Reinforcement Learning Assignment

Assignment Overview

Following our previous discussion, you have been assigned a Reinforcement Learning task. This task will help you understand the basics of APIs and RL and how they can be applied to solve simple yet interesting problems.

Your objective is to train an RL agent to navigate through a 5x5 maze until it reaches the final goal. This environment is in a remote VM, and you can interact with it through API in order to train and evaluate your agent. For this task, you can use a library to implement the selected algorithms to train your agents or even use your own algorithms. We strongly recommend using stable-baselines3 for this task.

Environment Setup

In order to set up an RL environment, we need to:

- Install Python on your local machine (version greater than 3.4+)
- Install prerequisites libraries
    - OpenAI Gym (Gymnasium)
    - Stable-baselines3 (or your preferred library to train your agents)
    - Requests (for the API calls)

First, familiarise yourself with the OpenAI Gymnasium logic and functions.
You can find more information and examples in the following link:
https://gymnasium.farama.org/index.html#

More information regarding the stable-baselines3 library can be found here:
https://stable-baselines3.readthedocs.io/en/master/

Then, create a Gymnasium env Wrapper class that makes the necessary calls to the API provided in order to have a proper connection to the environment. Information regarding the calls, parameters and the structure of the API can be found in the swagger file here: [redacted]

Algorithm Selection

After setting up the environment, you need to train our first agent using an appropriate algorithm. We recommend using two of the most common algorithms, DQN (Deep Q Network) and PPO (Proximal Policy Optimization), which can be also found in stable-baselines3. This library provides a large variety of algorithms and tools to the user in PyTorch. You can use as many algorithms as you want or whichever you think is the best algorithm for this problem.

Training and Evaluation

Once you have implemented your chosen algorithm, you need to train the agent. After the agent is trained, we must evaluate the trained agent's performance in terms of successful episode completion and average reward. You can also use evaluate_policy from stable-baselines3.

Objectives

The acceptance criteria for this task are the following:

- First, set up the connection between your machine and the API and check that every call is executed successfully.
- At least two of the algorithms must be implemented. When the agent is trained with these algorithms, try to fine-tune the hyperparameters to achieve maximum reward (at least 80 mean reward and maximum 20 steps until it reaches the final goal).
- When the training is completed, provide the training curves from tensorboard (Mean Episode Reward).
- Evaluate the agents and provide mean episode reward and standard deviation for each algorithm for 100 episodes.
- **Optional**: Reproduce the same experiment, only this time try to implement a custom Wrapper to apply a different reward logic based on your observations in order to increase the possible reward.

The deliverable must include a pdf report detailing the progress of the task, a brief overview of the implemented algorithms, final results containing images from TensorBoard and the necessary scripts along with the training/evaluate scripts for each algorithm.

Use of AI tools like Copilot, ChatGPT, or similar for this task is strictly prohibited. We employ sophisticated detection methods to identify any use of such tools. Violation of this policy will result in immediate termination of the process. Please rely solely on your own skills and the provided resources. Contact us for any clarifications.