SI252 Reinforcement Learning: Homework #06

Due on June 1, 2025 at 11:59 p.m.(CST)

Name: **Zhou Shouchen**
Student ID: 2021533042

Problem 1

Required: OpenAI Spinning Up in Deep RL. Welcome to Spinning Up in Deep RL! This is an educational resource produced by OpenAI that makes it easier to learn about deep reinforcement learning (deep RL). Please study the documents and install the environment with PyTorch version.
(a) Finish the problem set 1: "Basics of Implementation". It includes three exercises: Gaussian Log-Likelihood, Policy for PPO, and Computation Graph for TD3.
(b) Finish the problem set 2: "Algorithm Failure Modes". It includes two exercises: Value Function Fitting in TRPO, and Silent Bug in DDPG.

Solution

(a) After setting up the environment in the methods in 'README.md', we can successfully run the codes. The codes cuold be check in 'code/spinningup/spinup/exercises/pytorch/problem_set_1'.

- exercise 1_1: For a k dimension diagnoal Gaussian distribtion $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, where $\boldsymbol{\Sigma} = \text{diag}(\sigma_1^2, \ldots, \sigma_k^2)$, thus its PDF is

$$p(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{\frac{k}{2}} |\boldsymbol{\Sigma}|^{\frac{1}{2}}} \exp\left[-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1} (\mathbf{x}\boldsymbol{\mu})\right]$$

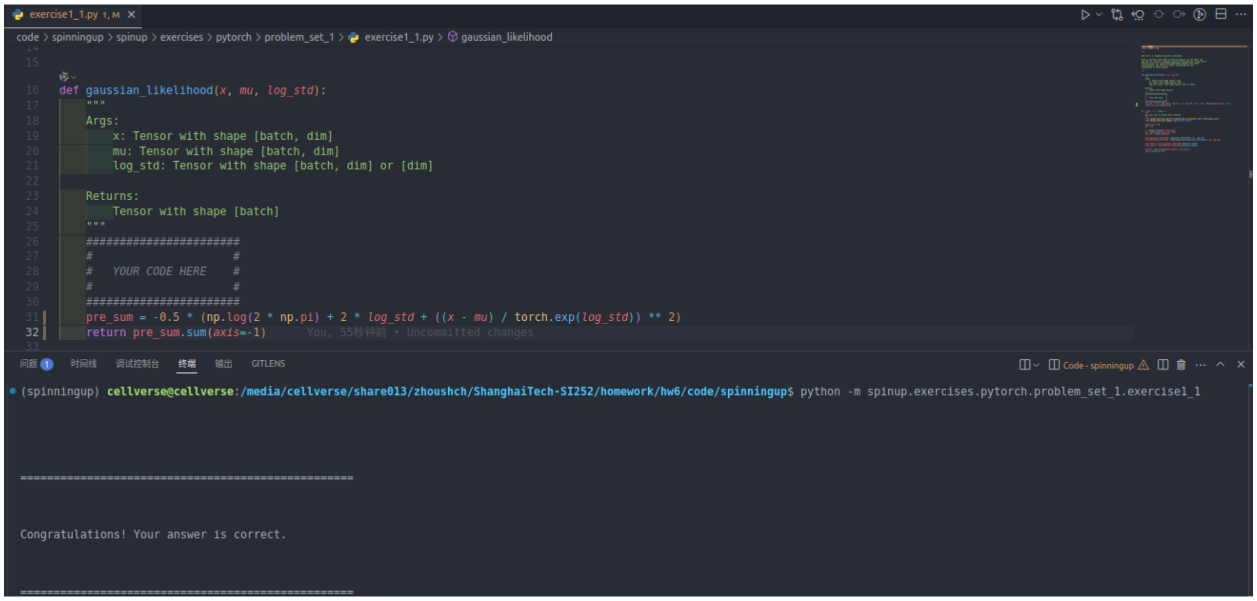
Where

$$|\boldsymbol{\Sigma}| = \prod_{i=1}^{k} \sigma_i^2, \qquad \boldsymbol{\Sigma}^{-1} = \text{diag}\left(\sigma_1^{-2}, \ldots, \sigma_k^{-2}\right), \qquad (\mathbf{x} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) = \sum_{i=1}^{k} \frac{(x_i - \mu_i)^2}{\sigma_i^2}$$

So above all, the log-likelihood is

$$\begin{aligned} \log p(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) &= -\frac{k}{2}\log(2\pi) - \sum_{i=1}^{k} \log \sigma_i - \frac{1}{2}\sum_{i=1}^{k} \frac{(x_i - \mu_i)^2}{\sigma_i^2} \\ &= -\frac{1}{2}\sum_{i=1}^{k}\left(\log(2\pi) + 2\log\sigma_i + \left(\frac{x_i - \mu_i}{\sigma_i}\right)^2\right) \end{aligned}$$

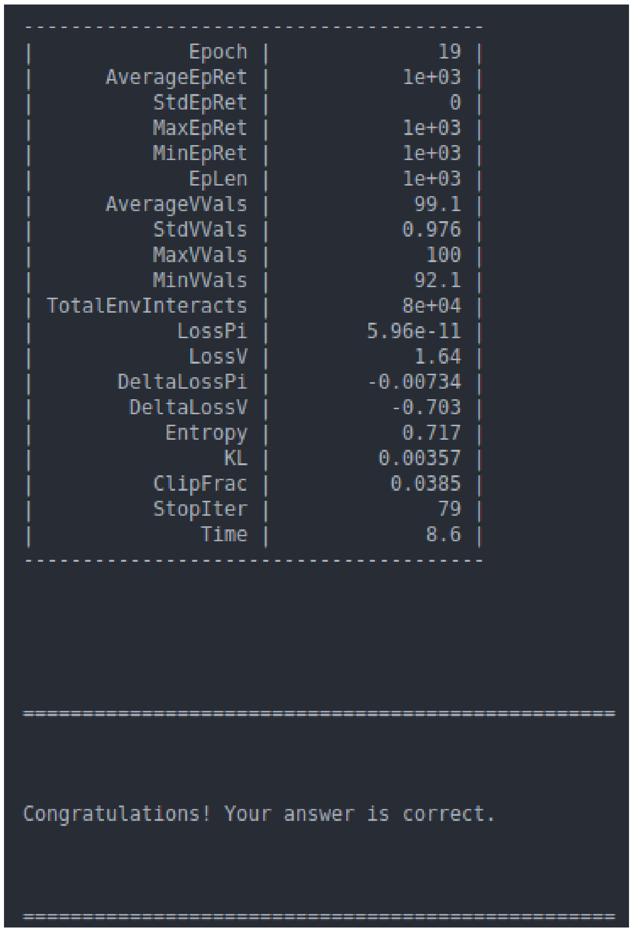
The implementation are check results are as follows:



```
def gaussian_likelihood(x, mu, log_std):
    """
    Args:
        x: Tensor with shape [batch, dim]
        mu: Tensor with shape [batch, dim]
        log_std: Tensor with shape [batch, dim] or [dim]

    Returns:
        Tensor with shape [batch]
    """
    #######################
    #                     #
    #   YOUR CODE HERE    #
    #                     #
    #######################
    pre_sum = -0.5 * (np.log(2 * np.pi) + 2 * log_std + ((x - mu) / torch.exp(log_std)) ** 2)
    return pre_sum.sum(axis=-1)
```

```
(spinningup) cellverse@cellverse:/media/cellverse/share013/zhoushch/ShanghaiTech-SI252/homework/hw6/code/spinningup$ python -m spinup.exercises.pytorch.problem_set_1.exercise1_1



==================================================


Congratulations! Your answer is correct.


==================================================
```

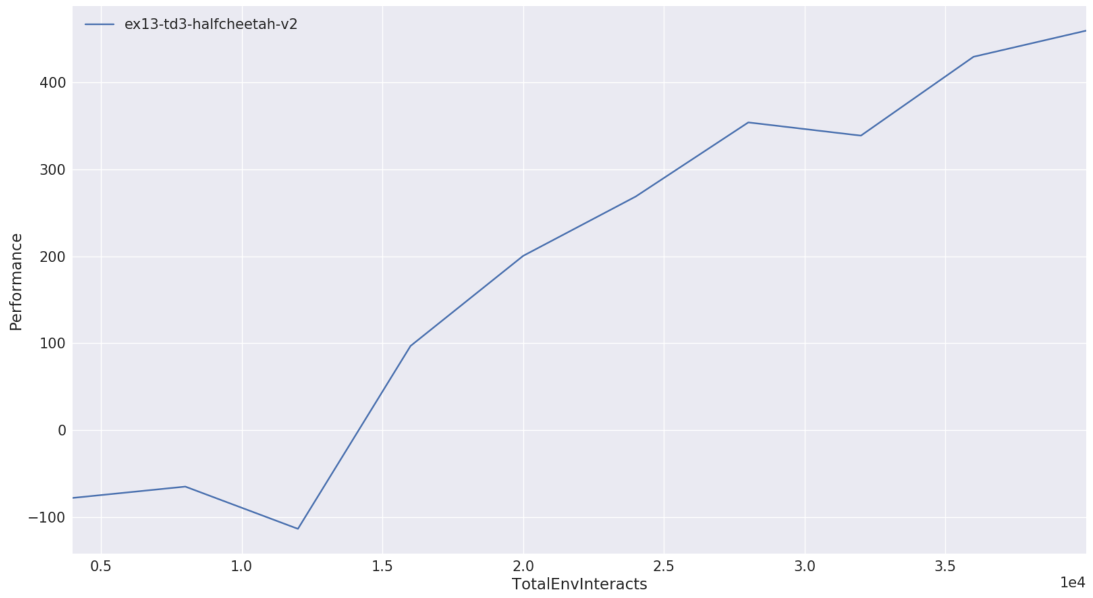
- exercise 1_2: The implementation are check results are as follows:



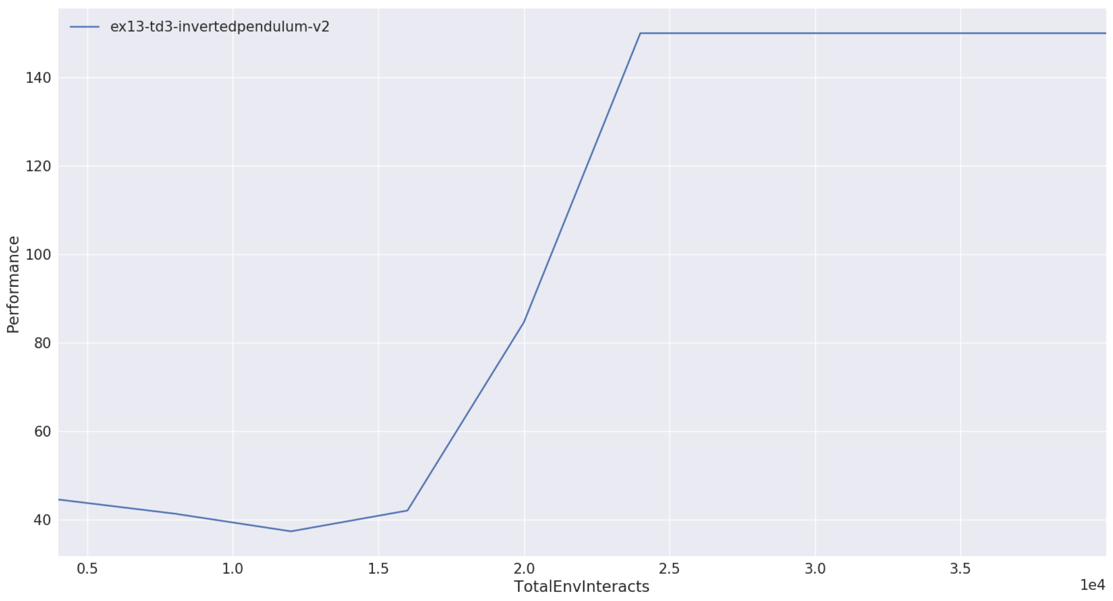
```
---------------------------------------
|             Epoch |              19 |
|      AverageEpRet |           1e+03 |
|          StdEpRet |               0 |
|          MaxEpRet |           1e+03 |
|          MinEpRet |           1e+03 |
|             EpLen |           1e+03 |
|      AverageVVals |            99.1 |
|          StdVVals |           0.976 |
|          MaxVVals |             100 |
|          MinVVals |            92.1 |
| TotalEnvInteracts |           8e+04 |
|            LossPi |        5.96e-11 |
|             LossV |            1.64 |
|       DeltaLossPi |        -0.00734 |
|        DeltaLossV |          -0.703 |
|           Entropy |           0.717 |
|                KL |         0.00357 |
|          ClipFrac |          0.0385 |
|          StopIter |              79 |
|              Time |             8.6 |
---------------------------------------



==================================================



Congratulations! Your answer is correct.



==================================================
```

- exercise 1_3: According to the discription, within 10 rounds, the score in HalfSheetah should exceed 300, while the score in InvertedPendulum should reach 150.

  And the following curves show that the performance achieved the expected results:



Problem 1 continued on next page...

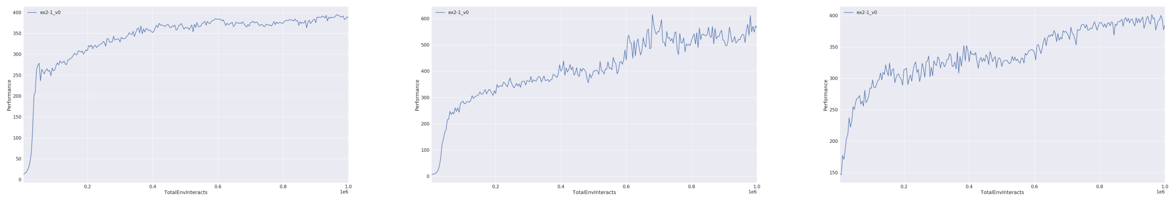


(b) Run the commands in the ‘README.md’, we can get the following reults. The codes cuold be check in ‘code/spinningup/spinup/exercises/pytorch/problem_set_2’.

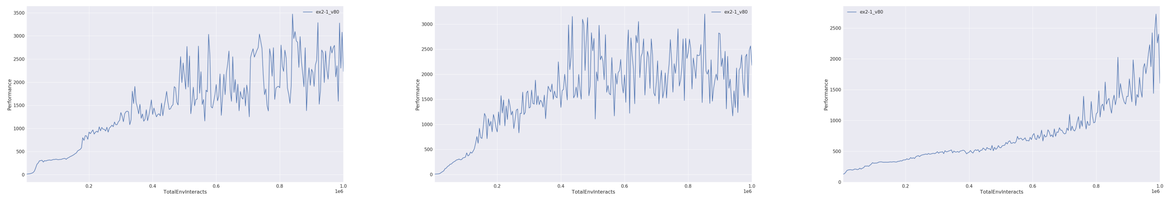
- exercise 2_1:

  The difference is quite substantial: with a trained value function, the agent is able to quickly make progress. With an untrained value function, the agent gets stuck early on.

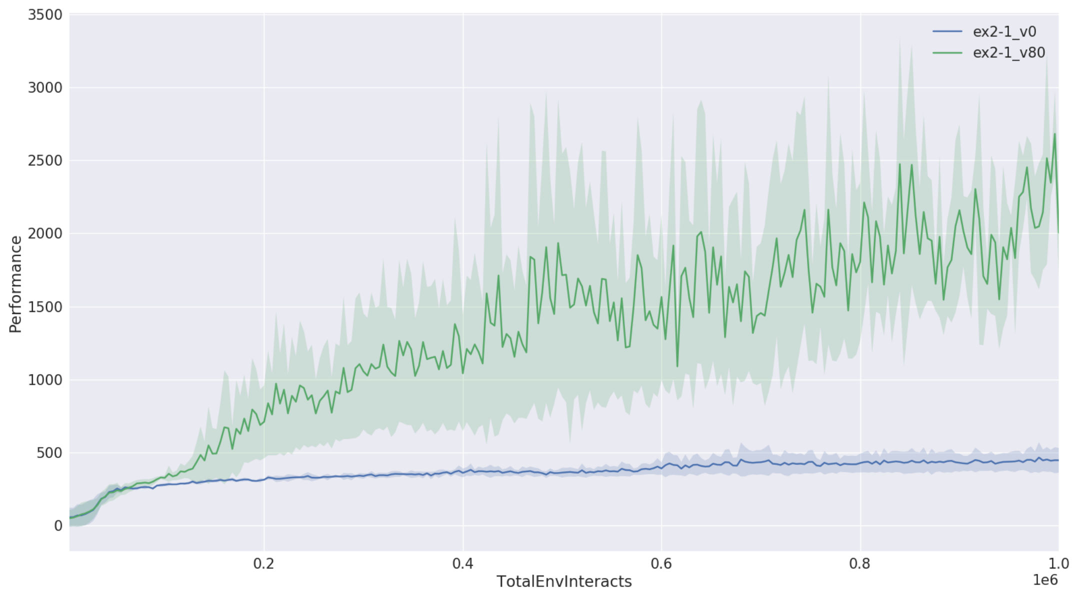
  The curve of v0 with different seeds are as follows:



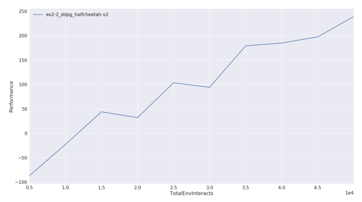
  The curve of v80 with different seeds are as follows:

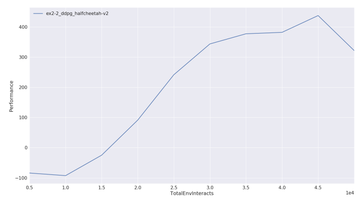


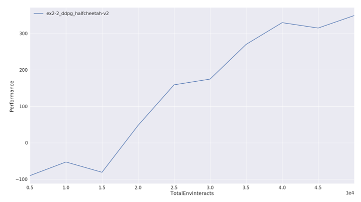
  The comparison between curve of v0 and v80, and their average and variance are as follows:



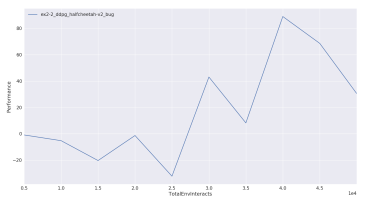
- exercise 2_2: The curve of the code without bug and with different seeds are as follows:

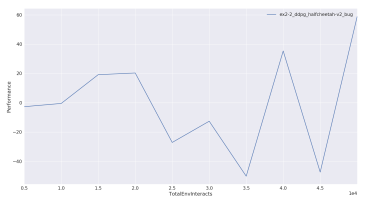


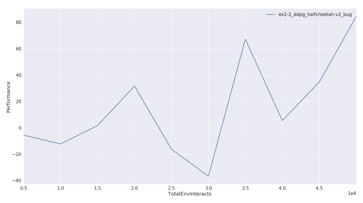




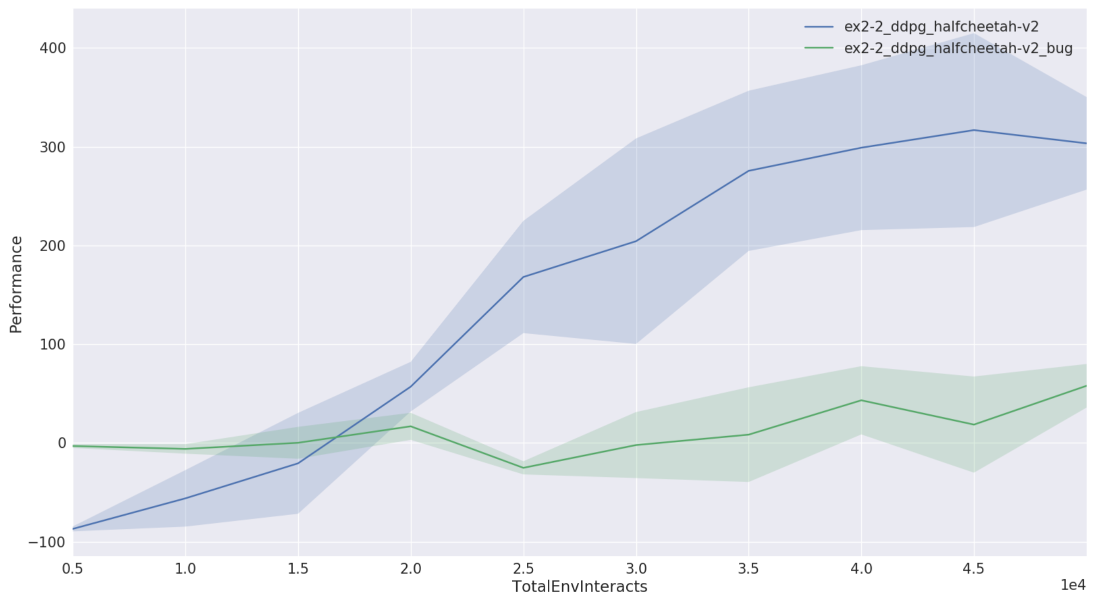
The curve of the code with bug and with different seeds are as follows:



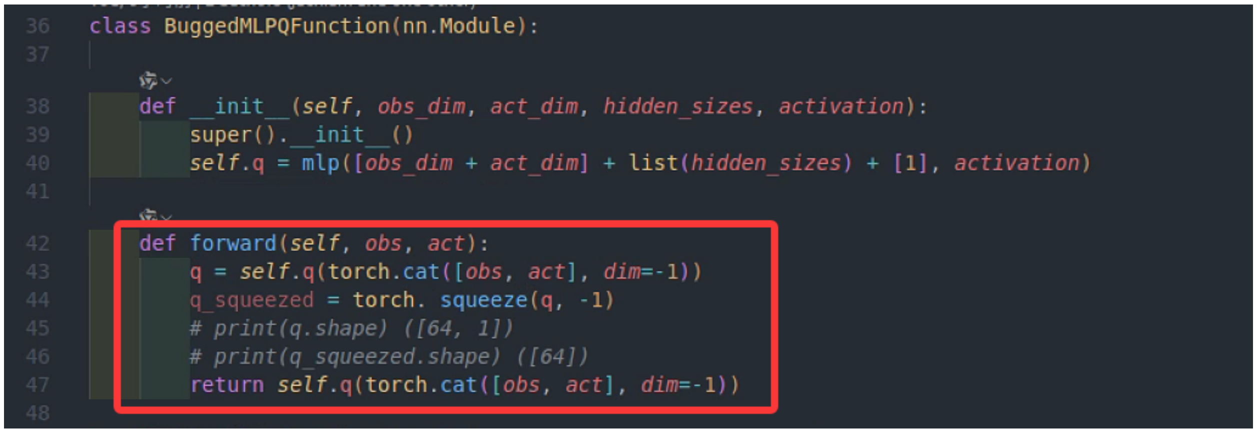




The comparison between curve with and without bug, and their average and variance are as follows:



The code's bug is shown in the following figure:



```
class BuggedMLPQFunction(nn.Module):

    def __init__(self, obs_dim, act_dim, hidden_sizes, activation):
        super().__init__()
        self.q = mlp([obs_dim + act_dim] + list(hidden_sizes) + [1], activation)

    def forward(self, obs, act):
        q = self.q(torch.cat([obs, act], dim=-1))
        q_squeezed = torch. squeeze(q, -1)
        # print(q.shape) ([64, 1])
        # print(q_squeezed.shape) ([64])
        return self.q(torch.cat([obs, act], dim=-1))
```

The error in the code is due to a missing squeeze(-1) operation, which results in the output tensor having a shape of [B, 1] instead of the expected [B]. This can cause issues in subsequent operations, as many functions (such as loss computation or tensor arithmetic) expect inputs with shape [B], and using [B, 1] may lead to shape mismatches or broadcasting errors.

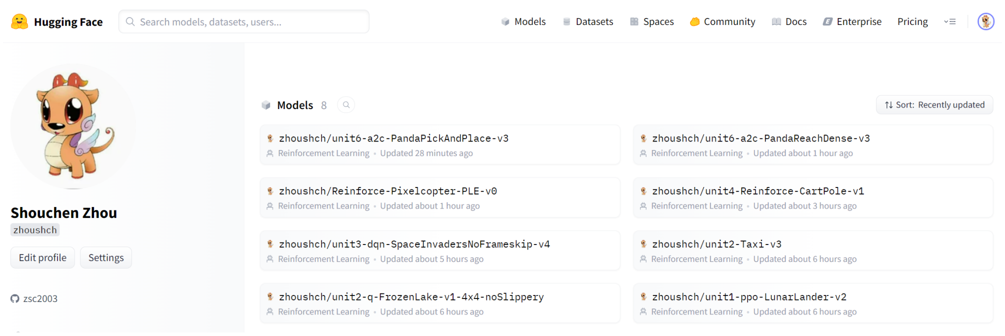
- Bonus: Are there any choices of hyperparameters which would have hidden the effects of the bug? Adjusting the hyperparameter: set batchsize into 1, in this case, the bug disappears.

Problem 2

Optional: Hugging Face Course in Deep RL. Welcome to Hugging Face Course in Deep RL! This is an educational resource produced by Hugging Face community that makes it easier to learn about deep reinforcement learning (deep RL).
(a) Finish basic units including unit 1(Introduction to Deep Reinforcement Learning), unit 2(Introduction to Q-Learning), unit 3(Deep Q-Learning with Atari Games), unit 4(Policy Gradient with Pytorch), unit 6(Actor-Critic Methods with Robotics Environments), unit 8(Proximal Policy Optimization)
(b) (Bonus)Finish all remaining units

Solution

The finished units: 1, 2, 3, 4, 6, 8's codes cuold be check in 'code/Huggingface_Deep_RL'.
Aftering running the implemented codes, several models from different units were pushed to to the Huggingface Hub, and the models are as follows:



Shouchen Zhou
zhoushch
zsc2003

Models 8
Sort: Recently updated

zhoushch/unit6-a2c-PandaPickAndPlace-v3
Reinforcement Learning · Updated 28 minutes ago

zhoushch/unit6-a2c-PandaReachDense-v3
Reinforcement Learning · Updated about 1 hour ago

zhoushch/Reinforce-Pixelcopter-PLE-v0
Reinforcement Learning · Updated about 1 hour ago

zhoushch/unit4-Reinforce-CartPole-v1
Reinforcement Learning · Updated about 3 hours ago

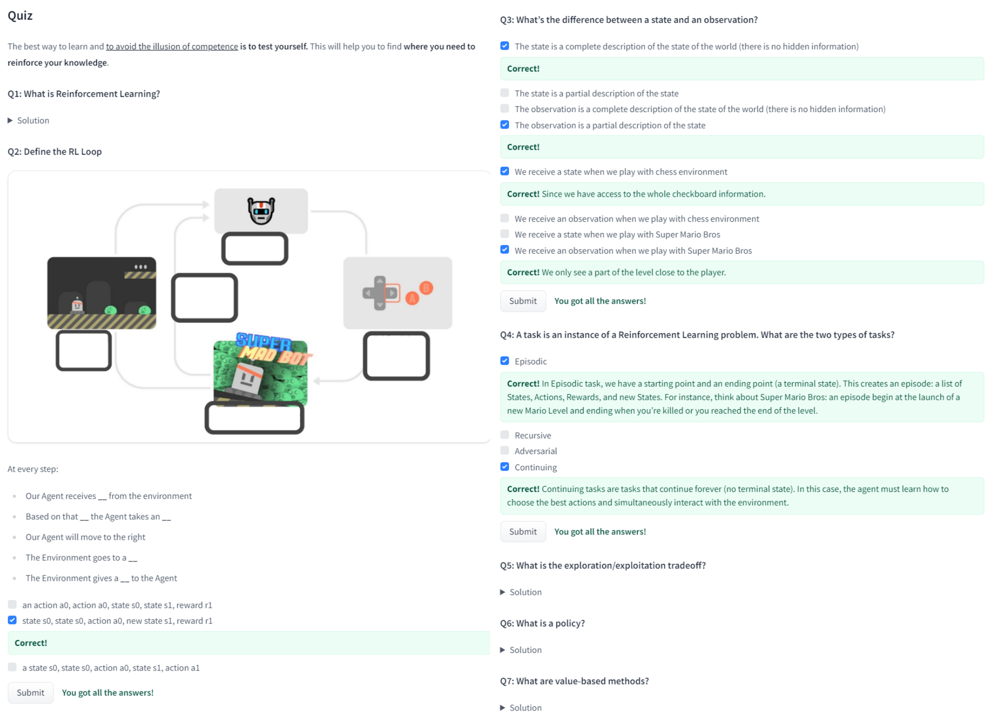
zhoushch/unit3-dqn-SpaceInvadersNoFrameskip-v4
Reinforcement Learning · Updated about 5 hours ago

zhoushch/unit2-Taxi-v3
Reinforcement Learning · Updated about 6 hours ago

zhoushch/unit2-q-FrozenLake-v1-4x4-noSlippery
Reinforcement Learning · Updated about 6 hours ago

zhoushch/unit1-ppo-LunarLander-v2
Reinforcement Learning · Updated about 6 hours ago

The quizzes of each units are also finished, which could be seen as follows:



Quiz

The best way to learn and to avoid the illusion of competence **is to test yourself.** This will help you to find **where you need to reinforce your knowledge**.

Q1: What is Reinforcement Learning?

▶ Solution

Q2: Define the RL Loop

At every step:

- Our Agent receives __ from the environment
- Based on that __ the Agent takes an __
- Our Agent will move to the right
- The Environment goes to a __
- The Environment gives a __ to the Agent

an action a0, action a0, state s0, state s1, reward r1
[✓ state s0, state s0, action a0, new state s1, reward r1]
Correct!
a state s0, state s0, action a0, state s1, action a1

You got all the answers!

Q3: What's the difference between a state and an observation?

[✓ The state is a complete description of the state of the world (there is no hidden information)]
Correct!
The state is a partial description of the state
The observation is a complete description of the state of the world (there is no hidden information)
[✓ The observation is a partial description of the state]
Correct!
[✓ We receive a state when we play with chess environment]
Correct! Since we have access to the whole checkboard information.
We receive an observation when we play with chess environment
We receive a state when we play with Super Mario Bros
[✓ We receive an observation when we play with Super Mario Bros]
Correct! We only see a part of the level close to the player.

You got all the answers!

Q4: A task is an instance of a Reinforcement Learning problem. What are the two types of tasks?

[✓ Episodic]
Correct! In Episodic task, we have a starting point and an ending point (a terminal state). This creates an episode: a list of States, Actions, Rewards, and new States. For instance, think about Super Mario Bros: an episode begin at the launch of a new Mario Level and ending when you're killed or you reached the end of the level.
Recursive
Adversarial
[✓ Continuing]
Correct! Continuing tasks are tasks that continue forever (no terminal state). In this case, the agent must learn how to choose the best actions and simultaneously interact with the environment.

You got all the answers!

Q5: What is the exploration/exploitation tradeoff?

▶ Solution

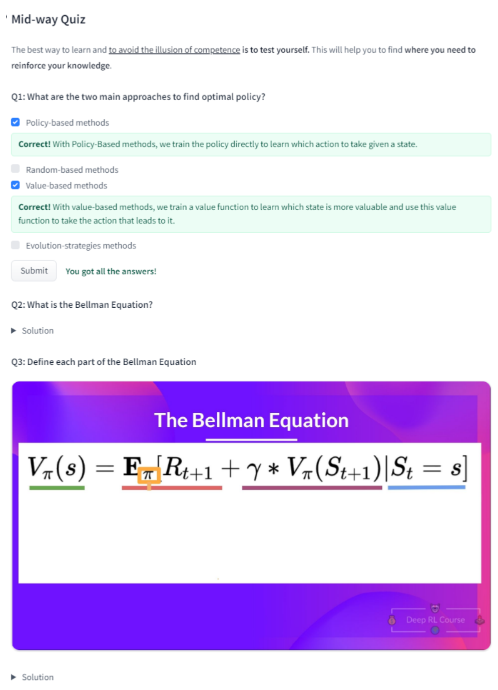
Q6: What is a policy?

▶ Solution

Q7: What are value-based methods?

▶ Solution

Problem 2 continued on next page...



Mid-way Quiz

The best way to learn and <u>to avoid the illusion of competence</u> **is to test yourself.** This will help you to find **where you need to reinforce your knowledge**.

Q1: What are the two main approaches to find optimal policy?

[✓ Policy-based methods]

Correct! With Policy-Based methods, we train the policy directly to learn which action to take given a state.

Random-based methods

[✓ Value-based methods]

Correct! With value-based methods, we train a value function to learn which state is more valuable and use this value function to take the action that leads to it.

Evolution-strategies methods

Submit You got all the answers!

Q2: What is the Bellman Equation?

▶ Solution

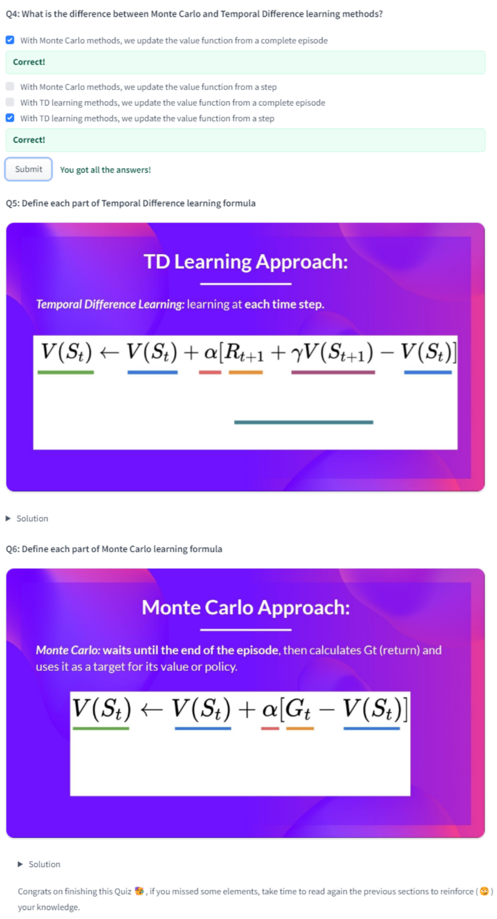
Q3: Define each part of the Bellman Equation

The Bellman Equation

$$V_\pi(s) = \mathbf{E}_\pi[R_{t+1} + \gamma * V_\pi(S_{t+1}) | S_t = s]$$

Deep RL Course

▶ Solution



Q4: What is the difference between Monte Carlo and Temporal Difference learning methods?

[✓ With Monte Carlo methods, we update the value function from a complete episode]

Correct!

With Monte Carlo methods, we update the value function from a step

With TD learning methods, we update the value function from a complete episode

[✓ With TD learning methods, we update the value function from a step]

Correct!

Submit You got all the answers!

Q5: Define each part of Temporal Difference learning formula

TD Learning Approach:

Temporal Difference Learning: learning at **each time step.**

$$V(S_t) \leftarrow V(S_t) + \alpha[R_{t+1} + \gamma V(S_{t+1}) - V(S_t)]$$

▶ Solution

Q6: Define each part of Monte Carlo learning formula

Monte Carlo Approach:

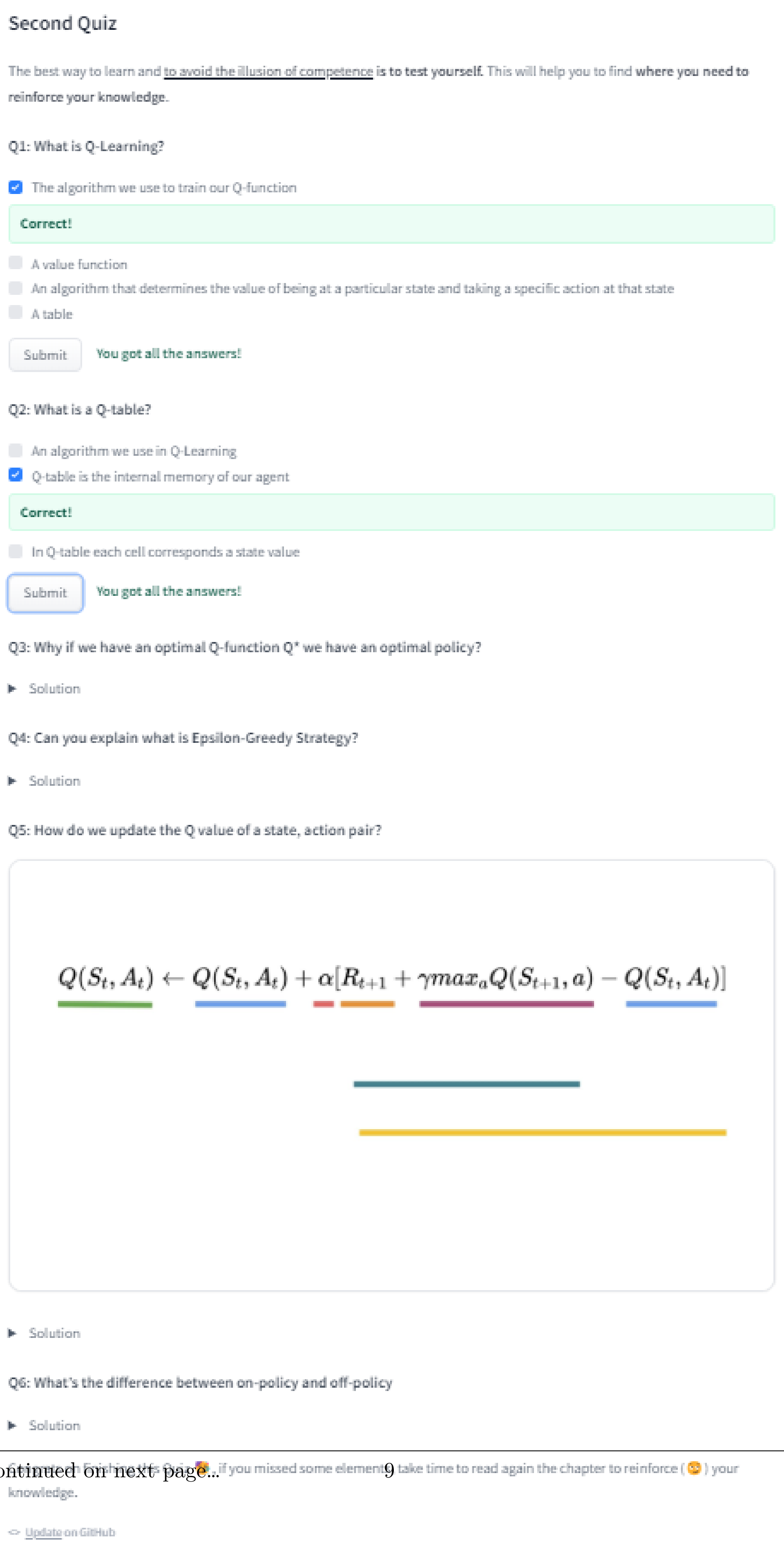
Monte Carlo: **waits until the end of the episode,** then calculates Gt (return) and uses it as a target for its value or policy.

$$V(S_t) \leftarrow V(S_t) + \alpha[G_t - V(S_t)]$$

▶ Solution

Congrats on finishing this Quiz 🥳, if you missed some elements, take time to read again the previous sections to reinforce (😏) your knowledge.

Problem 2 continued on next page...



Second Quiz

The best way to learn and to avoid the illusion of competence **is to test yourself.** This will help you to find **where you need to reinforce your knowledge.**

Q1: What is Q-Learning?

- [✓ The algorithm we use to train our Q-function]

Correct!

- A value function
- An algorithm that determines the value of being at a particular state and taking a specific action at that state
- A table

Submit You got all the answers!

Q2: What is a Q-table?

- An algorithm we use in Q-Learning
- [✓ Q-table is the internal memory of our agent]

Correct!

- In Q-table each cell corresponds a state value

Submit You got all the answers!

Q3: Why if we have an optimal Q-function Q* we have an optimal policy?

▸ Solution

Q4: Can you explain what is Epsilon-Greedy Strategy?

▸ Solution

Q5: How do we update the Q value of a state, action pair?

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma max_a Q(S_{t+1}, a) - Q(S_t, A_t)]$$

▸ Solution

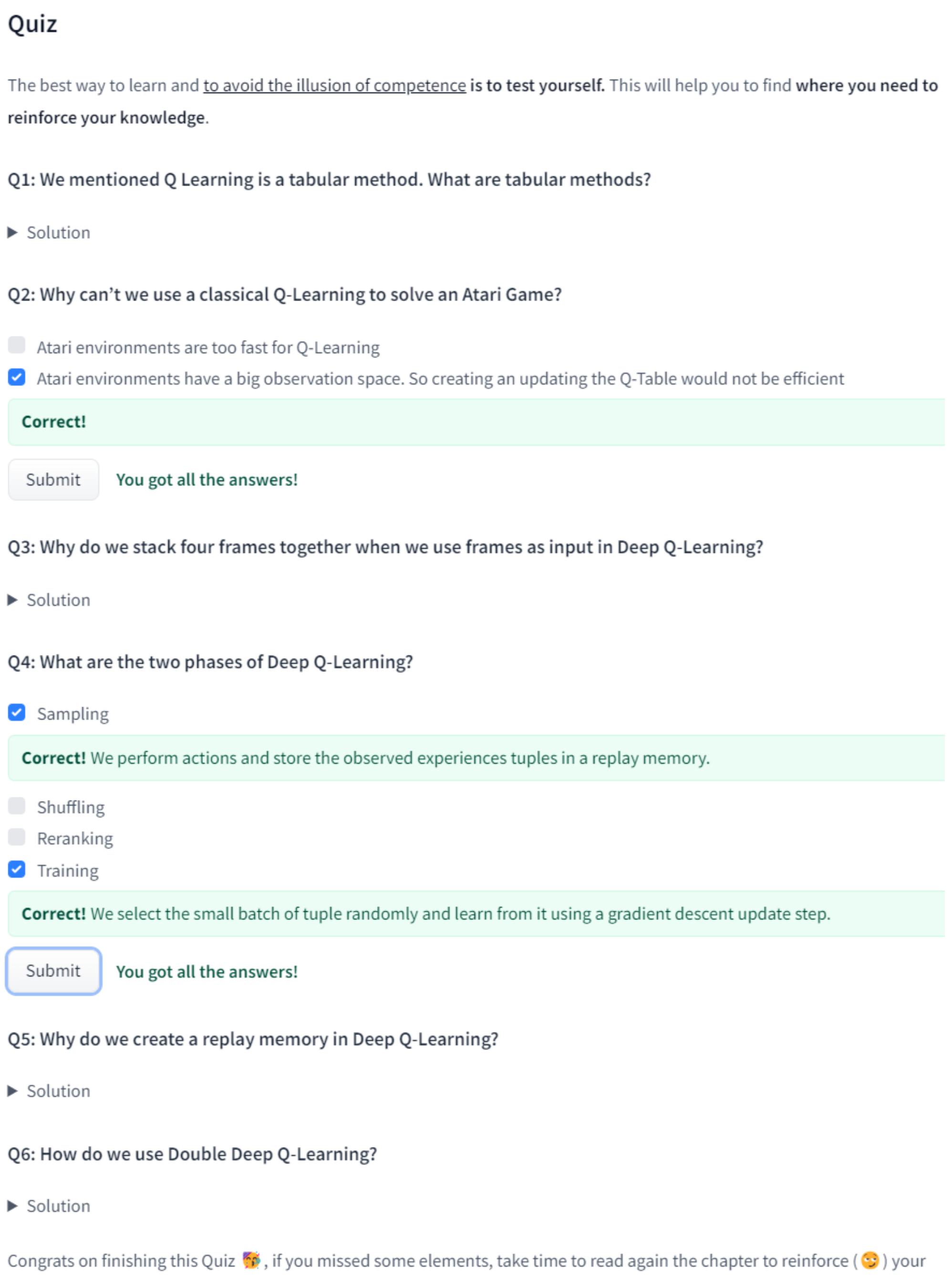
Q6: What's the difference between on-policy and off-policy

▸ Solution

Congrats on finishing this Quiz 🥳, if you missed some elements, take time to read again the chapter to reinforce (😏) your knowledge.

<> Update on GitHub

Problem 2 continued on next page...



Quiz

The best way to learn and to avoid the illusion of competence **is to test yourself.** This will help you to find **where you need to reinforce your knowledge**.

Q1: We mentioned Q Learning is a tabular method. What are tabular methods?

▶ Solution

Q2: Why can't we use a classical Q-Learning to solve an Atari Game?

☐ Atari environments are too fast for Q-Learning

☑ Atari environments have a big observation space. So creating an updating the Q-Table would not be efficient

Correct!

Submit **You got all the answers!**

Q3: Why do we stack four frames together when we use frames as input in Deep Q-Learning?

▶ Solution

Q4: What are the two phases of Deep Q-Learning?

☑ Sampling

Correct! We perform actions and store the observed experiences tuples in a replay memory.

☐ Shuffling

☐ Reranking

☑ Training

Correct! We select the small batch of tuple randomly and learn from it using a gradient descent update step.

Submit **You got all the answers!**

Q5: Why do we create a replay memory in Deep Q-Learning?

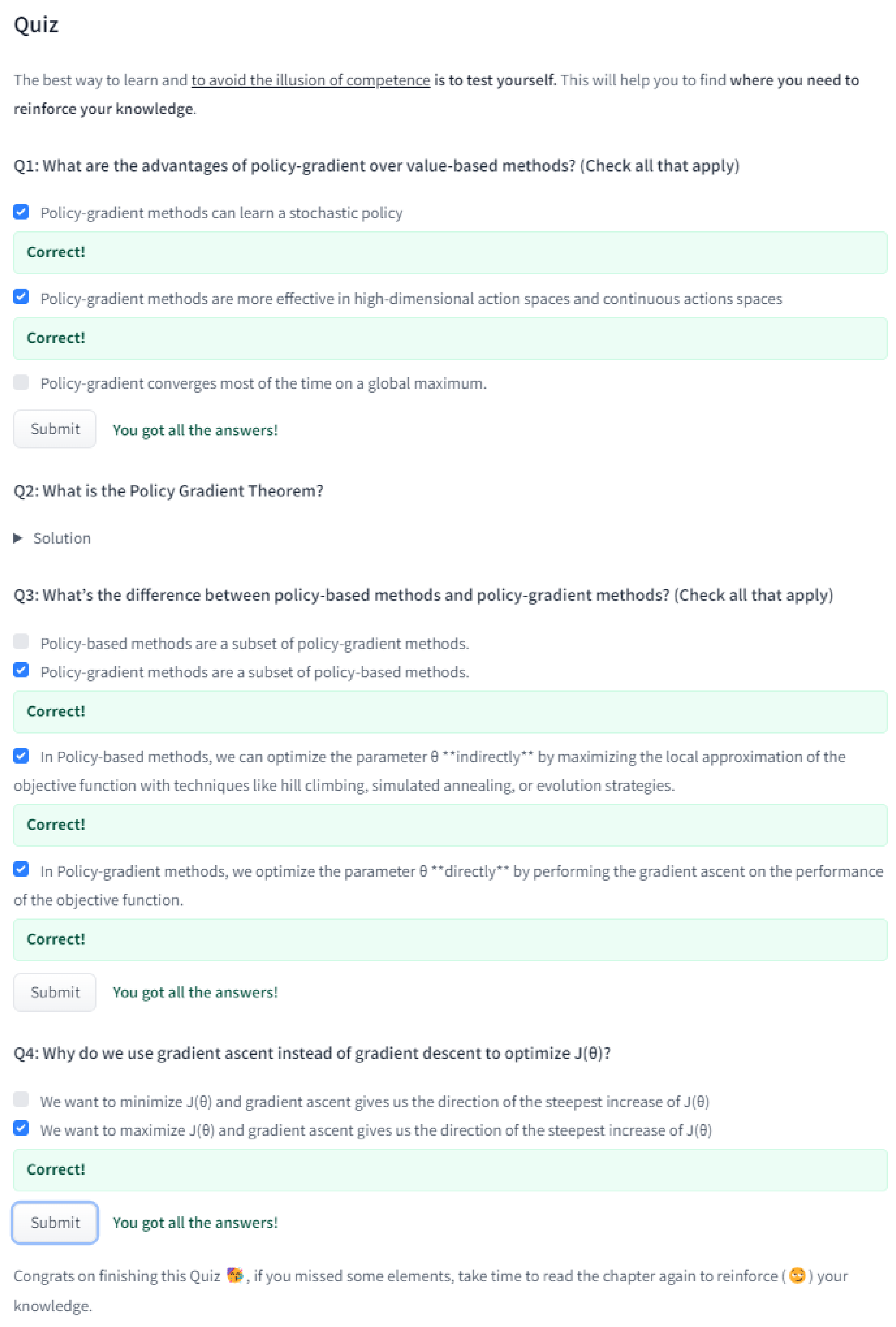
▶ Solution

Q6: How do we use Double Deep Q-Learning?

▶ Solution

Congrats on finishing this Quiz 🥳, if you missed some elements, take time to read again the chapter to reinforce (😏) your

Problem 2 continued on next page...



Quiz

The best way to learn and to avoid the illusion of competence **is to test yourself.** This will help you to find **where you need to reinforce your knowledge**.

Q1: What are the advantages of policy-gradient over value-based methods? (Check all that apply)

[✓ Policy-gradient methods can learn a stochastic policy]

Correct!

[✓ Policy-gradient methods are more effective in high-dimensional action spaces and continuous actions spaces]

Correct!

Policy-gradient converges most of the time on a global maximum.

Submit You got all the answers!

Q2: What is the Policy Gradient Theorem?

▶ Solution

Q3: What's the difference between policy-based methods and policy-gradient methods? (Check all that apply)

Policy-based methods are a subset of policy-gradient methods.

[✓ Policy-gradient methods are a subset of policy-based methods.]

Correct!

[✓ In Policy-based methods, we can optimize the parameter θ **indirectly** by maximizing the local approximation of the objective function with techniques like hill climbing, simulated annealing, or evolution strategies.]

Correct!

[✓ In Policy-gradient methods, we optimize the parameter θ **directly** by performing the gradient ascent on the performance of the objective function.]

Correct!

Submit You got all the answers!

Q4: Why do we use gradient ascent instead of gradient descent to optimize J(θ)?

We want to minimize J(θ) and gradient ascent gives us the direction of the steepest increase of J(θ)

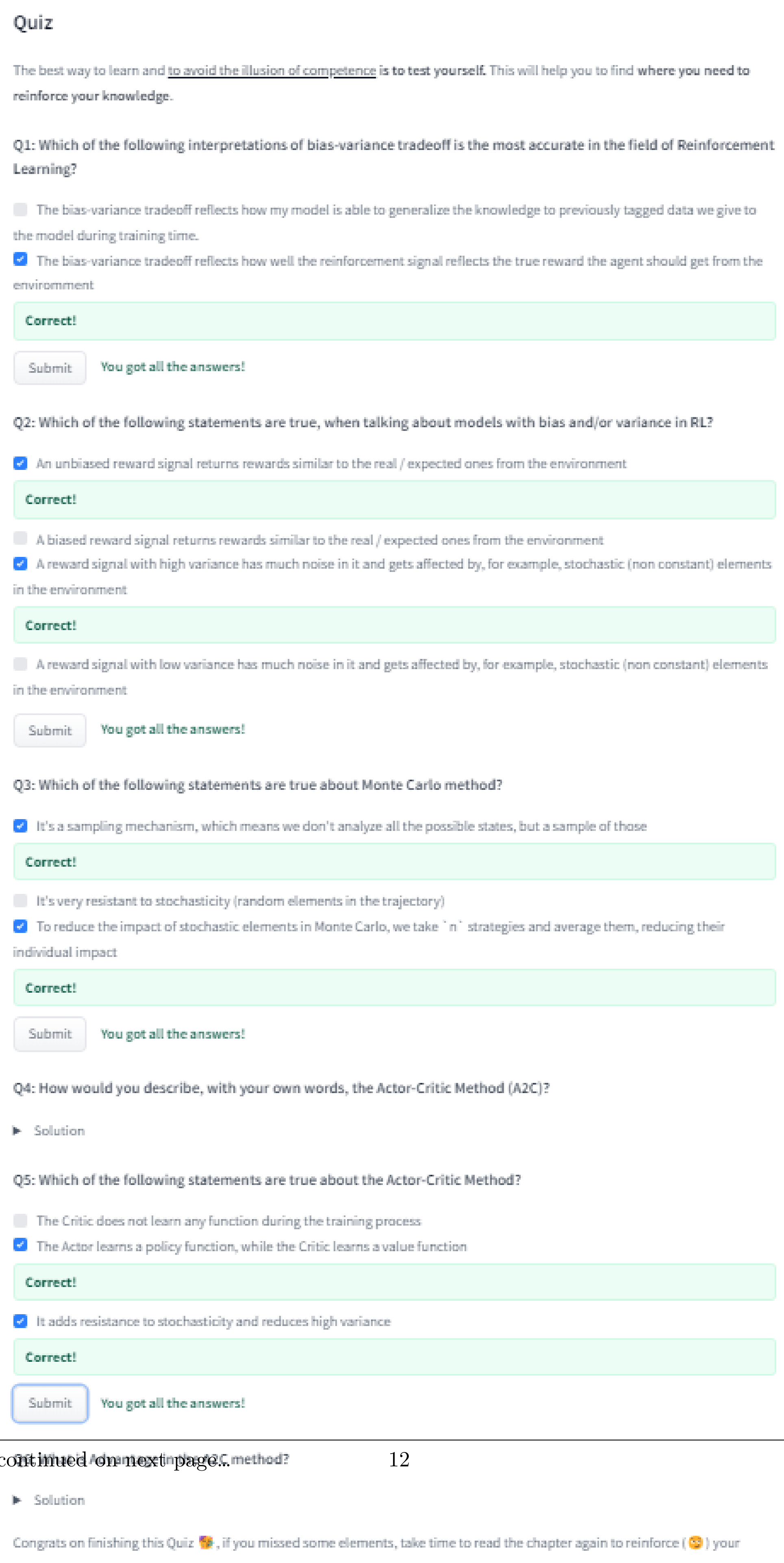
[✓ We want to maximize J(θ) and gradient ascent gives us the direction of the steepest increase of J(θ)]

Correct!

Submit You got all the answers!

Congrats on finishing this Quiz 🥳, if you missed some elements, take time to read the chapter again to reinforce (😋) your knowledge.

Problem 2 continued on next page...



Quiz

The best way to learn and to avoid the illusion of competence **is to test yourself.** This will help you to find **where you need to reinforce your knowledge**.

Q1: Which of the following interpretations of bias-variance tradeoff is the most accurate in the field of Reinforcement Learning?

[ ] The bias-variance tradeoff reflects how my model is able to generalize the knowledge to previously tagged data we give to the model during training time.

[✓] The bias-variance tradeoff reflects how well the reinforcement signal reflects the true reward the agent should get from the enviromment

Correct!

Submit You got all the answers!

Q2: Which of the following statements are true, when talking about models with bias and/or variance in RL?

[✓] An unbiased reward signal returns rewards similar to the real / expected ones from the environment

Correct!

[ ] A biased reward signal returns rewards similar to the real / expected ones from the environment

[✓] A reward signal with high variance has much noise in it and gets affected by, for example, stochastic (non constant) elements in the environment

Correct!

[ ] A reward signal with low variance has much noise in it and gets affected by, for example, stochastic (non constant) elements in the environment

Submit You got all the answers!

Q3: Which of the following statements are true about Monte Carlo method?

[✓] It's a sampling mechanism, which means we don't analyze all the possible states, but a sample of those

Correct!

[ ] It's very resistant to stochasticity (random elements in the trajectory)

[✓] To reduce the impact of stochastic elements in Monte Carlo, we take `n` strategies and average them, reducing their individual impact

Correct!

Submit You got all the answers!

Q4: How would you describe, with your own words, the Actor-Critic Method (A2C)?

▶ Solution

Q5: Which of the following statements are true about the Actor-Critic Method?

[ ] The Critic does not learn any function during the training process

[✓] The Actor learns a policy function, while the Critic learns a value function

Correct!

[✓] It adds resistance to stochasticity and reduces high variance

Correct!

Submit You got all the answers!

Problem 2 continued on next page...

Q6: What is Advantage in the A2C method?

▶ Solution

Congrats on finishing this Quiz 🥳, if you missed some elements, take time to read the chapter again to reinforce (😏) your