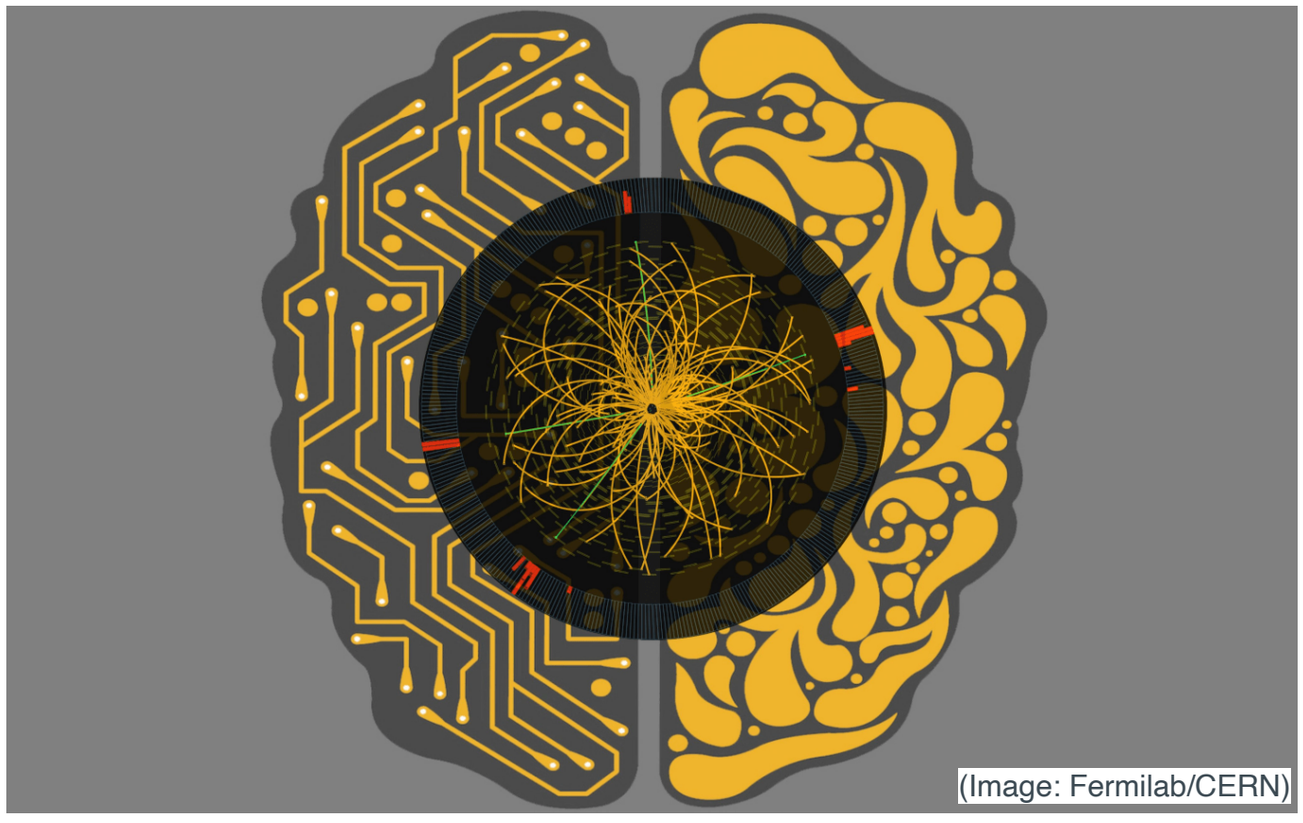
PHY 835: Collider Physics Phenomenology

Machine Learning in Fundamental Physics

Gary Shiu, UW-Madison



Lecture 16: Generative Adversarial Networks

Recap of Lecture 15

- Autoencoder
- Variational methods and Mean Field Theory (MFT)
- Expectation-Maximization (EM)

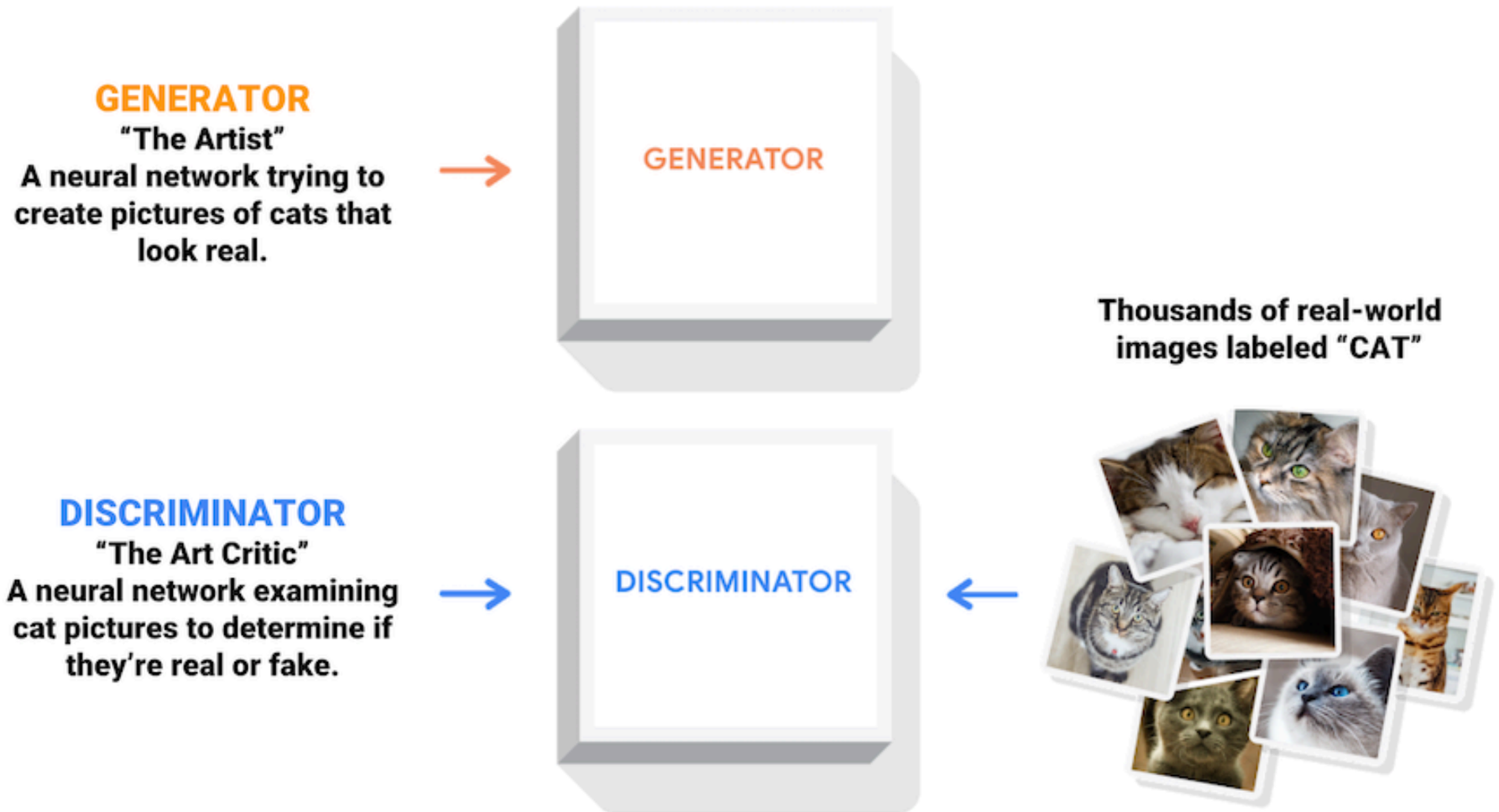
Outline for today

- Generative Adversarial Networks (GANs)
- Limitations of Maximizing Likelihood
- Adversarial Learning
- Wasserstein Loss and WGAN

References: 1803.08823, Deep Learning Book
https://www.tensorflow.org/tutorials/generative/dcgan

Generative Adversarial Networks (GANs)

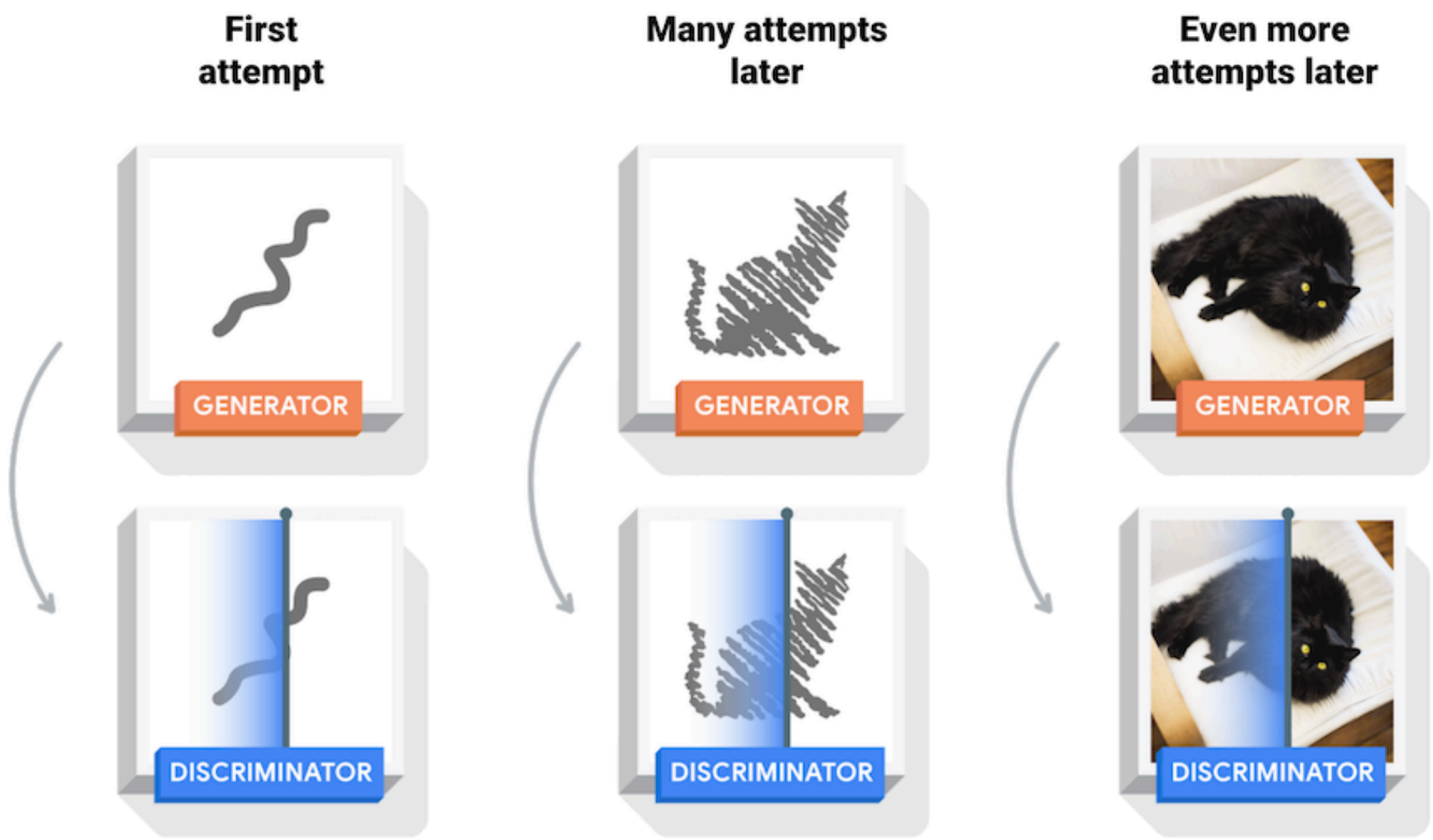
- Differential network, can be trained with backpropagation methods
- Idea: train two neural networks known as Generator G and Discriminator D that compete against each other.



Generative Adversarial Networks (GANs)

- During training, G progressively becomes better at creating images that look real, while D becomes better at telling them apart.

- The process reaches equilibrium (Nash equilibrium) when the discriminator can no longer distinguish real images from fakes.



GAN and VAE

- GANs and Variational Autoencoder (VAE) are generative models that do not directly seek to maximize likelihood.

- Numerous applications, e.g.,

    - Jet substructure in QCD: https://arxiv.org/abs/1808.08979;

    - Ising Model: https://arxiv.org/abs/1710.04987;

    - Many-body quantum systems: https://arxiv.org/abs/1710.00725;

    - Phase identification: https://arxiv.org/abs/1703.02435;

    - Galaxy images for dark energy science: https://arxiv.org/abs/1609.05796

- See a notebook on how to use GAN on MNIST dataset: https://www.tensorflow.org/tutorials/generative/dcgan

Real or Fake?

Figure 10: Top: Our CELEBA-HQ results. Bottom: The nearest neighbors found from the training data, using the center crop (half vertically, half horizontally) and L_1 distance in pixel space.

https://research.nvidia.com/sites/default/files/pubs/2017-10_Progressive-Growing-of/karras2018iclr-paper.pdf

https://www.youtube.com/watch?v=XOxxPcy5Gr4

Limitations of Maximizing Likelihood

- KL divergence measures the similarity between two probability distributions; not a distance since $D_{KL}(p\,||\,q) \neq D_{KL}(q\,||\,p)$:

$$D_{KL}(p||q) = \int \mathrm{d}\mathbf{x} p(\mathbf{x}) \log \frac{p(\mathbf{x})}{q(\mathbf{x})}$$
$$D_{KL}(q \parallel p) = \int \mathrm{d}\mathbf{x} q(\mathbf{x}) \log \frac{q(\mathbf{x})}{p(\mathbf{x})}.$$

- A related quantity is the Jensen-Shannon divergence:

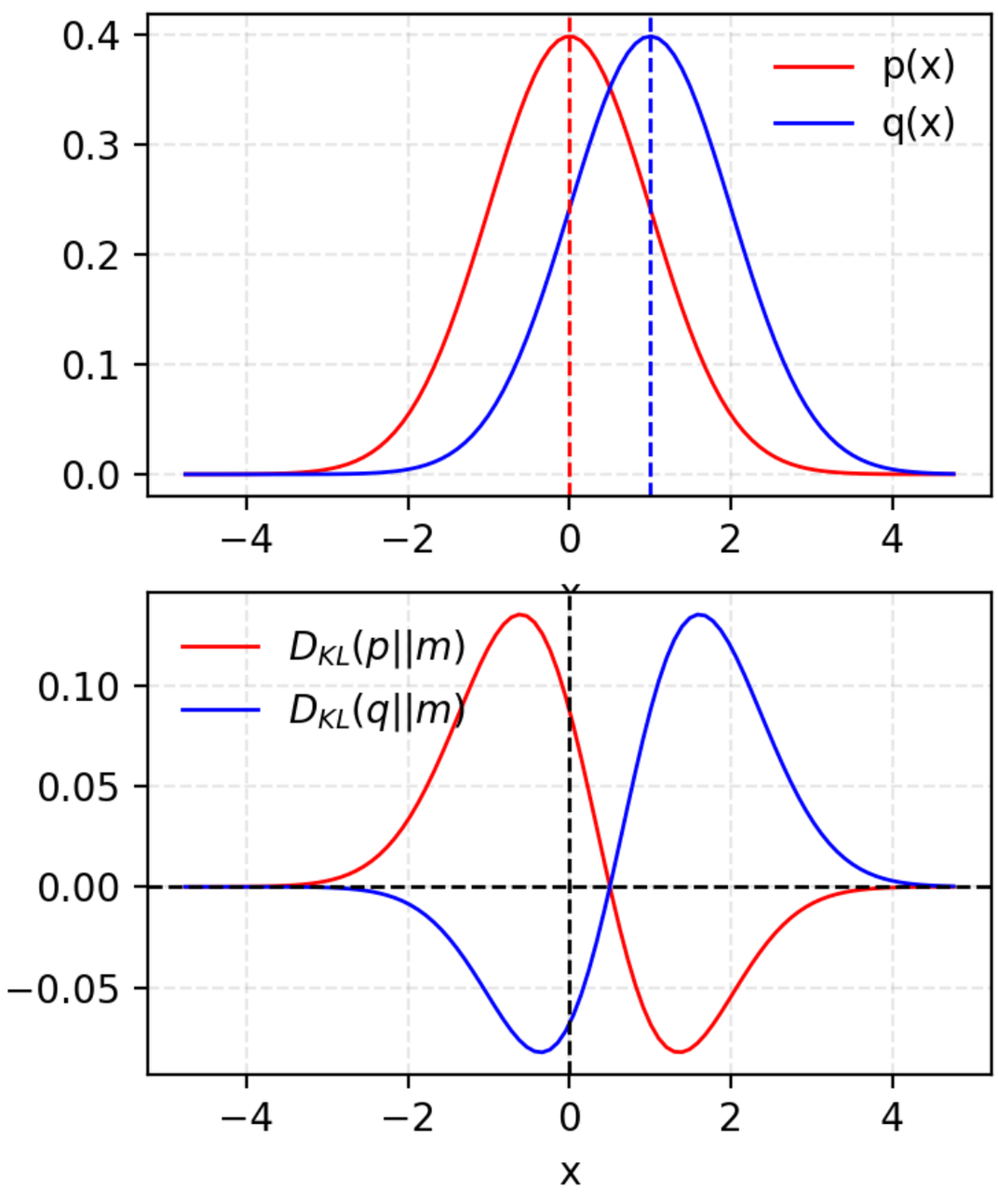
$$D_{JS}(p, q) = \frac{1}{2}\left[D_{KL}\left(p \middle\| \frac{p+q}{2}\right) + D_{KL}\left(q \middle\| \frac{p+q}{2}\right)\right]$$

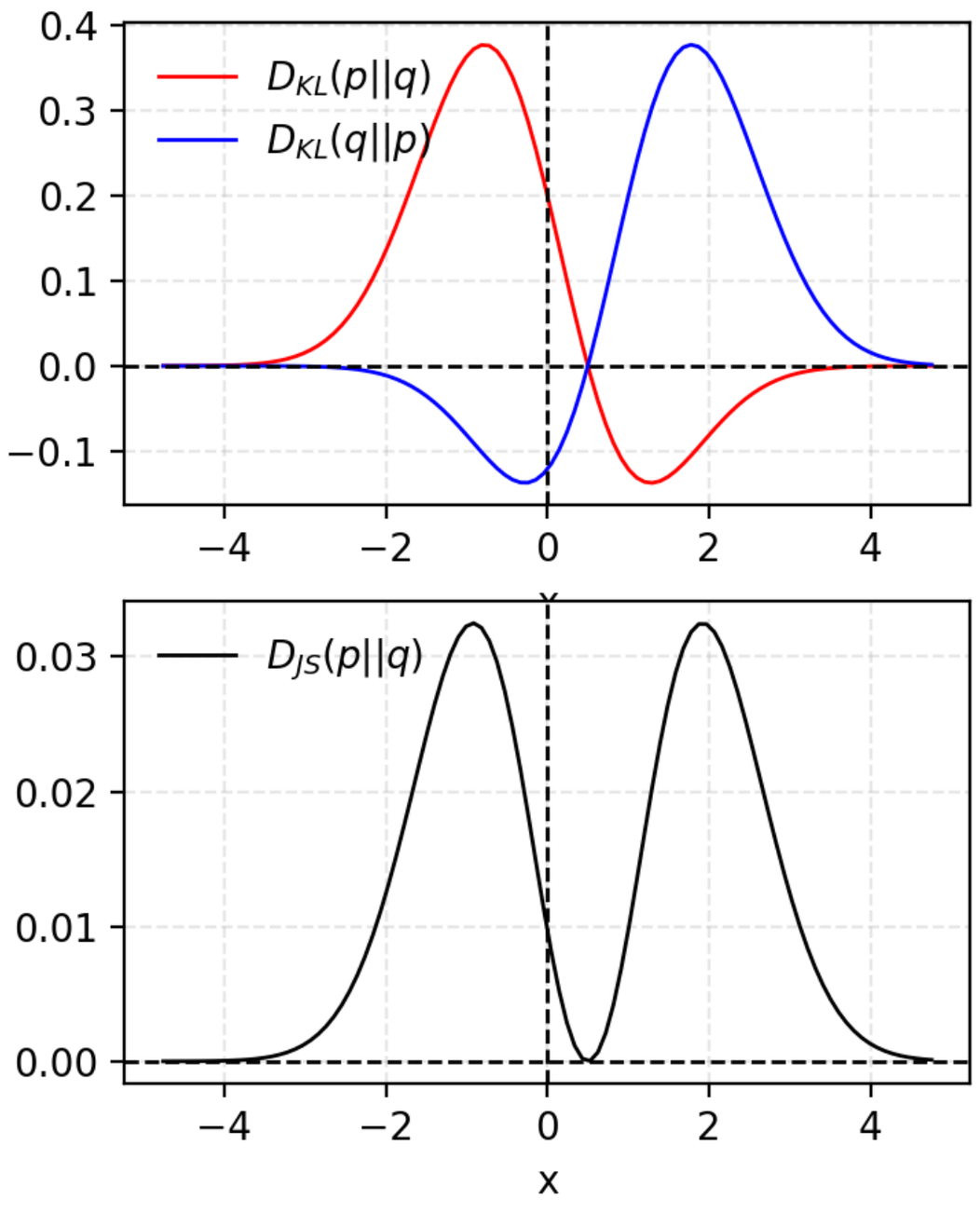
which satisfies all the properties of a squared metric (see next page).

- Maximizing likelihood = minimizing KL divergence:

$$D_{KL}(p_{\text{data}}||p_\theta) = \int \mathrm{d}\mathbf{x} p_{\text{data}}(\mathbf{x}) \log p_{\text{data}}(\mathbf{x}) - \int \mathrm{d}\mathbf{x} p_{\text{data}}(\mathbf{x}) \log p_\theta(\mathbf{x}) = -S[p_{\text{data}}] - \langle \log p_\theta(\mathbf{x}) \rangle_{\text{data}}$$

Limitations of Maximizing Likelihood





$$m = (p + q)/2$$

Limitations of Maximizing Likelihood

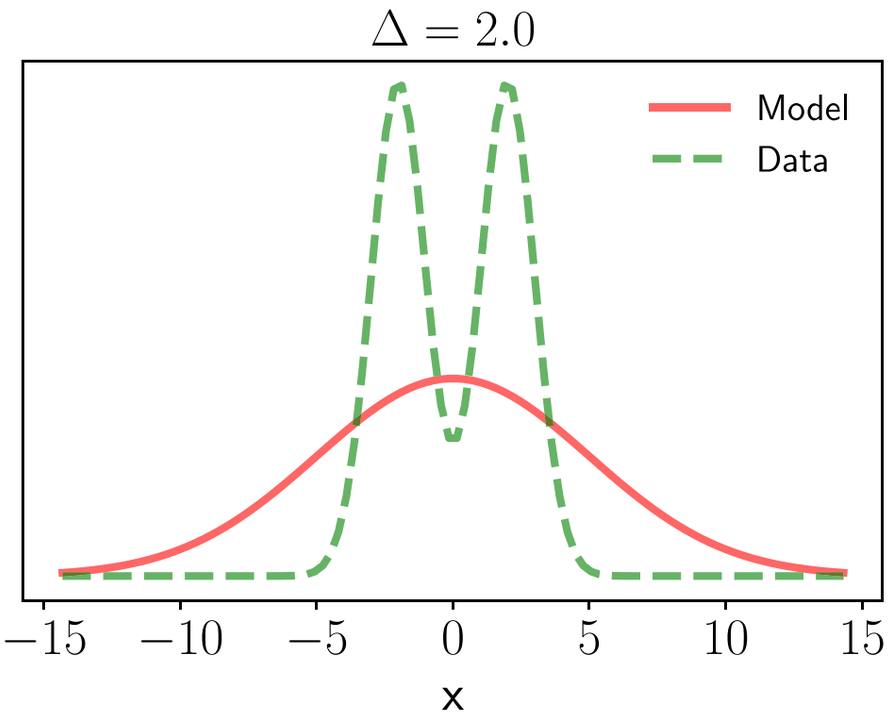
- The equivalence follows from $D_{KL} \geq 0$ and $S[p_{\text{data}}]$ is θ-independent:

$$\langle \log p_\theta(\mathbf{v}) \rangle_{\text{data}} = -S[p_{\text{data}}] - D_{KL}(p_{\text{data}} || p_\theta)$$

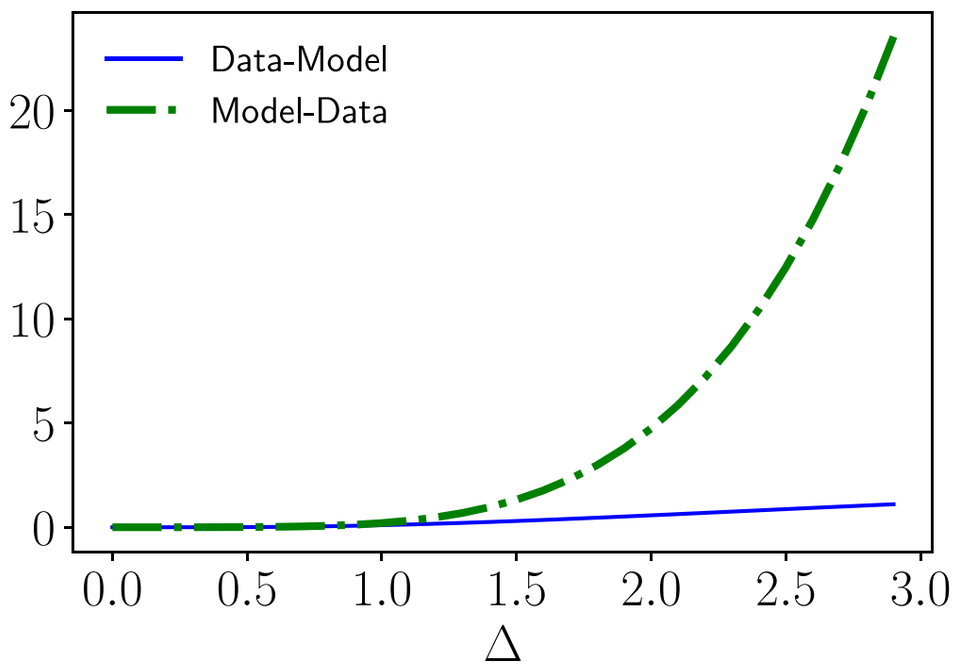
- The original formulation of GANs minimizes the Jensen-Shannon divergence (but other measures e.g. Wasserstein distance are used).
- $D_{KL}(p_{\text{data}} || p_\theta)$ and $D_{KL}(p_\theta || p_{\text{data}})$ measure similarities between the two distributions, but they are sensitive to very different things.
- $D_{KL}(p_\theta || p_{\text{data}})$ is insensitive to setting $p_\theta \approx 0$ even when $p_{\text{data}} \neq 0$ while $D_{KL}(p_{\text{data}} || p_\theta)$ punishes this harshly.
- Likelihood-based methods may fail by improperly “filling in” low probability density regions between peaks in the data distribution.

Limitations of Maximizing Likelihood

- The Jensen–Shannon divergence which underlies GANs is sensitive **both** to placing weight where there is data since it has information about $D_{KL}(p_{\text{data}}||p_\theta)$ and to not placing weight where no data has been observed since it has information about $D_{KL}(p_\theta||p_{\text{data}})$.

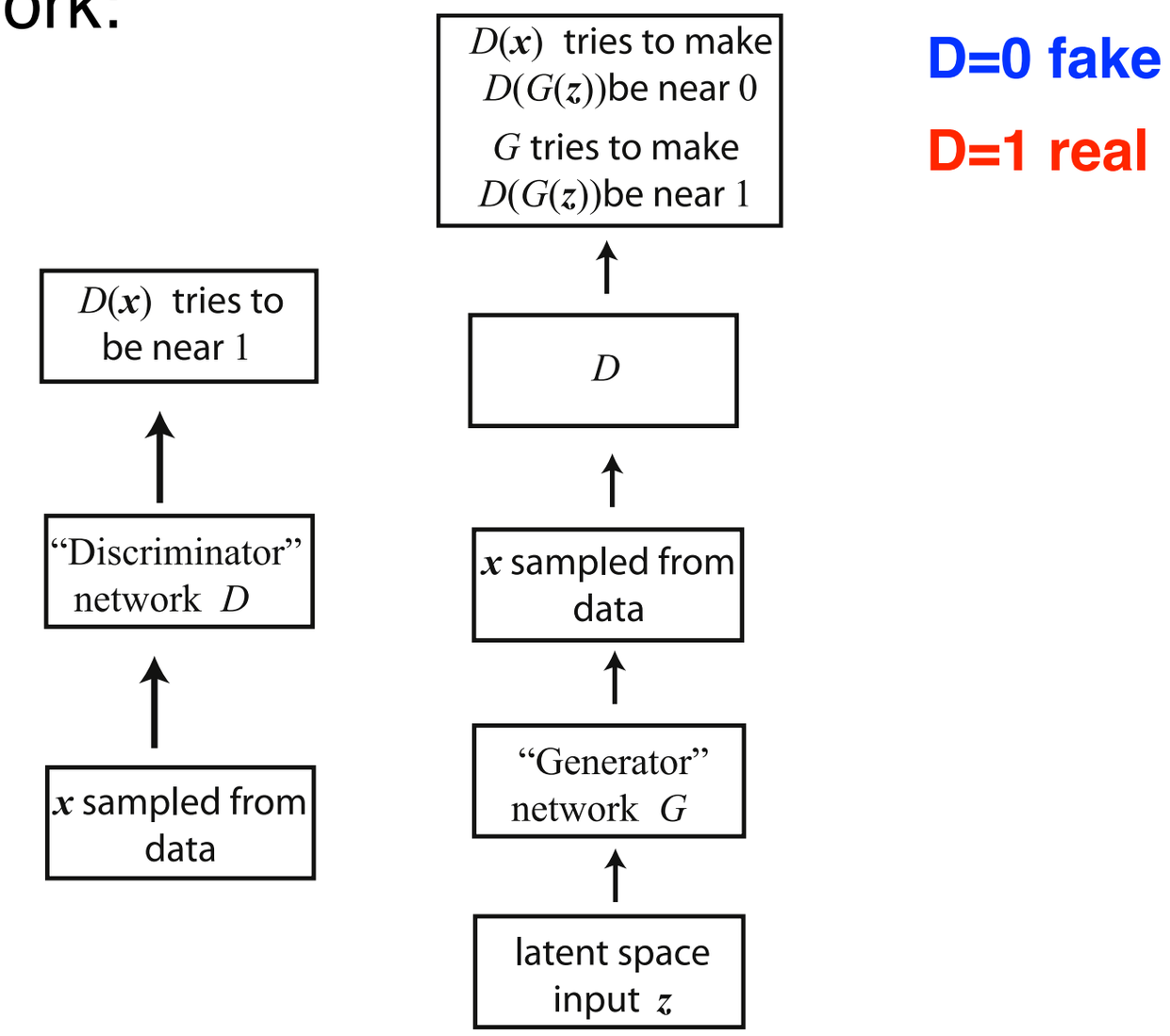


- **Adversarial learning:** teaches a discriminator network to distinguish between real data points and samples generated from the model. By punishing the model for generating points that can be easily discriminated from the data, adversarial learning decreases the weight of regions in the model space that are far away from data points.



Adversarial Learning

- The theory of GANs draws deeply from concepts in Game Theory such as Nash Equilibrium.
- Hard to train, see e.g. tutorial: https://github.com/soumith/ganhacks
- Two differential network:



Adversarial Learning: Cost Function

- To define the cost functions, first introduce the following functional:

$$V(D, G) = \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}} \left(\log D(\mathbf{x})\right) + \mathbb{E}_{\mathbf{z} \sim p_{\text{prior}}} \left(\log \left[1 - D(G(\mathbf{z}))\right]\right)$$

- The first term is the log probability of D predicting that real data is indeed real, while the second term is the log probability of D predicting that the generator's data $D(G(z))$ is indeed generated.

- The cost functions for the discriminator and generator are:

$$C_D = V(D, G) = -C_G$$

generator designed to do the opposite

- Training: alternate between generator and discriminator (keep weights of the other network fixed during training).

Adversarial Learning: Cost Function

- In terms of the real and generated distributions $\rho_r(x)$ and $\rho_g(x)$:

$$V(D,G) = \mathbb{E}_{x\sim p_{\text{data}}} \log(D(x)) + \mathbb{E}_{z\sim p_{\text{prior}}} \log[1 - D(G(z))]$$

$$= \int dx\ \rho_r(x)\ \log(D(x)) + \int dz\ \rho_{\text{prior}}(z)\ \log[1 - D(G(z))]$$

$$= \int dx\ \rho_r(x)\ \log(D(x)) + \int dx\ \rho_g(x)\ \log[1 - D(x)]$$

- What is the optimal D for fixed generator?
- Extremizing $f(y) = a\log y + b\log(1-y)$ gives $y = a/(a+b)$.
- After successful training, $\rho_r = \rho_g$ and so $y = 1/2$. As a result

$$V(D,G) = \log\left(\frac{1}{2}\right)\left[\int dx\rho_r(x) + \int dx\rho_g(x)\right] = -2\log 2$$

Adversarial Learning: Cost Function

- What does the loss function represent (in this idealized setting)?

$$D_{JS}(\rho_r||\rho_g) = \frac{1}{2}D_{KL}(\rho_r||\frac{\rho_r+\rho_g}{2}) + \frac{1}{2}D_{KL}(\rho_g||\frac{\rho_r+\rho_g}{2})$$

$$= \frac{1}{2}\left(\log 2 + \int dx \rho_r \log\frac{\rho_r}{\rho_r+\rho_g}\right) + \frac{1}{2}\left(\log 2 + \int dx \rho_g \log\frac{\rho_g}{\rho_r+\rho_g}\right)$$

$$= \frac{1}{2}\left(\log 4 + V(D,G)\right)$$

$$\rightarrow 0 \text{ for ideaized training}$$

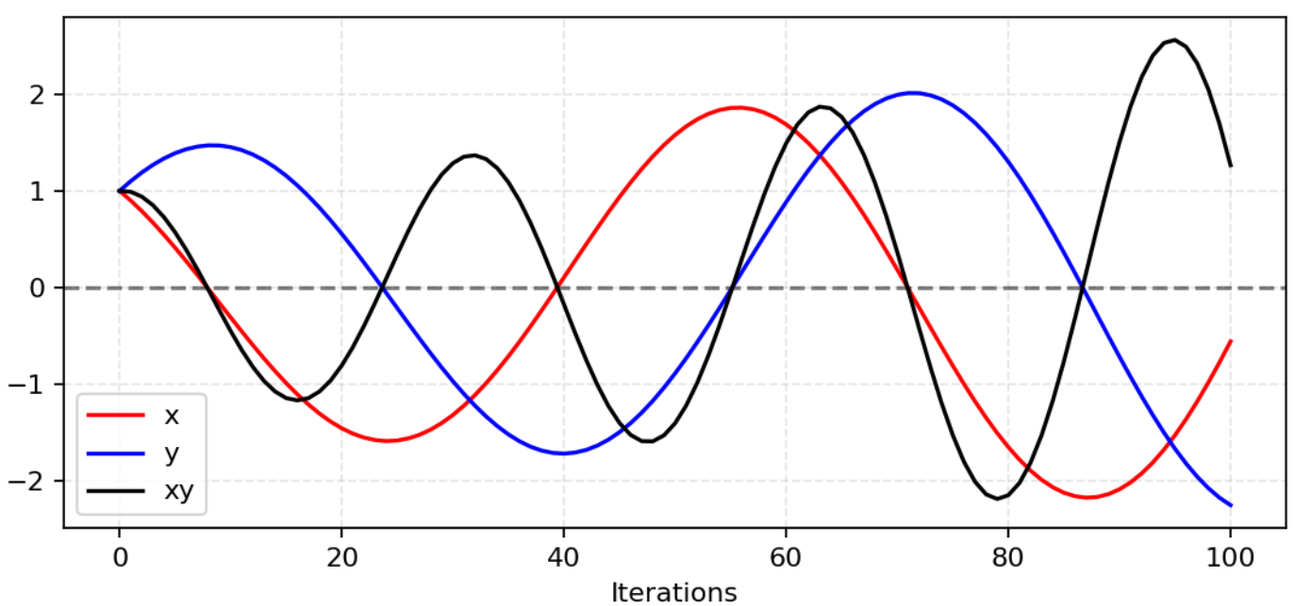
- In this limit, loss of GAN quantifies similarity between generator data distribution ρ_g & real data distribution ρ_r by evaluating JS divergence.

GANs: Training

- Alternating training of generator and discriminator.
- Intuition: discriminator and generator get better at the same speed, the results of the generator get more accurate and the discriminator stronger to distinguish real from fake samples.
- Keep generator (respectively discriminator) constant during training of the respective other network.
- Perfect success of generator: 50% success rate in discriminator (discriminator only guessing).
- Problem: discriminator feedback less useful and generator might not improve anymore (or even collapse).

GANs: Problems

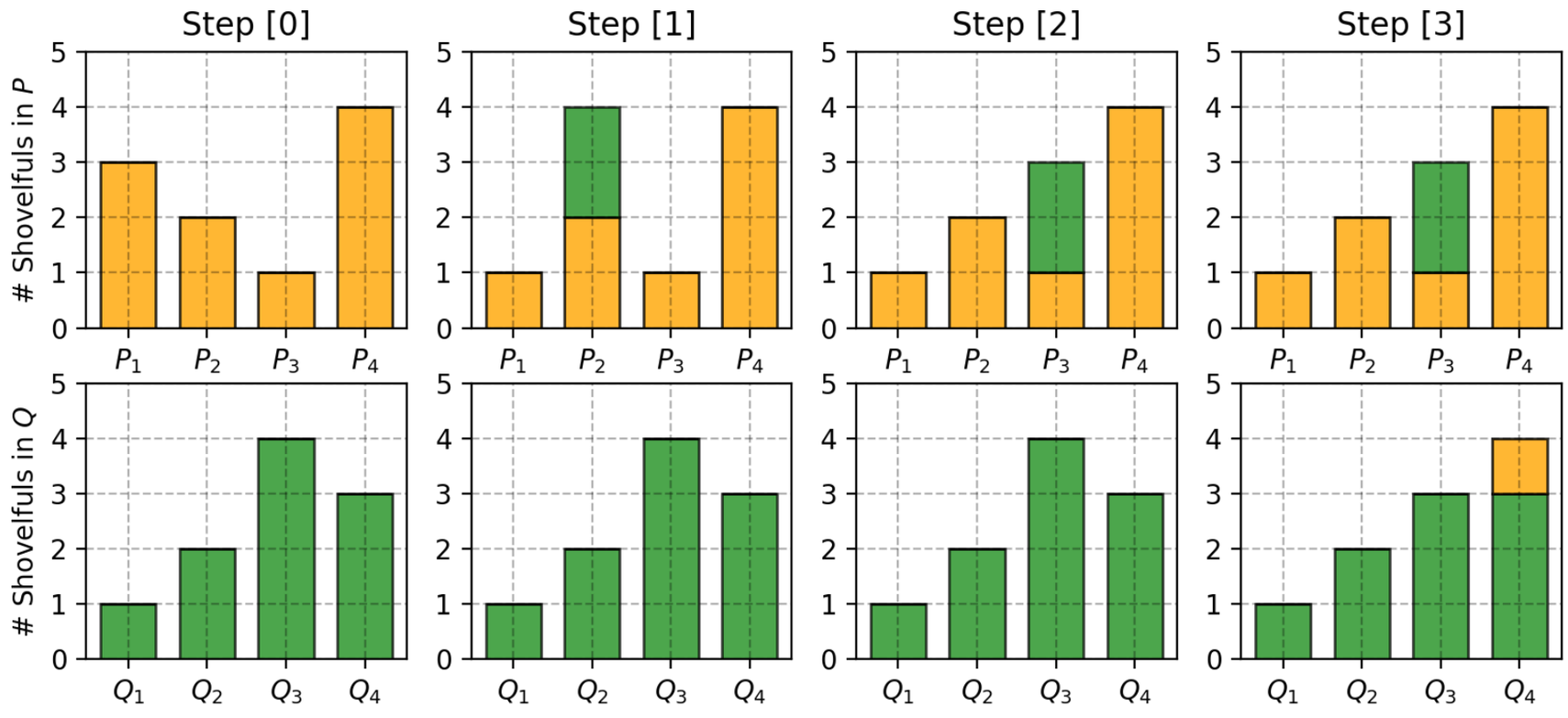
- **Vanishing gradients:** discriminator or generator too good. Modify loss to train one network faster than the other, vary strength of respective networks.
- **Mode collapse:** generator provides always the same output which might be a perfect example. Generator finds a way to trick the discriminator. Modify loss to make it harder for generator to trick discriminator (unrolled GANs)
- No convergence to Nash equilibrium (zero sum game) guaranteed: $f(x, y) = xy$.



- Intrinsically no proper evaluation metric. When is performance good?

Wasserstein Loss

- Different measure to quantify the difference between data and model distribution: earth-mover distance (Wasserstein distance).
- Idea: Earth-mover distance = Minimum energy cost to transform a pile of dirt from one shape to another. Cost = Amount x Distance.
- Consider an example where the probability domain is discrete.



$$P_1 = 3, P_2 = 2, P_3 = 1, P_4 = 4$$
$$Q_1 = 1, Q_2 = 2, Q_3 = 4, Q_4 = 3$$

Cost to make P_i and Q_i match $\equiv \delta_i$

$$\delta_{i+1} = \delta_i + P_i - Q_i$$

$$\delta_0 = 0$$
$$\delta_1 = 0 + 3 - 1 = 2$$
$$\delta_2 = 2 + 2 - 2 = 2$$
$$\delta_3 = 2 + 1 - 4 = -1$$
$$\delta_4 = -1 + 4 - 3 = 0$$

$$W = \sum |\delta_i| = 5.$$

Wasserstein Loss

- When dealing with the continuous probability domain, the distance formula becomes:

$$W(p_r, p_g) = \inf_{\gamma \sim \Pi(p_r, p_g)} \mathbb{E}_{(x,y)\sim\gamma}[\|x - y\|]$$

- $\pi(p_r, p_g)$ is the set of all possible joint probability distributions between ρ_r and ρ_g, and when marginalized over x and y gives:

$$\sum_x \gamma(x, y) = p_g(y) \qquad \sum_y \gamma(x, y) = p_r(x).$$

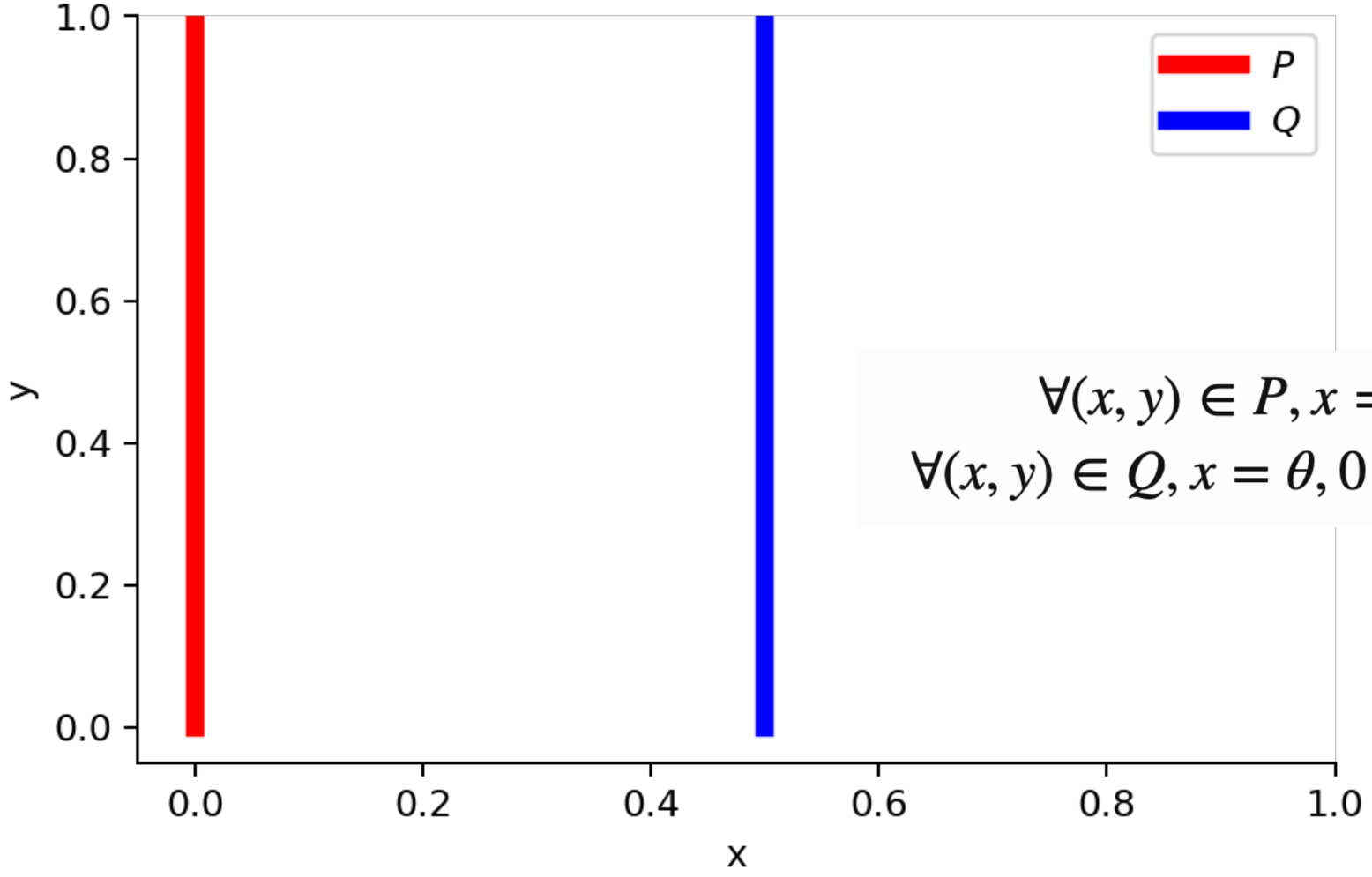
- Each $\gamma \in \Pi(p_r, p_g)$ describes one dirt transport plan: $\gamma(x, y)$= percentage of dirt should be transported from point x to point y so as to make x follows the same probability distribution of y.

- Cost = amount x distance = $\gamma(x, y)||x - y||$. Expected cost averaged across all the (x, y) pairs:

$$\sum_{x,y} \gamma(x, y)\|x - y\| = \mathbb{E}_{x,y\sim\gamma}\|x - y\|$$

WGAN

- Even when two distributions are located in lower dimensional manifolds without overlaps, Wasserstein distance can still provide a meaningful and smooth representation of the distance in-between.
- Consider the following two distributions P and Q:



$$\forall (x, y) \in P, x = 0 \text{ and } y \sim U(0, 1)$$
$$\forall (x, y) \in Q, x = \theta, 0 \leq \theta \leq 1 \text{ and } y \sim U(0, 1)$$

WGAN

- There is no overlap between P and Q when $\theta \neq 0$:

$$D_{KL}(P\|Q) = \sum_{x=0, y\sim U(0,1)} 1 \cdot \log \frac{1}{0} = +\infty$$
$$D_{KL}(Q\|P) = \sum_{x=\theta, y\sim U(0,1)} 1 \cdot \log \frac{1}{0} = +\infty$$
$$D_{JS}(P, Q) = \frac{1}{2}\left(\sum_{x=0, y\sim U(0,1)} 1 \cdot \log \frac{1}{1/2} + \sum_{x=0, y\sim U(0,1)} 1 \cdot \log \frac{1}{1/2}\right) = \log 2$$
$$W(P, Q) = |\theta|$$

- When $\theta = 0$, the two distributions are fully overlapped:

$$D_{KL}(P\|Q) = D_{KL}(Q\|P) = D_{JS}(P, Q) = 0$$
$$W(P, Q) = 0 = |\theta|$$

- D_{KL} gives us infinity when two distributions are disjoint. The value of D_{JS} has sudden jump, not differentiable at $\theta = 0$. Only Wasserstein metric provides a smooth measure, which is helpful for a stable learning process using gradient descents.

Summary

- Generative Adversarial Networks (GANs)
- Limitations of Maximizing Likelihood
- Adversarial Learning
- Wasserstein Loss and WGAN