RL 15: Apprenticeship Learning and Inverse RL

Michael Herrmann

University of Edinburgh, School of Informatics

10/03/2015

Reminder: Rewards and Values in RL

- Reward is often modelled as a function of previous state, most recent action and state reached as a consequence of this action:

  $$R\left(s_t, a_t, s_{t+1}\right)$$

  R may depend also on earlier states and actions: Suppose an action a_t was executed in state s_t and contributed to the reward $r_{t+\tau}$ that was received τ time steps later. The algorithm (if it runs correctly) represents this reward in the value function of state s_t.
- Vice versa, if at state $s_{t+\tau}$ a reward is received, then the learning rule will backpropagate this information towards s_t (for an appropriate value of the discount factor $\gamma < 1$). The value of s_t is (among others) due to the fact that an action a_t can be applied there that (often) leads towards $s_{t+\tau}$ where a $r_{t+\tau}$ can be expected.

Inverse reinforcement learning

- Given measurements of an agent's behaviour over time (s_t, a_t, s_{t+1}), find out how reward is distributed if it is known that the agent derives its actions from a value function based on the rewards.
- In other words, analyse the behaviour (i.e. action choices) of the agents in order to find $R\left(s_t, a_t, s_{t+1}\right)$.
- Obviously, the reconstructed $\hat{R}$ will recover R only up to some transformation and baseline, even after successful learning.
- The task can be simplified (such that a realistic problem complexity is reached) if a transition model or time horizon is given.

Find (set of) possible reward function(s) R such that (empirically given) π is an optimal policy

T. S. Reddy et al. (2912) Inverse reinforcement learning for decentralized non-cooperative multiagent systems. To appear in Proc. IEEE Int. Conf. Systems, Man, and Cybernetics, Seoul, South Korea.

Motivation

Goals

- Learning from Demonstration
- What are the driving forces for human and animal behaviour?
- Enabling robots to “understand” humans or other robots.
- Improving the ability of robots to learn by imitation

Examples

- Car driving simulation (Abbeel et al 2004, Syed and Schapire, NIPS 2007)
- Autonomous helicopter flight (Andrew Ng et. al., 2006)
- Aerial imagery based navigation (Ratliff, Bagnell and Zinkevich, ICML 2006)
- Parking lot navigation Abbeel, Dolgov, Ng and Thrun, IROS 2008
- Urban navigation, route recommendation, and destination prediction (Ziebart, Maas, Bagnell and Dey, 2008)
- Human path planning (Mombaur, Truong and Laumond, AURO 2009)
- Human goal inference (Baker, Saxe and Tenenbaum, Cognition 2009)
- Quadruped locomotion (Ratliff, Bradley, Bagnell and Chestnutt, NIPS 2007; Kolter, Abbeel and Ng, NIPS 2008)

Why not “behavioural cloning”?

Use a supervised learning algorithm to find a optimal representation of the data $\{(s_t, a_t) | \, t \in \mathcal{T}\}$, i.e. find a policy $\hat{\pi}$ that minimises

$$\sum_{t \in \mathcal{T}'} \left(\hat{\pi}\left(s_t\right) - a_t\right)^2$$

Restrict family of estimators for the policy and choose $\mathcal{T}$ and $\mathcal{T}'$ differently in order to avoid overfitting.

Problems: Agent is inflexible, cannot change goals and may fail if more than present state is needed in order to perform (i.e. in a non-Markovian environment.

Possible solution: Hierarchical RL could use cloned partial policies.(see also: Pomerleau, NIPS 1989, ALVINN; Sammut et al., ICML 1992: Learning to fly.)

Here: Instead of the policy π, estimate the reward R (*inverse* RL)

IRL: Mathematical Formulation

- Given
  - State space $\mathcal{S}$, action space $\mathcal{A}$
  - Transition model $T(s, a, s') = P(s'|s, a)$
  - **not** given: reward function $R(s, a, s')$
  - Teacher's demonstration (from teacher's policy π^*): $s_0,\ a_0,\ s_1,\ a_1, \ldots$
- Find $\hat{R}$, such that $V^{\pi^*} \geq V^{\pi}$, $\forall \pi$, i.e.

$$E\left[\sum_{t=0}^{\infty} \gamma^t \hat{R}\left(s_t\right)|\pi^*\right] \geq E\left[\sum_{t=0}^{\infty} \gamma^t \hat{R}\left(s_t\right)|\pi\right],\ \forall \pi$$

- Problems:
  - π^* is given in terms of state transitions (trajectories). How to evaluate the expectations?
  - The conditions are satisfied by the trivial solution $\hat{R} = 0$. Solution methods should prefer non-trivial solutions.

Inverse reinforcement learning

Evaluation of the expectations by sample average of the data

$$\left\langle \sum_{t=0}^{\infty} \gamma^t \hat{R}(s_t) \,|\pi^* \right\rangle \geq \left\langle \sum_{t=0}^{\infty} \gamma^t \hat{R}(s_t) \,|\pi \right\rangle, \ \forall \pi$$

But trajectory is determined by value, while we need the rewards.

Use Bellman equations

$$V^\pi(s) = R + \gamma P^a(s, s') \, V^\pi(s')$$

Matrix form

$$R = (I - \gamma P^a) \, V^\pi$$

i.e.

$$V^\pi = (I - \gamma P^a)^{-1} R$$

A. Y. Ng & S. Russell (2000) Algorithms for inverse reinforcement learning. ICML.

N.B.: Why reward rather than value? Reward often independent of trajectory, more easily generalisable; value is "smoothed" reward.

Inverse reinforcement learning

Assuming there is a single best action

$$a = \arg\max_{b \in \mathcal{A}} R(s) + \sum_{s'} P^b(s, s') V(s')$$

i.e.

$$\sum_{s'} P^a(s, s') V(s') \succeq \sum_{s'} P^b(s, s') V(s') \quad \forall b \in \mathcal{A} \setminus \{a\}$$

in matrix form and using $R = (I - \gamma P^a)^{-1} V^\pi$

$$\sum_{s'} P^a(s, s')(I - \gamma P^a)^{-1} R(s) \succeq \sum_{s'} P^b(s, s')(I - \gamma P^a)^{-1} R(s) \;\; \forall b \in \mathcal{A} \setminus a$$

i.e. $\pi(s) = a$ is optimal if R satisfies ($\succeq$ element-wise greater-than)n)

$$P^a (I - \gamma P^a)^{-1} R \succeq P^b (I - \gamma P^a)^{-1} R$$

P^a denotes the $|\mathcal{S}| \times |\mathcal{S}|$ transition matrix given action a
R denotes the $|\mathcal{S}|$ vector $\{R(s, \pi(s))\}$

Inverse Reinforcement Learning

$$(P_a - P_b)(I - \gamma P_a)^{-1} R \succeq 0$$

- $R = 0$ is always a solution (give an advantage to nonzero R)
- Multiple solutions: Favour solutions that make single-step deviations from π as costly as possible (this does not reflect a property of the reward function we are looking for, but provides us with a statistically robust solution)
- Maximise

$$\sum_{i=1}^{n} \min_{b \in \mathcal{A} \setminus a} (P_a(s_i) - P_b(s_i))(I - \gamma P_a)^{-1} R$$

subject to $(P_a - P_b)(I - \gamma P_a)^{-1} R \succeq 0$ and $R \leq R_{\text{max}}$

- Solve as linear programming problem

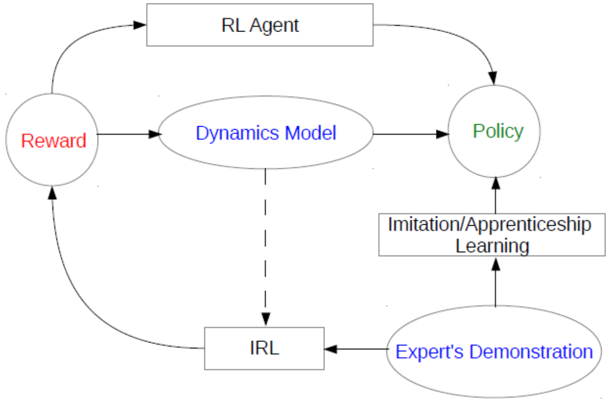
IRL with function approximation from sampled trajectories

Policy given only by a set of trajectories in state space (transition probabilities unknown), assume that the system can be simulated or queried

- Trajectory starts at $s \in \mathcal{S}_0$ with partial policy π given by trajectory
- Calculate $V^{\pi}(s_0)$ given candidate $R(s) = \sum_{i=1}^{d} \alpha_i \varphi(s)$ using function approximation
- Find α_i to maximise $\sum_{s \in \mathcal{S}_0} \min_{b \in \mathcal{A}} \left\{ P\left(\mathbb{E}_{s' \sim P_{sa}}\left[V^{\pi}(s') \right] - \mathbb{E}_{s' \sim P_{sb}}\left[V^{\pi}(s') \right] \right) \right\}$ subject to $|\alpha_i| < 1$
- New values for the α_i define a new reward function
- Repeat

It may be reasonable to add a regularisation term $-\lambda \|R\|_1$ where $\|\cdot\|_1$ denotes the absolute norm (ℓ_1) and $\lambda \geq 0$, see below.

Inverse RL and Apprenticeship Learning



From V. Ngo & M. Toussaint (inspired from a poster of Boularias, Kober, Peters)

Apprenticeship learning

- Inverse RL analyses the problem. Apprenticeship learning try to use the results of IRL. (Apprenticeship via inverse reinforcement learning by P. Abbeel)
- Markov decision process with unknown reward function: Observe behaviour of an expert
- Similar to *Learning from Demonstration*, but is expected to generalise better because reward structure is reconstructed in addition to the reproduction of behaviours
- May never exactly recover the expert's reward function, but will often attain performance close to that of the expert (measured by the unknown reward function of the expert)

P. Abbeel, A. Ng: Apprenticeship learning via inverse reinforcement learning." 21st ICML 2004.

Apprenticeship learning

Algorithm: For t = 1,2,...

- Inverse RL step: Estimate expert's reward function

  $$R(s) = w^\top \varphi(s)$$

  such that under $R(s)$ a model expert performs better than for all previously found policies π_i.
- RL step: Compute optimal policy π_t for the estimated reward weights w.

Because of

$$E\left[\gamma^t R\left(s_t\right)|\pi\right] = E\left[\gamma^t w^\top \varphi(s_t)|\pi\right] = w^\top E\left[\gamma^t \varphi(s_t)|\pi\right] = w^\top \eta\left(\pi\right)$$

the same weights can be used for the value function. Based on N sample trajectories we will use

$$\eta\left(\pi\right) = \frac{1}{N}\sum_{i=1}^{N}\sum_{t=0}^{T_i}\gamma^t \varphi\left(s_t\right)$$

Apprenticeship learning: Algorithm

Finding a policy π whose performance is as close to the expert policy's performance as possible

$$\left\| {w^*}^\top \eta(\pi^*) - w^\top \eta(\pi) \right\| \leq \varepsilon$$

Optimise margin between policies by margin of feature values:

- assume $R(s) = w^\top \varphi(s)$, where $w \in \mathbb{R}^n$ and $\varphi : \mathcal{S} \to \mathcal{R}$
- initialise π_0
- for $i = 1, \ldots$ do
  - Find a reward function such that the teacher maximally outperforms all previously found controllers

    $$\max_{\gamma, \|w\| \leq 1} \|\kappa\|$$

    such that

    $$w^\top \eta(\pi^*) \geq w^\top \eta(\pi) + \kappa \quad \forall \pi \in \{\pi_0, \pi_1, \ldots, \pi_{i-1}\}$$
  - Find optimal policy π_i for reward function R_w w.r.t current w.
- end for

Convergence and sampling

Convergence:

Let an MDP\R, k-dimensional feature vector φ be given. Then the algorithm will terminate with $\max \|\tau\| \leq \varepsilon$ after at most

$$O\left(\frac{k}{(1-\gamma)^2\varepsilon^2}\right)\log\left(\frac{k}{(1-\gamma)\varepsilon}\right)$$

iterations.

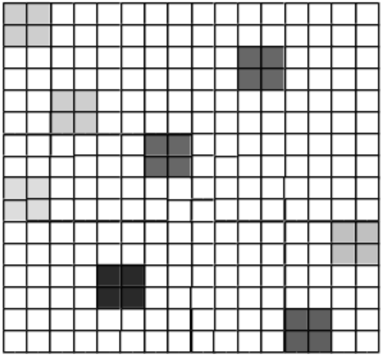
Sampling:

In practice, we have to use sampling estimates for the feature distribution of the expert. We still have ε-optimal performance w.p. $(1-\delta)$ for the following number of samples

$$m \geq \frac{9k}{2(1-\gamma)^2\varepsilon^2}\log\frac{2k}{\delta}.$$

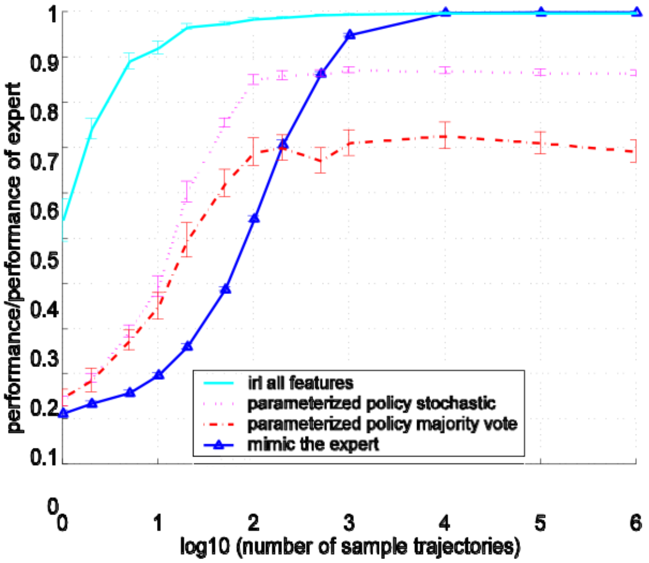
... complexity is polynomial.

Apprenticeship learning: Example

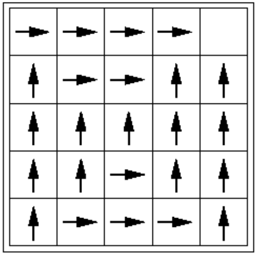


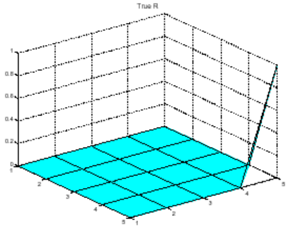
- Reward function is piece-wise constant over small regions. Features φ for IRL are these small regions.
- 128x128 grid, small regions of size 16x16 (macrocell), a few of which have positive reward.
- Learning using 8x8 macrocells (features)
- 30% of actions are random

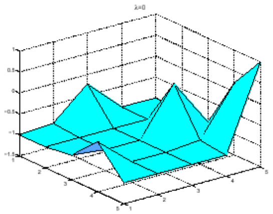
Apprenticeship learning: Example

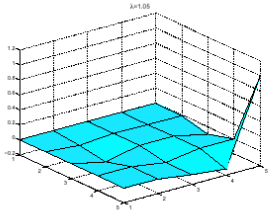


Experiments (Data and ground truth vs. reconstruction)







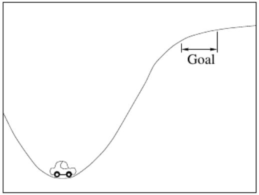


Bottom: true reward function.

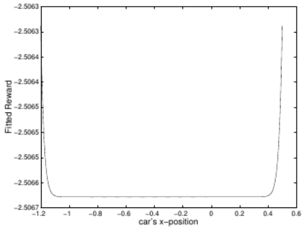
Results for two different regularisation parameters (top $\lambda = 0$, bottom $\lambda = 1.05$).

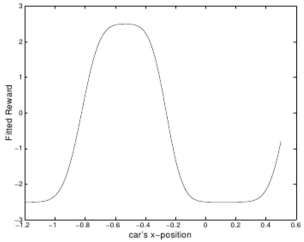
Ng, A. Y., & Russell, S. J. (2000). Algorithms for inverse reinforcement learning. In Icml (pp. 663-670).

Experiment: IRL with function approximation



Mountain car problems:
(i) Going up
(ii) Parking down
State space: x, v
Reconstruction of the optimal policy (given on a 120× 120 discretisation of the state space
Based on 5000 samples





Apprenticeship Learning: Conclusion

- Given expert demonstrations, the inverse RL algorithm returns a policy with performance as good as the expert as evaluated according to the expert's unknown reward function.
- Given an initial demonstration, there is no need to explicitly explore the state/action space. Even if you repeatedly "exploit" (use your best policy), you will collect enough data to learn a sufficiently accurate dynamical model to carry out your control task.

P. Abbeel, A. Ng: Apprenticeship learning via inverse reinforcement learning." 21st ICML 2004.

Challenges

- Experts (in particular human experts) are not optimal.
- Time series may contain errors or rare deviations
- In competitive environments opponents may play tricks or be not optimal themselves
- Bayesian problem for probabilistic policies (Neu and Szepesvari, UAI2007)
- In POMDPs: Choi, J., & Kim, K. E. (2011) Inverse reinforcement learning in partially observable environments. *JMLR* **12**, 691-730.
- Transfer learning: Taylor, M. E., & Stone, P. (2009). Transfer learning for reinforcement learning domains: A survey. *JMLR* **10**, 1633-1685.

Acknowledgements & References

See also: Neu, G., & Szepesvári, C. (2012). Apprenticeship learning using inverse reinforcement learning and gradient methods. arXiv preprint arXiv:1206.5264.

Some material was adapted from web resources associated with Sutton and Barto's Reinforcement Learning book and C. Szepesvári: *Algorithms for RL*.

Also a talk on "Inverse Reinforcement Learning" by Pieter Abbeel has been used.

and on the slides by Dr. Subramanian Ramamoorthy from the previous years.

See also: http://umichrl.pbworks.com/w/page/7597585/Myths of Reinforcement Learning