Generative Adversarial Networks

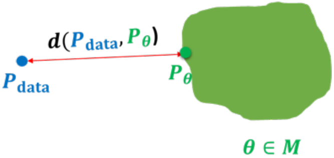
Stefano Ermon

Stanford University

Lecture 9

Recap

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



Model family

- 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: $p_X(\mathbf{x}; \theta) = p_Z\left(\mathbf{f}_\theta^{-1}(\mathbf{x})\right) \left|\det\left(\frac{\partial \mathbf{f}_\theta^{-1}(\mathbf{x})}{\partial \mathbf{x}}\right)\right|$
- All the above families are trained by minimizing KL divergence $D_{KL}(p_{\text{data}}||p_\theta)$, or equivalently maximizing likelihoods (or approximations)

Why maximum 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 $\theta^* \in \mathcal{M}$ that satisfies $p_{\theta^*} = p_{\text{data}}$.
  - The convergence of $\hat{\theta}$ to θ^* when $M \to \infty$ is the "fastest" among all statistical methods when using maximum likelihood training.
- **Higher likelihood = better lossless compression**.
- Is the likelihood a good indicator of the quality of samples generated by the model?

Towards likelihood-free learning

- **Case 1:** Optimal generative model will give best **sample quality** and highest test **log-likelihood**
- For imperfect models, achieving high log-likelihoods might not always imply good sample quality, and vice-versa (Theis et al., 2016)

Towards likelihood-free learning

- **Case 2:** Great test log-likelihoods, poor samples. E.g., For a discrete noise mixture model $p_\theta(\mathbf{x}) = 0.01 p_{\text{data}}(\mathbf{x}) + 0.99 p_{\text{noise}}(\mathbf{x})$
  - 99% of the samples are just noise (most samples are poor)
  - Taking logs, we get a lower bound

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

  - For expected log-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 $D_{KL}(p_{\text{data}} \| p_\theta) \geq 0$)

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

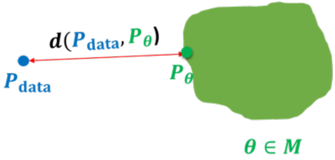
  - As we increase the dimension n of $\mathbf{x} = (x_1, \cdots, x_n)$, absolute value of $\log p_{\text{data}}(\mathbf{x}) = \sum_{i=1}^n \log p_\theta(x_i | \mathbf{x}_{<i})$ increases proportionally to n but $\log 100$ remains constant. Hence, likelihoods are great $E_{p_{\text{data}}}[\log p_\theta(\mathbf{x})] \approx E_{p_{\text{data}}}[\log p_{\text{data}}(\mathbf{x})]$ in very high dimensions

Towards likelihood-free learning

- **Case 3:** Great samples, poor test log-likelihoods. E.g., Memorizing 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 sample quality
- **Likelihood-free learning** consider alternative training objectives that do not depend directly on a likelihood function

Recap

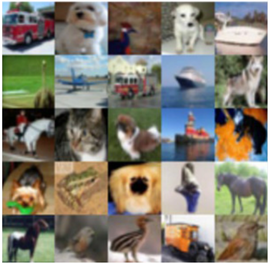
$\mathbf{x}_i \sim P_{\text{data}}$
$i = 1, 2, \ldots, n$



Model family

- Model families
  - Autoregressive Models: $p_\theta(\mathbf{x}) = \prod_{i=1}^{n} p_\theta(x_i|\mathbf{x}_{<i})$
  - Variational Autoencoders: $p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x}, \mathbf{z})dz$
  - Normalizing Flow Models: $p_X(\mathbf{x}; \theta) = p_Z\left(\mathbf{f}_\theta^{-1}(\mathbf{x})\right)\left|\det\left(\frac{\partial \mathbf{f}_\theta^{-1}(\mathbf{x})}{\partial \mathbf{x}}\right)\right|$
- All the above families are trained by minimizing KL divergence $d_{KL}(p_{\mathrm{data}}||p_\theta)$, or equivalently maximizing likelihoods (or approximations)
- Today: alternative choices for $d(p_{\mathrm{data}}||p_\theta)$

Comparing distributions via samples



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

vs.

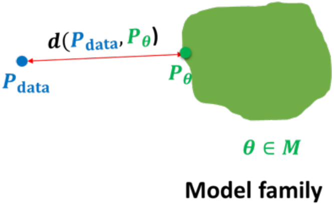
$S_2 = \{\mathbf{x} \sim Q\}$

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: $P = Q$
  - Alternative hypothesis H_1: $P \neq Q$
- Test statistic T compares S_1 and S_2. For example: difference in means, variances of the two sets of samples
  - $T(S_1, S_2) = \left| \frac{1}{|S_1|} \sum_{x \in S_1} x - \frac{1}{|S_2|} \sum_{x \in S_2} x \right|$
- If T is larger than a threshold α, then reject H_0 otherwise we say H_0 is consistent with observation.
- **Key observation:** Test statistic is **likelihood-free** since it does not involve the densities P or Q (only samples)

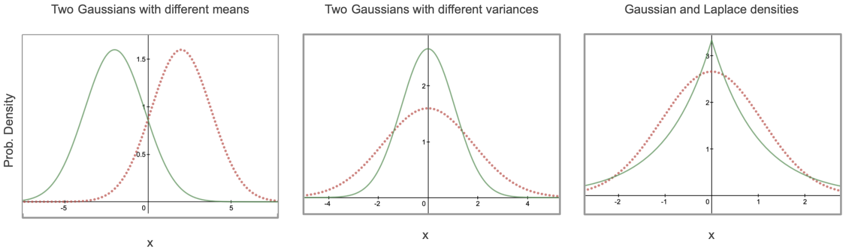
Generative modeling and two-sample tests



- A priori 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 $S_2 = \{\mathbf{x} \sim p_\theta\}$
- **Alternative notion of distance between distributions:** Train the generative model to minimize a two-sample test objective between S_1 and S_2

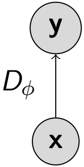
Two-Sample Test via a Discriminator

- Finding a good 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 to automatically identify in what way the two sets of samples S_1 and S_2 differ from each other
- How? Train a classifier (called a discriminator)!

Two-Sample Test via a Discriminator



- **Two-Sample Test via a Discriminator**
  - Any binary classifier D_ϕ (e.g., neural network) which tries to distinguish "real" ($y = 1$) samples from the dataset and "fake" ($y = 0$) samples generated from the model
  - Test statistic: -loss of the classifier. Low loss, real and fake samples are easy to distinguish (different). High loss, real and fake samples are hard to distinguish (similar).
  - Goal: Maximize the two-sample test statistic (in support of the alternative hypothesis $p_{\mathrm{data}} \neq p_\theta$), or equivalently minimize classification loss

Two-Sample Test via a Discriminator

- **Training objective for discriminator**:

$$\max_{D_\phi} V(p_\theta, D_\phi) = E_{\mathbf{x}\sim p_{\rm data}}[\log D_\phi(\mathbf{x})] + E_{\mathbf{x}\sim p_\theta}[\log(1 - D_\phi(\mathbf{x}))]$$
$$\approx \sum_{\mathbf{x}\in S_1} \log D_\phi(\mathbf{x}) + \sum_{\mathbf{x}\in S_2}[\log(1 - D_\phi(\mathbf{x}))]$$

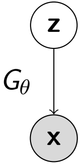
- For a fixed generative model p_θ, the discriminator is performing binary classification with the cross entropy objective
  - Assign probability 1 to true data points $\mathbf{x} \sim p_{\rm data}$ (in set S_1)
  - Assign probability 0 to fake samples $\mathbf{x} \sim p_\theta$ (in set S_2)
- Optimal discriminator

$$D^*_\theta(\mathbf{x}) = \frac{p_{\rm data}(\mathbf{x})}{p_{\rm data}(\mathbf{x}) + p_\theta(\mathbf{x})}$$

- Sanity check: if $p_\theta = p_{\rm data}$, classifier cannot do better than chance ($D^*_\theta(\mathbf{x}) = 1/2$)

Generative Adversarial Networks

- A two player minimax game between a **generator** and a **discriminator**



- **Generator**
  - Directed, latent variable model with a deterministic mapping between $\mathbf{z}$ and $\mathbf{x}$ given by G_θ
    - Sample $\mathbf{z} \sim p(\mathbf{z})$, where $p(\mathbf{z})$ is a simple prior, e.g. Gaussian
    - Set $\mathbf{x} = G_\theta(\mathbf{z})$
  - Similar to a flow model, but mapping G_θ need not be invertible
  - Distribution over $p_\theta(\mathbf{x})$ over $\mathbf{x}$ is implicitly defined (no likelihood!)
  - Minimizes a two-sample test objective (in support of the null hypothesis $p_{\mathrm{data}} = p_\theta$)

Example of GAN objective

- **Training objective for generator**:

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

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

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

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

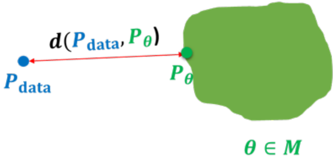
$$p_G = p_{\mathrm{data}}$$

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

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

Recap of GANs

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



Model family

- Choose $d(p_{\text{data}}, p_\theta)$ to be a two-sample test statistic
  - Learn the statistic by training a classifier (discriminator)
  - Under ideal conditions, equivalent to choosing $d(p_{\text{data}}, p_\theta)$ to be $D_{JSD}[p_{\text{data}}, p_\theta]$
- Pros:
  - Loss only requires samples from p_θ. No likelihood needed!
  - Lots of flexibility for the neural network architecture, any G_θ defines a valid sampling procedure
  - Fast sampling (single forward pass)
- Cons: very difficult to train in practice

The GAN training algorithm

- 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**

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

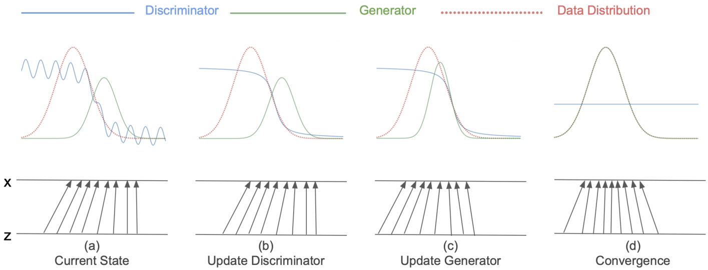
- Update the generator parameters θ by stochastic gradient **descent**

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

- Repeat for fixed number of epochs

Alternating optimization in GANs

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

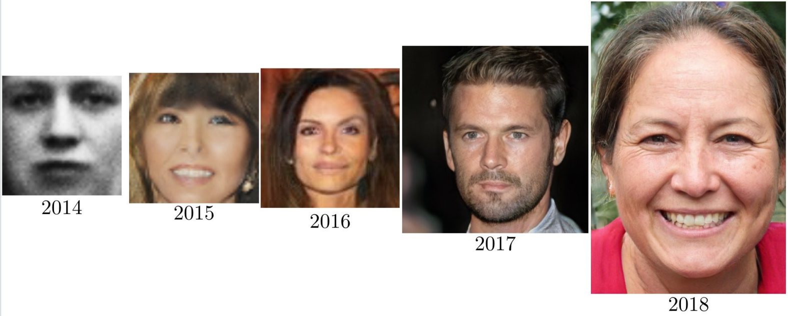


Which one is real?

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

Both images are generated via GANs!

Frontiers in GAN research

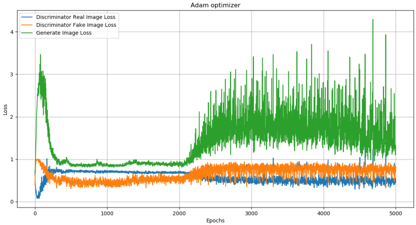


- GANs have been successfully applied to several domains and tasks
- However, working with GANs can be very challenging in practice
  - Unstable optimization
  - Mode collapse
  - Evaluation
- Bag of tricks needed to train GANs successfully

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

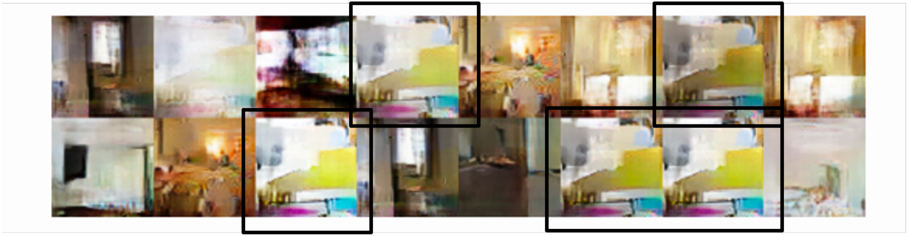


Source: Mirantha Jayathilaka

- No robust stopping criteria in practice (unlike MLE)

Mode Collapse

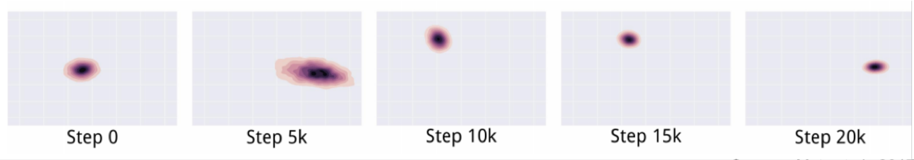
- GANs are notorious for suffering from **mode collapse**
- Intuitively, this refers to the phenomena where the generator of a GAN collapses to one or few samples (dubbed as "modes")



Arjovsky et al., 2017

Mode Collapse

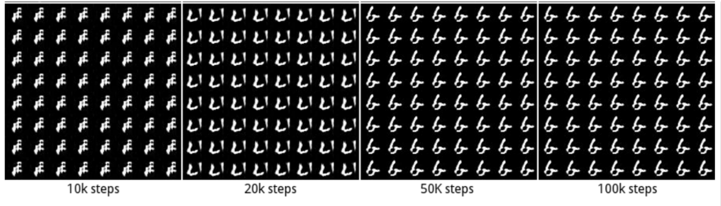
- True distribution is a mixture of Gaussians



Source: Metz et al., 2017

- The generator distribution keeps oscillating between different modes

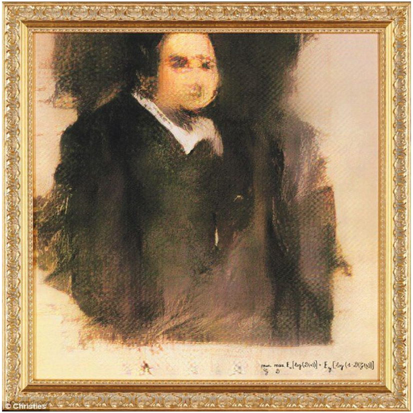
Mode Collapse



Source: Metz et al., 2017

- Fixes to mode collapse are mostly empirically driven: alternative architectures, alternative GAN loss, adding regularization terms, 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



Source: Robbie Barrat, Obvious

GAN generated art auctioned at Christie's.
Expected Price: $\$7,000 - \$10,000$
True Price: $\$432,500$