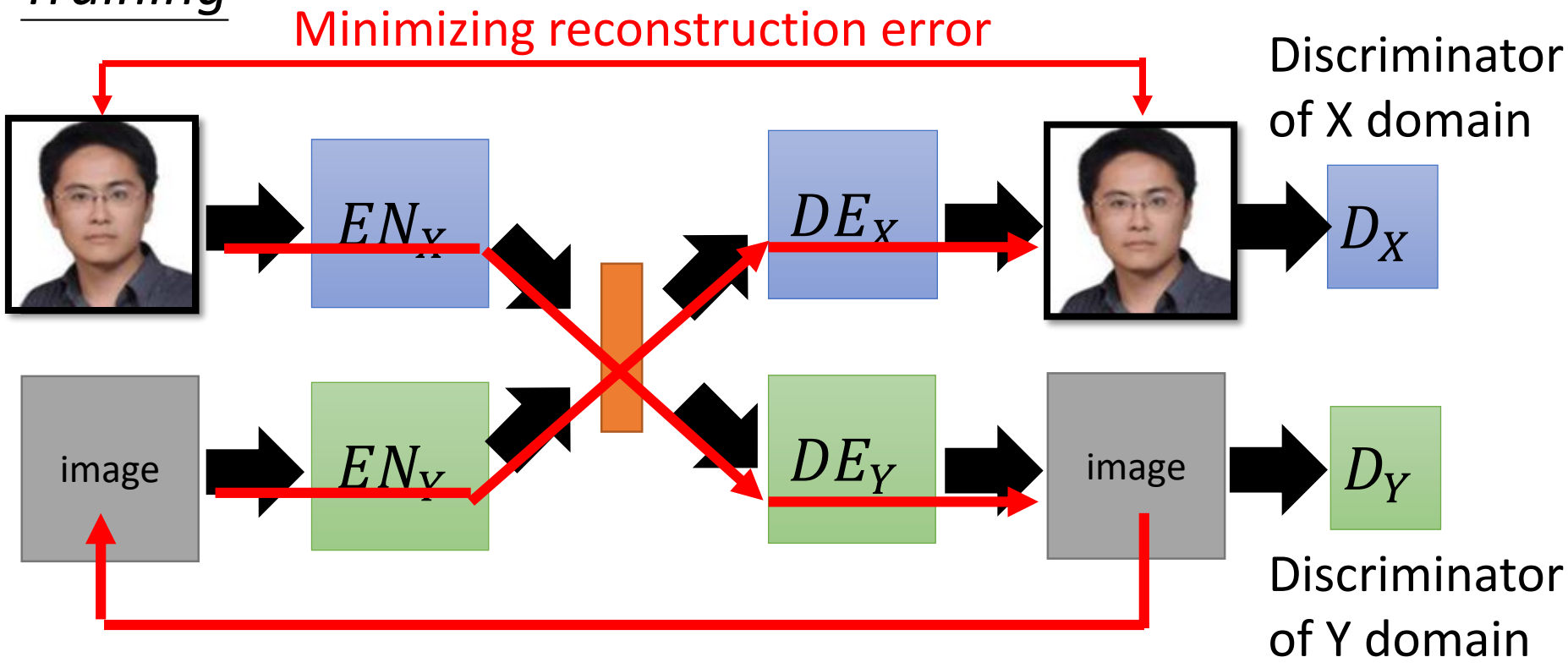


The domain discriminator forces the output of EN_X and EN_Y have the same distribution.

[Guillaume Lample, et al., NIPS, 2017]

Projection to Common Space

Training

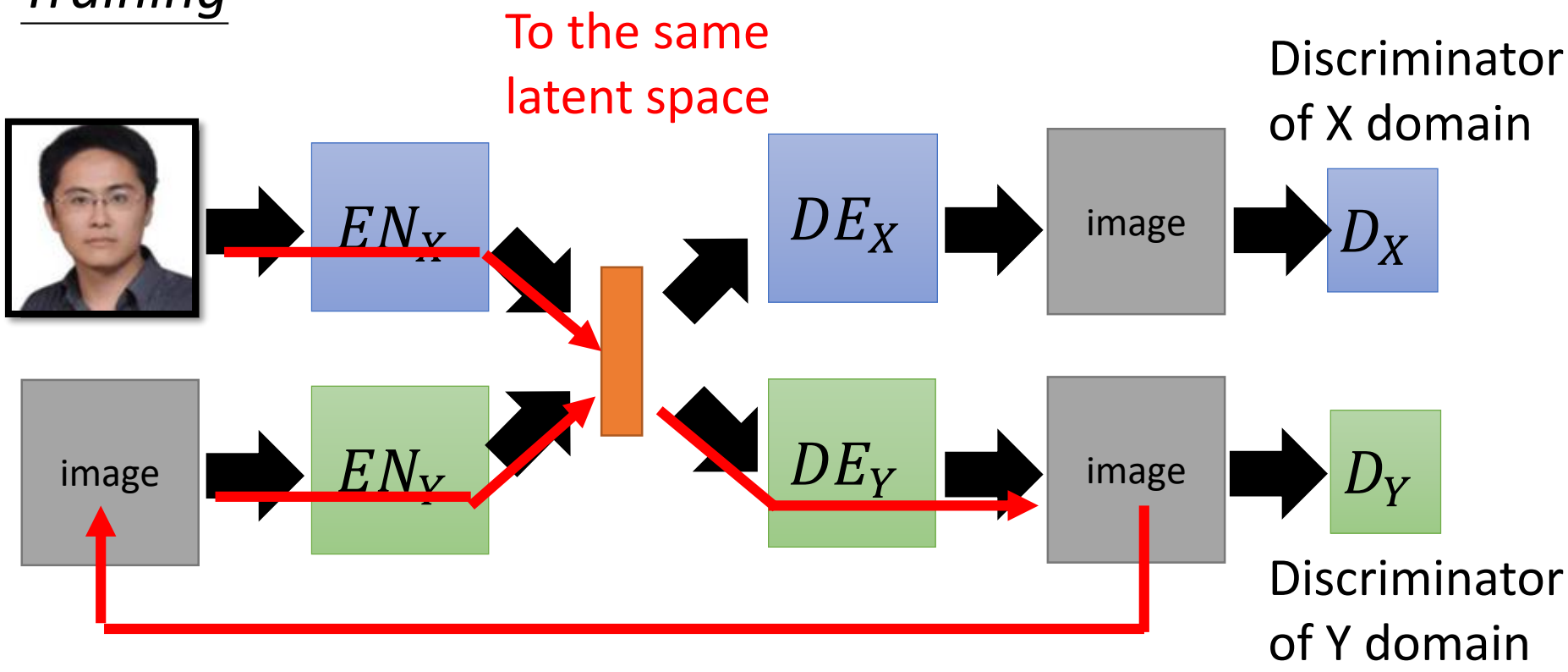


Cycle Consistency:

Used in ComboGAN [Asha Anoosheh, et al., arXiv, 017]

Projection to Common Space

Training



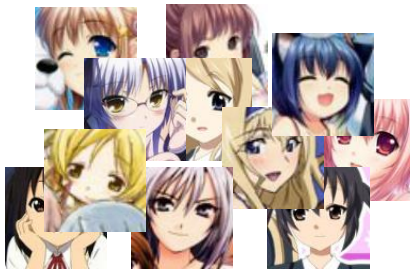
Semantic Consistency:

Used in DTN [Yaniv Taigman, et al., ICLR, 2017] and XGAN [Amélie Royer, et al., arXiv, 2017]

世界二次元化

- Using the code: https://github.com/Hi-king/kawaii_creator
- It is not cycle GAN, Disco GAN

input

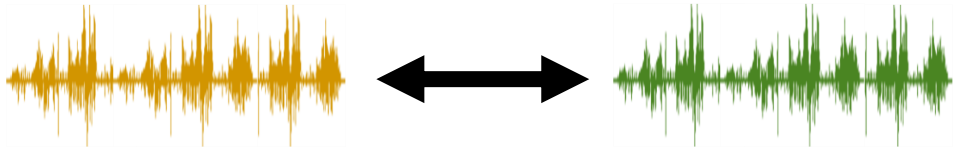


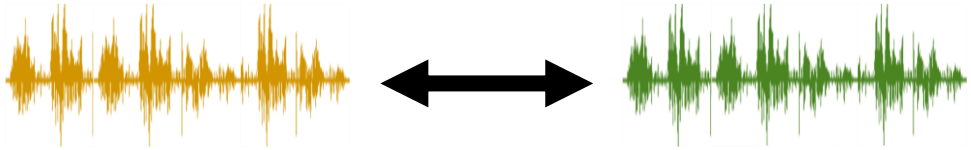
output ***domain***

Voice Conversion

In the past

Speaker A Speaker B

How are you?  How are you?

Good morning  Good morning

Today

Speaker A Speaker B

天氣真好  How are you?

再見囉  Good morning

Speakers A and B are talking about completely different things.

Speaker A	Speaker B
我	

感謝周儒杰同學提供實驗結果

Reference

- Jun-Yan Zhu, Taesung Park, Phillip Isola, Alexei A. Efros, Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks, ICCV, 2017
- Zili Yi, Hao Zhang, Ping Tan, Minglun Gong, DualGAN: Unsupervised Dual Learning for Image-to-Image Translation, ICCV, 2017
- Tomer Galanti, Lior Wolf, Sagie Benaim, The Role of Minimal Complexity Functions in Unsupervised Learning of Semantic Mappings, ICLR, 2018
- Yaniv Taigman, Adam Polyak, Lior Wolf, Unsupervised Cross-Domain Image Generation, ICLR, 2017
- Asha Anoosheh, Eirikur Agustsson, Radu Timofte, Luc Van Gool, ComboGAN: Unrestrained Scalability for Image Domain Translation, arXiv, 2017
- Amélie Royer, Konstantinos Bousmalis, Stephan Gouws, Fred Bertsch, Inbar Mosseri, Forrester Cole, Kevin Murphy, XGAN: Unsupervised Image-to-Image Translation for Many-to-Many Mappings, arXiv, 2017

Reference

- Guillaume Lample, Neil Zeghidour, Nicolas Usunier, Antoine Bordes, Ludovic Denoyer, Marc'Aurelio Ranzato, Fader Networks: Manipulating Images by Sliding Attributes, NIPS, 2017
- Taeksoo Kim, Moonsu Cha, Hyunsoo Kim, Jung Kwon Lee, Jiwon Kim, Learning to Discover Cross-Domain Relations with Generative Adversarial Networks, ICML, 2017
- Ming-Yu Liu, Oncel Tuzel, “Coupled Generative Adversarial Networks”, NIPS, 2016
- Ming-Yu Liu, Thomas Breuel, Jan Kautz, Unsupervised Image-to-Image Translation Networks, NIPS, 2017
- Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, Jaegul Choo, StarGAN: Unified Generative Adversarial Networks for Multi-Domain Image-to-Image Translation, arXiv, 2017

Theory behind GAN

Generation

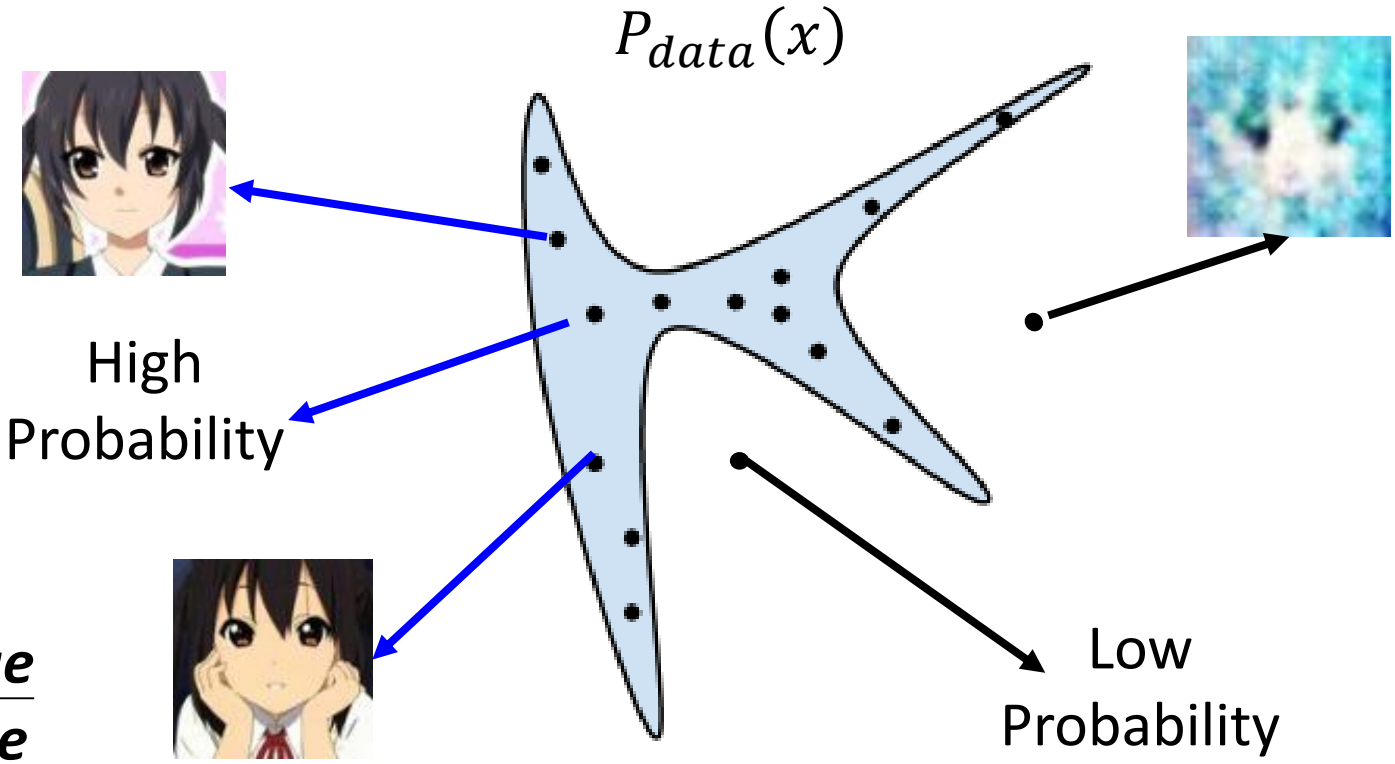
Using Generative Adversarial Network (GAN)

Drawing?

Generation

x: an image (a high-dimensional vector)

- We want to find data distribution $P_{data}(x)$



Maximum Likelihood Estimation

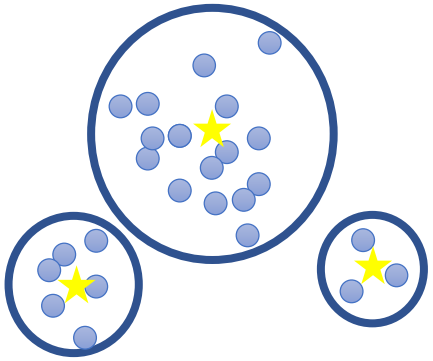
- Given a data distribution $P_{data}(x)$ (We can sample from it.)
- We have a distribution $P_G(x;\theta)$ parameterized by θ
  - We want to find θ such that $P_G(x;\theta)$ close to $P_{data}(x)$
  - E.g. $P_G(x;\theta)$ is a Gaussian Mixture Model, θ are means and variances of the Gaussians

Sample $\{x^1, x^2, \dots, x^m\}$ from $P_{data}(x)$

We can compute $P_G(x^i;\theta)$

Likelihood of generating the samples

$$L = \prod_{i=1}^{m} P_G(x^i;\theta)$$



Find θ^* maximizing the likelihood

Maximum Likelihood Estimation = Minimize KL Divergence

$$\theta^* = arg\max_\theta \prod_{i=1}^{m} P_G(x^i;\theta) = arg\max_\theta log \prod_{i=1}^{m} P_G(x^i;\theta)$$

$$= arg\max_\theta \sum_{i=1}^{m} logP_G(x^i;\theta)$$

$\{x^1, x^2, \ldots, x^m\}$ from $P_{data}(x)$

$$\approx arg\max_\theta E_{x\sim P_{data}}[logP_G(x;\theta)]$$

$$= arg\max_\theta \int_x P_{data}(x)logP_G(x;\theta)dx - \int_x P_{data}(x)logP_{data}(x)dx$$

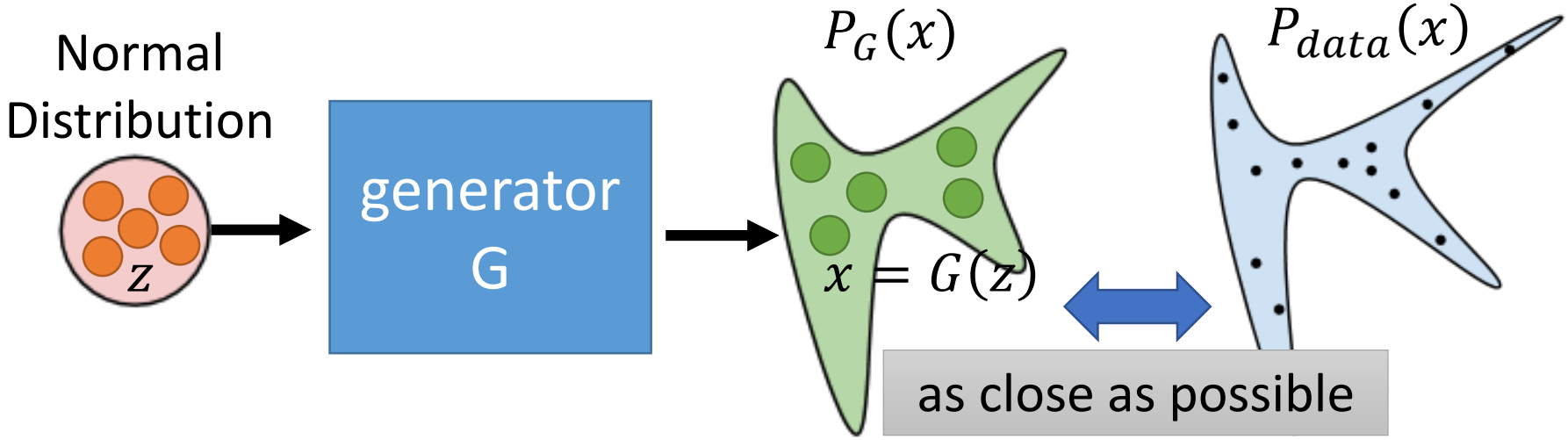
$$= arg\min_\theta KL(P_{data}||P_G)$$

How to define a general P_G?

Generator

x: an image (a high-dimensional vector)

- A generator G is a network. The network defines a probability distribution P_G



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

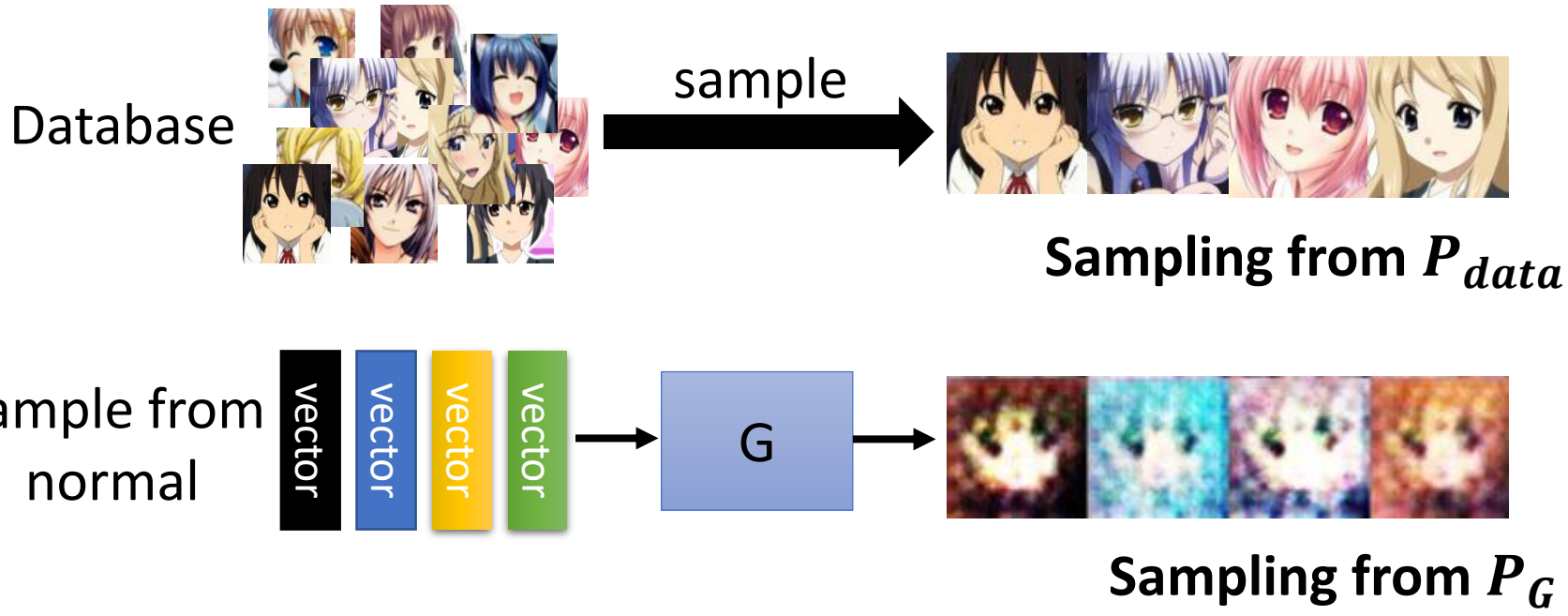
Divergence between distributions P_G and P_{data}

How to compute the divergence?

Discriminator

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

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



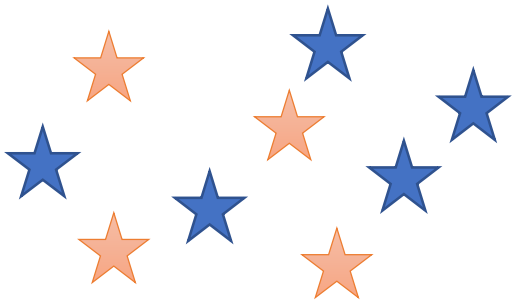
Discriminator

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

: data sampled from P_{data}

: data sampled from P_G

Using the example objective function is exactly the same as training a binary classifier.



train

Discriminator

Sigmoid Output

Example Objective Function for D

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

(G is fixed)

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

The maximum objective value is related to JS divergence.

[Goodfellow, et al., NIPS, 2014]

Discriminator

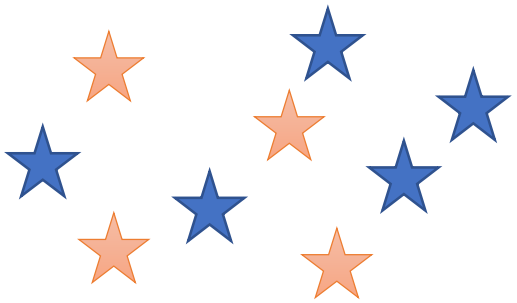
$$G^* = arg \min_G Div(P_G, P_{data})$$

: data sampled from P_{data}

: data sampled from P_G

Training:

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

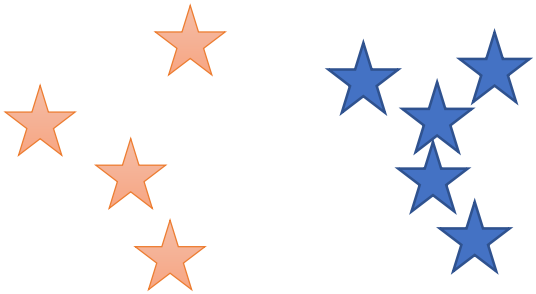


small divergence

train

Discriminator

hard to discriminate
(cannot make objective large)



large divergence

train

Discriminator

easy to discriminate

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

$$V = E_{x\sim P_{data}}[logD(x)] + E_{x\sim P_G}\left[log\left(1-D(x)\right)\right]$$

- Given G, what is the optimal D* maximizing

$$V = E_{x\sim P_{data}}[logD(x)] + E_{x\sim P_G}\left[log\left(1-D(x)\right)\right]$$

$$= \int_x P_{data}(x)logD(x)\,dx + \int_x P_G(x)log\left(1-D(x)\right)dx$$

$$= \int_x \left[P_{data}(x)logD(x) + P_G(x)log\left(1-D(x)\right)\right]dx$$

Assume that D(x) can be any function

- Given x, the optimal D* maximizing

$$P_{data}(x)logD(x) + P_G(x)log\left(1-D(x)\right)$$

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

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

- Given x, the optimal D* maximizing

$$P_{data}(x)logD(x) + P_G(x)log(1-D(x))$$

a D b D

- Find D* maximizing: $f(D) = alog(D) + blog(1-D)$

$$\frac{df(D)}{dD} = a\times\frac{1}{D} + b\times\frac{1}{1-D}\times(-1) = 0$$

$$a\times\frac{1}{D^*} = b\times\frac{1}{1-D^*} \qquad a\times(1-D^*) = b\times D^*$$

$$a - aD^* = bD^* \qquad a = (a+b)D^*$$

$$D^* = \frac{a}{a+b} \Rightarrow D^*(x) = \frac{P_{data}(x)}{P_{data}(x)+P_G(x)}$$

$0 <$ < 1

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

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

$$\max_D V(G,D) = V(G,D^*) \qquad D^*(x) = \frac{P_{data}(x)}{P_{data}(x)+P_G(x)}$$

$$= E_{x\sim P_{data}}\left[\log \frac{P_{data}(x)}{P_{data}(x)+P_G(x)}\right] + E_{x\sim P_G}\left[\log \frac{P_G(x)}{P_{data}(x)+P_G(x)}\right]$$

$$= \int_x P_{data}(x)\log \frac{\frac{1}{2}P_{data}(x)}{\frac{P_{data}(x)+P_G(x)}{2}}dx + \int_x P_G(x)\log \frac{\frac{1}{2}P_G(x)}{\frac{P_{data}(x)+P_G(x)}{2}}dx$$

$$+2\log\frac{1}{2} \quad -2\log 2$$

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

$$\mathrm{JSD}(P \parallel Q) = \frac{1}{2}D(P \parallel M) + \frac{1}{2}D(Q \parallel M)$$

$$M = \frac{1}{2}(P+Q)$$

$$\max_D V(G,D) = V(G,D^*) \qquad D^*(x) = \frac{P_{data}(x)}{P_{data}(x) + P_G(x)}$$

$$= -2log2 + \int_x P_{data}(x) log \frac{P_{data}(x)}{\big(P_{data}(x) + P_G(x)\big)/2} dx$$

$$+ \int_x P_G(x) log \frac{P_G(x)}{\big(P_{data}(x) + P_G(x)\big)/2} dx$$

$$= -2\log 2 + \mathrm{KL}\left(\mathrm{P_{data}} || \frac{\mathrm{P_{data}} + \mathrm{P_G}}{2}\right) + \mathrm{KL}\left(\mathrm{P_G} || \frac{\mathrm{P_{data}} + \mathrm{P_G}}{2}\right)$$

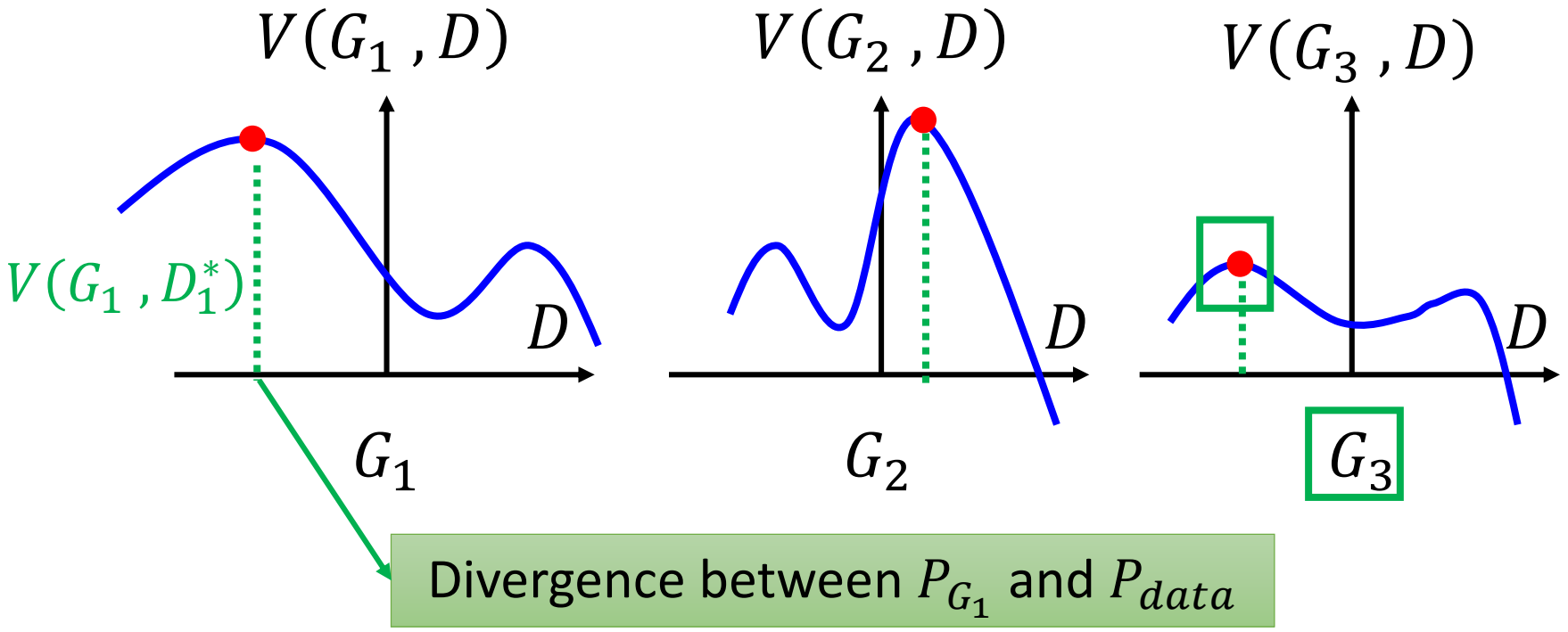
$$= -2log2 + 2JSD(P_{data}||P_G)$$

Jensen-Shannon divergence

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

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

The maximum objective value is related to JS divergence.



[Goodfellow, et al., NIPS, 2014]

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

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

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

Algorithm

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

$L(G)$

- To find the best G minimizing the loss function $L(G)$,

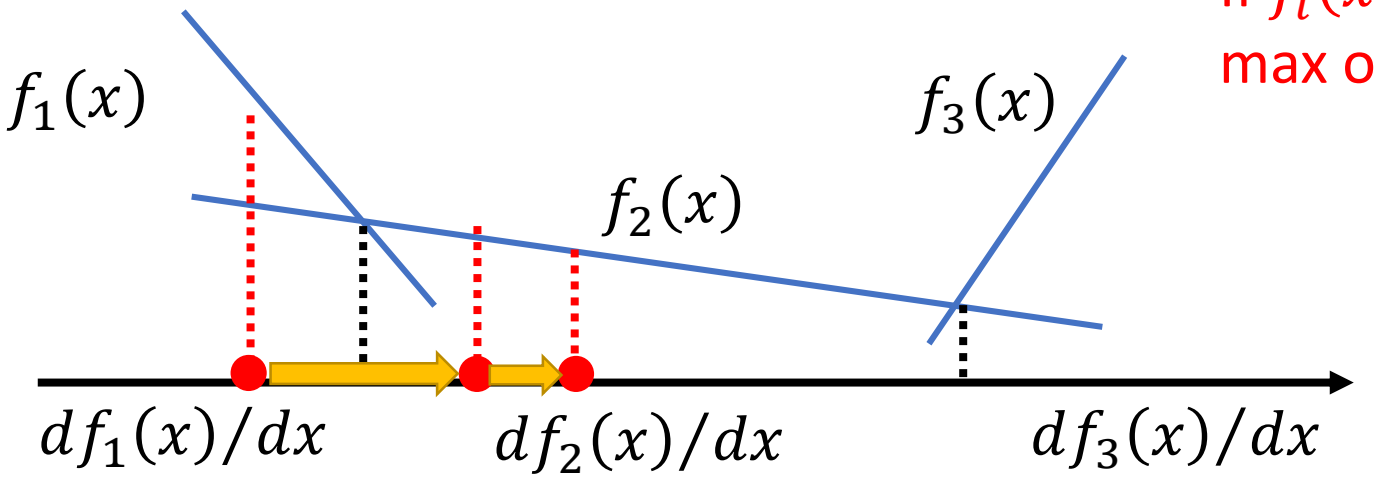
$$\theta_G \leftarrow \theta_G - \eta \, \partial L(G)/\partial \theta_G$$

θ_G defines G

$$f(x) = \max\{f_1(x), f_2(x), f_3(x)\}$$

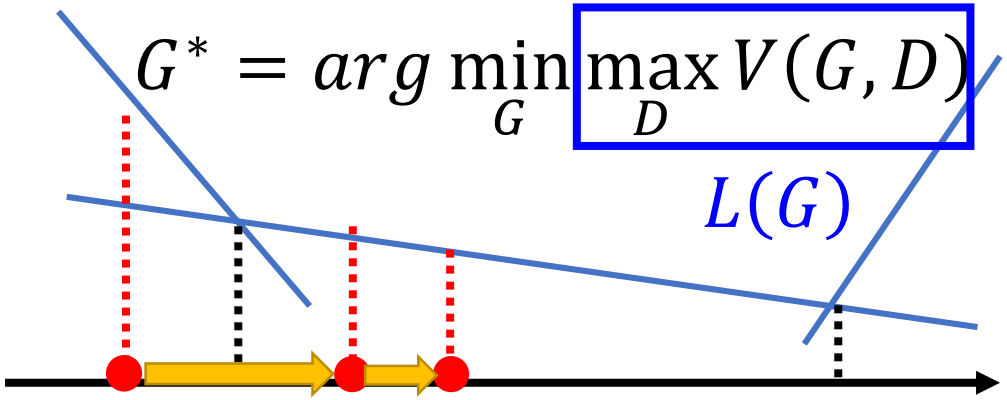
$$\frac{df(x)}{dx} =? \; df_i(x)/dx$$

If $f_i(x)$ is the max one



Algorithm

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



- Given G_0
- Find D_0^* maximizing $V(G_0, D)$ **Using Gradient Ascent**

$V(G_0, D_0^*)$ is the JS divergence between $P_{data}(x)$ and $P_{G_0}(x)$

- $\theta_G \leftarrow \theta_G - \eta \, \partial V(G, D_0^*)/\partial \theta_G$ → Obtain G_1
- Find D_1^* maximizing $V(G_1, D)$

Decrease JS divergence(?)

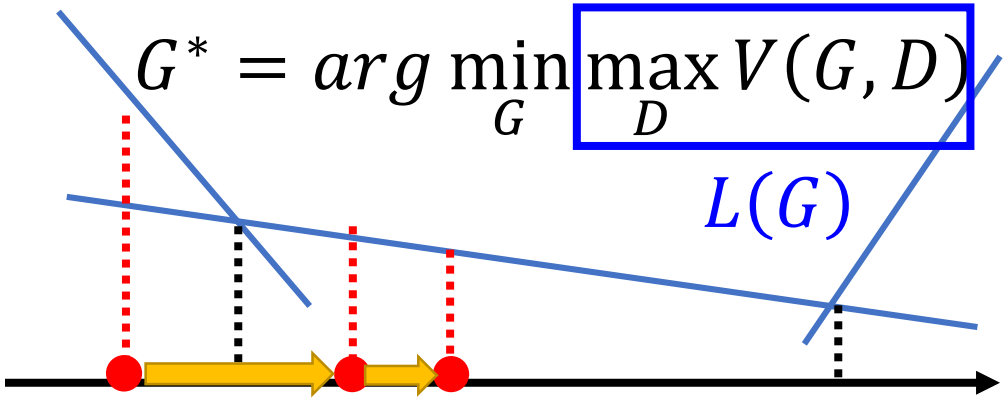
$V(G_1, D_1^*)$ is the JS divergence between $P_{data}(x)$ and $P_{G_1}(x)$

- $\theta_G \leftarrow \theta_G - \eta \, \partial V(G, D_1^*)/\partial \theta_G$ → Obtain G_2
- ……

Decrease JS divergence(?)

Algorithm

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

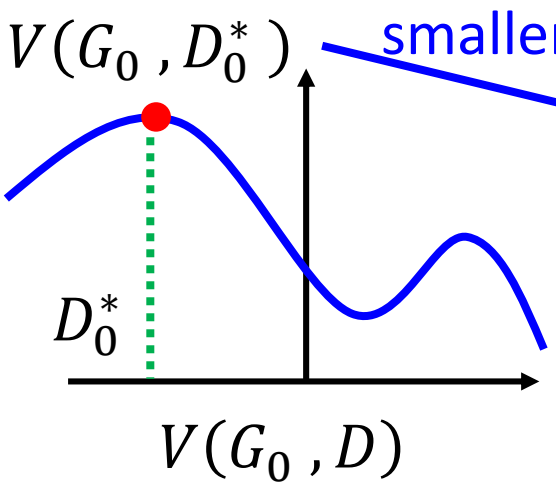


- Given G_0
- Find D_0^* maximizing $V(G_0, D)$

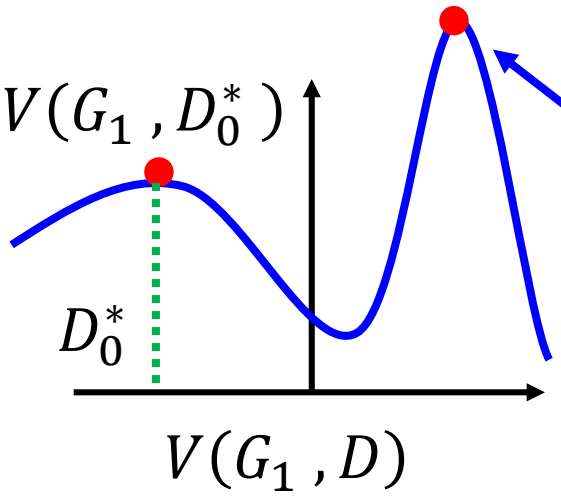
$V(G_0, D_0^*)$ is the JS divergence between $P_{data}(x)$ and $P_{G_0}(x)$

- $\theta_G \leftarrow \theta_G - \eta \, \partial V(G, D_0^*)/\partial \theta_G$ ➡ Obtain G_1

Decrease JS divergence(?)



smaller



$V(G_1, D_1^*)$ ……

Assume $D_0^* \approx D_1^*$

Don't update G too much

In practice …

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

- Given G, how to compute $\max_D V(G, D)$
  - Sample $\{x^1, x^2, \ldots, x^m\}$ from $P_{data}(x)$, sample $\{\tilde{x}^1, \tilde{x}^2, \ldots, \tilde{x}^m\}$ from generator $P_G(x)$

Maximize $\tilde{V} = \frac{1}{m}\sum_{i=1}^{m} logD(x^i) + \frac{1}{m}\sum_{i=1}^{m} log\left(1 - D(\tilde{x}^i)\right)$

=

Binary Classifier

D is a binary classifier with sigmoid output (can be deep)

$\{x^1, x^2, \ldots, x^m\}$ from $P_{data}(x)$ → Positive examples

$\{\tilde{x}^1, \tilde{x}^2, \ldots, \tilde{x}^m\}$ from $P_G(x)$ → Negative examples

Minimize Cross-entropy

Algorithm

Initialize θ_d for D and θ_g for G

Can only find lower found of $\max_D V(G,D)$

- In each training iteration:

Learning D

Repeat k times

- Sample m examples $\{x^1, x^2, \dots, x^m\}$ from data distribution $P_{data}(x)$
- Sample m noise samples $\{z^1, z^2, \dots, z^m\}$ from the prior $P_{prior}(z)$
- Obtaining generated data $\{\tilde{x}^1, \tilde{x}^2, \dots, \tilde{x}^m\}$, $\tilde{x}^i = G(z^i)$
- Update discriminator parameters θ_d to maximize
  - $\tilde{V} = \frac{1}{m}\sum_{i=1}^{m} logD(x^i) + \frac{1}{m}\sum_{i=1}^{m} log\left(1 - D(\tilde{x}^i)\right)$
  - $\theta_d \leftarrow \theta_d + \eta\nabla\tilde{V}(\theta_d)$

Learning G

Only Once

- Sample another m noise samples $\{z^1, z^2, \dots, z^m\}$ from the prior $P_{prior}(z)$
- Update generator parameters θ_g to minimize
  - $\tilde{V} = \cancel{\frac{1}{m}\sum_{i=1}^{m} logD(x^i)} + \frac{1}{m}\sum_{i=1}^{m} log\left(1 - D\left(G(z^i)\right)\right)$
  - $\theta_g \leftarrow \theta_g - \eta\nabla\tilde{V}(\theta_g)$

Objective Function for Generator in Real Implementation

$$V = E_{x \sim P_{data}}[logD(x)]$$

$$+E_{x \sim P_G}[log(1 - D(x))]$$

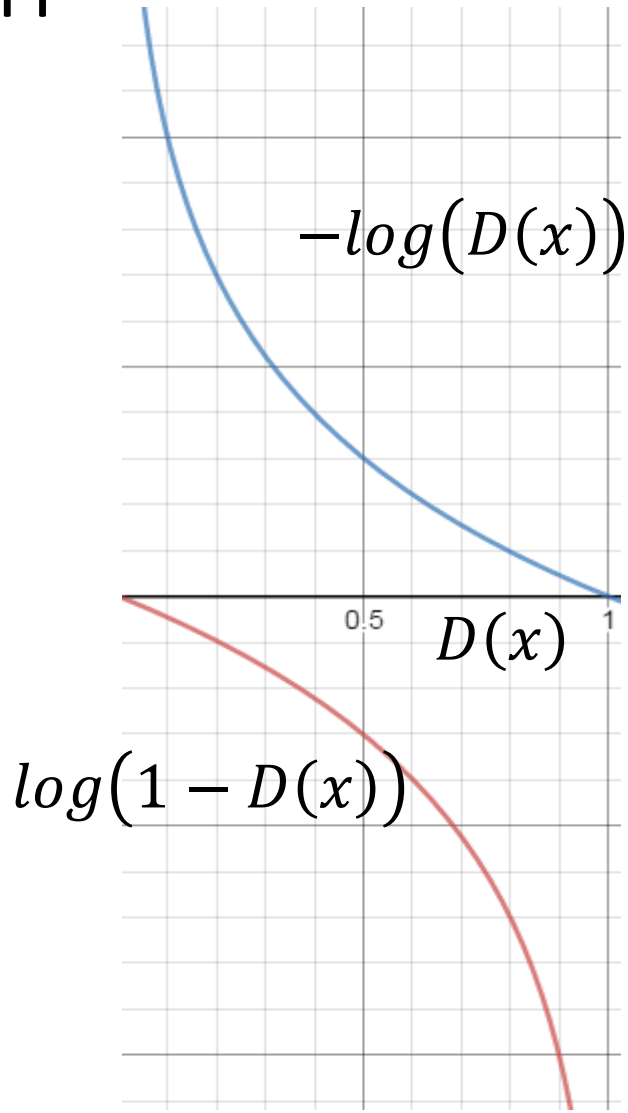
Slow at the beginning

Minimax GAN (MMGAN)

$$V = E_{x \sim P_G}[-log(D(x))]$$

Real implementation:
label x from P_G as positive

Non-saturating GAN (NSGAN)



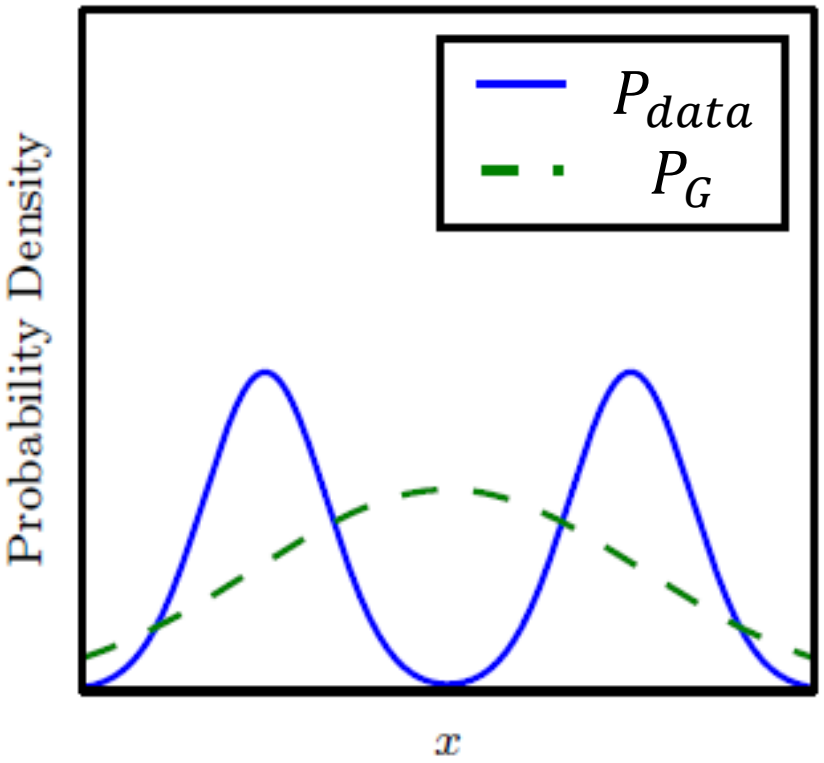
fGAN: General Framework of GAN

Modified from Ian Goodfellow's tutorial

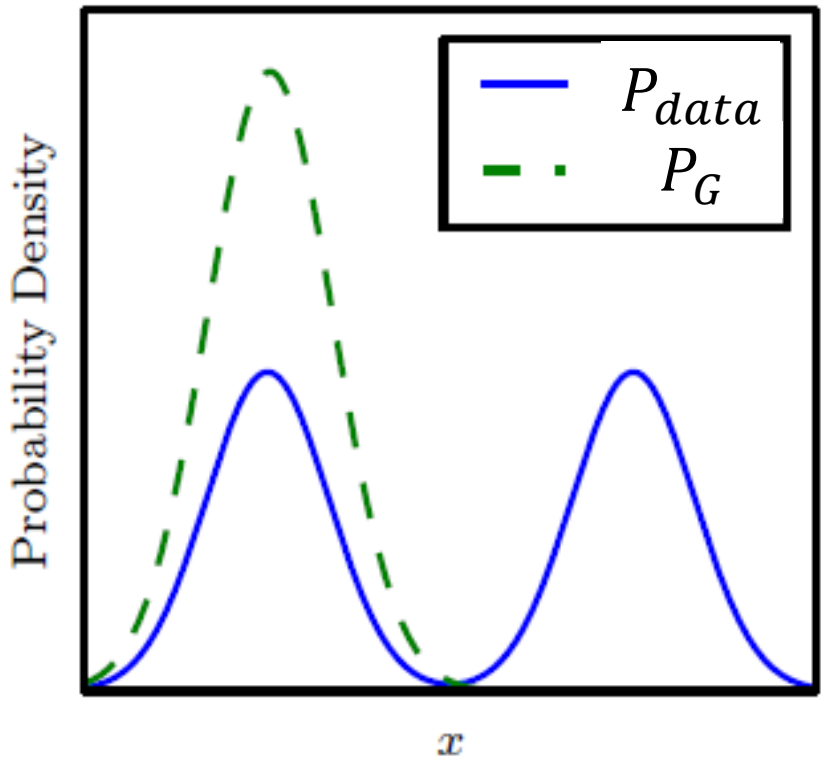
Flaw in Optimization?

$$KL = \int P_{data} log \frac{P_{data}}{P_G} dx$$

Reverse $KL = \int P_G log \frac{P_G}{P_{data}} dx$



Maximum likelihood
(minimize KL($P_{data}||P_G$))



Minimize KL($P_G||P_{data}$)
(reverse KL)

f-divergence

P and Q are two distributions. $p(x)$ and $q(x)$ are the probability of sampling x.

$$D_f(P||Q) = \int_x q(x) f\left(\frac{p(x)}{q(x)}\right) dx$$

f is convex

f(1) = 0

$D_f(P||Q)$ evaluates the difference of P and Q

If $p(x) = q(x)$ for all x

smallest $\quad D_f(P||Q) = \int_x q(x) f\left(\underbrace{\left(\frac{p(x)}{q(x)}\right)}_{=1}\right) dx = \underbrace{0}_{=0}$

$$D_f(P||Q) = \int_x q(x) f\left(\frac{p(x)}{q(x)}\right) dx$$

Because f is convex

$$\geq f\left(\int_x \cancel{q(x)} \frac{p(x)}{\cancel{q(x)}} dx\right)$$

$$= f(1) = 0$$

If P and Q are the same distributions,

$D_f(P||Q)$ has the smallest value, which is 0

f-divergence

$$D_f(P||Q) = \int_x q(x) f\left(\frac{p(x)}{q(x)}\right) dx$$

f is convex

f(1) = 0

$f(x) = xlogx$

KL

$$D_f(P||Q) = \int_x q(x)\frac{p(x)}{q(x)} log\left(\frac{p(x)}{q(x)}\right) dx = \int_x p(x) log\left(\frac{p(x)}{q(x)}\right) dx$$

$f(x) = -logx$

Reverse KL

$$D_f(P||Q) = \int_x q(x)\left(-log\left(\frac{p(x)}{q(x)}\right)\right) dx = \int_x q(x) log\left(\frac{q(x)}{p(x)}\right) dx$$

$f(x) = (x-1)^2$

Chi Square

$$D_f(P||Q) = \int_x q(x)\left(\frac{p(x)}{q(x)} - 1\right)^2 dx = \int_x \frac{\left(p(x) - q(x)\right)^2}{q(x)} dx$$

Fenchel Conjugate

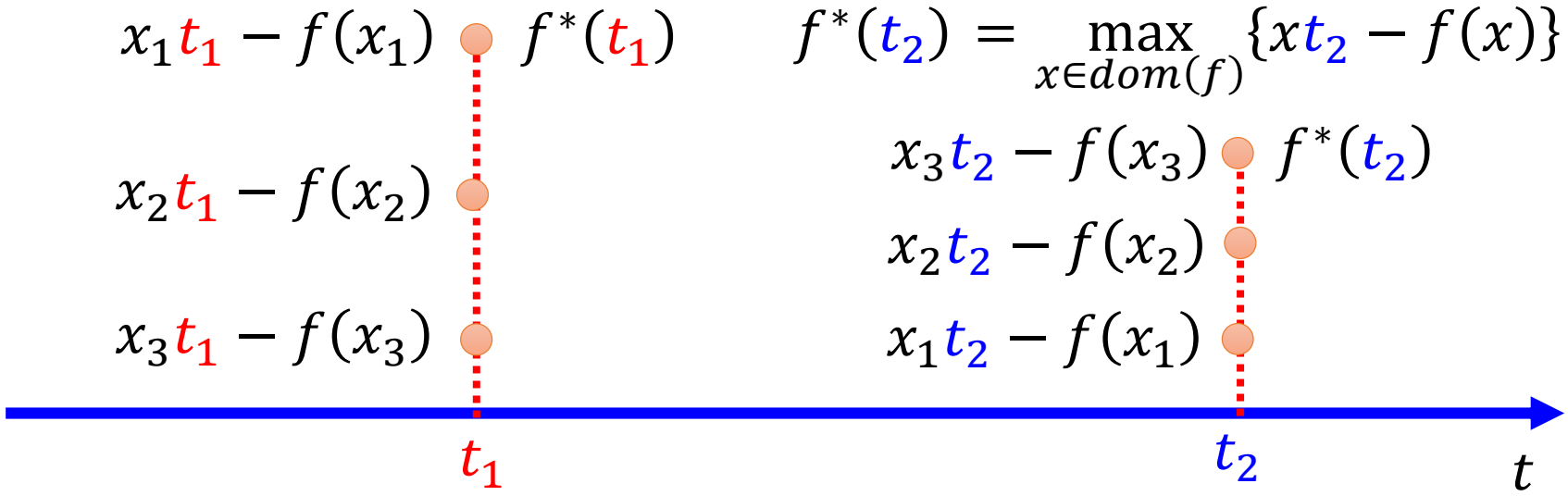
$$D_f(P||Q) = \int_x q(x) f\left(\frac{p(x)}{q(x)}\right) dx$$

f is convex, f(1) = 0

- Every convex function f has a conjugate function f*

$$f^*(t) = \max_{x \in dom(f)} \{xt - f(x)\}$$

$$f^*(t_1) = \max_{x \in dom(f)} \{xt_1 - f(x)\}$$



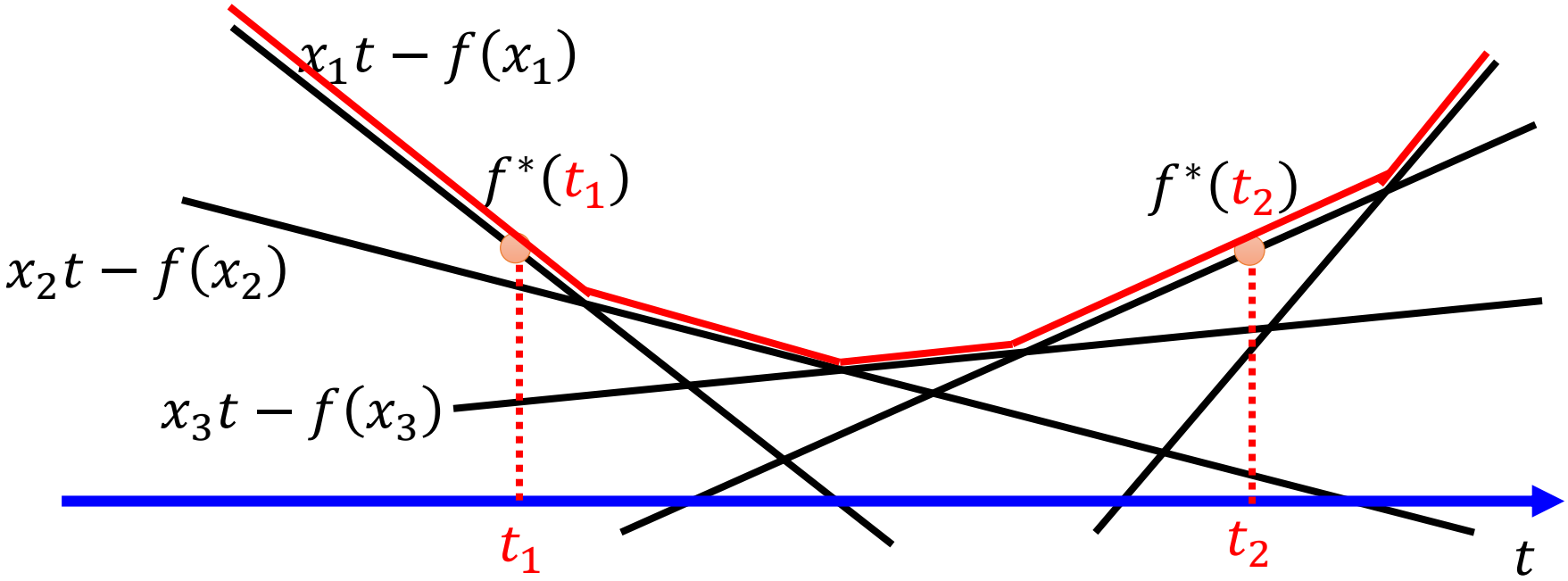
Fenchel Conjugate

$$D_f(P||Q) = \int_x q(x) f\left(\frac{p(x)}{q(x)}\right) dx$$

f is convex, f(1) = 0

- Every convex function f has a conjugate function f*

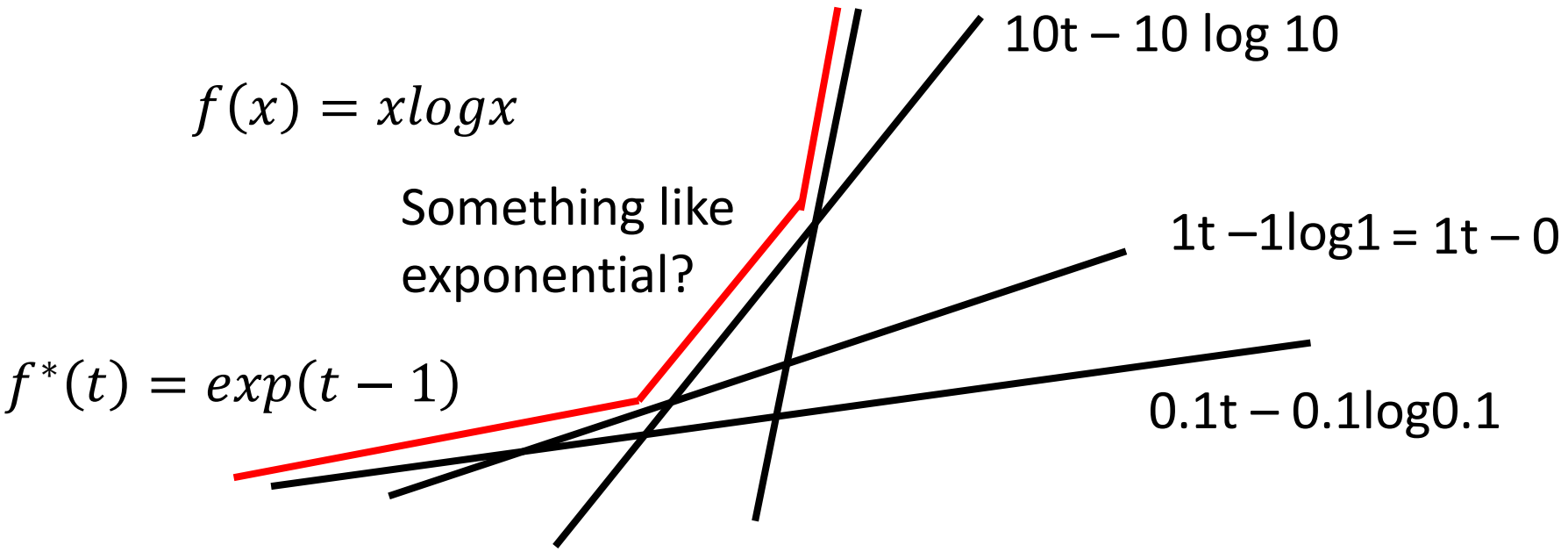
$$f^*(t) = \max_{x \in dom(f)} \{xt - f(x)\}$$



Fenchel Conjugate

- Every convex function f has a conjugate function f*

$$f^*(t) = \max_{x \in dom(f)} \{xt - f(x)\}$$



Fenchel Conjugate

- Every convex function f has a conjugate function f*
- (f*)* = f

$$f^*(t) = \max_{x \in dom(f)} \{xt - f(x)\}$$

$$f(x) = xlogx \longleftrightarrow f^*(t) = exp(t-1)$$

$$f^*(t) = \max_{x \in dom(f)} \{xt - xlogx\}$$

$g(x) = xt - xlogx$ Given t, find x maximizing $g(x)$

$t - logx - 1 = 0 \qquad x = exp(t-1)$

$$f^*(t) = exp(t-1) \times t - exp(t-1) \times (t-1) = exp(t-1)$$

Connection with GAN

$$f^*(t) = \max_{x \in dom(f)} \{xt - f(x)\} \longleftrightarrow f(x) = \max_{t \in dom(f^*)} \{xt - f^*(t)\}$$

$\frac{p(x)}{q(x)}$ $\frac{p(x)}{q(x)}$

$$D_f(P||Q) = \int_x q(x) f\left(\frac{p(x)}{q(x)}\right) dx$$

$$= \int_x q(x) \left(\max_{t \in dom(f^*)} \left\{ \frac{p(x)}{q(x)} t - f^*(t) \right\} \right) dx$$

$$\approx \max_D \int_x p(x)D(x)dx - \int_x q(x) f^*\big(D(x)\big)dx$$

D is a function whose input is x, and output is t

$$D_f(P||Q) \geq \int_x q(x) \left(\frac{p(x)}{q(x)} D(x) - f^*\big(D(x)\big) \right) dx$$

$$= \int_x p(x)D(x)dx - \int_x q(x) f^*\big(D(x)\big)dx$$

Connection with GAN

$$D_f(P||Q) \approx \max_D \int_x p(x)D(x)dx - \int_x q(x)f^*\big(D(x)\big)dx$$

$$= \max_D \{E_{x\sim P}[D(x)] - E_{x\sim Q}[f^*\big(D(x)\big)]\}$$

Samples from P Samples from Q

$$D_f(P_{data}||P_G) = \max_D \{E_{x\sim P_{data}}[D(x)] - E_{x\sim P_G}[f^*\big(D(x)\big)]\}$$

$$G^* = arg\min_G D_f(P_{data}||P_G)$$

Original GAN has different V(G,D)

$$= arg\min_G \max_D \{E_{x\sim P_{data}}[D(x)] - E_{x\sim P_G}[f^*\big(D(x)\big)]\}$$

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

familiar? ☺

$$D_f(P_{data}||P_G) = \max_{\mathrm{D}}\{E_{x\sim P_{data}}[D(x)] - E_{x\sim P_G}[f^*(D(x))]\}$$

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

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 f-divergence you like ☺

https://arxiv.org/pdf/1606.00709.pdf

Tips for Improving GAN

Martin Arjovsky, Soumith Chintala, Léon Bottou, Wasserstein GAN, arXiv prepring, 2017

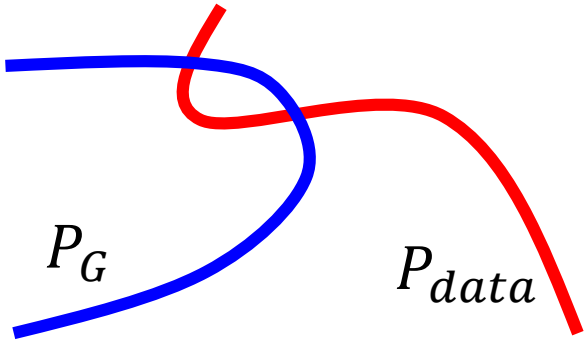
Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, Aaron Courville, “Improved Training of Wasserstein GANs”, arXiv prepring, 2017

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.

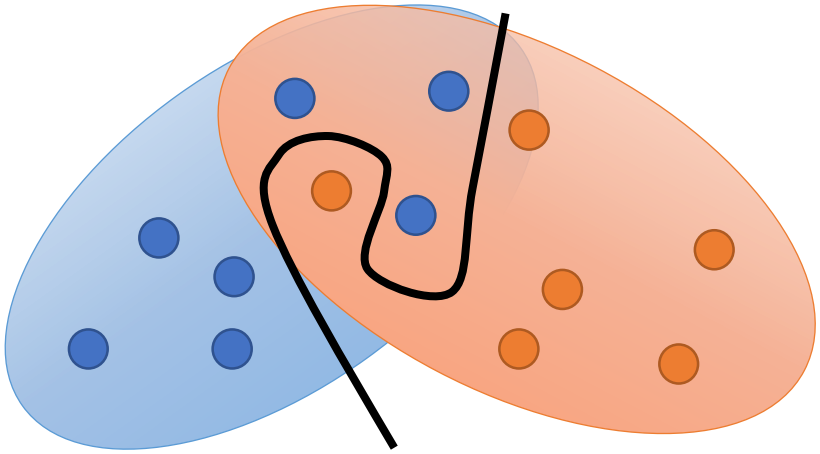
  The overlap can be ignored.



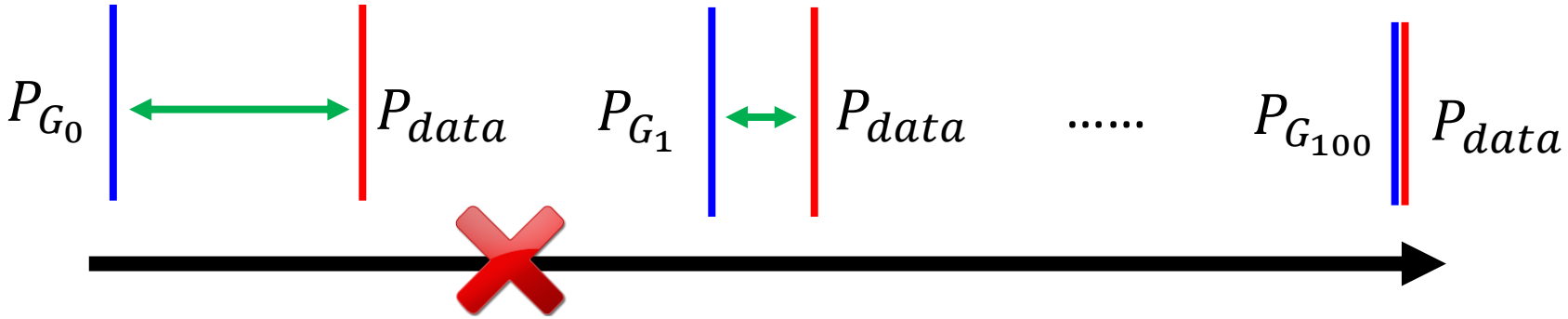
- 2. Sampling

  Even though P_{data} and P_G have overlap.

  If you do not have enough sampling ……



What is the problem of JS divergence?



Equally bad

$JS(P_{G_0}, P_{data}) = log2$

$JS(P_{G_1}, P_{data}) = log2$

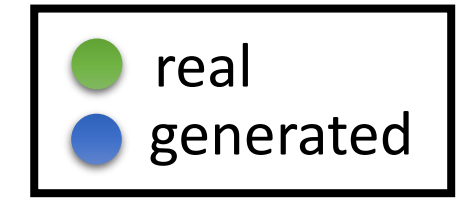
..... $JS(P_{G_{100}}, P_{data}) = 0$

JS divergence is log2 if two distributions do not overlap.

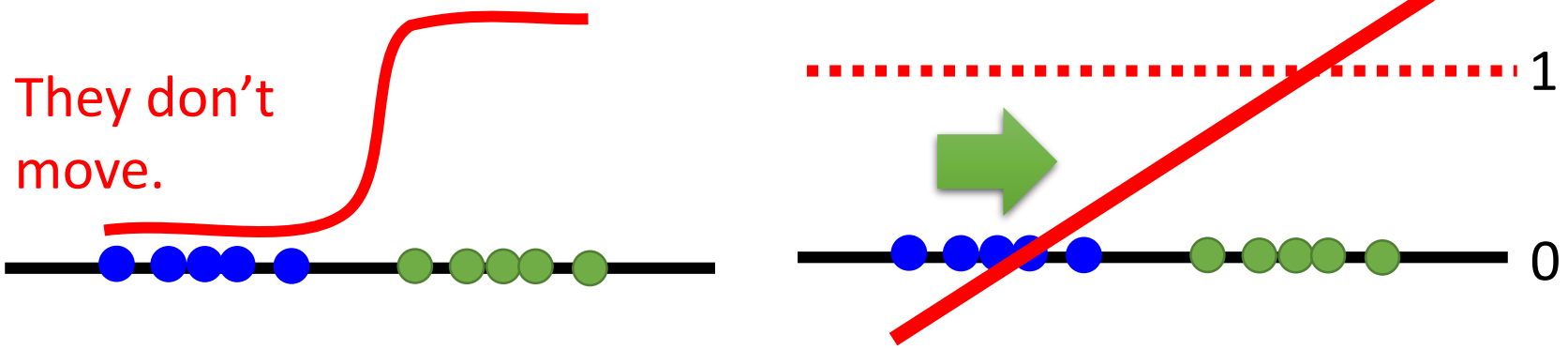
Intuition: If two distributions do not overlap, binary classifier achieves 100% accuracy

➡ Same objective value is obtained. ➡ Same divergence

Least Square GAN (LSGAN)

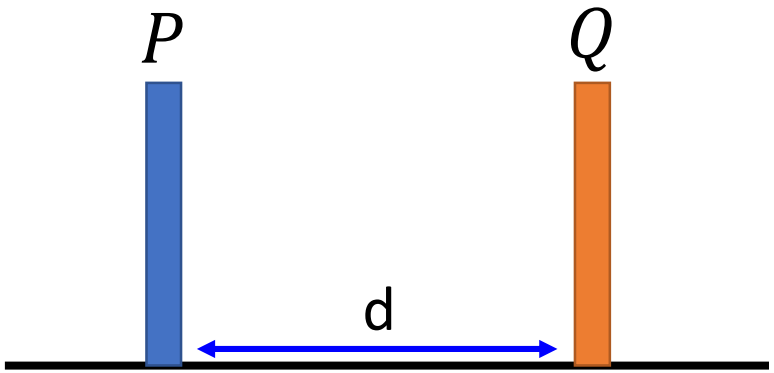


- Replace sigmoid with linear (replace classification with regression)



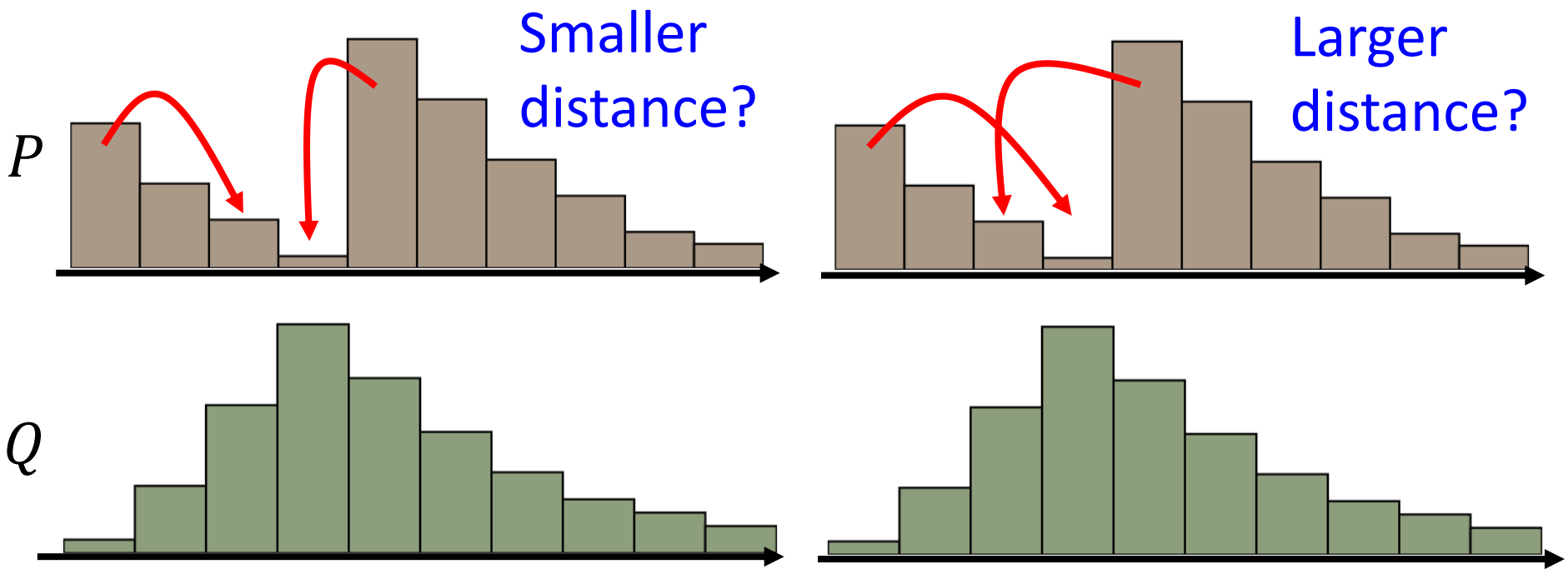
Wasserstein GAN (WGAN): Earth Mover's Distance

- 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.



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

WGAN: Earth Mover's Distance



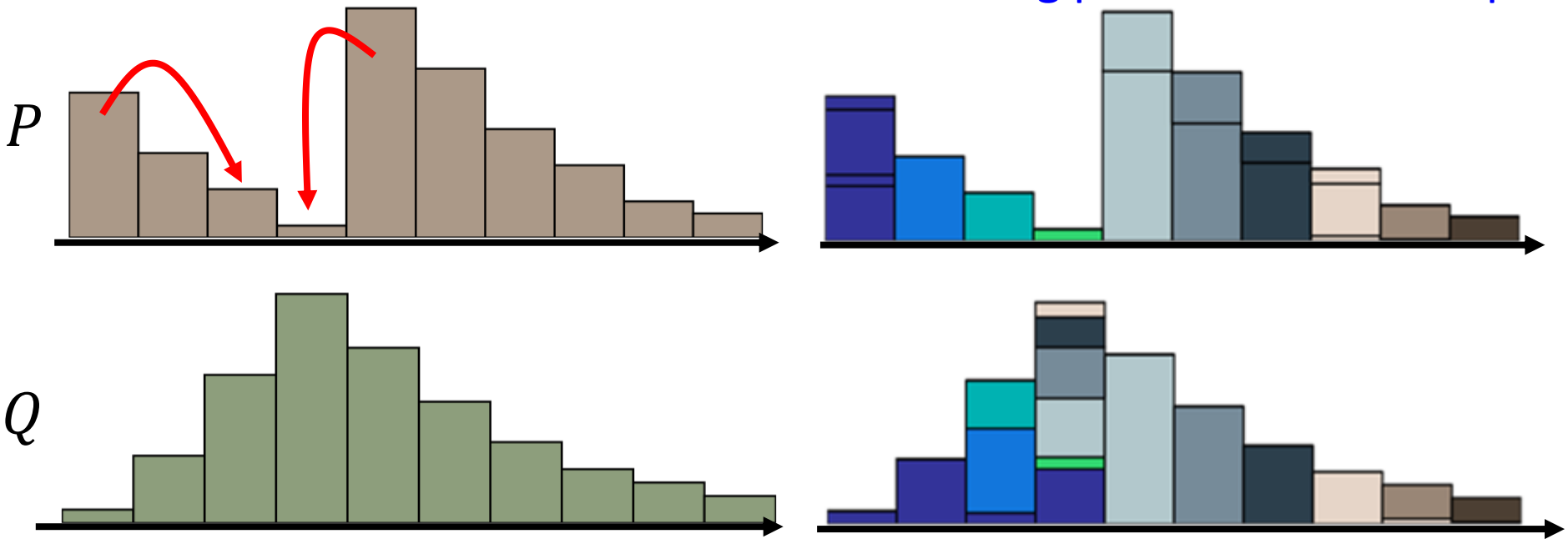
There many possible "moving plans".

Using the "moving plan" with the smallest average distance to define the earth mover's distance.

Source of image: https://vincentherrmann.github.io/blog/wasserstein/

WGAN: Earth Mover's Distance

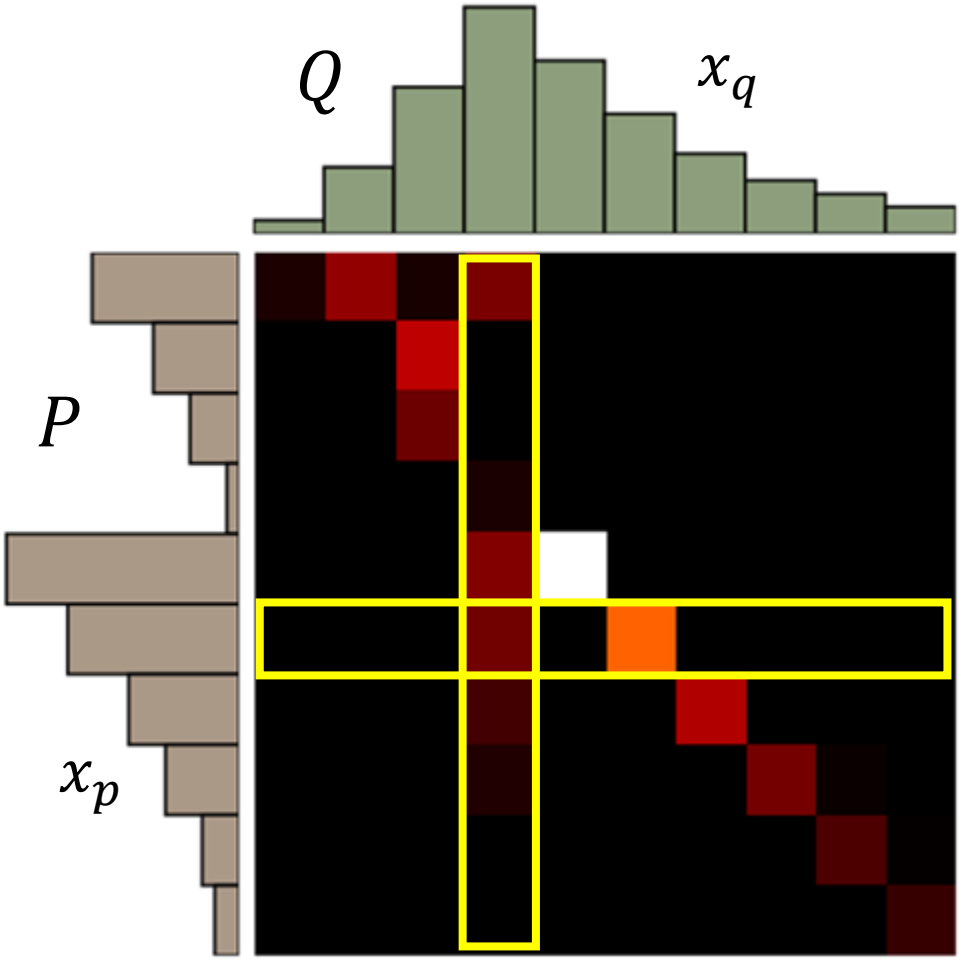
Best "moving plans" of this example



There many possible "moving plans".

Using the "moving plan" with the smallest average distance to define the earth mover's distance.

Source of image: https://vincentherrmann.github.io/blog/wasserstein/



moving plan γ

All possible plan Π

A “moving plan” is a matrix

The value of the element is the amount of earth from one position to another.

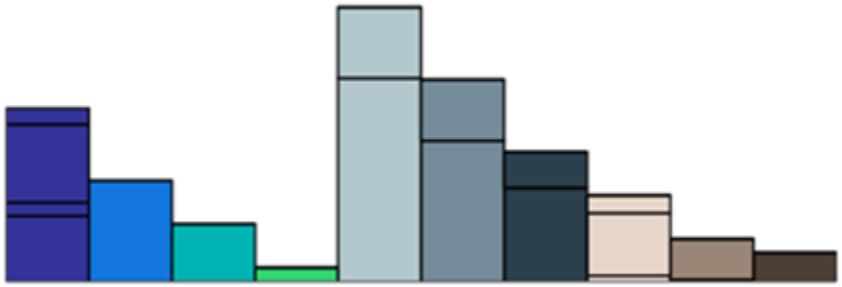
Average distance of a plan γ:

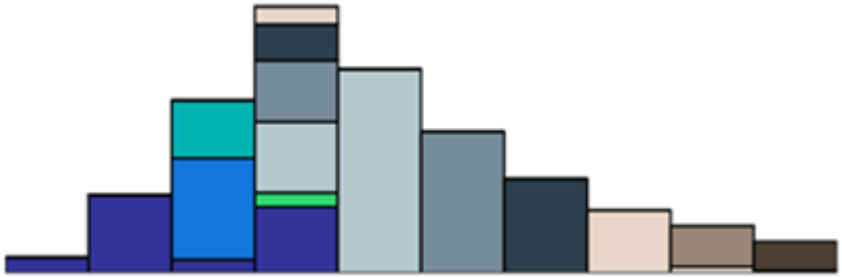
$$B(\gamma) = \sum_{x_p, x_q} \gamma(x_p, x_q) \|x_p - x_q\|$$

Earth Mover’s Distance:

$$W(P, Q) = \min_{\gamma \in \Pi} B(\gamma)$$

The best plan

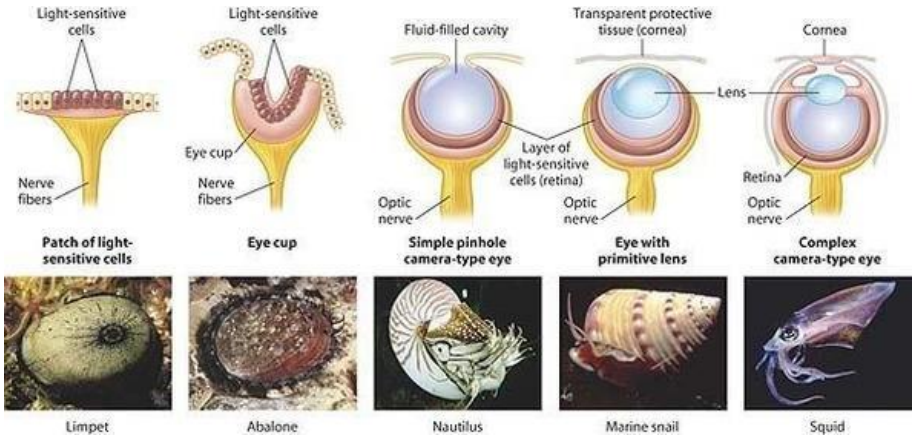


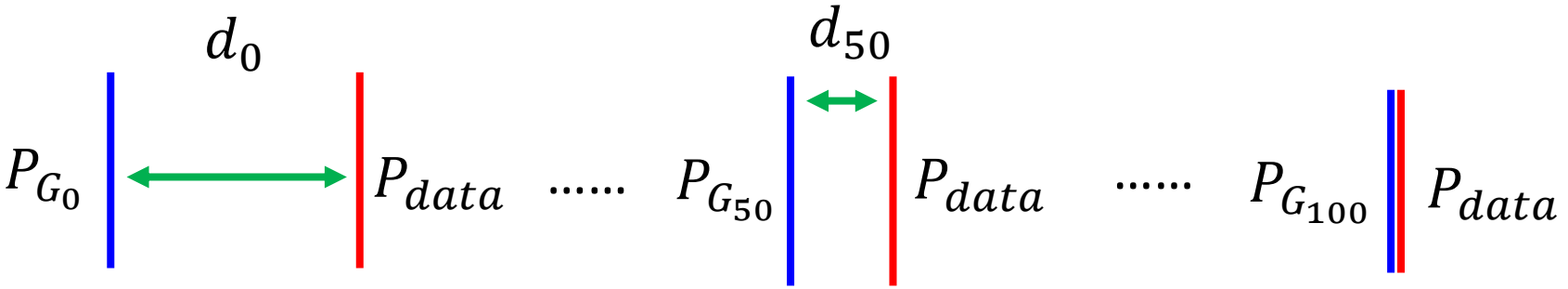


Why Earth Mover's Distance?

$$D_f(P_{data}||P_G)$$

$$W(P_{data}, P_G)$$





$$JS(P_{G_0}, P_{data}) = log2$$

$$JS(P_{G_{50}}, P_{data}) = log2$$

$$JS(P_{G_{100}}, P_{data}) = 0$$

$$W(P_{G_0}, P_{data}) = d_0$$

$$W(P_{G_{50}}, P_{data}) = d_{50}$$

$$W(P_{G_{100}}, P_{data}) = 0$$

[Martin Arjovsky, et al., arXiv, 2017]

WGAN

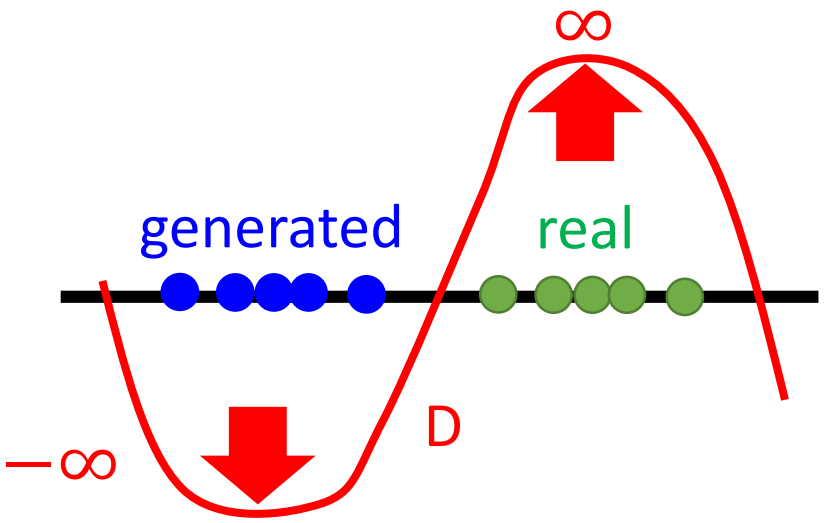
Evaluate wasserstein distance between P_{data} and P_G

$$V(G,D) = \max_{D \in 1-Lipschitz} \left\{ E_{x \sim P_{data}}[D(x)] - E_{x \sim P_G}[D(x)] \right\}$$

D has to be smooth enough.

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

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



Weight Clipping [Martin Arjovsky, et al., arXiv, 2017]

WGAN

Force the parameters w between c and -c

After parameter update, if w > c, w = c;

if w < -c, w = -c

Evaluate wasserstein distance between P_{data} and P_G

$$V(G,D) = \max_{D \in 1-Lipschitz} \{E_{x \sim P_{data}}[D(x)] - E_{x \sim P_G}[D(x)]\}$$

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

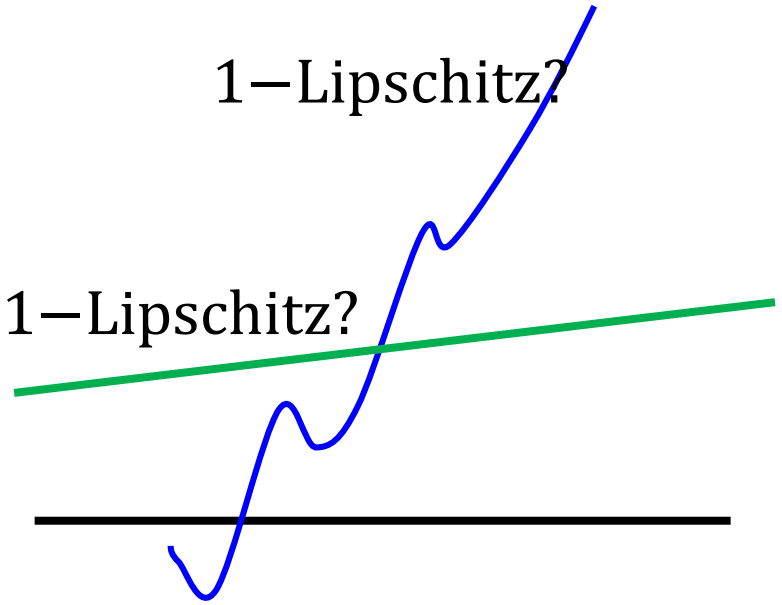
Lipschitz Function

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

Output change / Input change

K=1 for "$1-Lipschitz$"

Do not change fast



Improved WGAN (WGAN-GP)

$$V(G,D) = \max_{D \in 1-Lipschitz} \{E_{x \sim P_{data}}[D(x)] - E_{x \sim P_G}[D(x)]\}$$

A differentiable function is 1-Lipschitz if and only if it has gradients with norm less than or equal to 1 everywhere.

$$D \in 1-Lipschitz \iff \|\nabla_x D(x)\| \leq 1 \text{ for all x}$$

$$V(G,D) \approx \max_D \{E_{x \sim P_{data}}[D(x)] - E_{x \sim P_G}[D(x)] - \lambda \int_x max(0, \|\nabla_x D(x)\| - 1) dx\}$$

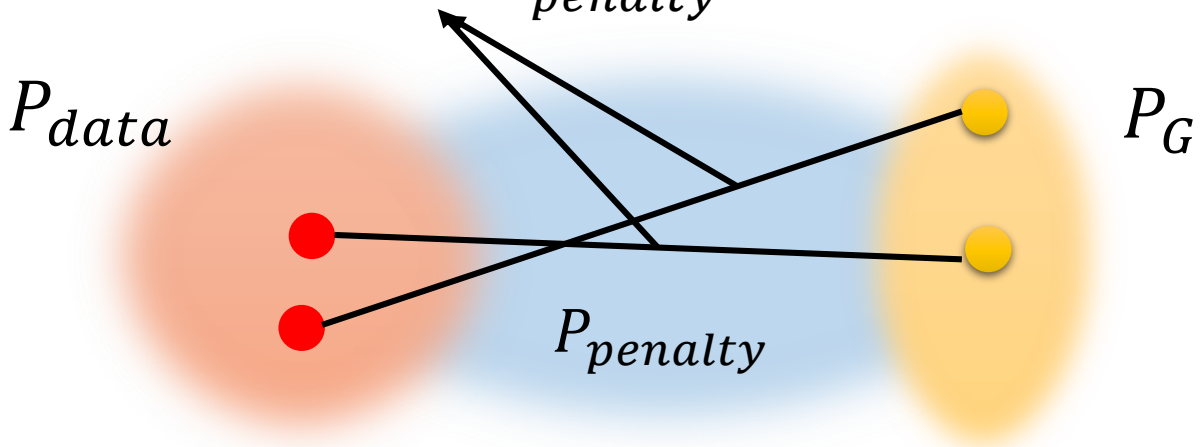
Prefer $\|\nabla_x D(x)\| \leq 1$ for all x

$$-\lambda E_{x \sim P_{penalty}}[max(0, \|\nabla_x D(x)\| - 1)]\}$$

Prefer $\|\nabla_x D(x)\| \leq 1$ for x sampling from $x \sim P_{penalty}$

Improved WGAN (WGAN-GP)

$$V(G,D) \approx \max_{D}\{E_{x\sim P_{data}}[D(x)] - E_{x\sim P_G}[D(x)] - \lambda E_{x\sim P_{penalty}}[max(0, \|\nabla_x D(x)\| - 1)]\}$$



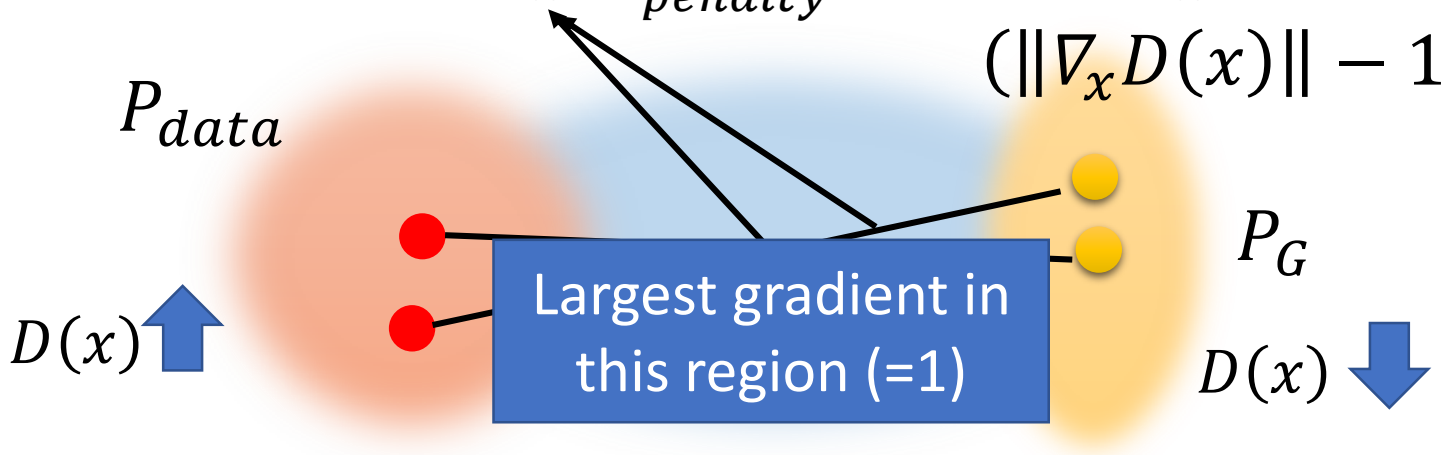
"Given that enforcing the Lipschitz constraint everywhere is intractable, enforcing it ***only along these straight lines*** seems sufficient and experimentally results in good performance."

Only give gradient constraint to the region between P_{data} and P_G because they influence how P_G moves to P_{data}

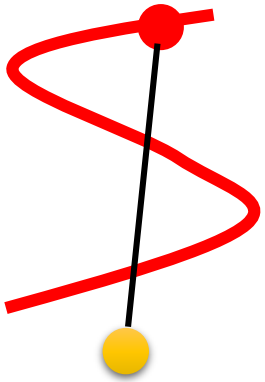
Improved WGAN (WGAN-GP)

$$V(G,D) \approx \max_D \{E_{x\sim P_{data}}[D(x)] - E_{x\sim P_G}[D(x)] - \lambda E_{x\sim P_{penalty}}[\cancel{max(0, \|\nabla_x D(x)\| - 1)}]\}$$

$$(\|\nabla_x D(x)\| - 1)^2$$



"Simply penalizing overly large gradients also works in theory, but experimentally we found that this approach converged faster and to better optima."



Spectrum Norm

Spectral Normalization → Keep gradient norm smaller than 1 everywhere [Miyato, et al., ICLR, 2018]

Algorithm of WGAN

- In each training iteration:

No sigmoid for the output of D

Learning D

Repeat k times

- Sample m examples $\{x^1, x^2, \dots, x^m\}$ from data distribution $P_{data}(x)$
- Sample m noise samples $\{z^1, z^2, \dots, z^m\}$ from the prior $P_{prior}(z)$
- Obtaining generated data $\{\tilde{x}^1, \tilde{x}^2, \dots, \tilde{x}^m\}$, $\tilde{x}^i = G(z^i)$
- Update discriminator parameters θ_d to maximize
  - $\tilde{V} = \frac{1}{m}\sum_{i=1}^{m} D(x^i) - \frac{1}{m}\sum_{i=1}^{m} D(\tilde{x}^i)$
  - $\theta_d \leftarrow \theta_d + \eta \nabla \tilde{V}(\theta_d)$

Weight clipping / Gradient Penalty …

Learning G

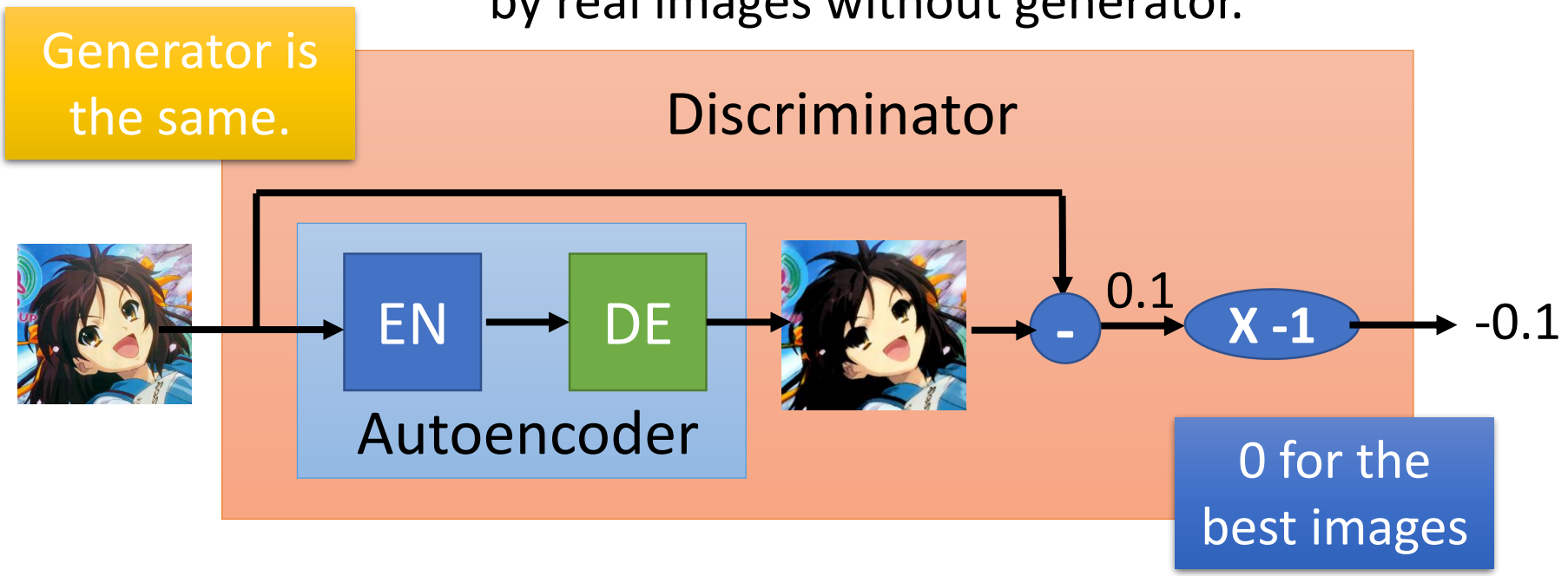
Only Once

- Sample another m noise s … } from the prior $P_{prior}(z)$
- Update generator parameters θ_g to minimize
  - $\tilde{V} = \frac{1}{m}\sum_{i=1}^{m} \log D(x^i) - \frac{1}{m}\sum_{i=1}^{m} D\left(G(z^i)\right)$
  - $\theta_g \leftarrow \theta_g - \eta \nabla \tilde{V}(\theta_g)$

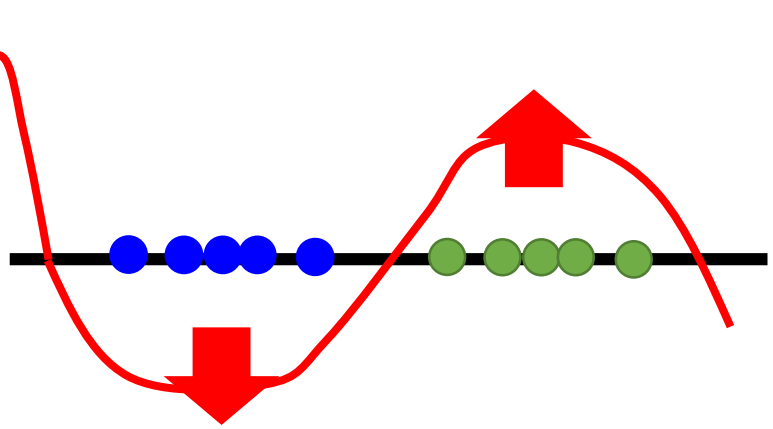
[Junbo Zhao, et al., arXiv, 2016]

Energy-based GAN (EBGAN)

- Using an autoencoder as discriminator D
  - Using the negative reconstruction error of auto-encoder to determine the goodness
  - **Benefit**: The auto-encoder can be pre-train by real images without generator.

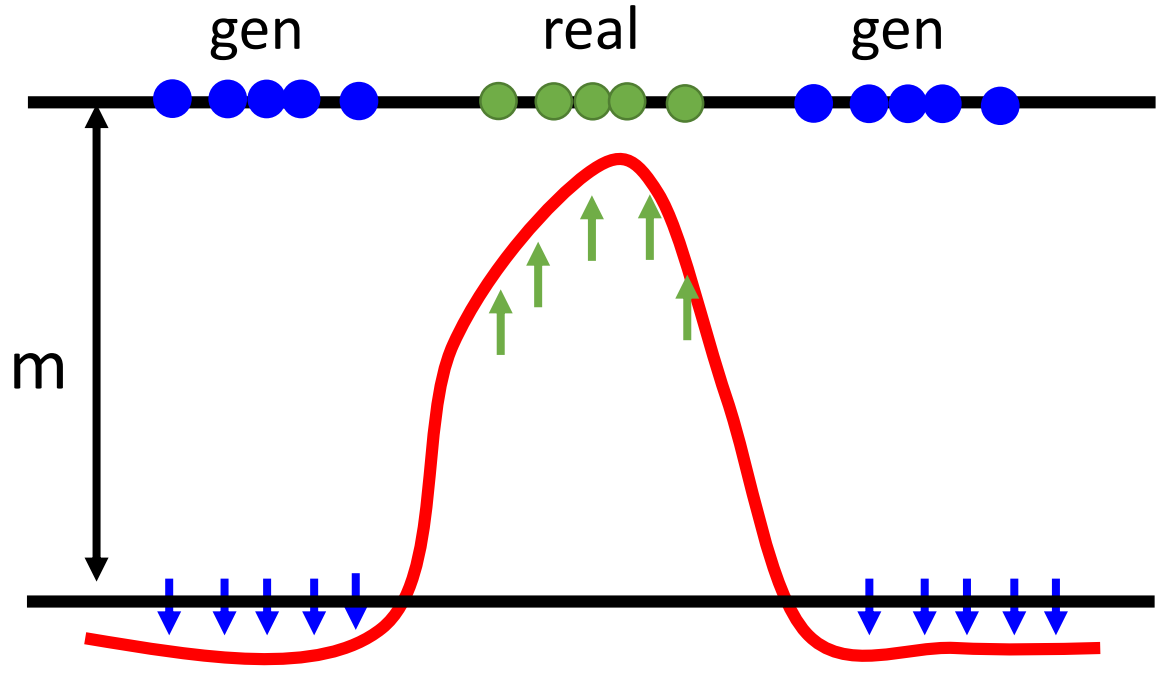


EBGAN



Auto-encoder based discriminator
only gives limited region large value.

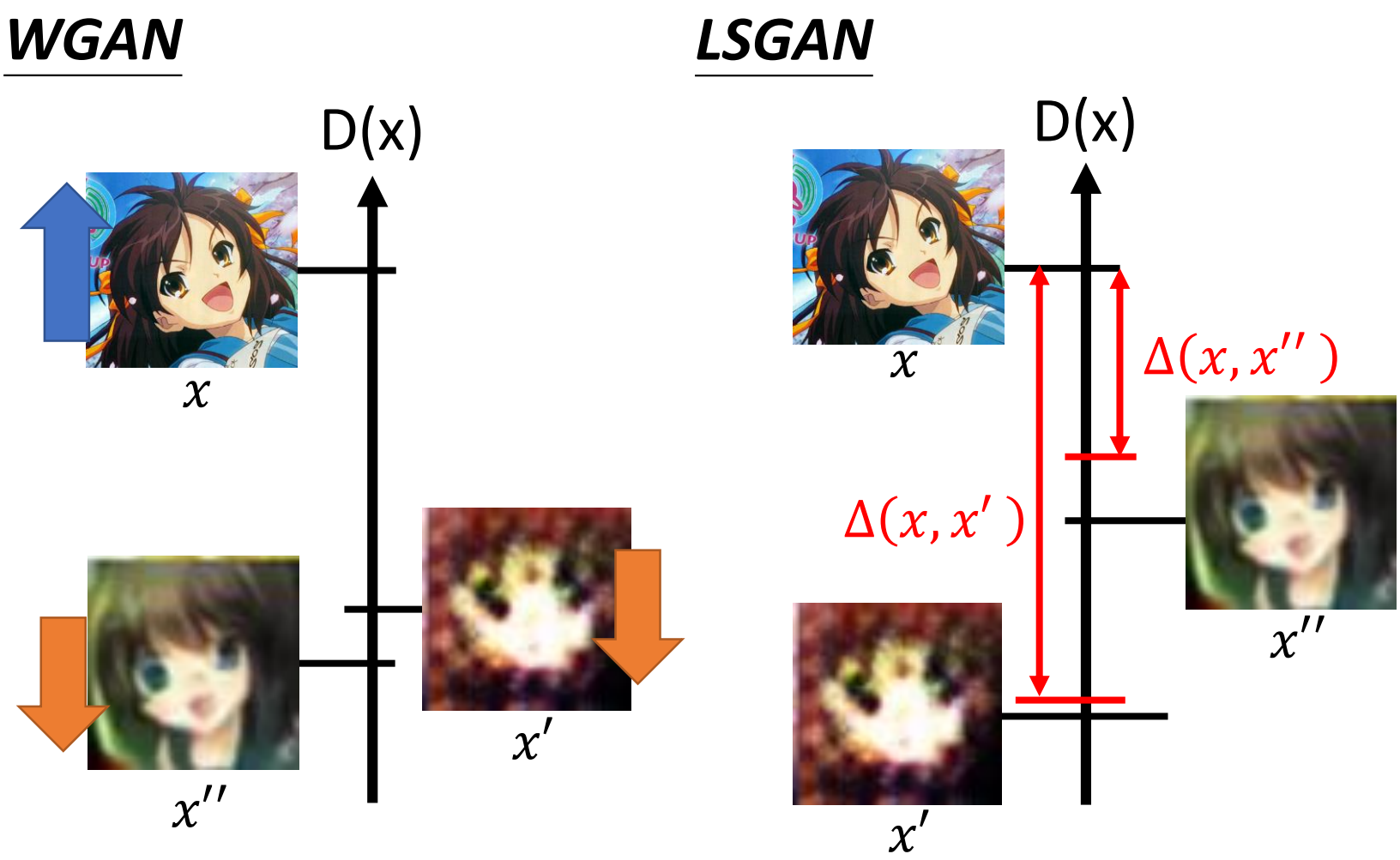
0 is for
the best.



Do not have to
be very negative

Hard to reconstruct, easy to destroy

Outlook:
Loss-sensitive GAN (LSGAN)



Reference

- Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio, Generative Adversarial Networks, NIPS, 2014
- Sebastian Nowozin, Botond Cseke, Ryota Tomioka, “f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization”, NIPS, 2016
- Martin Arjovsky, Soumith Chintala, Léon Bottou, Wasserstein GAN, arXiv, 2017
- Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, Aaron Courville, Improved Training of Wasserstein GANs, NIPS, 2017
- Junbo Zhao, Michael Mathieu, Yann LeCun, Energy-based Generative Adversarial Network, arXiv, 2016
- Mario Lucic, Karol Kurach, Marcin Michalski, Sylvain Gelly, Olivier Bousquet, “Are GANs Created Equal? A Large-Scale Study”, arXiv, 2017
- Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen Improved Techniques for Training GANs, NIPS, 2016

Reference

- Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, Sepp Hochreiter, GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium, NIPS, 2017
- Naveen Kodali, Jacob Abernethy, James Hays, Zsolt Kira, “On Convergence and Stability of GANs”, arXiv, 2017
- Xiang Wei, Boqing Gong, Zixia Liu, Wei Lu, Liqiang Wang, Improving the Improved Training of Wasserstein GANs: A Consistency Term and Its Dual Effect, ICLR, 2018
- Takeru Miyato, Toshiki Kataoka, Masanori Koyama, Yuichi Yoshida, Spectral Normalization for Generative Adversarial Networks, ICLR, 2018

Feature Extraction

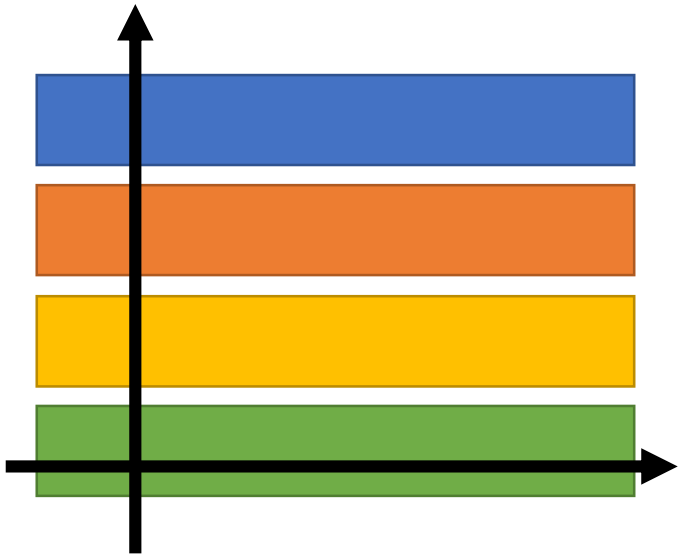
InfoGAN

(The colors represents the characteristics.)

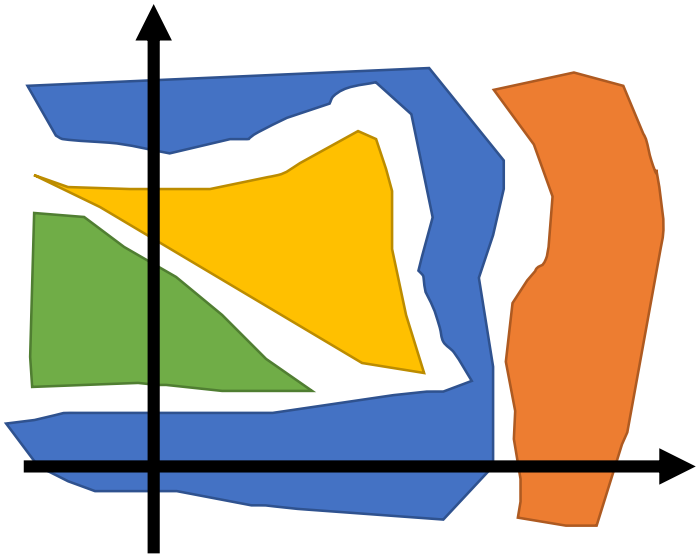
Regular GAN

Modifying a specific dimension, no clear meaning

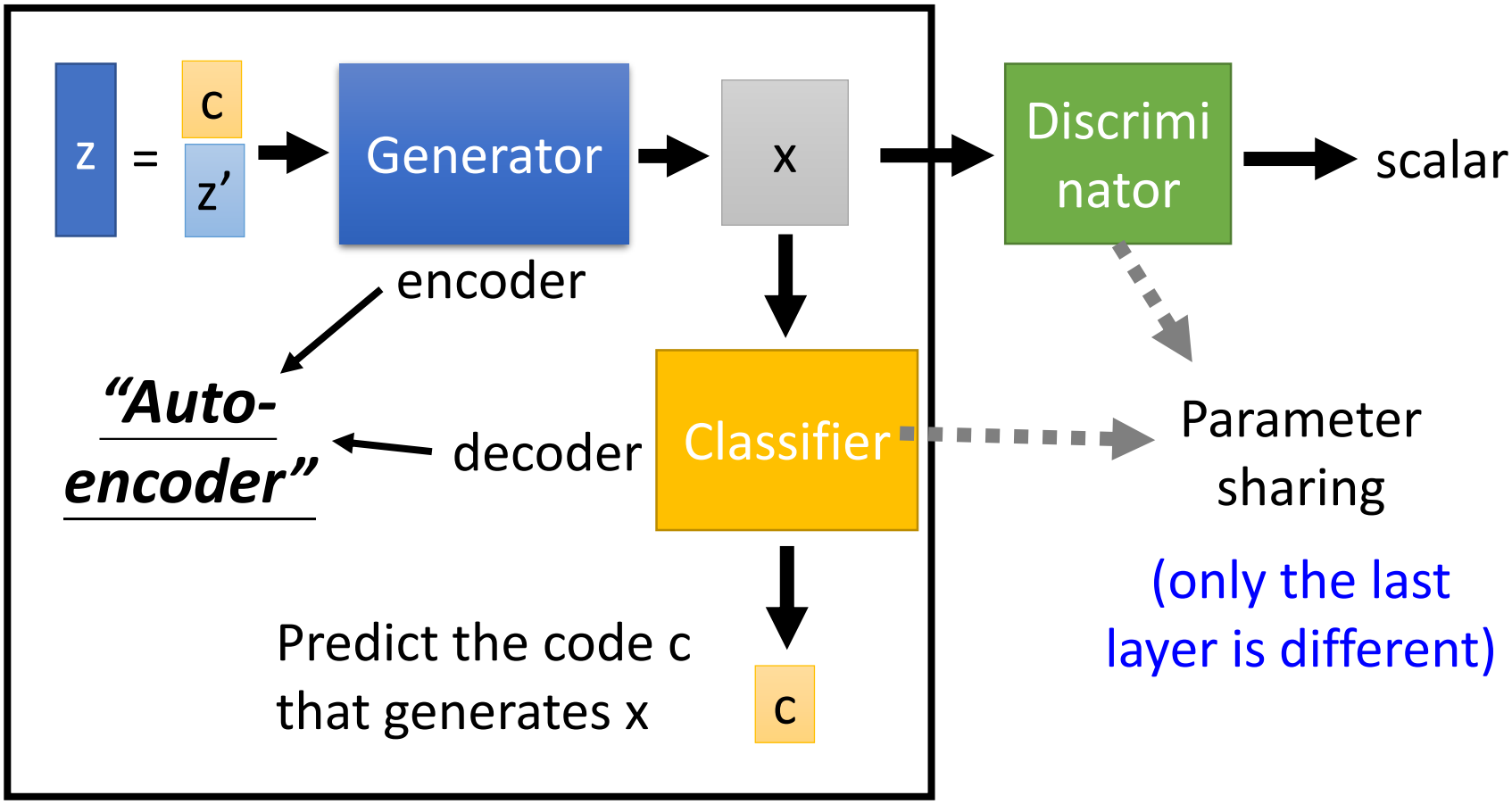
What we expect



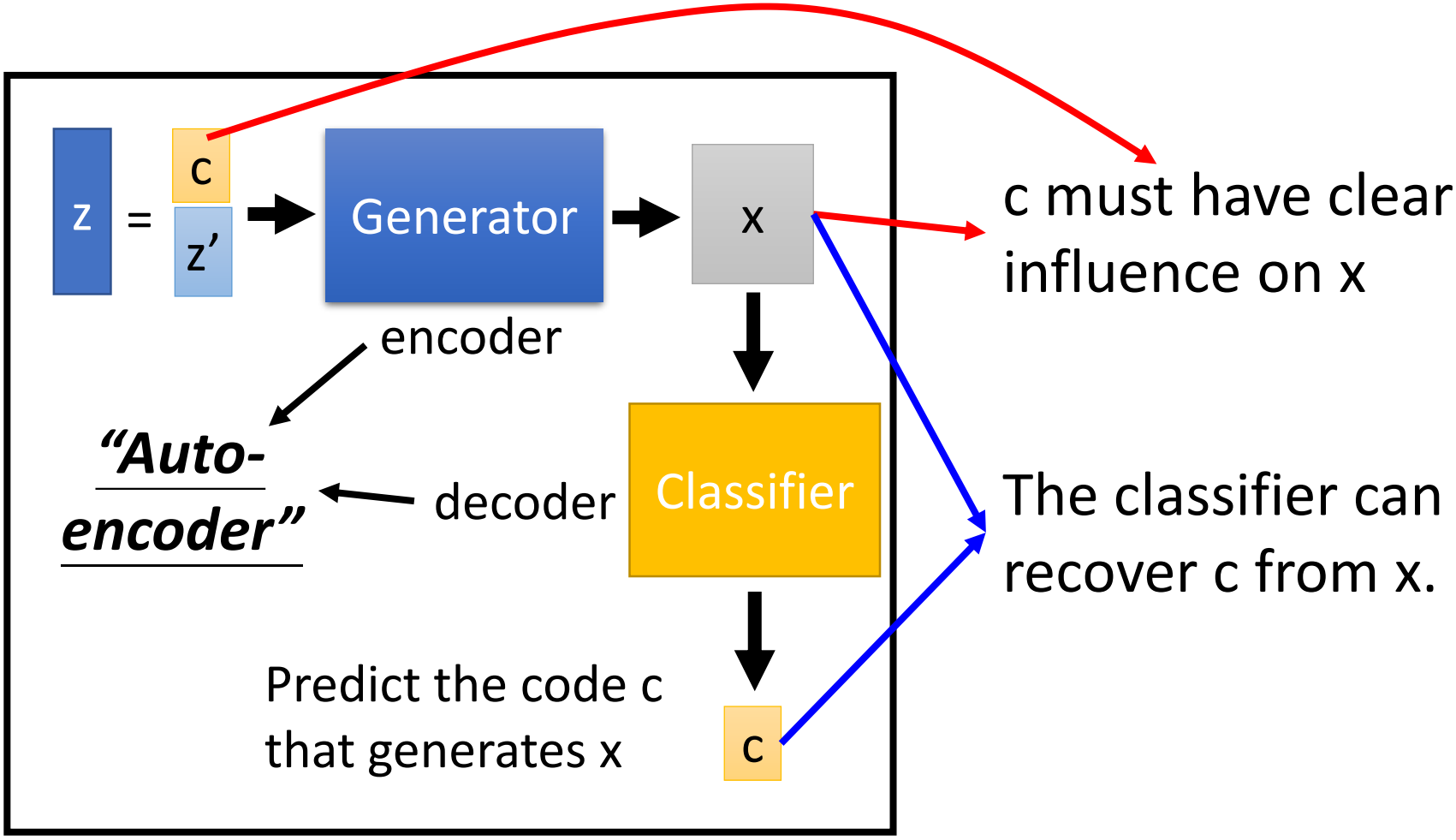
Actually …

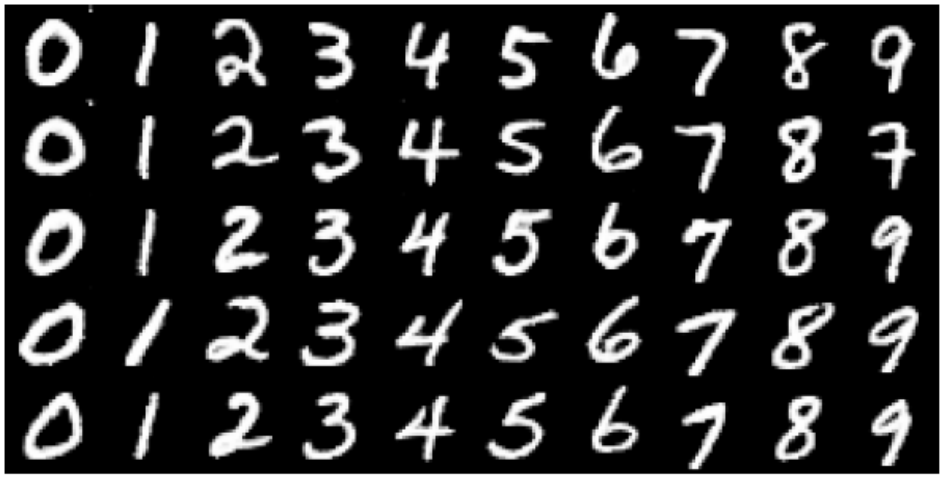


What is InfoGAN?

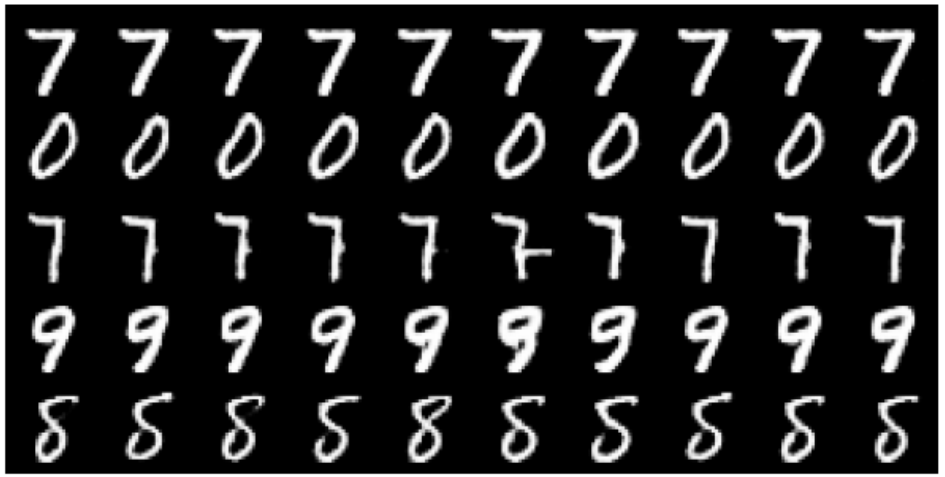


What is InfoGAN?

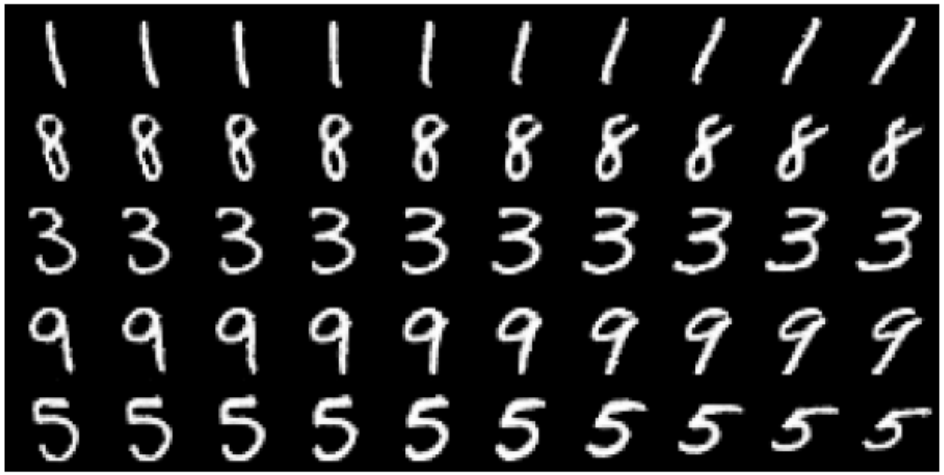




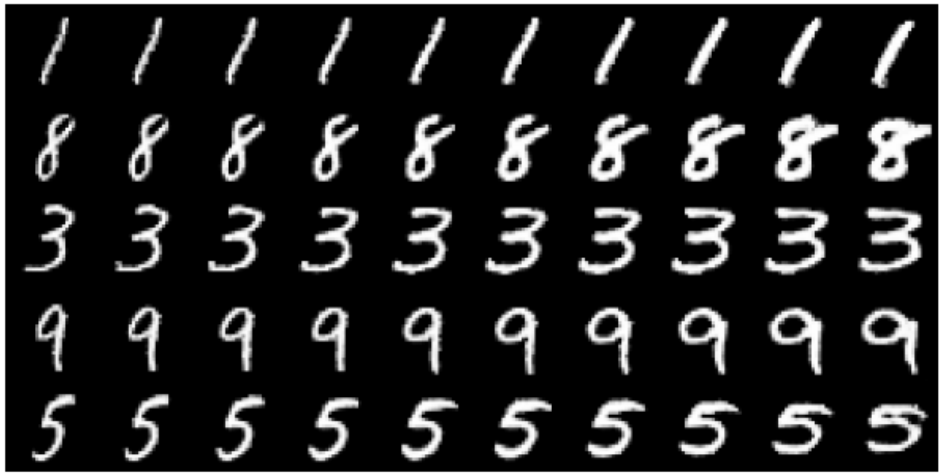
(a) Varying c_1 on InfoGAN (Digit type)



(b) Varying c_1 on regular GAN (No clear meaning)



(c) Varying c_2 from -2 to 2 on InfoGAN (Rotation)



(d) Varying c_3 from -2 to 2 on InfoGAN (Width)

https://arxiv.org/abs/1606.03657

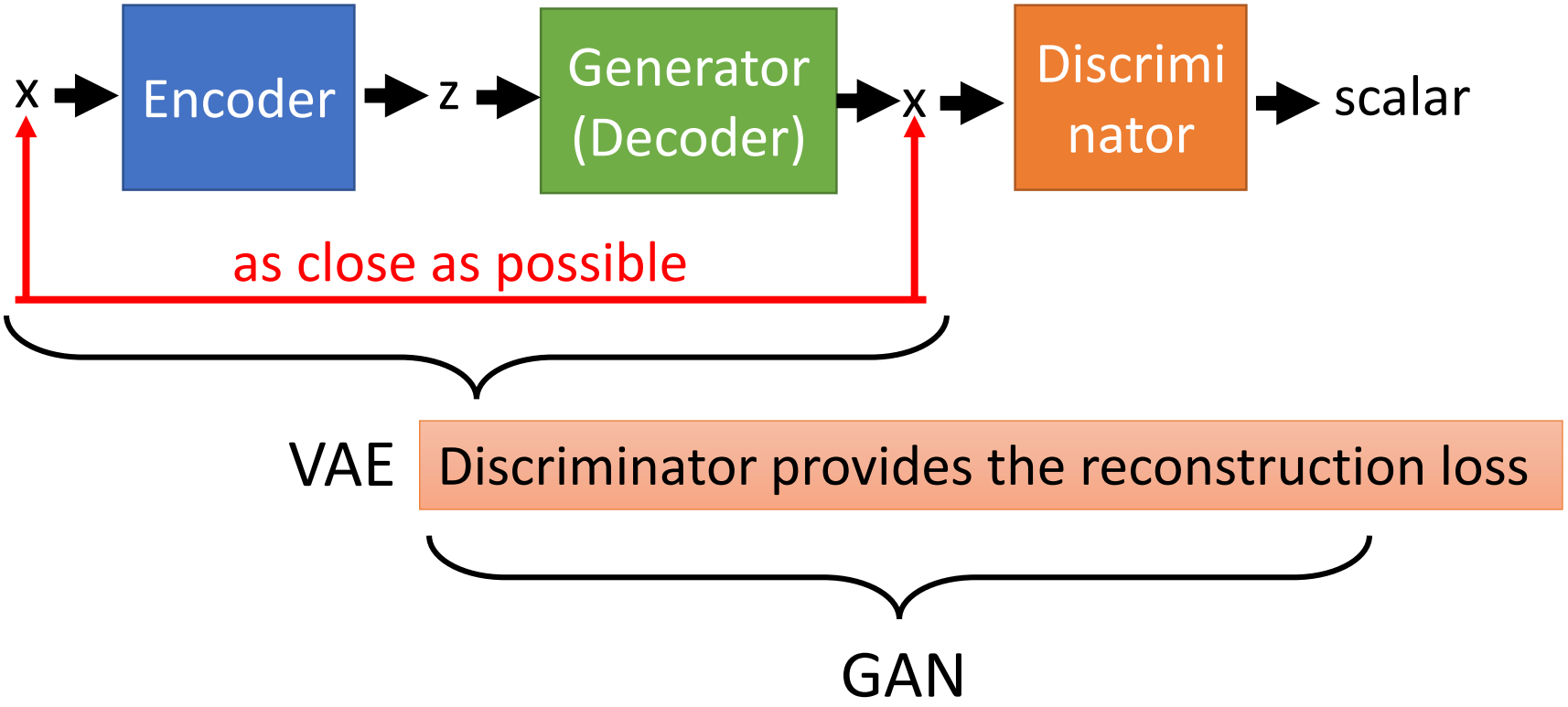
VAE-GAN

Anders Boesen, Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, Ole Winther, "Autoencoding beyond pixels using a learned similarity metric", ICML. 2016

- Minimize reconstruction error
- z close to normal

- Minimize reconstruction error
- Cheat discriminator

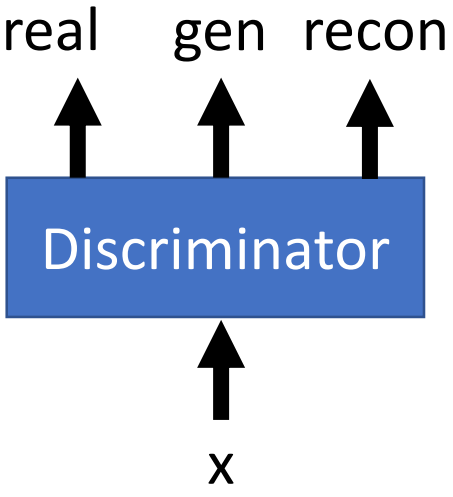
- Discriminate real, generated and reconstructed images



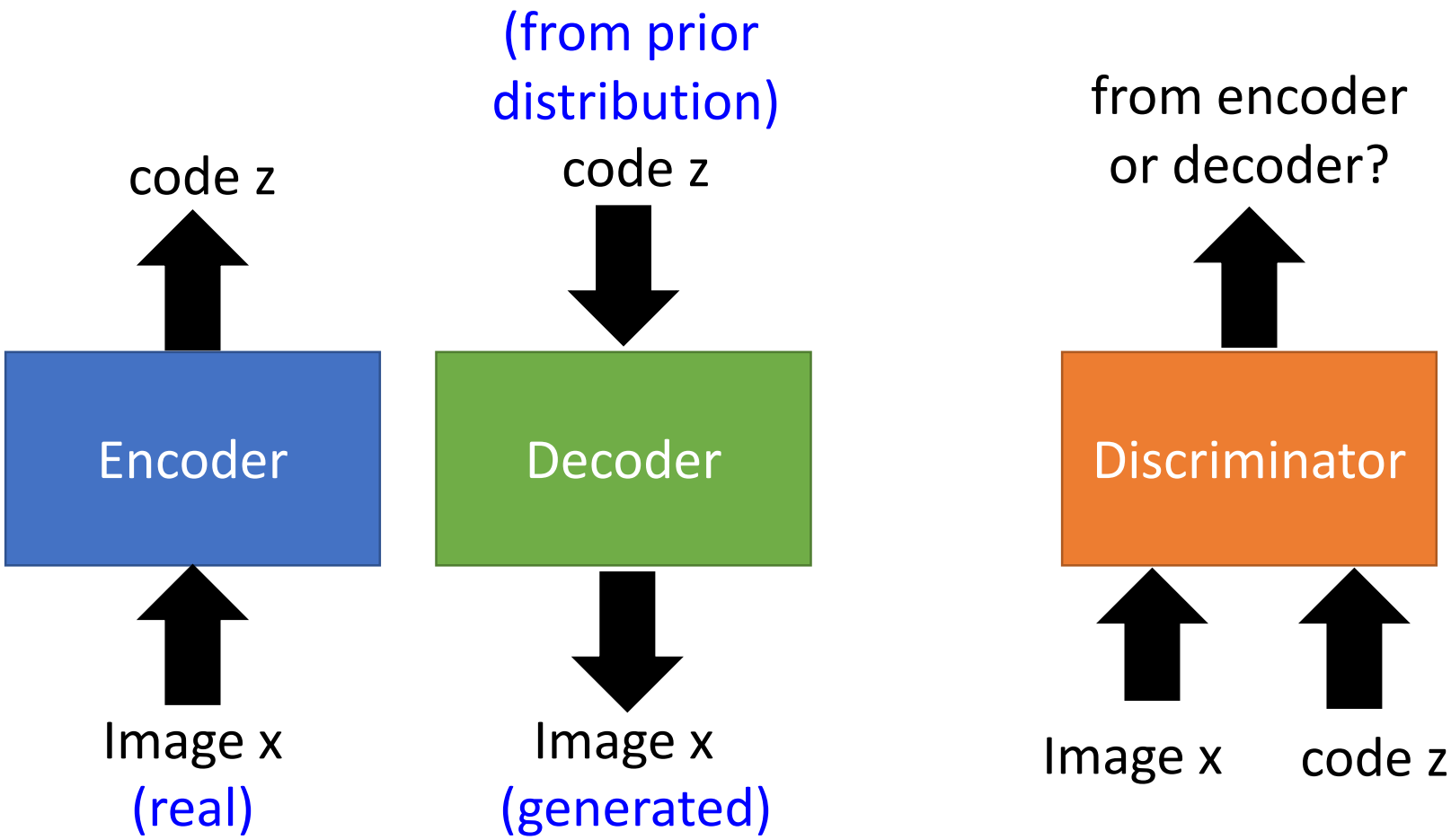
Algorithm

- Initialize En, De, Dis
- In each iteration:
  - Sample M images $x^1, x^2, \cdots, x^M$ from database
  - Generate M codes $\tilde{z}^1, \tilde{z}^2, \cdots, \tilde{z}^M$ from encoder
    - $\tilde{z}^i = En(x^i)$
  - Generate M images $\tilde{x}^1, \tilde{x}^2, \cdots, \tilde{x}^M$ from decoder
    - $\tilde{x}^i = De(\tilde{z}^i)$
  - Sample M codes $z^1, z^2, \cdots, z^M$ from prior P(z)
  - Generate M images $\hat{x}^1, \hat{x}^2, \cdots, \hat{x}^M$ from decoder
    - $\hat{x}^i = De(z^i)$
  - Update En to decrease $\|\tilde{x}^i - x^i\|$, decrease KL(P($\tilde{z}^i$|x^i)||P(z))
  - Update De to decrease $\|\tilde{x}^i - x^i\|$, increase $Dis(\tilde{x}^i)$ and $Dis(\hat{x}^i)$
  - Update Dis to increase $Dis(x^i)$, decrease $Dis(\tilde{x}^i)$ and $Dis(\hat{x}^i)$

Another kind of discriminator:

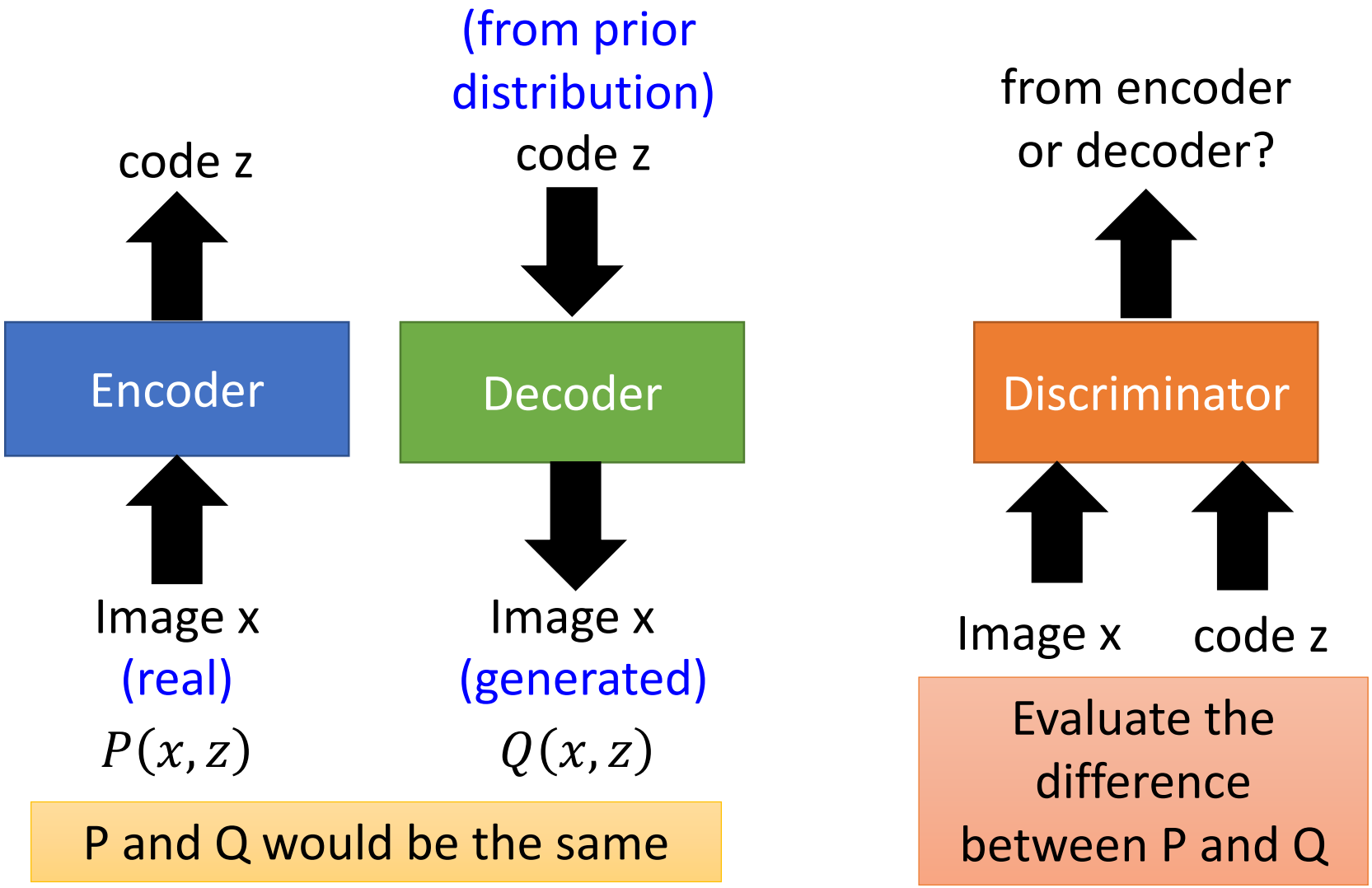


BiGAN


code z
Encoder
Image x
(real)
(from prior
distribution)
code z
Decoder
Image x
(generated)
from encoder
or decoder?
Discriminator
Image x
code z

Algorithm

- Initialize encoder En, decoder De, discriminator Dis
- In each iteration:
  - Sample M images $x^1, x^2, \cdots, x^M$ from database
  - Generate M codes $\tilde{z}^1, \tilde{z}^2, \cdots, \tilde{z}^M$ from encoder
    - $\tilde{z}^i = En(x^i)$
  - Sample M codes $z^1, z^2, \cdots, z^M$ from prior P(z)
  - Generate M codes $\tilde{x}^1, \tilde{x}^2, \cdots, \tilde{x}^M$ from decoder
    - $\tilde{x}^i = De(z^i)$
  - Update Dis to increase $Dis(x^i, \tilde{z}^i)$, decrease $Dis(\tilde{x}^i, z^i)$
  - Update En and De to decrease $Dis(x^i, \tilde{z}^i)$, increase $Dis(\tilde{x}^i, z^i)$



Optimal encoder and decoder:

En(x') = z' ⇒ De(z') = x' For all x'

De(z'') = x'' ⇒ En(x'') = z'' For all z''

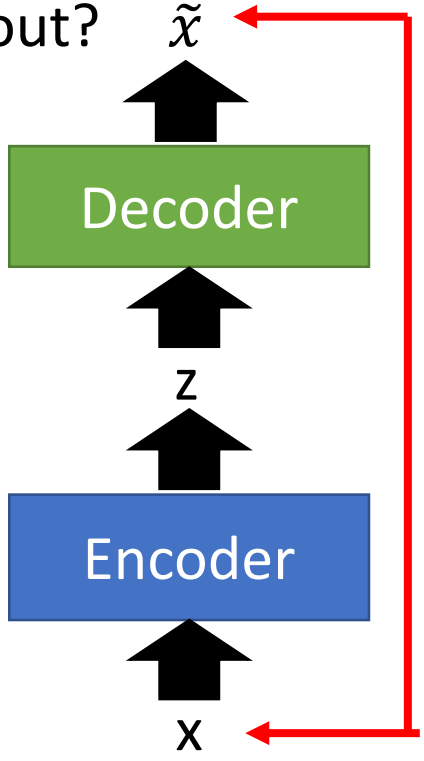
BiGAN

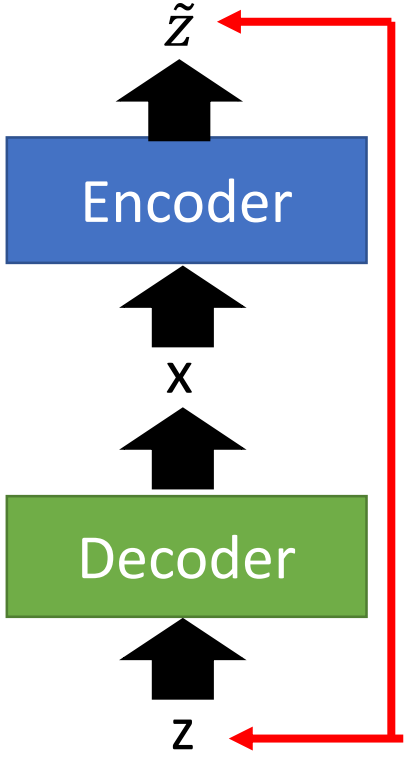
Optimal encoder and decoder:

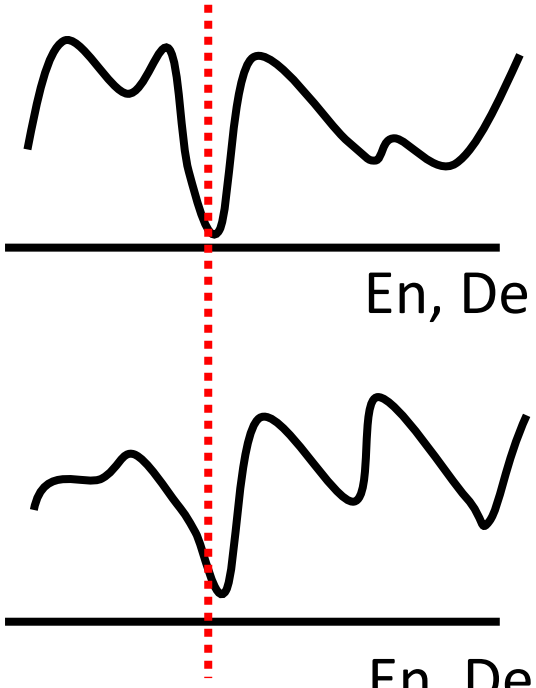
En(x') = z' ⇒ De(z') = x' For all x'

De(z'') = x'' ⇒ En(x'') = z'' For all z''

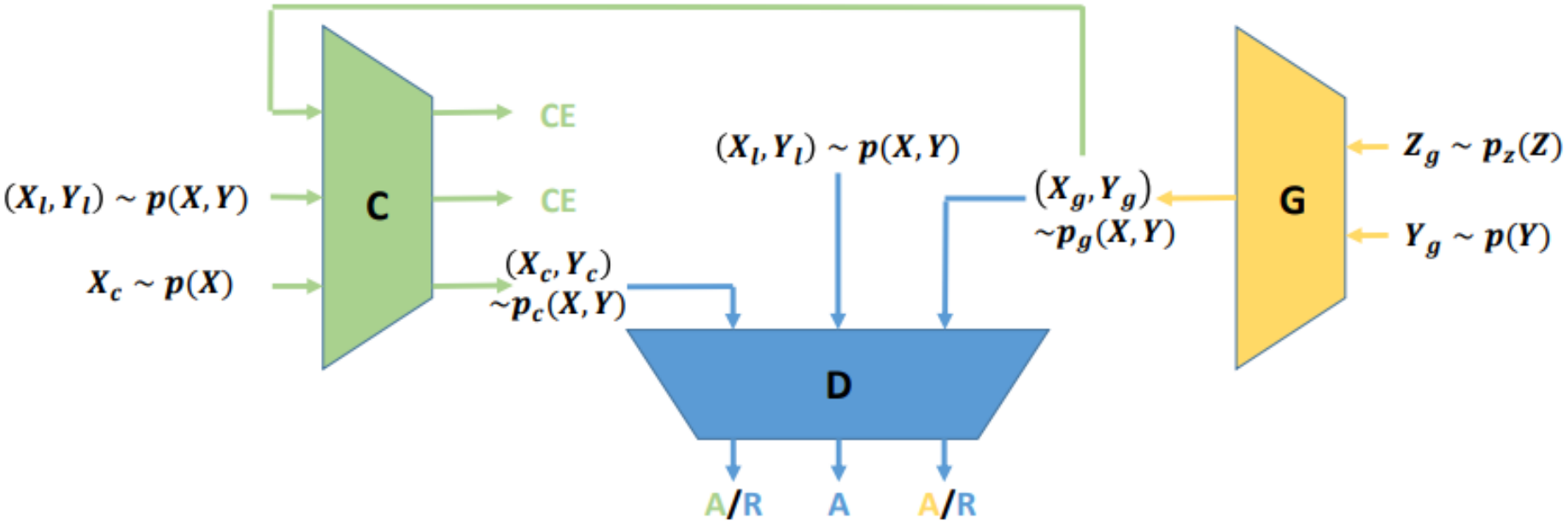
How about?







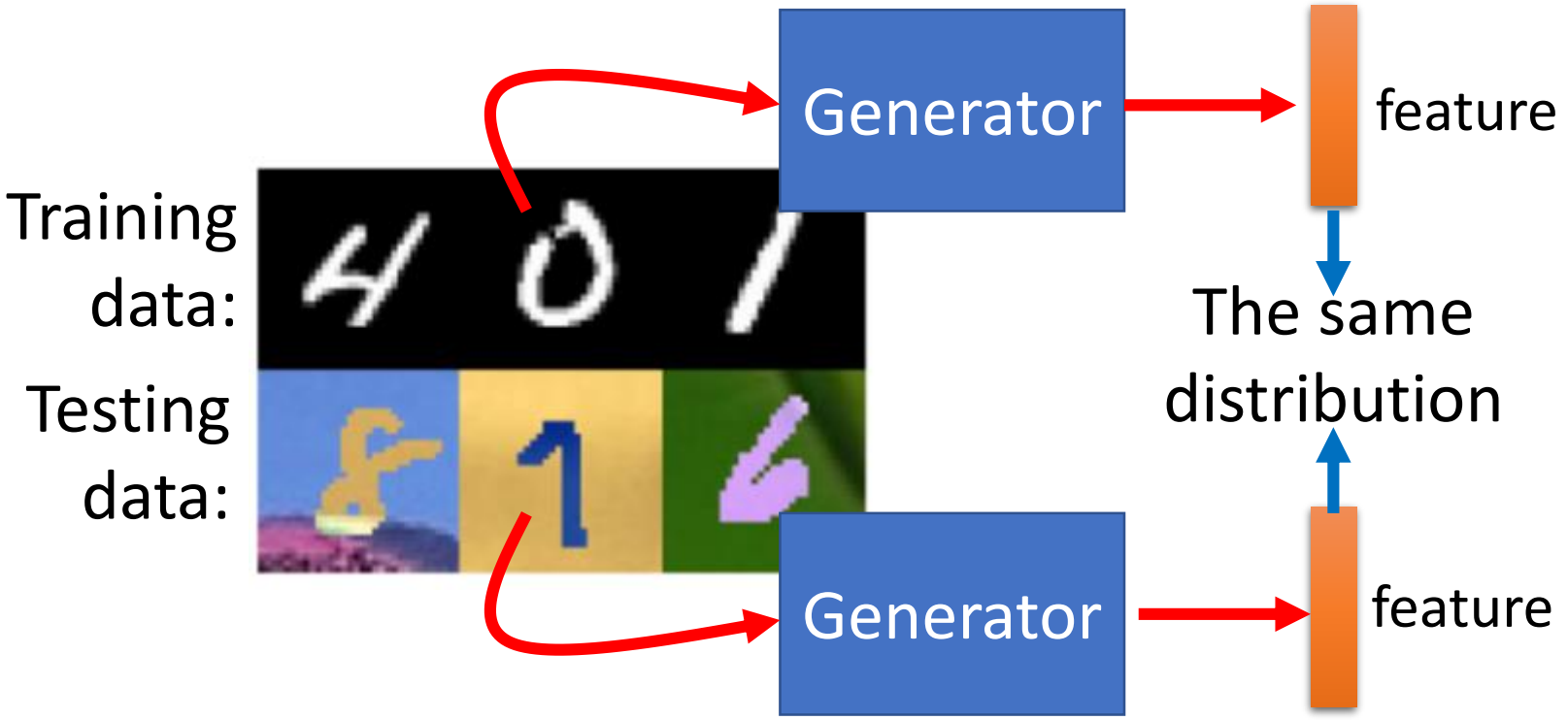
Triple GAN



Chongxuan Li, Kun Xu, Jun Zhu, Bo Zhang, “Triple Generative Adversarial Nets”, arXiv 2017

Domain-adversarial training

- Training and testing data are in different domains

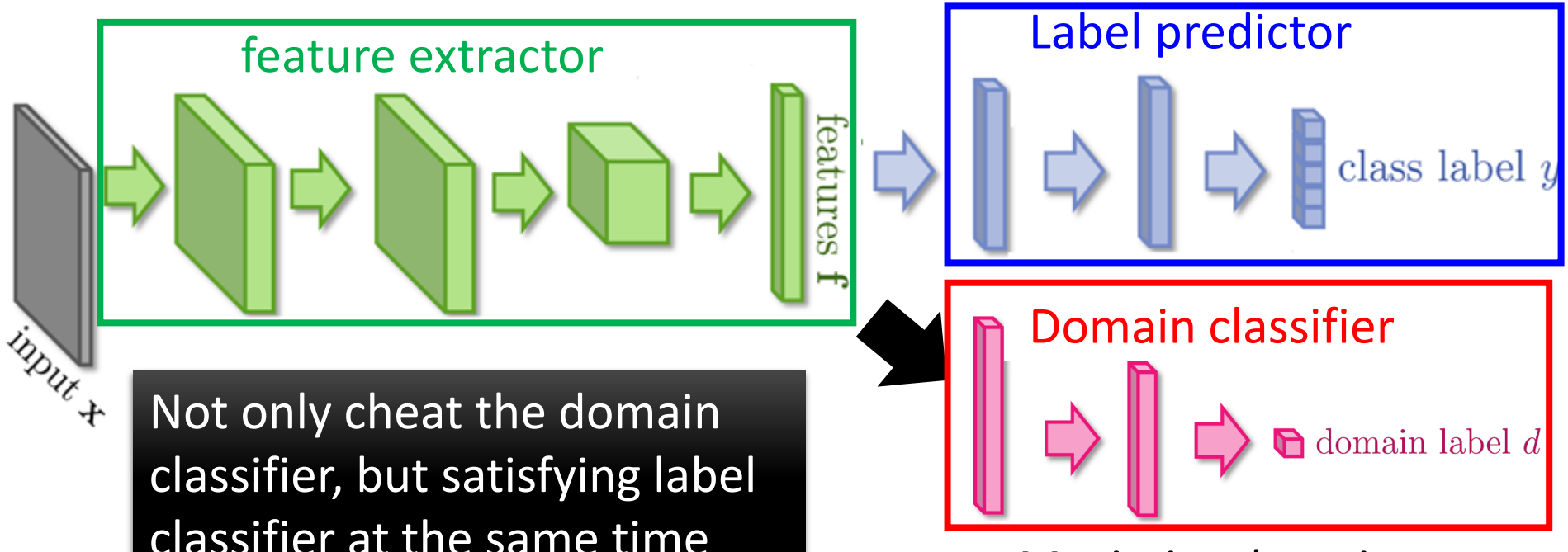


Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, Domain-Adversarial Training of Neural Networks, JMLR, 2016

Domain-adversarial training

Maximize label classification accuracy + minimize domain classification accuracy

Maximize label classification accuracy

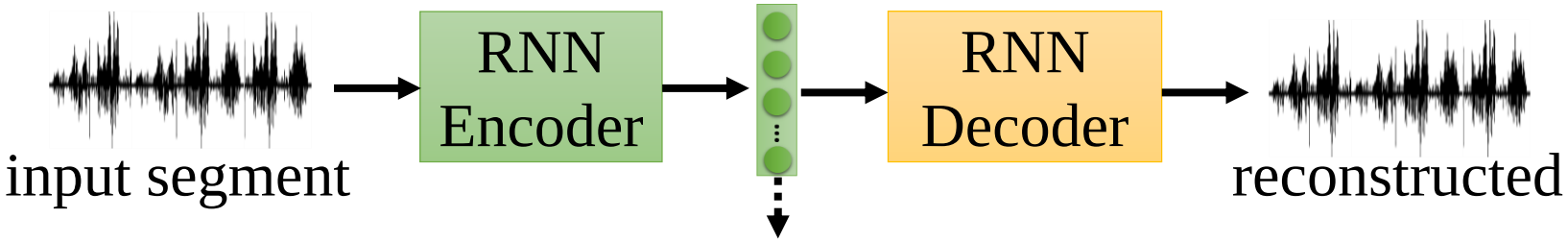


Not only cheat the domain classifier, but satisfying label classifier at the same time

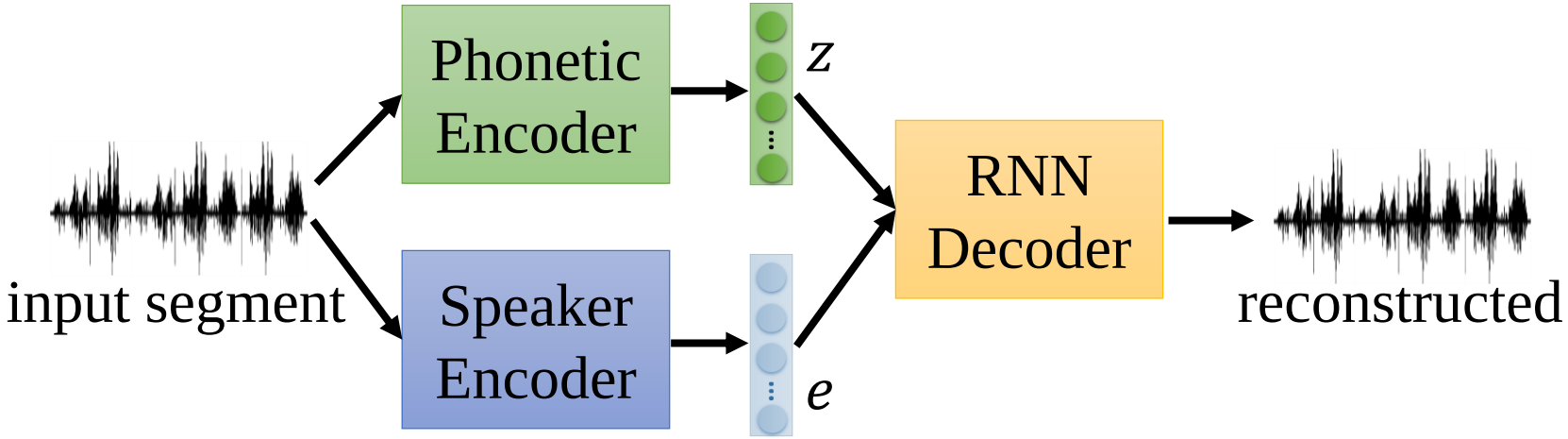
Maximize domain classification accuracy

This is a big network, but different parts have different goals.

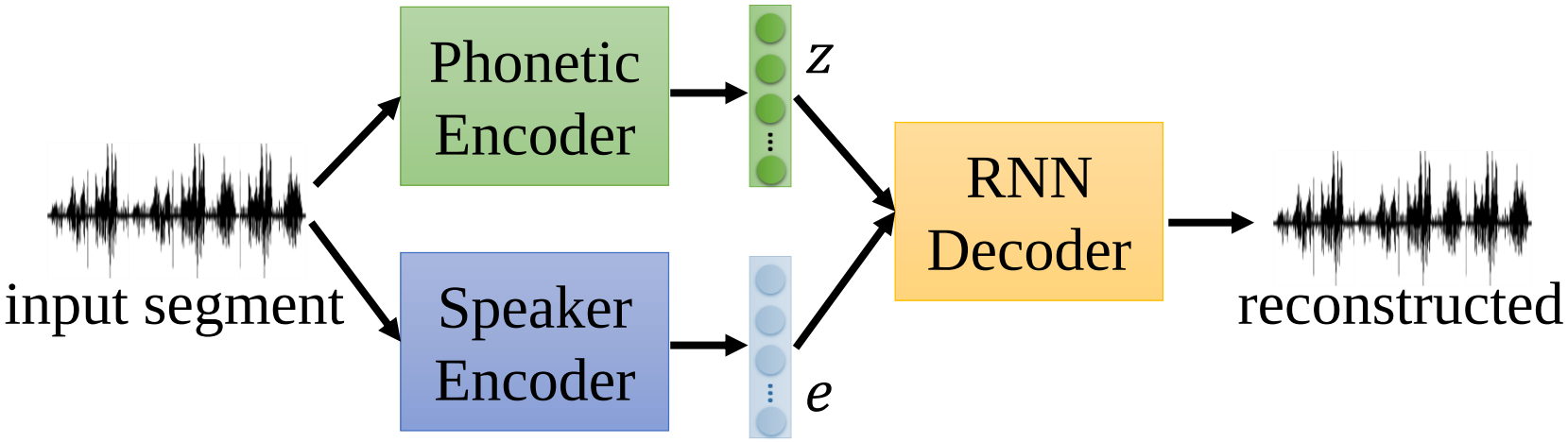
Original Seq2seq Auto-encoder

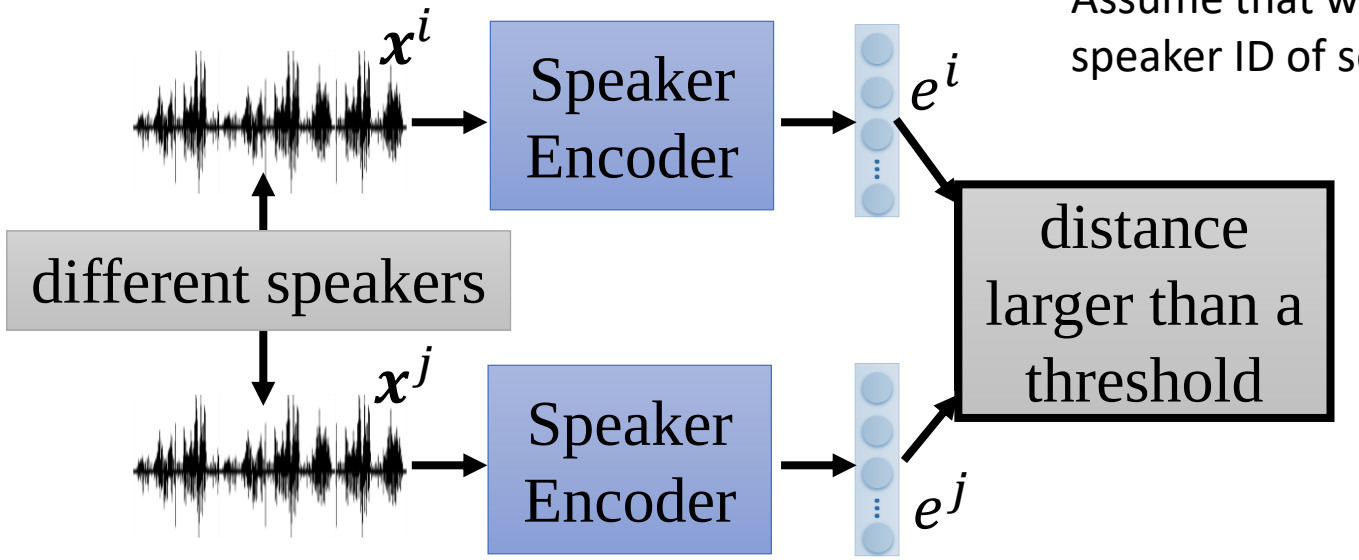


Feature Disentangle



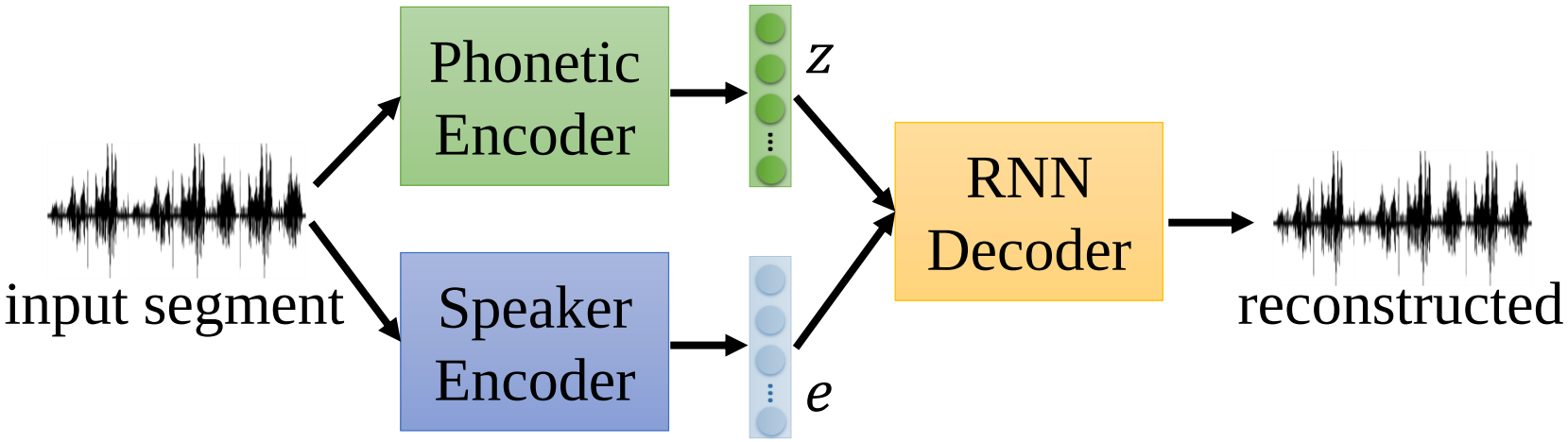
Feature Disentangle



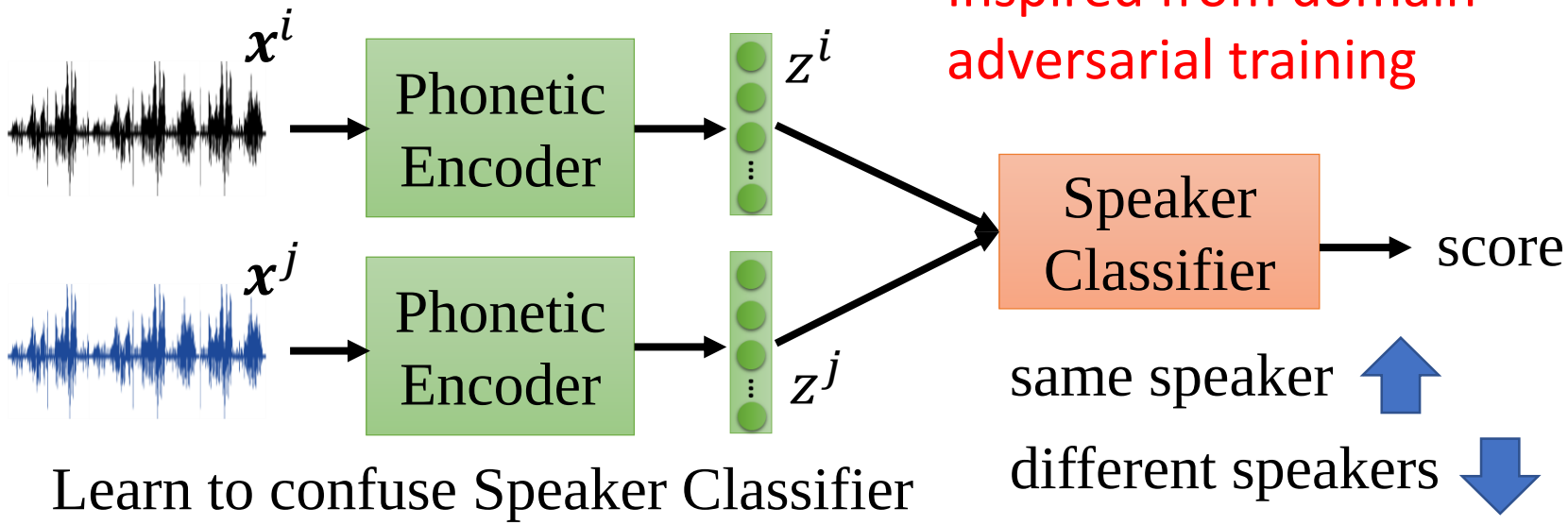


Assume that we know speaker ID of segments.

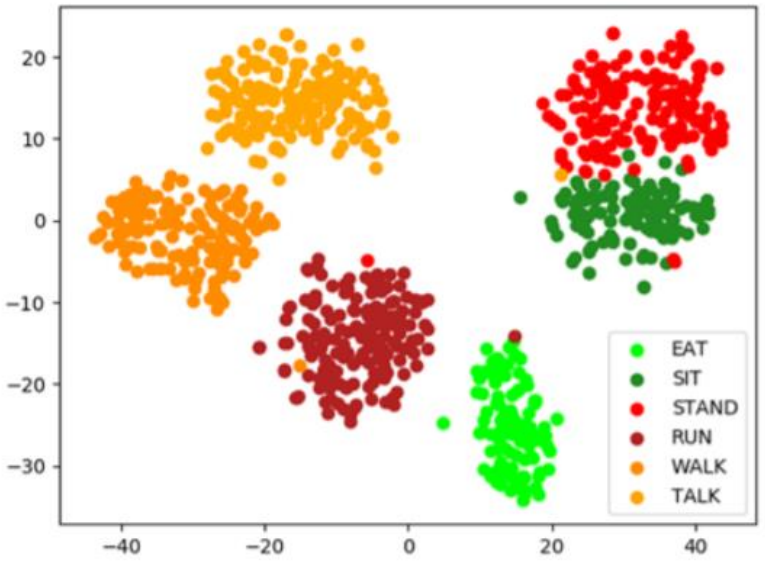
Feature Disentangle



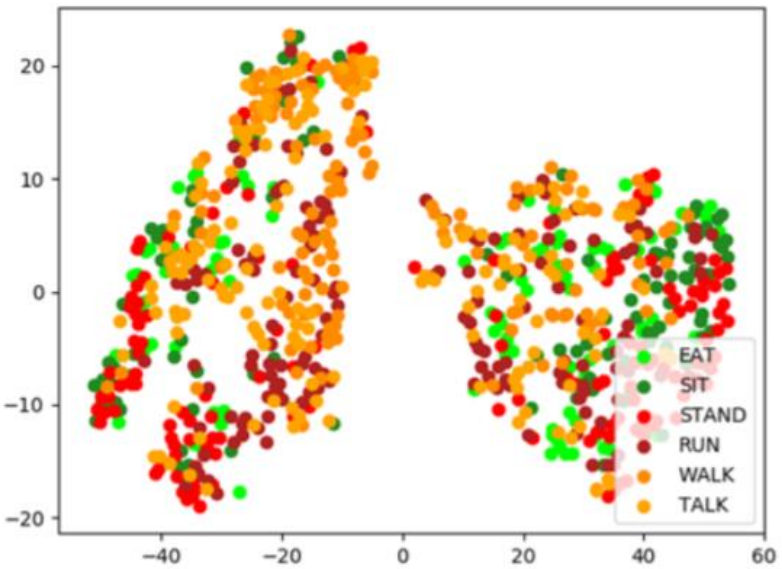
Inspired from domain adversarial training



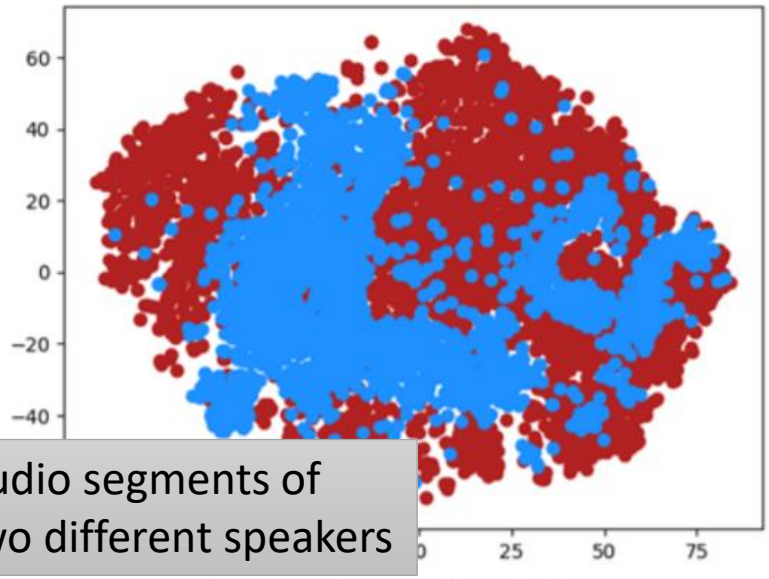
Learn to confuse Speaker Classifier



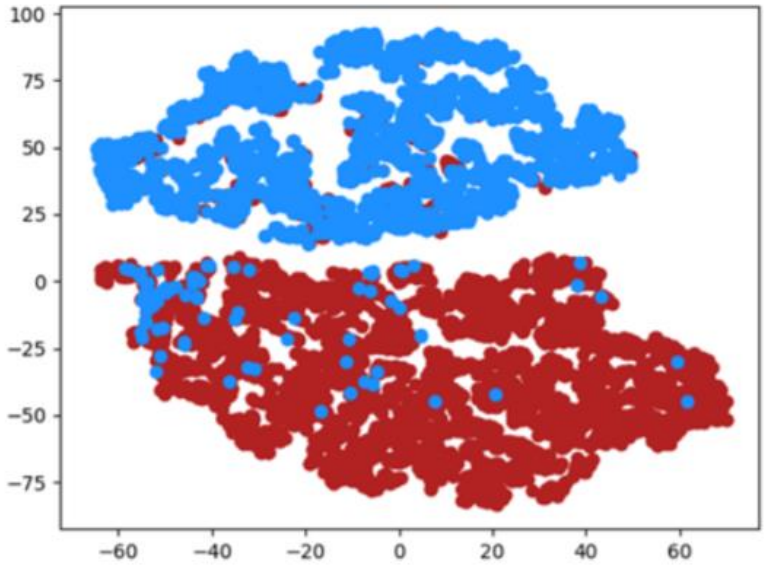
phonetic embedding z



speaker embedding e



phonetic embedding z



speaker embedding e

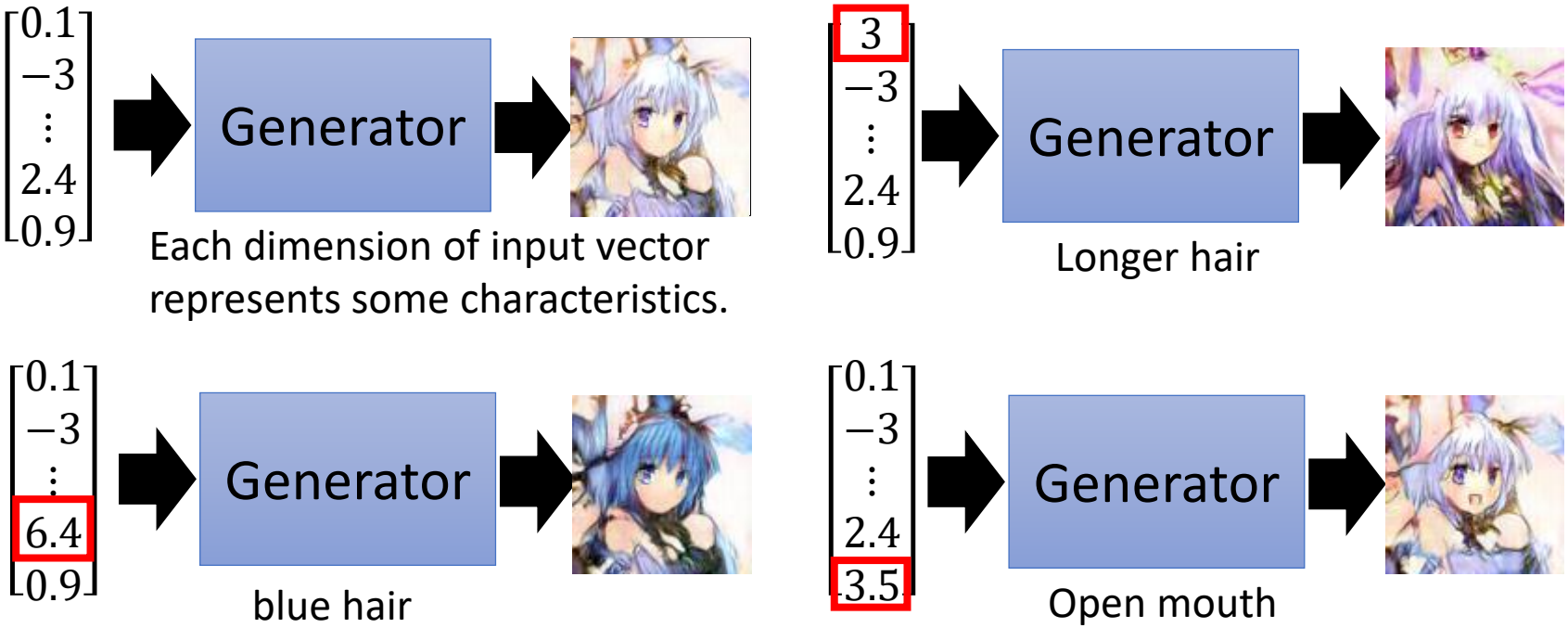
Audio segments of two different speakers

Acknowledgement

- 感謝 許傑盛 同學發現投影片 上的打字錯誤

Intelligent Photo Editing

Modifying Input Code



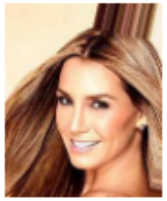
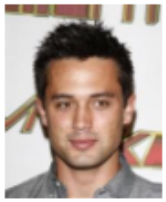
- The input code determines the generator output.
- Understand the meaning of each dimension to control the output.

Connecting Code and Attribute

(c) Hair style

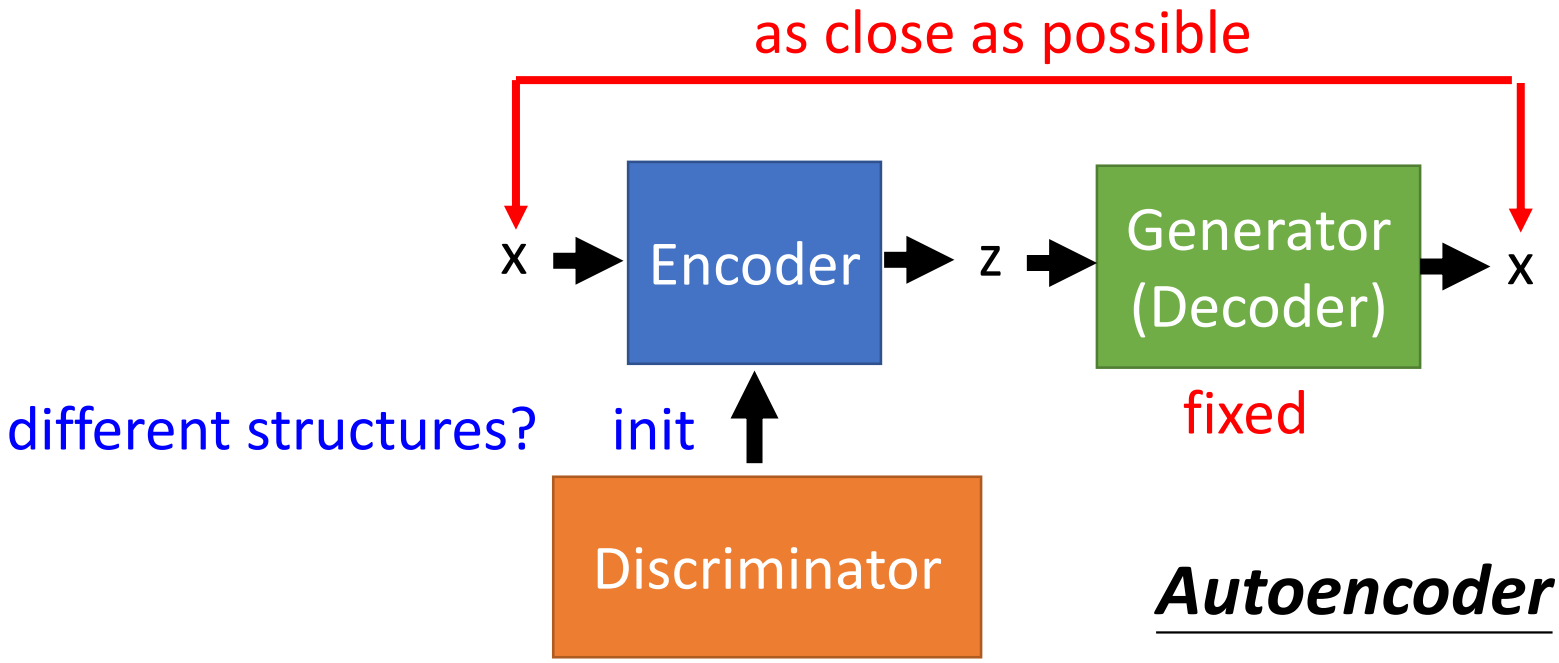
(d) Emotion

CelebA

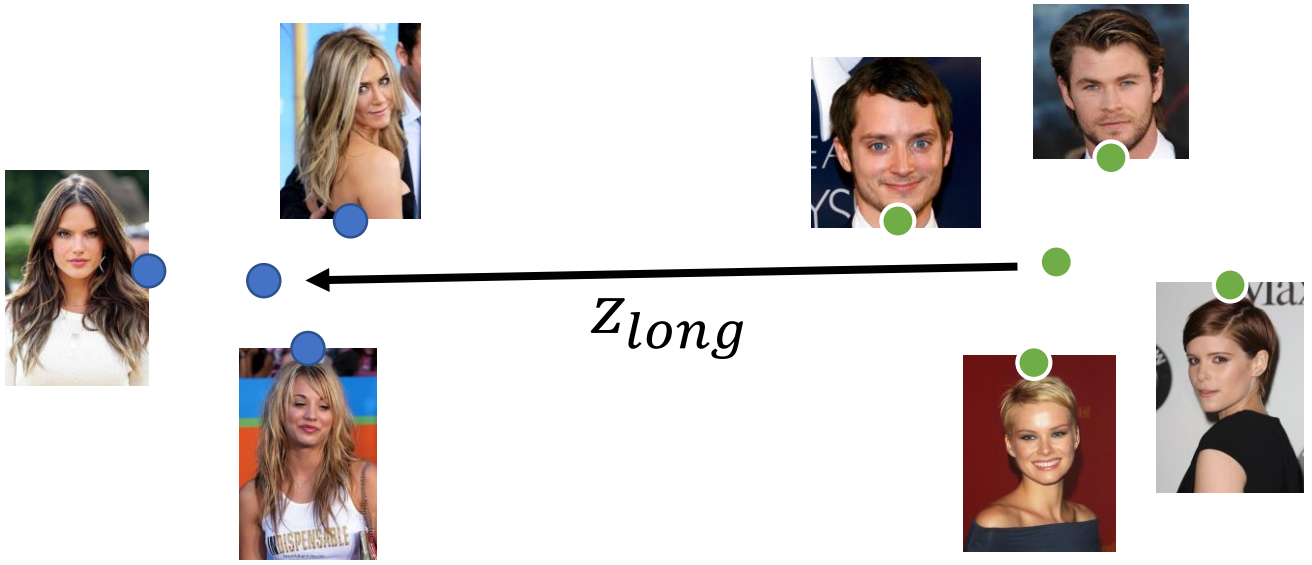
Image	Attributes
	Arched eyebrows, attractive, brown hair, heavy makeup, high cheekbones, mouth slightly open, no beard, pointy nose, smiling, straight hair, wearing earrings, wearing lipstick, young.
	5 o'clock shadows, attractive, bags under eyes, big lips, big nose, black hair, bushy eyebrows, male, no beard, pointy nose, straight hair, young.

GAN+Autoencoder

- We have a generator (input z, output x)
- However, given x, how can we find z?
  - Learn an encoder (input x, output z)



Attribute Representation



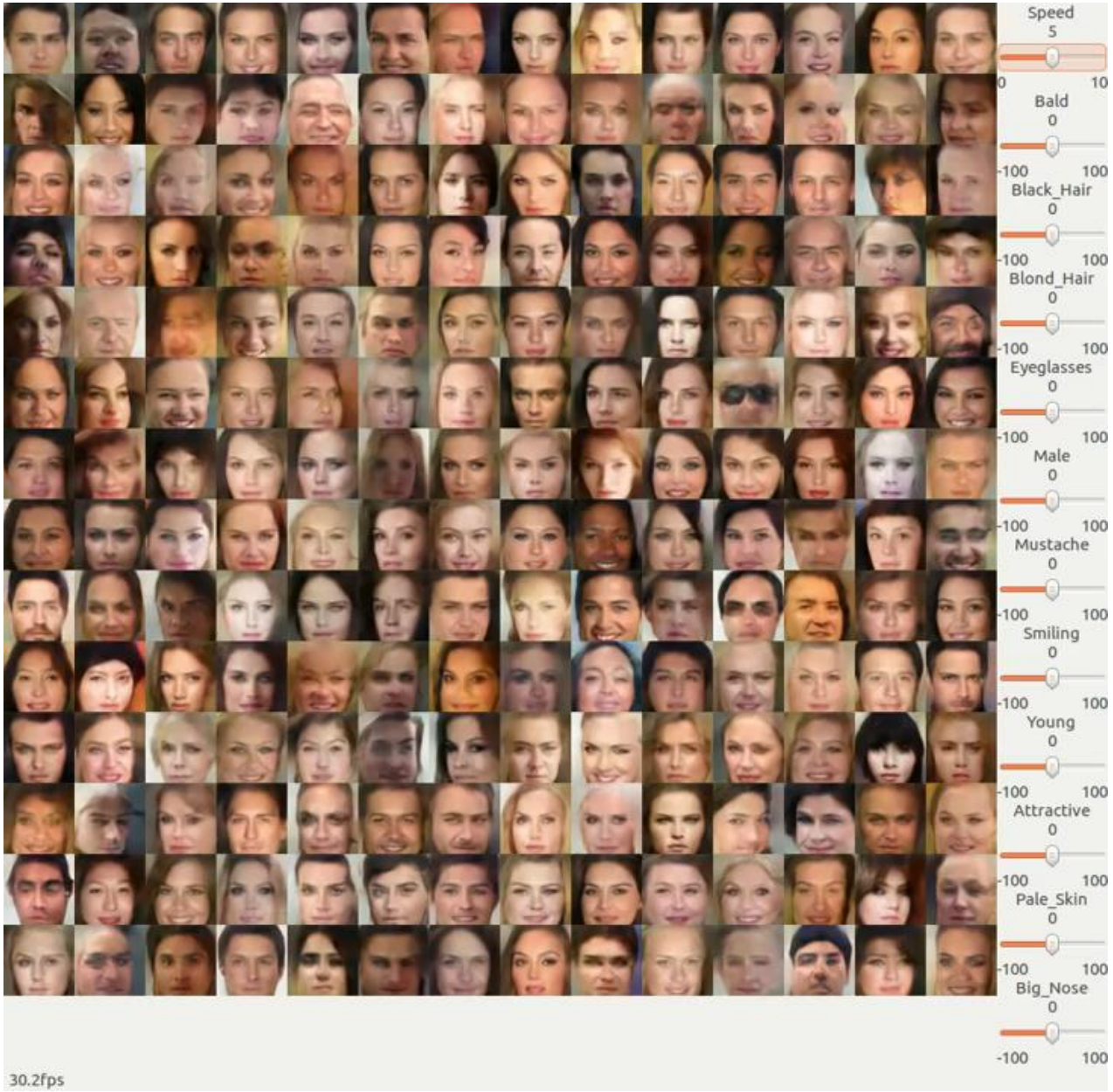
$$z_{long} = \frac{1}{N_1}\sum_{x \in long} En(x) - \frac{1}{N_2}\sum_{x' \notin long} En(x')$$

Short Hair

$$x \Rightarrow En(x) + z_{long} = z' \Rightarrow Gen(z')$$

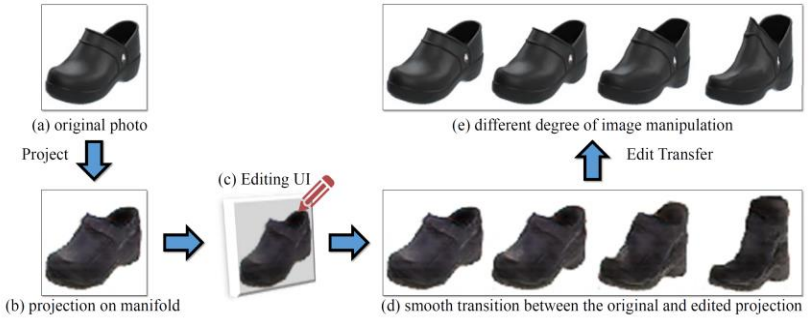
Long Hair

Photo Editing



https://www.youtube.com/watch?v=kPEIJJsQr7U

Basic Idea

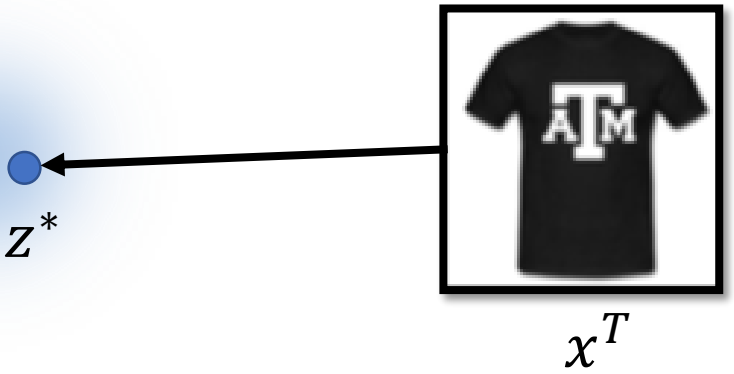


space of code z

Fulfill the constraint

Why move on the code space?

Back to z



- ***Method 1***

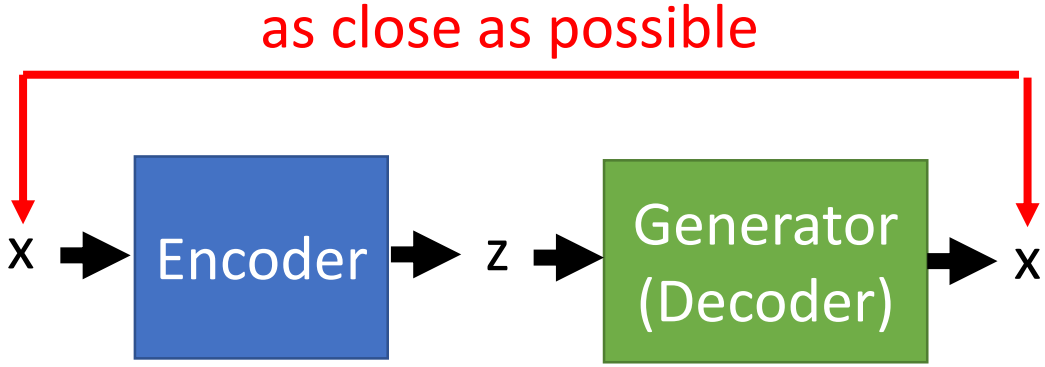
$$z^* = arg\min_z L(G(z), x^T)$$

Gradient Descent

Difference between G(z) and x^T

- Pixel-wise
- By another network

- ***Method 2***



- ***Method 3***

Using the results from ***method 2*** as the initialization of ***method 1***

Editing Photos

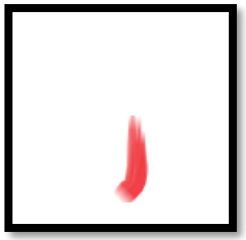
- z_0 is the code of the input image

image

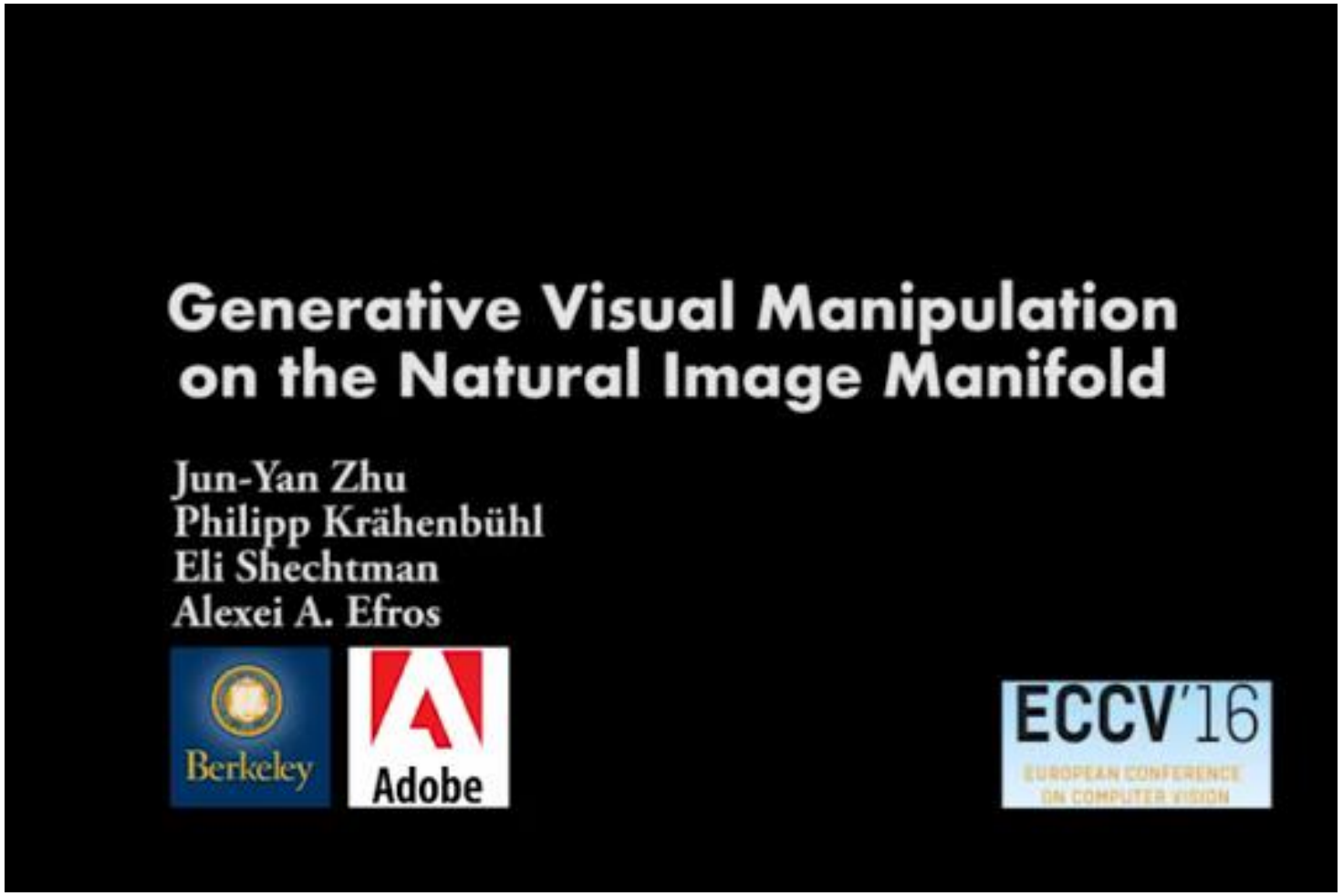
Using discriminator to check the image is realistic or not

$$z^* = arg\min_z U\big(G(z)\big) + \lambda_1 \|z - z_0\|^2 - \lambda_2 D\big(G(z)\big)$$

Not too far away from the original image

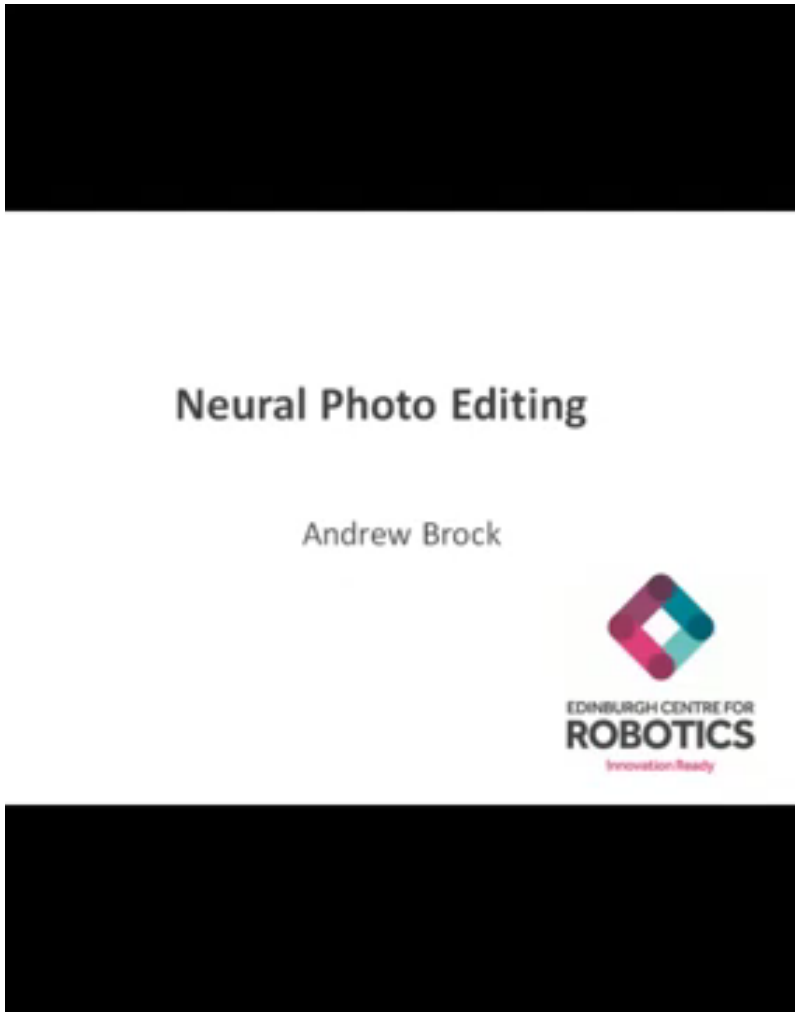


Does it fulfill the constraint of editing?



https://www.youtube.com/watch?v=9c4z6YsBGQ0

Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman and Alexei A. Efros. "Generative Visual Manipulation on the Natural Image Manifold", ECCV, 2016.



Andrew Brock, Theodore Lim, J.M. Ritchie, Nick Weston, **Neural Photo Editing with Introspective Adversarial Networks,** arXiv preprint, 2017

Image super resolution

- Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, Wenzhe Shi, “Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network”, CVPR, 2016

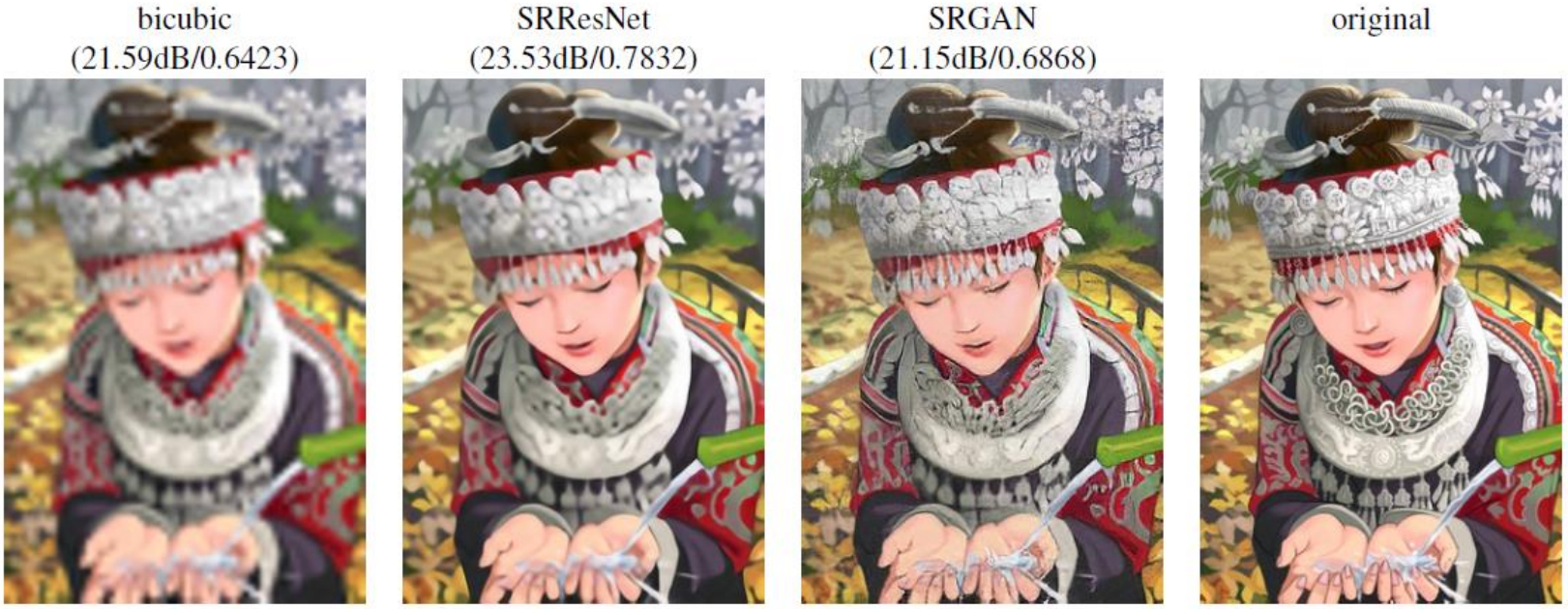
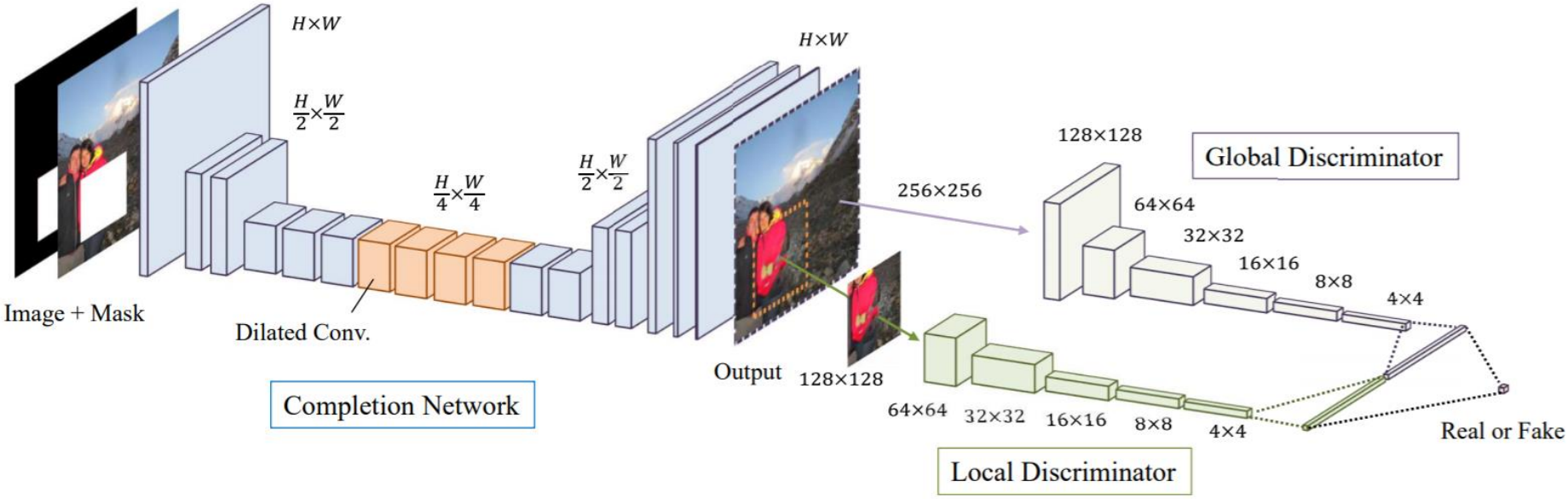


Figure 2: From left to right: bicubic interpolation, deep residual network optimized for MSE, deep residual generative adversarial network optimized for a loss more sensitive to human perception, original HR image. Corresponding PSNR and SSIM are shown in brackets. [4× upscaling]

Image Completion

http://hi.cs.waseda.ac.jp/~iizuka/projects/completion/en/



Demo

https://www.youtube.com/watch?v=5Ua4NUKowPU

Improving Sequence Generation by GAN

李宏毅

Hung-yi Lee

Outline

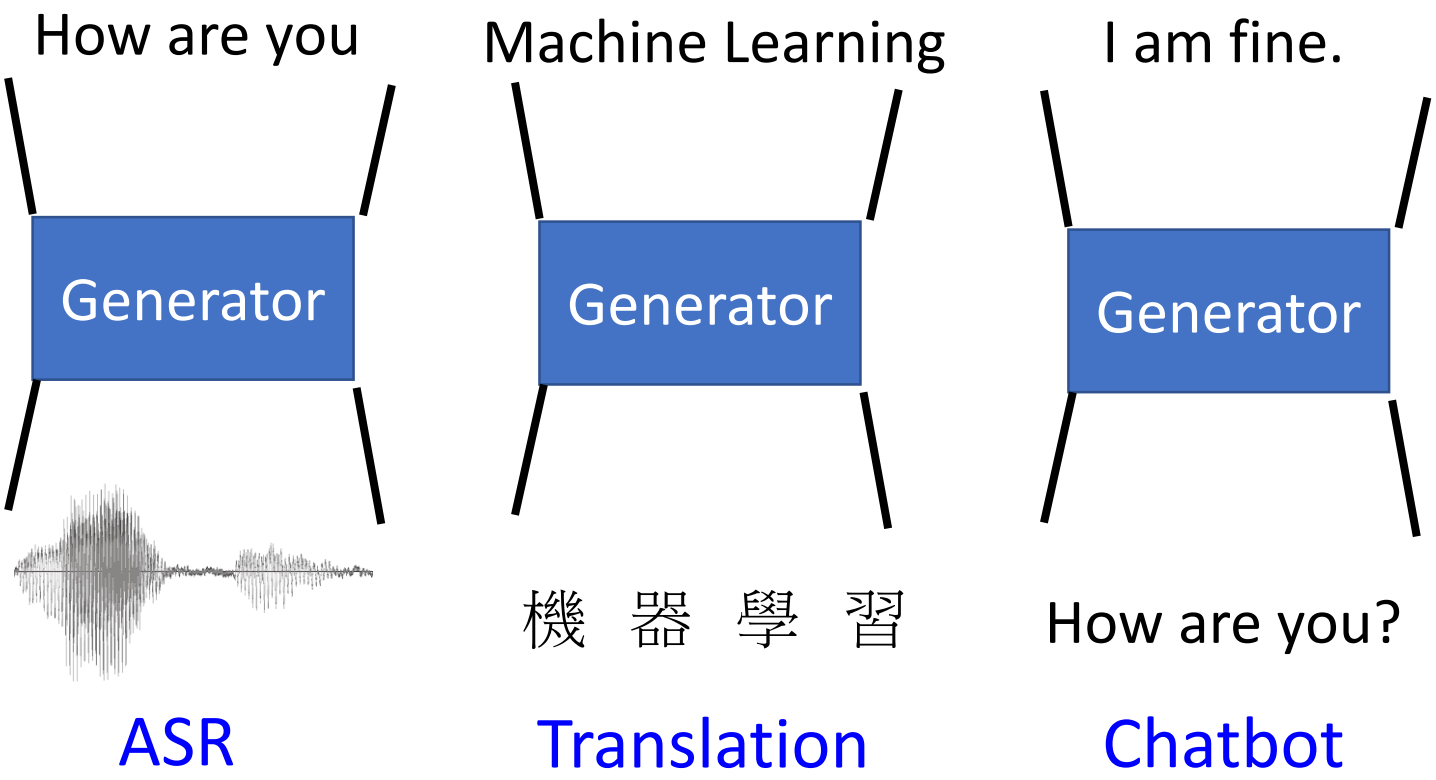
Conditional Sequence Generation

- RL (human feedback)
- GAN (discriminator feedback)

Unsupervised Conditional Sequence Generation

- Text Style Transfer
- Unsupervised Abstractive Summarization
- Unsupervised Translation

Conditional Sequence Generation



The generator is a typical seq2seq model.

With GAN, you can train seq2seq model in another way.

Review: Sequence-to-sequence

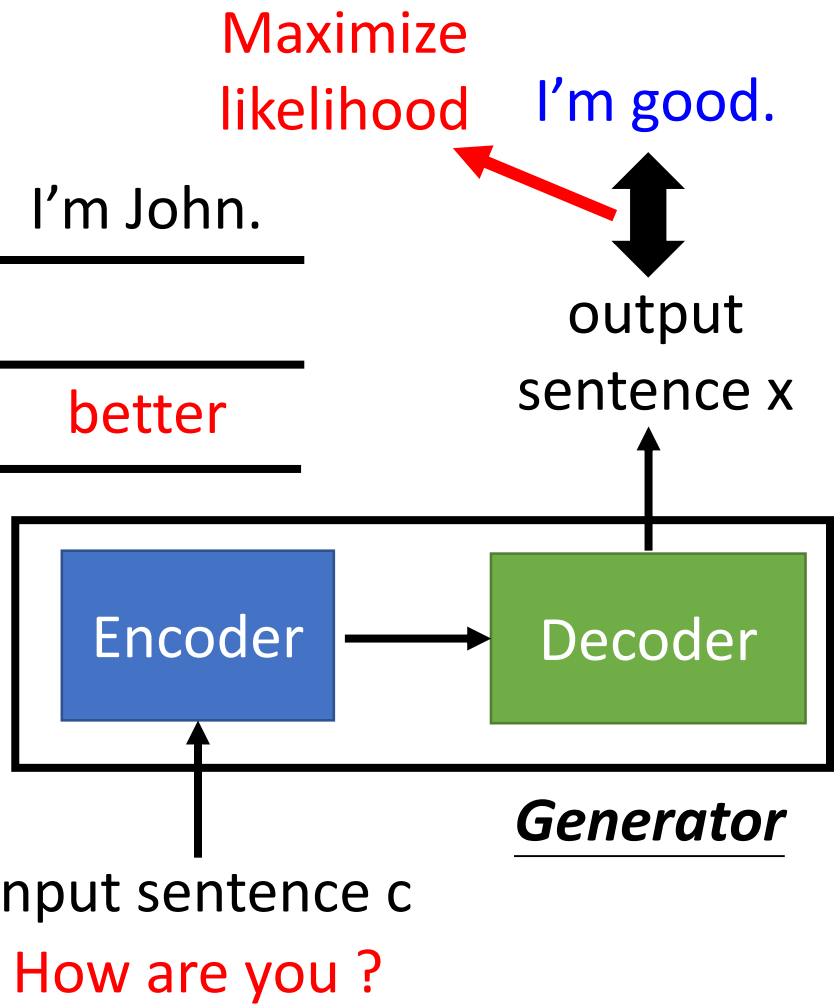
- Chat-bot as example

Output:	Not bad	I'm John.
Human	better	
Training Criterion		better

Training data:

A: How are you ?

B: I'm good.



Outline of Part III

Improving Supervised Seq-to-seq Model

- RL (human feedback)
- GAN (discriminator feedback)

Unsupervised Seq-to-seq Model

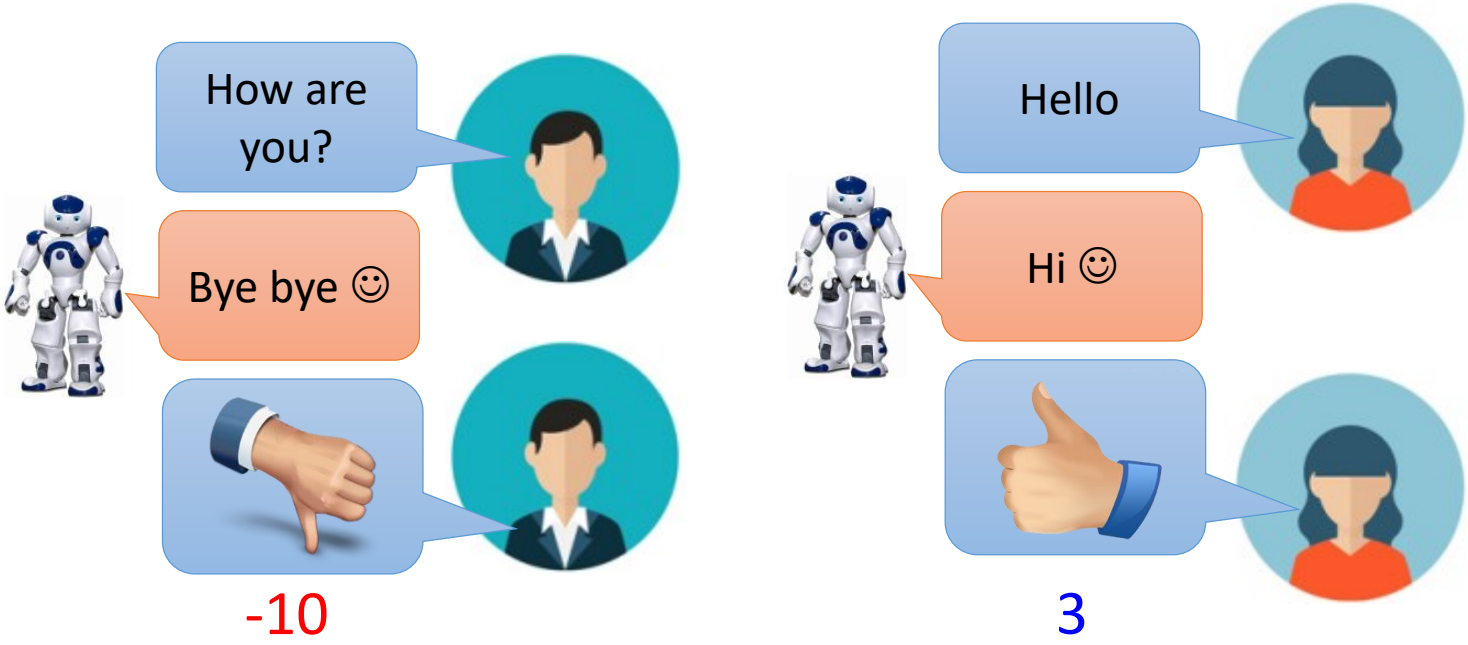
- Text Style Transfer
- Unsupervised Abstractive Summarization
- Unsupervised Translation

https://image.freepik.com/free-vector/variety-of-human-avatars_23-2147506285.jpg

http://www.freepik.com/free-vector/variety-of-human-avatars_766615.htm

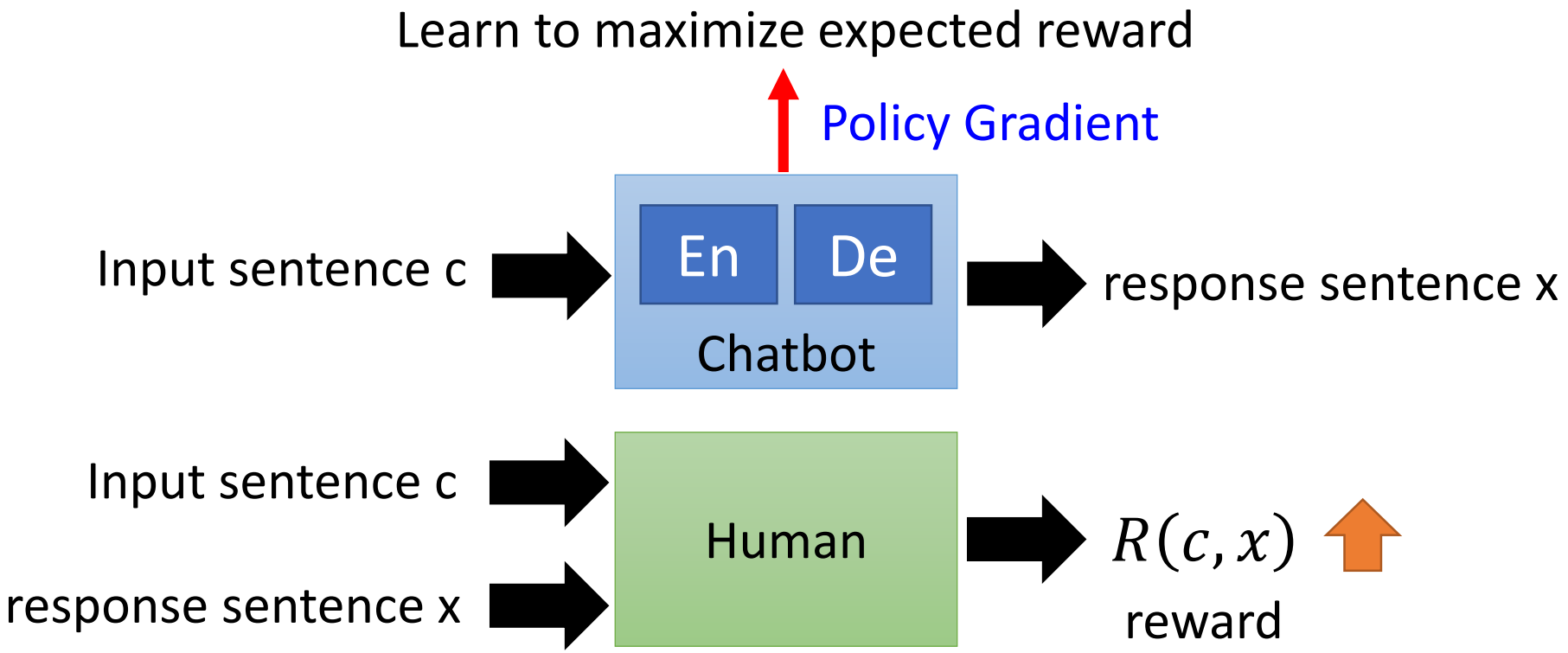
Introduction

- Machine obtains feedback from user



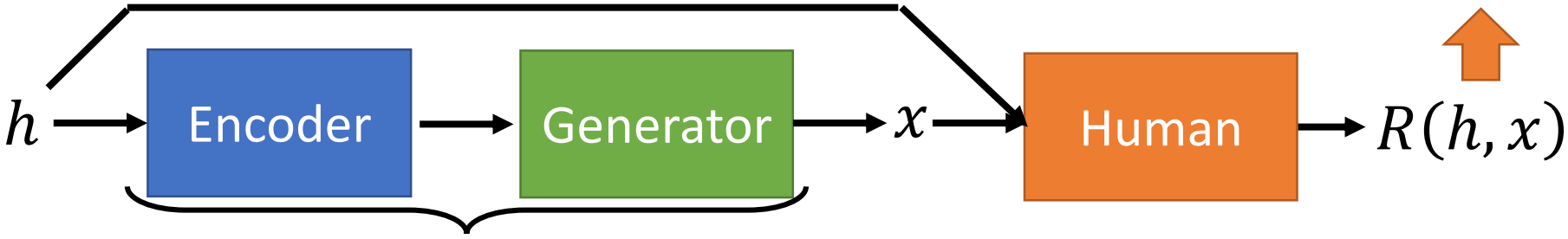
- Chat-bot learns to maximize the ***expected reward***

Maximizing Expected Reward



[Li, et al., EMNLP, 2016]

Maximizing Expected Reward



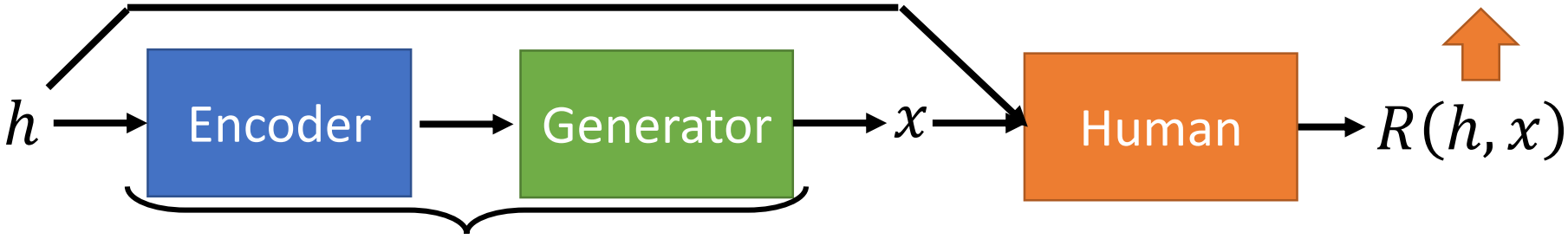
$$\theta^* = arg\max_{\theta} \bar{R}_\theta$$ ⟸ Maximizing expected reward

$$\bar{R}_\theta = \sum_h P(h) \sum_x R(h,x)\, P_\theta(x|h)$$

Randomness in generator

Probability that the input/history is h

Maximizing Expected Reward



$\theta^* = arg\max_\theta \bar{R}_\theta$ ⟵ Maximizing expected reward

$$\bar{R}_\theta = \sum_h P(h) \sum_x R(h,x)\, P_\theta(x|h) = E_{h\sim P(h)}\left[E_{x\sim P_\theta(x|h)}[R(h,x)]\right]$$

$$= E_{h\sim P(h),x\sim P_\theta(x|h)}[R(h,x)] \approx \frac{1}{N}\sum_{i=1}^{N} R\left(h^i, x^i\right)$$

Sample: $(h^1, x^1), (h^2, x^2), \cdots, (h^N, x^N)$

Where is θ?

Policy Gradient

$$\frac{d\log(f(x))}{dx} = \frac{1}{f(x)}\frac{df(x)}{dx}$$

$$\bar{R}_\theta = \sum_h P(h) \sum_x R(h,x)\, P_\theta(x|h) \approx \frac{1}{N}\sum_{i=1}^{N} R\left(h^i, x^i\right)$$

$$\nabla \bar{R}_\theta = \sum_h P(h) \sum_x R(h,x) \nabla P_\theta(x|h) \approx \frac{1}{N}\sum_{i=1}^{N} R\left(h^i, x^i\right) \nabla \log P_\theta(x|h)$$

Sampling

$$= \sum_h P(h) \sum_x R(h,x) P_\theta(x|h) \frac{\nabla P_\theta(x|h)}{P_\theta(x|h)}$$

$$= \sum_h P(h) \sum_x R(h,x) P_\theta(x|h) \nabla \log P_\theta(x|h)$$

$$= E_{h\sim P(h), x\sim P_\theta(x|h)}[R(h,x) \nabla \log P_\theta(x|h)]$$

Policy Gradient

- Gradient Ascent

$$\theta^{new} \leftarrow \theta^{old} + \eta \nabla \bar{R}_{\theta^{old}}$$

$$\nabla \bar{R}_\theta \approx \frac{1}{N} \sum_{i=1}^{N} R\left(h^i, x^i\right) \nabla log P_\theta\left(x^i | h^i\right)$$

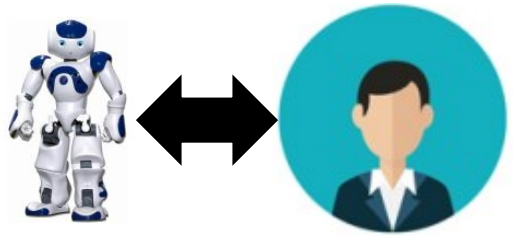
$R\left(h^i, x^i\right)$ is positive

➡ After updating θ, $P_\theta\left(x^i | h^i\right)$ will increase

$R\left(h^i, x^i\right)$ is negative

➡ After updating θ, $P_\theta\left(x^i | h^i\right)$ will decrease

Policy Gradient - Implemenation

θ^t 

(c^1, x^1) $R(c^1, x^1)$

(c^2, x^2) $R(c^2, x^2)$

(c^N, x^N) $R(c^N, x^N)$

$$\theta^{t+1} \leftarrow \theta^t + \eta \nabla \bar{R}_{\theta^t}$$

$$\frac{1}{N}\sum_{i=1}^{N} R(c^i, x^i) \nabla log P_{\theta^t}(x^i|c^i)$$

$R(c^i, x^i)$ is positive

Updating θ to increase $P_\theta(x^i|c^i)$

$R(c^i, x^i)$ is negative

Updating θ to decrease $P_\theta(x^i|c^i)$

Comparison

	Maximum Likelihood	Reinforcement Learning
Objective Function	$\frac{1}{N}\sum_{i=1}^{N} logP_\theta(\hat{x}^i \mid c^i)$	$\frac{1}{N}\sum_{i=1}^{N} R(c^i, x^i) logP_\theta(x^i \mid c^i)$
Gradient	$\frac{1}{N}\sum_{i=1}^{N} \nabla logP_\theta(\hat{x}^i \mid c^i)$	$\frac{1}{N}\sum_{i=1}^{N} R(c^i, x^i) \nabla logP_\theta(x^i \mid c^i)$
Training Data	$\{(c^1, \hat{x}^1), \dots, (c^N, \hat{x}^N)\}$ $R(c^i, \hat{x}^i) = 1$	$\{(c^1, x^1), \dots, (c^N, x^N)\}$ obtained from interaction weighted by $R(c^i, x^i)$

Alpha GO style training !

- Let two agents talk to each other


How old are you?


See you.


How old are you?

I am 16.



See you.


See you.


I though you were 12.

What make you think so?


Using a pre-defined evaluation function to compute R(h,x)

Outline of Part III

Improving Supervised Seq-to-seq Model

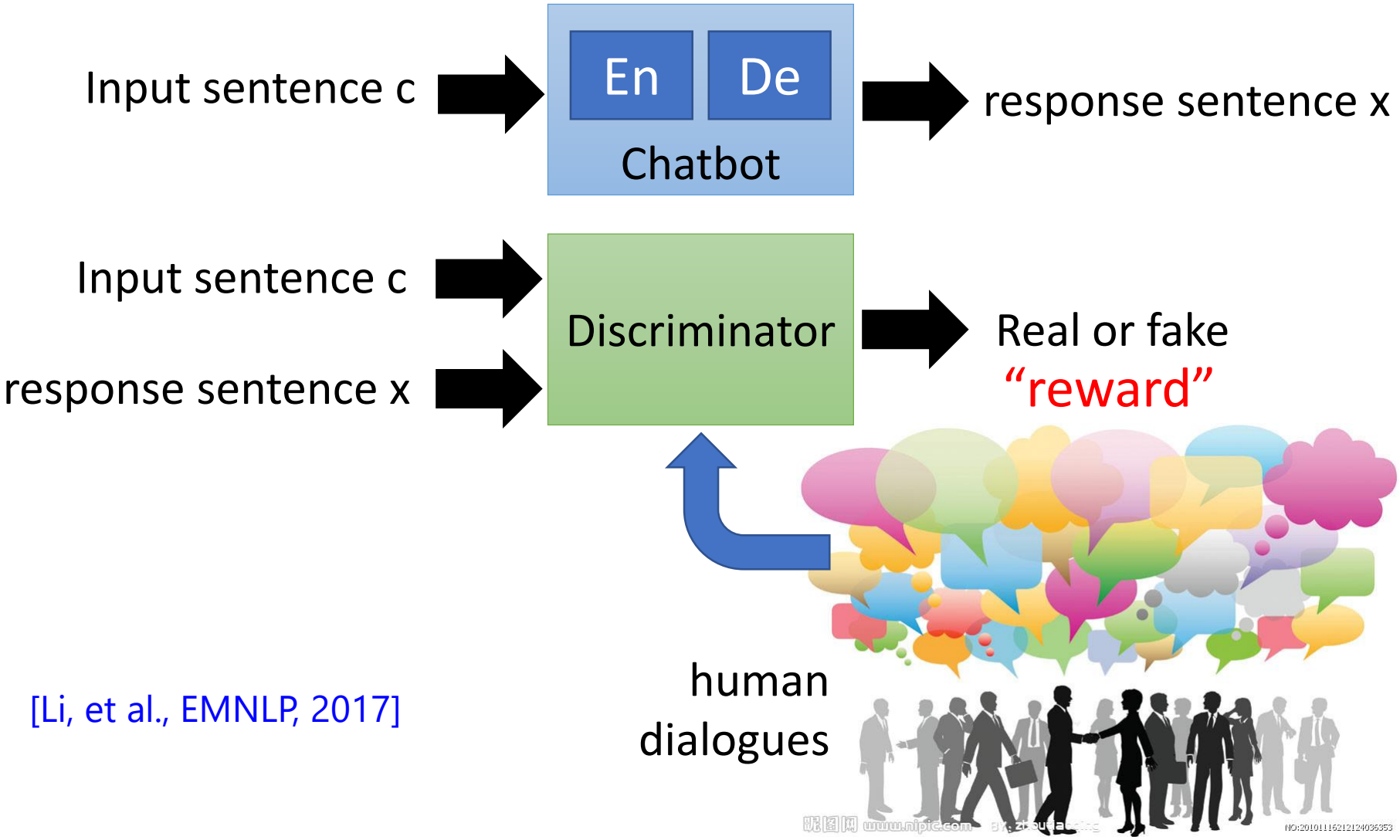
- RL (human feedback)
- GAN (discriminator feedback)

Unsupervised Seq-to-seq Model

- Text Style Transfer
- Unsupervised Abstractive Summarization
- Unsupervised Translation

http://www.nipic.com/show/3/83/3936650kd7476069.html

Conditional GAN



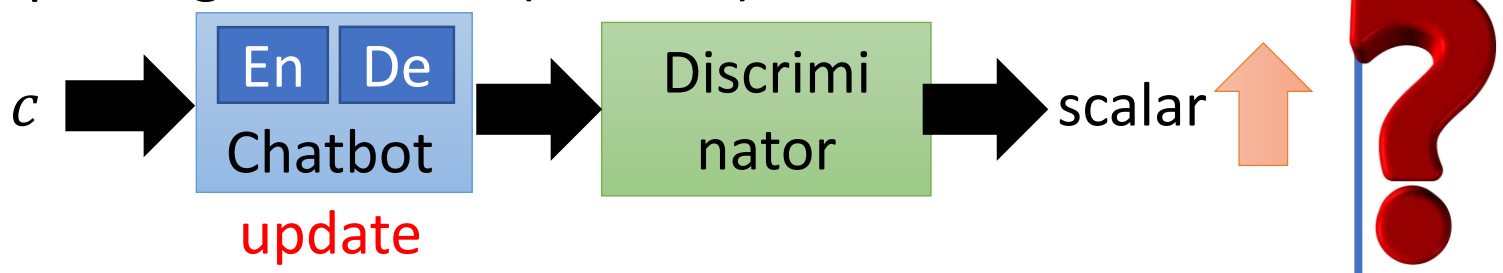
[Li, et al., EMNLP, 2017]

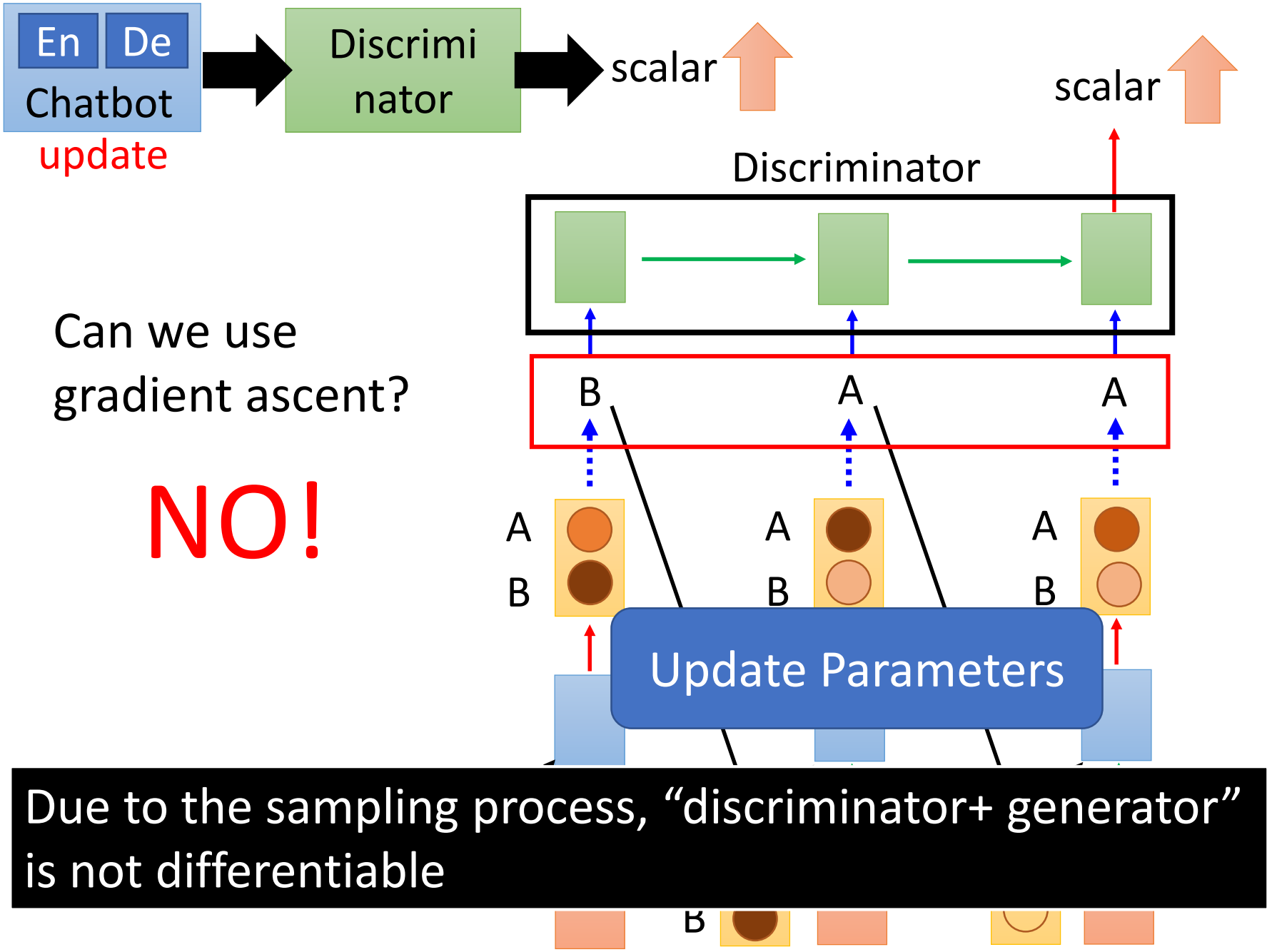
Algorithm

Training data:

Pairs of conditional input c and response x

- Initialize generator G (chatbot) and discriminator D
- In each iteration:
  - Sample input c and response x from training set
  - Sample input c' from training set, and generate response $\tilde{x}$ by G(c')
  - Update D to increase $D(c, x)$ and decrease $D(c', \tilde{x})$
- Update generator G (chatbot) such that




En
De
Chatbot
update
Discrimi
nator
scalar
scalar
Discriminator
Can we use
gradient ascent?
NO!
B
A
A
A
B
A
B
A
B
Update Parameters
Due to the sampling process, “discriminator+ generator” is not differentiable

Three Categories of Solutions

Gumbel-softmax

- [Matt J. Kusner, et al, arXiv, 2016]

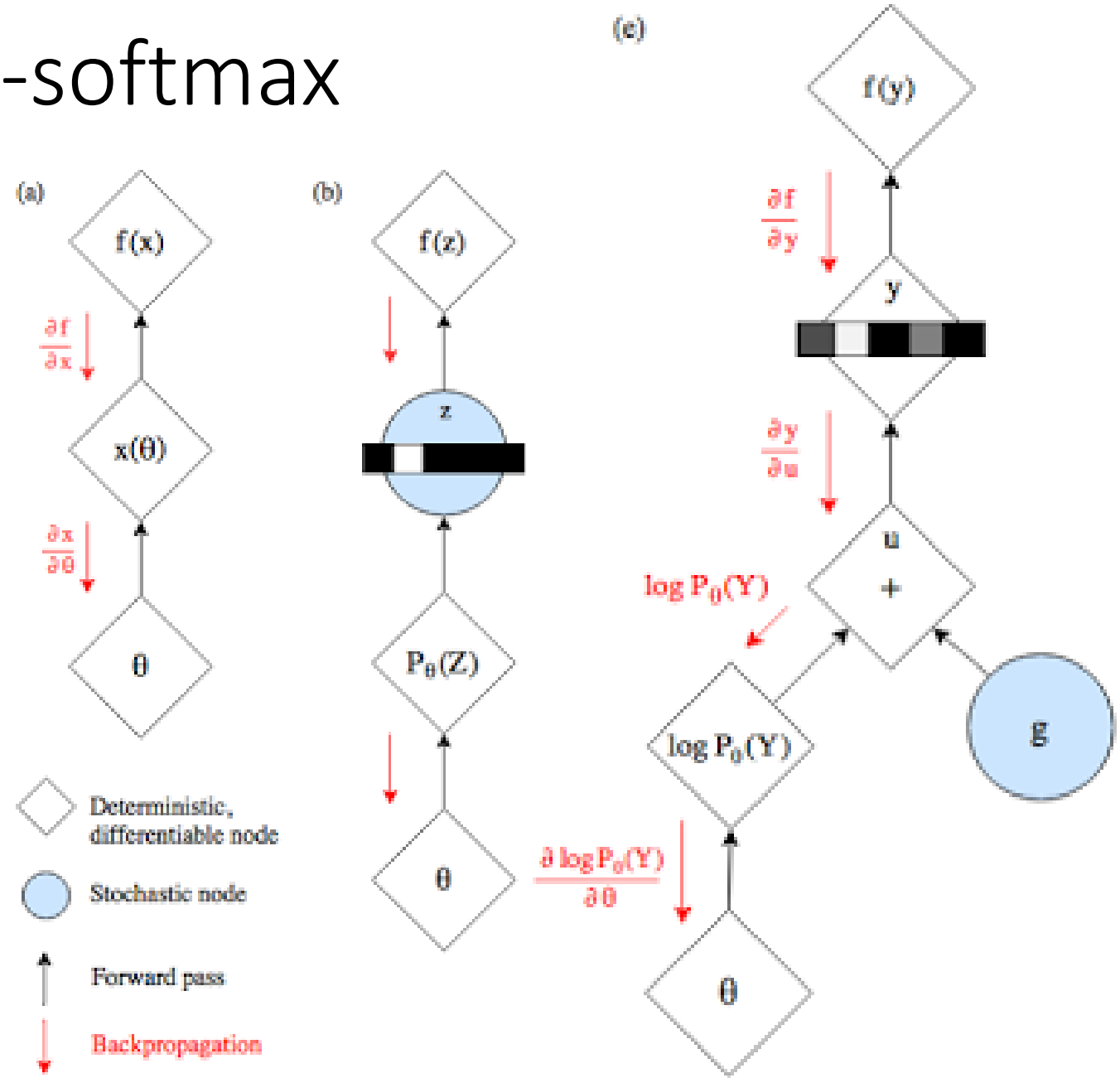
Continuous Input for Discriminator

- [Sai Rajeswar, et al., arXiv, 2017][Ofir Press, et al., ICML workshop, 2017][Zhen Xu, et al., EMNLP, 2017][Alex Lamb, et al., NIPS, 2016][Yizhe Zhang, et al., ICML, 2017]

"Reinforcement Learning"

- [Yu, et al., AAAI, 2017][Li, et al., EMNLP, 2017][Tong Che, et al, arXiv, 2017][Jiaxian Guo, et al., AAAI, 2018][Kevin Lin, et al, NIPS, 2017][William Fedus, et al., ICLR, 2018]

Gumbel-softmax



https://gabrielhuang.gitbooks.io/machine-learning/reparametrization-trick.html

https://casmls.github.io/general/2017/02/01/GumbelSoftmax.html

http://blog.evjang.com/2016/11/tutorial-categorical-variational.html

Three Categories of Solutions

Gumbel-softmax

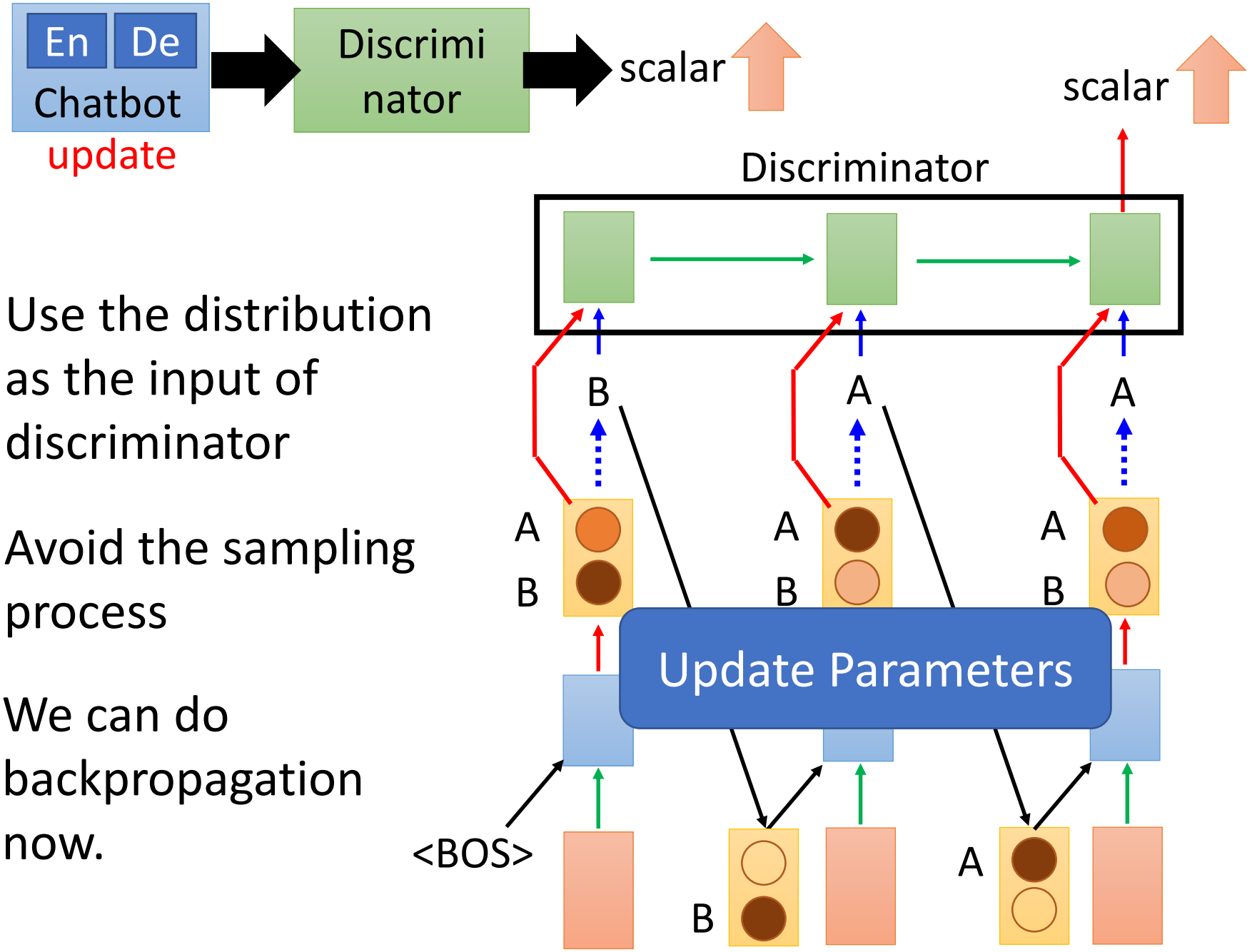
- [Matt J. Kusner, et al, arXiv, 2016]

Continuous Input for Discriminator

- [Sai Rajeswar, et al., arXiv, 2017][Ofir Press, et al., ICML workshop, 2017][Zhen Xu, et al., EMNLP, 2017][Alex Lamb, et al., NIPS, 2016][Yizhe Zhang, et al., ICML, 2017]

"Reinforcement Learning"

- [Yu, et al., AAAI, 2017][Li, et al., EMNLP, 2017][Tong Che, et al, arXiv, 2017][Jiaxian Guo, et al., AAAI, 2018][Kevin Lin, et al, NIPS, 2017][William Fedus, et al., ICLR, 2018]


En
De
Chatbot
update
Discrimi
nator
scalar
scalar
Discriminator
Use the distribution as the input of discriminator
Avoid the sampling process
We can do backpropagation now.
B
A
A
A
B
A
B
A
B
Update Parameters
<BOS>
B
A

What is the problem?

- Real sentence

1	0	0	0	0
0	1	0	0	0
0	0	1	0	0
0	0	0	1	0
0	0	0	0	1

Discriminator can immediately find the difference.

- Generated

0.9	0.1	0.1	0	0
0.1	0.9	0.1	0	0
0	0	0.7	0.1	0
0	0	0.1	0.8	0.1
0	0	0	0.1	0.9

Can never be 1-of-N

WGAN is helpful

Three Categories of Solutions

Gumbel-softmax

- [Matt J. Kusner, et al, arXiv, 2016]

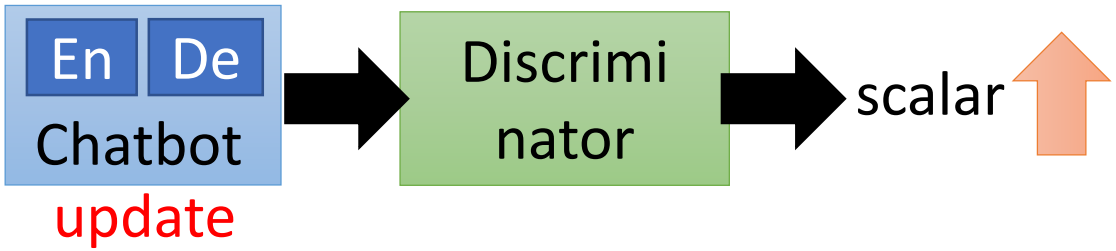
Continuous Input for Discriminator

- [Sai Rajeswar, et al., arXiv, 2017][Ofir Press, et al., ICML workshop, 2017][Zhen Xu, et al., EMNLP, 2017][Alex Lamb, et al., NIPS, 2016][Yizhe Zhang, et al., ICML, 2017]

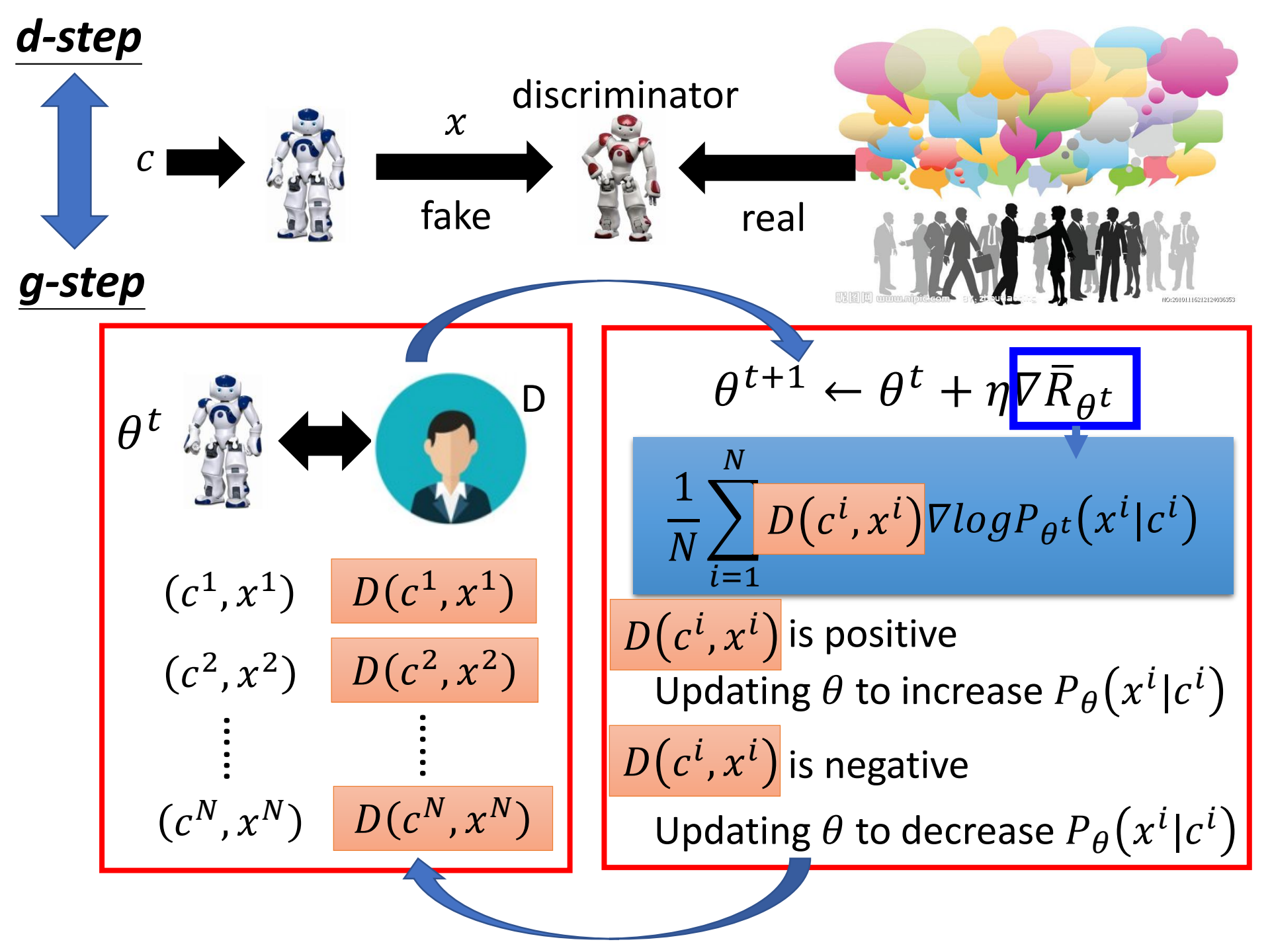
"Reinforcement Learning"

- [Yu, et al., AAAI, 2017][Li, et al., EMNLP, 2017][Tong Che, et al, arXiv, 2017][Jiaxian Guo, et al., AAAI, 2018][Kevin Lin, et al, NIPS, 2017][William Fedus, et al., ICLR, 2018]

Reinforcement Learning?



- Consider the output of discriminator as reward
  - Update generator to increase discriminator = to get maximum reward
  - Using the formulation of policy gradient, replace reward $R(c, x)$ with discriminator output $\mathrm{D}(c, x)$
- Different from typical RL
  - The discriminator would update


d-step
g-step
c
x
discriminator
fake
real
θ^t
D
(c^1, x^1)
D(c^1, x^1)
(c^2, x^2)
D(c^2, x^2)
(c^N, x^N)
D(c^N, x^N)
θ^{t+1} ← θ^t + η∇R̄_{θ^t}
1/N Σ_{i=1}^{N} D(c^i, x^i) ∇log P_{θ^t}(x^i|c^i)
D(c^i, x^i) is positive
Updating θ to increase P_θ(x^i|c^i)
D(c^i, x^i) is negative
Updating θ to decrease P_θ(x^i|c^i)

Reward for Every Generation Step

$$\nabla \bar{R}_\theta \approx \frac{1}{N}\sum_{i=1}^{N} D(c^i, x^i)\nabla log P_\theta(x^i|c^i)$$

c^i = "What is your name?"

x^i = "I don't know"

$D(c^i, x^i)$ is negative

Update θ to decrease $\log P_\theta(x^i|c^i)$

$$log P_\theta(x^i|c^i) = log P(x_1^i|c^i) + log P(x_2^i|c^i, x_1^i) + log P(x_3^i|c^i, x_{1:2}^i)$$

$P("I"|c^i)$ ↓ ? ↓ ↓

c^i = "What is your name?"

x^i = "I am John"

$D(c^i, x^i)$ is positive

Update θ to increase $\log P_\theta(x^i|c^i)$

$$log P_\theta(x^i|c^i) = log P(x_1^i|c^i) + log P(x_2^i|c^i, x_1^i) + log P(x_3^i|c^i, x_{1:2}^i)$$

$P("I"|c^i)$ ↑ ↑ ↑

Reward for Every Generation Step

h^i = "What is your name?" x^i = "I don't know"

$$logP_\theta(x^i|h^i) = logP(x_1^i|c^i) + logP(x_2^i|c^i, x_1^i) + logP(x_3^i|c^i, x_{1:2}^i)$$

$P("I"|c^i)$ $P("don't"|c^i, "I")$ $P("know"|c^i, "I\ don't")$

$$\nabla \bar{R}_\theta \approx \frac{1}{N}\sum_{i=1}^{N} D(c^i, x^i)\nabla logP_\theta(x^i|c^i)$$

$$\nabla \bar{R}_\theta \approx \frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{T}\left(Q(c^i, x_{1:t}^i) - b\right)\nabla logP_\theta(x_t^i|c^i, x_{1:t-1}^i)$$

Method 1. Monte Carlo (MC) Search [Yu, et al., AAAI, 2017]

Method 2. Discriminator For Partially Decoded Sequences [Li, et al., EMNLP, 2017]

Tips: RankGAN

Kevin Lin, Dianqi Li, Xiaodong He, Zhengyou Zhang, Ming-Ting Sun, "Adversarial Ranking for Language Generation", NIPS 2017

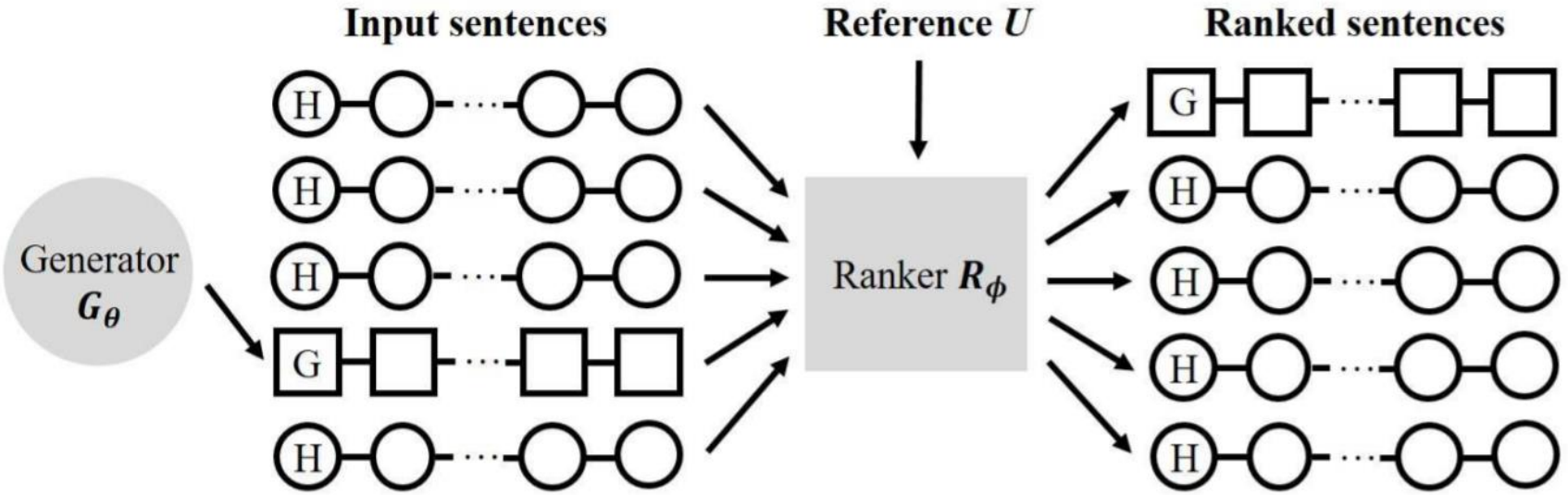


Image caption generation:

Method	BLEU-2	BLEU-3	BLEU-4
MLE	0.781	0.624	0.589
SeqGAN	0.815	0.636	0.587
RankGAN	**0.845**	**0.668**	**0.614**

Method	Human score
SeqGAN	3.44
RankGAN	4.61
Human-written	**6.42**

Experimental Results

Input	We've got to look for another route.
MLE	I'm sorry.
GAN	You're not going to be here for a while.

Input	You can save him by talking.
MLE	I don't know.
GAN	You know what's going on in there, you know what I mean?

- MLE frequently generates “I’m sorry”, “I don’t know”, etc. (corresponding to fuzzy images?)
- GAN generates longer and more complex responses (however, no strong evidence shows that they are better)

Find more comparison in the survey papers.

[Lu, et al., arXiv, 2018][Zhu, et al., arXiv, 2018]

More Applications

- Supervised machine translation [Wu, et al., arXiv 2017][Yang, et al., arXiv 2017]
- Supervised abstractive summarization [Liu, et al., AAAI 2018]
- Image/video caption generation [Rakshith Shetty, et al., ICCV 2017][Liang, et al., arXiv 2017]

If you are using seq2seq models,
consider to improve them by GAN.

Outline of Part III

Conditional Sequence Generation

- RL (human feedback)
- GAN (discriminator feedback)

Unsupervised Conditional Sequence Generation

- Text Style Transfer
- Unsupervised Abstractive Summarization
- Unsupervised Translation

Text Style Transfer

Domain X

male

Domain Y

female

It is good.

It's a good day.

I love you.

positive sentences

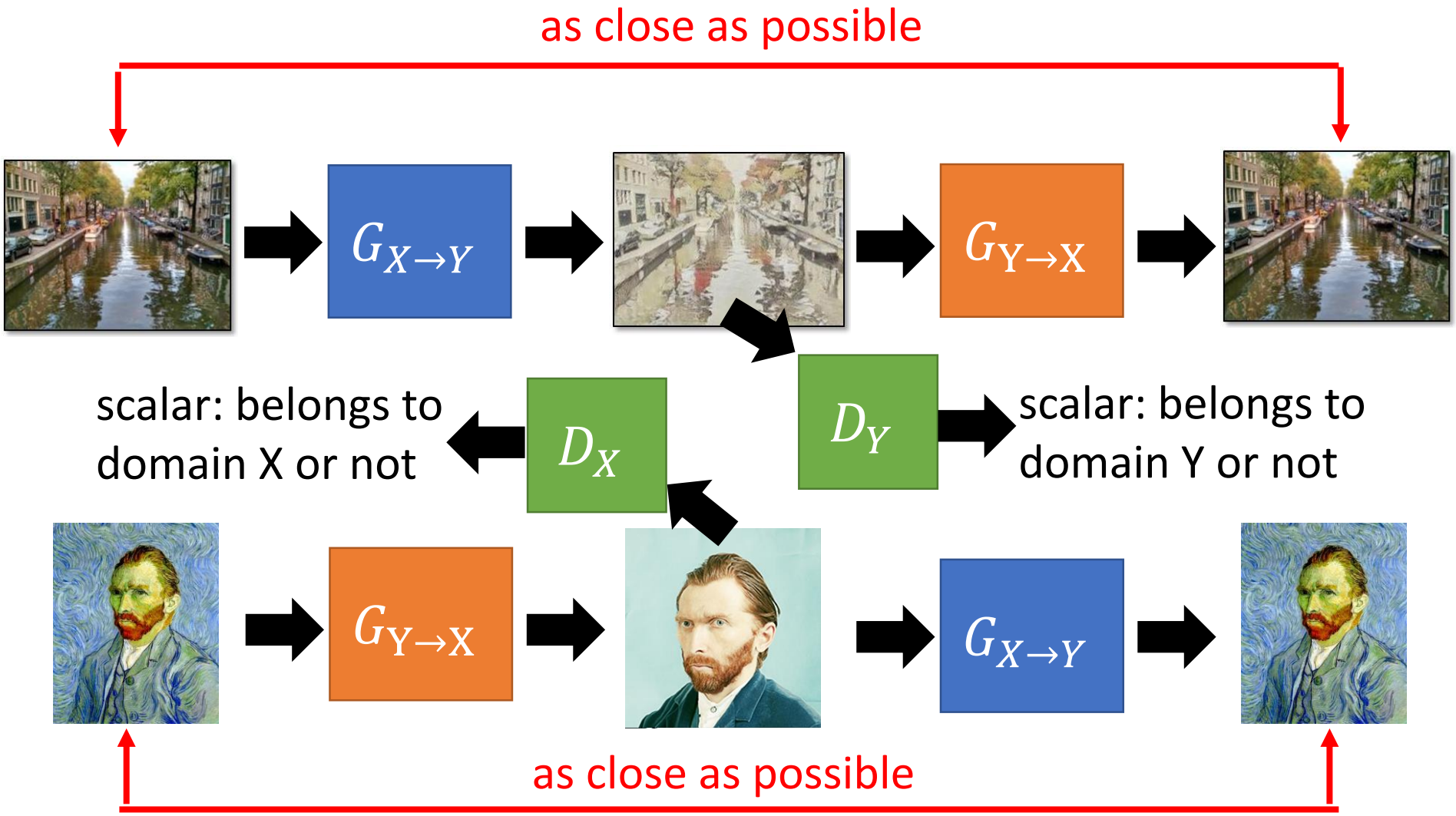
It is bad.

It's a bad day.

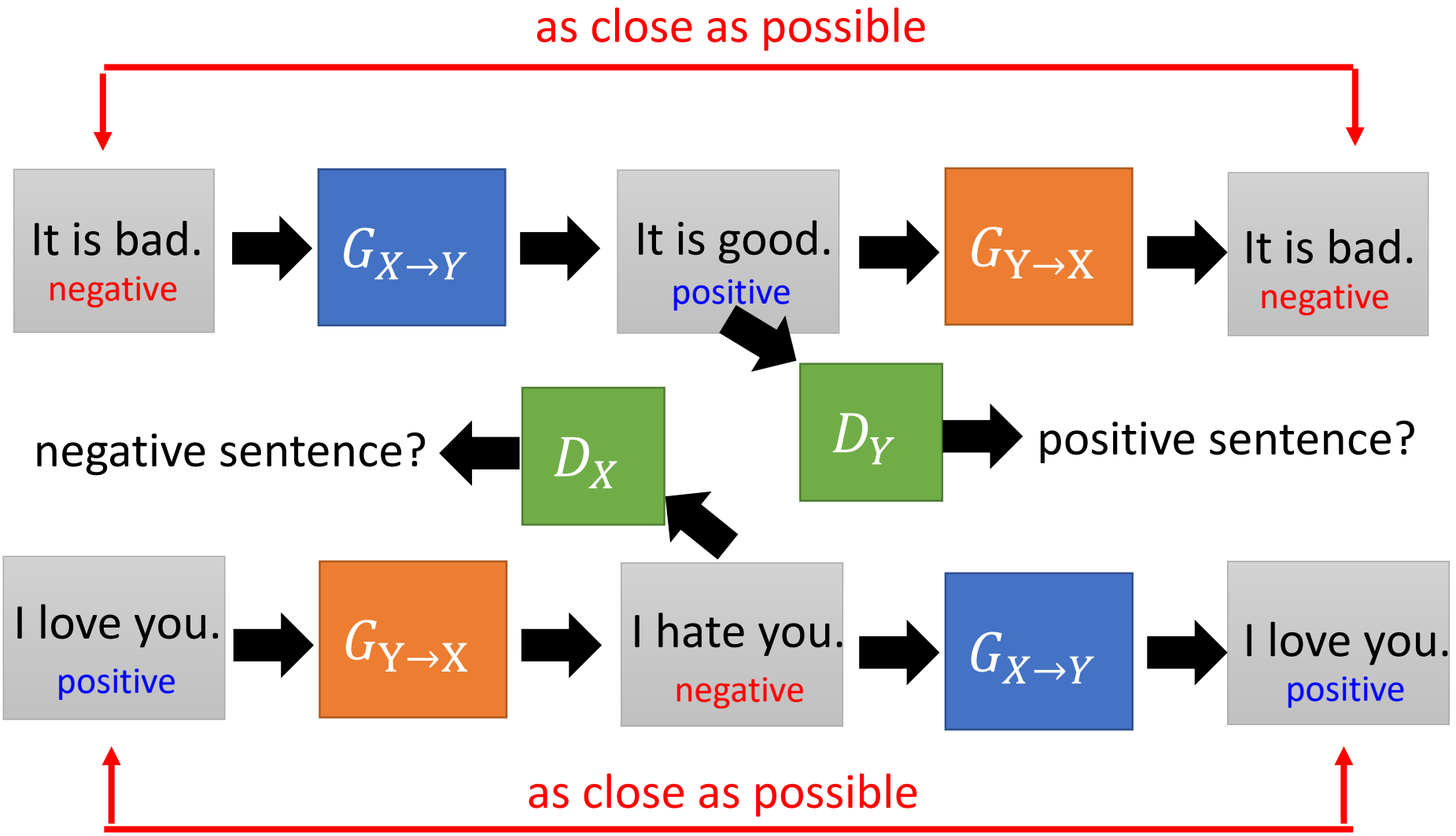
I don't love you.

negative sentences

Direct Transformation



Direct Transformation

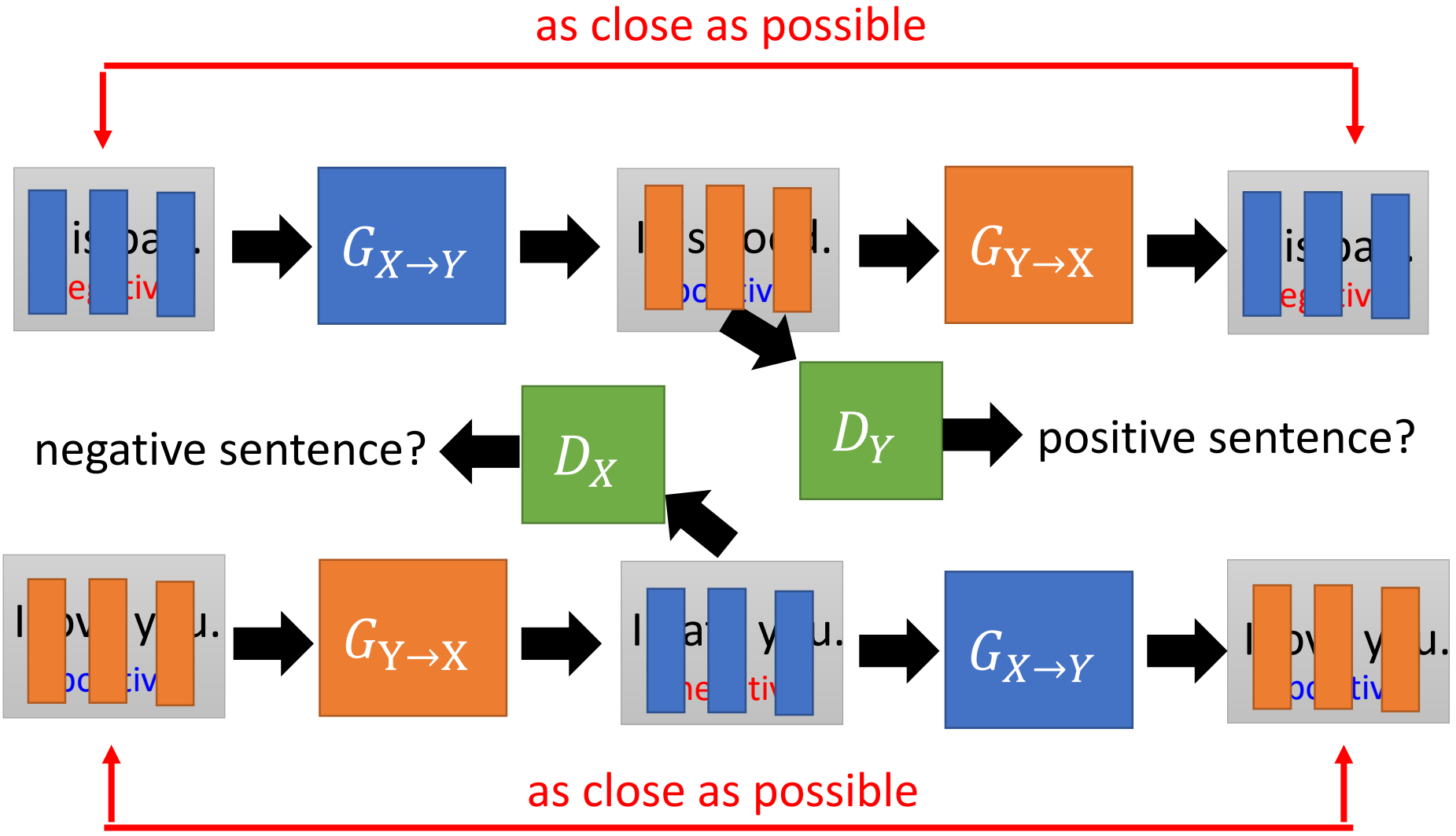


Direct Transformation

Discrete?

Word embedding

[Lee, et al., ICASSP, 2018]



- Negative sentence to positive sentence:
  it's a crappy day → it's a great day
  i wish you could be here → you could be here
  it's not a good idea → it's good idea
  i miss you → i love you
  i don't love you → i love you
  i can't do that → i can do that
  i feel so sad → i happy
  it's a bad day → it's a good day
  it's a dummy day → it's a great day
  sorry for doing such a horrible thing → thanks for doing a great thing
  my doggy is sick → my doggy is my doggy
  my little doggy is sick → my little doggy is my little doggy

Projection to Common Space

EN_X

EN_Y

DE_X

DE_Y

D_X

D_Y

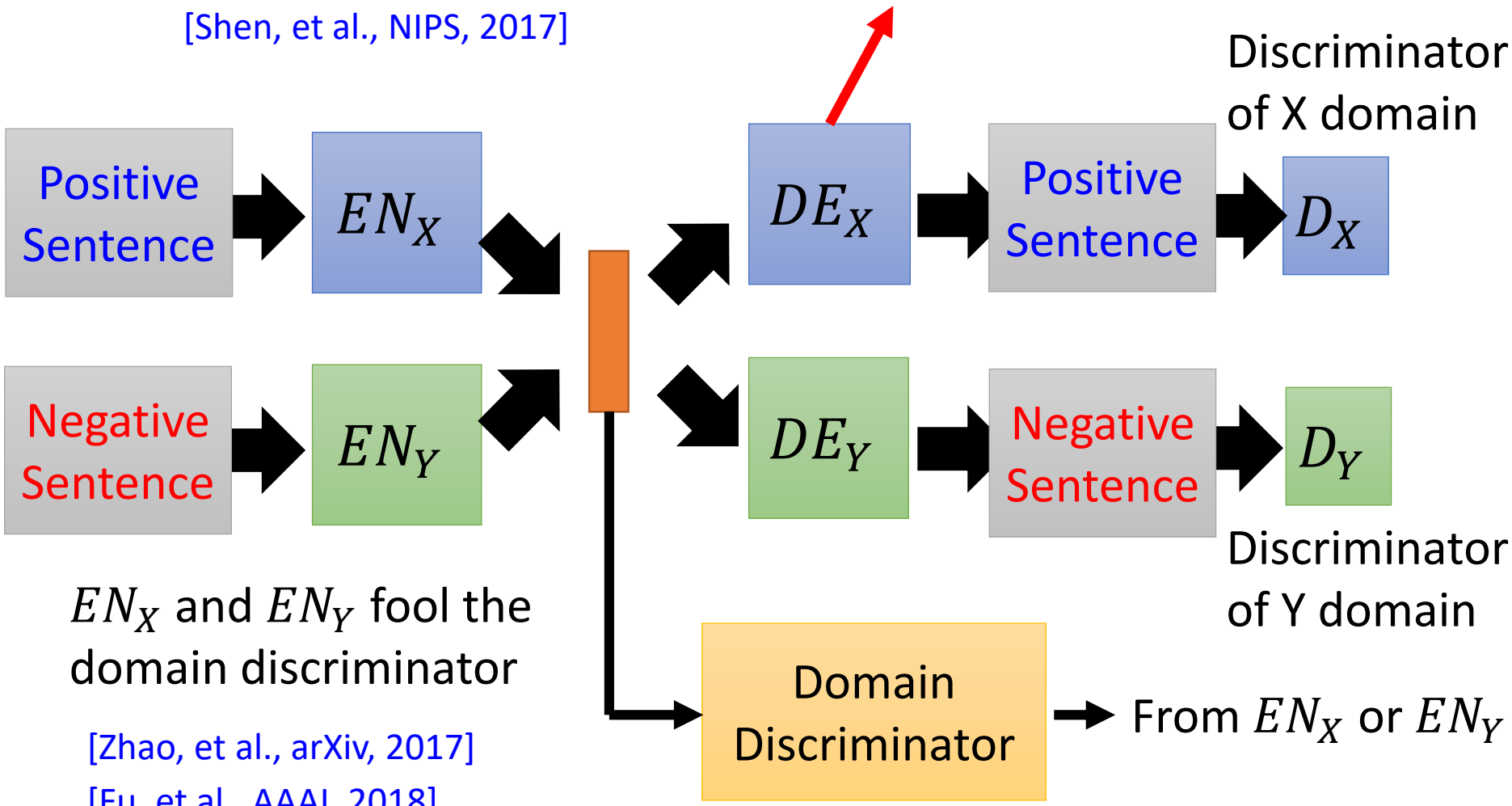
Discriminator of X domain

Discriminator of Y domain

Projection to Common Space

Decoder hidden layer as discriminator input

[Shen, et al., NIPS, 2017]



EN_X and EN_Y fool the domain discriminator

[Zhao, et al., arXiv, 2017]

[Fu, et al., AAAI, 2018]

Outline of Part III

Improving Supervised Seq-to-seq Model

- RL (human feedback)
- GAN (discriminator feedback)

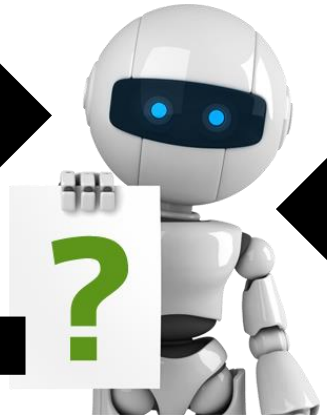
Unsupervised Seq-to-seq Model

- Text Style Transfer
- Unsupervised Abstractive Summarization
- Unsupervised Translation

Abstractive Summarization

- Now machine can do **abstractive summary** by seq2seq (write summaries in its own words)

summary
(in its own words)



seq2seq

summary 1

summary 2

summary 3

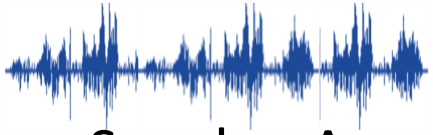
Training Data

Supervised: We need lots of labelled training data.

Review:
Unsupervised Conditional Generation

Domain X

Domain Y



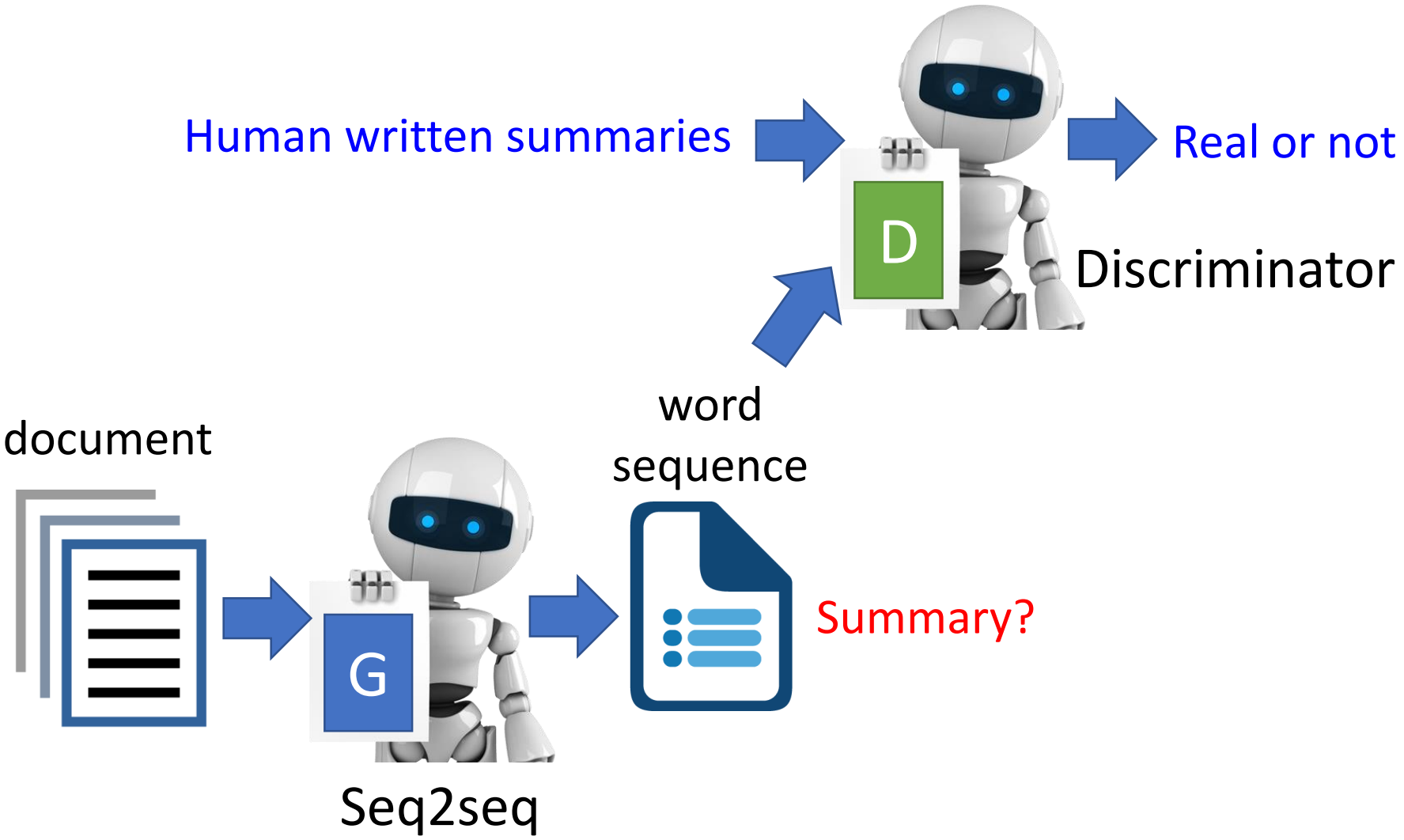
Speaker A

Speaker B

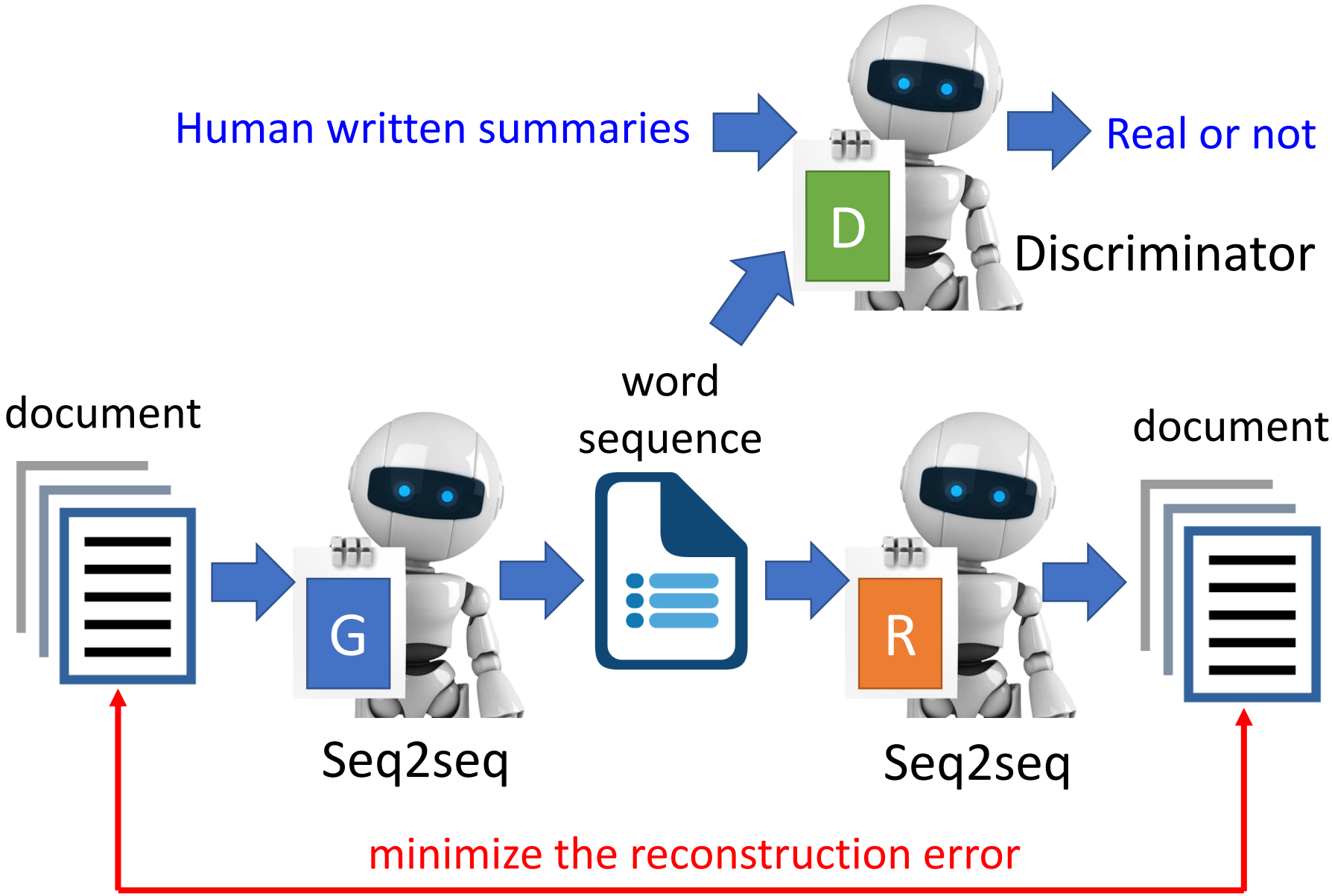
document

summary

Unsupervised Abstractive Summarization



Unsupervised Abstractive Summarization

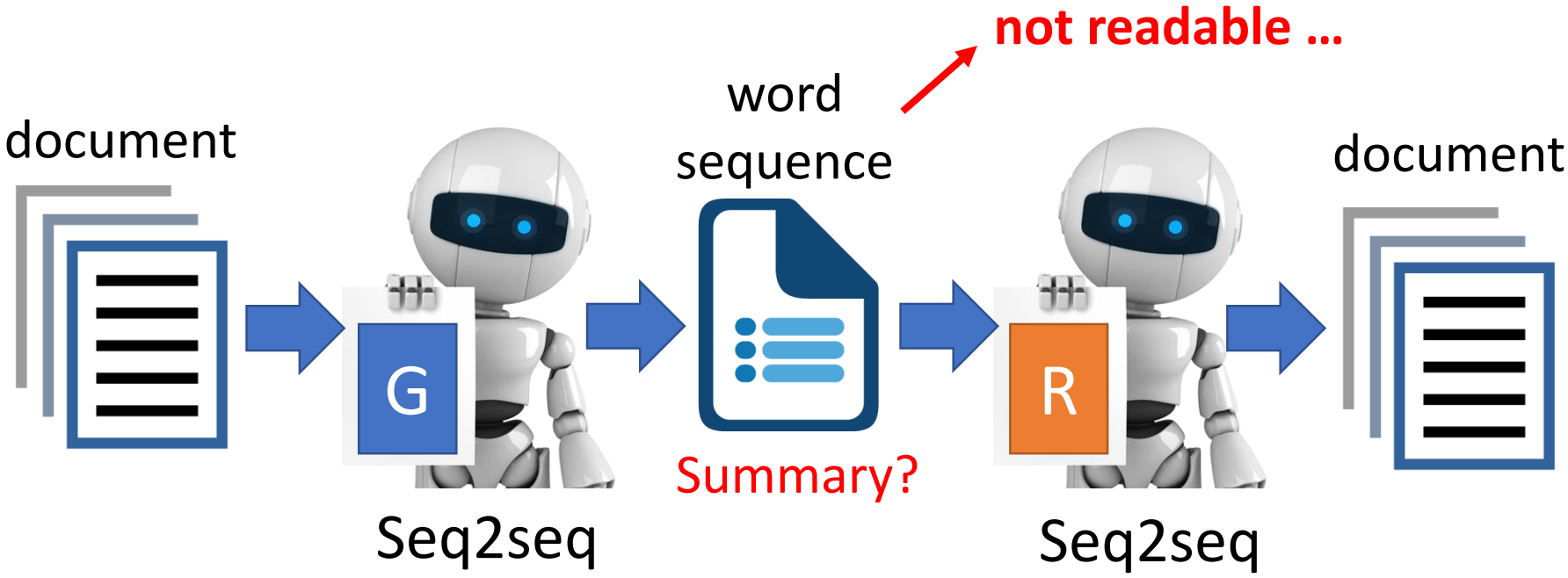


Unsupervised Abstractive Summarization

Only need a lot of documents to train the model

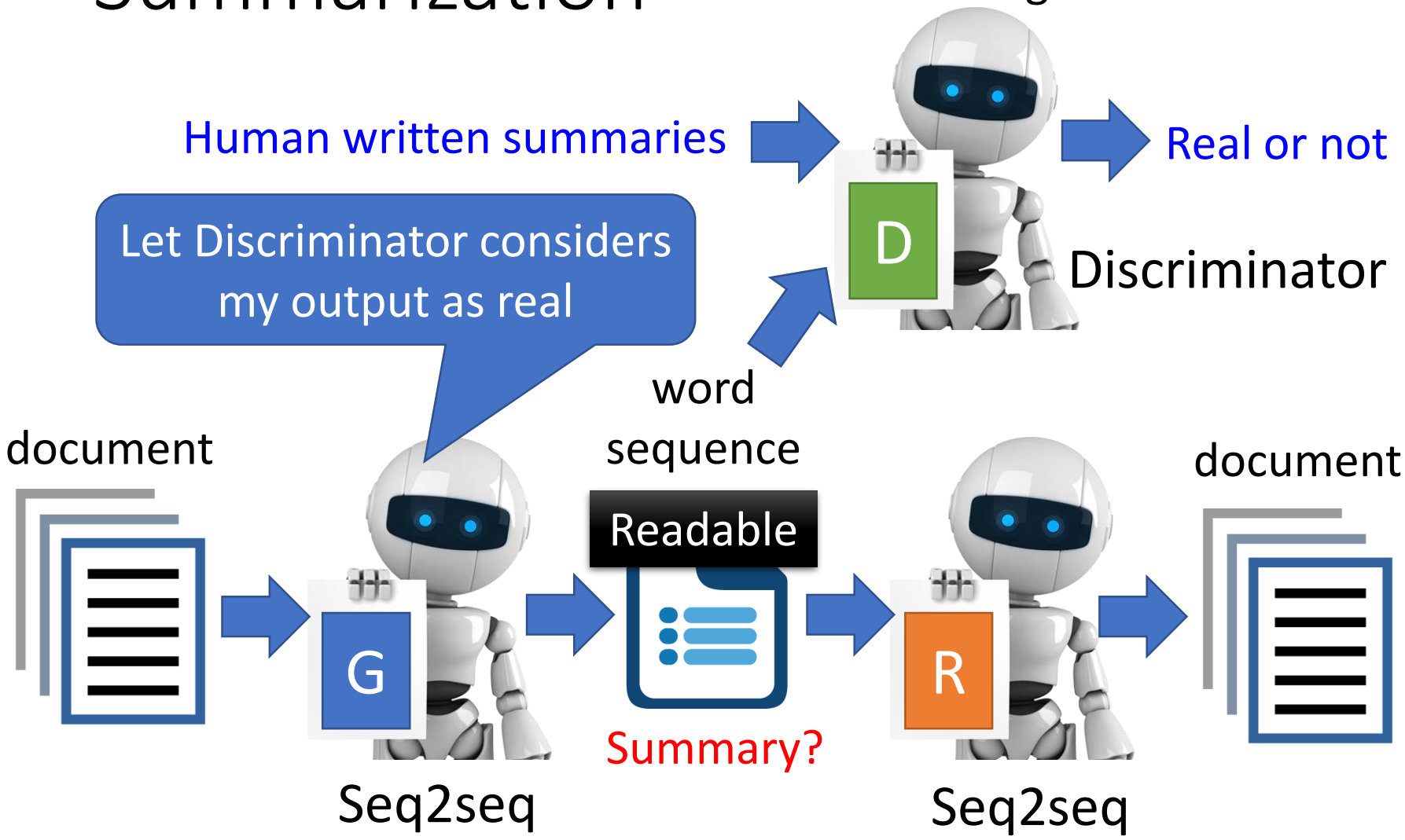
This is a ***seq2seq2seq auto-encoder***.

Using a sequence of words as latent representation.



Unsupervised Abstractive Summarization

REINFORCE algorithm is used.



感謝 王耀賢 同學提供實驗結果

Unsupervised Abstractive Summarization

- ***Document***:澳大利亞今天與13個國家簽署了反興奮劑雙邊協議,旨在加強體育競賽之外的藥品檢查並共享研究成果 ……
- ***Summary***:
  - Human:澳大利亞與13國簽署反興奮劑協議
  - Unsupervised:澳大利亞加強體育競賽之外的藥品檢查
- ***Document***:中華民國奧林匹克委員會今天接到一九九二年冬季奧運會邀請函,由於主席張豐緒目前正在中南美洲進行友好訪問,因此尚未決定是否派隊赴賽 ……
- ***Summary***:
  - Human:一九九二年冬季奧運會函邀我參加
  - Unsupervised:奧委會接獲冬季奧運會邀請函

感謝 王耀賢 同學提供實驗結果

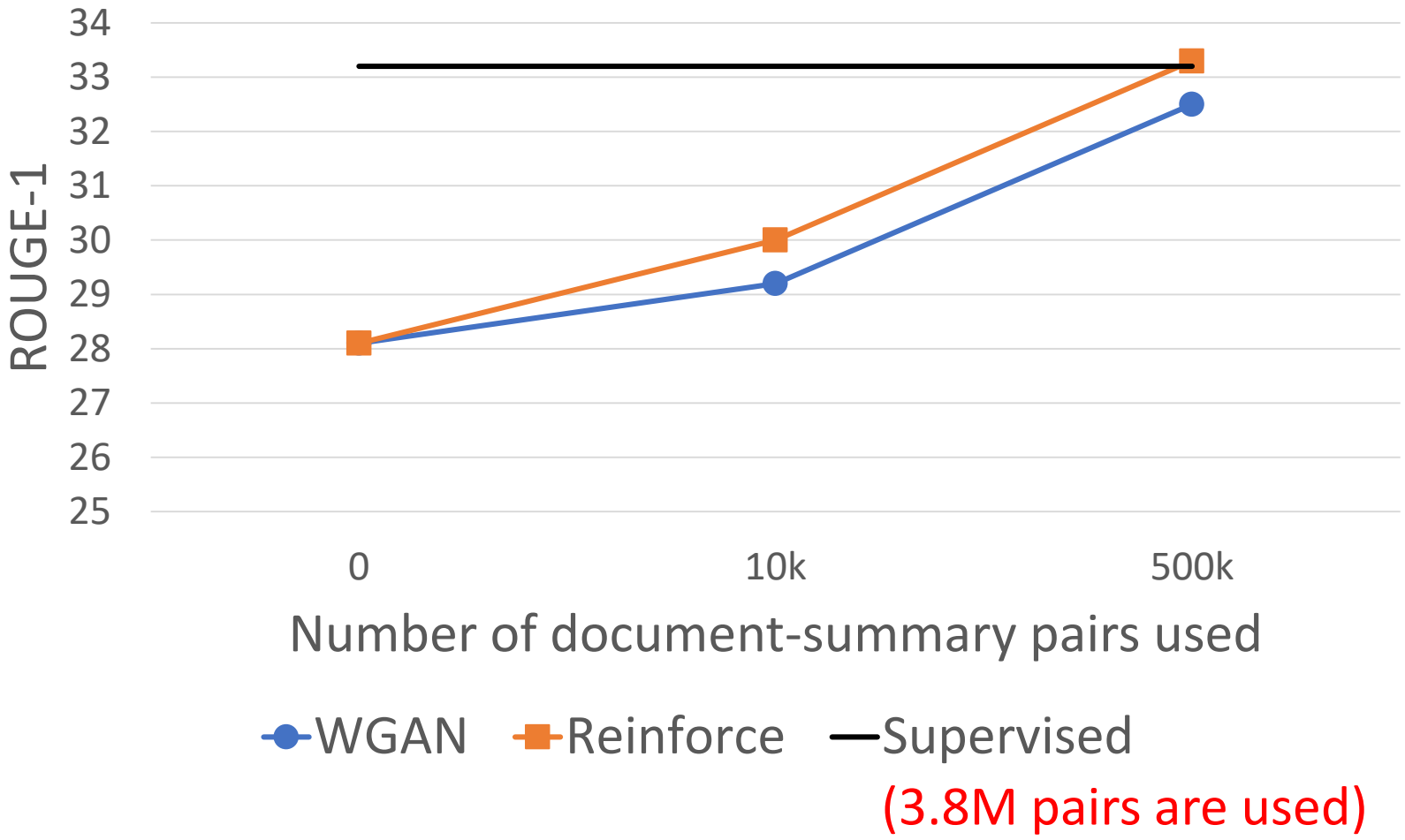
Unsupervised Abstractive Summarization

- ***Document***:據此間媒體27日報道,印度尼西亞蘇門答臘島的兩個省近日來連降暴雨,洪水泛濫導致塌方,到26日為止至少已有60人喪生,100多人失蹤 ……
- ***Summary***:
  - Human:印尼水災造成60人死亡
  - Unsupervised:印尼門洪水泛濫導致塌雨
- ***Document***:安徽省合肥市最近為領導幹部下基層做了新規定:一律輕車簡從,不準搞迎來送往、不準搞層層陪同 ……
- ***Summary***:
  - Human:合肥規定領導幹部下基層活動從簡
  - Unsupervised:合肥領導幹部下基層做搞迎來送往規定:一律簡

(unpublished result)

Semi-supervised Learning

Using matched data



Outline of Part III

Improving Supervised Seq-to-seq Model

- RL (human feedback)
- GAN (discriminator feedback)

Unsupervised Seq-to-seq Model

- Text Style Transfer
- Unsupervised Abstractive Summarization
- Unsupervised Translation

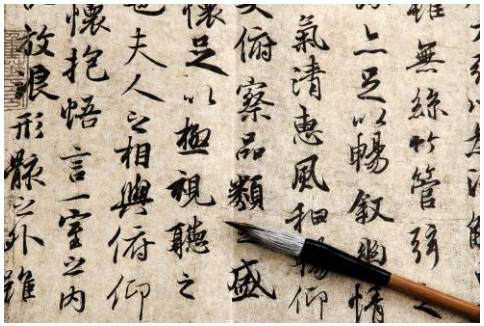
Unsupervised Machine Translation

Domain X

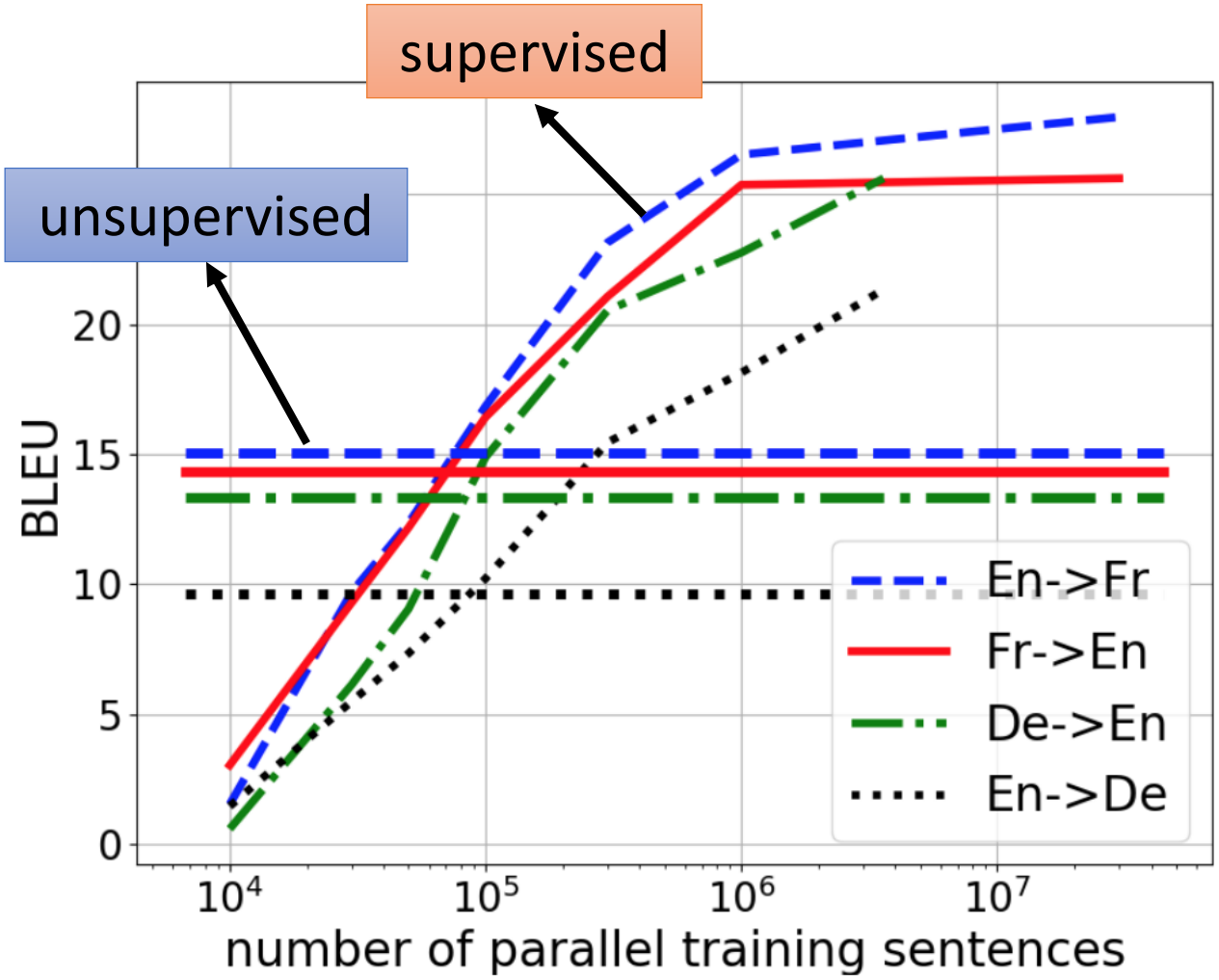
male

Domain Y

female



[Alexis Conneau, et al., ICLR, 2018]
[Guillaume Lample, et al., ICLR, 2018]

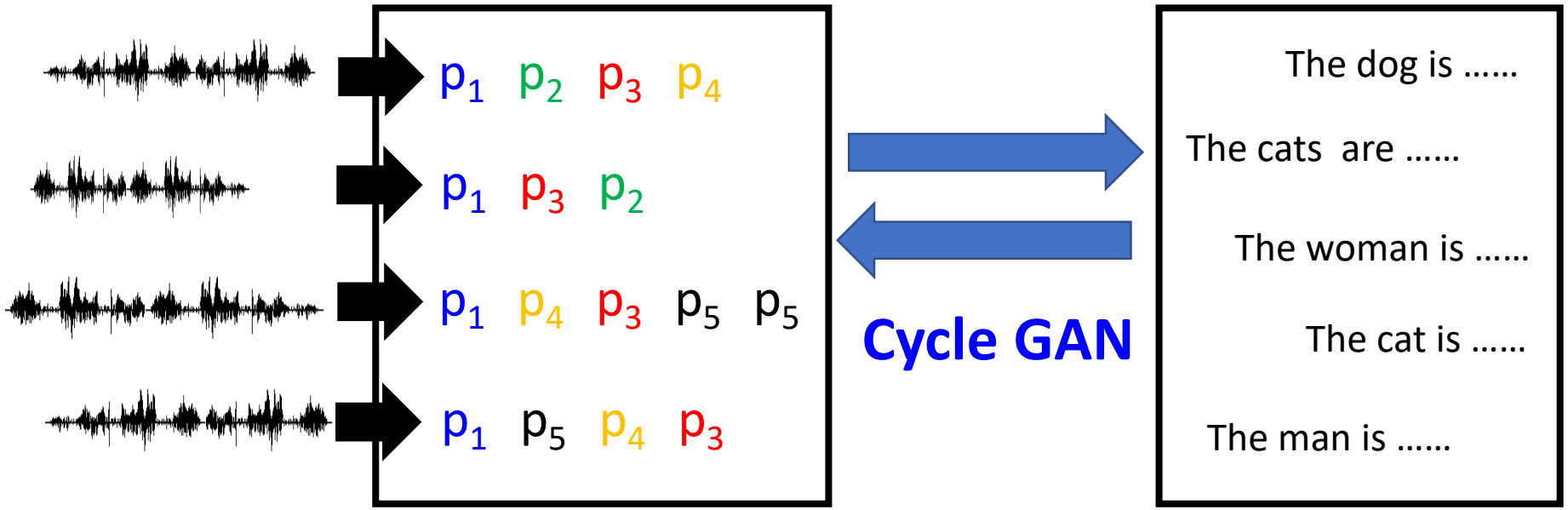


Unsupervised learning with 10M sentences

=

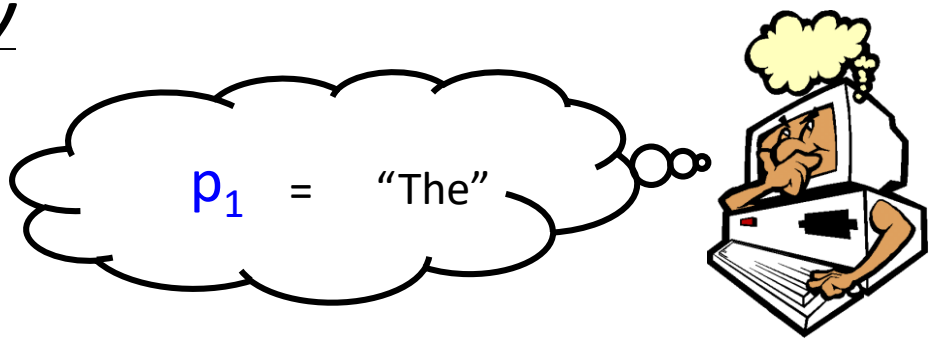
Supervised learning with 100K sentence pairs

Unsupervised Speech Recognition



Acoustic Pattern Discovery

Can we achieve unsupervised speech recognition?

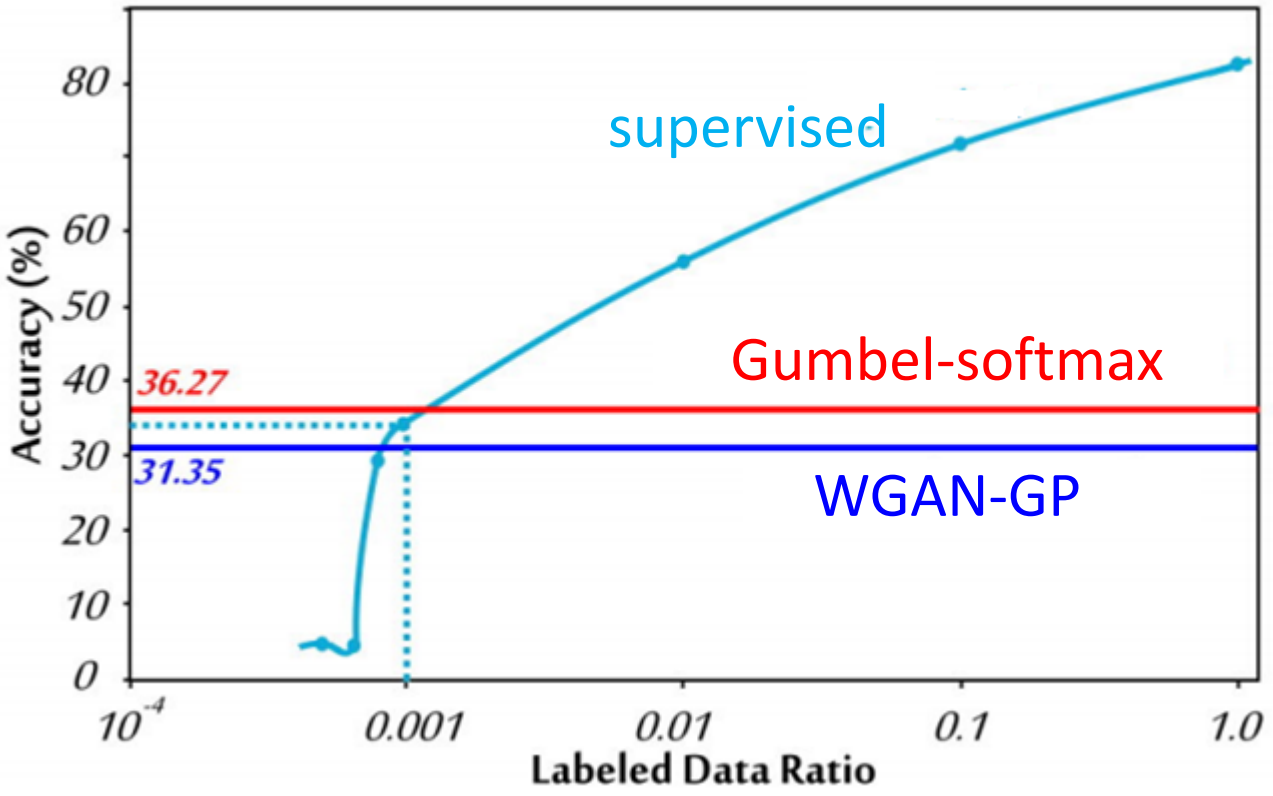


[Liu, et al., arXiv, 2018] [Chen, et al., arXiv, 2018]

Unsupervised Speech Recognition

- Phoneme recognition

Audio: TIMIT
Text: WMT



Concluding Remarks

Conditional Sequence Generation

- RL (human feedback)
- GAN (discriminator feedback)

Unsupervised Conditional Sequence Generation

- Text Style Transfer
- Unsupervised Abstractive Summarization
- Unsupervised Translation

Concluding Remarks

from A to Z

A	B	C	D	E	F
ACGAN	BiGAN	CycleGAN	DCGAN DuelGAN	EBGAN	fGAN

G	H	I	J	K	L
GAN	?	InfoGAN	?	?	LSGAN

(only list those mentioned in class)

M

MMGAN

N

NSGAN

O

?

P

Progressive GAN

Q

?

R

Rank GAN

S

StackGAN

StarGAN

SeqGAN

T

Triple GAN

U

Unroll GAN

V

VAEGAN

W

WGAN

X

XGAN

Y ?

Z ?

Reference

- **Conditional Sequence Generation**
  - Jiwei Li, Will Monroe, Alan Ritter, Michel Galley, Jianfeng Gao, Dan Jurafsky, Deep Reinforcement Learning for Dialogue Generation, EMNLP, 2016
  - Jiwei Li, Will Monroe, Tianlin Shi, Sébastien Jean, Alan Ritter, Dan Jurafsky, Adversarial Learning for Neural Dialogue Generation, EMNLP, 2017
  - Matt J. Kusner, José Miguel Hernández-Lobato, GANS for Sequences of Discrete Elements with the Gumbel-softmax Distribution, arXiv 2016
  - Tong Che, Yanran Li, Ruixiang Zhang, R Devon Hjelm, Wenjie Li, Yangqiu Song, Yoshua Bengio, Maximum-Likelihood Augmented Discrete Generative Adversarial Networks, arXiv 2017
  - Lantao Yu, Weinan Zhang, Jun Wang, Yong Yu, SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient, AAAI 2017

Reference

- **Conditional Sequence Generation**
  - Sai Rajeswar, Sandeep Subramanian, Francis Dutil, Christopher Pal, Aaron Courville, Adversarial Generation of Natural Language, arXiv, 2017
  - Ofir Press, Amir Bar, Ben Bogin, Jonathan Berant, Lior Wolf, Language Generation with Recurrent Generative Adversarial Networks without Pre-training, ICML workshop, 2017
  - Zhen Xu, Bingquan Liu, Baoxun Wang, Chengjie Sun, Xiaolong Wang, Zhuoran Wang, Chao Qi , Neural Response Generation via GAN with an Approximate Embedding Layer, EMNLP, 2017
  - Alex Lamb, Anirudh Goyal, Ying Zhang, Saizheng Zhang, Aaron Courville, Yoshua Bengio, Professor Forcing: A New Algorithm for Training Recurrent Networks, NIPS, 2016
  - Yizhe Zhang, Zhe Gan, Kai Fan, Zhi Chen, Ricardo Henao, Dinghan Shen, Lawrence Carin, Adversarial Feature Matching for Text Generation, ICML, 2017
  - Jiaxian Guo, Sidi Lu, Han Cai, Weinan Zhang, Yong Yu, Jun Wang, Long Text Generation via Adversarial Training with Leaked Information, AAAI, 2018
  - Kevin Lin, Dianqi Li, Xiaodong He, Zhengyou Zhang, Ming-Ting Sun, Adversarial Ranking for Language Generation, NIPS, 2017
  - William Fedus, Ian Goodfellow, Andrew M. Dai, MaskGAN: Better Text Generation via Filling in the_____, ICLR, 2018

Reference

- **Conditional Sequence Generation**
  - Sidi Lu, Yaoming Zhu, Weinan Zhang, Jun Wang, Yong Yu, Neural Text Generation: Past, Present and Beyond, arXiv, 2018
  - Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, Yong Yu, Texygen: A Benchmarking Platform for Text Generation Models, arXiv, 2018
  - Zhen Yang, Wei Chen, Feng Wang, Bo Xu, Improving Neural Machine Translation with Conditional Sequence Generative Adversarial Nets, NAACL, 2018
  - Lijun Wu, Yingce Xia, Li Zhao, Fei Tian, Tao Qin, Jianhuang Lai, Tie-Yan Liu, Adversarial Neural Machine Translation, arXiv 2017
  - Linqing Liu, Yao Lu, Min Yang, Qiang Qu, Jia Zhu, Hongyan Li, Generative Adversarial Network for Abstractive Text Summarization, AAAI 2018
  - Rakshith Shetty, Marcus Rohrbach, Lisa Anne Hendricks, Mario Fritz, Bernt Schiele, Speaking the Same Language: Matching Machine to Human Captions by Adversarial Training, ICCV 2017
  - Xiaodan Liang, Zhiting Hu, Hao Zhang, Chuang Gan, Eric P. Xing, Recurrent Topic-Transition GAN for Visual Paragraph Generation, arXiv 2017

Reference

- **Text Style Transfer**
  - Zhenxin Fu, Xiaoye Tan, Nanyun Peng, Dongyan Zhao, Rui Yan, Style Transfer in Text: Exploration and Evaluation, AAAI, 2018
  - Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola, Style Transfer from Non-Parallel Text by Cross-Alignment, NIPS 2017
  - Chih-Wei Lee, Yau-Shian Wang, Tsung-Yuan Hsu, Kuan-Yu Chen, Hung-Yi Lee, Lin-shan Lee, Scalable Sentiment for Sequence-to-sequence Chatbot Response with Performance Analysis, ICASSP, 2018
  - Junbo (Jake) Zhao, Yoon Kim, Kelly Zhang, Alexander M. Rush, Yann LeCun, Adversarially Regularized Autoencoders, arxiv, 2017

Reference

- **Unsupervised Machine Translation**
  - Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, Hervé Jégou, Word Translation Without Parallel Data, ICRL 2018
  - Guillaume Lample, Ludovic Denoyer, Marc'Aurelio Ranzato, Unsupervised Machine Translation Using Monolingual Corpora Only, ICRL 2018
- **Unsupervised Speech Recognition**
  - Da-Rong Liu, Kuan-Yu Chen, Hung-Yi Lee, Lin-shan Lee, Completely Unsupervised Phoneme Recognition by Adversarially Learning Mapping Relationships from Audio Embeddings, arXiv, 2018
  - Yi-Chen Chen, Chia-Hao Shen, Sung-Feng Huang, Hung-yi Lee, Towards Unsupervised Automatic Speech Recognition Trained by Unaligned Speech and Text only, arXiv, 2018

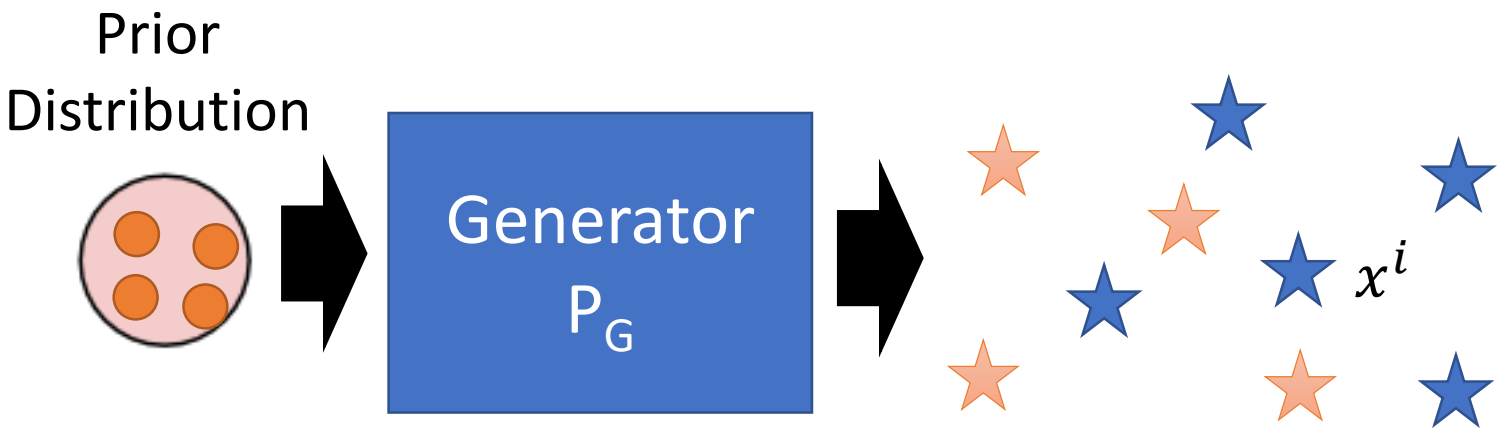
Evaluation

Ref: Lucas Theis, Aäron van den Oord, Matthias Bethge, “A note on the evaluation of generative models”, arXiv preprint, 2015

Likelihood

: real data (not observed during training)

: generated data

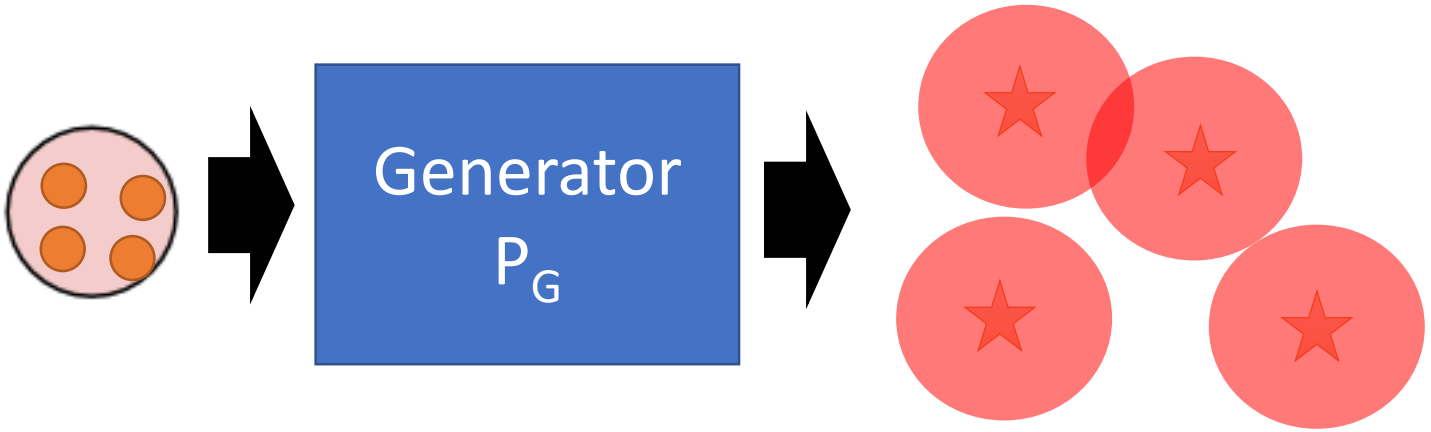


Log Likelihood: $L = \frac{1}{N}\sum_i log P_G(x^i)$

We cannot compute $P_G(x^i)$. We can only sample from P_G.

Likelihood
- Kernel Density Estimation

- Estimate the distribution of $P_G(x)$ from sampling



Each sample is the mean of a Gaussian with the same covariance.

Now we have an approximation of P_G, so we can compute $P_G(x^i)$ for each real data x^i

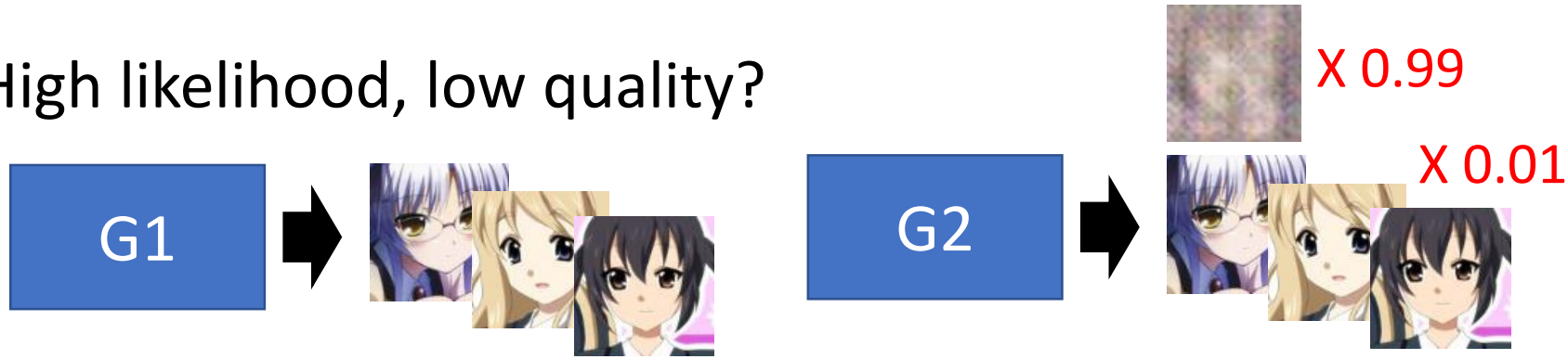
Then we can compute the likelihood.

Likelihood v.s. Quality

- Low likelihood, high quality?

  Considering a model generating good images (small variance)

- High likelihood, low quality?



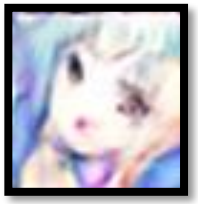
$$L = \frac{1}{N}\sum_i log\underbrace{P_G(x^i)}_{100} = -log100 + \underbrace{\frac{1}{N}\sum_i logP_G(x^i)}_{4.6}$$

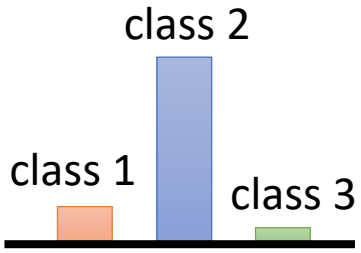
[Tim Salimans, et al., NIPS, 2016]

Objective Evaluation

x: image

y: class (output of CNN)

x  → Off-the-shelf Image Classifier → $P(y|x)$



e.g. Inception net, VGG, etc.

Concentrated distribution means higher visual quality

---

x^1 → CNN → $P(y^1|x^1)$

x^2 → CNN → $P(y^2|x^2)$

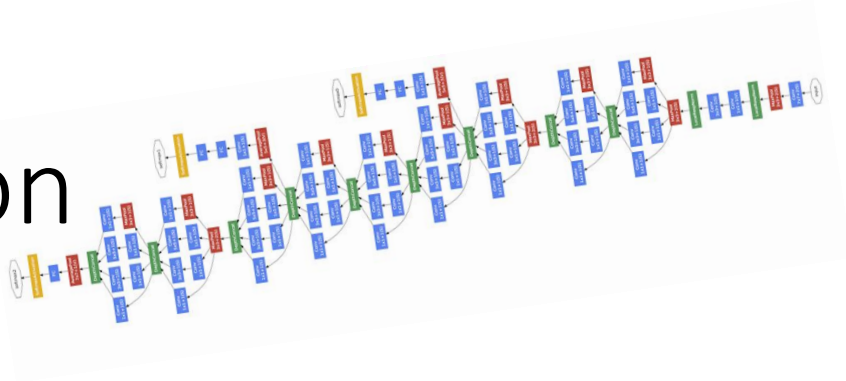
x^3 → CNN → $P(y^3|x^3)$

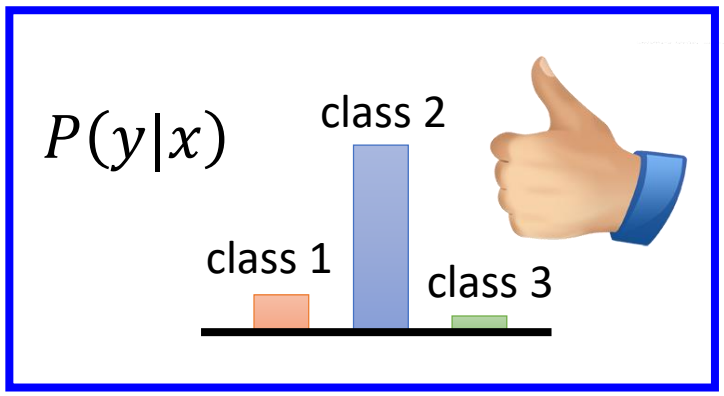
⋮

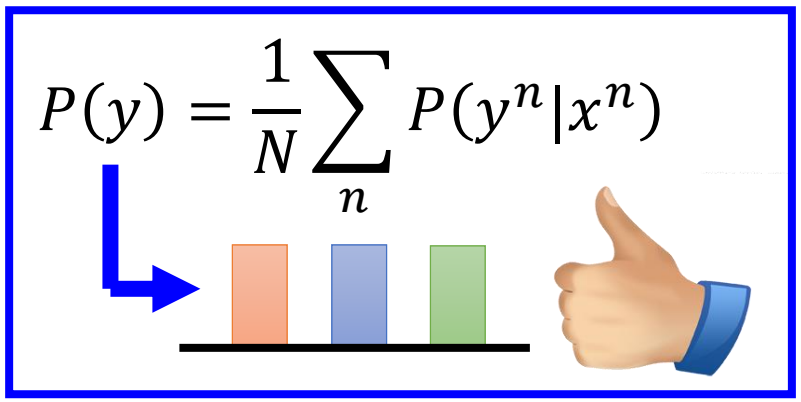
$$P(y) = \frac{1}{N}\sum_n P(y^n|x^n)$$

Uniform distribution means higher variety

Objective Evaluation





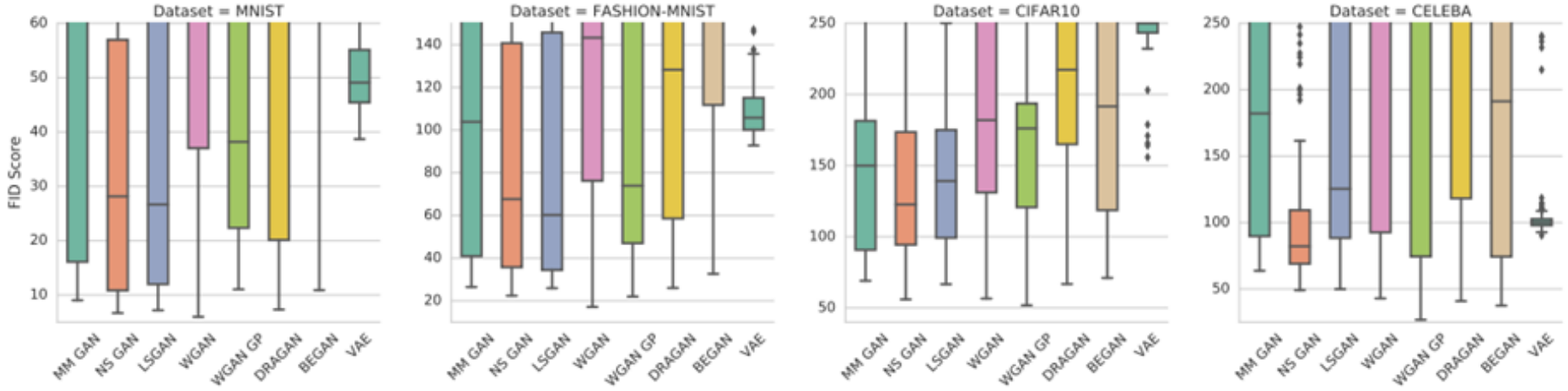


Inception Score

$$= \sum_{x}\sum_{y} P(y|x)logP(y|x)$$ Negative entropy of P(y|x)

$$-\sum_{y} P(y)logP(y)$$ Entropy of P(y)

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



Smaller is better

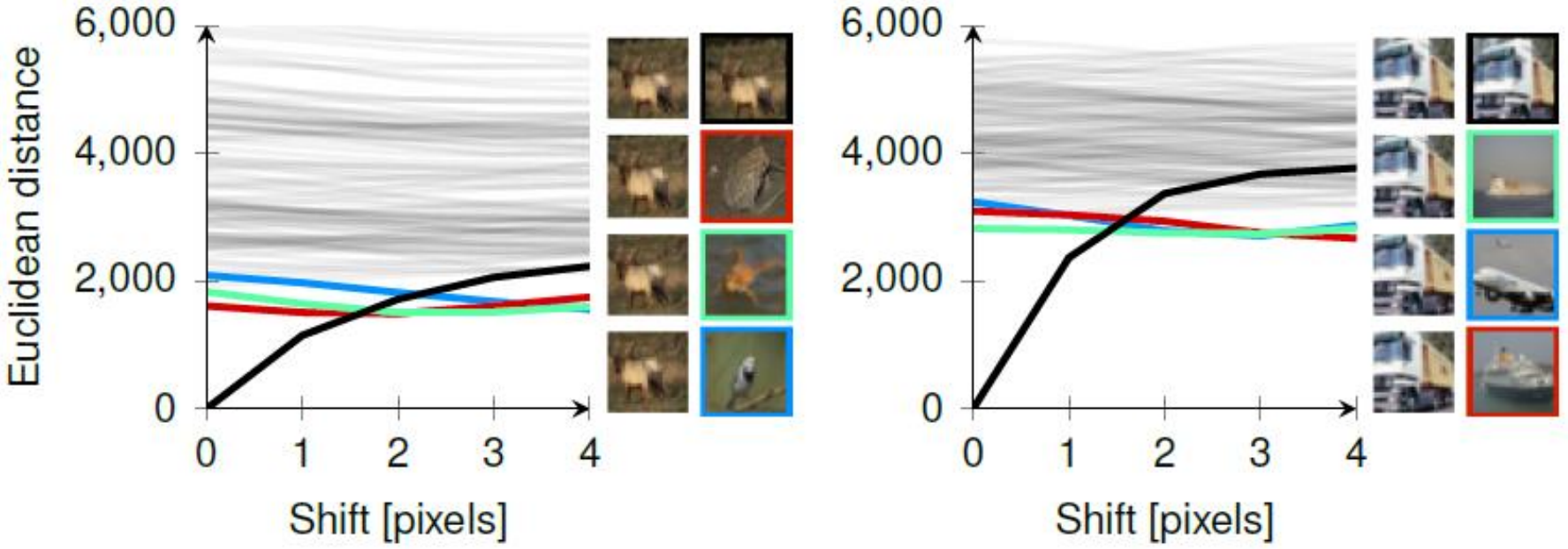
FIT:
https://arxiv.org/pdf/1706.08500.pdf

Mario Lucic, Karol Kurach, Marcin Michalski, Sylvain Gelly, Olivier Bousquet, "Are GANs Created Equal? A Large-Scale Study", arXiv, 2017

https://arxiv.org/pdf/1511.01844.pdf

We don’t want memory GAN.

- Using k-nearest neighbor to check whether the generator generates new objects



Missing Mode ?

Mode collapse is easy to detect.

?