ECEN 743: Reinforcement Learning

Inverse Reinforcement Learning

Dileep Kalathil
Assistant Professor
Department of Electrical and Computer Engineering
Texas A&M University

References

- "Imitation Learning Tutorial",Yisong Yue, Hoang M. Le, ICML 2018. [Link]
- [AJKS, Chapter 15]

RL and Reward

- RL learns the optimal policy guided by the reward feedback

RL and Reward

- RL learns the optimal policy guided by the reward feedback
- Where does the reward come from?

RL and Reward

- RL learns the optimal policy guided by the reward feedback
- Where does the reward come from?
- In games like Atari, reward is naturally defined

RL and Reward

- RL learns the optimal policy guided by the reward feedback
- Where does the reward come from?
- In games like Atari, reward is naturally defined
  - Objective is to maximize the (cumulative) score

RL and Reward

- RL learns the optimal policy guided by the reward feedback
- Where does the reward come from?
- In games like Atari, reward is naturally defined
  - Objective is to maximize the (cumulative) score
- However, in many real world scenarios, reward function is not obvious and not specified as a part of the problem

RL and Reward

- RL learns the optimal policy guided by the reward feedback
- Where does the reward come from?
- In games like Atari, reward is naturally defined
  - Objective is to maximize the (cumulative) score
- However, in many real world scenarios, reward function is not obvious and not specified as a part of the problem
  - Autonomous driving, robotic control

RL and Reward

- RL learns the optimal policy guided by the reward feedback
- Where does the reward come from?
- In games like Atari, reward is naturally defined
  - Objective is to maximize the (cumulative) score
- However, in many real world scenarios, reward function is not obvious and not specified as a part of the problem
  - Autonomous driving, robotic control
- Rewards may be sparse and not well defined

RL and Reward

- RL learns the optimal policy guided by the reward feedback
- Where does the reward come from?
- In games like Atari, reward is naturally defined
  - Objective is to maximize the (cumulative) score
- However, in many real world scenarios, reward function is not obvious and not specified as a part of the problem
  - Autonomous driving, robotic control
- Rewards may be sparse and not well defined
- Domain experts/engineers can hand design/tune rewards . Unfortunately, hand designed/tuned reward function may lead to counterintuitive behaviors [Link]

RL and Reward

- RL learns the optimal policy guided by the reward feedback
- Where does the reward come from?
- In games like Atari, reward is naturally defined
  - Objective is to maximize the (cumulative) score
- However, in many real world scenarios, reward function is not obvious and not specified as a part of the problem
  - Autonomous driving, robotic control
- Rewards may be sparse and not well defined
- Domain experts/engineers can hand design/tune rewards . Unfortunately, hand designed/tuned reward function may lead to counterintuitive behaviors [Link]
- Designing rewards which can induce the desired behavior of the system is incredibly challenging

RL without Reward

- It is often easier to provide some human expert demonstration

[1] [Finn et. al. 2016], "Guided cost learning: Deep inverse optimal control via policy optimization", *ICML* 2016.
[2] [Bojarski et. al. 2016], "End to end learning for self-driving cars", arXiv:1604.07316, 2016.

RL without Reward

- It is often easier to provide some human expert demonstration
- Example videos:

[1] [Finn et. al. 2016], "Guided cost learning: Deep inverse optimal control via policy optimization", *ICML* 2016.
[2] [Bojarski et. al. 2016], "End to end learning for self-driving cars", arXiv:1604.07316, 2016.

RL without Reward

- It is often easier to provide some human expert demonstration
- Example videos:
  - Demonstration based learning in robotics[1] [Link]

[1] [Finn et. al. 2016], "Guided cost learning: Deep inverse optimal control via policy optimization", *ICML* 2016.
[2] [Bojarski et. al. 2016], "End to end learning for self-driving cars", arXiv:1604.07316, 2016.

RL without Reward

- It is often easier to provide some human expert demonstration
- Example videos:
  - Demonstration based learning in robotics[1] [Link]
  - Demonstration based learning in self-driving cars[2] [Link]

[1] [Finn et. al. 2016], "Guided cost learning: Deep inverse optimal control via policy optimization", *ICML* 2016.
[2] [Bojarski et. al. 2016], "End to end learning for self-driving cars", arXiv:1604.07316, 2016.

RL without Reward

- It is often easier to provide some human expert demonstration
- Example videos:
  - Demonstration based learning in robotics[1] [Link]
  - Demonstration based learning in self-driving cars[2] [Link]
- How do we learn the optimal control policy from demonstration data?

[1] [Finn et. al. 2016], "Guided cost learning: Deep inverse optimal control via policy optimization", *ICML* 2016.
[2] [Bojarski et. al. 2016], "End to end learning for self-driving cars", arXiv:1604.07316, 2016.

RL without Reward

- It is often easier to provide some human expert demonstration
- Example videos:
  - Demonstration based learning in robotics[1] [Link]
  - Demonstration based learning in self-driving cars[2] [Link]
- How do we learn the optimal control policy from demonstration data?
- Inverse RL, Imitation Learning, Offline RL

[1] [Finn et. al. 2016], "Guided cost learning: Deep inverse optimal control via policy optimization", *ICML* 2016.
[2] [Bojarski et. al. 2016], "End to end learning for self-driving cars", arXiv:1604.07316, 2016.

RL without Reward

- It is often easier to provide some human expert demonstration
- Example videos:
  - Demonstration based learning in robotics[1] [Link]
  - Demonstration based learning in self-driving cars[2] [Link]
- How do we learn the optimal control policy from demonstration data?
- Inverse RL, Imitation Learning, Offline RL
- This lecture: Inverse Reinforcement Learning (IRL)

[1] [Finn et. al. 2016], "Guided cost learning: Deep inverse optimal control via policy optimization", *ICML* 2016.
[2] [Bojarski et. al. 2016], "End to end learning for self-driving cars", arXiv:1604.07316, 2016.

RL without Reward

- It is often easier to provide some human expert demonstration
- Example videos:
  - Demonstration based learning in robotics[1] [Link]
  - Demonstration based learning in self-driving cars[2] [Link]
- How do we learn the optimal control policy from demonstration data?
- Inverse RL, Imitation Learning, Offline RL
- This lecture: Inverse Reinforcement Learning (IRL)
  - Infer the reward function that explains the expert behavior

[1] [Finn et. al. 2016], "Guided cost learning: Deep inverse optimal control via policy optimization", *ICML* 2016.
[2] [Bojarski et. al. 2016], "End to end learning for self-driving cars", arXiv:1604.07316, 2016.

RL without Reward

- It is often easier to provide some human expert demonstration
- Example videos:
  - Demonstration based learning in robotics[1] [Link]
  - Demonstration based learning in self-driving cars[2] [Link]
- How do we learn the optimal control policy from demonstration data?
- Inverse RL, Imitation Learning, Offline RL
- This lecture: Inverse Reinforcement Learning (IRL)
  - Infer the reward function that explains the expert behavior
  - Recover the expert policy

[1] [Finn et. al. 2016], "Guided cost learning: Deep inverse optimal control via policy optimization", *ICML* 2016.
[2] [Bojarski et. al. 2016], "End to end learning for self-driving cars", arXiv:1604.07316, 2016.

IRL Formulation

- Consider MDP *without* the reward function specified: $(\mathcal{S}, \mathcal{A}, P)$

IRL Formulation

- Consider MDP *without* the reward function specified: $(\mathcal{S}, \mathcal{A}, P)$
- We have the demonstration data form an expert

IRL Formulation

- Consider MDP *without* the reward function specified: $(\mathcal{S}, \mathcal{A}, P)$
- We have the demonstration data form an expert
  - Some trajectories generated according to an expert policy

$$\mathcal{D} = \{(s_0^i, a_0^i, s_1^i, a_1^i, \ldots, s_t^i, a_t^i, \ldots), i = 1, \ldots, n\}, \quad a_t^i \sim \pi_e(s_t^i)$$

IRL Formulation

- Consider MDP *without* the reward function specified: $(\mathcal{S}, \mathcal{A}, P)$
- We have the demonstration data form an expert
  - Some trajectories generated according to an expert policy

$$\mathcal{D} = \{(s_0^i, a_0^i, s_1^i, a_1^i, \ldots, s_t^i, a_t^i, \ldots), i = 1, \ldots, n\}, \quad a_t^i \sim \pi_e(s_t^i)$$

- Assumption: Expert is using the optimal policy with respect to a reward function r. This reward function is unknown to the algorithm.

$$\pi_e = \pi^* = \arg\max_{\pi} \ \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r(s_t, \pi(s_t))]$$

IRL Formulation

- Consider MDP *without* the reward function specified: $(\mathcal{S}, \mathcal{A}, P)$
- We have the demonstration data form an expert
  - Some trajectories generated according to an expert policy

$$\mathcal{D} = \{(s_0^i, a_0^i, s_1^i, a_1^i, \ldots, s_t^i, a_t^i, \ldots), i = 1, \ldots, n\}, \quad a_t^i \sim \pi_e(s_t^i)$$

- Assumption: Expert is using the optimal policy with respect to a reward function r. This reward function is unknown to the algorithm.

$$\pi_e = \pi^* = \arg\max_{\pi} \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r(s_t, \pi(s_t))]$$

- IRL objective: Estimate r and use that to recover the expert policy

IRL with Feature Vectors

- Assume that true reward function has the form $r(s,a) = (w^*)^\top \phi(s,a)$

IRL with Feature Vectors

- Assume that true reward function has the form $r(s,a) = (w^*)^\top \phi(s,a)$
- We know the feature ϕ. The goal is to find w^*

IRL with Feature Vectors

- Assume that true reward function has the form $r(s,a) = (w^*)^\top \phi(s,a)$
- We know the feature ϕ. The goal is to find w^*
- For any given π, define $\rho_\pi^\phi = \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t \phi(s_t, a_t)]$. Then

IRL with Feature Vectors

- Assume that true reward function has the form $r(s,a) = (w^*)^\top \phi(s,a)$
- We know the feature ϕ. The goal is to find w^*
- For any given π, define $\rho_\pi^\phi = \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t \phi(s_t, a_t)]$. Then

IRL with Feature Vectors

- Assume that true reward function has the form $r(s,a) = (w^*)^\top \phi(s,a)$
- We know the feature ϕ. The goal is to find w^*
- For any given π, define $\rho_\pi^\phi = \mathbb{E}_\pi[\sum_{t=0}^\infty \gamma^t \phi(s_t, a_t)]$. Then

$$V_\pi = \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t r(s_t,a_t)] = \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t w^{*\top} \phi(s_t,a_t)] = w^{*\top} \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t \phi(s_t,a_t)] = w^{*\top} \rho_\pi^\phi$$

IRL with Feature Vectors

- Assume that true reward function has the form $r(s,a) = (w^*)^\top \phi(s,a)$
- We know the feature ϕ. The goal is to find w^*
- For any given π, define $\rho_\pi^\phi = \mathbb{E}_\pi[\sum_{t=0}^\infty \gamma^t \phi(s_t, a_t)]$. Then

$$V_\pi = \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t r(s_t,a_t)] = \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t w^{*\top} \phi(s_t,a_t)] = w^{*\top} \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t \phi(s_t,a_t)] = w^{*\top} \rho_\pi^\phi$$

- For the optimal policy π^*, we have $V_{\pi^*} \geq V_\pi, \forall \pi$

IRL with Feature Vectors

- Assume that true reward function has the form $r(s,a) = (w^*)^\top \phi(s,a)$
- We know the feature ϕ. The goal is to find w^*
- For any given π, define $\rho_\pi^\phi = \mathbb{E}_\pi[\sum_{t=0}^\infty \gamma^t \phi(s_t, a_t)]$. Then

$$V_\pi = \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t r(s_t,a_t)] = \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t w^{*\top} \phi(s_t,a_t)] = w^{*\top} \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t \phi(s_t,a_t)] = w^{*\top} \rho_\pi^\phi$$

- For the optimal policy π^*, we have $V_{\pi^*} \geq V_\pi, \forall \pi$
- This can be written as $w^{*\top} \rho_{\pi^*}^\phi \geq w^{*\top} \rho_\pi^\phi, \ \forall \pi$

IRL with Feature Vectors

- Assume that true reward function has the form $r(s,a) = (w^*)^\top \phi(s,a)$
- We know the feature ϕ. The goal is to find w^*
- For any given π, define $\rho_\pi^\phi = \mathbb{E}_\pi[\sum_{t=0}^\infty \gamma^t \phi(s_t, a_t)]$. Then

$$V_\pi = \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t r(s_t,a_t)] = \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t w^{*\top} \phi(s_t,a_t)] = w^{*\top} \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t \phi(s_t,a_t)] = w^{*\top} \rho_\pi^\phi$$

- For the optimal policy π^*, we have $V_{\pi^*} \geq V_\pi, \forall \pi$
- This can be written as $w^{*\top} \rho_{\pi^*}^\phi \geq w^{*\top} \rho_\pi^\phi, \ \forall \pi$
- Equivalently, $w^{*\top}(\rho_{\pi^*}^\phi - \rho_\pi^\phi) \geq 0, \ \forall \pi$

IRL with Feature Vectors

- Assume that true reward function has the form $r(s,a) = (w^*)^\top \phi(s,a)$
- We know the feature ϕ. The goal is to find w^*
- For any given π, define $\rho_\pi^\phi = \mathbb{E}_\pi[\sum_{t=0}^\infty \gamma^t \phi(s_t, a_t)]$. Then

$$V_\pi = \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t r(s_t,a_t)] = \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t w^{*\top} \phi(s_t,a_t)] = w^{*\top} \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t \phi(s_t,a_t)] = w^{*\top} \rho_\pi^\phi$$

- For the optimal policy π^*, we have $V_{\pi^*} \geq V_\pi, \forall \pi$
- This can be written as $w^{*\top} \rho_{\pi^*}^\phi \geq w^{*\top} \rho_\pi^\phi, \ \forall \pi$
- Equivalently, $w^{*\top}(\rho_{\pi^*}^\phi - \rho_\pi^\phi) \geq 0, \ \forall \pi$
- Clearly, we can find infinitely many solutions (w^*) that satisfies the above inequality

IRL with Feature Vectors

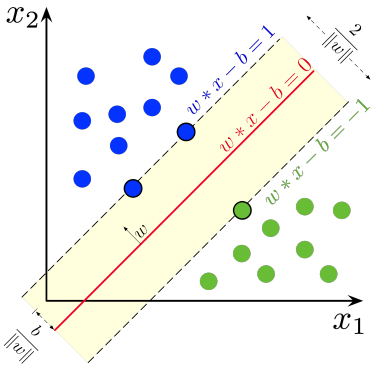
- Assume that true reward function has the form $r(s,a) = (w^*)^\top \phi(s,a)$
- We know the feature ϕ. The goal is to find w^*
- For any given π, define $\rho_\pi^\phi = \mathbb{E}_\pi[\sum_{t=0}^\infty \gamma^t \phi(s_t, a_t)]$. Then

$$V_\pi = \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t r(s_t,a_t)] = \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t w^{*\top} \phi(s_t,a_t)] = w^{*\top} \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t \phi(s_t,a_t)] = w^{*\top} \rho_\pi^\phi$$

- For the optimal policy π^*, we have $V_{\pi^*} \geq V_\pi, \forall \pi$
- This can be written as $w^{*\top} \rho_{\pi^*}^\phi \geq w^{*\top} \rho_\pi^\phi, \ \forall \pi$
- Equivalently, $w^{*\top}(\rho_{\pi^*}^\phi - \rho_\pi^\phi) \geq 0, \ \forall \pi$
- Clearly, we can find infinitely many solutions (w^*) that satisfies the above inequality
- How do we select one w (equivalently one r) from possibly infinite solutions?

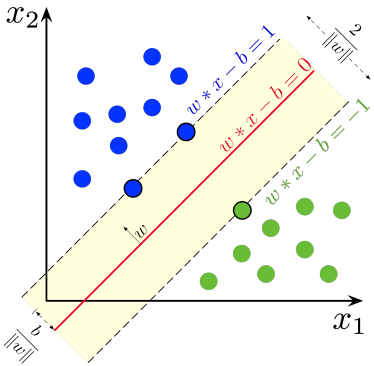
Maximum Margin Learning in SVM

- Support Vector Machine (SVM): Find a hyperplane that separates two classes
- There are infinitely many solutions
- How do we select one?



Maximum Margin Learning in SVM

- Support Vector Machine (SVM): Find a hyperplane that separates two classes
- There are infinitely many solutions
- How do we select one?



- Maximum margin principle

$$\max_{w} \frac{1}{||w||}$$

s.t. w separates the data set

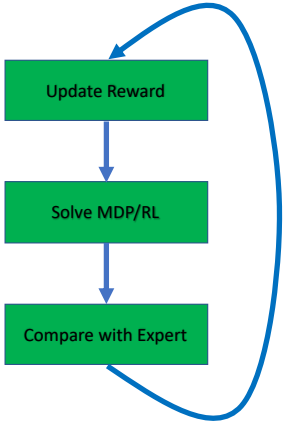
Maximum Margin IRL

Algorithm 1 Maximum Margin IRL

1: Initialization: Set $i = 0$. Select initial policy π_0. Estimate $\rho^{\phi}_{\pi_0}$ by generating data according to π_0 using the simulator
2: **for** $i = 1, 2, \ldots$ **do**
3: Solve for w_i, m_i:

$$\max_{m,w} \; m$$
$$\text{s.t.} \;\; w^\top \rho^{\phi}_{\pi^*} \geq w^\top \rho^{\phi}_{\pi_j} + m, \;\; j = 0, 1, \ldots, i-1$$
$$||w||_2 \leq 1$$

4: **if** $m_i \leq \epsilon$ **then**
5: Terminate the loop
6: **end if**
7: Learn the optimal policy π^i for the MDP using rewards $r(s,a) = w_i^\top \phi(s,a)$
8: Estimate $\rho^{\phi}_{\pi_i}$
9: **end for**



- P. Abbeel and A. Ng, "Apprenticeship Learning via Inverse Reinforcement Learning ", International Conference on Machine Learning (ICML), 2004
- Ratliff, Bagnell, Zinkevich, "Maximum Margin Planning", International Conference on Machine Learning (ICML), 2006

Principle of Maximum Entropy

- The basic idea used in apprenticeship learning is to find a π_{al} such $\rho^{\phi}_{\pi_{al}} \approx \rho^{\phi}_{\pi^*}$

Principle of Maximum Entropy

- The basic idea used in apprenticeship learning is to find a π_{al} such $\rho^{\phi}_{\pi_{al}} \approx \rho^{\phi}_{\pi^*}$
- There are infinitely many stochastic policies that can match the feature distribution of the optimal expert policy. How do we select one policy from this?

Principle of Maximum Entropy

- The basic idea used in apprenticeship learning is to find a π_{al} such $\rho^{\phi}_{\pi_{al}} \approx \rho^{\phi}_{\pi^*}$
- There are infinitely many stochastic policies that can match the feature distribution of the optimal expert policy. How do we select one policy from this?
- Principle of Maximum Entropy: The probability distribution which best represents the current state of knowledge is the one with largest entropy [Wiki Link]

Principle of Maximum Entropy

- The basic idea used in apprenticeship learning is to find a π_{al} such $\rho^{\phi}_{\pi_{al}} \approx \rho^{\phi}_{\pi^*}$
- There are infinitely many stochastic policies that can match the feature distribution of the optimal expert policy. How do we select one policy from this?
- Principle of Maximum Entropy: The probability distribution which best represents the current state of knowledge is the one with largest entropy [Wiki Link]
- Entropy of a distribution is a measure of its inherent uncertainty

Principle of Maximum Entropy

- The basic idea used in apprenticeship learning is to find a π_{al} such $\rho^{\phi}_{\pi_{al}} \approx \rho^{\phi}_{\pi^*}$
- There are infinitely many stochastic policies that can match the feature distribution of the optimal expert policy. How do we select one policy from this?
- Principle of Maximum Entropy: The probability distribution which best represents the current state of knowledge is the one with largest entropy [Wiki Link]
- Entropy of a distribution is a measure of its inherent uncertainty

Principle of Maximum Entropy

- The basic idea used in apprenticeship learning is to find a π_{al} such $\rho^{\phi}_{\pi_{al}} \approx \rho^{\phi}_{\pi^*}$
- There are infinitely many stochastic policies that can match the feature distribution of the optimal expert policy. How do we select one policy from this?
- Principle of Maximum Entropy: The probability distribution which best represents the current state of knowledge is the one with largest entropy [Wiki Link]
- Entropy of a distribution is a measure of its inherent uncertainty

Principle of Maximum Entropy

- The basic idea used in apprenticeship learning is to find a π_{al} such $\rho^{\phi}_{\pi_{al}} \approx \rho^{\phi}_{\pi^*}$
- There are infinitely many stochastic policies that can match the feature distribution of the optimal expert policy. How do we select one policy from this?
- Principle of Maximum Entropy: The probability distribution which best represents the current state of knowledge is the one with largest entropy [Wiki Link]
- Entropy of a distribution is a measure of its inherent uncertainty

- Let X be a random variable and $p(\cdot)$ be a distribution on X. The entropy of the distribution p, denote as $H(p)$, is defined as

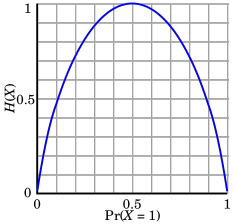
$$H(p) = -\sum_{x} p(x) \log p(x)$$

Principle of Maximum Entropy

- The basic idea used in apprenticeship learning is to find a π_{al} such $\rho^{\phi}_{\pi_{al}} \approx \rho^{\phi}_{\pi^*}$
- There are infinitely many stochastic policies that can match the feature distribution of the optimal expert policy. How do we select one policy from this?
- Principle of Maximum Entropy: The probability distribution which best represents the current state of knowledge is the one with largest entropy [Wiki Link]
- Entropy of a distribution is a measure of its inherent uncertainty

- Let X be a random variable and $p(\cdot)$ be a distribution on X. The entropy of the distribution p, denote as $H(p)$, is defined as

$$H(p) = -\sum_{x} p(x) \log p(x)$$



Maximum Entropy IRL

- Maximum Entropy IRL:

$$\begin{aligned} \max_{\pi} \quad & -\mathbb{E}_{(s,a)\sim\rho_\pi(\cdot)}[\log \pi(s,a)] \\ \text{s.t.} \quad & \mathbb{E}_{(s,a)\sim\rho_\pi(\cdot)}[\phi(s,a)] = \mathbb{E}_{(s,a)\sim\rho_{\pi^*}(\cdot)}[\phi(s,a)] \end{aligned}$$

Maximum Entropy IRL

- Maximum Entropy IRL:

$$\begin{aligned} \max_{\pi} \quad & -\mathbb{E}_{(s,a)\sim\rho_\pi(\cdot)}[\log \pi(s,a)] \\ \text{s.t.} \quad & \mathbb{E}_{(s,a)\sim\rho_\pi(\cdot)}[\phi(s,a)] = \mathbb{E}_{(s,a)\sim\rho_{\pi^*}(\cdot)}[\phi(s,a)] \end{aligned}$$

- Using Lagrangian

$$\min_{w} \max_{\pi} \left(\mathbb{E}_{(s,a)\sim\rho_\pi}[w^\top \phi(s,a)] - \mathbb{E}_{(s,a)\sim\rho_\pi}[\log \pi(s,a)] - \mathbb{E}_{(s,a)\sim\rho_{\pi^*}}[w^\top \phi(s,a)] \right)$$

Maximum Entropy IRL

- Maximum Entropy IRL:

$$\max_{\pi} \quad -\mathbb{E}_{(s,a)\sim\rho_\pi(\cdot)}[\log \pi(s,a)]$$
$$\text{s.t.} \quad \mathbb{E}_{(s,a)\sim\rho_\pi(\cdot)}[\phi(s,a)] = \mathbb{E}_{(s,a)\sim\rho_{\pi^*}(\cdot)}[\phi(s,a)]$$

- Using Lagrangian

$$\min_{w} \max_{\pi} \left(\mathbb{E}_{(s,a)\sim\rho_\pi}[w^\top \phi(s,a)] - \mathbb{E}_{(s,a)\sim\rho_\pi}[\log \pi(s,a)] - \mathbb{E}_{(s,a)\sim\rho_{\pi^*}}[w^\top \phi(s,a)]\right)$$

- The inner optimization problem (find the best policy given w)

$$\max_{\pi} \left(\mathbb{E}_{(s,a)\sim\rho_\pi}[w^\top \phi(s,a)] - \mathbb{E}_{(s,a)\sim\rho_\pi}[\log \pi(s,a)]\right)$$

  - Find the policy that maximizes the value function with an additional entropy regularization cost

Maximum Entropy IRL

- Maximum Entropy IRL:

$$\max_{\pi} \quad -\mathbb{E}_{(s,a)\sim\rho_\pi(\cdot)}[\log \pi(s,a)]$$
$$\text{s.t.} \quad \mathbb{E}_{(s,a)\sim\rho_\pi(\cdot)}[\phi(s,a)] = \mathbb{E}_{(s,a)\sim\rho_{\pi^*}(\cdot)}[\phi(s,a)]$$

- Using Lagrangian

$$\min_{w} \max_{\pi} \left(\mathbb{E}_{(s,a)\sim\rho_\pi}[w^\top \phi(s,a)] - \mathbb{E}_{(s,a)\sim\rho_\pi}[\log \pi(s,a)] - \mathbb{E}_{(s,a)\sim\rho_{\pi^*}}[w^\top \phi(s,a)]\right)$$

- The inner optimization problem (find the best policy given w)

$$\max_{\pi} \left(\mathbb{E}_{(s,a)\sim\rho_\pi}[w^\top \phi(s,a)] - \mathbb{E}_{(s,a)\sim\rho_\pi}[\log \pi(s,a)]\right)$$

  - Find the policy that maximizes the value function with an additional entropy regularization cost
  - This is maximum entropy RL where because the objective is

$$\max_{\pi} \left(\mathbb{E}_{(s,a)\sim\rho_\pi}[w^\top \phi(s,a) + H(\pi(s,\cdot))]\right)$$

Maximum Entropy IRL

- Maximum Entropy IRL:

$$\max_{\pi} \quad -\mathbb{E}_{(s,a)\sim\rho_\pi(\cdot)}[\log \pi(s,a)]$$
$$\text{s.t.} \quad \mathbb{E}_{(s,a)\sim\rho_\pi(\cdot)}[\phi(s,a)] = \mathbb{E}_{(s,a)\sim\rho_{\pi^*}(\cdot)}[\phi(s,a)]$$

- Using Lagrangian

$$\min_{w} \max_{\pi} \left(\mathbb{E}_{(s,a)\sim\rho_\pi}[w^\top \phi(s,a)] - \mathbb{E}_{(s,a)\sim\rho_\pi}[\log \pi(s,a)] - \mathbb{E}_{(s,a)\sim\rho_{\pi^*}}[w^\top \phi(s,a)]\right)$$

- The inner optimization problem (find the best policy given w)

$$\max_{\pi} \left(\mathbb{E}_{(s,a)\sim\rho_\pi}[w^\top \phi(s,a)] - \mathbb{E}_{(s,a)\sim\rho_\pi}[\log \pi(s,a)]\right)$$

  - Find the policy that maximizes the value function with an additional entropy regularization cost
  - This is maximum entropy RL where because the objective is

$$\max_{\pi} \left(\mathbb{E}_{(s,a)\sim\rho_\pi}[w^\top \phi(s,a) + H(\pi(s,\cdot))]\right)$$

- Outer optimization problem (update w)

$$w \leftarrow w + \alpha \left(\mathbb{E}_{(s,a)\sim\rho_\pi}[\phi(s,a)] - \mathbb{E}_{(s,a)\sim\rho_{\pi^*}}[\phi(s,a)]\right)$$

A Relaxed Problem

- Maximum Entropy IRL:

$$\max_{\pi} \; H(\pi) \;\; \text{s.t.} \;\; \mathbb{E}_{(s,a)\sim\rho_{\pi}(\cdot)}[\phi(s,a)] = \mathbb{E}_{(s,a)\sim\rho_{\pi^*}(\cdot)}[\phi(s,a)]$$

A Relaxed Problem

- Maximum Entropy IRL:

$$\max_{\pi} \; H(\pi) \;\; \text{s.t.} \;\; \mathbb{E}_{(s,a)\sim\rho_{\pi}(\cdot)}[\phi(s,a)] = \mathbb{E}_{(s,a)\sim\rho_{\pi^*}(\cdot)}[\phi(s,a)]$$

- We will replace the above formulation as

$$\max_{\pi} \; H(\pi) \;\; \text{s.t.} \;\; d(\rho_{\pi}, \rho_{\pi_e}) \leq \epsilon$$

A Relaxed Problem

- Maximum Entropy IRL:

$$\max_{\pi} \; H(\pi) \;\; \text{s.t.} \;\; \mathbb{E}_{(s,a)\sim\rho_{\pi}(\cdot)}[\phi(s,a)] = \mathbb{E}_{(s,a)\sim\rho_{\pi^*}(\cdot)}[\phi(s,a)]$$

- We will replace the above formulation as

$$\max_{\pi} \; H(\pi) \;\; \text{s.t.} \;\; d(\rho_{\pi}, \rho_{\pi_e}) \leq \epsilon$$

  - Here, $d(\cdot,\cdot)$ is an appropriate distance metric between two distributions

A Relaxed Problem

- Maximum Entropy IRL:

$$\max_{\pi} \; H(\pi) \;\; \text{s.t.} \;\; \mathbb{E}_{(s,a)\sim\rho_{\pi}(\cdot)}[\phi(s,a)] = \mathbb{E}_{(s,a)\sim\rho_{\pi^*}(\cdot)}[\phi(s,a)]$$

- We will replace the above formulation as

$$\max_{\pi} \; H(\pi) \;\; \text{s.t.} \;\; d(\rho_{\pi}, \rho_{\pi_e}) \leq \epsilon$$

  - Here, $d(\cdot,\cdot)$ is an appropriate distance metric between two distributions

- We will then solve the relaxed optimization problem

$$\min_{\pi} \; d(\rho_{\pi}, \rho_{\pi_e}) - H(\pi)$$

A Relaxed Problem

- Maximum Entropy IRL:

$$\max_{\pi} \; H(\pi) \;\; \text{s.t.} \;\; \mathbb{E}_{(s,a)\sim\rho_{\pi}(\cdot)}[\phi(s,a)] = \mathbb{E}_{(s,a)\sim\rho_{\pi^*}(\cdot)}[\phi(s,a)]$$

- We will replace the above formulation as

$$\max_{\pi} \; H(\pi) \;\; \text{s.t.} \;\; d(\rho_{\pi}, \rho_{\pi_e}) \leq \epsilon$$

  - Here, $d(\cdot,\cdot)$ is an appropriate distance metric between two distributions

- We will then solve the relaxed optimization problem

$$\min_{\pi} \; d(\rho_{\pi}, \rho_{\pi_e}) - H(\pi)$$

- How do we evaluate $d(\cdot,\cdot)$?

Discriminator as Pseudo-Reward Function

- Imitation learning problem: $\min_\pi \quad d(\rho_\pi, \rho_{\pi_e}) - H(\pi)$

Discriminator as Pseudo-Reward Function

- Imitation learning problem: $\min_\pi \quad d(\rho_\pi, \rho_{\pi_e}) - H(\pi)$
- How do we evaluate $d(\cdot, \cdot)$?

Discriminator as Pseudo-Reward Function

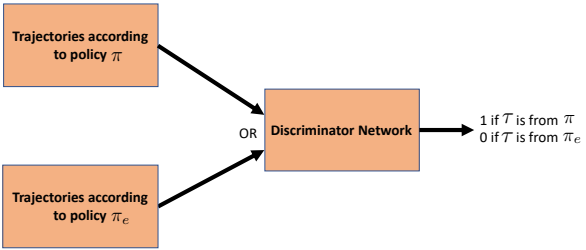
- Imitation learning problem: $\min_\pi \quad d(\rho_\pi, \rho_{\pi_e}) - H(\pi)$
- How do we evaluate $d(\cdot, \cdot)$?
- We can use a discriminator! (motivated from the Generative Adversarial Network (GAN) algorithms)

Discriminator as Pseudo-Reward Function

- Imitation learning problem: $\min_\pi \quad d(\rho_\pi, \rho_{\pi_e}) - H(\pi)$
- How do we evaluate $d(\cdot, \cdot)$?
- We can use a discriminator! (motivated from the Generative Adversarial Network (GAN) algorithms)

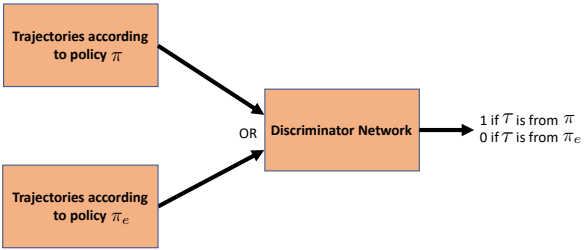
Discriminator as Pseudo-Reward Function

- Imitation learning problem: $\min_\pi \quad d(\rho_\pi, \rho_{\pi_e}) - H(\pi)$
- How do we evaluate $d(\cdot, \cdot)$?
- We can use a discriminator! (motivated from the Generative Adversarial Network (GAN) algorithms)



Discriminator as Pseudo-Reward Function

- Imitation learning problem: $\min_\pi \quad d(\rho_\pi, \rho_{\pi_e}) - H(\pi)$
- How do we evaluate $d(\cdot, \cdot)$?
- We can use a discriminator! (motivated from the Generative Adversarial Network (GAN) algorithms)



- This can be written as

$$d(\rho_\pi, \rho_{\pi_e}) = \max_D \quad \mathbb{E}_{(s,a)\sim\rho_\pi}[\log D(s,a)] + \mathbb{E}_{(s,a)\sim\rho_{\pi_e}}[\log(1 - D(s,a))]$$

GAIL

- Generative Adversarial Imitation Learning (GAIL)[3]

$$\min_{\theta} \max_{w} \quad \mathbb{E}_{(s,a)\sim\rho_{\pi_\theta}}[\log D_w(s,a)] + \mathbb{E}_{(s,a)\sim\rho_{\pi_e}}[\log(1 - D_w(s,a))] - \lambda H(\pi_\theta)$$

[3] Ho and Ermon, "Generative adversarial imitation learning", NeurIPS, 2016.

GAIL

- Generative Adversarial Imitation Learning (GAIL)[3]

$$\min_{\theta} \max_{w} \; \mathbb{E}_{(s,a)\sim\rho_{\pi_\theta}}[\log D_w(s,a)] + \mathbb{E}_{(s,a)\sim\rho_{\pi_e}}[\log(1 - D_w(s,a))] - \lambda H(\pi_\theta)$$

- For a given discriminator parameter w, the policy optimization problem is

$$\min_{\theta} \; \Big(\mathbb{E}_{(s,a)\sim\rho_{\pi_\theta}}[\log D_w(s,a)] - \lambda H(\pi_\theta)\Big)$$

[3] Ho and Ermon, "Generative adversarial imitation learning", NeurIPS, 2016.

GAIL

- Generative Adversarial Imitation Learning (GAIL)[3]

$$\min_{\theta} \max_{w} \; \mathbb{E}_{(s,a)\sim\rho_{\pi_\theta}}[\log D_w(s,a)] + \mathbb{E}_{(s,a)\sim\rho_{\pi_e}}[\log(1 - D_w(s,a))] - \lambda H(\pi_\theta)$$

- For a given discriminator parameter w, the policy optimization problem is

$$\min_{\theta} \left(\mathbb{E}_{(s,a)\sim\rho_{\pi_\theta}}[\log D_w(s,a)] - \lambda H(\pi_\theta) \right)$$

- This is a maximum entropy RL problem, with reward function $R(s,a) = -\log D_w(s,a)$

[3] Ho and Ermon, "Generative adversarial imitation learning", NeurIPS, 2016.

GAIL

- Generative Adversarial Imitation Learning (GAIL)[3]

$$\min_{\theta} \max_{w} \; \mathbb{E}_{(s,a)\sim\rho_{\pi_\theta}}[\log D_w(s,a)] + \mathbb{E}_{(s,a)\sim\rho_{\pi_e}}[\log(1 - D_w(s,a))] - \lambda H(\pi_\theta)$$

- For a given discriminator parameter w, the policy optimization problem is

$$\min_{\theta} \left(\mathbb{E}_{(s,a)\sim\rho_{\pi_\theta}}[\log D_w(s,a)] - \lambda H(\pi_\theta) \right)$$

- This is a maximum entropy RL problem, with reward function $R(s,a) = -\log D_w(s,a)$
- Update w (discriminator):

$$w \leftarrow w + \alpha_k \, \nabla_w \left(\mathbb{E}_{(s,a)\sim\rho_{\pi_\theta}}[\log D_w(s,a)] + \mathbb{E}_{(s,a)\sim\rho_{\pi_e}}[\log(1 - D_w(s,a))] \right)$$

[3] Ho and Ermon, "Generative adversarial imitation learning", NeurIPS, 2016.

GAIL

- Generative Adversarial Imitation Learning (GAIL)[3]

$$\min_{\theta} \max_{w} \; \mathbb{E}_{(s,a)\sim\rho_{\pi_\theta}}[\log D_w(s,a)] + \mathbb{E}_{(s,a)\sim\rho_{\pi_e}}[\log(1 - D_w(s,a))] - \lambda H(\pi_\theta)$$

- For a given discriminator parameter w, the policy optimization problem is

$$\min_{\theta} \; \Big(\mathbb{E}_{(s,a)\sim\rho_{\pi_\theta}}[\log D_w(s,a)] - \lambda H(\pi_\theta)\Big)$$

- This is a maximum entropy RL problem, with reward function $R(s,a) = -\log D_w(s,a)$
- Update w (discriminator):

$$w \leftarrow w + \alpha_k \; \nabla_w \Big(\mathbb{E}_{(s,a)\sim\rho_{\pi_\theta}}[\log D_w(s,a)] + \mathbb{E}_{(s,a)\sim\rho_{\pi_e}}[\log(1 - D_w(s,a))]\Big)$$

- Update θ (policy):

$$\theta \leftarrow \theta - \beta_k \; \nabla_\theta \Big(\mathbb{E}_{(s,a)\sim\rho_{\pi_\theta}}[\log D_w(s,a)] - \lambda H(\pi_\theta)\Big)$$

[3] Ho and Ermon, "Generative adversarial imitation learning", NeurIPS, 2016.

Generative Adversarial Networks

References for GAN

- "Introduction to GANs", CVPR 2018 Tutorial on GANs by Ian Goodfellow [Link]
- Goodfellow et al. “Generative adversarial networks”, NeurIPS, 2014

- Figures from the above two references are used in this lecture

Generative Models

- We have a set of data $\{x_i\}_{i=1}^n$

Generative Models

- We have a set of data $\{x_i\}_{i=1}^n$
- Data is generated according to an unknown distribution, $x_i \sim p_{\text{data}}(\cdot)$

Generative Models

- We have a set of data $\{x_i\}_{i=1}^n$
- Data is generated according to an unknown distribution, $x_i \sim p_{\text{data}}(\cdot)$
- Suppose we want more data samples (why?)

Generative Models

- We have a set of data $\{x_i\}_{i=1}^n$
- Data is generated according to an unknown distribution, $x_i \sim p_{\text{data}}(\cdot)$
- Suppose we want more data samples (why?)
- Given the training data, how do we generate new data samples that are not in the original training data?

Generative Models

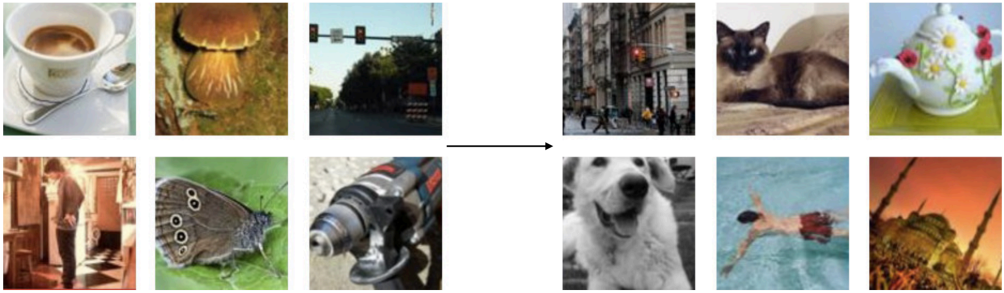
- We have a set of data $\{x_i\}_{i=1}^n$
- Data is generated according to an unknown distribution, $x_i \sim p_{\text{data}}(\cdot)$
- Suppose we want more data samples (why?)
- Given the training data, how do we generate new data samples that are not in the original training data?
- Can we estimate a model p_{model} and generate samples $x_i' \sim p_{\text{model}}(\cdot)$, according to this model?

Generative Models

- Generative models: Given training data, generate new samples from same distribution

Generative Models

- Generative models: Given training data, generate new samples from same distribution

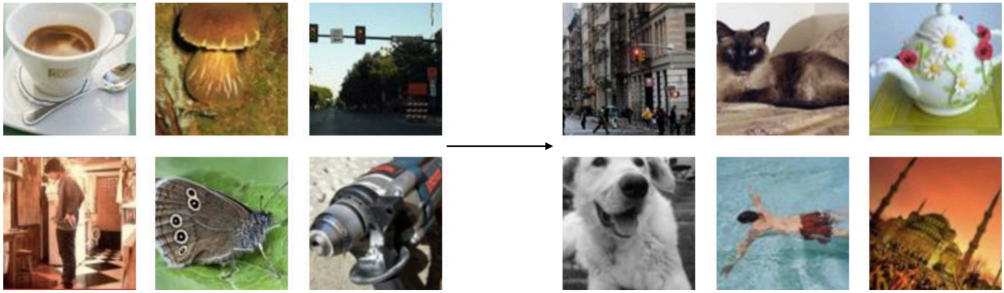


Training examples

Model samples

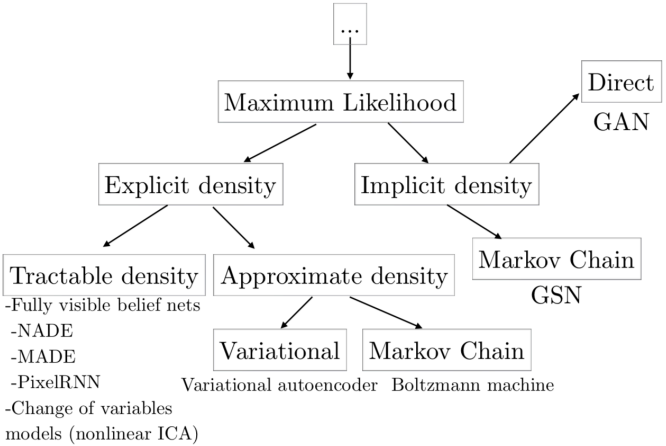
Generative Models

- Generative models: Given training data, generate new samples from same distribution



- Given the training data, say, samples from ImageNet, can we generate new images that are not in the original training data?

Generative Models

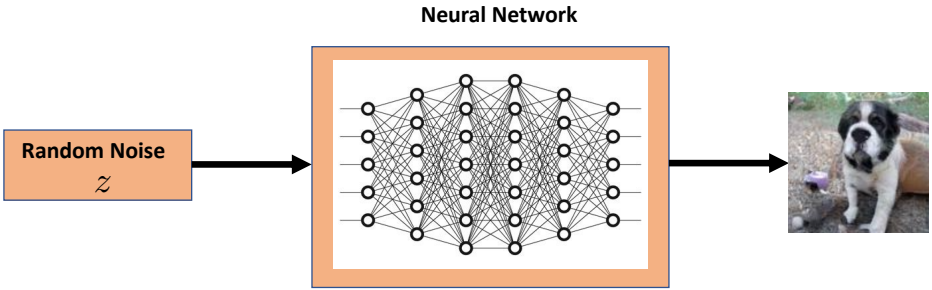


Generative Adversarial Network (GAN)

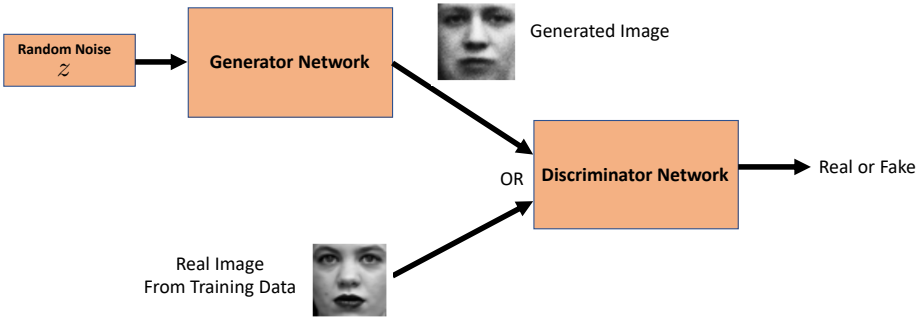
- GAN: A neural network that takes a random noise sample as the input and output a realistic image which was not in the training data!

Generative Adversarial Network (GAN)

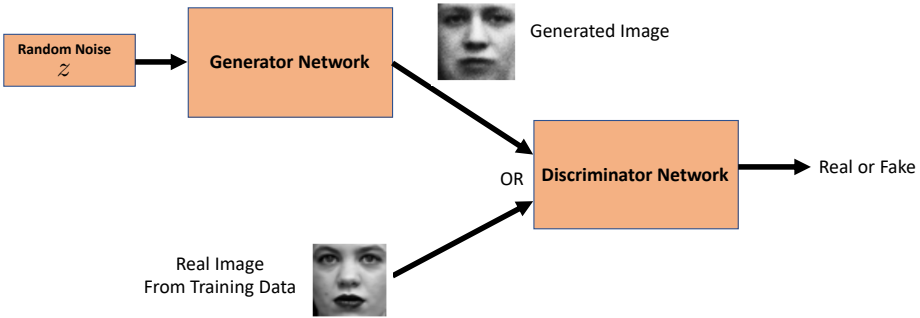
- GAN: A neural network that takes a random noise sample as the input and output a realistic image which was not in the training data!



Generator and Discriminator

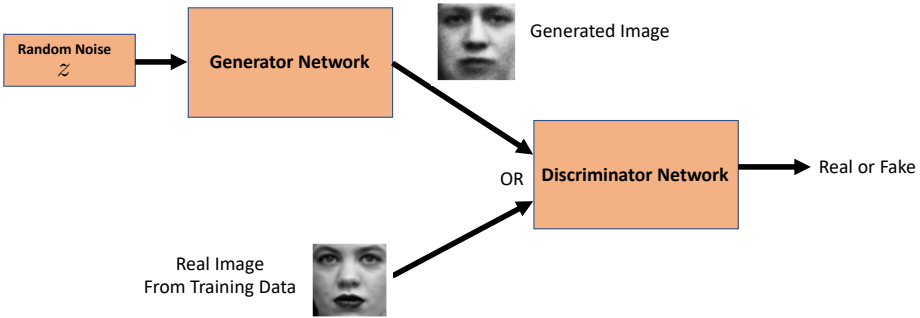


Generator and Discriminator



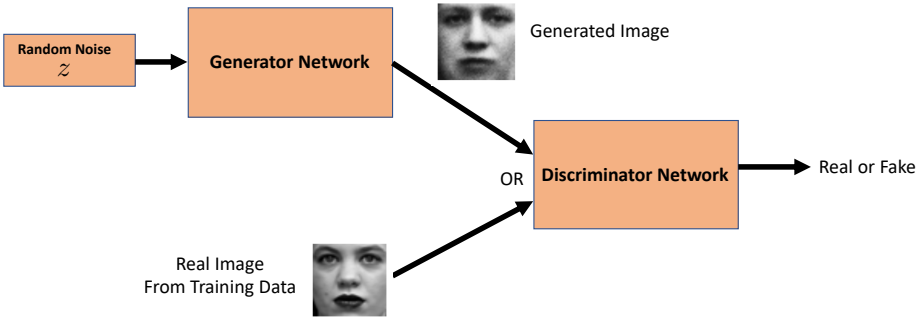
- Generator: Goal is to produce realistic-looking images. It tries to fool the discriminator by generating realistic-looking images.

Generator and Discriminator



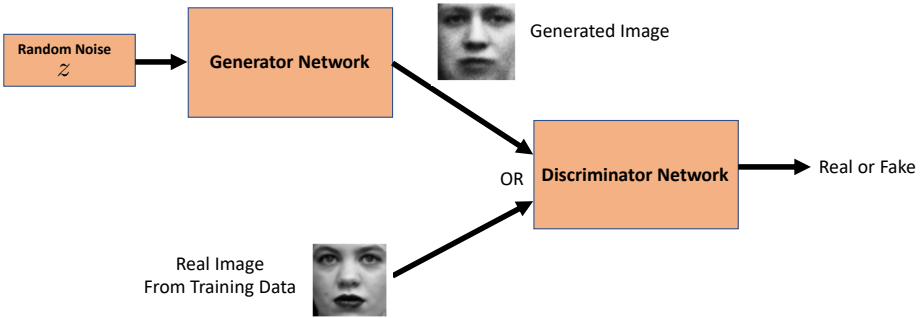
- Generator: Goal is to produce realistic-looking images. It tries to fool the discriminator by generating realistic-looking images.
  - "analogous to a team of counterfeiters, trying to produce fake currency and use it without detection"

Generator and Discriminator



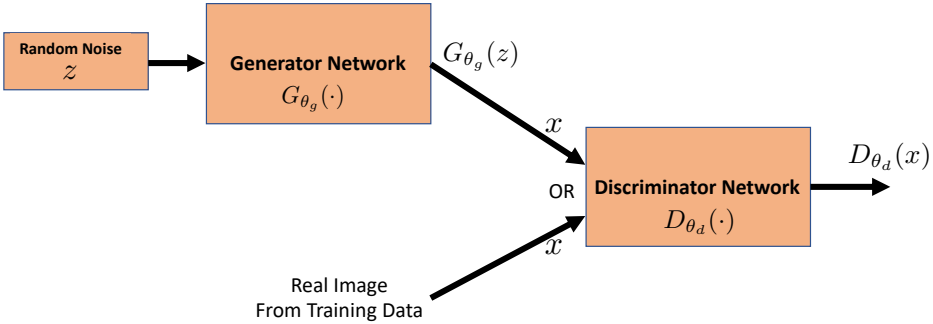
- Generator: Goal is to produce realistic-looking images. It tries to fool the discriminator by generating realistic-looking images.
  - "analogous to a team of counterfeiters, trying to produce fake currency and use it without detection"
- Discriminator: Goal is to distinguish between real and fake images. It tries to figure out whether an image came from the generator network.

Generator and Discriminator

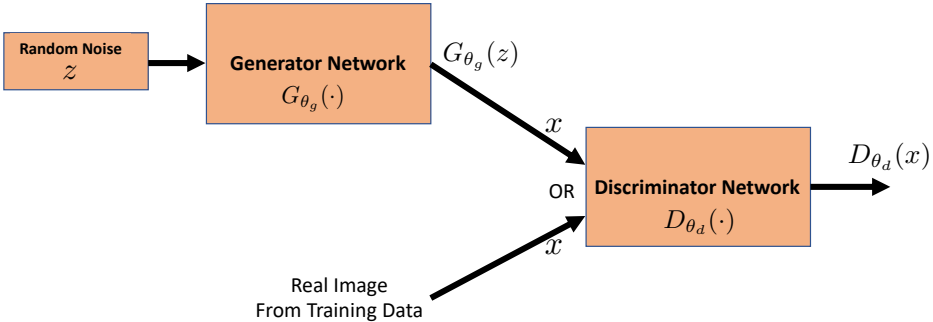


- Generator: Goal is to produce realistic-looking images. It tries to fool the discriminator by generating realistic-looking images.
  - "analogous to a team of counterfeiters, trying to produce fake currency and use it without detection"
- Discriminator: Goal is to distinguish between real and fake images. It tries to figure out whether an image came from the generator network.
  - "analogous to the police, trying to detect the counterfeit currency"

GAN Objective

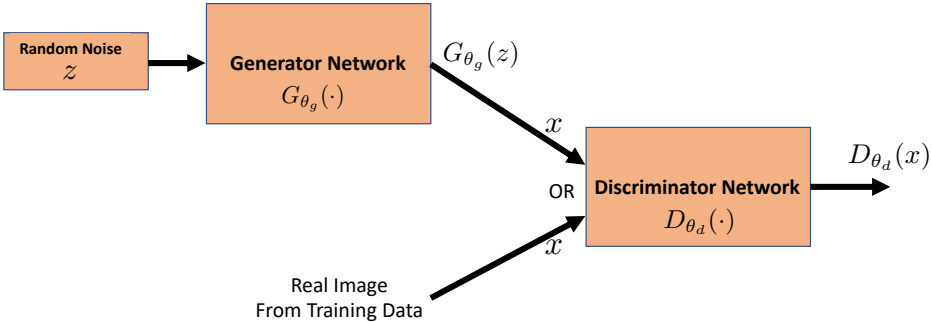


GAN Objective



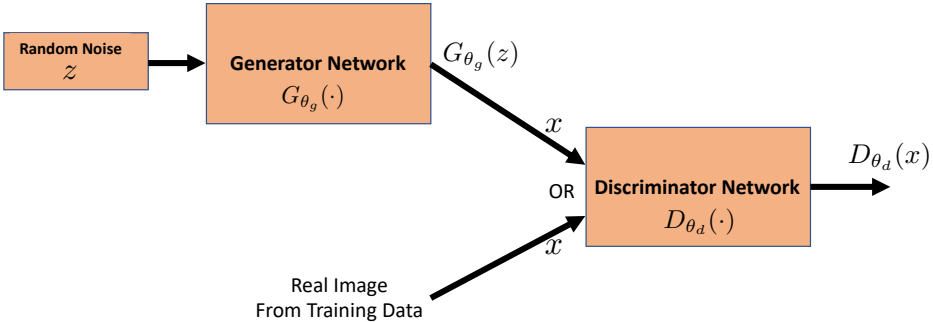
- Data distribution: p_{data}, Noise distribution: p_z, Generator output distribution: p_g

GAN Objective



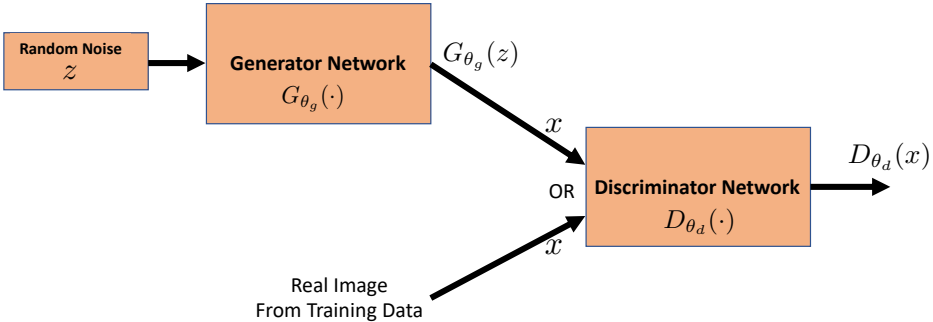
- Data distribution: p_{data}, Noise distribution: p_z,
  Generator output distribution: p_g
- Discriminator output $D_{\theta_d}(x)$ represents the probability that x is generated according to p_{data} rather than p_g

GAN Objective



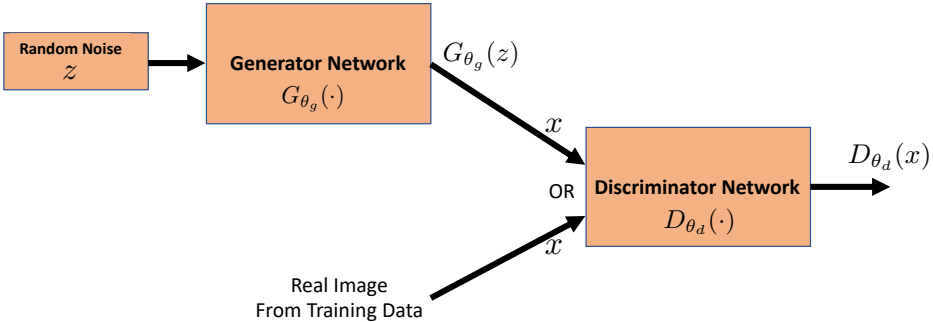
- Data distribution: p_{data}, Noise distribution: p_z,
  Generator output distribution: p_g
- Discriminator output $D_{\theta_d}(x)$ represents the probability that x is generated according to p_{data} rather than p_g
  - If x is a real-image from training data, then we want $D_{\theta_d}(x)$ to be close to 1.

GAN Objective



- Data distribution: p_{data}, Noise distribution: p_z,
  Generator output distribution: p_g
- Discriminator output $D_{\theta_d}(x)$ represents the probability that x is generated according to p_{data} rather than p_g
  - If x is a real-image from training data, then we want $D_{\theta_d}(x)$ to be close to 1.
  - On the other hand, we want $D_{\theta_d}(G_{\theta_g}(z))$ to be close to zero.

GAN Objective



- Data distribution: p_{data}, Noise distribution: p_z,
  Generator output distribution: p_g
- Discriminator output $D_{\theta_d}(x)$ represents the probability that x is generated according to p_{data} rather than p_g
  - If x is a real-image from training data, then we want $D_{\theta_d}(x)$ to be close to 1.
  - On the other hand, we want $D_{\theta_d}(G_{\theta_g}(z))$ to be close to zero.
- Our goal is to train generator and discriminator such that after training we will have $p_g \approx p_{\text{data}}$

Game and Value Function

- Discriminator objective

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

Game and Value Function

- Discriminator objective

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

- Generator objective

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

Game and Value Function

- Discriminator objective

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

- Generator objective

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

- This is a zero-sum game (minimax game) between the generator and the discriminator

Game and Value Function

- Discriminator objective

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

- Generator objective

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

- This is a zero-sum game (minimax game) between the generator and the discriminator
- Value function of the game

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

Game and Value Function

- Discriminator objective

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

- Generator objective

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

- This is a zero-sum game (minimax game) between the generator and the discriminator
- Value function of the game

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

Game and Value Function

- Discriminator objective

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

- Generator objective

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

- This is a zero-sum game (minimax game) between the generator and the discriminator
- Value function of the game

$$V(D, G) = \min_G \max_D \; \mathbb{E}_{x \sim p_{\textbf{data}}(\cdot)}[\log D(x)] + \mathbb{E}_{z \sim p_z(\cdot)}[\log(1 - D(G(z)))]$$

$$V(\theta_d, \theta_g) = \min_{\theta_g} \max_{\theta_d} \; \mathbb{E}_{x \sim p_{\textbf{data}}(\cdot)}[\log D_{\theta_d}(x)] + \mathbb{E}_{z \sim p_z(\cdot)}[\log(1 - D_{\theta_d}(G_{\theta_g}(z)))]$$

Training GAN

- Gradient ascent on discriminator

$$\max_{\theta_d} \mathbb{E}_{x \sim p_{\textbf{data}}(\cdot)}[\log D_{\theta_d}(x)] + \mathbb{E}_{z \sim p_z(\cdot)}[\log(1 - D_{\theta_d}(G_{\theta_g}(z)))]$$

Training GAN

- Gradient ascent on discriminator

$$\max_{\theta_d} \mathbb{E}_{x \sim p_{\mathbf{data}}(\cdot)}[\log D_{\theta_d}(x)] + \mathbb{E}_{z \sim p_z(\cdot)}[\log(1 - D_{\theta_d}(G_{\theta_g}(z)))]$$

- Gradient descent on generator

$$\min_{\theta_g} \mathbb{E}_{z \sim p_z(\cdot)}[\log(1 - D_{\theta_d}(G_{\theta_g}(z)))]$$

Training GAN

- Gradient ascent on discriminator

$$\max_{\theta_d} \mathbb{E}_{x \sim p_{\textbf{data}}(\cdot)}[\log D_{\theta_d}(x)] + \mathbb{E}_{z \sim p_z(\cdot)}[\log(1 - D_{\theta_d}(G_{\theta_g}(z)))]$$

- Gradient descent on generator

$$\min_{\theta_g} \mathbb{E}_{z \sim p_z(\cdot)}[\log(1 - D_{\theta_d}(G_{\theta_g}(z)))]$$

  - In practice, optimizing this generator objective does not work well!

Training GAN

- Gradient ascent on discriminator

$$\max_{\theta_d} \mathbb{E}_{x \sim p_{\textbf{data}}(\cdot)}[\log D_{\theta_d}(x)] + \mathbb{E}_{z \sim p_z(\cdot)}[\log(1 - D_{\theta_d}(G_{\theta_g}(z)))]$$

- Gradient descent on generator

$$\min_{\theta_g} \mathbb{E}_{z \sim p_z(\cdot)}[\log(1 - D_{\theta_d}(G_{\theta_g}(z)))]$$

  - In practice, optimizing this generator objective does not work well!
  - In practice, a modified approach is used

Training GAN

- Gradient ascent on discriminator

$$\max_{\theta_d} \mathbb{E}_{x \sim p_{\textbf{data}}(\cdot)}[\log D_{\theta_d}(x)] + \mathbb{E}_{z \sim p_z(\cdot)}[\log(1 - D_{\theta_d}(G_{\theta_g}(z)))]$$

- Gradient descent on generator

$$\min_{\theta_g} \mathbb{E}_{z \sim p_z(\cdot)}[\log(1 - D_{\theta_d}(G_{\theta_g}(z)))]$$

  - In practice, optimizing this generator objective does not work well!
  - In practice, a modified approach is used

- Gradient ascent on generator with modified objective

$$\max_{\theta_g} \mathbb{E}_{z \sim p_z(\cdot)}[\log D_{\theta_d}(G_{\theta_g}(z))]$$

GAN Training

Algorithm GAN Training

for number of training iterations **do**
  for k steps **do**
    Sample minibatch of m noise samples $\{z^{(1)}, \ldots, z^{(m)}\}$ according p_z.
    Sample minibatch of m data samples $\{x^{(1)}, \ldots, x^{(m)}\}$ according p_{data}.
    Update the discriminator by ascending its stochastic gradient:

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

  end for
  Sample minibatch of m noise samples $\{z^{(1)}, \ldots, z^{(m)}\}$ according p_z
  Update the generator by ascending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log D(G(z^{(i)}))$$

end for

GAN Training

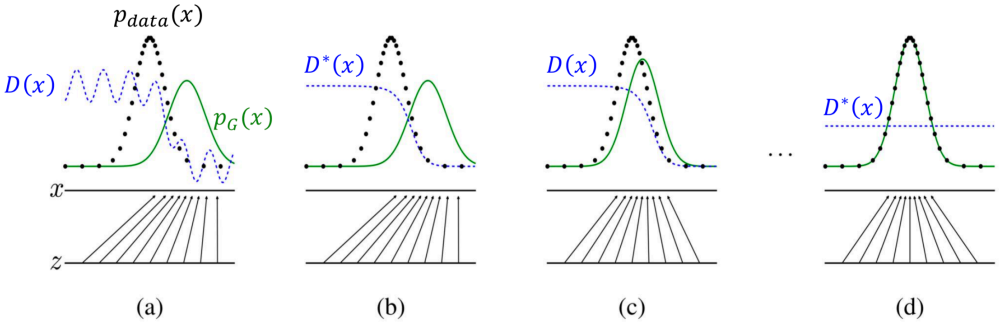
Proposition

For a fixed G, the optimal discriminator D_G^* is given by $D_G^* = \frac{p_{\textbf{data}}(x)}{p_{\textbf{data}}(x)+p_g(x)}$

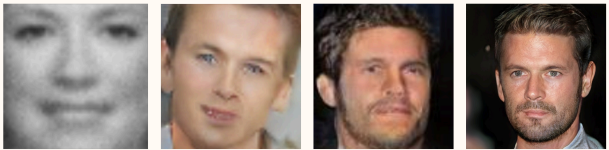
GAN Training

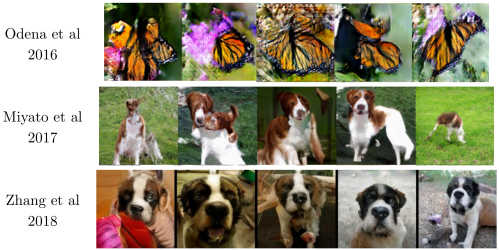
Proposition

For a fixed G, the optimal discriminator D_G^* is given by $D_G^* = \frac{p_{\textbf{data}}(x)}{p_{\textbf{data}}(x)+p_g(x)}$



GAN Examples





Images are taken from CVPR 2018 Tutorial slides by Ian Goodfellow

GAN Examples

- GAN vector algebra

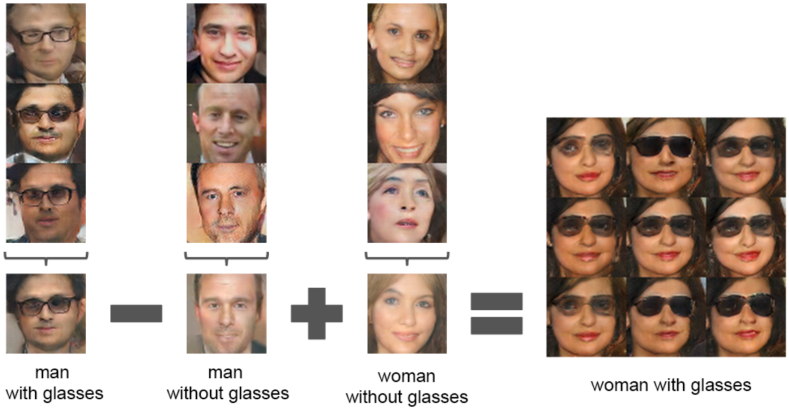


Image from: Radford and Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks", 2016

GAN Examples

- Image-to-Image Translation

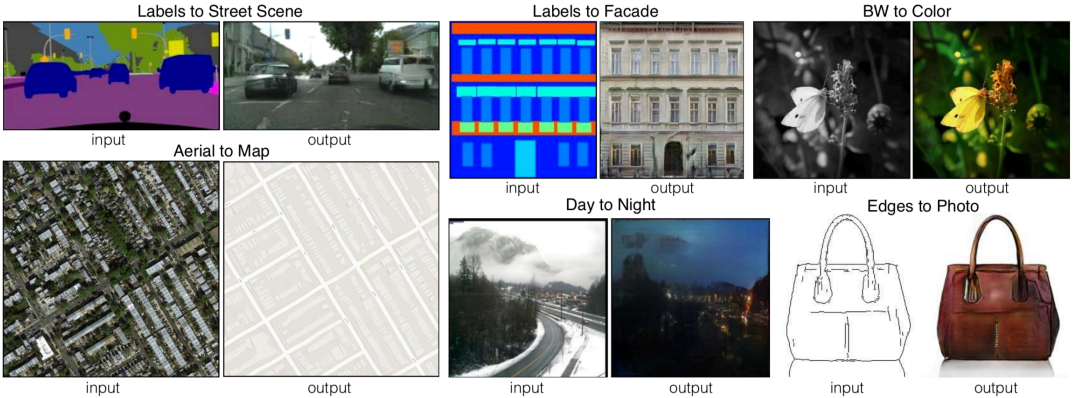


Image from: Isola et al, "Image-to-Image Translation with Conditional Adversarial Networks", 2017

GAN Examples

- Text-to-Image Synthesis

this small bird has a pink breast and crown, and black primaries and secondaries.

this magnificent fellow is almost all black with a red crest, and white cheek patch.

the flower has petals that are bright pinkish purple with white stigma

this white and yellow flower have thin white petals and a round yellow stamen

Image from: Reed et al, "Generative Adversarial Text to Image Synthesis", 2016

GAN Examples

Image from: Brock et al, "Large Scale GAN Training for High Fidelity Natural Image Synthesis", 2019