Our Extraordinary Reinforcement Learning Assignment

Farrukh Baratov (s3927083), Darie Petcu (s3990044)

December 6, 2021

1 Algorithms used

1.1 Greedy

$$A_t = \arg\max_{a \in \mathcal{A}} Q_t(a)$$

The greedy always goes for the action with the highest expected utility, focusing on exploitation over exploration. This is done using the above equation, which selects the arm with the highest utility at time t.

1.2 ϵ -Greedy

$$\pi_t(a) = \begin{cases} (1-\epsilon) + \epsilon/|\mathcal{A}| & \text{if } a = \arg\max_{a \in \mathcal{A}} Q_t(a) \\ \epsilon/|\mathcal{A}| & \text{otherwise} \end{cases}$$

ϵ-Greedy works in the same way, excepts it also uses a probability: at each timestep, there is a probability ϵ of exploring a random arm, and otherwise behaving greedily.

1.3 Upper-Confidence Bound

$$a_t \doteq \arg\max_{a \in \mathcal{A}} \left[Q_t(a) + c\sqrt{\frac{ln(t)}{N_a(t)}} \right]$$

UCB greedily selects the arm with the highest predicted upper confidence bound using the above equation, while also encouraging exploration by reducing the chance of picking a frequently selected arm and increasing the chance of picking a less frequently selected arm as time goes on.

As is visible, having a high estimated utility and a low uncertainty increases the chance of being selected. Uncertainty for an arm is incremented every time it is selected, causing an arm that has been selected before to have a lower added chance of being selected and encouraging exploration. The hyper-parameter c dictates how much exploration is prioritized over exploitation, as increasing this parameter increases the chance of selecting an uncommonly picked arm in the long run.

1.4 Optimistic Initial Values

Optimistic initial values works by initializing the expected utility for an arm at a high value within its reward distribution. Afterwards, it greedily selects the arms with the highest expected utility. It is expected that, when selecting a new arm, the actual reward will be much lower than the optimistic expectation, causing the expectation to drop significantly and encouraging the agent to select other options. Early exploration is encouraged this way.

1.5 Softmax Policy

$$\pi(a) = \frac{e^{Q_t(a)/\tau}}{\sum_{k=0}^{N} e^{Q_t(k)/\tau}}$$

The softmax policy algorithm works by computing probabilities of choosing each arm. The probability depends on the value estimation function, as well as the hyperparameter τ. The hyperparameter controls the relevance of the value estimations, with higher values resulting in less influence for the Q-values. This is an enhanced version of the previous algorithms, where exploration is not blindly performed, but rather the arm to be explored is chosen based on prior knowledge (stored using the value estimations)

1.6 Action Preferences

$$\pi(a) = \frac{e^{H_t(a)}}{\sum_{k=0}^{N} e^{H_t(k)}}$$

$$H_{t+1}(a') \doteq H_t(a') + \alpha\,(r_t - \bar{r}_t)(1 - \pi_t(a'))$$

$$H_{t+1}(a) \doteq H_t(a) - \alpha\,(r_t - \bar{r}_t)\,\pi_t(a) \quad \text{if} \quad a \neq a'$$

The Action Preferences algorithm is similar to the Softmax algorithm when it comes to the involved formulas. However, action preferences is improved further by factoring in the regret (difference between expected and real reward) and the prior knowledge (new H is computed using old H and old policies π).

2 Experimental Setup

The experimental setup was as follows: Each algorithm was tested using 1000 agents, each running and training for 1000 epochs. 7 arms were used during testing with arbitrarily assigned distributions.

2.1 Reward Functions Definitions

2.1.1 Gaussian Distribution

The mean of the Gaussian distribution is a randomly generated value (using a uniform distribution) between 0.3 and 1, while the standard deviation was fixed at 0.2. The reward for an arm is randomly sampled from this distribution. Furthermore, the reward was limited to values from 0 to 1 so if a value outside this range is sampled, it is rounded to the minimum/maximum value within the range.

2.1.2 Bernoulli Distribution

The probability of each Bernoulli distribution is randomly generated using a uniform distribution. When an arm is successfully pulled (i.e. reward is sampled), 1 is returned. Otherwise, 0 is returned. The "success" is based on whether or not a value randomly generated from a uniform distribution is lower than the arm's probability.

2.2 Hyperparameters for each algorithm

Once the algorithms and multi-armed bandit have been implemented, we started experimenting with various values for the hyperparameters.

We wrote a script that creates plots for several hyperparameter values of each action selection algorithm. The result were four plots which allowed us to intuitively pick the optimum hyperparameter value. An example of the plots we used for selecting hyperparameters can be observed in Figure 1

We arrived at the following hyperparameters:

- $(\epsilon - greedy)$: $\epsilon = 0.38$
- (UCB algorithm): $c = 0.38$
- (softmax): $\tau = 0.12$
- (action preferences): $\alpha = 0.9$

3 Results

3.1 Gaussian Distribution

The performance and accuracy of the algorithms against Gaussian Distribution-based arms is shown in Figure 2.

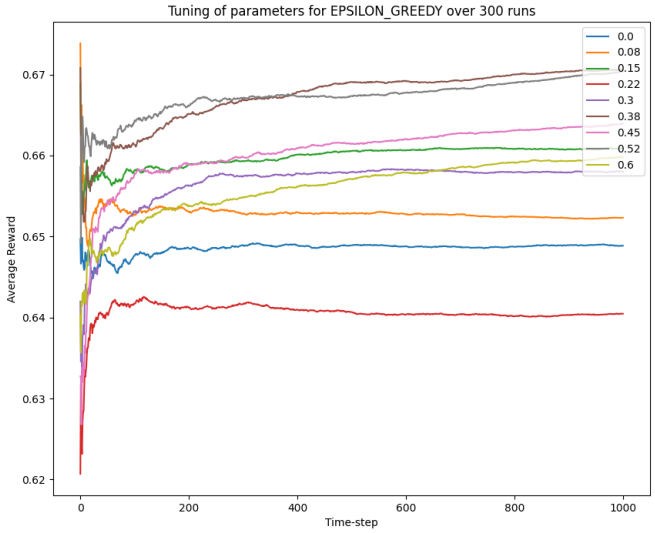

Figure 1: The performance of the ϵ-greedy algorithm for different hyperparameter ϵ values.

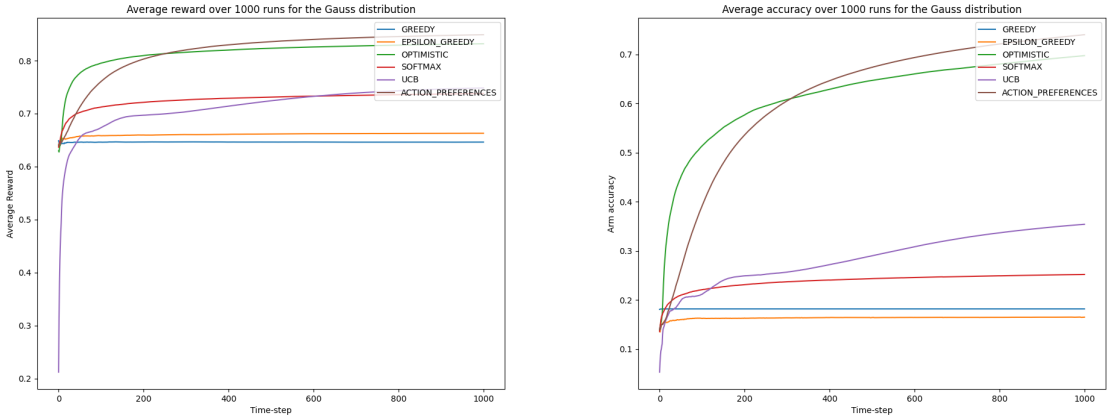

Figure 2: Average rewards and accuracy obtained by algorithms for the Gaussian distribution.

3.2 Bernoulli Distribution

The performance and accuracy of the algorithms Bernoulli Distribution-based arms is shown in Figure 3.

4 Algorithms Comparison

4.1 Conceptual comparison

We understood each algorithm as adding one more layer of complexity to the decisional factors. ϵ-greedy is an improved greedy because it sometimes attempts some exploration. Although these attempts occur at random times, this is still an improvement over not performing exploration.

Upper-Confidence Bounds is the first step in realizing the value estimations are not the only relevant criteria to take into account. The frequency of picking an arm is factored into the equation now, but we have yet to take the rewards into account probabilistically.

The softmax algorithm takes that step, by computing the π policies in terms of the value estimations and the time step. Furthermore, action preferences also benefit from stepwise updating of the parameters

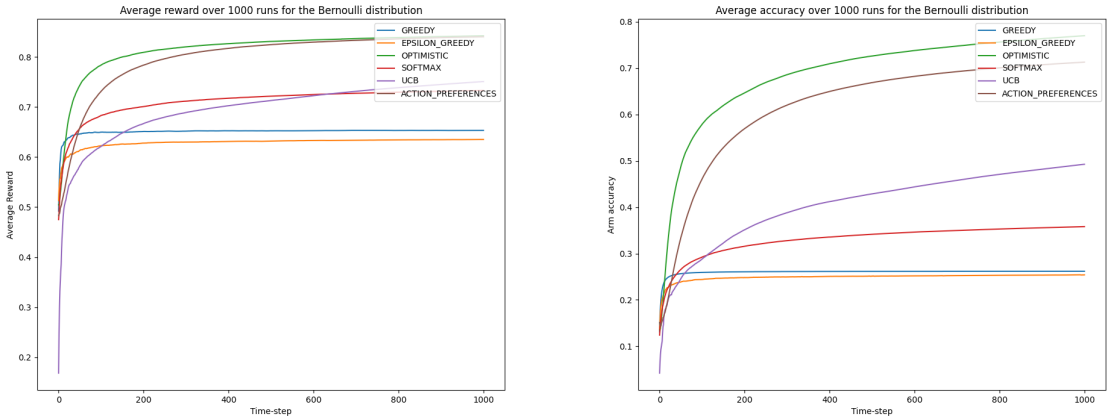

Figure 3: Average rewards and accuracy obtained by algorithms for the Bernoulli distribution.

(π and H) that allows for using past knowledge, and from using the regret as a coefficient to the learned information.

Apart from these algorithms, there is also the *optimistic initial values (OIV)*. This still requires an action selection algorithm. When used in combination with greedy, their performance is much better than standard greedy. This is because the agent is allowed to explore the environment (by pursuing the arms that are still "thought to be" valuable). Once the agent has sampled each arm, due to the small number of arms present, chances are high that one of the more valuable arms will be identified as favorable due to the high value estimation. The algorithm will then be focused on exploitation, since the OIV effects have been overwritten, but it will have already settled upon at least a near-optimal arm.

4.2 Results comparison

Average Rewards: The algorithms performed very well, given a few timesteps to learn the arm distributions. Their hierarchy is not aligned with our aforementioned expectations. For example, we expected the softmax and action performance algorithms to perform better than all others; however, softmax presents dissapointing results, being inferior to UCB or OIV. Despite the boolean rewards given by the Bernoulli distribution problem, it had comparatively similar results to the Gaussian distribution problem. The optimistic initial values trend performs the best. This occurs due to the exploitation-focused approached of greedy selection being combined with the early exploration caused by optimistic values, leading to it finding and settling on a highly rewarding branch early on in the process.

By sources of information, Upper Confidence Bounds is comparable to ϵ / standard - greedy; it only adds a simple histogram of past choices to the equation. This ends up being a valuable improvement, as it is able to bring the algorithm up to the standards of the other more complicated algorithms. It is worthy to mention these algorithms have access to more information.

Accuracies: As expected, the accuracies for Bernoulli are higher due to the boolean rewards that are used. These allow for an easier distinction between profitable and unprofitable arms, since the rewards are not fuzzy anymore.

The shape and ranking of the accuracy curves for the algorithms between distributions is generally similar, however the biggest visible difference is that the optimistic initial values algorithm performs much better for the Bernoulli distribution. This may be due to the boolean rewards given by the Bernoulli distribution, meaning that accurately estimating the probability would allow it to find the generally best performing arm.

The action preferences algorithm is eventually capable of overtaking all other algorithms in accuracy, mainly because it finally manages to learn the available information. It requires, however, about 300 time steps before it manages to perform better than optimistic initial values.