3. Assignment 2: REINFORCE and basic Actor-Critic Methods

3.1. Motivation

In the last assignment you have familiarized yourself with the use of neural networks for approximating value functions. Yet, the policy so far was hard-coded, as we behaved (epsilon-)greedy w.r.t. our current value estimate.

Policy Network In the first part of the assignment we will take a step back, and explore the REINFORCE algorithm (). In brief, the idea of this algorithm is to parametrize the policy by a neural network, while only providing a Monte-Carlo estimate for the value. In other words, we drop our value network and estimate the values in a hard-coded way, but now have a policy network. Our policy network is a function

$$\pi_\theta : \mathcal{S} \rightarrow \Delta(\mathcal{A}), \tag{1}$$

that maps each state to a distribution over action spaces, parametrized by a θ. If θ_k denotes our current set of parameters, the loss function for REINFORCE (also referred to as the vanilla policy loss) is the following:

$$\mathcal{L}(\theta) = \mathbb{E}_{s\sim\mathcal{D}}\left[\mathbb{E}_{a\sim\pi_\theta(a|s)}[-Q^{\pi_{\theta_k}}(s,a)]\right]. \tag{2}$$

This loss can then be used to update our policy by gradient descent. Note, that we would have to a step of gradient *ascent*, if we would remove the minus-sign. The intricacy is now, that we don't have access to a value-function and have to estimate $Q^{\pi_{\theta_k}}(s,a)$ via Monte-Carlo estimation (as you did it in tabular environments).

Actor-Critic Methods Why not combine these two ideas, you may ask. Indeed, this is exactly what we will do now. We now assume, that we have a parametrized policy network π_θ as above (called the actor) and a value network Q_ϕ (called the critic), that tries to approximate the policy $Q_\phi \approx Q^{\pi_\theta}$. Methods, that use both a policy and a value network are then referred to as Actor-Critic methods, and most modern RL algorithms follow this approach. For the policy network and its loss, we can copy-paste what we had for REINFORCE, but we need to adjust the Q network, as we can not simply plug-and-play our Q-Learning loss (why?). The resulting algorithm will then repeat the following steps:

(i) Sample data,

(ii) Update Policy network via gradient descent with policy loss,

(iii) Update value network via gradient descent with value loss.

Does this remind you of something?

Advantage Actor-Critic (A2C) We have learned that neural networks seldom work out of the box, and we have to use some tricks to make things numerically more stable. One of these tricks is to use the advantage function, which is defined as $A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s)$ rather than the Q-function. The resulting loss is the following:

$$\mathcal{L}(\theta) = \mathbb{E}_{s\sim\mathcal{D}}\left[\mathbb{E}_{a\sim\pi_\theta(a|s)}[-A^{\pi_{\theta_k}}(s,a)]\right]. \tag{3}$$

Interestingly, to simplify implementation, we usually implement a V network, i.e. $V_\varphi : \mathcal{S} \rightarrow \mathbb{R}$, and make a MC estimate (as in REINFORCE) for the Q-value, and use both to calculate the advantage. As use an Actor-Critic approach with advantages, this is often referred to as Advantage Actor-Critic (A2C).

3.2. Tasks

The assignment will resolve around two major parts: A more theoretical component and some experiments.

Theory

This assignment covers a lot of new ideas, that need to be explained. First, while we have provided you the expression for the policy Loss in Eq. (2), and describe how this can be used for updating the policy. Using this, you should explain the steps in the REINFORCE algorithm (if you use pseudo-code, at most 8 lines).

Moving to Actor-Critic Methods (Bonus if you can provide a reason for the choice of the terms ”actor” and ”critic”), we want you to explain, what would prevent us to use the Q-Learning loss in this basic Actor-Critic approach. Additionally, you should explain which loss should be used instead. This sets you up, to explain the steps, that our basic AC algorithm follows (if you use pseudo-code, at most 8 lines).

Lastly, we want you to describe why we can use the advantage loss Eq. (3) instead of the loss in Eq. (2). Also please explain how your A2C algorithm would then work (please use pseudo code, around 10 algorithmic steps). For more information you may also check spinning up.

Experiments

We stay in the Cartpole environment. Implement all the three aforementioned algorithms (REINFORCE, AC, A2C), all with the same policy network. The results of these algorithms should be compared to your results from Assignment 1, you may also reuse the baseline provided there as a point of reference. Discuss your results.

3.3. Task-Checklist

1.1 Explain the origin of the vanilla policy loss

1.2 Explain the REINFORCE algorithm

1.3 Explain your basic Actor-Critic Algorithm (especially explain your Q-Network + loss) (pseudo code required)

1.4 Explain why we can use Advantages over Q-Values.

1.5 Explain your Advantage Actor-Critic Algorithm (including the loss for your value network) (pseudo code required)

2.1 Implement REINFORCE

2.2 Implement AC

2.3 Implement A2C

2.4 Plot the Learning Curves (Return over environment steps) in a single graph, also reference the results of Assignment 1.

Bonus To obtain a bonus of up to 1 point, you can do either perform additional experiments or answer some of the additional questions mentioned in the text above. Alternatively we reward concise (yet complete) reports, where we reward half a point per page below the 6 page limit (up to a full point). Please note that this should not come at the cost of the completeness of your assignment, writing concisely is challenging and requires work.

3.4. Submission

Make sure to nicely document everything that you do. Your final submission consists of:

(i) Your source code, together with a requirements file and a README with instructions that allows us to easily (single command per experiment / sub task) rerun your experiments on a university machine booted into Linux (DSLab or computer lab). To generate a requirements file boot up a terminal, activate your virtual/conda environment (if you used one), navigate to the folder in which you want to generate the file and run the following command: `python -m pip freeze > requirements.txt`.

(ii) You have to use the ICML Latex template for your Report. (Obtainable here)

(iii) Chapter Structure: Abstract (at most quarter of a page), Introduction, Theory (Theory only, around one page), Experiments (setup & results; explaining environment should take at most quarter of a page), Discussion (discussion of results & weaknesses), Conclusion (reflect own work & outline future work).

(iv) A self-contained scientific pdf report of 4-6 pages with figures etc. This report contains an explanation of the techniques, your experimental design, results (performance statistics, other measurements,...), and overall conclusions, in which you briefly summarize the goal of your experiments, what you have done, and what you have observed/learned.

If you have any questions about this assignment, please visit our lab sessions on Friday where we can help you out. In case you cannot make it, you can post questions about the contents of the course on the Brightspace discussion forums, where other students can also read and reply to your questions.

3.5. Deadline

April 11, 23:59:59

References

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 1861–1870. PMLR, July 2018.

Haarnoja, T., Zhou, A., Hartikainen, K., Tucker, G., Ha, S., Tan, J., Kumar, V., Zhu, H., Gupta, A., Abbeel, P., and Levine, S. Soft Actor-Critic Algorithms and Applications, January 2019.

Hessel, M., Modayil, J., Van Hasselt, H., Schaul, T., Ostrovski, G., Dabney, W., Horgan, D., Piot, B., Azar, M., and Silver, D. Rainbow: Combining improvements in deep reinforcement learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Mnih, V. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal Policy Optimization Algorithms, August 2017.

Sutton, R. S. Reinforcement learning: An introduction. *A Bradford Book*, 2018.

Towers, M., Kwiatkowski, A., Terry, J., Balis, J. U., De Cola, G., Deleu, T., Goulao, M., Kallinteris, A., Krimmel, M., KG, A., et al. Gymnasium: A standard interface for reinforcement learning environments. *arXiv preprint arXiv:2407.17032*, 2024.