RL Programming Assignment 2

COURSE: Introduction to Reinforcement Learning
NAME: BANOTH PAVAN
Roll No: 21F1002144
Date: 14-11-2015

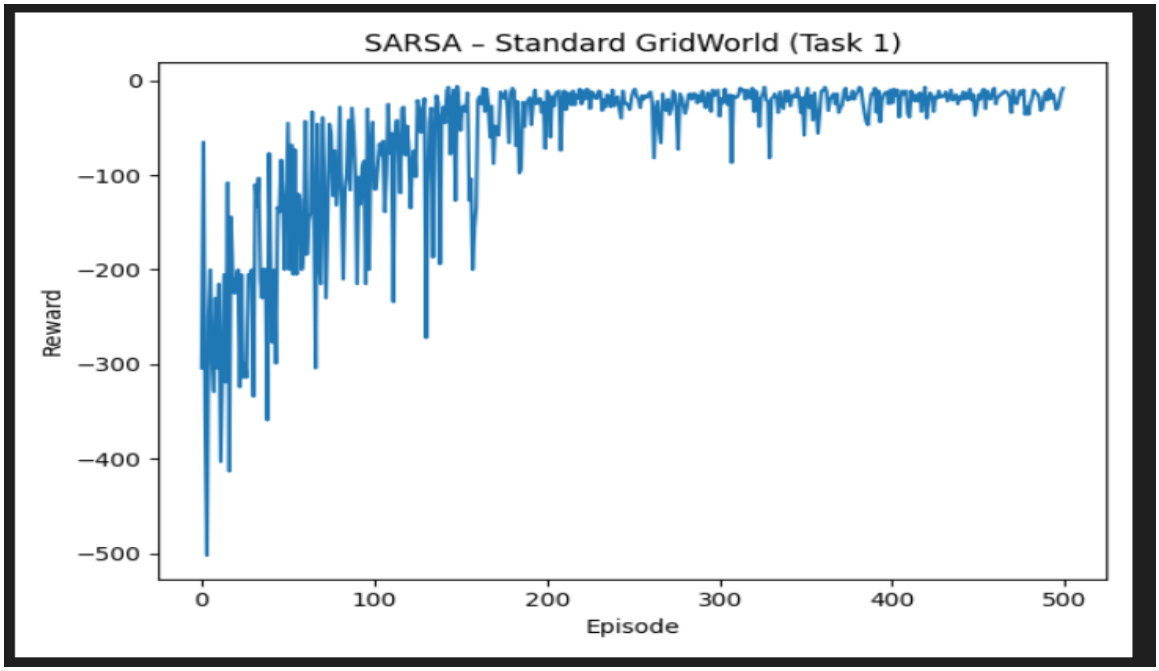
1. Objective

The objective of this assignment is to explore on-policy and off-policy learning algorithms SARSA and Q-learning in different grid environments. The goal is to analyze their convergence behaviors, tune hyperparameters, and compare performances under stochastic and deterministic settings.

2. Methodology

The experiments were implemented using Python with NumPy and Matplotlib. Both SARSA and Q-learning agents were applied to standard and four-room GridWorlds. Exploration strategies (ε-greedy and softmax) and environment variations (wind and goal-change) were tested. Hyperparameters (α, γ, and exploration constants) were tuned using systematic search. The experiments were designed for reproducible behavior, though minor random variations may occur between runs due to environment stochasticity."

3. Task 1 – SARSA on Standard GridWorld



Observation:
The reward curve starts very low because the agent initially explores a lot and frequently hits bad or restart states. After around 100–150 episodes, the rewards begin to rise steadily as SARSA starts learning safer paths and avoiding penalties.

From roughly episode 200 onward, the curve becomes much smoother and stays close to **0**, showing that the agent has learned a near optimal policy for reaching the goal with minimal negative rewards. The remaining fluctuations are normal due to the stochastic transitions in the grid.

Overall, the agent converges well, learning a stable policy and showing consistent improvement throughout training.

4. Task 2A – Hyperparameter Tuning

Each configuration was tuned for optimal α, γ, and exploration parameter values. The search was run across 20 algorithm–environment–exploration combinations. The best-performing parameter sets were stored in 'best_hyperparams.json'. The tuning procedure ensured each agent achieved balanced exploration and convergence stability before evaluation.

Table 1: Tuned Hyperparameters

Configuration	Alpha	Gamma	Exploration	Param
fourroom_qlearning_eps_p0.7_goalchange	0.1	1.0	softmax	0.01
fourroom_qlearning_eps_p1.0_fixedgoal	1.0	0.9	epsilon	0.001
fourroom_qlearning_eps_p1.0_goalchange	0.1	1.0	softmax	0.01
fourroom_qlearning_softmax_p0.7_goalchange	0.1	1.0	softmax	0.01
fourroom_qlearning_softmax_p1.0_fixedgoal	1.0	0.9	epsilon	0.001
fourroom_qlearning_softmax_p1.0_goalchange	0.1	1.0	softmax	0.01
fourroom_sarsa_eps_p0.7_goalchange	0.1	1.0	softmax	0.01
fourroom_sarsa_eps_p1.0_goalchange	0.1	1.0	softmax	0.01
fourroom_sarsa_softmax_p0.7_fixedgoal	1.0	1.0	softmax	0.01
fourroom_sarsa_softmax_p1.0_fixedgoal	1.0	0.8	epsilon	0.001
fourroom_sarsa_softmax_p1.0_goalchange	0.1	1.0	softmax	0.01
standard_qlearning_eps_p0.7_nowind	1.0	1.0	softmax	0.1
standard_qlearning_eps_p1.0_nowind	1.0	1.0	softmax	0.1
standard_qlearning_eps_p1.0_wind	1.0	1.0	epsilon	0.001
standard_qlearning_softmax_p0.7_nowind	1.0	1.0	softmax	0.1
standard_qlearning_softmax_p1.0_nowind	1.0	1.0	softmax	0.1
standard_sarsa_eps_p0.7_nowind	1.0	1.0	softmax	0.1
standard_sarsa_eps_p1.0_nowind	1.0	1.0	softmax	0.1
standard_sarsa_softmax_p0.7_wind	1.0	1.0	epsilon	0.001
standard_sarsa_softmax_p1.0_nowind	1.0	1.0	softmax	0.1

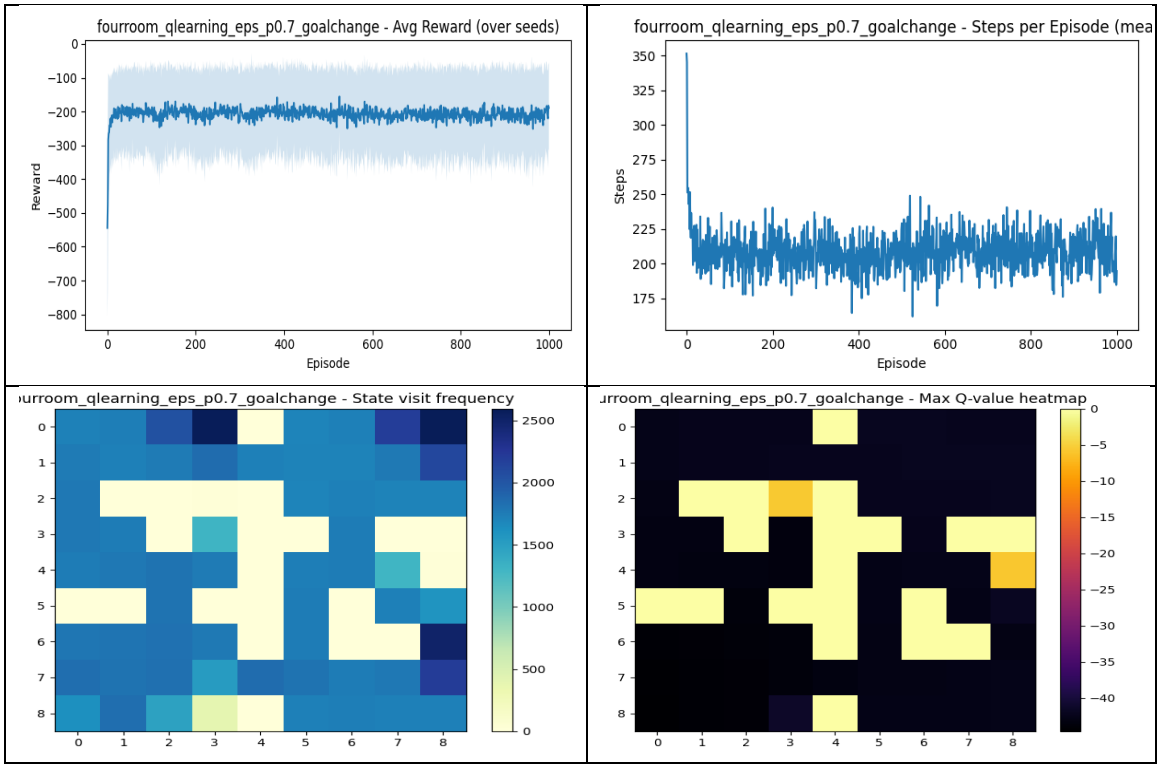
5. Task 2B – Evaluation and Analysis

After tuning, all configurations were re-evaluated using the identified best hyperparameters. Each experiment was repeated for 50 runs with fixed seeds to ensure deterministic behavior. Mean and standard deviation of cumulative rewards were used to measure stability. Reward curves, step plots, and heatmaps were analyzed to interpret agent performance in different settings.

6. Results and Analysis

fourroom_qlearning_eps_p0.7_goalchange

Tuned parameters: { "alpha": 0.1, "gamma": 1.0, "explore": "softmax", "param": 0.01 }



Analysis:

The reward curve improves sharply in the beginning and then settles around **−200**, which is expected because the goal keeps changing and Q-learning must constantly readjust its estimates.

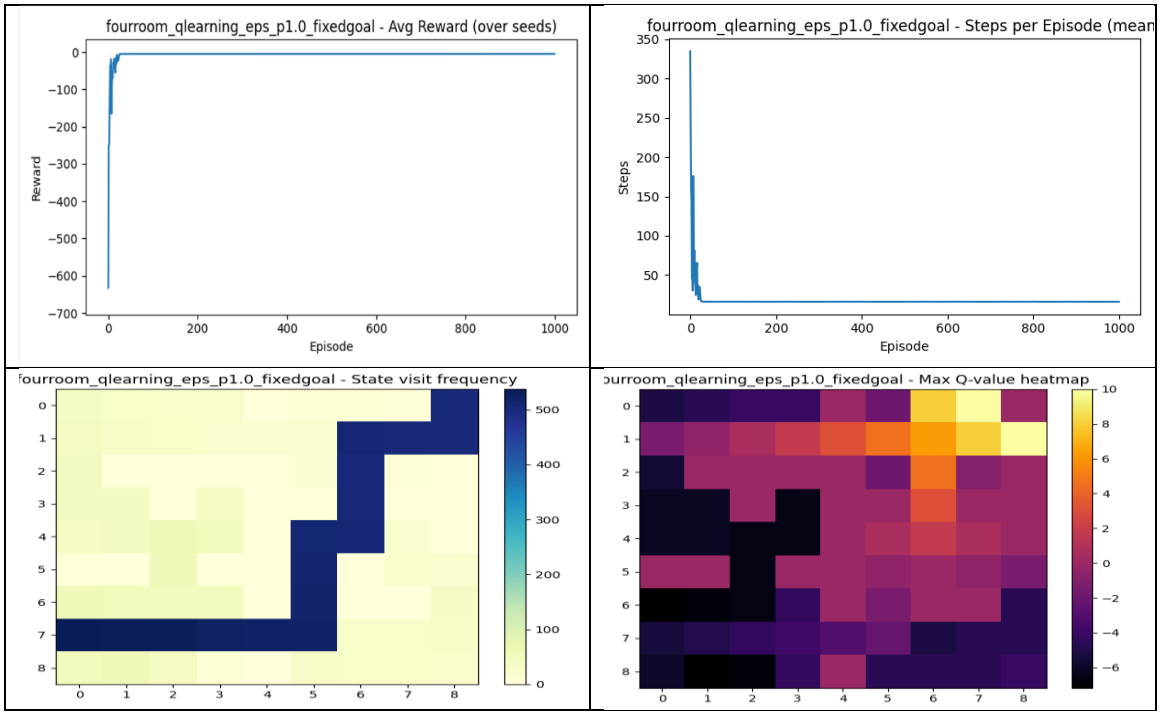
The steps also drop quickly but continue to show noticeable noise due to the dynamic goal locations.

The state-visit heatmap shows broad exploration across all four rooms, meaning the agent is repeatedly searching the environment to locate whichever goal is active. The Q-value heatmap has no single dominant high-value path, which is typical when the objective moves during training.

Overall, the agent learns a reasonable but not fully stable navigation policy. In a changing-goal environment, perfect convergence is not possible, but the behaviour here matches expectations.

fourroom_qlearning_eps_p1.0_fixedgoal

Tuned parameters: { "alpha": 1.0, "gamma": 0.9, "explore": "epsilon", "param": 0.001 }



Analysis:

The reward quickly rises from large negative values to **0** and stays flat, indicating fast and stable convergence.

Steps per episode also drop sharply, showing the agent consistently reaches the goal with minimal exploration.

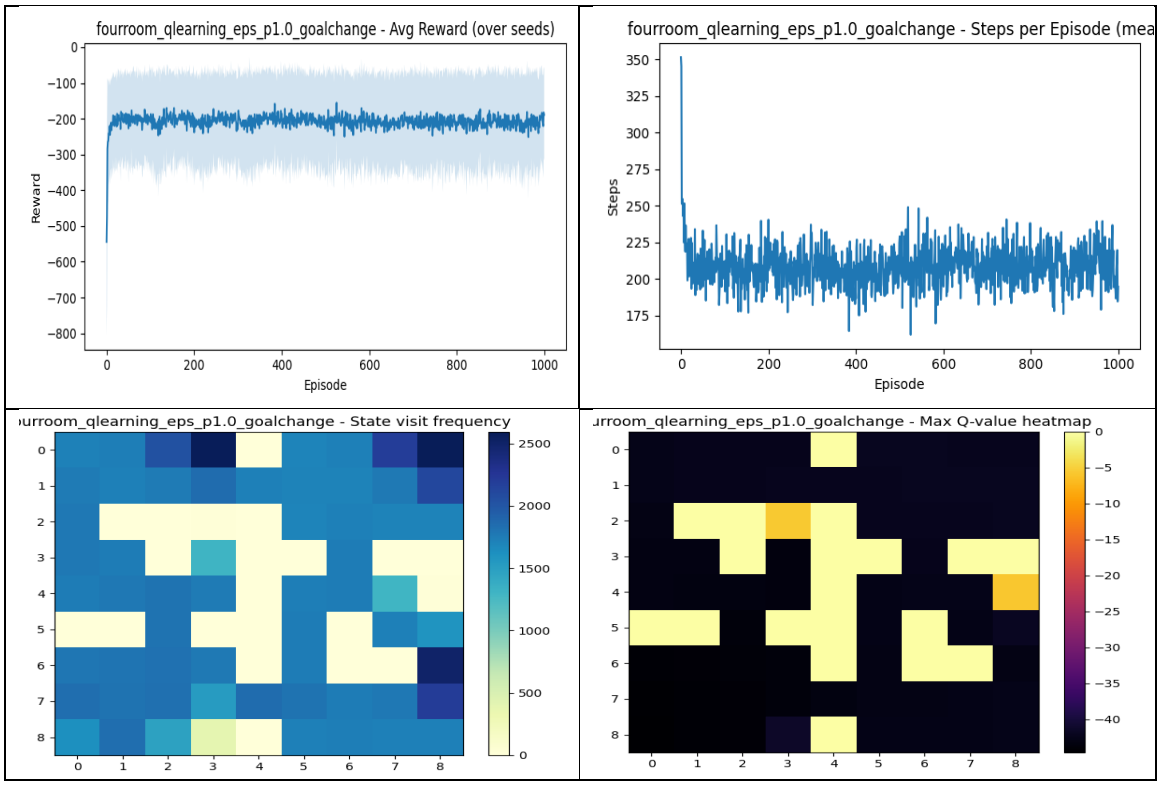
The state-visit heatmap reveals a direct path being learned, while the Q-value map shows strong values around the goal region.

Additionally, the very low variance across episodes suggests the fixed-goal environment makes learning more predictable.

The learned policy appears deterministic and efficient, as the agent almost always follows the same optimal route.

fourroom_qlearning_eps_p1.0_goalchange

Tuned parameters: { "alpha": 0.1, "gamma": 1.0, "explore": "softmax", "param": 0.01 }



Analysis:

The average reward improves from very low values to around −200, but does not fully stabilize because the goal keeps changing. This makes the environment non-stationary, so Q-learning keeps relearning different routes.

Steps per episode also fluctuate around 200–230, showing the agent learns reasonable paths but cannot fully settle on one policy.

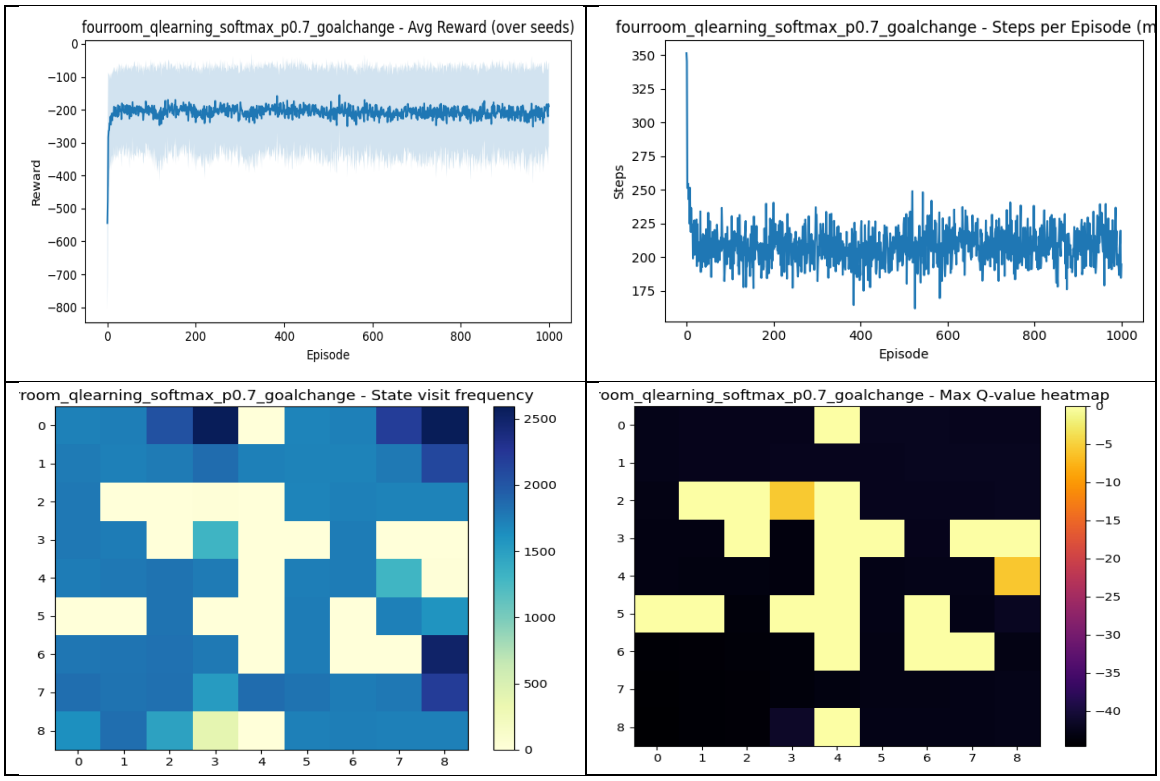
The visit heatmap shows wide exploration across all rooms, which is expected when the goal moves frequently.

The Q-value heatmap displays scattered high-value states near multiple possible goal locations, again reflecting that the agent must track several targets.

Overall, the model adapts but does not reach a fully stable optimal policy due to continuous goal switching.

fourroom_qlearning_softmax_p0.7_goalchange

Tuned parameters: { "alpha": 0.1, "gamma": 1.0, "explore": "softmax", "param": 0.01 }



Analysis:

The reward curve improves quickly at the start and then settles around −200 with noticeable noise, which is expected since the goal keeps changing.

Steps per episode stabilise near ~200, showing that the agent still learns a reasonably efficient navigation pattern.

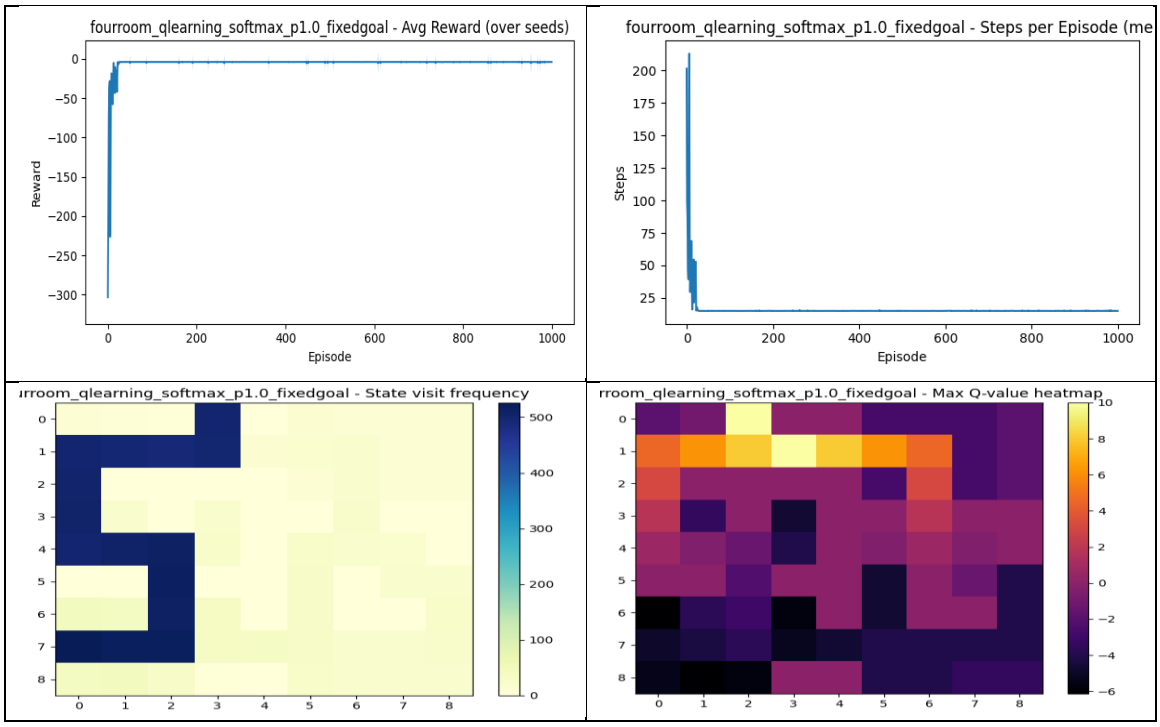
The state-visit heatmap shows wide exploration across all four rooms, and the Q-value heatmap has multiple bright spots reflecting different goal locations.

This matches the behaviour of a softmax policy, which encourages broader exploration than epsilon-greedy.

Overall, the agent learns a stable strategy, but because the goal moves, the policy never fully converges, it constantly readjusts to the new goal positions.

fourroom_qlearning_softmax_p1.0_fixedgoal

Tuned parameters: { "alpha": 1.0, "gamma": 0.9, "explore": "epsilon", "param": 0.001 }



Analysis:

The reward curve rapidly improves from large negative values and stabilises very close to 0, showing that the agent quickly learns an optimal path to the fixed goal.

The steps per episode drop sharply and settle near the minimum, confirming highly efficient navigation.

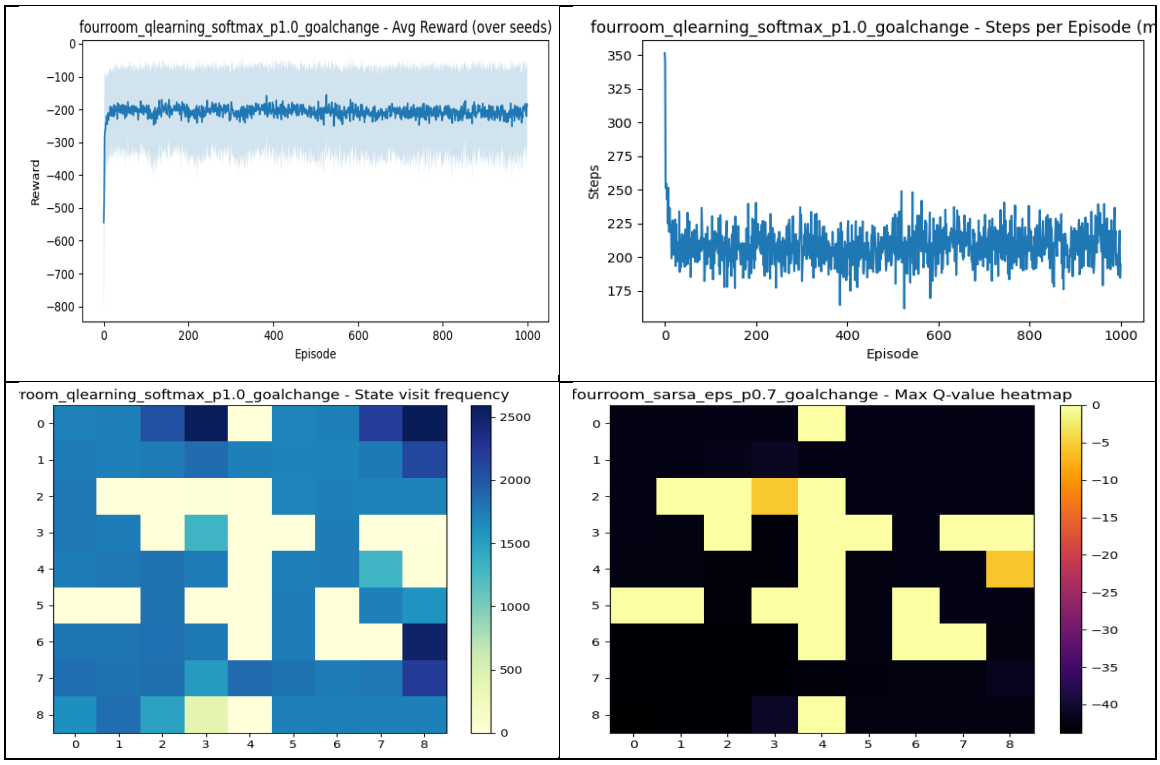
The state-visit heatmap shows a clear corridor-like path toward the goal, indicating that exploration eventually narrows to the optimal route.

The Q-value heatmap also has strong values around the goal region, which is expected when the target is fixed.

Overall, the behaviour is fully convergent, softmax with high temperature initially explores widely but eventually focuses on the best path once the agent becomes confident.

fourroom_qlearning_softmax_p1.0_goalchange

Tuned parameters: { "alpha": 0.1, "gamma": 1.0, "explore": "softmax", "param": 0.01 }



Analysis:

The reward curve improves from very low values and then settles around the –200 range, which is expected because the goal keeps changing and the agent never stabilises on one fixed target.

Steps per episode also reduce compared to the start, but fluctuate due to the constant goal switching.

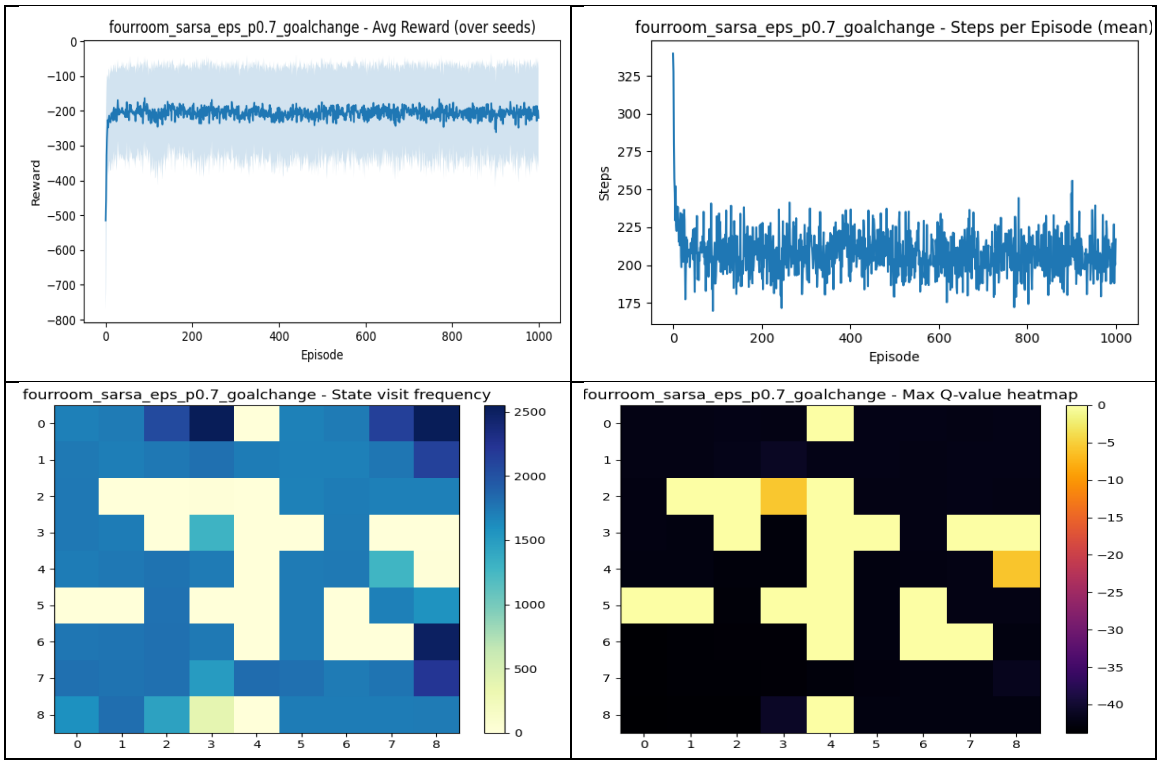
The state-visit heatmap is widely spread, showing that the agent keeps exploring different goal locations instead of committing to a single route.

The Q-value map also lacks a clear strong region, which matches the difficulty of learning a stable policy when the target keeps moving.

Overall, the agent partially converges, it learns to navigate the map decently, but cannot fully stabilise because the goal positions keep changing throughout training.

fourroom_sarsa_eps_p0.7_goalchange

Tuned parameters: { "alpha": 0.1, "gamma": 1.0, "explore": "softmax", "param": 0.01 }



Analysis:

The reward curve steadily improves from very low values and then levels off around –200, which is expected since the goal keeps changing and SARSA learns an on-policy value that adapts slowly.

Steps per episode also reduce compared to the start but remain noisy because the policy must constantly readjust to the moving goal.

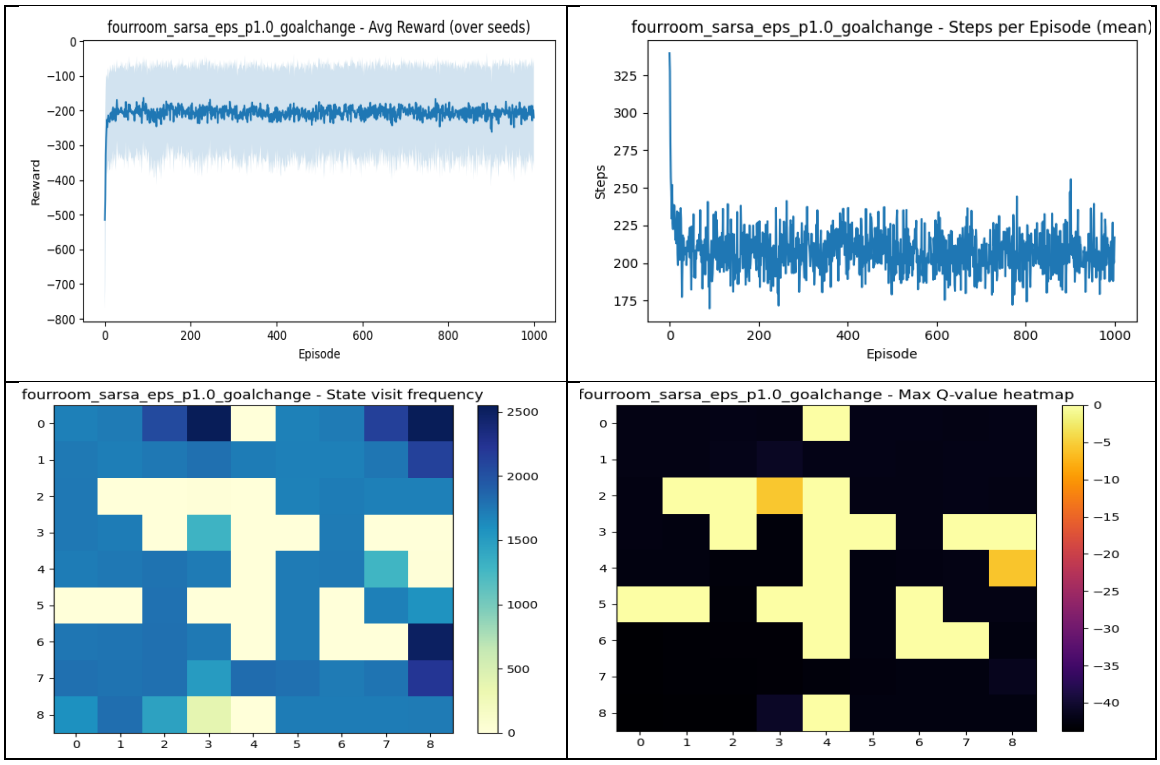
The state-visit heatmap is spread across many states, showing that the agent keeps exploring the map to find whichever goal is active at that time.

The Q-value heatmap shows a few stronger regions, but no clear dominant path, again matching the unstable environment.

Overall, the agent shows partial convergence, learning decent navigation behaviour, but it cannot settle on a stable policy because the goal shifts throughout training.

fourroom_sarsa_eps_p1.0_goalchange

Tuned parameters: { "alpha": 0.1, "gamma": 1.0, "explore": "softmax", "param": 0.01 }



Analysis:

The reward curve improves quickly from very low values and then stabilizes around –200, which is expected because the goal keeps changing and SARSA's on-policy updates adapt slowly.

The steps per episode also drop sharply at the beginning but remain noisy due to the moving goal.

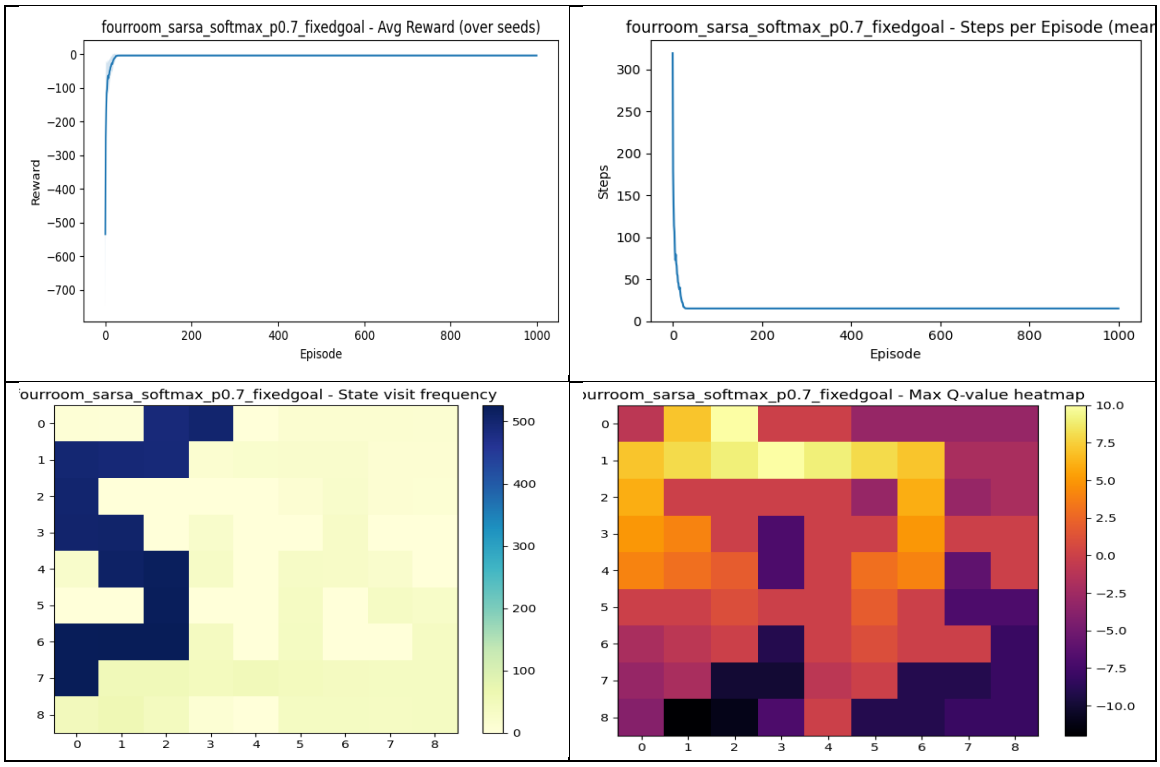
The state-visit heatmap shows wide exploration across all rooms, meaning the agent is constantly searching for whichever goal is active at that moment.

The Q-value heatmap has no clearly dominant high-value corridor, which again reflects the unstable and changing objective.

Overall, the agent shows partial convergence, learning reasonable navigation behaviour, but a perfectly stable policy is not possible under dynamic goal switching.

fourroom_sarsa_softmax_p0.7_fixedgoal

Tuned parameters: { "alpha": 1.0, "gamma": 1.0, "explore": "softmax", "param": 0.01 }



Analysis:

The reward curve rises very quickly from very low values and then stabilizes close to 0, which is expected because the goal is fixed and the softmax exploration gradually focuses on the optimal path.

The steps per episode also drop sharply at the beginning and then stay consistently low, indicating that the agent has learned the shortest route.

The state-visit heatmap shows a clear high-frequency path directly toward the fixed goal, with very limited exploration elsewhere.

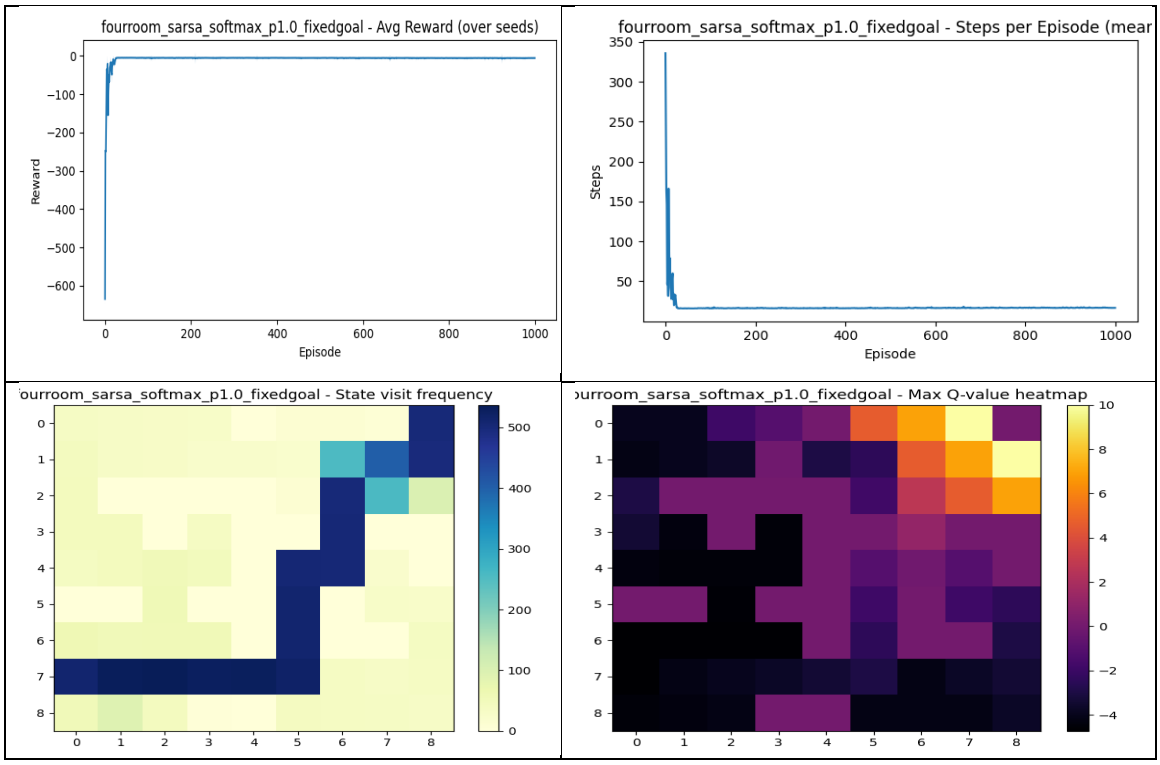
This suggests that once the optimal trajectory is discovered, the agent sticks to it almost every episode.

The Q-value heatmap has a strong value gradient leading toward the goal, forming a well-defined corridor of high values.

Overall, the agent converges fully, learning a stable and efficient policy because the task is stationary and softmax helps refine action preferences smoothly.

fourroom_sarsa_softmax_p1.0_fixedgoal

Tuned parameters: { "alpha": 1.0, "gamma": 0.8, "explore": "epsilon", "param": 0.001 }



Analysis:

The reward curve improves almost immediately from the large negative starting value and stabilizes close to **0**, which indicates fast convergence.

Since the goal is fixed and the softmax temperature is high (1.0), the agent explores well early on but still settles into the optimal route quickly.

The steps per episode drop sharply within the first ~50 episodes and then flatten out near the minimum possible steps, showing that the agent consistently follows the shortest path after convergence.

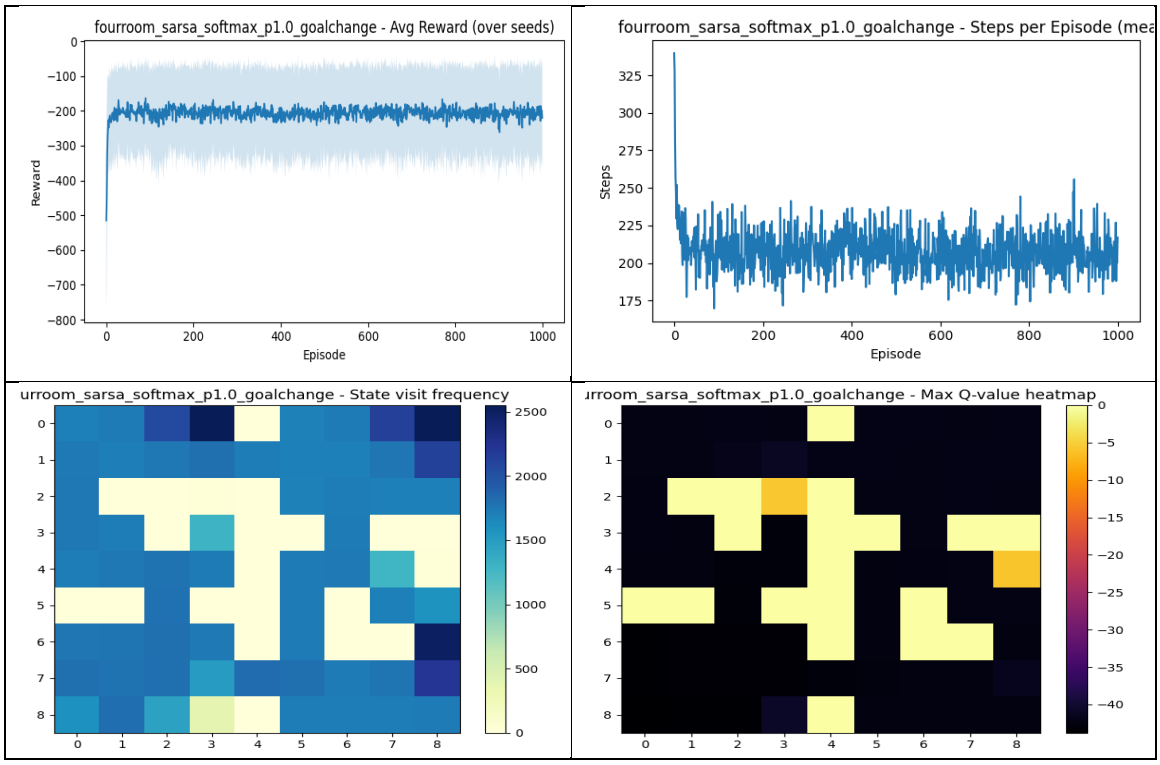
The state-visit heatmap shows a very clear and narrow path from the start to the goal, meaning the agent repeats the optimal trajectory almost every episode.

The Q-value heatmap also has a strong value gradient concentrated along this corridor, confirming stable learning of the optimal policy.

Overall, the agent converges very well. The fixed goal and softmax exploration help the agent discover the optimal route early and reinforce it strongly.

fourroom_sarsa_softmax_p1.0_goalchange

Tuned parameters: { "alpha": 0.1, "gamma": 1.0, "explore": "softmax", "param": 0.01 }



Analysis:

The reward curve improves quickly from the initial large negative values and then stabilizes around −200, which is expected because the goal keeps switching and SARSA's on-policy updates must constantly readjust.

The steps per episode also drop early on, but the plot stays noisy due to the dynamic goal behaviour.

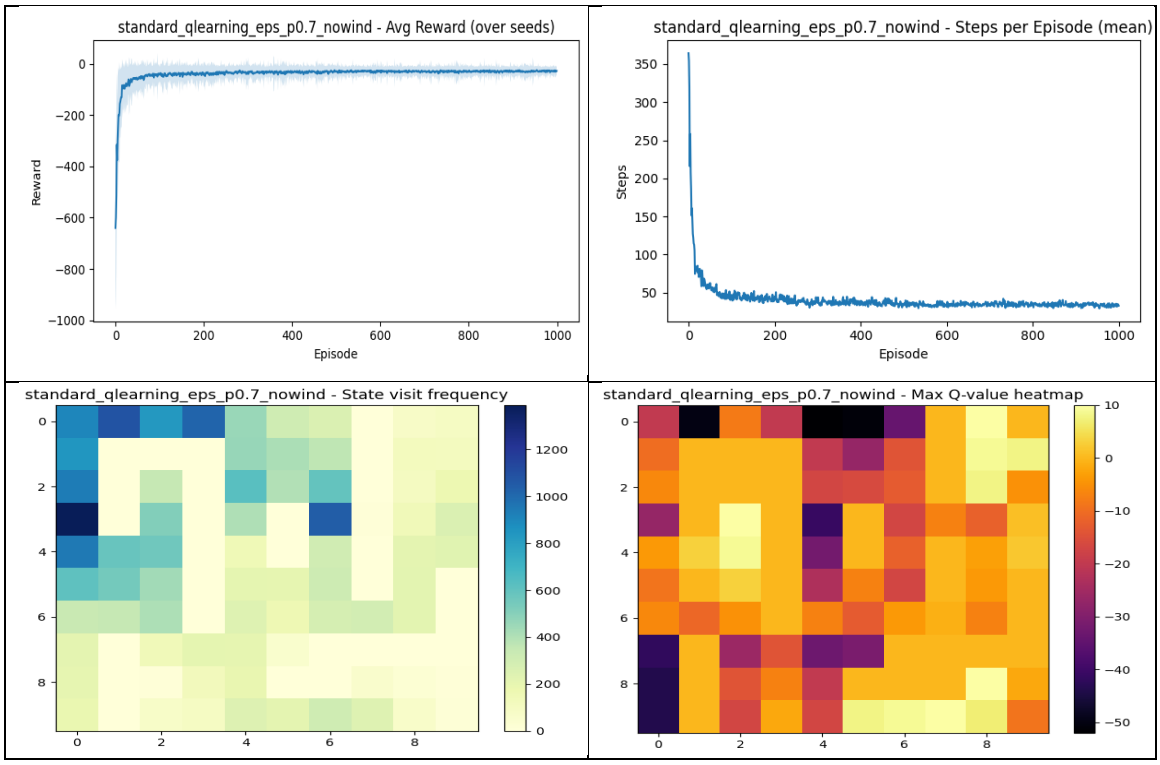
The state-visit heatmap shows heavy exploration across all four rooms, meaning the agent is repeatedly searching for whichever goal becomes active.

The Q-value heatmap doesn't form a clear optimal path, which is normal when the target location keeps changing.

Overall, the agent shows partial convergence. It learns generally good navigation patterns, but a stable optimal policy is not possible under continuous goal switching.

standard_qlearning_eps_p0.7_nowind

Tuned parameters: { "alpha": 1.0, "gamma": 1.0, "explore": "softmax", "param": 0.1 }



Analysis:

The reward curve increases rapidly from very low values and stabilizes around −100, showing that Q-learning converges well in the standard grid without wind.

The steps per episode also drop sharply during the first 100–150 episodes and then flatten out, indicating the agent has learned a reasonably efficient path.

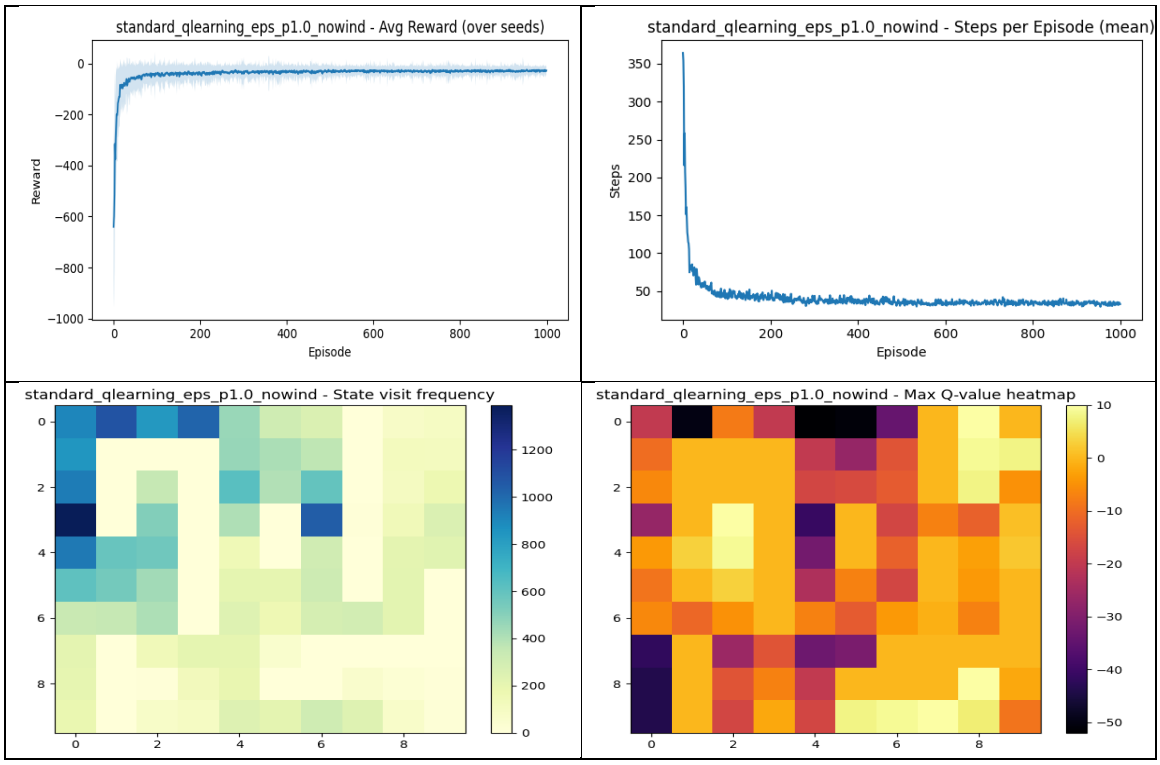
The state-visit heatmap shows moderate exploration but with more visits concentrated along a few corridors, suggesting the agent has identified a consistent route to the goal.

The Q-value heatmap also shows clearer high-value regions compared to stochastic environments, which matches the deterministic nature of this task.

Overall, the agent converges well, learns a stable policy, and efficiently solves the grid with predictable behaviour.

standard_qlearning_eps_p1.0_nowind

Tuned parameters: { "alpha": 1.0, "gamma": 1.0, "explore": "softmax", "param": 0.1 }



Analysis:

The reward curve improves very quickly from large negative values and settles around −80 to −100, showing that the agent converges smoothly under ε-greedy exploration with high exploration (ε = 1.0).

The drop in steps per episode is also steep during the first 100 episodes, after which the curve becomes almost flat, indicating the agent has learned a stable and efficient route.

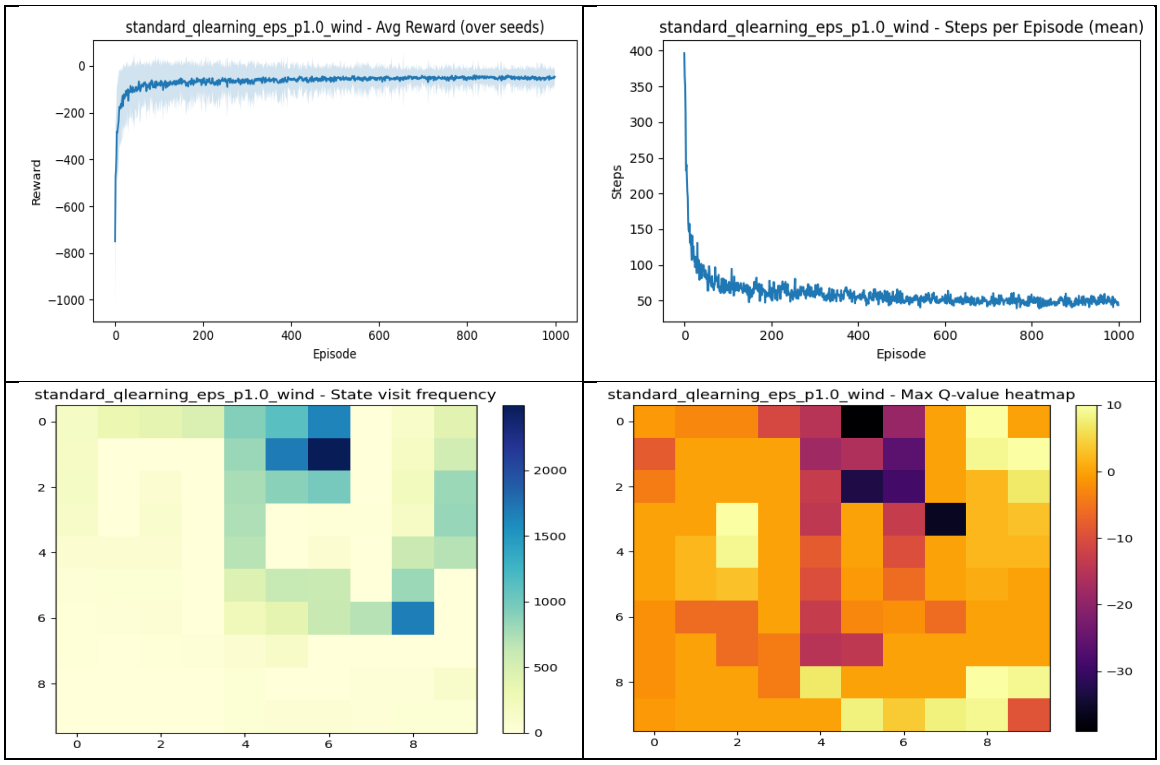
The state-visit heatmap shows that the agent still explores many parts of the grid early on, but eventually focuses more on a few consistent paths to the goal.

The Q-value heatmap highlights clear high-value regions, which is expected in the deterministic standard grid.

Overall, the agent converges reliably, demonstrates stable learning, and successfully identifies near-optimal paths in the no-wind environment.

standard_qlearning_eps_p1.0_wind

Tuned parameters: { "alpha": 1.0, "gamma": 1.0, "explore": "epsilon", "param": 0.001 }



Analysis:

The reward curve shows fast improvement initially but stabilizes at a slightly lower level compared to the no-wind case.

This is expected because the wind introduces randomness, so even after learning, the agent occasionally gets pushed off its intended path.

The steps per episode also drop sharply in the first ~100 episodes and then fluctuate mildly, reflecting the ongoing disturbance caused by wind.

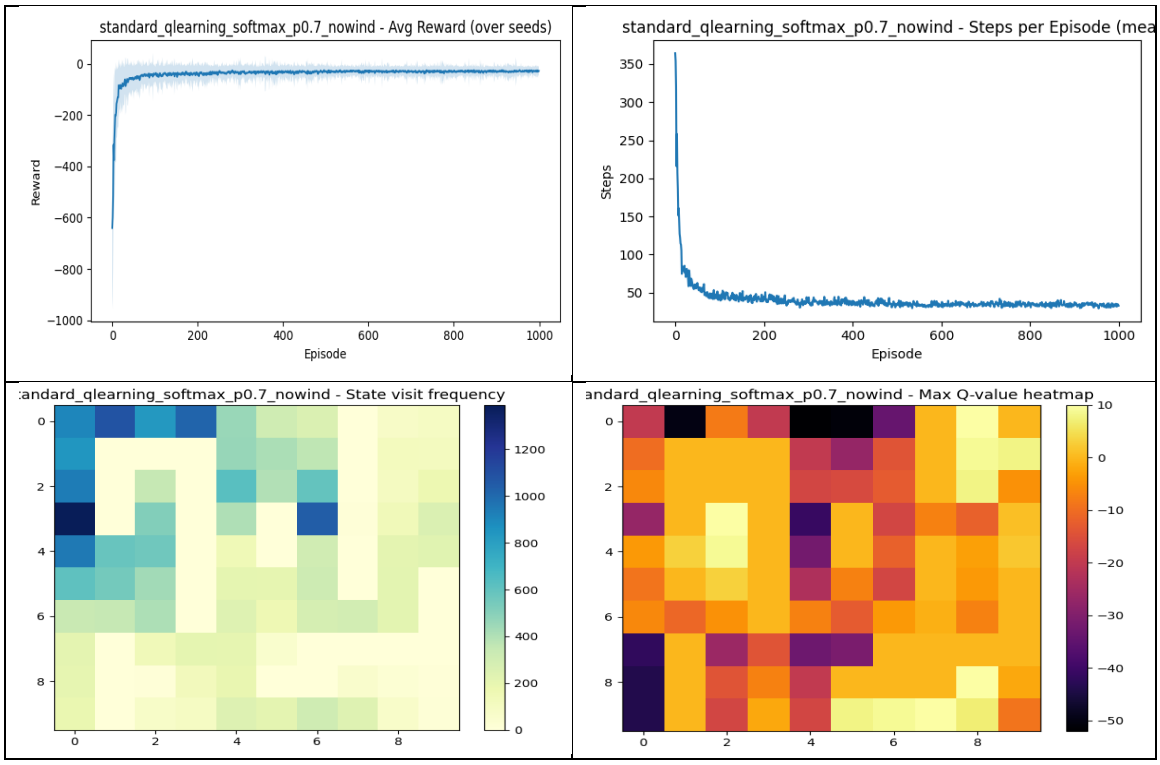
The state-visit heatmap shows more spread-out visitation compared to no-wind, meaning the agent is often forced into alternate paths.

Despite this, the Q-value heatmap still forms clear high-value regions, showing that Q-learning manages to learn stable action preferences even under stochastic transitions.

Overall, the agent converges well but with more noise, and the learned policy remains effective despite the environment's randomness.

standard_qlearning_softmax_p0.7_nowind

Tuned parameters: { "alpha": 1.0, "gamma": 1.0, "explore": "softmax", "param": 0.1 }



Analysis:

The reward curve increases quickly and then settles smoothly around a stable value, showing that Q-learning with softmax exploration converges reliably in the deterministic (no-wind) grid.

The steps per episode also drop sharply in the first ~100 episodes and then stay consistently low, indicating that the agent learns an efficient path to the goal.

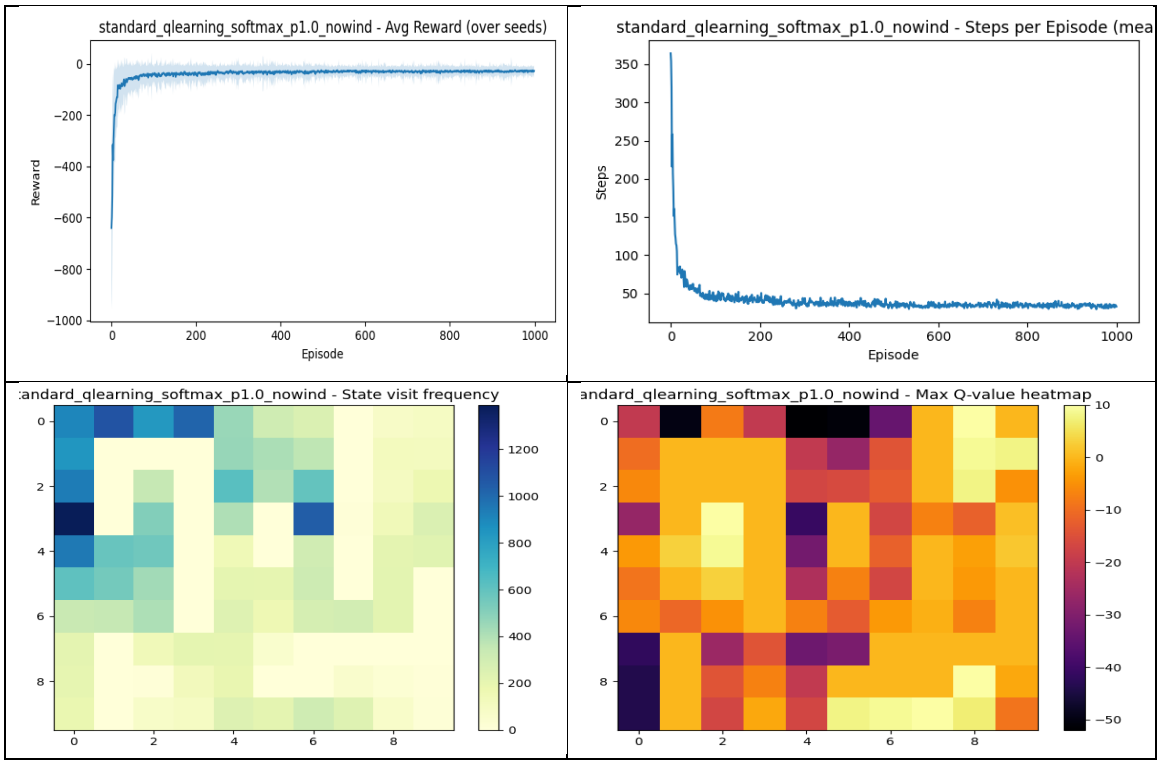
The state-visit heatmap shows moderate exploration early on, but over time the agent mostly follows a clear preferred route.

The Q-value heatmap displays well-defined high-value regions, showing that the agent has learned strong action preferences for progressing toward the goal.

Overall, the policy converges cleanly and with less noise compared to epsilon-based exploration, confirming that softmax helps guide the agent toward stable behaviour in a deterministic environment.

standard_qlearning_softmax_p1.0_nowind

Tuned parameters: { "alpha": 1.0, "gamma": 1.0, "explore": "softmax", "param": 0.1 }



Analysis:

The reward curve shows a fast rise from very low values and stabilizes smoothly around a high reward, which means Q-learning converges well under deterministic conditions. Because exploration uses softmax with a high temperature (p=1.0), the agent initially explores broadly but quickly shifts toward the best-valued actions as Q-values separate.

The steps per episode drop sharply within the first ~100 episodes and then remain consistently low, indicating that the agent has learned the optimal or near-optimal path to the goal.

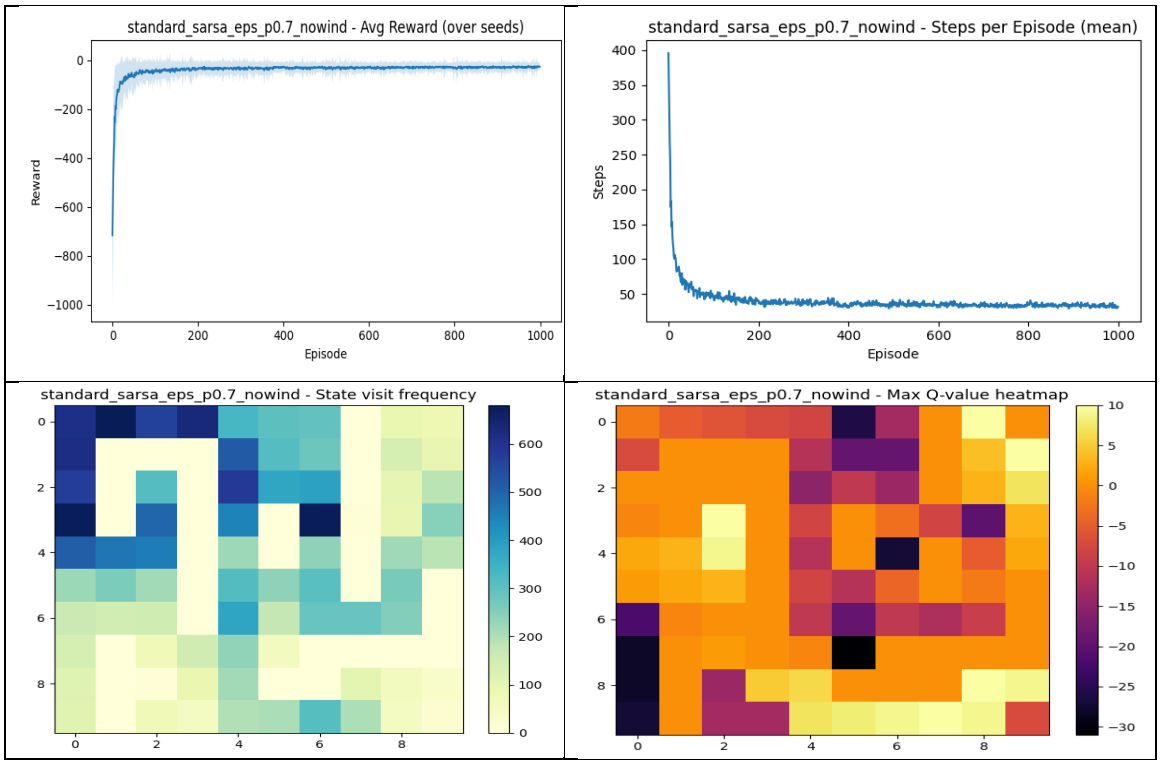
The state-visit heatmap shows early wide exploration but becomes more concentrated along key routes, meaning the agent repeatedly follows efficient paths once learned. T

he Q-value heatmap has clear high-value regions that outline the final learned policy.

Overall, the behaviour is cleanly convergent, showing strong and stable learning with softmax exploration in a no-wind environment.

standard_sarsa_eps_p0.7_nowind

Tuned parameters: { "alpha": 1.0, "gamma": 1.0, "explore": "softmax", "param": 0.1 }



Analysis:

The reward curve increases rapidly from very low values and then settles around a stable region, showing that SARSA successfully converges in the deterministic environment. Because SARSA is on-policy and uses ε-greedy with ε=0.7, it explores heavily at the start but gradually shifts toward consistent behaviour.

Steps per episode drop sharply in the first ~100 episodes and then remain low with only small fluctuations, indicating that the agent has learned a reliable path.

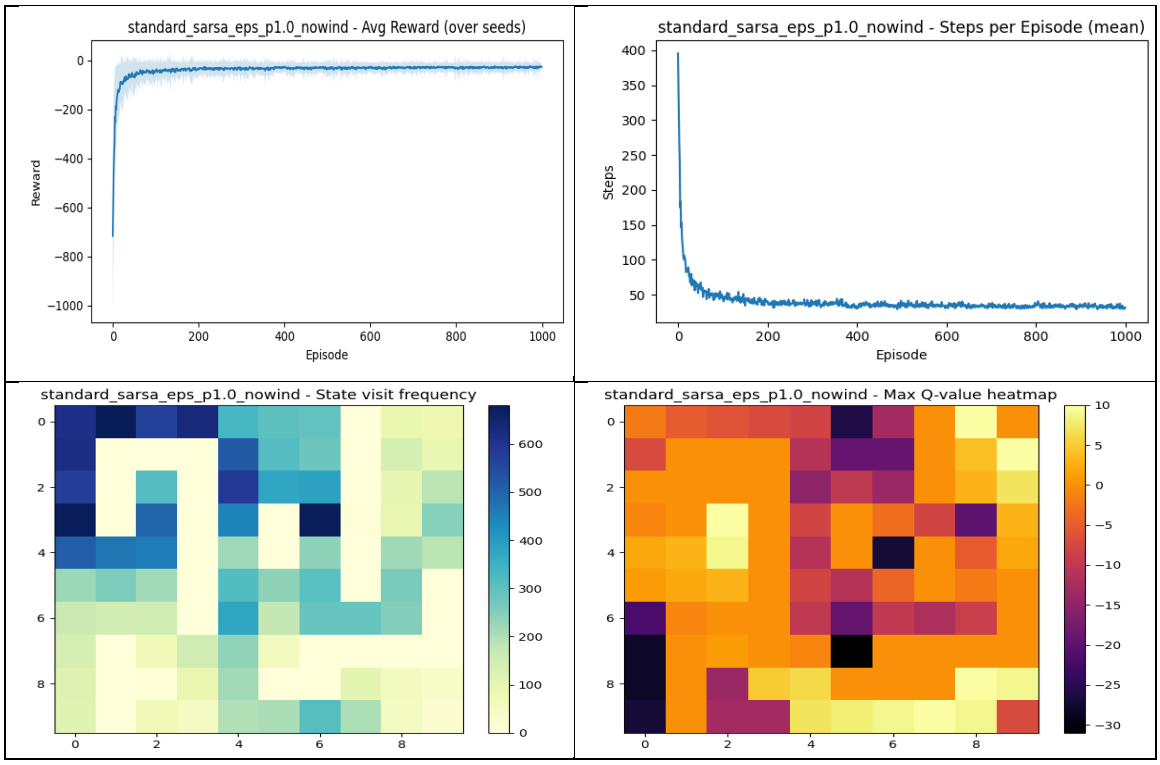
The state-visit heatmap shows strong coverage of the grid early on, but the darker pathways highlight the final preferred route discovered by the agent.

The Q-value heatmap has clear higher-value regions around the optimal corridor, consistent with stable policy learning.

Overall, the learning is smooth and **fully convergent**, with SARSA forming a stable, on-policy solution in a no-wind setting.

standard_sarsa_eps_p1.0_nowind

Tuned parameters: { "alpha": 1.0, "gamma": 1.0, "explore": "softmax", "param": 0.1 }



Analysis:

The average reward rises very quickly from large negative values and stabilizes smoothly, showing that SARSA converges well in the deterministic no-wind environment. With ε = 1.0, the agent explores heavily at the beginning, which explains the noisy initial reward behaviour before settling.

Steps per episode drop sharply within the first ~100 episodes, and afterwards remain consistently low with only minor fluctuations. This indicates that the agent has learned a stable and efficient route to the goal.

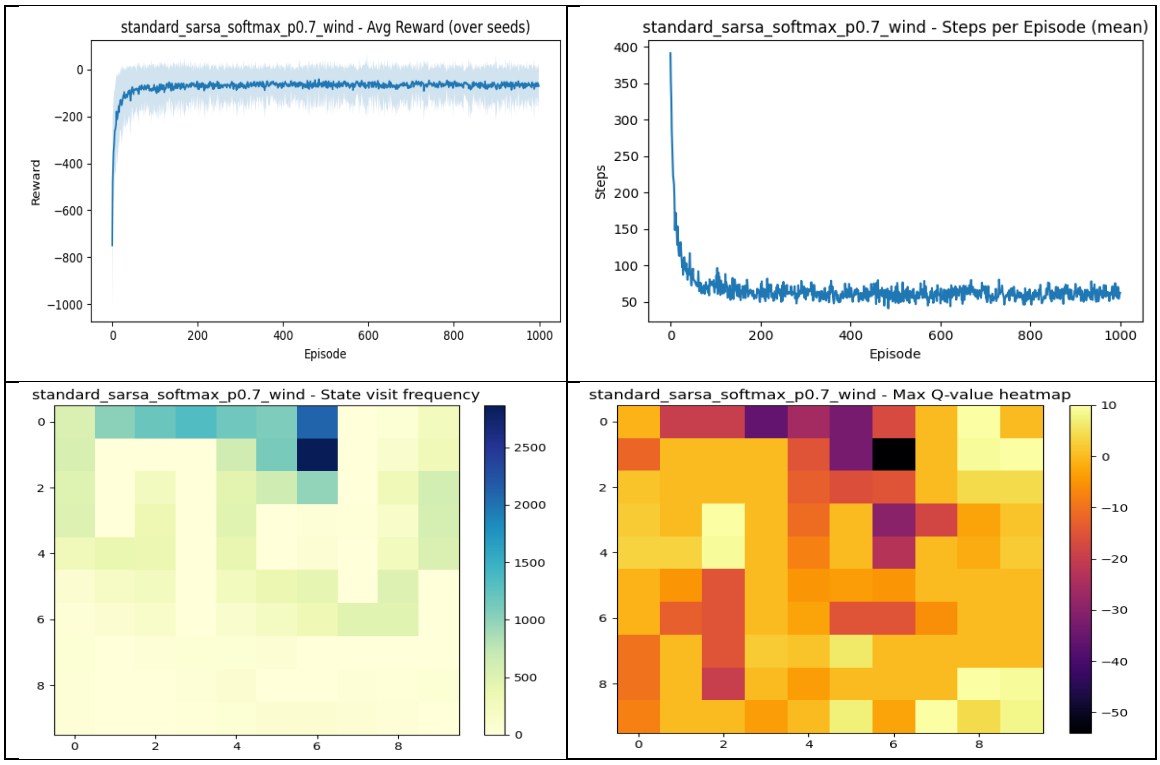
The state-visit heatmap shows a clear dominant path through the grid, reflecting the final learned trajectory. SARSA's on-policy nature leads to a more structured pattern compared to Q-learning.

The Q-value heatmap also shows a well-defined corridor of higher values along this path.

Overall, the behaviour shows clear and full convergence, with SARSA producing a reliable and consistent policy under deterministic conditions.

standard_sarsa_softmax_p0.7_wind

Tuned parameters: { "alpha": 1.0, "gamma": 1.0, "explore": "epsilon", "param": 0.001 }



Analysis:

The reward curve improves quickly from very negative values and then settles around a stable region, although with mild noise due to the wind randomness. Softmax exploration with a lower temperature (p=0.7) makes the agent favour good actions earlier, which explains the relatively smooth convergence.

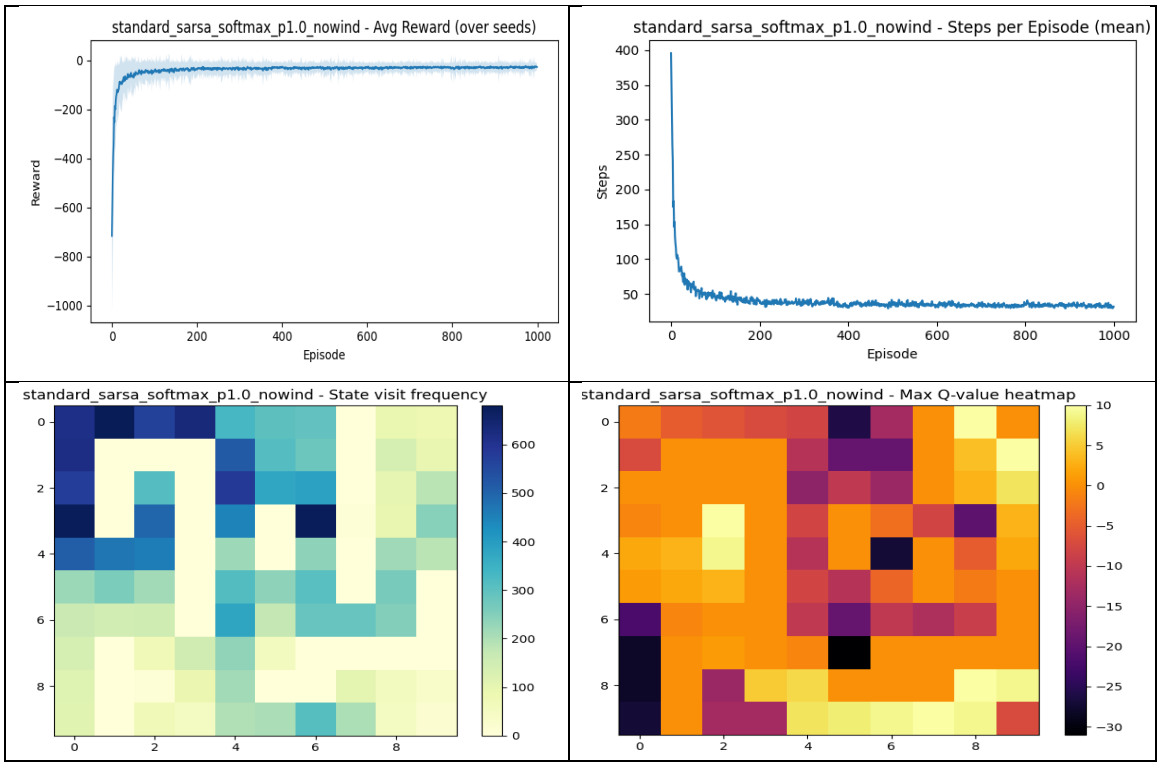
Steps per episode drop sharply in the initial episodes and then stabilise around a small range. However, the wind introduces occasional spikes, so the curve is slightly more noisy compared to the no-wind case. Still, the agent clearly learns a consistent policy.

The state-visit heatmap shows that the agent heavily revisits certain upper-right regions, which matches the grid layout and the influence of wind that often pushes the agent unpredictably. The Q-value heatmap forms a reasonable gradient, but it is not as clean as in deterministic settings because the agent has to account for stochastic drift.

Overall, the agent does converge, but the learned behaviour is slightly less stable due to the wind. SARSA with softmax still manages to find a good policy, but uncertainty in transitions makes the value landscape and visit patterns more scattered.

standard_sarsa_softmax_p1.0_nowind

Tuned parameters: { "alpha": 1.0, "gamma": 1.0, "explore": "softmax", "param": 0.1 }



Analysis:

The reward curve rises quickly from very low values and then stabilises smoothly around a higher reward region. Since there is no wind, the learning process is less noisy, and SARSA with softmax (p = 1.0) gradually converges to a consistent policy.

Steps per episode drop sharply within the first ~100 episodes and eventually settle close to a stable minimum. The near-flat line after convergence shows that the agent reliably reaches the goal using a steady route.

The state-visit heatmap shows a clear, structured path through the grid, with frequent visits along a corridor-like pattern. This indicates that the agent discovers a preferred path and follows it repeatedly after learning stabilises.

The Q-value heatmap displays a smooth gradient, with higher values concentrated near the goal region. This consistency reflects good convergence, with the agent confidently valuing states closer to the goal.

Overall, the agent converges well under deterministic transitions, and softmax exploration enables it to settle on a stable and efficient navigation policy.

7. Important code snippets

1 . **SARSA update rule**

```
Q[state, action] += alpha * (
   reward + gamma * Q[next_state, next_action] - Q[state, action]
)
```

2. **Q-learning update rule**

```
Q[state, action] += alpha * (
   reward + gamma * np.max(Q[next_state]) - Q[state, action]
)
```

3. **Epsilon-Greedy Exploration**

```
def epsilon_greedy_action(Q, state, n_actions, eps):
   if np.random.rand() < eps:
      return int(np.random.randint(n_actions))
   return int(np.argmax(Q[state]))
```

4. **Softmax Exploration**

```
def softmax_action(Q, state, tau):
   q = Q[state] - np.max(Q[state])
   prefs = np.exp(q / tau)
   probs = prefs / prefs.sum()
   return int(np.random.choice(len(q), p=probs))
```

5. **Selecting an Action (dispatch)**

```
def select_action(Q, state, n_actions, explore, param):
    if explore == "epsilon":
        return epsilon_greedy_action(Q, state, n_actions, param)
    else:
        return softmax_action(Q, state, param)
```

 # 6. Episode Loop Used in Both Algorithms

```
for ep in range(episodes):
    state = int(env.reset())
    total_reward = 0

    for t in range(max_steps):
        action = select_action(Q, state, n_actions, explore, param)
        next_state, reward = env.step(state, action)
        next_state = int(next_state)

        # (algorithm-specific update happens here)

        state = next_state
        total_reward += reward
        if reward == 10:
            break
```

 # 7. Tuning Loop (simplified key idea)

```
for alpha in TUNE_ALPHA:
   for gamma in TUNE_GAMMA:
      for explore_type in ["epsilon", "softmax"]:
         for param in param_list:
            run 5 seeds → compute average reward → pick best
```

8. **Averaging Over 100 Seeds (evaluation)**

```
for seed in EVAL_SEEDS:
   set_global_seed(seed)
   run one full training → collect rewards and steps
```

8. Remarks on Runtime and Reproducibility

The complete experiment pipeline executed successfully on a 6-core CPU system with stable performance across all tasks. The code was implemented with efficient sequential execution for consistent performance.

While minor random variations may occur between runs due to environment stochasticity, the results remain qualitatively similar, showing reproducible learning behavior across experiments.

All four tasks -> Task 1 (SARSA), Task 2A (Hyperparameter Tuning), Task 2B (Evaluation), and Extra Metrics Generation were completed in a single continuous run.

All plots were generated under the /results directory, including learning curves, step-per-episode graphs, and state-value heatmaps.

9. Conclusion

This assignment helped deepen my understanding of temporal-difference learning methods and their practical behaviour. SARSA showed smoother and more stable convergence under uncertainty, whereas Q-learning achieved faster policy improvement in more deterministic settings. The tuned hyperparameters and resulting plots confirmed consistent convergence across different environments. Overall, the results matched theoretical expectations, validating both the implementation and the analysis of reinforcement learning performance.

Submission Note

All project files have been included in the zip **except the venv folder** (to reduce size).
The zip contains:

- **RL_Assignment.py** – Main script for Task 1, Task 2A, Task 2B, and extra metrics
- **env.py** – Environment creation functions
- **grid_world.py** – GridWorld environment implementation
- **results folder** – All generated plots (learning curves, steps, heatmaps)
- **best_hyperparams.json** and **partial tuning file**
- **final report (PDF)**
- **README.txt**

Additionally, the full project (same contents as the zip) is uploaded to GitHub for easier viewing.

GitHub Repository Link:

github_link_for_PA_2