Review of Inverse Reinforcement Learning frameworks for Imitation Learning

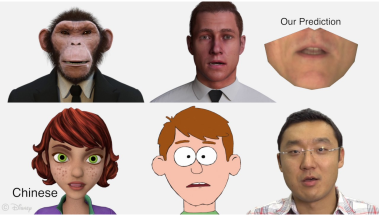
Josselin Bonnevie
Shahine Bouabid

MVA

Outline

Introduction

Bojarski et al. [2]

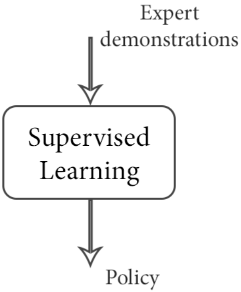


Taylor et al. [5]

$\boxed{\textbf{Given demonstrations} \longrightarrow \textbf{Find followed policy}}$

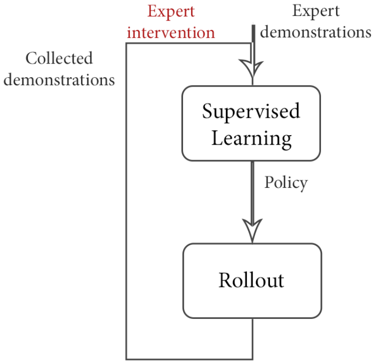
Given demonstrations ⟶ **Find** followed policy

Introduction



- Poor behavior recovery if step deviation
- Quadratic loss in task horizon

Introduction



- Linear loss in task horizon
- Requires interactive feedback

Idea : Use reward learning as a mean for policy derivation

Problem Setup

Framework :

- MDP : $\mathcal{M} = (S, A, P, R^*, \gamma, I)$ endowed with expert policy π^*
- Energy : $\mathcal{E}(\pi, R) = \mathbb{E}_{s \sim I}[V^\pi(s)]$

Given :

- $\mathcal{M} \backslash R^*$
- Environment simulation
- Demonstrations $\mathcal{D} = \{\zeta_i^*\}_i$ where $\zeta_i^* \sim \pi^*$
- Energy evaluation

Objective :

$$\boxed{\text{Find } R \text{ s.t. } \pi^* \in \arg\max_{\pi} \mathcal{E}(\pi, R)}$$

Approximation : Leverage $\mathcal{D} \rightarrow$ Find R s.t. $\pi_{\mathcal{D}} \in \arg\max_{\pi} \mathcal{E}(\pi, R)$

Problem Setup

Environment feature functions : $\phi = (\phi_j)$, $\phi_j : S \times A \to \mathbb{R}$

- Model environment facets
- Introduce prior knowledge

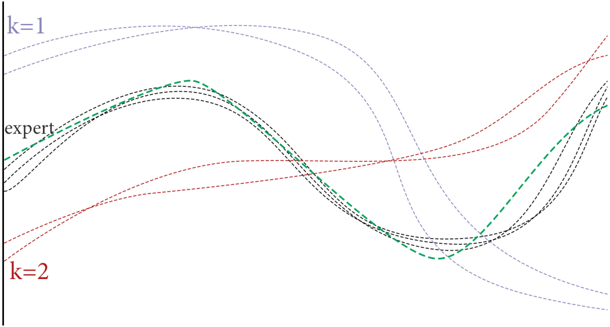
$\to$ *Linear reward parametrization* : $R = \theta^\top \phi$

Expected feature count :

$$\mu(\pi) = \mathbb{E}\left[\sum_{t \geq 0} \gamma^t \phi(s_t, a_t) \mid \pi\right] \Rightarrow \mathcal{E}(\pi, \theta) = \theta^\top \mu(\pi)$$

A Linear Programming Approach

Idea : Favor reward functions making deviation wrt optimal policy costly



Given $\{\pi_1, \ldots, \pi_k\}$ and linear penalization h, maximize :

$$\sum_{i=1}^{k} h(\mathcal{E}(\pi_{\mathcal{D}}, \theta) - \mathcal{E}(\pi_i, \theta)), \quad |\theta_j| \leq 1 \, \forall j \qquad (\mathrm{LP}_k)$$

A Linear Programming Approach

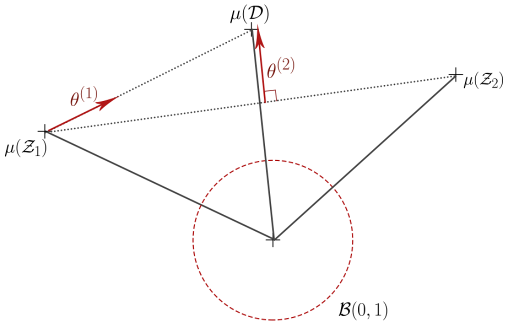
Algorithm 1 Ng and Russell [3]

Initialize $k \leftarrow 1$; θ randomly ; $\mathcal{E}(\pi_{\mathcal{D}}, \theta) \leftarrow \theta^\top \mu(\mathcal{D})$
Initialize base case policy π_1 and collect trajectories $\mathcal{Z}_1$
Initialize $\mathcal{E}(\pi_1, \theta) \leftarrow \theta^\top \mu(\mathcal{Z}_1)$
while Termination condition not met **do**
    $\theta \leftarrow$ solve (LP$_k$)
    Derive $\pi_{k+1} = \arg\max_{\pi} \mathcal{E}(\pi, \theta)$
    Roll out to collect trajectories $\mathcal{Z}_{k+1}$
    Update $\mathcal{E}(\pi_{\mathcal{D}}, \theta) \leftarrow \theta^\top \mu(\mathcal{D})$ and $\mathcal{E}(\pi_j, \theta) \leftarrow \theta^\top \mu(\mathcal{Z}_j) \forall j \in \{1, \dots, k\}$
    $k \leftarrow k + 1$
end while
return π_k

$\rightarrow$ Simple, intuitive but no convergence guarantee

A Quadratic Programming approach

Idea : Move problem to feature expectation space : $\mathcal{E}(\pi, \theta) = \theta^\top \mu(\pi)$ with constraint $\|\theta\|_2 \leq 1$



$$\max_{\eta\,\theta} \quad \eta$$
$$\text{s.t.} \quad \theta^\top \mu(\mathcal{D}) - \eta \geq \theta^\top \mu(\mathcal{Z}_i) \;\; \forall i \in [\![1, k]\!] \text{ and } \|\theta\|_2 \leq 1 \qquad (\text{SVM}_k)$$

A Quadratic Programming approach

Algorithm 2 Abbeel and Ng [1]

1: Initialize $\eta \leftarrow 0$; $k \leftarrow 1$; θ randomly
2: Initialize base case policy π_1 and collect trajectories $\mathcal{Z}_1$
3: Compute $\mu(\mathcal{D})$ and $\mu(\mathcal{Z}_1)$
4: **while** $\eta > \varepsilon$ **do**
5: $\quad$ η, $\theta \leftarrow$ solve (SVM$_k$)
6: $\quad$ Derive $\pi_{k+1} = \arg\max_{\pi} \mathcal{E}(\pi, \theta)$ under R
7: $\quad$ Roll out to collect trajectories $\mathcal{Z}_{k+1}$ and compute $\mu(Z_{k+1})$
8: $\quad$ $k \leftarrow k + 1$
9: **end while**
10: **return** π_k

A Quadratic Programming approach

Theorem

Let $\varepsilon > 0$, $\delta \in]0, 1[$, d the dimensionality of θ and m the number of Monte Carlo samples used.

• If $m \geq \frac{2d}{(1-\gamma)^2\varepsilon^2} \log \frac{2d}{\delta}$ and $\|\theta\|_2 \leq 1$

• Then $\exists K = \mathcal{O}\left(\frac{d}{(1-\gamma)^2\varepsilon^2} \log \frac{d}{(1-\gamma)\varepsilon}\right)$ s.t. $|\mathcal{E}(\pi^*, \theta) - \mathcal{E}(\pi_K, \theta)| \leq \varepsilon$ with probability $1 - \delta$

Pros :

- Convergence in a finite number of steps
- Control over solution's accuracy

Cons :

- Strong assumption : existence of a near-optimal policy
- Lack of generalization for start/goal variations

Max-margin Planning

Idea :

- Map each demonstration ζ_i^* to a different MDP endowed with its own expert policy π_i^* and set of feature functions
- But unique weight vector θ
- Introduce for each MDP an occupancy loss vector $\ell_i \to$ weights in closeness to π_i^*

$$\begin{aligned} \min_{\eta\,\theta} \quad & \frac{\lambda}{2}\|\theta\|_2^2 + \frac{\kappa}{n}\sum_{i=1}^{n} \beta_i \eta_i^q \\ \text{s.t.} \quad & \theta^\top \mu(\zeta_i^*) + \eta_i \geq \max_{\zeta} \theta^\top \mu(\zeta) + \ell_i^\top \rho^\zeta \ \ \forall i \end{aligned} \tag{MMP_k}$$

Number of constraints $\propto |\mathcal{D}| \prod_i |S_i \times A_i| \to$ subgradient resolution

Max-margin Planning

Algorithm 3 Ratliff et al. [4]

1: Initialize hyperparameters, learning rate α_t, horizon T, $t \leftarrow 1$, $\theta \leftarrow 0$
2: **while** $t \leq T$ **do**
3:     **for** i in $1 \dots n$ **do**
4:         Update $R_i \leftarrow \theta^\top \phi^{(i)}$
5:         Derive $\hat{\pi}_i^* = \arg\max_\pi \mathcal{E}(\pi, R_i) + \mathcal{L}_i(\pi)$
6:     **end for**
7:     Update $\theta \leftarrow \theta - \alpha_t \cdot \text{subgradient}(\hat{\pi}_1^*, \dots, \hat{\pi}_n^*)$ and increment t
8: **end while**
9: **return** θ

Great :

- Weight vector prone to extrapolate to new feature maps
- Convergence guaranteed in batch and online settings (resp. linear and sublinear)

Feature and occupancy matching

Expected feature count

$$\mu(\pi) = \mathbb{E}\left[\sum_{t\geq 0} \gamma^t \phi(s_t, a_t) \mid \pi\right] \Rightarrow \mathcal{E}(\pi, \theta) = \theta^\top \mu(\pi)$$

Occupancy measure

$$\rho^\pi(s, a) = \mathbb{E}_\pi\left[\sum_{t\geq 0} \gamma^t \mathbf{1}_{(s_t, a_t)=(s,a)}\right] \Rightarrow \mathcal{E}(\pi, \theta) = \sum_{(s,a)} \rho^\pi(s, a) R(s, a) = R^\top \rho^\pi$$

Idea : Find policy with same performance for whatever the reward

Maximum Entropy Principle - Deterministic MDP

Many policies can match the the expected features of the expert.

Entropy Maximization on the trajectory distribution

$$\max_{p_\pi} H(p_\pi) \quad \text{s.t.} \quad \mu(p_\pi) = \hat{\mu}(\mathcal{D})$$

Maximum likelihood in the exponential family

$$\hat{p}(\zeta) = \frac{e^{\theta^\top \mu(\zeta)}}{Z(\theta)} \quad \text{with } Z(\theta) = \int e^{\theta^\top \mu(\zeta)} d\zeta$$

Gradient

$$\nabla_\theta L(\theta) = \hat{\mu}(\mathcal{D}) - \mu(\pi_\theta) = \hat{\mu}(\mathcal{D}) - \sum_{(s,a) \in S \times A} \rho^\theta(s,a)\phi(s,a)$$

Maximum Causal Entropy for non Deterministic MDPs:

Causal Entropy (γ discounted):

$$\tilde{H}^{\gamma}(\pi) = \mathbb{E}\left[-\sum_{t\geq 0}\gamma^{t}\log\pi(a_t \mid s_t)\right]$$

The associated IRL problem

$$\max_{\pi}\tilde{H}^{\gamma}(\pi) \ \text{ s.t. } \ \mu(\pi) = \hat{\mu}(\mathcal{D})$$

Soft Bellman Equations:

$$\hat{\pi}(a \mid s) = \exp\left(Q_{\hat{\theta}}^{soft}(s,a) - V_{\hat{\theta}}^{soft}(s)\right) \tag{1}$$

Where

$$\begin{aligned} Q_{\hat{\theta}}^{soft}(s,a) &= \theta^{\top}\phi(s,a) + \beta\sum_{s'\in\mathcal{S}}P(s' \mid s,a)V_{\hat{\theta}}^{soft}(s') \\ V_{\hat{\theta}}^{soft}(s) &= \operatorname*{softmax}_{a\in A} Q_{\hat{\theta}}^{soft}(s,a) = \log\left(\sum_{a\in A}\exp(Q_{\hat{\theta}}^{soft}(s,a))\right) \end{aligned} \tag{2}$$

An other vision of MaxEnt IRL :

Maximum Entropy and Maximum Likelihood are dual problems :

$$\begin{aligned}\hat{\pi} &= \underset{\pi}{\operatorname{argmax}}\ \hat{H}^{\gamma}(\pi) + \hat{\theta}^{\top}\left(\mu(\pi) - \hat{\mu}(\mathcal{D})\right) \\ &= \underset{\pi}{\operatorname{argmax}}\ \hat{H}^{\gamma}(\pi) + \mathcal{E}(\pi, \hat{\theta})\end{aligned}$$

Maximum Entropy IRL can somehow be seen as a paradigm with a model on the agent's behaviour.

- We assume that for some R^* :

$$\pi^* \in \arg\max_{\pi} \hat{H}^{\gamma}(\pi) + \mathcal{E}(\pi, R^*)$$

- We want to find $\hat{R}$ such that

$$\pi^* \in \arg\max_{\pi} \hat{H}^{\gamma}(\pi) + \mathcal{E}(\pi, \hat{R})$$

Maximum Causal Entropy - Guarantees

Trajectories with equal reward have equal probabilities

Guarantees :

$$\hat{\pi} \in \arg\min_{\pi} \sup_{\tilde{\pi}} \mathbb{E}_{\tilde{\pi}} \left[\sum_{t \geq 0} -\beta^t \log \pi(a_t \mid s_t) \right]$$

Resolution

Gradient of the demonstration's likelihood:

$$\nabla_\theta L(\theta) = \hat{\mu}(\mathcal{D}) - \mu(\pi_\theta) = \hat{\mu}(\mathcal{D}) - \sum_{(s,a)\in S\times A} \rho^\theta(s,a)\phi(s,a)$$

Solving the MDP

- Soft Value Iteration

$$T_\theta^{\text{soft}}(V)(s) = \underset{a}{\text{softmax}}\left(\theta^\top f(s,a) + \sum_{s'\in\mathcal{S}} P(s' \mid s,a)V(s')\right)$$

- Soft Q-learning :

$$Q_\theta^{soft}(s_t,a_t) \leftarrow Q_\theta^{soft}(s_t,a_t) +$$
$$\eta(t)\left[\theta^\top f(s_t,a_t) + \gamma \underset{a_{t+1}}{\text{softmax}} Q_\theta^{soft}(s_{t+1},a_t) - Q_\theta^{soft}(s_t,a_t)\right]$$

Estimating occupancy measures

- Dynamic programming or Monte-Carlo methods

Deep Inverse Reinforcement Learning

- Feature engineering : hard and painful
- Occupancy measure matching : computing a reward for each action-state pair

Middle ground :

$$R(s, a) = R_\theta(\mathbf{f}(s_t, a_t))$$

- Back-propagation of the likelihood's gradient to the network's parameter
- A way to include expert's knowledge

Generative Adversarial Imitation Learning

- Directly Recovering a policy
- No RL in a loop
- Using Neural Networks

Causal Entropy Inverse Reinforcement Learning Problem

$$\min_{R \in \mathcal{C}} \left(\max_{\pi \in \Pi} \hat{H}^{\gamma}(\pi) + \mathbb{E}_{\pi}[R(s,a)] \right) - \mathbb{E}_{\pi^*}[R(s,a)]$$

But we don't want exact occupancy measure matching

$$\min_{R \in \mathcal{C}} \left(\max_{\pi \in \Pi} \hat{H}^{\gamma}(\pi) + \mathbb{E}_{\pi}[R(s,a)] \right) - \mathbb{E}_{\pi^*}[R(s,a)] \underbrace{+ \psi(R)}_{\text{Regularization}}$$

Generative Adversarial Imitation Learning

$$RL \circ IRL_\psi(\pi^*) = \arg\max_{\pi} \hat{H}^\gamma(\pi) - \psi^*(\rho^\pi - \rho^{\pi^*})$$

By taking:

$$\psi_{GA}(R) = \begin{cases} \mathbb{E}_{\pi^*}[g(R(s,a))] & \text{if } R > 0 \\ +\infty & \text{otherwise} \end{cases}$$

$$\text{With} \quad g(R) = R - \log(1 - e^{-R}) \text{ for all } R > 0$$

We get

$$\begin{aligned} \psi^*_{GA}(\rho^\pi - \rho^{\pi^*}) = & \max_{D \in (0,1)^{S \times A}} \mathbb{E}_\pi[\log D(s,a)] + \mathbb{E}_{\pi^*}[\log(1 - D(s,a))] \\ = & D_{JS}(\pi, \pi^*) + cst \end{aligned}$$

Generative Adversarial Imitation Learning

Final formulation of Generative Adversarial Imitation Learning

$$\text{Find } \hat{\pi} \in \arg\min_{\pi} D_{\mathrm{JS}}(\rho^{\pi}, \rho^{\pi^*}) - \lambda \tilde{H}^{\gamma}(\pi)$$

Or equivalently,

$$\hat{\pi} \in \arg\min_{\pi} \max_{D \in (0,1)^{\mathcal{S}\times\mathcal{A}}} \mathbb{E}_{\pi}[\log(D(s,a)] + \mathbb{E}_{\pi^*}[\log(1 - D(s,a)] - \lambda \tilde{H}^{\gamma}(\pi)$$

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*, ICML '04, pages 1–, New York, NY, USA, 2004. ACM. ISBN 1-58113-838-5. doi: 10.1145/1015330.1015430. URL http://doi.acm.org/10.1145/1015330.1015430.

[2] M. Bojarski, D. D. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba. End to end learning for self-driving cars. *CoRR*, abs/1604.07316, 2016. URL http://arxiv.org/abs/1604.07316.

[3] A. Y. Ng and S. J. Russell. Algorithms for inverse reinforcement learning. In *Proceedings of the Seventeenth International Conference on Machine Learning*, ICML '00, pages 663–670, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc. ISBN 1-55860-707-2. URL http://dl.acm.org/citation.cfm?id=645529.657801.

[4] N. D. Ratliff, J. A. Bagnell, and M. A. Zinkevich. Maximum margin planning. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pages 729–736, New York, NY, USA, 2006. ACM. ISBN 1-59593-383-2. doi: 10.1145/1143844.1143936. URL http://doi.acm.org/10.1145/1143844.1143936.

[5] S. Taylor, T. Kim, Y. Yue, M. Mahler, J. Krahe, A. Garcia Rodriguez, J. Hodgins, and I. Matthews. A deep learning approach for generalized speech animation. *ACM Transactions on Graphics*, 36:1–11, 07 2017. doi: 10.1145/3072959.3073699.