Reinforcement Learning with the Use of Costly Features

Robby Goetschalckx[2], **Scott Sanner**[1] and **Kurt Driessens**[2]

Abstract.

A common solution approach to reinforcement learning problems with large state spaces (where value functions cannot be represented exactly) is to compute an approximation of the value function in terms of state features. However, little attention has been paid to the cost of computing these state features (e.g., search-based features). To this end, we introduce a cost-sensitive sparse linear-value function approximation algorithm — FOVEA — and demonstrate its performance on an experimental domain with a range of feature costs.

1 Introduction

Reinforcement learning problems with large state spaces often preclude the representation of a fully enumerated value function. In this case, a common solution approach is to compute an approximation of the value function in terms of state features. While value function approximation is well-addressed in the reinforcement learning literature (c.f., Chapter 8 of [4]), the cost of feature computation is often ignored. Yet in the presence of costly features, this cost must be traded off with its impact on value prediction accuracy.

While reinforcement learning is often modelled as a Markov decision process (MDP) [3], one might consider modelling the function approximation setting with costly features as a partially observable MDP (POMDP) [2] by using information-gathering actions to represent the computation of costly features. In theory, an optimal policy for this POMDP would select those features to compute at any decision stage in order to optimally trade-off feature cost w.r.t. its impact on future reward. However, such a framework requires embedding an already difficult-to-solve MDP inside a POMDP; in general, solutions to such a POMDP will not be feasible in practice.

Here we propose a more pragmatic approach where we learn the relative value of features in an explicit way. To do this, we approximate the value function using cost-sensitive sparse linear regression techniques, directly trading off prediction errors with feature costs.

2 MDPs and Reinforcement Learning

We briefly review *Markov decision processes* (MDPs) [3] and *reinforcement learning* (RL) [4]. Formally, an MDP can be defined as a tuple $\langle S, A, T, R, \gamma \rangle$. $S = \{s_1, \ldots, s_n\}$ is a finite set of fully observable states. $A = \{a_1, \ldots, a_m\}$ is a finite set of actions. $T : S \times A \times S \rightarrow [0, 1]$ is a stationary, Markovian transition function. A reward $R : S \times A \rightarrow \mathbb{R}$ is associated with every state and action. γ is a discount factor s.t. $0 \leq \gamma < 1$ used to specify that a reward obtained t timesteps into the future is discounted by γ^t. $\gamma = 1$ is permitted if total accumulated reward is finite.

[1] National ICT Australia, email: Scott.Sanner@nicta.com.au

[2] Declarative Languages and Artificial Intelligence, Katholieke Universiteit Leuven, Leuven, Belgium, email: {robby,kurtd}@cs.kuleuven.be

A policy $\pi : S \rightarrow A$ specifies the action $a = \pi(s)$ to take in each state s. The value $Q^\pi(s, a)$ of taking an action a in state s and then following the policy π thereafter can be defined using the infinite horizon, expected discounted reward criterion:

$$Q^\pi(s, a) = E_\pi \left[\sum_{t=0}^{\infty} \gamma^t \cdot r_t \middle| s_0 = s, a_0 = a \right] \quad (1)$$

where r_t is the reward obtained at time t (assuming s_0 and a_0 respectively represent the state and action at $t = 0$). The objective in an MDP is to find a policy π^* such that $\forall \pi, s.\ Q^{\pi^*}(s, \pi^*(s)) \geq Q^\pi(s, \pi(s))$. An optimal policy π^* is guaranteed to exist.

In the RL setting, the transition and reward model may not be explicitly known to the agent although both can be sampled from experience. Here, we use the *generalized policy iteration* (GPI) framework known to capture most reinforcement learning algorithms [4]. GPI interleaves policy evaluation and update stages as follows:

Generalized Policy Iteration (GPI)

1. Start with arbitrary initial policy π_0 and set $i = 0$.
2. Estimate $Q^{\pi_i}(s, a)$ (e.g., from samples using Equation 1).
3. Let $\pi_{i+1}(s) = \arg\max_{a \in A} Q^{\pi_i}(s, a)$.
4. If termination criteria not met, let $i = i + 1$ and goto step 2.

Every RL algorithm that is an instance of GPI algorithm may prescribe its own method for performing each step and many GPI instances guarantee convergence to π^* or an approximation thereof. We keep our treatment of reinforcement learning with costly features as general as possible. Specifically, this means that in the context of GPI, we can restrict our discussion of RL with costly features to that of cost-efficient Q-value approximation in step 2 of GPI.

3 Cost-efficient Value-approximation

We represent a Q-value approximation $\hat{Q}^\pi_{\vec{w}}(s, a)$ w.r.t. policy π as a linear combination of a feature set $\mathcal{F} = \{f_1, \ldots, f_k\}$ with weights $\vec{w} = \langle w_0, \ldots, w_k \rangle$ where each $f_i : S \times A \rightarrow \mathbb{R}$ and each $w_i \in \mathbb{R}$:

$$\hat{Q}^\pi_{\vec{w}}(s, a) = w_0 + \sum_{f_i \in \mathcal{F}} w_i f_i(s, a) \quad (2)$$

We assume each feature f_i is associated with cost $c_{f_i} \in \mathbb{R}$ expressed in the same units as prediction error. Our task will be to find feature weights $\vec{w}$ that trade-off Q-value accuracy with feature cost.

At step 2 of GPI, we assume that we are given data $D = \{Q^\pi_{s,a}\}$ consisting of sampled Q-values to approximate. Then we define the optimal cost-efficient value approximation $\vec{w}^*$ as follows:

$$\vec{w}^* = \operatorname*{argmin}_{\vec{w}} \frac{1}{|D|} \sum_{Q^\pi_{s,a} \in D} [Q^\pi_{s,a} - \hat{Q}^\pi_{\vec{w}}(s, a)]^2 + \sum_{f_i \in \mathcal{F}} \mathbb{I}[w_i \neq 0] \frac{c_{f_i}}{1 - \gamma}$$

Here, $\mathbb{I}[\cdot]$ is 1 when its argument is true and 0 otherwise. We see

that the optimal setting of $\vec{w}^*$ directly trades off prediction error with feature cost (divided by $(1-\gamma)$ to account for the future discounted cost of feature evaluation at every time step). Unfortunately, this optimization objective is not convex due to step discontinuities where any $w_i = 0$ and thus not easily amenable to finding a global optima.

However, observing that weight sparsity encourages low feature cost, we can modify sparse linear regression approaches to encourage sparsity for a feature weight in a manner proportional to its cost. To do this, we focus on a class of sparse linear regression techniques collectively referred to as *least-angle regression* (LAR) methods, such as *lasso* and *forward stepwise regression* [1]. Fortunately, a simple modification of forward-stepwise regression provides us with an efficient algorithm — FOVEA — for approximating the solution to our optimization problem. We present FOVEA below and refer the reader to the detailed discussion in [1] for the original algorithm.

***Fo*rward-stepwise *V*alu*e* *A*pproximation (FOVEA)**

1. **Input:** Q-value samples $D = \{Q^\pi_{s,a}\}$ for policy π (e.g., computed from sample trajectories for a policy π using Equation 1).
2. Initialize $\mathcal{F} = \emptyset$.
3. Initialize $w_i = 0$ for $i \geq 1$ and w_0 with the average value of $Q_{s,a} \in D$ (this gives the residuals a mean of 0).
4. Normalize all feature predictions to have 0 mean and a standard deviation of 1.
5. Initialize the step-size η to some small positive value.
6. Repeat the following:
   (a) Compute a vector of residuals $\vec{r}$ and a vector of feature values $\vec{f}_i$ with entries for each data sample $Q^\pi_{s,a} \in D$ where the residual is $Q^\pi_{s,a} - \hat{Q}^\pi_{\vec{w}}(s,a)$ and the feature value is $f_i(s,a)$.
   (b) Calculate cost-penalized correlation score for all $f_i \in \mathcal{F}$:
   $$score_i = \frac{1}{|D|}\left|\vec{f}_i \cdot \vec{r}\right| - \mathbb{I}[f_i \in \mathcal{F}]\sqrt{c_{f_i}}$$
   (c) Find the feature f_i with the highest $score_i \geq \eta$; if no such feature found then halt and **Output:** $\vec{w}$.
   (d) **If** $f_i \notin \mathcal{F}$, let $\mathcal{F} = \mathcal{F} \cup \{f_i\}$; $w_i = w_i + \text{sgn}(\vec{f}_i \cdot \vec{r})\sqrt{c_{f_i}}$.[3]
   (e) **Else** let $w_i = w_i + \text{sgn}(\vec{f}_i \cdot \vec{r})\eta$.

It is important to note that the forward stepwise approach is a *greedy* selection approach and thus the result obtained might not be the optimal one in all cases. However, we can still prove a form of local optimality during the progression of the FOVEA algorithm:

Theorem 1 *Every feature f_i which is introduced in step 6d of the FOVEA algorithm immediately reduces the mean squared error of the prediction by the value of its cost c_{f_i}.*

4 Experiments

We evaluated GPI using FOVEA on a simple deterministic corridor domain. The state space consists of five rooms, labeled $s_1, \ldots, s_5$. From each state two actions, $+1, -1$ can be performed. Performing action $+1$ in state s_i for $i < 5$ leads to s_{i+1} and performing -1 in state s_i for $i > 1$ leads to s_{i-1}. All other actions take the agent to the center s_3. A reward of 1 is assigned for taking $+1$ in s_5 and -1 is assigned for taking action -1 in s_1. All other rewards are equal to 0. We used a discount factor $\gamma = 0.9$.

[3] $\text{sgn}(\cdot)$ is $+1$ if its argument is non-negative and -1 otherwise.

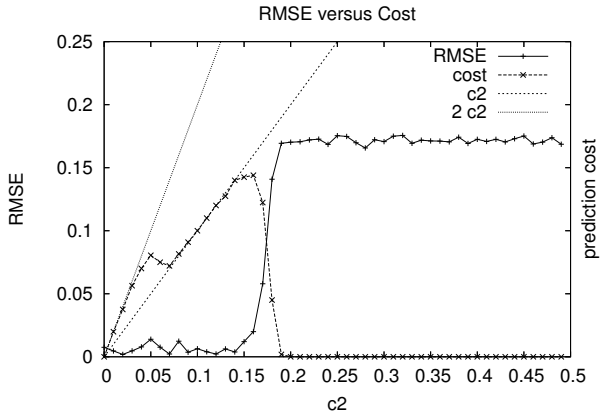

Figure 1. Prediction error (RMSE) vs. the feature cost of the prediction.

We provided seven state-action indicator features f_i for $0 \leq i \leq 6$ to the agent where taking action $a \in \{+1, -1\}$ in i results in $f_{i+a} = 1$ with all remaining indicator features set to 0. f_0, f_2, f_3, f_5 and f_6 are free and are assigned cost $c_1 = 0$ while f_1 and f_4 have a cost c_2. Furthermore two random number generators were provided to the agent, one which was free and another one which had a cost $c_3 > c_2$. Finally, the state-action feature indicators $f_0, \ldots, f_6$ were copied but now with the higher cost c_3. We used forward-stepwise value approximation to approximate Q-values using the state-action features defined above. We used 100 samples for each forward-stepwise update. All results shown are averages over 10 runs.

We varied the value of c_2 over a range of 0 to 0.5. If the agent does not pay c_2 for f_1 or f_4, it can not distinguish between s_1 and s_4 (if it pays the cost of only one of f_1 or f_4, it can still infer the other by absence). The results in Figure 1 demonstrate the effectiveness of FOVEA. Initially the agent pays $2c_2$ for both f_1 and f_4 (illustrating slight sub-optimality by paying for both features due to inherent statistical noise in the estimation process, but still avoiding the useless features that cost c_3) until it realizes for $c_2 > 0.05$ that it can just pay c_2 for one of these features and still obtain low prediction error. However, for $c_2 > 0.185$, the agent refuses to pay the cost for either f_1 or f_4 since the cost exceeds the future expected reward. As such, there is a clear phase transition near $c_2 = 0.185$ as the paid feature cost decreases rapidly while the prediction error likewise increases.

5 Future Work

Perhaps the most important area of future work is to explore efficient extensions to handle state- and action-dependent feature selection.

Acknowledgements

This research was sponsored by the fund for scientific research (FWO) of Flanders, of which Kurt Driessens is a postdoctoral fellow, and by National ICT Australia.

REFERENCES

[1] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, 'Least angle regression', Tech. report, Statistics Department, Stanford University, (2002).
[2] Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra, 'Planning and acting in partially observable stochastic domains', *Artificial Intelligence*, **101**, 99–134, (1998).
[3] Martin L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, Wiley, New York, 1994.
[4] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*, The MIT Press, Cambridge, MA, 1998.