Statistical Techniques in Robotics (16-831, S22) Lecture #24 (Monday, April 21)

Inverse Reinforcement Learning

Lecturer: Kris Kitani *Scribes: Rishabh Pahuja, Sarthak Shetty*

1 Review

In the last few lectures, we studied about different reinforcement algorithms to find the optimum policy for the agent to follow. Some of the algorithms that we learnt were, value based iteration, policy based iteration, Q-learning, and so on. Even though reinforcement learning is theoretically effective, it takes a lot of time for the agent to learn a policy. For faster learning, we can learn from demonstrations (such as humans or other experts). This field of reinforcement learning is known as **Imitation Learning (IL)**. The agent learns the policy by demonstration from an expert. Imitation learning can be classified into three main types: 1. Passive IL, 2. Active IL, 3. Interactive IL. Table 1 gives the details about each kind of imitation learning.

Table 1: **Types of Imitation Learning**

	Passive IL	**Active IL**	**Interactive IL**
Demonstrations $\mathcal{D}$	yes	yes	optional
Environment $\mathcal{E}$	no	yes	yes
Oracle π	no	no	yes
Dynamics $\mathcal{T}$	no	optional	optional
Reward $\mathcal{R}$	no	optional	optional

1.1 Passive Imitation Learning

In passive IL, the agent has access to the demonstration given by the expert but does not carry out any actions, since it does not have acces to the environment. From the demonstrations it estimates the best policy which it then leverages in a given environment.

1.2 Active Imitation Learning

In active IL, the agent has access to not only to the demonstration given by the expert but also to the environment. The access to the environment allows the agent to test the policy and reward function being learnt in real time.

1.3 Interactive Imitation Learning

In interactive imitation learning, the agent can actively query the best policy or the "oracle" in order to improve the policy that it is learning. However, the agent may or may not have access to

the dynamics. Since the agent has access to the environment, it can estimate its performance and receive regular feedback from the oracle.

In this lecture we further look at inverse reinforcement learning which is a type of active imitation learning where we try to estimate the best policy and the reward function. The advantage of estimating the reward function is that there is better generalisation; here, we try to understand the intent of taking a specific action, rather than following a given policy. Once, the reward function is learnt, it can then be transferred to generate new trajectories in a given environment. In the previous lecture we also saw how to formulate the Linear Programming - Inverse Reinforcement Learning problem.

2 Lecture Summary

2.1 Variations of IRL Objectives

1. Q funtion: In this variation, Q function is used to find the optimal policy and the action corresponding to that state shall be the best action. This means that the Q value function shall be greatest for that particular action as compared to all other actions in that state:

$$\mathbf{Q}^{\pi}(\mathbf{s}, \mathbf{a}^{*}) \geq \mathbf{Q}^{\pi}(\mathbf{s}, \mathbf{a}) \tag{1}$$

2. Future payoff: In this variation, the linear approximation of the reward function is used to find the best action. This is particularly useful in large state spaces:

$$\sum_{\mathbf{s}'} \mathbf{P}(\mathbf{s}'|\mathbf{s}, \mathbf{a}^{*})\mathbf{V}^{\pi}(\mathbf{s}') \geq \sum_{\mathbf{s}'} \mathbf{P}(\mathbf{s}'|\mathbf{s}, \mathbf{a})\mathbf{V}^{\pi}(\mathbf{s}') \tag{2}$$

3. Value function: In this variant, the value function of optimal policy is compared with the value function of other policies:

$$\hat{V}^{\pi^{*}}(s) \geq \hat{V}^{\pi}(s) \tag{3}$$

For today's lecture, we will be focusing on **Policy Value**:

$$\mathbf{E}[\mathbf{V}^{\pi^{*}}(\mathbf{s})] \geq \mathbf{E}[\mathbf{V}^{\pi}(\mathbf{s})] \tag{4}$$

The short hand notation for the same can be used as:

$$\mathbf{V}^{\pi^{*}}(\mathbf{s}) \geq \mathbf{V}^{\pi}(\mathbf{s}) \tag{5}$$

2.2 Policy Value

$$\mathbf{E}[\mathbf{V}(\pi)] = \mathbf{E}_{\mathbf{p}}[\sum_{\mathbf{t=0}}^{\infty} \gamma^{t} r(s_t)] \tag{6}$$

(assume a linear reward function)

$$\mathbf{V}(\pi) = \mathbf{E_p}[\sum_{\mathbf{t=0}}^{\infty} \gamma^{\mathbf{t}} \theta.\phi(\mathbf{s_t})] \tag{7}$$

(pull out reward parameter vector)

$$\mathbf{V}(\pi) = \theta.\mathbf{E}[\sum_{\mathbf{t=0}}^{\infty} \gamma^{\mathbf{t}} \phi(\mathbf{s_t})] \tag{8}$$

$$\mathbf{V}(\pi) = \theta.\mu(\pi) \tag{9}$$

$\phi(s_t)$ indicates the probabilities of set of features, for e.g. people in a particular state in a parking lot on grass, near a car, parking area, etc. θ includes the parameters and $\mu(\pi)$ includes expected features calculated over all possibles trajectories over all possible states and that is one of the reasons why μ is represented as a function of π and not as a function of states.

The p in E_p includes the probability of a particular trajectory along with prior state distribution.

3 Objective Function for IRL

We refer to the [1] paper for much of this section.

Let there be two policies such that a trajectory is sampled from optimal policy and a trajectory is sampled from some other policy such that:

$$\zeta^* \sim \pi^* \ \ and \ \zeta \sim \pi \tag{10}$$

where π^* indicates the optimal policy. Therefore:

$$\mathbf{V}(\pi^*) \geq \mathbf{V}(\pi) \tag{11}$$

$$\theta.\mu(\pi^*) \geq \theta \cdot \mu(\pi) \tag{12}$$

(Since, we assume that the reward function is linear)

Lets introduce a margin variable called t, such that:

$$\theta^{\mathbf{T}}\mu(\pi^*) \geq \theta^{\mathbf{T}}\mu(\pi) + \mathbf{t} \ \ \forall\pi \tag{13}$$

Eq. 13 in this form has the following problems:

1. No bound on the reward parameters, so by scaling θ, the inequality can be made arbitrarily large.

2. All policies cannot be compared, therefore, it is difficult to find the optimal policy

These problems can be tackled by:

1. Regularizing parameters
2. Use reinforcement learning to find most competitive policies

3.1 Regularization

The Eq. 13 can be modified such that:

$$\theta^{\mathbf{T}}.\mu(\pi^*) \geq \theta^{\mathbf{T}}\mu(\pi) + \mathbf{t} \ \ \forall\pi \tag{14}$$

$$||\theta||_{\mathbf{2}} \leq \mathbf{1} \tag{15}$$

$$min_{\theta,t}\lambda||\theta||_2 - t \tag{16}$$

Such that

$$\theta^T(\mu(\pi^*) - \mu(\pi_n)) \geq t \ \ \forall\pi_n \in \Pi \tag{17}$$

where Π is a finite set of competitive policies

Equation 17. looks very similar to the equations of SVM (Support Vector Machine). The equations for soft SVM and Max-margin IRL (MM IRL) side to side can be seen as:

3.1.1 Soft SVM

$$min_{w,\xi}||w||^2 + C\sum_i \xi_i \tag{18}$$

Such that:

$$y_i(w^T x_i + b) \geq 1 - \xi_i \tag{19}$$

3.2 Max Margin Inverse Reinforcement Learning

$$\max_\theta \ \ t \tag{20}$$

Such that:

$$\theta^T\mu(\pi^*) \geq \theta^T\mu(\pi) + t \ \ \forall\pi \tag{21}$$

$$||\theta||_2 \leq 1 \tag{22}$$

Converting equations to 'SVM-like':

$$max_\theta t \tag{23}$$

Such that:

$$\theta^T \mu(\pi^*) \geq \theta^T \mu(\pi) + t \forall \pi \tag{24}$$

$$||\theta||_2 \leq 1 \tag{25}$$

$$min_\theta \lambda ||\theta||_2 + \sum_i \xi_i \tag{26}$$

Such that:

$$\theta^T (\mu(\pi^*) - \mu(\pi_i)) \geq (1 - \xi_i) \ \ \forall i \tag{27}$$

(Replacing t_i by (1- ξ_i)

We borrow the math and algorithms from [2] for this section.

Now, we can put together the function of the Max Margin Inverse RL problem as follow:

Algorithm 1 function MaxMarginIRL $\mu(\pi^*)$

1: **for** $t = 1, \cdots, T$ **do**
2: $\hat{\pi} = argmax_\pi \theta^T \mu(\pi)$
3: $\mu(\hat{\pi}_i) = \mathbb{E}_{p_0, \tau, \hat{\pi}}[\sum_{t=0}^{\infty} \gamma^t \phi(s_t)]$
4: $\theta = QuadProg(\lambda, \mu(\pi^*), \{\mu(\hat{\pi_n})\}_{n=1}^i)$
5: **end for** Return θ

In Algo. 1, line 3 is of the ways to write the reinforcement learning objective, with the assumption to that the reward function is linearized. Line 4, is the expected feature counts for the competitive policy. Line 5 can be an SVM or even subsituted with a gradient descent algorithm. $\mu(\pi^*)$ is computed from the expert trajectories, whereas $\mu(\hat{\pi_n})$ is computed from the competitive policies.

3.3 Structured Large Margin Criteria

Recall that in an earlier lecture we prove that the Policy Value is linear in features. Here we quickly go through some of the math from that proof:

$$V(\pi) = \mathbb{E}_p[\sum_{t=0}^{\infty} \gamma^t r(s_t)] \tag{28}$$

We pull out the reward parameter vector:

$$V(\pi) = \mathbb{E}_p[\sum_{t=0}^{\infty} \gamma^t \theta \cdot \phi(s_t)] \tag{29}$$

$$V(\pi) = \theta \cdot \mathbb{E}_p[\sum_{t=0}^{\infty} \gamma^t \cdot \phi(s_t)] \tag{30}$$

$$V(\pi) = \theta \cdot \mu(\pi) \tag{31}$$

Let's now assume that we have trajectories drawn from some ζ^* trajectories that are being sampled from an expert policy denoted by π^*. We also have a regular policy π from which we are sampling trajectories denoted by ζ. We can represent the value function from these two policies as:

$$V(\pi^*) \geq V(\pi) \tag{32}$$

From the proof derived in Eq. 31, we can write the value function in terms of parameters and expected policy features as:

$$\theta \cdot \mu(\pi^*) \geq \theta \cdot \mu(\pi) \tag{33}$$

We can now re-write the optimization problem that we saw earlier as:

$$min_{\theta,\xi} \;\; \lambda||\theta||_2^2 + \sum_i \xi_i \tag{34}$$

such that,

$$\theta^T(\mu(\pi^*) - \mu(\pi_i)) \geq (1 - \xi_i) \;\; \forall i \tag{35}$$

Here ξ_i represent the slack, whereas the 1 is the margin, and is derived from convention.

We now ask if this 1 changes if the properties of the policy are changed, and if we can encode additional structure into the optimization problem if we can change this 1.

We now change this margin 1 into a function that takes into consideration the distance between the expert policy and the policy that we are using right now. We call this function a 'variable margin' and represent it as:

$$l(\pi^*, \pi_i) \tag{36}$$

Substituting this variable margin function back into our Eq. 35 we now get:

$$\theta^T(\mu(\pi^*) - \mu(\pi_i)) \geq (l((\pi^*, \pi_i) - \xi_i) \;\; \forall i \tag{37}$$

The intuition behind introducing this criteria is:

1. In the case of a discrete policy (for example, a policy operating in grid world), this function will look at the number of disagreements between the expert policy and the policy that we are evaluating.
2. If this policy is continuous then we look at the physical distance between the trajectories that result from the two policies.

Therefore, with this criteria, we are injecting some additional inductive biases and structure into the optimization problem.

Some notation before we move to the next section:

1. **Feature Count** is represented as $\mu \in \mathbb{R}^N$. For example, we can represent the total counts resulting from a policy as $\mu(\pi)$ or the total counts resulting from a trajectory as $\mu(\zeta)$.
2. We represent the **Feature Map** as $\mathbb{F} \in \mathbb{R}^{Nx|S||A|}$. This is a matrix that holds all the state features for all possible state-action pairs.
3. **Occupancy Measure** is represented as $\eta \in \mathbb{R}^{|S||A|}$. It represents the 'visitation count' or the number times a given state-action pair is invoked during a given episode, while a particular policy is being followed.

We can now establish a relationship between the cumulative feature count μ and the occupancy measure η as:

$$\mu = F\eta \tag{38}$$

We now modify the objective function that we derived in Eq. 37 and receive:

$$min_\theta \ \frac{1}{2}||\theta||^2 + \lambda \sum_d \{(\theta^T F_d + l_d^T)\eta - \theta^T F_d \eta_d\} \tag{39}$$

In Eq. 39. θ are the reward parameters. The first term represents regularization. We are summing over d expert demonstrations. The term $\theta^T F_d \eta_d$ represents the value function of an expert demonstrator. The first term inside the summation represents a policy as well. Here, η_d^* represents the occupancy vector contains the distribution of the sate-action pairs, that we receive after running the optimal policy (which we receive by running RL).

This η_d^* gets multiplied with $\theta^T F_d$. We also have the variable margin l_d^T. We make the assumption here that this l_d^T has a linear relationship with the occupancy η^*.

The expression in Eq. 39. is very similar to the SVM hinge loss, with a regularization term and a linear constraint. The SVM hinge loss was represented as:

$$min_w \ (\frac{\lambda}{2}||w||^2 + \frac{1}{M}\sum_{m=1}^{M} max\{0, 1 - y_m w^T x_m\}) \tag{40}$$

We solve this Eq. 40 using online sub-gradient descent:

$$g = \frac{\partial L}{\partial \theta} = \theta + \frac{\lambda}{D} \sum_d \beta_d F_d (\eta_d^* - \eta_d) \tag{41}$$

4 Conclusion

In this lecture, we explored some key-concepts in Inverse Reinforcement Learning, specifically in the domain of imitation learning. We saw how we can frame Policy-Value Objective Function as a max margin classification, which can be solved as a online gradient descent problem, by the introduction of a quadratic regularization term on the reward parameters and a loss term. We also showed how such problems can also be solved using quadratic programming. In the lecture, we also saw how the max-margin planning was one of the first such algorithm to be applied to real world robots.

Note: We referred to the previous years' scribe notes while preparing this document, including [6] and [5].

5 Appendix

We went through a few papers as well in order to better understand how imitation learning and learning from demonstrations actually works. We are linking a few of them here along with a one-line description that readers might find helpful.

1. **Robotic Telekinesis: Learning a Robotic Hand Imitator by Watching Humans on YouTube**, Sivakumar et al. (R:SS 2022) - In this paper the authors developed a system where a human can simply teleoperate a robot by demonstrating the actions with their own hands. The system learns to do so by analyzing videos of human hand videos. [7]
2. **A System for General In-Hand Object Re-Orientation**, Chen et al. (CoRL 2021) - In this paper, the authors incorporate teacher-student learning with a gravity curriculum to learn complex in-hand reorientation of common household objects[3]
3. **Replacing Rewards with Examples: Example-Based Policy Search via Recursive Classification**, Eysenbach et al. (ICRA 2021) - This paper doesn't deal with learning from demonstrations exactly, but is a reinforcement learning problem where the model frames reward functions as successful examples of the task that the agent is trying to learn. [4]

References

[1] P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, page 1, 2004.

[2] J. A. Bagnell, N. Ratliff, and M. Zinkevich. Maximum margin planning. In *Proceedings of the International Conference on Machine Learning (ICML)*. Citeseer, 2006.

[3] T. Chen, J. Xu, and P. Agrawal. A system for general in-hand object re-orientation. In *Conference on Robot Learning*, pages 297–307. PMLR, 2022.

[4] B. Eysenbach, S. Levine, and R. R. Salakhutdinov. Replacing rewards with examples: Example-based policy search via recursive classification. *Advances in Neural Information Processing Systems*, 34, 2021.

[5] Z. Kai. Lecture 22. https://github.com/Xingyu-Lin/16831-spring-2021/blob/master/scribes/lecture_22_A, 2021.

[6] I. Navarro and P. Zach. Lecture 23a. https://github.com/Xingyu-Lin/16831-spring-2021/blob/master/scribes/lecture_23_A, 2021.

[7] A. Sivakumar, K. Shaw, and D. Pathak. Robotic telekinesis: Learning a robotic hand imitator by watching humans on youtube. *arXiv preprint arXiv:2202.10448*, 2022.