Adaptive Exploration using Stochastic Neurons

Michel Tokic[1,2] and Günther Palm[1]

[1] Institute of Neural Information Processing, University of Ulm, Germany
[2] Institute of Applied Research, University of Applied Sciences Ravensburg-Weingarten, Germany

Abstract. Stochastic neurons are deployed for efficient adaptation of exploration parameters by gradient-following algorithms. The approach is evaluated in model-free temporal-difference learning using discrete actions. The advantage is in particular memory efficiency, because memorizing exploratory data is only required for starting states. Hence, if a learning problem consist of only one starting state, exploratory data can be considered as being global. Results suggest that the presented approach can be efficiently combined with standard off- and on-policy algorithms such as Q-learning and Sarsa.

Keywords: reinforcement learning, exploration/exploitation

1 Introduction

One of the most challenging tasks in reinforcement learning (RL) is balancing the amount of exploration and exploitation [1]. If the behavior of an agent is too exploratory, the outcome of randomly selected bad actions can prevent from maximizing short-term reward. In contrast, if an agent is too exploitative, the selection of only sub-optimal actions prevents from maximizing long-term reward, because the outcome of true optimal actions is underestimated. Conclusively, the optimal balance is somewhere in between, dependent on many parameters such as the learning rate, discounting factor, learning progress, and of course on the learning problem itself.

Many different approaches exist for trading off exploration and exploitation. Based on a single exploration parameter, some basic policies select random actions either equally distributed (ε-Greedy) or value sensitively (Softmax) [1], or by a combination of both [2], with the advantage of not requiring to memorize any exploratory data. In contrast, approaches utilizing counters in every state (exploration bonuses) direct the exploration process towards finding the optimal policy in polynomial time under certain circumstances [3, 4]. Nevertheless, basic policies can be effective having a proper exploration parameter configured, which has been successfully shown for example in board games with huge discrete state spaces like Othello [5] or English Draughts [6]. For such state spaces, utility functions are hard to approximate, and conducting experiments for determining a proper exploration parameter can be time consuming. A non-convergent counter function is even harder to approximate than a convergent utility function [7]. Interestingly, Daw et al. revealed in biologically-motivated studies on exploratory

decisions in humans that there is *[...] no evidence to justify the introduction of an extra parameter that allowed exploration to be directed towards uncertainty (softmax with an uncertainty bonus): at optimal fit, the bonus was negligible, making the model equivalent to the simpler softmax* [8]. However, the search for an appropriate exploration parameter for such policy remains.

In the following, a stochastic neuron model [9] is deployed for adapting the exploration parameter of basic exploration policies instead of tuning such parameter by hand in advance. We evaluate the algorithm in discrete and continuous state spaces using variants of the cliff-walking and mountain-car problems.

2 Methodology

The learning problems considered in this paper can be described as Markovian Decision Processes (MDP) [1], which basically consist of a set of states, $\mathcal{S}$, and a set of possible actions within each state, $\mathcal{A}(s) \in \mathcal{A}, \forall s \in \mathcal{S}$. A stochastic transition function $\mathcal{P}(s, a, s')$ describes the (stochastic) behavior of the environment, i.e. the probability of reaching successor state s' after selecting action $a \in \mathcal{A}(s)$ in state s. The selection of an action is rewarded by a numerical signal from the environment, $r \in \Re$, used for evaluating the utility of the selected action. The goal of an agent is finding an optimal policy, $\pi^* : \mathcal{S} \rightarrow \mathcal{A}$, maximizing the cumulative reward. In the following, it is allowed for $\mathcal{S}$ to be continuous, but assumed that $\mathcal{A}$ is a finite set of actions. Action-selection decisions are taken at regular time steps, $t \in \{1, 2, \ldots, T\}$, until a maximum number of T actions is exceeded or a terminal state is reached.

2.1 Learning Algorithms

In reinforcement learning, behavior can be derived from a utility function estimating so far learned knowledge as the expected and discounted future reward, $Q(s, a) = E\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s, a_t = a\right\}$, for taking action a in state s [1]. If no model of the environment is present, utility functions must be sampled from observations of the interaction between the agent and its environment, which is also known as *temporal-difference learning*. Two commonly used learning algorithms are Sarsa [7] and Q-learning [10], where the technical difference between both algorithms is the kind of successor-state information used for evaluating action a_t in state s_t. Sarsa includes the discounted value of the actual selected action in the successor state, for which reason it is classified being an on-policy algorithm. In contrast, Q-learning includes the discounted value of the estimated optimal action in the successor state for which reason it is classified being an off-policy algorithm. On-policy algorithms have the advantage of including into Q respective costs from stochastic action-selection policies, but have in turn no convergence guarantee, except when the policy has a greedy behavior[3] [1].

For sampling Q accurately, a proper trade-off between exploration and exploitation is required. Basic exploration policies such as ε-Greedy [10] select a

[3] i.e. taking in state s an action having the highest estimated utility $Q(s, a)$.

certain amount of random actions with regard to an exploration rate ε. A disadvantage of such policy is that exploration actions are selected equally distributed among all possible actions, which might cause the income of high negative rewards from several bad actions, even if their true utility is correctly estimated. For this, another basic policy is selecting actions according to their weighting in a Boltzmann distribution (the Softmax policy), which also takes so far estimated utility into account [1]. A known problem of Softmax is that it *[...] has large problems of focusing on the best actions while still being able to sometimes deviate from them* [2]. For this, Wiering proposed to combine Softmax with ε-Greedy into the *Max-Boltzmann Exploration* (MBE) rule [2], which selects exploration actions according to Softmax instead of being equally distributed.

A drawback of the above mentioned basic exploration policies is that an exploration parameter (ε or τ, or both for MBE) needs to be found. Such parameter varies dependent on the learning problem, and typically an exploratory behavior is desired at the beginning of the learning process, when everything is unknown (in model-free RL). For this, some applications make use of an decreasing exploration rate [11], but which is known as to be inefficient for non-stationary environment responses. Other approaches such as the VDBE-Softmax policy dynamically adapt a state-dependent exploration rate, $\varepsilon(s)$, based on learning progress measured as fluctuations in Q [12].

3 Exploration Control using Stochastic Neurons

Finding a near optimal exploration parameter by trial-and-error can be a very time consuming task, and conclusively it is desired having algorithms adapting this parameter based on current operating conditions. The proposed idea is adapting the parameter, a_e, of an action-selection policy, $\pi(a_e, \cdot, \cdot)$, towards improving the future outcome of π with regard to a performance measure ρ. For maximizing ρ in the future, we deploy Williams' *"REINFORCE with multiparameter distributions"* algorithm using a stochastic neuron model [9]. The input to such neuron is a weighted parameter vector θ, from which the neuron determines an adaptable stochastic scalar as its output, i.e. reinforcement learning in continuous action spaces. However, our investigated domain consists of discrete actions, thus the algorithm is applied for adapting the continuous-valued exploration parameter, i.e. **R**EINFORCE **E**xploration-Parameter **C**ontrol (REC). For example, if $\pi(a_e, \cdot, \cdot)$ is an ε-Greedy policy, the exploration parameter a_e refers to the exploration rate ε, i.e. $a_e \equiv \varepsilon$.

Assuming one starting state, the exploration parameter a_e is drawn at the beginning of an episode (being valid over the whole episode) from a Gaussian distribution, $a_e \sim \mathcal{N}(\mu, \sigma)$, whose density function is given by

$$g(a_e, \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(a_e-\mu)^2/2\sigma^2} \quad . \tag{1}$$

Let θ denote the vector of adaptable parameters consisting of

$$\theta = \begin{pmatrix} \mu \\ \sigma \end{pmatrix} \quad . \tag{2}$$

At the end of episode i, the components of θ are adapted towards the gradient with regard to the outcome ρ of the current episode

$$\theta_{i+1} \approx \theta_i + \alpha \nabla_\theta \rho \ . \tag{3}$$

For improving the future performance of $\pi(a_e, \cdot, \cdot)$, the policies outcome is measured as the cumulative reward in the current episode

$$\rho = E\{r_1 + r_2 + \cdots + r_T | \pi(a_e, \cdot, \cdot)\} \ . \tag{4}$$

Next, the characteristic eligibility of each component of θ is estimated by

$$\frac{\partial \ln g(a_e, \mu, \sigma)}{\partial \mu} = \frac{a_e - \mu}{\sigma^2} \tag{5}$$

$$\frac{\partial \ln g(a_e, \mu, \sigma)}{\partial \sigma} = \frac{(a_e - \mu)^2 - \sigma^2}{\sigma^3} \ , \tag{6}$$

and a reasonable algorithm for adapting μ and σ has the following form

$$\Delta\mu = \alpha_R (\rho - \bar{\rho}) \frac{a_e - \mu}{\sigma^2} \tag{7}$$

$$\Delta\sigma = \alpha_R (\rho - \bar{\rho}) \frac{(a_e - \mu)^2 - \sigma^2}{\sigma^3} \ . \tag{8}$$

The learning rate α_R has to be chosen appropriately, e.g. as a small positive constant, $\alpha_R = \alpha\sigma^2$, [9]. The baseline $\bar{\rho}$ is adapted by a simple reinforcement-comparison scheme

$$\bar{\rho} = \bar{\rho} + \alpha(\rho - \bar{\rho}) \ . \tag{9}$$

Analytically, in Eqn. 7 the mean μ is shifted towards a_e in case of $\rho \geq \bar{\rho}$. On the contrary, μ is shifted towards the opposite direction if ρ is less than $\bar{\rho}$. Similarly, in Eqn. 8 the standard deviation σ is adapted in a way that the occurrence of a_e is increased if $\rho \geq \bar{\rho}$, and decreased otherwise (see proof in [9]). In simple words, the standard deviation controls exploration in the space of a_e.

Importantly, a proper functioning of the proposed algorithm depends on some requirements. In order to limit the search of reasonable parameters, the exploration parameter, mean and standard deviation must be bounded for obtaining reasonable performance. Furthermore, if the learning problem consists of more than one starting state, all parameters must be associated to each occurring starting state, i.e. $\mu \rightarrow \mu(s)$, $\sigma \rightarrow \sigma(s)$ and $\bar{\rho} \rightarrow \bar{\rho}(s)$, since way costs might affect ρ unevenly. However, if a learning problem consists of just one starting state, all utilized parameters can be considered as global parameters.

4 Experiments

The presented approach is evaluated in two environments using Q-learning and Sarsa. First, a variation of the cliff-walking problem [1] is proposed as the *non-stationary cliff-walking problem* comprising a non-stationary environment.

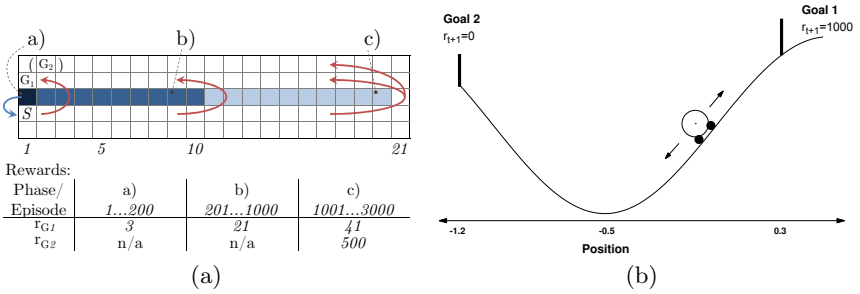


Fig. 1. The non-stationary cliff-walking problem (a) and the mountain-car problem with two goals (b).

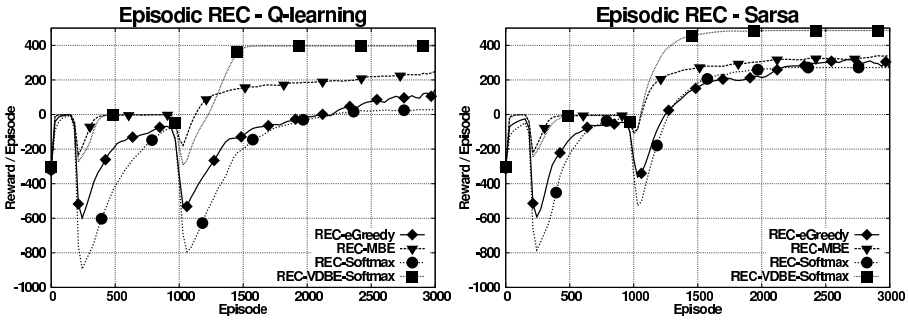
Second, a variation of the mountain-car problem is investigated comprising a continuous-valued state space approximated by a table, which causes partial observability of the actual coordinates. Investigated basic exploration policies are ε-Greedy, Softmax and MBE. Since MBE requires two parameters to be set (ε and τ), we only adapt ε of this policy, while setting the temperature parameter constantly to the value of $\tau = 1$, and normalizing all Q-values in state s into the interval $[-1, 1]$. For comparison, the recently proposed VDBE-Softmax policy is also evaluated, which works similar to the MBE policy, but adapts a state-dependent exploration rate $\varepsilon(s)$ based on fluctuations in the utility function Q [12]. For this policy the adaptable parameter is the *sensitivity parameter* used for controlling greediness.

4.1 The Non-Stationary Cliff-Walking Problem

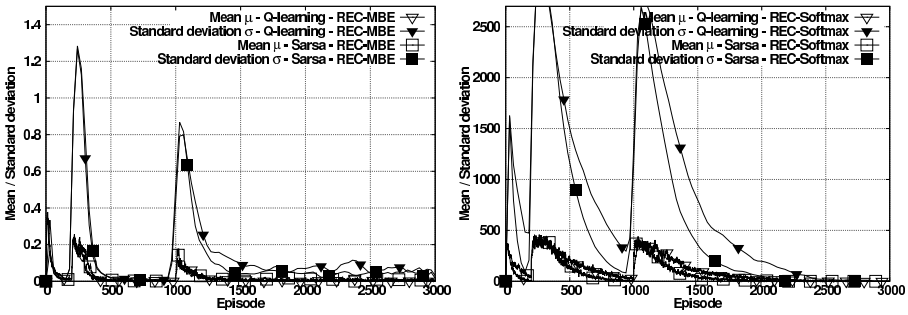
The *non-stationary cliff-walking problem* is a modification of the cliff-walking problem presented by Sutton and Barto [1], but additionally comprising non-stationary responses of the environment. The goal for the agent is learning a path from starting state S to the goal state G_1, which is rewarded with the absolute costs of the shortest path minus 1 if successful (see Fig. 1(a)). The way costs (reward) for each action are defined as $r_{\text{step}} = -1$. The environment also comprises unsafe *cliff states*, which lead to a high negative reward of $r_{\text{cliff}} = -100$ when entered, and also reset the agent back to the starting state S.

At the beginning of the experiment, learning takes place in phase (a) of Fig. 1(a) having one cliff state (at left border). After 200 learning episodes, the grid world changes to phase (b), now comprising 10 cliffs. After additional 800 episodes, the problem is tightened as shown in phase (c), where the number of cliffs is increased to 20. Additionally, an alternative goal state appears, $G2$, which is much higher rewarded with $r_{\text{G2}} = 500$ when entered.

In each episode the agent starts in state S, as well an episode terminates when a goal state, $G1$ or $G2$, is entered or the time limit of $T_{\max} = 200$ actions is exceeded. Since the learning problem is episodic, no discounting ($\gamma = 1$) is used. Finally, all utility values are optimistically initialized with $Q_{t=0}(s, a) = 0$, and learned using a constant step-size parameter of $\alpha = 0.2$.



(a) Averaged reward per episode.



(b) Mean and standard deviation for MBE and Softmax.

Fig. 2. The non-stationary cliff-walking problem: Averaged results (smoothed) using Q-learning and Sarsa. Note the dynamics of exploration for non-stationary environment responses.

Results Figure 2(a) shows the averaged reward per episode over 500 runs for the non-stationary cliff-walking problem. Averages of the mean μ and standard deviation σ are shown in Figure 2(b). It is observable that VDBE-Softmax maximizes the reward/episode. MBE shows best performance of the remaining three basic policies with the advantage of not requiring to memorize any further exploratory data such as utilized by VDBE-Softmax. The Sarsa learning algorithm shows better results for all four investigated policies. All REC policies have a much higher reward in episode 3000 compared to when using a pure greedy policy, which converges to a reward per episode of 0, and is only the optimal policy for the first 1000 episodes. In contrast, a pure random policy converges to a reward per episode of -2750 respectively.

4.2 The Mountain-Car Problem With Two Goals

In the mountain-car problem, the goal is driving an underpowered car up a mountain road [1], by initially standing in the valley between two mountains. The problem is that gravity is stronger than the car's engine, thus reaching the mountain top by full throttle only is not possible. Instead, the car has to swing

up for collecting enough inertia for overcoming gravity. In the here presented modification of the original learning problem, two goal states are utilized as depicted in Fig. 1(b), which are rewarded differently upon arrival, because a simple greedy policy leads to optimal performance in the original description of the learning problem. The bounded state variables are continuously valued consisting of the position of the car, $-1.2 \leq x \leq 0.3$, and its velocity $-0.07 \leq \dot{x} \leq 0.07$. The car's dynamics are described by differential equations

$$\begin{aligned} x_{t+1} &= \mathit{bound}\big[x_t + \dot{x}_{t+1}\big] \\ \dot{x}_{t+1} &= \mathit{bound}\big[\dot{x}_t + 0.001a_t - 0.0025\cos(3x_t)\big] \ . \end{aligned} \tag{10}$$

At each discrete time step, the agent can chose between one of seven actions, $a_t \in \{-1.0, -0.66, -0.33, 0, 0.33, 0.66, 1.0\}$, each rewarded by $r_{t+1} = -1$, except for reaching the right goal, which is rewarded by $r_{t+1} = 1000$. An episode terminates when either one of the two goals has been arrived or when a maximum number of actions, $T_{\max} = 10000$, is exceeded. At the beginning of each episode, the car is positioned in the valley at position $x = -0.5$ with initial velocity $\dot{x} = 0.0$. The state space is approximated by a 100×100 matrix, thus causing the actual state to be only partially observable. Since the learning problem is episodic, no discounting ($\gamma = 1$) is used. Finally, all utility values are optimistically initialized with $Q_{t=0}(s, a) = 0$, and learned using a step-size parameter of $\alpha = 0.7$.

Results The averaged results over 200 runs are shown in Figure 3. Similar to the non-stationary cliff-walking problem, episodic adaptation of MBE outperforms ε-Greedy and Softmax. Furthermore, the Sarsa algorithm shows only to be advantageous in combination with ε-Greedy and Softmax policies, in contrast to MBE and VDBE-Softmax behaving more efficiently in combination with Q-learning. In the first phase of learning, a degradation of performance is recognizable for episodic MBE and VDBE-Softmax, which is due to the reason of first learning a path to the left goal and afterwards learning the path to the right (better) goal. For comparison, a greedy policy converges to an average reward per episode of 114, in contrast to a pure random policy converging to -5440.

5 Discussion and Conclusions

In this paper, stochastic neurons have been deployed for adapting the exploration parameter of basic exploration policies using gradient-following algorithms. A global variant is evaluated with two popular learning algorithms, Q-learning and Sarsa, in two different learning problems showing performance improvements when trading off exploration and exploitation. Results from the non-stationary cliff-walking problem show how the exploration parameter is readapted based on learning progress when non-stationary environment responses are received. The MBE policy turned out to be reliable for achieving good performance without requiring to memorize any exploratory data. However, additional exploratory data might further improve results as shown using VDBE-Softmax in the non-stationary cliff-walking problem, but which is not always the case as it is observable in the mountain-car problem with two goals.

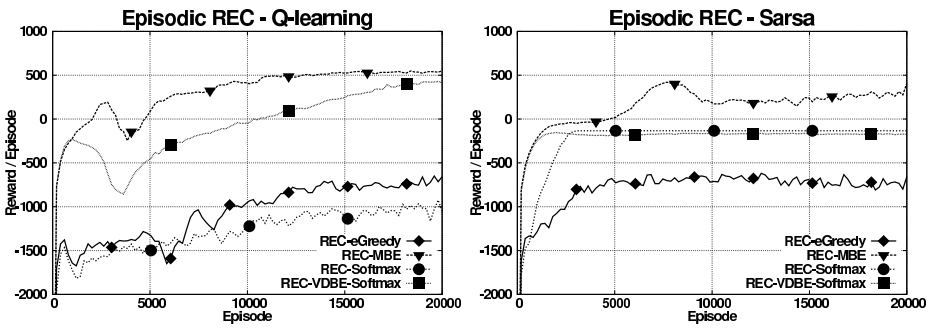


Fig. 3. The mountain-car problem with two goals: Averaged reward using Q-learning and Sarsa (smoothed).

Acknowledgements. Michel Tokic received funding by the collaborative center for applied research *ZAFH-Servicerobotik*. The authors gratefully acknowledge the research grants of the federal state Baden-Württemberg and the European Union.

References

[1] Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, Cambridge, MA (1998)
[2] Wiering, M.: Explorations in Efficient Reinforcement Learning. PhD thesis, University of Amsterdam, Amsterdam (1999)
[3] Thrun, S.B.: Efficient exploration in reinforcement learning. Technical Report CMU-CS-92-102, Carnegie Mellon University, Pittsburgh, USA (1992)
[4] Auer, P.: Using confidence bounds for exploitation-exploration trade-offs. The Journal of Machine Learning Research **3** (2002) 397–422
[5] van Eck, N.J., van Wezel, M.: Application of reinforcement learning to the game of Othello. Computers and Operations Research **35** (2008) 1999–2017
[6] Faußer, S., Schwenker, F.: Learning a strategy with neural approximated temporal-difference methods in english draughts. In: Proceedings of the 20th International Conference on Pattern Recognition. ICPR'10, IEEE Computer Society (2010) 2925–2928
[7] Rummery, G.A., Niranjan, M.: On-line Q-learning using connectionist systems. Technical Report CUED/F-INFENG/TR 166, Cambridge University (1994)
[8] Daw, N.D., O'Doherty, J.P., Dayan, P., Seymour, B., Dolan, R.J.: Cortical substrates for exploratory decisions in humans. Nature **441**(7095) (2006) 876–879
[9] Williams, R.J.: Simple statistical Gradient-Following algorithms for connectionist reinforcement learning. Machine Learning **8** (1992) 229–256
[10] Watkins, C.: Learning from Delayed Rewards. PhD thesis, University of Cambridge, England (1989)
[11] Grzes, M., Kudenko, D.: Online learning of shaping rewards in reinforcement learning. Neural Networks **23**(4) (2010) 541–550
[12] Tokic, M., Palm, G.: Value-difference based exploration: Adaptive exploration between epsilon-greedy and softmax. In: KI 2011: Advances in Artificial Intelligence. Springer Berlin / Heidelberg (2011) 335–346