

Generative Adversarial Network (GAN)

Restricted Boltzmann Machine:

[http://speech.ee.ntu.edu.tw/~tlkagk/courses/MLDS_2015_2/Lecture/RBM%20\(v2\).ecm.mp4/index.html](http://speech.ee.ntu.edu.tw/~tlkagk/courses/MLDS_2015_2/Lecture/RBM%20(v2).ecm.mp4/index.html)

Outlook:

Gibbs Sampling:

[http://speech.ee.ntu.edu.tw/~tlkagk/courses/MLDS_2015_2/Lecture/MRF%20\(v2\).ecm.mp4/index.html](http://speech.ee.ntu.edu.tw/~tlkagk/courses/MLDS_2015_2/Lecture/MRF%20(v2).ecm.mp4/index.html)

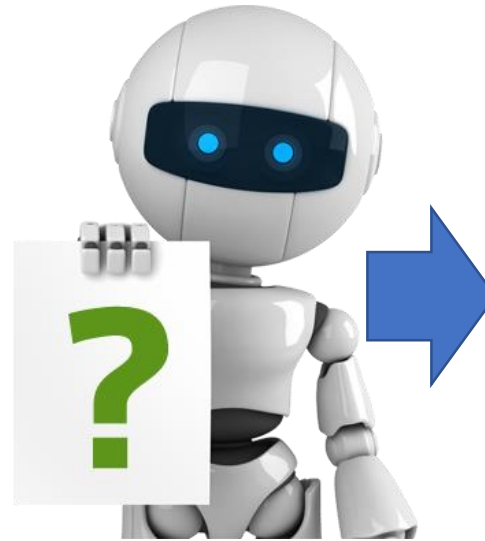
NIPS 2016 Tutorial: Generative Adversarial Networks

- Author: Ian Goodfellow
- Paper: <https://arxiv.org/abs/1701.00160>
- Video: <https://channel9.msdn.com/Events/Neural-Information-Processing-Systems-Conference/Neural-Information-Processing-Systems-Conference-NIPS-2016/Generative-Adversarial-Networks>

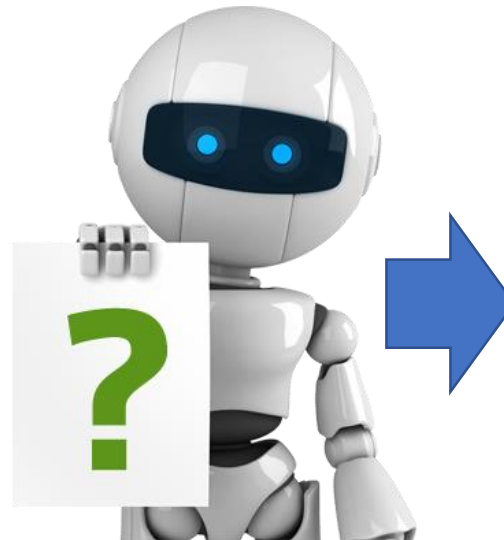
You can find tips for training GAN here:
<https://github.com/soumith/ganhacks>

Review

Generation

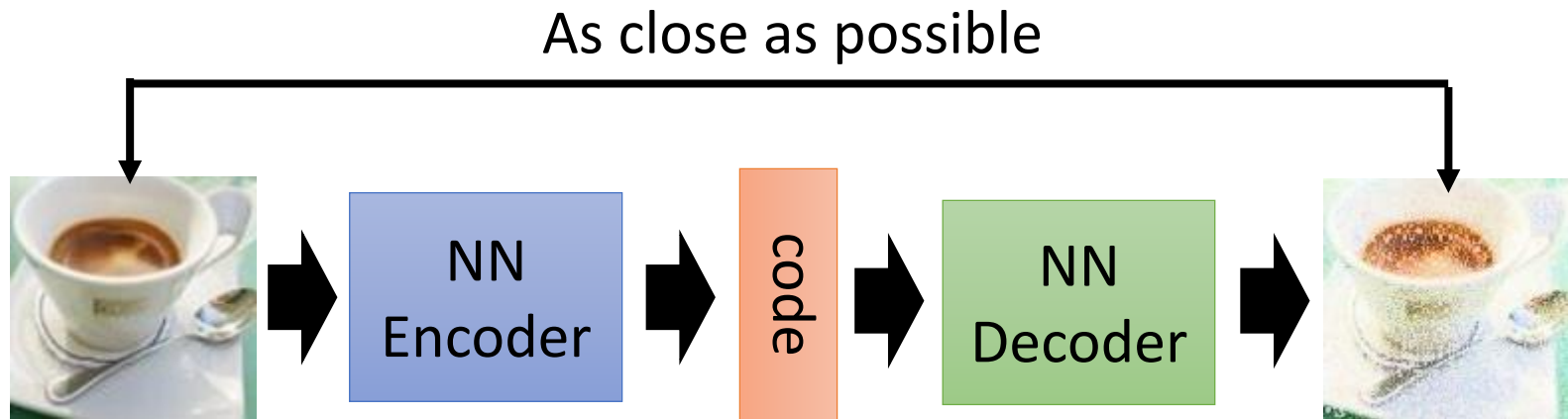


Writing
Poems?

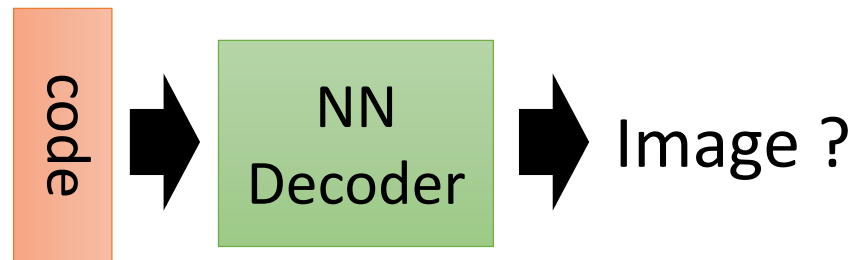


Drawing?

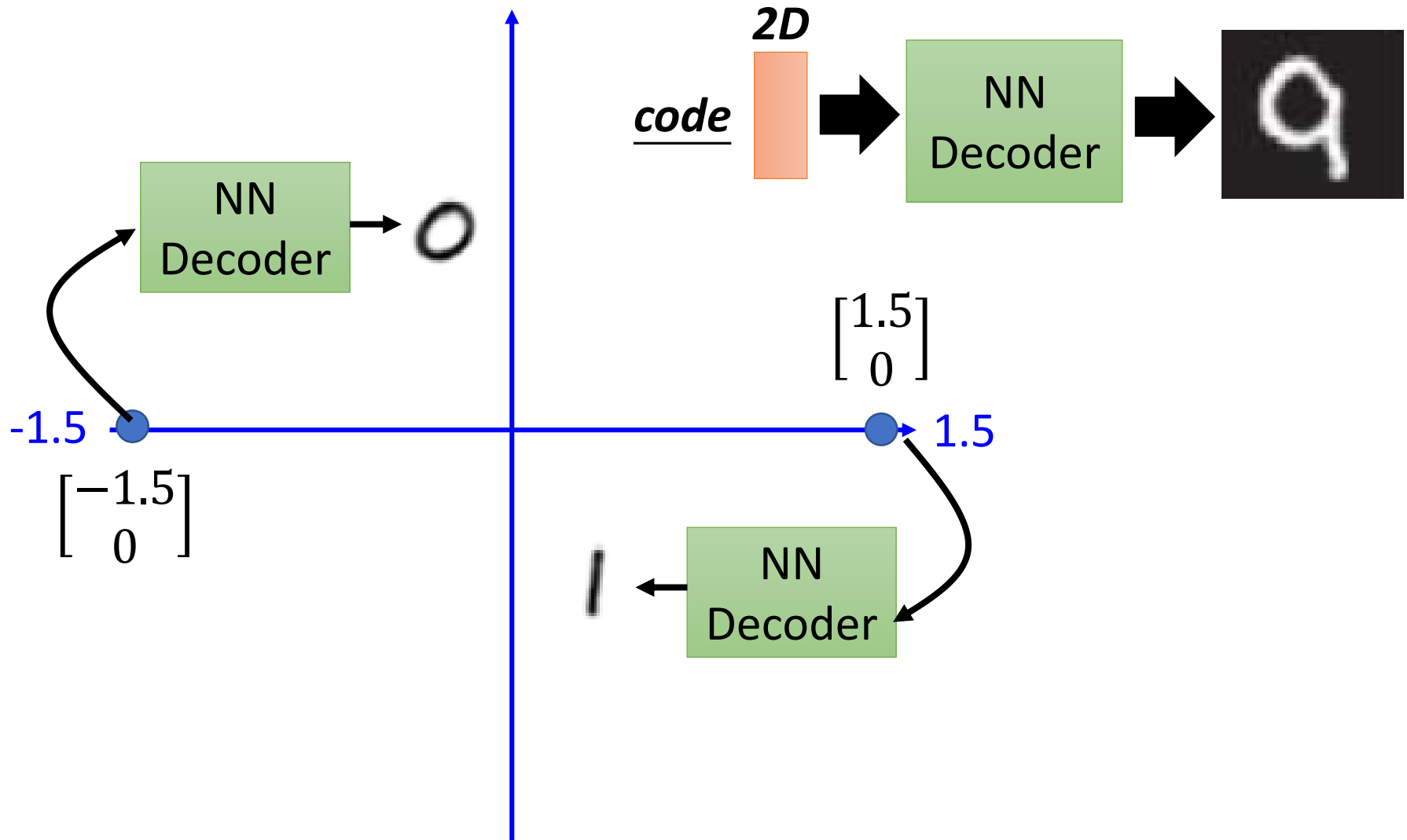
Review: Auto-encoder



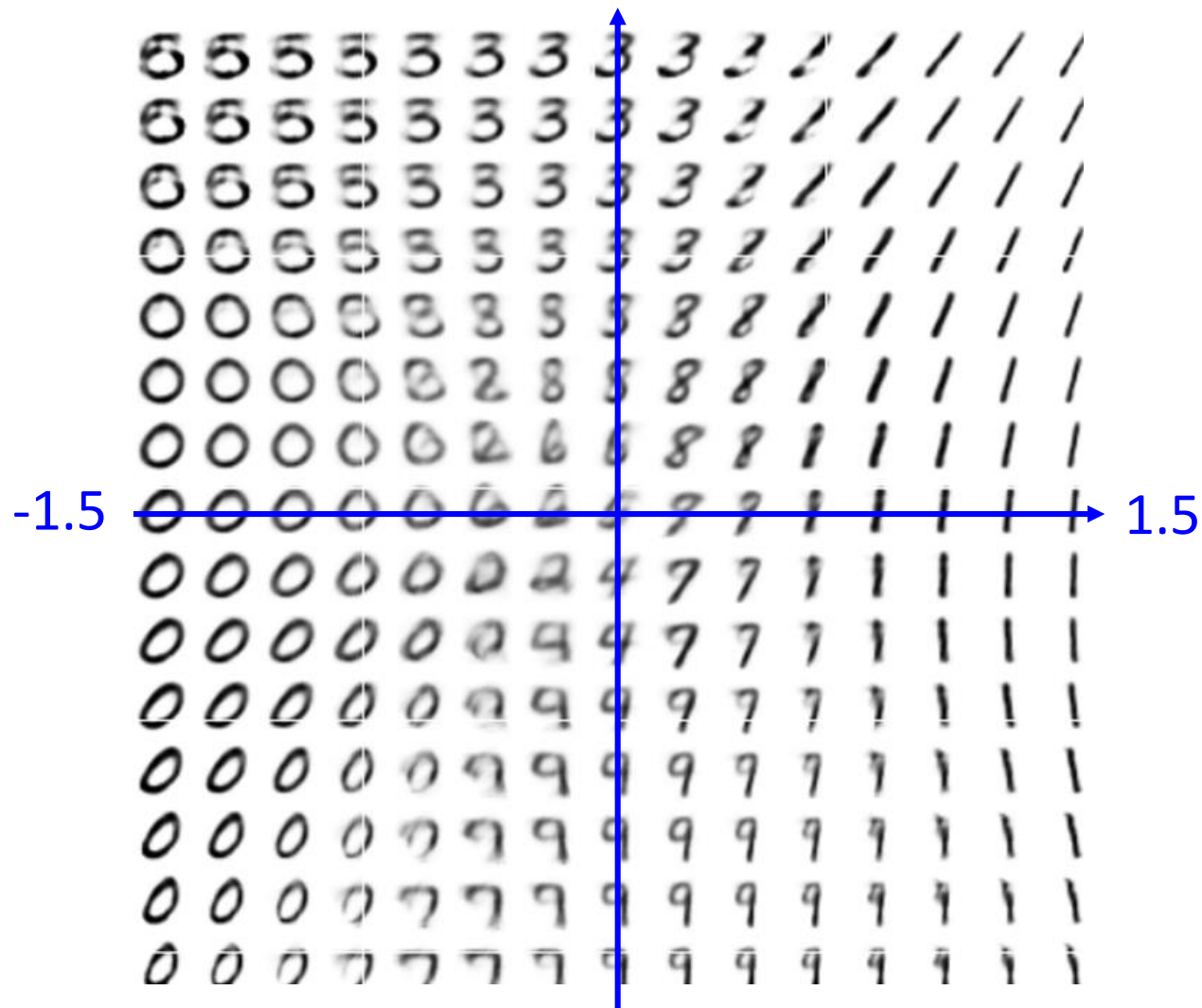
Randomly generate
a vector as code



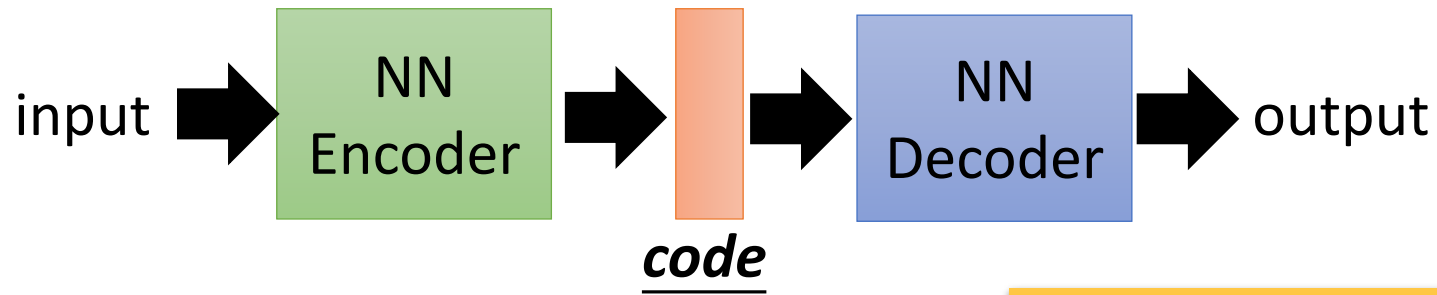
Review: Auto-encoder



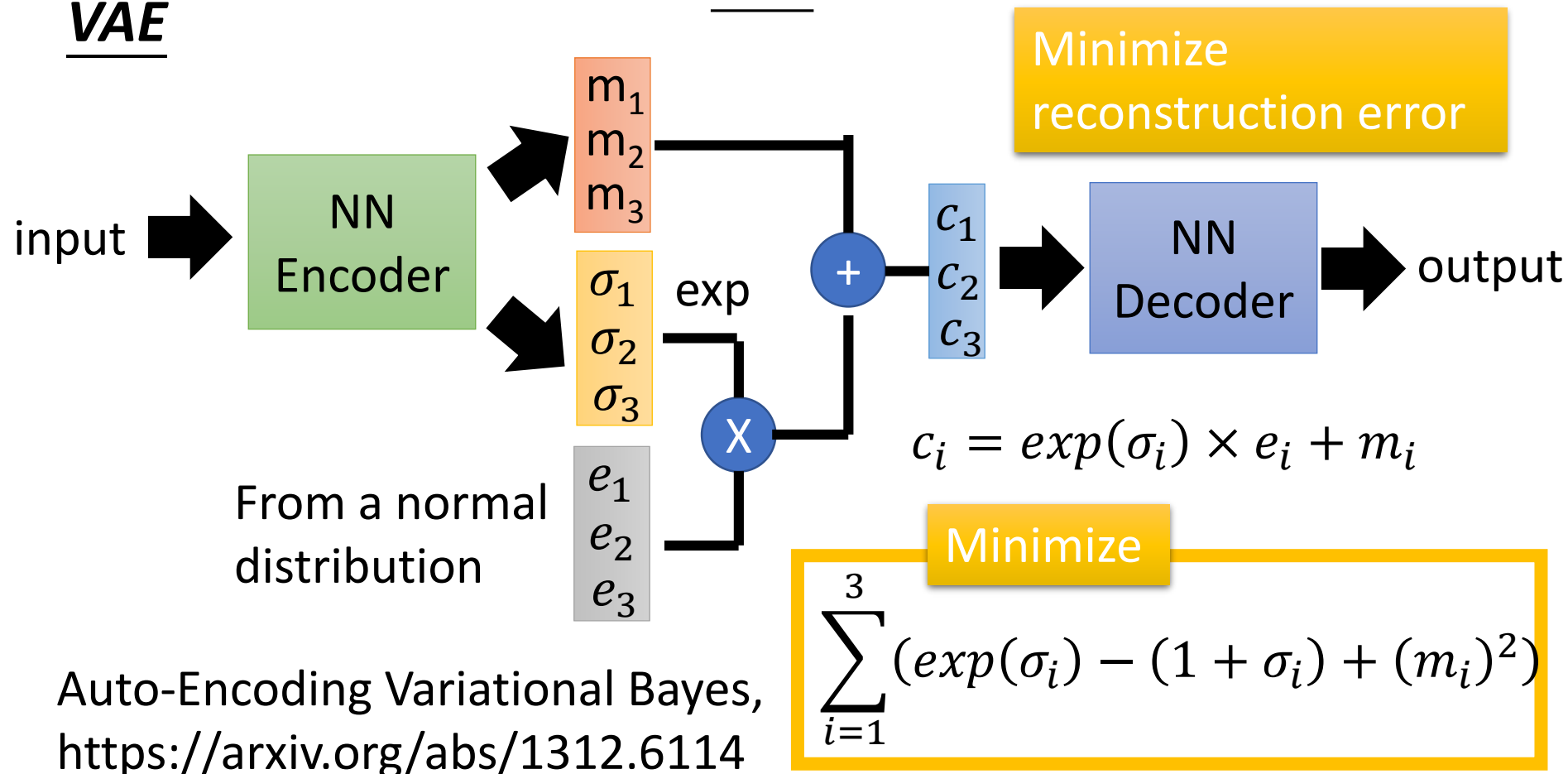
Review: Auto-encoder



Auto-encoder

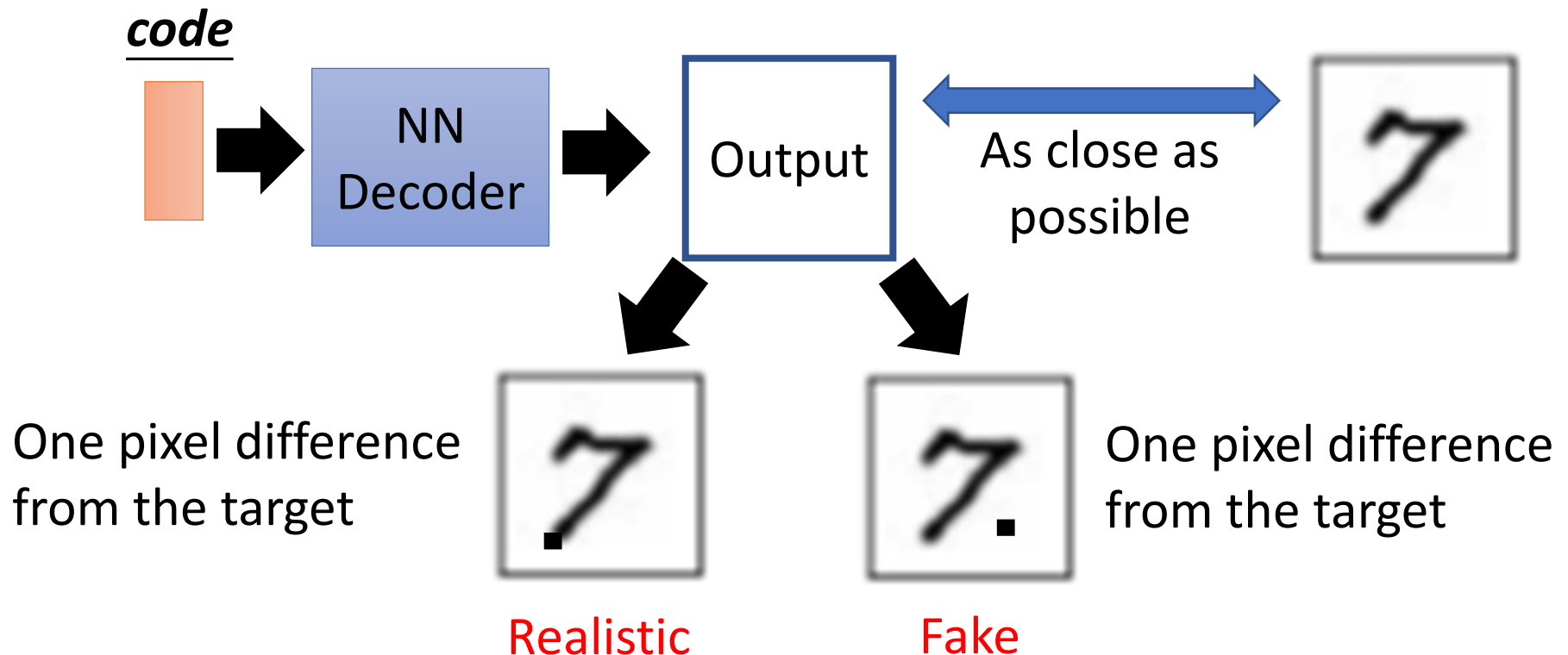


VAE

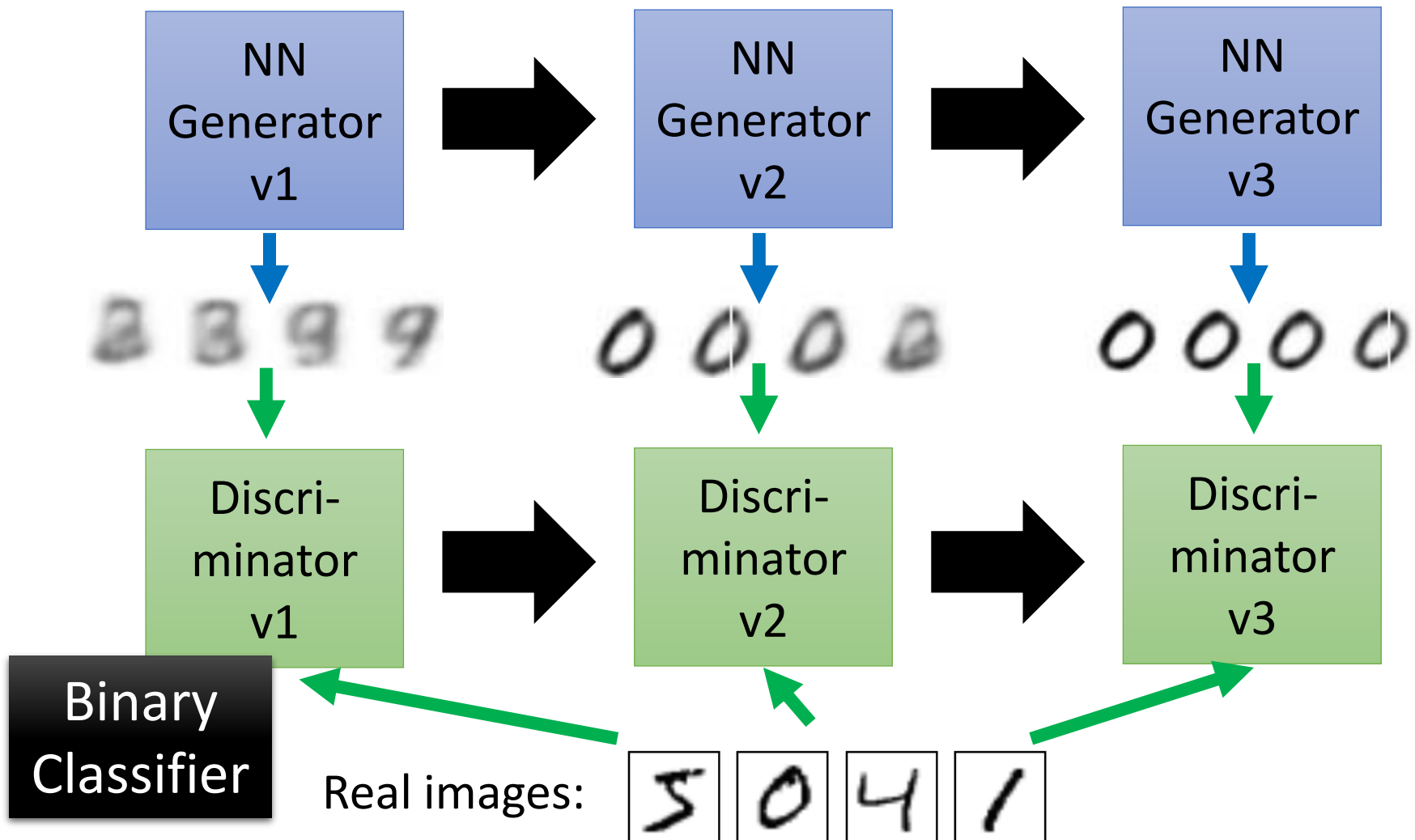


Problems of VAE

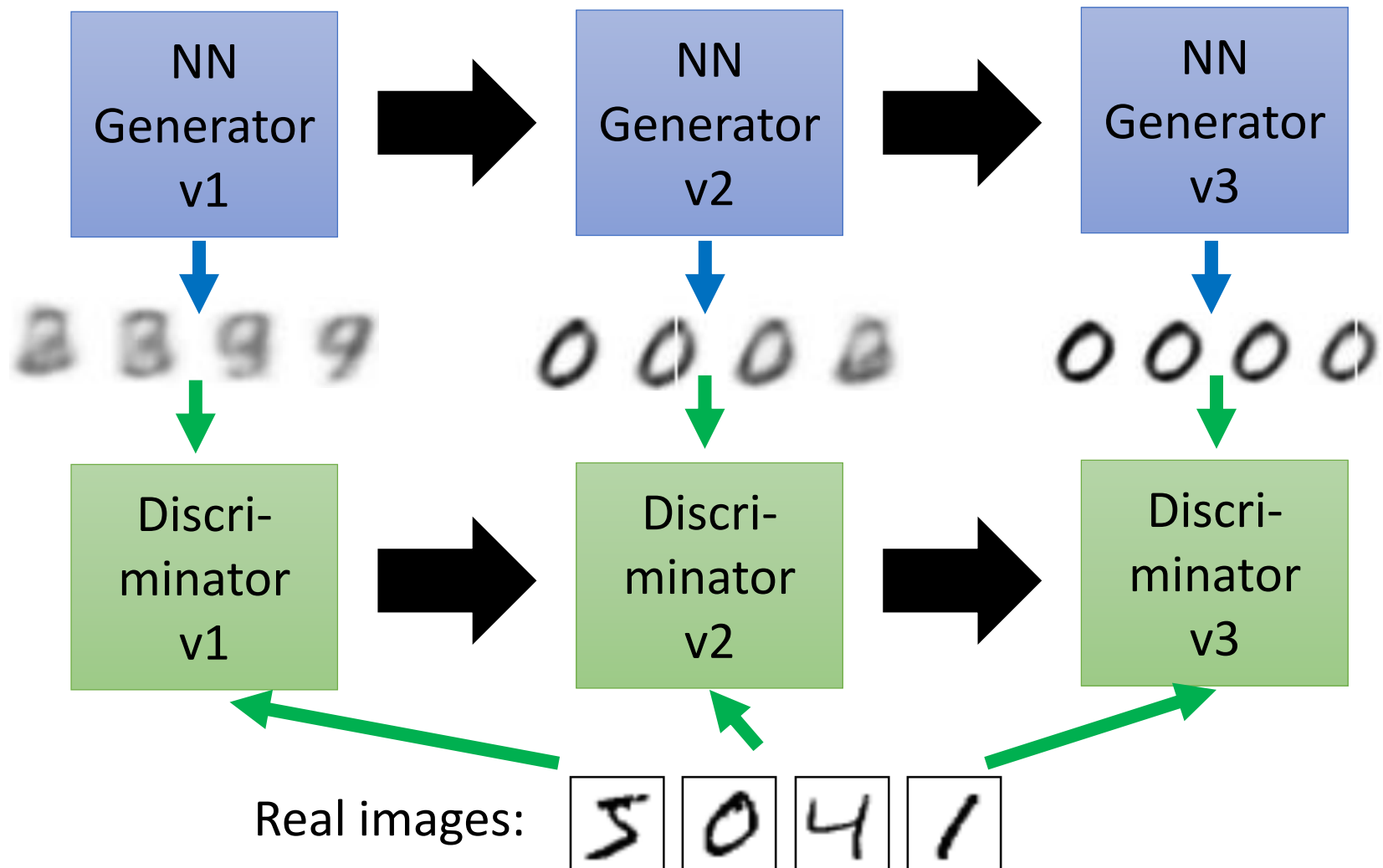
- It does not really try to simulate real images



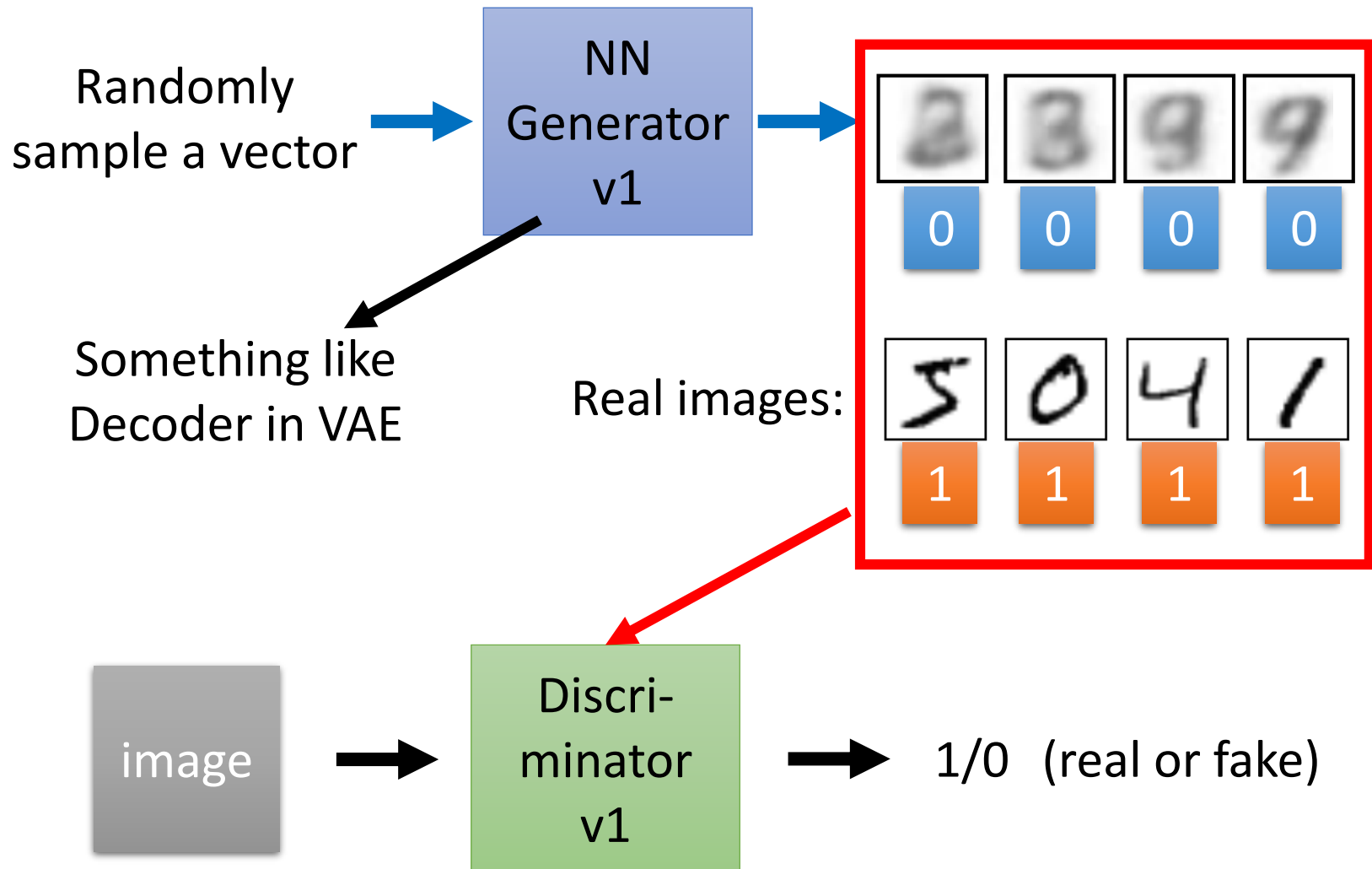
The evolution of generation



The evolution of generation



GAN - Discriminator



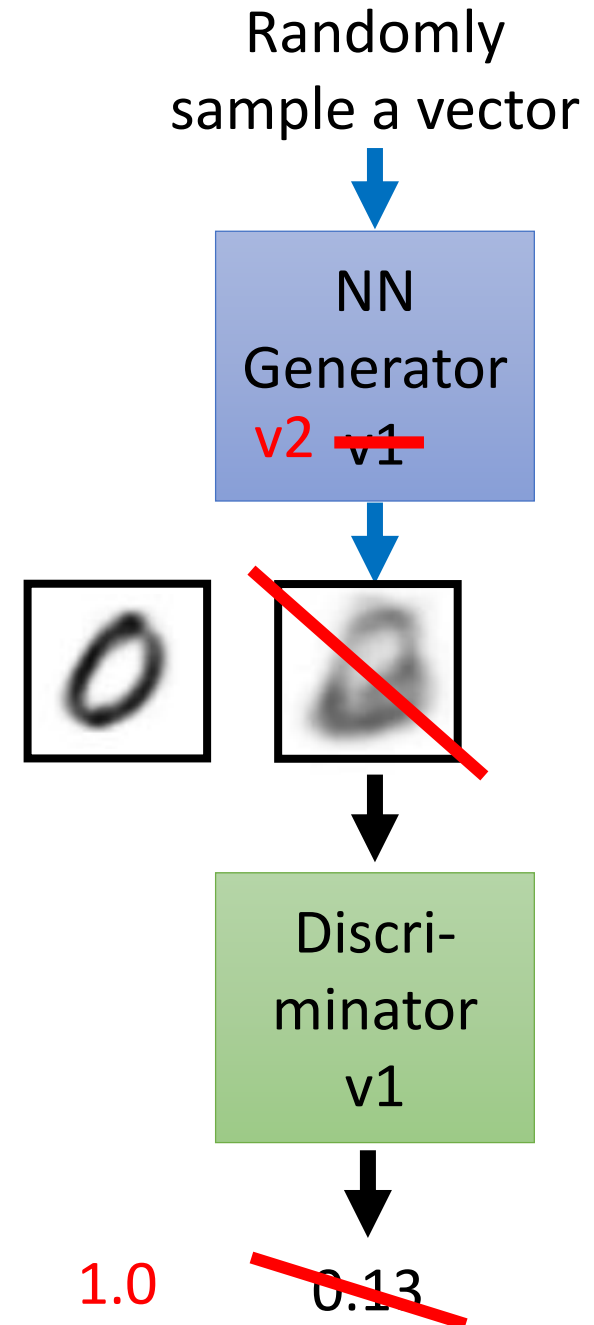
GAN - Generator

Updating the parameters of generator

➡ The output be classified as “real” (as close to 1 as possible)

Generator + Discriminator
= a network

Using gradient descent to update the parameters in the generator, but fix the discriminator



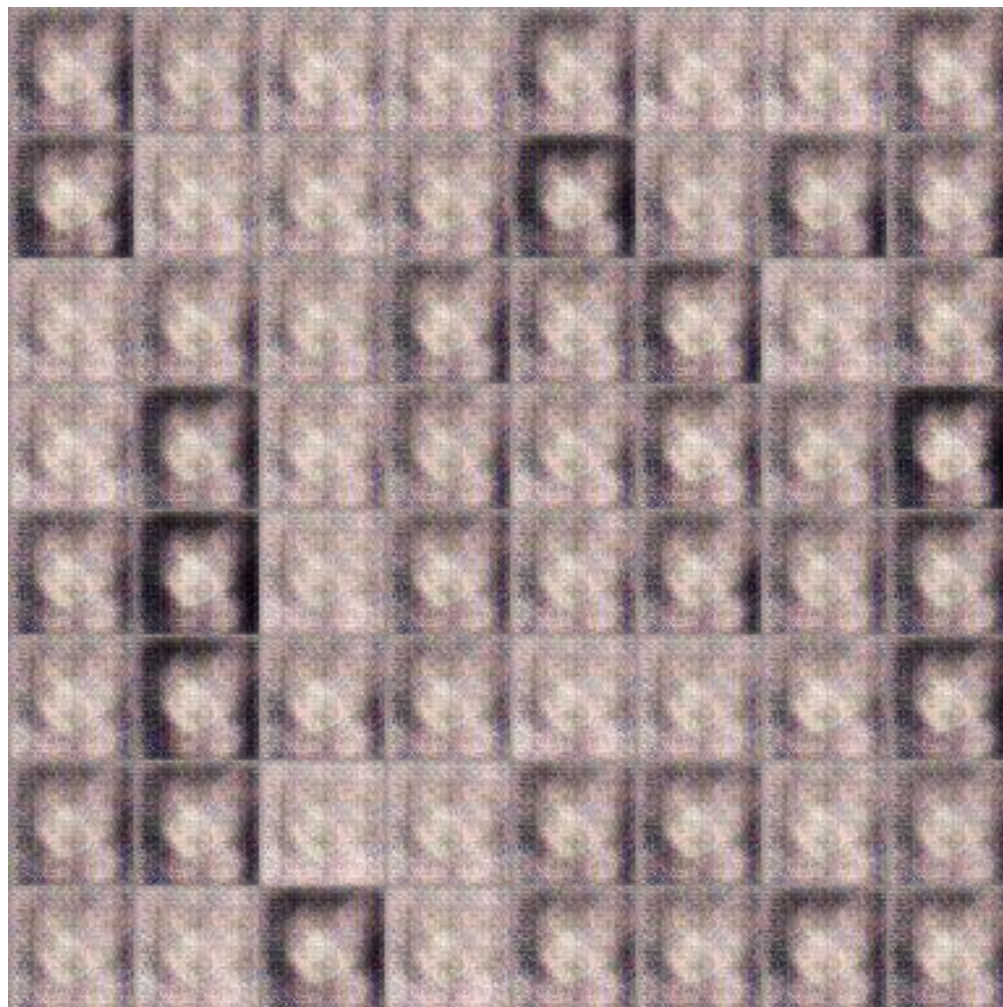
GAN – 二次元人物頭像鍊成



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

DCGAN: <https://github.com/carpedm20/DCGAN-tensorflow>

GAN – 二次元人物頭像鍊成



100 rounds

GAN – 二次元人物頭像鍊成



1000 rounds

GAN – 二次元人物頭像鍊成



2000 rounds

GAN – 二次元人物頭像鍊成



5000 rounds

GAN – 二次元人物頭像鍊成



10,000 rounds

GAN – 二次元人物頭像鍊成



20,000 rounds

GAN – 二次元人物頭像鍊成



50,000 rounds

Basic Idea of GAN

Maximum Likelihood Estimation

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

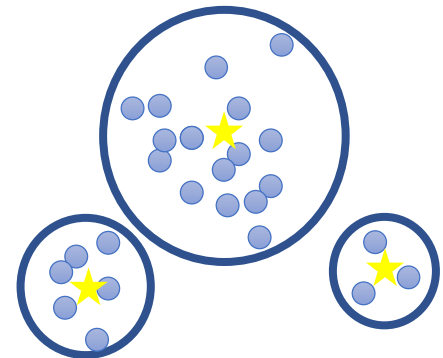
Sample $\{x^1, x^2, \dots, x^m\}$ from $P_{data}(x)$

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

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

$$= \arg \max_{\theta} \sum_{i=1}^m \log P_G(x^i; \theta) \quad \{x^1, x^2, \dots, x^m\} \text{ from } P_{data}(x)$$

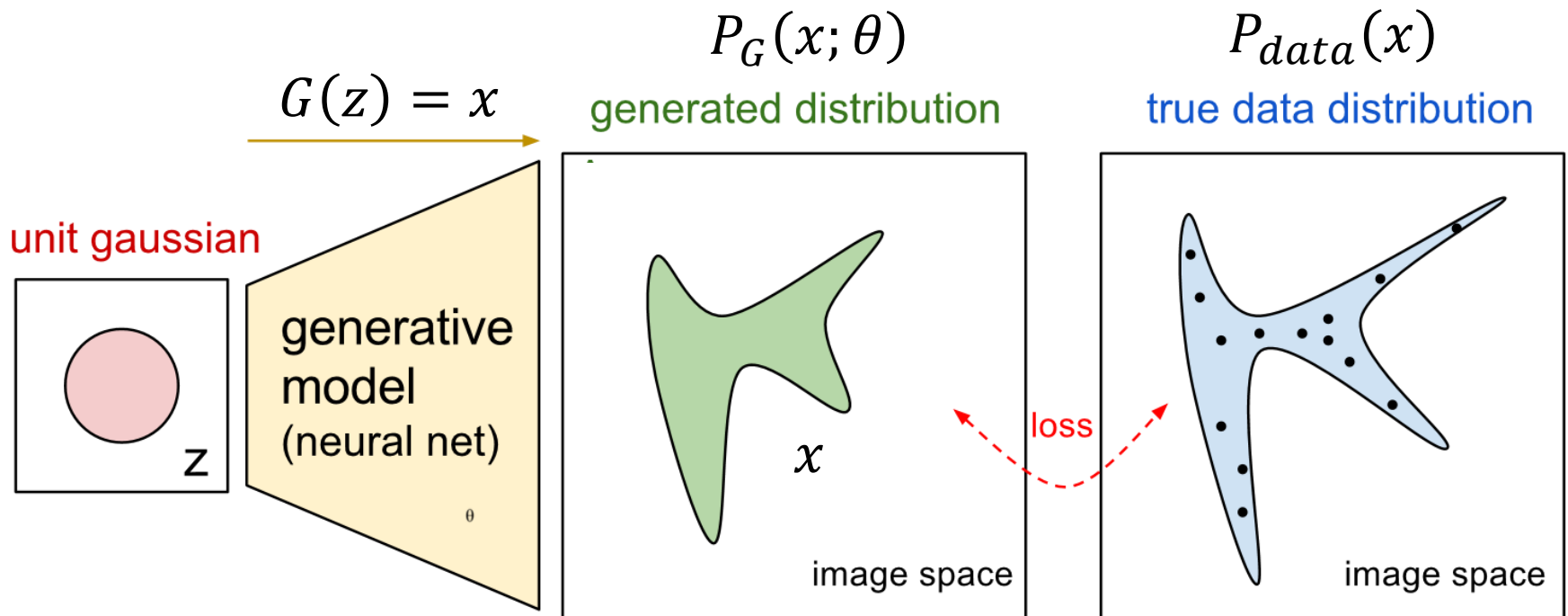
$$\approx \arg \max_{\theta} E_{x \sim P_{data}}[\log P_G(x; \theta)]$$

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

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

How to have a very
general $P_G(x; \theta)$?

Now $P_G(x; \theta)$ is a NN



$$P_G(x) = \int_z P_{prior}(z) I_{[G(z)=x]} dz$$

It is difficult to
compute the likelihood.

Basic Idea of GAN

- Generator G Hard to learn by maximum likelihood
 - G is a function, input z , output x
 - Given a prior distribution $P_{\text{prior}}(z)$, a probability distribution $P_G(x)$ is defined by function G
- Discriminator D
 - D is a function, input x , output scalar
 - Evaluate the “difference” between $P_G(x)$ and $P_{\text{data}}(x)$
- There is a function $V(G,D)$.

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

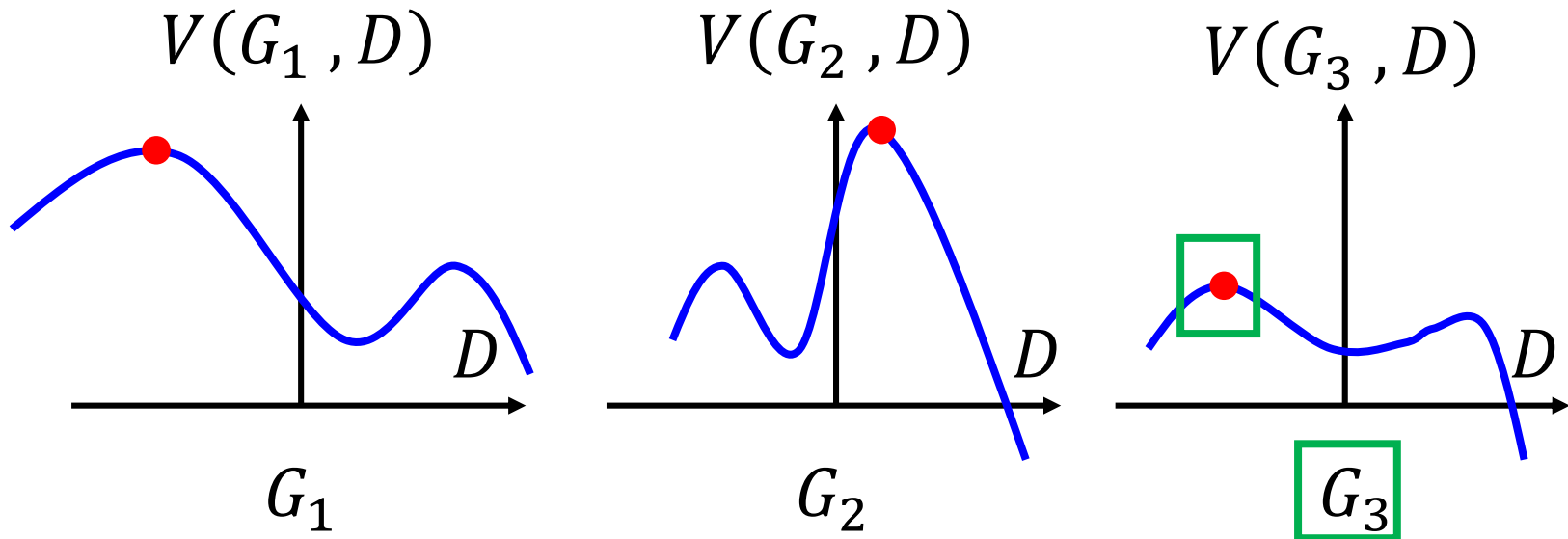
Basic Idea

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

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

Given a generator G , $\max_D V(G, D)$ evaluate the “difference” between P_G and P_{data}

Pick the G defining P_G most similar to P_{data}



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

- Given G , what is the optimal D^* maximizing

$$\begin{aligned} V &= E_{x \sim P_{data}}[\log D(x)] + E_{x \sim P_G}[\log(1 - D(x))] \\ &= \int_x P_{data}(x) \log D(x) dx + \int_x P_G(x) \log(1 - D(x)) dx \\ &= \int_x [P_{data}(x) \log D(x) + P_G(x) \log(1 - D(x))] dx \end{aligned}$$

Assume that $D(x)$ can have any value here

- Given x , the optimal D^* maximizing

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

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

- Given x , the optimal D^* maximizing

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

a
 D
 b
 D

- Find D^* maximizing: $f(D) = a \log(D) + b \log(1 - D)$

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

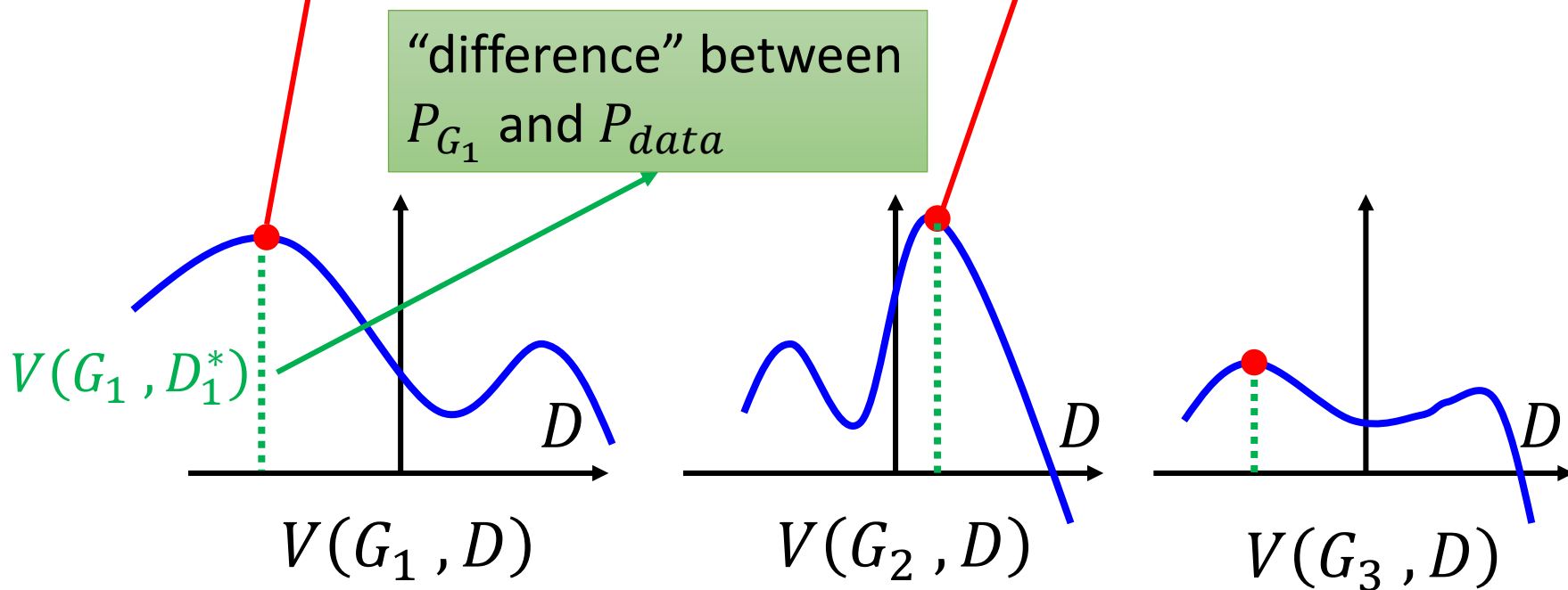
$$a \times \frac{1}{D^*} = b \times \frac{1}{1 - D^*} \quad a \times (1 - D^*) = b \times D^* \quad a - aD^* = bD^*$$

$$D^* = \frac{a}{a + b} \rightarrow D^*(x) = \frac{P_{data}(x)}{P_{data}(x) + P_G(x)} \quad 0 < \quad < 1$$

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

$$D_1^*(x) = \frac{P_{data}(x)}{P_{data}(x) + P_{G_1}(x)}$$

$$D_2^*(x) = \frac{P_{data}(x)}{P_{data}(x) + P_{G_2}(x)}$$



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

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

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

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

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

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

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

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

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

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

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

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

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

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

$$= -2\log 2 + \text{KL} \left(P_{data}(x) \parallel \frac{P_{data}(x) + P_G(x)}{2} \right) + \text{KL} \left(P_G(x) \parallel \frac{P_{data}(x) + P_G(x)}{2} \right)$$

$$= -2\log 2 + 2\text{JSD}(P_{data}(x) \parallel P_G(x)) \quad \text{Jensen-Shannon divergence}$$

In the end

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

- Generator G, Discriminator D
- Looking for G^* such that

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

- Given G, $\max_D V(G, D)$
 $= -2 \log 2 + 2 JSD(P_{data}(x) || P_G(x))$
 $0 < \quad < \log 2$
- What is the optimal G?

$$P_G(x) = P_{data}(x)$$

Algorithm

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

$$L(G)$$

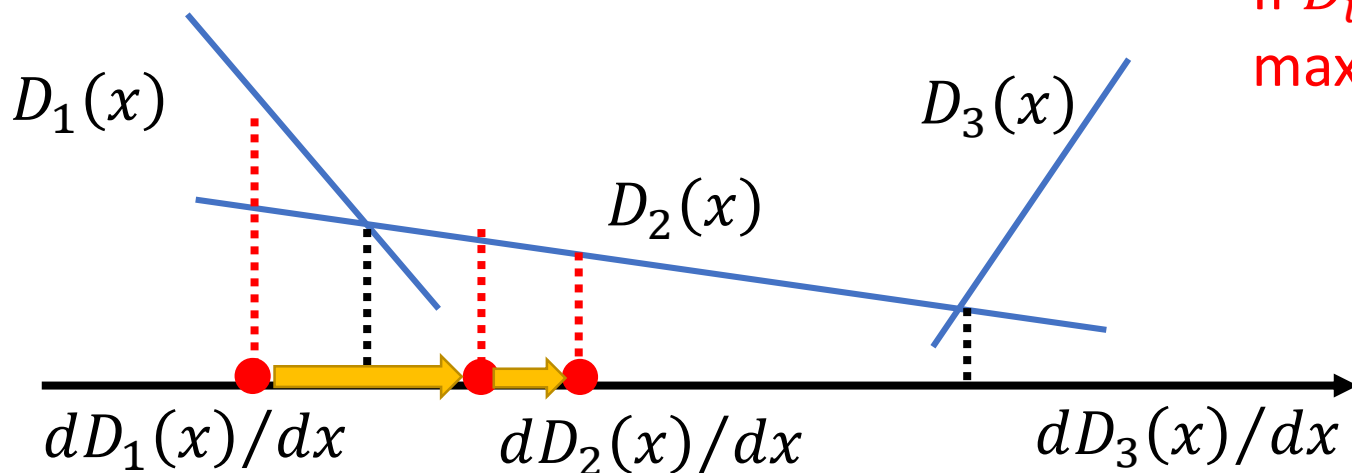
- To find the best G minimizing the loss function $L(G)$,

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

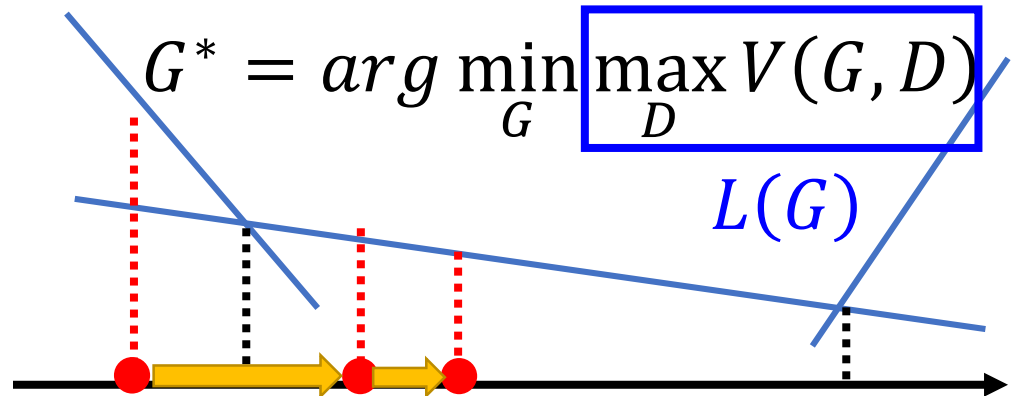
θ_G defines G

$$f(x) = \max\{D_1(x), D_2(x), D_3(x)\} \quad \frac{df(x)}{dx} = ? \quad dD_i(x)/dx$$

If $D_i(x)$ is the
max one



Algorithm



- 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 $P_{G_0}(x)$

- $\theta_G \leftarrow \theta_G - \eta \partial V(G, D_0^*) / \partial \theta_G \longrightarrow$ Obtain G_1 Decrease JS divergence(?)
- Find D_1^* maximizing $V(G_1, D)$

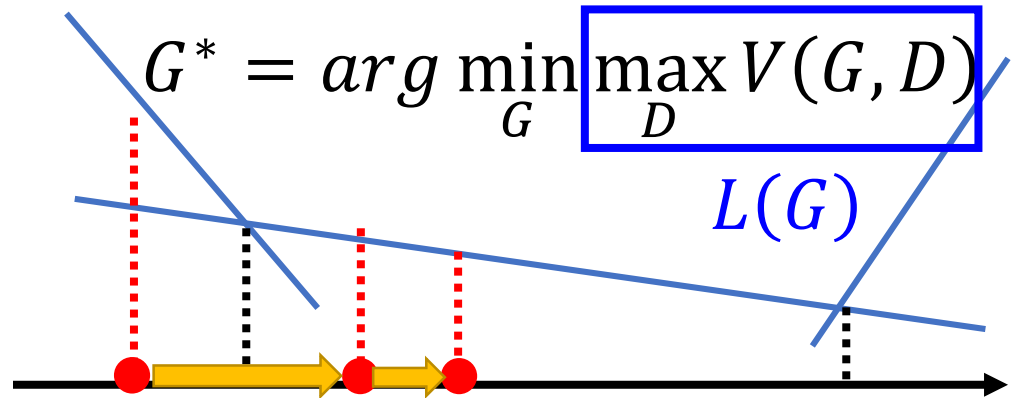
$V(G_1, D_1^*)$ is the JS divergence between $P_{data}(x)$ and $P_{G_1}(x)$

- $\theta_G \leftarrow \theta_G - \eta \partial V(G, D_1^*) / \partial \theta_G \longrightarrow$ Obtain G_2 Decrease JS divergence(?)
-

Algorithm

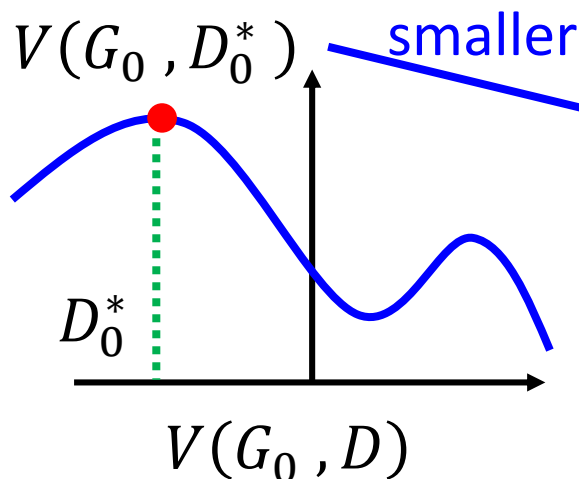
- Given G_0
- Find D_0^* maximizing $V(G_0, D)$

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

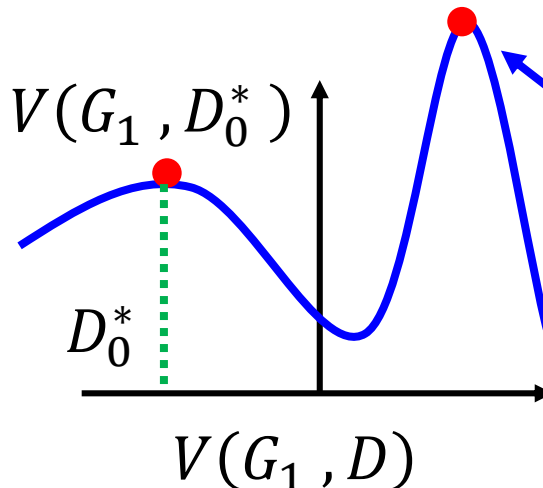


- $\theta_G \leftarrow \theta_G - \eta \partial V(G, D_0^*) / \partial \theta_G \longrightarrow$ Obtain G_1

Decrease JS divergence(?)



smaller



$V(G_1, D_1^*) \dots$

Assume $D_0^* \approx D_1^*$

Don't update G too much

In practice ...

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

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

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

Binary Classifier

Output is $D(x)$ Minimize Cross-entropy

If x is a positive example \Rightarrow Minimize $-\log D(x)$

If x is a negative example \Rightarrow Minimize $-\log(1-D(x))$

Binary Classifier

Output is $f(x)$ Minimize Cross-entropy

If x is a positive example \Rightarrow Minimize $-\log f(x)$

If x is a negative example \Rightarrow Minimize $-\log(1-f(x))$

D is a binary classifier (can be deep) with parameters θ_d

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

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

Minimize
$$L = \frac{1}{m} \sum_{i=1}^m \log D(x^i) + \frac{1}{m} \sum_{i=1}^m \log (1 - D(\tilde{x}^i))$$

||

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

Algorithm

Initialize θ_d for D and θ_g for G

Can only find
lower bound 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
 - $\tilde{V} = \frac{1}{m} \sum_{i=1}^m \log D(x^i) + \frac{1}{m} \sum_{i=1}^m \log (1 - D(\tilde{x}^i))$
 - $\theta_d \leftarrow \theta_d + \eta \nabla \tilde{V}(\theta_d)$

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
 - $\tilde{V} = \frac{1}{m} \sum_{i=1}^m \log D(x^i) + \frac{1}{m} \sum_{i=1}^m \log (1 - D(G(z^i)))$
 - $\theta_g \leftarrow \theta_g - \eta \nabla \tilde{V}(\theta_g)$

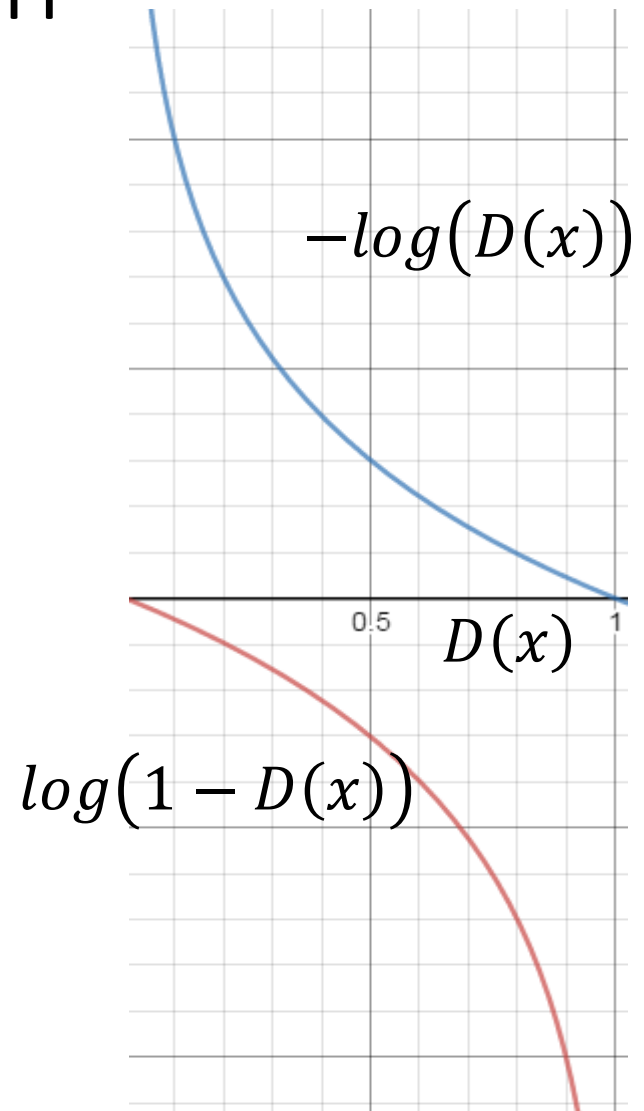
Objective Function for Generator in Real Implementation

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

Slow at the beginning

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

Real implementation:
label x from P_G as positive

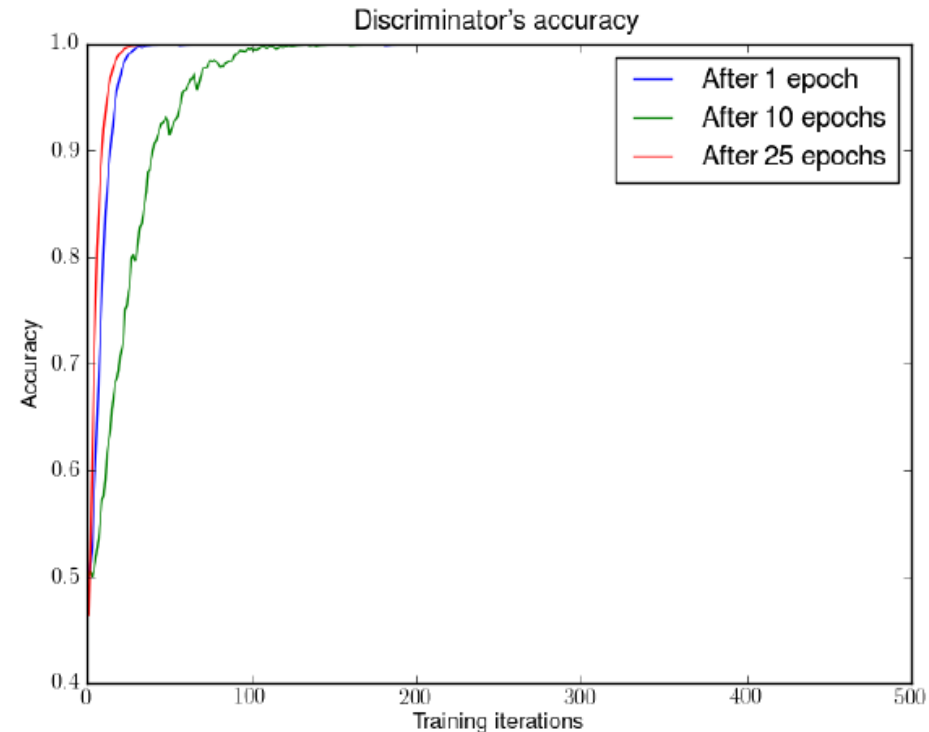
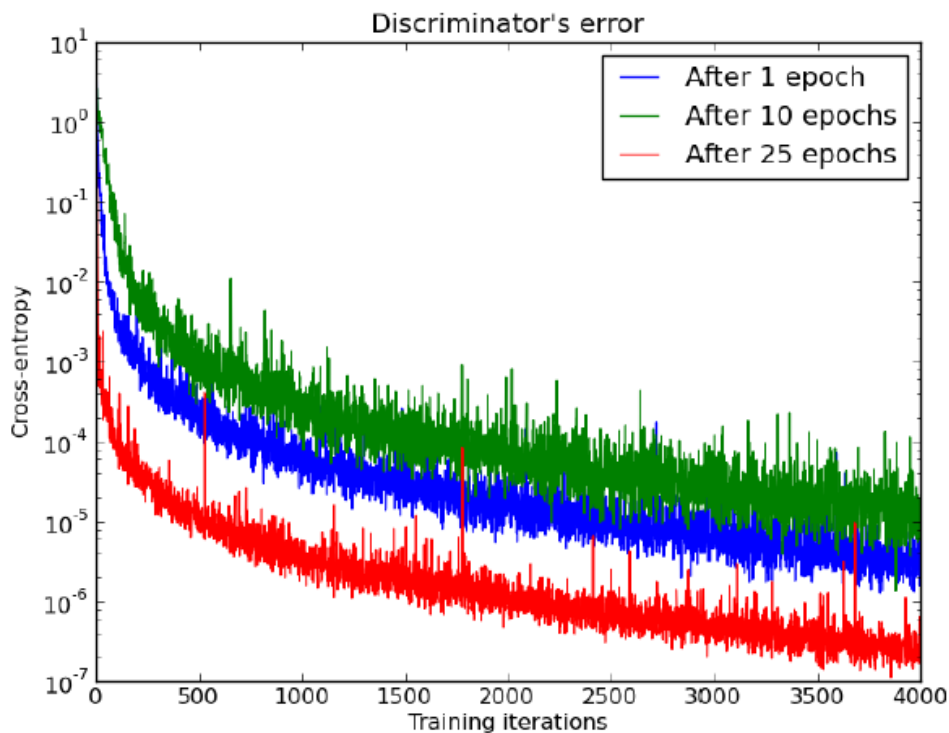


Demo

- The code used in demo from:
 - https://github.com/osh/KerasGAN/blob/master/MNIST_CNN_GAN_v2.ipynb

Issue about Evaluating the Divergence

Evaluating JS divergence

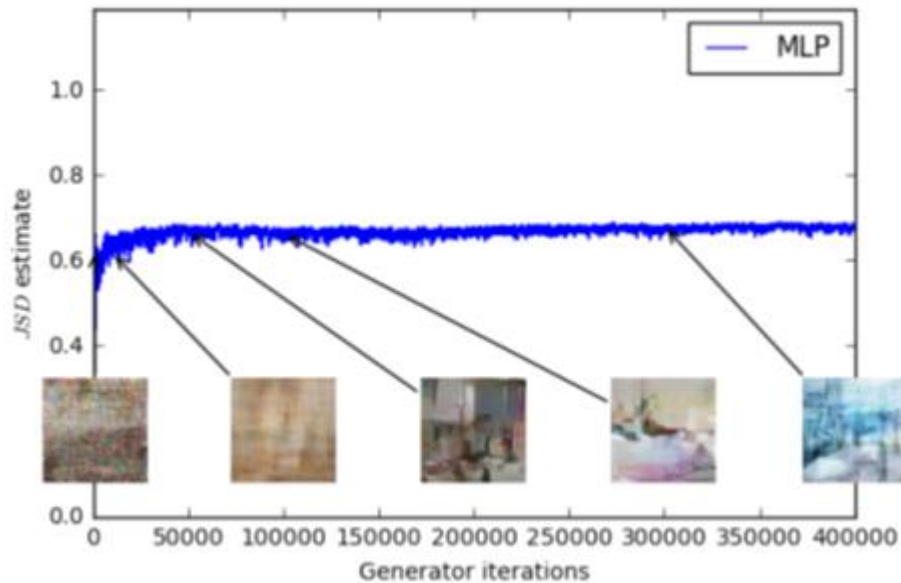


Martin Arjovsky, Léon Bottou, Towards Principled Methods for Training Generative Adversarial Networks, 2017, arXiv preprint

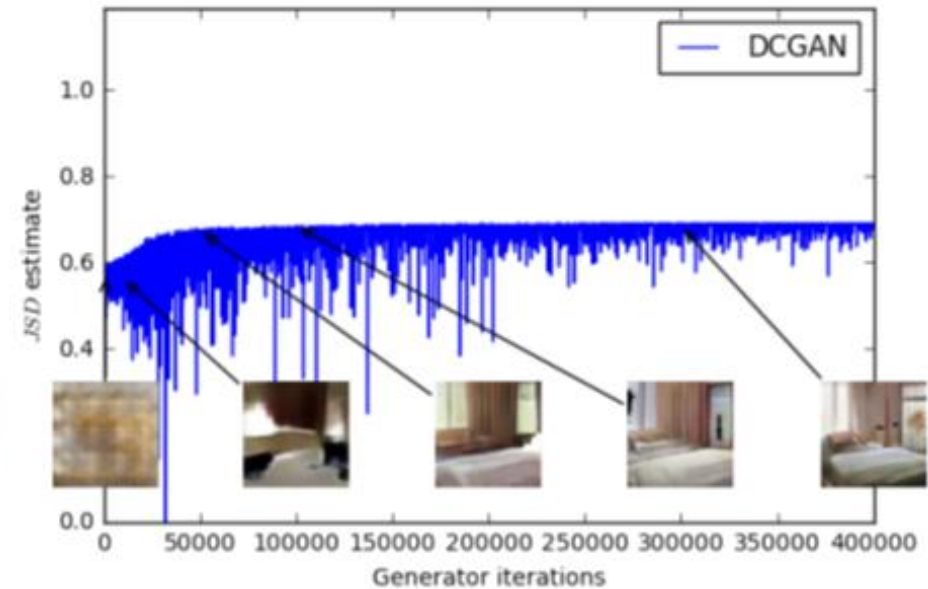
Evaluating JS divergence

<https://arxiv.org/abs/1701.07875>

- JS divergence estimated by discriminator telling little information



Weak Generator



Strong Generator

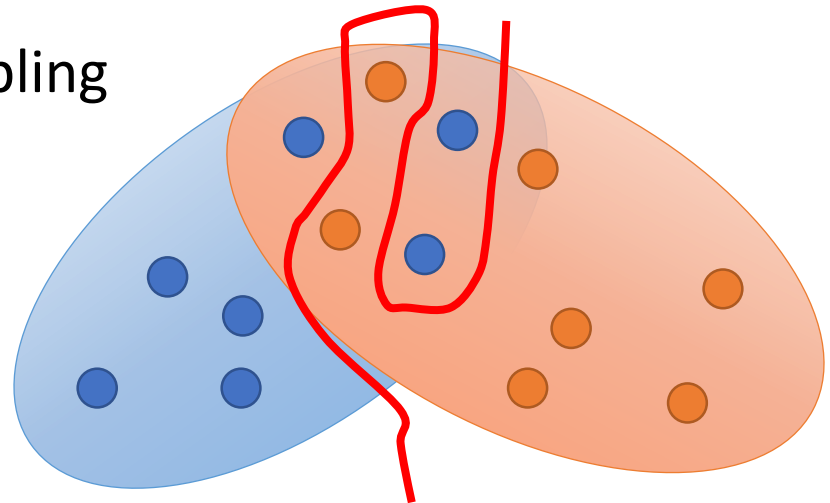
Discriminator

$$\begin{aligned} V &= E_{x \sim P_{data}} [\log D(x)] + E_{x \sim P_G} [\log(1 - D(x))] \\ &\approx \frac{1}{m} \sum_{i=1}^m \log D(x^i) + \frac{1}{m} \sum_{i=1}^m \log(1 - D(\tilde{x}^i)) \\ \max_D V(G, D) &= -2 \log 2 + 2 \underbrace{JSD(P_{data}(x) || P_G(x))}_{\log 2} = 0 \end{aligned}$$

Reason 1. Approximate by sampling

Weaken your
discriminator?

Can weak discriminator
compute JS divergence?



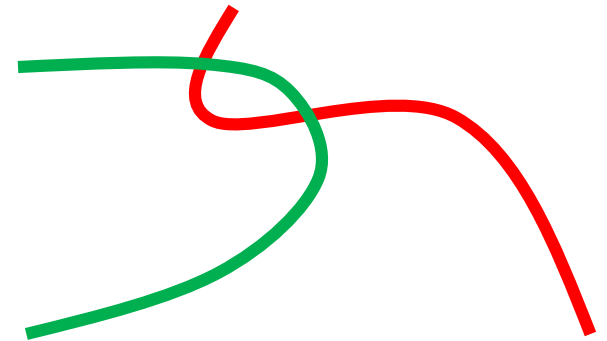
Discriminator

$$\begin{aligned} V &= E_{x \sim P_{data}} [\log D(x)] + E_{x \sim P_G} [\log(1 - D(x))] \\ &\approx \frac{1}{m} \sum_{i=1}^m \log D(x^i) + \frac{1}{m} \sum_{i=1}^m \log(1 - D(\tilde{x}^i)) \\ \max_D V(G, D) &= -2 \log 2 + 2 \underbrace{JSD(P_{data}(x) || P_G(x))}_{\log 2} = 0 \end{aligned}$$

Reason 2. the nature of data

Both $P_{data}(x)$ and $P_G(x)$ are low-dim manifold in high-dim space

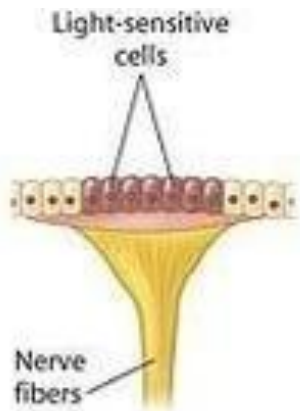
Usually they do not have any overlap



Evaluation

<http://www.guokr.com/post/773890/>

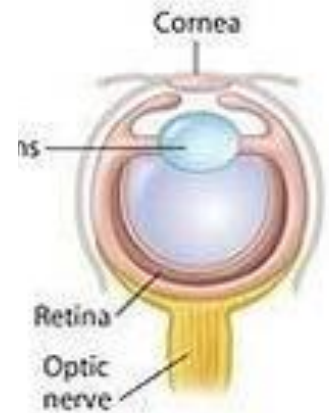
Better



Patch of light-sensitive cells



Limpet

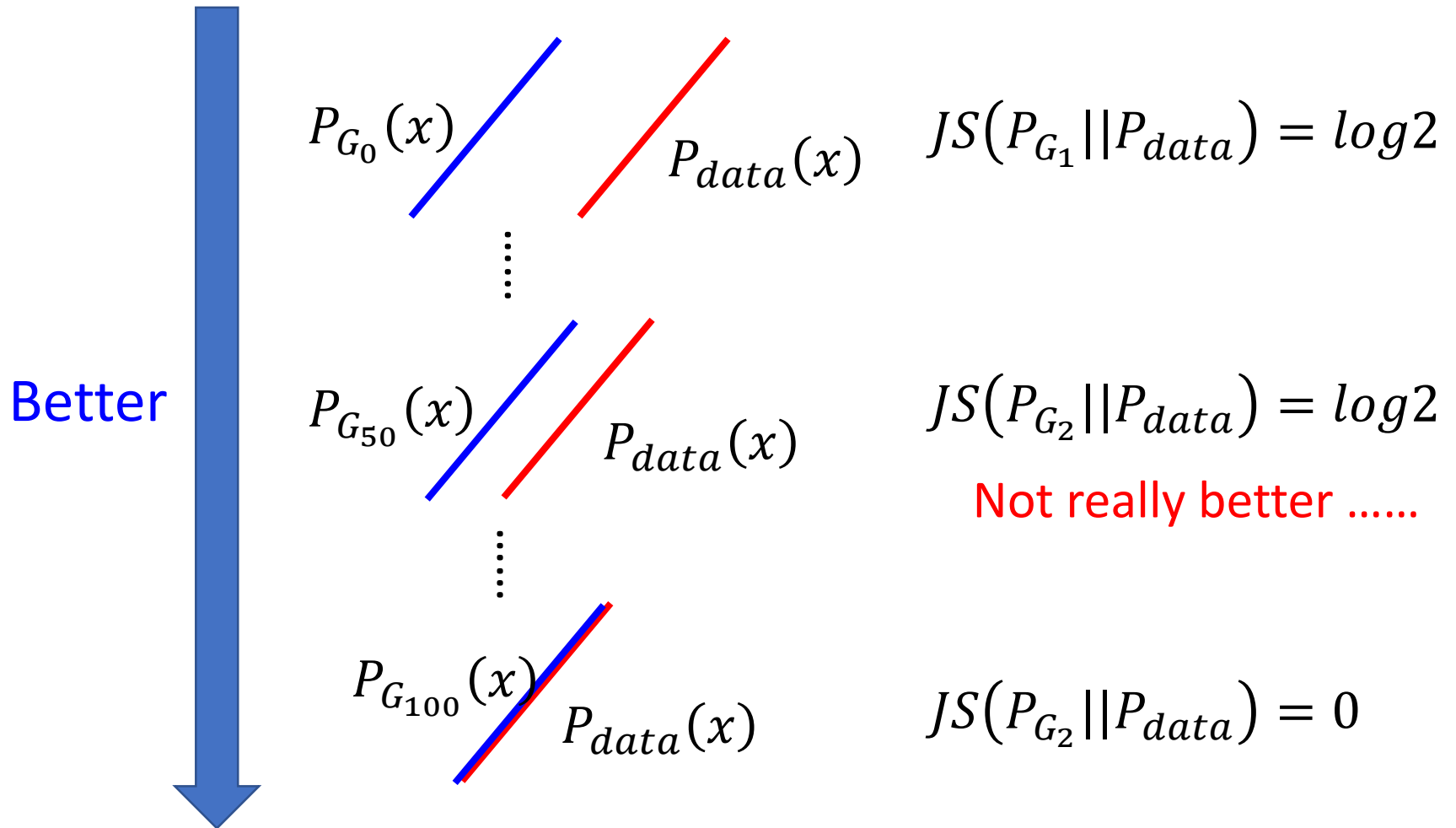


Complex camera-type eye



Squid

Evaluation



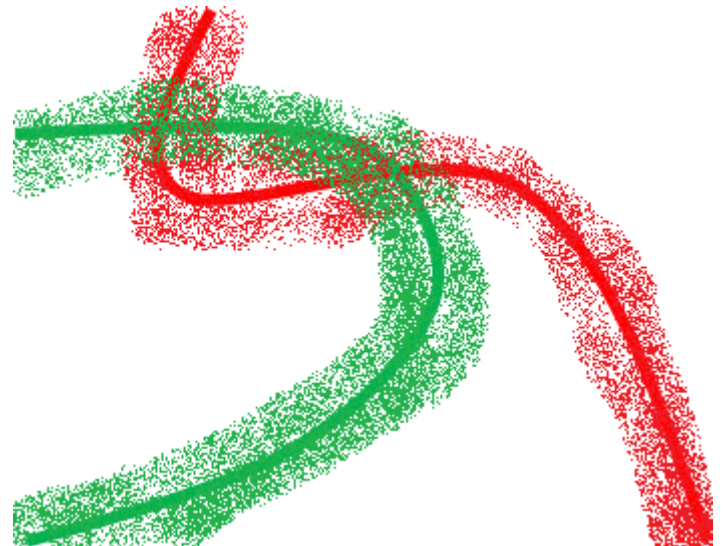
Add Noise

- Add some artificial noise to the inputs of discriminator
- Make the labels noisy for the discriminator

Discriminator cannot perfectly separate real and generated data

$P_{data}(x)$ and $P_G(x)$
have some overlap

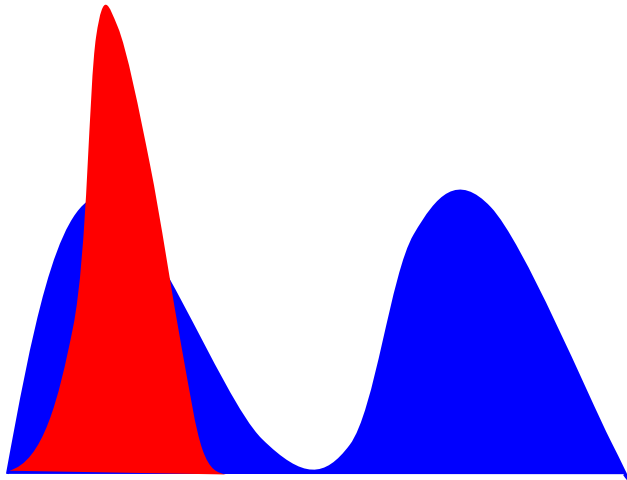
Noises decay over time



Mode Collapse

Mode Collapse

Generated
Distribution

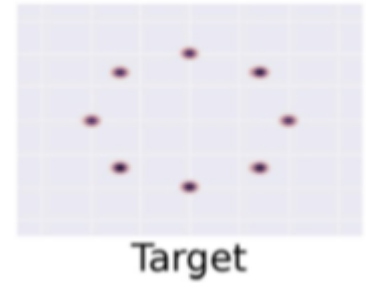


Data
Distribution

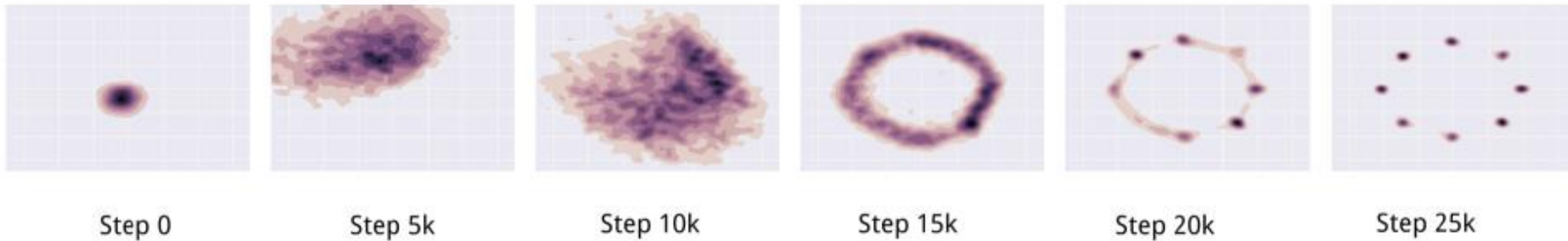


Mode Collapse

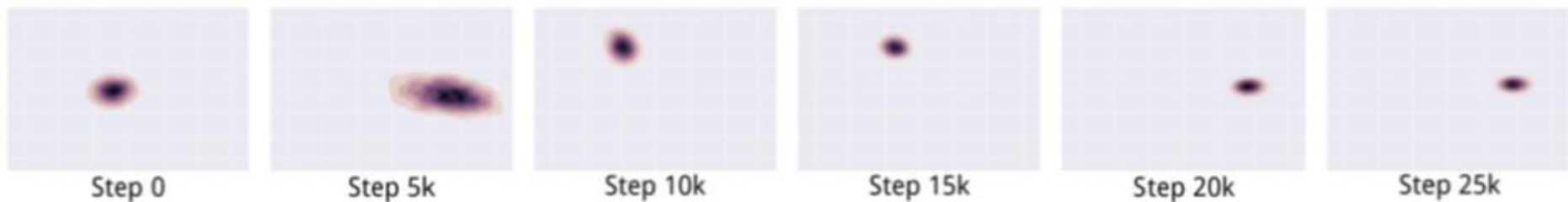
P_{data}



What we want ...



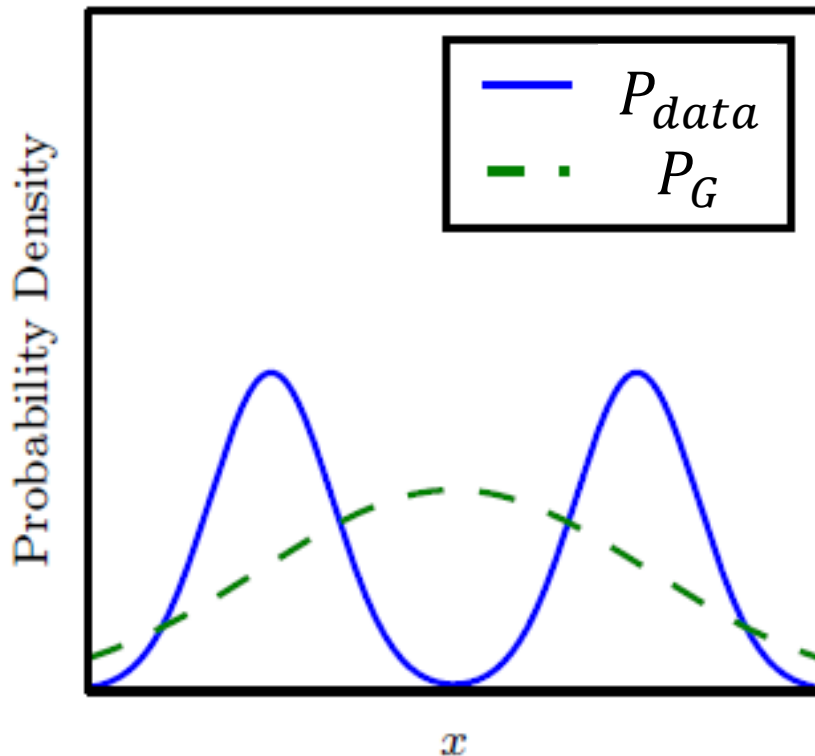
In reality ...



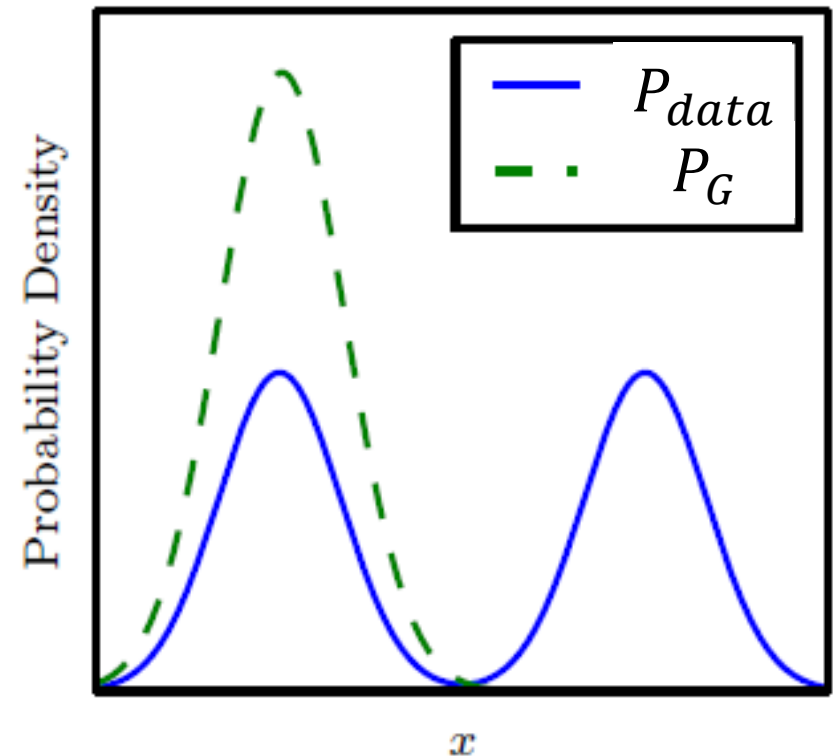
Flaw in Optimization?

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

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



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



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

This may not be the reason (based on Ian Goodfellow's tutorial)

So many GANs

Modifying the Optimization of GAN

fGAN

WGAN

Least-square GAN

Loss Sensitive GAN

Energy-based GAN

Boundary-seeking GAN

Unroll GAN

.....

Different Structure from the Original GAN

Conditional GAN

Semi-supervised GAN

InfoGAN

BiGAN

Cycle GAN

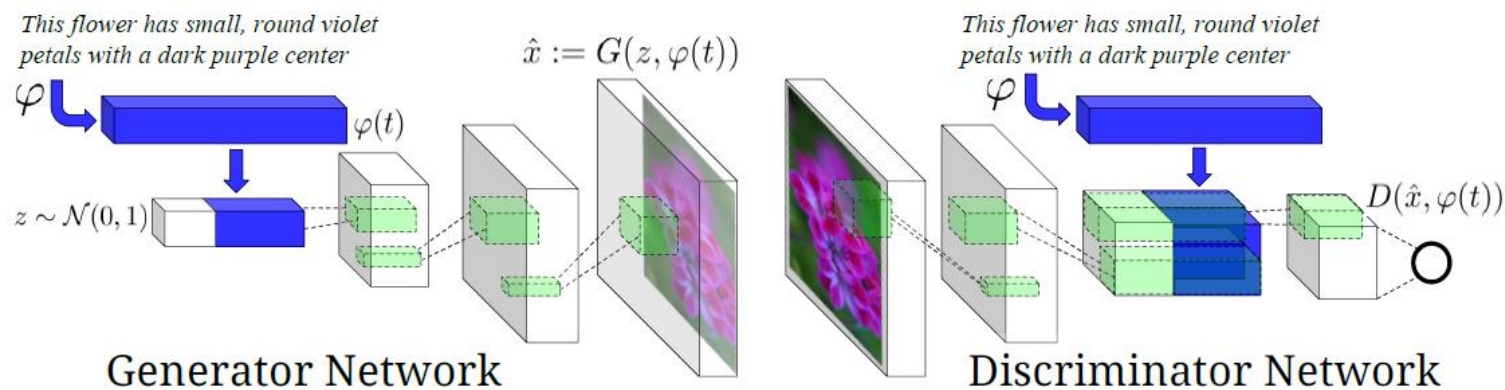
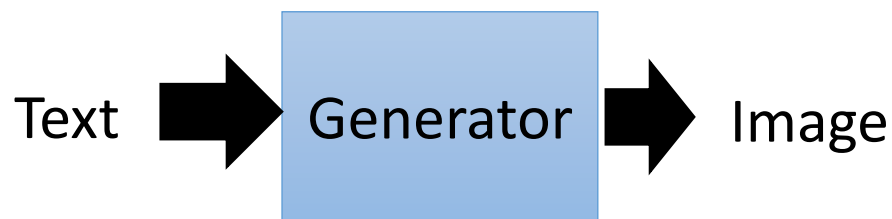
Disco GAN

VAE-GAN

.....

Conditional GAN

Motivation



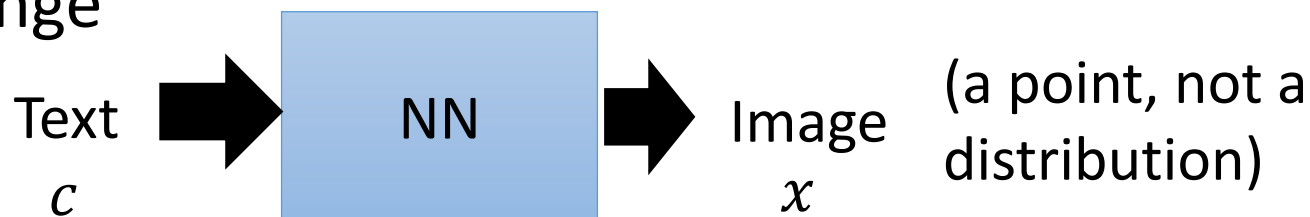
Scott Reed, Zeynep Akata, Xincheng Yan, Lajanugen Logeswaran, Bernt Schiele, Honglak Lee, "Generative Adversarial Text-to-Image Synthesis", ICML 2016

Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaolei Huang, Xiaogang Wang, Dimitris Metaxas, "StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks", arXiv preprint, 2016

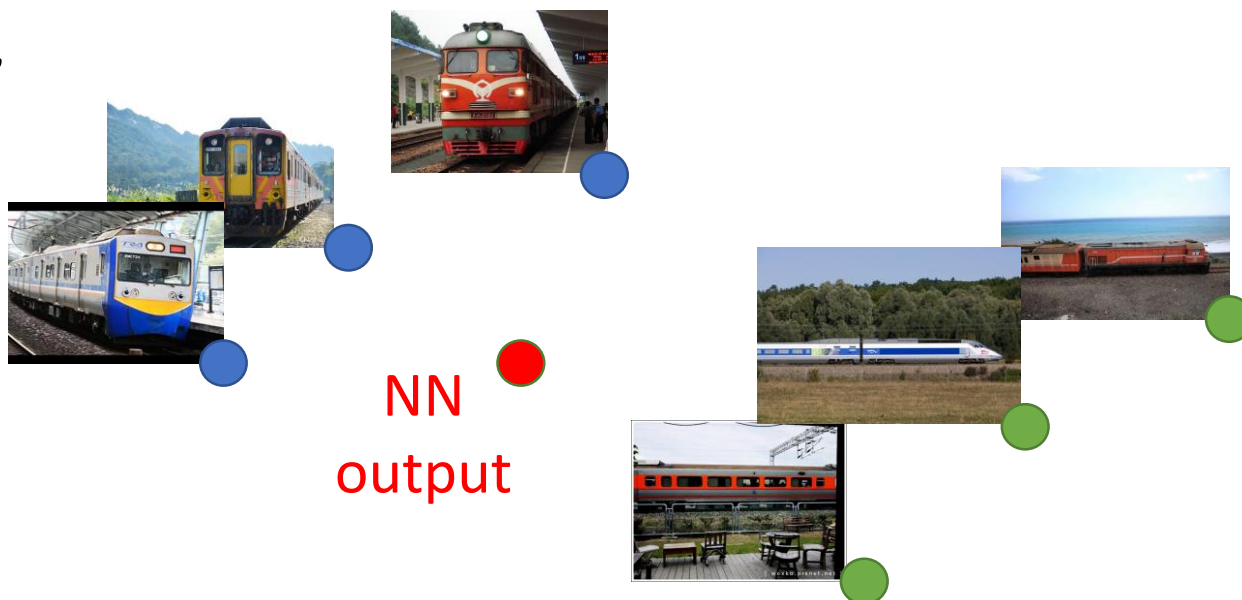
Scott Reed, Zeynep Akata, Santosh Mohan, Samuel Tenka, Bernt Schiele, Honglak Lee, "Learning What and Where to Draw", NIPS 2016

Motivation

- Challenge

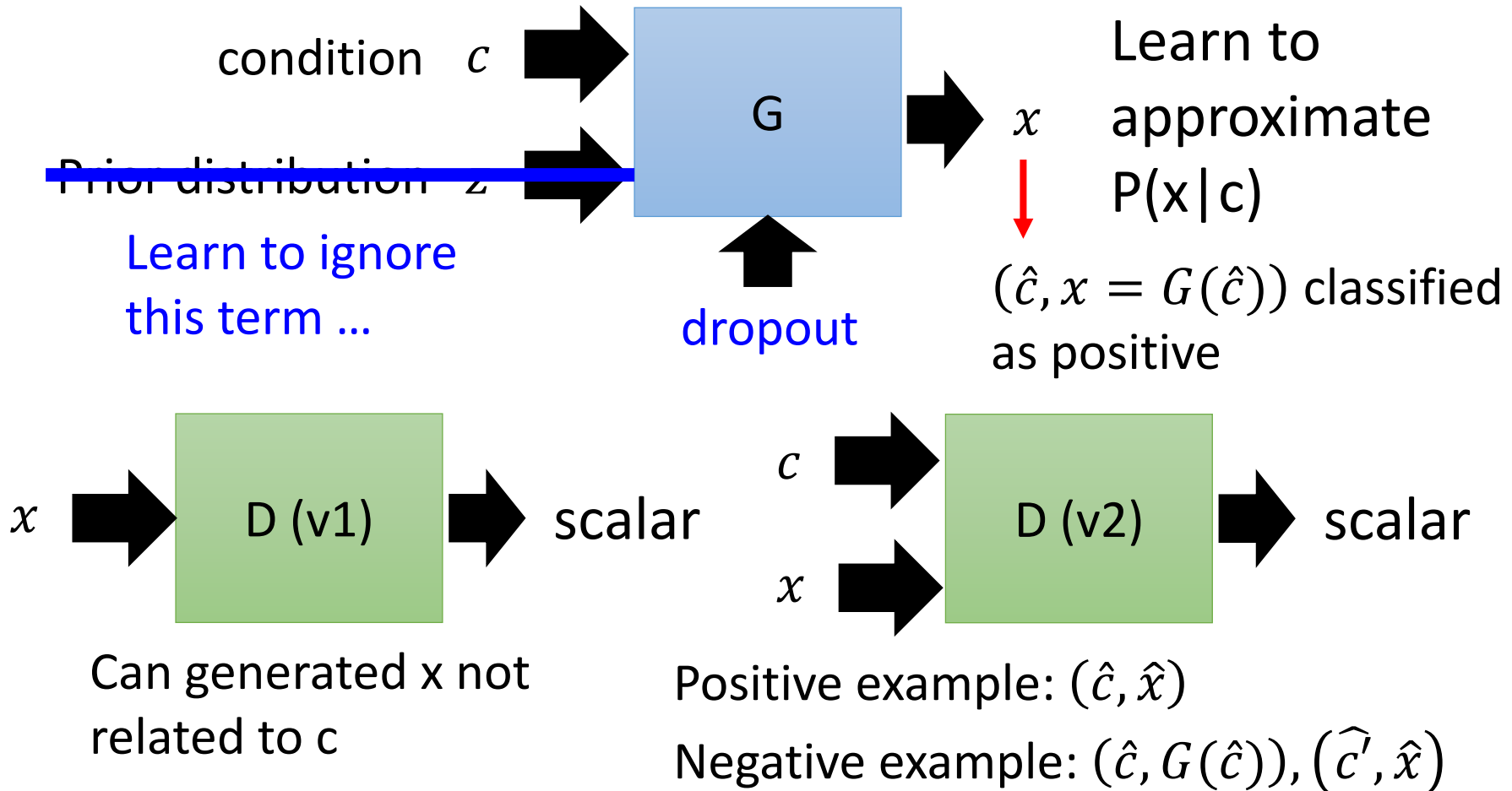
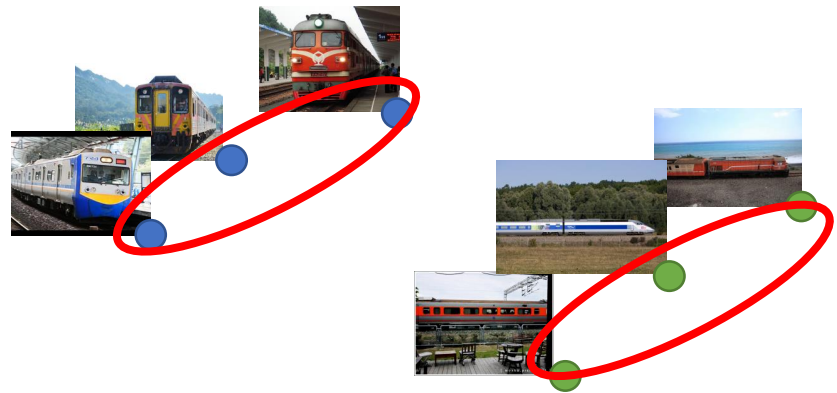


Text: "train"






Conditional GAN

Training data: (\hat{c}, \hat{x})

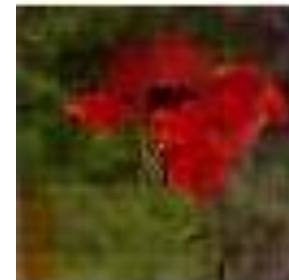


Text to Image - Results

Caption	Image
a pitcher is about to throw the ball to the batter	
a group of people on skis stand in the snow	
a man in a wet suit riding a surfboard on a wave	

Text to Image - Results

"red flower with
black center"



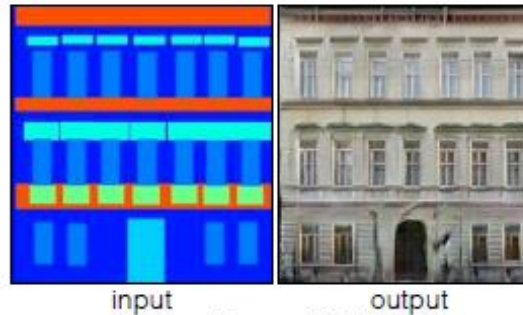
Caption	Image
this flower has white petals and a yellow stamen	A grid of 16 small images showing various white flowers with yellow centers, arranged in two rows of eight.
the center is yellow surrounded by wavy dark purple petals	A grid of 16 small images showing various purple flowers with yellow centers, arranged in two rows of eight.
this flower has lots of small round pink petals	A grid of 16 small images showing various pink flowers, arranged in two rows of eight.

Image-to-image Translation

Labels to Street Scene



Labels to Facade



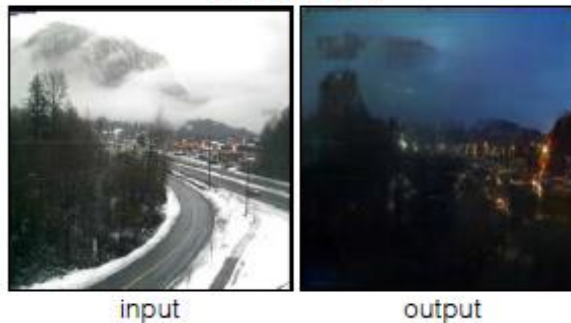
BW to Color



Aerial to Map



Day to Night

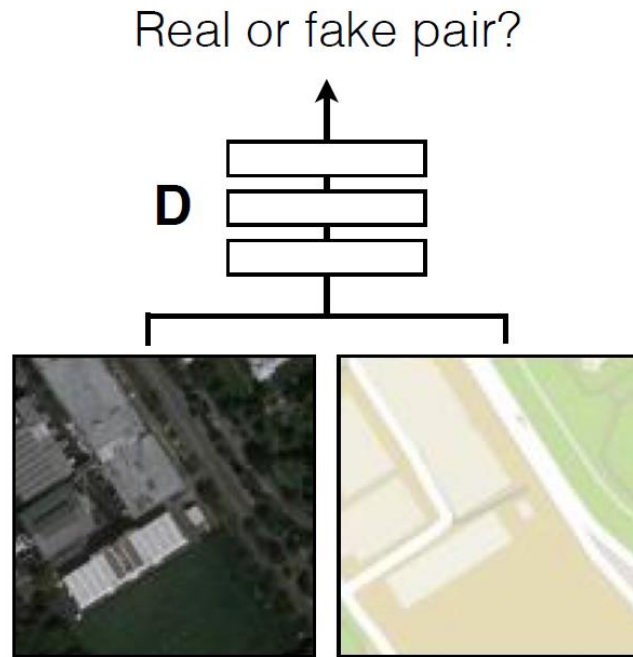


Edges to Photo



Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, Alexei A. Efros, "Image-to-Image Translation with Conditional Adversarial Networks", arXiv preprint, 2016

Positive examples



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

D tries to identify the fakes

Negative examples

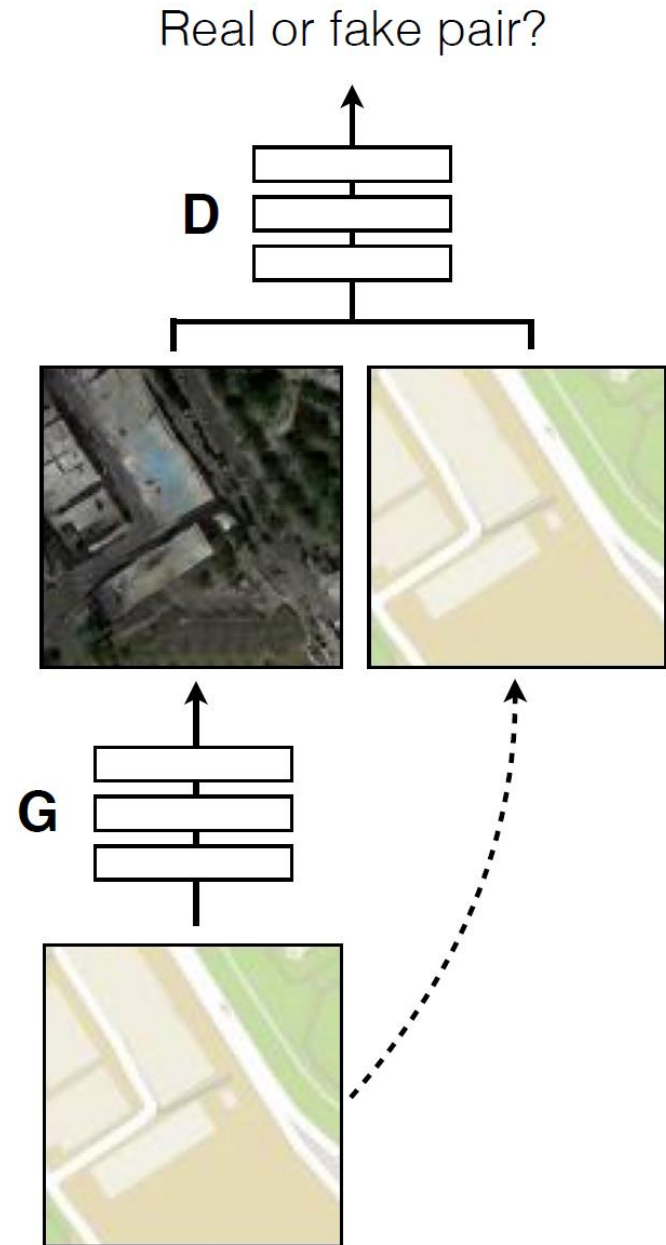


Image-to-image Translation

- Results

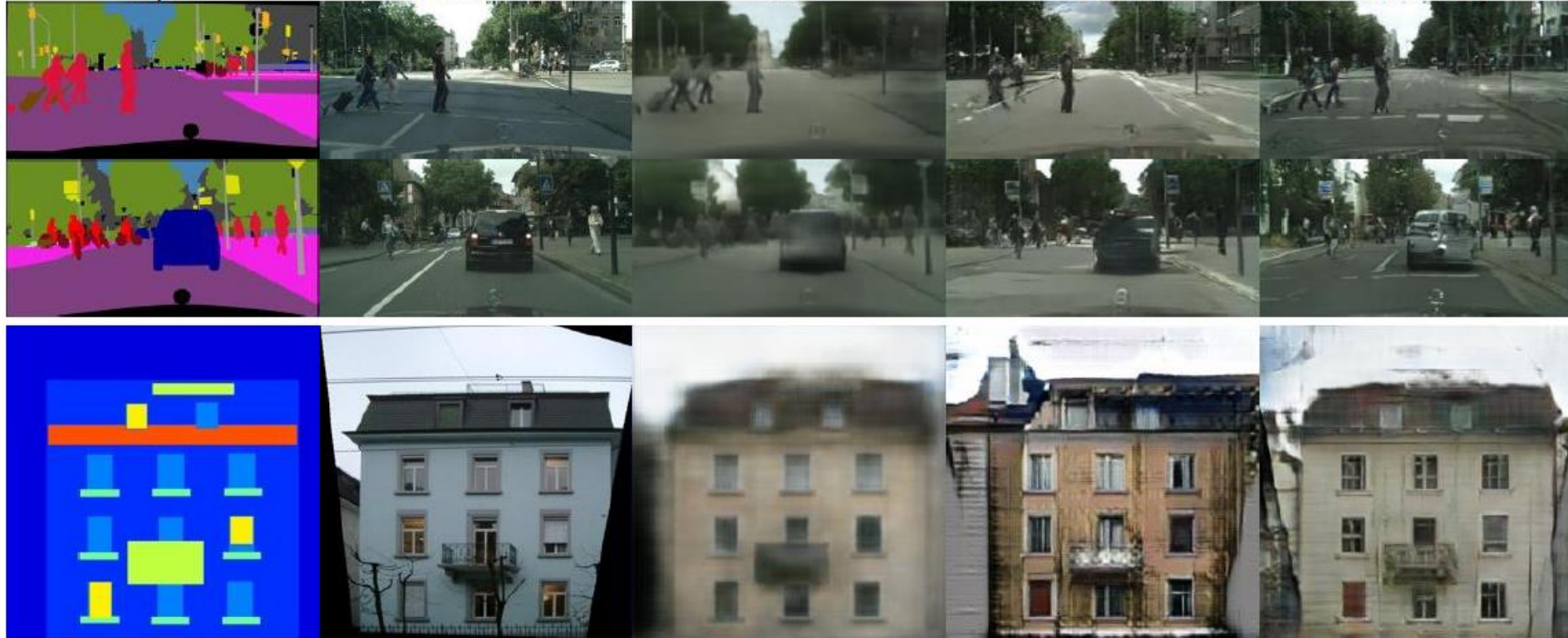
Input

Ground truth

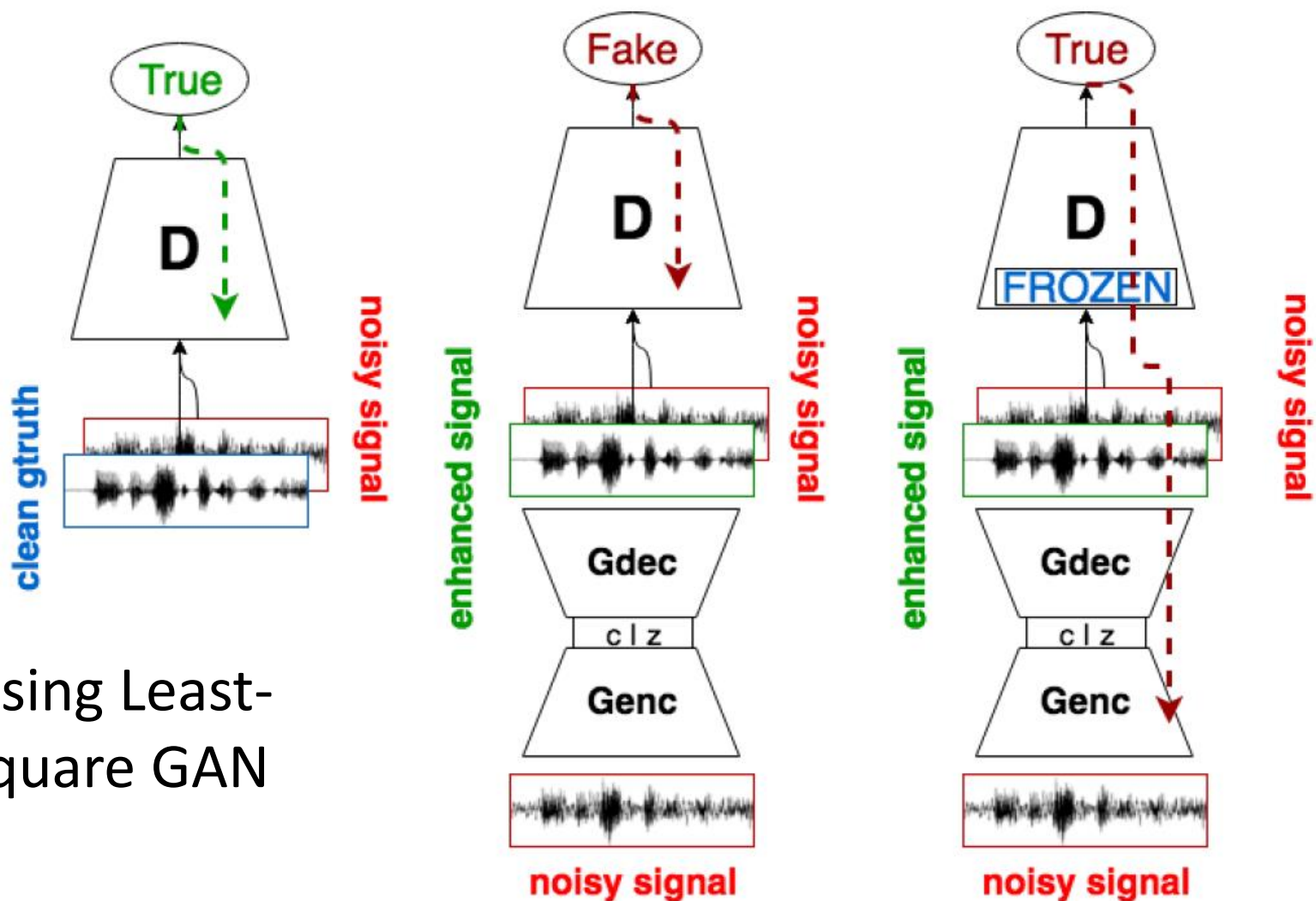
L1

cGAN

L1 + cGAN



Speech Enhancement GAN



Least-square GAN

- For discriminator

D has linear output

$$\min_D \frac{1}{2} E_{x \sim P_{data}} [(D(x) - \underset{1}{b})^2] + \frac{1}{2} E_{x \sim P_G} [(D(x) - \underset{0}{a})^2]$$

- For Generator

$$\min_D \frac{1}{2} E_{z \sim P_{data}} \left[(D(G(z)) - \underset{1}{c})^2 \right]$$

Least-square GAN

- The code used in demo from:
 - https://github.com/osh/KerasGAN/blob/master/MNIST_CNN_GAN_v2.ipynb