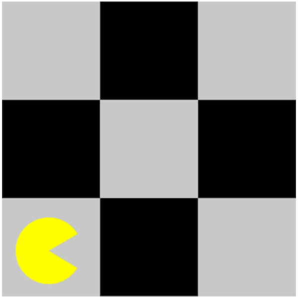
Inverse Reinforcement Learning
Model-Free Approaches

Olivier Pietquin
olivier.pietquin@supelec.fr

Supélec - UMI 2958 (GeorgiaTech - CNRS), France
MaLIS research Group

September 6, 2012
GDR Robotique

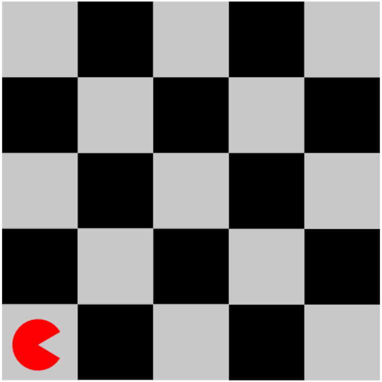
Reinforcement Learning



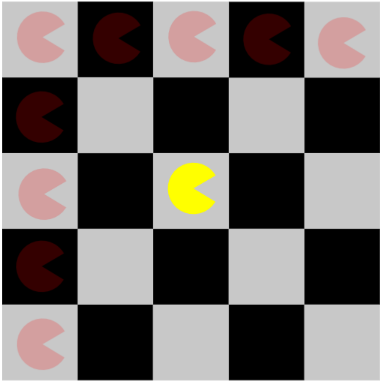
Notions

- Agent
- Task
- Environment

Imitation: Expert



Generalization



Inverse RL

Reward inference

Given 1) measurements of an agent's behaviour over time, in a variety of circumstances, 2) measurements of the sensory inputs to that agent; 3) a model of the physical environment (if available).

Determine the reward function that the agent is optimizing

Seminal paper by Russell [Russell, 1998] (and [Kalman, 1964])

Why?

- In RL, the reward is often a heuristic that might be wrong
- Understand human and animal behavior
- Most compact representation of the task (transfert imitation)

Questions

- To what extend is a reward uniquely recoverable?
- What are appropriate error metrics for fitting?
- Computational complexity?
- Can we identify local inconsistencies?
- Can we determine the reward before learning?
- How many observations are required?

First IRL algorithms [Ng and Russell, 2000]

Inversing Bellman Equations for finite MDPs

$$\forall a \neq a^* : \; (\mathbf{P}_{a^*} - \mathbf{P}_a)(\mathbf{I} - \gamma \mathbf{P}_{a^*})^{-1}\mathbf{R} \succeq 0$$

Problems

- $\mathbf{0}$ is a solution
- There is an infinite set of solutions

Solutions: add constrains

- Penalize deviations w.r.t. π^*
- Linear representation of R ($\hat{r} = \theta^T \psi(s)$)
- Solve with linear programming

Apprenticeship learning via IRL [Abbeel and Ng, 2004] I

Find **a policy** whose performance is close to that of the expert's, on the unknown reward function $\hat{r}^* = {\theta^*}^T \psi(s)$.

Feature expectation : $\mu^\pi(s)$

$$V^\pi(s_t) = E\left[\sum_i \gamma^i r_{t+i} \middle| \pi\right]$$

$$V^\pi(s_t) = E\left[\sum_i \gamma^i \theta^T \psi(s_{t+i}) \middle| \pi\right]$$

$$V^\pi(s_t) = \theta^T \underbrace{E\left[\sum_i \gamma^i \psi(s_{t+i}) \middle| \pi\right]}_{\mu^\pi(s_t)} = \theta^T \mu^\pi(s_t)$$

Apprenticeship learning via IRL [Abbeel and Ng, 2004] II

Fitness to performance

$if\, \|\theta\|_2 \leq 1 : \|\mu^\pi(s) - \mu^E(s)\|_2 < \epsilon \Rightarrow \|V^\pi(s) - V^E(s)\|_2 < \epsilon$

1. Initiate Π with random policy π_0 and **compute** $\mu^{\mathbf{0}} = \mu^{\pi_0}$
2. Compute t and θ such that

$$t = \max_\theta \{ \min_{\pi_i \in \Pi} \theta^T (\mu^E - \mu^i) \} \text{ s.t. } \|\theta\|^2 \leq 1$$

If $t \leq \xi$ Terminate

3. Train a new policy π_i optimizing $R = \theta^T \psi(s)$
4. **Compute** $\mu^{\mathbf{i}}$ for π_i; $\Pi \leftarrow \pi_i$
5. Goto to step 2.

Apprenticeship learning via IRL [Abbeel and Ng, 2004] III

Many methods based on matching feature expectations

- Game theoretic point of view [Syed and Schapire, 2008]
- Linear programming (generates only 1 policy) [Syed *et al.*, 2008]
- Maximum entropy [Ziebart *et al.*, 2008]

All iterative algorithms requiring computing μ^π for many π

Max-Margin Planning [Ratliff *et al.*, 2006] I

Ideas

- Maximize the margin between the best policy obtained with the current estimation and the expert policy
- Allow the expert to be non-optimal (slack variables)

Quadratic program

$$\min_{\theta,\xi_i} \frac{1}{2}\|\theta\|_2 + \frac{\gamma}{n}\sum_i \beta_i \xi_i^2$$

$$s.t. \forall i : \theta^T \mu^{\pi_i^*}_{d_0,\mathcal{M}_i} + \xi_i \geq \max_\pi \theta^T \mu^{\pi}_{d_0,\mathcal{M}_i} + \mathcal{L}_{\mathcal{M}_i,\pi}$$

Max-Margin Planning [Ratliff *et al.*, 2006] II

Optimization problem : min $J(\theta)$

$$J(\theta) = \frac{1}{N}\sum_{1=1}^{N} \max_{\pi}(\theta^T \mu^{\pi}_{d_0,\mathcal{M}_i} + \mathcal{L}_{\mathcal{M}_i,\pi}) - \theta^T \mu^{\pi_i^*}_{d_0,\mathcal{M}_i} + \frac{\lambda}{2}\|\theta\|_2$$

Problems

- max operator : sub-gradient descent
- Need to **solve an MDP** for each constraint : A*
- Need to estimate μ^{π} for any π

Max-Margin Planning [Ratliff *et al.*, 2006] III

Classification based [Ratliff *et al.*, 2007][Taskar *et al.*, 2005]

$$J(q) = \frac{1}{N} \sum_{i=1}^{N} \max_a (q(s_i, a) + l(s_i, a)) - q(s_i, a_i).$$

Problems

- max operator : sub-gradient descent
- Not taking the structure into account (inconsistent behaviour)

Advantages

- $q(s, a)$ can be of any (differentiable) form
- No need of knowing the MDP

Structured Classification of IRL: SCIRL [Klein *et al.*, 2012]

Best of both worlds

- No need of MDP model
- Take structure into account

Contributions

- Classification with the structure of the MDP
- Only needs data from the expert
- Can use other data if available

Special case of classification

Score function based classifiers

- Classifier : map inputs $x \in \mathcal{X}$ to labels $y \in \mathcal{Y}$
- Data : $\{(x_i, y_i)_{1 \leq i \leq N}\}$
- Decision rule : $g \in \mathcal{Y}^{\mathcal{X}}$
- Score function : $g_s(x) \in \arg\max_{y \in \mathcal{Y}} s(x, y)$

Linearly parameterized score function

$$s(x, y) = \theta^T \phi(x, y) \tag{1}$$

Let's compare

Linearly parametrized score function based classifiers

- $g_s(x) \in \arg\max_{y \in \mathcal{Y}} s(x, y)$
- $s(x, y) = \theta^T \phi(x, y)$
- $g_s(x) \in \arg\max_{y \in \mathcal{Y}} \theta^T \phi(x, y)$

Expert's decision rule

- $\pi^E(s) = \arg\max_a Q^{\pi^E}(s, a)$
- $Q^\pi(s_t, a) = \theta^T \mu^\pi(s_t, a)$
- $\pi^E(s) = \arg\max_a \theta^T \mu^{\pi^E}(s, a)$

Let's compare

Putting it all together

$\mathcal{X} \equiv S,$,

Linearly parametrized score function based classifiers

- $g_s(x) \in \arg\max_{y \in \mathcal{Y}} s(x, y)$
- $s(x, y) = \theta^T \phi(x, y)$
- $g_s(x) \in \arg\max_{y \in \mathcal{Y}} \theta^T \phi(x, y)$

Expert's decision rule

- $\pi^E(s) = \arg\max_a Q^{\pi^E}(s, a)$
- $Q^\pi(s_t, a) = \theta^T \mu^\pi(s_t, a)$
- $\pi^E(s) = \arg\max_a \theta^T \mu^{\pi^E}(s, a)$

Let's compare

Putting it all together

$\mathcal{X} \equiv S$, $\mathcal{Y} \equiv A$,

Linearly parametrized score function based classifiers

- $g_s(x) \in \arg\max_{y \in \mathcal{Y}} s(x, y)$
- $s(x, y) = \theta^T \phi(x, y)$
- $g_s(x) \in \arg\max_{y \in \mathcal{Y}} \theta^T \phi(x, y)$

Expert's decision rule

- $\pi^E(s) = \arg\max_a Q^{\pi^E}(s, a)$
- $Q^\pi(s_t, a) = \theta^T \mu^\pi(s_t, a)$
- $\pi^E(s) = \arg\max_a \theta^T \mu^{\pi^E}(s, a)$

Let's compare

Putting it all together

$\mathcal{X} \equiv S$, $\mathcal{Y} \equiv A$, $s \equiv Q^{\pi^E}$

Linearly parametrized score function based classifiers

- $g_s(x) \in \arg\max_{y \in \mathcal{Y}} s(x, y)$
- $s(x, y) = \theta^T \phi(x, y)$
- $g_s(x) \in \arg\max_{y \in \mathcal{Y}} \theta^T \phi(x, y)$

Expert's decision rule

- $\pi^E(s) = \arg\max_a Q^{\pi^E}(s, a)$
- $Q^{\pi}(s_t, a) = \theta^T \mu^{\pi}(s_t, a)$
- $\pi^E(s) = \arg\max_a \theta^T \mu^{\pi^E}(s, a)$

Let's compare

Putting it all together

$$\mathcal{X} \equiv S,\ \mathcal{Y} \equiv A,\ s \equiv Q^{\pi^E} \Rightarrow \phi \equiv \mu_E$$

Linearly parametrized score function based classifiers

- $g_s(x) \in \arg\max_{y \in \mathcal{Y}} s(x, y)$
- $s(x, y) = \theta^T \phi(x, y)$
- $g_s(x) \in \arg\max_{y \in \mathcal{Y}} \theta^T \phi(x, y)$

Expert's decision rule

- $\pi^E(s) = \arg\max_a Q^{\pi^E}(s, a)$
- $Q^\pi(s_t, a) = \theta^T \mu^\pi(s_t, a)$
- $\pi^E(s) = \arg\max_a \theta^T \mu^{\pi^E}(s, a)$

SCIRL Pseudo-code

Algorithm 1: SCIRL algorithm

Given a training set $\mathcal{D} = \{(s_i, a_i = \pi_E(s_i))_{1 \leq i \leq N}\}$, an estimate $\hat{\mu}^{\pi_E}$ of the expert feature expectation μ^{π_E} and a classification algorithm;

Compute the parameter vector θ_c using the classification algorithm fed with the training set $\mathcal{D}$ and considering the parameterized score function $\theta^T \hat{\mu}^{\pi_E}(s, a)$;

Output the reward function $R_{\theta_c}(s) = \theta_c^T \psi(s)$;

Why is it nice?

Advantages

- Only μ^E is required
- No need to compute μ for other policies
- Based on transitions (not trajectories or policies)
- Is based on standard classification methods
- Bounds can be computed
- Returns a reward

Error bound

Definitions

- $C_f = (1-\gamma)\sum_{t\geq 0}\gamma^t c(t)$ with $c(t) = \max_{\pi_1,\dots,\pi_t,s\in S}\frac{(\rho_E^T P_{\pi_1}\dots P_{\pi_t})(s)}{\rho_E(s)}$

Error bound

Definitions

- $C_f = (1-\gamma)\sum_{t\geq 0}\gamma^t c(t)$ with $c(t) = \max_{\pi_1,\dots,\pi_t,s\in S}\frac{(\rho_E^T P_{\pi_1}\dots P_{\pi_t})(s)}{\rho_E(s)}$
- $\epsilon_c = \underset{s\sim\rho_E}{E}[\mathbf{1}_{\{\pi_c(s)\neq\pi_E(s)\}}] \in [0,1]$

Error bound

Definitions

- $C_f = (1-\gamma)\sum_{t\geq 0}\gamma^t c(t)$ with $c(t) = \max_{\pi_1,\dots,\pi_t,s\in S} \frac{(\rho_E^T P_{\pi_1}\dots P_{\pi_t})(s)}{\rho_E(s)}$
- $\epsilon_c = \underset{s\sim\rho_E}{E}[\mathbf{1}_{\{\pi_c(s)\neq\pi_E(s)\}}] \in [0,1]$
- $\epsilon_\mu = \hat{\mu}^{\pi_E} - \mu^{\pi_E} : S \times A \to \mathbb{R}^p$

Error bound

Definitions

- $C_f = (1-\gamma)\sum_{t\geq 0}\gamma^t c(t)$ with $c(t) = \max_{\pi_1,\dots,\pi_t,s\in S} \frac{(\rho_E^T P_{\pi_1}\dots P_{\pi_t})(s)}{\rho_E(s)}$
- $\epsilon_c = \underset{s\sim\rho_E}{E}[\mathbf{1}_{\{\pi_c(s)\neq\pi_E(s)\}}] \in [0,1]$
- $\epsilon_\mu = \hat{\mu}^{\pi_E} - \mu^{\pi_E} : S \times A \to \mathbb{R}^p$ $\qquad \epsilon_Q = \theta_c^T \epsilon_\mu : S \times A \to \mathbb{R}$

Error bound

Definitions

- $C_f = (1-\gamma)\sum_{t\geq 0}\gamma^t c(t)$ with $c(t) = \max_{\pi_1,\dots,\pi_t,s\in S} \frac{(\rho_E^T P_{\pi_1}\dots P_{\pi_t})(s)}{\rho_E(s)}$
- $\epsilon_c = \underset{s\sim\rho_E}{E}[\mathbf{1}_{\{\pi_c(s)\neq\pi_E(s)\}}] \in [0,1]$
- $\epsilon_\mu = \hat{\mu}^{\pi_E} - \mu^{\pi_E} : S\times A \to \mathbb{R}^p$ $\qquad \epsilon_Q = \theta_c^T\epsilon_\mu : S\times A\to\mathbb{R}$
- $\bar{\epsilon}_Q = \underset{s\sim\rho_E}{E}[\max_{a\in A}\epsilon_Q(s,a) - \min_{a\in A}\epsilon_Q(s,a)] \geq 0$

Error bound

Definitions

- $C_f = (1-\gamma)\sum_{t\geq 0}\gamma^t c(t)$ with $c(t) = \max_{\pi_1,\dots,\pi_t,s\in S}\frac{(\rho_E^T P_{\pi_1}\dots P_{\pi_t})(s)}{\rho_E(s)}$
- $\epsilon_c = \underset{s\sim\rho_E}{E}[\mathbf{1}_{\{\pi_c(s)\neq\pi_E(s)\}}] \in [0,1]$
- $\epsilon_\mu = \hat{\mu}^{\pi_E} - \mu^{\pi_E} : S\times A \to \mathbb{R}^p$ $\qquad \epsilon_Q = \theta_c^T\epsilon_\mu : S\times A\to\mathbb{R}$
- $\bar{\epsilon}_Q = \underset{s\sim\rho_E}{E}[\max_{a\in A}\epsilon_Q(s,a) - \min_{a\in A}\epsilon_Q(s,a)] \geq 0$

Theorem

$$0 \leq \underset{s\sim\rho_E}{E}[v^*_{R_{\theta_c}} - v^{\pi_E}_{R_{\theta_c}}] \leq \frac{C_f}{1-\gamma}\left(\bar{\epsilon}_Q + \epsilon_c\frac{2\gamma\|R_{\theta_c}\|_\infty}{1-\gamma}\right) \qquad (2)$$

Instanciation

Structured large margin classifier [Taskar *et al.*, 2005]

$$J(\theta) = \frac{1}{N}\sum_{i=1}^{N} \max_a \theta^T \hat{\mu}^{\pi_E}(s_i, a) + \mathcal{L}(s_i, a) - \theta^T \hat{\mu}^{\pi_E}(s_i, a_i) + \frac{\lambda}{2}\|\theta\|^2. \quad (3)$$

- $\mathcal{L}(s_i, a) = 1$ si $a \neq a_i$, 0 otherwise
- Sub-gradient descend

Computing μ^E

LSTD-μ [?]

Based on already known *Least-square temporal differences* method

Characteristics

- Can be fed with mere transitions
- No model
- Off or on policy evaluation

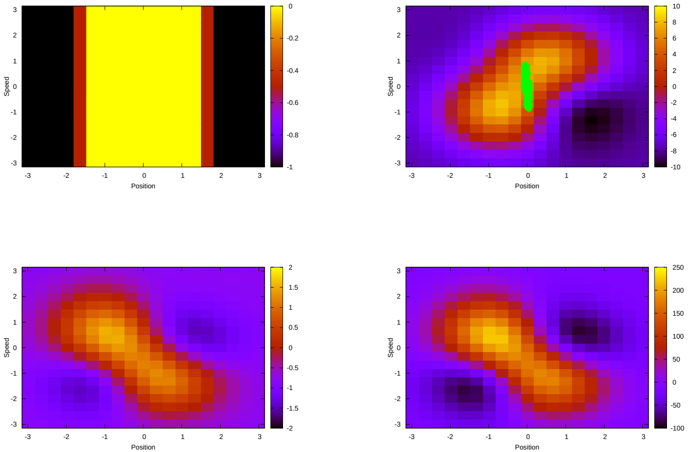
Monte carlo with heuristics

- $\hat{\mu}^\pi(s_0, a_0) = \frac{1}{M} \sum_{j=1}^{M} \sum_{i \geq 0} \gamma^i \phi(s_i^j)$
- $\hat{\mu}^\pi(s_0, a \neq a_0) = \gamma \hat{\mu}^\pi(s_0, a_0)$

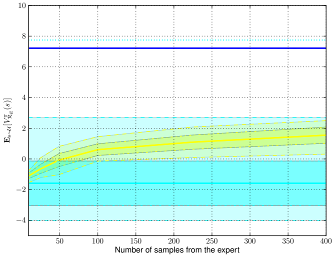
Characteristics

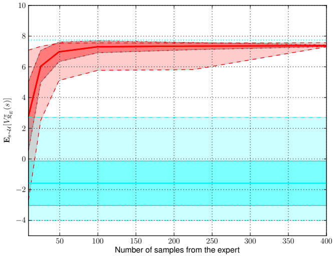
- Only on-policy
- Seems more robust in dire conditions

Inverted Pendulum



Results on the driving problem





Description

- Goal of the expert : avoid other cars, do not go off-road, go fast
- Using only data from the expert and natural features
- Non trivial (State of the art does not work)

Future work

- No computation of μ^E everywhere
- Real-world problem
- Task Transfer ?

Thank you. . .

. . . for your attention

References I

Pieter Abbeel and Andrew Y. Ng.
Apprenticeship learning via inverse reinforcement learning.
In *Proceedings of the 21st International Conference on Machine learning (ICML)*, 2004.

Rudolph Kalman.
When is a linear control system optimal?
Transactions of the ASME, Journal of Basic Engineering, Series D, 86:81.90, 1964.

Edouard Klein, Matthieu Geist, Bilal PIOT, and Olivier Pietquin.
Inverse Reinforcement Learning through Structured Classification.
In *Advances in Neural Information Processing Systems (NIPS 2012)*, Lake Tahoe (NV, USA), December 2012.

Andrew Y. Ng and Stuart Russell.
Algorithms for Inverse Reinforcement Learning.
In *Proceedings of 17th International Conference on Machine Learning (ICML)*, 2000.

Nathan Ratliff, Andrew D. Bagnell, and Martin Zinkevich.
Maximum Margin Planning.
In *Proceedings of the 23rd International Conference on Machine Learning (ICML)*, 2006.

Nathan Ratliff, Andrew D. Bagnell, and Martin Zinkevich.
Imitation learning for locomotion and manipulation.
In *Proceedings of ICHR*, 2007.

References II

Stuart Russell.
Learning agents for uncertain environments (extended abstract).
In *Proceedings of the 11th annual Conference on Computational Learning Theory (COLT)*, 1998.

Umar Syed and Robert Schapire.
A game-theoretic approach to apprenticeship learning.
In *Advances in Neural Information Processing Systems 20 (NIPS)*, 2008.

Umar Syed, Michael Bowling, and Robert E. Schapire.
Apprenticeship learning using linear programming.
In *Proceedings of the 25th International Conference on Machine learning (ICML)*, 2008.

Ben Taskar, Vassil Chatalbashev, Daphne Koller, and Carlos Guestrin.
Learning Structured Prediction Models: a Large Margin Approach.
In *Proceedings of 22nd International Conference on Machine Learning (ICML)*, 2005.

Brian D. Ziebart, Andrew Maas, J. Andrew Bagnell, and Anind K. Dey.
Maximum entropy inverse reinforcement learning.
In *Proceedings of the 23rd national conference on Artificial intelligence (AAAI)*, 2008.