Theoretical Aspects of Generative Adversarial Nets (GAN)

NIC Lab

Wei-Ning Chen (wnchen@ntu.edu.tw)

National Taiwan University

Outline

- **GAN**
  - Basic idea and minimax game
  - Equivalent JS divergence minimization
- **f-GAN : from JS divergence to f-divergence**
  - Variational Estimation
  - Practical difficulties
- **WGAN**
  - A better distance measure: Wasserstein distance
  - Dual representation

Recap: Generative Adversarial Nets[1]

- Real samples $X \sim P_{data}(x)$, synthetic samples $X \sim P_{G_\theta}(x)$
- $P_{G_\theta}(x)$ is constructed as distribution of $G_\theta(Z)$, where Z follows standard Gaussian and $\{G_\theta : \theta \in \Theta\}$ is a family of deterministic function
- Mathematically,

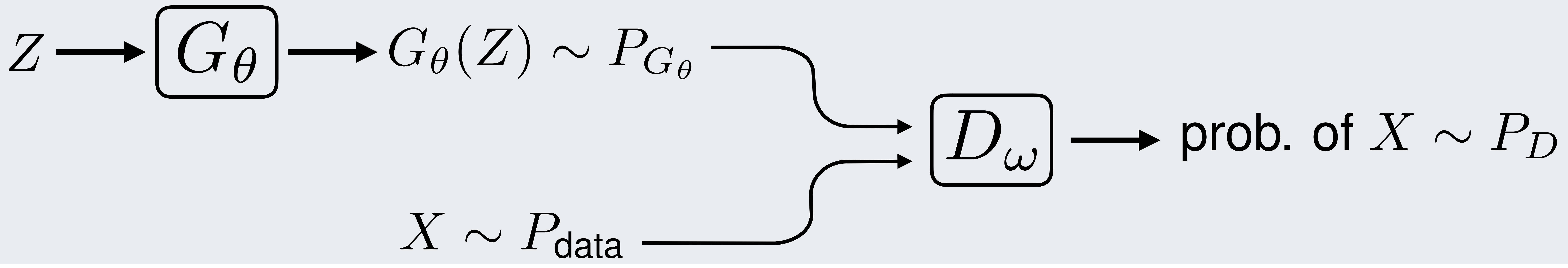
$$G_\theta : \mathcal{Z} \to \mathcal{X},\ P_{G_\theta}(x) = \mathcal{P}_Z \left\{ G_\theta^{-1}(x) \right\}.$$

- Typically $\{G_\theta\}$ is implemented by a neural network (in this case, θ represents weights of the networks)
- Gaol : find the "closest" P_{G_θ} to P_{data}

[1] Ian J. Goodfellow, et. al *"Generative Adversarial Nets,"* NIPS, 2014

Recap: Generative Adversarial Nets

- A first attempt : maximum likelihood estimator $\hat{\theta} = \underset{\theta}{\operatorname{argmax}} \sum_{i=1}^{n} \log P_{G_\theta}(x_i)$
  - Equivalent to minimize $D_{\text{KL}}\left(P_{\text{data}} || P_{G_\theta}\right)$
  - Impractical since computing P_{G_θ} involves inverting a neural network
- GAN : introduce another auxiliary discriminator
  - Compute $\min_\theta \max_\omega \mathbb{E}_{X \sim P_{data}}[\log D_\omega(X)] + \mathbb{E}_{X \sim P_{G_\theta}}[1 - \log D_\omega(X)]$



Recap: Generative Adversarial Nets

- A first attempt : maximum likelihood estimator $\hat{\theta} = \operatorname*{argmax}_{\theta} \sum_{i=1}^{n} \log P_{G_\theta}(x_i)$
  - Equivalent to minimize $D_{\text{KL}}\left(P_{\text{data}} || P_{G_\theta}\right)$
  - Impractical since computing P_{G_θ} involves inverting a neural network

What do we actually do by playing this minimax game ?

Recap: Generative Adversarial Nets

$$\boxed{\min_{\theta}\max_{\omega} \underbrace{\mathbb{E}_{X\sim P_{\text{data}}}[\log D_\omega(X)] + \mathbb{E}_{X\sim P_{G_\theta}}[1-\log D_\omega(X)]}_{V(G_\theta, D_\omega)}}$$

- Optimal discriminator (given generator being G_θ) : $D^*(x) = \dfrac{P_{\text{data}}(x)}{P_{\text{data}}(x) + P_{G_\theta}(x)}$
  - Assume $D^* \in \{D_\omega : \omega \in \Omega\}$

- Optimal generator :

$$\begin{aligned} G^* &= \operatorname*{argmin}_{G_\theta} V(G_\theta, D^*) \\ &= \operatorname*{argmin}_{G_\theta} \mathbb{E}_{P_{\text{data}}}\left[\log \frac{P_{\text{data}}(X)}{P_{\text{data}}(X) + P_{G_\theta}(X)}\right] + \mathbb{E}_{P_{G_\theta}}\left[\log \frac{P_{G_\theta}(X)}{P_{\text{data}}(X) + P_{G_\theta}(X)}\right] \\ &= \operatorname*{argmin}_{G_\theta} -2\log 2 + D_{\text{KL}}\left(P_{\text{data}} || \frac{P_{\text{data}} + P_{G_\theta}}{2}\right) + D_{\text{KL}}\left(P_{G_\theta} || \frac{P_{\text{data}} + P_{G_\theta}}{2}\right) \\ &= \operatorname*{argmin}_{G_\theta} D_{\text{JS}}\left(P_{\text{data}} || P_{G_\theta}\right) \end{aligned}$$

Recap: Generative Adversarial Nets

$$\boxed{\min_{\theta}\max_{\omega}\mathbb{E}_{X\sim P_{\text{data}}}[\log D_\omega(X)] + \mathbb{E}_{X\sim P_{G_\theta}}[1-\log D_\omega(X)]}$$

- Do we circumvent the difficulty to compute P_{G_θ}?
- Sampling from P_{G_θ} is easy
- By playing the game, we can find the optimal generator which minimizes the divergence to the real distribution
- Can this game-theoretic approach be utilized to minimize other divergence?

for *number of training iterations* **do**
  for k *steps* **do**
    Sample m noise samples $\{z_1, ..., z_m\}$ from standard normal P_Z ;
    Sample m examples (real data) $\{x_1, ..., x_m\}$ from $P_{\text{data}}(x)$;
    Update discriminator by ascending its stochastic gradient:

$$\nabla_\omega \frac{1}{m}\sum_{i=1}^{m}\left(\log D_\omega(x_i) + \log\left(1 - D_\omega\left(G_\theta\left(z_i\right)\right)\right)\right)$$

  end
  Sample m noise samples $\{z_1, ..., z_m\}$ from standard normal P_Z ;
  Update generator by descending its stochastic gradient:

$$\nabla_\theta \frac{1}{m}\sum_{i=1}^{m}\log\left(1 - D_\omega\left(G_\theta\left(z_i\right)\right)\right)$$

end

Algorithm 1: minibatch GAN

f-GAN [2]

- Traditional GAN minimizes the (empirical) JS divergence
- Our next goal is to minimize *f-divergence*

<u>Definition</u> : f-divergence

Let f be a convex function on $\mathbb{R}$ with $f(1) = 0$, and P, Q be two probability density on $\mathcal{X}$. Then the f-divergence between P, Q is defined as

$$D_f(P||Q) \triangleq \mathbb{E}_Q\left[f\left(\frac{P(X)}{Q(X)}\right)\right]$$

[2] Sebastian Nowozin, et. al *"f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization,"* NIPS, 2016

f-divergence

Definition : f-divergence

Let f be a convex function on $\mathbb{R}$ with $f(1) = 0$, and P, Q be two probability density on $\mathcal{X}$. Then the f-divergence between P, Q is defined as

$$D_f(P\|Q) \triangleq \mathbb{E}_Q\left[f\left(\frac{P(X)}{Q(X)}\right)\right]$$

Some common f-divergence

Name	$D_f(P\|Q)$	Generator $f(u)$	$T^*(x)$
Kullback-Leibler	$\int p(x)\log\frac{p(x)}{q(x)}\,\mathrm{d}x$	$u\log u$	$1+\log\frac{p(x)}{q(x)}$
Reverse KL	$\int q(x)\log\frac{q(x)}{p(x)}\,\mathrm{d}x$	$-\log u$	$-\frac{q(x)}{p(x)}$
Pearson χ^2	$\int\frac{(q(x)-p(x))^2}{p(x)}\,\mathrm{d}x$	$(u-1)^2$	$2(\frac{p(x)}{q(x)}-1)$
Squared Hellinger	$\int\left(\sqrt{p(x)}-\sqrt{q(x)}\right)^2\,\mathrm{d}x$	$(\sqrt{u}-1)^2$	$(\sqrt{\frac{p(x)}{q(x)}}-1)\cdot\sqrt{\frac{q(x)}{p(x)}}$
Jensen-Shannon	$\frac{1}{2}\int p(x)\log\frac{2p(x)}{p(x)+q(x)}+q(x)\log\frac{2q(x)}{p(x)+q(x)}\,\mathrm{d}x$	$-(u+1)\log\frac{1+u}{2}+u\log u$	$\log\frac{2p(x)}{p(x)+q(x)}$
GAN	$\int p(x)\log\frac{2p(x)}{p(x)+q(x)}+q(x)\log\frac{2q(x)}{p(x)+q(x)}\,\mathrm{d}x-\log(4)$	$u\log u-(u+1)\log(u+1)$	$\log\frac{p(x)}{p(x)+q(x)}$

Variational Estimation of f-divergence [3,4]

- A general method to estimate f-divergence with *only* samples from P and Q
- Define the convex conjugate function as

$$f^*(t) = \sup_{u \in \mathrm{dom}_f} \{ut - f(u)\}$$

- Also, the duality suggests

$$f = f^{**} \left(= \sup_{t \in \mathrm{dom}_{f^*}} \{tu - f^*(t)\} \right)$$

[3] X. Nguyen et.al, "Estimating divergence functionals and the likelihood ratio by convex risk minimization," IEEE Transaction on Information Theory, 2010
[4] M. D. Reid et.al, "Information, divergence and risk for binary experiments," Journal of Machine Learning Research, 2011

Variational Estimation of f-divergence

A lower bound on the divergence :

$$D_f(P||Q) = \int_{\mathcal{X}} q(x) \underline{\sup_{t \in \mathrm{dom} f^*} \left\{ t \frac{p(x)}{q(x)} - f^*(t) \right\}} dx \quad \left(\because f(u) = f^{**}(u) = \sup_{t \in \mathrm{dom} f^*} \{tu - f^*(t)\} \right)$$

$$\geq \sup_{\underline{T(x) \in \mathcal{T}}} \left(\int_{\mathcal{X}} p(x) T(x) dx - \int_{\mathcal{X}} q(x) f^* \left(T(x) \right) dx \right) \quad \left(\text{“ = ” holds when } T(x) = \operatorname*{argmax}_{t \in \mathrm{dom} f^*} \left\{ t \frac{p(x)}{q(x)} - f^*(t) \right\} \right)$$

$$= \sup_{T \in \mathcal{T}} \left(\mathbb{E}_{X \sim P} \left[T(X) \right] - \mathbb{E}_{X \sim Q} \left[f^*(T(X)) \right] \right)$$

$$\boxed{\mathcal{T} \triangleq \{ T(x) : \mathcal{X} \to \mathbb{R} \}}$$

Variational Estimation of *f*-divergence

- Another representation of f-divergence

$$D_f(P||Q) = \sup_{T \in \mathcal{T}} \left(\mathbb{E}_{X \sim P}[T(X)] - \mathbb{E}_{X \sim Q}[f^*(T(X))] \right)$$

- Plug into GAN's setting

$$D_f(P_{\text{data}}||P_{G_\theta}) = \sup_{T \in \mathcal{T}} \left(\mathbb{E}_{X \sim P_{\text{data}}}[T(X)] - \mathbb{E}_{X \sim P_{G_\theta}}[f^*(T(X))] \right)$$

- Game-theoretic form

$$\min_\theta \max_{T \in \mathcal{T}} D_f(P_{\text{data}}||P_{G_\theta}) = \sup_{T \in \mathcal{T}} \left(\mathbb{E}_{X \sim P_{\text{data}}}[\boxed{T(X)}] - \mathbb{E}_{X \sim P_{G_\theta}}[f^*(T(X))] \right)$$

same role as a discriminator

- Optimal discriminator

$$T^*(x) = f'\left(\frac{p(x)}{q(x)}\right)$$

f-GAN Algorithm

- Parametrize $\mathcal{T}$: $\mathcal{T}_\omega$

- Define the new objective function

$$F(\theta,\omega) \triangleq \mathbb{E}_{X\sim P_{\text{data}}}[T_\omega(X)] + \mathbb{E}_{X\sim P_{G_\theta}}[f^*(T_\omega(X)]$$

- Setup minimax game

$$\theta^* = \underset{\theta}{\text{argmin}}\,\max_{\omega} F(\theta,\omega)$$

for *number of training iterations* **do**
  for k *steps* **do**
    Sample m noise samples $\{z_1, ..., z_m\}$ from standard normal P_Z ;
    Sample m examples (real data) $\{x_1, ..., x_m\}$ from $P_{\text{data}}(x)$;
    Update discriminator by ascending its stochastic gradient:

$$\nabla_\omega \frac{1}{m}\sum_{i=1}^{m}\left(T_\omega(x_i) + f^*\left(T_\omega\left(G_\theta\left(z_i\right)\right)\right)\right)$$

  end
  Sample m noise samples $\{z_1, ..., z_m\}$ from standard normal P_Z ;
  Update generator by descending its stochastic gradient:

$$\nabla_\theta \frac{1}{m}\sum_{i=1}^{m} f^*\left(T_\omega\left(G_\theta\left(z_i\right)\right)\right)$$

end

Algorithm 1: f-GAN

Practical Issues in Training : Vanishing Gradient

- Previous theory tells us that the cost of discriminator will at most

$$-2\log 2 + D_{\text{JS}}\left(P_{\text{data}}||P_{G_\theta}\right)$$

- However, in practice, if we train the discriminator until it converges, its error will go 0!

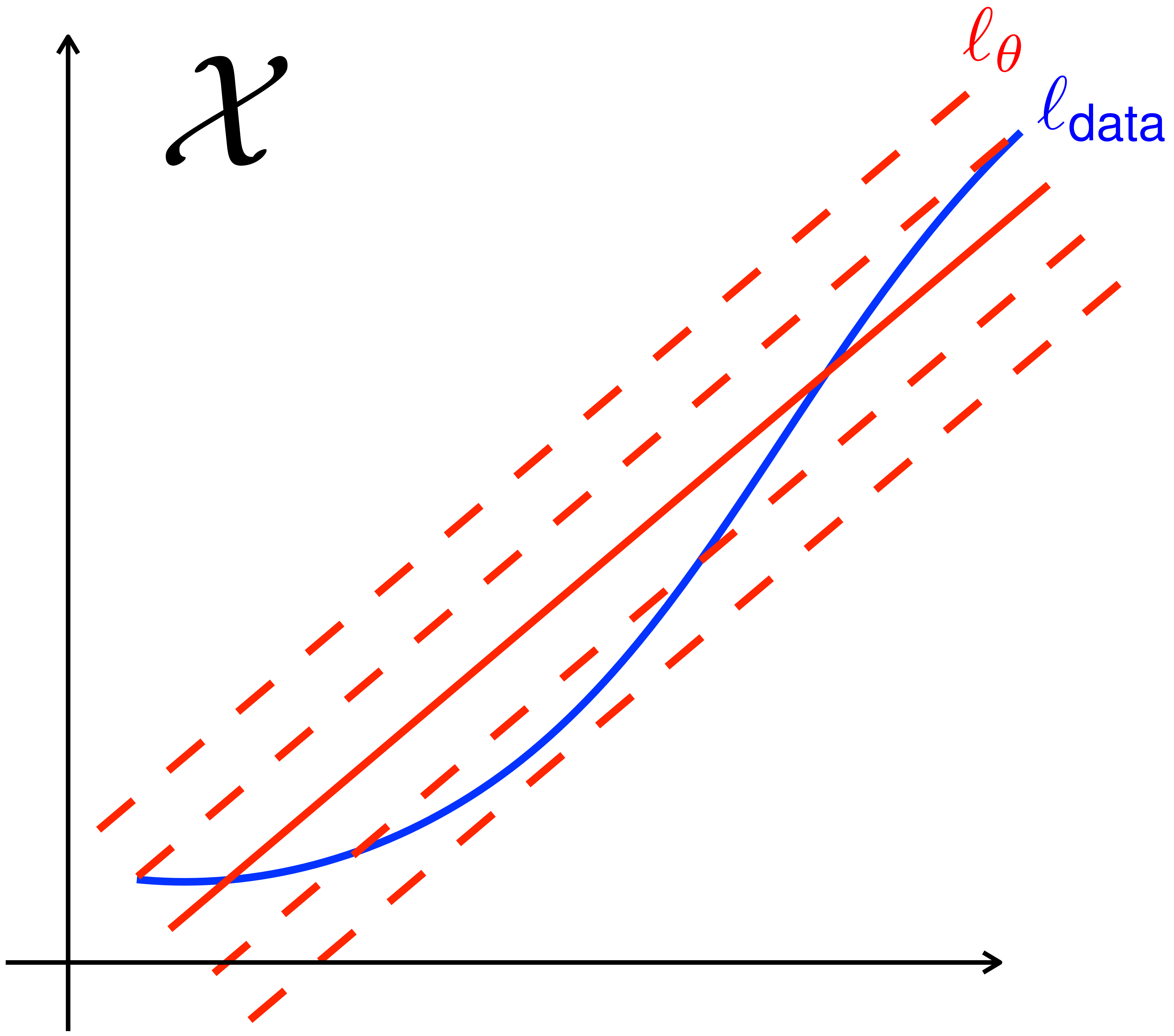
- The only way this can happen is if the distributions have disjoint support

$$D_{\text{JS}}\left(P_{\text{data}}||P_{G_\theta}\right) = \log 2 \;\; (\text{ recall } D_{\text{JS}}\left(P||Q\right) \leq \log 2,\; \forall P,\, Q)$$

- In this case, gradient vanishes when training generator

Low Dimensional Support

- Since $G_\theta(z) : \mathbb{R}^k \to \mathbb{R}^d$ with $k \ll d$, $P_{G_\theta}(x)$ lies on a low dimensional manifold (with dimension $\leq k$)



$P_{\text{data}}(x)$: uniform (ℓ_{data})

$P_{G_\theta}(x)$: uniform (ℓ_θ)

$$D_{\text{JS}}\left(P_{G_\theta} \| P_{\text{data}}\right) = \log 2, \ \forall \theta$$
$$(\because \mathcal{P}\left\{\ell_\theta \cap \ell_{\text{data}}\right\} = 0)$$

$$\Rightarrow \nabla_\theta D_{\text{JS}}\left(P_{G_\theta} \| P_{\text{data}}\right) = 0$$

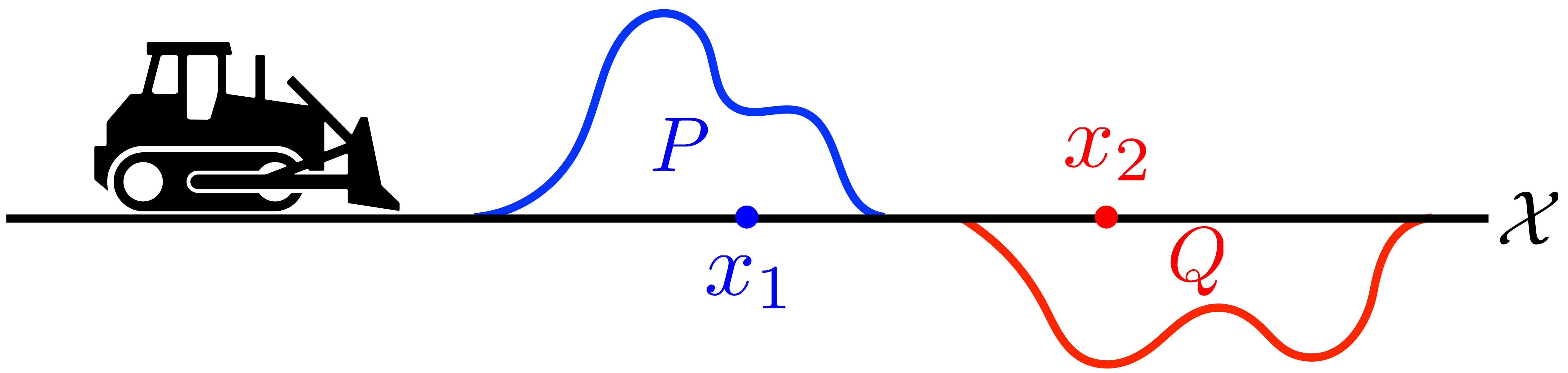
JS divergence is not a good measure !

- To overcome the shortcoming, Arjovsky proposed to use Wasserstein metric

Definition Wasserstein Metric

Let $(\mathcal{X}, d(\cdot,\cdot))$ be a metric space, and P, Q be two distributions on $\mathcal{X}$. The Wasserstein distance is defined as

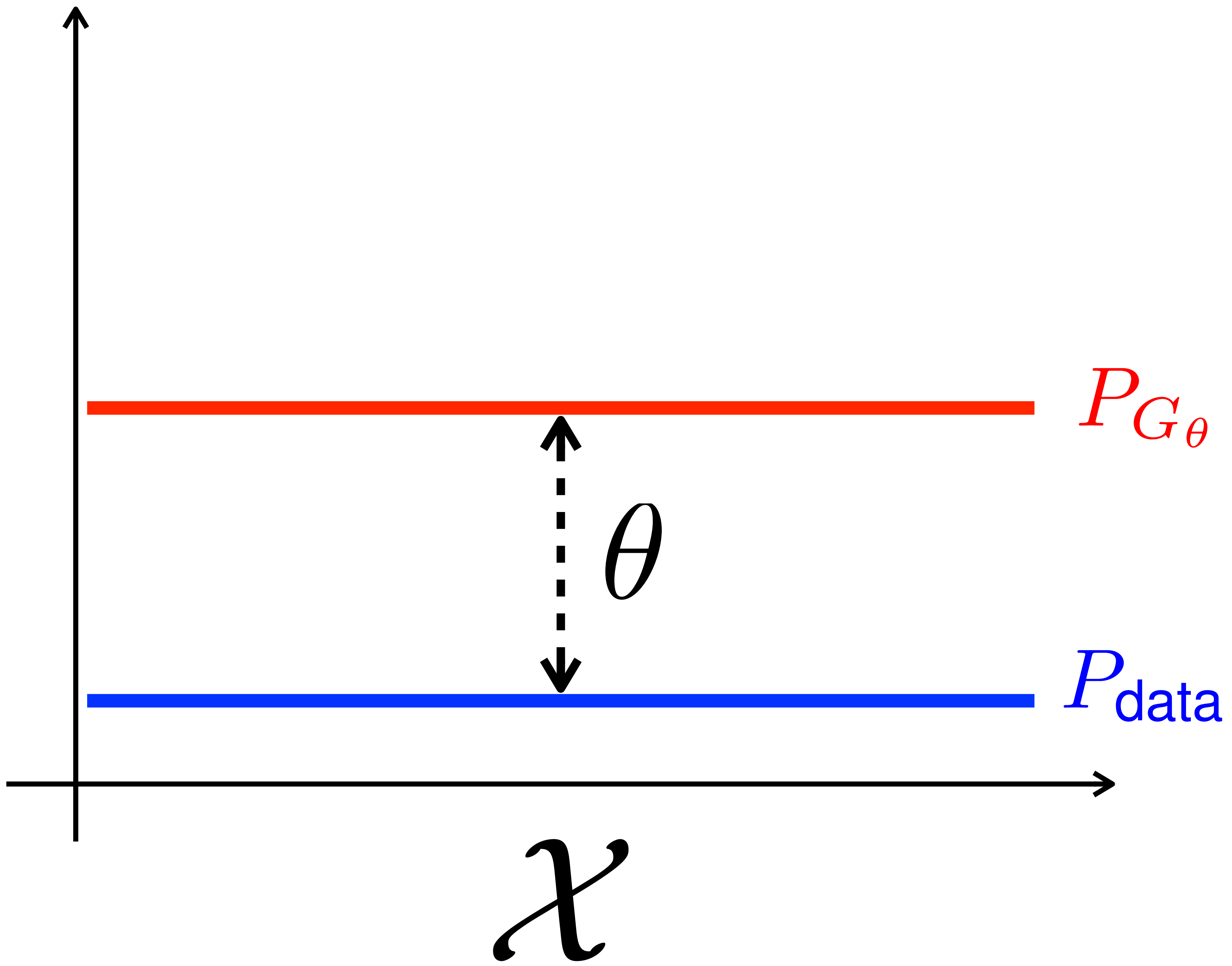
$$W(P,Q) \triangleq \inf_{\gamma \in \Gamma(P,Q)} \int_{\mathcal{X}\times\mathcal{X}} d(x,y)\,d\gamma(x,y)$$



$\gamma(x_1, x_2)$: amount of mass moved from x_1 to x_2

[5] Martin Arjovsky et.al, “Towards Principled Methods for Training Generative Adversarial Networks,” ICLR 2017
[6] Martin Arjovsky et.al, “Wasserstein GAN,” arXiv:1701.07875

Why Wasserstein a Better Distance?



$$D_{\text{JS}}\left(P_{\text{data}} || P_{G_\theta}\right) = \begin{cases} \log 2, & \text{if } \theta \neq 0 \\ 0, & \text{if } \theta = 0 \end{cases}$$

$$W\left(P_{\text{data}}, P_{G_\theta}\right) = |\theta|$$

Dual Representation

$$\boxed{\text{Goal : } \hat{\theta} = \underset{\theta}{\operatorname{argmin}}\, W\left(P_{\text{data}}, P_{G_\theta}\right)}$$

- Dual representation

$$W\left(P_{\text{data}}, P_{G_\theta}\right) = \sup_{\underline{\|f\|_{L\leq 1}}} \left\{\mathbb{E}_{P_{\text{data}}}[f(X)] - \mathbb{E}_{P_{G_\theta}}[f(X)]\right\}$$

1-Lipschitz

$$K \cdot W\left(P_{\text{data}}, P_{G_\theta}\right) = \sup_{\|f\|_{L\leq K}} \left\{\mathbb{E}_{P_{\text{data}}}[f(X)] - \mathbb{E}_{P_{G_\theta}}[f(X)]\right\}$$

- WGAN as a minimax game :

generalized discriminator

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} \max_{\underline{\|f_\omega\|\leq K}} \left\{\mathbb{E}_{P_{\text{data}}}\left[\boxed{f_\omega(X)}\right] - \mathbb{E}_{P_{G_\theta}}[f_\omega(X)]\right\}$$

choose $\|\omega\| \leq \tilde{K}$ instead $(\because \omega \rightarrow f_\omega \text{ conti. })$

Modified Algorithm

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} \max_{\|\omega\| \le \tilde{K}} \left\{ \mathbb{E}_{P_{\text{data}}} [f_\omega(X)] - \mathbb{E}_{P_{G_\theta}} [f_\omega(X)] \right\}$$

- Remove the $\log$ from discriminator
- Remove the sigmoid function in last layer (f_ω becomes a regressor of w-distance)
- Truncate the weights (to make f_ω Lipschitz)
- Do not use momentum-based optimizer (heuristic from experiments)

for *number of training iterations* **do**
  for k *steps* **do**
    Sample m noise samples $\{z_1, ..., z_m\}$ from standard normal P_Z ;
    Sample m examples (real data) $\{x_1, ..., x_m\}$ from $P_{\text{data}}(x)$;
    Update discriminator by ascending its stochastic gradient:

$$\nabla_\omega \frac{1}{m} \sum_{i=1}^{m} \left(f_\omega(x_i) - f_\omega \left(G_\theta \left(z_i \right) \right) \right)$$

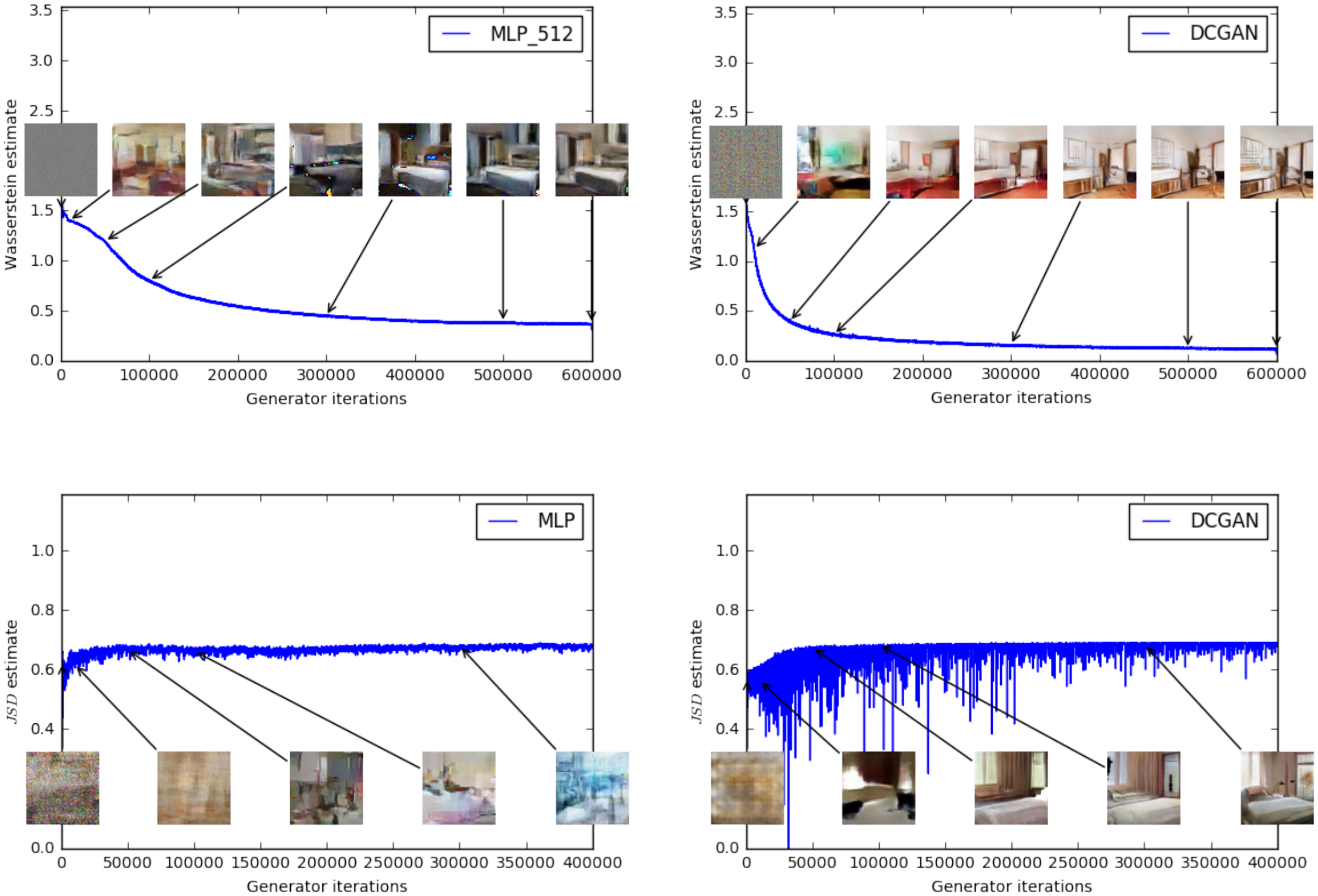
    $\omega \leftarrow \text{clip}(\omega, -\tilde{K}, \tilde{K})$;
  end
  Sample m noise samples $\{z_1, ..., z_m\}$ from standard normal P_Z ;
  Update generator by descending its stochastic gradient:

$$\nabla_\theta - \frac{1}{m} \sum_{i=1}^{m} f_\omega \left(G_\theta \left(z_i \right) \right)$$

end

Algorithm 1: Wasserstein GAN

Experiments



Some Take-home Messages

- Playing a minimax game is equivalent to minimize the divergence
- Minimax representation helps when evaluating density function is hard
- Wasserstein distance benefits when metric in $\mathcal{X}$ matters

Thanks for your attention !