Types of Reinforcement learning algorithm – adaptive ϵ methods

Ameet Rahane

April 10, 2019

1 Preface

All of the following methods are ways I'm considering changing explore/exploit behavior within models in order to fit it to a biological agent. Something of notes is that most of these models are built to optimize for reward gained. This is not necessarily our goal. We want to find the best fit to a mouse.

2 ϵ Greedy

With this method, the amount of exploration is globally controlled by a parameter $\epsilon \in [0, 1]$ that denotes the probability of action selection. You choose actions based on the following equation:

$$a_t = \begin{cases} a_t^* & p = 1 - \epsilon \\ \text{random} & p = \epsilon \end{cases} \tag{1}$$

This basically stochastically chooses between two choices – pure randomness or pure maximization (this is probably a bit idealistic and won't fit mice that well).

3 Softmax

- Bias exploration towards promising actions
- Softmax action selection methods grade action probabilities by estimated values
- Most Common is Boltzmann Distribution:

$$\pi(a|s) = \frac{e^{\frac{Q(s,a)}{\tau}}}{e^{\sum_{a' \in A} \frac{Q(s,a')}{\tau}}} \tag{2}$$

- Here τ is a temperature constant. That is $\tau \to \infty$ implies that $P = \dfrac{1}{|A|}$ and $T \to 0$ implies that it's greedy

4 Value Difference Based Exploration

This method extends $\epsilon-$greedy by adapting a state dependent exploration probability $\epsilon(s)$ instead of the classical hand tuning. 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.

Consider the RL framework where an agent interacts with a Markovian decision process. At each time step $t \in \mathbb{N}$, the agent is in a sertain state $s_t \in S$. After the selection of the action,$a_t \in A(s_t)$, the agent receives a reward signal from the environment $r_{t+1} \in \mathbb{R}$ and is passed into a successor state. The decision which action a is chosen in a certain state is characterized by a policy $\pi(s) = a$, which could also be stochastic $\pi(a|s) = p(a_t = a|s_t = s$. A policy that maximizes the cumulative reward is denoted as π^*.

A state action value denotes the expected cumulative reward R_t for following π by starting in state s and selection action a:

$$Q^\pi(s,a) = E_\pi\{R_t|s_t = s, a_t = a\} = E_\pi\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1}|s_t = s, a_t = a\} \tag{3}$$

γ is a discount factor in $(0, 1]$ for episodic learning tasks and $0 < \gamma < 1$ for continuous learning tasks.

The basic idea of VDBE is to extend ϵ greedy by controlling a state dependent explorations probability $\epsilon(s)$ in dependence of the value function error instead of manual tuning. The following equatins adapt such desired behavior according to a softmax Boltzmann distribution of the value fuction estimates. This is performed after eadch learning step by:

$$f(s,a,\sigma) = \frac{1 - e^{\frac{-|\alpha \text{TD-Error}|}{\sigma}}}{1 + e^{\frac{-|\alpha \text{TD-Error}|}{\sigma}}} \tag{4}$$

Then, we update ϵ as follows:

$$\epsilon_{t+1}(s) = \delta f(s_t, a_t, \sigma) + (1 - \delta)\epsilon_t(s) \tag{5}$$

σ is a positive constant called inverse sensitivity and $\delta \in [0, 1)$, a parameter determining the influence of the selection action on the exploration rate. An obvious setting for δ may be the inverse of the number of actions in the current state: $\delta = \frac{1}{|\mathcal{A}(s)|}$. Low inverse sensitivities cause full exploration even at small value changes. On the other hand, shigh inverse sensitivities cause a high level of exploration only at large value changes. In the limit, the exploration rate converges to zero as the Q-function converges, which results in pure greedy action selection. One drawback is the exploration actions are chosen uniformly distributed among all possible actions in the current state.

5 Adaptive control between ϵ Greedy and Softmax

5.1 Learning the Q function by on and off policy methods

Value functions are learned by sampling observations of the interactions between the agent and its environment. For this, the branch of temporal difference learing offers two commonly used alogrithm which are named Sarsa for on-policy control:

$$\Delta_{\text{Sarsa}} = [r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t] \tag{6}$$

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha \Delta_{\text{Sarsa}} \tag{7}$$

and Q learning for off policy control:

$$b^* = \text{argmax}_{b \in \mathcal{A}(s_{t+1})} Q(s_{t+1}, b) \tag{8}$$

$$\Delta_{\text{Qlearning}} = [r_{t+1} + \gamma Q(s_{t+1}, b^*) - Q(s_t, a_t)] \tag{9}$$

$$Q(s_t, a_t = Q(s_t, a_t) + \alpha \Delta_{\text{Qlearning}} \tag{10}$$

α is a stepsize parameter. The only difference in two algorithms is the inclusion of successor state information used for the evaluation of action a_t taken in state s_t while learning the value function.

5.2 VDBE-Softmax

One way we can fix the issue with VDBE is to do a softmax step instead of sampling uniformly for the explore. This is as follows:

$$\pi(s) = \begin{cases} \text{Softmax: } \dfrac{e^{Q(s,a)/\tau}}{\sum_b e^{Q(s,b)/\tau}} & \zeta < \epsilon \\ \text{argmax}_{a \in \mathcal{A}(s)} Q(s,a) & \zeta \geq \epsilon \end{cases} \tag{11}$$

This adapts the state dependent exploration rate according to VDBE but selects random actions according to Softmax in the case of exploration.

6 Creating a probability distribution over the possible q value and sampling that

7 Apply the Softmax at each step

8 Binary stochastic switching between greedy behavior and probabilistic sampling

9 Binary stochastic switching between greedy behavior and softmax

10 Adaptive Exploration using stochastic neurons

11 Meta learning of exporation and exploitation parameters

12 Gradient Algorithms for Exploration/Exploitation Trade-offs

13 Chunk based exploration/exploitation???