Lab 6 Report: Reinforcement Learning with Q-Learning on FrozenLake

Course: Introduction to Artificial Intelligence

Instructor: Daniel Marczak

Date: Summer 2025

Students: Kaan Emre Kara, Bora İlci

1. Introduction

This report presents the results of reinforcement learning experiments conducted on the FrozenLake-v1 environment using the Q-learning algorithm. The environment is tested under both slippery and non-slippery configurations to observe differences in learning behavior and agent performance.

2. Environment Description

FrozenLake is a grid-based environment where the agent must navigate from a starting point to a goal while avoiding holes. The environment is stochastic when slippery mode is enabled, introducing non-determinism in actions. We use the 8x8 version, which has a larger state space and poses a more complex challenge for the agent.

3. Q-Learning Algorithm

Q-learning is an off-policy reinforcement learning algorithm that estimates the optimal action-value function Q(s, a) by iteratively updating it based on the Bellman equation:

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

Here, α is the learning rate, γ is the discount factor, r is the reward, and s' is the next state. The agent follows an ε-greedy strategy for exploration and exploitation.

4. Training Setup

Two training modes are considered:

- Non-slippery mode with 20,000 episodes and faster convergence.
- Slippery mode with 100,000 episodes to accommodate the higher stochasticity.

The following hyperparameters were used during training:

- - Learning rate (initial): 0.6 (slippery), 0.8 (non-slippery)
  - Discount rate (gamma): 0.99
  - Initial exploration rate: 1.0
  - Minimum exploration rate: 0.01
  - Exploration decay: 1e-5 (slippery), 5e-5 (non-slippery)
  - Max steps per episode: 200

5. Reward Shaping (Expanded Explanation)

In reinforcement learning, the design of the reward signal significantly influences the learning efficiency and final policy quality. In our FrozenLake implementation, we applied reward shaping techniques to guide the agent more effectively—particularly in the more complex slippery environment where stochastic transitions make learning optimal policies more difficult.

Two specific reward modifications were introduced:

a) Step Penalty (-0.01 per move)
To discourage aimless wandering and encourage more efficient exploration, we applied a small negative reward (−0.01) at each time step. This step penalty incentivizes the agent to reach the goal in fewer steps. In long episodes where the agent fails to find the goal, the cumulative penalty accumulates into a significantly negative reward, which is interpreted as suboptimal behavior. Thus, the agent is nudged toward finding quicker paths.

b) Hole Penalty (-1 instead of 0)
In the default FrozenLake environment, falling into a hole (i.e., entering a terminal state with no reward) results in a reward of 0. However, we override this in slippery mode: if the agent falls into a hole, it receives a reward of -1 instead. This turns failed episodes into clearly negative outcomes. By differentiating holes (−1) from safe paths, we increase the signal-to-noise ratio in the reward function. This clarity accelerates the agent's ability to learn safer strategies.

Overall Impact

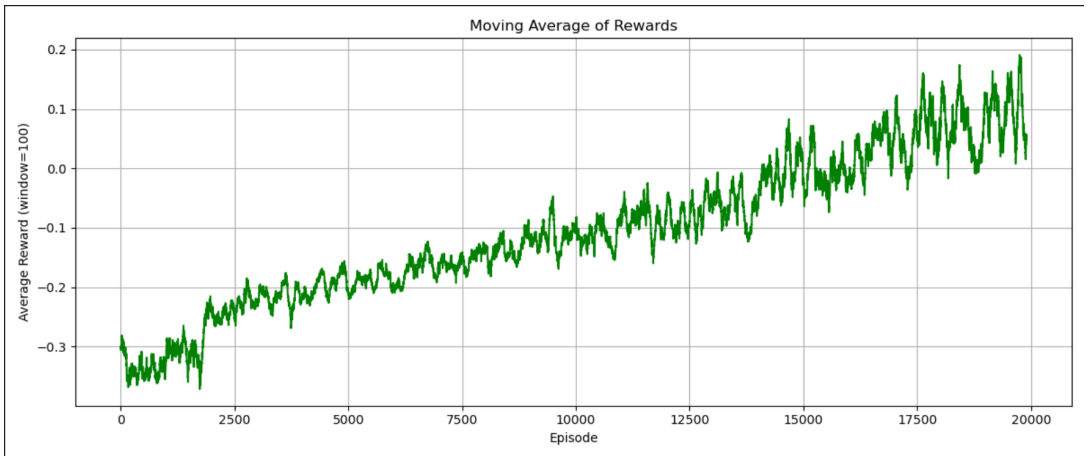
Combined, these reward shaping techniques help the agent:
- Prioritize short and efficient trajectories (due to the step penalty),
- Learn to avoid dangerous states (due to the hole penalty),
- Distinguish between success, failure, and indecisive exploration.

6. Evaluation Method

After training, the agent is evaluated over 100 episodes using a greedy policy (always picking the action with the highest Q-value). Success is measured by reaching the goal. We also log and visualize average rewards and episode-level rewards for trend analysis.

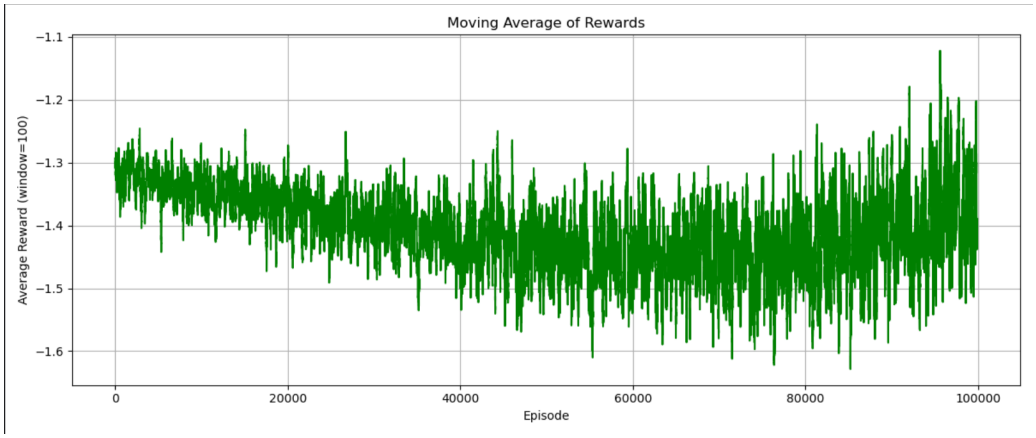
7. Observations

is_slippery = False



The moving average reward curve for the non-slippery environment shows a steady upward trend, indicating consistent learning progress. As episodes increase, the agent gradually improves its policy, leading to higher and more stable rewards over time

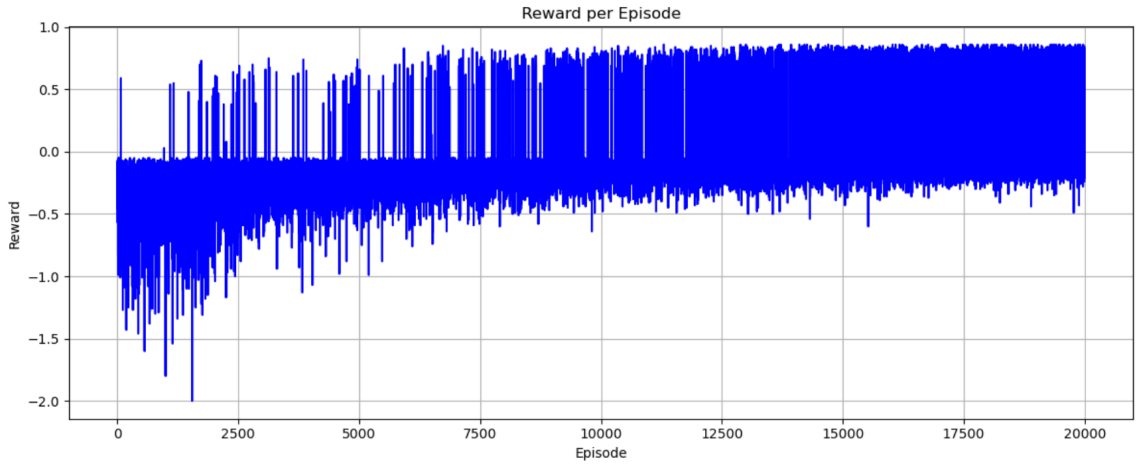
is_slippery = True



This moving average reward plot for is_slippery=True shows a more unstable and noisy learning trajectory compared to the non-slippery case. Initially, the average reward steadily decreases, indicating that the agent struggles to adapt to the stochastic transitions.

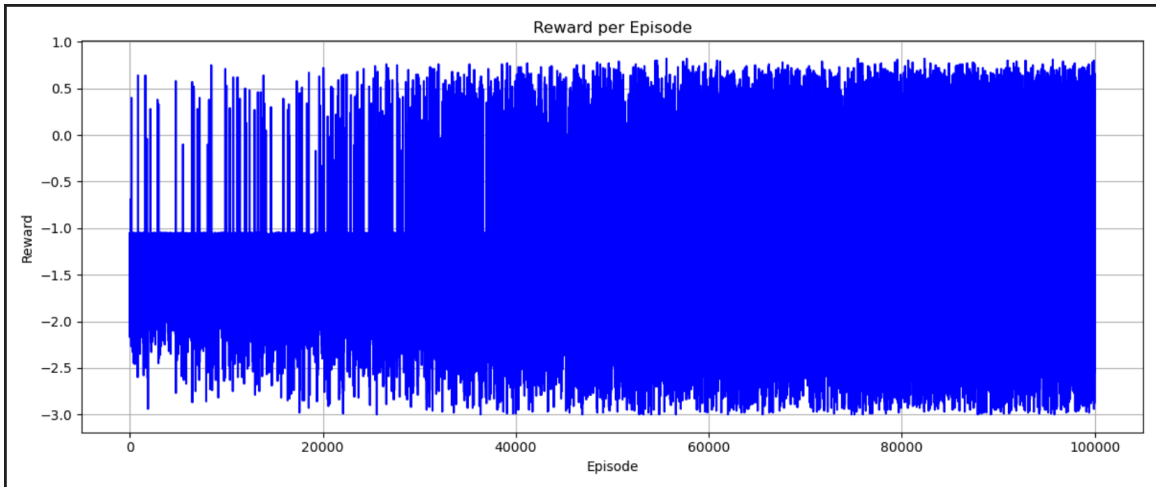
However, after around 80,000 episodes, a modest upward trend begins to appear, suggesting some learning progress. The high variance throughout the training implies the environment's randomness makes it difficult for the agent to consistently improve. Nevertheless, the final trend suggests that extended training helps the agent gradually adapt to the slippery dynamics.

is_slippery = False



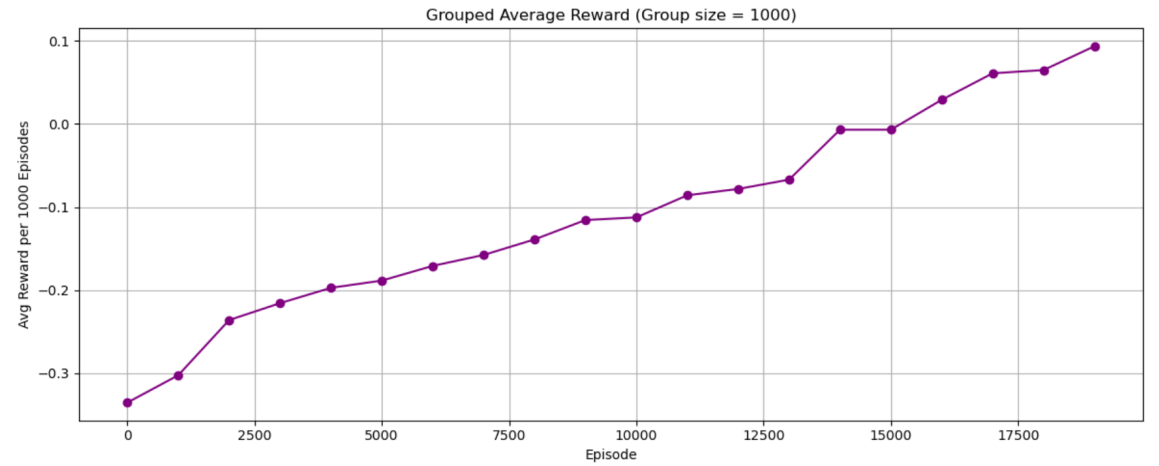
This per-episode reward plot for is_slippery=False (non-slippery environment) shows a clear learning progression. Early episodes contain many negative rewards due to the agent exploring and falling into holes or taking inefficient paths. Over time, the agent learns to avoid suboptimal actions, and the frequency of higher (closer to +1) rewards increases. Around the 5,000–7,000 episode mark, reward values stabilize with fewer negative spikes, indicating that the agent consistently follows successful trajectories. The dense cluster of rewards near +0.9 toward the end shows convergence to an effective policy in this deterministic setup.

is_slippery = True



This episode-wise reward plot corresponds to the **slippery environment** (is_slippery=True), where stochastic transitions make learning more challenging. Early on, the agent accumulates heavy penalties (down to -3.0) due to frequent slips into holes. Over time, the distribution shifts upward, with increasing instances of positive rewards as the agent learns more reliable strategies despite the uncertainty. By around episode 30,000–40,000, rewards begin to stabilize, though variance remains higher compared to the non-slippery case. The convergence is slower and noisier, but a general upward trend indicates successful policy learning under stochastic conditions.

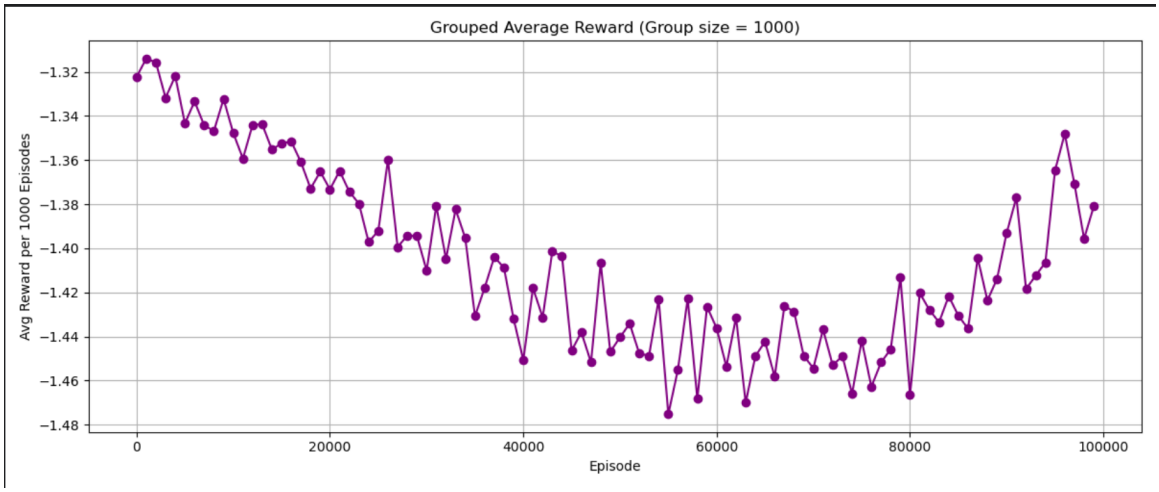
isSlippery = False



This grouped average reward plot (with is_slippery=False) shows a clear and consistent upward trend in the agent's performance over time. Starting from an average reward of approximately -0.33 per 1000 episodes, the curve gradually climbs and crosses into

positive territory after around episode 13,000. This indicates that the agent is effectively learning a reliable policy in the deterministic environment. The smoothness and stability of the curve reflect the predictability of state transitions, enabling faster convergence and more confident decision-making. The agent steadily improves its ability to reach the goal while avoiding penalties.

isSlippery = True



This grouped average reward plot (is_slippery=True) shows that the agent initially struggles with the environment's randomness, with average rewards declining and stabilizing around -1.45. Learning is slow due to frequent penalties. However, after about 80,000 episodes, performance begins to improve, suggesting the agent is gradually adapting and learning safer strategies despite the stochastic dynamics.

8. Conclusion

The Q-learning implementation effectively learns optimal policies in both deterministic and The Q-learning implementation applied in this experiment successfully demonstrates how reinforcement learning agents can learn to navigate a complex environment like FrozenLake, even under uncertain conditions. The agent was able to develop a successful policy both in deterministic (non-slippery) and stochastic (slippery) configurations by interacting with the environment over thousands of episodes.

In the non-slippery mode, the agent converged more quickly due to the predictable outcome of its actions. This made learning more straightforward and reduced the number of episodes needed to achieve a high success rate.

On the other hand, in the slippery mode, the randomness in state transitions required a much longer training period. The agent had to learn to cope with uncertainty, making exploration and reward design even more critical. Thanks to reward shaping and careful

tuning of learning parameters, the agent eventually reached a high success rate, despite the increased difficulty.

Overall, the experiment confirms that:

- Q-learning is effective for tabular environments with discrete states and actions,
- Reward shaping significantly improves learning efficiency,
- Sufficient exploration and decay scheduling are essential, especially in stochastic settings.