Model-free reinforcement learning

Quiz, 5 questions

Congratulations! You passed!

Next Item

1 / 1 point

1.

How is a model-free RL algorithm different from a model-based one?

- [x] Model-free algorithm does not rely on knowing environment dynamics: P(s'| s, a)

Correct

- [ ] Model-based algorithms rely on machine learning models (e.g. neural networks)

Un-selected is correct

- [x] Model-based algorithm knows rewards in advance for every state and action.

Correct

- [ ] Model-based algorithms always learn by interacting with physical environment

Un-selected is correct

- [ ] Model-free algorithms know all environment states in advance

Un-selected is correct

---

1 / 1 point

Model-free reinforcement learning

Quiz, 5 questions

2.

Imagine your are building an agent for Taxi-v2 env and you want to train it with (tabular) Q-learning. What's the minimal set of trainable parameters you will have to maintain?

- Best actions and action Q-value for each of 500 states = 1 000
- 500 Q-values and 500 optimal actions = 1 000
- Q-values for each of 500 states, 6 actions and 500 next states = 1 500 000
- Q-values for 6 actions and state values for 500 states = 506
- Q-values for 500 states * 6 actions = 3 000

Correct

---

1 / 1 point

3.

There's an atari game called "Skiing" in which player takes a role of a mountain skier that has to slide down the route and pass as many gates as possible along his way. Reward function r(s,a) for skiing is computed roughly like this:

- For every time-step that passes your agent receives -1 reward (i.e. -4 if it acts once every 4 frames).
- At the end of a game your agent receives -1000 for each gate he missed.
- A game lasts roughly 5000 time-steps

If you train a Q-learning agent for that game it will almost certainly fail to learn anything even.

- e-greedy exploration, e=0.5 and slowly decaying to 0
- gamma=0.99
- training on-policy
- best hand-crafted features you can imagine

Any ideas what could possibly go wrong?

- The problem is that agent's reward for passing first several gates will only manifest at the game end (in 5000 ticks). This causes agent to ignore main reward due to the way discounted rewards work.

Correct

- The problem is in the way rewards are organized. An agent is very likely to end up with a negative total reward for a session. Algorithms like Q-learning are not guaranteed to work in case of negative rewards.

- The problem is in the algorithm itself. Q-learning is an off-policy algorithm and it can't be used in an on-policy setting.

1 / 1 point

4.

You want to train a reinforcement learning algorithm that moves a bipedal human-like robot to walk forward. You have 1000 recordings from human pilots that perform decently but with some systematic errors. Finally, you have the robot itself and a couple of engineers who can help you with it if you ask them politely.

You don't have a Totally Accurate Robot Physics Simulator (tm) for this task.

How can you train such an algorithm? Please select the option which seems most practical to you.

- [x] Implement Expected Value SARSA with e-greedy exploration, intialize epsilon to something small. Pre-train Q-values on human sessions with state value expectation term computed using e-greedy policy. Then load them into the robot and continue learning with Expected Value SARSA.

Correct

- [ ] Take those 1000 human sessions and train on them using simple SARSA. Then take the Q-values, load them into the robot and continue learning with Q-learning but keep exploration to a minimum.
- [ ] Implement Expected Value SARSA or Q-learning with softmax exploration. Pick a gamma around 0.999. Upload it into the physical robot. Bring it outside your lab. Flick the on switch. Enjoy.

1 / 1 point

5.

Which of the following statements are true about Q-learning?

- [x] Q-values learned by Q-learning can be different from those learned by SARSA.

Correct

- [ ] Q-learning can't find optimal policy in any finite number of steps, it only converges asymptotically.

Un-selected is correct

Model-free reinforcement learning

Quiz, 5 questions

- [x] A minimum amount of data to update Q-values table with Q-learning is a tuple of (state, action, reward, next_state).

Correct

- [ ] Q-learning finds a policy that gets highest expected sum of reward per session.

Un-selected is correct

- [ ] Q-learning can't be applied in model-based setting (i.e. when you know P(s', r | s,a) distribution)

Un-selected is correct