INF8953DE Final Exam

Emile Dimas

TOTAL POINTS

60 / 65

QUESTION 1

Comparisons 10 pts

1.1 1.a **2 / 2**

✓ **- 0 pts** Correct

1.2 1.b **2 / 2**

✓ **- 0 pts** Correct

1.3 1.c **2 / 2**

✓ **- 0 pts** Correct

1.4 1.d **0 / 2**

✓ **- 2 pts** Incorrect Answer

1.5 1.e **2 / 2**

✓ **- 0 pts** Correct

QUESTION 2

Recommendation Systems 12 pts

2.1 2.a **4 / 4**

✓ **- 0 pts** Correct

2.2 2.b **2 / 2**

✓ **- 0 pts** Correct

2.3 2.c **2 / 2**

✓ **- 0 pts** Correct

2.4 2.d **2 / 2**

✓ **- 0 pts** Correct

2.5 2.e **2 / 2**

✓ **- 0 pts** Correct

QUESTION 3

Deterministic policy and MDP 7 pts

3.1 3.a **2 / 2**

✓ **- 0 pts** Correct

3.2 3.b **2 / 2**

✓ **- 0 pts** Correct

3.3 3.c **3 / 3**

✓ **- 0 pts** Correct

QUESTION 4

Monte-Carlo Methods 6 pts

4.1 4.a **3 / 3**

✓ **- 0 pts** Correct

4.2 4.b **3 / 3**

✓ **- 0 pts** Correct

QUESTION 5

5 Monte-carlo vs TD **5 / 5**

✓ **- 0 pts** Correct

QUESTION 6

Q-learning 14 pts

6.1 6.a **6 / 6**

✓ **- 0 pts** Correct

6.2 6.b **4 / 4**

✓ **- 0 pts** All Correct

6.3 6.c **2 / 4**

✓ **- 2 pts** Partial marks

QUESTION 7

7 Learning with options **4 / 5**

✓ **- 1 pts See comments.**

You need to discuss more the nature of SMDP learning and its behavior under the given option setting.

QUESTION 8

8 Non-markovian Options **6 / 6**

✓ **- 0 pts Correct**

QUESTION 9

9 Honor Code **0 / 0**

✓ **- 0 pts Correct**

Honour Code

By submitting this exam, I affirm that the Honour Code is in effect and the solutions submitted are my own. I neither consulted anyone else during the period of exam nor did I give away the solutions to other students taking the exam.

Question 1 ~20 min

a) On-policy vs off-policy

- in on-policy we only use one policy: the one we want to optimize. we also use this policy to explore (SARSA)
- in off-policy we use 2 policies: - behaviour policy which is used to interact with the world and explore
  (Q-learning)
  - target policy which we want to learn (which goal is to optimize the returns)

b) Model-free vs model based

- in model free we do not use or learn a model of the world the agent learns its policy by interacting with the environment: (SARSA)
- in model based we learn a model or this model is given to us and we use the model to do planning (Dynamic Programming)

c) Online RL vs Offline RL

- in online RL the agent interacts with the environment, while in offline the agent learns from a dataset of collected trajectories.
  Online RL example (SARSA)
  offline RL example (Dagger)

d) off-policy vs offline

as mentioned above off-policy is an online method as the agent interacts with the world. However the agent uses 2 different policies one to explore, the second being the one to optimize (ex: off policy MC with WIS)
On the contrary offline RL does not interact with the world (or at least most of the time); Like the off-policy, we can consider that there is more than one policy: the one to optimize and the ones used to collect the dataset of trajectories (ex: policy penalty methods)

1.1 1.a **2 / 2**

✓ **- 0 pts** Correct

Honour Code

By submitting this exam, I affirm that the Honour Code is in effect and the solutions submitted are my own. I neither consulted anyone else during the period of exam nor did I give away the solutions to other students taking the exam.

Question 1 ~20 min

a) On-policy vs off-policy

- in on-policy we only use one policy: the one we want to optimize. we also use this policy to explore (SARSA)
- in off-policy we use 2 policies: - behaviour policy which is used to interact with the world and explore
(Q-learning)
  - target policy which we want to learn (which goal is to optimize the returns)

b) Model-free vs model based

- in model free we do not use or learn a model of the world the agent learns its policy by interacting with the environment: (SARSA)
- in model based we learn a model or this model is given to us and we use the model to do planning (Dynamic Programming)

c) Online RL vs Offline RL

- in online RL the agent interacts with the environment, while in offline the agent learns from a dataset of collected trajectories.

Online RL example (SARSA)
Offline RL example (Dagger)

d) Off-policy vs Offline

as mentioned above off-policy is an online method as the agent interacts with the world. However the agent uses 2 different policies one to explore, the second being the one to optimize (ex: off policy MC with WIS)
On the contrary offline RL does not interact with the world (or at least most of the time); Like the off-policy, we can consider that there is more than one policy. the one to optimize and the ones used to collect the dataset of trajectories (ex: policy penalty methods)

1.2 1.b **2 / 2**

✓ **- 0 pts** Correct

Honour Code

By submitting this exam, I affirm that the Honour Code is in effect and the solutions submitted are my own. I neither consulted anyone else during the period of exam nor did I give away the solutions to other students taking the exam.

Question 1 ~20 min

a) On-policy vs off-policy

- in on-policy we only use one policy: the one we want to optimize. we also use this policy to explore (SARSA)
- in off-policy we use 2 policies: - behaviour policy which is used to interact with the world and explore
  (Q-learning)
  - target policy which we want to learn (which goal is to optimize the returns)

b) Model-free vs model based

- in model free we do not use or learn a model of the world the agent learns its policy by interacting with the environment: (SARSA)
- in model based we learn a model or this model is given to us and we use the model to do planning (Dynamic Programming)

c) Online RL vs Offline RL

- in online RL the agent interacts with the environment, while in offline the agent learns from a dataset of collected trajectories.
  Online RL example (SARSA)
  Offline RL example (Dagger)

d) Off-policy vs Offline

as mentioned above off-policy is an online method as the agent interacts with the world. However the agent uses 2 different policies one to explore, the second being the one to optimize (ex: off policy MC with WIS)

On the contrary offline RL does not interact with the world (or at least most of the time); Like the off-policy, we can consider that there is more than one policy: the one to optimize and the ones used to collect the dataset of trajectories (ex: policy penalty methods)

1.3 1.c 2 / 2

✓ **- 0 pts** Correct

Honour Code

By submitting this exam, I affirm that the Honour Code is in effect and the solutions submitted are my own. I neither consulted anyone else during the period of exam nor did I give away the solutions to other students taking the exam.

Question 1 ~20 min

a) On-policy vs off-policy

- in on-policy we only use one policy: the one we want to optimize. we also use this policy to explore (SARSA)
- in off-policy we use 2 policies: - behaviour policy which is used to interact with the world and explore
  (Q-learning)
  - target policy which we want to learn (which goal is to optimize the returns)

b) Model-free vs model based

- in model free we do not use or learn a model of the world the agent learns its policy by interacting with the environment: (SARSA)
- in model based we learn a model or this model is given to us and we use the model to do planning (Dynamic Programming)

c) Online RL vs Offline RL

- in online RL the agent interacts with the environment, while in offline the agent learns from a dataset of collected trajectories.

  Online RL example (SARSA)
  offline RL example (Dagger)

d) off-policy vs offline

as mentioned above off-policy is an online method as the agent interacts with the world. However the agent uses 2 different policies one to explore, the second being the one to optimize (ex: off policy MC with WIS)

On the contrary offline RL does not interact with the world (or at least most of the time); Like the off-policy, we can consider that there is more than one policy: the one to optimize and the ones used to collect the dataset of trajectories (ex: policy penalty methods)

1.4 1.d 0 / 2

✓ - 2 pts Incorrect Answer

Monte Carlo vs TD

- Monte carlo methods estimate the value of states by computing the discounted sum of returns (and averaging over many episodes) each estimate is independent of the other.
- TD on the other hand bootstrap its estimate which means it uses the estimate of the next state to compute the estimate of the current state.

TD is faster than MC and usually converges to better policies.

TD has lower variance than MC, but MC has lower bias

Question 2 ~30 min

a) states : would be the users

actions: would be the movies recommended

transitions: it depends here what are our assumptions. The problem can be modeled as a contextual bandit problem. If that's the case there is no effect on the user depending on the action that he/she takes now. However this is a very simplistic assumption. In fact the contextual bandit doesn't take into account long term dependencies. (S, A, R)

rewards: Rating of the movie by the user (if we assume he watches the movie) or his satisfaction measured by click through rate

b) I would assume a contextual bandit algorithm would be enough for this problem but we have not seen in class any specific contextual bandit problem. I would therefore choose a DQN for several reasons. we use FA which will be useful. it's one of the most advanced algorithm we have seen. it uses TD update which is good for continuous setting (our case)

c) Since I use DQN. I only need to use the right features to be able to seamlessly add new customers. (would have been the same if I used contextual bandit)

d) new movies correspond to new actions, so that I would need to add more actions (which is not optimal) FA does the job

if it was a contextual bandit algorithm we would have added a new "arm" (action) and trained again)

1.5 1.e 2 / 2

✓ **- 0 pts** Correct

Monte Carlo vs TD

- Monte carlo methods estimate the value of states by computing the discounted sum of returns (and averaging over many episodes) each estimate is independent of the other.
- TD on the other hand bootstrap its estimate which means it uses the estimate of the next state to compute the estimate of the current state.

TD is faster than MC and usually converges to better policies.

TD has lower variance than MC, but MC has lower bias

Question 2 ~30 min

a) states: would be the users

actions: would be the movies recommended

transitions: it depends here what are our assumptions. The problem can be modeled as a contextual bandit problem. If that's the case there is no effect on the user depending on the action that he/she takes now. However this is a very simplistic assumption. In fact the contextual bandit doesn't take into account long term dependencies. (S, A, R)

rewards: Rating of the movie by the user (if we assume he watches the movie) or his satisfaction measured by click through rate

b) I would assume a contextual bandit algorithm would be enough for this problem but we have not seen in class any specific contextual bandit problem. I would therefore choose a DQN for several reasons. we use FA which will be useful. it's one of the most advanced algorithm we have seen. it uses TD update which is good for continuous setting (our case)

c) Since I use DQN. I only need to use the right features to be able to seamlessly add new customers. (would have been the same if I used Contextual Bandit)

d) new movies correspond to new actions, so that I would ~~need~~ FA does the job to add more actions (which is not optimal)
if it was a contextual bandit algorithm we would have added a new "arm" (action) and trained again)

2.1 2.a 4 / 4

✓ - 0 pts Correct

Monte Carlo vs TD

- Monte carlo methods estimate the value of states by computing the discounted sum of returns (and averaging over many episodes) each estimate is independent of the other.
- TD on the other hand bootstrap its estimate which means it uses the estimate of the next state to compute the estimate of the current state.

TD is faster than MC and usually converges to better policies.

TD has lower variance than MC, but MC has lower bias

Question 2 ~30 min

a) states: would be the users

actions: would be the movies recommended

transitions: it depends here what are our assumptions. The problem can be modeled as a contextual bandit problem. If that's the case there is no effect on the user depending on the action that he/she takes now. However this is a very simplistic assumption. In fact the contextual bandit doesn't take into account long term dependencies. (S, A, R)

rewards: Rating of the movie by the user (if we assume he watches the movie) or his satisfaction measured by Click through rate

b) I would assume a contextual bandit algorithm would be enough for this problem but we have not seen in class any specific contextual bandit problem. I would therefore choose a DQN for several reasons. we use FA which will be useful. it's one of the most advanced algorithm we have seen. it uses TD update which is good for continuous setting (our case)

c) Since I use DQN. I only need to use the right features to be able to seamlessly add new customers. (would have been the same if I used Contextual Bandit)

d) new movies correspond to new actions, so that I would ~~need~~ FA does the job to add more actions (which is not optimal) if it was a contextual bandit algorithm we would have added a new "arm" (action) and trained again)

2.2 2.b **2 / 2**

✓ **- 0 pts** Correct

Monte Carlo vs TD

- Monte carlo methods estimate the value of states by computing the discounted sum of returns (and averaging over many episodes) each estimate is independent of the other.
- TD on the other hand bootstrap its estimate which means it uses the estimate of the next state to compute the estimate of the current state.

TD is faster than MC and usually converges to better policies.

TD has lower variance than MC, but MC has lower bias

Question 2 ~30 min

a) states: would be the users

actions: would be the movies recommended

transitions: it depends here what are our assumptions. The problem can be modeled as a contextual bandit problem. If that's the case there is no effect on the user depending on the action that he/she takes now. However this is a very simplistic assumption. In fact the contextual bandit doesn't take into account long term dependencies. (S, A, R)

rewards: Rating of the movie by the user (if we assume he watches the movie) or his satisfaction measured by click through rate

b) I would assume a contextual bandit algorithm would be enough for this problem but we have not seen in class any specific contextual bandit problem. I would therefore choose a DQN for several reasons. we use FA which will be useful. it's one of the most advanced algorithm we have seen. it uses TD update which is good for continuous setting (our case)

c) Since I use DQN. I only need to use the right features to be able to seamlessly add new customers. (would have been the same if I used Contextual Bandit)

d) new movies correspond to new actions, so that I would ~~need~~ FA does the job to add more actions (which is not optimal) if it was a contextual bandit algorithm we would have added a new "arm" (action) and trained again)

2.3 2.c **2 / 2**

✓ **- 0 pts** Correct

Monte Carlo vs TD

- Monte carlo methods estimate the value of states by computing the discounted sum of returns (and averaging over many episodes) each estimate is independent of the other.
- TD on the other hand bootstrap its estimate which means it uses the estimate of the next state to compute the estimate of the current state.

TD is faster than MC and usually converges to better policies.

TD has lower variance than MC, but MC has lower bias

Question 2 ~30 min

a) states: would be the users

actions: would be the movies recommended

transitions: it depends here what are our assumptions. The problem can be modeled as a contextual bandit problem. If that's the case there is no effect on the user depending on the action that he/she takes now. However this is a very simplistic assumption. In fact the contextual bandit doesn't take into account long term dependencies. (S, A, R)

rewards: Rating of the movie by the user (if we assume he watches the movie) or his satisfaction measured by click through rate

b) I would assume a contextual bandit algorithm would be enough for this problem but we have not seen in class any specific contextual bandit problem. I would therefore choose a DQN for several reasons. we use FA which will be useful. it's one of the most advanced algorithm we have seen. it uses TD update which is good for continuous setting (our case)

c) Since I use DQN. I only need to use the right features to be able to seamlessly add new customers. (would have been the same if I used Contextual Bandit)

d) new movies correspond to new actions, so that I would need to add more actions (which is not optimal) FA does the job

if it was a contextual bandit algorithm we would have added a new "arm" (action) and trained again)

2.4 2.d 2 / 2

✓ - 0 pts Correct

e) One thing that I can think off is control exploration by adding a layer of unsupervised learning. For example, we can create a variable that measure the favorite genre based on activity history of the user. we could also use RNNs. Then we could add a bonus to the same type of movies.
(priority Experience replay for example with prioritization based on previous history)

Question 3 (~10 min)

a) $v_\pi^*(s) = \max_a \sum_{s',r} p(s',r|s,a)\left[r + \gamma v_{\pi^*}(s')\right]$

$q_\pi^*(s,a) = \sum_{s',r} p(s',r|s,a)\left[r + \gamma \max_{a'} q_\pi^*(a',s')\right]$

nothing changes

b) $v_\pi^*(s) = \max_a \left[r + \gamma v_\pi(s')\right]$

$q^*(s,a) = \left[r + \gamma \max_{a'} q_\pi^*(a',s')\right]$

c) we just need to replace the 2 update rules in the algorithm

- $V(s) \leftarrow \max_a \left[r + \gamma v_\pi(s')\right]$ instead of $V(s) \leftarrow \max_a \sum_{s',r} p(s',r|s,a)\left(r+\gamma v_\pi(s)\right)$
- $\pi(s) = \operatorname{argmax}_a \left[r + \gamma v_\pi(s')\right]$ instead of $\pi(s) = \operatorname{argmax}_a \sum_{s',r} p(s',r|s,a)\left[r+\gamma V(s')\right]$

question 4 (~10min)

$S_1, A_1, 13, S_1, A_2, 7, S_1, A_1, 13, S_1, A_2, 14$ done

a) First visit MC

$q_\pi(S_1,A_1)$ = return from first seeing S_1, A_1 in the sequence

rewards $= [13, 7, 13, 14]$ $\quad G_t = 13 + \frac{7}{2} + \frac{13}{4} + \frac{14}{8} = 21.5$

$\gamma^k = \left[(\frac{1}{2})^0 ; \frac{1}{2}, (\frac{1}{2})^2, (\frac{1}{2})^3\right]$ $\quad \Rightarrow \boxed{q_\pi(S_1,A_1) = 21.5}$

$q_\pi(S_1,A_2) \Rightarrow$ rewards $= [7, 13, 14]$

$\gamma^k = (1, \frac{1}{2}, \frac{1}{4})$ $\quad \Rightarrow G_t = 7 + \frac{13}{2} + \frac{14}{4} = 17$

$\boxed{q_\pi(S_1,A_2) = 17}$

2.5 2.e **2 / 2**

✓ **- 0 pts** Correct

e) One thing that I can think off is control exploration by adding a layer of unsupervised learning. For example, we can create a variable that measure the favorite genre based on activity history of the user. we could also use RNNs. Then we could add a bonus to the same type of movies.
(priority Experience replay for example with prioritization based on previous history)

Question 3 (~10 min)

a) $v_\pi^*(s) = \max_a \sum_{s',r} p(s',r|s,a)\left[r + \gamma v_{\pi^*}(s')\right]$

$q_\pi^*(s,a) = \sum_{s',r} p(s',r|s,a)\left[r + \gamma \max_{a'} q_\pi^*(a',s')\right]$

nothing changes

b) $v_\pi^*(s) = \max_a \left[r + \gamma v_\pi(s')\right]$

$q^*(s,a) = \left[r + \gamma \max_{a'} q_\pi^*(a',s')\right]$

c) we just need to replace the 2 update rules in the algorithm

- $V(s) \leftarrow \max_a \left[r + \gamma V_\pi(s')\right]$ instead of $V(s) \leftarrow \max_a \sum_{s',r} p(s',r|s,a)\left[r + \gamma V_\pi(s')\right]$
- $\pi(s) = \operatorname{argmax}_a \left[r + \gamma V_\pi(s')\right]$ instead of $\pi(s) = \operatorname{argmax}_a \sum_{s',r} p(s',r|s,a)\left[r + \gamma V(s')\right]$

question 4 (~10 min)

$S_1, A_1, 13, S_1, A_2, 7, S_1, A_1, 13, S_1, A_2, 14$ done

a) First visit MC

$q_\pi(S_1,A_1)$ = return from first seeing S_1, A_1 in the sequence

rewards $= [13, 7, 13, 14]$

$\gamma^k = [(\frac{1}{2})^0; \frac{1}{2}, (\frac{1}{2})^2, (\frac{1}{2})^3]$

$G_t = 13 + \frac{7}{2} + \frac{13}{4} + \frac{14}{8} = 21.5$

$\Rightarrow \boxed{q_\pi(S_1,A_1) = 21.5}$

$q_\pi(S_1,A_2) \Rightarrow$ rewards $= [7, 13, 14]$

$\gamma^k = (1, \frac{1}{2}, \frac{1}{4})$

$\Rightarrow G_t = 7 + \frac{13}{2} + \frac{14}{4} = 17$

$\boxed{q_\pi(S_1,A_2) = 17}$

3.1 3.a **2 / 2**

✓ **- 0 pts** Correct

e) One thing that I can think off is control exploration by adding a layer of unsupervised learning. For example, we can create a variable that measure the favorite genre based on activity history of the user. we could also use RNNs. Then we could add a bonus to the same type of movies.
(priority Experience replay for example with prioritization based on previous history)

Question 3 (~10 min)

a) $v_\pi^*(s) = \max_a \sum_{s',r} p(s',r|s,a) [r + \gamma v_{\pi^*}(s')]$

$q_\pi^*(s,a) = \sum_{s',r} p(s',r|s,a) [r + \gamma \max_{a'} q_\pi^*(a',s')]$

nothing changes

b) $v_\pi^*(s) = \max_a [r + \gamma v_\pi(s')]$

$q^*(s,a) = [r + \gamma \max_{a'} q_\pi^*(a',s')]$

c) we just need to replace the 2 update rules in the algorithm

- $V(s) \leftarrow \max_a [r + \gamma v_\pi(s')]$ instead of $V(s) \leftarrow \max_a \sum_{s',r} p(s',r|s,a)(r+\gamma v_\pi(s'))$
- $\pi(s) = \arg\max_a [r + \gamma v_\pi(s')]$ instead of $\pi(s) = \arg\max_a \sum_{s',r} p(s',r|s,a)[r+\gamma V(s')]$

question 4 (~10 min)

$S_1, A_1, 13, S_1, A_2, 7, S_1, A_1, 13, S_1, A_2, 14$ done

a) First visit MC

$q_\pi(S_1,A_1)$ = return from first seeing S_1, A_1 in the sequence

rewards $= [13, 7, 13, 14]$ $\quad G_t = 13 + \frac{7}{2} + \frac{13}{4} + \frac{14}{8} = 21.5$

$\gamma^k = [(\frac{1}{2})^0; \frac{1}{2}, (\frac{1}{2})^2, (\frac{1}{2})^3]$

$\Rightarrow \boxed{q_\pi(S_1,A_1) = 21.5}$

$q_\pi(S_1,A_2) \Rightarrow$ rewards $= [7, 13, 14]$

$\gamma^k = (1, \frac{1}{2}, \frac{1}{4})$ $\quad \Rightarrow G_t = 7 + \frac{13}{2} + \frac{14}{4} = 17$

$\boxed{q_\pi(S_1,A_2) = 17}$

3.2 3.b **2 / 2**

✓ **- 0 pts** Correct

e) One thing that I can think off is control exploration by adding a layer of unsupervised learning. For example, we can create a variable that measure the favorite genre based on activity history of the user. we could also use RNNs. Then we could add a bonus to the same type of movies.
(priority Experience replay for example with prioritization based on previous history)

Question 3 (~10 min)

a) $v_{\pi}^*(s) = \max_a \sum_{s',r} p(s',r|s,a)[r + \gamma v_{\pi^*}(s')]$

$q_{\pi}^*(s,a) = \sum_{s',r} p(s',r|s,a)[r + \gamma \max_{a'} q_{\pi}^*(a',s')]$

nothing changes

b) $v_{\pi}^*(s) = \max_a [r + \gamma v_{\pi}(s')]$

$q^*(s,a) = [r + \gamma \max_{a'} q_{\pi}^*(a',s')]$

c) we just need to replace the 2 update rules in the algorithm

- $V(s) \leftarrow \max_a [r + \gamma v_{\pi}(s')]$ instead of $V(s) \leftarrow \max_a \sum_{s',r} p(s',r|s,a)(r+\gamma v_{\pi}(s))$
- $\pi(s) = \arg\max_a [r + \gamma v_{\pi}(s')]$ instead of $\pi(s) = \arg\max_a \sum_{s',r} p(s',r|s,a)[r+\gamma V(s')]$

question 4 (~10min)

$S_1, A_1, 13, S_1, A_2, 7, S_1, A_1, 13, S_1, A_2, 14$ done

a) First visit MC

$q_{\pi}(S_1, A_1)$ = return from first seeing S_1, A_1 in the sequence

rewards = $[13, 7, 13, 14]$ $\quad G_t = 13 + \frac{7}{2} + \frac{13}{4} + \frac{14}{8} = 21.5$

$\gamma^k = [(\frac{1}{2})^0; \frac{1}{2}, (\frac{1}{2})^2, (\frac{1}{2})^3]$

$\Rightarrow \boxed{q_{\pi}(S_1, A_1) = 21.5}$

$q_{\pi}(S_1, A_2) \Rightarrow$ rewards = $[7, 13, 14]$

$\gamma^k = (1, \frac{1}{2}, \frac{1}{4})$ $\Rightarrow G_t = 7 + \frac{13}{2} + \frac{14}{4} = 17$

$\boxed{q_{\pi}(S_1, A_2) = 17}$

3.3 3.c **3 / 3**

✓ **- 0 pts** Correct

e) One thing that I can think off is control exploration by adding a layer of unsupervised learning. For example, we can create a variable that measure the favorite genre based on activity history of the user. we could also use RNNs. Then we could add a bonus to the same type of movies.
(priority Experience replay for example with prioritization based on previous history)

Question 3 (~10 min)

a) $v_\pi^*(s) = \max_a \sum_{s',r} p(s',r|s,a)\left[r + \gamma v_{\pi^*}(s')\right]$

$q_\pi^*(s,a) = \sum_{s',r} p(s',r|s,a)\left[r + \gamma \max_{a'} q_\pi^*(a',s')\right]$

nothing changes

b) $v_\pi^*(s) = \max_a \left[r + \gamma v_\pi(s')\right]$

$q^*(s,a) = \left[r + \gamma \max_{a'} q_\pi^*(a',s')\right]$

c) we just need to replace the 2 update rules in the algorithm

- $V(s) \leftarrow \max_a \left[r + \gamma V_\pi(s')\right]$ instead of $V(s) \leftarrow \max_a \sum_{s',r} p(s',r|s,a)\left[r+\gamma V_\pi(s)\right]$
- $\pi(s) = \arg\max_a \left[r + \gamma V_\pi(s')\right]$ instead of $\pi(s) = \arg\max_a \sum_{s',r} p(s',r|s,a)\left[r+\gamma V(s')\right]$

question 4 (~10min)

$S_1, A_1, 13, S_1, A_2, 7, S_1, A_1, 13, S_1, A_2, 14$ done

a) First visit MC

$q_\pi(S_1,A_1)$ = return from first seeing S_1, A_1 in the sequence

rewards = $[13, 7, 13, 14]$

$\gamma^k = \left[(\tfrac{1}{2})^0 ; \tfrac{1}{2}, (\tfrac{1}{2})^2, (\tfrac{1}{2})^3\right]$

$G_t = 13 + \frac{7}{2} + \frac{13}{4} + \frac{14}{8} = 21.5$

$\Rightarrow \boxed{q_\pi(S_1,A_1) = 21.5}$

$q_\pi(S_1,A_2) \Rightarrow$ rewards = $[7, 13, 14]$

$\gamma^k = (1, \tfrac{1}{2}, \tfrac{1}{4})$

$\Rightarrow G_t = 7 + \frac{13}{2} + \frac{14}{4} = 17$

$\boxed{q_\pi(S_1,A_2) = 17}$

4.1 4.a 3 / 3

✓ - 0 pts Correct

b) Every visit MC

$q_\pi(S_1, A_1) = ?$

$G_{t_1} = 21.5$ same as in question a)

$G_{t_2} = ?$ rewards $= [13, 14]$ $\quad G_{t_2} = 13 + \frac{14}{2} = 20$

$\gamma^k = [1, \frac{1}{2}]$

$G_t = \frac{G_{t_1} + G_{t_2}}{2} = \frac{21.5 + 20}{2} = 20.75 \Rightarrow \boxed{q_\pi(S_1, A_1) = 20.75}$

$q_\pi(S_1, A_2) = ?$

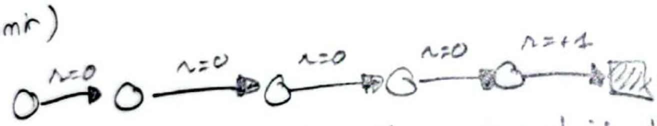
$G_{t_1} = 17$

$G_{t_2} = ?$ rewards $= [14]$ $\quad G_{t_2} = 14$

$\gamma^k = [1]$

$G_t = \frac{G_{t_1} + G_{t_2}}{2} = \frac{17 + 14}{2} = 15.5 \Rightarrow \boxed{q_\pi(S_1, A_2) = 15.5}$

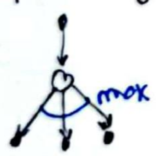
Question 5: (~5min)



in this case there is no stochasticity. The environment is deterministic. if we use TD this will take many steps to get the right values of all states. However with Monte carlo, only one update will yield the exact results.

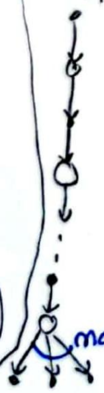
Question 6: ~ 30min

1-step Q-learning



$Q(S,A) = Q(S,A) + \alpha[R_{t+1} + \gamma \max_{a} Q(S_{t+1}, a) - Q(S,A)]$

n-step Q learning

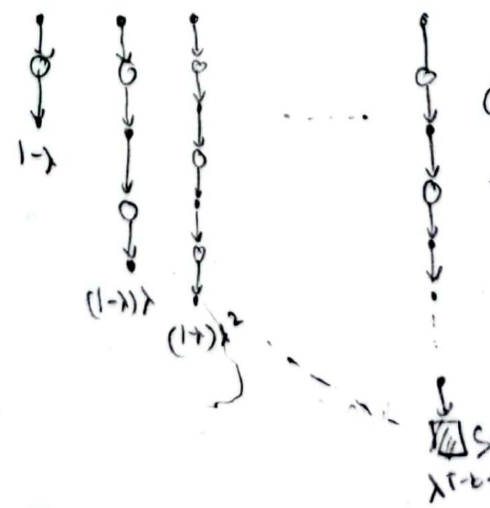


$G_{t:t+n} = R_{t+1} + \gamma R_{t+2} + \dots \gamma^n \max_a Q(s', a)$

$Q(S,A) = Q(S,A) + \alpha[G_{t:t+n} - Q(S,A)]$

Q(λ)

$Q(S,A) = Q(S,A) + \alpha[G_t^\lambda - Q(S,A)]z_t$



$G_t^\lambda = (1-\lambda)\sum_{n=1}^{T-t} \lambda^{n-1} G_{t:t+n}$

$z_t = \begin{cases} \gamma\lambda z_{t-1}(S) & \text{if } S \neq S_t \\ \gamma\lambda z_{t-1}(S) + 1 & \text{if } S = S_t \end{cases}$

(9)

4.2 4.b 3 / 3

✓ - 0 pts Correct

b) Every visit MC

$q_\pi(S_1, A_1) = ?$

$G_{t_1} = 21.5$ same as in question a)

$G_{t_2} = ?$ rewards $= [13, 14]$, $\gamma^k = [1, \frac{1}{2}]$ $\quad G_{t_2} = 13 + \frac{14}{2} = 20$

$G_t = \frac{G_{t_1} + G_{t_2}}{2} = \frac{21.5 + 20}{2} = 20.75 \Rightarrow \boxed{q_\pi(S_1, A_1) = 20.75}$

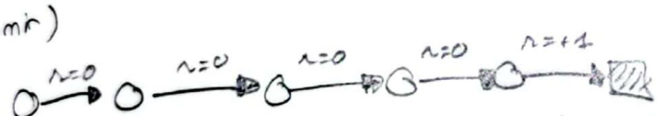
$q_\pi(S_1, A_2) = ?$

$G_{t_1} = 17$

$G_{t_2} = ?$ rewards $= [14]$, $\gamma^k = [1]$ $\quad G_{t_2} = 14$

$G_t = \frac{G_{t_1} + G_{t_2}}{2} = \frac{17 + 14}{2} = 15.5 \Rightarrow \boxed{q_\pi(S_1, A_2) = 15.5}$

Question 5: (~5min)



in this case there is no stochasticity. The environment is deterministic. if we use TD this will take many steps to get the right values of all states. However with Monte Carlo, only one update will yield the exact results.

Question 6: ~30min

1-step Q-learning

$Q(S,A) = Q(S,A) + \alpha[R_{t+1} + \gamma \max_{a} Q(S_{t+1}, a) - Q(S,A)]$

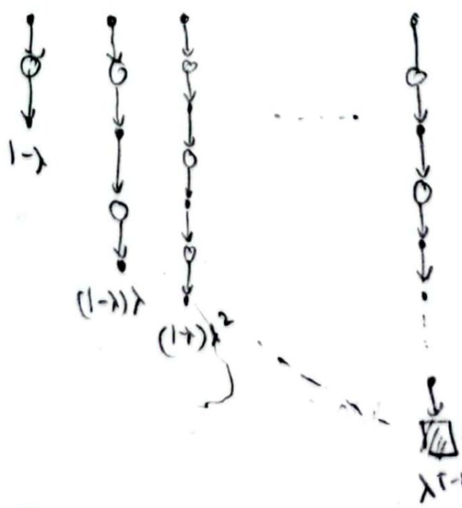
n-step Q learning

$G_{t:t+n} = R_{t+1} + \gamma R_{t+2} + \dots \gamma^n \max_a Q(s', a)$

$Q(S,A) = Q(S,A) + \alpha[G_{t:t+n} - Q(S,A)]$

$Q(\lambda)$

$Q(S,A) = Q(S,A) + \alpha[G_t^\lambda - Q(S,A)] z_t$



$G_t^\lambda = (1-\lambda) \sum_{n=1}^{T-t} \lambda^{n-1} G_{t:t+n}$

$z_t = \begin{cases} \gamma\lambda z_{t-1}(S) & \text{if } S \neq S_t \\ \gamma\lambda z_{t-1}(S) + 1 & \text{if } S = S_t \end{cases}$

9

5 Monte-carlo vs TD 5 / 5

✓ **- 0 pts** Correct

b) Every visit MC

$q_\pi(S_1, A_1) = ?$

$G_{t_1} = 21.5$ same as in question a)

$G_{t_2} = ?$ rewards $= [13, 14]$ $\quad G_{t_2} = 13 + \frac{14}{2} = 20$

$\gamma^k = [1, \frac{1}{2}]$

$$G_t = \frac{G_{t_1} + G_{t_2}}{2} = \frac{21.5 + 20}{2} = 20.75 \Rightarrow \boxed{q_\pi(S_1, A_1) = 20.75}$$

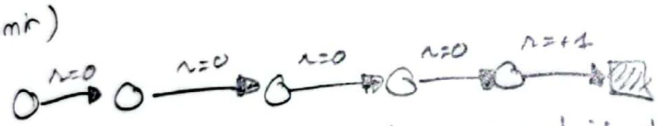
$q_\pi(S_1, A_2) = ?$

$G_{t_1} = 17$

$G_{t_2} = ?$ rewards $= [14]$, $\gamma^k = [1]$ $\quad G_{t_2} = 14$

$$G_t = \frac{G_{t_1} + G_{t_2}}{2} = \frac{17 + 14}{2} = 15.5 \Rightarrow \boxed{q_\pi(S_1, A_2) = 15.5}$$

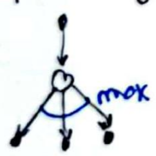
Question 5: (~5min)



in this case there is no stochasticity. The environment is deterministic. if we use TD this will take many steps to get the right values of all states. However with Monte carlo, only one update will yield the exact results.

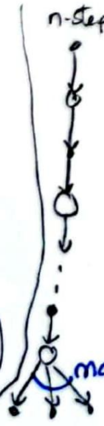
Question 6: ~30min

1-step Q-learning



$Q(S,A) = Q(S,A) + \alpha[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S,A)]$

n-step Q learning

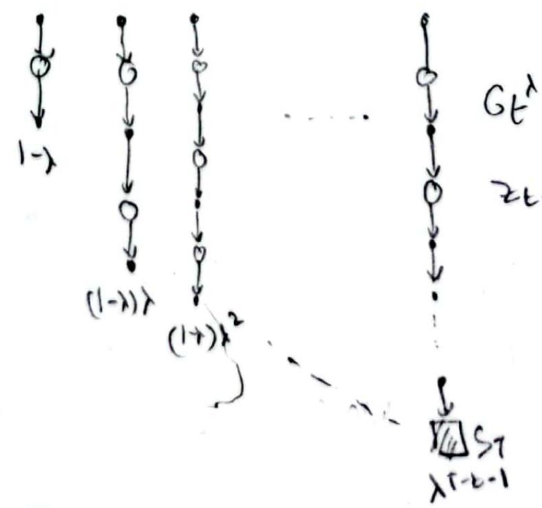


$G_{t:t+n} = R_{t+1} + \gamma R_{t+2} + \dots \gamma^n \max_a Q(s', a)$

$Q(S,A) = Q(S,A) + \alpha[G_{t:t+n} - Q(S,A)]$

Q(λ)

$Q(S,A) = Q(S,A) + \alpha[G_t^\lambda - Q(S,A)] z_t$



$G_t^\lambda = (1-\lambda)\sum_{n=1}^{T-t} \lambda^{n-1} G_{t:t+n}$

$z_t = \begin{cases} \gamma\lambda z_{t-1}(S) & \text{if } S \neq S_t \\ \gamma\lambda z_{t-1}(S) + 1 & \text{if } S = S_t \end{cases}$

4

6.1 6.a 6 / 6

✓ - 0 pts Correct

b) the 3 extra features added by DQN are

1- Experience Replay

DQN stores the agent's experiences at each time step in a buffer pooled over many episodes called replay buffer. Q-learning updates are computed by sampling a mini batch of transitions from the buffer

- each step of experience is potentially used in many weight updates
  → data efficiency
- randomizing samples break correlations and reduces the variance of updates
- by using the replay buffer, the behavior distribution is averaged over many of its previous states, avoiding divergence

2- Target Network

to avoid unstable learning we need to use a fixed version of the network which is the target network. otherwise the update target won't be stationary and we will result in unstable learning

3. clipping reward between (-1, 1)

in order to have small incremental updates

c) one problem is that in DQN we do the max over action. imagine these transitions have not been in the replay buffer or have been flushed out when we want to do the update.

I would try to increase the size of the buffer

6.2 6.b 4 / 4

✓ - 0 pts All Correct

b) the 3 extra features added by DQN are

1- Experience Replay
DQN stores the agent's experiences at each time step in a buffer pooled over many episodes called replay buffer. Q-learning updates are computed by sampling a mini batch of transitions from the buffer
- each step of experience is potentially used in many weight updates
  → data efficiency
- randomizing samples break correlations and reduces the variance of updates
- by using the replay buffer, the behavior distribution is averaged over many of its previous states, avoiding divergence

2- Target Network
to avoid unstable learning we need to use a fixed version of the network which is the target network. otherwise the update target won't be stationary and we will result in unstable learning

3. clipping reward between (-1, 1)
in order to have small incremental updates

c) one problem is that in DQN we do the max over action. imagine these transitions have not been in the replay buffer or have been flushed out when we want to do the update.
I would try to increase the size of the buffer

6.3 6.c **2 / 4**

✓ **- 2 pts** Partial marks

Question 7

by using SMDP Q-learning with normal q-learning updates, our algorithm is going to face some issues. Mainly the will find it very hard to know when to use the abstract actions vs the primitive actions. So let's say the agent is in the same room of the target. The agent doesn't know that. it will take an abstract action and get to another room. now it uses primitive actions but will not find the goal. This is probably why the agent doesn't learn faster. it is also important to note that the use of q-learning doesn't help since it uses max over all the actions (which in most cases will have equal values)..

Question 8

The markov property states that the entire history can be summarized in the last state and action

if we consider that the markov property is not respected, we can think of the following

TD: the fact that TD bootstrap will worsen the result. Since information in the present state is not enough. The information flow is biased a lot.

MC: will be more suitable since it computes the entire return and does not use only the next state

PG: Also use the entire return. Therefore they are more suitable than TD methods

7 Learning with options 4 / 5

✓ **- 1 pts See comments.**

You need to discuss more the nature of SMDP learning and its behavior under the given option setting.

Question 7

by using SMDP Q-learning with normal q-learning updates, our algorithm is going to face some issues. Mainly the will find it very hard to know when to use the abstract actions vs the primitive actions. So let's say the agent is in the same room of the target. The agent doesn't know that it will take an obstract action and get to another room. now it uses primitive actions but will not find the goal. This is probably why the agent doesn't learn faster. it is also important to note that the use of q-learning doesn't help since it uses max over all the actions (which in most cases will have equal values)..

Question 8

The markov property states that the entire history can be summarized in the last state and action

if we consider that the markov property is not respected, we can think of the following

TD: the fact that TD bootstrap will worsen the result. Since the information information in the present state is not enough, the information flow is biased a lot.

MC: will be more suitable since it computes the entire return and does not use only the next state

PG: Also use the entire return. Therefore they are more suitable than TD methods

8 Non-markovian Options 6 / 6

✓ - 0 pts Correct

Honour Code

By submitting this exam, I affirm that the Honour Code is in effect and the solutions submitted are my own. I neither consulted anyone else during the period of exam nor did I give away the solutions to other students taking the exam.

Question 1 ~20 min

a) On-policy vs off-policy

- in on-policy we only use one policy: the one we want to optimize. we also use this policy to explore (SARSA)
- in off-policy we use 2 policies: - behaviour policy which is used to interact with the world and explore
  (Q-learning)
  - target policy which we want to learn (which goal is to optimize the returns)

b) Model-free vs model based

- in model free we do not use or learn a model of the world the agent learns its policy by interacting with the environment: (SARSA)
- in model based we learn a model or this model is given to us and we use the model to do planning (Dynamic Programming)

c) Online RL vs Offline RL

- in online RL the agent interacts with the environment, while in offline the agent learns from a dataset of collected trajectories.
  Online RL example (SARSA)
  offline RL example (Dagger)

d) off-policy vs offline

as mentioned above off-policy is an online method as the agent interacts with the world. However the agent uses 2 different policies one to explore, the second being the one to optimize (ex: off policy MC with WIS)
On the contrary offline RL does not interact with the world (or at least most of the time); Like the off-policy, we can consider that there is more than one policy: the one to optimize and the ones used to collect the dataset of trajectories (ex: policy penalty methods)

9 Honor Code 0 / 0

✓ **- 0 pts** **Correct**