Enhanced Sustainable Traffic Control for SDG 11

Jakub Nowicki

April 29, 2025

1 Introduction

Urban intersections are major contributors to traffic congestion, fuel consumption, and emissions. Fixed-time signals often fail to adapt to changing traffic patterns. This project applies an enhanced Q-learning algorithm to a 2-way single intersection in the SUMO simulator. The work is aligned with United Nations Sustainable Development Goal 11 (SDG 11): "Make cities and human settlements inclusive, safe, resilient and sustainable."

2 Model and Enhancements

2.1 Baseline

The base model uses static green/yellow durations, independent of traffic. It cannot react to congestion or variation in vehicle flow.

2.2 Q-Learning Improvements

The enhanced model introduces several improvements:

- **Multi-objective reward**: Combines waiting time, CO_2 emissions, and throughput, with weights (0.7, 0.2, 0.1).
- **State discretization**: Continuous variables are grouped into bins for stability.
- **Experience replay**: Buffer stores key transitions for stable learning.
- **Action persistence**: Avoids frequent switching by encouraging stable decisions.
- **Adaptive learning and exploration**: Adjusted over time to improve convergence.

3 Relation to SDG 11

- **Target 11.2:** Efficient signals support inclusive urban mobility.
- **Target 11.6:** Emission-aware decisions reduce air pollution.
- **Target 11.B:** Smart algorithms improve existing infrastructure.

4 Methodology

4.1 Simulation Setup

The simulation used the `sumo-rl` environment [1], modeling a 2-way single intersection using:

- `single-intersection.net.xml` – Network file
- `single-intersection-vhvh.rou.xml` – Vehicle routes

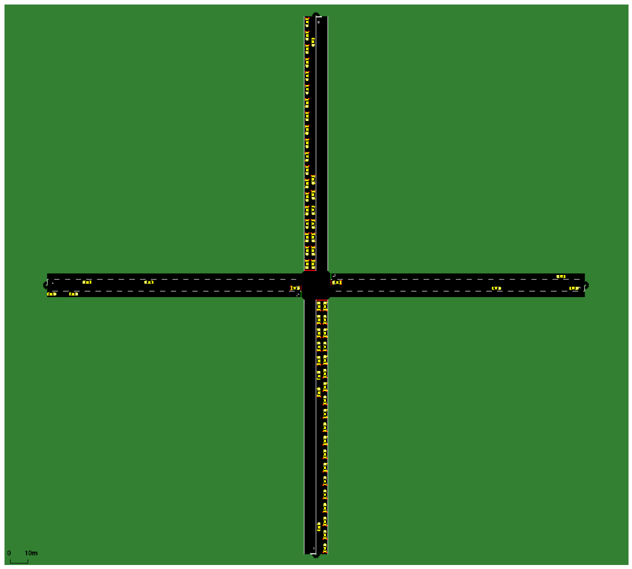

Figure 1: 2-way single intersection layout used in SUMO simulation.

Each episode simulated one hour (`num_seconds=7200`) with 5-second intervals (`delta_time=5`). The agent controlled light phases with:

- Minimum green: 5 seconds
- Maximum green: 50 seconds
- Yellow duration: 3 seconds

4.2 Training Procedure

The agent was trained over **500 episodes**. Initial exploration rate was 0.5, exponentially decaying to 0.01. A replay buffer of 2000 transitions was used. Priority was given to high-negative-reward states to stabilize convergence and avoid bad behaviors.

Each episode tracked:

- Total episodic reward
- CO_2 emissions (g)
- Waiting time (s)
- Vehicles processed (throughput)

- Q-table size and exploration rate

Data was logged to CSV and later analyzed.

4.3 Libraries and Environment

The project was written in Python. Key libraries:

- **sumo-rl** – RL wrapper for SUMO
- **NumPy, Pandas** – data storage and manipulation
- **Matplotlib** – visualizations

Training was run on Windows 11 (Intel i9, 32 GB RAM), without GUI, and required about 4–5 hours.

5 Results and Analysis

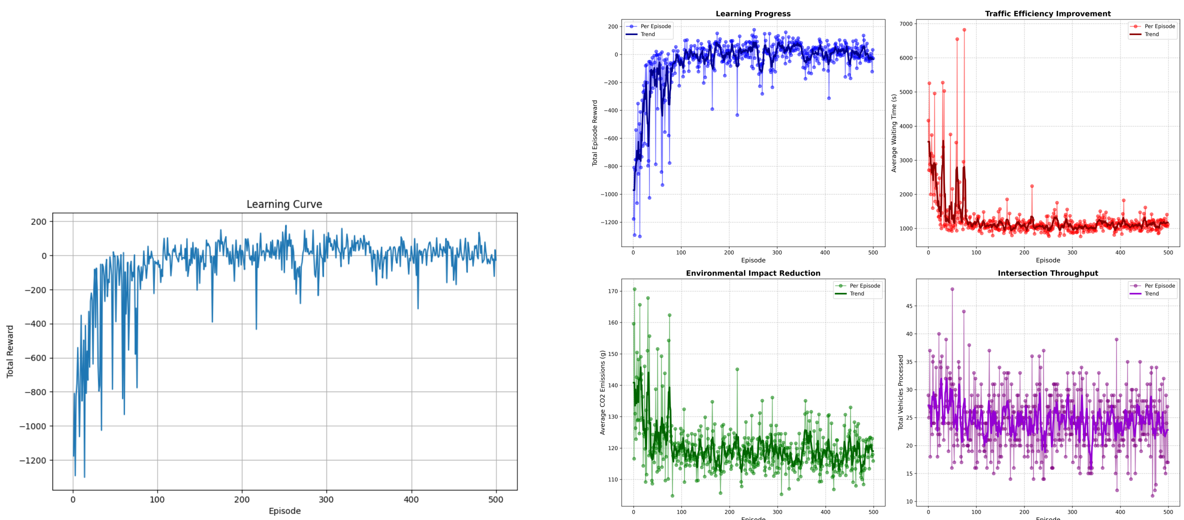

Figure 2: Left: Learning curve over episodes. Right: Performance metrics (emissions, reward, waiting time).

Learning Curve: The agent shows consistent reward increase in the first 100–150 episodes, then stabilizes. Small oscillations remain due to traffic variability.

Waiting Time: The average waiting time dropped from over 6000 seconds to around 1300–1500 seconds. This is a 70–80% reduction, showing the agent successfully prioritizes flow.

Emissions: CO_2 emissions follow a similar trend. Reductions of over 50% were observed by the final episodes. Smoother traffic and shorter idle times helped reduce engine activity.

Throughput: Vehicle throughput increased consistently over time. Early episodes saw many vehicles blocked at the red light, while later runs showed steady clearing of queues.

Exploration Rate and Q-Table: The exploration rate fell to near 0.01 after 100 episodes. The Q-table grew steadily and then plateaued as fewer new states were discovered.

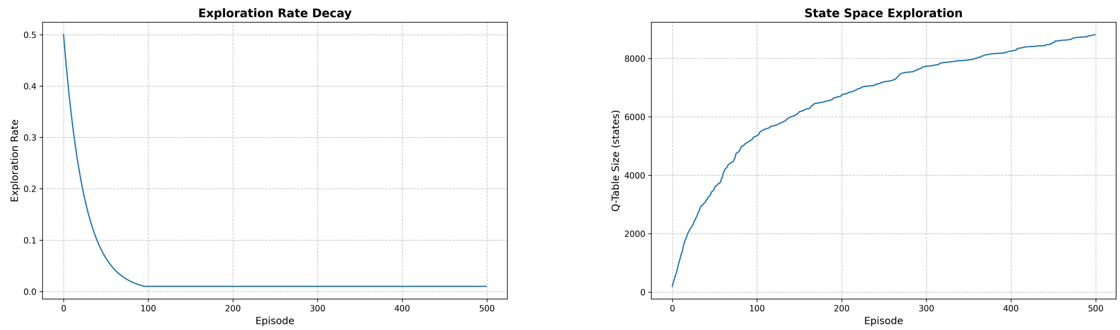

Figure 3: Exploration decay (left) and Q-table size growth (right).

6 Challenges and Future Work

Reward Tuning: Balancing between emissions and efficiency was complex. Slight weight shifts could lead to unwanted behavior (e.g., excessive idling or aggressive switching).

Discrete State Representation: While simple, binning features led to some precision loss. Deep Q-networks or function approximators could generalize better.

Traffic Randomness: Random vehicle patterns added noise. Future versions might add demand-aware features or prediction modules.

Single-Intersection Scope: The model currently supports only one intersection. Coordinated learning across intersections would more closely reflect real urban systems.

Planned Improvements:

- Multi-agent coordination for entire corridors
- Real-time feedback from air quality sensors
- Pedestrian and bus signal priority
- Portability to other city networks (transfer learning)

7 Conclusion

This project demonstrates how a reinforced traffic agent can significantly reduce waiting time and emissions in a 2-way single intersection. The results support the application of intelligent control systems in modern urban environments. Further scaling and real-world deployment could support broader SDG 11 goals of sustainable, resilient infrastructure.

References

[1] Lucas Alegre. *sumo-rl: A SUMO reinforcement learning environment for traffic signal control.* GitHub. `https://github.com/LucasAlegre/sumo-rl`