2026 Volume: 02 Issue: 05 www.ijrpa.com ISSN 2456-9995 Research Article

International Journal Research Publication Analysis

Page: 01-13

TRAFFIC SIGNAL CONTROL SYSTEM USING REINFORCEMENT LEARNING

Dr. Ramya B. N. *[1], **Varun S.[2], Vishal R. Rajput[3]**

[1]Associate Professor, Dept. of Computer Science and Engineering, Jyothy Institute of Technology, Bengaluru, India.

[23]Department of Computer Science and Engineering, Jyothy Institute of Technology, Bengaluru, India.

Article Received: 25 March 2026
Article Revised: 15 April 2026
Published on: 05 May 2026

***Corresponding Author: Dr. Ramya B. N.**
Associate Professor, Dept. of Computer Science and Engineering, Jyothy Institute of Technology, Bengaluru, India.

DOI: https://doi-doi.org/101555/ijrpa.7951

ABSTRACT

Urban traffic congestion has become one of the defining infrastructure challenges of modern cities, driven by rising vehicle ownership and road networks that were never designed to accommodate today's demand. Traditional traffic signal systems rely on fixed-cycle timers or basic threshold rules that are incapable of adapting to the moment-to-moment fluctuations that characterise real-world traffic. This paper investigates the use of Reinforcement Learning (RL) as a principled, data-driven alternative for intelligent traffic signal control at a four-lane intersection. The system is formulated as a Markov Decision Process (MDP) in which an agent observes per-lane vehicle density, queue length, and waiting time, then selects signal phases to maximise a delay-reduction reward. Two RL formulations are evaluated—tabular Q-Learning and Deep Q-Networks (DQN)—and compared against fixed-time, rule-based, and random baselines across multiple performance metrics. The RL agent achieves an average vehicle waiting time of 18.5 seconds, representing a 32 % reduction relative to the fixed-time baseline of 27.2 seconds, and a 22 % improvement over the rule-based system. A dedicated learning-rate sensitivity study using the CartPole benchmark confirms that a moderate learning rate of $\alpha = 0.25$ offers the best balance between convergence speed and training stability. Taken together, the findings demonstrate that RL-based signal control is a scalable, adaptive, and practically viable framework for smart-city traffic management.

KEYWORDS: Reinforcement Learning; Q-Learning; Deep Q-Network; Traffic Signal

Control; Markov Decision Process; Intelligent Transportation Systems; Urban Congestion; Cart Pole.

I. INTRODUCTION

Traffic signals are among the most consequential infrastructure elements in any city. A single poorly timed intersection can propagate delays across an entire corridor, reducing throughput, increasing fuel consumption, and elevating tailpipe emissions. For decades, traffic engineers have addressed this challenge through fixed-cycle signal plans—predetermined sequences of Red, Yellow, and Green phases with durations calibrated to typical demand patterns. Although these plans can be carefully tuned for expected conditions, they are inherently brittle: a minor incident, an unexpected event, or simply an off-peak day can shift demand in ways that fixed plans cannot accommodate.

Adaptive signal control systems emerged as a partial remedy. By reading loop detector data in real time, such systems can extend or truncate phases in response to observed queues. However, most commercially deployed adaptive systems operate on hand-crafted decision trees or linear threshold rules that must be tuned manually and offer limited generalization to scenarios outside their design envelope. They respond to traffic but do not learn from it.

Reinforcement Learning (RL) offers a fundamentally different approach. Rather than specifying a control policy explicitly, RL allows an agent to discover effective behavior through trial and error, guided by a reward signal that quantifies traffic efficiency. The agent is not told how to control the intersection; it learns by observing outcomes and adjusting its strategy accordingly. This capacity for autonomous policy discovery makes RL particularly well suited to the high-dimensional, stochastic, and non-stationary nature of urban traffic.

This paper presents a systematic investigation of RL-based traffic signal control for a simulated four-lane intersection. We formulate the control problem as a Markov Decision Process, implement and train Q-Learning and Deep Q- Network (DQN) agents, and evaluate their performance against three baselines: a fixed-time timer, a rule-based greedy controller, and a random signal selector. In addition, we conduct a learning-rate sensitivity study using the Cart Pole benchmark to provide transferable insights on hyperparameter selection. The objective is to deliver a thorough, end-to-end analysis—from problem formulation through experimental evaluation—that can serve as a practical reference for researchers and engineers working at the intersection of machine learning and intelligent transportation.

The remainder of this paper is structured as follows. Section II surveys related work on traffic signal control and RL-based approaches. Section III describes the methodology, including the MDP formulation, neural network architecture, and training configuration. Section IV presents experimental results with quantitative comparisons. Section V provides a detailed discussion of the findings. Section VI concludes the paper. Section VII outlines directions for future work, followed by the Acknowledgment and References.

II. RELATED WORK

Traffic signal control has attracted research attention for several decades, and the literature spans a wide range of approaches. The earliest and most widely deployed systems use fixed-cycle timing, in which signal phases follow a predetermined schedule. Webster [1] established foundational timing formulae relating cycle length, phase splits, and intersection throughput, and these relationships continue to underpin much practical signal engineering. Fixed-time systems perform reliably when traffic demand is stable and predictable but deteriorate rapidly under uneven or dynamic loading, since no mechanism exists for real-time adaptation.

Actuated control systems were introduced to address the rigidity of fixed-cycle plans. By instrumenting approach lanes with loop detectors or video sensors, actuated systems can extend a green phase while vehicles are detected and terminate it early when a lane clears. While this represents a meaningful improvement over purely fixed timing, actuated systems still operate on threshold rules and do not have any capacity to learn or to plan across multiple cycles. They react to immediate sensor inputs rather than anticipating downstream effects.

Data-driven and model-based adaptive control methods represent a step further. Systems such as SCOOT and SCATS use rolling estimates of traffic flow to continuously re- optimize signal offsets and splits. More recent work has incorporated machine learning for demand prediction, feeding forecast queue lengths into an optimization layer. However, even these approaches rely on hand-crafted objective functions and do not adapt their decision strategy in response to experience.

The application of Reinforcement Learning to traffic signal control was pioneered by Abdulhai et al. [3], who demonstrated that a Q-Learning agent could outperform actuated controllers on simulated arterial networks without any hand-crafted timing rules. Their work established the conceptual foundation for the current paper. Khamis, Gomaa, and Karray [2]

subsequently extended the RL formulation with a multi-objective reward that simultaneously penalizes delay, queue length, and vehicle stops, showing that richer reward design leads to more balanced traffic outcomes. More recent studies have substituted tabular Q-Learning with Deep Q-Networks, allowing the state representation to scale to larger intersections and more complex sensor inputs without the curse of dimensionality that limits tabular methods.

Despite this body of work, important gaps remain. Many RL- based studies optimise a single metric, making it difficult to assess whether gains on one dimension come at the expense of another. Evaluation is often confined to single isolated intersections, leaving scalability to multi-intersection networks unaddressed. Systematic hyperparameter sensitivity studies are rarely reported, so practitioners lack guidance on how to configure RL agents for new deployment scenarios. The present work addresses each of these gaps: it evaluates multiple metrics simultaneously, provides a learning-rate sensitivity analysis using a controlled testbed, and discusses scalability explicitly.

III. METHODOLOGY

A. Problem Formulation as a Markov Decision Process

The traffic signal control problem is formulated as a discrete- time Markov Decision Process (MDP) defined by the tuple (S, A, R, γ). The state space S captures the traffic condition at each lane of the intersection. At every decision step t, the agent observes a state vector $s_t \in S$ composed of the vehicle count, queue length, and instantaneous waiting time for each of the four lanes, yielding a 12-dimensional observation when these three features are concatenated. All raw vehicle counts are normalised to the unit interval [0, 1] prior to being fed to the agent, which promotes numerical stability during gradient-based learning.

The action space A contains one action per feasible signal phase. For a four-lane single-phase intersection, this corresponds to four discrete actions—each assigning the green phase to one lane while the remaining three display red. The agent selects an action $a_t \in A$ at each step, and the simulation advances by applying the resulting signal state for a fixed duration.

The reward function R is designed to penalize vehicle delay directly. After each action, the environment computes the total accumulated waiting time across all lanes, and the reward r_t is defined as the negative of this quantity, so that reducing delay increases the reward. The agent's objective is to find a policy $\pi: S \rightarrow A$ that maximizes the discounted cumulative reward over an episode of T time steps:

$$R_{total} = \Sigma_{t0}^{T} \gamma^t r_t$$

where $\gamma = 0.9$ is the discount factor, which balances the agent's preference between immediate and future rewards. A value of 0.9 causes the agent to weigh future rewards meaningfully without becoming myopic, which is appropriate for a task where the consequences of a single signal decision propagate over several subsequent time steps.

B. Dataset and State Preprocessing

All experiments are conducted in a purpose-built simulation of a four-lane road intersection. Vehicle arrivals are modelled stochastically, with arrival rates drawn from distributions calibrated to generate light, moderate, and peak-congestion scenarios across training episodes. This variability is essential for training a robust policy: an agent trained only on light traffic would not learn to handle peak-hour conditions, and vice versa.

The preprocessing pipeline applies three transformations to the raw simulation output before the state is presented to the agent. First, normalization maps vehicle counts and queue length values to the range [0, 1] by dividing by the maximum observed capacity of each lane, ensuring that no single feature dominates the state vector due to scale differences. Second, noise handling applies to smooth transient spikes in vehicle arrivals, which can otherwise cause the agent to react to measurement noise rather than genuine demand changes. Third, the preprocessed state is formatted as a flat vector suitable for input to the Q-network.

C. Neural Network Architecture for DQN

For the Deep Q-Network variant, the Q-function $Q(s, a; \theta)$— which estimates the expected cumulative reward of taking action a in states under the current parameter vector θ—is approximated by a fully connected feedforward network. The input layer accepts the normalised 12-dimensional state vector. The first hidden layer contains 64 neurons and applies the Rectified Linear Unit (ReLU) activation:

$$f(x) = max(0, x)$$

ReLU is chosen for its computational efficiency, its avoidance of the vanishing gradient problem that afflicts sigmoid and tanh activations in deeper networks, and its empirically strong performance across a wide range of reinforcement learning benchmarks. The second hidden layer contains 32 neurons, again with ReLU. The output layer contains one neuron per action and applies a linear activation, producing an unbounded Q-value estimate for each

candidate signal phase. The network is trained using the Mean Squared Bellman Error loss, optimized with Adam.

Two standard DQN stabilization techniques are employed. Experience replay stores transitions (s_t, a_t, r_t, s_{t+1}) in a circular buffer and samples random mini batches for each gradient update, breaking the temporal correlation between consecutive transitions that would otherwise destabilize training. A target network—a periodically synchronized copy of the Q-network—provides stable Bellman targets by decoupling the network being trained from the network generating those targets.

Method	Avg Wait (sec)	vs Baseline	Rank
Reinforcement Learning	18.5	−32 %	1st
Fixed-Time Baseline	27.2	Reference	2nd
Rule-Based System	23.8	−12 %	3rd
Random Controller	35.4	+30 %	4th

D. Training Configuration

Table I summarizes the hyperparameters used across all RL experiments. The learning rate for the DQN variant is set to 0.01, which was selected following pilot runs. The discount factor $\gamma = 0.9$ is held constant across all experiments. Training runs for a minimum of 500 episodes, with each episode simulating a full intersection cycle under stochastically generated demand. Mini-batches of size 32 are sampled from the replay buffer at each gradient update. Exploration follows an ε-greedy policy in which ε decays linearly from 1.0 to 0.01 over the first 80 % of training, after which it remains at 0.01 for the remainder.

TABLE I: ***RL Training Hyperparameter Configuration.***

Parameter	Value	Purpose
Algorithm	Q-Learning / DQN	Policy optimization
Learning Rate	0.01 (DQN)	Gradient update magnitude
Discount Factor	0.9	Future reward weighting
Episodes	500+	Training duration
Batch Size	32 / 64	Replay sampling
Reward Function	Delay-based	Penalize vehicle waiting time
ε Schedule	1.0 → 0.01	ε-greedy decay
Target Network	Periodic sync	Stable Bellman targets

E. Baseline Control Strategies

Three baseline strategies are used for comparison. Fixed- Time Baseline cycles through all four signal phases with equal durations of a fixed number of seconds per phase, irrespective

of observed demand. This represents thelong-run impact of each action rather than just its immediate effect.

The random controller's average waiting time of 35.4 seconds—30 % higher than the fixed-time baseline— confirms that intelligent decision-making matters even in its simplest form: uniform random phase selection is clearly worse than naive equal-time allocation.

B. Convergence and Learning Behavior

Table III compares the convergence properties and qualitative learning characteristics of the three strategies that involve some form of decision logic. predominant practice in lower-income urban settings and serves as the primary reference point for evaluating improvement. The Rule-Based Controller assigns the green phase at each step to the lane currently reporting the highest vehicle count, implementing a greedy priority rule without any learning. The Random Controller selects signal phases uniformly at random, providing a lower-bound reference that quantifies the performance floor below which no reasonable controller should fall.

IV. EXPERIMENTAL RESULTS & ANALYSIS

A. Traffic Optimization Performance

Table II reports the average vehicle waiting time achieved by each control strategy after training (for the RL agent) or across the full evaluation period (for the baselines). Results are averaged over 100 evaluation episodes to account for the stochasticity of vehicle arrivals.

TABLE II ***Traffic Optimization Performance Comparison.***

Attribute	RL Agent	Rule-Based	Fixed- Time
Convergence Speed	Moderate	Fast	Immediate
Post-Conv. Stability	High	Medium	High
Demand Adaptability	Excellent	Low	None
Learn from Data	Yes	No	No
Handles Unseen Scenarios	Yes (generalizes)	Partially	No

The RL agent clearly leads all other methods, achieving an average waiting time of 18.5 seconds—a reduction of 8.7 seconds (32 %) from the fixed-time baseline. This margin is achieved because the agent has learned, through hundreds of training episodes, to extend the green phase for whichever lane carries the greatest instantaneous load rather than following a rigid schedule. The result is a dynamic, demand- responsive policy that fixed-time systems

cannot replicate by design.

The rule-based system performs better than the fixed-time baseline (23.8 s versus 27.2 s, a 12 % reduction), which confirms that even a simple greedy heuristic—always favouring the most congested lane—outperforms uniform phase allocation. However, the rule-based system lacks any capacity to consider downstream consequences or to recover from self-induced oscillations when two lanes carry similar densities. The RL agent's policy accounts for these dynamics implicitly through the Q-value function, which estimates the

TABLE III ***Convergence and Learning Behavior Comparison.***

The RL agent requires the most time to converge, which is an inherent cost of policy discovery through environment interaction. During the early stages of training, when ε is high and the agent explores widely, average episode rewards are low and noisy. As training progresses and the ε-greedy policy shifts toward exploitation, the reward curve rises monotonically and variance decreases, indicating that the agent is consolidating a stable, effective policy. This learning trajectory—characterized by an initial exploratory phase followed by steady policy improvement—is consistent with healthy Q-Learning convergence and was observed across multiple independent training runs.

The rule-based system converges immediately since it requires no training, but its performance ceiling is limited by the expressiveness of the underlying heuristic. Once the rule is specified, performance is fixed; no amount of operational experience can improve it. Fixed-time control similarly requires no convergence time but cannot improve beyond its pre-calibrated schedule.

C. Learning Rate Sensitivity Study

To isolate the effect of the learning rate from the complexity of traffic dynamics, a dedicated sensitivity study is conducted using the Cart Pole balancing task from OpenAI Gym [4]—a standard RL benchmark with a well-understood reward landscape. Four values of the learning rate α are tested: 0.1, 0.25, 0.75, and 1.0. Each configuration is evaluated over 30 000 episodes, with average reward per episode serving as the performance metric.

At α = 0.1, the agent converges reliably but slowly. The reward curve rises gradually and does not approach near- optimal performance until approximately episode 20 000, making this setting impractical for applications requiring rapid policy acquisition. At α = 0.25, the

agent achieves the best overall outcome: convergence occurs around episode 12 000, the reward curve is smooth with minimal episode-to- episode variance, and the asymptotic reward level is the highest among all settings tested. This configuration represents the optimal trade-off between update speed and stability.

At $\alpha = 0.75$, early convergence is faster than at 0.25, but the reward curve becomes oscillatory beyond episode 15 000, with visible spikes and dips that indicate overshooting of the optimal Q-values. The agent's step sizes are large enough to interfere with previously learned associations, causing performance to fluctuate rather than stabilise. At $\alpha = 1.0$, this instability is severe: Q-values diverge intermittently, the reward remains persistently low, and the variance is high throughout the entire training run. These observations confirm that excessively large learning rates are incompatible with stable policy learning, regardless of the task.

The practical implication for the traffic domain is clear: a learning rate near 0.25 should be used as a default starting point for Q-Learning-based traffic signal controllers, with smaller values reserved for scenarios where stability is paramount and longer training time is acceptable.

V. DISCUSSION

A. Why RL Outperforms Fixed-Time and Rule-Based Control

The performance advantage of the RL agent stems from two complementary properties that neither the fixed-time nor the rule-based system possesses. The first is temporal credit assignment: because the Q-function estimates future discounted reward, the agent evaluates actions not only by their immediate effect on the current time step but also by their downstream consequences across subsequent steps. This allows the agent to, for example, temporarily extend a lightly loaded lane's green phase to prevent a developing queue from growing into a major bottleneck strategy that a greedy rule would never adopt because it ignores future states.

The second property is policy generalization. Because the DQN learns a parametric function approximation over the continuous state space, it can respond sensibly to traffic states it has never seen before, provided they are sufficiently similar to states encountered during training. This is qualitatively different from a rule-based system, which can only act on the specific rules it was given and has no ability to interpolate to novel situations. In practice, this means

the RL agent adapts gracefully to new traffic patterns—such as a sudden peak caused by a nearby event—without requiring recalibration.

B. Trade-offs and Practical Considerations

The RL approach is not without costs. Training requires time and computational resources that rule-based systems do not. In this study, training runs of 500 or more episodes are needed before the agent reaches stable performance, which means that RL controllers cannot be deployed immediately without a prior offline training phase. This requirement for training infrastructure represents a practical barrier to adoption in resource-constrained settings.

Additionally, the quality of the learned policy depends heavily on the fidelity of the simulation used for training. If the simulation does not accurately capture real-world traffic dynamics—such as driver behavior, incident effects, or pedestrian interference, the agent may learn a policy that is well-optimized for the simulator but performs poorly in deployment. This sim-to-real gap is a known challenge in applied RL and would need to be addressed through high- fidelity simulation, domain randomization, or real-world fine-tuning before field deployment.

C. Implications for Scalability

The present experiments focus on a single isolated intersection, which is the natural starting point for demonstrating the viability of RL-based signal control. Extending the framework to multi-intersection networks introduces additional challenges. At the most basic level, the state space grows with the number of intersections, potentially exceeding the capacity of a centralized single- agent approach. Multi-agent RL frameworks, in which each intersection hosts its own learning agent while communicating with its neighbors, offer a natural solution and have been explored in the literature. The policy and training configuration developed in this paper could be extended to such a setting with relatively modest modifications to the reward function, for example, incorporating a cooperative term that penalizes queue propagation across intersections.

D. Reward Design and its Consequences

The delay-based reward function used in this study directly aligns the agent's objective with the outcome that matters most to road users: minimizing time spent waiting. This alignment is one of the strengths of formulation. An alternative reward design—for example, maximizing throughput rather than minimizing delay—could yield different policy

characteristics, potentially prioritizing high- flow lanes at the expense of lower-flow ones. The choice of reward function is therefore a design decision with real-world consequences, and practitioners adapting this framework to new settings should consider carefully which metric or combination of metrics best reflects the goals of the deployment context.

VI. CONCLUSION

This paper has demonstrated, through simulation experiments and systematic analysis, that Reinforcement Learning provides a compelling and practically viable approach to adaptive traffic signal control. The RL agent trained in this study achieved an average vehicle waiting time of 18.5 seconds at a four-lane intersection—a 32 % reduction compared to a fixed-time baseline and a 22 % improvement over a greedy rule-based controller. These margins are not marginal; they translate, at real-world scale, to meaningful reductions in fuel consumption, vehicle emissions, and commuter stress.

The key enabler of this performance is the agent's ability to learn a temporally aware policy through interaction with the environment, rather than being programmed with explicit rules. By estimating the long-run value of each signal decision, the agent naturally avoids the myopic failures of greedy heuristics and the rigidity of fixed-cycle timing. The learning-rate sensitivity study on Cart Pole provides a concrete and transferable finding: a moderate value of $\alpha = 0.25$ consistently delivers the best trade-off between convergence speed and stability across the range tested, while values of 0.75 and above produce oscillation or divergence.

Taken together, the results confirm that RL is not merely a theoretically attractive framework for traffic signal control but a practically effective one that can be trained, evaluated, and interpreted within a realistic simulation workflow. The methods and findings presented here provide a solid foundation for further development toward real-world deployment.

VII. FUTURE WORK

Several promising directions emerge from this work. The most immediate extension is the integration of real-time vehicle detection using computer vision, replacing the simulated vehicle count with measurements derived from intersection cameras via OpenCV or similar frameworks. This would allow the trained policy to be applied to real traffic streams without requiring loop detector infrastructure.

A second priority is scaling from the single-intersection scenario to coordinated multi-

intersection networks. Multi- agent RL approaches, where each agent governs one intersection and shares state information with its neighbors, are the natural next step. Designing reward functions that balance local and global objectives in such settings is a non-trivial research challenge that warrants dedicated investigation.

Third, exploring state-of-the-art deep RL algorithms such as Proximal Policy Optimization (PPO), Advantage Actor- Critic (A2C), and Soft Actor-Critic (SAC) in place of DQN could yield further performance improvements, particularly in settings with larger state and action spaces. These policy- gradient and actor-critic methods have demonstrated advantages over Q-Learning-based approaches in several continuous control benchmarks and may offer similar benefits for traffic signal control.

Finally, embedding IoT-based sensing infrastructure— including infrared detectors, ultrasonic sensors, and connected vehicle data feeds—would enrich the state representation available to the agent and enable cloud-based traffic monitoring at city scale. Emergency vehicle pre- emption, which currently requires dedicated hardware, could also be integrated into the RL framework as a constrained optimization component, ensuring that learned policies never delay emergency responders while still optimizing for general traffic efficiency.

ACKNOWLEDGMENT.

The authors gratefully acknowledge the Department of Computer Science and Engineering at Jyothy Institute of Technology, Bengaluru, for the academic environment and institutional support that made this research possible. The authors also thank Dr. Prabhanjan S, Head of Department, for providing the opportunity and resources to carry out this investigation, and Dr. Gopalakrishna K, Principal, for ensuring access to adequate computing facilities throughout the project. The open-source communities behind Python, PyTorch, OpenAI Gym [4], and Matplotlib [5] provided indispensable tools without which the experimental work could not have been conducted.

REFERENCES.

1. R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction, 2nd ed. Cambridge, MA, USA: MIT Press, 2018.
2. M. A. Khamis, W. Gomaa, and F. Karray, “Adaptive multi-objective reinforcement learning with hybrid exploration for traffic signal control based on cooperative multi-

agent framework," Engineering Applications of Artificial Intelligence, vol. 29, pp. 134–151, Mar. 2014.

3. B. Abdulhai, R. Pringle, and G. J. Karakoulas, "Reinforcement learning for true adaptive traffic signal control," Journal of Transportation Engineering, vol. 129, no. 3, pp. 278–285, Jun. 2003.
4. OpenAI, "Gym: A toolkit for developing and comparing reinforcement learning algorithms," 2023. [Online]. Available: https://gym.openai.com.
5. J. D. Hunter, "Matplotlib: A 2D graphics environment," Computing in Science & Engineering, vol. 9, no. 3, pp. 90–95, 2007.