Generative Adversarial Networks

Volodymyr Kuleshov

Cornell Tech

Lecture 10

Announcements

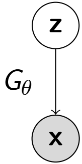
- Project proposal due tonight
  - Submit on Gradescope.
  - Can submit as a team.
- Working on resolving presentation slots

Lecture Outline

1. Optimization Issues in GANs
2. Optimization of f-Divergences
3. Latent Variable Modeling in GANs
4. Domain Translation

Generative Adversarial Networks: Recap

- 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
  - Minimizes a two-sample test objective (in support of the null hypothesis $p_{\mathrm{data}} = p_\theta$)
- **Discriminator**
  - Any function (e.g., neural network) which tries to distinguish "real" samples from the dataset and "fake" samples generated from the model
  - Maximizes the two-sample test objective (in support of the alternate hypothesis $p_{\mathrm{data}} \neq p_\theta$)

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 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

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:
    - Only works for continuous variables
    - Difficult to train
    - Suffers from mode collapse

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

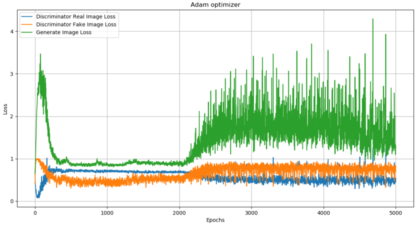
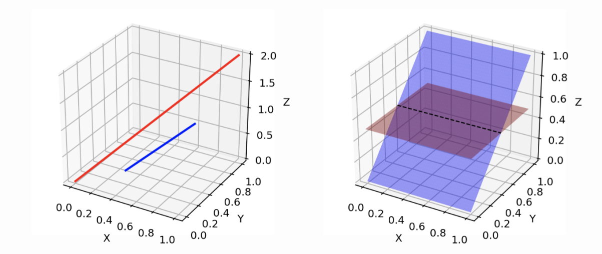


Figure: *

Source: Mirantha Jayathilaka

Low Dimensional Support

- Images are a small subset of all possible $n \times n$ matrices. They represent a small subset of $\mathbb{R}^{n \times n}$.
- Similarly, the manifold of outputs from the generator is also small.



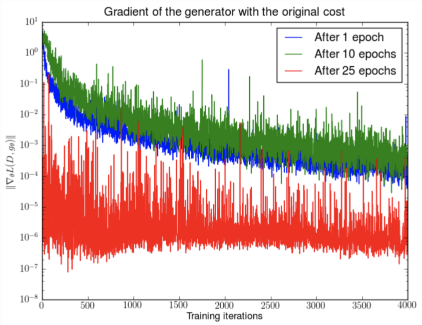
- Hence, their intersection is small (Arjofsky and Bottou, 2017).

Vanishing gradients

- Recall that the GAN objective is

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

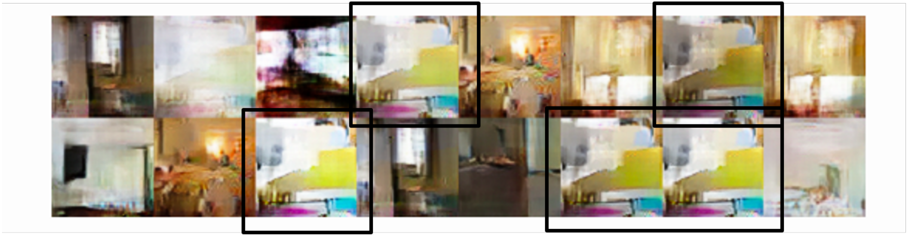
- When discriminator is overconfident, second term is small and has vanishing grads.



- This can happen when the manifolds are disjoint.
- A lot of tricks to address this: change the objective, constrain the power of the GAN, add noise, etc.

Mode Collapse

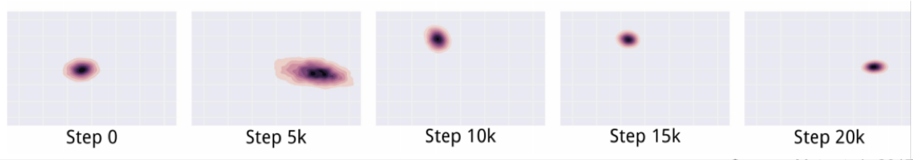
- GANs are notorious for suffering from **mode collapse**
- Intuitively, this refers to the phenomena where the generator of a GAN collapses to one or few samples (dubbed as "modes")



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

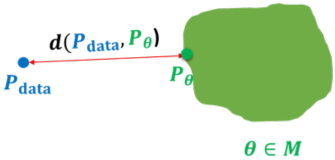
Mode Collapse

Source: Metz et al., 2017

- Fixes to mode collapse are mostly empirically driven:
  - Alternate architectures
  - Feature matching: $E_{\mathbf{x}\sim p_{\text{data}}}[\log F(\mathbf{x})] - E_{\mathbf{x}\sim p_G}[(F(\mathbf{x})]$
  - Label Smoothing: D outputs numbers close but $\neq$ to 0, 1.
  - Adding noise to data: make the manifolds closer to each other.
  - Evaluation criteria based on heuristics and pre-trained vision models.
  - Better metrics of distribution similarity!

Beyond KL and Jenson-Shannon Divergence

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



Model family

What choices do we have for $d(\cdot)$?

- KL divergence: Autoregressive Models, Flow models
- (scaled and shifted) Jenson-Shannon divergence: original GAN objective

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

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

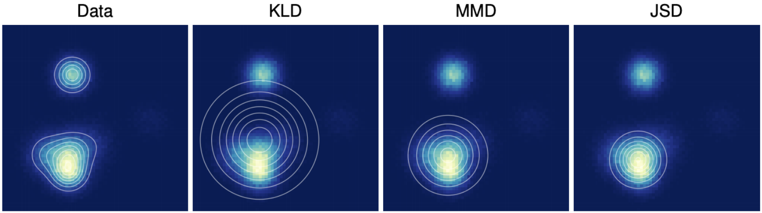
- For the optimal discriminator $D^*_{G^*}(\cdot)$ and generator $G^*(\cdot)$, we have

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

Jenson-Shannon Divergence

The Jenson-Shannon divergence is mode-seeking.

- Consider a multi-modal data distribution that we are trying to approximating with a uni-model estimator.



- The KL divergence (log-likelihood objective) tries to average both modes. The JSD objective favors fitting one mode well. Recall:

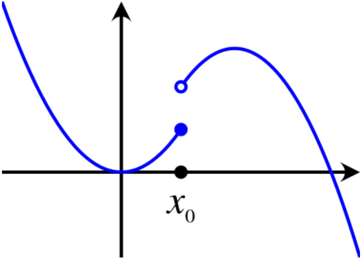
$$\mathbf{D}(P_{\mathrm{data}}||P_\theta) = \mathbf{E}_{\mathbf{x}\sim P_{\mathrm{data}}}\left[\log\left(\frac{P_{\mathrm{data}}(\mathbf{x})}{P_\theta(\mathbf{x})}\right)\right] = \sum_{\mathbf{x}} P_{\mathrm{data}}(\mathbf{x})\log\frac{P_{\mathrm{data}}(\mathbf{x})}{P_\theta(\mathbf{x})}$$

f divergences

- Given two densities p and q, the f-divergence is given by

$$D_f(p,q) = E_{\mathbf{x}\sim q}\left[f\left(\frac{p(\mathbf{x})}{q(\mathbf{x})}\right)\right]$$

  where f is any convex, lower-semicontinuous function with $f(1) = 0$.
- Convex: Line joining any two points lies above the function
- Lower-semicontinuous: function value at any point $\mathbf{x}_0$ is close to $f(\mathbf{x}_0)$ or greater than $f(\mathbf{x}_0)$



- Example: KL divergence with $f(u) = u \log u$

f divergences

Many more f-divergences!

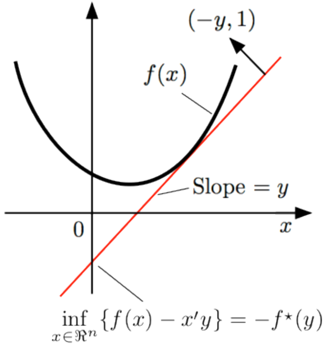
Name	$D_f(P\|Q)$	Generator $f(u)$
Total variation	$\frac{1}{2}\int \lvert p(x)-q(x)\rvert\,\mathrm{d}x$	$\frac{1}{2}\lvert u-1\rvert$
Kullback-Leibler	$\int p(x)\log\frac{p(x)}{q(x)}\,\mathrm{d}x$	$u\log u$
Reverse Kullback-Leibler	$\int q(x)\log\frac{q(x)}{p(x)}\,\mathrm{d}x$	$-\log u$
Pearson χ^2	$\int \frac{(q(x)-p(x))^2}{p(x)}\,\mathrm{d}x$	$(u-1)^2$
Neyman χ^2	$\int \frac{(p(x)-q(x))^2}{q(x)}\,\mathrm{d}x$	$\frac{(1-u)^2}{u}$
Squared Hellinger	$\int \left(\sqrt{p(x)}-\sqrt{q(x)}\right)^2\,\mathrm{d}x$	$(\sqrt{u}-1)^2$
Jeffrey	$\int (p(x)-q(x))\log\left(\frac{p(x)}{q(x)}\right)\,\mathrm{d}x$	$(u-1)\log u$
Jensen-Shannon	$\frac{1}{2}\int p(x)\log\frac{2p(x)}{p(x)+q(x)}+q(x)\log\frac{2q(x)}{p(x)+q(x)}\,\mathrm{d}x$	$-(u+1)\log\frac{1+u}{2}+u\log u$
Jensen-Shannon-weighted	$\int p(x)\pi\log\frac{p(x)}{\pi p(x)+(1-\pi)q(x)}+(1-\pi)q(x)\log\frac{q(x)}{\pi p(x)+(1-\pi)q(x)}\,\mathrm{d}x$	$\pi u\log u-(1-\pi+\pi u)\log(1-\pi+\pi u)$
GAN	$\int p(x)\log\frac{2p(x)}{p(x)+q(x)}+q(x)\log\frac{2q(x)}{p(x)+q(x)}\,\mathrm{d}x-\log(4)$	$u\log u-(u+1)\log(u+1)$
α-divergence ($\alpha\notin\{0,1\}$)	$\frac{1}{\alpha(\alpha-1)}\int\left(p(x)\left[\left(\frac{q(x)}{p(x)}\right)^\alpha-1\right]-\alpha(q(x)-p(x))\right)\mathrm{d}x$	$\frac{1}{\alpha(\alpha-1)}(u^\alpha-1-\alpha(u-1))$

Source: Nowozin et al., 2016

f-GAN: Variational Divergence Minimization

- To use f-divergences as a two-sample test objective for likelihood-free learning, we need to be able to estimate it only via samples
- Fenchel conjugate: For any function $f(\cdot)$, its convex conjugate is defined as

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



f-GAN: Variational Divergence Minimization

- To use f-divergences as a two-sample test objective for likelihood-free learning, we need to be able to estimate it only via samples
- Duality: $f^{**} = f$. When $f(\cdot)$ is convex, lower semicontinous, so is $f^*(\cdot)$

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

- Example: KL divergence with $f(u) = u \log u$

f-GAN: Variational Divergence Minimization

- We can obtain a lower bound to any f-divergence via its Fenchel conjugate

$$
\begin{aligned}
D_f(p,q) &= E_{\mathbf{x}\sim q}\left[f\left(\frac{p(\mathbf{x})}{q(\mathbf{x})}\right)\right] \\
&= E_{\mathbf{x}\sim q}\left[\sup_{t\in \mathrm{dom}_{f^*}}\left(t\frac{p(\mathbf{x})}{q(\mathbf{x})} - f^*(t)\right)\right] \\
&= \int_{\mathcal{X}} \sup_{T\in\mathcal{T}} \left(T(\mathbf{x})p(\mathbf{x}) - f^*(T(\mathbf{x}))q(\mathbf{x})\right) \mathrm{d}\mathbf{x} \\
&\geq \sup_{T\in\mathcal{T}} \int_{\mathcal{X}} \left(T(\mathbf{x})p(\mathbf{x}) - f^*(T(\mathbf{x}))q(\mathbf{x})\right) \mathrm{d}\mathbf{x} \\
&= \sup_{T\in\mathcal{T}} \left(E_{\mathbf{x}\sim p}\left[T(\mathbf{x})\right] - E_{\mathbf{x}\sim q}\left[f^*(T(\mathbf{x}))\right]\right)
\end{aligned}
$$

  where $\mathcal{T}:\mathcal{X}\mapsto\mathbb{R}$ is an arbitrary class of functions

- **Note:** Lower bound is likelihood-free w.r.t. p and q

f-GAN: Variational Divergence Minimization

- Variational lower bound

$$D_f(p, q) \geq \sup_{T \in \mathcal{T}} \left(E_{\mathbf{x} \sim p} [T(\mathbf{x})] - E_{\mathbf{x} \sim q} [f^*(T(\mathbf{x}))] \right)$$

- Choose any f-divergence
- Let $p = p_{\text{data}}$ and $q = p_G$
- Parameterize T by ϕ and G by θ
- Consider the following f-GAN objective

$$\min_\theta \max_\phi F(\theta, \phi) = E_{\mathbf{x} \sim p_{\text{data}}} [T_\phi(\mathbf{x})] - E_{\mathbf{x} \sim p_{G_\theta}} [f^*(T_\phi(\mathbf{x}))]$$

- Generator G_θ tries to minimize the divergence estimate and discriminator T_ϕ tries to tighten the lower bound

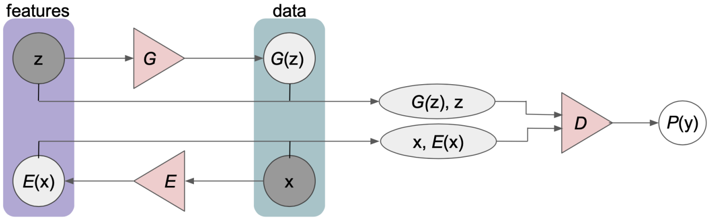
Inferring latent representations in GANs

- The generator of a GAN is typically a directed, latent variable model with latent variables $\mathbf{z}$ and observed variables $\mathbf{x}$ How can we infer the latent feature representations in a GAN?
- Unlike a normalizing flow model, the mapping $G : \mathbf{z} \mapsto \mathbf{x}$ need not be invertible
- Unlike a variational autoencoder, there is no inference network $q(\cdot)$ which can learn a variational posterior over latent variables
- **Solution 1**: For any point $\mathbf{x}$, use the activations of the prefinal layer of a discriminator as a feature representation
- Intuition: Similar to supervised deep neural networks, the discriminator would have learned useful representations for $\mathbf{x}$ while distinguishing real and fake $\mathbf{x}$

Inferring latent representations in GANs

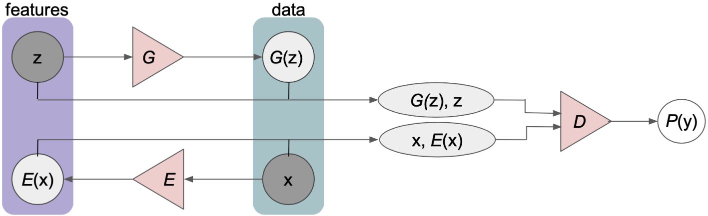
- If we want to directly infer the latent variables $\mathbf{z}$ of the generator, we need a different learning algorithm
- A regular GAN optimizes a two-sample test objective that compares samples of $\mathbf{x}$ from the generator and the data distribution
- **Solution 2**: To infer latent representations, we will compare samples of $\mathbf{x}, \mathbf{z}$ from the joint distributions of observed and latent variables as per the model and the data distribution
- For any $\mathbf{x}$ generated via the model, we have access to $\mathbf{z}$ (sampled from a simple prior $p(\mathbf{z})$)
- For any $\mathbf{x}$ from the data distribution, the $\mathbf{z}$ is however unobserved (latent)

Bidirectional Generative Adversarial Networks (BiGAN)



- In a BiGAN, we have an encoder network E in addition to the generator network G
- The encoder network only observes $\mathbf{x} \sim p_{\text{data}}(\mathbf{x})$ during training to learn a mapping $E : \mathbf{x} \mapsto \mathbf{z}$
- As before, the generator network only observes the samples from the prior $\mathbf{z} \sim p(\mathbf{z})$ during training to learn a mapping $G : \mathbf{z} \mapsto \mathbf{x}$

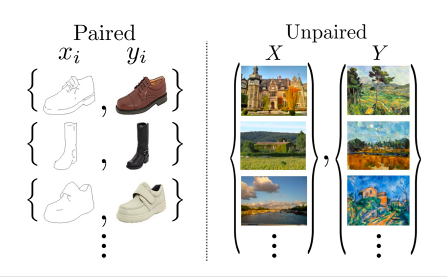
Bidirectional Generative Adversarial Networks (BiGAN)



- The discriminator D observes samples from the generative model $\mathbf{z}, G(\mathbf{z})$ and the encoding distribution $E(\mathbf{x}), \mathbf{x}$
- The goal of the discriminator is to maximize the two-sample test objective between $\mathbf{z}, G(\mathbf{z})$ and $E(\mathbf{x}), \mathbf{x}$
- After training is complete, new samples are generated via G and latent representations are inferred via E

Translating across domains

- Image-to-image translation: We are given images from two domains, $\mathcal{X}$ and $\mathcal{Y}$
- Paired vs. unpaired examples

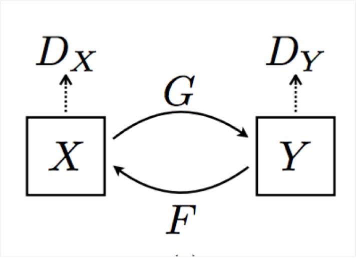


Source: Zhu et al., 2016

- Paired examples can be expensive to obtain. Can we translate from $\mathcal{X} \leftrightarrow \mathcal{Y}$ in an unsupervised manner?

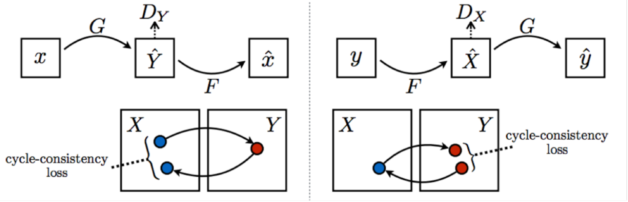
CycleGAN: Adversarial training across two domains

- To match the two distributions, we learn two parameterized conditional generative models $G : \mathcal{X} \to \mathcal{Y}$ and $F : \mathcal{Y} \to \mathcal{X}$
- G maps an element of $\mathcal{X}$ to an element of $\mathcal{Y}$. A discriminator $D_{\mathcal{Y}}$ compares the observed dataset Y and the generated samples $\hat{Y} = G(X)$
- Similarly, F maps an element of $\mathcal{Y}$ to an element of $\mathcal{X}$. A discriminator $D_{\mathcal{X}}$ compares the observed dataset X and the generated samples $\hat{X} = F(Y)$



Source: Zhu et al., 2016

- **Cycle consistency:** If we can go from X to $\hat{Y}$ via G, then it should also be possible to go from $\hat{Y}$ back to X via F
  - $F(G(X)) \approx X$
  - Similarly, vice versa: $G(F(Y)) \approx Y$

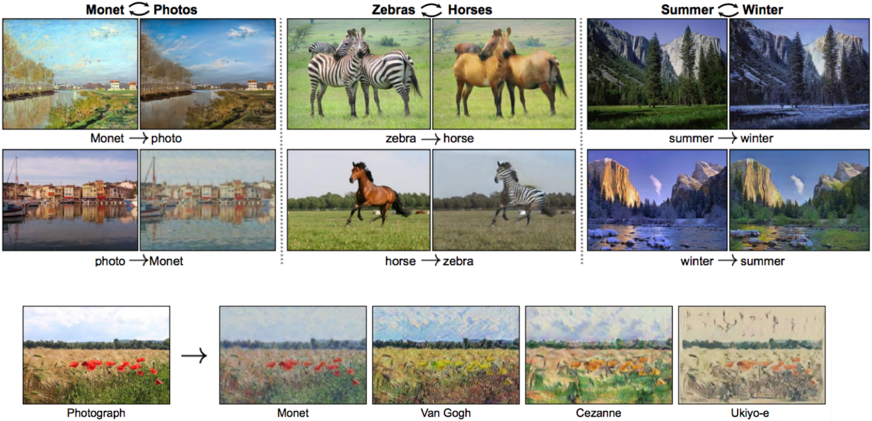


Source: Zhu et al., 2016

- Overall loss function

$$\min_{F,G,D_{\mathcal{X}},D_{\mathcal{Y}}} \mathcal{L}_{\text{GAN}}(G, D_{\mathcal{Y}}, X, Y) + \mathcal{L}_{\text{GAN}}(F, D_{\mathcal{X}}, X, Y)$$
$$+\lambda \underbrace{\left(E_X[\|F(G(X)) - X\|_1] + E_Y[\|G(F(Y)) - Y\|_1]\right)}_{\text{cycle consistency}}$$

CycleGAN in practice



Source: Zhu et al., 2016

CycleGAN in practice

CycleGANs can also be applied to movies.

CycleGAN in practice

CycleGAN in practice

AlignFlow

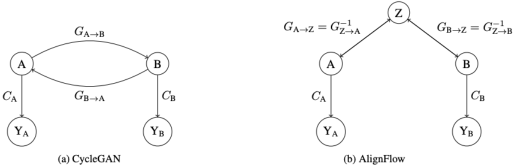


Figure 1: CycleGAN v.s. AlignFlow for unpaired cross-domain translation. Unlike CycleGAN, AlignFlow specifies a single invertible mapping $G_{A\to Z} \circ G^{-1}_{B\to Z}$ that is exactly cycle-consistent, represents a shared latent space Z between the two domains, and can be trained via both adversarial training and exact maximum likelihood estimation. Double-headed arrows denote invertible mappings. Y_A and Y_B are random variables denoting the output of the critics used for adversarial training.

- What if G is a flow model?
- No need to parameterize F separately! $F = G^{-1}$
- Can train via MLE and/or adversarial learning!
- Exactly cycle-consistent
  F(G(X)) = X
  G(F(Y)) = Y

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:
  - Only works for continuous variables
  - Difficult to train
  - Suffers from mode collapse

Summary of Generative Adversarial Networks

- Key observation: Samples and likelihoods are not correlated in practice
- Two-sample test objectives allow for learning generative models only via samples (likelihood-free)
- Wide range of two-sample test objectives covering f-divergences (and more)
- Latent representations can be inferred via BiGAN
- Cycle-consistent domain translations via CycleGAN and AlignFlow