Focus of Attention in Sequential Decision Making

Lihong Li and **Vadim Bulitko** and **Russell Greiner**

University of Alberta
Department of Computing Science
Edmonton, Alberta, Canada T6G 2E8
{lihong,bulitko,greiner}@cs.ualberta.ca

Abstract

We investigate the problem of using function approximation in reinforcement learning (RL) where the agent's control policy is represented as a classifier mapping states to actions. The innovation of this paper lies with introducing a measure of state's decision-making importance. We then use an efficient approximation to this measure as misclassification costs in learning the agent's policy. As a result, the focused learning process is shown to converge faster to better policies.

Introduction

Reinforcement learning (RL) (Sutton & Barto 1998) provides a general framework for many sequential decision-making problems and has succeeded in a number of important applications. Two primary approaches to learning agent's policy have been investigated. Methods based on value functions learn an approximation to the long-term return and then use it greedily (Sutton & Barto 1998; Watkins 1989). Policy-search methods, on the other hand, seek an approximation to the optimal policy *directly* in the policy space (Kearns, Mansour, & Ng 2000; Ng & Jordan 2000; Sutton *et al.* 2000; Williams 1992).

Recent developments in the latter group of methods include application of modern classifiers (Fern, Yoon, & Givan 2004; Lagoudakis & Parr 2003; Langford & Zadrozny 2003; Yoon, Fern, & Givan 2002). In the framework, acquisition of agent's policy is expressed via learning a classifier labeling states with optimal actions. Recent implementations of this idea have demonstrated promising performance in several domains by learning high-quality policies through high-accuracy classifiers. It should be noted, however, that in sequential decision-making the classification error is *not* the target performance measure of a reward-collecting agent. Consequently, increasing the classification accuracy may actually *lower* the policy value (Li 2004). The intuition behind this phenomenon lies with the fact that not all states are equally important in terms of preferring one action to another. Therefore, the efficiency of classification-based reinforcement learning methods can be increased by *focusing* the learning process on *more important* states, which is the problem investigated in this paper.

Related Work

Throughout this paper, the state and action spaces are denoted by $\mathcal{S}$ and $\mathcal{A}$, respectively. The agent starts from some state $s_0 \in \mathcal{S}$ according to a start-state distribution D. At each time step t, it perceives the current state s_t, takes an action $a_t \in \mathcal{A}$, receives an immediate reward $r_{t+1} \in \mathbb{R}$, and reaches the next state s_{t+1}; then it repeats the same process from state s_{t+1}. The goal of the agent is to find a policy $\pi : \mathcal{S} \mapsto \mathcal{A}$ to maximize its expected discounted long-term return starting from s_0 by following π: $\mathbf{E}\left\{\sum_{t=0}^{\infty} \gamma^k r_{t+1} \middle| s_0, \pi\right\}$. Given a policy π, we define $V^\pi(s)$, the state-value function, as the expected return received by following π from state s. Similarly, the action-value function $Q^\pi(s,a)$ is defined as the expected return received by taking action a in state s and following π thereafter. In the paper, we use *policy value* $V(\pi)$ as the measure of a policy's performance, which is defined as the expected return obtained by following π from the start state s_0 drawn according to D (Ng & Jordan 2000):

$$V(\pi) = \mathop{\mathbf{E}}_{s_0 \sim D}\{V^\pi(s_0)\}. \tag{1}$$

A reinforcement learning agent tries to *learn* the optimal policy with the maximum value: $\pi^* = \arg\max_\pi V(\pi)$. The corresponding optimal state- and action-value functions are denoted by $V^*(s)$ and $Q^*(s,a)$. Another way to measure the quality of a policy is through *policy loss*:

$$L(\pi) = V(\pi^*) - V(\pi). \tag{2}$$

Since $V(\pi^*)$ is a constant for a given problem, $\arg\max_\pi V(\pi) \equiv \arg\min_\pi L(\pi)$. Therefore, policy value and policy loss are two equivalent measures.

There are two primary approaches to computing the optimal policy. Value-function based methods approximate the optimal value function V^* assuming that the greedy policy π_V with respect to the approximation V will be close to π^* when V is close to V^*. In particular, a class of algorithms called *temporal difference learning* (TD(λ)) (Sutton 1988) implement this approach by updating the approximate value function V iteratively online.

On the other hand, policy-search methods work directly in a policy space Π and build an approximation $\widehat{\pi}^* \in \Pi$ to the optimal policy π^*. Recent examples include (Kearns, Mansour, & Ng 2000; Ng & Jordan 2000; Sutton *et al.* 2000; Williams 1992). In this paper we will focus on *classification-based* policy-search RL methods where the policy is represented as a classifier mapping states to actions.

Reinforcement Learning As Classification

When the state space is small, the policy can be stored in a lookup table with one entry per state. In such a case, *policy iteration* (Howard 1960) has been found efficient in solving MDPs. Two steps are involved in each policy iteration. In *policy evaluation*, the agent evaluates the value function of its current policy π, such as $Q^\pi(s,a)$; then in *policy improvement*, it computes the *greedy policy* π' from π by (3). This process repeats until convergence.

$$\pi'(s) = \arg\max_a Q^\pi(s,a), \quad \forall s \in \mathcal{S}. \tag{3}$$

It can be shown that $V(\pi') > V(\pi)$, unless $\pi' = \pi = \pi^*$ in which case $V(\pi') = V(\pi) = V(\pi^*)$.

If the state space is prohibitively large, however, function approximation has to be used to represent the policy or the value function. A general framework of approximation within policy iteration is known as *approximate policy iteration* (API) (Bertsekas & Tsitsiklis 1996), which is similar to policy iteration, except that (i) the policy and value functions are represented by function approximators such as neural networks (in contrast to lookup tables in policy iteration), and/or (ii) the value function is not computed exactly but is estimated. Consequently, the greedy policy $\widehat{\pi}'$ computed in the inner loop of API is an approximation to π'. Unlike π', $\widehat{\pi}'$ may be worse than the original policy π, due to the errors introduced by the approximation. By using classifiers to represent policies, API has been successfully applied in several domains (Lagoudakis & Parr 2003).

Challenges to Existing Methods

Recent implementations of classification-based API attempt to acquire a high-quality policy by employing high-accuracy classifiers (Fern, Yoon, & Givan 2004; Lagoudakis & Parr 2003; Yoon, Fern, & Givan 2002). While promising, this approach is susceptible to two problems.

First, increasing accuracy of the classifier can actually result in policies with *lower* values. This is so because in sequential decision-making the classification accuracy (i.e., the probability that a policy π outputs the optimal action) is *not* the target performance measure of a reward-maximizing agent. Consequently, *increasing* classification accuracy in the sense of supervised learning may actually *lower* the policy value. The underlying intuition is that not all states are equally important in terms of preferring one action to another (Li 2004). In playing chess, for example, a player should pay more attention to avoiding mistakes in critical positions where a wrong move will likely lead to material loss, while in many other positions suboptimal moves are less important. Therefore, it would be helpful for the agent to *focus* the learning process on more important states.

The second problem lies with the fact that decisions in some states make very little difference in terms of the expected returns. Therefore, an importance-insensitive classification-based RL agent can use learning time and resources inefficiently by trying to approximate the optimal policy in *all* states.

A similar idea has been suggested for the T-step RL problem in (Kim, Langford, & Ng 2003; Langford & Zadrozny 2003). Our work is different in that (i) our results apply to MDPs of any horizon; (ii) we define the state importance in a way that enables an exact relation with policy values; and (iii) we train only one classifier as the policy, instead of T classifiers each per time step.

Novel Approach

In this section, we formalize the idea of focused learning by introducing the novel concept of *state importance*. The policy switching theorem will be presented first; then two classes of RL problems, batch RL and online RL, are investigated. The performance advantage of our approach is guaranteed by several theorems. Due to space limitation, proofs of the theorems and detailed empirical evaluation results are omitted, and only binary-action RL problems with deterministic policies are examined. A detailed account can be found in (Li 2004).

The Policy Switching Theorem

Theorem 1 below forms the basis of our proposed methods. It established how the policy value is changed when a switch between two policies is implemented. A concept similar to $G^\pi(s, \pi \to \tau)$, defined below, was introduced in (Baird 1993) and called advantage. The following theorem parallels a result in (Kakade & Langford 2002).

Theorem 1 *Let π and τ be two arbitrary policies for a discounted, infinite-horizon MDP; $\mu_t^{\tau,D}(s)$ is the state visitation distribution, i.e., the probability that state s is visited at time t by following policy τ with start states drawn according to D. Define $d^{\tau,D}(s) = \sum_{t=0}^{\infty} \gamma^t \cdot \mu_t^{\tau,D}(s)$ and $G^\pi(s, \pi \to \tau) = Q^\pi(s, \tau(s)) - Q^\pi(s, \pi(s))$, then*

$$V(\tau) - V(\pi) = \sum_{s \in \mathcal{S}} \left(G^\pi(s, \pi \to \tau) \cdot d^{\tau,D}(s) \right). \tag{4}$$

Focused Batch Reinforcement Learning

In this section, we will consider a simpler class of RL problems called *batch reinforcement learning*, where the agent is presented with a *fixed* set of experiences and the goal is to compute a control policy *offline*. Such a learning problem is useful in the domains where online learning is not feasible (e.g., when the reward data are limited (Levner & Bulitko 2004)), and therefore a fixed set of experiences has to be acquired and used for offline policy learning. Specifically, we assume that the state space is *sampled* and the optimal action values, $Q^*(s,a)$, for these sampled states can be computed or estimated through full trajectory tree expansions (Kearns, Mansour, & Ng 2000). Indeed, such a method is computationally expensive, but it can be useful in some cases (Levner & Bulitko 2004; Wang & Dietterich 1999). A more comprehensive discussion is found in (Li 2004). In summary, the problem considered in this section is to compute a policy with the maximum value by inducing a classifier from a set of training data in the following form:

$$T_{Q^*} = \{ \langle s, Q^*(s,a) \rangle \mid s \in \mathcal{T}, a \in \mathcal{A} \}. \tag{5}$$

where $\mathcal{T} \subset \mathcal{S}$ is the sampled state space. We denote the optimal action in state s by $a^*(s)$, and the other (sub-optimal) action by $\bar{a}(s)$.

An importance-insensitive RL agent induces a classifier $\widehat{\pi}^*_{\mathsf{CI}}$ from the training set $T_{\mathsf{CI}} = \{ \langle s, a^*(s) \rangle \mid s \in \mathcal{T} \}$ by minimizing the classification error:

$$\widehat{\pi}^*_{\mathsf{CI}} = \arg\min_{\widehat{\pi}^*} \frac{\mathcal{I}(\widehat{\pi}^*(s) \neq a^*(s))}{|\mathcal{S}|}, \tag{6}$$

where $\mathcal{I}(\cdot)$ is the *indicator function* defined as:

$$\mathcal{I}(A) = \begin{cases} 1, & A \text{ is true} \\ 0, & A \text{ is false} \end{cases}.$$

We will now introduce an appropriate measure of state importance. Intuitively, a state is important from the decision-making perspective when making a wrong decision in it can have significant repercussions. Formally, the *importance* of a state s, $G^*(s)$, is defined as the difference in the optimal values of $a^*(s)$ and the other action $\bar{a}(s)$ (as previously mentioned, we focus on the binary-action case in this paper):

$$G^*(s) = Q^*(s, a^*(s)) - Q^*(s, \bar{a}(s)). \tag{7}$$

Similarly, $G^*(s, \pi)$ is defined as:

$$\begin{aligned} G^*(s,\pi) &= Q^*(s, a^*(s)) - Q^*(s, \pi(s)) \\ &= G^*(s) \cdot \mathcal{I}(\pi(s) \neq \pi^*(s)). \end{aligned} \tag{8}$$

From Theorem 1, the following corollary can be obtained.

Corollary 1 *It holds true for any policy π that*

$$L(\pi) = \sum_{s \in \mathcal{S}} \left(G^*(s,\pi) \cdot d^{\pi,D}(s) \right). \tag{9}$$

This result is still, however, of a limited practical use since the visitation distribution $\mu_t^{\pi,d}$ is usually unavailable to the agent. Thus, we instead minimize the upper bound of $L(\pi)$:

$$L(\pi) \leq \sum_{s \in \mathcal{S}} \left(G^*(s,\pi) \cdot \sum_{t=0}^{\infty} \gamma^t \right) = \frac{1}{1-\gamma} \sum_{s \in \mathcal{S}} G^*(s,\pi)$$

Consequently, a practical approach is proposed:

$$\begin{aligned} \arg\min_{\pi} L(\pi) &\approx \arg\min_{\widehat{\pi}^*} \frac{1}{1-\gamma} \sum_{s \in \mathcal{S}} G^*(s, \widehat{\pi}^*) \\ &= \arg\min_{\widehat{\pi}^*} \sum_{s \in \mathcal{S}} G^*(s, \widehat{\pi}^*). \end{aligned} \tag{10}$$

Thus, given a set of training data T_{Q^*} described in (5), the agent can first compute $G^*(s)$ for all states $s \in \mathcal{T}$ by (7), build a training set $T_{\mathsf{CS}} = \{\langle s, a^*(s), G^*(s) \rangle \mid s \in \mathcal{T}\}$, and then solve the optimization problem:

$$\widehat{\pi}^*_{\mathsf{CS}} = \arg\min_{\widehat{\pi}^*} \sum_{s \in \mathcal{S}} G^*(s, \widehat{\pi}^*). \tag{11}$$

By replacing $G^*(s,\pi)$ in (11) with (8), solving $\widehat{\pi}^*_{\mathsf{CS}}$ is turned precisely in the cost-sensitive classification problem with the *misclassification costs* conditional on individual cases (Turney 2000). Indeed, in the RL settings, s is the attribute, $a^*(s)$ is the desired class label, and $G^*(s)$ is the misclassification cost.

A problem of both theoretical and practical interest is therefore: Is it preferable to solve $\widehat{\pi}^*_{\mathsf{CS}}$ as oppose to $\widehat{\pi}^*_{\mathsf{CI}}$? Theorem 2 below provides an upper bound of the policy loss of $\widehat{\pi}^*_{\mathsf{CS}}$. In contrast, Theorem 3 establishes that $\widehat{\pi}^*_{\mathsf{CI}}$ can be arbitrarily poor in the sense that the policy loss can be arbitrarily close to its upper bound, $\sum_{s \in \mathcal{S}} G^*(s)/(1-\gamma)$.

Theorem 2 *If $\widehat{\pi}^*_{\mathsf{CS}}$ has a sufficiently high quality, i.e.,*

$$\exists \epsilon > 0, \quad \text{s.t.} \quad \frac{\sum_{s \in \mathcal{S}} G^*(s, \widehat{\pi}^*_{\mathsf{CS}})}{\sum_{s \in \mathcal{S}} G^*(s)} < \epsilon,$$

then

$$L(\widehat{\pi}^*_{\mathsf{CS}}) \quad \leq \quad \frac{\epsilon}{1-\gamma} \sum_{s \in \mathcal{S}} G^*(s).$$

Theorem 3 *Let ϵ be the classification error (6) of $\widehat{\pi}^*_{\mathsf{CI}}$, then $\forall \epsilon > 0, \forall \xi > 0$, there exists an MDP and $\widehat{\pi}^*_{\mathsf{CI}}$, s.t.*

$$L(\widehat{\pi}^*_{\mathsf{CI}}) \quad > \quad \frac{1-\xi}{1-\gamma} \sum_{s \in \mathcal{S}} G^*(s).$$

Focused Online Reinforcement Learning

In this section, we extend the idea of focused learning to the online RL problem, where the agent interacts with the environment online, has experiences in the form of $\langle s_t, a_t, r_{t+1}, s_{t+1} \rangle$, and the $Q^*(s,a)$ are *not* available for any (s,a)-pair. Our work is based on the API implementation in (Lagoudakis & Parr 2003), which will be referred to as CI-cRL (Cost-Insensitive classification-based RL) henceforth. Note that CI-cRL does not compute the exact greedy policy π' (3). Instead, it uses a learning algorithm in each iteration to construct a classifier $\widehat{\pi}'_{\mathsf{CI-cRL}}$ as the approximate greedy policy by solving:

$$\widehat{\pi}'_{\mathsf{CI-cRL}} = \arg\min_{\widehat{\pi}'} \frac{\mathcal{I}(\widehat{\pi}'(s) \neq \pi'(s))}{|\mathcal{S}|}. \tag{12}$$

Although it is guaranteed that $V(\pi') \geq V(\pi)$ where π' is the greedy policy of π, it is possible that the value of the policy constructed by CI-cRL decreases: $V(\widehat{\pi}'_{\mathsf{CI-cRL}}) < V(\pi)$, due to the errors introduced by the approximation. Thus, in order to improve the value of approximate greedy policy in each iteration, we prefer a policy whose value is close to $V(\pi')$. Similar to the batch RL case, we define the importance of a state as:

$$\begin{aligned} G^\pi(s) &= Q^\pi(s, a^*_\pi(s)) - Q^\pi(s, \bar{a}_\pi(s)) \\ &= G^\pi(s) \cdot \mathcal{I}(\widehat{\pi}'(s) \neq \pi'(s)). \end{aligned} \tag{13}$$

where $a^*_\pi(s)$ and $\bar{a}_\pi(s)$ are the greedy and non-greedy actions in s with respect to π. Intuitively, $G^\pi(s)$ measures how much additional reward can be obtained by switching the action from $\bar{a}_\pi(s)$ to $a^*_\pi(s)$ in state s, and then following the policy π thereafter. Likewise, we define

$$\begin{aligned} G^\pi(s,\tau) &= Q^\pi(s, a^*_\pi(s)) - Q^\pi(s, \tau(s)) \\ &= \begin{cases} G^\pi(s), & \tau(s) \neq a^*_\pi(s) \\ 0, & \tau(s) = a^*_\pi(s) \end{cases}. \end{aligned} \tag{14}$$

Corollary 2 *If during policy improvement a policy π is changed to $\widehat{\pi}'$ which is an approximation to the greedy policy π', and*

$$\forall s \in \mathcal{S}, \ |d^{\pi',D}(s) - d^{\widehat{\pi}',D}(s)| < \varepsilon, \tag{15}$$

then

$$V(\pi') - V(\widehat{\pi}') \leq \sum_{s \in \mathcal{S}} G^\pi(s, \widehat{\pi}') \cdot d^{\widehat{\pi}',D}(s) + \varepsilon \sum_{s \in \mathcal{S}} G^\pi(s). \tag{16}$$

This corollary bounds $V(\pi') - V(\widehat{\pi}')$ via $G^\pi(s, \widehat{\pi}')$ in each individual state. Note that π' and $V(\pi')$ are constant if π is fixed. Therefore, $\arg\max_{\widehat{\pi}'} V(\widehat{\pi}') \equiv \arg\min_{\widehat{\pi}'} [V(\pi') - V(\widehat{\pi}')]$. In other words, we can minimize $V(\pi') - V(\widehat{\pi}')$ instead of maximizing $V(\widehat{\pi}')$. The representation (16) is, however, still of a limited practical value since $d^{\widehat{\pi}',D}(s)$ and ε are usually unavailable to the agent. Thus, a similar practical approximation is proposed:

$$\begin{aligned} \widehat{\pi}'_{\mathsf{CS-cRL}} &= \arg\min_{\widehat{\pi}'} \sum_{s \in \mathcal{S}} G^\pi(s, \widehat{\pi}') && (17) \\ &\approx \arg\min_{\widehat{\pi}'} \left[V(\pi') - V(\widehat{\pi}') \right]. && (18) \end{aligned}$$

We prefer to solve $\widehat{\pi}'_{\mathsf{CS-cRL}}$ as it takes state importance into account which is highly related to the policy value according to Corollary 2. Again, by replacing $G^\pi(s, \pi)$ in (17) with (13), solving $\widehat{\pi}'_{\mathsf{CS-cRL}}$ is turned precisely into a cost-sensitive classification problem, where s is the attribute, $a^*_\pi(s)$ is the desired class label, and $G^\pi(s)$ is the misclassification cost. The analysis above becomes the basis of CS-cRL (Cost-Sensitive classification-based RL) in (Li 2004), which is a cost-sensitive version of CI-cRL. Advantage of a similar idea was also suggested by (Fern, Yoon, & Givan 2004) independently, but lacks analysis.

The following two theorems establish (i) whether (18) is reasonable, and (ii) whether it is preferable to solve $\widehat{\pi}'_{\mathsf{CS-cRL}}$ as opposed to $\widehat{\pi}'_{\mathsf{CI-cRL}}$. Theorem 4 states that if ε is not large and if $\widehat{\pi}'_{\mathsf{CS-cRL}}$ is of a sufficiently high quality, then it will be close to π' in terms of policy value. In contrast, Theorem 5 asserts that as long as $\widehat{\pi}'_{\mathsf{CI-cRL}}$ has a non-zero classification error, the difference $V(\pi') - V(\widehat{\pi}'_{\mathsf{CI-cRL}})$ can be arbitrarily close to its upper bound, $\sum_{s\in\mathcal{S}} G^\pi(s)/(1-\gamma)$.

Theorem 4 *If $\widehat{\pi}'_{\mathsf{CS-cRL}}$ has a sufficiently high quality, i.e.,*

$$\exists \epsilon > 0, \quad s.t. \quad \frac{\sum_{s\in\mathcal{S}} G^\pi(s, \widehat{\pi}'_{\mathsf{CS-cRL}})}{\sum_{s\in\mathcal{S}} G^\pi(s)} < \epsilon,$$

and assume the notation in (15), then

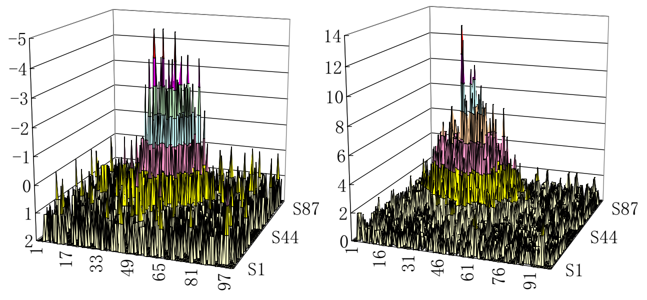
$$V(\pi') - V(\widehat{\pi}'_{\mathsf{CS-cRL}}) \quad \le \quad (\frac{\epsilon}{1-\gamma} + \varepsilon) \sum_{s\in\mathcal{S}} G^\pi(s).$$

Theorem 5 *Let ϵ be the classification error (12) of $\widehat{\pi}'_{\mathsf{CI-cRL}}$, then $\forall \epsilon > 0$, $\forall \xi > 0$, there exists an MDP and $\widehat{\pi}'_{\mathsf{CI-cRL}}$, s.t.*

$$V(\pi') - V(\widehat{\pi}'_{\mathsf{CI-cRL}}) \quad > \quad \frac{1-\xi}{1-\gamma} \sum_{s\in\mathcal{S}} G^\pi(s).$$

Empirical Evaluation

In this section, we report the empirical results in a 2D grid-world domain to evaluate the advantages of importance-sensitive classification-based reinforcement learning. This domain is a variation on that in (Dietterich & Wang 2002). We consider grid-worlds of $N \times N$ cells with no walls. Each state is represented as a tuple, (x, y) where $x, y \in \{1, \cdots, N\}$. Start states are randomly chosen in the leftmost column, $(1, y)$. At every step, the agent has two possible actions, north-east and south-east, which deterministically take the agent from the position (x, y) to $(x+1, y+1)$ or $(x+1, y-1)$, respectively. If the agent attempts to step out of the grid-world (i.e., $y-1$ or $y+1$ exceeds the range $[1, N]$), it will continue to move along the boundary. An episode is terminated when the agent reaches the rightmost column.



(a) reward (b) importance

Figure 1: A typical distribution of immediate reward and importance values in the 2D grid-world.

Experiment I: Batch Learning

In the first experiment, $N = 100$. In order to make the optimal policy have a positive value (which is more intuitive), we assigned a small positive reward of 2 to each action (note that this does not change any policy). Additionally, 3000 units of negative rewards, each with a value of -1, are randomly positioned in the grid-world according to a distribution scheme. If more than one reward was placed in the same cell, then the rewards were accumulated. The distribution scheme used in our experiment was a mixture of a uniform distribution and a two-dimensional Gaussian distribution centered at the cell $(50, 50)$ with the variance $\sigma = 10$ in each dimension. The uniform and Gaussian distributions carried the weights of 0.4 and 0.6, respectively. Formally, for all $x, y \in \{1, 2, \cdots, 100\}$, the weight of cell (x, y) is computed by:

$$d(x,y) = \frac{0.4}{100 \times 100} + \frac{0.6}{2\pi\sigma} \exp\left\{ \frac{(x-50)^2 + (y-50)^2}{2\sigma^2} \right\}.$$

Figure 1 illustrate an example of the reward and importance value distribution in the 100×100 grid world, respectively. This problem is a non-discounted finite-horizon MDP. The goal of the agent is to learn a policy to maximize its cumulative rewards by avoiding the negative rewards throughout the grid-world.

Two methods are compared. The first one, called CI, is the baseline algorithm that solves the importance-insensitive optimization problem (6); the other, called CS, solves the importance-sensitive classification problem (11). We used a feed-forward multi-layer artificial neural network (ANN) as the classifier. The topology and parameters of the ANN were fixed throughout the experiments, so that both classifiers, $\widehat{\pi}^*_{\mathsf{CI}}$ and $\widehat{\pi}^*_{\mathsf{CS}}$, have the same learning ability. Cost-sensitivity in classification can be achieved in different ways (Zadrozny & Langford 2003). We adopted simple resampling which samples training data according to the distribution:

$$\Pr(\text{select state } s \in \mathcal{T} \text{ for training}) = \frac{G^*(s)}{\sum_{s'\in\mathcal{T}} G^*(s')}.$$

Each of the algorithms was evaluated along the two performance measures:

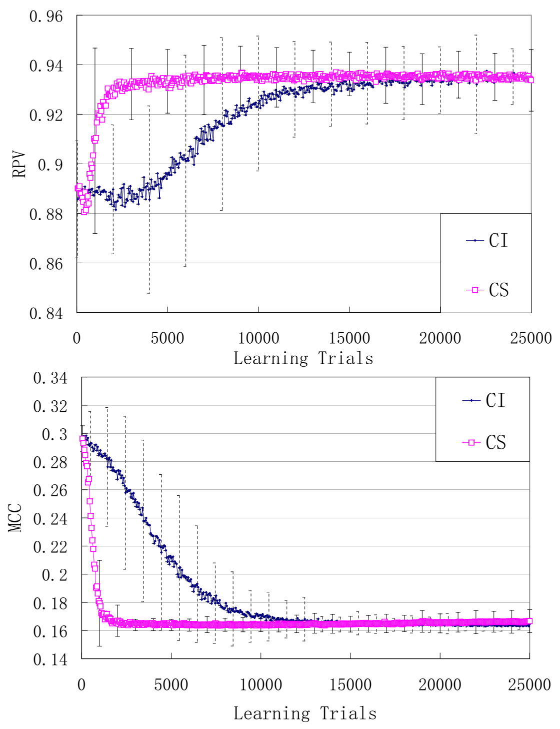


Figure 2: Policy value and misclassification cost in the grid-world experiment, with standard deviations plotted.

- *Relative Policy Value* (RPV)[1], defined as $\mathcal{V}(\pi)/\mathcal{V}(\pi^*)$;
- *Average Misclassification Cost* (MCC) over the entire state space, defined as $\sum_s G^*(s, \pi)/|\mathcal{S}|$.

The true objective of our reinforcement learning agent is to maximize RPV while the classifier's objective is to reduce MCC.

In the experiments, $\mathcal{T}$ contained 1000 states randomly sampled from the original state space: 700 of them were used for training, and the other 300 were used for validation to guard against overfitting. Ten random 100×100 grid-worlds were generated according to the reward distribution, and 20 learning sessions were conducted in each of them resulting in a total of 400 trials for each algorithm. The results are plotted in Figures 2 with standard deviation as the error bars. Several observations are in order.

First, note that the importance-sensitive algorithm CS increased the policy value substantially faster than the importance-insensitive CI. The significant advantage was observed early in the learning process. Note that the importance values of the grid-world states vary significantly, as shown in Figure 1(b). For many states, the importance is negligible or even zero; on the other hand, some of the states have much greater importance which makes them more significant in affecting the policy value and deserve more attention in learning. This observation leads to the conjecture that CS can learn even better when there is a high variance in the importance distribution over the state space.

Second, we note that the importance-sensitive learner CS was able to reduce the average misclassification cost (MCC) faster than CI [2]. The superior ability to reduce MCC appears to be the reason for a more rapid RPV improvement exhibited by CS.

Finally, we find that the standard deviation of the policies obtained by CS is lower (Figure 2(a)). This seems to be due to the fact that CS pays more attention to more important states.

[1]NB: since $V^*(\pi)$ is guaranteed to be positive, higher RPV indicates a better policy.

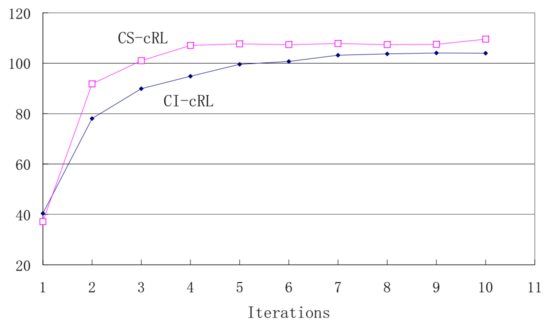


Figure 3: Policy values in the first ten policy iterations, averaged over 50 runs.

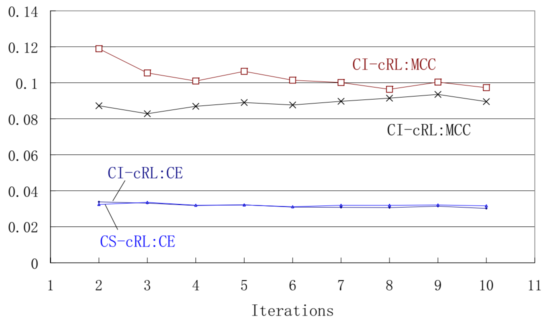


Figure 4: (Weighted) classification errors on the training set in the first ten policy iterations, averaged over 50 runs.

Experiment II: Online Learning

For the online learning experiment, smaller mazes of 50×50 cells were used. 2000 pieces of rewards each with a value of 1 were placed in the state space according to a similar reward distribution. The center of the Gaussian distribution was moved to the cell $(25, 25)$ and the variance was $\sigma = 5$. The training state set $\mathcal{T}$ contained 500 randomly selected states. We used support vector machines[3] as the classifiers. In the original form, SVMs are cost insensitive and do not take the misclassification costs into account. We again used the simple resampling technique as in the previous section.

Three metrics were used to evaluate the approximate greedy policy $\widehat{\pi}'$ in each iteration:

- *Policy Value* (PV): $\mathcal{V}(\widehat{\pi}')$;
- *Classification Error* (CE): $\sum_{s \in \mathcal{S}} \mathcal{I}(\widehat{\pi}' \neq \pi')/|\mathcal{S}|$.
- *Weighted Classification Costs* (WCE), the classification error weighted by the state importance $G^{\pi}(s)$:

[2]Note that MCC is different from the standard uniform classification error insomuch as it is weighted by state importance (i.e., the misclassification cost).

[3]We used LS-SVMlab (Suykens *et al.* 2002), a publicly available implementation of SVMs in the MATLAB environment.

Iteration	1	2	3	4	5
Advantage	–	13.6±15.7	11.1±17.4	12.2±14.3	8.1±11.0
Prob(better)	–	81%	76%	86%	76%

Table 1: Statistics of the policy value improvements of CS-cRL obtained from 50 runs of experiment.

$\sum_{s\in\mathcal{S}} G^{\pi}(s)\mathcal{I}(\widehat{\pi}' \neq \pi') / \sum_{s\in\mathcal{S}} G^{\pi}(s)$.

In the experiment, a maze was randomly generated and 50 runs of experiments were conducted. In each run, 500 states were randomly selected as the training state set T. The average policy value of CI-cRL and CS-cRL in the first 10 API iterations are shown in Figures 3.

First, observe that CS-cRL successfully increased the policy values by 10–20% on the second to fifth iterations[4], compared with CI-cRL. In most runs, CS-cRL did produce better policies. Statistics for the second to the fifth iterations are given in Table 1, including the average policy value improvement of CS-cRL (with variances) defined as $\mathcal{V}(\widehat{\pi}'_{\mathsf{CS-cRL}}) - \mathcal{V}(\widehat{\pi}'_{\mathsf{CI-cRL}})$, as well as the probability that the true value of CS-cRL's policy was higher.

Next, we examine the relations between CE/WCE and PV in the experiments to see whether it is helpful to focus on more important states. Figure 4 gives the results (CE/WCE) on the training set. It is clear that CS-cRL managed to decrease the WCE while it has an almost identical performance with CI-cRL in terms of reducing CE. These results together with Figure 3 suggest that the proposed method is able to focus on more important states, which appeared to be the reason why important-sensitive learning is beneficial.

Conclusions and Future Work

In this paper we investigated the problem of focusing attention in classification-based RL. Two classes of RL problems were investigated and two suitable state importance measures were defined. The advantages, as supported by several theoretical and empirical results, include faster convergence to better policies. The promising initial results open several avenues for future research. First, by placing a threshold on the importance level of a training state we can further prune down the training data set. An immediate advantage is a reduction of training time. However, it remains unknown whether this reduction will cause overfitting. Another direction for future research is to investigate other ways of defining state importance. Note that Corollary 2 depends on the factor ε, and approximation steps were taken in (10) and (18). It is possible to find better measures of state importance and/or better approximation (e.g., estimating $d^{\pi,D}(s)$) leading to more efficient RL algorithms. Finally, we are presently extending the idea of focused learning to value-function based RL methods, such as SARSA (Sutton & Barto 1998) and Q-learning (Watkins 1989).

Acknowledgements We gratefully appreciate the contributions from Rich Sutton, Ilya Levner, and Greg Lee. Doina Precup and the reviewers provided several helpful suggestions that improved the paper. Ron Parr drew our attention to independent work by Fern et al., which suggested a similar idea to CS-cRL. This research is supported by the Alberta Ingenuity Center for Machine Learning (AICML), the University of Alberta, and NSERC.

[4]Note that the policies in the first iteration were random.

References

Baird, L. 1993. Advantage updating. Technical Report WL-TR-93-1146, Wright-Patterson Air Force Base, OH.

Bertsekas, D. P., and Tsitsiklis, J. 1996. *Neuro-Dynamic Programming*. Athena Scientific.

Dietterich, T. G., and Wang, X. 2002. Batch value function approximation via support vectors. In *Advances in Neural Information Processing Systems*, volume 14.

Fern, A.; Yoon, S.; and Givan, R. 2004. Approximate policy iteration with a policy language bias. In *Advances in Neural Information Processing Systems*, volume 16.

Howard, R. A. 1960. *Dynamic Programming and Markov Processes*. Cambridge, MA: MIT Press.

Kakade, S., and Langford, J. 2002. Approximately optimal approximate reinforcement learning. In *Proc. of ICML*.

Kearns, M.; Mansour, Y.; and Ng, A. Y. 2000. Approximate planning in large POMDPs via reusable trajectories. In *Advances in Neural Information Processing Systems*, volume 12.

Kim, H. J.; Langford, J.; and Ng, A. Y. 2003. Policy search via supervised learning. In *Machine Learning Reductions Workshop*.

Lagoudakis, M., and Parr, R. 2003. Reinforcement learning as classification: Leveraging modern classifiers. In *Proc. of ICML*.

Langford, J., and Zadrozny, B. 2003. Reducing T-step reinforcement learning to classification. In *Machine Learning Reductions Workshop*.

Levner, I., and Bulitko, V. 2004. Machine learning for adaptive image interpretation. In *Proc. of IAAI*.

Li, L. 2004. Focus of attention in reinforcement learning. Master's thesis, Department of Computing Science, University of Alberta, Edmonton, Alberta, Canada. In preparation.

Ng, A. Y., and Jordan, M. 2000. PEGASUS: A policy search method for large MDPs and POMDPs. In *Proc. of UAI*.

Sutton, R. S., and Barto, A. G. 1998. *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press.

Sutton, R. S.; McAllester, D.; Singh, S.; and Mansour, Y. 2000. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*, volume 12.

Sutton, R. S. 1988. Learning to predict by the methods of temporal differences. *Machine Learning* 3:9–44.

Suykens, J.; Gestel, T. V.; Brabanter, J. D.; Moor, B. D.; and Vandewalle, J. 2002. *Least Squares Support Vector Machines*. Singapore: World Scientific Pub. Co.

Turney, P. 2000. Types of cost in inductive concept learning. In *The ICML-00 Workshop on Cost-Sensitive Learning*, 15–21.

Wang, X., and Dietterich, T. G. 1999. Efficient value function approximation using regression trees. In *IJCAI-99 Workshop on Statistical Machine Learning for Large-Scale Optimization*.

Watkins, C. 1989. *Learning from Delayed Rewards*. Ph.D. Dissertation, King's College, University of Cambridge, UK.

Williams, R. J. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning* 8:229–256.

Yoon, S.; Fern, A.; and Givan, R. 2002. Inductive policy selection for first-order MDPs. In *Proc. of UAI*.

Zadrozny, B., and Langford, J. 2003. Cost-sensitive learning by cost-proportionate example weighting. In *Proc. of ICDM*.