**COMP3702 A3**

**Question 1. Q-learning vs Value Iteration**

Q-learning is closely related to the Value Iteration algorithm for Markov decision processes.

**a) Describe two key similarities between Q-learning and Value Iteration. Answer in no more than 5 lines of text. (5 marks)**

Both methods use the Bellman equation to iteratively update value estimates—Value Iteration updates state values, while Q-learning updates state-action values.

Both algorithms aim to discover the optimal policy that maximizes the cumulative reward in a Markov decision process (MDP).

**b) Give one key difference between Q-learning and Value Iteration. Answer in no more than 5 lines of text. (5 marks)**

Value Iteration is model-based, requiring knowledge of the environment’s transition probabilities and rewards, whereas Q-learning is model-free, learning the optimal policy directly from experience without prior knowledge of the environment's dynamics.

**Value Iteration Bellman Update**: V(s)=max​s′∑​P(s′∣s,a)[R(s,a,s′)+γV(s′)] where V(s)) is the value of state s, P(s′∣s,a) is the transition probability, R(s,a,s′) is the reward, and γ is the discount factor.

**Q-learning Update Rule**: Q(s,a)←Q(s,a)+α[r+γ′max​Q(s′,a′)−Q(s,a)] where Q(s,a) is the state-action value, rrr is the immediate reward, and α\alphaα is the learning rate.

a) Implement a function to plot the R100 value vs Episode number. You will need to import a plotting library, e.g. import matplotlib.pyplot as plt. Copy or screenshot your code implementation for your answer. As part of this, you may also want to implement saving and loading of plots. (5 marks)

b) Plot the R100 value vs Episode number for CartPole-v0 and CartPole-v1 DQN models. Ensure your axes are correctly labelled and indicate what each plot represents (e.g., using a legend or caption). (5 marks)

CartPole-v0 single-hidden

A graph of a number

Description automatically generated

CartPole-v1 single-hidden

A graph of a number

Description automatically generated

c) Describe and compare the learnt policies for CartPole-v0 and CartPole-v1.

You may make use of the saved video examples on Blackboard titled “CartPole-v0.mp4” and “CartPole-v1.mp4” (v0 keeps moving to the left or right).

Based on your observation of these learnt policies, the definition of the environment and your plots, explain why you think the values of max\_episode\_steps and reward\_threshold were increased from v0 to v1. (5 marks)

The main difference between the learnt policies for **CartPole-v0** and **CartPole-v1** is evident in the plots you provided and can be explained by the environment settings.

**Exploration vs. Exploitation**:

In the early stages of training, the agent is primarily exploring the environment, meaning that it is taking random actions to discover which actions lead to higher rewards. During this exploration phase, the agent may perform poorly for several episodes, which causes a temporary dip in the R100 values. As the training progresses, the agent gradually transitions from exploration to exploitation (choosing actions based on learned policies). However, this transition isn't always smooth, and exploration can sometimes lead to actions that reduce the short-term reward.

In **CartPole-v0**, the environment terminates episodes earlier due to its smaller max\_episode\_steps (200 steps). This can be observed in the slower progression in reward improvement in your plot for CartPole-v0, with a smoother increase in R100 (running average of 100 episodes) over a longer time (over 1000 episodes). The agent is learning to balance the pole but has limited time to improve performance before the episode terminates.

In **CartPole-v1**, with an increased max\_episode\_steps (500 steps), the agent has more time in each episode to maximize rewards. As shown in your plot for CartPole-v1, the R100 curve shows a sharp increase within fewer episodes (~300-400), indicating that the agent learned to balance the pole more effectively within fewer episodes compared to v0.

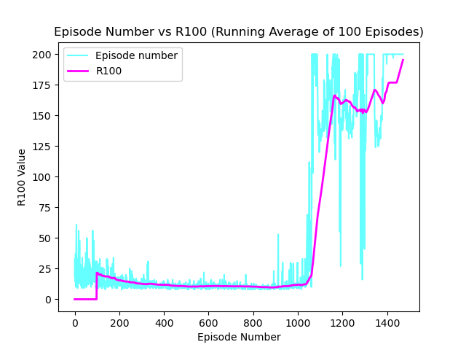
**CartPole-v0** has a lower reward\_threshold and max\_episode\_steps. The environment stops after 200 steps, limiting how long the agent can keep the pole balanced. As a result, the policy learned tends to make the cart move more to the left or right to keep the pole upright but might fail after the 200-step limit.

**CartPole-v1** increases both the max\_episode\_steps to 500 and the reward\_threshold, allowing for longer episodes. This gives the agent more time to perfect its balancing strategy, allowing it to maintain the pole in an upright position for a longer time. This helps explain why the agent reaches a higher cumulative reward and learns faster in v1.

**Max Episode Steps:** In v0, the 200-step limit may have been too short for the agent to learn optimal long-term balancing behavior. By increasing it to 500 in v1, the agent can learn to stabilize the pole for a more extended period and achieve better performance.

**Reward Threshold:** Since the agent can maintain balance for longer in v1, the reward threshold must be increased to reflect the more difficult task. The higher reward threshold in v1 ensures the agent is evaluated based on long-term stability rather than short-term, which is why it needs to achieve a score closer to 500.

CartPole-v0 two-hidden



CartPole-v1 -n two-hidden

A graph with lines and numbers

Description automatically generated

CartPole-v0 -n duelling-dqn

A graph of a number

Description automatically generated

CartPole-v1 -n duelling-dqn

