



A Lightweight  
Benchmark

# *Logging Farm*

Zekun Song



# *Content*

*01*

Background of  
Benchmark

*02*

Pipeline of  
Implementation

*03*

Results  
Demonstration

*04*

Summary

# Implementation

## Version 1

State Space: 10x10 grid, value of tree

Action Space: Cut trees with growing  
state 1 to 7, or not cut

Reward: [0, 1, 4, 9, 16, 25, 36, 49]

## Version 2

State Space: 10x10 grid, value of tree

Action Space: Cut trees with growing  
state 1 to 7, or not cut

Reward: [0, 1, 4, 9, 16, 25, 36, 49],

0 if not surpass threshold

## Version 3

State Space: 10x10 grid, value of tree

Action Space: Cut trees with growing  
state 1 to 7, or not cut

Reward: [0, 1, 4, 9, 16, 25, 36, 49] +  
[0, 5, 10, 15, 20, 25, 30]

## Version 4

State Space: 10x10 grid, value of tree  
fertility of soil

Action Space: Cut trees with growing  
state 1 to 7, or not cut

Reward:  $0.5\pi(\text{growing state})^2$

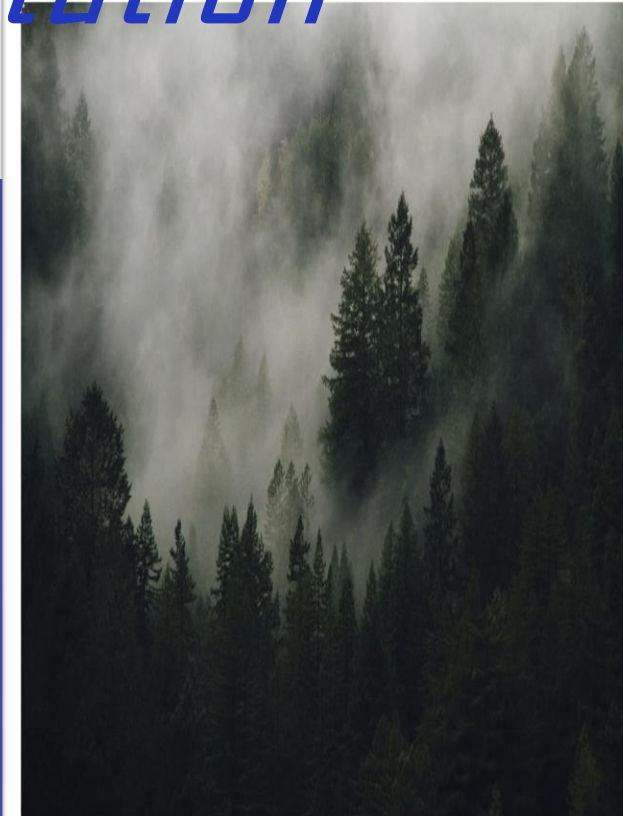
## Version 5

State Space: 10x10 grid, value of tree  
fertility of soil

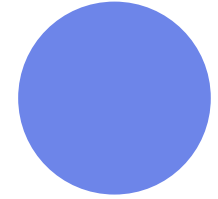
Action Space: Cut trees with growing  
state 1 to 7, or not cut

Reward:

$0.5\pi(\text{growing state})^2 + 0.05(\text{CO}_2 \text{ absorbency})$



# Graphic and Interaction of Environment



## Basic Control of Game



Arrow keys control the movement of yellow frame.

Key "n" to move to next year.

Key "0" to "7" cut trees with corresponding growing state.





---

# Impact of Seeds on Training

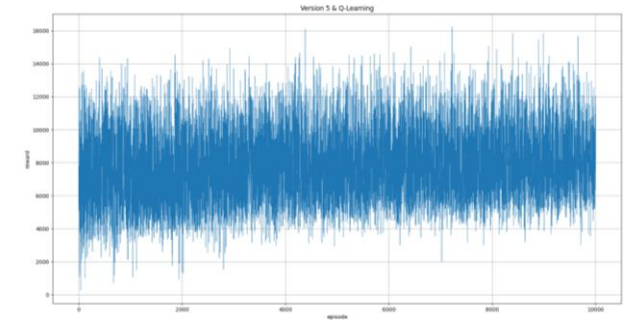
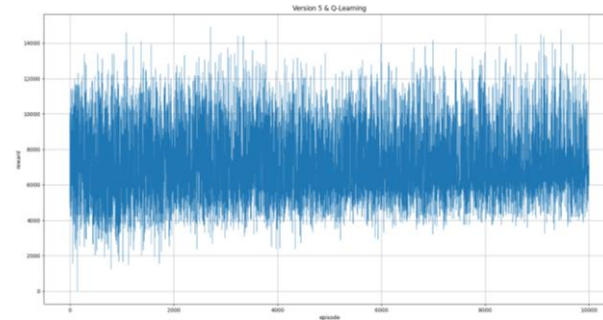
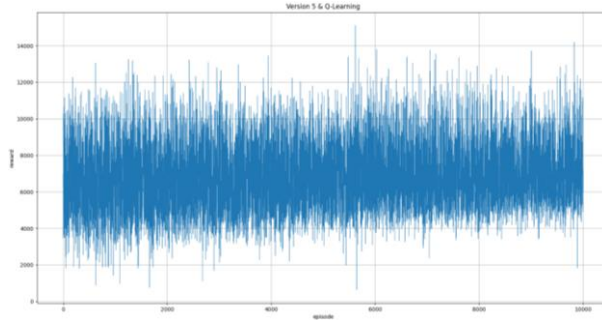
# Version 1

Seed 0

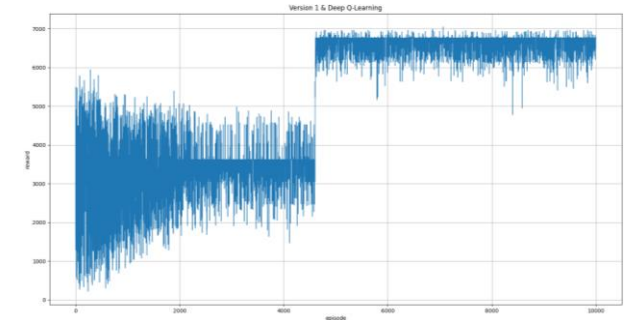
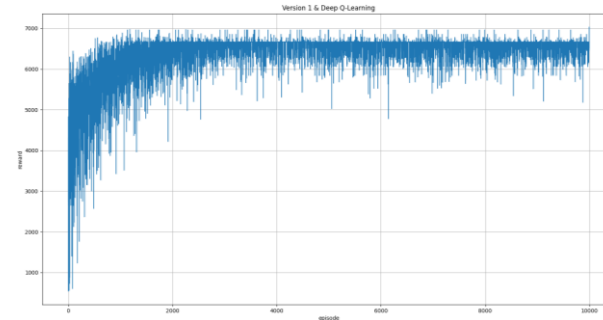
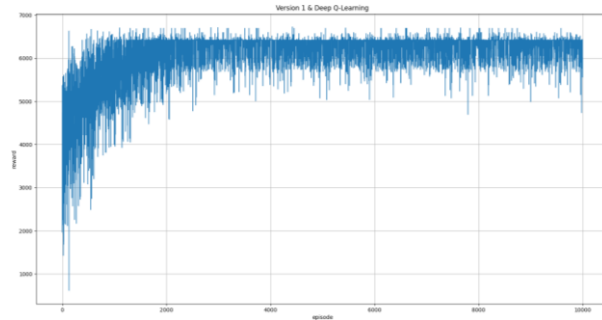
Seed 1

Seed 2

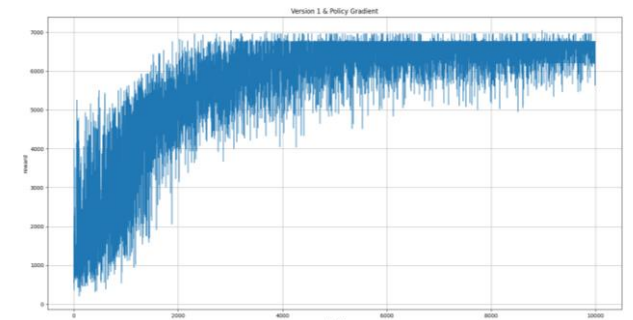
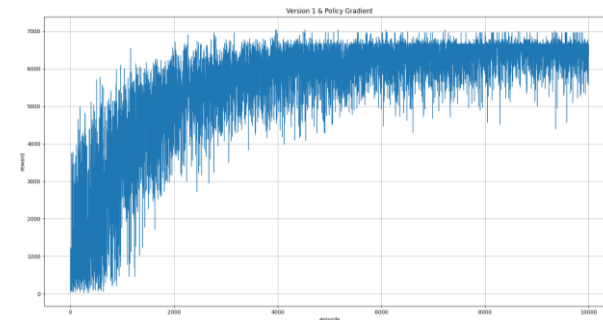
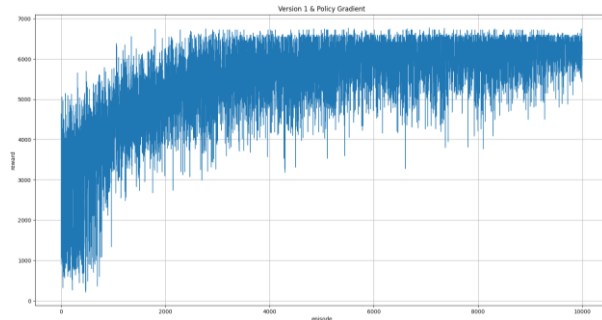
Q-Learning



Deep Q-Learning



Policy Gradient



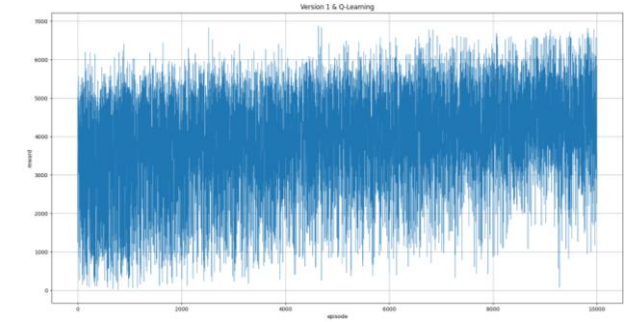
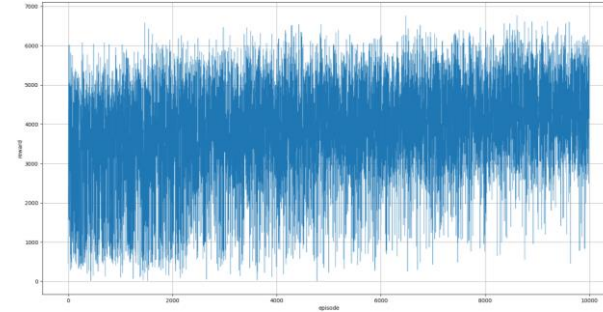
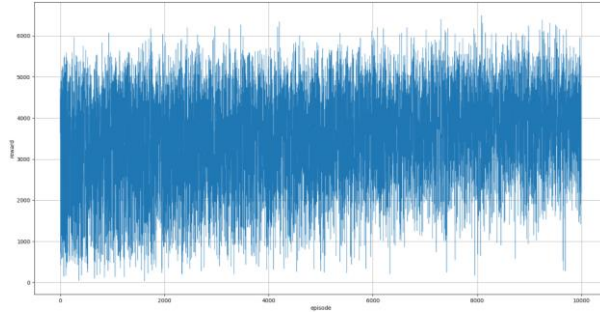
# Version 5

Seed 0

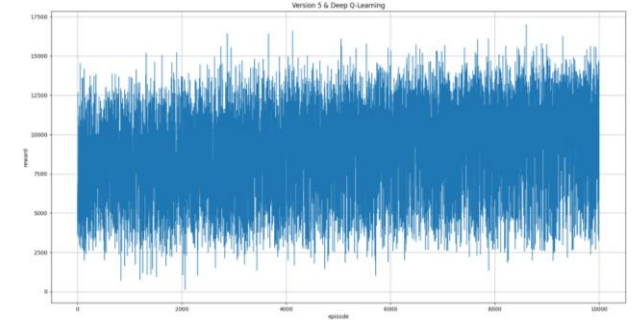
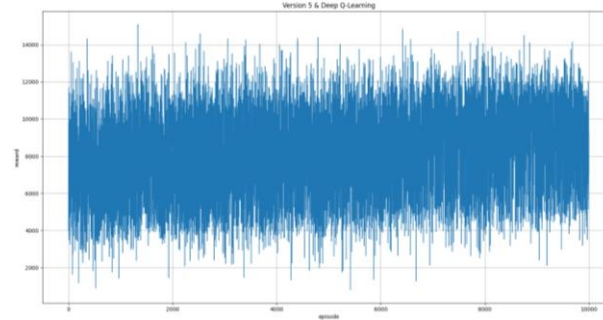
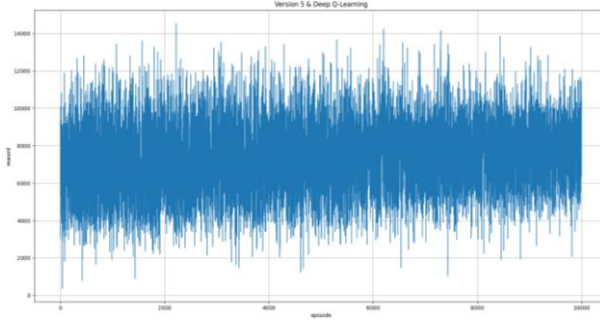
Seed 1

Seed 2

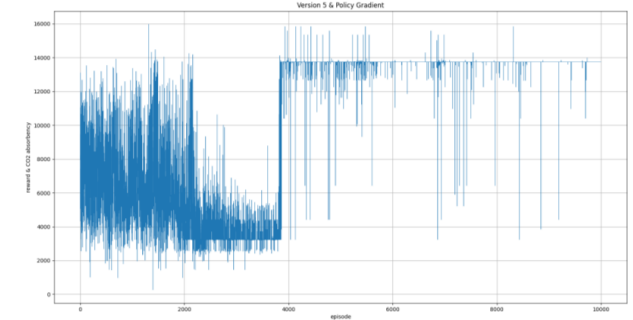
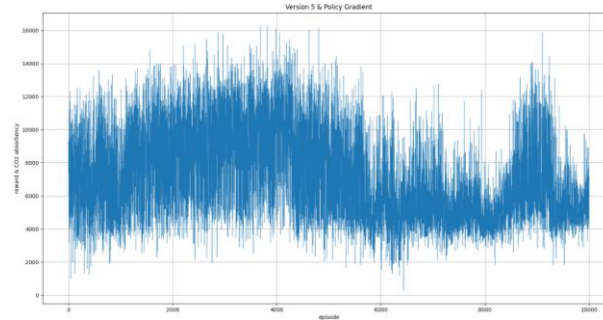
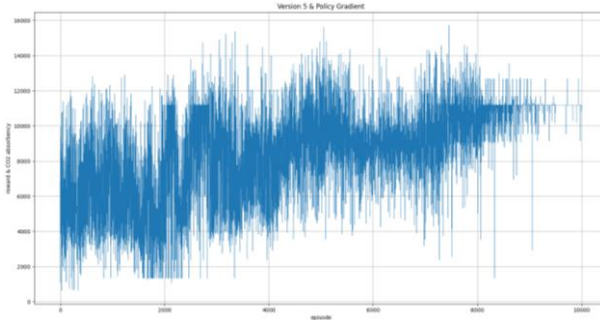
Q-Learning



Deep Q-Learning



Policy Gradient







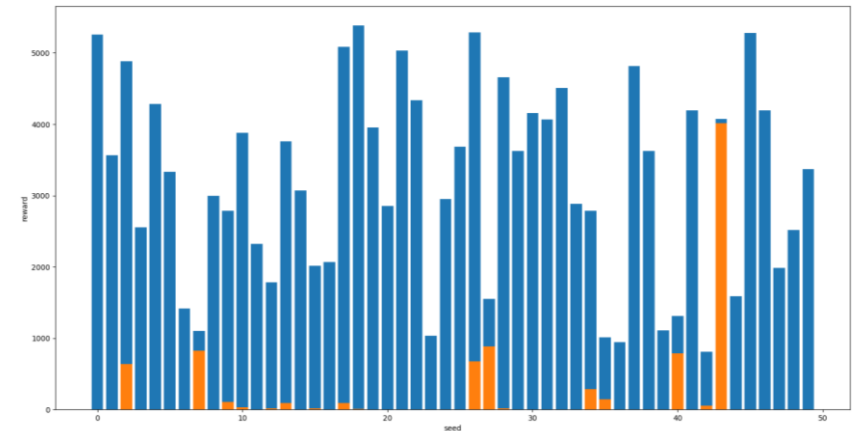
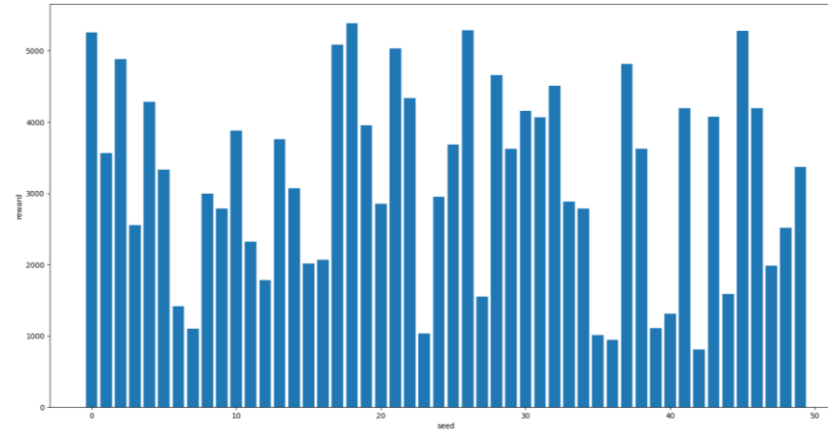
---

# Evaluation Results

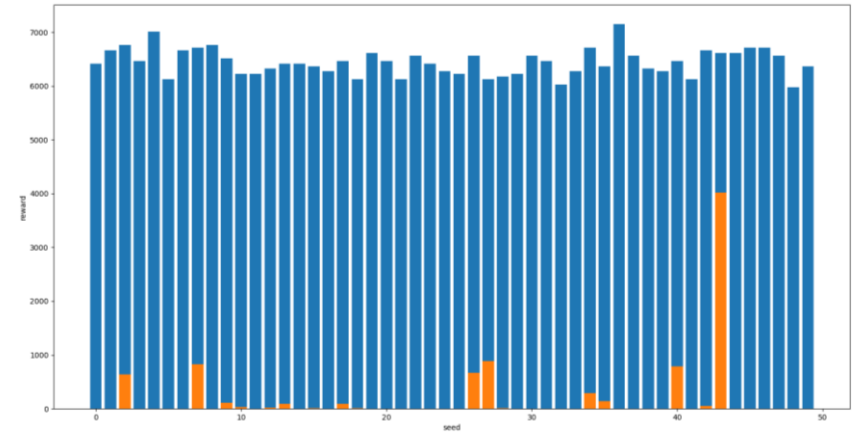
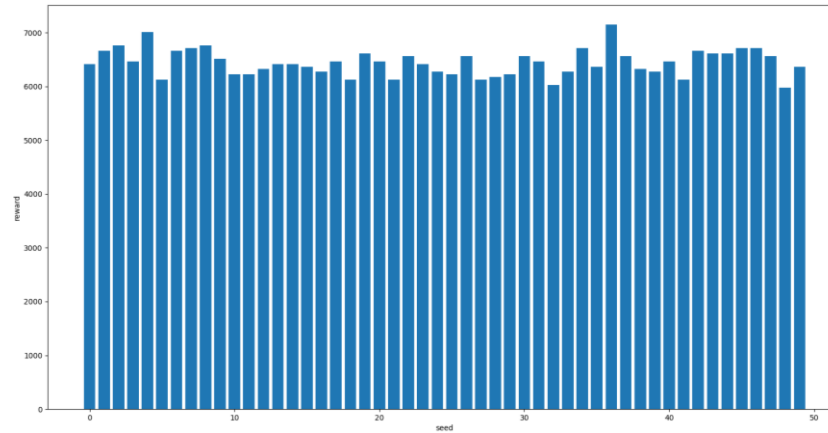


# Version 1

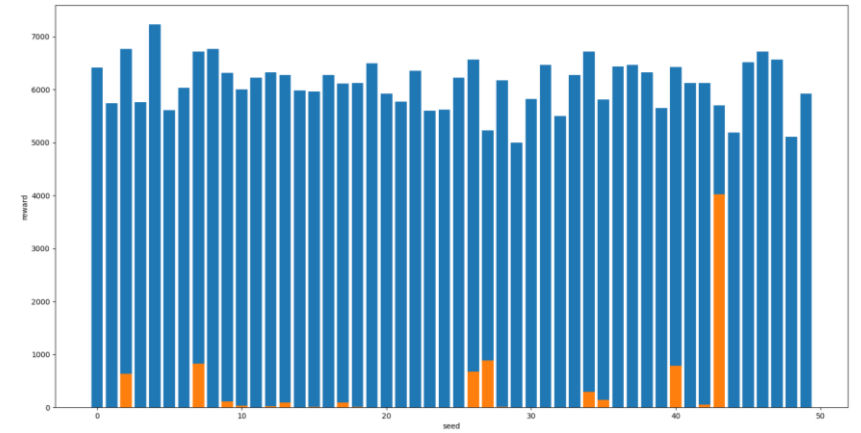
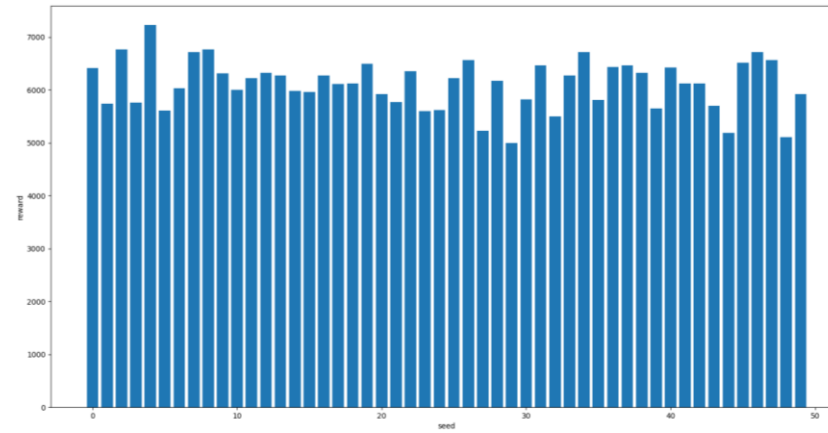
Q-Learning



Deep Q-Learning

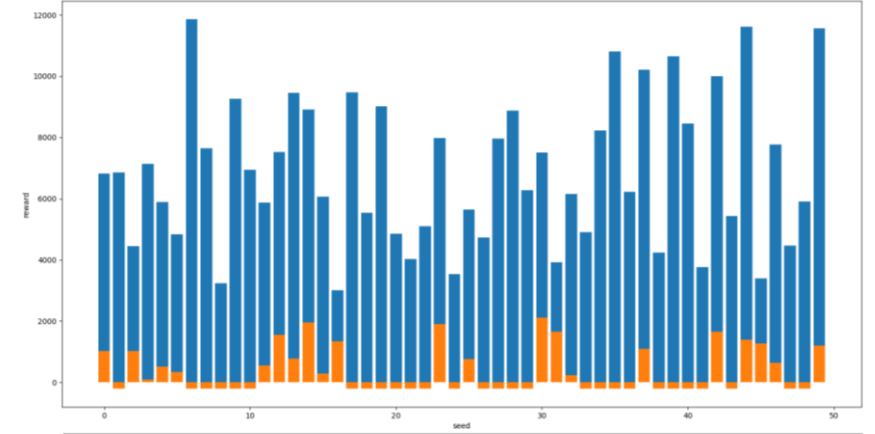
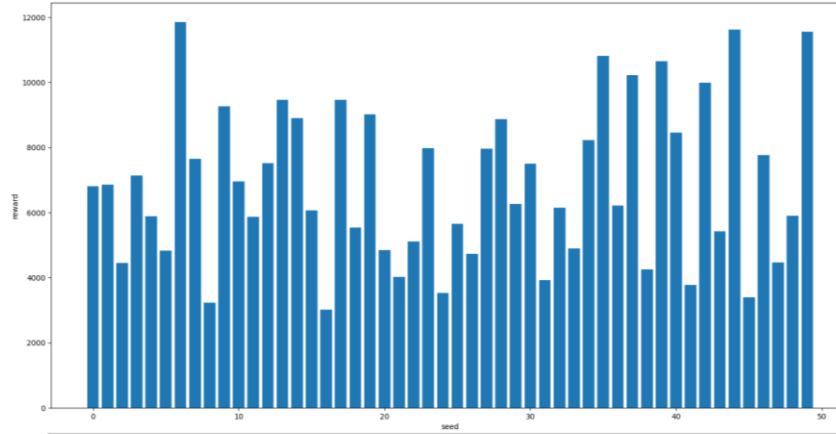


Policy Gradient

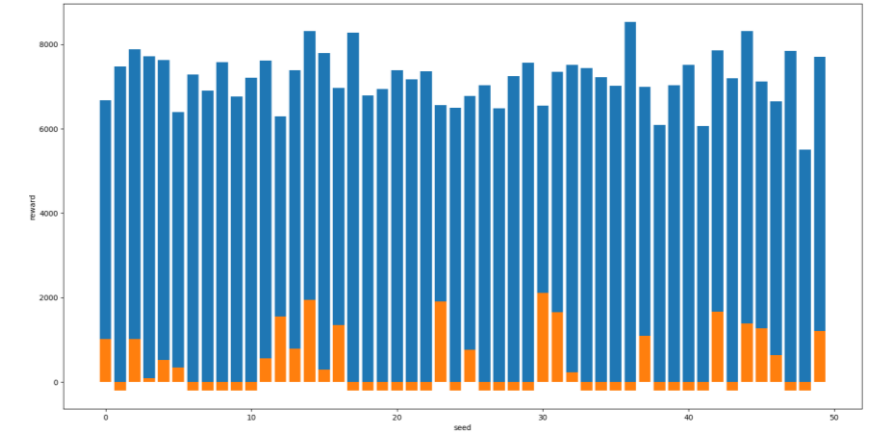
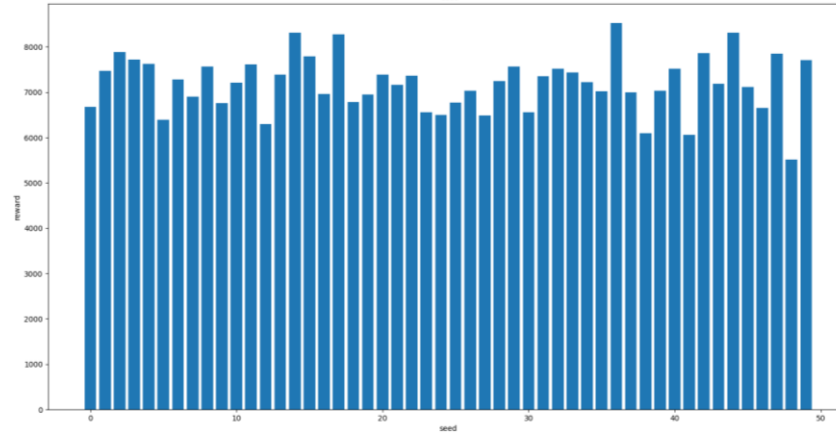


# Version 5

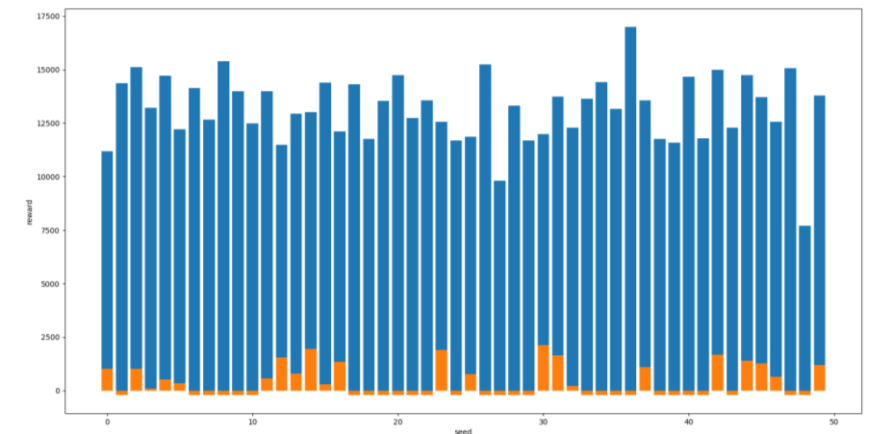
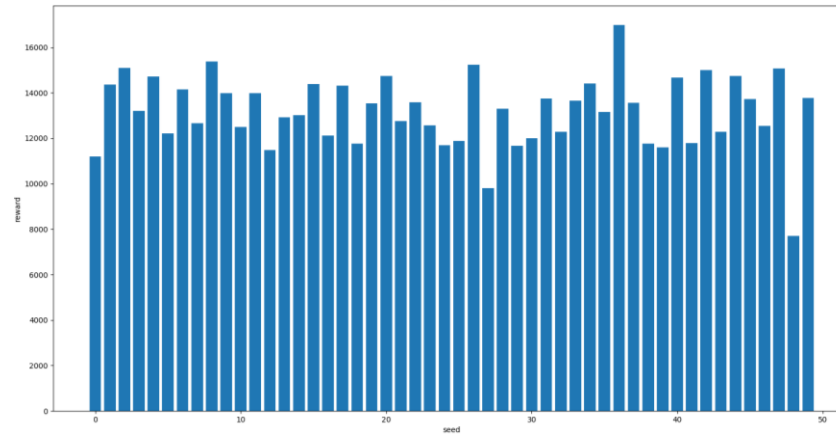
Q-Learning



Deep Q-Learning



Policy Gradient



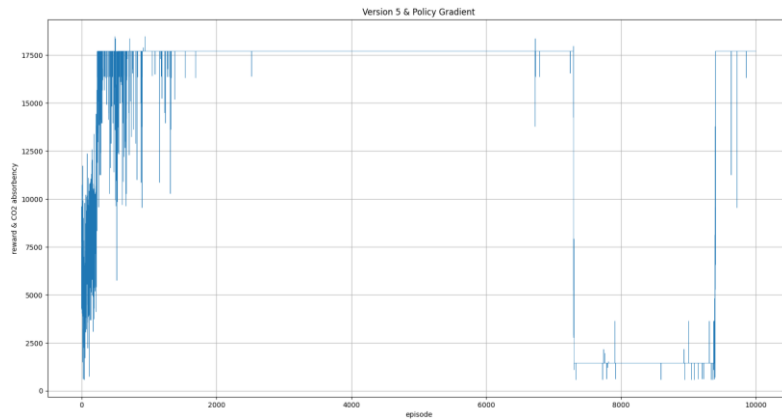


# Impact of Hyperparameters

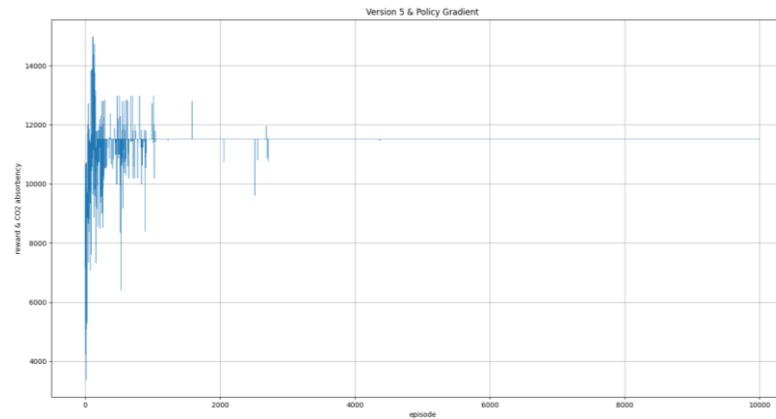
# Version 5

## Different Hyperparameters settings of Policy Gradient

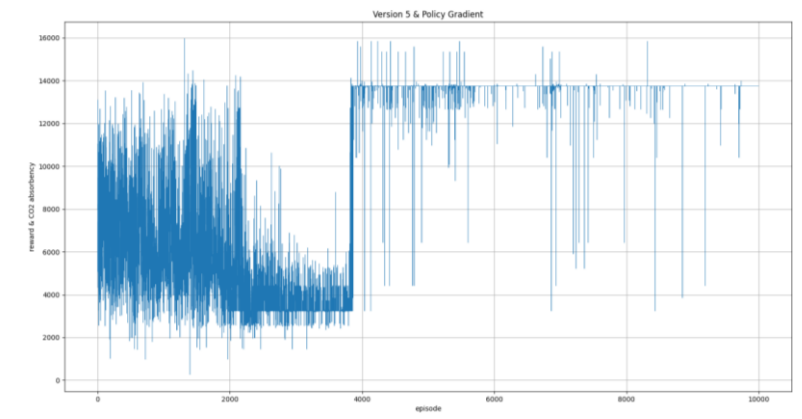
**Hyperparameters:** Learning rate:  $1e-4$ , **Discount factor:** 1.0, **Batch size:** 128, **Hidden layer:** 256x512 sigmoid 512x64, **Optimizer:** Adam



**Hyperparameters:** Learning rate:  $1e-4$ , **Discount factor:** 1.0, **Batch size:** 128, **Hidden layer:** 256x1024 sigmoid 1024x256, **Optimizer:** Adam



**Hyperparameters:** Learning rate:  $1e-4$ , **Discount factor:** 1.0, **Batch size:** 128, **Hidden layer:** 256x64 sigmoid 64x256, **Optimizer:** Adam





*From the whole building progress of this benchmark we clearly noticed how the complexity of benchmark influences the performance of RL algorithms. And we learned the importance of selecting appropriate RL algorithm for specific environment. Though Gradient Policy generally is powerful than simple RL algorithms e.g. Q-Learning, it still could perform worse if we do not configure it appropriately. Last but not least, our benchmark still has much space to improve, for example, it can be added more factors of nature environment, such as sunlight, temperature etc. Though our benchmark is quite lightweight, it's still pretty challengeable for state-of-the-art RL algorithms.*

# Summary

