CENG 796
Deep Generative Models

Generative Adversarial Networks - Part I (Fundamentals)

ORTA DOĞU TEKNİK ÜNİVERSİTESİ
MIDDLE EAST TECHNICAL UNIVERSITY

Motivation: GAN Progress

- Ian Goodfellow is first-author on the first GAN paper
- GAN is most prominent of Implicit Models

Motivation: BigGAN

[BigGAN, Brock, Donahue, Simonyan, 2018]

Slide originally by Pieter Abbeel, Peter Chen, Jonathan Ho, Aravind Srinivas, Alex Li, Wilson Yan

Motivation: GAN Art

Slide originally by Pieter Abbeel, Peter Chen, Jonathan Ho, Aravind Srinivas, Alex Li, Wilson Yan

So far...

- Autoregressive models
- Flow models
- Latent Variable Models

- Common aspect: Likelihood-based models
  - exact (autoregressive and flows)
  - approximate (VAE)

Generative Models

- Sample
- Evaluate likelihood
- Train
- Representation

→ What if all we care about is sampling?

Slide originally by Pieter Abbeel, Peter Chen, Jonathan Ho, Aravind Srinivas, Alex Li, Wilson Yan

Building a sampler

- How about this sampler?

```
import glob, cv2, numpy as np
files = glob.glob('*.jpg')
def _sample():
    idx = np.random.randint(len(files))
    return cv2.imread(files[idx])
def sample(*, n_samples):
        samples = np.array([_sample() for _ in range(n_samples)])
    return samples
```

Slide originally by Pieter Abbeel, Peter Chen, Jonathan Ho, Aravind Srinivas, Alex Li, Wilson Yan

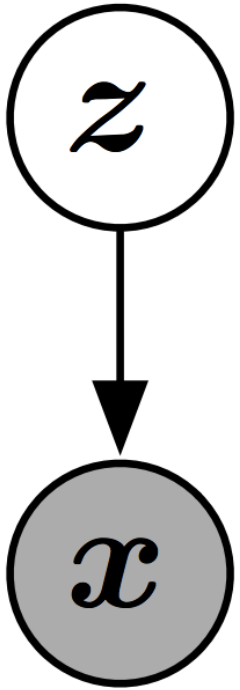
Building a sampler

- You don’t just want to sample the exact data points you have.
- You want to build a generative model that can understand the underlying distribution of data points and
  - smoothly interpolate across the training samples
  - output samples similar but not the same as training data samples
  - output samples representative of the underlying factors of variation in the training distribution.
  - Example: digits with unseen strokes, faces with unseen poses, etc.

Slide originally by Pieter Abbeel, Peter Chen, Jonathan Ho, Aravind Srinivas, Alex Li, Wilson Yan

Implicit Models

- Sample z from a fixed noise source distribution (uniform or gaussian).
- Pass the noise through a deep neural network to obtain a sample x.
- Sounds familiar? Right:
  - Flow Models
  - VAE
- *What's going to be different here?*
  - *Learning the deep neural network **without** explicit density estimation*



Slide originally by Pieter Abbeel, Peter Chen, Jonathan Ho, Aravind Srinivas, Alex Li, Wilson Yan

Implicit Models

Given samples from data distribution $p_{\text{data}} : x_1, x_2, \cdots, x_n$

Given a sampler $q_\phi(z) = \text{DNN}(z; \phi)$ where $z \sim p(z)$

- $x = q_\phi(z)$ induces a density function p_{model}

Do not have an explicit form for p_{data} or p_{model}; can only draw samples

Make p_{model} as close to p_{data} as possible by learning an appropriate ϕ

Slide originally by Pieter Abbeel, Peter Chen, Jonathan Ho, Aravind Srinivas, Alex Li, Wilson Yan

Departure from maximum likelihood

We need some measure of how far apart p_{data} and induced p_{model} are

With density models, we used $KL(p_{\text{data}}||p_{\text{model}})$ which gave us the objective $\mathbb{E}_{x\sim p_{\text{data}}}[\log p_\theta(x)]$ (discarding the term independent of θ)

- where we explicitly modeled p_{model} as $p_\theta(x)$

Not having an explicit $p_\theta(x)$ requires us to come up distance measures that potentially behave differently from maximum likelihood.

Example, Maximum Mean Discrepancy (MMD); Jensen Shannon Divergence (JSD); Earth Mover's Distance, etc.

Slide originally by Pieter Abbeel, Peter Chen, Jonathan Ho, Aravind Srinivas, Alex Li, Wilson Yan

Generative Adversarial Networks

Generative Adversarial Nets

Ian J. Goodfellow[*], **Jean Pouget-Abadie**[†], **Mehdi Mirza**, **Bing Xu**, **David Warde-Farley**, **Sherjil Ozair**[‡], **Aaron Courville**, **Yoshua Bengio**[§]
Département d'informatique et de recherche opérationnelle
Université de Montréal
Montréal, QC H3C 3J7

Abstract

We propose a new framework for estimating generative models via an adversarial process, in which we simultaneously train two models: a generative model G that captures the data distribution, and a discriminative model D that estimates the probability that a sample came from the training data rather than G. The training procedure for G is to maximize the probability of D making a mistake. This framework corresponds to a minimax two-player game. In the space of arbitrary functions G and D, a unique solution exists, with G recovering the training data distribution and D equal to $\frac{1}{2}$ everywhere. In the case where G and D are defined by multilayer perceptrons, the entire system can be trained with backpropagation. There is no need for any Markov chains or unrolled approximate inference networks during either training or generation of samples. Experiments demonstrate the potential of the framework through qualitative and quantitative evaluation of the generated samples.

Slide originally by Pieter Abbeel, Peter Chen, Jonathan Ho, Aravind Srinivas, Alex Li, Wilson Yan

Generative Adversarial Networks

$$\min_G \max_D \mathbb{E}_{x \sim p_{\text{data}}} [\log D(x)] + \mathbb{E}_{z \sim p(z)} [\log(1 - D(G(z)))]$$

- Two player minimax game between generator (G) and discriminator (D)
- (D) tries to maximize the log-likelihood for the binary classification problem
  - data: real (1)
  - generated: fake (0)
- (G) tries to minimize the log-probability of its samples being classified as “fake” by the discriminator (D)

Slide originally by Pieter Abbeel, Peter Chen, Jonathan Ho, Aravind Srinivas, Alex Li, Wilson Yan

Generative Adversarial Networks

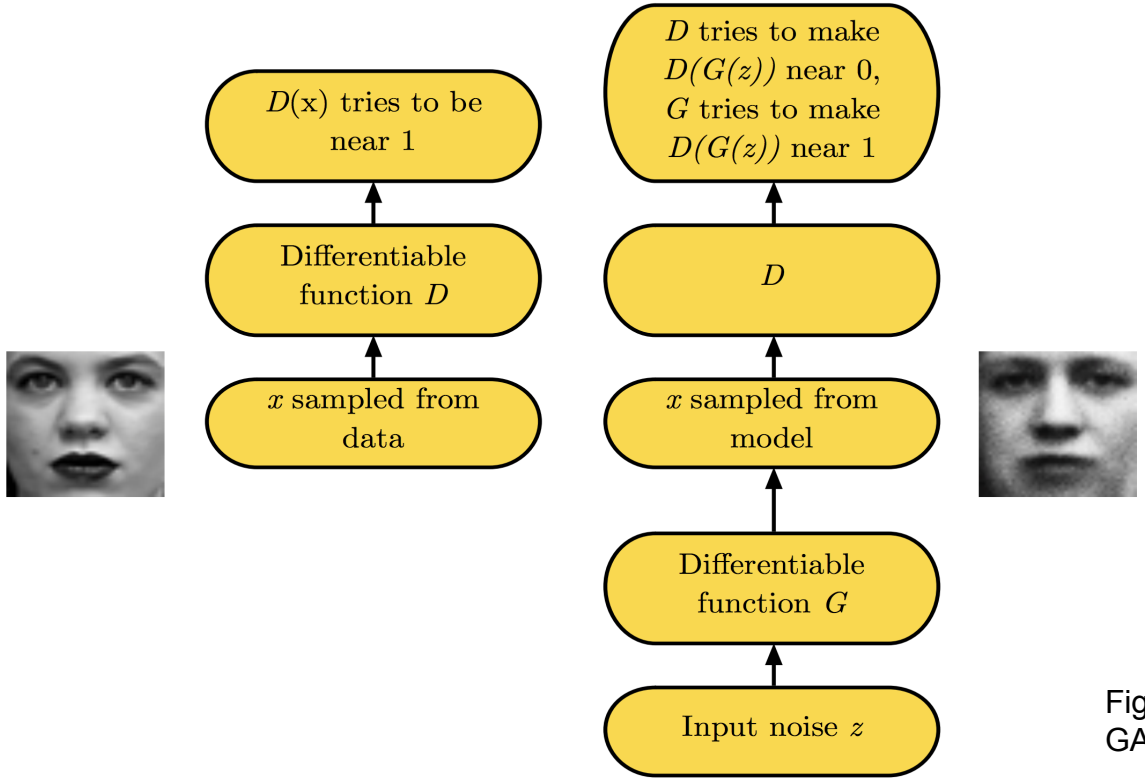


Figure from NeurIPS 2016 GAN Tutorial (Goodfellow)

Slide originally by Pieter Abbeel, Peter Chen, Jonathan Ho, Aravind Srinivas, Alex Li, Wilson Yan

GANs - Pseudocode

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

  for k steps **do**

  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
  - Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

  end for

  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

[Goodfellow et al 2014]

Slide originally by Pieter Abbeel, Peter Chen, Jonathan Ho, Aravind Srinivas, Alex Li, Wilson Yan

GAN

See it in action: https://poloclub.github.io/ganlab/

GAN samples from 2014

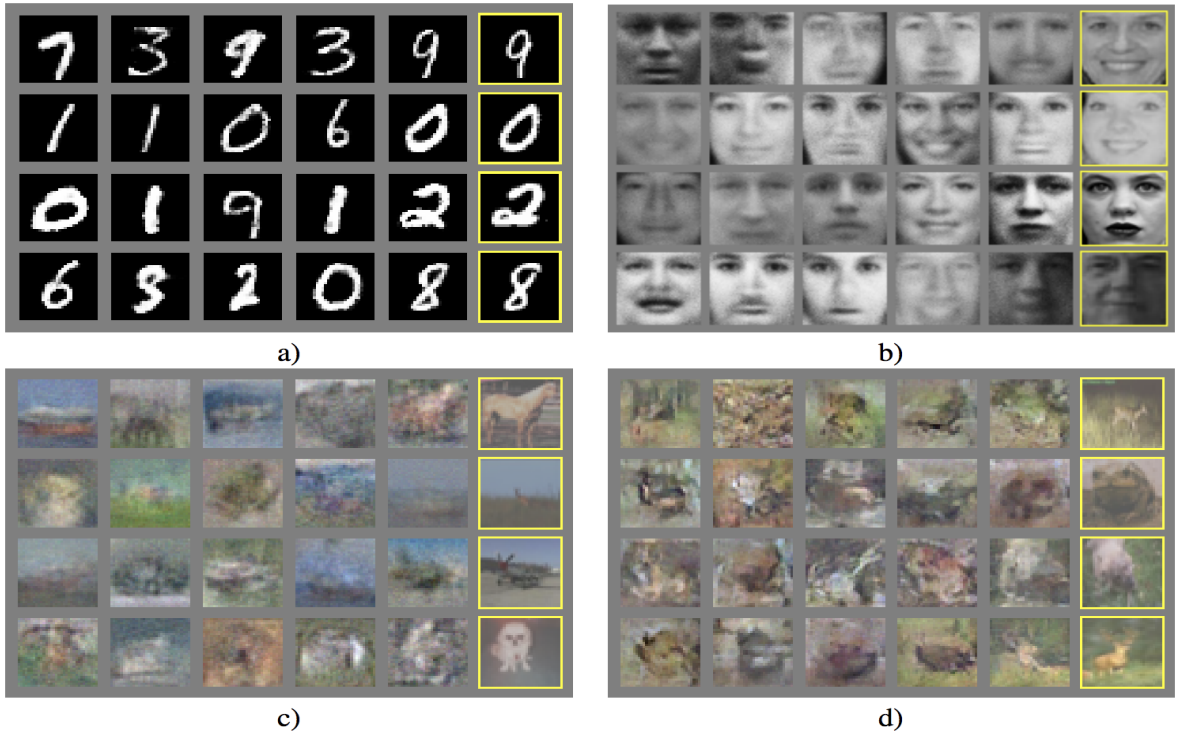


Figure from Goodfellow et al 2014

Slide originally by Pieter Abbeel, Peter Chen, Jonathan Ho, Aravind Srinivas, Alex Li, Wilson Yan

GAN: Bayes-Optimal Discriminator

- What's the optimal discriminator given generated and true distributions?

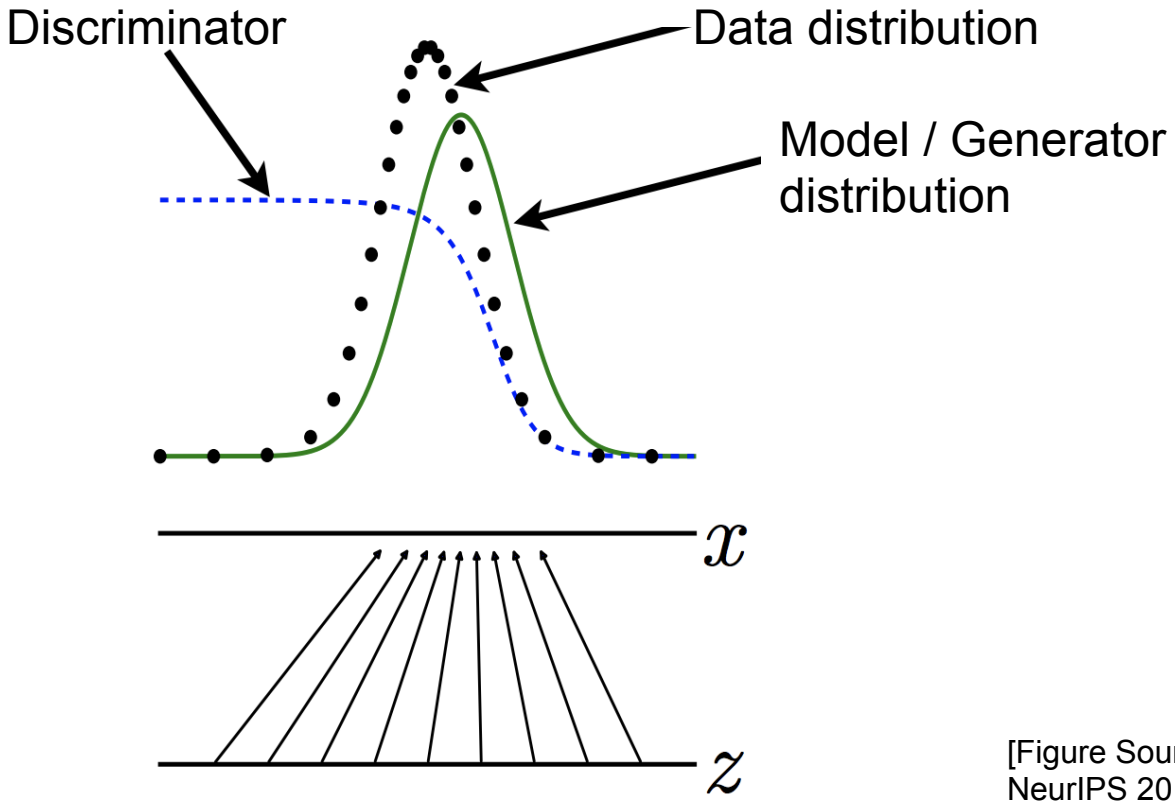
$$
\begin{aligned}
V(G,D) &= \mathbb{E}_{x\sim p_{\text{data}}}\left[\log D(x)\right] + \mathbb{E}_{z\sim p(z)}\left[\log(1-D(G(z)))\right] \\
&= \int_x p_{\text{data}}(x)\log D(x)dx + \int_z p(z)\log(1-D(G(z)))dz \\
&= \int_x p_{\text{data}}(x)\log D(x)dx + \int_x p_g(x)\log(1-D(x))dx \\
&= \int_x \left[p_{\text{data}}(x)\log D(x) + p_g(x)\log(1-D(x))\right]dx
\end{aligned}
$$

$$
\nabla_y \left[a\log y + b\log(1-y)\right] = 0 \implies y^* = \frac{a}{a+b} \quad \forall \quad [a,b]\in\mathbb{R}^2\backslash[0,0]
$$

$$
\implies D^*(x) = \frac{p_{\text{data}}(x)}{(p_{\text{data}}(x)+p_g(x))}
$$

Slide originally by Pieter Abbeel, Peter Chen, Jonathan Ho, Aravind Srinivas, Alex Li, Wilson Yan

GAN: Bayes-Optimal Discriminator



[Figure Source: Goodfellow NeurIPS 2016 Tutorial on GANs]

Behaviors across divergence measures

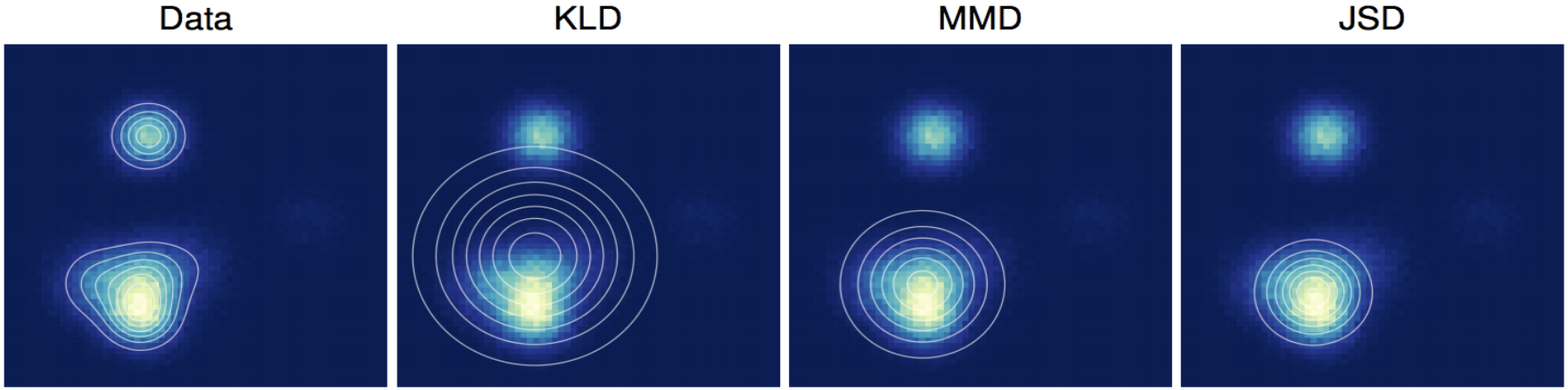
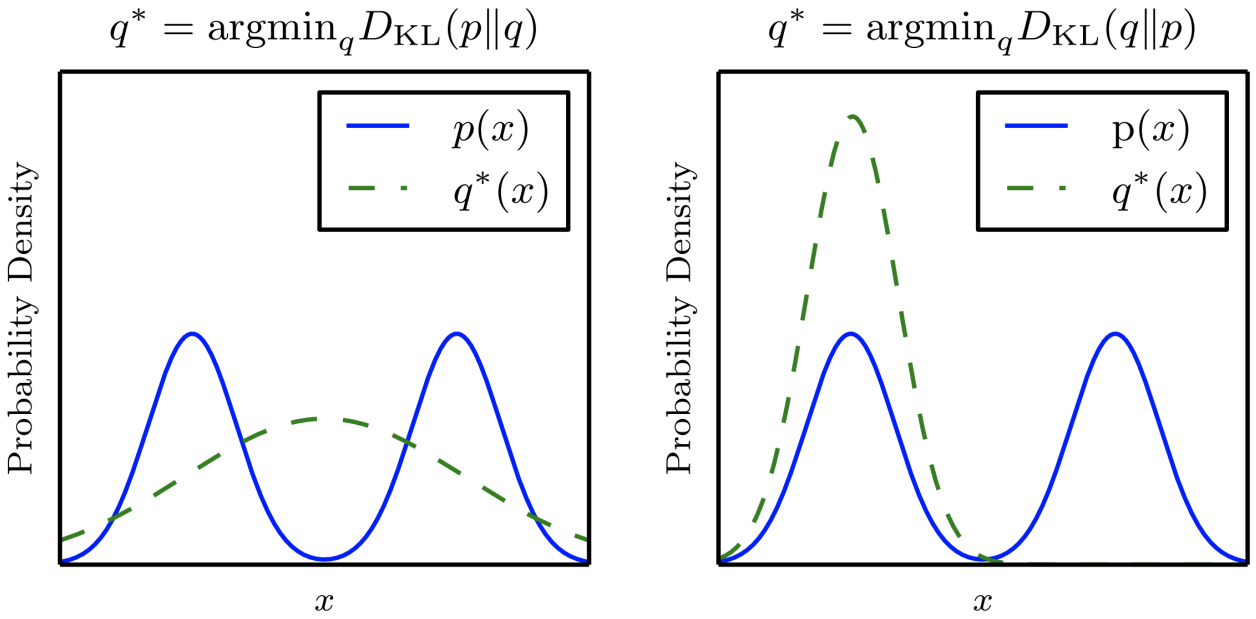


Figure 1: An isotropic Gaussian distribution was fit to data drawn from a mixture of Gaussians by either minimizing Kullback-Leibler divergence (KLD), maximum mean discrepancy (MMD), or Jensen-Shannon divergence (JSD). The different fits demonstrate different tradeoffs made by the three measures of distance between distributions.

["A note on the evaluation of generative models" -- Theis, Van den Oord, Bethge 2015]

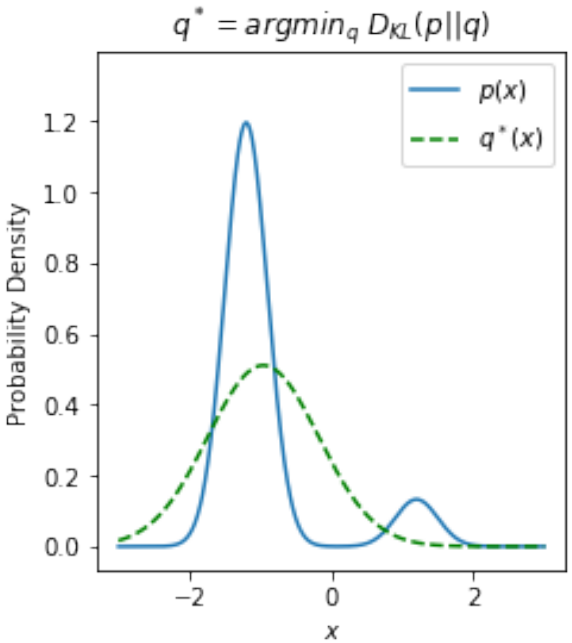
Slide originally by Pieter Abbeel, Peter Chen, Jonathan Ho, Aravind Srinivas, Alex Li, Wilson Yan

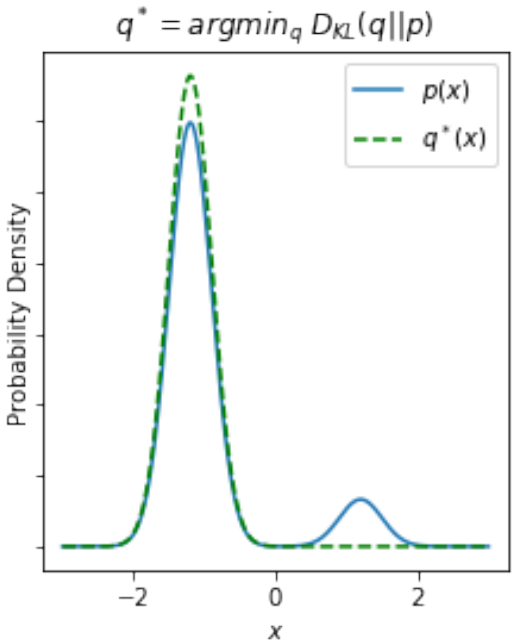
Direction of KL divergence

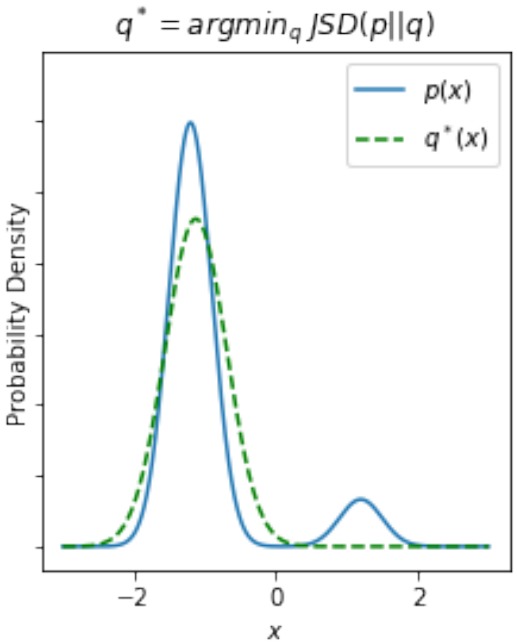


Deep Learning Textbook (Goodfellow 2016)- Chapter 3

KL and JSD







Slide originally by Pieter Abbeel, Peter Chen, Jonathan Ho, Aravind Srinivas, Alex Li, Wilson Yan

KLD = Int{ .. pdata log pdata - pdata log pg (MLE)

$\longrightarrow$ - pdata log pg

JSD = Int{.. pdata log pdata - pdata log [0.5 (pdata + pg)]

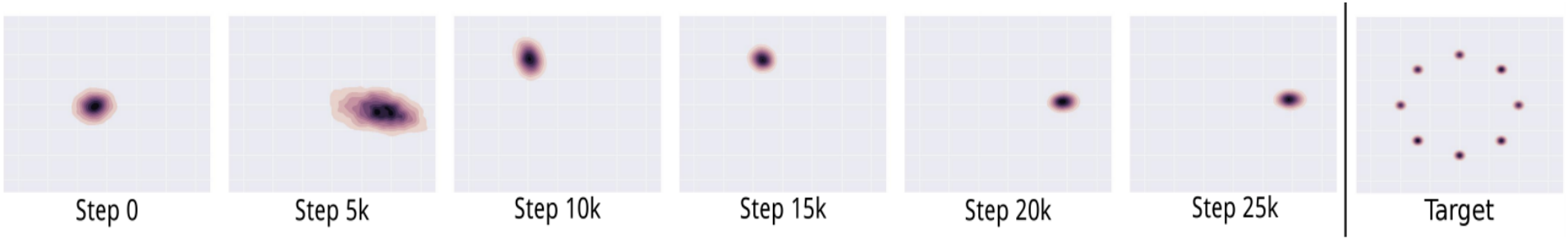
+ pg log pg - pg log [0.5 (pg+pdata)]

Mode covering vs Mode seeking: Tradeoffs

- For compression, one would prefer to ensure all points in the data distribution are assigned probability mass.
- For generating good samples, blurring across modes spoils perceptual quality because regions outside the data manifold are assigned non-zero probability mass.
- Picking one mode without assigning probability mass on points outside can produce “better-looking” samples.
- Caveat: More expressive density models can place probability mass more accurately.
  Example: Using mixture of Gaussians as opposed to a single isotropic gaussian.

Slide originally by Pieter Abbeel, Peter Chen, Jonathan Ho, Aravind Srinivas, Alex Li, Wilson Yan

Mode Collapse



Standard GAN training collapses when the true distribution is a mixture of gaussians (Figure from Metz et al 2016)

Slide originally by Pieter Abbeel, Peter Chen, Jonathan Ho, Aravind Srinivas, Alex Li, Wilson Yan

Back to GANs

Recall

$$\min_G \max_D \underbrace{\mathbb{E}_{x \sim p_{\text{data}}} [\log D(x)] + \mathbb{E}_{z \sim p(z)} [\log(1 - D(G(z)))]}_{\text{Discriminator}}$$

Mini-Exercise

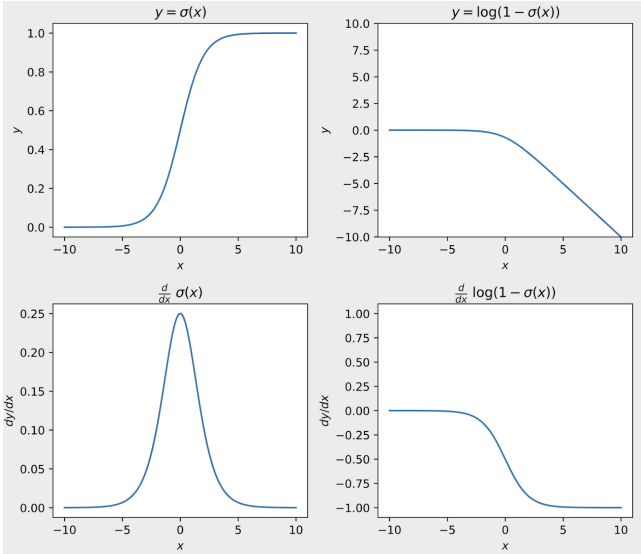
- Is it feasible to run the inner optimization to completion?
- For this specific objective, would it create problems if we were able to do so?

Slide originally by Pieter Abbeel, Peter Chen, Jonathan Ho, Aravind Srinivas, Alex Li, Wilson Yan

Discriminator Saturation

- Generator samples confidently classified as fake by the discriminator receive no gradient for the generator update.

$$\nabla_{G(z)} \log(1 - D(G(z))) \text{ where } D(x) = \text{sigmoid}(x; \theta) = \sigma(x; \theta) \qquad \nabla_x \sigma(x) = \sigma(x)(1 - \sigma(x))$$



Slide originally by Pieter Abbeel, Peter Chen, Jonathan Ho, Aravind Srinivas, Alex Li, Wilson Yan

Avoiding Discriminator Saturation: (1) Alternating Optimization

- Alternate gradient steps on discriminator and generator objectives

$$L^{(D)}(\theta_D, \theta_G) = -\mathbb{E}_{x \sim p_{\text{data}}} [\log D(x; \theta_D)] - \mathbb{E}_{z \sim p(z)} [\log(1 - D(G(z; \theta_G), \theta_D))]$$

$$L^{(G)}(\theta_D, \theta_G) = \mathbb{E}_{z \sim p(z)} [\log(1 - D(G(z; \theta_G), \theta_D))]$$

$$\theta_D := \theta_D - \alpha^{(D)} \nabla_{\theta_D} L^{(D)}(\theta_D, \theta_G)$$

$$\theta_G := \theta_G - \beta^{(G)} \nabla_{\theta_G} L^{(G)}(\theta_D, \theta_G)$$

- Balancing these two updates is hard for the zero sum game

Slide originally by Pieter Abbeel, Peter Chen, Jonathan Ho, Aravind Srinivas, Alex Li, Wilson Yan

Avoiding Discriminator Saturation: (2) Non Saturating Formulation

$$L^{(D)} = -\mathbb{E}_{x \sim p_{\text{data}}} [\log D(x)] - \mathbb{E}_{z \sim p(z)} [\log(1 - D(G(z)))]$$

$$L^{(G)} = -L^{D} \equiv \min_{G} \mathbb{E}_{z \sim p(z)} \log(1 - D(G(z)))$$

Not zero-sum

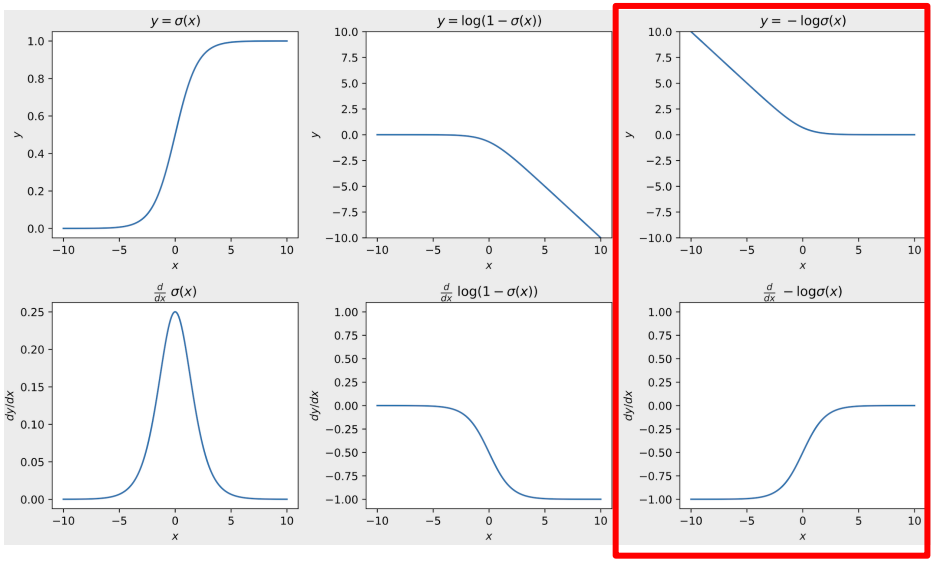
$$L^{(D)} = -\mathbb{E}_{x \sim p_{\text{data}}} [\log D(x)] - \mathbb{E}_{z \sim p(z)} [\log(1 - D(G(z)))]$$

$$L^{(G)} = -\mathbb{E}_{z \sim p(z)} \log(D(G(z)) \equiv \max_{G} \mathbb{E}_{z \sim p(z)} \log(D(G(z))$$

Slide originally by Pieter Abbeel, Peter Chen, Jonathan Ho, Aravind Srinivas, Alex Li, Wilson Yan

Avoiding Discriminator Saturation: (2) Non Saturating Formulation

- ORIGINAL ISSUE: Generator samples confidently classified as fake by the discriminator receive no gradient for the generator update.
- FIX: non-saturating loss for when discriminator confident about fake



Slide originally by Pieter Abbeel, Peter Chen, Jonathan Ho, Aravind Srinivas, Alex Li, Wilson Yan