Assignment 2: Deep Q Learning with LSTD [15 points]

Deadline: 22 March 2024, 23:55 CET

Description:

1. **[3 points]** Devise the Least Squares Temporal Differences (LSTD) version of the deep Q-learning algorithm. The algorithm should use a deep neural network for nonlinear feature extraction. For instance, it could be an auto-encoder that maps a state to a lower-dimensional latent embedding. The action-value function should be a linear function of the output of the feature extractor. Express your algorithm as pseudocode and justify your design choices.
2. **[2 points]** Implement your algorithm in PyTorch and evaluate its performance on the Acrobot, Pendulum, and Mountain Car environments of OpenAI Gym. Apply a proper software engineering practice in your design and write descriptive comments wherever required.
3. **[6 points]** Train your algorithm until you observe it to solve the control tasks. Run your algorithm in the evaluation mode in reasonable regular intervals (e.g. every 10 episodes) for five repetitions, each with different random seeds and store its total episode reward for each repetition. Plot the evolution of its "evaluation-time" performance at the end of the training as a curve that contains the number of gradient-descent steps taken until an evaluation run on the x axis and the mean of the total episode rewards over five repetitions on the y axis. Plot the resulting curve in solid lines and shade its one-standard deviation surroundings.
4. **[4 points]** Write a one-paragraph explanation of the resulting learning curves. Inspect them from angles including but not limited to: i) the comparative training step counts to solve different tasks, ii) the comparative stability of the algorithm across different environments, iii) the absolute stability of the algorithm across different repetitions, iv) the effect of the capacity of the used autoencoder on performance.

This is an individual assignment. The submitted deliverables are your own intellectual properties. Group discussions on solutions, brainstorming, and comparing your results with each other are allowed. However, implementation and reporting should be done on individual basis. As the implementation platform, use Python and PyTorch.

Please submit the following deliverables via the itsLearning portal (emails do not count as submissions):

- Your one-page long report in PDF format that contains pseudocode, a one-paragraph justification of its design choices, results figures, the verbal explanation of the results, and any other details that are necessary to understand and run your code.
- All source code required to replicate the numbers and draw the figures you use in your report in Python (.py) files that saves the result plots in PNG format with names *"learning-curve-acrobot.png"*, *"learning-curve-pendulum.png"*, *"learning-curve-mountaincar.png"*

until March 22nd, 2024 23:55 Copenhagen Time. Late submissions will be graded according to the formula below:

[Valid Points] = max([Earned Points] – [Number of Calendar Days after DL]*3, 0).

For example, [Number of Calendar Days after DL] = 1 if you submit on March 22 at 23:56, and [Number of Calendar Days after DL] = 2 if you submit on March 23, 23:56 and so forth.