Uncertainty Estimate with Pseudo-Entropy in Reinforcement Learning

ZHANG Ping and Stéphane Canu

(URA CNRS 817, HEUDIASYC, UTC, BP 529, 60205 Compiègne, France)

Abstract: A reinforcement learning (RL) system interacts with an unrestricted, unknown environment. Its goal is to maximize cumulative rewards, to be obtained throughout its limited, unknown lifetime. One of difficulties for a RL system is that reward signal is sparse, specially for RL system with very delayed rewards. In this paper, we describe an algorithm based on a model of the state's uncertainty estimate. It uses efficiently reward information stored in value function. The experiments show that the algorithm has a very good performance.

Key words: reinforcement learning; Q-learning; entropy estimate; uncertainty; Markov decision

1 Introduction

In Reinforcement learning (RL) scenario, the agent must choose an output to generate in response to each input. The reinforcement signal (reward) it receives indicates only how successful that output was; it carries no information about how successful other outputs might have been. RL has been studied extensively and its properties are well known, seen in [1, 2], It has been shown also to perform well in Markov domains, such as [3～7].

The RL is difficulty for reward signal is sparse and often very delayed. As a quality signal, reward carries less information about environment. A very important issue in RL is efficient use of reward. Rewards are generally stored in form of value function, for example Q-value. The algorithm proposed in this paper takes use of Q-value to estimate state's uncertainty and then uses this estimate to modify Q-value. The algorithm is compared with Q-learning[2] and SCIQ-learning[6] in empirical trials and is shown to have very good performance.

2 State's Uncertainty Estimate with Pseudo-Entropy

2.1 Reinforcement Learning

The general RL problem is typically stated as finding a policy that maximizes expected discounted future reinforcement. A policy π is a mapping from S (a state space) to A_i (a set of action in state i). The evaluation function maintains an estimate, V_i, of optimal value function in each state i. It is updated as follows:

$$V_i \leftarrow (1-\alpha)V_i + \alpha(r + \gamma V_j) \tag{1}$$

where r is the actual reinforcement value received for taking action k in state i, j is the next state, α, $0 < \alpha < 1$, is a learning rate, and γ is a discounted factor.

Manuscript received Fed. 26, 1996, revised Oct. 30, 1996.

Q-learning stores and updates the espected discounted reinforcement estimates, called Q-values, for each state-action pair. The agent's policy is to choose the action with the maximal Q-value available from the current state. Let $Q_{i,k}$ be the expected discounted reinforcement for taking action k in state i and continuing thereafter with the optimal policy. The Q-values are updated as follows:

$$Q_{i,k} \leftarrow (1-\alpha)Q_{i,k} + \alpha(r + \gamma \max_{k' \in A_j} Q_{j,k'}). \tag{2}$$

Given the Q-values, there is a policy defined by taking, in any state i, the action k that maximizes $Q_{i,k}$. It is called the greedy policy. For the sake of exploration, one can choose actions randomly according to the Boltzmann distribution. Both policies are used in this work.

2.2 State's Uncertainty Estimate

The basis of RL is a class of stochastic optimal control problems called Markov decision problems (MDP). A MDP is defined in terms of a discrete-time stochastic dynamic system with finite state set $S = \{1, \cdots, n\}$. At each time step, a controller observes the system's current state and selects an action, which is executed by being applied as input to the system. If i is the observed state, then the action is selected from a finite set A_i of admissible actions. When the controller executes action $k \in A_i$, the system's state at the next time step will be j with state-transition probability $P_{ij}(k)$. The application of action k in state i at time t incurs an immediate reward $r(t)$.

In information theory seen in [8], the entropy is a measure of uncertainty of a random variable. Let X be discrete random variable with alphabet χ and probability mass function $p(x) = Pr\{X = x\}, x \in \chi$. The entropy $H(X)$ of a discrete random variable X is defined by

$$H(X) = -\sum_{x \in \chi} p(x) \log p(x). \tag{3}$$

The maximum value of the entropy is achieved when all $p(x)$'s are equal, the minimum when one of $p(x)$'s is 1, and $0 \log 0 = 0$ is easily justified by continuity since $x \log x \to 0$ as $x \to 0$.

For the MDP, the maximun-likelihood estimate of state-transition probabilities needs extra memory. So we propose to use Q-values for this estimate as follows:

$$\hat{p}_{ij}^{k}(t) = Q_{ik}(t) \Big/ \sum\nolimits_j Q_{ij}(t). \tag{4}$$

In fact, this is not a real probability estimate. We will see in the following that the purpose is just to take use of it for pseudo-entropy estimate of a state.

Then we define pseudo-entropy estimate (PEE) of the state i at time t using $\hat{p}_{ij}^{k}(t)$ described in (4) as follows:

$$\hat{H}^{t}(i) = -\sum_{j=1}^{m} \hat{p}_{ij}^{k}(t) \log(\hat{p}_{ij}^{k}(t)) \tag{5}$$

where m is the number of the actions available in state i.

We take in the state i the PEE of the following state j as a bias of the value V_j of the state j, since we think that the really achieved value of state j depends on its certainty. So we propose the following updating rule associated to Q-learning:

$$Q_{i,k} \leftarrow (1-\alpha)Q_{i,k} + \alpha(r + \gamma \max_{k' \in A_i} Q_{j,k'} - \hat{H}(j)). \tag{6}$$

$\hat{H}(j)$ is used in (6), since bigger is the uncertainty of the state j, smaller is the certainty of taking the maximum benefice from this state. This algorithm is noted PEQ-learning because the pseudo-entropy estimate is combined with Q-learning.

3 Simulations

3.1 Pole Balancing Problem

The pole balancing problem (Fig. 1) is token for examining the proposed reinforcement algorithm. The objective is to learn to push the cart left or right so as to keep the pole balanced more or less vertical above the cart, and also to keep the cart from colliding with the ends of track. The learner has access to the state vector $(x,\dot{x},\theta,\dot{\theta})$ at each time step and can select ont of two actions, a rightward or leftward force on the cart. If the pole falls over more than 12 degrees from vertical, or if the cart hits the track boundary, a failure is said to occur.

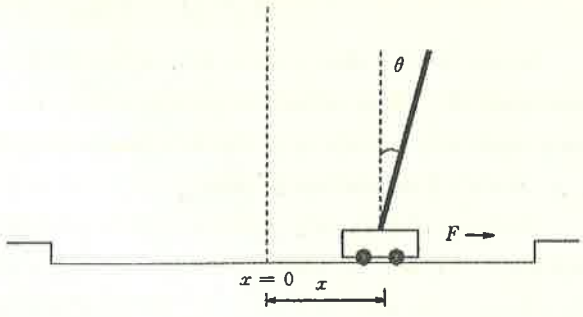


Fig. 1 Pole balancing system

The representation used in experiments is reported as follows:

$$\ddot{\theta}_t = \frac{g\sin\theta_t + \cos\theta_t\left[\dfrac{-F_t - ml\dot{\theta}_t^2\sin\theta + \mu_c\mathrm{sgn}(\dot{x}_t)}{m_c+m}\right] - \dfrac{\mu_p\dot{\theta}_t}{ml}}{l\left[\dfrac{4}{3} - \dfrac{m\cos^2\theta_t}{m_c+m}\right]}, \tag{7}$$

$$\ddot{x}_t = \frac{F_t + ml[\dot{\theta}_t^2\sin\theta_t - \ddot{\theta}_t\cos\theta_t] - \mu_c\mathrm{sgn}(\dot{x}_t)}{m_c+m}, \tag{8}$$

where the parameters are the same as used in [3]. All rewards are zero except upon failure, when a reward of negative one is delivered.

A series of runs are carried out, where each run consists of a sequence of trials. Each trial is terminated when a failure occurs. Each run consists of a number of trials until the system remains balanced for more than 100000 time steps, in which case the run is terminated. The learning rate $\alpha = 0.2$ is picked that seemed to give the best result. The discounted factor, γ, is set to 0.95 for both methods.

Fig. 2 shows the accumulated time steps measured against the number of the failures (trials) for PEQ-learning and Q-learning. It is the average of 5 runs. In terms of the accumulated time steps until failure, PEQ-learning achieves a higher level of performance than Q-

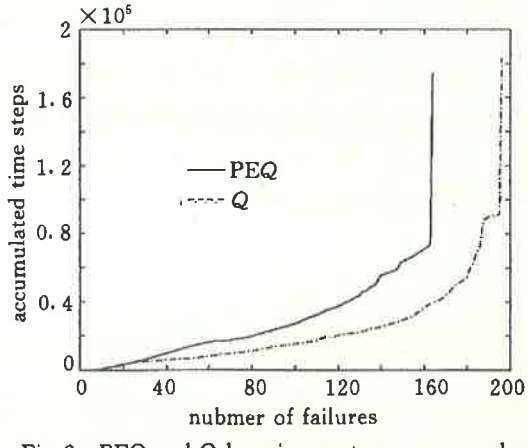


Fig. 2 PEQ and Q learning systems, measured against the number of the failures

learning.

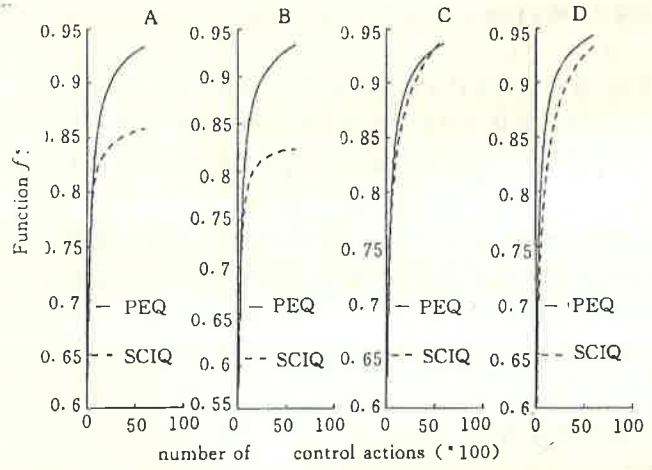
3.2 A MDP Example

A MDP with 5 states and 3 actions is taken as an example. The arrays of transition probabilities and rewards are described in detail seen in [6]. The relative frequency coefficient is defined to be

$$f^*(t) = \frac{1}{t-1}\sum_i n_i^*(t), \tag{9}$$

which gives the average number of optimal decisions made before time t, where n_i^* is the number of times the optimal action was taken in state i.

SCIQ is an algorithm proposed in [6]. The experiment is a comparison between PEQ and Q, PEQ and SCIQ. Fig. 3 shows the evolution of the relative frequency coefficient, $f^*(t)$, defined by (9). Each graph is the average of hundred simulation experiments. Each experiment consists of six thousand control actions. The graphs in Fig. 3 show that, in this learning task, the PEQ achieves a higher level of performance after a given number of control actions than Q-learning and a slightly better than SCIQ.



Panel A, B: comparison between PEQ and Q.

Panel C, D: comparison between PEQ and SCIQ.

Panel A, C: using greedy policy.

Panel B, D: using Boltzmann distribution.

Fig. 3 Graphs of $f^*(t)$, for PEQ (solid line) and Q, SCIQ (dashed line).

4 Conclusion

The algorithm proposed here, called PEQ-learning, is of interest because of being a new idea for an agent to explore in RL environment, although the entropy is not a new definition. The results of the experiments show that for the learning tasks simulated in this work, PEQ-learning achieves a higher level of performance than Q-learning. There exists still the great possibility to develop significantly PEQ-learning. Our future work will be focused on finding more efficient formulation with pseudo-entropy estimate for RL, and then use it to more difficult learning tasks.

References

1 Barto, A., Bradtke, S. and Singh, S.. Learning to act using real-time dynamic programming. Artificial Intelligence,

Elsevier Science B. V., 1995, 72(1):81—138

2 Watkins, C.. Learning from delayed rewards. PhD Thesis. Cambridge University, Cambridge, England, 1989

3 Barto, A., Sutton, R. and Anderson, C.. Neuronlike elements that can solve difficult learning control problems. IEEE Trans. Syst. Man, and Cybern, 1983,13(4):835—846

4 Lin, L.. Self-improving reactive agents based on reinforcement learning, planning and teaching. Machine Learning, Kluwer Academic, Boston, 1992,8(3/4): 293—321

5 Sutton, R.. Integrated modeling and control based on reinforcement learning and dynamic programming. Advances in Neural Information Processing Systems. Morgan Kaufmann, 1991,3:471—478

6 Zhang, P. and Canu, S.. Self-Confidence Increasing Q-learning. Proceedings of Neural Networks and Their Applications, Marseilles, 1994,245—254

7 Zhang, P. and Canu, S.. Indirect Adaptive Exploration in Entropy Based Reinforcement Learning. Proceedings of ICANN95, Paris, 1995,175—182

8 Cover, T. and Thomas, J.. Elements of Information Theory. John Wilpeqy & Sons, INC, 1991

在加强型学习系统中用伪熵进行不确定性估计

张 平 斯特凡·卡纽

(国家科研中心 817 号,启发与诊断实验室·贡比涅科技大学·法国)

摘要:加强型学习系统是一种与没有约束的、未知的环境相互作用的系统.学习系统的目标在于最大可能地获取累积奖励信号.这个奖励信号在有限、未知的生命周期内由系统所处的环境中得到.对于一个加强型学习系统,困难之一在于奖励信号非常稀疏,尤其是对于只有时延信号的系统.已有的加强型学习方法以价值函数的形式贮存奖励信号,例如著名的 Q-学习.

本文提出了一个基于状态的不确定性估计模型的方法.这个算法有效地利用了存贮于价值函数中的奖励信息.它同时适用于带有立即奖励和时延奖励信号两种情况.实验结果表明,本文的算法具有很好的学习行为.

关键词:加强型学习;Q-学习;熵估计;不确定性;马尔柯夫过程

本文作者简介

ZHANG Ping received his B. Sc. degree in 1982 in computer science and his M. Sc. degree in 1986 in automatic control both from Northeast Institute of Heavy Machinery, China. From 1986 to 1991 he was an assistant professor and then a lecturer both at the Department of Computer Engineering at Northeast Institute of Heavy Machinery. He is currently working for has Ph. D degree in the area of system control at Université de Technologie de Compiègne, France. His current research interest is in dynamic system control, machine learning, artificial intelligence, neural networks.

Stéphane Canu received his PhD degree in System Control from Compiegne University of Technology in 1986. He joined the faculty of the department of Computer Science at Compiègne University of technology in 1987. He is currently Associate professor and responsible of Heudiasyc devision of the CNRS laboratory. His current research interest is in learning theory for neural networks and in the application of neural networks to control.