Carnegie Mellon

School of Computer Science

Deep Reinforcement Learning and Control

Adversarial imitation learning, Goal-conditioned Imitation learning

Fall 2020, CMU 10-703

Katerina Fragkiadaki

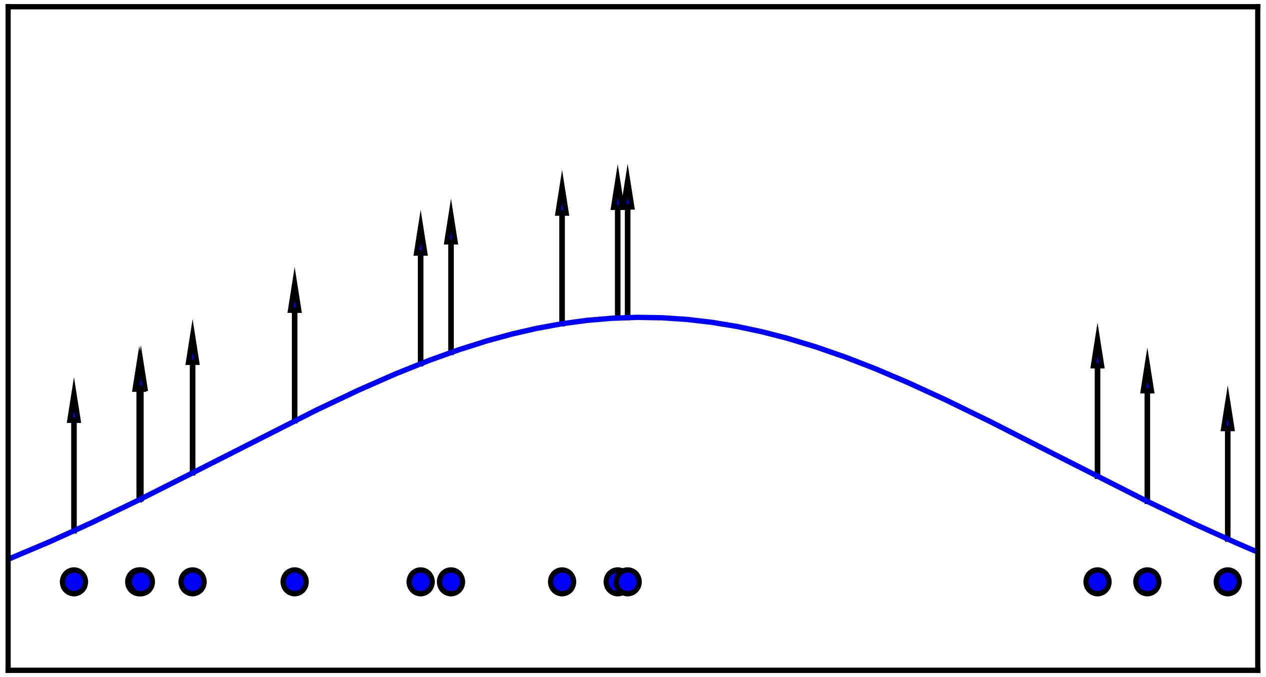
Last lecture

- Behaviour cloning for imitation learning. Assumes access to a set of trajectories $\mathscr{T} = \{o_1^j, a_1^j, o_2^j, a_2^j, o_3^j, a_3^j, \ldots, o_T^j, a_T^j, j = 1 \ldots T\}$. Trains a policy by minimizing a standard supervised learning objective:

$$\mathscr{L}_{BC}(\theta, \mathscr{T}) = \mathbb{E}_{(s_t^j, a_t^j) \sim \mathscr{T}} \left[\| a_t^j - \pi_\theta(s_t^j) \|_2^2 \right]$$

- Self-supervised visual feature learning to train policies from images directly using a keypoint bottleneck comprised of (x,y) coordinates of a set of keypoints.

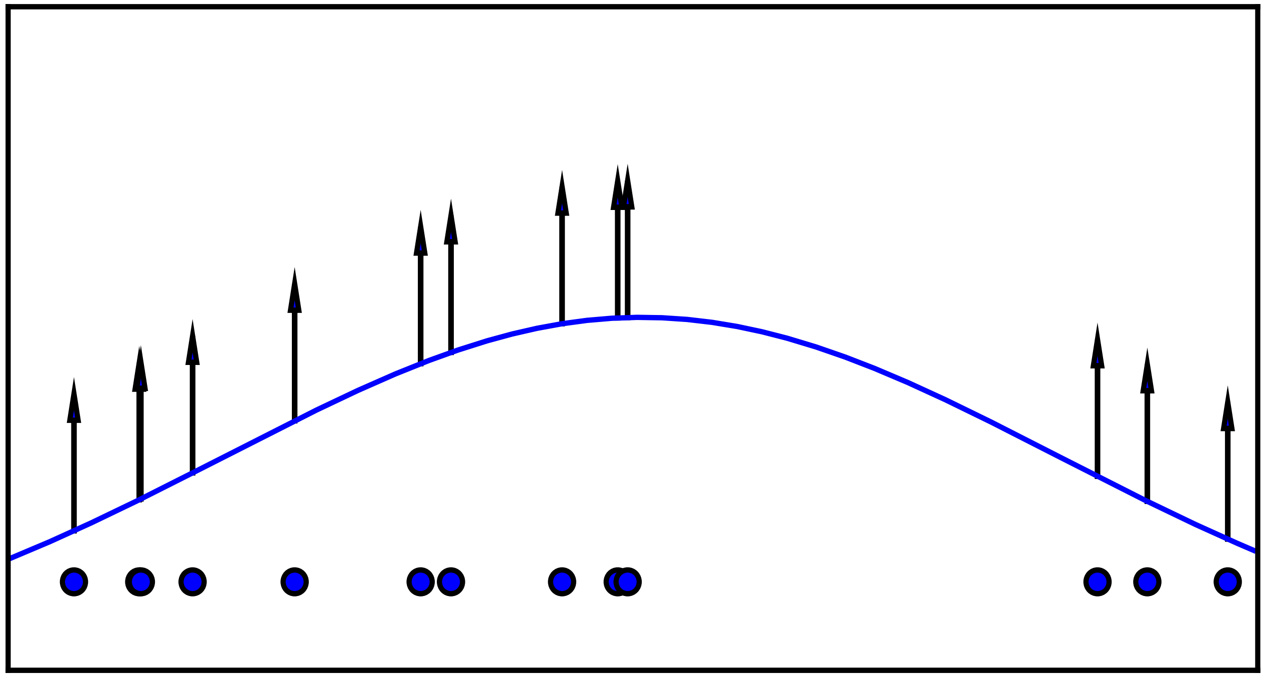
Maximum Likelihood



$$\theta^* = \arg\max_\theta \mathbb{E}_{x\sim p_{\text{data}}} \log p_{\text{model}}(\mathbf{x} \mid \theta)$$

$$\theta^* = \arg\max_\theta \sum_{i=1}^{N} \log p_{\text{model}}(\mathbf{x}_i \mid \theta)$$

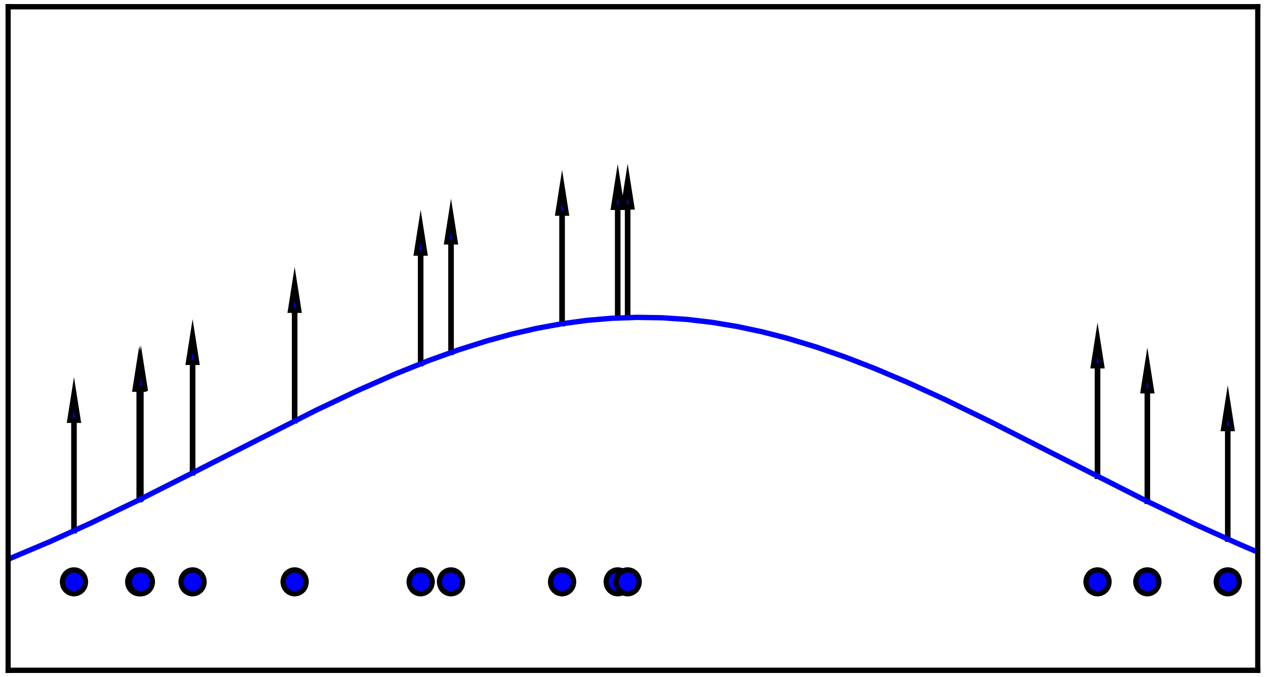
Maximum Likelihood



$$\theta^* = \arg\max_{\theta} \mathbb{E}_{x\sim p_{\text{data}}} \log p_{\text{model}}(\mathbf{x} \mid \theta)$$

explicit density

Maximum Conditional Likelihood



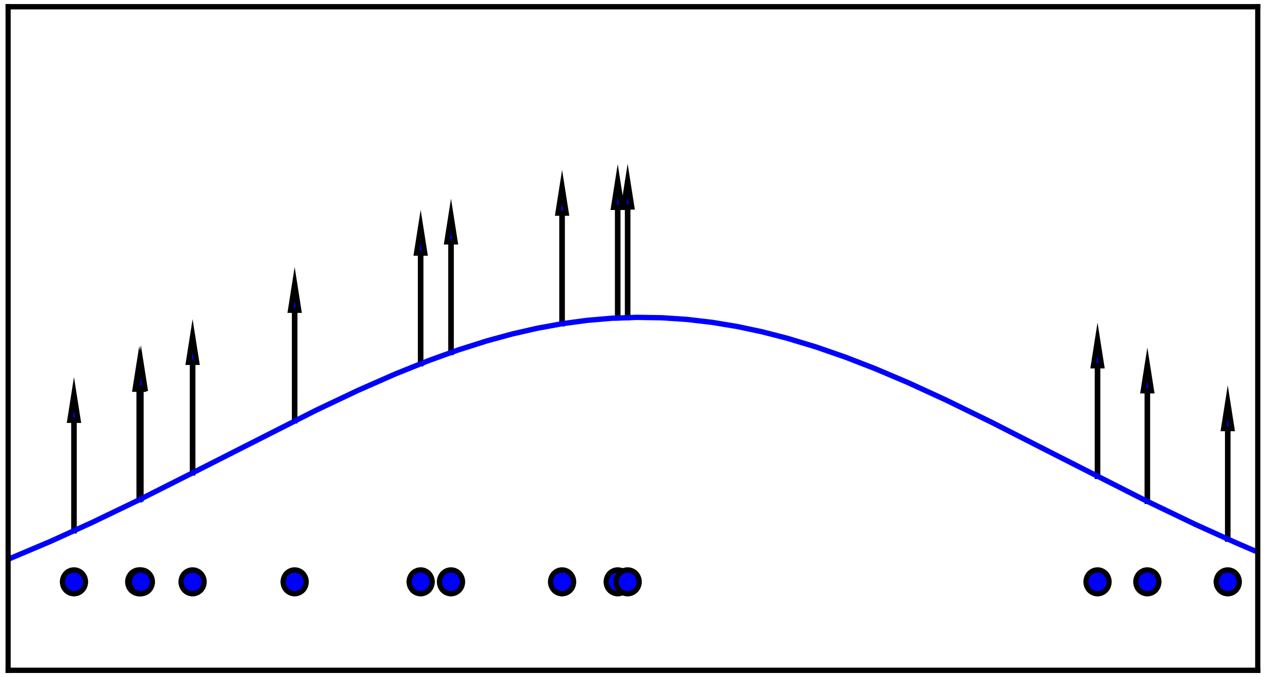
$$\theta^* = \arg\max_{\theta} \mathbb{E}_{x \sim p_{\text{data}}} \log p_{\text{model}}(\mathbf{x} \mid \theta, c)$$

explicit density

extra conditioning information

Maximum Conditional Likelihood

$$D_{KL}(P \| Q) = -\sum_{x \in X} P(x) \log \left(\frac{Q(x)}{P(x)} \right)$$

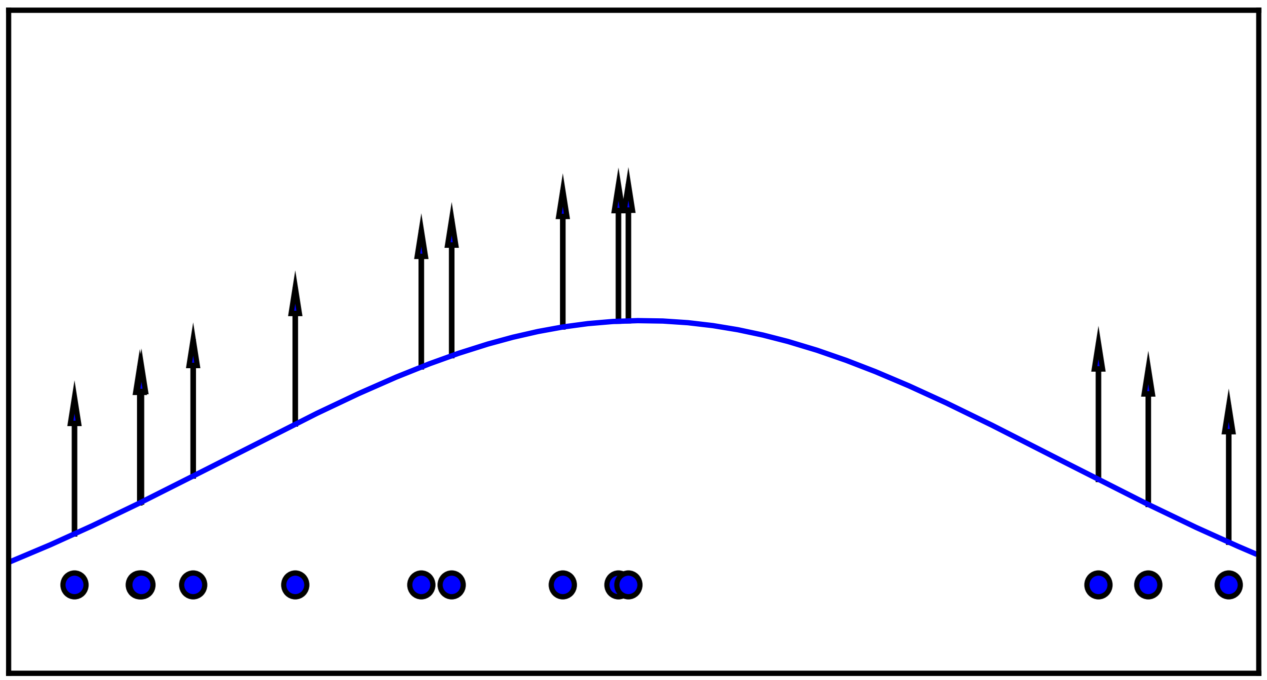


$$\theta^* = \arg\max_{\theta} \mathbb{E}_{x \sim p_{\text{data}}} \log p_{\text{model}}(\mathbf{x} \mid \theta, c)$$

equiv. to

$$\theta^* = \arg\min_{\theta} D_{KL}\left(p_{\text{data}} \| p_{\text{model}}(\mathbf{x} \mid \theta, c)\right)$$

Maximum Likelihood-Gaussian with fixed covariance

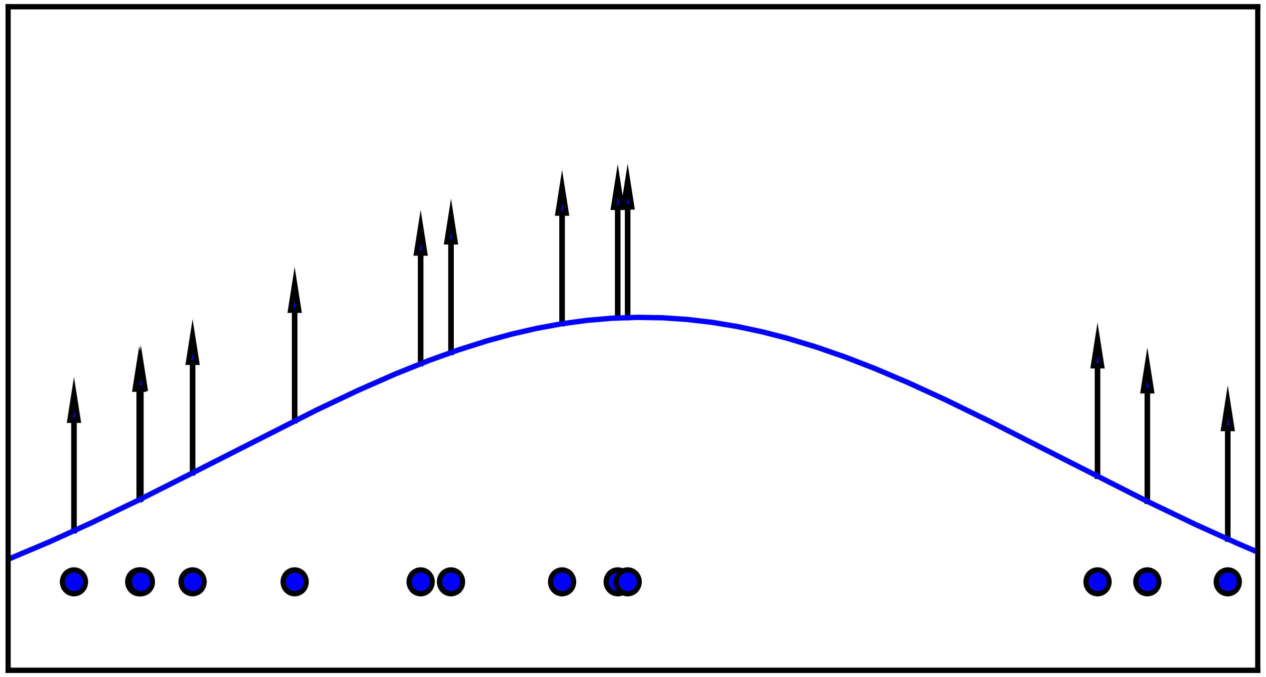


$$\theta^* = \arg\max_{\theta} \mathbb{E}_{x \sim p_{\text{data}}} \log p_{\text{model}}(\mathbf{x} \mid \theta, c)$$

$$p_{\text{model}}(\mathbf{x} \mid \theta, c) = \frac{1}{(2\pi)^{-\frac{k}{2}} \det(\Sigma)^{-\frac{1}{2}}} \exp\left(-\frac{1}{2}(\mathbf{x} - \mu(\theta, \mathbf{c}))^\top \Sigma^{-1} (\mathbf{x} - \mu(\theta, \mathbf{c}))\right), \text{where } \Sigma = \mathrm{I}$$

Maximum Likelihood-Gaussian with fixed covariance

$$p_{\text{model}}(\mathbf{x} \mid \theta, c) = \frac{1}{(2\pi)^{-\frac{k}{2}} \det(\Sigma)^{-\frac{1}{2}}} \exp\left(-\frac{1}{2}(\mathbf{x} - \mu(\theta, \mathbf{c}))^{\top} \Sigma^{-1} (\mathbf{x} - \mu(\theta, \mathbf{c}))\right), \text{ where } \Sigma = \mathrm{I}$$



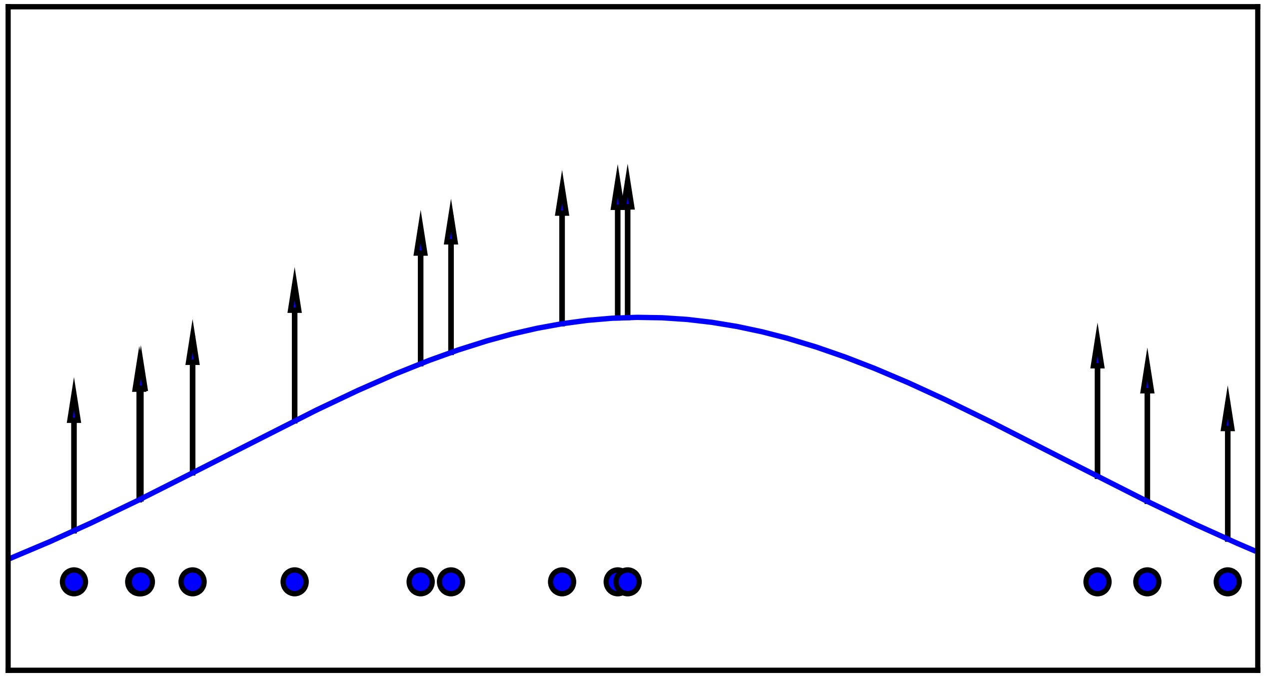
$$\theta^* = \arg\max_{\theta} \mathbb{E}_{x \sim p_{\text{data}}} \log p_{\text{model}}(\mathbf{x} \mid \theta, c)$$

$$\max_{\theta} \mathbb{E}_{x \sim p_{\text{data}}} \log p_{\text{model}}(\mathbf{x} \mid \theta, c)$$ equiv. to $$\min_{\theta} \mathbb{E}_{x \sim p_{\text{data}}} \|\mathbf{x} - \mu(\theta, \mathbf{c})\|_2^2$$

e.g. behavior cloning with continuous actions

$$\mathcal{L}_{BC}(\theta, \mathcal{T}) = \mathbb{E}_{(s_t^j, a_t^j) \sim \mathcal{T}} \left[\|a_t^j - \pi_\theta(s_t^j)\|_2^2 \right]$$

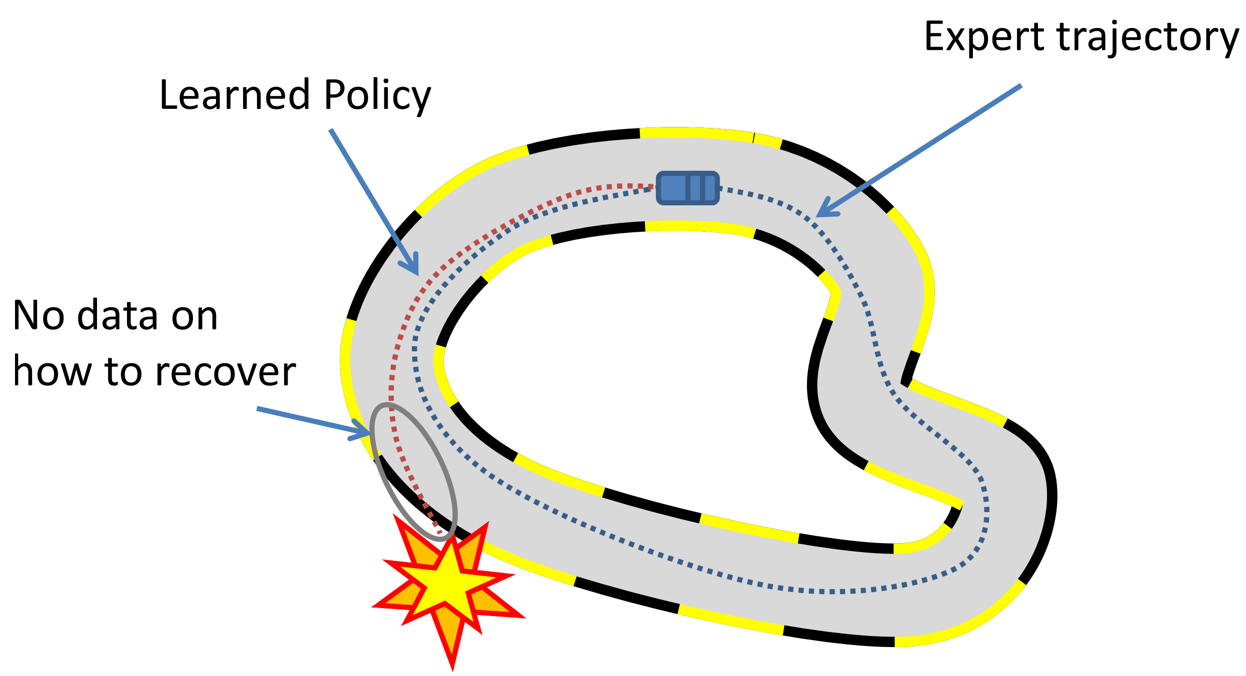
BC Maximizes Conditional Likelihood



$$\mathscr{L}_{BC}(\theta, \mathscr{T}) = \mathbb{E}_{(s_t^j, a_t^j) \sim \mathscr{T}} \left[\| a_t^j - \pi_\theta(s_t^j) \|_2^2 \right]$$

BC Maximizes Conditional Likelihood

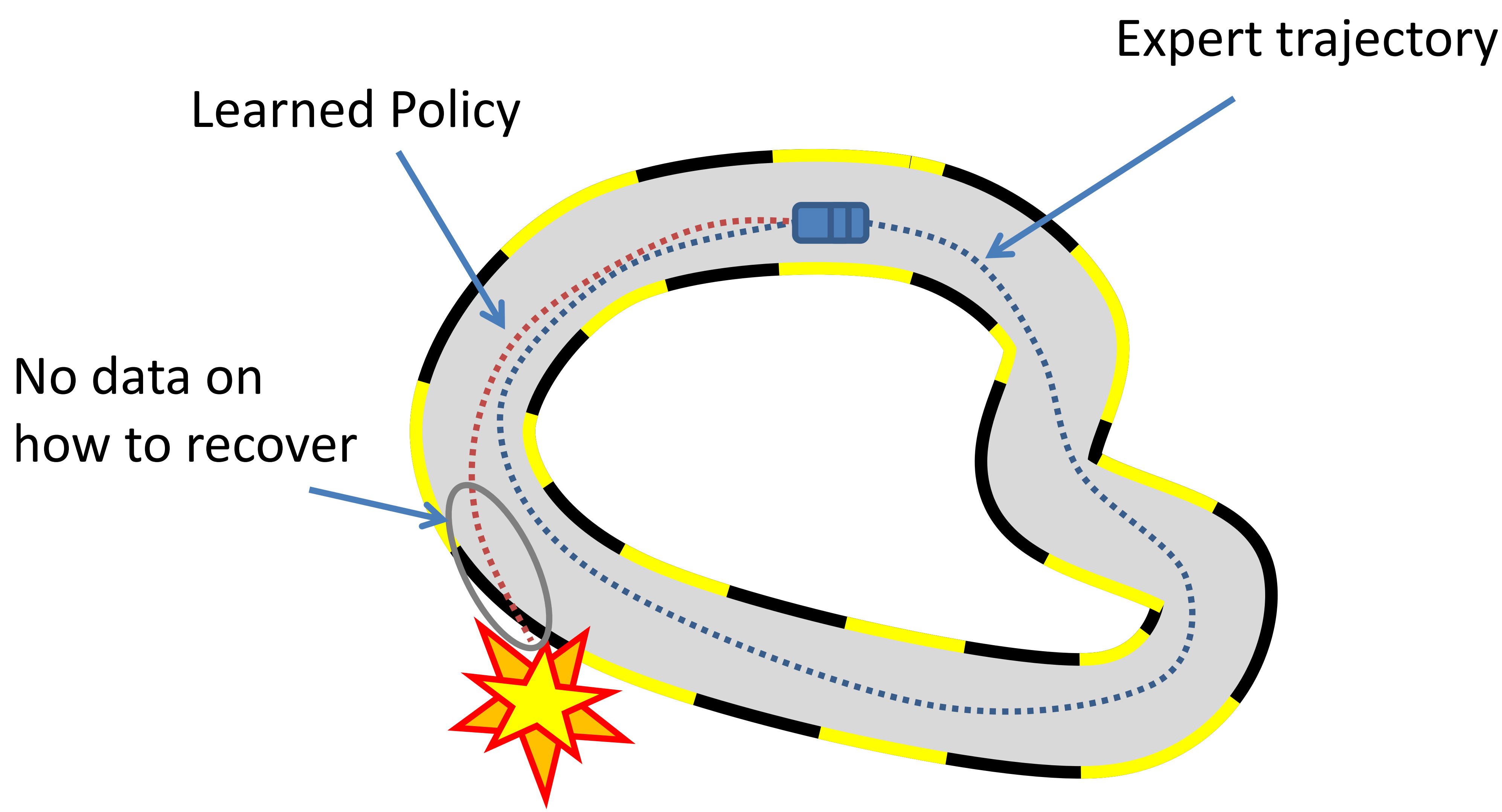
$$\mathscr{L}_{BC}(\theta, \mathscr{T}) = \mathbb{E}_{(s_t^j, a_t^j)\sim\mathscr{T}}\left[\|a_t^j - \pi_\theta(s_t^j)\|_2^2\right]$$



- Makes the expert actions most likely in the states of the expert trajectories.
- But what about the states not on the expert trajectories? There the actions are unconstrained!

Distribution mismatch (distribution shift)

$$P_{\pi^*}(\mathbf{o}_t) \neq P_{\pi_\theta}(\mathbf{o}_t)$$

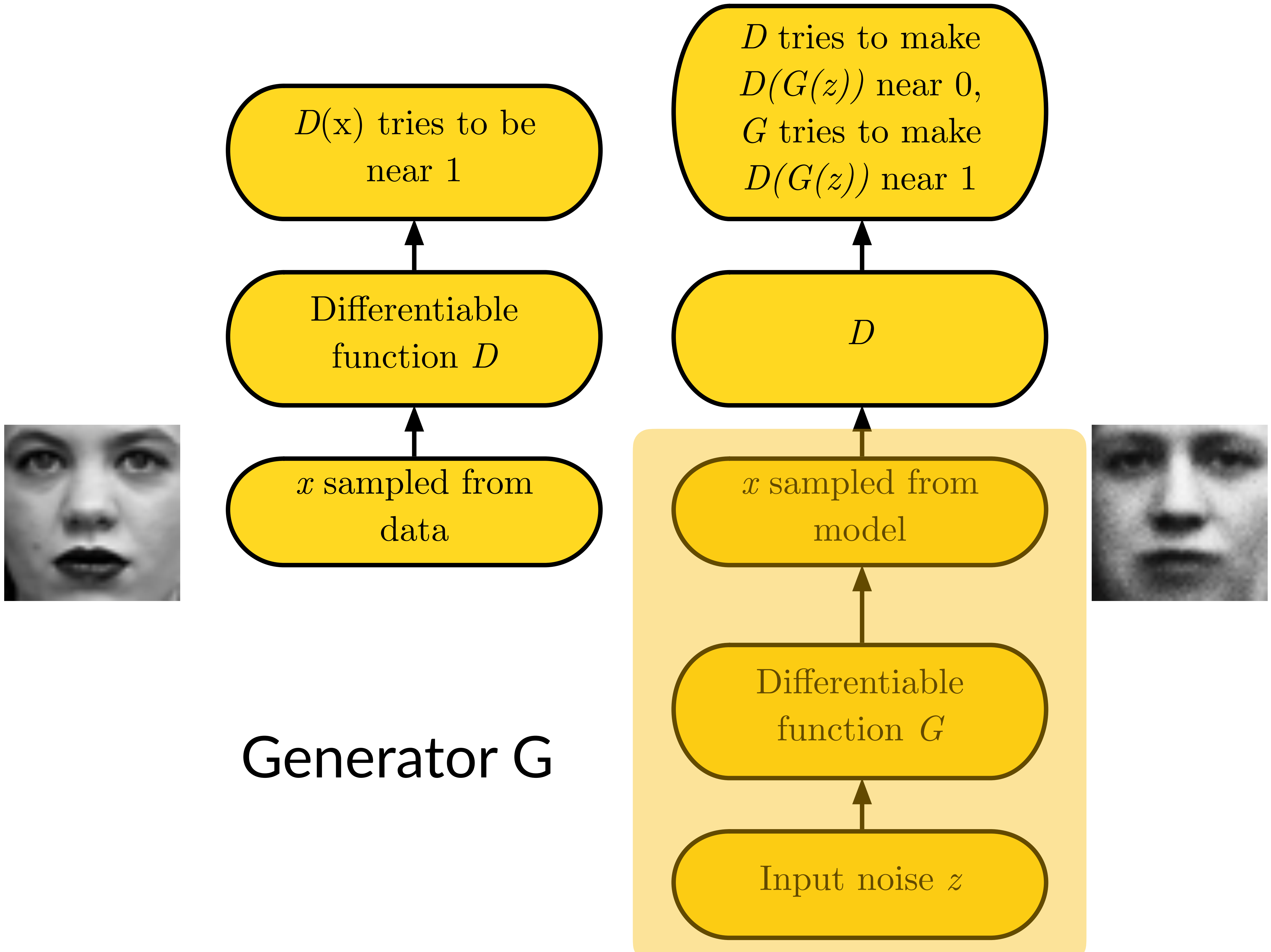


State-action distribution matching objective

- The state-action distribution from the expert trajectories and the state-action distribution that the agent visits by deploying the policy in the environment need to match.

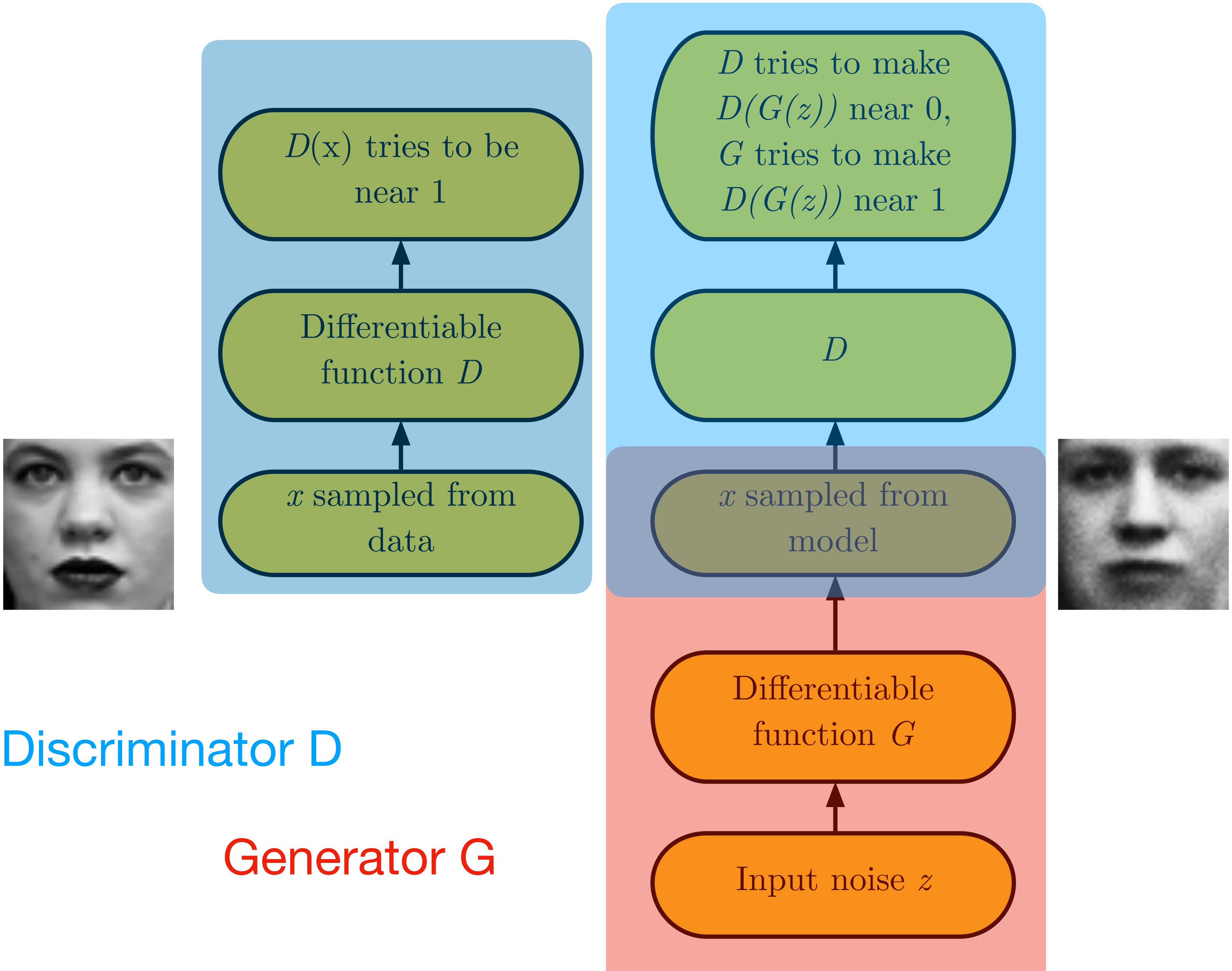
New solution to the compounding error problem of BC!
Let's see how we can optimize this distribution matching objective!

Adversarial Nets Framework



(Goodfellow 2016)

$$\min_{G} \max_{D} V(D,G) = \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))]$$



Discriminator D

Generator G

(Goodfellow 2016)

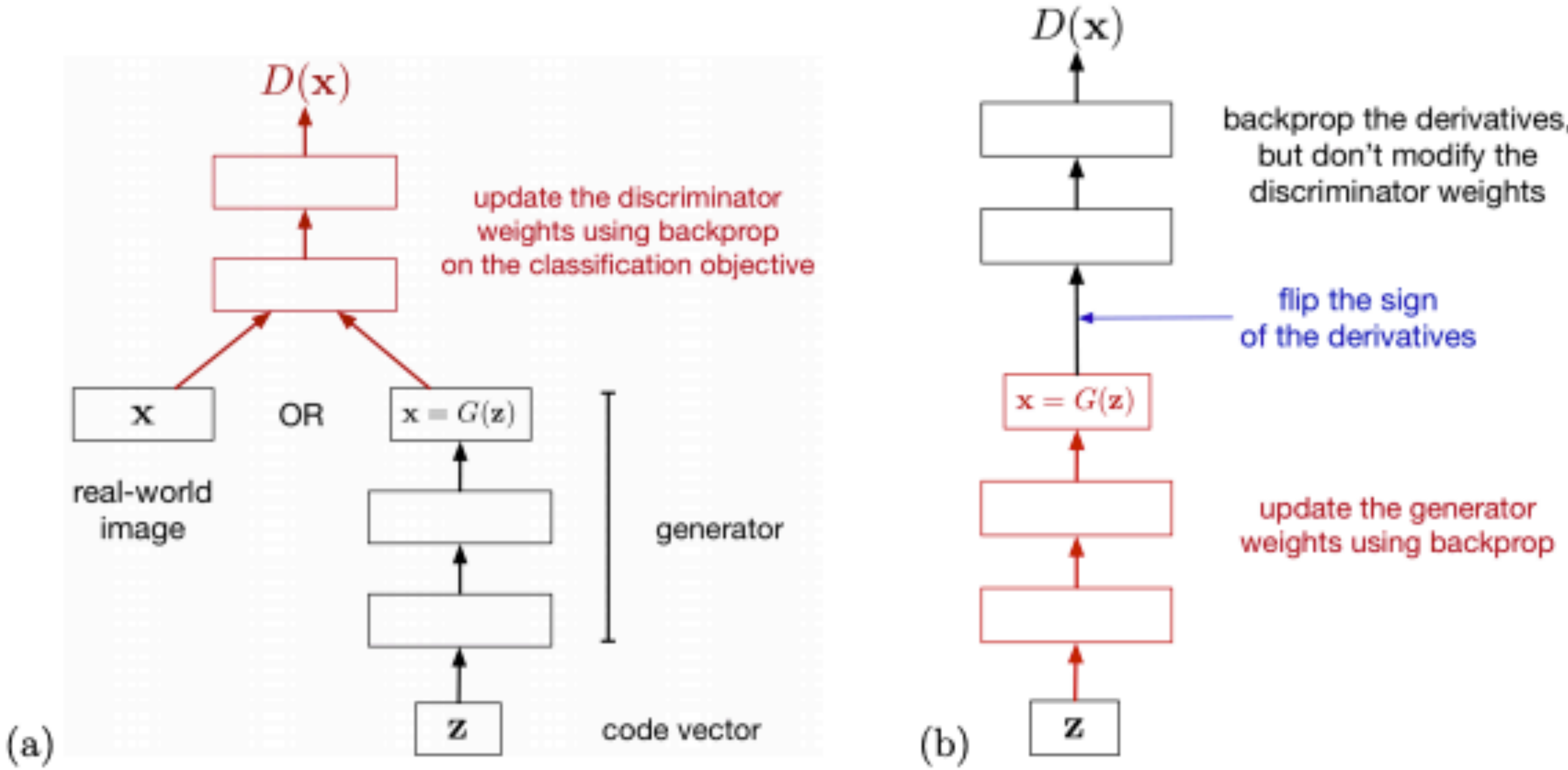
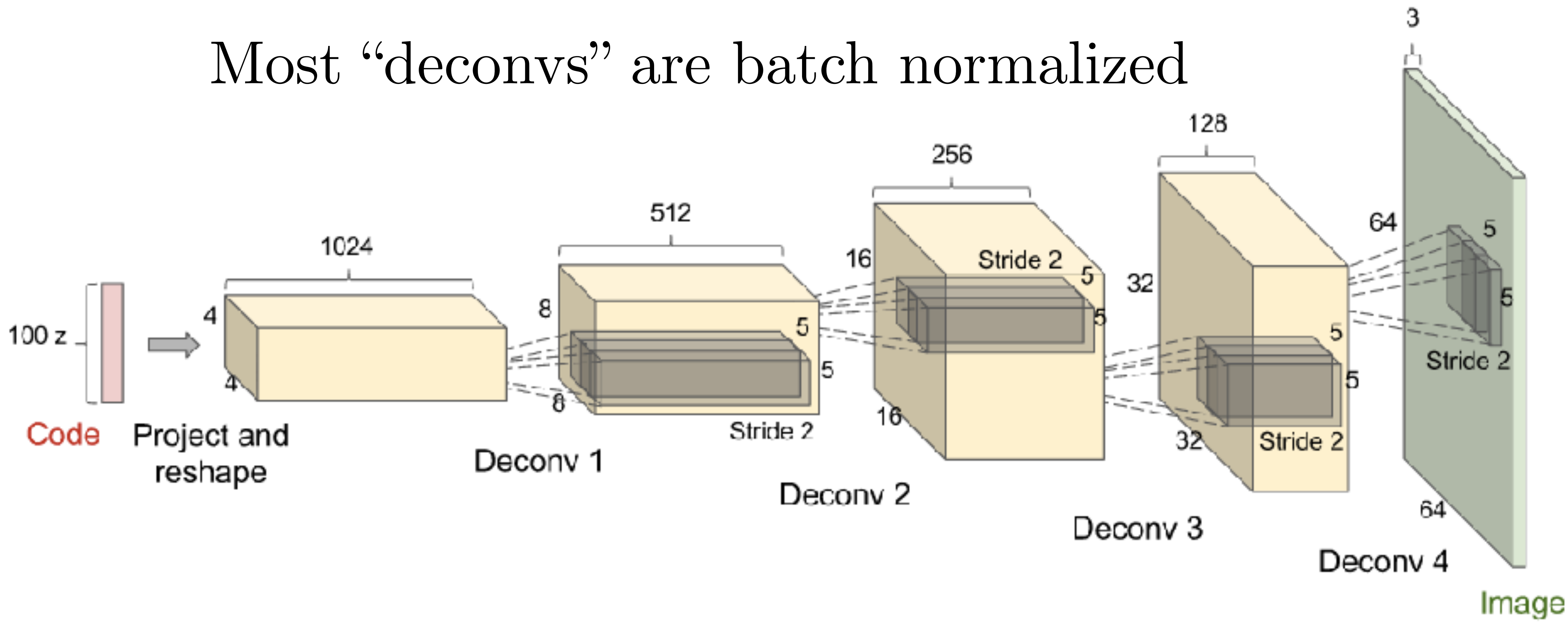


Figure 3: **(a)** Updating the discriminator. **(b)** Updating the generator.

A Generator network (DCGAN)

Most "deconvs" are batch normalized



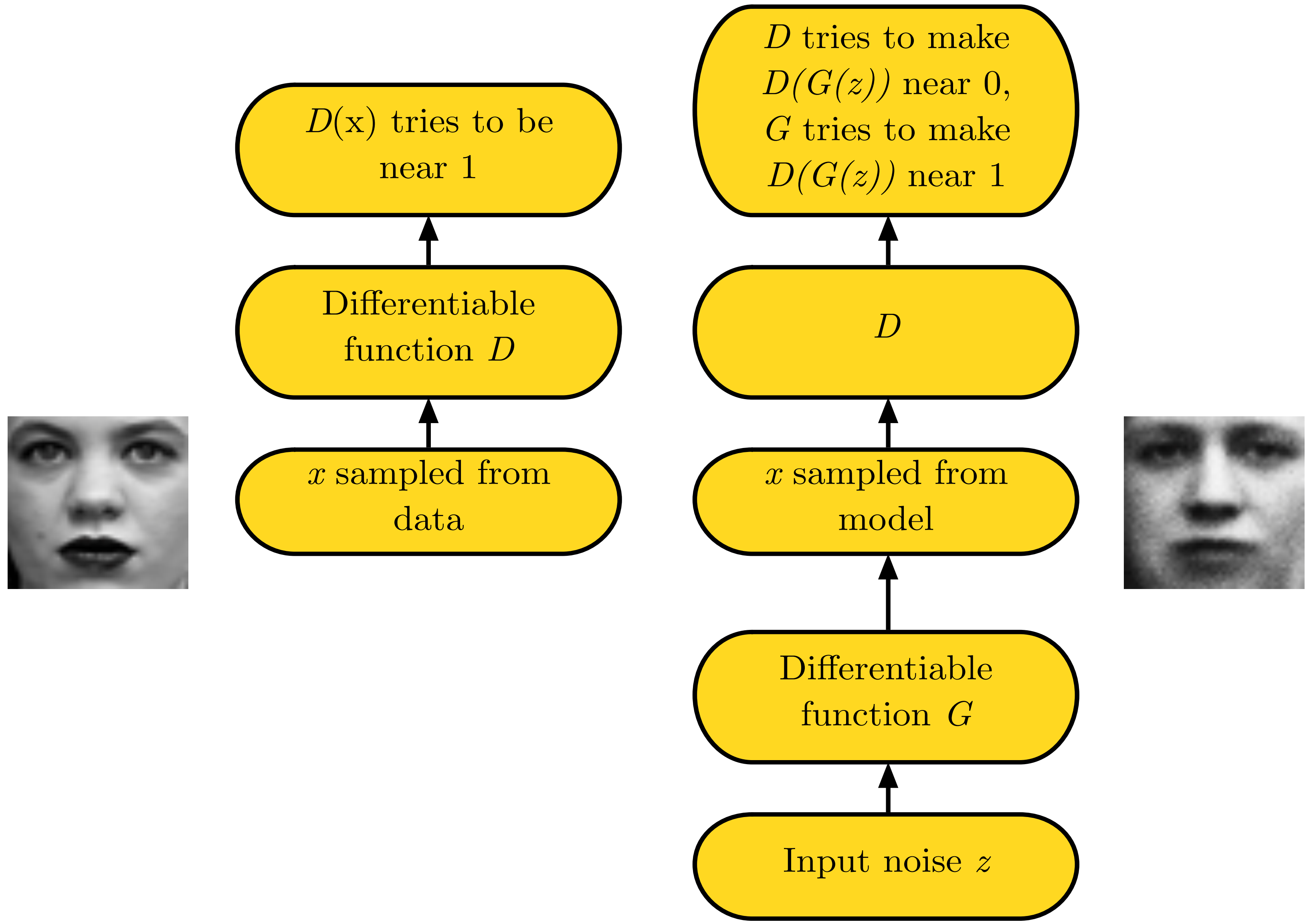
(Radford et al 2015)

(Goodfellow 2016)

Training Procedure

- Use SGD-like algorithm of choice (Adam) on two minibatches simultaneously:
    - A minibatch of training examples
    - A minibatch of generated samples
- Optional: run k steps of one player for every step of the other player.

(Goodfellow 2016)



(Goodfellow 2016)

Questions:

What if the generator maps all noise vectors to a single super photorealistic image?

What if we train the discriminator till convergence (it is just a supervised classifier…) and becomes perfect in distinguishing real from generated images?

A minimax game

$$\min_{G} \max_{D} V(D, G) = \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))]$$

A better cost function

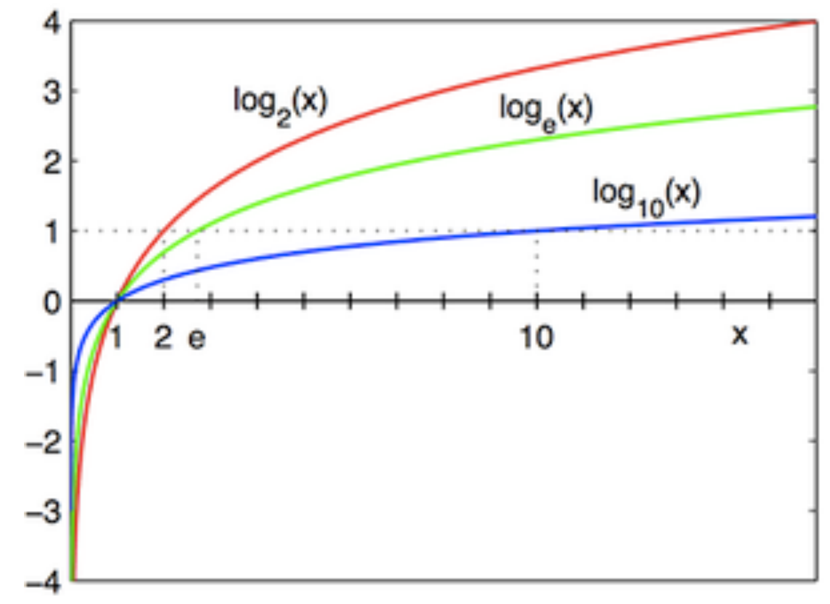
$$\min_{G}\max_{D} V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1-D(G(z)))]$$

$$\min_{G} \quad \mathbb{E}_{z\sim p_z(z)}[\log(1-D(G(z)))]$$

Gradients not informative when D close to 0

$$\Downarrow$$

$$\min_{G} \quad \mathbb{E}_{z\sim p_z(z)}[-\log(D(G(z)))]$$



$$\max_{D} \quad \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1-D(G(z)))]$$

$$\Downarrow$$

$$\min_{D} \quad \mathbb{E}_{x\sim p_{data}(x)}[\log(1-D(x))] + \mathbb{E}_{z\sim p_z(z)}[\log(D(G(z)))]$$

Optimal discriminator strategy

$$\min_{G} \max_{D} V(D, G) = \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))]$$

$$V(D, G) = \int_x p_{\mathrm{data}}(x) \log D(x) dx + \int_z p_z(z) \log(1 - D(G(z))) dz$$

Optimal discriminator strategy

$$\min_{G} \max_{D} V(D, G) = \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))]$$

$$V(D, G) = \int_x p_{\text{data}}(x) \log D(x) dx + \int_z p_z(z) \log(1 - D(G(z))) dz$$

$$\int_x p_{\text{data}}(x) \log D(x) dx + \int_x p_G(x) \log(1 - D(x)) dx$$

Optimal discriminator strategy

$$\min_{G} \max_{D} V(D, G) = \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))]$$

$$\begin{aligned}
V(D, G) &= \int_x p_{\text{data}}(x) \log D(x) dx + \int_z p_z(z) \log(1 - D(G(z))) dz \\
&= \int_x p_{\text{data}}(x) \log D(x) dx + \int_x p_G(x) \log(1 - D(x)) dx \\
&= \int_x p_{\text{data}}(x) \log D(x) + p_G(x) \log(1 - D(x)) dx
\end{aligned}$$

Optimal discriminator strategy

$$V(D,G) = \int_x p_{\text{data}}(x) \log D(x) + p_G(x) \log(1 - D(x))dx$$

The discriminator assigns values D(x) to each image x. Let's take the derivative to see where the optimum is attained.

Optimal discriminator strategy

$$V(D,G) = \int_x p_{\text{data}}(x)\log D(x) + p_G(x)\log(1 - D(x))dx$$

$$\frac{d}{dD(x)}\left(p_{\text{data}}(x)\log D(x) + p_G(x)\log(1 - D(x))\right) = 0$$

Optimal discriminator strategy

$$V(D,G) = \int_x p_{\text{data}}(x)\log D(x) + p_G(x)\log(1-D(x))dx$$

$$\frac{d}{dD(x)}\left(p_{\text{data}}(x)\log D(x) + p_G(x)\log(1-D(x))\right) = 0$$

$$\Leftrightarrow p_{\text{data}}(x)\frac{1}{D(x)} - p_G(x)\frac{1}{1-D(x)} = 0$$

Optimal discriminator strategy

$$V(D,G) = \int_x p_{\text{data}}(x)\log D(x) + p_G(x)\log(1 - D(x))dx$$

$$\frac{d}{dD(x)}\left(p_{\text{data}}(x)\log D(x) + p_G(x)\log(1 - D(x)\right) = 0$$

$$\Leftrightarrow p_{\text{data}}(x)\frac{1}{D(x)} - p_G(x)\frac{1}{1 - D(x)} = 0$$

$$\Leftrightarrow p_{\text{data}}(x)\frac{1}{D(x)} = p_G(x)\frac{1}{1 - D(x)}$$

Optimal discriminator strategy

$$V(D,G) = \int_x p_{\text{data}}(x)\log D(x) + p_G(x)\log(1 - D(x))dx$$

$$\frac{d}{dD(x)}\left(p_{\text{data}}(x)\log D(x) + p_G(x)\log(1 - D(x))\right) = 0$$

$$\Leftrightarrow p_{\text{data}}(x)\frac{1}{D(x)} - p_G(x)\frac{1}{1 - D(x)} = 0$$

$$\Leftrightarrow p_{\text{data}}(x)\frac{1}{D(x)} = p_G(x)\frac{1}{1 - D(x)}$$

$$\Leftrightarrow p_{\text{data}}(x)(1 - D(x)) = p_G(x)D(x)$$

Optimal discriminator strategy

$$V(D,G) = \int_x p_{\text{data}}(x) \log D(x) + p_G(x) \log(1 - D(x)) dx$$

$$\frac{d}{dD(x)} \left(p_{\text{data}}(x) \log D(x) + p_G(x) \log(1 - D(x)) \right) = 0$$

$$\Leftrightarrow p_{\text{data}}(x) \frac{1}{D(x)} - p_G(x) \frac{1}{1 - D(x)} = 0$$

$$\Leftrightarrow p_{\text{data}}(x) \frac{1}{D(x)} = p_G(x) \frac{1}{1 - D(x)}$$

$$\Leftrightarrow p_{\text{data}}(x)(1 - D(x)) = p_G(x) D(x)$$

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

Optimal generator strategy

$$C(G) = \max_D V(G, D)$$

Optimal generator strategy

$$
\begin{aligned}
C(G) &= \max_D V(G, D) \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log D_G^*(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D_G^*(G(z))]
\end{aligned}
$$

Optimal generator strategy

$$
\begin{aligned}
C(G) &= \max_D V(G, D) \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log D_G^*(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D_G^*(G(z))] \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log D_G^*(x)] + \mathbb{E}_{x \sim p_G(x)}[\log(1 - D_G^*(x)]
\end{aligned}
$$

Optimal generator strategy

$$
\begin{aligned}
C(G) &= \max_D V(G, D) \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log D_G^*(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D_G^*(G(z))] \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log D_G^*(x)] + \mathbb{E}_{x \sim p_G(x)}[\log(1 - D_G^*(x)] \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)}] + \mathbb{E}_{x \sim p_G(x)}[\log(1 - \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)})]
\end{aligned}
$$

Optimal generator strategy

$$
\begin{aligned}
C(G) &= \max_D V(G, D) \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log D_G^*(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D_G^*(G(z))] \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log D_G^*(x)] + \mathbb{E}_{x \sim p_G(x)}[\log(1 - D_G^*(x)] \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)}] + \mathbb{E}_{x \sim p_G(x)}[\log(1 - \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)})] \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)}] + \mathbb{E}_{x \sim p_G(x)}[\log(\frac{p_{\text{G}}(x)}{p_{\text{data}}(x) + p_G(x)})]
\end{aligned}
$$

Optimal generator strategy

$$
\begin{aligned}
C(G) &= \max_{D} V(G, D) \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log D_G^*(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D_G^*(G(z))] \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log D_G^*(x)] + \mathbb{E}_{x \sim p_G(x)}[\log(1 - D_G^*(x)] \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)}] + \mathbb{E}_{x \sim p_G(x)}[\log(1 - \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)})] \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)}] + \mathbb{E}_{x \sim p_G(x)}[\log(\frac{p_{\text{G}}(x)}{p_{\text{data}}(x) + p_G(x)})] \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)}] + \mathbb{E}_{x \sim p_G(x)}[\log(\frac{p_{\text{G}}(x)}{p_{\text{data}}(x) + p_G(x)})] - \log 4 + \log 4
\end{aligned}
$$

Optimal generator strategy

$$
\begin{aligned}
C(G) &= \max_D V(G, D) \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log D_G^*(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D_G^*(G(z))] \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log D_G^*(x)] + \mathbb{E}_{x \sim p_G(x)}[\log(1 - D_G^*(x)] \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)}] + \mathbb{E}_{x \sim p_G(x)}[\log(1 - \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)})] \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)}] + \mathbb{E}_{x \sim p_G(x)}[\log(\frac{p_{\text{G}}(x)}{p_{\text{data}}(x) + p_G(x)})] \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)}] + \mathbb{E}_{x \sim p_G(x)}[\log(\frac{p_{\text{G}}(x)}{p_{\text{data}}(x) + p_G(x)})] - \log 4 + \log 4 \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log \frac{2p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)}] + \mathbb{E}_{x \sim p_G(x)}[\log(\frac{2p_{\text{G}}(x)}{p_{\text{data}}(x) + p_G(x)})] - \log 4
\end{aligned}
$$

Optimal generator strategy

$$
\begin{aligned}
C(G) &= \max_D V(G, D) \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log D_G^*(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D_G^*(G(z))] \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log D_G^*(x)] + \mathbb{E}_{x \sim p_G(x)}[\log(1 - D_G^*(x)] \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)}] + \mathbb{E}_{x \sim p_G(x)}[\log(1 - \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)})] \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)}] + \mathbb{E}_{x \sim p_G(x)}[\log(\frac{p_{\text{G}}(x)}{p_{\text{data}}(x) + p_G(x)})] \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)}] + \mathbb{E}_{x \sim p_G(x)}[\log(\frac{p_{\text{G}}(x)}{p_{\text{data}}(x) + p_G(x)})] - \log 4 + \log 4 \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log \frac{2p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)}] + \mathbb{E}_{x \sim p_G(x)}[\log(\frac{2p_{\text{G}}(x)}{p_{\text{data}}(x) + p_G(x)})] - \log 4 \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log \frac{p_{\text{data}}(x)}{\frac{p_{\text{data}}(x) + p_G(x)}{2}}] + \mathbb{E}_{x \sim p_G(x)}[\log \frac{p_{\text{G}}(x)}{\frac{p_{\text{data}}(x) + p_G(x)}{2}}] - \log 4
\end{aligned}
$$

Optimal generator strategy

$$
\begin{aligned}
C(G) &= \max_D V(G,D) \\
&= \mathbb{E}_{x\sim p_{data}(x)}[\log D_G^*(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D_G^*(G(z))] \\
&= \mathbb{E}_{x\sim p_{data}(x)}[\log D_G^*(x)] + \mathbb{E}_{x\sim p_G(x)}[\log(1 - D_G^*(x)] \\
&= \mathbb{E}_{x\sim p_{data}(x)}[\log \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)}] + \mathbb{E}_{x\sim p_G(x)}[\log(1 - \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)})] \\
&= \mathbb{E}_{x\sim p_{data}(x)}[\log \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)}] + \mathbb{E}_{x\sim p_G(x)}[\log(\frac{p_{\text{G}}(x)}{p_{\text{data}}(x) + p_G(x)})] \\
&= \mathbb{E}_{x\sim p_{data}(x)}[\log \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)}] + \mathbb{E}_{x\sim p_G(x)}[\log(\frac{p_{\text{G}}(x)}{p_{\text{data}}(x) + p_G(x)})] - \log 4 + \log 4 \\
&= \mathbb{E}_{x\sim p_{data}(x)}[\log \frac{2p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)}] + \mathbb{E}_{x\sim p_G(x)}[\log(\frac{2p_{\text{G}}(x)}{p_{\text{data}}(x) + p_G(x)})] - \log 4 \\
&= \mathbb{E}_{x\sim p_{data}(x)}[\log \frac{p_{\text{data}}(x)}{\frac{p_{\text{data}}(x) + p_G(x)}{2}}] + \mathbb{E}_{x\sim p_G(x)}[\log \frac{p_{\text{G}}(x)}{\frac{p_{\text{data}}(x) + p_G(x)}{2}}] - \log 4 \\
&= \mathrm{D_{KL}}\left(p_{data}(x)\,||\,\frac{p_{\text{data}}(x) + p_G(x)}{2}\right) + \mathrm{D_{KL}}\left(p_G(x)\|\frac{p_{\text{data}}(x) + p_G(x)}{2}\right) - \log 4
\end{aligned}
$$

Optimal generator strategy

$$
\begin{aligned}
C(G) &= \max_D V(G, D) \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log D_G^*(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D_G^*(G(z))] \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log D_G^*(x)] + \mathbb{E}_{x \sim p_G(x)}[\log(1 - D_G^*(x)] \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)}] + \mathbb{E}_{x \sim p_G(x)}[\log(1 - \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)})] \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)}] + \mathbb{E}_{x \sim p_G(x)}[\log(\frac{p_{\text{G}}(x)}{p_{\text{data}}(x) + p_G(x)})] \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)}] + \mathbb{E}_{x \sim p_G(x)}[\log(\frac{p_{\text{G}}(x)}{p_{\text{data}}(x) + p_G(x)})] - \log 4 + \log 4 \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log \frac{2p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)}] + \mathbb{E}_{x \sim p_G(x)}[\log(\frac{2p_{\text{G}}(x)}{p_{\text{data}}(x) + p_G(x)})] - \log 4 \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log \frac{p_{\text{data}}(x)}{\frac{p_{\text{data}}(x) + p_G(x)}{2}}] + \mathbb{E}_{x \sim p_G(x)}[\log \frac{p_{\text{G}}(x)}{\frac{p_{\text{data}}(x) + p_G(x)}{2}}] - \log 4 \\
&= \mathrm{D_{KL}}\left(p_{data}(x) \,||\, \frac{p_{\text{data}}(x) + p_G(x)}{2}\right) + \mathrm{D_{KL}}\left(p_G(x) \| \frac{p_{\text{data}}(x) + p_G(x)}{2}\right) - \log 4 \\
&= 2\mathrm{D_{JSD}}\left(p_{data}(x) \,||\, p_G(x)\right) - \log 4
\end{aligned}
$$

Optimal generator strategy

$$
\begin{aligned}
C(G) &= \max_D V(G, D) \\
&= \mathbb{E}_{x \sim p_{data}(x)}[\log \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)}] + \mathbb{E}_{x \sim p_G(x)}[\log(\frac{p_{\text{G}}(x)}{p_{\text{data}}(x) + p_G(x)})] \\
&= 2\mathrm{D}_{\text{JSD}}\left(p_{data}(x) \,||\, p_G(x)\right) - \log 4
\end{aligned}
$$

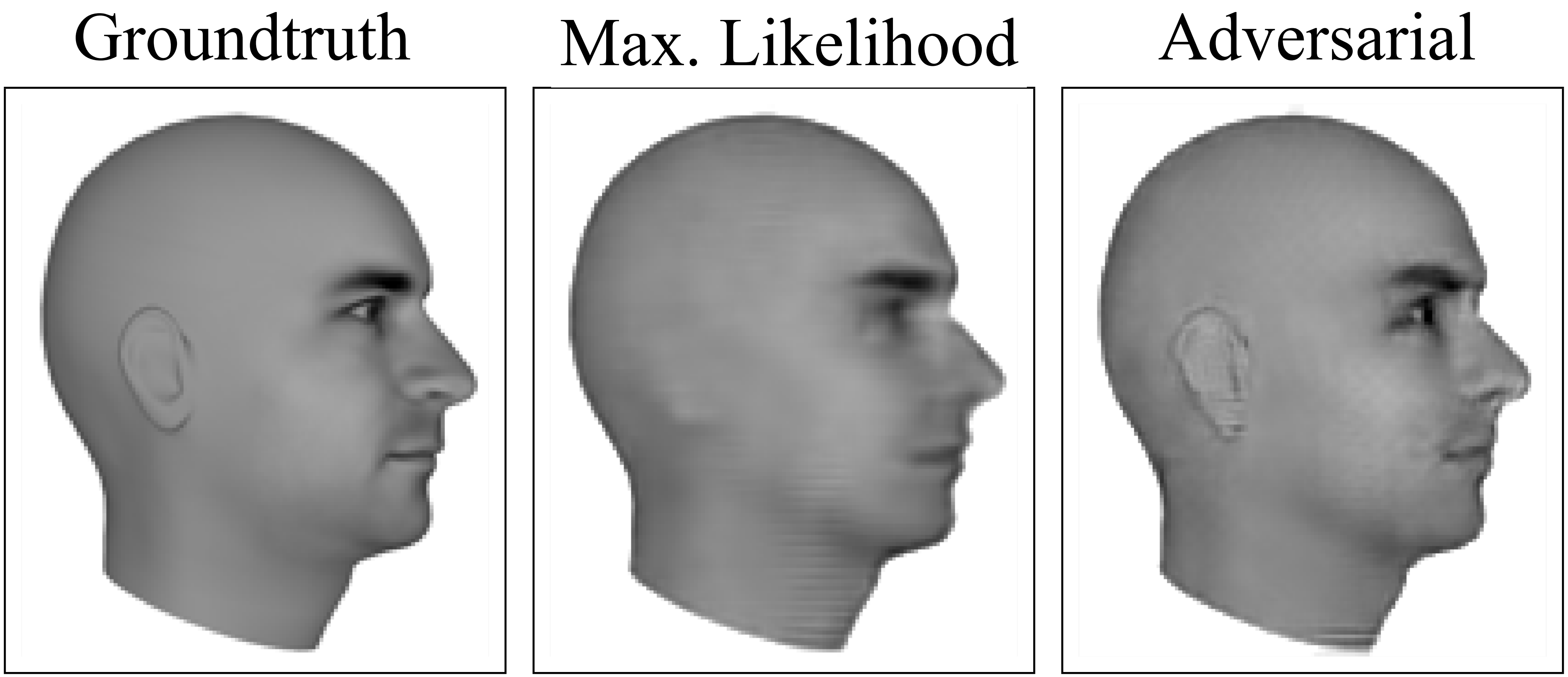
Since $\mathrm{D}_{\text{JSD}} \geq 0, \quad C(G) \geq -\log 4$

By setting $P_G(x) = \mathrm{p}_{\text{data}}(\mathrm{x})$ in the equation above, we get:

$$
C(G) = \mathbb{E}_{x \sim \mathrm{p}_{\text{data}}(x)} \log \frac{1}{2} + \mathbb{E}_{x \sim p_G(x)} \log \frac{1}{2} = -\log 4
$$

Thus generator achieves the optimum when $P_G(x) = \mathrm{p}_{\text{data}}(\mathrm{x})$.

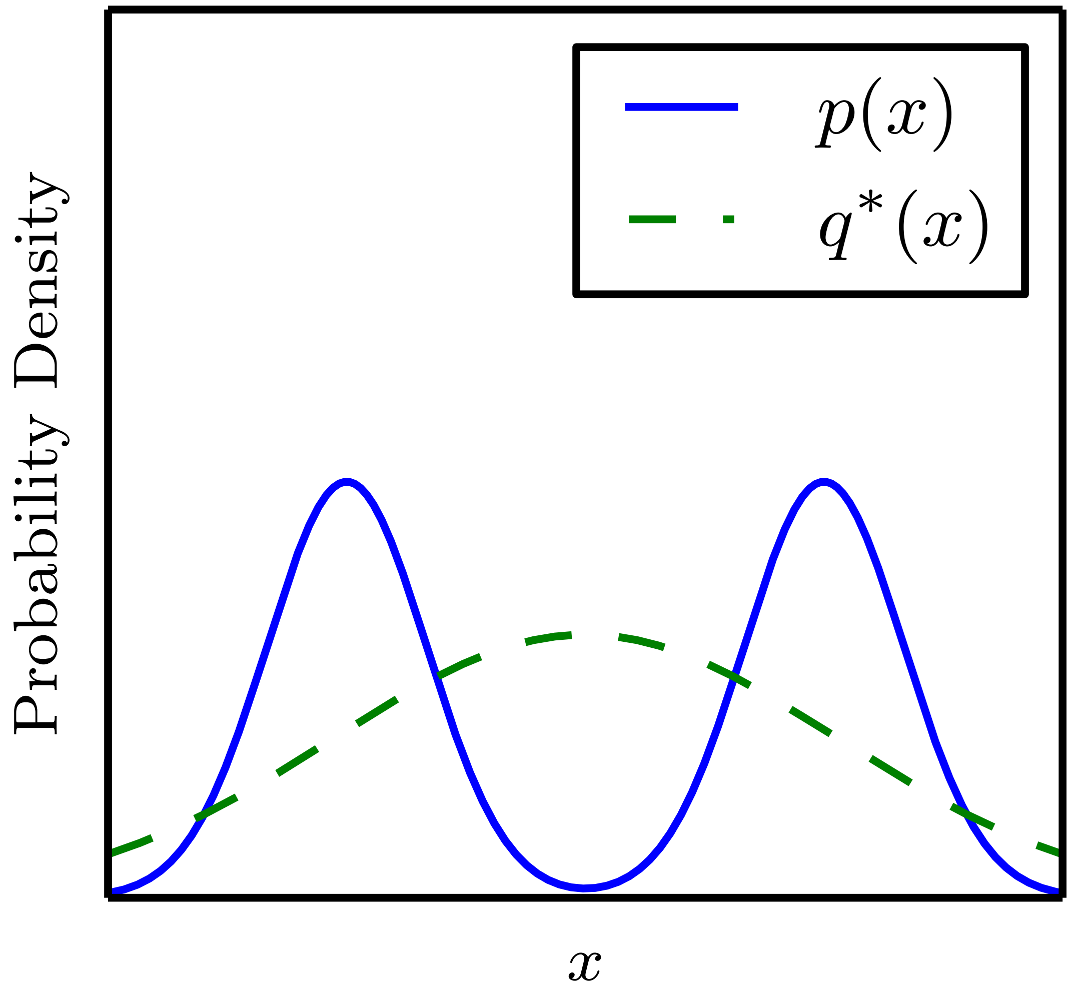
Next Video Frame Prediction



(Lotter et al 2016)

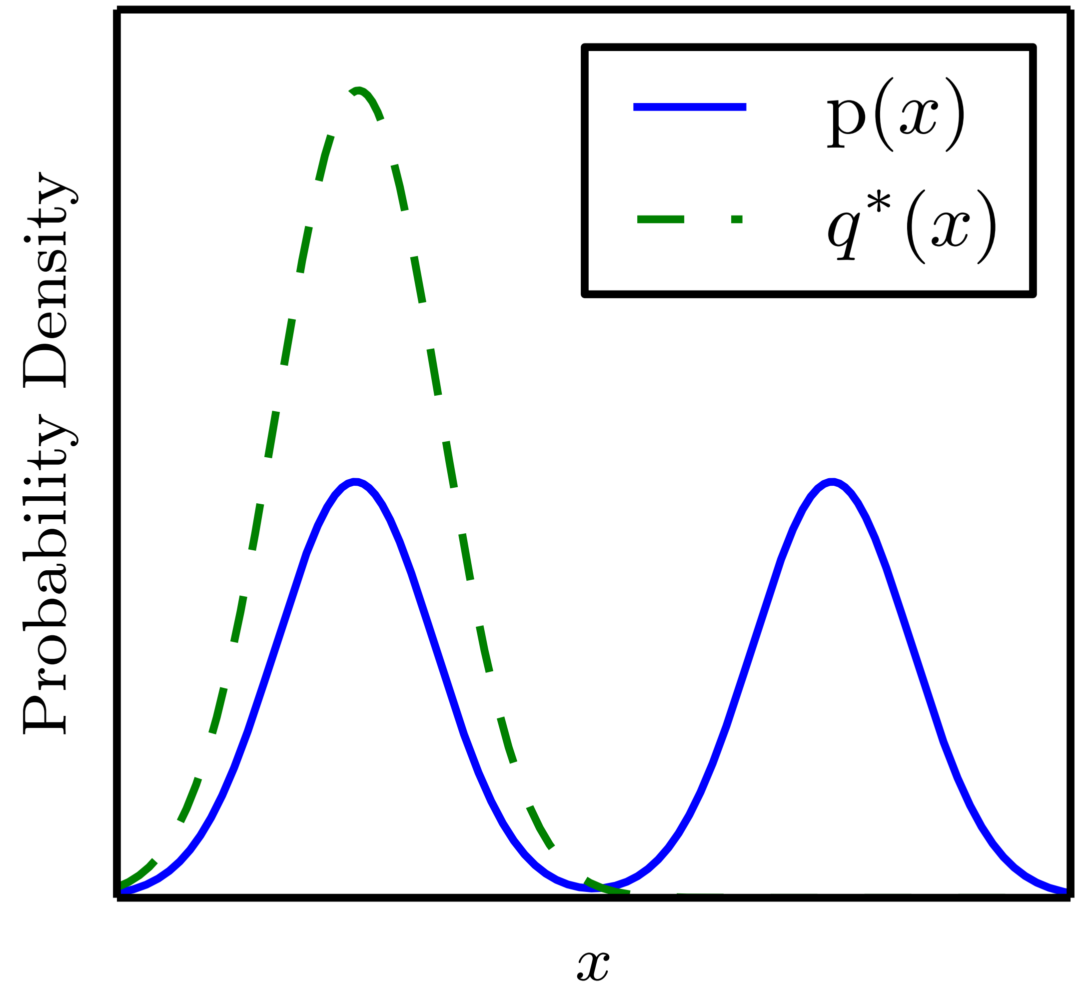
Maybe an explanation of why GANs work

$q^* = \text{argmin}_q D_{\text{KL}}(p\|q)$



Maximum likelihood

$q^* = \text{argmin}_q D_{\text{KL}}(q\|p)$



Reverse KL

Generative Adversarial Imitation learning

The policy network will be our generator, that conditions on the state:

$$\pi_\theta(s) \to a$$

Generative Adversarial Imitation learning

Find a policy π_θ that makes it impossible for a discriminator network to distinguish between state-actions from the expert demonstrations and state-action pairs visited by the agent's policy π_θ:

$$\min_{\pi_\theta} \quad \mathbb{E}_{(s,a)\sim\pi_\theta}[-\log(D_\phi(s,a))]$$

$$\min_{D_\phi} \quad \mathbb{E}_{(s,a)\sim\mathrm{Demo}}[\log(1-D_\phi(s,a))] + \mathbb{E}_{(s,a)\sim\pi_\theta}[\log(D_\phi(s,a))]$$

The reward for the policy optimization is how well I matched the demonstrator's trajectory distribution, else, how well I confused the discriminator.

$$r(s,a) = \log D_\phi(s,a), (s,a) \sim \pi_\theta$$

Generative Adversarial Imitation learning

Input: Expert trajectories , initial policy parameters θ_0 and initial discriminator weights ϕ_0.

For i=0,1,2,3… **do**

1. Sample agent trajectories $\tau_i \sim \pi_{\theta_i}$
2. Update the discriminator parameters with the gradient:

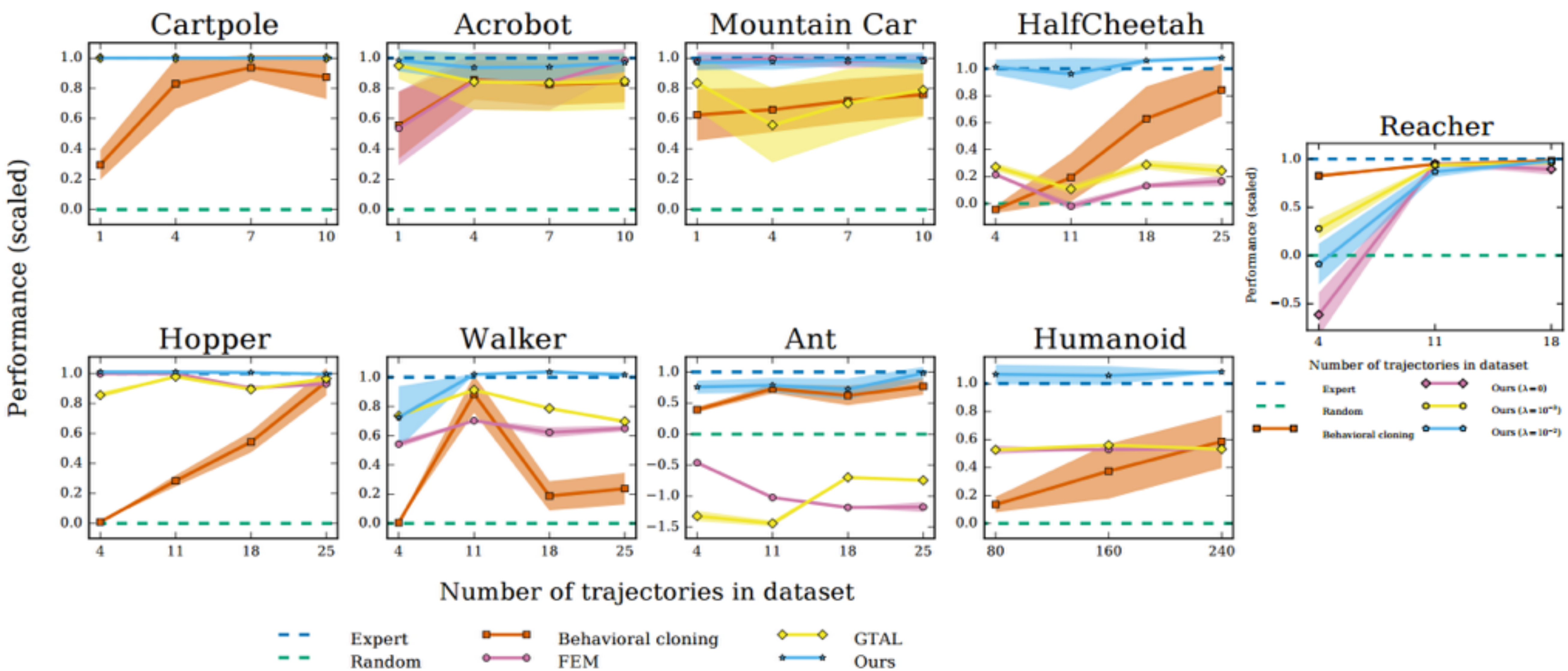
$$\mathbb{E}_{(s,a)\sim \mathrm{Demo}}[\nabla_\phi \log(1 - D_\phi(s,a))] + \mathbb{E}_{(s,a)\in\tau_i}[\nabla_\phi \log(D_\phi(s,a))]$$

3. Update the policy using a policy gradient computed with the rewards, e.g., the REINFORCE policy gradient would be:

$$\mathbb{E}_{(s,a)\in\tau_i}[\nabla_\theta \log \pi_\theta \log D_{\phi_{i+1}}(s,a)]$$

end for

Generative Adversarial Imitation learning



- GAIL: a reinforcement learning method with a reward based on trajectory distribution matching between the agent and an expert.
- BC: reduces imitation learning to supervised learning for individual actions.
- GAIL performs better than behaviour cloning but it requires MORE interactions with the environment.
- Q:Can BC or GAIL outperform the expert?

Imitation learning for diverse goals

- Pushing to diverse locations
- Pouring to different bottles
- Driving to different destinations

We need a way to communicate the goal during learning of the policy

Generalized policies

- Often times we care about policies that achieve many related goals
- For example push object A to (10,10,10) and to (10,12,10)
- The two policies should have many things in common
- Training such policies jointly may be beneficial

$$\pi(s;\theta) \quad \Rightarrow \quad \pi(s, g; \theta)$$

$$s, g \in \mathcal{S}$$

Universal value function Approximators

$$V(s;\theta) \rightarrow V(s,g;\theta)$$

$$\pi(s;\theta) \rightarrow \pi(s,g;\theta)$$

- All methods we have learnt so far can be used.
- At the beginning of an episode, **we sample not only a start state but also a goal g**, which stays constant throughout the episode
- The experience tuples should contain the goal.

$$(s,a,r,s') \rightarrow \left(s,g,a,r,s'\right)$$

Universal Value Function Approximators, Schaul et al.

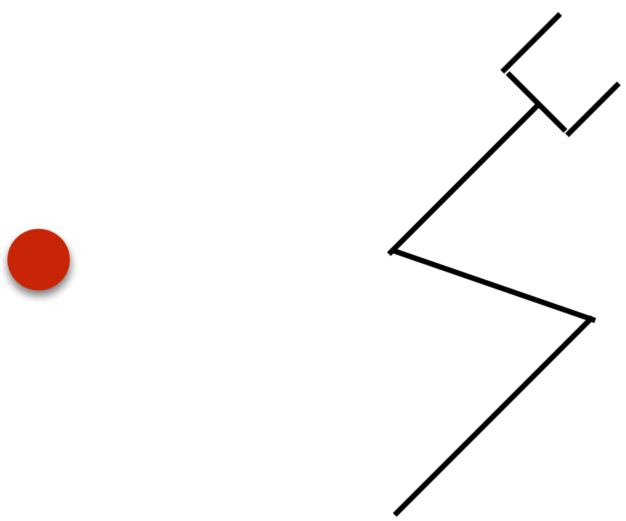
Goal conditioned behavior cloning

- Assumes access to a set of trajectories $\mathcal{T} = \{o_1^j, a_1^j, o_2^j, a_2^j, o_3^j, a_3^j, \ldots, o_T^j, a_T^j, g^j, j = 1...T\}$. Trains a policy by minimizing a standard supervised learning objective:

$$\mathcal{L}_{BC}(\theta, \mathcal{T}) = \mathbb{E}_{(s_t^j, a_t^j, g^j) \sim \mathcal{T}} \left[\| a_t^j - \pi_\theta(s_t^j, g^j) \|_2^2 \right]$$

Goal relabelling!

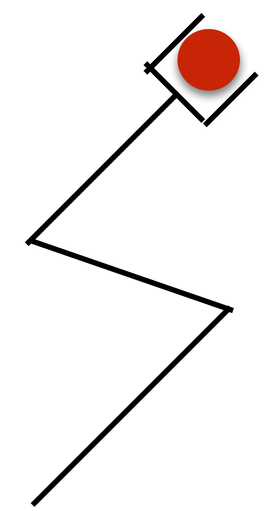
Initial idea: use failed executions under one goal g, as successful executions under an alternative goal g'.



No reward :-(

Goal g

Our reacher at the end of the episode $(s, g, a, 0, s')$



Goal g'

reward :-)

Our reacher at the end of the episode $(s, g', a, 1, s')$

We will use goal relabelling also for expert demonstrations

Hindsight Experience Replay

Marcin Andrychowicz*, **Filip Wolski**, **Alex Ray**, **Jonas Schneider**, **Rachel Fong**,
Peter Welinder, **Bob McGrew**, **Josh Tobin**, **Pieter Abbeel**†, **Wojciech Zaremba**†
OpenAI

Main idea: use failed executions under one goal g, as successful executions under an alternative goal g' (which is where we ended at the end of the episode).

RL with goal relabelling

Algorithm 1 Hindsight Experience Replay (HER)

Given:

- an off-policy RL algorithm $\mathbb{A}$, ▷ e.g. DQN, DDPG, NAF, SDQN
- a strategy $\mathbb{S}$ for sampling goals for replay, ▷ e.g. $\mathbb{S}(s_0, \ldots, s_T) = m(s_T)$
- a reward function $r : \mathcal{S} \times \mathcal{A} \times \mathcal{G} \to \mathbb{R}$. ▷ e.g. $r(s, a, g) = -[f_g(s) = 0]$

Initialize $\mathbb{A}$ ▷ e.g. initialize neural networks
Initialize replay buffer R
for episode $= 1, M$ **do**
  Sample a goal g and an initial state s_0.
  for $t = 0, T-1$ **do**
    Sample an action a_t using the behavioral policy from $\mathbb{A}$:
      $a_t \leftarrow \pi_b(s_t||g)$ ▷ || denotes concatenation
    Execute the action a_t and observe a new state s_{t+1}
  end for
  for $t = 0, T-1$ **do**
    $r_t := r(s_t, a_t, g)$
    Store the transition $(s_t||g, a_t, r_t, s_{t+1}||g)$ in R ▷ standard experience replay
    Sample a set of additional goals for replay $G := \mathbb{S}($**current episode**$)$
    for $g' \in G$ **do**
      $r' := r(s_t, a_t, g')$
      Store the transition $(s_t||g', a_t, r', s_{t+1}||g')$ in R ▷ HER
    end for
  end for
  for $t = 1, N$ **do**
    Sample a minibatch B from the replay buffer R
    Perform one step of optimization using $\mathbb{A}$ and minibatch B
  end for
end for

The reward here is $||s_t - g||$

G : the states of the current episode

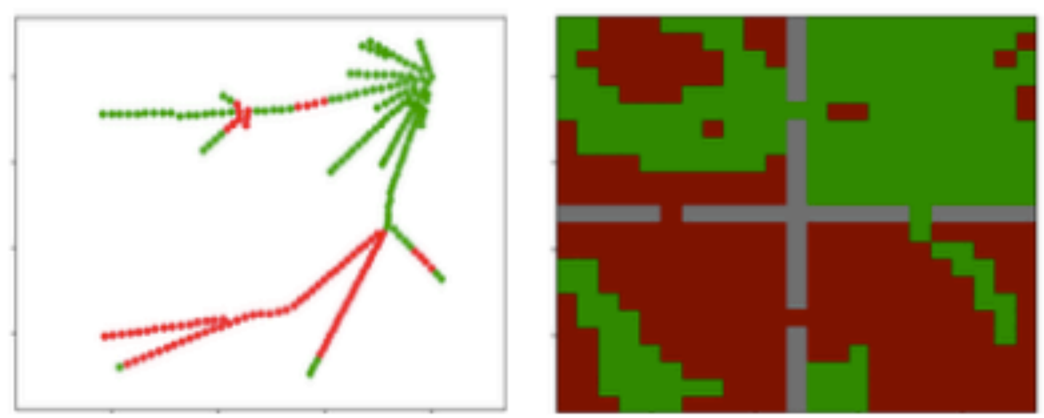
Usually as additional goal we pick the goal that this episode achieved, and the reward becomes non zero

Relabelling expert demonstations

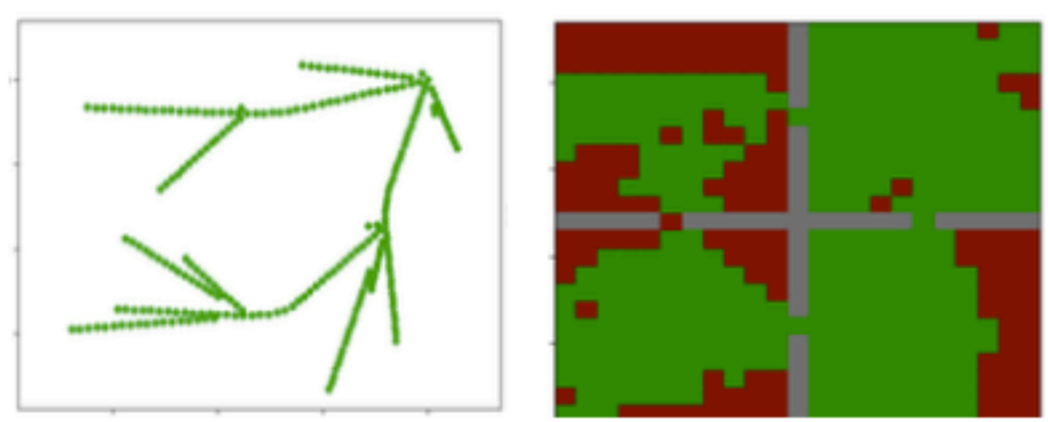
If $(s_t^j, a_t^j, s_{t+1}^j, g^j)$ is in a demonstration, we also add $(s_t^j, a_t^j, s_{t+1}^j, g' = s_{t+k}^j)$

Green mans the policy visited these goals

BC



BC with goal relabelling



Goal-conditioned GAIL with goal relabelling

Goal-conditioned Imitation Learning

Yiming Ding*
Department of Computer Science
University of California, Berkeley
dingyiming0427@berkeley.edu

Carlos Florensa*
Department of Computer Science
University of California, Berkeley
florensa@berkeley.edu

Mariano Phielipp
Intel AI Labs
mariano.j.phielipp@intel.com

Pieter Abbeel
Department of Computer Science
University of California, Berkeley
pabbeel@berkeley.edu

Goal GAIL

Input: Expert trajectories , initial policy parameters θ_0 and initial discriminator weights ϕ_0.

For i=0,1,2,3… **do**

1. Sample agent trajectories $\tau_i \sim \pi_{\theta_i}$
2. Update the discriminator parameters with the gradient:

$$\mathbb{E}_{(s,a,g)\sim \mathrm{Demo}}[\nabla_\phi \log(1-D_\phi(s,a,g))] + \mathbb{E}_{(s,a,g)\in\tau_i}[\nabla_\phi \log(D_\phi(s,a,g))]$$

3. Update the policy using a policy gradient computed with the rewards, e.g., the REINFORCE policy gradient would be:

$$\mathbb{E}_{(s,a,g)\in\tau_i}[\nabla_\theta \log \pi_\theta \log D_{\phi_{i+1}}(s,a,g)]$$

end for

Algorithm 1 Goal-conditioned GAIL with Hindsight: *goalGAIL*

Input: Demonstrations $\mathcal{D} = \left\{(s_0^j, a_0^j, s_1^j, ..., g^j)\right\}_{j=0}^{D}$, replay buffer $\mathcal{R} = \{\}$, policy $\pi_\theta(s, g)$, discount γ, hindsight probability p

while not done **do**

  # Sample rollout

  $g \sim \texttt{Uniform}(\mathcal{S})$

  $\mathcal{R} \leftarrow \mathcal{R} \cup (s_0, a_0, s_1, ...)$ sampled using $\pi(\cdot, g)$

  # Sample from expert buffer and replay buffer

  $\left\{(s_t^j, a_t^j, s_{t+1}^j, g^j)\right\} \sim \mathcal{D}$, $\left\{(s_t^i, a_t^i, s_{t+1}^i, g^i)\right\} \sim \mathcal{R}$

  # Relabel agent transitions

  for each i, with probability p **do**

    $g^i \leftarrow s_{t+k}^i, \quad k \sim \text{Unif}\{t+1, \ldots, T^i\}$ ▷ Use *future* HER strategy

  end for

  # Relabel expert transitions

  $g^j \leftarrow s_{t+k'}^j, \quad k' \sim \text{Unif}\{t+1, \ldots, T^j\}$

  $r_t^h = \mathbb{1}\left[s_{t+1}^h == g^h\right]$

  $\psi \leftarrow \min_\psi \mathcal{L}_{GAIL}(D_\psi, \mathcal{D}, \mathcal{R})$ (Eq. 3)

  $r_t^h = (1 - \delta_{GAIL}) r_t^h + \delta_{GAIL} \log D_\psi(a_t^h, s_t^h, g^h)$ ▷ Add annealed GAIL reward

  # Fit Q_ϕ

  $y_t^h = r_t^h + \gamma Q_\phi(\pi(s_{t+1}^h, g^h), s_{t+1}^h, g^h)$ ▷ Use target networks $Q_{\phi'}$ for stability

  $\phi \leftarrow \min_\phi \sum_h \|Q_\phi(a_t^h, s_t^h, g^h) - y_t^h\|$

  # Update Policy

  $\theta += \alpha \nabla_\theta \hat{J}$ (Eq. 2)

  Anneal δ_{GAIL} ▷ Ensures outperforming the expert

end while

Goal GAIL without actions

Input: Expert trajectories , initial policy parameters θ_0 and initial discriminator weights ϕ_0.

For i=0,1,2,3… **do**

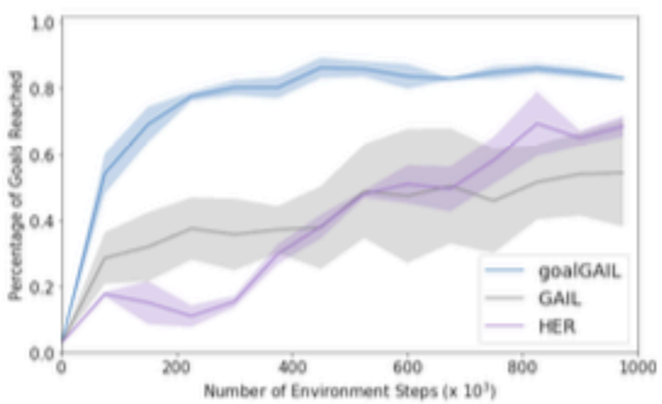
1. Sample agent trajectories $\tau_i \sim \pi_{\theta_i}$
2. Update the discriminator parameters with the gradient:

$$\mathbb{E}_{(s,s',g)\sim \text{Demo}}[\nabla_\phi \log(1 - D_\phi(s, s', g))] + \mathbb{E}_{(s,s',g)\in\tau_i}[\nabla_\phi \log(D_\phi(s, s', g))]$$

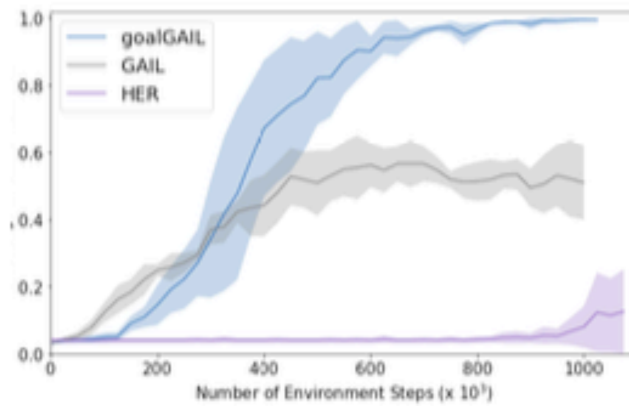
3. Update the policy using a policy gradient computed with the rewards, e.g., the REINFORCE policy gradient would be:

$$\mathbb{E}_{(s,s',g)\in\tau_i}[\nabla_\theta \log \pi_\theta \log D_{\phi_{i+1}}(s, s', g)]$$

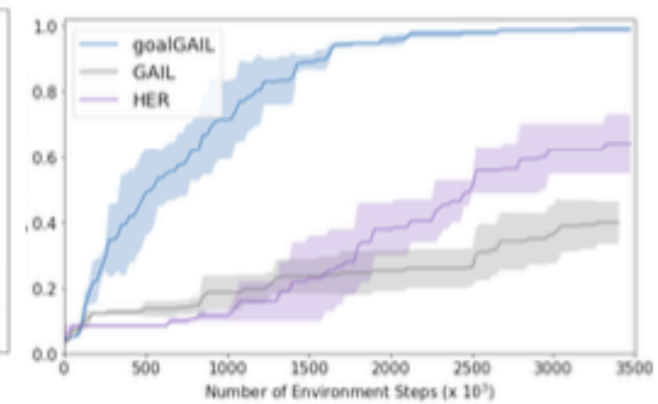
end for



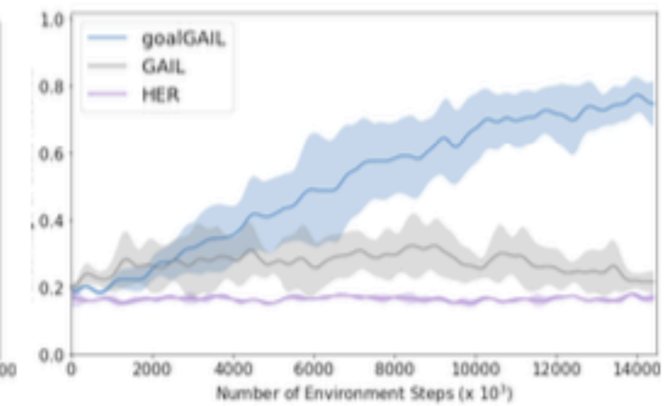
(a) Continuous Four rooms



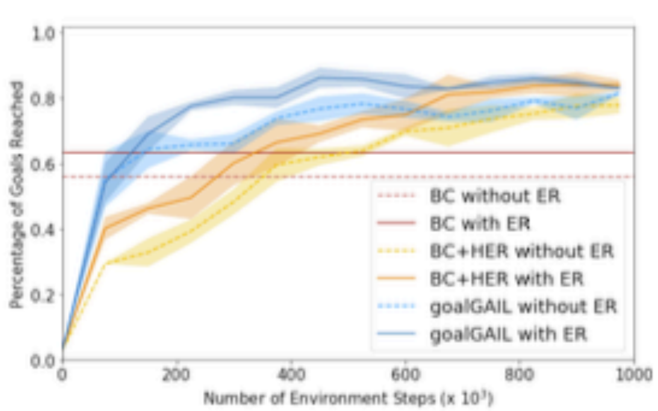
(b) Pointmass block pusher



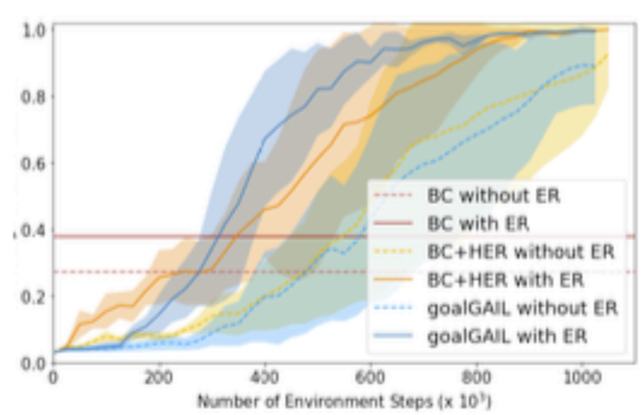
(c) Fetch Pick & Place



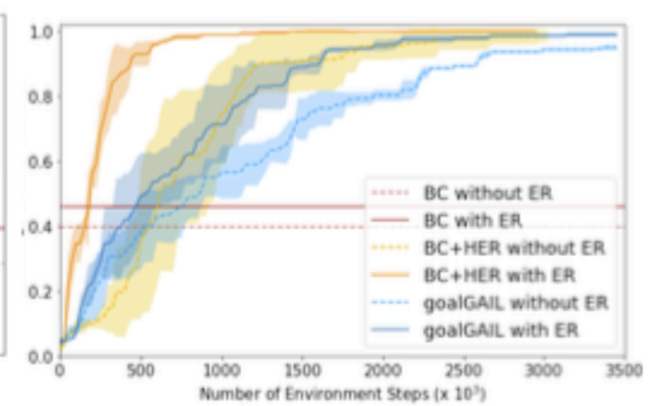
(d) Fetch Stack Two



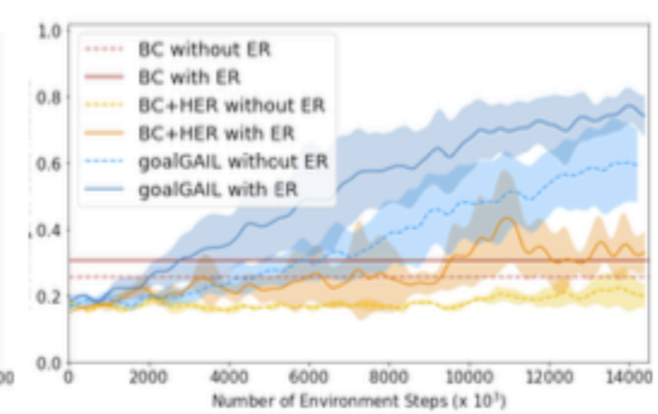
(a) Continuous Four rooms



(b) Pointmass block pusher



(c) Fetch Pick & Place



(d) Fetch Stack Two

https://sites.google.com/view/goalconditioned-il/