人工智能技术及应用

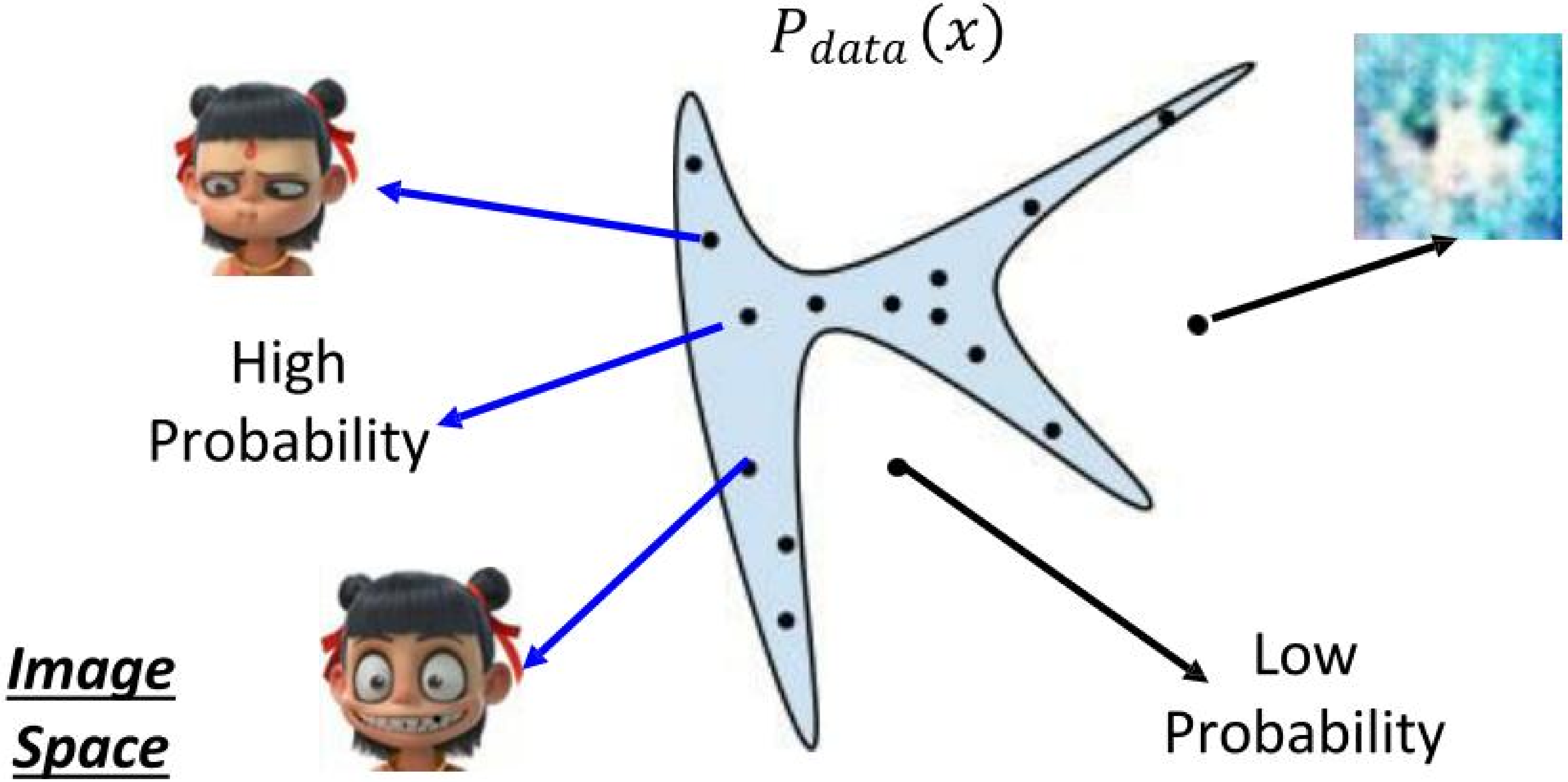
Artificial Intelligence and Application

Theory behind GAN

Generation

x: an image (a high-dimensional vector)

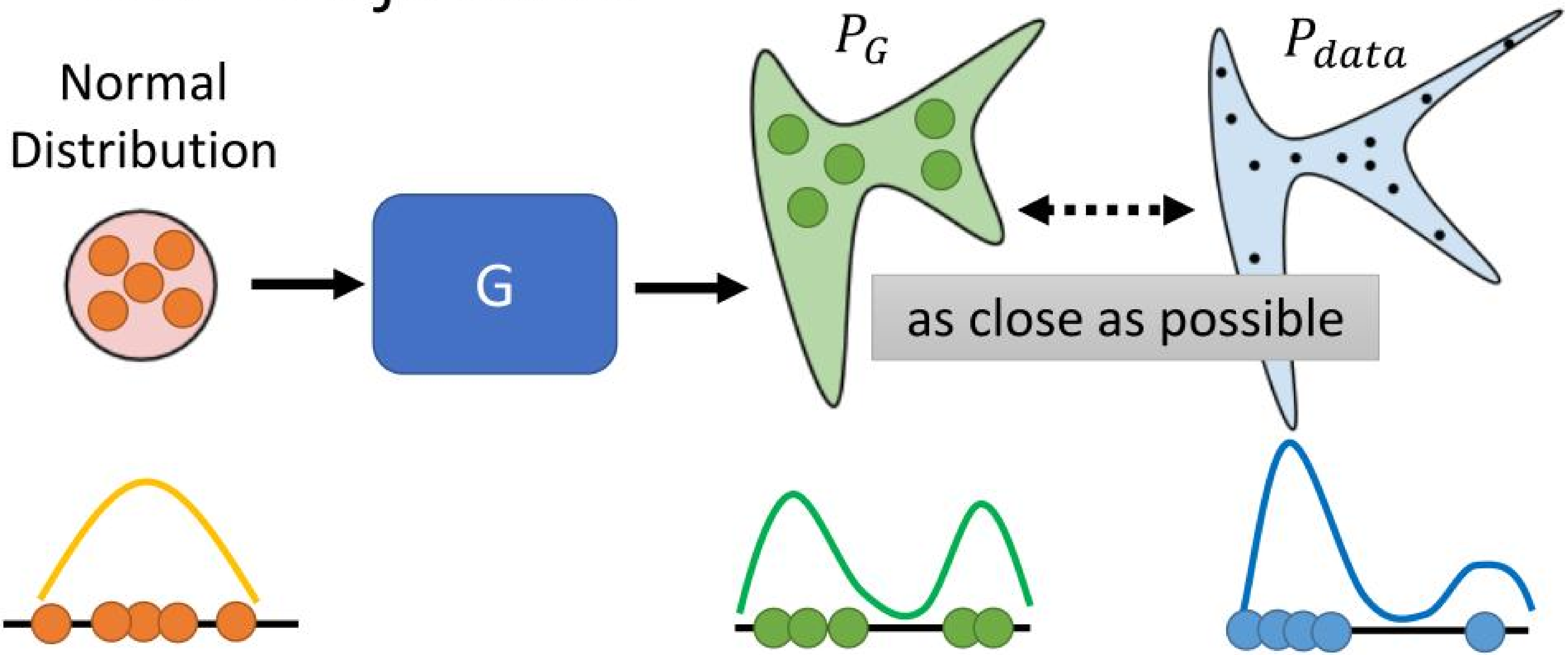
- We want to find data distribution $P_{data}\ (x)$



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

c.f.

Our Objective



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

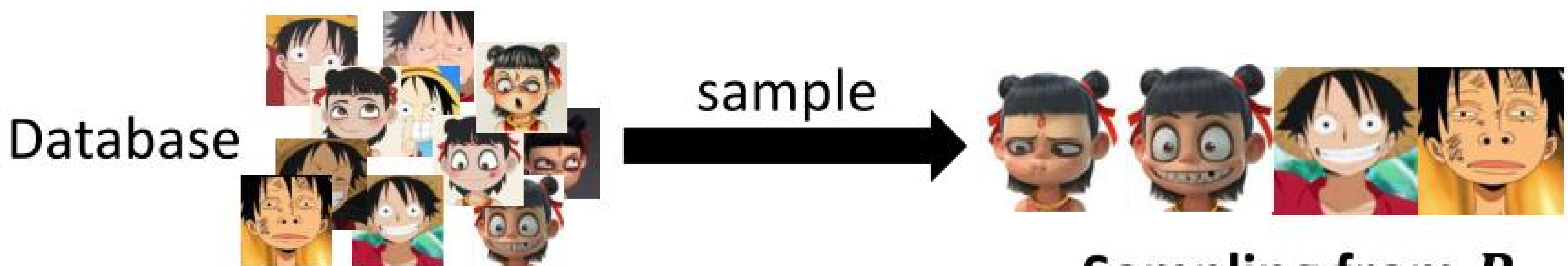
Divergence between distributions P_G and P_{data}

How to compute the divergence?

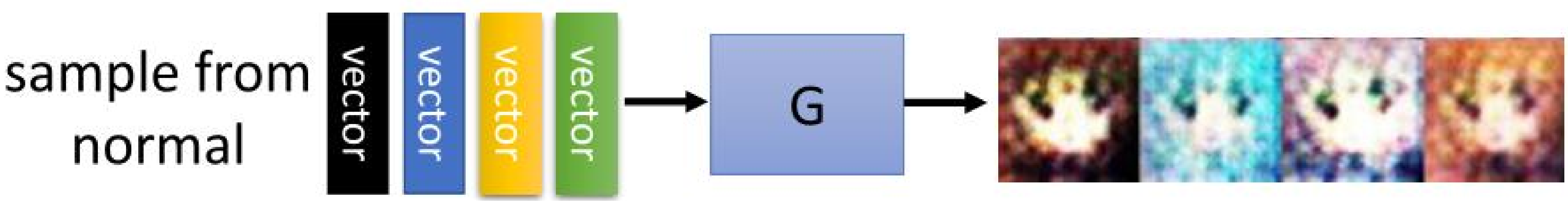
Sampling is good enough ……

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

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



Sampling from P_{data}



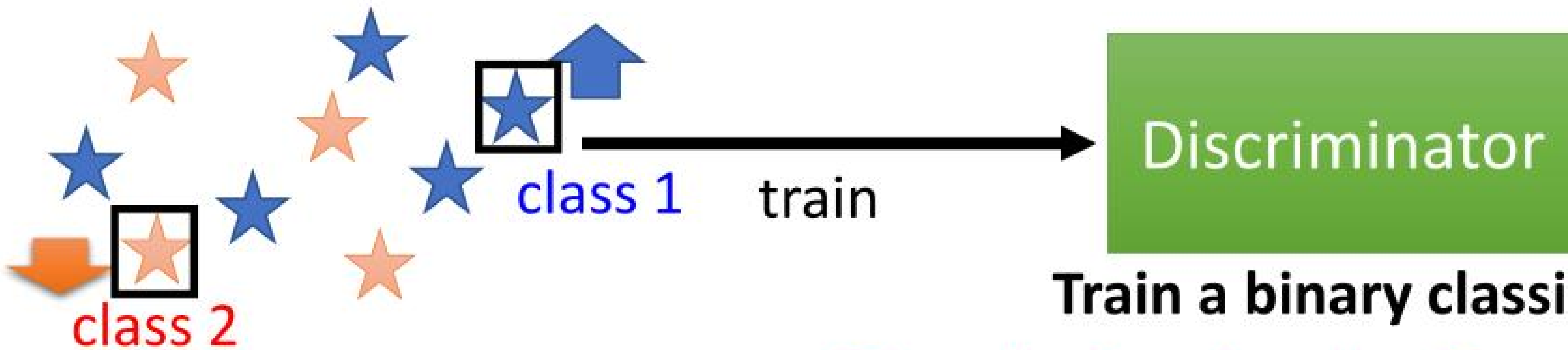
Sampling from P_G

https://arxiv.org/abs/1406.2661

Discriminator

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

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



Train a binary classifier

Training: $D^* = arg\boxed{\max_D V(D,G)}$ The value is related to JS divergence.

Objective Function for D

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

$D^* = arg\max_D \boxed{V(D,G)}$ = Training classifier: minimize cross entropy

negative cross entropy

Discriminator

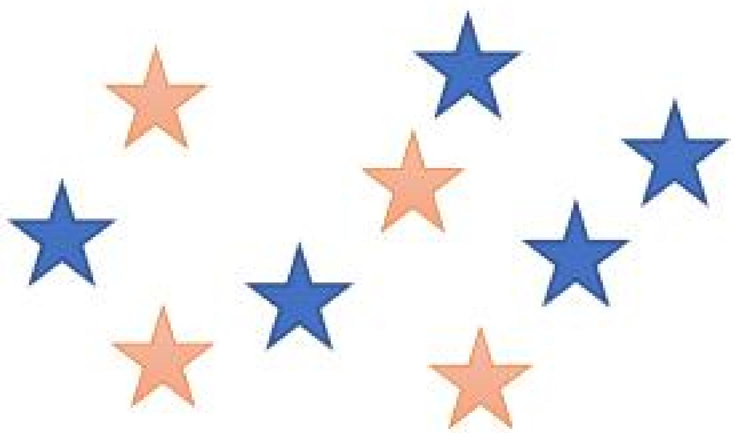
$$G^* = arg \min_G Div(P_G, P_{data})$$

★ : data sampled from P_{data}

★ : data sampled from P_G

Training:

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



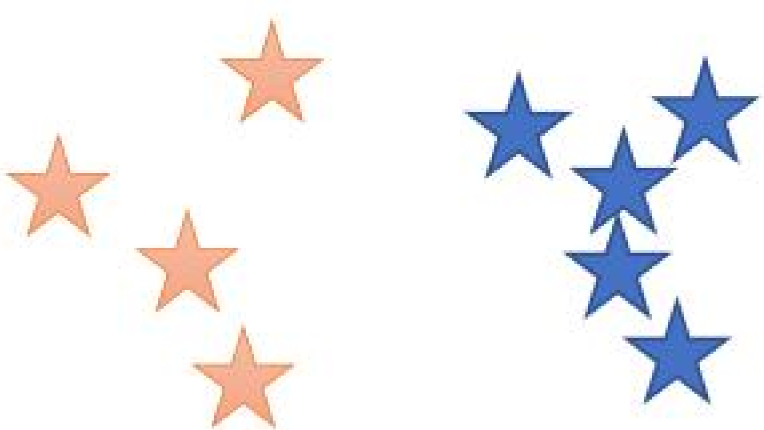
small divergence

train

Discriminator

hard to discriminate

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



large divergence

train

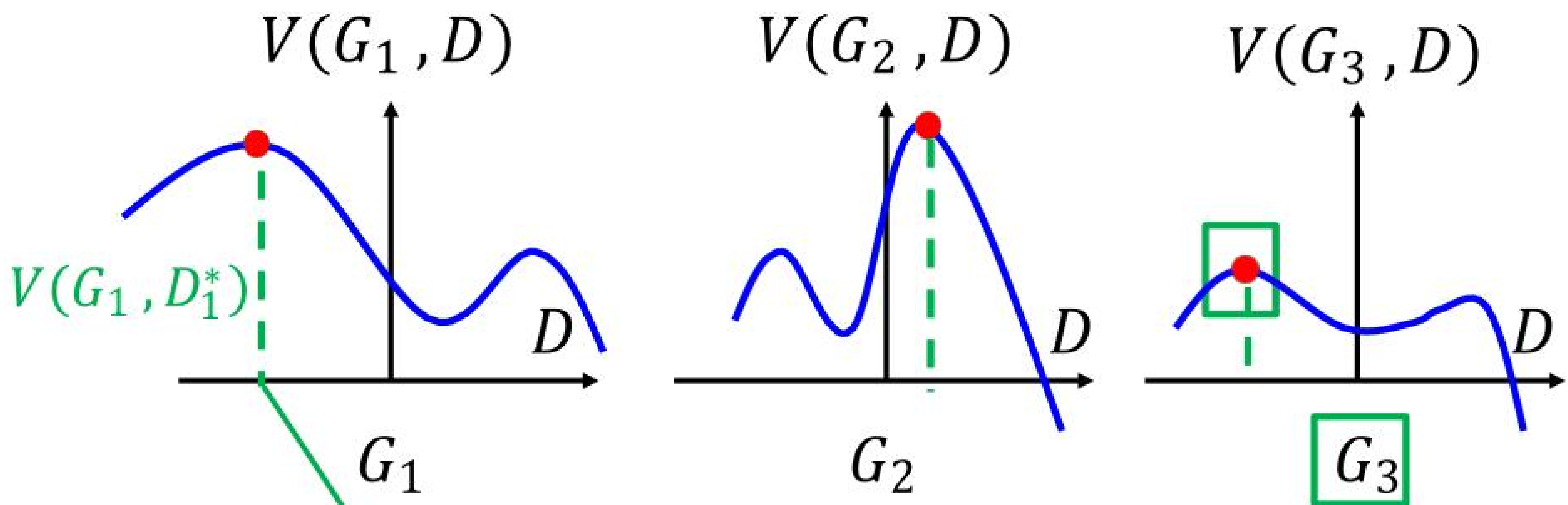
Discriminator

easy to discriminate

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

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

The maximum objective value is related to JS divergence.



Divergence between P_{G_1} and P_{data}

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

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

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 ☺

https://arxiv.org/abs/1606.00709

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

GAN is difficult to train ……

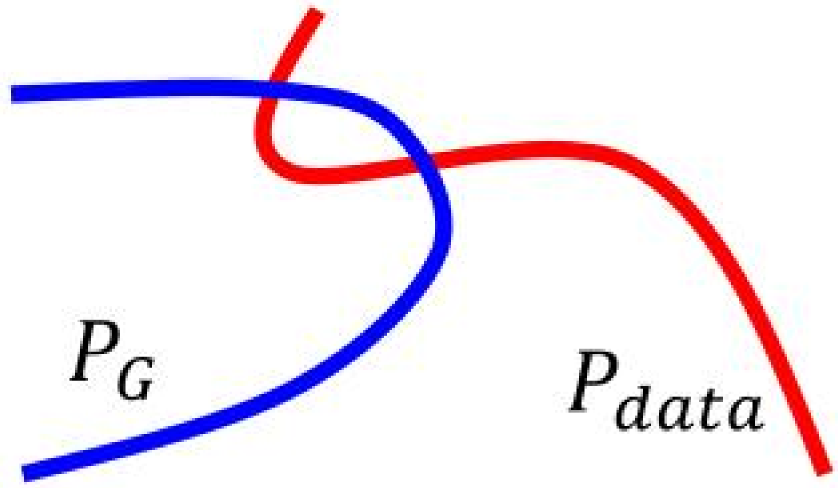
Tips for GAN

JS divergence is not suitable

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

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

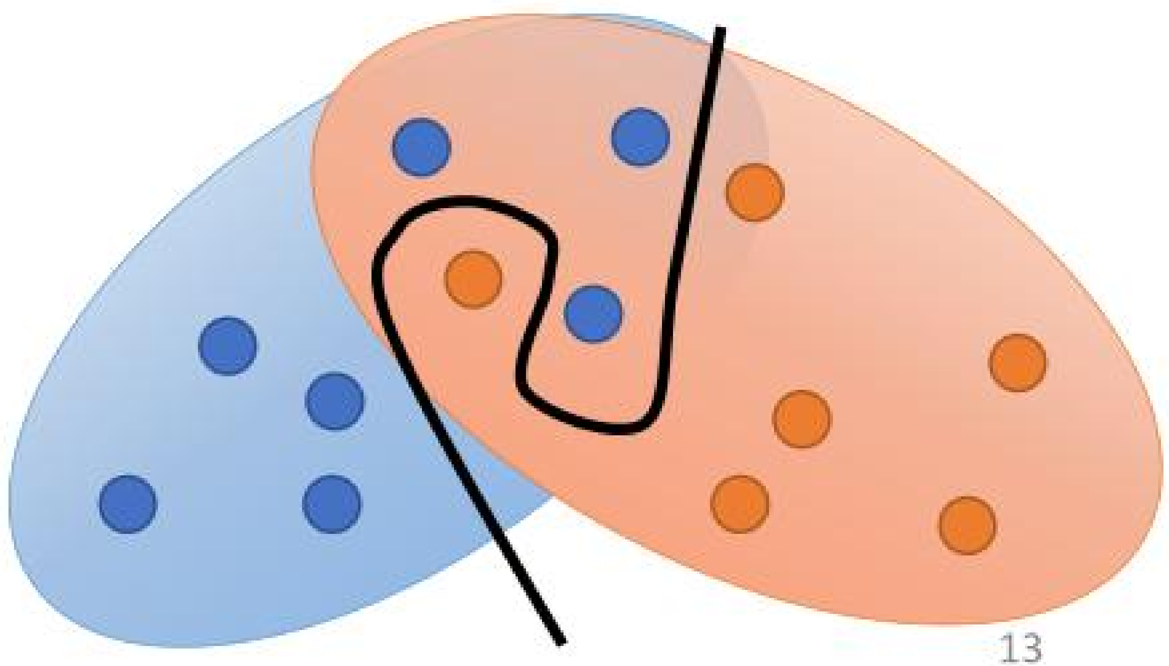
The overlap can be ignored.



- 2. Sampling

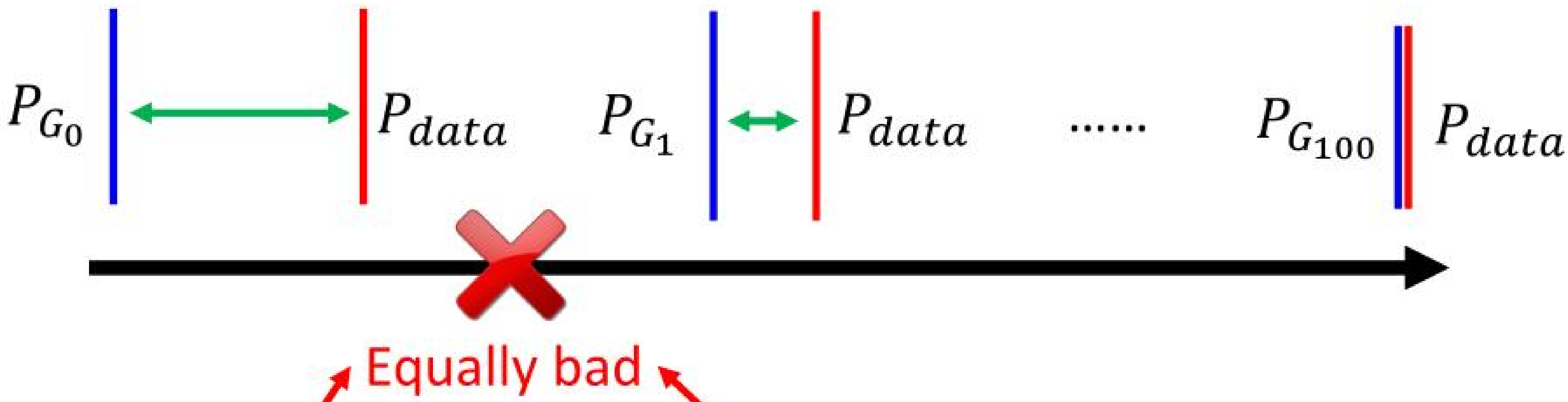
Even though P_{data} and P_G have overlap.

If you do not have enough sampling ……



What is the problem of JS divergence?

JS divergence is always log2 if two distributions do not overlap.



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

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

……

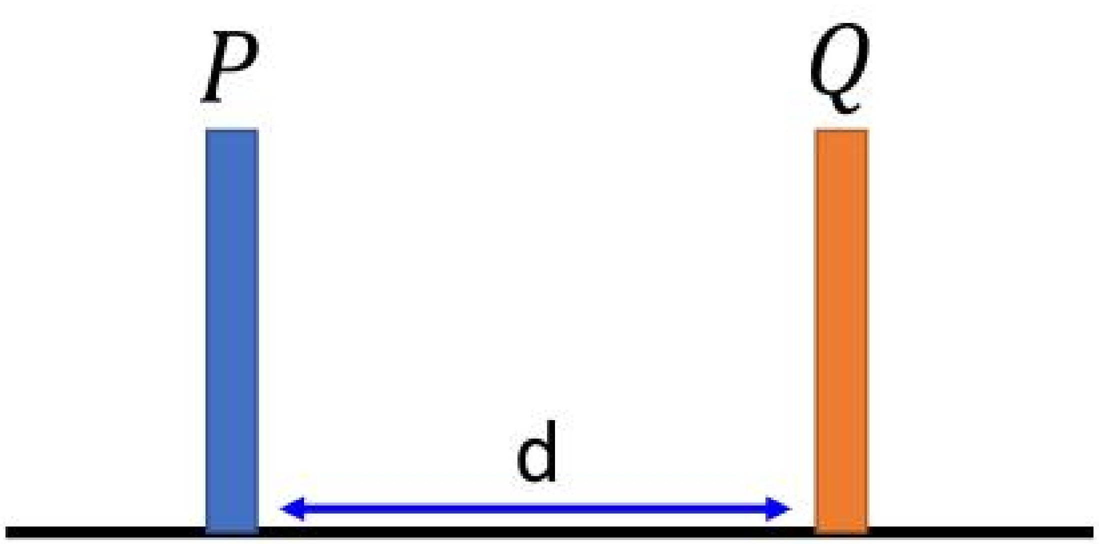
$JS(P_{G_{100}}, P_{data})$
$= 0$

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

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

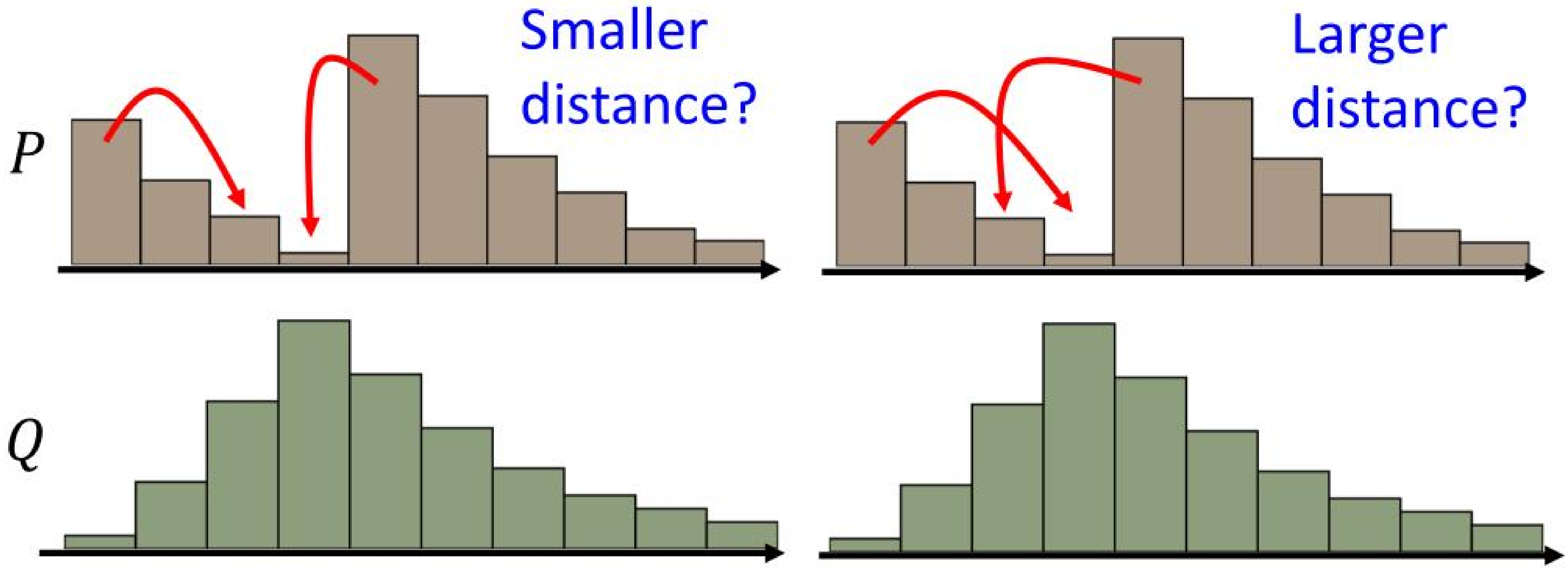
Wasserstein distance

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



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

Wasserstein distance



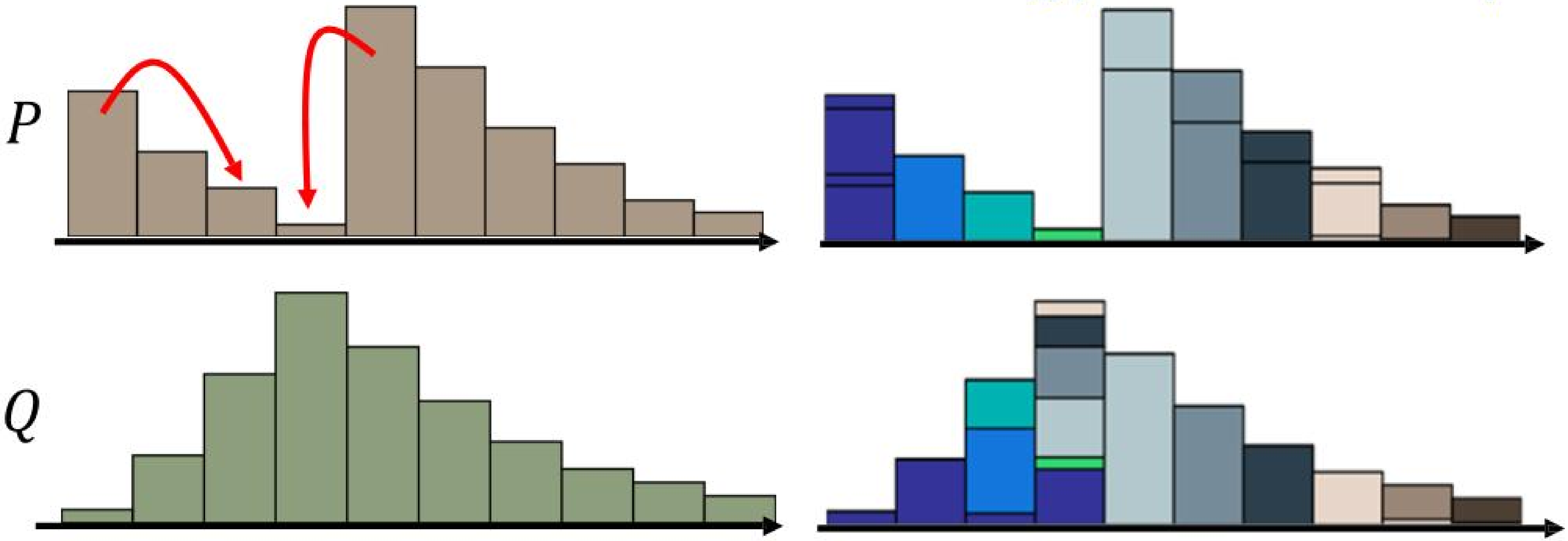
There are many possible "moving plans".

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

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

WGAN: Earth Mover’s Distance

Best “moving plans” of this example

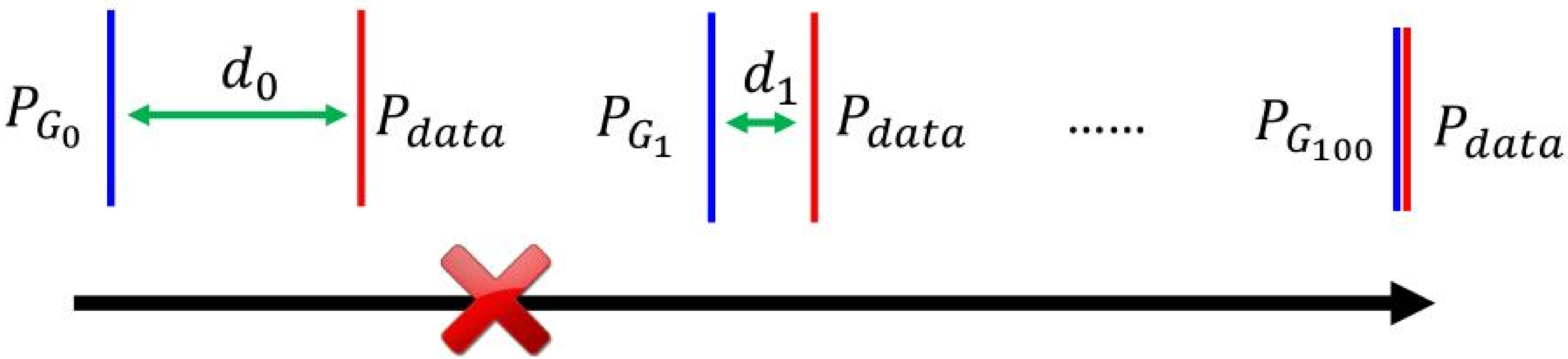


There many possible “moving plans”.

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/

What is the problem of JS divergence?



$JS(P_{G_0}, P_{data}) = log2$ $JS(P_{G_1}, P_{data}) = log2$ …… $JS(P_{G_{100}}, P_{data}) = 0$

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

Better!

https://arxiv.org/abs/1701.07875

WGAN

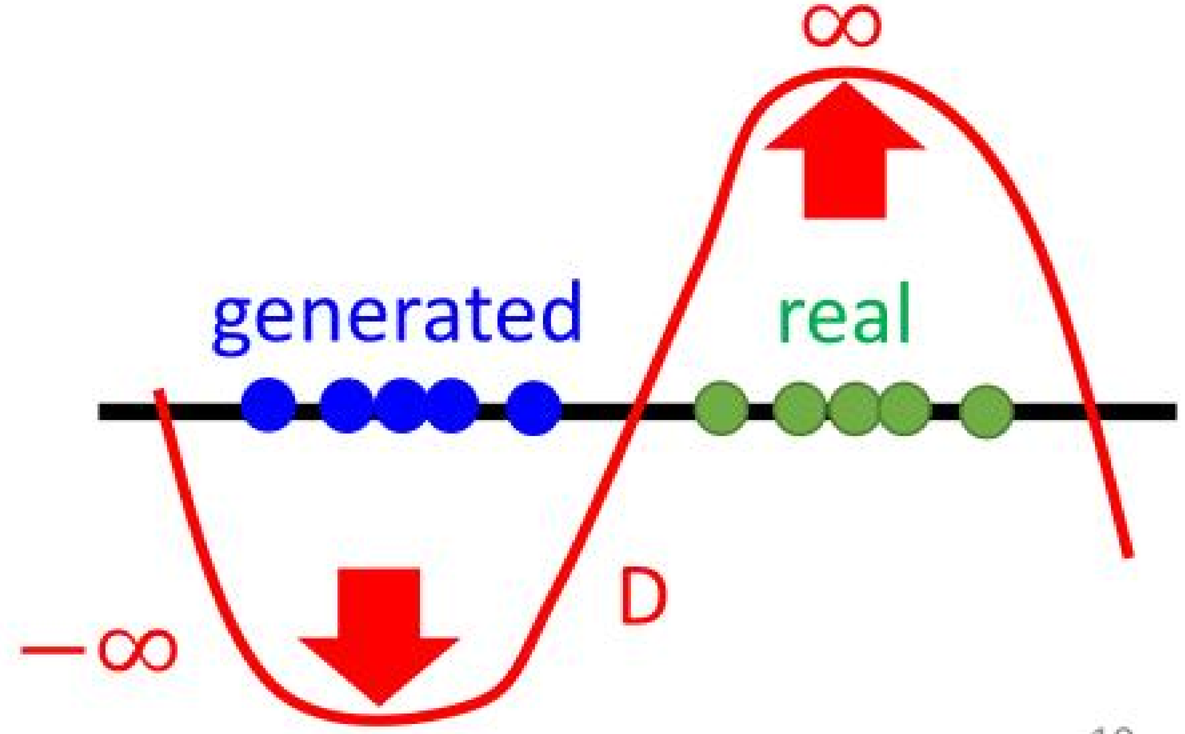
Evaluate Wasserstein distance between P_{data} and P_G

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

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

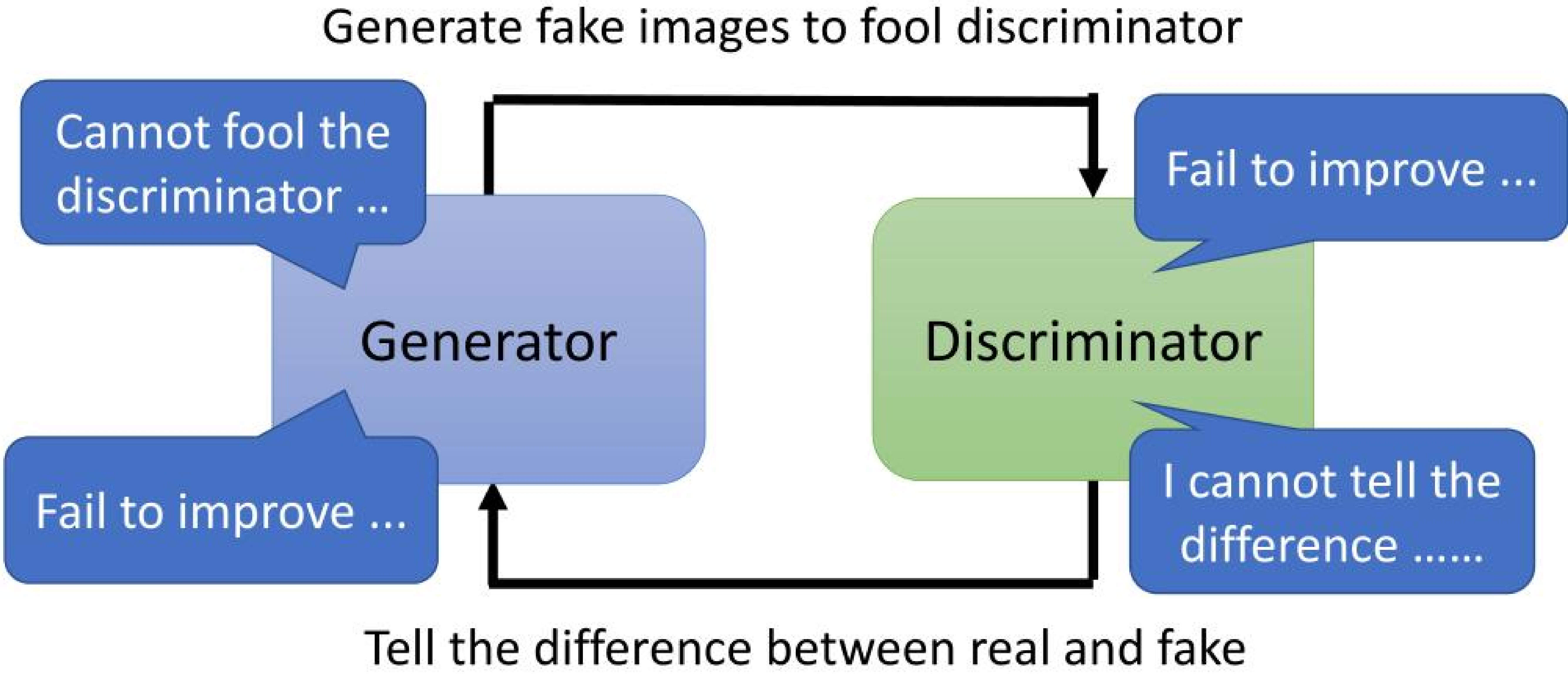
Without the constraint, the training of D will not converge.

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



GAN is still challenging …

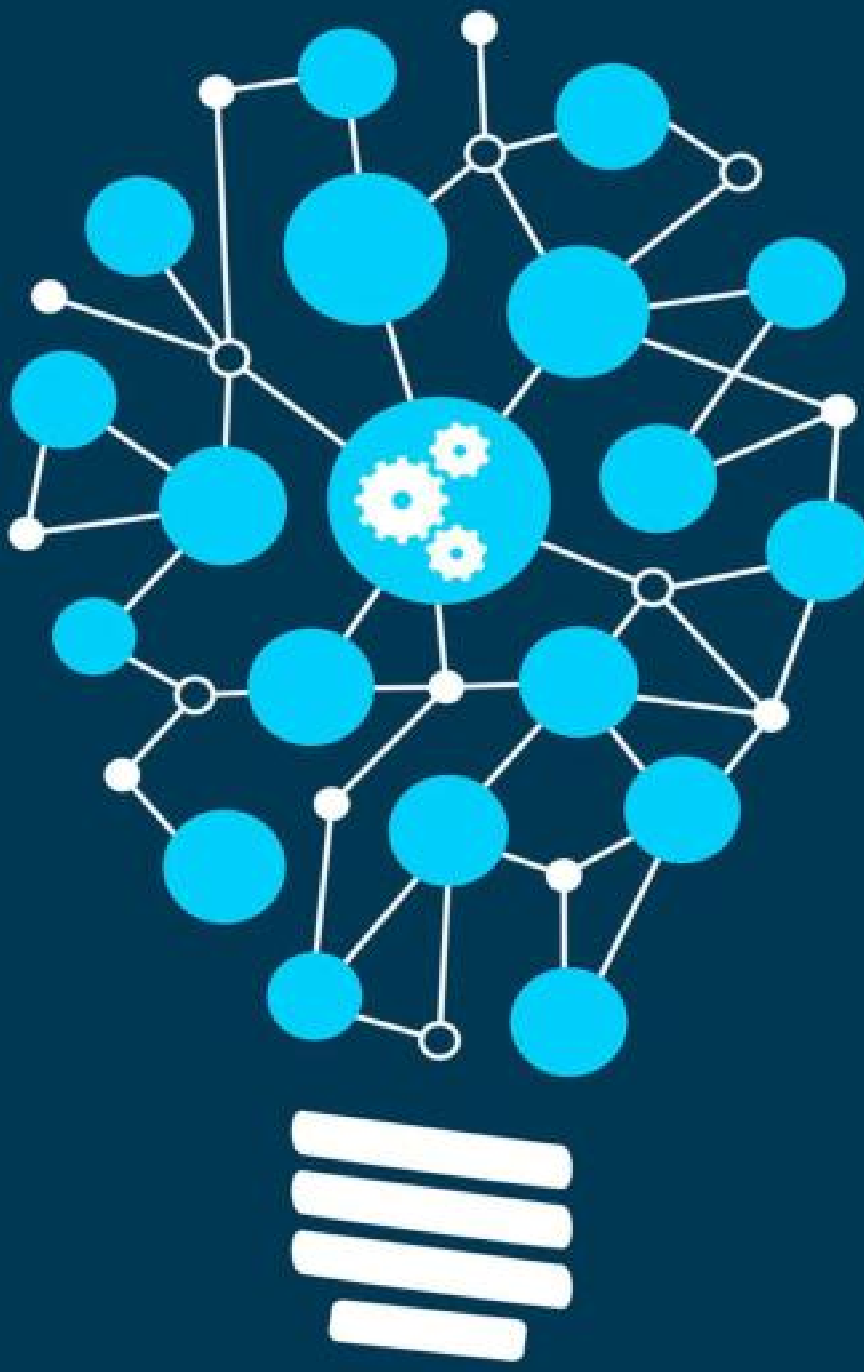
- Generator and Discriminator needs to match each other (棋逢对手)



More Tips

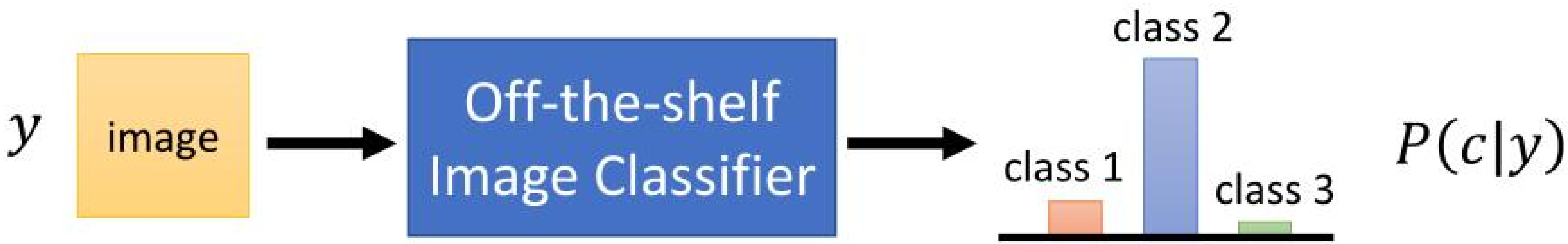
- Tips from Soumith
  - https://github.com/soumith/ganhacks
- Tips in DCGAN: Guideline for network architecture design for image generation
  - https://arxiv.org/abs/1511.06434
- Improved techniques for training GANs
  - https://arxiv.org/abs/1606.03498
- Tips from BigGAN
  - https://arxiv.org/abs/1809.11096

Evaluation of Generation



Quality of Image

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



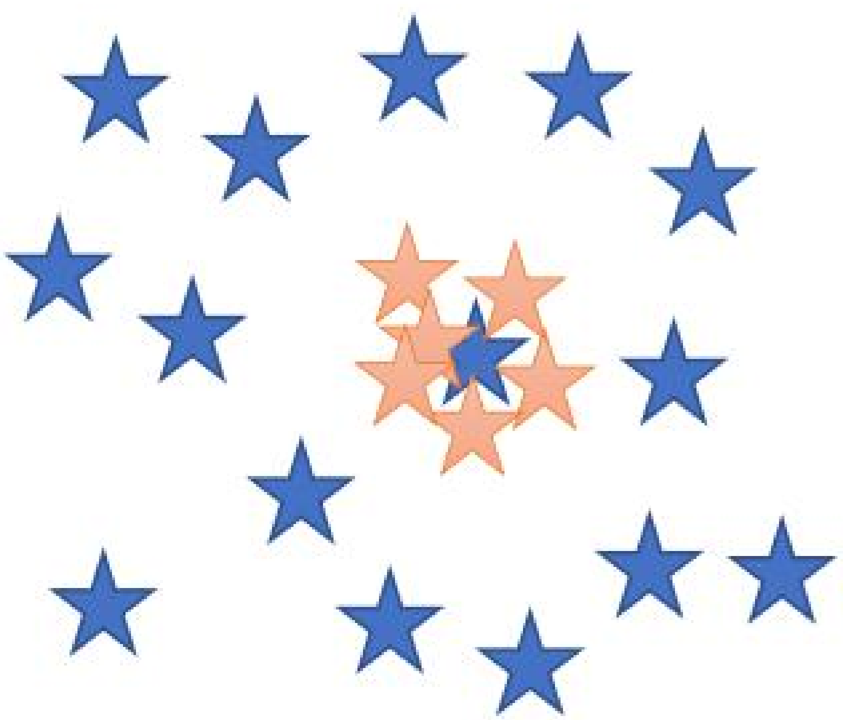
e.g., Inception net, VGG, etc.

Concentrated distribution means higher visual quality

Diversity - Mode Collapse

: real data

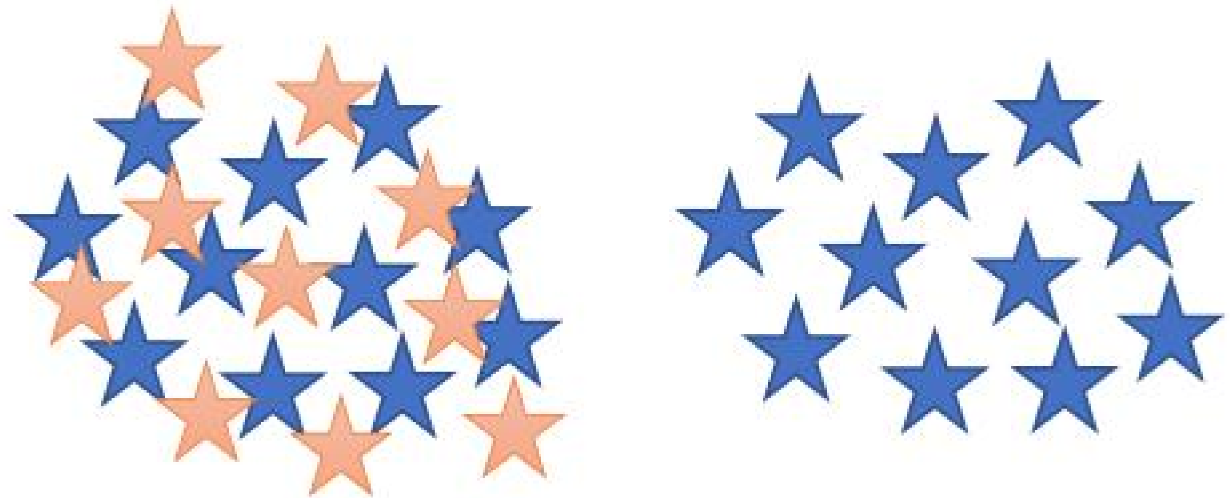
: generated data



Diversity - Mode Dropping

★ : real data

★ : generated data

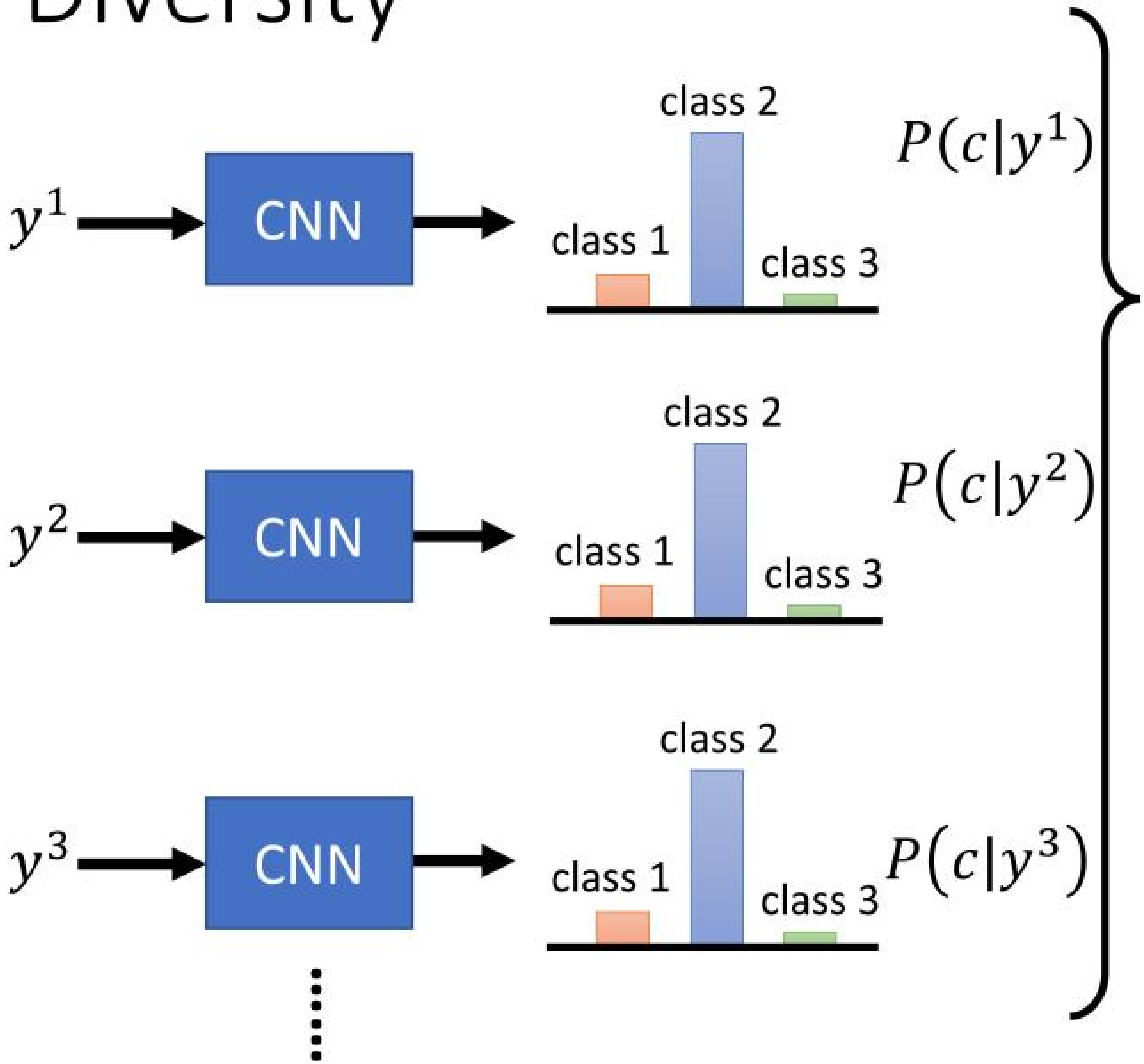


Generator
at iteration t

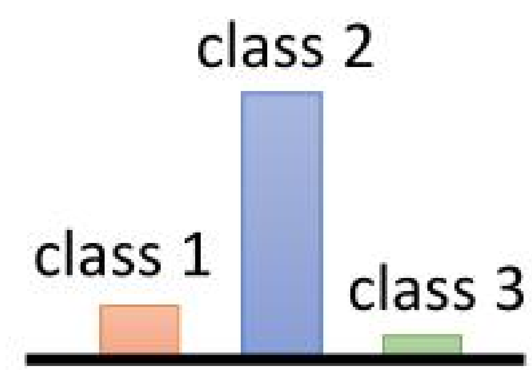
Generator
at iteration t+1

(BEGAN on CelebA)

Diversity



$$P(c) = \frac{1}{N}\sum_{n} P(c|y^n)$$

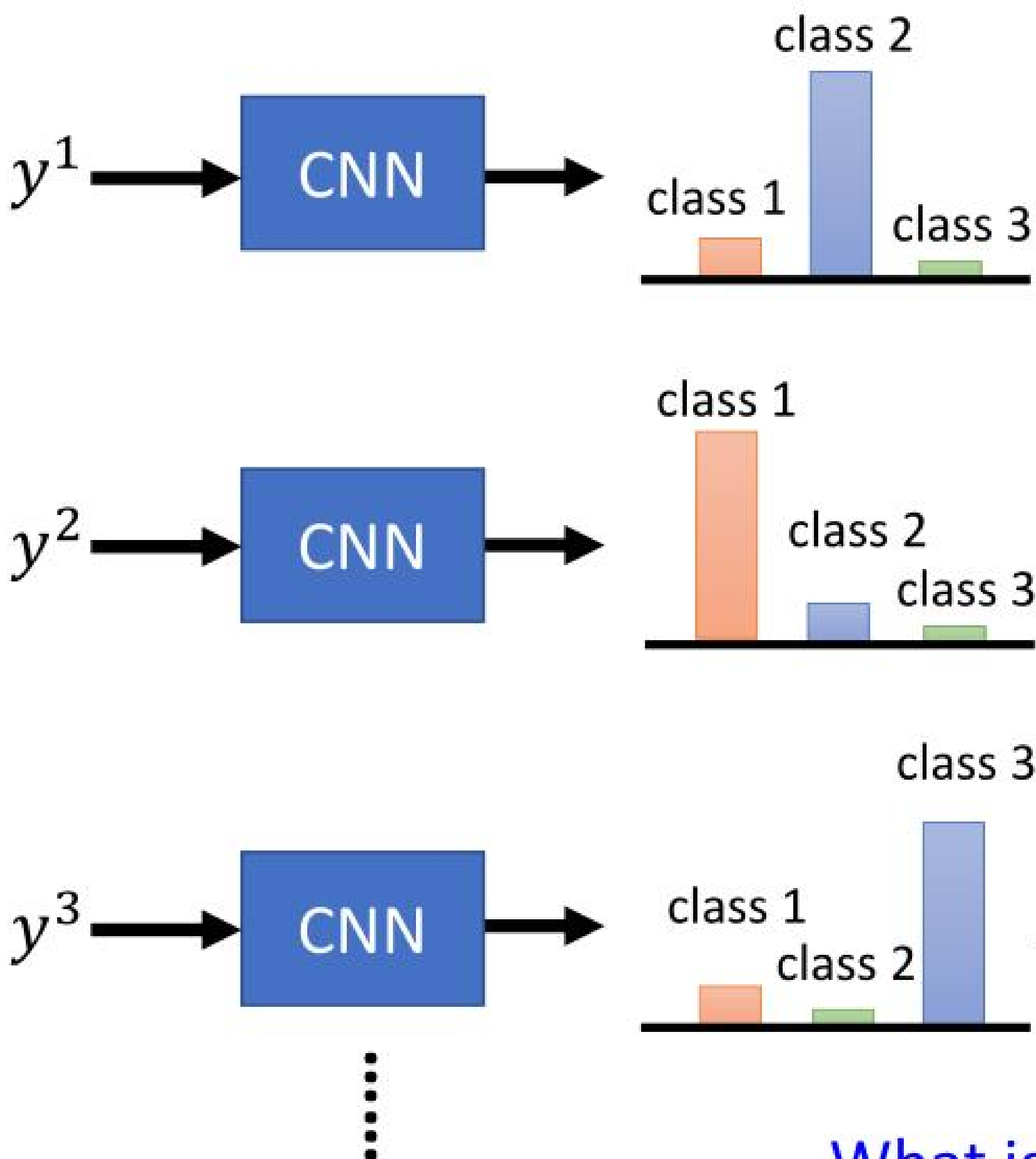


low diversity

Diversity

Inception Score (IS):

Good quality, large diversity → Large **IS**

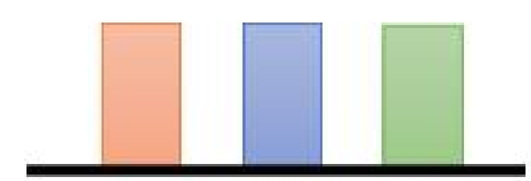


$P(c|y^1)$

$P(c|y^2)$

$P(c|y^3)$

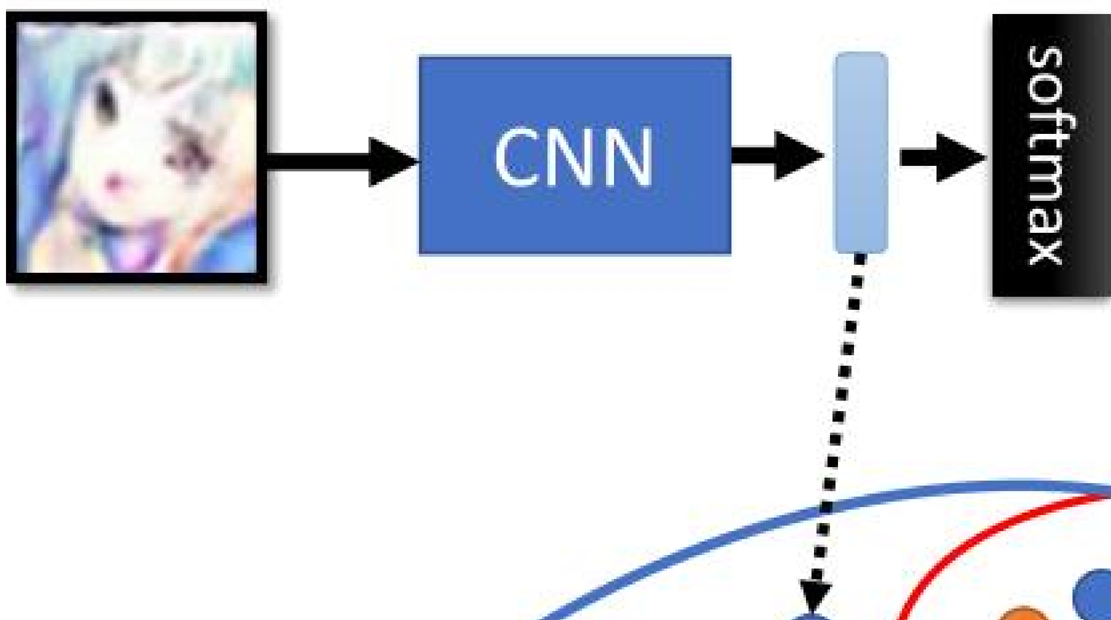
$$P(c) = \frac{1}{N}\sum_n P(c|y^n)$$



Uniform means higher variety

What is the problem here? ☺

Fréchet Inception Distance (FID)

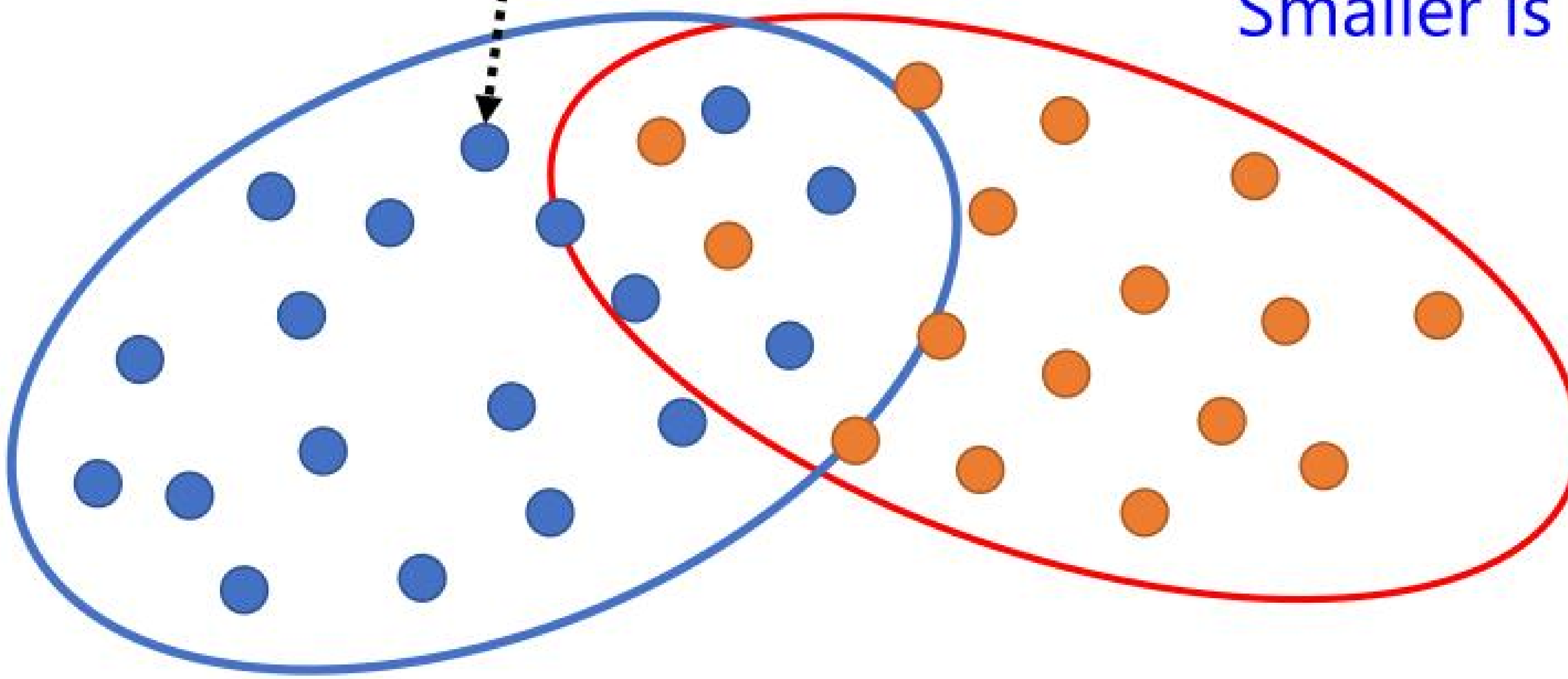


red points: real images

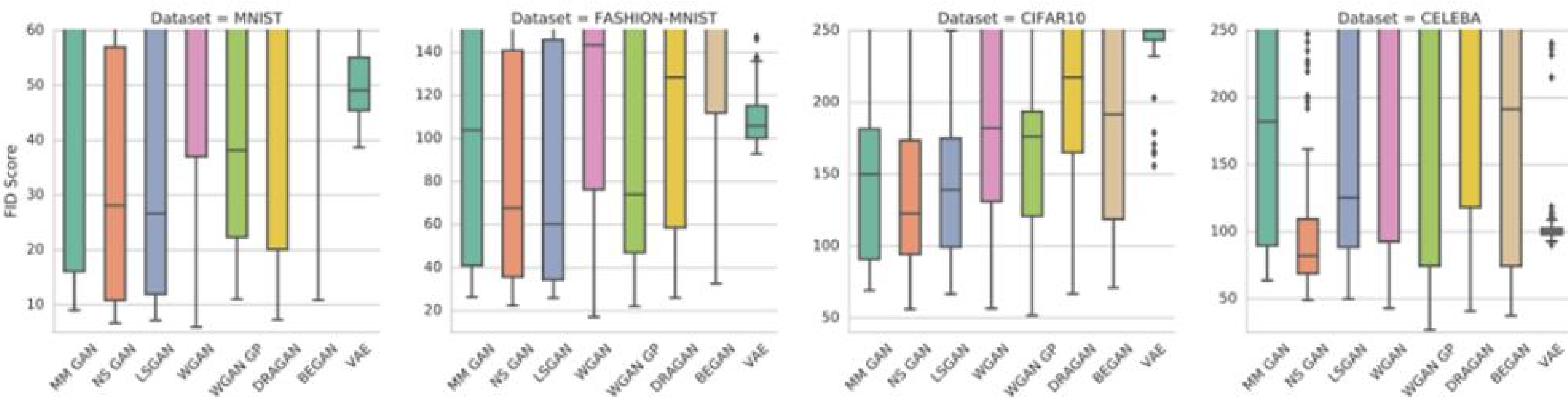
blue points: generated images

FID = Fréchet distance between the two **Gaussians** ???

Smaller is better



GAN	Discriminator Loss	Generator Loss
MM GAN	$\mathcal{L}_{\mathrm{D}}^{\mathrm{GAN}} = -\mathbb{E}_{x\sim p_d}[\log(D(x))] + \mathbb{E}_{\hat{x}\sim p_g}[\log(1 - D(\hat{x}))]$	$\mathcal{L}_{\mathrm{G}}^{\mathrm{GAN}} = -\mathcal{L}_{\mathrm{D}}^{\mathrm{GAN}}$
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	$\mathcal{L}_{\mathrm{D}}^{\mathrm{WGAN}} = \mathcal{L}_{\mathrm{D}}^{\mathrm{WGAN}} + \lambda\mathbb{E}_{\hat{x}\sim p_g}[(\lvert\lvert\nabla D(\alpha x + (1-\alpha\hat{x})\rvert\rvert_2 - 1)^2]$	$\mathcal{L}_{\mathrm{G}}^{\mathrm{WGAN}} = -\mathbb{E}_{\hat{x}\sim p_g}[D(\hat{x})]$
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	$\mathcal{L}_{\mathrm{D}}^{\mathrm{DRAGAN}} = \mathcal{L}_{\mathrm{D}}^{\mathrm{GAN}} + \lambda\mathbb{E}_{\hat{x}\sim p_d+\mathcal{N}(0,c)}[(\lvert\lvert\nabla D(\hat{x})\rvert\rvert_2 - 1)^2]$	$\mathcal{L}_{\mathrm{G}}^{\mathrm{DRAGAN}} = -\mathcal{L}_{\mathrm{D}}^{\mathrm{NS\ GAN}}$
BEGAN	$\mathcal{L}_{\mathrm{D}}^{\mathrm{BEGAN}} = \mathbb{E}_{x\sim p_d}[\lvert\lvert x - \mathrm{AE}(x)\rvert\rvert_1] - k_t\mathbb{E}_{\hat{x}\sim p_g}[\lvert\lvert\hat{x} - \mathrm{AE}(\hat{x})\rvert\rvert_1]$	$\mathcal{L}_{\mathrm{G}}^{\mathrm{BEGAN}} = \mathbb{E}_{\hat{x}\sim p_g}[\lvert\lvert\hat{x} - \mathrm{AE}(\hat{x})\rvert\rvert_1]$



FIT: Smaller is better

Are GANs Created Equal? A Large-Scale Study

https://arxiv.org/abs/1711.10337

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

We don't want memory GAN.

Real Data

Generated Data

Same as real data ...

Generated Data

Simply flip real data ...

To learn more about evaluation …

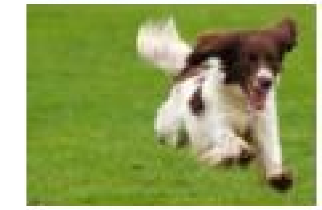
	Measure	Description
Quantitative	1. Average Log-likelihood [18, 22]	• Log likelihood of explaining realworld held out/test data using a density estimated from the generated data (*e.g.* using KDE or Parzen window estimation). $L = \frac{1}{N}\sum_i \log P_{model}(\mathbf{x}_i)$
	2. Coverage Metric [33]	• The probability mass of the true data "covered" by the model distribution $C := P_{data}(dP_{model} > t)$ with t such that $P_{model}(dP_{model} > t) = 0.95$
	3. Inception Score (IS) [3]	• KLD between conditional and marginal label distributions over generated data. $\exp\left(\mathbb{E}_{\mathbf{x}}\left[\mathbb{KL}\left(p(y \mid \mathbf{x}) \parallel p(y)\right)\right]\right)$
	4. Modified Inception Score (m-IS) [34]	• Encourages diversity within images sampled from a particular category. $\exp(\mathbb{E}_{\mathbf{x}_i}[\mathbb{E}_{\mathbf{x}_j}[(\mathbb{KL}(P(y\|\mathbf{x}_i)\|\|P(y\|\mathbf{x}_j)))]])$
	5. Mode Score (MS) [35]	• Similar to IS but also takes into account the prior distribution of the labels over real data. $\exp\left(\mathbb{E}_{\mathbf{x}}\left[\mathbb{KL}\left(p(y \mid \mathbf{x}) \parallel p(y^{train})\right)\right] - \mathbb{KL}\left(p(y) \parallel p(y^{train})\right)\right)$
	6. AM Score [36]	• Takes into account the KLD between distributions of training labels vs. predicted labels, as well as the entropy of predictions. $\mathbb{KL}(p(y^{train}) \parallel p(y)) + \mathbb{E}_{\mathbf{x}}[H(y\|\mathbf{x})]$
	7. Fréchet Inception Distance (FID) [37]	• Wasserstein-2 distance between multi-variate Gaussians fitted to data embedded into a feature space $FID(r,g) = \|\|\mu_r - \mu_g\|\|_2^2 + Tr(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{\frac{1}{2}})$
	8. Maximum Mean Discrepancy (MMD) [38]	• Measures the dissimilarity between two probability distributions P_r and P_g using samples drawn independently from each distribution. $M_k(P_r, P_g) = \mathbb{E}_{\mathbf{x},\mathbf{x}' \sim P_r}[k(\mathbf{x},\mathbf{x}')] - 2\mathbb{E}_{\mathbf{x} \sim P_r, \mathbf{y} \sim P_g}[k(\mathbf{x},\mathbf{y})] + \mathbb{E}_{\mathbf{y},\mathbf{y}' \sim P_g}[k(\mathbf{y},\mathbf{y}')]$
	9. The Wasserstein Critic [39]	• The critic (*e.g.* an NN) is trained to produce high values at real samples and low values at generated samples $\hat{W}(\mathbf{x}_{test}, \mathbf{x}_g) = \frac{1}{N}\sum_{i=1}^{N}\hat{f}(\mathbf{x}_{test}[i]) - \frac{1}{N}\sum_{i=1}^{N}\hat{f}(\mathbf{x}_g[i])$
	10. Birthday Paradox Test [27]	• Measures the support size of a discrete (continuous) distribution by counting the duplicates (near duplicates)
	11. Classifier Two Sample Test (C2ST) [40]	• Answers whether two samples are drawn from the same distribution (*e.g.* by training a binary classifier)
	12. Classification Performance [1, 15]	• An indirect technique for evaluating the quality of unsupervised representations (*e.g.* feature extraction; FCN score). See also the GAN Quality Index (GQI) [41].
	13. Boundary Distortion [42]	• Measures diversity of generated samples and covariate shift using classification methods.
	14. Number of Statistically-Different Bins (NDB) [43]	• Given two sets of samples from the same distribution, the number of samples that fall into a given bin should be the same up to sampling noise
	15. Image Retrieval Performance [44]	• Measures the distributions of distances to the nearest neighbors of some query images (*i.e.* diversity)
	16. Generative Adversarial Metric (GAM) [31]	• Compares two GANs by having them engaged in a battle against each other by swapping discriminators or generators. $p(\mathbf{x}\|y=1;M_1')/p(\mathbf{x}\|y=1;M_2') = (p(y=1\|\mathbf{x};D_1)p(\mathbf{x};G_2))/(p(y=1\|\mathbf{x};D_2)p(\mathbf{x};G_1))$
	17. Tournament Win Rate and Skill Rating [45]	• Implements a tournament in which a player is either a discriminator that attempts to distinguish between real and fake data or a generator that attempts to fool the discriminators into accepting fake data as real.
	18. Normalized Relative Discriminative Score (NRDS) [32]	• Compares n GANs based on the idea that if the generated samples are closer to real ones, more epochs would be needed to distinguish them from real samples.
	19. Adversarial Accuracy and Divergence [46]	• Adversarial Accuracy. Computes the classification accuracies achieved by the two classifiers, one trained on real data and another on generated data, on a labeled validation set to approximate $P_g(y\|\mathbf{x})$ and $P_r(y\|\mathbf{x})$. Adversarial Divergence: Computes $\mathbb{KL}(P_g(y\|\mathbf{x}), P_r(y\|\mathbf{x}))$
	20. Geometry Score [47]	• Compares geometrical properties of the underlying data manifold between real and generated data.
	21. Reconstruction Error [48]	• Measures the reconstruction error (*e.g.* L_2 norm) between a test image and its closest generated image by optimizing for z (*i.e.* $min_{\mathbf{z}}\|\|G(\mathbf{z}) - \mathbf{x}^{(test)}\|\|^2$)
	22. Image Quality Measures [49, 50, 51]	• Evaluates the quality of generated images using measures such as SSIM, PSNR, and sharpness difference
	23. Low-level Image Statistics [52, 53]	• Evaluates how similar low-level statistics of generated images are to those of natural scenes in terms of mean power spectrum, distribution of random filter responses, contrast distribution, etc.
	24. Precision, Recall and F_1 score [23]	• These measures are used to quantify the degree of overfitting in GANs, often over toy datasets.
Qualitative	1. Nearest Neighbors	• To detect overfitting, generated samples are shown next to their nearest neighbors in the training set
	2. Rapid Scene Categorization [18]	• In these experiments, participants are asked to distinguish generated samples from real images in a short presentation time (*e.g.* 100 ms); *i.e.* real v.s fake
	3. Preference Judgment [54, 55, 56, 57]	• Participants are asked to rank models in terms of the fidelity of their generated images (*e.g.* pairs, triples)
	4. Mode Drop and Collapse [58, 59]	• Over datasets with known modes (*e.g.* a GMM or a labeled dataset), modes are computed as by measuring the distances of generated data to mode centers
	5. Network Internals [1, 60, 61, 62, 63, 64]	• Regards exploring and illustrating the internal representation and dynamics of models (*e.g.* space continuity) as well as visualizing learned features

Pros and cons of GAN evaluation measures

https://arxiv.org/abs/1802.03446

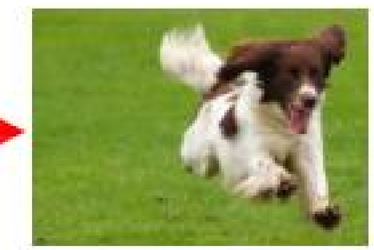
Conditional Generation

Text-to-Image

a dog is running 

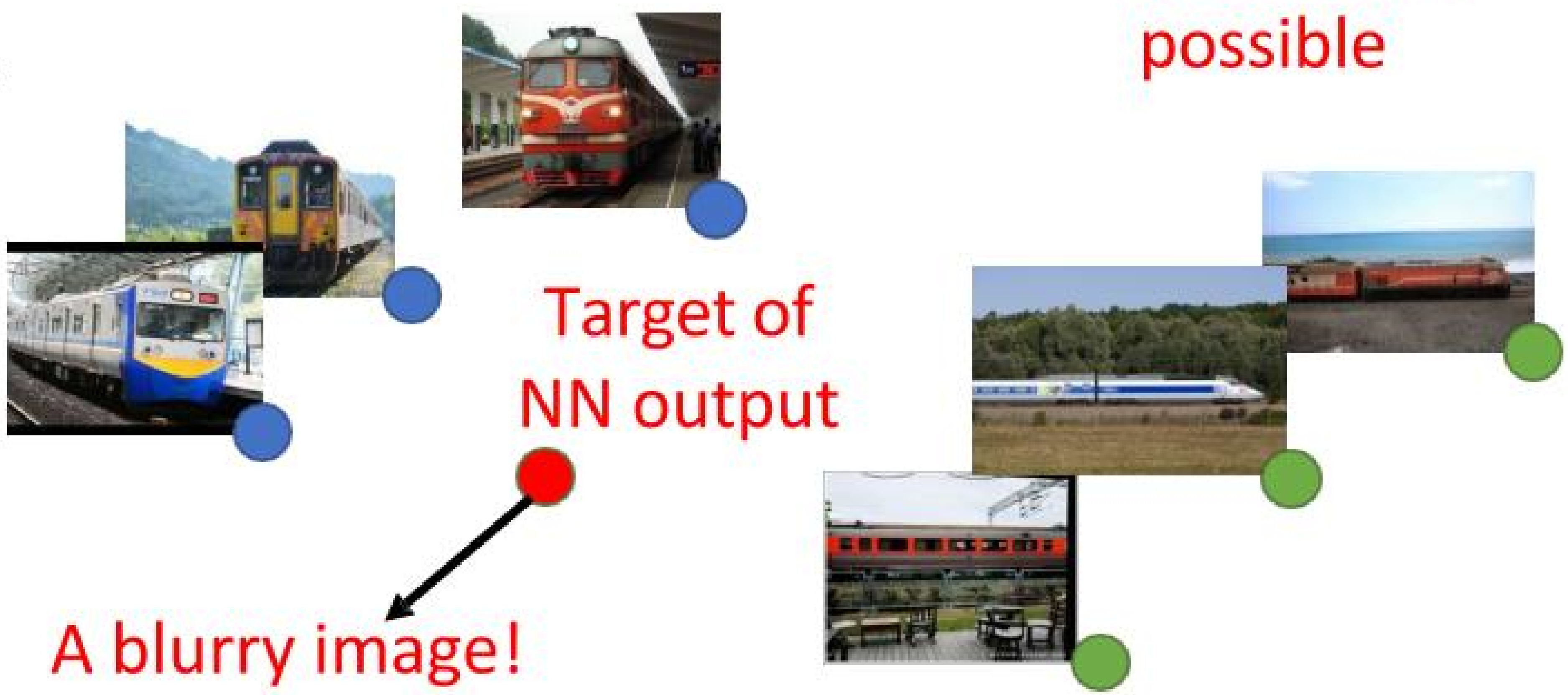
a bird is flying 

- **Traditional supervised approach**

c^1: a dog is running → NN → Image ↔ 

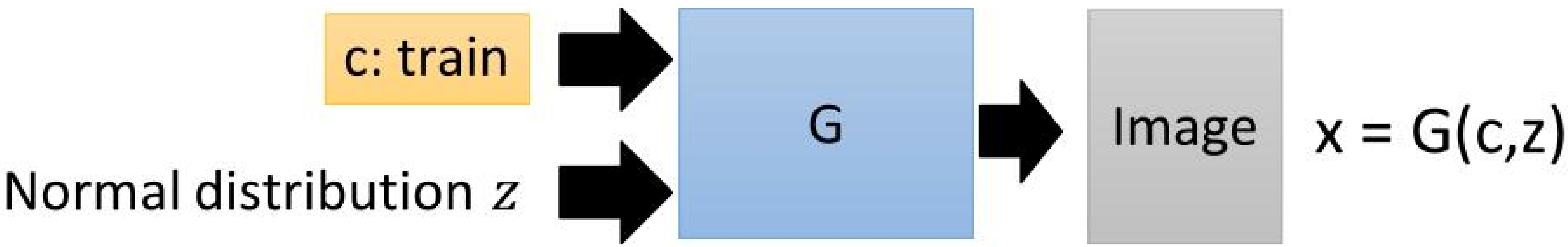
as close as possible

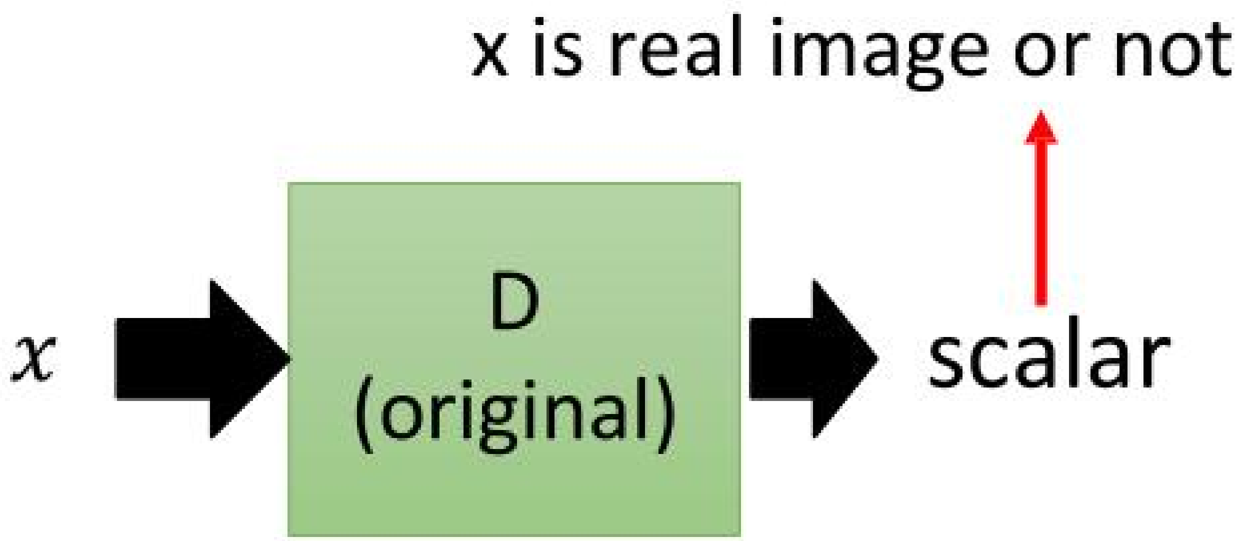
Text: “train”



[Scott Reed, et al, ICML, 2016]

Conditional GAN





Generator will learn to generate realistic images ….

But completely ignore the input conditions.

Real images:  1

Generated images: Image 0

[Scott Reed, et al, ICML, 2016]

Conditional GAN

c: train → G ← Normal distribution z

G → Image $x = G(c,z)$

c → D (better) ← x

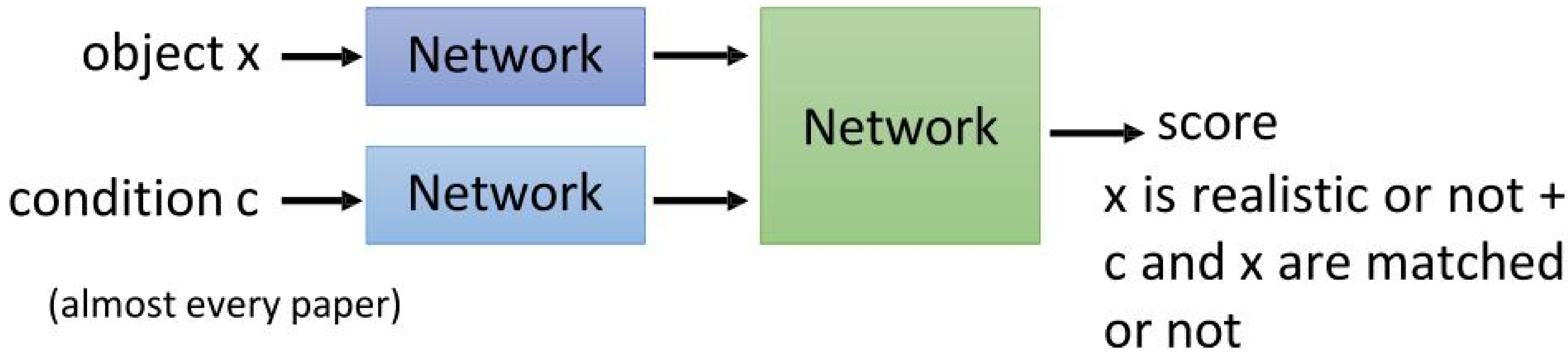
D (better) → scalar

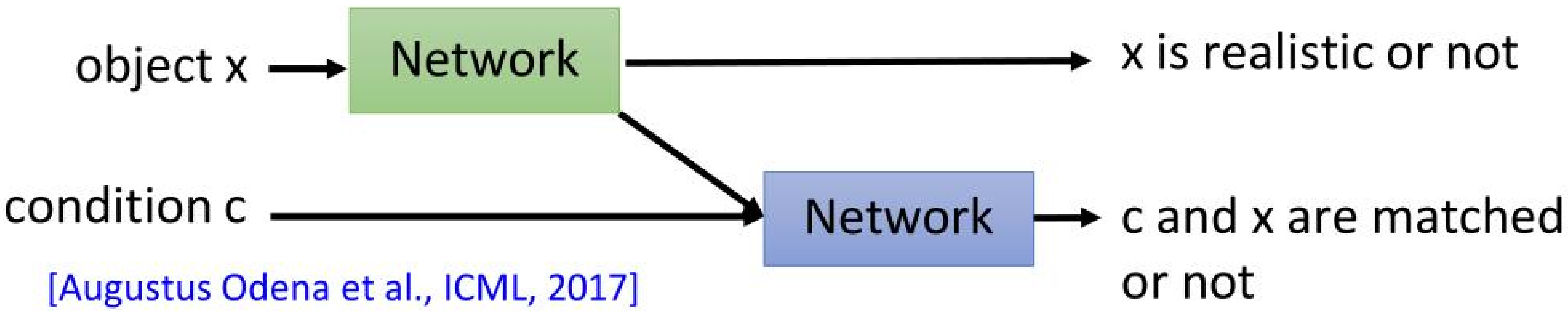
x is realistic or not +
c and x are matched or not

True text-image pairs: (train , ) 1

(cat , ) 0 (train , Image) 0

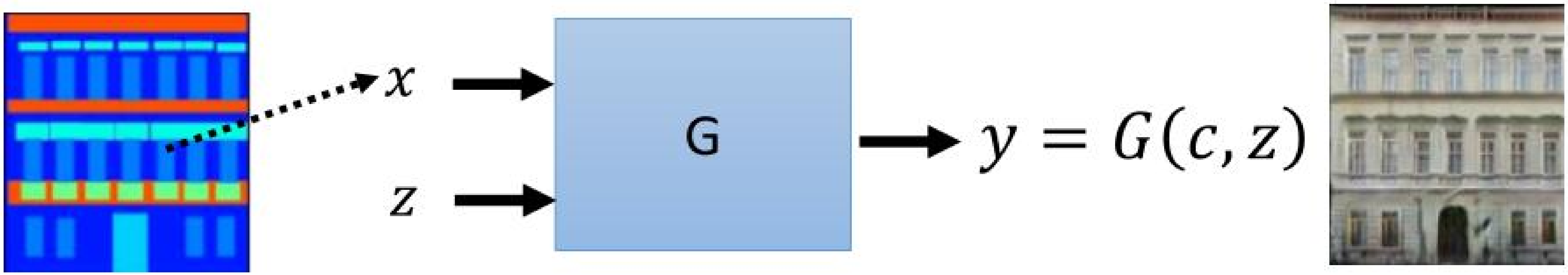
Conditional GAN - Discriminator





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

Conditional GAN



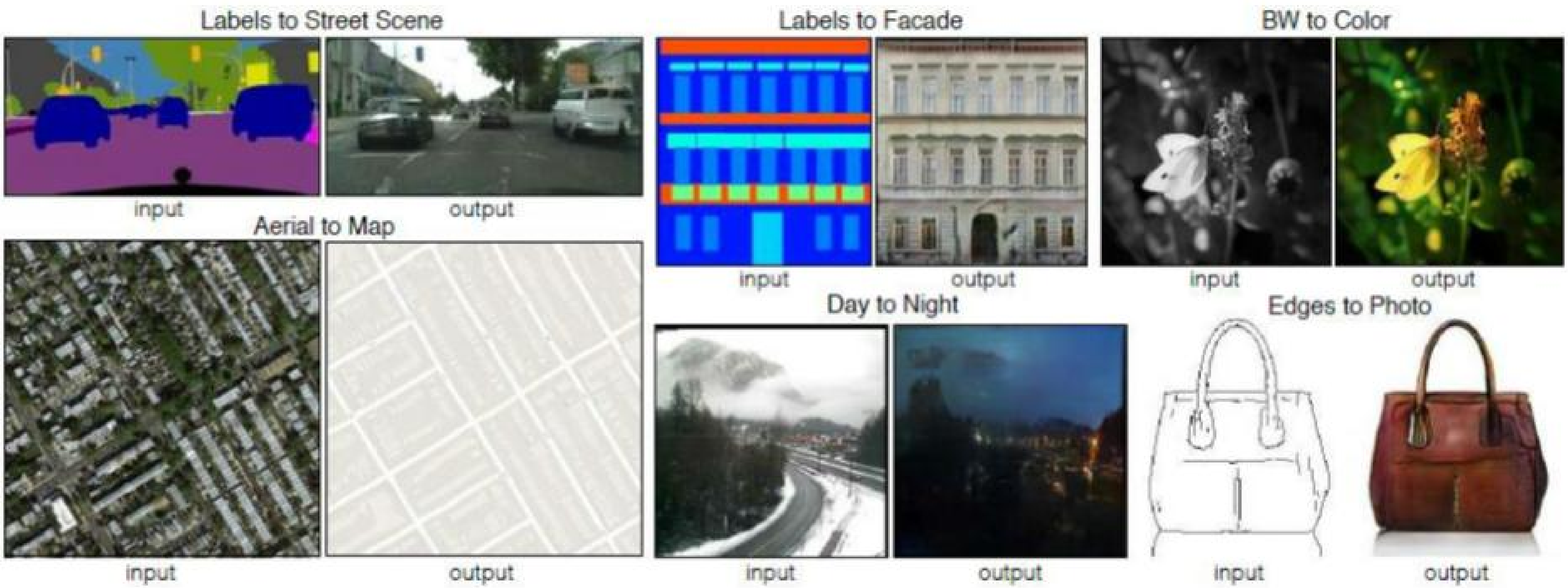
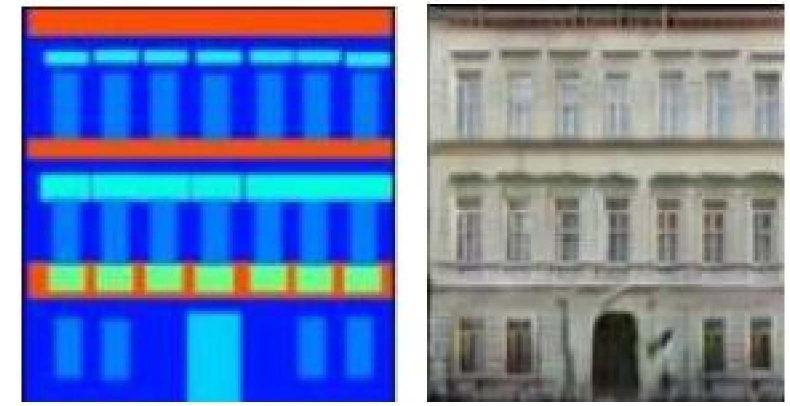
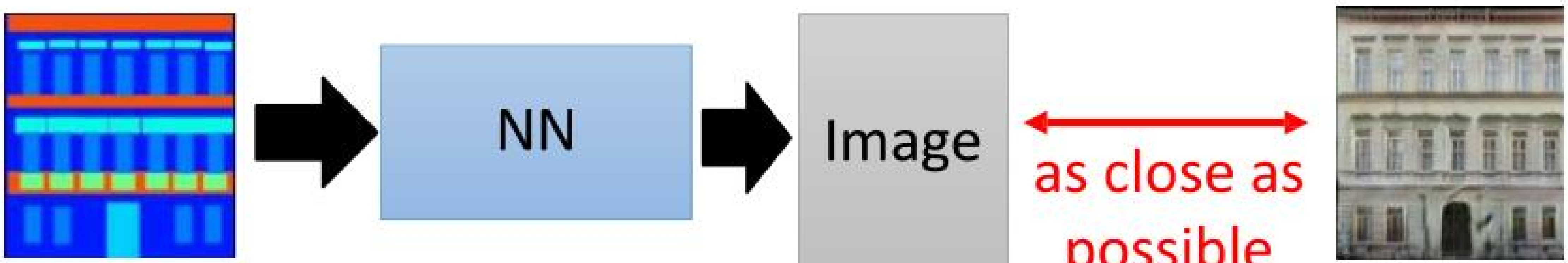


Image translation, or pix2pix

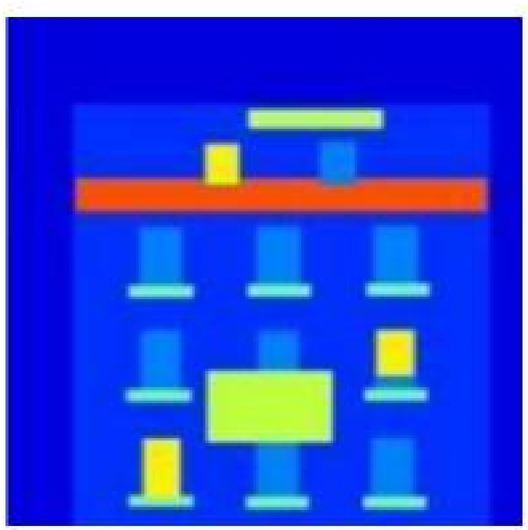
Image-to-image

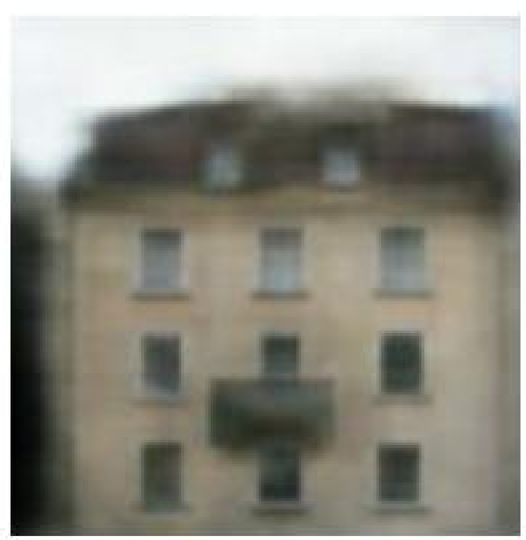


- Traditional supervised approach



Testing:

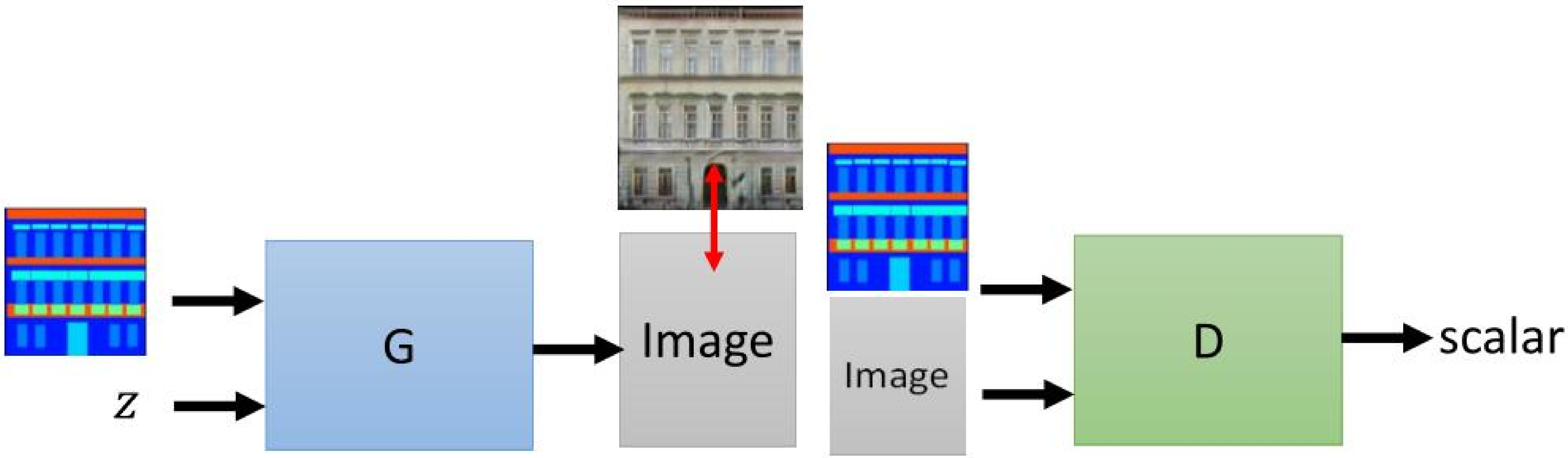

input


close

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

https://arxiv.org/abs/1611.07004

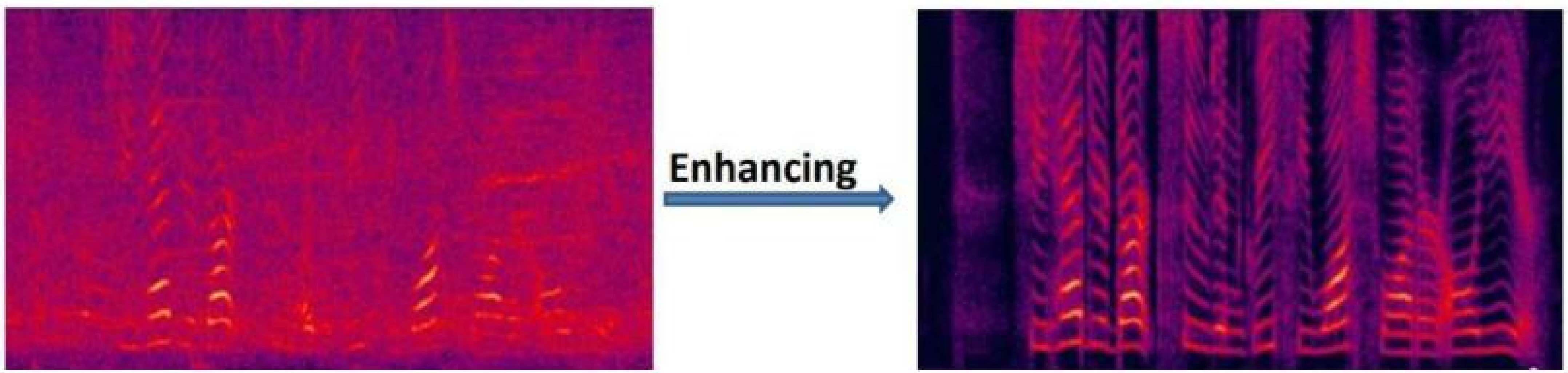
Conditional GAN



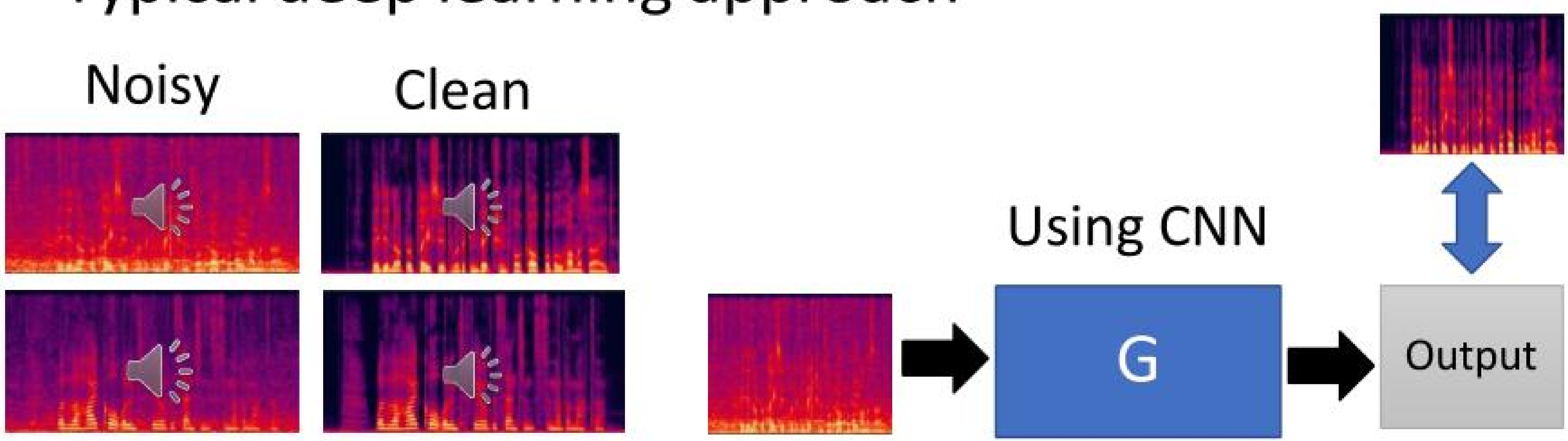
Testing:

input supervised GAN GAN + supervised

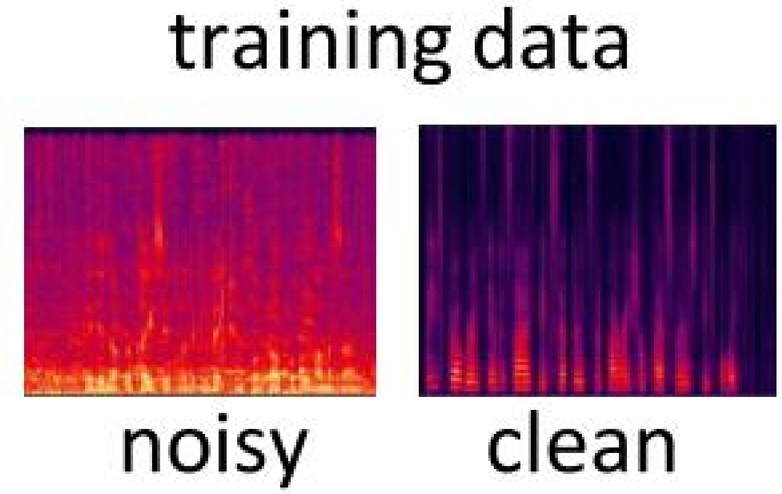
Speech Enhancement



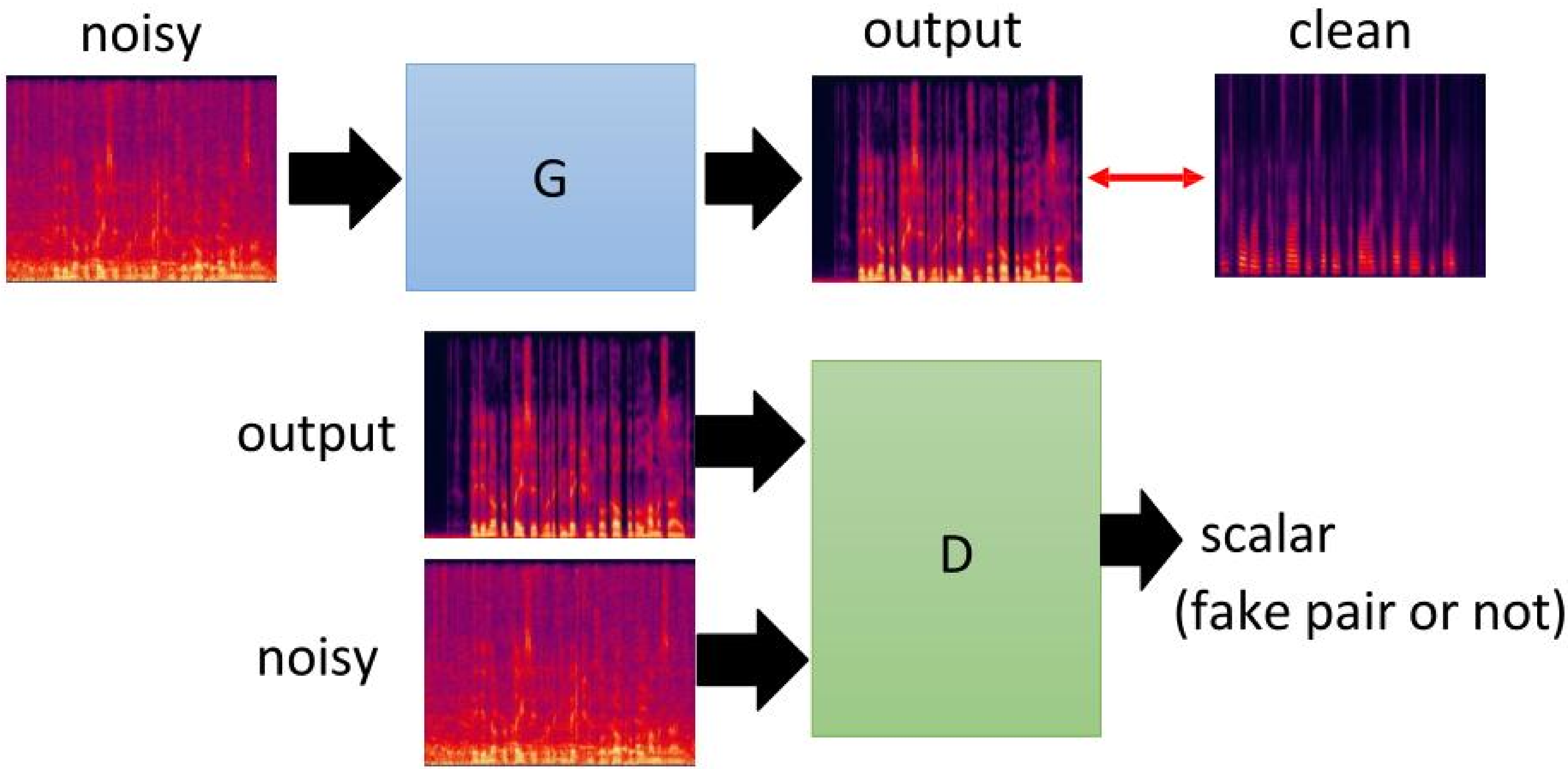
- Typical deep learning approach



Speech Enhancement



- Conditional GAN



Conditional GAN

Talking Head Generation

https://arxiv.org/abs/1905.08233

Concluding Remarks

- Introduction of Generative Models
- Generative Adversarial Network (GAN)
- Can Generator learn by itself?/Can Discriminator generate?
- Theory behind GAN
- Tips for GAN
- Evaluation of Generative Models
- Conditional Generation


Q&A