Released: 8th Apr Deadline: 12th May

Final Project

INF368A - Reinforcement Learning
Spring 2023

Part I.
Multi-Armed Bandits

In the first part (*Multi-Armed Bandits*), you are required to provide a report of maximum 1 page describing how you analyzed the problem, explaining the solution you designed, and showing the results you obtained.

Exercise1 MAB

Recall the old scenario for evaluating the effectiveness of three medicines A, B, C to fight headache. We previously solved this problem with a multi-armed bandit formalization aimed at discovering the most effective medicine in the most efficient way.

Consider now the scenario where again we have to evaluate the effectiveness of three new medicines A, B, C in fighting headache. However, in the current case, we want to take into consideration also effects of assuefaction and tolerance change among the population. This means that the effect of the drugs we developed may change with time as the biological response to the medicine adapts across the population. Practically, we will simulate this by having the average effectiveness of the three drugs A, B, C randomly change at random time steps.

1. How would this modified problem fit in the MAB or in the MDP formalism? What challenges arise in the current setting?
2. Assume your aim is still to minimize regret or maximize reward. How would you change or adapt a MAB algorithm to deal with this scenario in which you do not know when changes in effectiveness happen? Explain and justify your algorithmic choices. Describe assumptions and/or dependence of your solution on the parameters of the problem.
3. Download the file `bandit.py` and run the environment `Bandits_final()` representing the scenario discussed above for (at least) 1000 episodes. Run the algorithmic solution proposed above alongside (at least) one of the standard MAB algorithm discussed in the course. Compare and comment on the result.

Part II.
Reinforcement Learning

In the second part (*Reinforcement Learning*), you are required to provide a report of maximum 1 page describing how you analyzed the problem, explaining the solution you designed, and showing the results you obtained. You are also required to submit the code of your trained agent in the format specified in the `README.md` file.

Exercise2 RL

Recall the old problem of a robot patrolling a factory floor and retrieving cookies falling from a conveyor belt. We previously solved this problem using standard reinforcement learning algorithms producing an optimal policy for the given environment.

Consider now the problem of training an agent able to patrol a number of factories with varied characteristics: floors with different lengths (longer or shorter than 10 meters), different falling objects (cookies, cakes, scones), different lifetime of the objects on the floor (longer or shorter than 5 seconds), different formulation for friction. Practically you will be given a few environments on which to train your agent, and at the end we will consider running it on unseen environments.

1. How would this problem be expressed in the MDP formalism? What challenges arise in the current setting?
2. Assume your aim is to train an agent to maximize reward on a set of possible environments. How would you change or adapt your training to deal with this scenario? Explain and justify your algorithmic choices. Describe assumptions and/or dependence of your solution on the parameters of the problem.
3. Download and instantiate the environments `cookiedisaster-v1`, `cookiedisaster-v2`, `cookiedisaster-v3`, train and run your agent there. Compare and comment on the result across the environment.
4. Prepare your trained agent for submission according to the template `BaseAgent.py`, as specified in the `README.md` file. Notice that your agent does not have to be trained on the provided environments. Your agent will be tested on unseen environment.

Part III.
Proximal Policy Optimization

In the third part (*Proximal Policy Optimization*), you are required to provide a report of maximum 1 page on the code you implemented and the results you collected. Notice that the questions in the initial four exercises are all *optional*. Exercise3, Exercise4, Exercise5 are preliminaries designed to familiarize you with possibly new concepts; if you are already familiar with them, feel free to skip these exercises. Exercise6 is meant to guide you through the reading of the assigned paper. Exercise7 contains the deliverables.

Exercise3 Entropy

During the RL course, we worked with distributions, particularly with stochastic policies representing probability distributions over actions. Measuring and characterize a single distribution is crucial to work with them.

Consider the following discrete distributions:

- $p_1 = [.2, .2, .2, .2, .2]$ on domain $\Omega_1 = [0, 1, 2, 3, 4]$;
- $p_2 = [.1, .2, .4, .2, .1]$ on domain $\Omega_1 = [0, 1, 2, 3, 4]$;
- $p_3 = [.1, .2, .4, .2, .1]$ on domain $\Omega_2 = [.0, .2, .4, .6, .8]$
- $p_4 = [.1, .2, .2, .4, .1]$ on domain $\Omega_2 = [.0, .2, .4, .6, .8]$.

We said that the expected value provides a basic statistics to summarize a distribution.

1. Compute the expected value of the random variable X with the distributions above, that is, $E_{p_1}[X], E_{p_2}[X], E_{p_3}[X], E_{p_4}[X]$.
2. Which distributions above are undistinguishable when you summarize them with the expected value?

Not surprisingly, multiple different distributions may have the same expected value. Adding more descriptors may help us discriminating better.

3. Compute the variance of the random variable X with the distributions above, that is, $Var_{p_1}[X], Var_{p_2}[X], Var_{p_3}[X], Var_{p_4}[X]$.
4. How does the variance help with the problem of discrimination?

Considering expected value and variance helps discriminating, but it does not solve the fundamental problem; we may still have different distributions with identical expected value and variance. The more descriptors we have, the more fine-grained distinctions we can make.

Notice also, that p_2 and p_3 have the same shape although they have different variance. Sometimes we may be interested in considering only the shape of a distribution, independently of the value of the underlying domain.

5. Why could we be interested only in the shape of a distribution?
   A descriptor summarizing the shape of a distribution is the entropy, defined as:

$$H_p[X] = -\sum_{x \in \Omega} p(x) \log p(x).$$

6. Compute the entropy of the random variable X with the distributions above, that is, $H_{p_1}[X], H_{p_2}[X], H_{p_3}[X], H_{p_4}[X]$.
7. Which distributions are now undistinguishable?
8. Can you explain in which sense p_4 has the same shape as p_2 or p_3?
9. Are indeed the values of the domains Ω_1, Ω_2 used in the computation of the entropy?
10. How does the entropy relate to the spread of the distribution? Does entropy increase or decrease if the distribution becomes more concentrated?
11. Why might we want to promote entropy in RL? What would happen if the entropy is minimized? What advantage would you get from promoting entropy?

Exercise4 KL

While descriptors like expected value, variance and entropy allow us to characterize a single distribution, we often have to work with multiple distributions. For instance, we may have to relate multiple policies and assess how similar they are.

We could compare them using the descriptors above, but, because of the undistinguishability problem that we have experienced, we may end up judging different distributions as identical.

The Kullback-Leibler (KL) divergence is a function that provides us with an (asymmetric) measure of dissimilarity between two distributions:

$$KL(p; q) = \sum_{x \in \Omega} p(x) \log \frac{p(x)}{q(x)}$$

It evaluates the divergence in terms of shape and location of the mass of probability.

1. Can you explain why KL(p;q) is asymmetric in p, q?
2. Compute $KL(p_1; p_1)$. Does the result make sense?
3. Compute $KL(p_1; p_2)$. Does the result make sense?
4. Compute $KL(p_1; p_3)$. Does the result make sense?

If we are learning a distribution q and we are trying to approximate or approach another distribution p, we can use the KL divergence to evaluate how close we are.

Exercise5 First- and second-order optimization, and trust region optimization

By Taylor decomposition, we know that a continuous function $f(x)$ can be approximated around point p with polynomials. A first-order approximation is:

$$f(p) + \frac{1}{1!}f'(p)(x-p),$$

a second-order approximation is:

$$f(p) + \frac{1}{1!}f'(p)(x-p) + \frac{1}{2!}f''(p)(x-p)^2,$$

a third-order approximation is:

$$f(p) + \frac{1}{1!}f'(p)(x-p) + \frac{1}{2!}f''(p)(x-p)^2 + \frac{1}{3!}f'''(x-p)^3,$$

and so on.

Let us now consider a function $f(x)$ and suppose we want to find the value x that minimizes the function $f(x)$. We will adopt a gradient descent approach, that is, we will start at a given value of x and move downhill.

1. Assume our function is:

$$f(x) = \frac{1}{4}x^3 + \frac{3}{4}x^2 - 2x + 1$$

   Plot the function on the domain $x \in [-4, 4]$.
2. Assume that we are currently at value $x = 2$. What is the value of $f(x)$ at $x = 2$?

We want to improve our current solution $x = 2$ by gradient descent.

3. First, compute the first derivative $f'(x)$.
4. Then, compute the Taylor approximation of first-order around $x = 2$.
5. Plot the first-order Taylor approximation.
6. What information is the first-order Taylor approximation providing and how would you use it to perform gradient descent?
7. Compute now also the second derivative $f''(x)$.
8. Compute the Taylor approximation of second-order around $x = 2$.
9. Plot the second-order Taylor approximation.
10. How would you use the additional information of the second-order Taylor approximation to perform gradient descent?

Second-order optimization methods (such as Newton method) use information contained in the second derivative to perform gradient descent.

11. If there is more information in the second derivative, why are not second-order optimization methods more widely used in machine learning?

When minimizing $f(x)$, the gradient descent methods we considered above are examples of *line-search methods*: we consider the solution in current position p, we compute the derivative (first- or second-order) and then we perform an ϵ step in that direction.

An alternative approach is offered by *trust-region methods*: given the solution in current position p, we first find the largest region R within which the function $f(x)$ may be approximated by a simpler (first- or second-order) function $g(x)$, and then we move to the minimum of $g(x)$.

Exercise6 PPO

Retrieve and read the paper `https://arxiv.org/pdf/1707.06347.pdf`.
Consider the **Introduction** section (Section 1).

1. *Challenges:* what are the challenges and the limitations in the field detected by the authors?
2. *Aim:* what do the authors want to achieve with their work?
3. *Contributions:* what are the contributions of this paper to the field?
4. *Evaluation:* how do the authors assess the quality of their results?

Consider the **Policy Gradient Methods** section (Section 2.1), where policy gradient is explained.

5. Relate Equation (1) to the corresponding formula we used in the course.
6. In the course, we used the notation $f_\pi(a, s|\mathbf{w}_\pi)$ and $\mathbf{w}_\pi$ for a policy function estimator and its parameters. What notation does the paper use?
7. Explain what the paper means by *using automatic differentiation software*.
8. Consider how the paper goes from Equation (1) to Equation (2). It goes in the opposite way compared to what we did in the course. Can you make sense why it does that?
9. Express policy gradient as the maximization of the objective in Equation (2).

Consider the **Trust Region Methods** section (Section 2.2), where an alternative approach to policy gradient is presented.

10. What is the meaning and role of a surrogate objective function?
11. What is meaning of $\pi_{\theta_{old}}$?
12. What is the role of $\pi_{\theta_{old}}$ in the maximization of Equation (3)? How would $\pi_{\theta_{old}}$ behave if we were to take the gradient ∇_θ of the expectation in Equation (3)?
13. What do you expect is the effect of the ratio $\frac{\pi_\theta}{\pi_{\theta_{old}}}$ in Equation (3)?
14. What does the constraint of Equation (4) add?
15. How does the objective overall change from Equation (2) to Equation (3-4)?
16. Two TRPO maximization problems are presented, a hard version with constraints in Equation (3-4) and a soft version with a trade-off parameter β in Equation (5). Which maximization problem is encoded in standard TRPO?
17. Why does standard TRPO use one maximization problem instead of the other?
18. Does the solver for standard TRPO use first-order or second-order optimization?

Consider the **Clipped Surrogate Objective** section (Section 3), where the main algorithm is presented.

19. How does Equation (6) relate to Equation (3-4)?
20. What is the limitation of Equation (6)?
21. How does Equation (7) relate to Equation (3-4)?
22. Consider Figure 1. Explain the meaning of the axes.
23. Consider Figure 1. Why are there two plots? Can L^{CLIP} be negative for $A > 0$, or L^{CLIP} be positive for $A < 0$?
24. Consider Figure 1. What value does π_θ assume at the red dot?
25. *Critical questions:* would you have any question to ask the authors about this algorithm?

Consider the **Adaptive KL Penalty Coefficient** section (Section 4), where an alternative baseline algorithm is presented.

26. Recall the soft formulation of the TRPO problem from Section 2.2. What was the main difficulty in implementing this objective?
27. How do the authors suggest to tackle that challenge now?
28. How many hyperparameters does their approach introduce?
29. What is the role of the Adaptive KL penalty coefficient loss?
30. *Critical questions:* would you have any question to ask the authors about this algorithm?

Consider the **Algorithm** section (Section 5), where the actual PPO algorithm is built.

31. How do the authors suggest you would compute the gradient of L^{CLIP} or L^{KLPEN}? How would that work?
32. Why are the authors considering the idea of computing $V(s)$?
33. What class of RL algorithms would PPO belong to?
34. We considered a few methods in class for estimating $V(s)$; however the authors proposed different solutions. What are they suggesting?
35. The authors consider the possibility of a network sharing parameters between the policy and the value function. Where did we see in class a similar architecture?
36. What is the rationale for adding an entropy bonus?
37. Consider Equation(9). Explain the role of each term (L^{CLIP}, L^{VF}, S) in making up the overall loss ($L^{CLIP+VF+S}$).
38. Consider Equation(9). What is the role of c_1 and c_2?
39. Consider Equation(9). How would you practically deal with the expected value $\mathbb{E}$ when you implement your algorithm?
40. The authors follow the approach of running T timesteps and then compute the estimator in Equation (10). What approach does this remind you?
41. What approach instead would you have if you were to run episodes until the end instead of T timesteps?

42. Equation (11) provides a generalization of Equation (10). Verify that indeed Equation (11) reduces to Equation (10) when $\lambda = 1$.
43. The final PPO algorithm relies on another couple of tricks: parallelization and mini-batches. Explain them.
44. *Critical questions:* would you have any question to ask the authors about this algorithm?

Consider the **Comparison of Surrogate Objectives** section (Section 6.1), where different PPO losses are compared.

45. Reconnect the losses at the end of Page 5 with the Equations in the text.
46. *Environment*: on which environments did they run their experiments? What are the action sets?
47. *Hyperparameters*: which hyperparameters are involved in the models and what is their meaning?
48. *Hyperparameters*: which hyperparameters are fixed and which were searched over?
49. *Hyperparameters*: why are there no hyperparameters for c_1 and c_2?
50. *Hyperparameters*: why are some hyperparameters fixed and others not?
51. *Architecture*: can you graphically represent the architecture of the neural network they implemented?
52. *Architecture*: can you explain the output of the neural network in relation to the actions in the environment considered?
53. *Evaluation metrics*: why did they compute average total reward over the last 100 episodes?
54. *Evaluation metrics*: how do they normalize the scores?
55. *Results*: why is the average over 21 runs?
56. *Results*: check Table 1. Why is there no score of 1?
57. *Results*: check Table 1. How come there are values below 0?
58. *Critical questions:* would you have any question about the setup?
59. *Critical questions:* what would you conclude from the results? Would you consider running other experiments?

Consider the **Comparison to Other Algorithms in the Continuous Domain** section (Section 6.2), where PPO is compared to other algorithms.

60. Only one PPO loss is being considered now. Refer to the Equation expressing the loss.
61. What is the implicit justification for focusing on this single loss?
62. *Environment:* on which environments did they run their experiments?
63. *Algorithms:* how many algorithms do they compare with?
64. *Hyperparameters:* how are hyperparameters chosen? Do they still explore possible values for certain hyperparameters? If not, how do they choose them?

65. *Evaluation metrics*: how do they compare the algorithms?
66. *Results:* review Figure 3. What is the meaning of the shaded areas around the lines?
67. *Results:* review Figure 3. What would you infer from the plots?
68. *Critical questions:* would you have any question about the setup?
69. *Critical questions:* what would you conclude from the results? Would you consider running other experiments?

Consider the **Showcase in the Continuous Domain** section (Section 6.2), where PPO is benchmarked on another task.

70. What is the purpose of this further experiment?
71. *Environment:* what is the environment and the tasks they consider? What is the challenge in these environment?
72. *Algorithms:* how many algorithms do they compare with?
73. *Evaluation metrics*: how do they compare the algorithms?
74. *Results:* review Figure 4. Why do we have multiple lines in each plot?
75. *Results:* review Figure 4. What information can you infer from looking at the x-axis?
76. *Critical questions:* would you have any question about the setup?
77. *Critical questions:* what would you conclude from the results? Would you consider running other experiments?

Consider the **Comparison to Other Algorithms on the Atari Domain** section (Section 6.4), where PPO is benchmarked on further tasks.

78. What is the purpose of this further experiment?
79. *Environments*: on which environments did they run their experiments?
80. *Architectures and hyperparameters:* how are architecture and hyperparameters selected?
81. *Algorithms:* which algorithms do they compare?
82. *Evaluation metrics:* what metrics do they use to compare the algorithms? What is the meaning of these metrics?
83. *Results:* review Table 2. How would you characterize the difference between ACER and PPO?
84. *Results:* review Figure 6. Does it look like the results in Table 2 are backed up by these plots?
85. *Critical questions:* would you have any question about the setup?
86. *Critical questions:* what would you conclude from the results? Would you consider running other experiments?

Consider the **Conclusion** section (Section 7), where the main contributions are summarized.

87. What are the main conclusions of this paper?
88. *Critical questions:* are you left with any question?

Exercise7 PPO Implementation

Reconsider the paper `https://arxiv.org/pdf/1707.06347.pdf`.

1. Take an implementation of your actor-critic algorithm and transform it by changing its objective to the equivalent of the clipping objective of Equation (7).
2. Run your algorithm on the `InvertedPendulum` environment in *gymnasium*. Explore the results for different configurations of relevant parameters.
3. Compare the results across different hyperparameter settings and against the results presented in the paper.