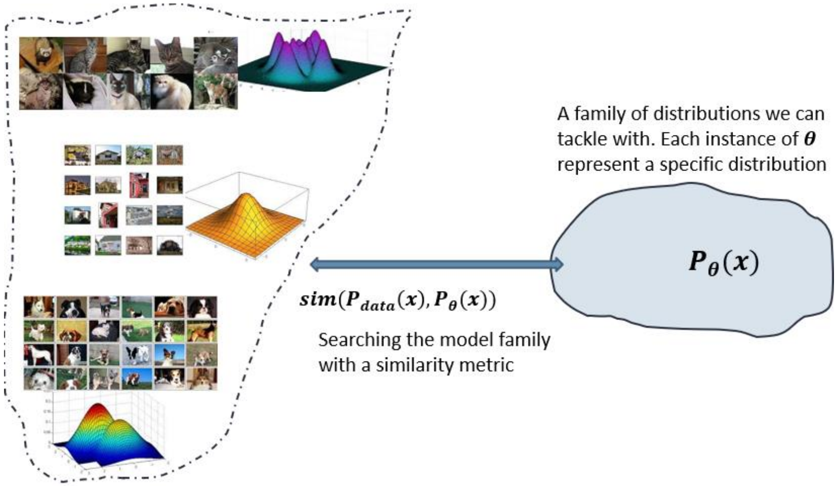
Generative adversarial networks

22-808: Generative models
Sharif University of Technology
Fall 2025

Fatemeh Seyyedsalehi

Recap



- We need a framework to interact with distributions for statistical generative models.
  - Probabilistic generative models
  - Deep generative models
    - Autoregressive models $p_\theta(\mathbf{x}) = \prod_{i=1}^{n} p_\theta(x_i|\mathbf{x}_{<i})$
    - Variational Autoencoders $p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x}, \mathbf{z})d\mathbf{z}$
    - **Generative adversarial networks**
  - Both AR and VAE model families attempted to minimize the KL divergence between model family and data distribution, or equivalently attempt to maximize the likelihood.
  - In GAN we are going to use an alternative choice for the similarity measure between model distribution and data distribution.

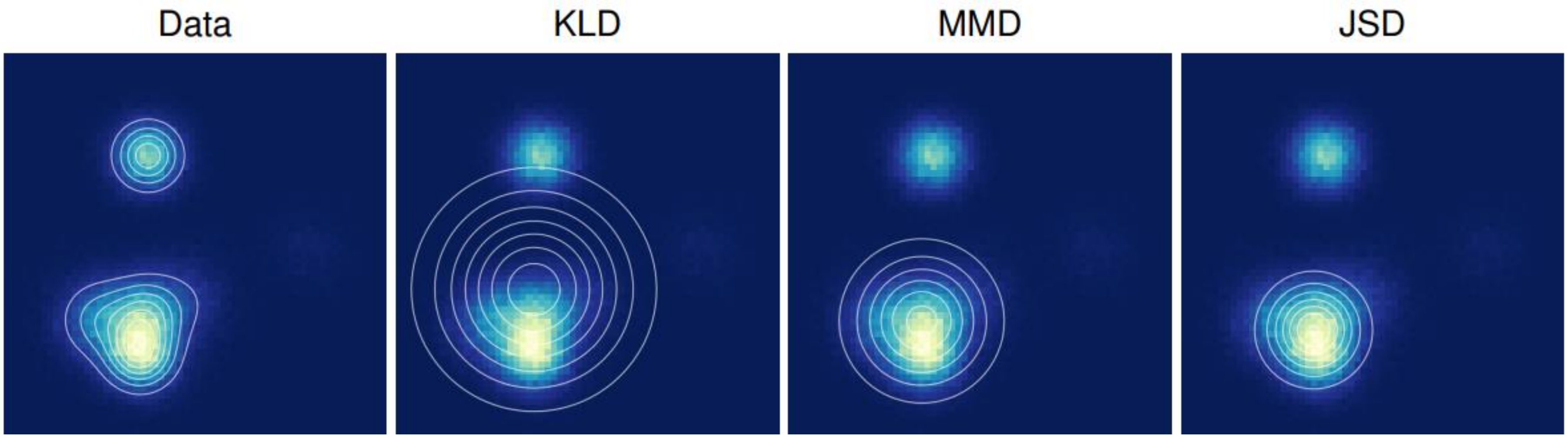
Maximizing the likelihood

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}} \sum_{i=1}^{M} \log p_\theta(\mathbf{x}_i), \quad \mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_M \sim p_{\text{data}}(\mathbf{x})$$

- Optimal statistical efficiency
  - Assume sufficient model capacity, such that there exists a unique θ^* that satisfy $p_{\theta^*} = p_{data}$.
  - The convergence of $\hat{\theta}$ to θ^* when $M \rightarrow \infty$, is the fastest among all statistical methods when using maximum likelihood training.

Maximizing the likelihood

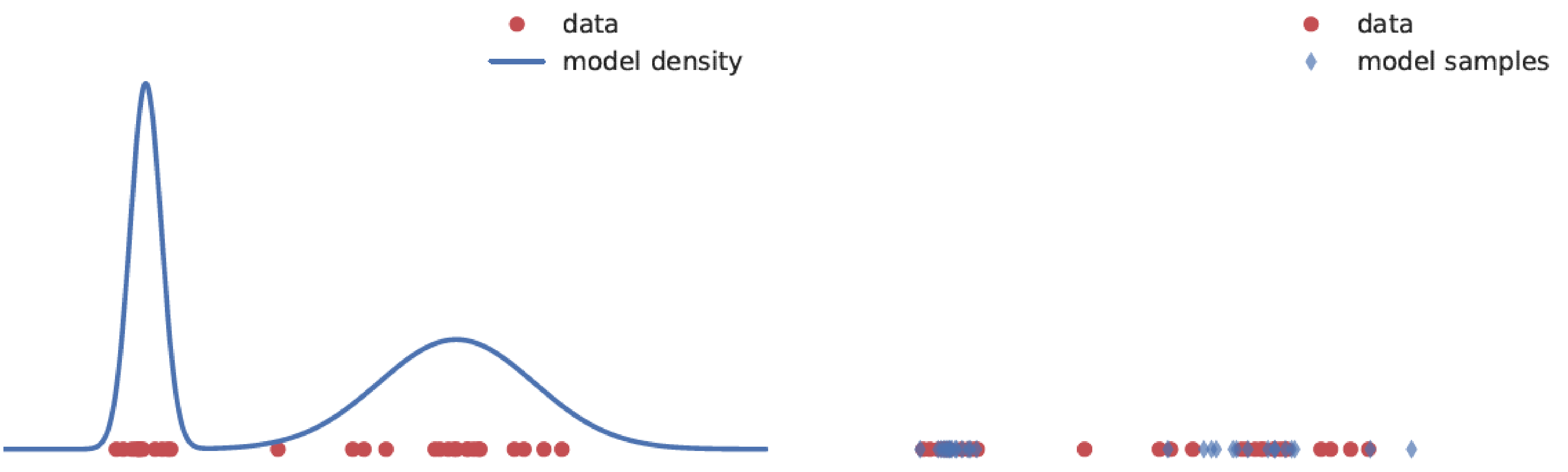
- For imperfect models, achieving high log-likelihoods might not always imply good sample quality.



An isotropic Gaussian distribution was fit to data drawn from a mixture of Gaussians by either minimizing KL divergence (KLD), maximum mean discrepancy (MMD), or Jensen-Shannon divergence (JSD). The different fits demonstrate different tradeoffs made by the three measures of distance between distributions.

Implicit generative models

- Kind of probabilistic generative models without an explicit likelihood function
- We use a likelihood-free approach to train these models
  - Training by comparing samples



Explicit models vs. implicit models

Learning by comparing samples

- We should define a distance(similarity) measure between two distributions that:
  - Provides guarantees about learning the data distribution.

$$\underset{p_\theta}{\operatorname{argmin}}\, D(p_{data}, p_\theta) = p_{data}$$

  - Can be evaluated only using samples from the data and model distribution.
  - Are computationally cheap to evaluate.

- Many distributional distances and divergences fail to satisfy the later two requirements

Learning by comparing samples

- The main approach to overcome these challenges is to approximate the desired quantity through optimization by introducing a comparison model, often called a **discriminator** or a **critic** D, such that:

$$\mathcal{D}(p^*, q) = \underset{D}{\operatorname{argmax}}\, \mathcal{F}(D, p^*, q)$$

- where $\mathcal{F}$ is a functional that can be estimated using only samples from $p^*(p_{data})$ and q. One way is that it depends on distributions only in expectations.
  - Therefore, it can be estimated using Monte Carlo estimation.

Learning by comparing samples

- As we usually use parametric functions (ex. Neural networks) for both the model and discriminator.
- Therefore, by the following optimization we estimate the distance measure $\mathcal{D}(p^*, q_\theta)$

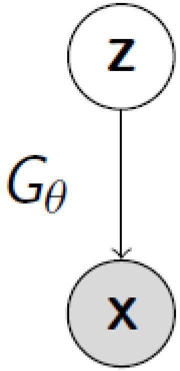
$$\operatorname{argmax}_{\phi} \mathcal{F}(D_{\phi}, p^*, q_\theta)$$

- Then, instead of optimizing the exact objective $\mathcal{D}(p^*, q_\theta)$

we use the tractable approximation provided through the optimal D_ϕ.

Generative adversarial networks (Goodfellow GAN)

- A finite number of samples from the desired real distribution is available: $x_1, x_2, \ldots, x_n$



- Like VAEs, we consider a latent variable model for the model generation process and attempt to learn G_θ. However, here we learn this function by Comparing samples.

The Goodfellow GAN
The probabilistic classification view

- Assuming $D(x)$ as a binary classifier which predicts whether a given point x was sampled from the real distribution or it is a fake sample from the generator G_θ.

- A cross entropy loss to train this classifier:

$$E_{\mathbf{x}\sim p_{\mathrm{data}}}[\log D_\phi(\mathbf{x})] + E_{\mathbf{x}\sim p_\theta}[\log(1 - D_\phi(\mathbf{x}))]$$

- We can see that the optimal discriminator for a fixed generator G_θ is:

$$\frac{p(x)}{p(x) + p_\theta(x)}$$

The Goodfellow GAN
The objective function

- By substitution the optimal discriminator into the cross-entropy loss, we have:

$$
\begin{aligned}
V^*(q_\theta, p^*) &= \frac{1}{2}\mathbb{E}_{p^*(\boldsymbol{x})}[\log \frac{p^*(\boldsymbol{x})}{p^*(\boldsymbol{x}) + q_\theta(\boldsymbol{x})}] + \frac{1}{2}\mathbb{E}_{q_\theta(\boldsymbol{x})}[\log(1 - \frac{p^*(\boldsymbol{x})}{p^*(\boldsymbol{x}) + q_\theta(\boldsymbol{x})})] \\
&= \frac{1}{2}\mathbb{E}_{p^*(\boldsymbol{x})}[\log \frac{p^*(\boldsymbol{x})}{\frac{p^*(\boldsymbol{x})+q_\theta(\boldsymbol{x})}{2}}] + \frac{1}{2}\mathbb{E}_{q_\theta(\boldsymbol{x})}[\log(\frac{q_\theta(\boldsymbol{x})}{\frac{p^*(\boldsymbol{x})+q_\theta(\boldsymbol{x})}{2}})] - \log 2 \\
&= \frac{1}{2}D_{\mathbb{KL}}\left(p^* \,\|\, \frac{p^* + q_\theta}{2}\right) + \frac{1}{2}D_{\mathbb{KL}}\left(q_\theta \,\|\, \frac{p^* + q_\theta}{2}\right) - \log 2 \\
&= JSD(p^*, q_\theta) - \log 2
\end{aligned}
$$

where JSD is the Jensen-Shannon divergence.

The Goodfellow GAN

- This establishes a connection between optimal binary classification and distributional divergences.
- By using binary classification, we were able to compute the distributional divergence using only samples, which is the important property needed for learning implicit generative models
- We have turned an intractable estimation problem (how to estimate the JSD divergence) into an optimization problem (how to learn a classifier) which can be used to approximate that divergence.

The Goodfellow GAN

- With optimal discriminator, we attempt to find the generative model G_θ that minimizes the JSD divergence.

$$
\begin{aligned}
\min_{\boldsymbol{\theta}} JSD(p^*, q_\theta) &= \min_{\boldsymbol{\theta}} V^*(q_\theta, p^*) + \log 2 \\
&= \min_{\boldsymbol{\theta}} \frac{1}{2}\mathbb{E}_{p^*(\boldsymbol{x})} \log D^*(\boldsymbol{x}) + \frac{1}{2}\mathbb{E}_{q_\theta(\boldsymbol{x})} \log(1 - D^*(\boldsymbol{x})) + \log 2
\end{aligned}
$$

Training procedure of GAN

Sample minibatch of m training points $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(m)}$ from $\mathcal{D}$

Sample minibatch of m noise vectors $\mathbf{z}^{(1)}, \mathbf{z}^{(2)}, \ldots, \mathbf{z}^{(m)}$ from p_z

Update the discriminator parameters ϕ by stochastic gradient **ascent**

$$\nabla_\phi V(G_\theta, D_\phi) = \frac{1}{m}\nabla_\phi \sum_{i=1}^{m}[\log D_\phi(\mathbf{x}^{(i)}) + \log(1 - D_\phi(G_\theta(\mathbf{z}^{(i)})))]$$

Update the generator parameters θ by stochastic gradient **descent**

$$\nabla_\theta V(G_\theta, D_\phi) = \frac{1}{m}\nabla_\theta \sum_{i=1}^{m} \log(1 - D_\phi(G_\theta(\mathbf{z}^{(i)})))$$

Repeat for fixed number of epochs

Training convergence

- If G and D have enough capacity, and at each step of training procedure, the discriminator is allowed to reach its optimum for a specific G_θ , and then p_θ is updated so as to improve

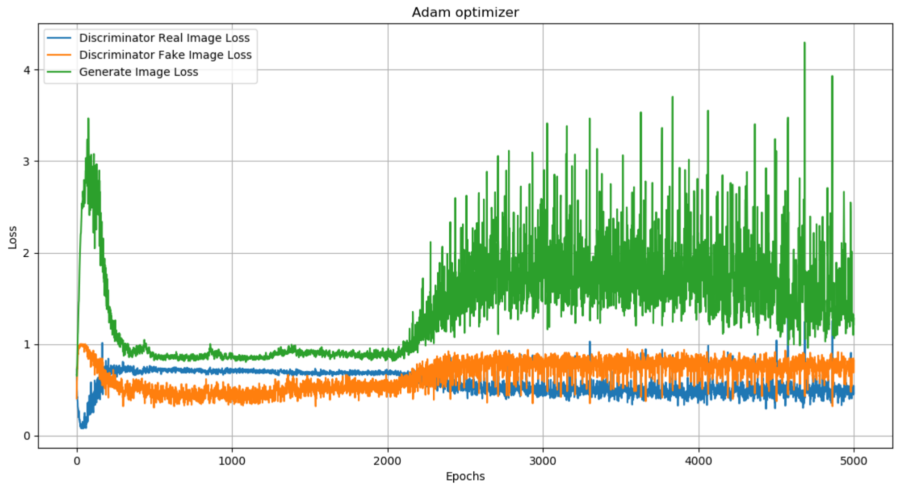
$$\mathbb{E}_{\boldsymbol{x}\sim p_{data}}[\log D_G^*(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{x}\sim p_g}[\log(1 - D_G^*(\boldsymbol{x}))]$$

then p_θ converges to p_{data}.

- Unrealistic assumptions ☹

Training convergence

- However, we do not have access to the optimal discriminator and only we can approximate it with a parametrized function: neural network D_ϕ
  - No guarantee for convergence
  - In practice, the generator and discriminator loss keeps oscillating during GAN training

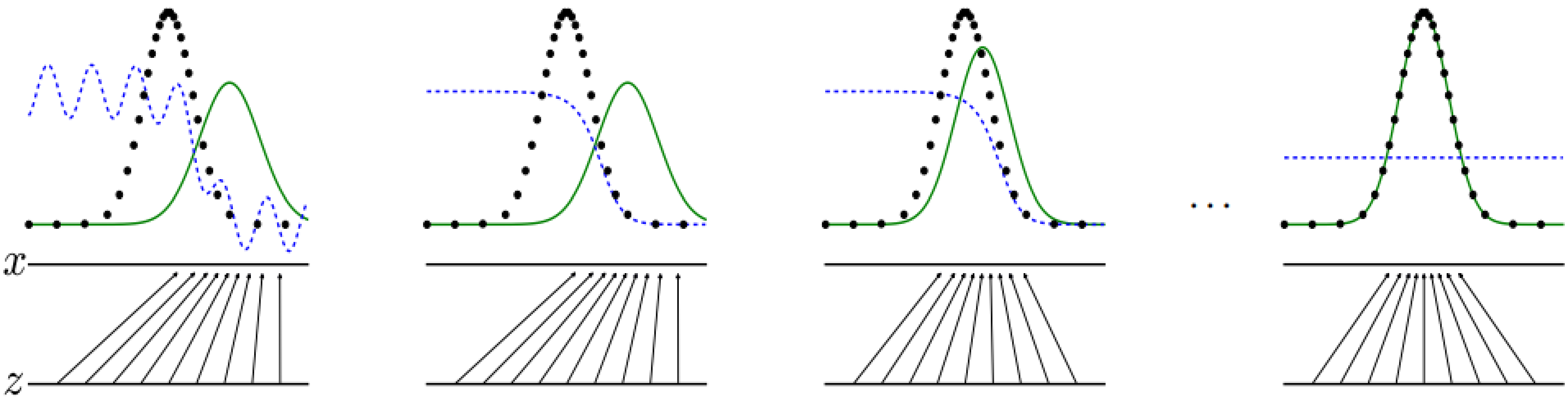


The min-max game

- The minmax game

$$\min_{\theta} \max_{\phi} V(G_\theta, D_\phi) = E_{\mathbf{x}\sim p_{\text{data}}}[\log D_\phi(\mathbf{x})] + E_{\mathbf{z}\sim p(\mathbf{z})}[\log(1 - D_\phi(G_\theta(\mathbf{z})))]$$

  - It is a game not an optimization problem
  - It should reach to a Nash equilibria



Example

- Which one is real?

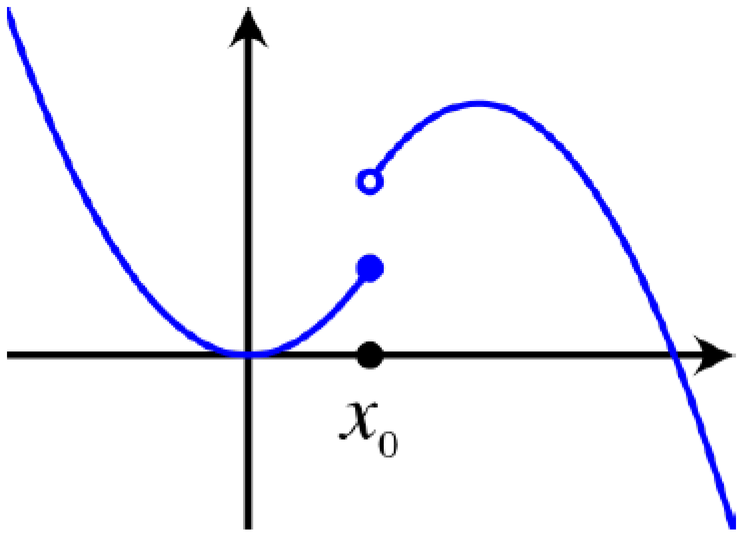
F-divergence

- Let $f: R \rightarrow R$ be a convex lower-semicontinuous function, such that $f(1) = 0$. We define the *f-divergence* between two distributions with densities p and q by:

$$D_f(p \parallel q) \equiv \int_{\mathcal{X}} q(x) f\left(\frac{p(x)}{q(x)}\right) dx.$$

- What's interesting about *f-divergence* is that we can construct a variational representation for it.
  - Alternating the integral to an optimization

Convex lower-semicontinuous function



Fenchel duality

- The idea is to use the convex conjugate of the function *f*, which is defined as follows:

$$f^*(t) \equiv \sup_x \{tx - f(x)\}.$$

- Fenchel duality: repeat application of the conjugate operation to convex lower-semicontinuous function *f* yields $f^{**} = f$. Therefore, we have:

$$f(x) = \sup_t \{tx - f^*(t)\}.$$

Variational representation of *F-divergence*

- Using Fenchel duality, we obtain the variational representation of the *f-divergence.*

$$\boxed{D_f(p \parallel q)} = \int_{\mathcal{X}} q(x) \sup_t \left[t\frac{p(x)}{q(x)} - f^*(t) \right] dx$$
$$= \int_{\mathcal{X}} \sup_t \left[tp(x) - f^*(t)q(x) \right] dx$$
$$= \sup_{T:\mathcal{X}\to\mathbb{R}} \int_{\mathcal{X}} \left(T(x)p(x) - f^*(T(x))q(x) \right) dx$$
$$= \boxed{\sup_{T:\mathcal{X}\to\mathbb{R}} \left[\mathop{\mathbb{E}}_{x\sim p} T(x) - \mathop{\mathbb{E}}_{x\sim q} f^*(T(x)) \right].}$$

F-GAN

- The dual form can be approximated using Monte Carlo estimation.
- Assuming a parametric family of functions $T\varphi$ (ex. a neural network) and the generator function g_θ, and a valid f-divergence, the F-GAN objective is,

$$\begin{aligned}\theta_f &= \arg\min_\theta \sup_\varphi \left[\mathop{\mathbb{E}}_{x\sim p} T_\varphi(x) - \mathop{\mathbb{E}}_{x\sim p_\theta} f^*(T_\varphi(x)) \right] \\ &= \arg\min_\theta \sup_\varphi \left[\mathop{\mathbb{E}}_{x\sim p} T_\varphi(x) - \mathop{\mathbb{E}}_{z\sim q} f^*(T_\varphi(g_\theta(z))) \right].\end{aligned}$$

- Generator g_θ tries to minimize the divergence estimate and discriminator $T\varphi$ tries to tighten the lower bound

F-divergence

distance or divergence	corresponding $g(t)$ $(t = \frac{p_i(x)}{p_j(x)})$
Bhattacharyya distance [1]	$\sqrt{t}$
KL-divergence	$t\log(t)$
Symmetric KL-divergence	$t\log(t) - \log(t)$
Hellinger distance	$(\sqrt{t}-1)^2$
Total variation	$\lvert t-1\rvert$
Pearson divergence	$(t-1)^2$
Jensen-Shannon divergence	$\frac{1}{2}(t\log\frac{2t}{t+1} + \log\frac{2}{t+1})$

The Goodfellow GAN as F-GAN

- The Goodfellow GAN is an instances of the *f-GAN.*
- Modified version of the Jensen-Shannon

$$2\mathrm{JSD}(p,q) - \log(4) = D_{\mathrm{KL}}\left(p \middle\| \frac{p+q}{2}\right) + D_{\mathrm{KL}}\left(p_g \middle\| \frac{p+q}{2}\right) - \log(4).$$

- The *f-divergence:*

$$f(x) = x \log x - (x+1)\log(x+1)$$

$$f^*(t) = -\log(1 - e^t).$$

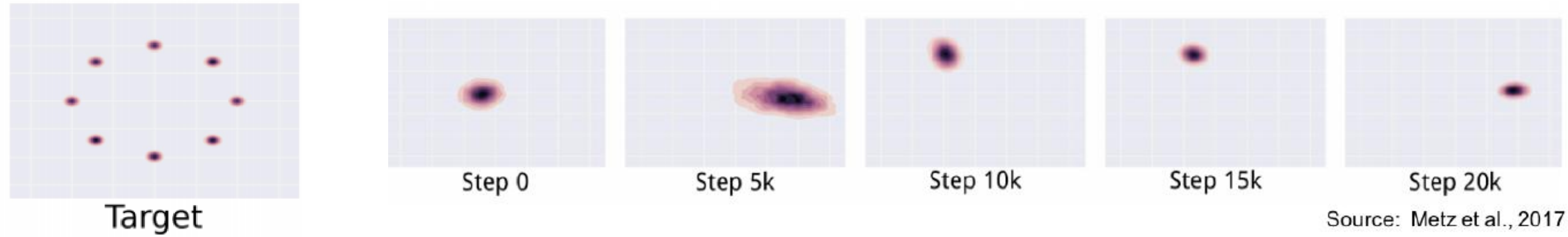
$$T_\varphi(x) = \log(d_\varphi(x))$$

- *We can obtain the* Goodfellow GAN :

$$\theta_f = \arg\min_\theta \sup_\varphi \left[\mathop{\mathbb{E}}_{x\sim p} \log d_\varphi(x) + \mathop{\mathbb{E}}_{z\sim q} \log(1 - d_\varphi(g_\theta(z)))\right]$$

Mode collapse and catastrophic forgetting

- In the case of mode collapse, the generator might focus on producing only a limited set of outputs that it knows will deceive the discriminator, completely ignoring other parts of the data distribution.
- As the generator iterates over epochs, it starts to forget the diversity it initially captured.
  - This happens because it gets reinforced to produce only certain types of outputs that are effective in fooling the discriminator.

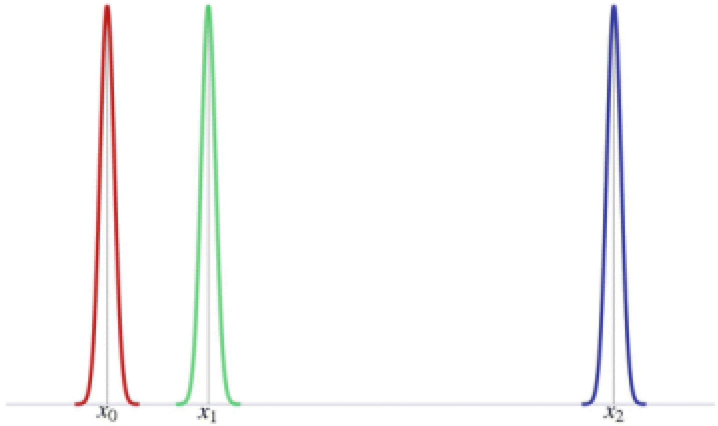


Source: Metz et al., 2017

The generator distribution keeps oscillating between different modes

The problem with KL divergence

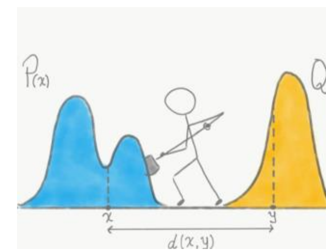
- KL divergence problem:
  - When distributions' supports are different, the KL does not defined.
  - As it consider the ratio of probability values, it shows a big difference between wo distributions when one has a very small value in even in a small region.



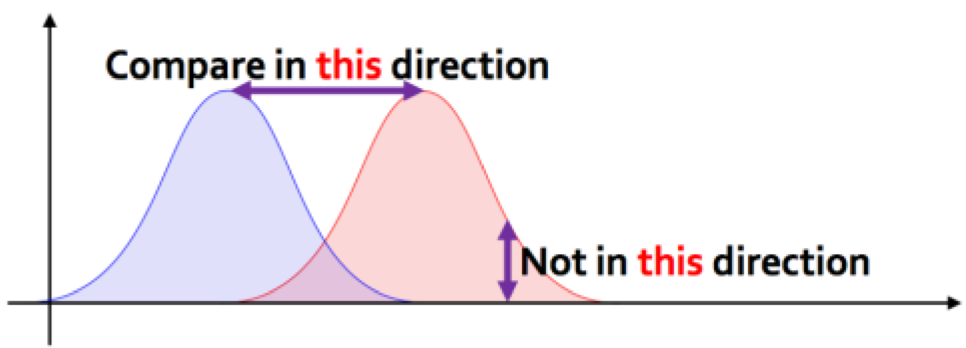
Wasserstein GAN

- Earth-Mover (EM) distance (Wasserstein-1)
  - $\Pi(P_r, P_g)$ shows the set of all joint distributions whose marginals are P_r and P_g, respectively.

$$W(\mathbb{P}_r, \mathbb{P}_g) = \inf_{\gamma \in \Pi(\mathbb{P}_r, \mathbb{P}_g)} \mathbb{E}_{(x,y)\sim\gamma} \big[\|x - y\| \big]$$



  - It is the cost of optimal transport between two distributions P_r and P_g.



Jeremy Kun, (blog post, 2018)

Example

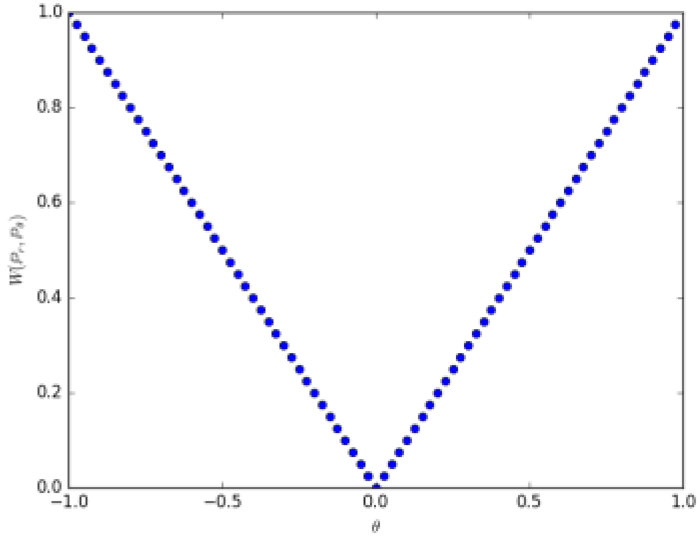
- Consider $z \sim U[0,1]$
  - P_0 a distribution over $(0,\ z)$
  - P_θ a distribution over $(\theta,\ z)$

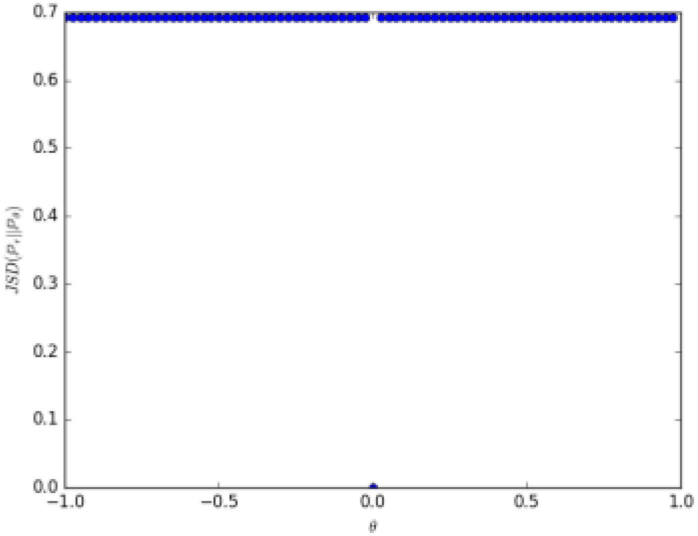
- Different distance measure for these two distributions:

- $W(\mathbb{P}_0, \mathbb{P}_\theta) = |\theta|$,
- $JS(\mathbb{P}_0, \mathbb{P}_\theta) = \begin{cases} \log 2 & \text{if } \theta \neq 0 \ , \\ 0 & \text{if } \theta = 0 \ , \end{cases}$
- $KL(\mathbb{P}_\theta \| \mathbb{P}_0) = KL(\mathbb{P}_0 \| \mathbb{P}_\theta) = \begin{cases} +\infty & \text{if } \theta \neq 0 \ , \\ 0 & \text{if } \theta = 0 \ , \end{cases}$
- and $\delta(\mathbb{P}_0, \mathbb{P}_\theta) = \begin{cases} 1 & \text{if } \theta \neq 0 \ , \\ 0 & \text{if } \theta = 0 \ . \end{cases}$

Example

- Learning can not be done with the other distances and divergences because the resulting loss function is not even continuous.
- Comparing EM and JSD for different θ





Kantorovich-Rubinstein Duality

- We can approximate the Wasserstein distance with its dual form:

$$W(\mathbb{P}_r, \mathbb{P}_\theta) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{x\sim\mathbb{P}_r}[f(x)] - \mathbb{E}_{x\sim\mathbb{P}_\theta}[f(x)]$$

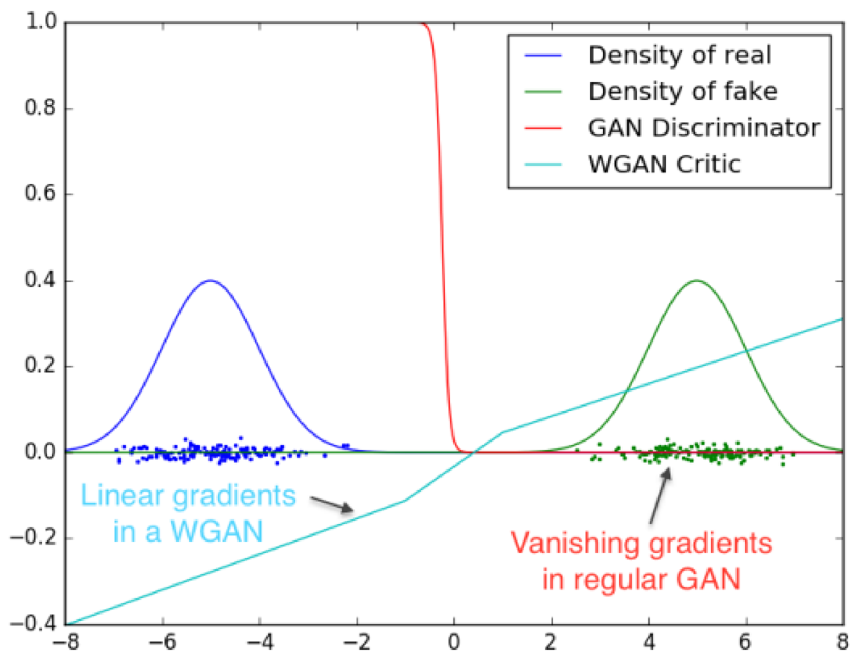
where the sup is over all 1-Lipschitz functions f : X → R.

- Considering a parameterized family of functions for f:
  - However, we need to be sure that this family satisfy the 1-Lipschitz constraint.
  - The Lipschitz constraint is essentially that a function must have a maximum gradient. The specific maximum gradient is a hyperparameter.

$$\max_{w\in\mathcal{W}} \mathbb{E}_{x\sim\mathbb{P}_r}[f_w(x)] - \mathbb{E}_{z\sim p(z)}[f_w(g_\theta(z))]$$

Constraint on the discriminator function

- The red line is a good discriminator but its gradient is nearly 0 at most points. The cyan line is clearly much worse as a discriminator, but is much better for training the generator because its gradient is not zero.
- The Lipschitz constraint limits the discriminator function



Lipschitz constraint

- Quick and dirty solution: clamp the size of the weights

$$-c < W < c$$

- Or clipping the gradient.

- However, a better solution is to add a soft penalty to the loss function as follows: (WGAN-GP)

$$\underbrace{\underset{\tilde{\boldsymbol{x}} \sim \mathbb{P}_g}{\mathbb{E}}[D(\tilde{\boldsymbol{x}})] - \underset{\boldsymbol{x} \sim \mathbb{P}_r}{\mathbb{E}}[D(\boldsymbol{x})]}_{\text{Original critic loss}} + \underbrace{\lambda \underset{\hat{\boldsymbol{x}} \sim \mathbb{P}_{\hat{\boldsymbol{x}}}}{\mathbb{E}}\left[(\|\nabla_{\hat{\boldsymbol{x}}} D(\hat{\boldsymbol{x}})\|_2 - 1)^2\right]}_{\text{Our gradient penalty}}$$

Where $\hat{x}$ is uniformly sampled from the line between samples of two distributions