SOLVING TAXI-V3

INTRODUCTION

Observation Space

500 discrete states:

- 25 taxi positions
- 5 possible locations of the passenger (including when he is in the taxi)
- 4 destination locations

Action Space

Discrete 6 actions

Reward

- 1 per step unless other reward is triggered.

+ 20 delivering passenger.

-10 executing “pickup” and “drop-off” actions illegally.

Q-LEARNING

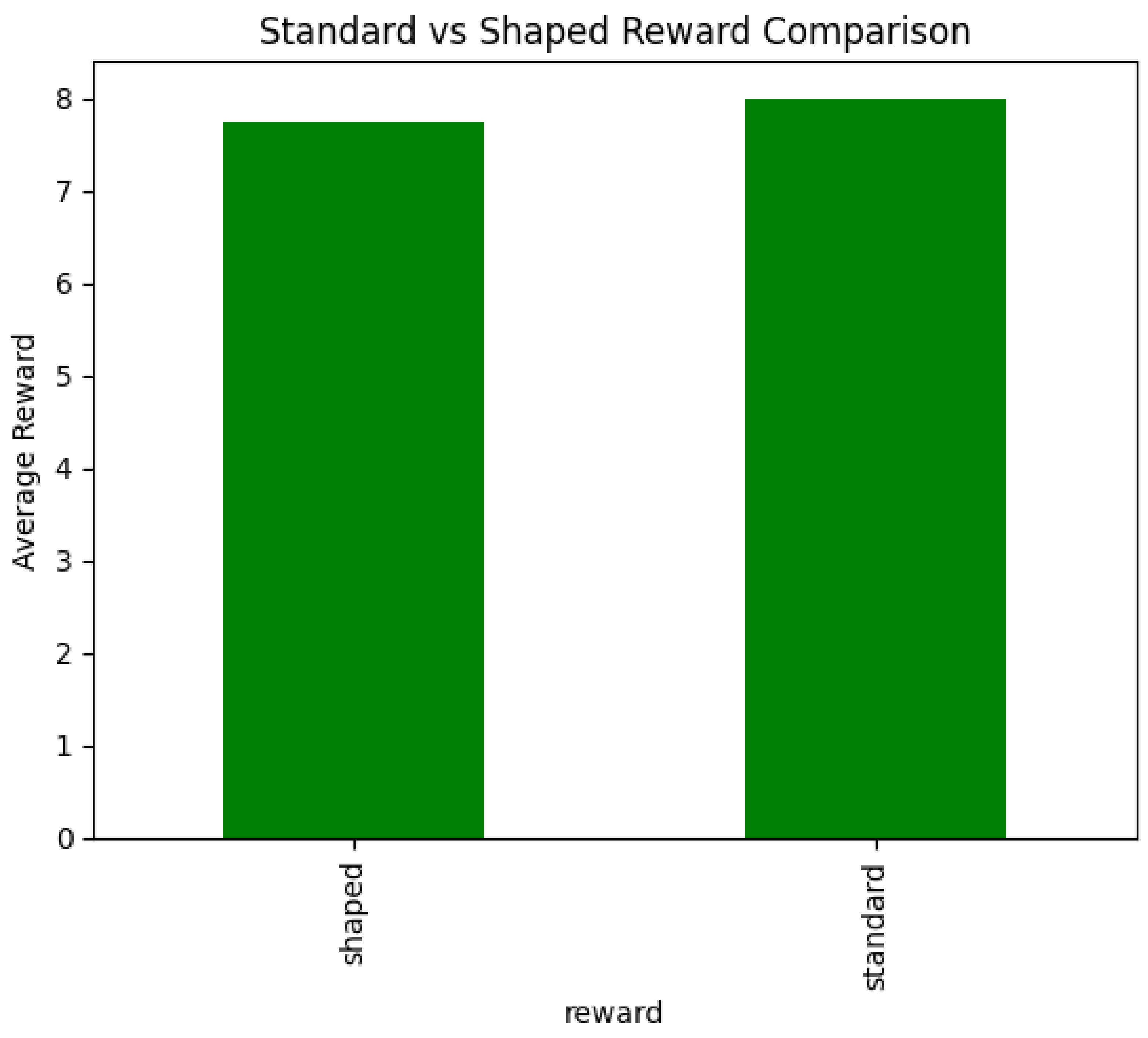
- Model-free, off-policy RL method
- Learns action-value function Q(s, a)
- Uses Bellman update rule

$$Q(s,a) \leftarrow Q(s,a)+\alpha[r+\gamma a \max Q(s',a)-Q(s,a)]$$

- The Q-table stores the expected cumulative reward for each state–action pair.

REWARD SHAPING

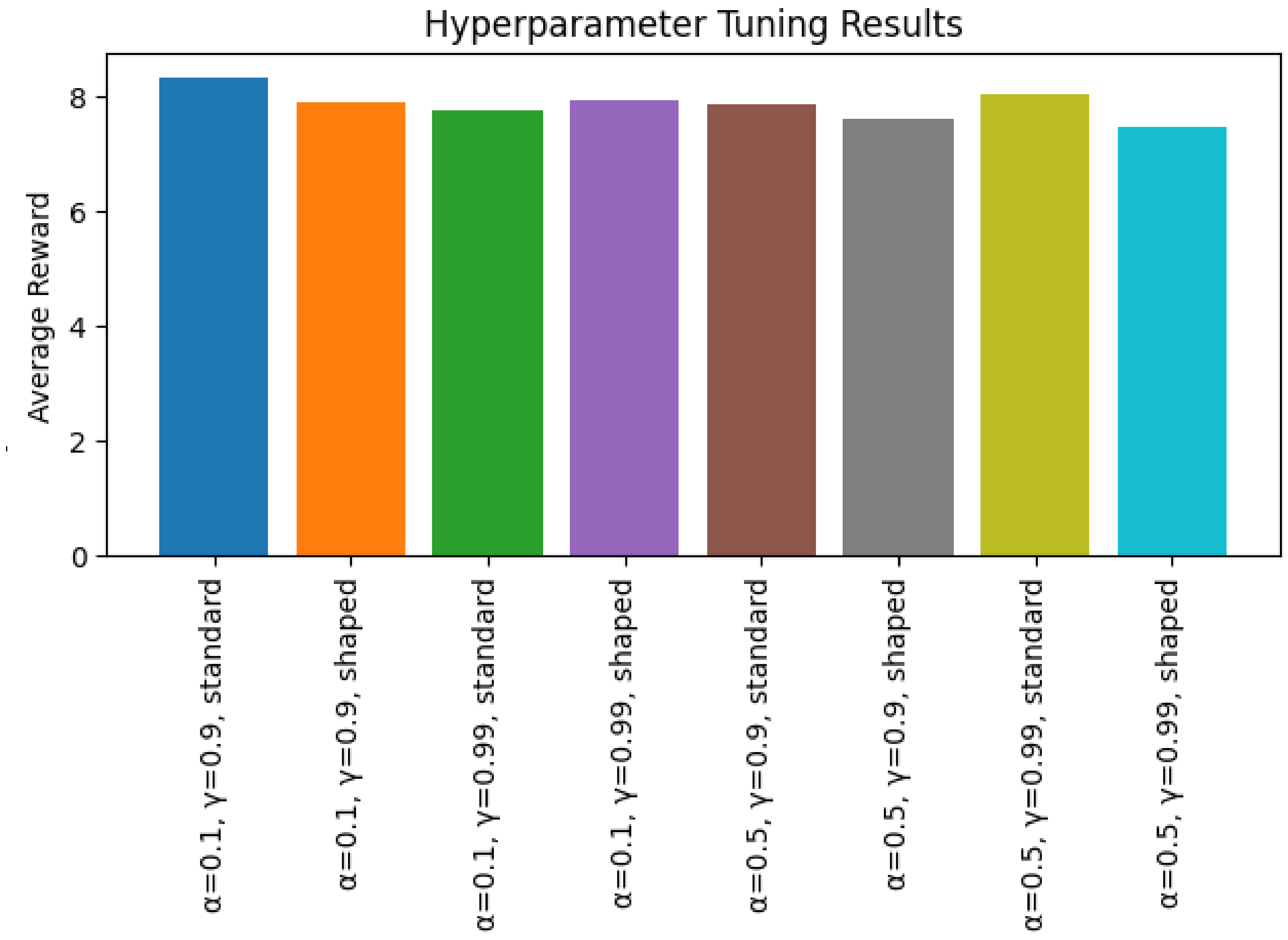
	alpha	gamma	reward	avg_reward
0	0.1	0.90	standard	8.33
6	0.5	0.99	standard	8.03
3	0.1	0.99	shaped	7.95
1	0.1	0.90	shaped	7.91
4	0.5	0.90	standard	7.87
2	0.1	0.99	standard	7.75
5	0.5	0.90	shaped	7.63
7	0.5	0.99	shaped	7.46



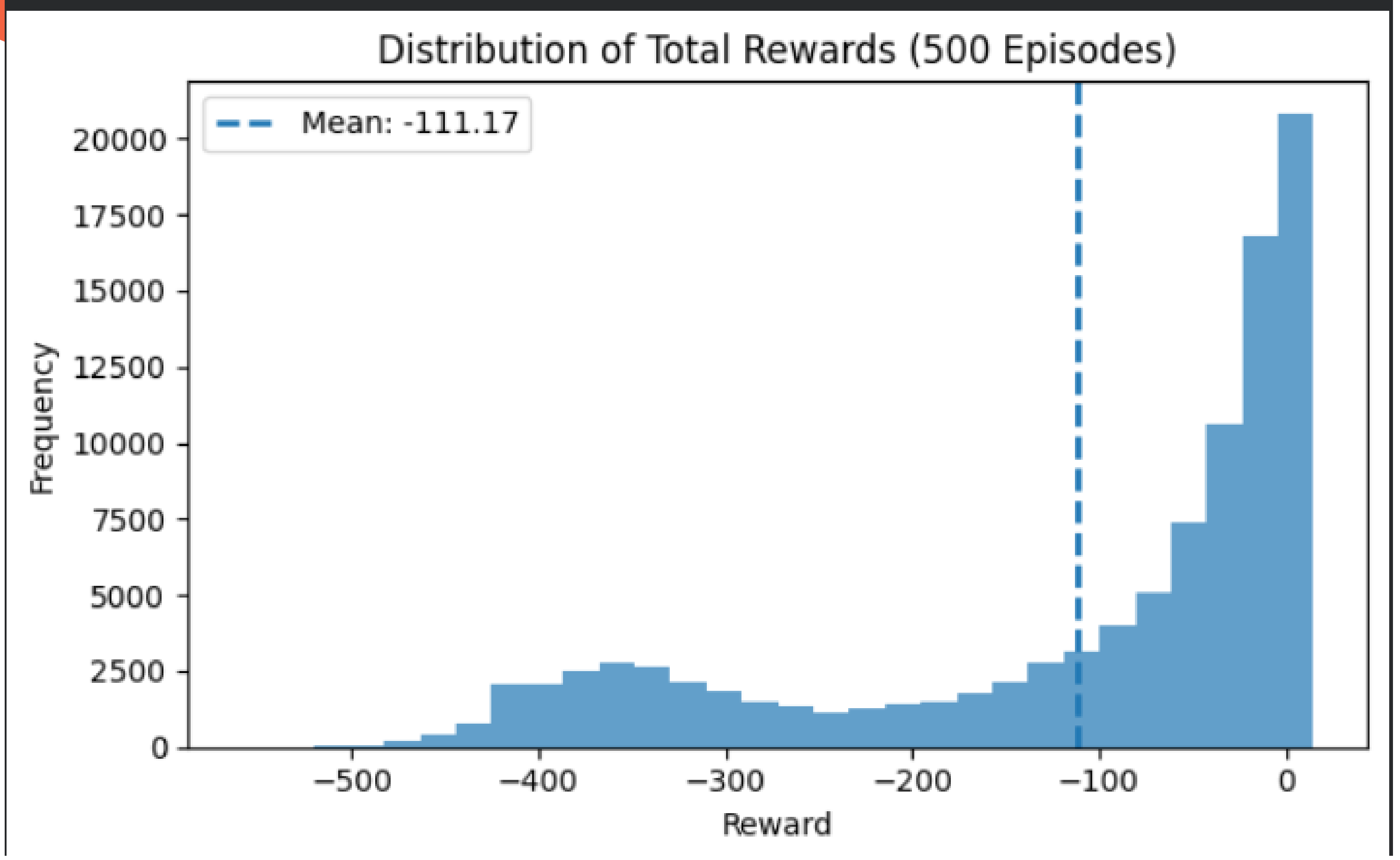
Q-LEARNING

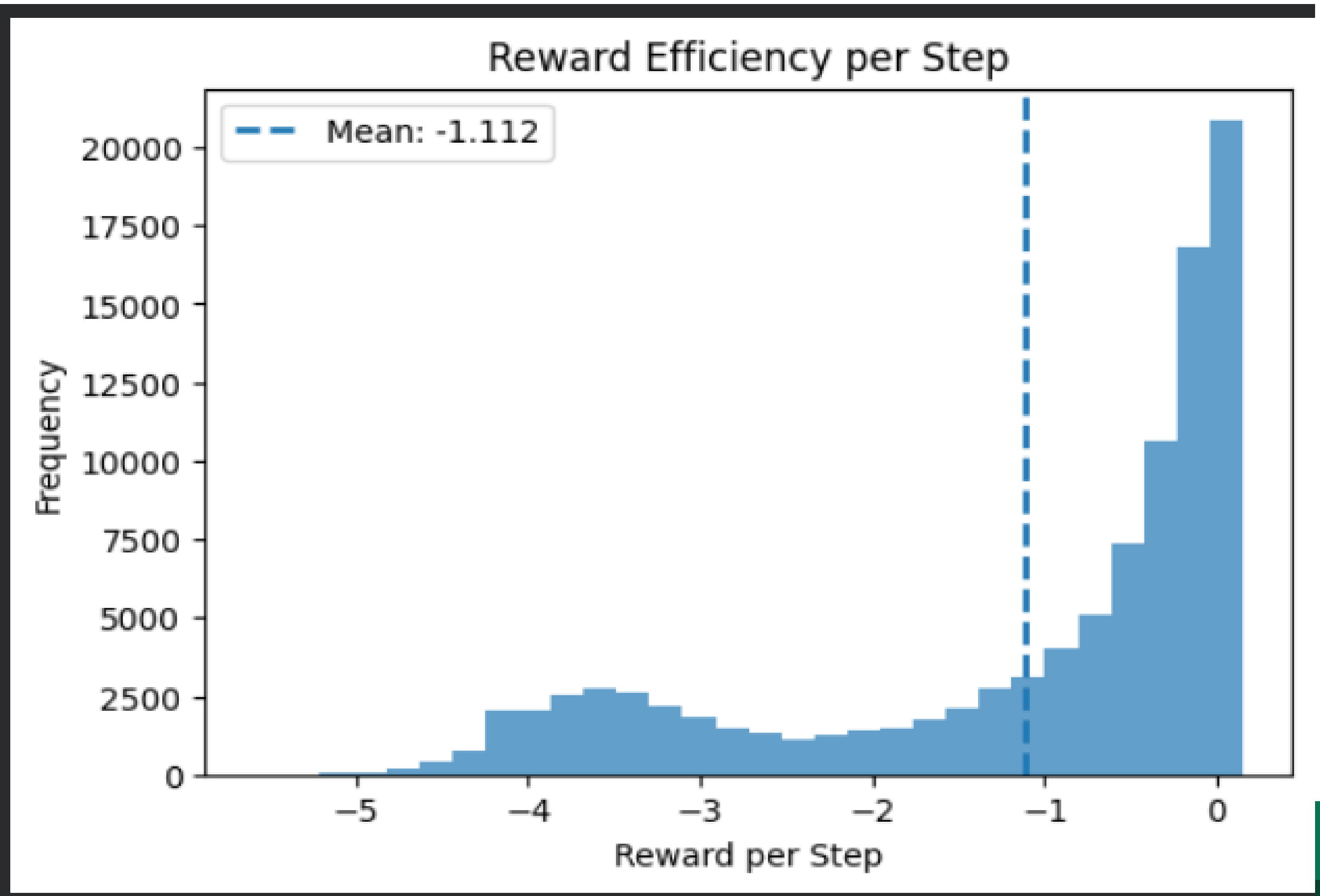
- The policy is $\pi(s) = \text{argmax}_a\ Q(s,a)$
- ε-Greedy Exploration
- With probability $\varepsilon \to$ random action
- With probability $1 - \varepsilon \to$ greedy action
- ε decays over time

Parameter	Value
Learning rate (α)	0.1
Discount factor (γ)	0.9
Initial ε	1.0
ε decay	0.995
Episodes	100,000



FINAL EVALUATION






```
Median return: -47.00
Standard deviation: 135.44
Minimum return: -559.00
Maximum return: 15.00
```

STOCHASTIC POLICY GRADIENT

Algorithm

REINFORCE (Monte Carlo Policy Gradient)

Neural Network

Input layer: 500-unit vector

Hidden Layers: two fully connecte layers (128 neurons each)

Output layer: 6-unit Softmax layer

Enhanced reward

$$R'(s_t, a_t, s_{t+1}) = \ r_t + F(s_t, s_{t+1})$$

$$dist = |x_{taxi} - x_{target}| + |y_{taxi} - y_{target}|$$

$$F(s_t) = -(dist(s_t, target) * \omega)$$

PARAMETERS TUNING RESULTS

Method

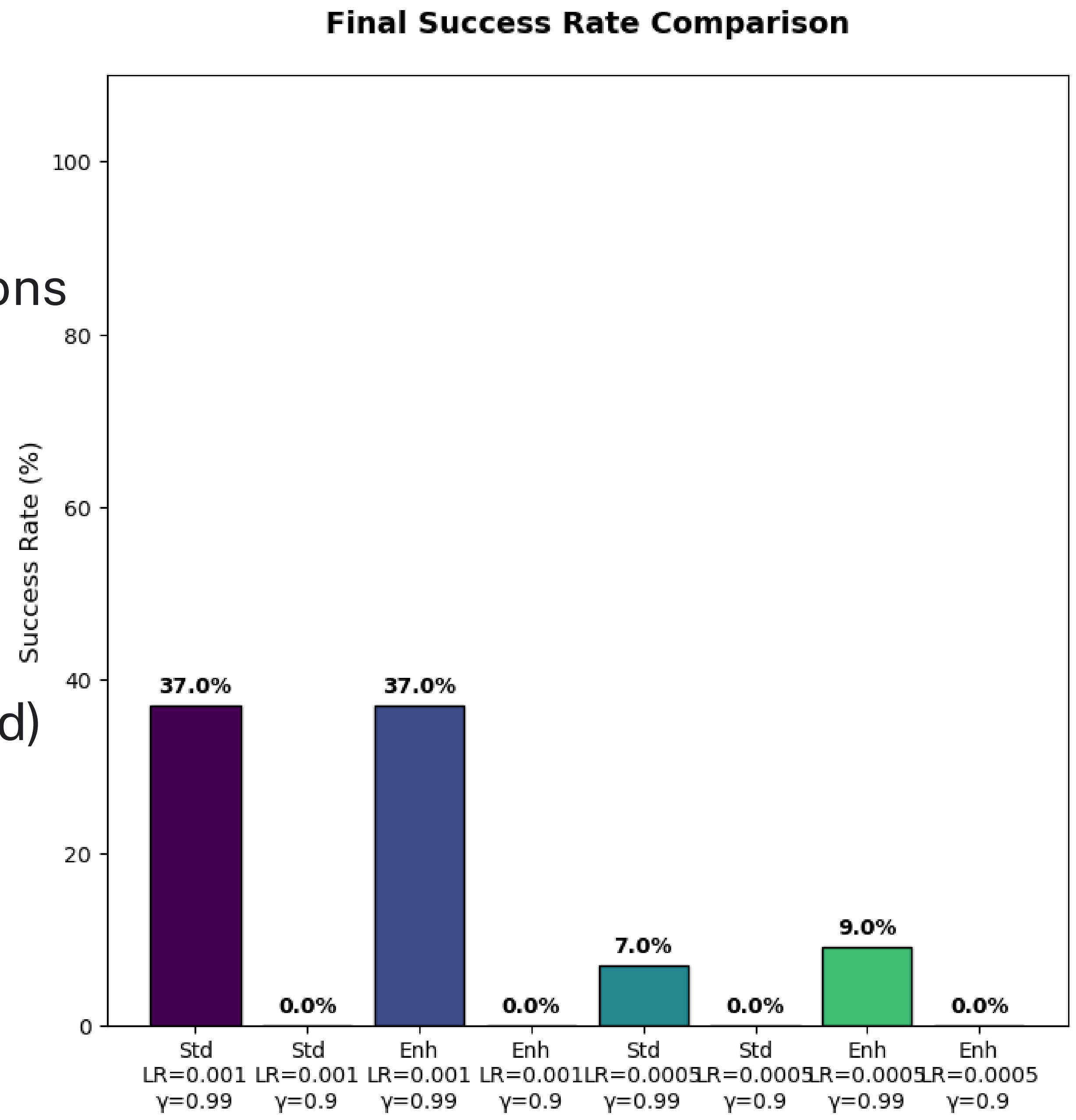
Grid Search over 8 distinct combinations

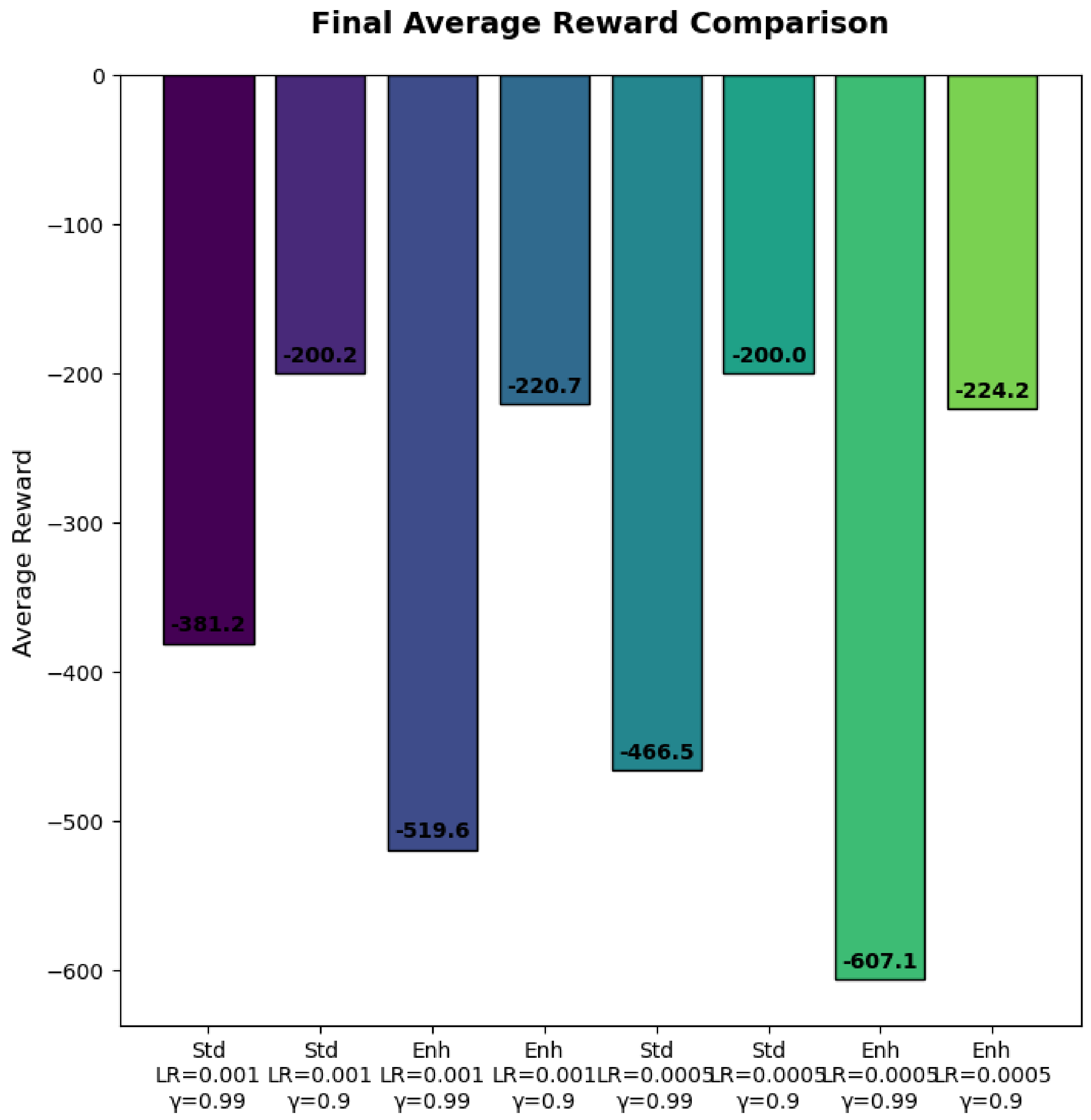
Parameters

- Learning Rate (0.001 vs. 0.0005)
- Discount Factor (0.99 vs. 0.90)
- Reward Models (Standard vs. Enhanced)

Number of episodes

from 0 to 5000 with 500 step

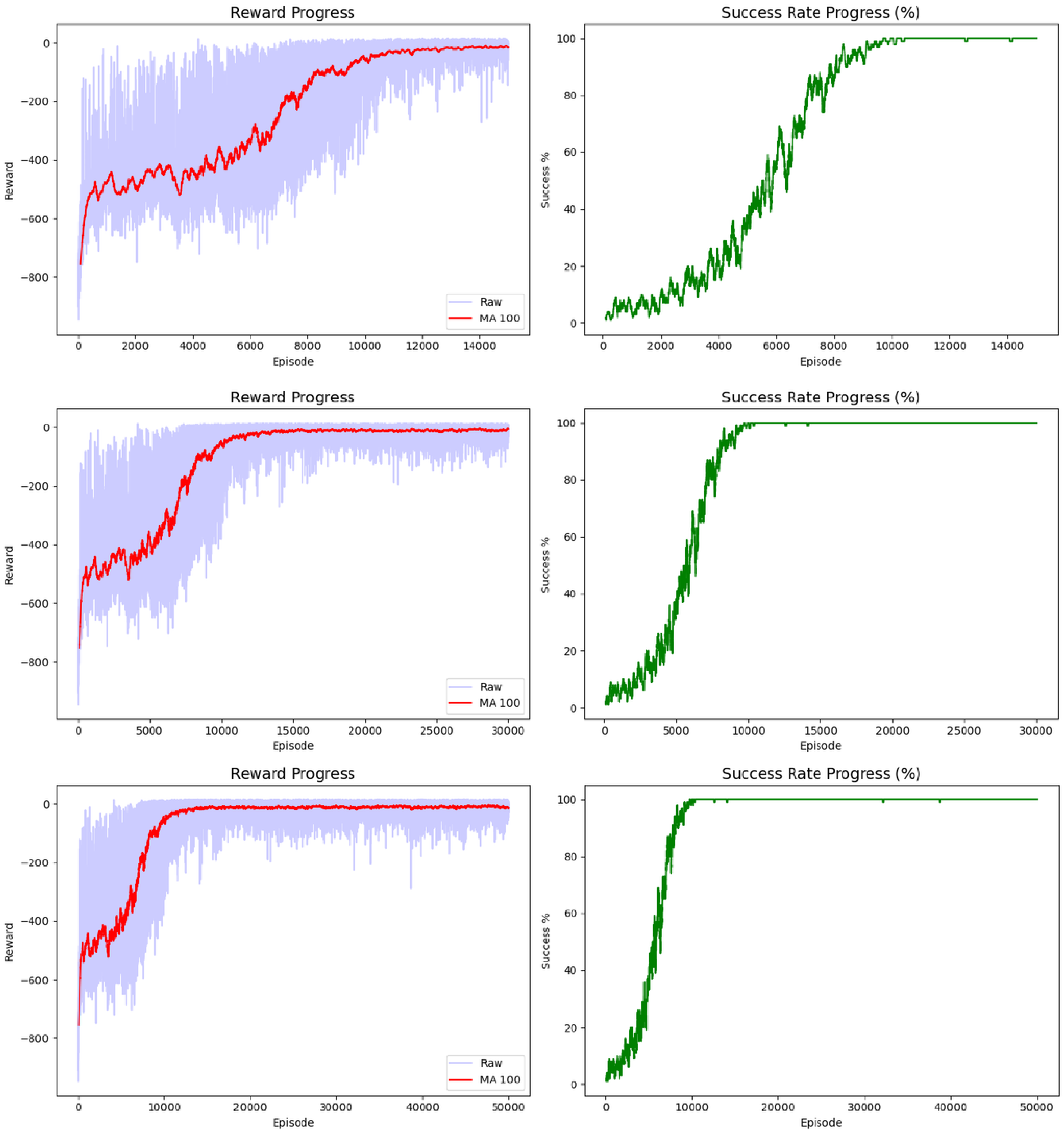




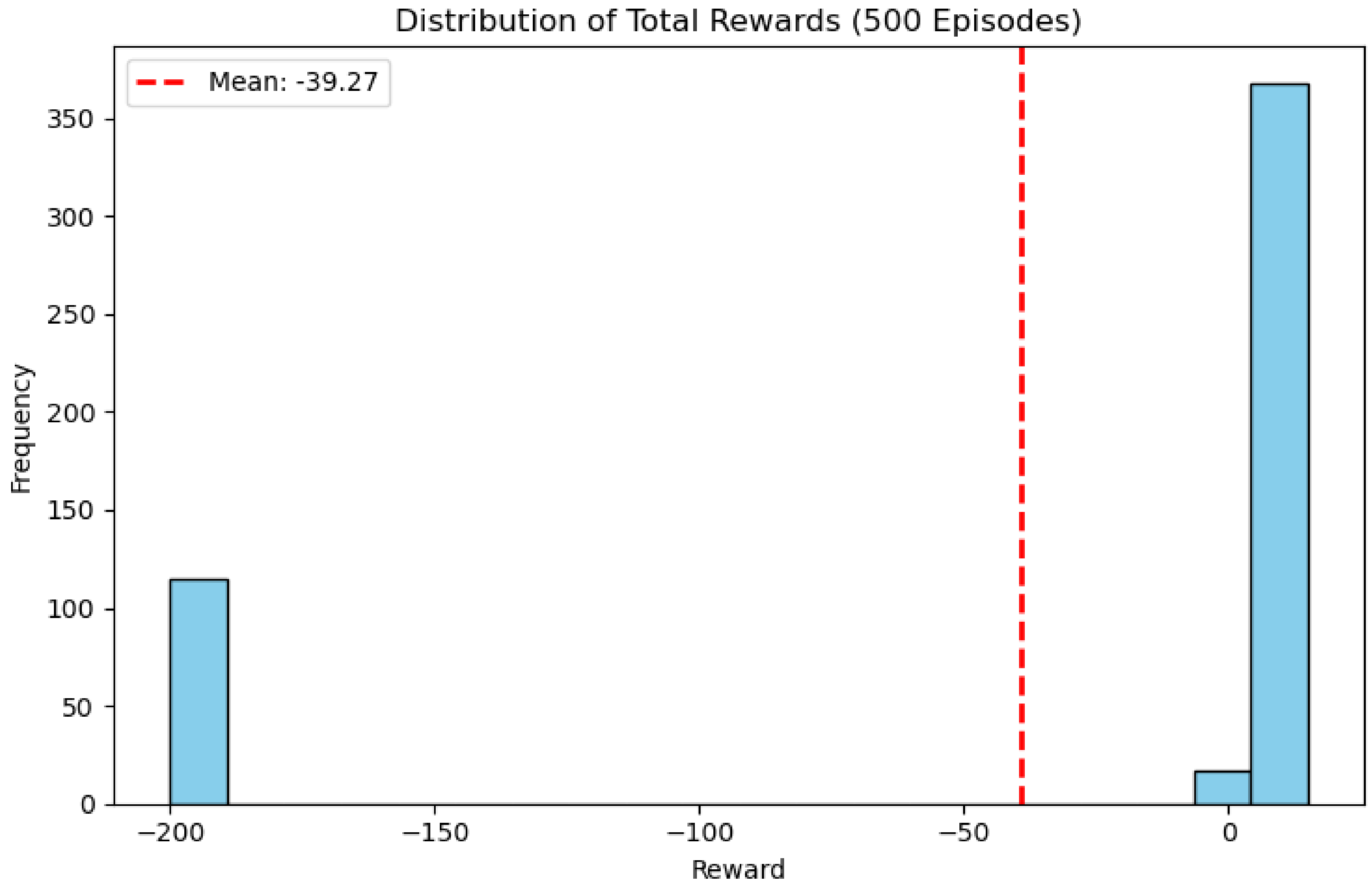
.

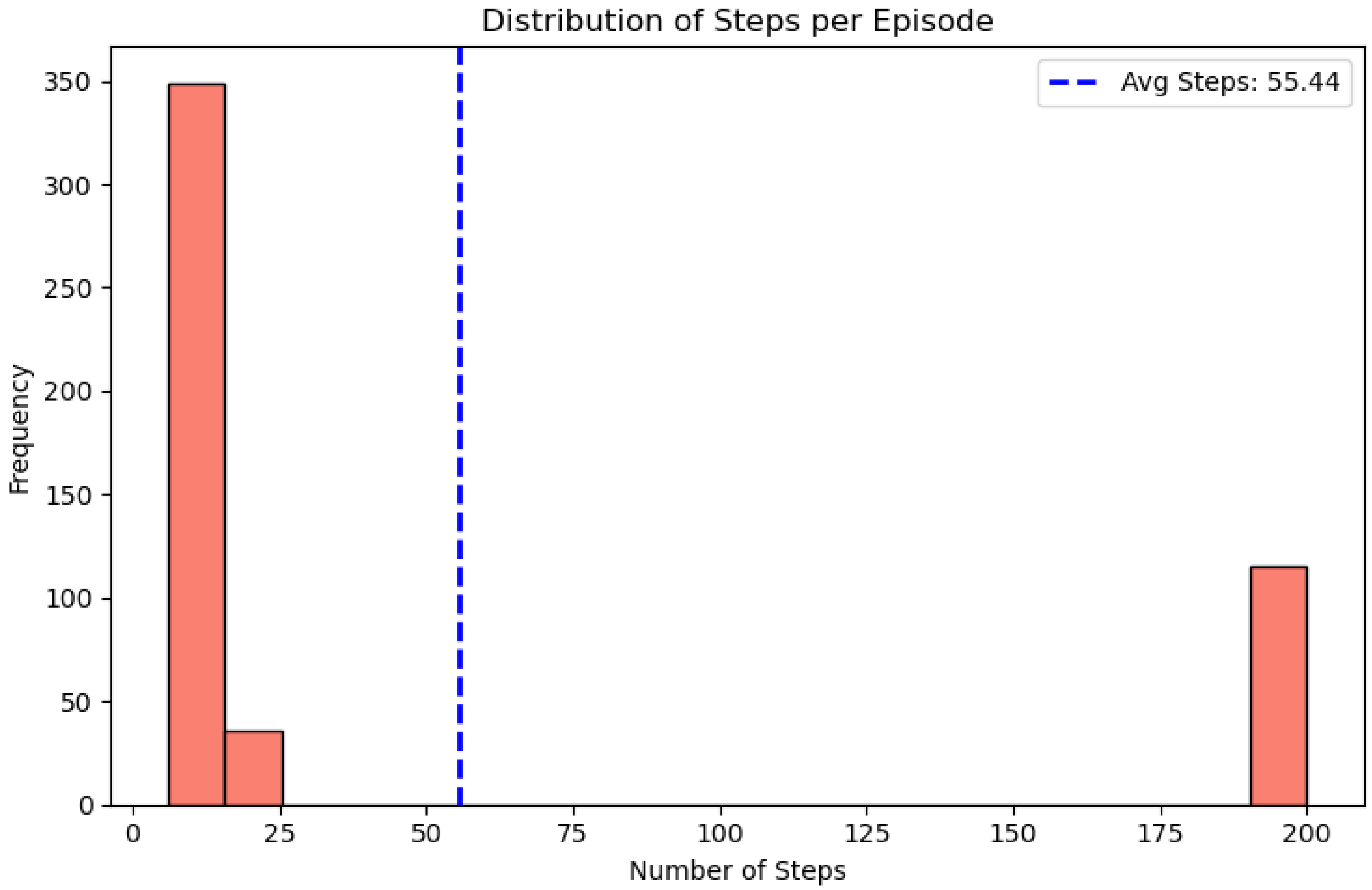
EPISODES TRAINING

parameters	15k	30k	50k	75k
Mean reward over 100 episodes	-91.40	-68.41	-41.49	-37.11
Success rate	52%	63%	76%	78%



FINAL EVALUATION





Results

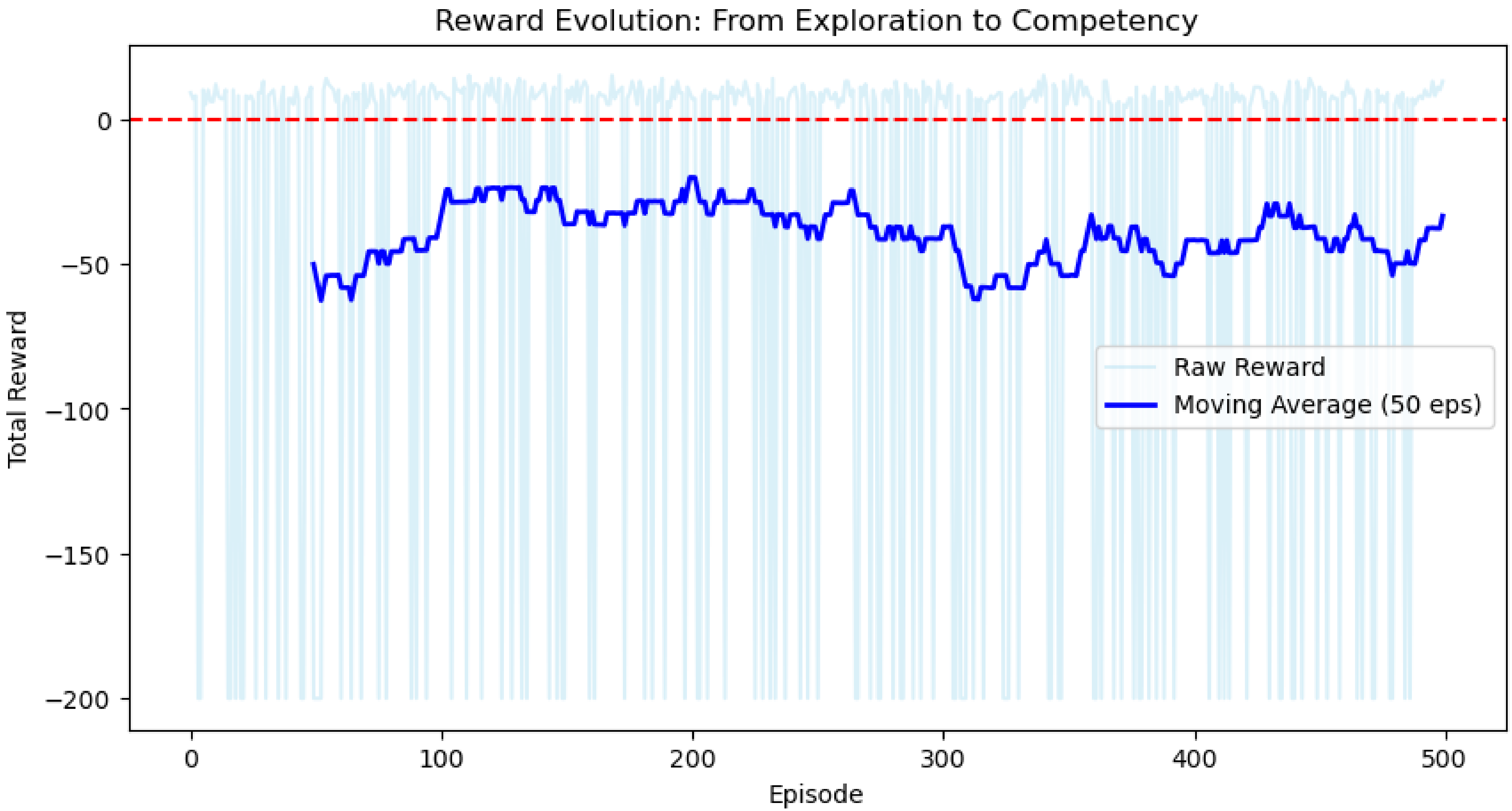
- **500** test episodes
- Success Rate **77%**
- Average Reward **-39.27**
- Average Steps **55.44**

FINAL EVALUATION

Results

- **500** test episodes
- Success Rate **77%**
- Average Reward **-39.27**
- Average Steps **55.44**

.



COMPARISON

Q-Learning

- Converges much faster
- Updated after every single step and uses "Bellman Equation" to estimate future reward based on the very next state
- Positive mean reward (around 7) after 100k episodes
- Discount factor 0.9 was effective
- Q-table is limited to env where you can fit every state into a table
- More stable for Taxi env

Stochastic Policy Gradient

- REINFORCE takes longer
- Uses Monte Carlo method, it must finish a full episode before it can update its weights
- Achieved negative mean reward (-37) with 78% success rate at 75k episodes
- Discount factor 0.99 was a must for REINFORCE
- Model (using NN) can handle continuous or much larger state spaces
- Could suffer from high-variance

LETS SEE NOW! Q-LEARNING

LETS SEE NOW! POLICY GRADIENT

THANK YOU