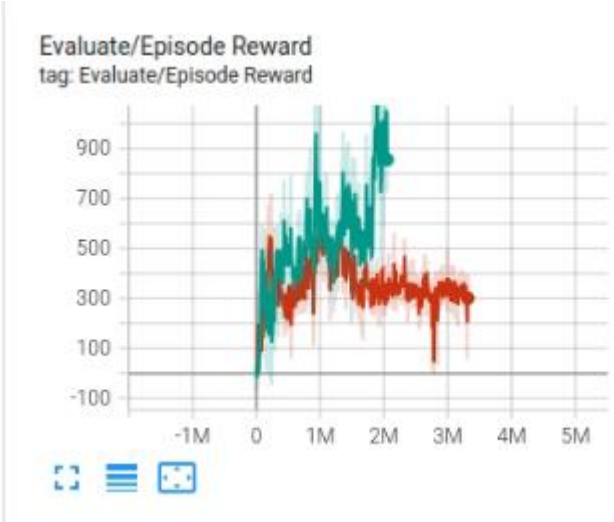
Training curve:

Testing result:

```
Evaluating...
Episode: 1      Length: 508     Total reward: 990.54
Episode: 2      Length: 576     Total reward: 1369.32
Episode: 3      Length: 587     Total reward: 1376.17
Episode: 4      Length: 455     Total reward: 1024.93
Episode: 5      Length: 243     Total reward: 492.36
Episode: 6      Length: 999     Total reward: 1359.37
Episode: 7      Length: 260     Total reward: 472.19
Episode: 8      Length: 425     Total reward: 879.56
Episode: 9      Length: 607     Total reward: 1431.55
Episode: 10     Length: 512     Total reward: 1442.58
average score: 1083.857473438253
```

Bonus 1:



Performance Differences:

1.More stable learning curves

2.Better final performance

3.Less susceptibility to overestimation

4.More conservative value estimates

single Q-network might show:

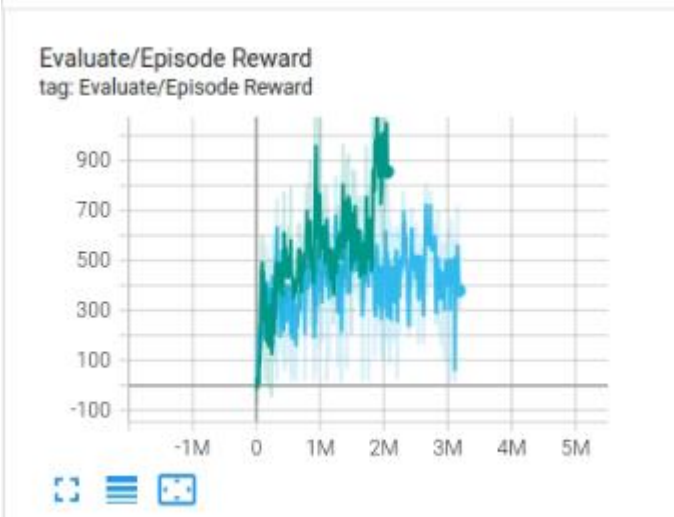
1.Faster initial learning in some cases

2.More variance in performance

3.Higher likelihood of overestimation

4.Less stable learning curves

Bonus 2:



Advantages:

1.Reduces variance in target Q-values

2.Prevents overfitting to narrow peaks in the value function

3.Improves robustness to function approximation errors

4.Creates a smoother policy that's less likely to exploit Q-function errors

With Smoothing:

- More stable learning curves
- Better final performance in complex environments
- More robust to hyperparameter choices
- Better generalization

Without Smoothing:

- Potentially faster learning in simple environments
- More precise policies in some cases
- Higher variance in performance
- More susceptible to Q-function approximation errors

Bonus 3:

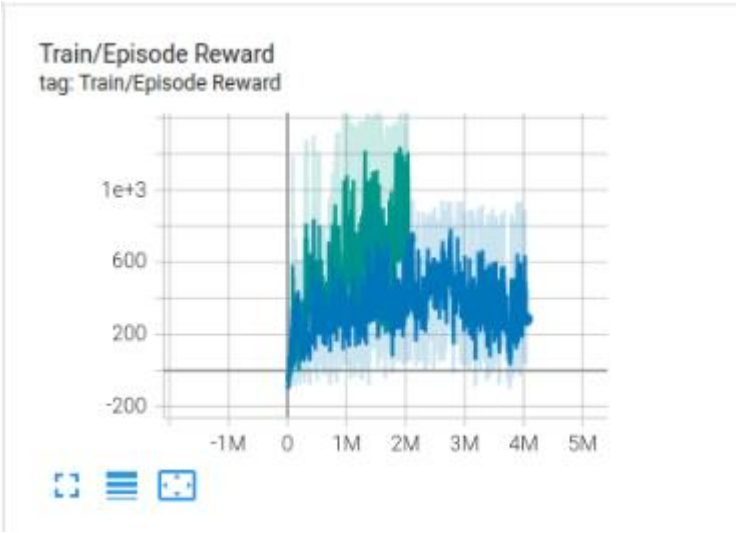
No Delay:

1. Faster initial learning possible
2. Higher variance in performance
3. Risk of premature policy updates
4. Less stable Q-value estimates

Standard Delay:

1. Balanced learning speed and stability
2. Better convergence properties
3. More reliable Q-value estimates
4. Reduced policy variance

Bonus 5:



Original reward function: actual reward of the game.

My reward function:

```
def completion_focused(self, reward):
    if reward > 0:  # Track tile visited
        return reward * 1.5  # Increase reward for staying on track
    return reward * 0.8  # Reduce time penalty
def createVideo(self, source, fps, output_name):
```

Make agent stay more on track and reduce the penalty to make more explorations.