

Model-free Imitation Learning with Policy Optimization

Jun Gao

2019-02-25

Background - Reinforcement Learning

- ▶ **State space:** \mathcal{S} .
- ▶ **Action space:** \mathcal{A} .
- ▶ **Dynamics model:** $p(s'|s, a), p_0(s_0)$.
- ▶ **Policy:** $\pi(a|s)$.
- ▶ **Cost function:** $c : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$.
- ▶ **State action value:** $Q_{\pi}^c(s_t, a_t) = \mathbb{E}_{p_0, p, \pi} \left[\sum_{t'=t}^{\infty} \gamma^{t'-t} c(s_{t'}, a_{t'}) \right]$.
- ▶ **State value:** $V_{\pi}^c(s) = \mathbb{E}_{a \sim \pi(\cdot|s)} [Q_{\pi}^c(s, a)]$.
- ▶ **Advantage function:** $A_{\pi}^c(s, a) = Q_{\pi}^c(s, a) - V_{\pi}^c(s)$.

Background - Reinforcement Learning

- ▶ **State space:** \mathcal{S} .
- ▶ **Action space:** \mathcal{A} .
- ▶ **Dynamics model:** $p(s'|s, a), p_0(s_0)$.
- ▶ **Policy:** $\pi(a|s)$.
- ▶ **Cost function:** $c : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$.
- ▶ **State action value:** $Q_{\pi}^c(s_t, a_t) = \mathbb{E}_{p_0, p, \pi} \left[\sum_{t'=t}^{\infty} \gamma^{t'-t} c(s_{t'}, a_{t'}) \right]$.
- ▶ **State value:** $V_{\pi}^c(s) = \mathbb{E}_{a \sim \pi(\cdot|s)} [Q_{\pi}^c(s, a)]$.
- ▶ **Advantage function:** $A_{\pi}^c(s, a) = Q_{\pi}^c(s, a) - V_{\pi}^c(s)$.
- ▶ **Expected cost of a policy π :**

$$\eta^c(\pi) = \mathbb{E}_{s_0 \sim p_0} [V_{\pi}^c(s_0)] = \sum_{s, a} \rho_{\pi}(s, a) c(s, a), \quad (1)$$

where $\rho_{\pi}(s, a) = \pi(a|s) [\sum_{t=0}^{\infty} \gamma^t P_{p_0, \pi}(s_t = s)]$

- ▶ **Imitation Learning:** learning from demonstrations of expert.
- ▶ **Behavioral cloning:** Suffer the *cascading errors*, and *covariate shift*.
- ▶ **Inverse RL:** Learn a cost function that fits expert's policy π_E at each iteration, but extremely expensive.
- ▶ **This paper:** Learn a policy from expert trajectories, through a class of cost functions.

- ▶ **Idea:** find a policy that performs at least as well as the expert π_E on an unknown true cost function.

$$\eta^{c_{true}}(\pi) \leq \eta^{c_{true}}(\pi_E). \quad (2)$$

- ▶ **Assumption:** c_{true} lies in one cost function class \mathcal{C} .

- ▶ **Idea:** find a policy that performs at least as well as the expert π_E on an unknown true cost function.

$$\eta^{c_{true}}(\pi) \leq \eta^{c_{true}}(\pi_E). \quad (2)$$

- ▶ **Assumption:** c_{true} lies in one cost function class \mathcal{C} .
- ▶ **Relaxation:**

$$\eta^c(\pi) \leq \eta^c(\pi_E), \text{ for all } c \in \mathcal{C}. \quad (3)$$

$$\delta_{\mathcal{C}}(\pi, \pi_E) = \sup_{c \in \mathcal{C}} \eta^c(\pi) - \eta^c(\pi_E) \quad (4)$$

- ▶ **Objective:**

$$\min_{\pi} \delta_{\mathcal{C}}(\pi, \pi_E) \quad (5)$$

► **Objective:**

$$\min_{\pi} \delta_{\mathcal{C}}(\pi, \pi_E) \quad (6)$$

► **Two ingredients to go:**

1. maximization over cost function class \mathcal{C} .
2. minimization over policy π .

- **Gradient of δ w.r.t model's parameters θ :**

$$\nabla_{\theta} \delta_{\mathcal{C}}(\pi, \pi_E) = \nabla_{\theta} \eta^{c^*}(\pi_{\theta}), \quad (7)$$

where c^* is the cost function that achieves supremum.

- **Policy Gradient:**

$$\nabla_{\theta} \eta^{c^*}(\pi_{\theta}) = \mathbb{E}_{\rho_{\pi_{\theta}}} [\nabla_{\theta} \log \pi_{\theta}(a|s) Q_{\pi_{\theta}}^{c^*}(s, a)] \quad (8)$$

- **Algorithm:**

Algorithm 1 IM-REINFORCE

Input: Expert trajectories τ_E , initial policy parameters.

θ_0

for $i = 0, 1, 2, \dots$ **do**

Roll out trajectories $\tau \sim \pi_{\theta_i}$

Compute \hat{c} achieving the supremum in (10),

with expectations taken over τ and τ_E

Estimate the gradient $\nabla_{\theta} \eta^{\hat{c}}(\pi_{\theta})|_{\theta=\theta_i}$ (8) with τ

Use the gradient to take a step from θ_i to θ_{i+1}

end for

TRPO for apprenticeship learning

Background - Trust Region Policy Optimization (TRPO)

- ▶ performance of a new policy π in terms of the performance of current policy π_0 :

$$\eta(\pi) = \eta(\pi_0) + \mathbb{E}_{s \sim \rho_\pi} \mathbb{E}_{a \sim \pi(\cdot|s)} [A_{\pi_0}(s, a)]$$

- ▶ first order approximation of $\eta(\pi)$:

$$L(\pi) \triangleq \eta(\pi_0) + \mathbb{E}_{s \sim \rho_{\pi_0}} \mathbb{E}_{a \sim \pi(\cdot|s)} [A_{\pi_0}(s, a)]$$

- ▶ a surrogate loss function of $L(\pi)$:

$$M(\pi) \triangleq L(\pi) + \frac{2\epsilon\gamma}{(1-\gamma)^2} \max_s D_{KL}(\pi_0(\cdot|s) || \pi(\cdot|s)),$$

where $\epsilon = \max_{s,a} |A_{\pi_0}(s, a)|$, $\eta(\pi) \leq M(\pi)$.

- ▶ reformulate as a trust region constraint.

$$\min_{\pi} L(\pi), \text{ s.t. } \bar{D}_{KL}(\pi_0 || \pi) \leq \Delta,$$

where $\bar{D}_{KL}(\pi_0 || \pi) = \mathbb{E}_{s \sim \rho_{\pi_0}} D_{KL}(\pi_0(\cdot|s) || \pi(\cdot|s))$

TRPO for apprenticeship learning

- Define a surrogate function:

$$M^c(\pi) \triangleq L^c(\pi) + \frac{2\epsilon'\gamma}{(1-\gamma)^2} \max_s D_{KL}(\pi_0(\cdot|s) \parallel \pi(\cdot|s)),$$

where $\epsilon' = \frac{2C_{max}}{1-\gamma} \geq \sup_c \max_{s,a} |A_{\pi_0}(s, a)|$, C_{max} is the upper bound for all cost functions.

- With the same proof at TRPO, we could get:

$$\delta_{\mathcal{C}}(\pi, \pi_E) = \sup_{c \in \mathcal{C}} \eta^c(\pi) - \eta^c(\pi_E) \leq \sup_{c \in \mathcal{C}} M^c(\pi) - \eta^c(\pi_E).$$

- It's ready to formulate as TRPO:

$$M^c(\pi, \pi_E) = \sup_{c \in \mathcal{C}} (L^c(\pi) - \eta^c(\pi_E)) + \frac{2\epsilon'\gamma}{(1-\gamma)^2} \max_s D_{KL}(\pi_0(\cdot|s) \parallel \pi(\cdot|s))$$

TRPO for apprenticeship learning

- ▶ formulate as TRPO:

$$M^{\mathcal{C}}(\pi, \pi_E) = \sup_{c \in \mathcal{C}} (L^c(\pi) - \eta^c(\pi_E)) + \frac{2\epsilon'\gamma}{(1-\gamma)^2} \max_s D_{KL}(\pi_0(\cdot|s) \parallel \pi(\cdot|s))$$

- ▶ The problem becomes:

$$\min_{\pi} \sup_{c \in \mathcal{C}} (L^c(\pi) - \eta^c(\pi_E)), \text{ s.t. } \bar{D}_{KL}(\pi_0 \parallel \pi) \leq \Delta$$

TRPO for apprenticeship learning

- Simplify the objective:

$$f = \sup_{c \in \mathcal{C}} L^c(\pi) - \eta^c(\pi_E) \quad (9)$$

$$= \sup_{c \in \mathcal{C}} \mathbb{E}_{\rho_{\pi_0}}[c(s, a)] - \mathbb{E}_{\rho_{\pi_E}}[c(s, a)] + \mathbb{E}_{\rho_{\pi_0}} \mathbb{E}_{a \sim \pi}[A_{\pi_0}^c(s, a)] \quad (10)$$

$$= \sup_{c \in \mathcal{C}} \mathbb{E}_{\rho_{\pi_0}}[c(s, a)] - \mathbb{E}_{\rho_{\pi_E}}[c(s, a)] \quad (11)$$

$$+ \mathbb{E}_{\rho_{\pi_0}} \left[\frac{\pi_{\theta}(a|s)}{\pi(a|s)} (Q_{\pi_0}^c(s, a) - V_{\pi_0}^c(s)) \right] \quad (12)$$

$$= \sup_{c \in \mathcal{C}} \mathbb{E}_{\rho_{\pi_0}}[c(s, a)] - \mathbb{E}_{\rho_{\pi_E}}[c(s, a)] \quad (13)$$

$$+ \mathbb{E}_{\rho_{\pi_0}} \left[\left(\frac{\pi_{\theta}(a|s)}{\pi(a|s)} - 1 \right) Q_{\pi_0}^c(s, a) \right] \quad (14)$$

Algorithm 2 IM-TRPO

Input: Expert trajectories τ_E , initial policy params. θ_0 ,
trust region size Δ

for $i = 0, 1, 2, \dots$ **do**

Roll out trajectories $\tau \sim \pi_{\theta_i}$

Find $\pi_{\theta_{i+1}}$ minimizing Equation (23)

subject to $\overline{D}_{\text{KL}}(\pi_{\theta_i} \parallel \pi_{\theta_{i+1}}) \leq \Delta$,

with expectations taken over τ and τ_E (15)

end for

Finding cost functions

- **Assumption:** linear or convex.

$$\delta_{\mathcal{C}_{linear}}(\pi, \pi_E) = \sup_{\|w\|_2 \leq 1} w^T (\phi(\pi) - \phi(\pi_E)). \quad (15)$$

Finding cost functions

- **Assumption:** linear or convex.

$$\delta_{\mathcal{C}_{linear}}(\pi, \pi_E) = \sup_{\|w\|_2 \leq 1} w^T (\phi(\pi) - \phi(\pi_E)). \quad (15)$$

- closed form solution:

$$w \triangleq \frac{\phi(\pi) - \phi(\pi_E)}{\|\phi(\pi) - \phi(\pi_E)\|_2} \quad (16)$$

Finding cost functions

- **Assumption:** linear or convex.

$$\delta_{\mathcal{C}_{linear}}(\pi, \pi_E) = \sup_{\|w\|_2 \leq 1} w^T (\phi(\pi) - \phi(\pi_E)). \quad (15)$$

- closed form solution:

$$w \triangleq \frac{\phi(\pi) - \phi(\pi_E)}{\|\phi(\pi) - \phi(\pi_E)\|_2} \quad (16)$$

- for TRPO:

$$w \triangleq \frac{\phi(\pi_0) - \phi(\pi_E) + \psi(\pi_0)}{\|\phi(\pi_0) - \phi(\pi_E) + \psi(\pi_0)\|_2}, \quad (17)$$

where $\psi(\pi_0) = \mathbb{E}_{\rho_{\pi_0}} \left[\left(\frac{\pi_\theta(a|s)}{\pi(a|s)} - 1 \right) (\mathbb{E}_{\pi_0} [\sum_{t=0}^{\infty} \gamma^t \phi(s_t, a_t) | s_0, a_0]) \right]$

Compared to other apprenticeship learning

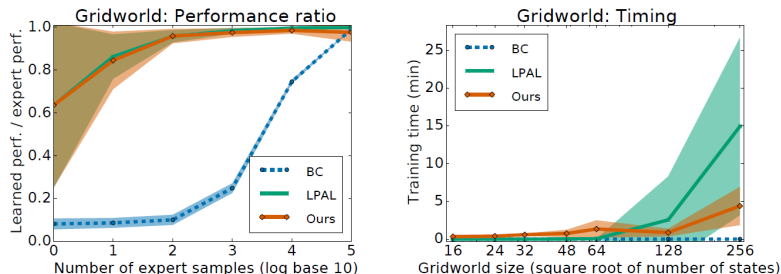


Figure 1. Left: Gridworld performance ratio across varying amounts of expert data. Right: Training time on increasing gridworld sizes. (*BC* stands for behavioral cloning.)

Compared to other IRL methods

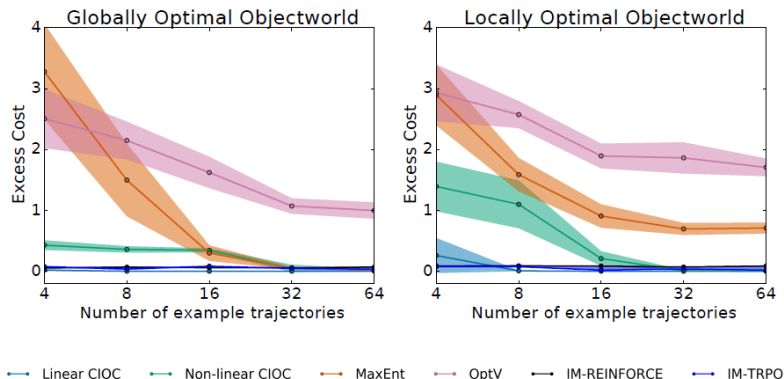


Figure 2. Excess cost for each algorithm for globally and locally optimal planar navigation examples, against variants of CIOC and other competing algorithms.

Influence of feature dimension

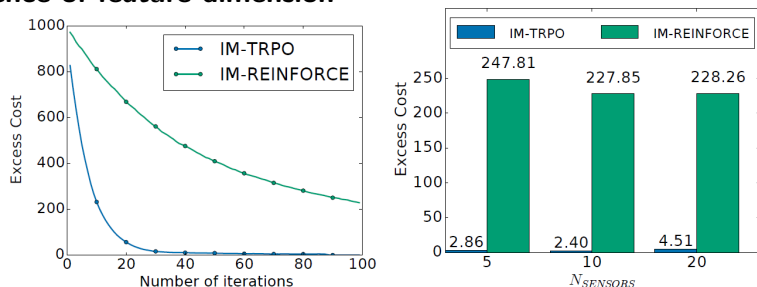


Figure 3. Left: Excess cost over time for one run of $N_{\text{sensors}} = 20$. Curves for other settings are similar. Right: Excess costs for learned policies on various sensor counts.

► Pros:

- blending state-of-the-art policy gradient algorithms for reinforcement learning.

► Cons

- linear (convex) cost function assumption, hard to scale to non-convex setting.