Team Members:
Ashwat Rajbhandari
Akshay Goenka
Yashwanth Gowda
Kaito Namatame

1. How do hyperparameters affect learning?

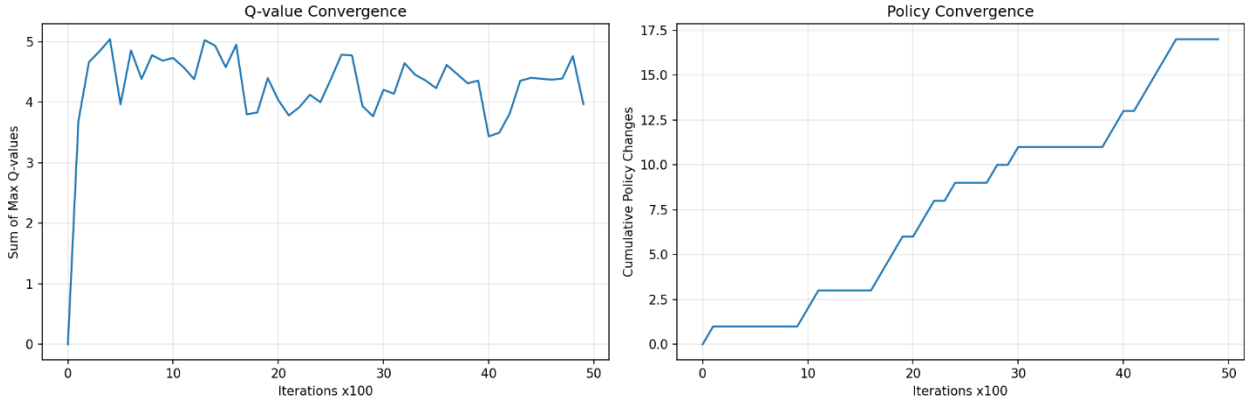
- When we run Q-learning for the gridworld problem, we noticed that the learning rate (α) determines how quickly the Q-value estimates get updated. If we set α high, our agent responds rapidly to the latest experience, which can help it adapt but also makes it prone to instability. If α, the learning rate is low, then learning is slower and more stable, because new experiences only gently influence the estimates.
- The discount factor (γ) tells our agent how much it should care about future rewards. With a high γ (close to 1), we find that the agent prefers longer, safer paths toward the goal, considering rewards that might be received in the future. With a lower γ, the agent is more greedy for immediate reward and may take riskier, shorter routes.
- Although the offline method uses a synthetic dataset, our offlinecode doesn't have an explicit exploration schedule (ϵ), we know that in online learning ϵ would decide how much the agent tries new actions. If ϵ is too high, the agent spends more time exploring, which is good early in training but inefficient later. Too little exploration could leave our agent stuck with a suboptimal policy.
    - Our choice of **α=0.1** provides a good balance, allowing for steady and stable convergence.
    - Our choice of **γ=0.9** encourages the agent to value future rewards for more balanced decision-making.
    - Our choice of $\boldsymbol{\epsilon = 0.1}$ for online Q learning. Epsilon was not used in offline method as we created a synthetic dataset.

---

2. Does Q-value or policy converge first?

The policy almost always converges before the Q-values.

In our implementation, we observe that the Q-values for each state-action pair gradually settle as the agent trains on experience data. When the updates become very small, we know that the Q-values have converged. Only after this stabilization do we extract the policy—by choosing, for every state, the action with the highest Q-value. So, Q-value convergence definitely comes first, and once the values stop changing, we can reliably read off the optimal policy for the gridworld. We got the following graph in one of our online Q learning method execution.



Optimal Policy For Penalty -1:

E	E	E	**+1**
N	**XXX**	N	**-1**
N	E	N	W

---

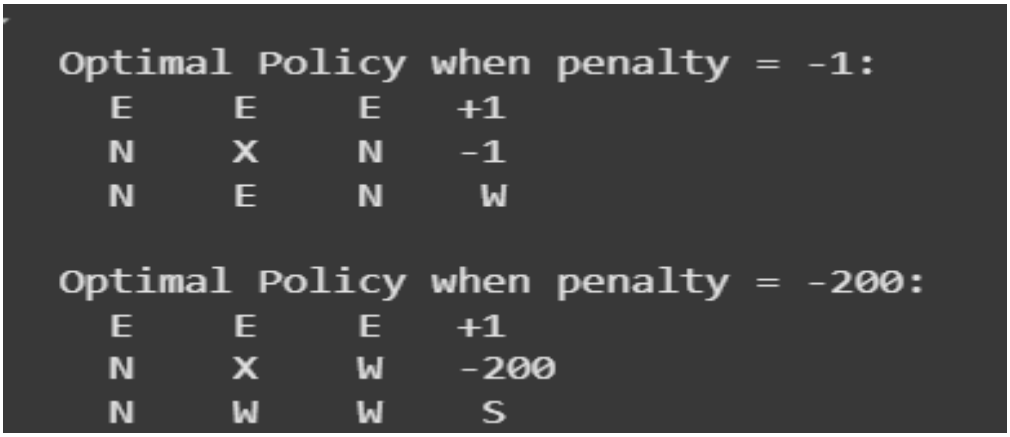
3. How does changing the trap penalty from -1 to -200 affect the solution?

We experimented with both trap penalties using our code. When the penalty in the trap state is -1, our agent sometimes risks traveling near it, and the optimal policy may sometimes steer toward the trap if the path seems worth it. But when we set the penalty to -200, we see a dramatic change the agent strongly avoids the trap and will only take safer and sometimes longer paths to reach the goal. This makes sense, because the value function punishes stepping into the trap much more severely in the second case. The comparison shows us how sensitive Q-learning is to the reward structure, changing one reward number can completely reshape the agent's behavior and policy.

Optimal Policy For Penalty -200:

E	E	E	**+1**
N	**XXX**	W	**-200**
N	W	W	S

Result:



Note: We tried solving the question using both offline and online methods and the Google collab links for both the running codes are below :-

Offline Method:

https://colab.research.google.com/drive/1GTWE55_IxJtys-3ZfRawCEqWQv1FHBZ8#scrollTo=w0Cp2ywQy6sJ

Online Method:

https://colab.research.google.com/drive/1wz88w7_1cvKBhEC40Bu5g1o1q-rMTrIl