A note on Apprenticeship Learning via Inverse Reinforcement Learning

Author name Wei-Lun Lai
Department of Computer Science
National Chiao Tung University
lai30942.cs04@nctu.edu.tw

1 Introduction

In this paper, the author proposes using two inverse reinforcement learning algorithms (*max-margin method* and *projection method*) for aprrenticeship learning. Inverse reinforcement learning (IRL) is a special case of Reinforcement learning. To be more specific, IRL learns in Markov Decision Process to recover the unknown reward function. Besides, IRL is especially useful for some applications where reward function is hard to be defined.

On the other hand, apprenticeship learning is trying to learn from expert's demonstration, and the expert interacting with the environment can be viewed as trying to maximize a reward function. Combining all above, IRL can be used to recover the unknown reward function, then given the reward function we can use appropriate RL algorithms to obtain the policy which can perform as good as or near the expert's performance.

As for apprenticeship learning, there have been numbers of prior works brought for apprenticeship learning. Most of them learn the expert's behavior directly with supervised learning. In other words, they learn the mapping from states to actions.

However, there are quite a few challenges by simply mapping state to action via supervised learning. Since this method is applicable only to problems where the task is to mimic the expert's trajectory. Obviously, task like driving would not be a appropriate scenario to directly apply these methods. Meanwhile, with reward function recovered, task like driving will not cause any problem with author's method.

The contribution of this paper is applying inverse reinforcement learning to apprenticeship learning. Besides, methods proposed are able to solve problem which the prior works of supervised learning can not tackle. In my opinion, these methods do tackle the challenge of the prior works. However, the ability of recover the reward function correctly still might not good enough.

2 Problem Formulation

A Markov decision process is a tuple (S, A, T, γ, D, R)**, where S is state space, A is action space, T is state transition probabilities,** γ **is discount factor where** $\gamma \in [0, 1)$**, D is the distribution of the initial state, from which the start state** s_0 **is drawn; and R: S -> A is the reward function. In the following,** (S, A, T, γ, D) **denotes an MDP without an reward function. Assume reward function can be represented by linear combination of features as** $R^*(s) = w^* \cdot \phi(s)$**, where** $\phi : S \mapsto [0, 1]^k$ **is vector of features over states and** $w^* \in \mathbb{R}^k$

Preprint. Under review.

A policy π is a mapping from states to probability distributions over actions. The value of a policy π is

$$\begin{aligned} \mathbb{E}[V^{\pi}(s_0] &= \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R(s_t)|\pi] && (1) \\ &= \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t w \cdot \phi(s_t)|\pi] && (2) \\ &= w \cdot \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t \phi(s_t)|\pi] && (3) \end{aligned}$$

$\mu(\pi)$ is the expected discounted accmulated feature vector. Besides, feature expectations to be

$$\mu(\pi) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t \phi(s_t)|\pi] \in \mathbb{R}^k \quad (4)$$

Combining (3) and (4), the value of a policy can be rewritten as $\mathbb{E}[V^{\pi}(s_0] = w \cdot \mu(\pi)$ Given this equation, the value of a policy can be associated with feature expectation. Then, assume expert policy is π_E

Given a set of m trajectories $\{s_0^{(i)}, s_1^{(i)}, ...\}_{i=1}^m$ generated by the expert, we denote the empirical estimate for μ_E by

$$\hat{\mu}_E = \frac{1}{m} \sum_{i=1}^{m} \sum_{t=0}^{\infty} \gamma^t \phi\left(s_t^{(i)}\right) \quad (5)$$

There are several assumptions in the problem formulation. First, the RL algorithm used to find the policy is assumed to find the optimal policy for simplicity of exposition. Second, expert's π_E is assumed to be accessible. To put it another way, the expert's trajectories(state sequence) is observable.

The target of this optimization is to find a policy $\widetilde{\pi}$ such that $\|\mu(\widetilde{\pi}) - \mu_E\|_2 \leq \epsilon$

3 Theoretical Analysis

3.1 Max-margin method

In Max-margin method, the apprenticeship learning algorithm which tries to find the policy $\widetilde{\pi}$ is as follows:

1. Randomly pick some policy $\pi^{(0)}$, compute (or approximate via Monte Carlo) $\mu^{(0)} = \mu\left(\pi^{(0)}\right)$, and set i=1
2. Compute $t^{(i)} = \max_{w:\|w\|_2 \leq 1} \min_{j \in \{0..(i-1)\}} w^T \left(\mu_E - \mu^{(j)}\right)$, and let $w^{(i)}$ be the value of w that attains this maximum.
3. If $t^{(i)} \leq \epsilon$, then terminate.
4. Using the RL algorithm, compute the optimal policy $\pi^{(i)}$ for the MDP using rewards $R = \left(w^{(i)}\right)^T \phi$
5. Compute (or estimate) $\mu^{(i)} = \mu\left(\pi^{(i)}\right)$
6. Set $i = i + 1$, and go back to step 2

The maximization in step 2 can be equivalent as follows:

$$\begin{aligned} \max_{t,w} \quad & t && (6) \\ \text{s.t.} \quad & w^T \mu_E \geq w^T \mu^{(j)} + t, j = 0, \ldots, i-1 && (7) \\ & \|w\|_2 \leq 1 && (8) \end{aligned}$$

By (7), it is obvious to see that this algorithm is trying to find a reward function $R = w(i) \cdot \phi$ such that $E_{s_0 \sim D}\left[V^{\pi_E}(s_0)\right] \geq E_{s_0 \sim D}\left[V^{\pi^{(i)}}(s_0)\right] + t$. In other words, algorithm tries to find a reward on which the expert does better, by a "margin" of t, than any of the i policies we had found previously.

With the 2-norm constraint on w, this maximization cannot be solved as linear program but quadratic program. This optimization can also be viewed as finding the maximum margin hyper-plane separating two sets of points. Since we can associate the expert's feature expectations μ_E with label 1 and associate the feature expectations $\{\mu(\pi^{(i)}) : j = 0..(i-1)\}$ with label -1. Thus, the vector $w(i)$ is the unit vector orthogonal to the maximum margin separating hyper-plane.

3.2 Projection method

In projection method, there is no need to use QP solver. It just replace the step 2 in Max-margin method

1. Set $\bar{\mu}^{(i-1)} = \bar{\mu}^{(i-2)} + \frac{\left(\mu^{(i-1)} - \bar{\mu}^{(i-2)}\right)^T \left(\mu_E - \bar{\mu}^{(i-2)}\right)}{\left(\mu^{(i-1)} - \bar{\mu}^{(i-2)}\right)^T \left(\mu^{(i-1)} - \bar{\mu}^{(i-2)}\right)} \left(\mu^{(i-1)} - \bar{\mu}^{(i-2)}\right)$ (This computes the orthogonal projection of μ_E onto the line through $\bar{\mu}^{(i-2)} and \mu^{(i-1)}$.)
2. Set $w^{(i)} = \mu_E - \bar{\mu}^{(i-1)}$
3. Set $t^{(i)} = \left\|\mu_E - \bar{\mu}^{(i-1)}\right\|_2$

3.3 Convergence

All methods mentioned above are under the assumption that the algorithms will terminate with $t \leq \epsilon$ eventually. Nevertheless, it is not the case. We can know that it will converge with an upper-bound of iterations with theorem 1 which is as follow:

Theorem 1. Let $an MDP \backslash R, features \phi : S \mapsto [0,1]^k$ *and any* $\epsilon > 0$ *be given. Then the apprenticeship learning algorithm (both max-margin and projection versions will terminate with* $t^{(i)} \leq \epsilon$ *after at most*

$$n = O\left(\frac{k}{(1-\gamma)^2 \epsilon^2} \log \frac{k}{(1-\gamma)\epsilon}\right) \tag{9}$$

iterations.

In addition to upper-bound iteration needed, there is another important issue, sample trajectory. Sample trajectory would directly influence the result of the learning. By the following theorem, we know that at least how many trajectories we need to sample.

Theorem 2. *Let an MDP* $\backslash R, features \phi : S \mapsto [0,1]^k$ *and any* $\epsilon > 0, \delta > 0$ *be given. Suppose the apprenticeship learning algorithm (either max-margin or projection version) is run using an estimate* $\hat{\mu}_E$ *for* μ_E *obtained by m Monte Carlo samples. In order to ensure that with probability at least* $1 - \delta$ *the algorithm terminates after at most a number of iterations n given by (8), and outputs a policy o that for any true reward* $R^*(s) = w^{*T}\phi(s)$ $(\|w^*\|_1 \leq 1)$ *we have*

$$E\left[\sum_{t=0}^{\infty} \gamma^t R^*(s_t) \mid \tilde{\pi}\right] \geq E\left[\sum_{t=0}^{\infty} \gamma^t R^*(s_t) \mid \pi_E\right] - \epsilon \tag{10}$$

it suffices that

$m \geq \frac{2k}{(\epsilon(1-\gamma))^2} \log \frac{2k}{\delta}$

4 Conclusion

In this paper, author use inverse reinforcement learning to find the policy having performance comparable to or better than that of the expert, without known reward function. However, there are still some limitations. Methods introduced by author including Maximum-margin method and projection method can not deal with unknown dynamic. To be more specific, they can not recover the unknown reward function without knowing transition probability T. In addition, these methods might not work when reward function may be non-linear functions of features.

As for future research direction, the author mentioned about using the dual to the LP which is used solve Bellman's equations

Recent researches on IRL have been quite diverse. Maximum entropy IRL [1] is based on probability to recover reward function. Others like multi-agent IRL [2] and Guided cost learning [3] are also the extension of the basis IRL. To sum up, with all these IRL, there might be better solutions for apprenticeship learning.

References

[1] J.Andrew Bagnell Brian D. Ziebart, Andrew Maas and Anind K. Dey. Maximum entropy inverse reinforcement learning. 2008.

[2] Kshitij Judah Prasad Tadepalli Kristian Kersting Jude Shavlik Sriraam Natarajan, Gautam Kunapuli. Multi-agent inverse reinforcement learning. 2010.

[3] Pieter Abbeel Chelsea Finn, Sergey Levine. Guided cost learning: Deep inverse optimal control via policy optimization. 2016.