Generative Adversarial Imitation Learning

Jonathan Ho and Stefano Ermon

Presented by Xinyuan Zhang

January 6, 2017
Reinforcement Learning

Reinforcement learning is a powerful method for finding the optimal solution for sequential decision process. It passes a number of state-action-pairs to find the optimal action in any state in order to maximize a numerical reward/ minimize a numerical cost.

- $S = \{s_1, ..., s_N\}$ is a set of $N$ states.
- $A = \{a_1, ..., a_k\}$ is a set of $k$ actions.
- $\pi : S \rightarrow A$ is a policy to decide which action to take at any state $s$. $a_t \sim \pi(\cdot|s_t)$. 
A (finite) MDP is a tuple \((S, \mathcal{A}, \{P_{sa}\}, \gamma, c(s, a))\), where

- \(\{P_{sa}\}\) are the state transition probabilities. \(s_{t+1} \sim P(\cdot|s_t, a_t)\).
- \(\gamma \in [0, 1)\) is the discount factor.
- \(c(s, a): S \times A \rightarrow \mathbb{R}\) is the cost function.

The goal of reinforcement learning is to find a policy \(\pi\) to minimize the following value function

\[
\mathbb{E}_{\pi}[c(s, a)] = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t c(s_t, a_t)\right]
\]  

(1)
Inverse reinforcement learning is used to solve the problems where the cost function is unknown.

- **Given**: 1) an expert policy $\pi_E$, 2) a set of trajectories sampled by executing $\pi_E$ in the environment; 3) a model of the physical environment.
- **Goal**: Find a cost function that describes observed behavior.
- IRL looks for a cost function $c \in \mathcal{C}$ that assigns low cost to the expert policy and high cost to other policies.
Imitation Learning

Maximum casual entropy IRL is adopted to find a cost function such that the expert performs better than all other policies, with the cost regularized by $\psi$:

$$\text{IRL}_\psi(\pi_E) = \arg \max_{c \in \mathcal{C}} -\psi(c) + \left( \min_{\pi \in \Pi} -H(\pi) + \mathbb{E}_\pi [c(s, a)] \right) - \mathbb{E}_{\pi_E} [c(s, a)]$$

(2)

where $H(\pi) = \mathbb{E}_\pi [-\log \pi(a|s)]$ is the policy regularizer. Then running RL on the output of IRL reveals the optimal policy

$$\text{RL}(c) = \arg \min_{\pi \in \Pi} -H(\pi) + \mathbb{E}_\pi [c(s, a)]$$

(3)
**Definition:** For a policy $\pi \in \Pi$, define its **occupancy measure** $\rho_\pi : S \times A \rightarrow \mathbb{R}$ as

$$
\rho_\pi(s, a) = \pi(a|s) \sum_{t=0}^{\infty} \gamma^t P(s_t = s|\pi)
$$

The occupancy measure can be interpreted as the distribution of state-action pairs with policy $\pi$. $\mathcal{D}$ is the set of valid occupancy measures. There is a one-to-one correspondence between $\Pi$ and $\mathcal{D}$.

**Proposition 3.1.** If $\rho \in \mathcal{D}$, then $\rho$ is the occupancy measure for $\pi_\rho(a|s) = \rho(s, a) / \sum_{a'} \rho(s, a')$, and $\pi_\rho$ is the only policy whose occupancy measure is $\rho$. 
The occupancy measure allows us to write

\[ \mathbb{E}_\pi[c(s, a)] = \sum_{s, a} \rho_\pi(s, a)c(s, a) \]  

(5)

**Lemma 3.1.** \( H(\pi) = \mathbb{E}_\pi[-\log \pi(a|s)] \) and \( \bar{H}(\rho) = -\sum_{s, a} \rho(s, a) \log(\rho(s, a))/\sum_{a'} \rho(s, a') \). Then \( \bar{H} \) is strictly concave, and for all \( \pi \in \Pi \) and \( \rho \in \mathcal{D} \), we have \( H(\pi) = \bar{H}(\rho_\pi) \) and \( \bar{H}(\rho) = H(\pi_\rho) \).

**Lemma 3.2.** \( L(\pi, c) = -H(\pi) + \mathbb{E}_\pi[c(s, a)] \) and \( \bar{L}(\rho, c) = -\bar{H}(\rho) + \sum_{s, a} \rho(s, a)c(s, a) \). Then for all \( c \in \mathcal{C} \), \( \pi \in \Pi \) and \( \rho \in \mathcal{D} \), we have \( L(\pi, c) = \bar{L}(\rho_\pi, c) \) and \( \bar{L}(\rho, c) = L(\pi_\rho, c) \).
Definition: For a function $f : \mathbb{R}^{S \times A} \to \mathbb{R}$, its convex conjugate $f^*$: $\mathbb{R}^{S \times A} \to \mathbb{R}$ is given by

$$f^*(x) = \sup_{y \in \mathbb{R}^{S \times A}} x^\top y - f(y)$$

(6)

Proposition 3.2.

$$\text{RL} \circ \text{IRL}_\psi(\pi_E) = \arg\min_{\pi \in \Pi} -H(\pi) + \psi^*(\rho_\pi - \rho_{\pi_E})$$

(7)

Proposition 3.2. tells us that $\psi$-regularized inverse reinforcement learning, implicitly, seeks a policy whose occupancy measure is close to the expert’s, as measured by the convex function $\psi^*$. 
Proposition 3.2.

Proof

Let \( \tilde{c} \in \text{IRL}_\psi(\pi_E) \), \( \tilde{\pi} \in \text{RL}(\tilde{c}) = \text{RL} \circ \text{IRL}_\psi(\pi_E) \) and

\[
\pi_A \in \arg\min_{\pi} -H(\pi) + \psi^*(\rho_\pi - \rho_{\pi_E}) \tag{8}
\]

\[
= \arg\min_{\pi} \max_c -H(\pi) - \psi(c) + \sum_{s,a} (\rho_\pi(s, a) - \rho_{\pi_E}(s, a))c(s, a) \tag{9}
\]

We want to show that \( \pi_A = \tilde{\pi} \), that is equivalent to show \( \rho_A = \tilde{\rho} \). Define

\[
\bar{L}(\rho, c) = -\bar{H}(\rho) - \psi(c) + \sum_{s,a} (\rho_\pi(s, a) - \rho_{\pi_E}(s, a))c(s, a) \tag{10}
\]

Due to convexity of \(-\bar{H}\) and \(\psi\), we can prove that \((\rho_A, \tilde{c})\) is a saddle point of \(\bar{L}\). Since \((\tilde{\rho}, \tilde{c})\) is a saddle point of \(\bar{L}\) and \(\bar{L}(\cdot, c)\) is strictly convex for all \(c\), we have

\[
\rho_A = \tilde{\rho} \iff \pi_A = \tilde{\pi} \tag{11}
\]
Proposition 3.2 suggests that various settings of $\psi$ lead to various imitation learning algorithms that directly solve the optimization problem given by Proposition 3.2. In this paper, three different regularizers are presented.

- Constant Regularizer
- Indicator Regularizer
- Generative Adversarial (GA) Regularizer
Constant Regularizer

**Corollary 3.2.1.** If $\psi$ is a constant function, $\tilde{c} \in IRL_\psi(\pi_E)$, and $\tilde{\pi} \in RL(\tilde{c})$, then $\rho_{\tilde{\pi}} = \rho_{\pi_E}$.

- If there were no cost regularization at all, then the recovered policy will exactly match the expert policy.
- However, such algorithm is not practically useful as it is intractable in large environment.
An apprenticeship learning algorithm finds a policy that performs better than the expert across $C$, by optimizing the objective

$$\min_{\pi} \max_{c \in C} \mathbb{E}_{\pi} \left[ c(s, a) \right] - \mathbb{E}_{\pi_E} \left[ c(s, a) \right]$$  \hspace{1cm} (12)

$C$ is typically restricted to convex sets given by linear combinations of basis functions $f_1, \ldots, f_d$. Abbeel and Ng use $C_{\text{linear}} = \{ \sum_i w_i f_i : \|w\|_2 \leq 1 \}$ and Syed et al. use $C_{\text{convex}} = \{ \sum_i w_i f_i : \sum_i w_i = 1, w_i \geq 0 \forall i \}$. 
With the indicator function \( \delta_c: \mathbb{R}^{S \times A} \rightarrow \mathbb{R} \), defined by

\[
\delta_c(c) = \begin{cases} 
0 & \text{if } c \in C; \\
+\infty & \text{otherwise.}
\end{cases}
\] (13)

Then the apprenticeship learning objective can be written as

\[
\max_{c \in C} \mathbb{E}_\pi [c(s, a)] - \mathbb{E}_{\pi_E} [c(s, a)] = \max_{c \in C} -\delta_c(c) + \sum_{s, a} (\rho_\pi(s, a) - \rho_{\pi_E}(s, a))c(s, a)
\] (14)

\[
= \delta^*_C (\rho_\pi - \rho_{\pi_E})
\]
Entropy-regularized apprenticeship learning is equivalent to performing RL following IRL with cost regularizer $\psi = \delta_C$.

- Apprenticeship learning with such $C$ can scale to large environment with policy function approximation.
- It forces the implicit IRL procedure to recover a cost function lying in $C$. If $C$ does not include a cost function that explains expert behavior well, however, it will fail to imitate.
Generative Adversarial (GA) Regularizer

$$\psi_{GA}(c) = \begin{cases} \mathbb{E}_{\pi_E}[g(c(s, a))] & \text{if } c < 0; \\ +\infty & \text{otherwise.} \end{cases}$$

(16)

where

$$g(x) = \begin{cases} -x - \log(1 - e^x) & \text{if } x < 0; \\ 0 & \text{otherwise.} \end{cases}$$

(17)

- $\psi_{GA}$ is an average over expert data, and therefore can adjust to arbitrary expert datasets.
- Unlike $\delta_C$ restricts cost functions in a small subspace, $\psi_{GA}$ allows for any cost function, as long as it is negative everywhere.
The choice of $\psi_{GA}$ is motivated by its convex conjugate

$$\psi_{GA}^*(\rho_\pi - \rho_{\pi_E}) = \max_{D \in (0,1)^{S \times A}} \mathbb{E}_{\pi_E}[\log(D(s, a))] + \mathbb{E}_\pi[\log(1 - D(s, a))]$$

(18)

Eq. (18) is the optimal negative log loss of distinguishing between $\pi$ and $\pi_E$. This optimal loss is the Jensen-Shannon divergence

$$D_{JS}(\rho_\pi, \rho_{\pi_E}) = D_{KL}(\rho_\pi \| (\rho_\pi + \rho_{\pi_E})/2) + D_{KL}(\rho_{\pi_E} \| (\rho_\pi + \rho_{\pi_E})/2)$$

(19)
Generative Adversarial Imitation Learning

\[
\min_{\pi} \psi^*_A(\rho_\pi - \rho_{\pi_E}) - \lambda H(\pi) 
\]  \hspace{1cm} (20)

\[ \iff \min_{\pi} \max_D \mathbb{E}_{\pi_E} [\log(D(s, a))] + \mathbb{E}_\pi [\log(1 - D(s, a))] - \lambda H(\pi) \] \hspace{1cm} (21)

\[ \iff \min_{\pi} D_{JS}(\rho_\pi, \rho_{\pi_E}) - \lambda H(\pi) \] \hspace{1cm} (22)

which finds a policy whose occupancy measure minimizes Jensen-Shannon divergence to the expert’s.
Generative adversarial networks

$$\min_G \max_D \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log (1 - D(G(z)))]$$  \hspace{1cm} (23)$$

The job of $D$ is to distinguish between the distribution of data generated by $G$ and the true data distribution.

- The learner’s occupancy measure $\rho_\pi$ is analogous to the data distribution generated by $G$.
- The expert’s occupancy measure $\rho_{\pi_E}$ is analogous to the true data distribution.
Generative Adversarial Imitation Learning

Algorithm

- Wish to find a saddle point \((\pi, D)\) of the expression

\[
\mathbb{E}_{\pi_E}[\log(D(s, a))] + \mathbb{E}_\pi[\log(1 - D(s, a))] - \lambda H(\pi) \tag{24}
\]

- \(\pi_\theta\) is a parameterized policy with weights \(\theta\), \(D_\omega\) is a discriminator with weights \(\omega\).

- Use an Adam gradient step on \(\omega\) to increase Eq. (24) with respect to \(D\).

- Use a TRPO step on \(\theta\) to decrease Eq. (24) with respect to \(\pi\).

- TRPO is a natural gradient step constrained to ensure that \(\pi_{\theta_{i+1}}\) does not stray too far from \(\pi_{\theta_i}\).
Algorithm 1 Generative adversarial imitation learning

1: **Input:** Expert trajectories $\tau_E \sim \pi_E$, initial policy and discriminator parameters $\theta_0, w_0$
2: **for** $i = 0, 1, 2, \ldots$ **do**
3: Sample trajectories $\tau_i \sim \pi_{\theta_i}$
4: Update the discriminator parameters from $w_i$ to $w_{i+1}$ with the gradient
   \[
   \mathbb{E}_{\tau_i}[\nabla_w \log(D_w(s, a))] + \mathbb{E}_{\tau_E}[\nabla_w \log(1 - D_w(s, a))]
   \]
5: Take a policy step from $\theta_i$ to $\theta_{i+1}$, using the TRPO rule with cost function $\log(D_{w_{i+1}}(s, a))$. Specifically, take a KL-constrained natural gradient step with
   \[
   \mathbb{E}_{\tau_i}[\nabla_\theta \log \pi_\theta(a|s)Q(s, a)] - \lambda \nabla_\theta H(\pi_\theta),
   \]
   where $Q(\bar{s}, \bar{a}) = \mathbb{E}_{\tau_i}[\log(D_{w_{i+1}}(s, a)) | s_0 = \bar{s}, a_0 = \bar{a}]$
6: **end for**
Algorithm 1 is tested against three baselines:

- **Behavioral cloning**: The policy is trained with supervised learning, using Adam with minibatches of 128 examples, until validation error stops decreasing.

- **Feature expectation matching (FEM)**: the algorithm of Ho et al. using the cost function class $C_{\text{linear}}$ of Abbeel and Ng.

- **Game-theoretic apprenticeship learning (GTAL)**: the algorithm of Ho et al. using the cost function class $C_{\text{convex}}$ of Syed et al.
Figure: Performance of learned policies. The y-axis is negative cost, scaled so that the expert achieves 1 and a random policy achieves 0.
Figure: Causal entropy regularization $\lambda$ on Reacher.