Adaptive ε-greedy Exploration in Reinforcement Learning Based on Value Differences

Michel Tokic[1,2]

[1] Institute of Applied Research, University of Applied Sciences
Ravensburg-Weingarten, 88241 Weingarten, Germany
[2] Institute of Neural Information Processing, University of Ulm, 89069 Ulm, Germany
michel@tokic.com

Abstract. This paper presents "Value-Difference Based Exploration" (VDBE), a method for balancing the exploration/exploitation dilemma inherent to reinforcement learning. The proposed method adapts the exploration parameter of ε-greedy in dependence of the *temporal-difference error* observed from value-function backups, which is considered as a measure of the agent's uncertainty about the environment. VDBE is evaluated on a multi-armed bandit task, which allows for insight into the behavior of the method. Preliminary results indicate that VDBE seems to be more parameter robust than commonly used ad hoc approaches such as ε-greedy or softmax.

1 Introduction

Balancing the ratio of exploration/exploitation is a great challenge in reinforcement learning (RL) that has a great bias on learning time and the quality of learned policies. On the one hand, too much exploration prevents from maximizing the short-term reward because selected "exploration" actions may yield negative reward from the environment. But on the other hand, exploiting uncertain environment knowledge prevents from maximizing the long-term reward because selected actions may not be optimal. For this reason, the described problem is well-known as the *dilemma of exploration and exploitation* [1].

This paper addresses the issue of adaptive exploration in RL and elaborates on a method for controlling the amount of exploration on basis of the agent's uncertainty. For this, the proposed VDBE method extends ε-greedy [2] by adapting a state dependent exploration probability, $\varepsilon(s)$, instead of the classical hand-tuning of this globally used parameter. The key idea is to consider the TD-error observed from value-function backups as a measure of the agent's uncertainty about the environment, which directly affects the exploration probability.

In the following, results are reported from evaluating VDBE and other methods on a multi-armed bandit task, which allows for understanding of VDBE's behavior. Indeed, it is important to mention that the proposed method is not specifically designed for just solving bandit problems, and thus, learning problems with even large state spaces may benefit from VDBE. For this reason, we do not compare the performance with methods that are unpractical in large state spaces because of their memory and computation time requirements.

1.1 Related Work

In the literature, many different approaches exists in order to balance the ratio of exploration/exploitation in RL: many methods utilize counters [3], model learning [4] or reward comparison in a biologically-inspired manner [5]. In practice, however, it turns out that the ε-greedy [2] method is often the method of first choice as reported by Sutton [6]. The reason for this seems to be due to the fact that (1) the method does not require to memorize any exploration specific data and (2) is known to achieve near optimal results in many applications by the hand-tuning of only a single parameter, see e.g. [7].

Even though the ε-greedy method is reported to be widely used, the literature still lacks on methods of adapting the method's exploration rate on basis of the learning progress. Only a few methods such as *ε-first* or *decreasing-ε* [8] consider "time" in order to reduce the exploration probability, but what is known to be less related to the true learning progress. For example, why should an agent be less explorative in unknown parts of a large state space due to a time-decayed exploration rate? In order to propose a possible solution to this problem, this paper introduces a method that takes advantage of the agent's learning progress.

2 Methodology

We consider the RL framework [1] where an agent interacts with a Markovian decision process (MDP). At each discrete time step $t \in \{0, 1, 2, ...\}$ the agent is in a certain state $s_t \in \mathcal{S}$. After the selection of an action, $a_t \in \mathcal{A}(s_t)$, the agent receives a reward signal from the environment, $r_{t+1} \in \mathbb{R}$, and passes into a successor state s'. The decision which action a is chosen in a certain state is characterized by a policy $\pi(s) = a$, which could also be stochastic $\pi(a|s) = Pr\{a_t = a|s_t = s\}$. A policy that maximizes the cumulative reward is denoted as π^*.

In RL, policies are often learned by the use of state-action value functions which denote how "valuable" it is to select action a in state s. Hereby, a state-action value denotes the expected cumulative reward R_t for following π by starting in state s and selecting action a

$$\begin{aligned} Q^{\pi}(s,a) &= E_{\pi}\left\{R_t|s_t = s, a_t = a\right\} \\ &= E_{\pi}\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1}|s_t = s, a_t = a\right\} \quad , \end{aligned} \tag{1}$$

where γ is a discount factor such that $0 < \gamma \leq 1$ for episodic learning tasks and $0 < \gamma < 1$ for continuous learning tasks.

2.1 Learning the *Q* function

Value functions are learned through the agent's interaction with its environment. For this, frequently used algorithms are Q-learning [2] or Sarsa [9] from the *temporal difference* approach, and which are typically used when the environment model is unknown. Other algorithms of the *dynamic programming* approach [10] are used when a model of the environment is available and therefore usually converge faster. In the following, we use a version of *temporal difference* learning with respect to a single-state MDP, which is suitable for experiments with the multi-armed bandit problem. For that reason the discount factor from Equation (1) is set to $\gamma = 0$ which causes the agent to maximize the immediate reward.

A single-state MDP with n different actions is considered where each single action is associated with a stochastic reward distribution. After the selection of action a_t, the environment responds with the reward signal, r_{t+1}, by which the mean reward of action a can be estimated by

$$Q_{t+1}(a) \leftarrow Q_t(a) + \alpha_t \Big[r_{t+1} - Q_t(a) \Big] \ , \tag{2}$$

where α is a positive step-size parameter such that $0 < \alpha \leq 1$. Larger rewards than the so far learned estimate will shift the estimate up into direction of the reward, and lower rewards vice versa. For this, the term in brackets is also called the *temporal-difference error* (TD-error) that indicates in which direction the estimate should be adapted.

Usually, the step-size parameter is a small constant if the reward distribution is assumed to be non-stationary. In contrast, when the reward distribution is assumed to be stationary, then the *sample average* method can be used in order to average the rewards incrementally by

$$\alpha_t(a) = \frac{1}{1 + k_a} \ , \tag{3}$$

where k_a indicates the number of preceding selections of action a. In general, it is important to know that the step-size parameter α is also a key for maximizing the speed of learning since small step-sizes cause long learning times, and however, large step-sizes cause oscillations in the value function. A more detailed overview of these and other step-size methods can be found in [11].

2.2 Exploration/Exploitation Strategies

Two widely used methods for balancing exploration/exploitation are ε-greedy and softmax [1]. With ε-greedy, the agent selects at each time step a random action with a fixed probability, $0 \leq \varepsilon \leq 1$, instead of selecting greedily one of the learned optimal actions with respect to the Q-function:

$$\pi(s) = \begin{cases} \text{random action from } \mathcal{A}(s) & \text{if } \xi < \varepsilon \\ \text{argmax}_{a \in \mathcal{A}(s)} Q(s,a) & \text{otherwise,} \end{cases} \tag{4}$$

where $0 \leq \xi \leq 1$ is a uniform random number drawn at each time step. In contrast, softmax utilizes action-selection probabilities which are determined by ranking the value-function estimates using a Boltzmann distribution:

$$\pi(a|s) = Pr\{a_t = a | s_t = s\} = \frac{e^{\frac{Q(s,a)}{\tau}}}{\sum_b e^{\frac{Q(s,b)}{\tau}}} \ , \tag{5}$$

where τ is a positive parameter called temperature. High temperatures cause all actions to be nearly equiprobable, whereas low temperatures cause greedy action selection.

In practice, both methods have advantages and disadvantages as described in [1]. Some derivatives of ε-greedy utilize time in order to reduce ε over time [8]. For example, the *decreasing-ε* method starts with a relative high exploration rate, which is reduced at each time step. Another example is the *ε-first* method, where full exploration is performed for a specific amount of time after that full exploitation is performed.

3 ε-greedy VDBE-Boltzmann

The basic idea of VDBE is to extend the ε-greedy method by controlling a state-dependent exploration probability, $\varepsilon(s)$, in dependence of the value-function error instead of manual tuning. The desired behavior is to have the agent more explorative in situations when the knowledge about the environment is uncertain, i.e. at the beginning of the learning process, which is recognized as large changes in the value function. On the other hand, the exploration rate should be reduced as the agent's knowledge becomes certain about the environment, which can be recognized as very small or no changes in the value function. For this, the following equations adapt such desired behavior according to a (softmax) Boltzmann distribution of the value-function estimates, which is performed after each learning step by

$$\begin{aligned} f(s,a,\sigma) &= \left| \frac{e^{\frac{Q_t(s,a)}{\sigma}}}{e^{\frac{Q_t(s,a)}{\sigma}} + e^{\frac{Q_{t+1}(s,a)}{\sigma}}} - \frac{e^{\frac{Q_{t+1}(s,a)}{\sigma}}}{e^{\frac{Q_t(s,a)}{\sigma}} + e^{\frac{Q_{t+1}(s,a)}{\sigma}}} \right| \\ &= \frac{1 - e^{\frac{-|Q_{t+1}(s,a) - Q_t(s,a)|}{\sigma}}}{1 + e^{\frac{-|Q_{t+1}(s,a) - Q_t(s,a)|}{\sigma}}} \\ &= \frac{1 - e^{\frac{-|\alpha \cdot \mathrm{TD-Error}|}{\sigma}}}{1 + e^{\frac{-|\alpha \cdot \mathrm{TD-Error}|}{\sigma}}} \qquad (6) \\ \varepsilon_{t+1}(s) &= \delta \cdot f(s_t, a_t, \sigma) + (1-\delta) \cdot \varepsilon_t(s) \ , \qquad (7) \end{aligned}$$

where σ is a positive constant called *inverse sensitivity* and $\delta \in [0,1)$ a parameter determining the influence of the selected action on the exploration rate. An obvious setting for δ may be the inverse of the number of actions in the current state, $\delta = \frac{1}{|\mathcal{A}(s)|}$, which led to good results in our experiments. The resulting effect of σ is depicted in Figure 1. It is shown that low inverse sensitivities cause full exploration even at small value changes. On the other hand, high inverse sensitivities cause a high level of exploration only at large value changes. In the limit, however, the exploration rate converges to zero as the Q-function converges, which results in pure greedy action selection. At the beginning of the learning process, the exploration rate is initialized by $\varepsilon_{t=0}(s) = 1$ for all states.

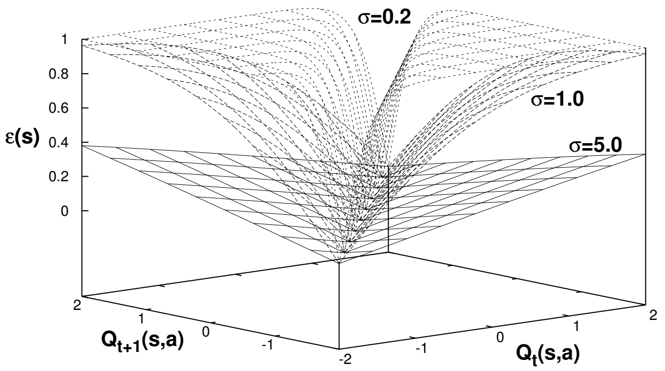


Fig. 1. Graph of $f(s,a,\sigma)$ in dependence of various sensitivities.

4 Experiments and Results

A typical scenario for evaluating exploration/exploitation methods is the multi-armed bandit problem [1, 12]. In this example a casino player can choose at each time step among n different levers of an n-armed bandit (slot machine analogy) with the goal of maximizing the cumulative reward within a series of trials. Each pull of a lever returns a numerical reward, i.e. the payoff for hitting the jackpot, which is drawn from a stationary probability distribution dependent on lever a, $a \in \{1,\ldots,n\}$. The player uses the rewards to estimate the "value" of each lever, for example, by averaging the rewards per lever in order to learn which lever optimizes the cumulative reward. During this process, the player has to decide at each time step whether he "exploits" greedily the lever having the highest estimated value or whether he "explores" one of the other levers to improve the estimates.

Some real-world problems analogous to the multi-armed bandit problem are, e.g., adaptive routing in networks with the goal of delay minimization [13] or the economic problem of selecting the best supplier on the basis of incomplete information [14].

4.1 Experiment Setup

The VDBE method is compared and evaluated on a set of 2000 randomly generated 10-armed bandit task as described in [1]. Each selection of a lever returns a stochastic reward drawn from a stationary normal (Gaussian) distribution with mean $Q^*(a)$ and variance 1, where all means are initialized randomly according to a normal distribution with mean 0 and variance 1. The results are averaged over 2000 randomly generated bandit tasks for each exploration parameter, where in each task the player can improve its action selection policy within 1000 trials. Throughout the experiments, learning after each action selection is based on Equation (2) in combination with the sample-average method from Equation (3), which simulates a convergent Q-function in a single-state MDP.

The state-dependent exploration rate of VDBE is immediately recomputed after the value-function backup of the selected action according to Equations (6, 7).

The overall performance is empirically measured and compared against ε-greedy and softmax action selection within a large bandwidth of constant parameter settings. For this, the exploration rate of ε-greedy has been investigated for different values within the interval $[0, 1]$, softmax within $[0.04, 25]$ and VDBE within $[0.04, 25]$, respectively[3]. The δ parameter of VDBE has been set to the inverse of the number of actions, i.e. $\delta = \frac{1}{|\mathcal{A}(s)|} = 0.1$.

4.2 Results

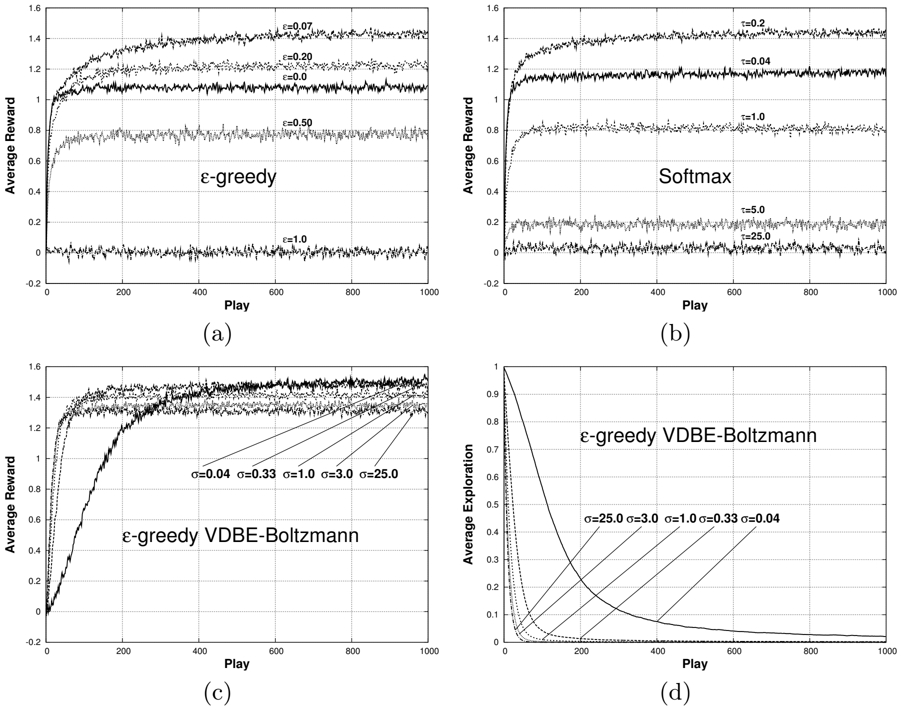
The results depicted in Figure 2 and Table 1 compare the three investigated methods on the multi-armed bandit task. First, from the comparison of ε-greedy and softmax it can be observed that the performance of these methods varies significantly depending on their parameters. The poor performance of both methods is not surprising and due to the fact that a large chosen exploration rate ε causes a large amount of random action selections, whereas the same applies also for high temperatures of softmax. Second, it is also observable that large parameter settings of both methods lead to a relative low level of the mean reward. In contrast, low parameter settings improve the results in the limit (even though very slowly) as the number of plays goes to infinity and when at least a constant bit of exploration is performed. In the case when no exploration is performed at all, the value function will get caught in local minima as it can be observed from the $\varepsilon = 0$ curve, which is equal to softmax and $\tau \rightarrow 0$.

Table 1. Empirical results: $r_{opt} = \frac{r_{t=1}+\cdots+r_{t=1000}}{1000}$ denotes the averaged reward per time step for the parameter that maximizes the cumulative reward within 1000 plays. r_{min} and r_{max} denote the minimum and maximum reward at play $t = 1000$. Numbers in brackets indicate the method's parameter for achieving the results.

Method	r_{opt}	r_{min}	r_{max}
ε-greedy	1.35 ($\varepsilon = 0.07$)	0.00 ($\varepsilon = 1.00$)	1.43 ($\varepsilon = 0.07$)
Softmax	1.38 ($\tau = 0.20$)	0.00 ($\tau = 25.0$)	1.44 ($\tau = 0.20$)
VDBE	1.42 ($\sigma = 0.33$)	1.30 ($\sigma = 25.0$)	1.50 ($\sigma = 0.04$)

In contrast, the advantage of adaptive exploration is shown in the plot of ε-greedy VDBE-Boltzmann. First, it can be observed that the range of the results after 1000 plays is much smaller and in the upper level of the overall reward range than with ε-greedy and softmax. Second, it can also be observed that the exploration rate converges to zero in dependence of the Q-function's convergence and independently of the chosen *inverse sensitivity*.

[3] In detail, the investigated parameters within the intervals have been:
ε-greedy: $\varepsilon \in \{0, 0.01, 0.05, 0.07, 0.08, 0.09, 0.10, 0.20, 0.30, 0.50, 0.80, 1.0\}$
Softmax and VDBE: $\tau, \sigma \in \{0.04, 0.10, 0.20, 0.33, 0.50, 1.0, 2.0, 3.0, 5.0, 25.0\}$



(a) (b) (c) (d)

Fig. 2. Comparison of the average reward on the 10-armed bandit task for (a) ε-greedy, (b) softmax and (c) VDBE. Graph (d) shows the exploration probability of VDBE.

5 Discussion and Conclusion

Based on the results, the VDBE method has been identified to be more robust over a wide range of parameter settings while still achieving acceptable performance results. In case VDBE is used in large state spaces where $\varepsilon(s)$ is approximated as a function (e.g. by a neural network), the method is also robust against errors from generalization since pure exploitation is performed in the limit. Although the method is demonstrated on a single-state MDP, the mathematical principle remains the same in multi-state MDPs where learning is additionally based on neighbor-state information ($\gamma > 0$), e.g. as in Q-learning. The only important assumption for VDBE is the convergence of the Q-function which depends (1) on the learning problem, (2) on the learning algorithm and (3) on the choice of the step-size parameter function. A non-convergent Q-function will cause a constant level of exploration, where the amount is dependent on the chosen inverse sensitivity.

To sum up, the results obtained from the experiments look promising which suggests that balancing the exploration/exploitation ratio based on value differences needs to be further investigated. In order to find out finally whether VDBE outperforms existing exploration strategies—or under which conditions it outperforms other strategies—the application to other more complex learning problems is required.

Acknowledgements. The author gratefully thanks G. Palm, M. Oubbati and F. Schwenker from Ulm University for their valuable comments and discussions on VDBE, which has also been discussed with W. Ertel, P. Ertle, M. Schneider and R. Cubek from UAS Ravensburg-Weingarten. The author also likes to thank P. Ertle, H. Bou Ammar and A. Usadel from UAS Ravensburg-Weingarten for proof reading the paper.

References

[1] Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, Cambridge, MA (1998)

[2] Watkins, C.: Learning from Delayed Rewards. PhD thesis, University of Cambridge, Cambridge, England (1989)

[3] Thrun, S.B.: Efficient exploration in reinforcement learning. Technical Report CMU-CS-92-102, Carnegie Mellon University, Pittsburgh, PA, USA (1992)

[4] Brafman, R.I., Tennenholtz, M.: R-MAX - a general polynomial time algorithm for near-optimal reinforcement learning. Journal of Machine Learning Research **3** (2002) 213–231

[5] Ishii, S., Yoshida, W., Yoshimoto, J.: Control of exploitation-exploration meta-parameter in reinforcement learning. Neural Networks **15**(4-6) (2002) 665–687

[6] Heidrich-Meisner, V.: Interview with Richard S. Sutton. Künstliche Intelligenz **3** (2009) 41–43

[7] Vermorel, J., Mohri, M.: Multi-armed bandit algorithms and empirical evaluation. In: Proceedings of the 16th European Conference on Machine Learning (ECML'05), Porto, Portugal (2005) 437–448

[8] Caelen, O., Bontempi, G.: Improving the exploration strategy in bandit algorithms. In: Learning and Intelligent Optimization. Number 5313 in LNCS. Springer (2008) 56–68

[9] Rummery, G.A., Niranjan, M.: On-line Q-learning using connectionist systems. Technical Report CUED/F-INFENG/TR 166, Cambridge University (1994)

[10] Bertsekas, D.P.: Dynamic Programming: Deterministic and Stochastic Models. Prentice Hall (1987)

[11] George, A.P., Powell, W.B.: Adaptive stepsizes for recursive estimation with applications in approximate dynamic programming. Machine Learning **65**(1) (2006) 167–198

[12] Robbins, H.: Some aspects of the sequential design of experiments. Bulletin of the American Mathematical Society **58** (1952) 527–535

[13] Awerbuch, B., Kleinberg, R.D.: Adaptive routing with end-to-end feedback: Distributed learning and geometric approaches. In: Proceedings of the 36th Annual ACM Symposium on Theory of Computing, Chicago, IL, USA, ACM (2004) 45–53

[14] Azoulay-Schwartz, R., Kraus, S., Wilkenfeld, J.: Exploitation vs. exploration: Choosing a supplier in an environment of incomplete information. Decision Support Systems **38**(1) (2004) 1–18