CS6700 - REINFORCEMENT LEARNING
PROGRAMMING ASSIGNMENT 2

Team member 1: Gokul Venkatesan (Roll number: ME18B047)
Team member 2: Vishal (Roll number: CH18B013)

ENVIRONMENT DESCRIPTIONS:

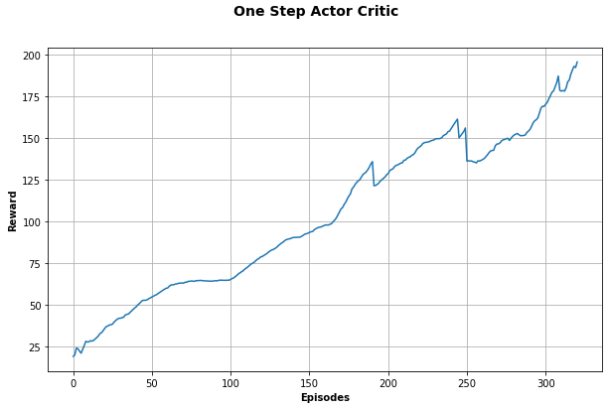
- Acrobot-v1: The acrobot system includes two joints and two links, where the joint between the two links is actuated. Initially, the links are hanging downwards, and the goal is to swing the end of the lower link up to a given height.
- CartPole-v1: A pole is attached by an un-actuated joint to a cart, which moves along a frictionless track. The system is controlled by applying a force of +1 or -1 to the cart. The pendulum starts upright, and the goal is to prevent it from falling over. A reward of +1 is provided for every timestep that the pole remains upright. The episode ends when the pole is more than 15 degrees from vertical, or the cart moves more than 2.4 units from the center.
- MountainCar-v0: A car is on a one-dimensional track, positioned between two ”mountains”. The goal is to drive up the mountain on the right; however, the car’s engine is not strong enough to scale the mountain in a single pass. Therefore, the only way to succeed is to drive back and forth to build up momentum.

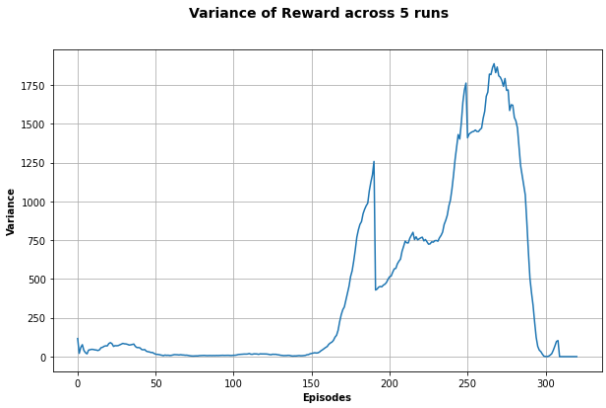
ACTOR CRITIC MODELS:

For Actor Critic, three different types of returns were tried for the three different environments: Single Return, Full Returns, and n-step returns. Below are the reward and variance plots for each variation averaged across 5 runs:

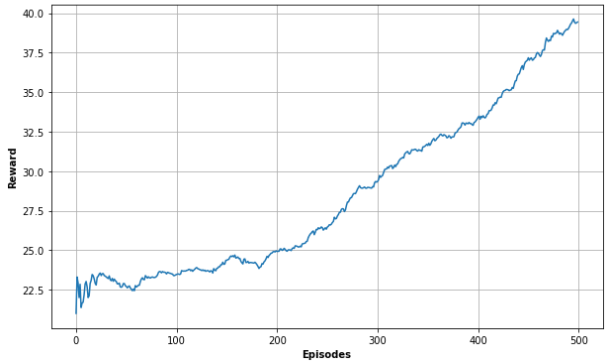
CartPole-v1:

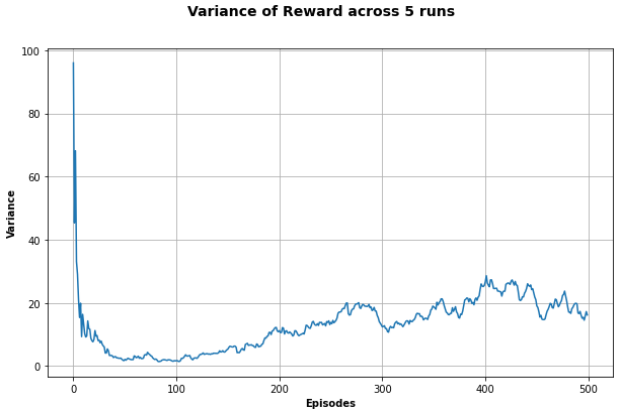
Single Return:



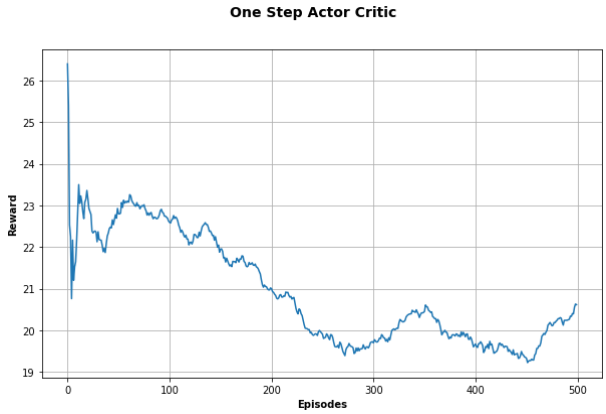


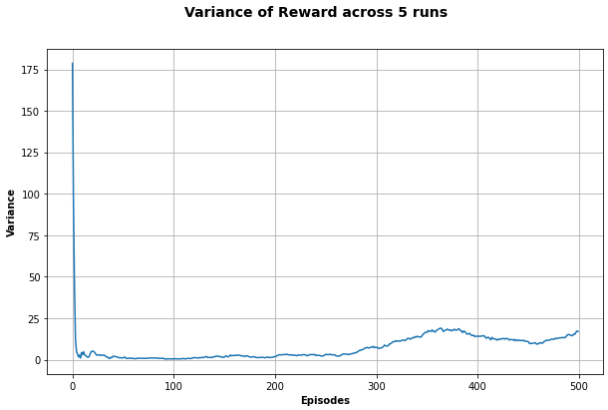
Full Returns:





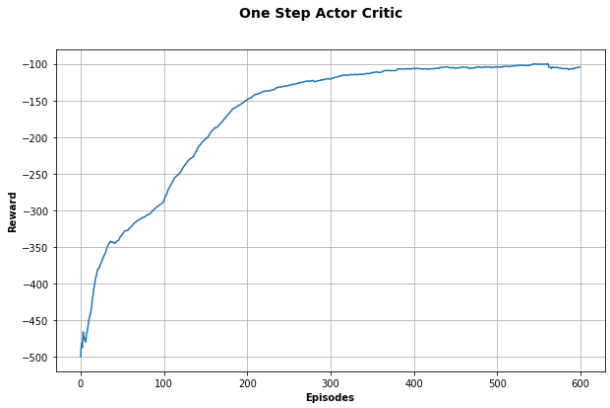
N-step Returns:

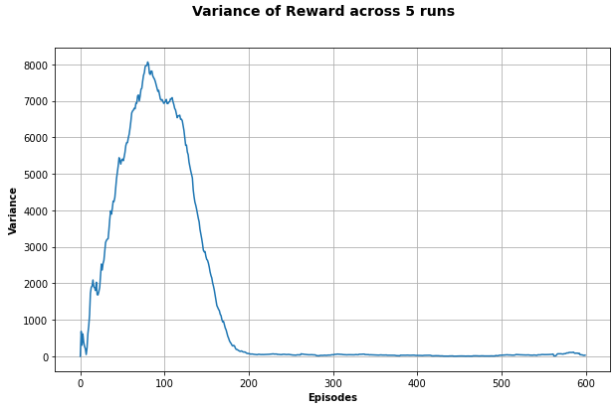




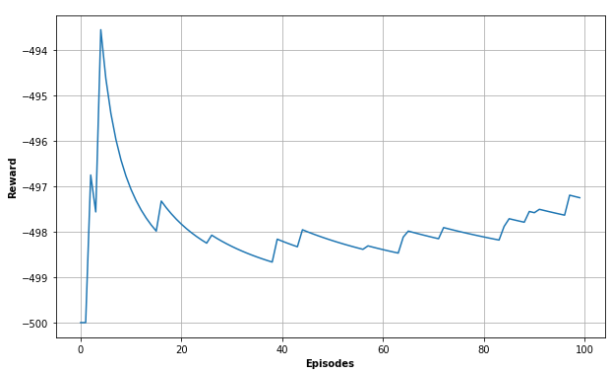
Actobot-v1:

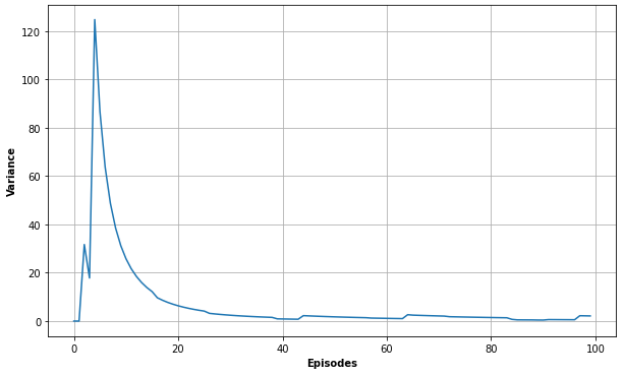
Single Return:



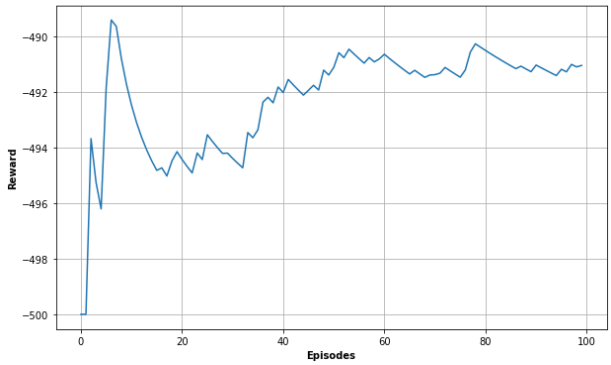


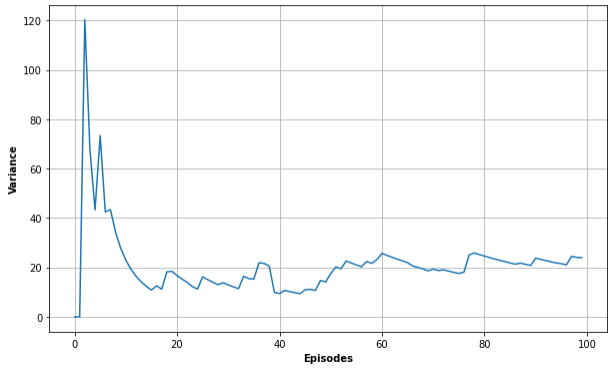
Full Returns:





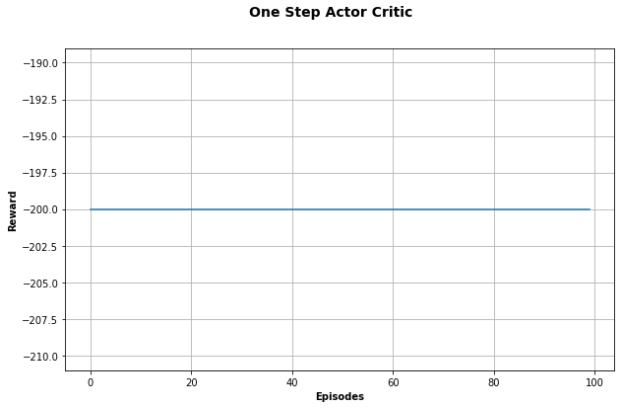
N-step returns:

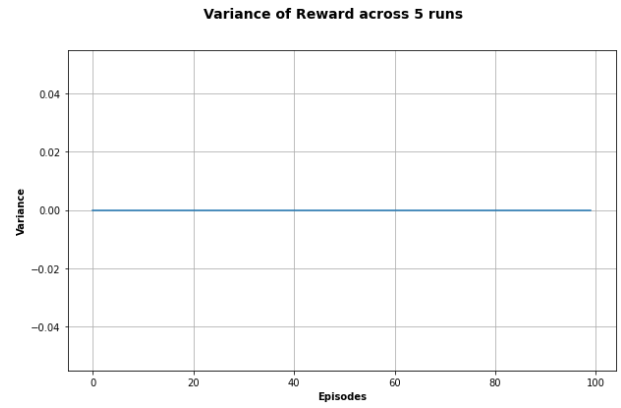




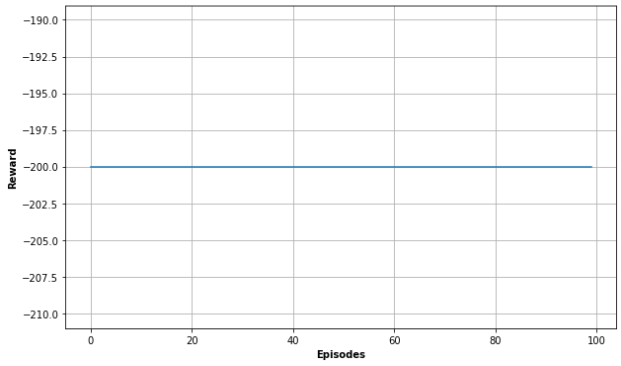
MountainCar-v0:

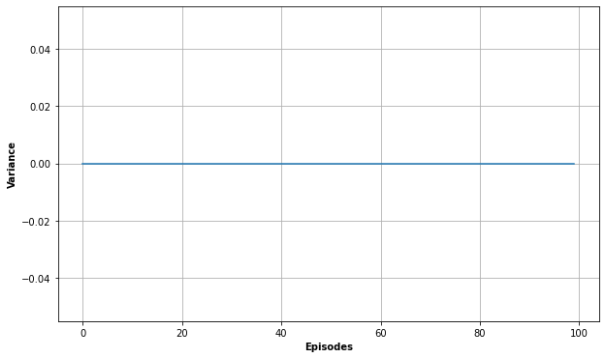
Single Return:



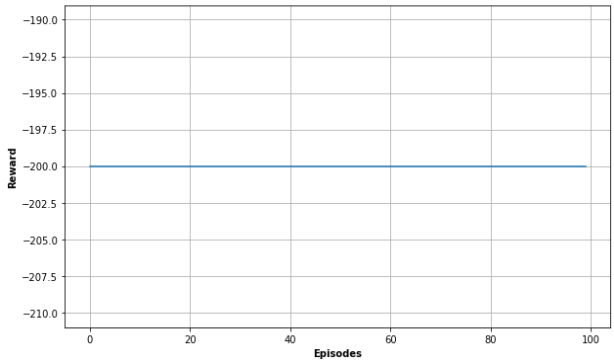


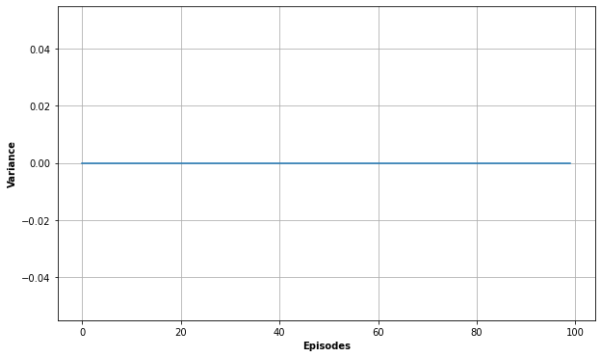
Full Returns:





N-step returns:





As you can see in the plots, convergence could not be achieved for the MountainCar environment. This could be because of the nature of the reward function of the environment, which allows for only 200 time steps to reach the goal which makes it very hard for the environment to be solved.

Hyperparameter values:

Environment	CartPole	Acrobot	MountainCar
Learning Rate	1e-4	2e-5	1e-5
No. of Hidden Layers	1024, 512	256, 512	512, 512

CS6700 - REINFORCEMENT LEARNING
PROGRAMMING ASSIGNMENT 2

Acrobot - v1:

Metric: The reward curve is computed for each variation. From this, we compute the moving average of the past 100 rewards and plot it against the episodes. We calculate the area under the graph. Since the total rewards are negative, the smaller the area under the curve, the better is the performance of the agent.

Variation - 1 (Tutorial):

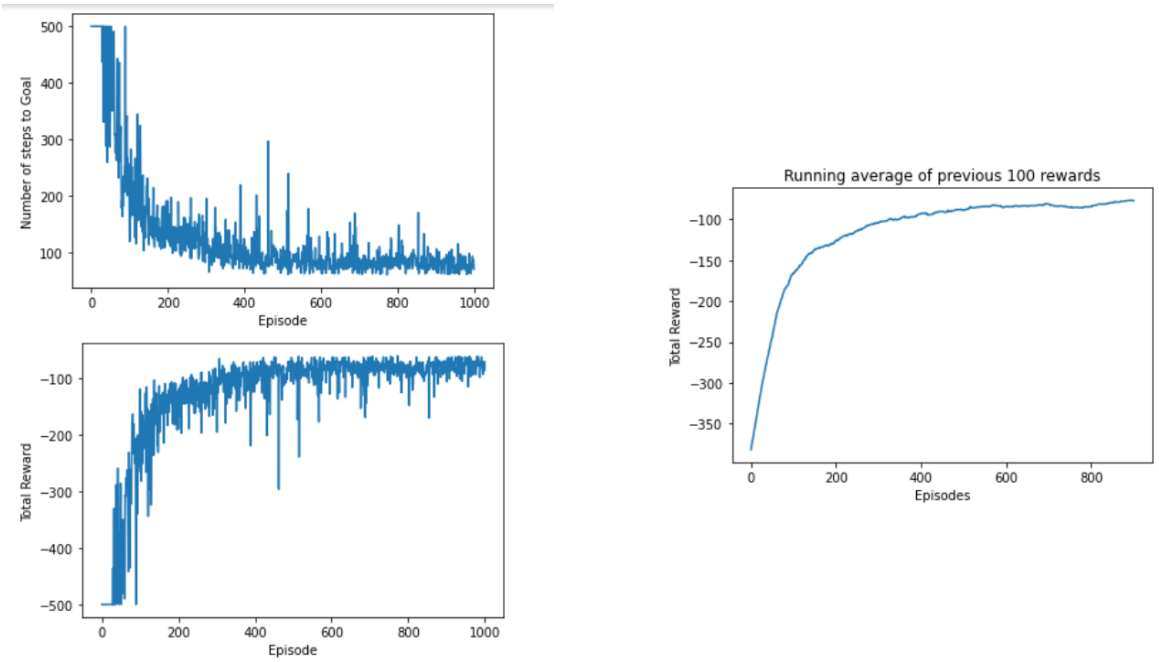
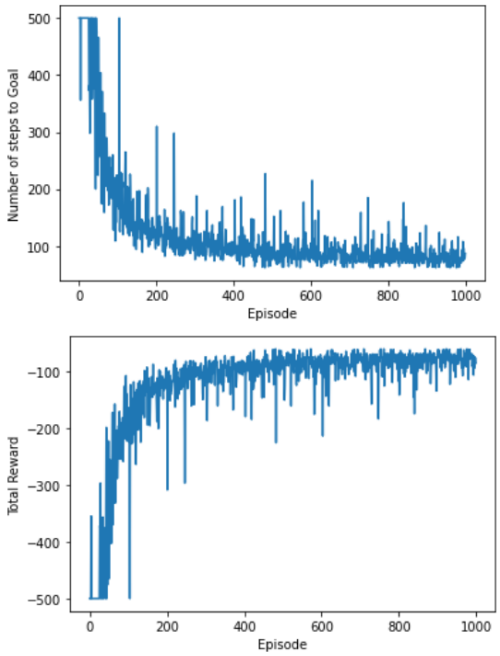


Figure 1: Number of steps in each episode and the total rewards against the episode (left); Running average of past 100 rewards (right)

Size of the Replay buffer: 100000
Batch Size: 64
Discount factor: 0.99
Learning rate: 0.0005
Update frequency of target Q-network: 20
Number of episodes: 1000
Gradient truncation: [-1, 1]
Number of neurons in the first and second layer of the Q-network: [128, 64]
Epsilon decay: 0.995

With an epsilon decay of 0.995, the value of epsilon reaches a very small value at the end of 1000 episodes. Hence, increasing the number of episodes beyond 1000 does not improve the network because the network would not be exploring much at that stage. With this setting, we find the area under the curve (AUC) to be **-103064.02.** From the reward curve, we can see that the agent has converged. Hence the area under the reward curve is an indicator of the agent's ability to quickly converge to the optimal Q-values.

Variation - 2:



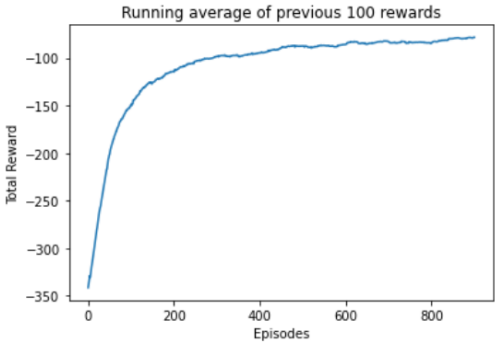
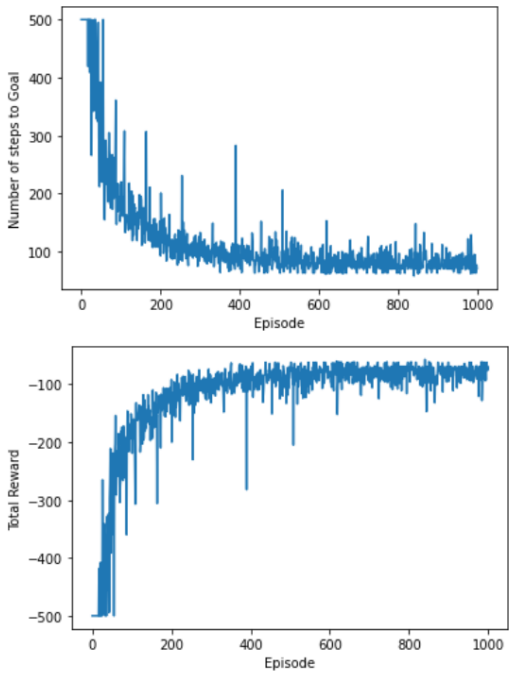


Figure 2: Number of steps in each episode and the total rewards against the episode (left); Running average of past 100 rewards (right)

Size of the Replay buffer: 100000
Batch Size: 64
Discount factor: 0.99
Learning rate: 0.0005
Update frequency of target Q-network: 50
Number of episodes: 1000
Gradient truncation: [-1, 1]
Number of neurons in the first and second layer of the Q-network: [128, 128]
Epsilon decay: 0.995

With an epsilon decay of 0.995, the value of epsilon reaches a very small value at the end of 1000 episodes. Hence, increasing the number of episodes beyond 1000 does not improve the network because the network would not be exploring much at that stage. With this setting, we find the area under the curve to be **-97394.57.** From the reward curve, we can see that the agent has converged. Also, a lower value of AUC than that of variation 1, indicates that the agent in variation 2 converges quicker than the agent in variation 1.

Variation - 3:



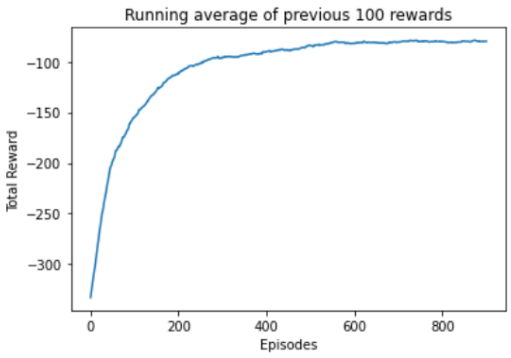


Figure 3: Number of steps in each episode and the total rewards against the episode (left); Running average of past 100 rewards (right)

Size of the Replay buffer: 100000
Batch Size: 100
Discount factor: 0.99
Learning rate: 0.0005
Update frequency of target Q-network: 50
Number of episodes: 1000
Gradient truncation: [-1, 1]
Number of neurons in the first and second layer of the Q-network: [128, 128]
Epsilon decay: 0.995

With an epsilon decay of 0.995, the value of epsilon reaches a very small value at the end of 1000 episodes. Hence, increasing the number of episodes beyond 1000 does not improve the network because the network would not be exploring much at that stage. With this setting, we find the area under the curve to be **-95281.095.** From the reward curve, we can see that the agent has converged. Also, a lower value of AUC than that of variation 2, indicates that the agent in variation 3 converges quicker than the agent in variation 2.

Variation - 4:

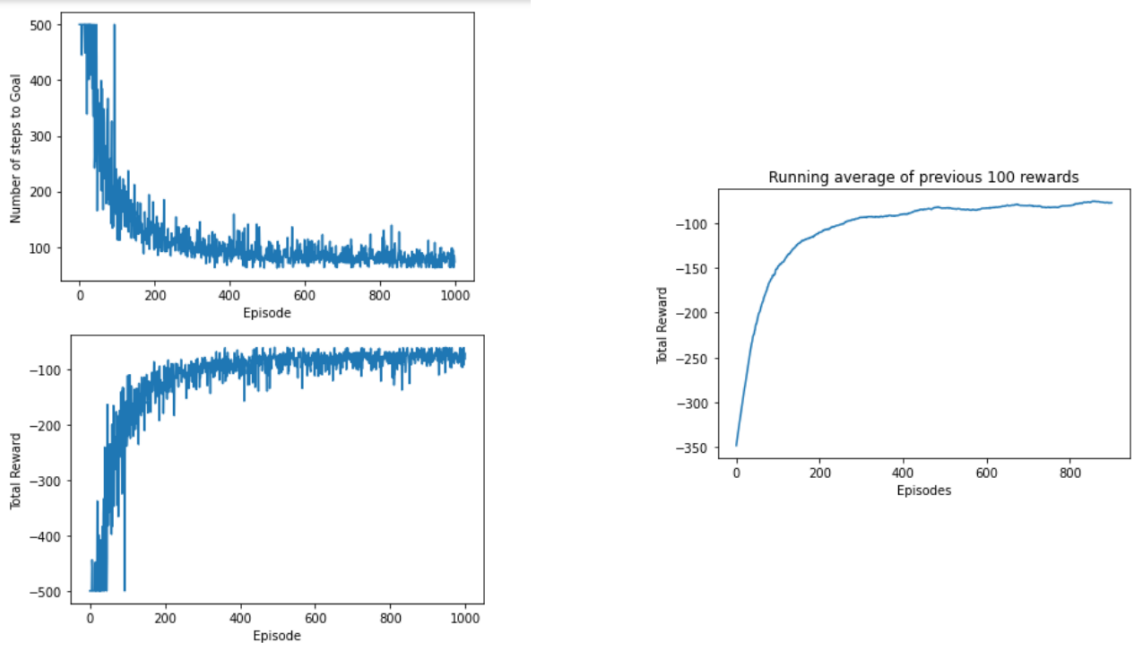


Figure 4: Number of steps in each episode and the total rewards against the episode (left); Running average of past 100 rewards (right)

Size of the Replay buffer: 100000
Batch Size: 100
Discount factor: 0.99
Learning rate: 0.0005
Update frequency of target Q-network: 50
Number of episodes: 1000
Gradient truncation: [-1, 1]
Number of neurons in the first and second layer of the Q-network: [256, 128]
Epsilon decay: 0.995

With an epsilon decay of 0.995, the value of epsilon reaches a very small value at the end of 1000 episodes. Hence, increasing the number of episodes beyond 1000 does not improve the network because the network would not be exploring much at that stage. With this setting, we find the area under the curve to be **-95378.13.** From the reward curve, we can see that the agent has converged. Also, a lower value of AUC than that of variation 2, indicates that the agent in variation 4 converges quicker than the agent in variation 2.

Variation - 5:

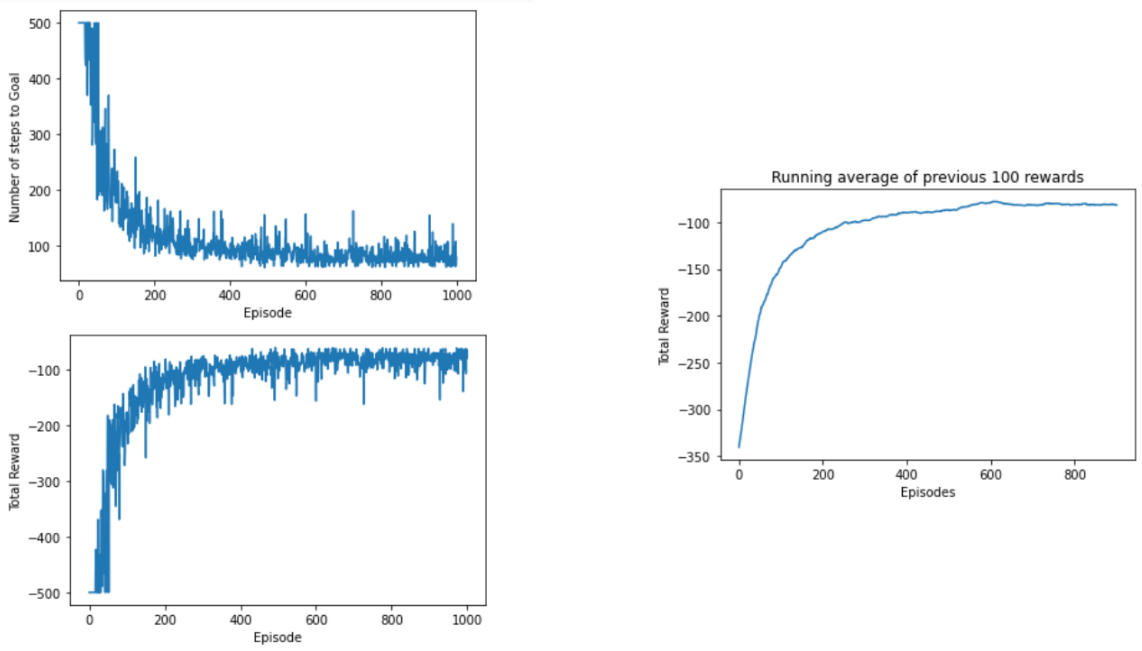


Figure 5: Number of steps in each episode and the total rewards against the episode (left); Running average of past 100 rewards (right)

Size of the Replay buffer: 100000
Batch Size: 120
Discount factor: 0.99
Learning rate: 0.0005
Update frequency of target Q-network: 50
Number of episodes: 1000
Gradient truncation: [-1.7, 1.7]
Number of neurons in the first and second layer of the Q-network: [256, 128]
Epsilon decay: 0.995

With an epsilon decay of 0.995, the value of epsilon reaches a very small value at the end of 1000 episodes. Hence, increasing the number of episodes beyond 1000 does not improve the network because the network would not be exploring much at that stage. With this setting, we find the area under the curve to be **-95216.11.** From the reward curve, we can see that the agent has converged. Also, a lower value of AUC than that of variation 3, indicates that the agent in variation 5 converges quicker than the agent in variation 3.

Inference:
1) In general, increasing the complexity of the Q-network enhanced the performance, provided the batch size, learning rate, and gradient truncation parameters are tuned appropriately. Increased complexity allows the network to extract complex features, which results in improved performance.
2) Increasing the batch size along with the model complexity enhanced the performance of the agent. As we increase the batch size, we estimate the gradient with a larger sample, which reduces the variance in the estimate. This resulted in smooth learning.

3) As we increase the model complexity, we need to increase the range of the gradient truncation. If it is narrow for large networks, learning becomes too slow and the network might not converge.
4) In general, increasing the update frequency of the target Q-network resulted in less variance in the reward curves.

CartPole - v1:

Metric: The reward curve is computed for each variation. From this, we compute the moving average of the past 100 rewards and plot it against the episodes. We calculate the area under the graph. Since the total rewards are positive, the greater the area under the curve, the better is the performance of the agent.

Variation - 1 (Tutorial):

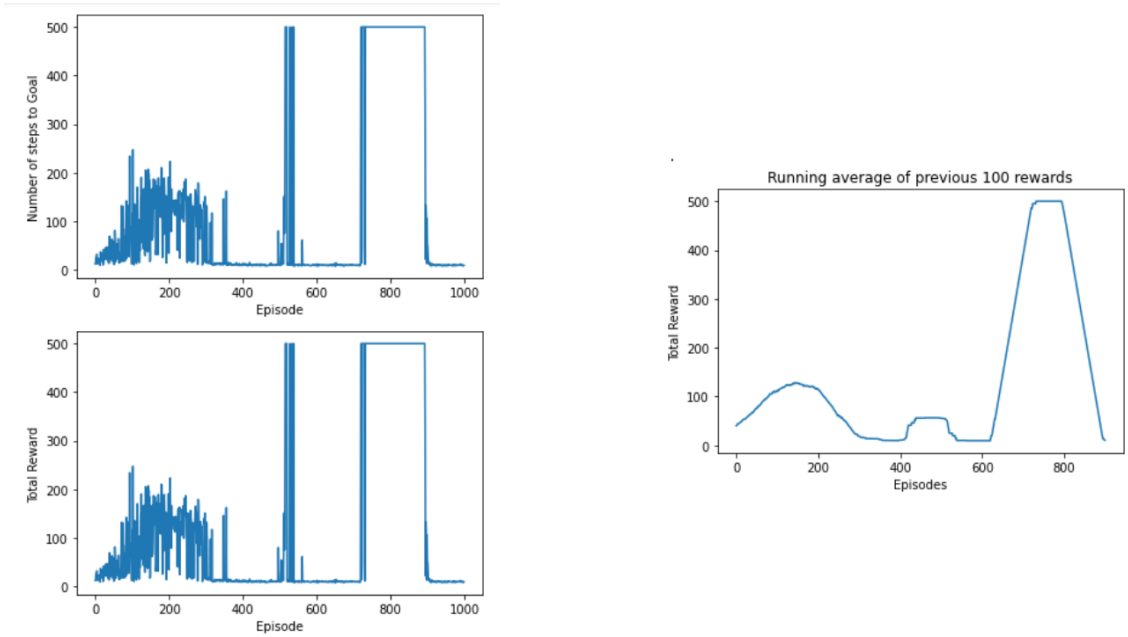


Figure 6: Number of steps in each episode and the total rewards against the episode (left); Running average of past 100 rewards (right)

Size of the Replay buffer: 100000
Batch Size: 64
Discount factor: 0.99
Learning rate: 0.0005
Update frequency of target Q-network: 20
Number of episodes: 1000
Gradient truncation: [-1, 1]
Number of neurons in the first and second layer of the Q-network: [128, 64]
Epsilon decay: 0.995

With an epsilon decay of 0.995, the value of epsilon reaches a very small value at the end of 1000 episodes. Hence, increasing the number of episodes beyond 1000 does not improve the network because the network would not be exploring much at that stage. With this setting, we find the area under the curve (AUC) to be **121699.48.** From the reward curve, we can see that

the performance of the agent shows oscillatory behavior. Still, the area under the reward curve is an indicator of the agent's ability to quickly converge to the optimal Q-values.

Variation - 2:

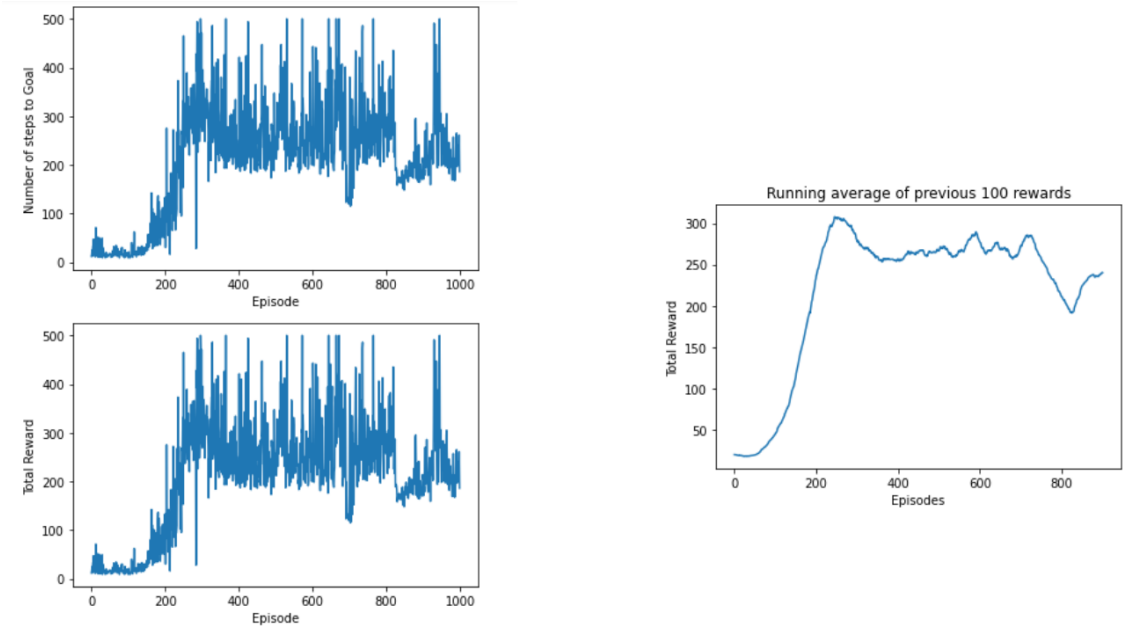


Figure 7: Number of steps in each episode and the total rewards against the episode (left); Running average of past 100 rewards (right)

Size of the Replay buffer: 100000
Batch Size: 64
Discount factor: 0.95
Learning rate: 0.00005
Update frequency of target Q-network: 50
Number of episodes: 1000
Gradient truncation: [-0.7, 0.7]
Number of neurons in the first and second layer of the Q-network: [128, 128]
Epsilon decay: 0.995

With an epsilon decay of 0.995, the value of epsilon reaches a very small value at the end of 1000 episodes. Hence, increasing the number of episodes beyond 1000 does not improve the network because the network would not be exploring much at that stage. With this setting, we find the area under the curve to be **197394.805.** From the reward curve, we can see that the performance of the agent shows oscillatory behavior, but not to the extent in variation 1. Though the agent is not able to solve the environment, a higher value of AUC than that of variation 1, indicates that the agent in variation 2 does show reduced instabilities during training.

Variation - 3:

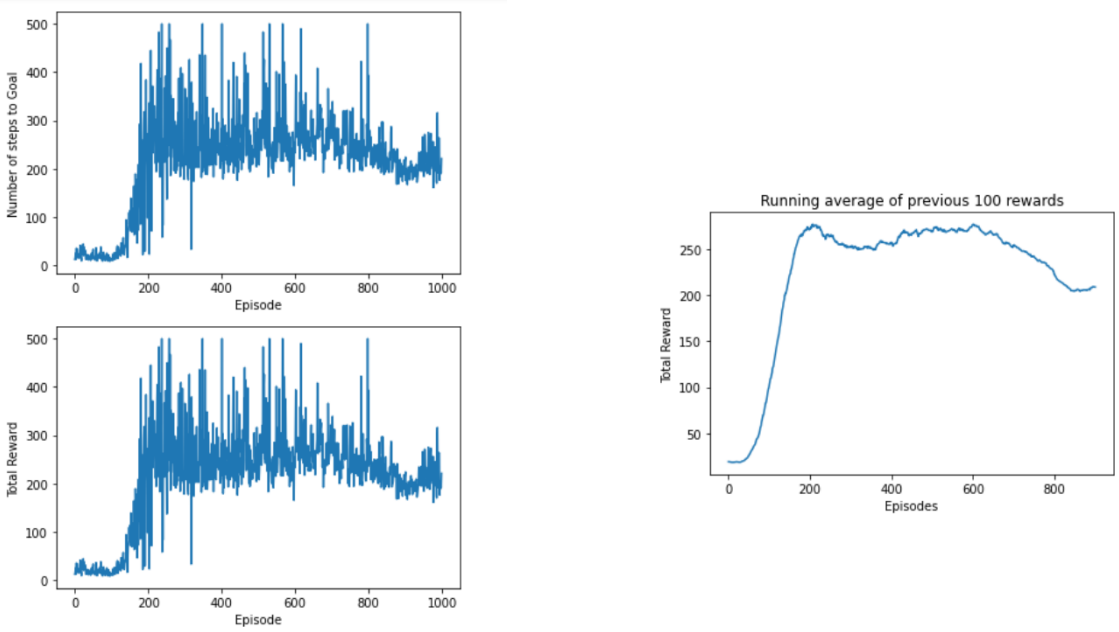


Figure 8: Number of steps in each episode and the total rewards against the episode (left); Running average of past 100 rewards (right)

Size of the Replay buffer: 100000
Batch Size: 96
Discount factor: 0.95
Learning rate: 0.00005
Update frequency of target Q-network: 50
Number of episodes: 1000
Gradient truncation: [-0.7, 0.7]
Number of neurons in the first and second layer of the Q-network: [128, 128]
Epsilon decay: 0.995

With an epsilon decay of 0.995, the value of epsilon reaches a very small value at the end of 1000 episodes. Hence, increasing the number of episodes beyond 1000 does not improve the network because the network would not be exploring much at that stage. With this setting, we find the area under the curve to be **201565.0.** From the reward curve, we can see that the performance of the agent shows oscillatory behavior, but not to the extent in variation 1. Though the agent is not able to solve the environment, a higher value of AUC than that of variation 2, indicates that the agent in variation 3 does show reduced instabilities during training.

Variation - 4:

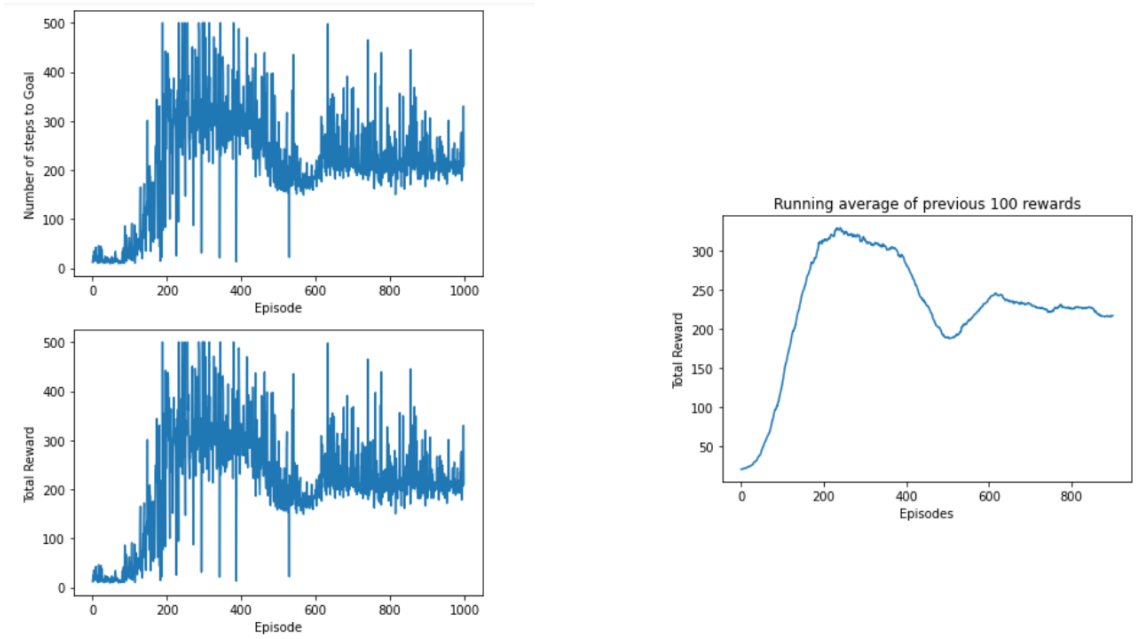


Figure 9: Number of steps in each episode and the total rewards against the episode (left); Running average of past 100 rewards (right)

Size of the Replay buffer: 100000
Batch Size: 96
Discount factor: 0.95
Learning rate: 0.00005
Update frequency of target Q-network: 50
Number of episodes: 1000
Gradient truncation: [-0.7, 0.7]
Number of neurons in the first and second layer of the Q-network: [200, 128]
Epsilon decay: 0.995

With an epsilon decay of 0.995, the value of epsilon reaches a very small value at the end of 1000 episodes. Hence, increasing the number of episodes beyond 1000 does not improve the network because the network would not be exploring much at that stage. With this setting, we find the area under the curve to be **204873.025.** From the reward curve, we can see that the performance of the agent shows oscillatory behavior, but not to the extent in variation 1. Though the agent is not able to solve the environment, a higher value of AUC than that of variation 3, indicates that the agent in variation 4 does show reduced instabilities during training.

Variation - 5:

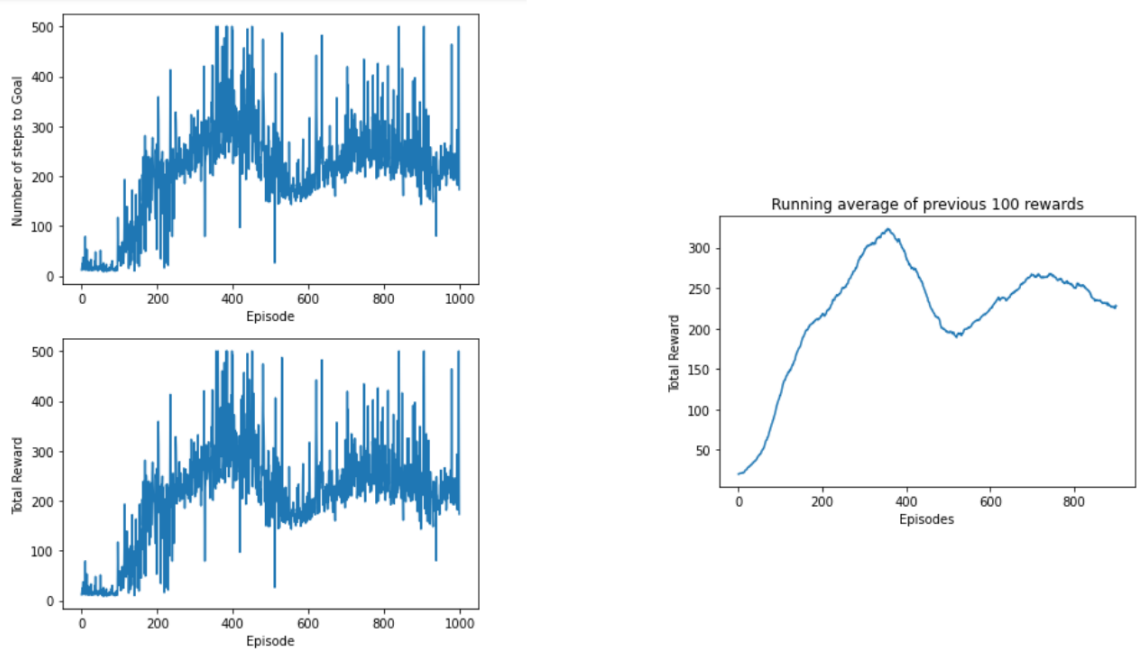


Figure 10: Number of steps in each episode and the total rewards against the episode (left); Running average of past 100 rewards (right)

Size of the Replay buffer: 100000
Batch Size: 110
Discount factor: 0.95
Learning rate: 0.00005
Update frequency of target Q-network: 50
Number of episodes: 1000
Gradient truncation: [-0.7, 0.7]
Number of neurons in the first and second layer of the Q-network: [200, 128]
Epsilon decay: 0.995

With an epsilon decay of 0.995, the value of epsilon reaches a very small value at the end of 1000 episodes. Hence, increasing the number of episodes beyond 1000 does not improve the network because the network would not be exploring much at that stage. With this setting, we find the area under the curve to be **198089.615.** From the reward curve, we can see that the performance of the agent shows oscillatory behavior, but not to the extent in variation 1. Though the agent is not able to solve the environment, a higher value of AUC than that of variation 2, indicates that the agent in variation 5 does show reduced instabilities during training.

Inference:
1) In general, increasing the complexity of the Q-network enhanced the performance, provided the batch size, learning rate, and gradient truncation parameters are tuned appropriately. Increased complexity allows the network to extract complex features, which results in improved performance.
2) Increasing the batch size along with the model complexity enhanced the performance of the agent. As we increase the batch size, we estimate the gradient with a larger sample, which reduces the variance in the estimate. This resulted in smooth learning.

3) As we increase the model complexity, we need to increase the range of the gradient truncation. If it is narrow for large networks, learning becomes too slow and the network might not converge.
4) In general, increasing the update frequency of the target Q-network resulted in less variance in the reward curves.
5) Changing the discount factor changes the dynamics of the problem. This is because we are trying to estimate the maximum possible value of the expected return (a function of the discount factor) at each and every state. For this environment, a discount factor of 0.95 seems to be appropriate.
6) As we increase the model complexity, we might need to tune the learning rate appropriately. A low value could slow down learning. A very high value could cause instabilities and oscillations during training, preventing the network from converging.

MountainCar - v0:

Metric: The reward curve is computed for each variation. From this, we compute the moving average of the past 100 rewards and plot it against the episodes. We calculate the area under the graph. Since the total rewards are positive, the greater the area under the curve, the better is the performance of the agent.

Variation - 1 (Tutorial):

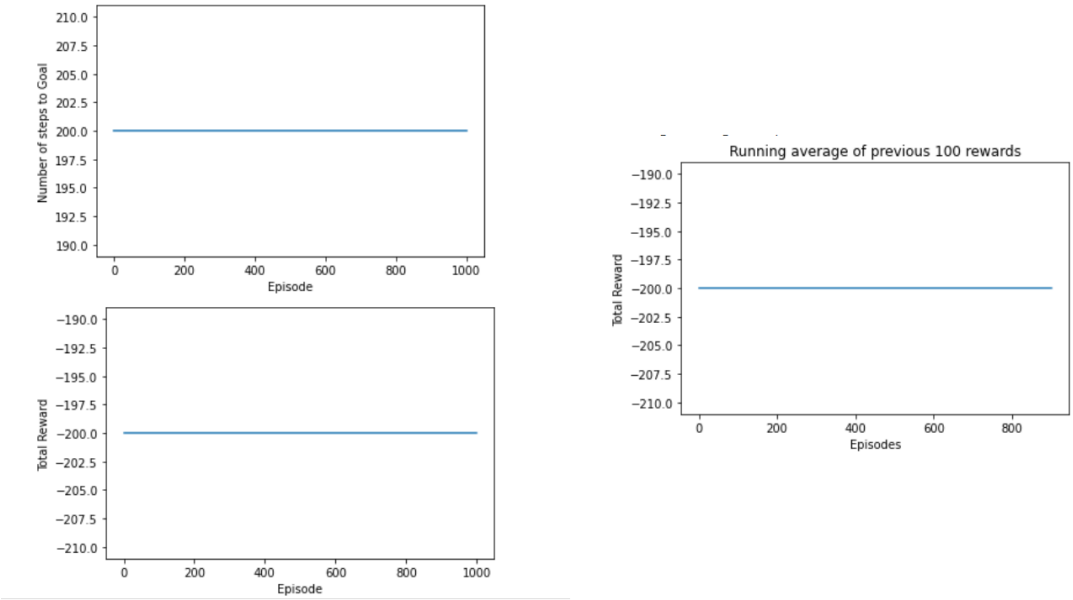


Figure 11: Number of steps in each episode and the total rewards against the episode (left); Running average of past 100 rewards (right)

Size of the Replay buffer: 100000
Batch Size: 64
Discount factor: 0.99
Learning rate: 0.0005
Update frequency of target Q-network: 20
Number of episodes: 1000
Gradient truncation: [-1, 1]
Number of neurons in the first and second layer of the Q-network: [128, 64]
Epsilon decay: 0.995

With an epsilon decay of 0.995, the value of epsilon reaches a very small value at the end of 1000 episodes. Hence, increasing the number of episodes beyond 1000 does not improve the network because the network would not be exploring much at that stage. With this setting, we find the area under the curve (AUC) to be **-180000.** From the reward curve, we can see that the agent has not learned an optimal policy. Still, the area under the reward curve is an indicator of the agent's ability to quickly converge to the optimal Q-values.

Variation - 2:

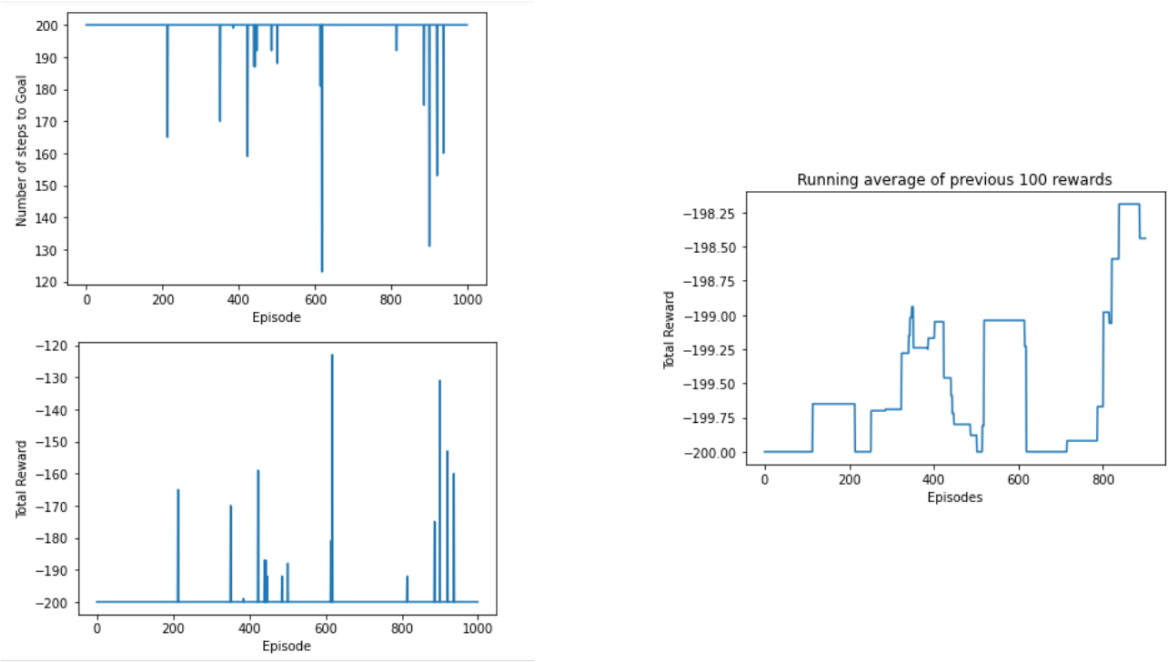


Figure 12: Number of steps in each episode and the total rewards against the episode (left); Running average of past 100 rewards (right)

Size of the Replay buffer: 100000
Batch Size: 32
Discount factor: 0.80
Learning rate: 0.02
Update frequency of target Q-network: 1
Number of episodes: 1000
Gradient truncation: [-1, 1]
Number of neurons in the first and second layer of the Q-network: [256, 256]
Epsilon decay: 0.995

With an epsilon decay of 0.995, the value of epsilon reaches a very small value at the end of 1000 episodes. Hence, increasing the number of episodes beyond 1000 does not improve the network because the network would not be exploring much at that stage. With this setting, we find the area under the curve (AUC) to be **-179578.98.** From the reward curve, we can see that the performance of the agent shows instabilities and oscillatory behavior.

Variation - 3:

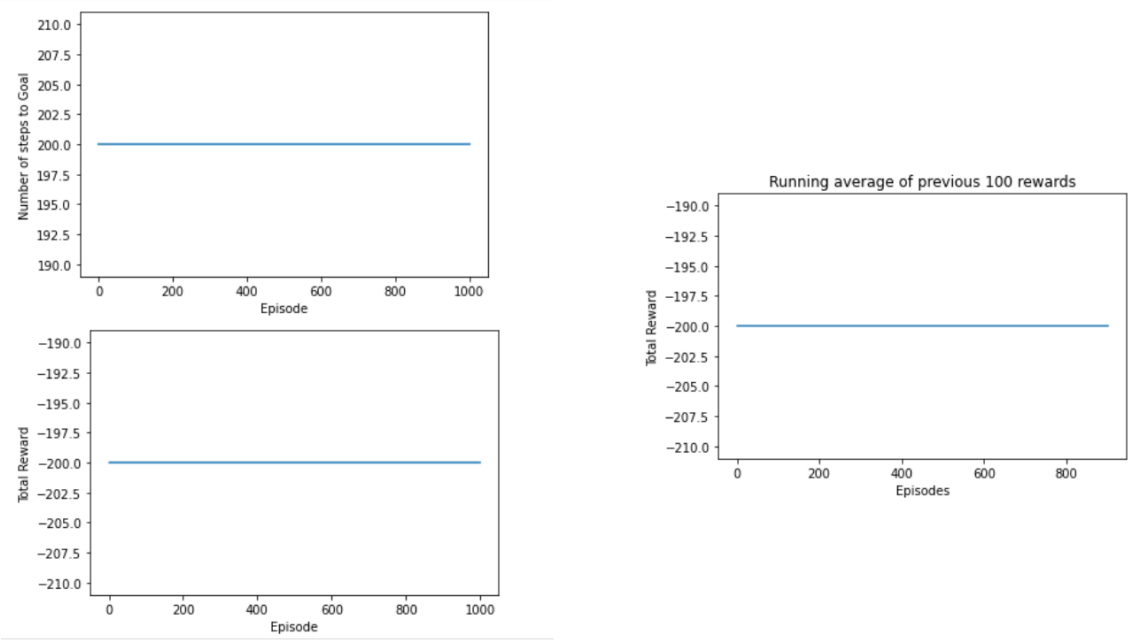


Figure 13: Number of steps in each episode and the total rewards against the episode (left); Running average of past 100 rewards (right)

Size of the Replay buffer: 100000
Batch Size: 32
Discount factor: 0.20
Learning rate: 0.02
Update frequency of target Q-network: 1
Number of episodes: 1000
Gradient truncation: [-1, 1]
Number of neurons in the first and second layer of the Q-network: [256, 256]
Epsilon decay: 0.995

With an epsilon decay of 0.995, the value of epsilon reaches a very small value at the end of 1000 episodes. Hence, increasing the number of episodes beyond 1000 does not improve the network because the network would not be exploring much at that stage. With this setting, we find the area under the curve (AUC) to be **-180000.** From the reward curve, we can see that the agent has not learned an optimal policy.

Variation - 4:

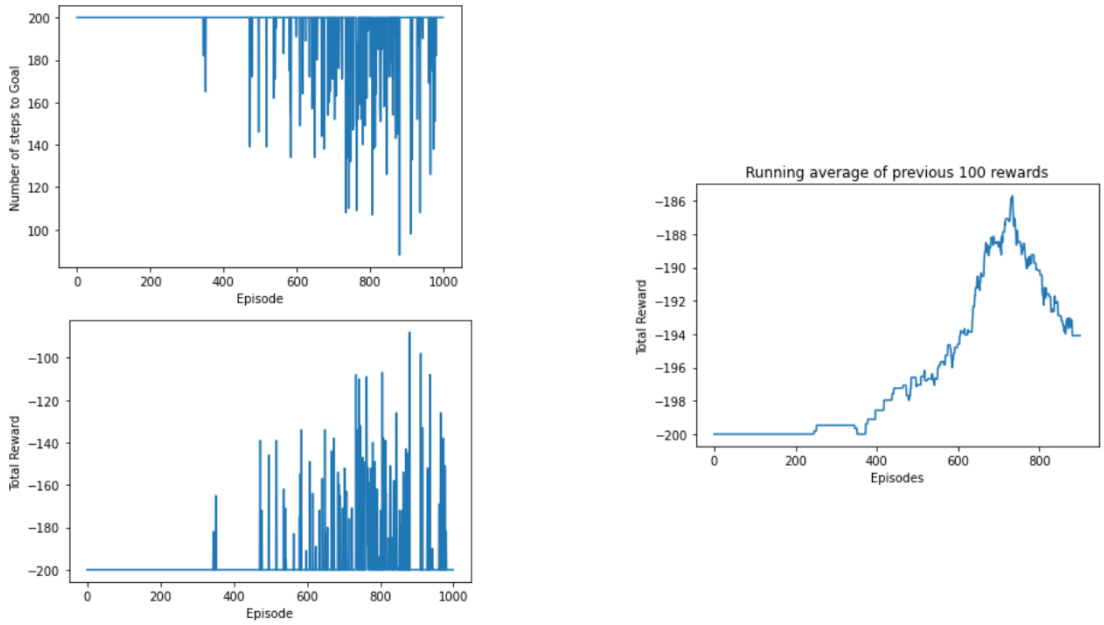


Figure 14: Number of steps in each episode and the total rewards against the episode (left); Running average of past 100 rewards (right)

Size of the Replay buffer: 100000
Batch Size: 32
Discount factor: 0.90
Learning rate: 0.002
Update frequency of target Q-network: 20
Number of episodes: 1000
Gradient truncation: [-2, 2]
Number of neurons in the first and second layer of the Q-network: [128, 128]
Epsilon decay: 0.995

With an epsilon decay of 0.995, the value of epsilon reaches a very small value at the end of 1000 episodes. Hence, increasing the number of episodes beyond 1000 does not improve the network because the network would not be exploring much at that stage. With this setting, we find the area under the curve (AUC) to be **-176531.745.** From the reward curve, we can see that the performance of the agent shows instabilities and oscillatory behavior. The maximum reward (running average of past 100 rewards) is about **-186.**

Variation - 5:

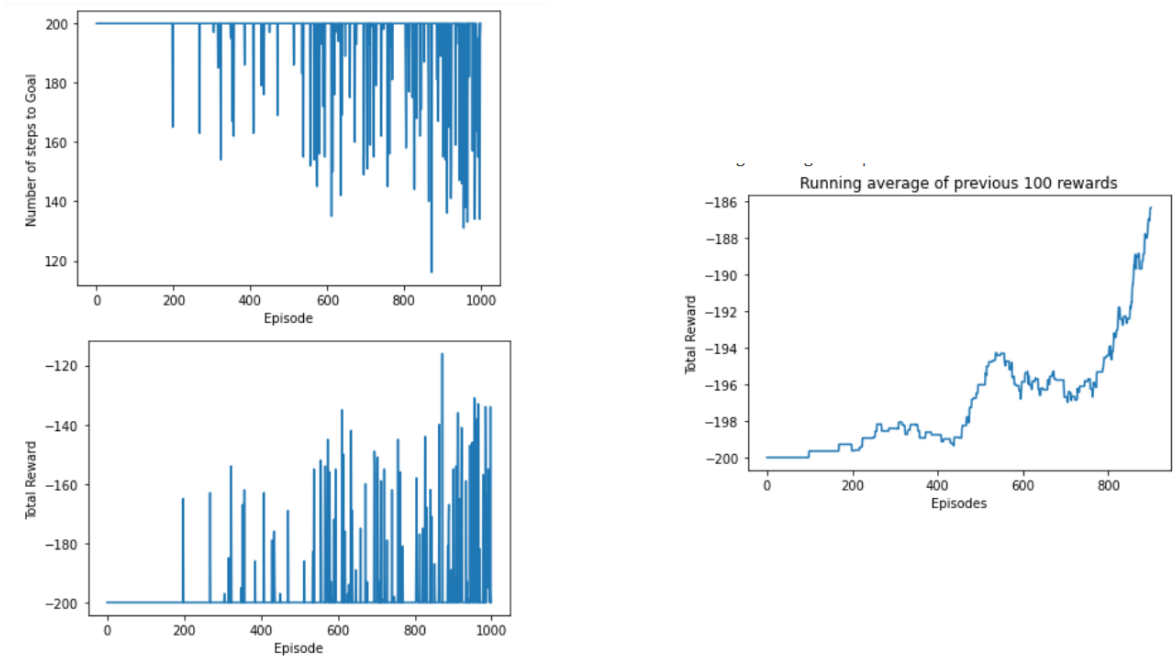


Figure 15: Number of steps in each episode and the total rewards against the episode (left); Running average of past 100 rewards (right)

Size of the Replay buffer: 100000
Batch Size: 64
Discount factor: 0.90
Learning rate: 0.002
Update frequency of target Q-network: 40
Number of episodes: 1000
Gradient truncation: [-2, 2]
Number of neurons in the first and second layer of the Q-network: [128, 128]
Epsilon decay: 0.995

With an epsilon decay of 0.995, the value of epsilon reaches a very small value at the end of 1000 episodes. Hence, increasing the number of episodes beyond 1000 does not improve the network because the network would not be exploring much at that stage. With this setting, we find the area under the curve (AUC) to be **-177314.635.** From the reward curve, we can see that the performance of the agent shows instabilities and oscillatory behavior. The maximum reward (running average of past 100 rewards) is about **-186.**

Variation - 6:

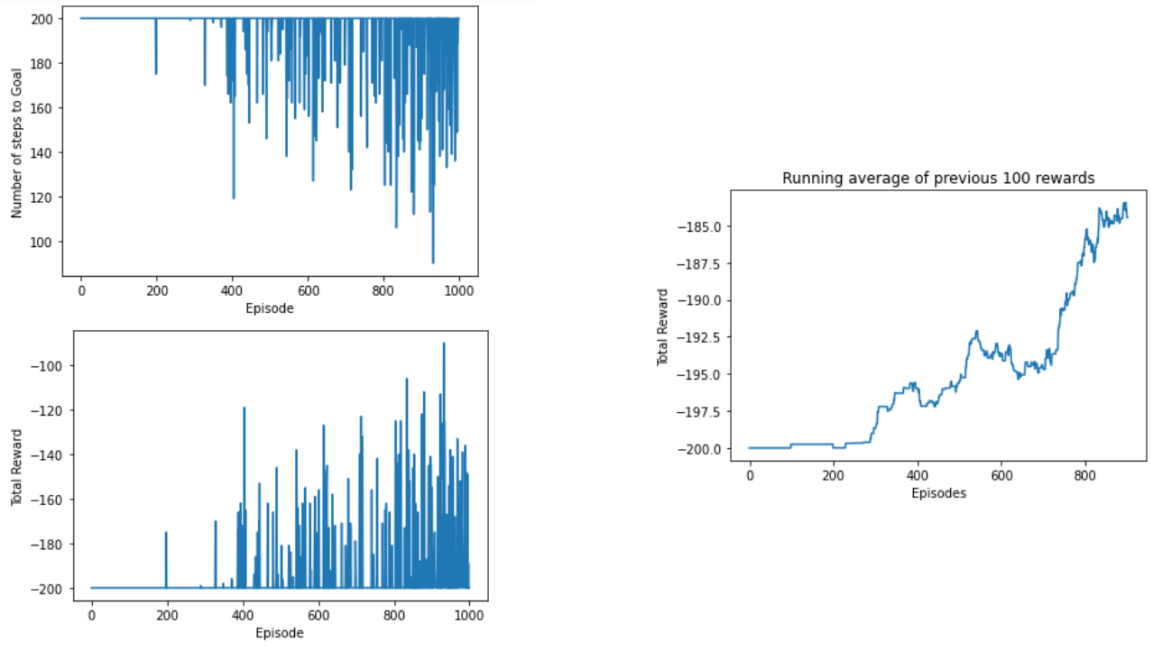


Figure 16: Number of steps in each episode and the total rewards against the episode (left); Running average of past 100 rewards (right)

Size of the Replay buffer: 100000
Batch Size: 64
Discount factor: 0.90
Learning rate: 0.002
Update frequency of target Q-network: 40
Number of episodes: 1000
Gradient truncation: [-2, 2]
Number of neurons in the first and second layer of the Q-network: [256, 128]
Epsilon decay: 0.995

With an epsilon decay of 0.995, the value of epsilon reaches a very small value at the end of 1000 episodes. Hence, increasing the number of episodes beyond 1000 does not improve the network because the network would not be exploring much at that stage. With this setting, we find the area under the curve (AUC) to be **-175640.935.** From the reward curve, we can see that the performance of the agent shows instabilities and oscillatory behavior. The maximum reward (running average of past 100 rewards) is about **-184.**

Variation - 7:

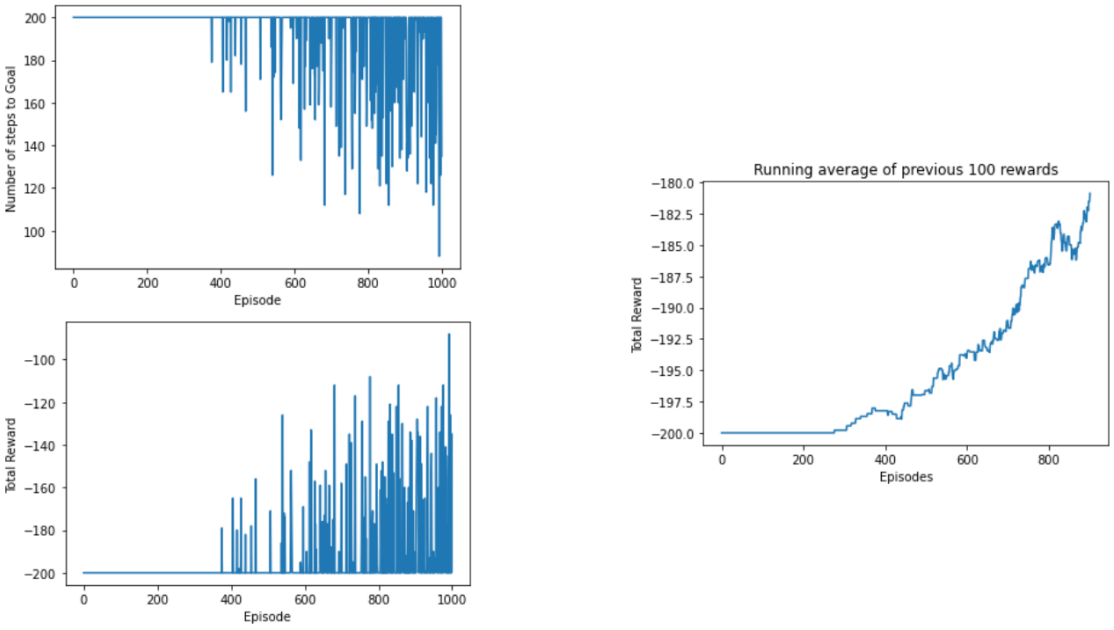


Figure 17: Number of steps in each episode and the total rewards against the episode (left); Running average of past 100 rewards (right)

Size of the Replay buffer: 100000
Batch Size: 64
Discount factor: 0.90
Learning rate: 0.0002
Update frequency of target Q-network: 50
Number of episodes: 1000
Gradient truncation: [-2, 2]
Number of neurons in the first and second layer of the Q-network: [256, 128]
Epsilon decay: 0.995

With an epsilon decay of 0.995, the value of epsilon reaches a very small value at the end of 1000 episodes. Hence, increasing the number of episodes beyond 1000 does not improve the network because the network would not be exploring much at that stage. With this setting, we find the area under the curve (AUC) to be **-175689.19.** From the reward curve, we can see that the performance of the agent shows instabilities and oscillatory behavior. The maximum reward (running average of past 100 rewards) is about **-180.**

Variation - 8:

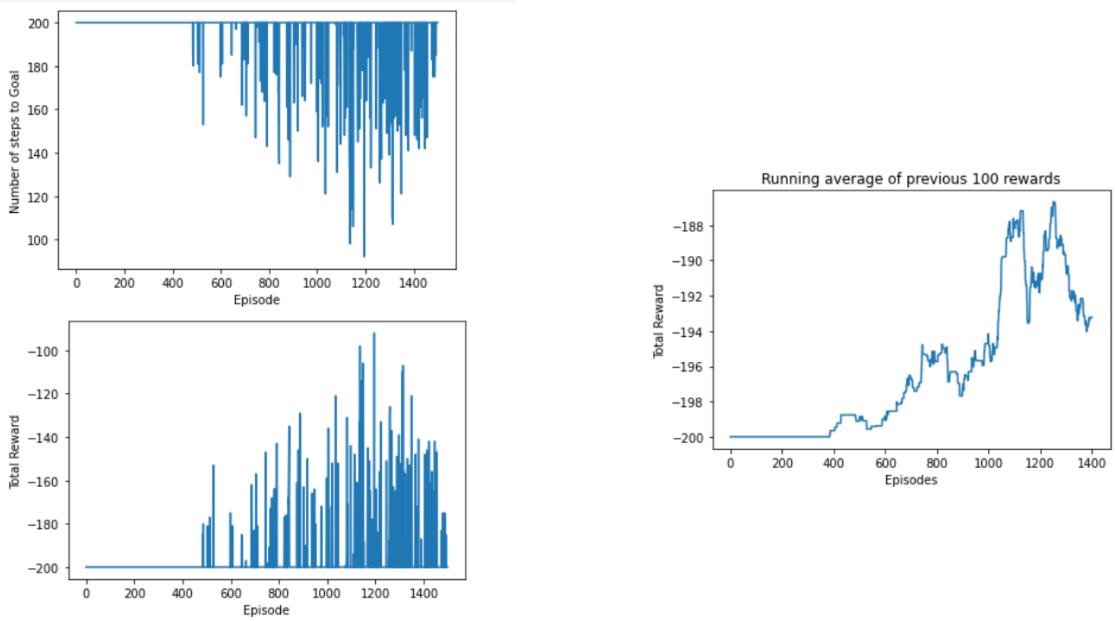


Figure 18: Number of steps in each episode and the total rewards against the episode (left); Running average of past 100 rewards (right)

Size of the Replay buffer: 100000
Batch Size: 64
Discount factor: 0.90
Learning rate: 0.0002
Update frequency of target Q-network: 50
Number of episodes: 1500
Gradient truncation: [-2, 2]
Number of neurons in the first and second layer of the Q-network: [256, 128]
Epsilon decay: 0.99

With an epsilon decay of 0.99, the value of epsilon reaches a very small value at the end of 1500 episodes. Hence, increasing the number of episodes beyond 1500 does not improve the network because the network would not be exploring much at that stage. With this setting, we find the area under the curve (AUC) to be **-274747.04.** From the reward curve, we can see that the performance of the agent shows instabilities and oscillatory behavior. The maximum reward (running average of past 100 rewards) is about **-187.**

Variation - 9:

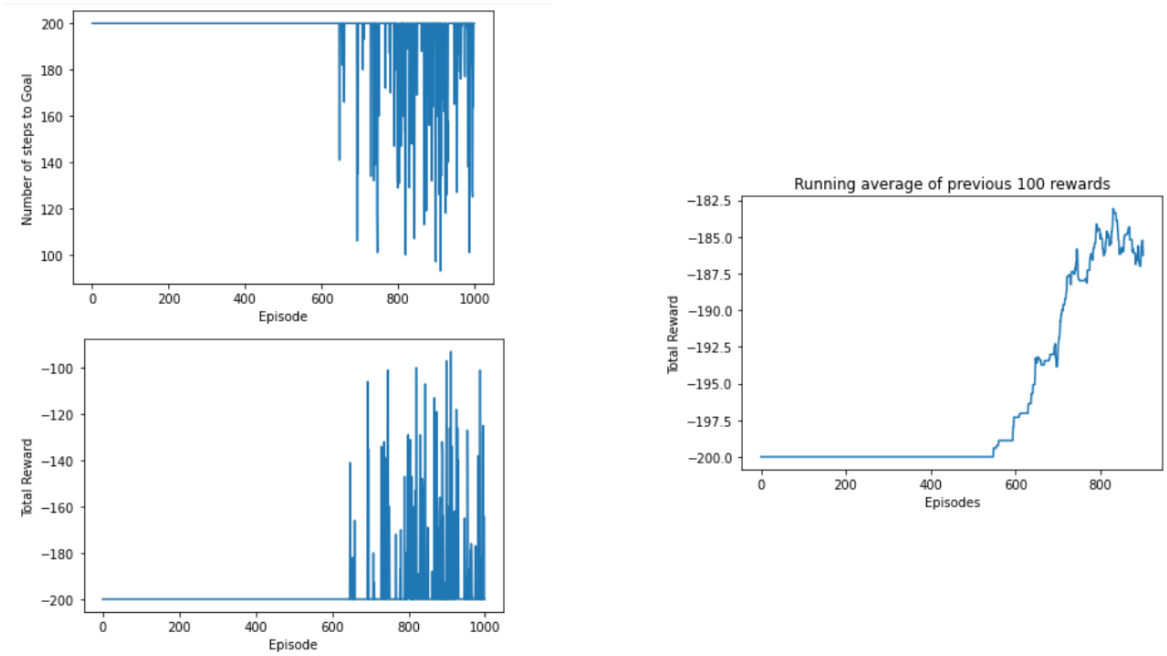


Figure 19: Number of steps in each episode and the total rewards against the episode (left); Running average of past 100 rewards (right)

Size of the Replay buffer: 100000
Batch Size: 96
Discount factor: 0.90
Learning rate: 0.0002
Update frequency of target Q-network: 50
Number of episodes: 1000
Gradient truncation: [-2, 2]
Number of neurons in the first and second layer of the Q-network: [256, 128]
Epsilon decay: 0.995

With an epsilon decay of 0.995, the value of epsilon reaches a very small value at the end of 1000 episodes. Hence, increasing the number of episodes beyond 1000 does not improve the network because the network would not be exploring much at that stage. With this setting, we find the area under the curve (AUC) to be -**176709.20.** From the reward curve, we can see that the performance of the agent shows instabilities and oscillatory behavior. The maximum reward (running average of past 100 rewards) is about **-182.**

Variation - 10:

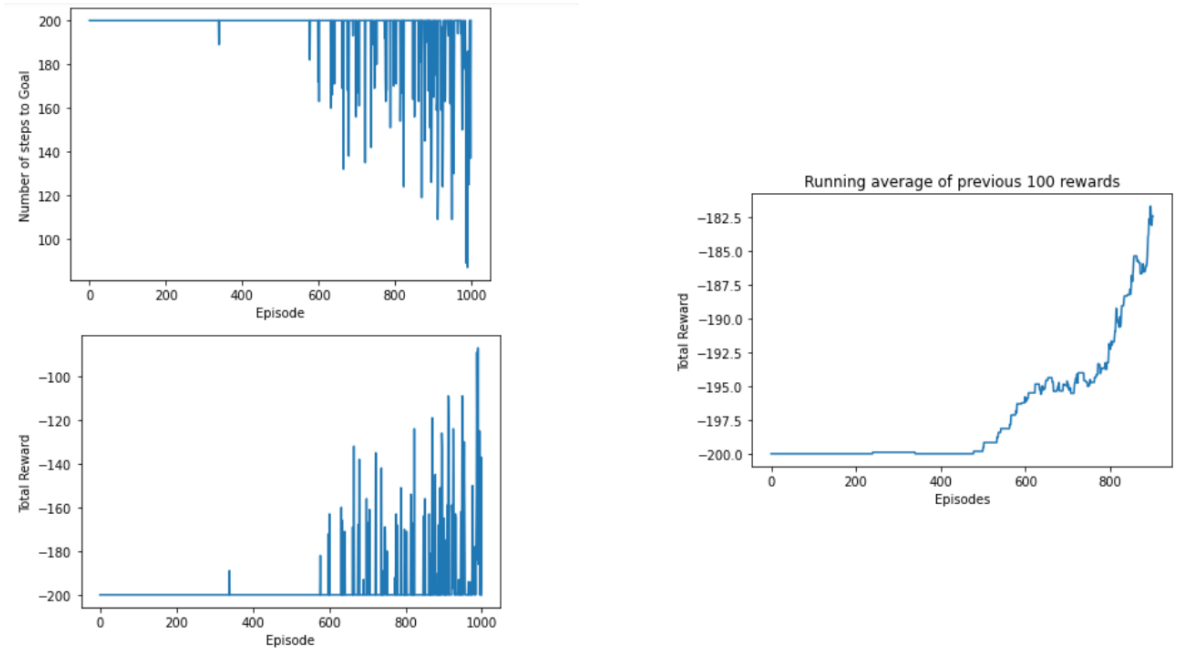


Figure 20: Number of steps in each episode and the total rewards against the episode (left); Running average of past 100 rewards (right)

Size of the Replay buffer: 100000
Batch Size: 96
Discount factor: 0.90
Learning rate: 0.0002
Update frequency of target Q-network: 50
Number of episodes: 1000
Gradient truncation: [-2, 2]
Number of neurons in the first and second layer of the Q-network: [256, 256]
Epsilon decay: 0.995

With an epsilon decay of 0.995, the value of epsilon reaches a very small value at the end of 1000 episodes. Hence, increasing the number of episodes beyond 1000 does not improve the network because the network would not be exploring much at that stage. With this setting, we find the area under the curve (AUC) to be **-177469.275.** From the reward curve, we can see that the performance of the agent shows instabilities and oscillatory behavior. The maximum reward (running average of past 100 rewards) is about **-182.**

Variation - 11:

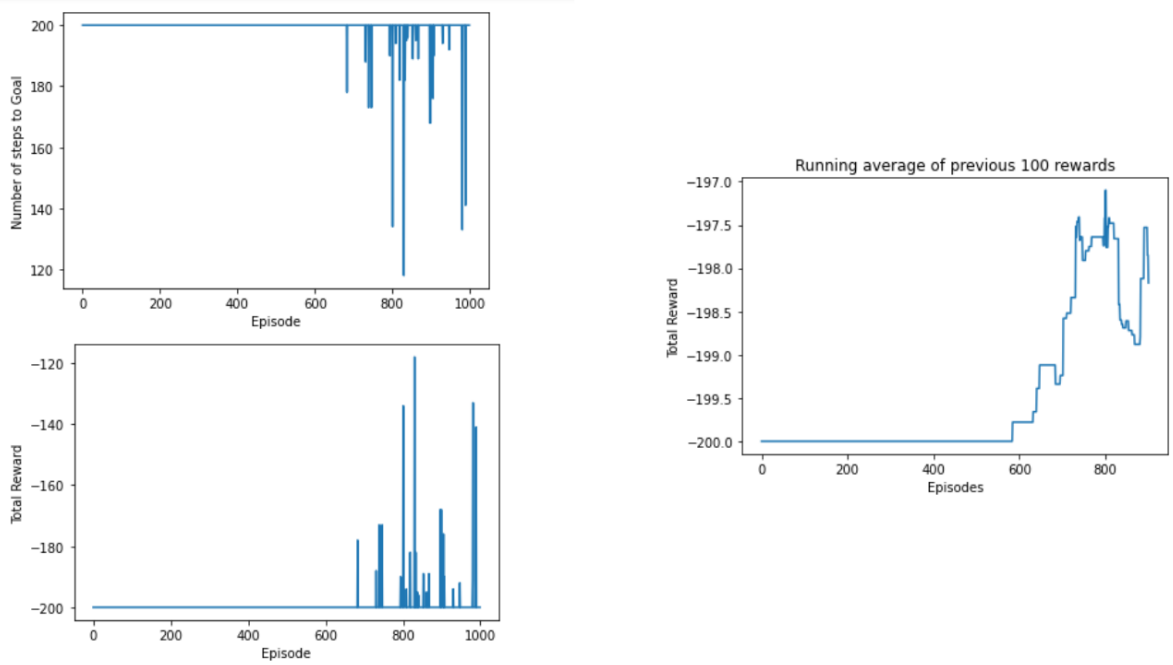


Figure 21: Number of steps in each episode and the total rewards against the episode (left); Running average of past 100 rewards (right)

Size of the Replay buffer: 100000
Batch Size: 96
Discount factor: 0.90
Learning rate: 0.0002
Update frequency of target Q-network: 80
Number of episodes: 1000
Gradient truncation: [-3, 3]
Number of neurons in the first and second layer of the Q-network: [256, 256]
Epsilon decay: 0.995

With an epsilon decay of 0.995, the value of epsilon reaches a very small value at the end of 1000 episodes. Hence, increasing the number of episodes beyond 1000 does not improve the network because the network would not be exploring much at that stage. With this setting, we find the area under the curve (AUC) to be -**179551.35.** From the reward curve, we can see that the performance of the agent shows instabilities and oscillatory behavior. The maximum reward (running average of past 100 rewards) is about **-197.**

Variation - 12:

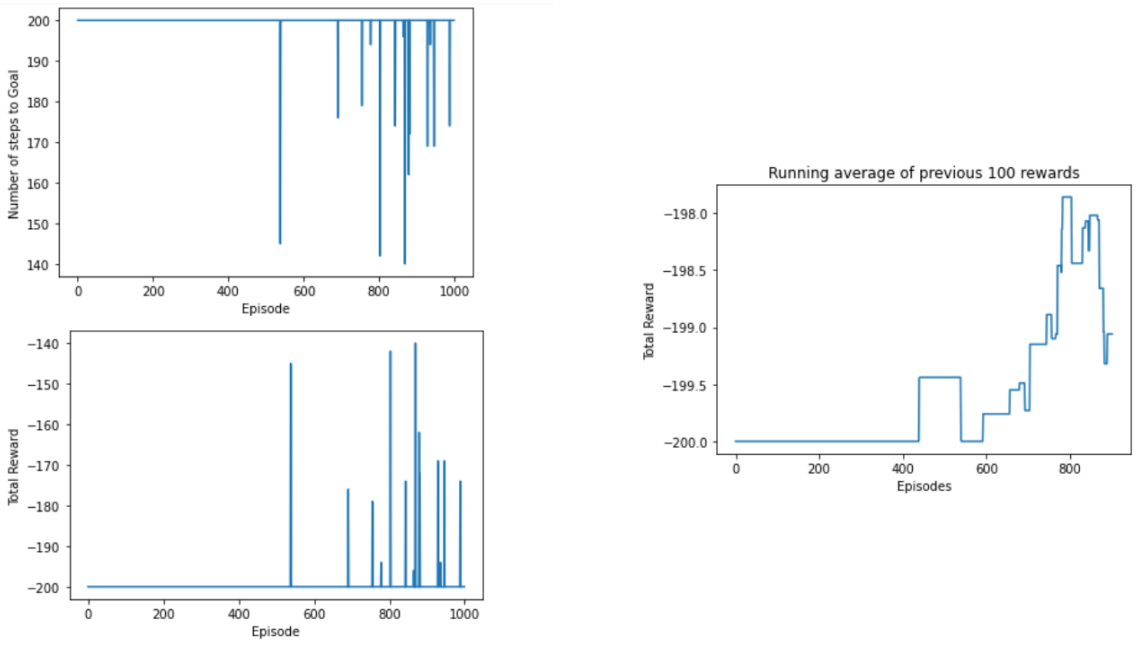


Figure 22: Number of steps in each episode and the total rewards against the episode (left); Running average of past 100 rewards (right)

Size of the Replay buffer: 100000
Batch Size: 64
Discount factor: 0.75
Learning rate: 0.01
Update frequency of target Q-network: 10
Number of episodes: 1000
Gradient truncation: [-1, 1]
Number of neurons in the first and second layer of the Q-network: [128, 64]
Epsilon decay: 0.995

With an epsilon decay of 0.995, the value of epsilon reaches a very small value at the end of 1000 episodes. Hence, increasing the number of episodes beyond 1000 does not improve the network because the network would not be exploring much at that stage. From the reward curve, we can see that the performance of the agent shows instabilities and oscillatory behavior. The maximum reward (running average of past 100 rewards) is about **-197.**

Inference:
1) In general, increasing the complexity of the Q-network enhanced the performance, provided the batch size, learning rate, and gradient truncation parameters are tuned appropriately. Increased complexity allows the network to extract complex features, which results in improved performance.
2) Increasing the batch size along with the model complexity enhanced the performance of the agent. As we increase the batch size, we estimate the gradient with a larger sample, which reduces the variance in the estimate. This resulted in smooth learning.
3) As we increase the model complexity, we need to increase the range of the gradient truncation. If it is narrow for large networks, learning becomes too slow and the network might not converge.

4) In general, increasing the update frequency of the target Q-network resulted in less variance in the reward curves.
5) Changing the discount factor changes the dynamics of the problem. This is because we are trying to estimate the maximum possible value of the expected return (a function of the discount factor) at each and every state. After several experiments, for this environment, a discount factor of 0.9 seems to be appropriate.
6) As we increase the model complexity, we might need to tune the learning rate appropriately. A low value could slow down learning. A very high value could cause instabilities and oscillations during training, preventing the network from converging.
7) By decreasing the value of epsilon decay, we have the luxury to increase the number of episodes. By reducing the value of epsilon decay, we allow the model to explore in more episodes, leading to convergence.

Among all variations, variation 7 was found to be the best, in terms of AUC. In general, all the variations did not learn the optimal policy. This might be due to the less number of episodes and a pretty high value of epsilon decay (which restricts exploration). Without adequate exploration, the agent cannot converge to an optimal policy. There are possibilities that the agent in variation 7 could have learned the optimal policy if trained with a small value of epsilon decay and a large number of episodes.