

Generative Adversarial Networks

Volodymyr Kuleshov

Cornell Tech

Lecture 9

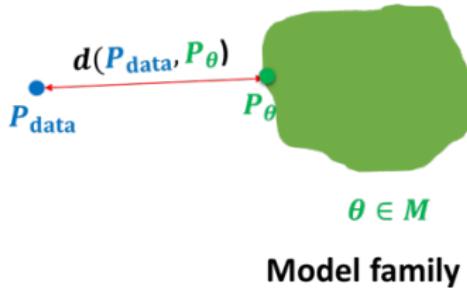
Announcements

- Assignment 2 has been released
- Project proposal is due this Wednesday
- Spreadsheet to sign up for presentations is now available

Recap So Far



$$\mathbf{x}_i \sim P_{\text{data}} \\ i = 1, 2, \dots, n$$



- We saw several generative model families:
 - Autoregressive Models: $p_\theta(\mathbf{x}) = \prod_{i=1}^n p_\theta(x_i | \mathbf{x}_{<i})$
 - Variational Autoencoders: $p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x}, \mathbf{z}) d\mathbf{z}$
 - Normalizing Flow Models: $p_X(\mathbf{x}; \theta) = p_Z(\mathbf{f}_\theta^{-1}(\mathbf{x})) \left| \det \left(\frac{\partial \mathbf{f}_\theta^{-1}(\mathbf{x})}{\partial \mathbf{x}} \right) \right|$
- All the above families are based on maximizing likelihoods (or approximations)
- Is the likelihood the right objective for measuring the similarity of a model to data?

Lecture Outline

① Towards Likelihood-Free Learning

- Motivation
- Two-Sample Tests

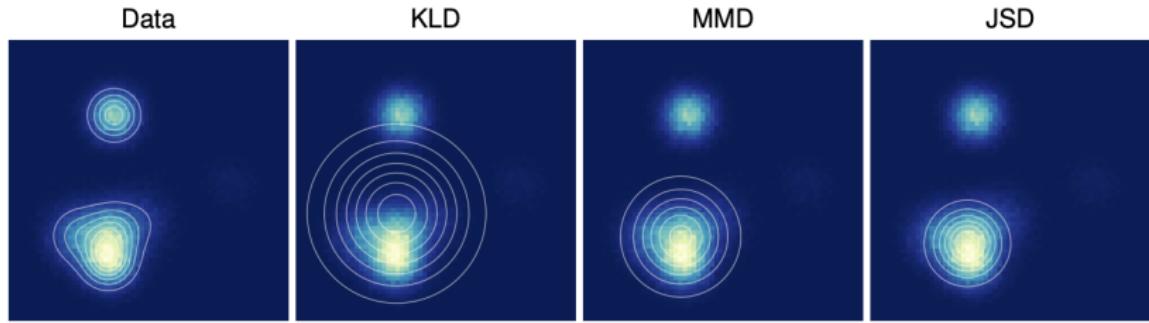
② Generative Adversarial Networks

- Definition
- Objective Functions
- Optimization Issues

Towards likelihood-free learning

What are some pros and cons of using likelihood?

- Optimal generative model will give best **sample quality** and highest test **log-likelihood** (simply because it's the data distribution).
- For imperfect models, achieving high log-likelihoods might not always imply good sample quality, and vice-versa (Theis et al., 2016)
- Likelihood is only one possible metric to define the distance between two distributions



High Likelihood Does Not Imply Great Samples

Example: Great test log-likelihoods, poor samples. Consider a mixture model $p_\theta(\mathbf{x}) = 0.01p_{\text{data}}(\mathbf{x}) + 0.99p_{\text{noise}}(\mathbf{x})$ of data and noise.

- 99% of the samples are just noise
- Taking logs, we get a lower bound

$$\begin{aligned}\log p_\theta(\mathbf{x}) &= \log[0.01p_{\text{data}}(\mathbf{x}) + 0.99p_{\text{noise}}(\mathbf{x})] \\ &\geq \log 0.01p_{\text{data}}(\mathbf{x}) = \log p_{\text{data}}(\mathbf{x}) - \log 100\end{aligned}$$

- For expected likelihoods, we know that
 - Lower bound

$$E_{p_{\text{data}}} [\log p_\theta(\mathbf{x})] \geq E_{p_{\text{data}}} [\log p_{\text{data}}(\mathbf{x})] - \log 100$$

- Upper bound (via non-negativity of KL)

$$E_{p_{\text{data}}} [\log p_{\text{data}}(\mathbf{x})] \geq E_{p_{\text{data}}} [\log p_\theta(\mathbf{x})]$$

- As we increase the dimension of \mathbf{x} , absolute value of $\log p_{\text{data}}(\mathbf{x})$ increases proportionally but $\log 100$ remains constant. Hence, $E_{p_{\text{data}}} [\log p_\theta(\mathbf{x})] \approx E_{p_{\text{data}}} [\log p_{\text{data}}(\mathbf{x})]$ in very high dimensions

Great Samples Don't Imply High Likelihood

Example: Great samples, poor test log-likelihoods. Consider a model that simply memorizes the training set.

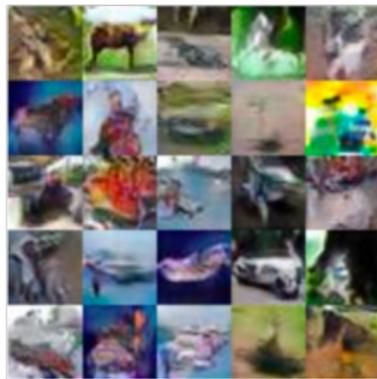
- Samples look exactly like the training set (cannot do better!)
- Test set will have zero probability assigned (cannot do worse!)
- The above cases suggest that it might be useful to disentangle likelihoods and samples
- **Likelihood-free learning** consider objectives that do not depend directly on a likelihood function

Towards Likelihood-Free Learning: Comparing Distributions via Samples

Rather than optimizing the likelihood and hope to get good generative properties, consider the idea of evaluating generative qualities directly.



vs.



$$S_1 = \{\mathbf{x} \sim P\}$$

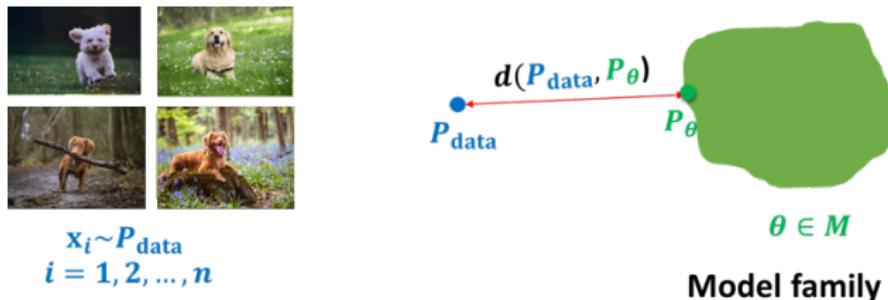
$$S_2 = \{\mathbf{x} \sim Q\}$$

Specifically, given a finite set of samples from two distributions $S_1 = \{\mathbf{x} \sim P\}$ and $S_2 = \{\mathbf{x} \sim Q\}$, how can we tell if these samples are from the same distribution? (i.e., $P = Q$?)

Two-Sample Tests

- Given $S_1 = \{\mathbf{x} \sim P\}$ and $S_2 = \{\mathbf{x} \sim Q\}$, a **two-sample test** considers the following hypotheses
 - Null hypothesis $H_0: P = Q$
 - Alternate hypothesis $H_1: P \neq Q$
- Test statistic T compares S_1 and S_2 e.g., difference in means, variances of the two sets of samples
- If T is less than a threshold α , then accept H_0 else reject it
- Key observation:** Test statistic is **likelihood-free** since it does not involve the densities P or Q (only samples)

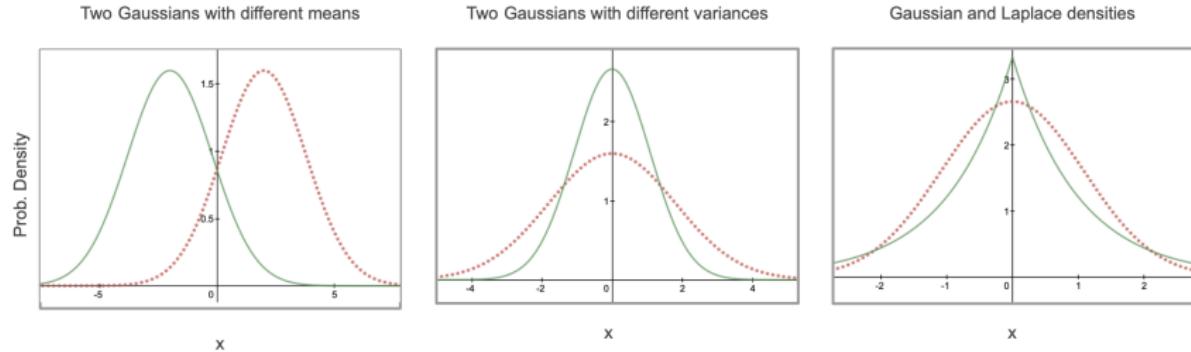
Generative Modeling and Two-Sample Tests



- Apriori we assume direct access to $S_1 = \mathcal{D} = \{\mathbf{x} \sim p_{\text{data}}\}$
- In addition, we have a model distribution p_θ
- Assume that the model distribution permits efficient sampling (e.g., directed models). Let $S_2 = \{\mathbf{x} \sim p_\theta\}$
- **Alternate notion of distance between distributions:** Train the generative model to minimize a two-sample test objective between S_1 and S_2

Two-Sample Test via a Discriminator

- Finding a two-sample test objective in high dimensions is hard



- In the generative model setup, we know that S_1 and S_2 come from different distributions p_{data} and p_{θ} respectively
- Key idea: Learn a statistic that maximizes a suitable notion of distance between the two sets of samples S_1 and S_2**

Lecture Outline

① Towards Likelihood-Free Learning

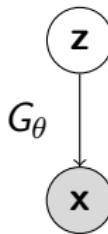
- Motivation
- Two-Sample Tests

② Generative Adversarial Networks

- Definition
- Objective Functions
- Optimization Issues

Generative Adversarial Networks (GANs)

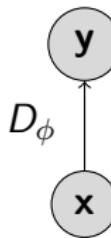
A GAN is defined by two models: a **generator** and a **discriminator**. Consider first the generator.



- The generator defines a directed, latent variable model with a deterministic mapping between z and x given by G_θ
- Directly outputs *samples* x , not parameters for a distribution over x as in a VAE. Hence probability is defined *implicitly*.
- Minimizes a two-sample test objective (in support of the null hypothesis $p_{\text{data}} = p_\theta$)

Generative Adversarial Networks (GANs)

The **discriminator** is a function (e.g., a neural network) which distinguishes “real” samples from the dataset and “fake” samples generated from the model.



- The discriminator maximizes the two-sample test objective (in support of the alternate hypothesis $p_{\text{data}} \neq p_\theta$)
- The generator and the discriminator can be seen as playing a two player minimax game between each other.

Example of GAN Objective

- **Training objective for discriminator:**

$$\max_D V(G, D) = E_{\mathbf{x} \sim p_{\text{data}}} [\log D(\mathbf{x})] + E_{\mathbf{x} \sim p_G} [\log(1 - D(\mathbf{x}))]$$

- For a fixed generator G , the discriminator is performing binary classification with the cross entropy objective
 - Assign probability 1 to true data points $\mathbf{x} \sim p_{\text{data}}$
 - Assign probability 0 to fake samples $\mathbf{x} \sim p_G$
- We can prove that the optimal discriminator equals

$$D_G^*(\mathbf{x}) = \frac{p_{\text{data}}(\mathbf{x})}{p_{\text{data}}(\mathbf{x}) + p_G(\mathbf{x})} = \sigma \left(\log \frac{p_{\text{data}}(\mathbf{x})}{p_G(\mathbf{x})} \right)$$

Unsupervised Learning as Supervised Learning

Suppose we have data coming from a balanced mixture of distributions $P(X|Y = 0)$ and $P(X|Y = 1)$ indexed by Y .

- Let's use supervised learning to predict the component Y from X :

$$\begin{aligned} P(Y = 1|X) &= \frac{P(X|Y = 1)P(Y = 1)}{P(X)} = \frac{P(X|Y = 1)}{P(X|Y = 0) + P(X|Y = 1)} \\ &= \frac{1}{1 + \frac{P(X|Y=0)}{P(X|Y=1)}} = \sigma\left(\log \frac{P(X|Y = 0)}{P(X|Y = 1)}\right) \end{aligned}$$

- Thus, logistic regression trained on $(X, Y) \sim P$ pairs estimates the log odds, which can form the basis of a two-sample test:

$$\log \frac{P(X|Y = 0)}{P(X|Y = 1)}.$$

- Old idea: can be used for outlier detection, density estimation, noise-contrastive learning, etc.

Example of a GAN Objective

- **Training objective for generator:**

$$\min_G V(G, D) = E_{\mathbf{x} \sim p_{\text{data}}} [\log D(\mathbf{x})] + E_{\mathbf{x} \sim p_G} [\log(1 - D(\mathbf{x}))]$$

- For the optimal discriminator $D_G^*(\cdot)$, we have

$$\begin{aligned} & V(G, D_G^*(\mathbf{x})) \\ &= E_{\mathbf{x} \sim p_{\text{data}}} \left[\log \frac{p_{\text{data}}(\mathbf{x})}{p_{\text{data}}(\mathbf{x}) + p_G(\mathbf{x})} \right] + E_{\mathbf{x} \sim p_G} \left[\log \frac{p_G(\mathbf{x})}{p_{\text{data}}(\mathbf{x}) + p_G(\mathbf{x})} \right] \\ &= E_{\mathbf{x} \sim p_{\text{data}}} \left[\log \frac{p_{\text{data}}(\mathbf{x})}{\frac{p_{\text{data}}(\mathbf{x}) + p_G(\mathbf{x})}{2}} \right] + E_{\mathbf{x} \sim p_G} \left[\log \frac{p_G(\mathbf{x})}{\frac{p_{\text{data}}(\mathbf{x}) + p_G(\mathbf{x})}{2}} \right] - \log 4 \\ &= \underbrace{D_{KL} \left[p_{\text{data}}, \frac{p_{\text{data}} + p_G}{2} \right] + D_{KL} \left[p_G, \frac{p_{\text{data}} + p_G}{2} \right]}_{2 \times \text{Jenson-Shannon Divergence (JSD)}} - \log 4 \\ &= 2D_{JSD}[p_{\text{data}}, p_G] - \log 4 \end{aligned}$$

Jenson-Shannon Divergence

- Also called as the symmetric KL divergence

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

- Properties
 - $D_{JSD}[p, q] \geq 0$
 - $D_{JSD}[p, q] = 0$ iff $p = q$
 - $D_{JSD}[p, q] = D_{JSD}[q, p]$
 - $\sqrt{D_{JSD}[p, q]}$ satisfies triangle inequality \rightarrow Jenson-Shannon Distance
- Optimal generator for the JSD/Negative Cross Entropy GAN

$$p_G = p_{\text{data}}$$

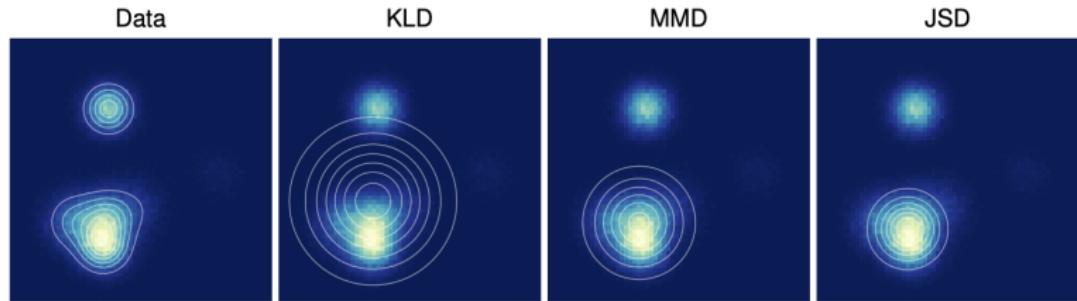
- For the optimal discriminator $D_{G^*}^*(\cdot)$ and generator $G^*(\cdot)$, we have

$$V(G^*, D_{G^*}^*(x)) = -\log 4$$

Jenson-Shannon Divergence

The Jenson-Shannon divergence is mode-seeking.

- Consider a multi-modal data distribution that we are trying to approximate with a unimodal estimator.



- The KL div. (log-likelihood objective) tries to average both modes.

$$\mathbf{D}(P_{\text{data}} || P_{\theta}) = \mathbf{E}_{\mathbf{x} \sim P_{\text{data}}} \left[\log \left(\frac{P_{\text{data}}(\mathbf{x})}{P_{\theta}(\mathbf{x})} \right) \right] = \sum_{\mathbf{x}} P_{\text{data}}(\mathbf{x}) \log \frac{P_{\text{data}}(\mathbf{x})}{P_{\theta}(\mathbf{x})}$$

- The JSD objective favors fitting one mode well.

The GAN Training Algorithm

- Sample minibatch of m training points $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(m)}$ from \mathcal{D}
- Sample minibatch of m noise vectors $\mathbf{z}^{(1)}, \mathbf{z}^{(2)}, \dots, \mathbf{z}^{(m)}$ from p_z
- Update the generator parameters θ by stochastic gradient **descent**

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

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

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

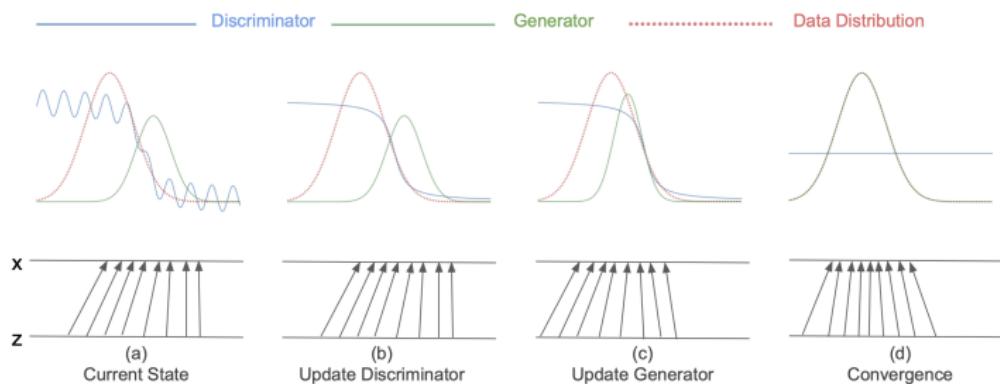
- Repeat for fixed number of epochs

Alternating Optimization in GANs

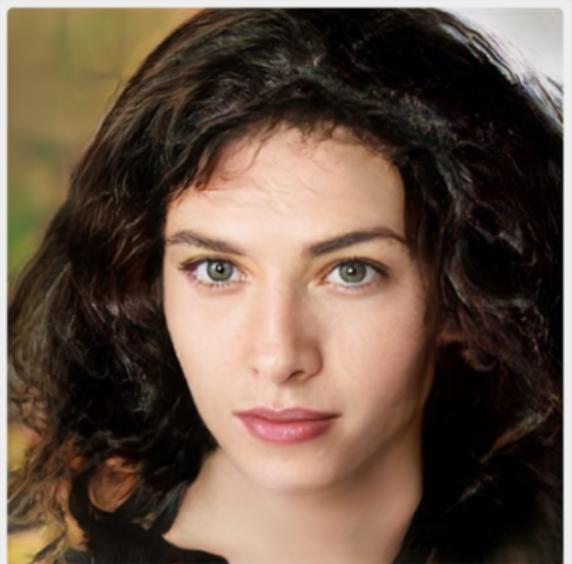
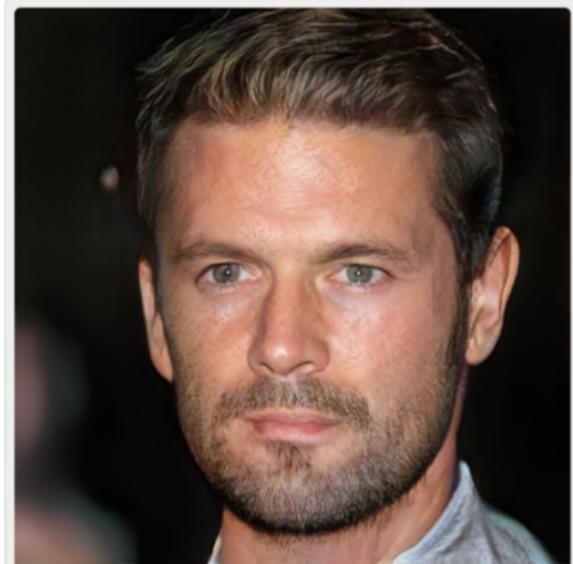
Recall the GAN objective:

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

We can visualize training in 1D via a picture:



Which Photo is Real?



Source: Karras et al., 2018; The New York Times

Both images are generated via GANs!

Frontiers in GAN Research

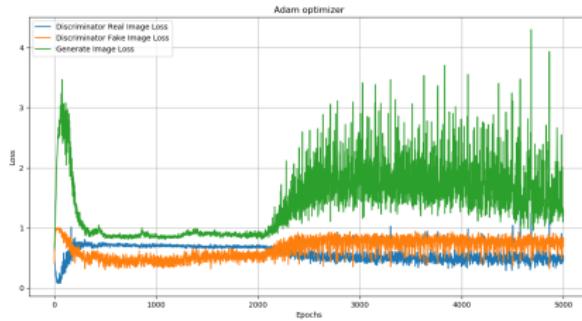


- GANs have been successfully applied to several domains and tasks
- However, working with GANs can be very challenging in practice
 - Unstable optimization
 - Mode collapse
 - Evaluation
- Many bag of tricks applied to train GANs successfully

Image Source: Ian Goodfellow. Samples from Goodfellow et al., 2014, Radford et al., 2015, Liu et al., 2016, Karras et al., 2017, Karras et al., 2018

Optimization Challenges

- **Theorem (informal):** If the generator updates are made in function space and discriminator is optimal at every step, then the generator is guaranteed to converge to the data distribution
- **Unrealistic assumptions!**
- In practice, the generator and discriminator loss keeps oscillating during GAN training (figure by Mirantha Jayathilaka).



- No robust stopping criteria in practice (unlike MLE)

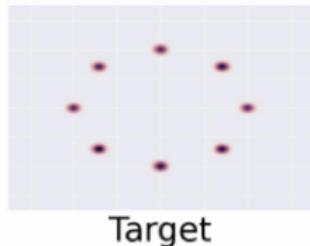
Mode Collapse

- GANs are notorious for suffering from **mode collapse**
- Intuitively, this refers to the phenomena where the generator of a GAN collapses to one or few samples (dubbed as “modes”)

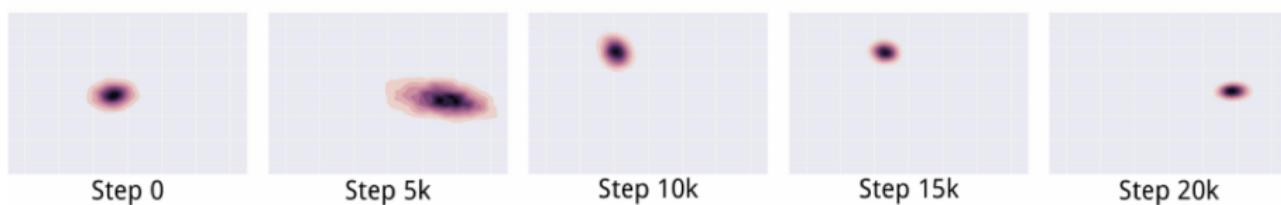


Arjovsky et al., 2017

Mode Collapse



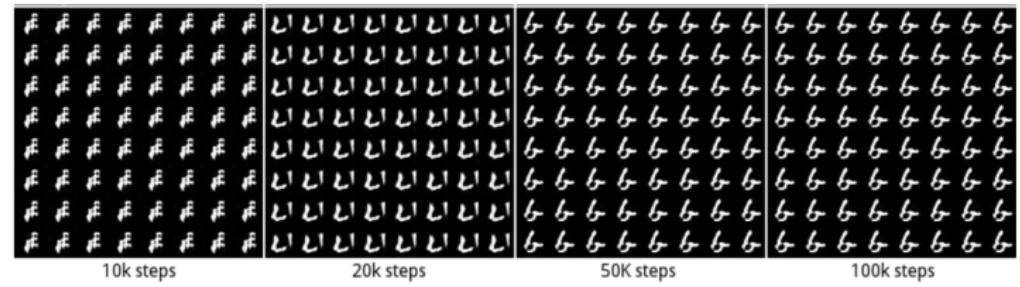
- True distribution is a mixture of Gaussians



Source: Metz et al., 2017

- The generator distribution keeps oscillating between different modes

Mode Collapse



Source: Metz et al., 2017

- Fixes to mode collapse are mostly empirically driven: alternate architectures, adding regularization terms, injecting small noise perturbations etc.
- <https://github.com/soumith/ganhacks>
How to Train a GAN? Tips and tricks to make GANs work by Soumith Chintala

Beauty lies in the eyes of the discriminator



GAN generated art auctioned at Christie's.

Expected Price: \$7,000 – \$10,000

True Price: \$432,500

Summary of GAN Models

- GAN Pros:

- Very high-quality samples.
- Can optimize a wide range of divergences between probabilities (next lecture)
- Broadly applicable: only need sampling from G !

- GAN Cons:

- Doesn't work as well for discrete variables
- Difficult to train
- Suffers from mode collapse