Received: 15 February 2023 | Revised: 21 March 2023 | Accepted: 23 March 2023

DOI: 10.1111/exsy.13301

ORIGINAL ARTICLE

Expert Systems WILEY

A citywide TD-learning based intelligent traffic signal control for autonomous vehicles: Performance evaluation using SUMO

Selim Reza[1] | Marta Campos Ferreira[1] | J. J. M. Machado[2] | João Manuel R. S. Tavares[2]

[1]Faculdade de Engenharia, Universidade do Porto, Rua Dr. Roberto Frias, 4200-465 Porto, Portugal

[2]Departamento de Engenharia Mecânica, Faculdade de Engenharia, Universidade do Porto, Rua Dr. Roberto Frias, Porto 4200-465, Portugal

Correspondence
João Manuel R. S. Tavares, Departamento de Engenharia Mecânica, Faculdade de Engenharia, Universidade do Porto, Rua Dr. Roberto Frias, Porto 4200-465, Portugal.
Email: tavares@fe.up.pt

Funding information
Fundação para a Ciência e Tecnologia (FCT), Grant/Award Number: 2022.12391.BD

Abstract

An autonomous vehicle can sense its environment and operate without human involvement. Its adequate management in an intelligent transportation system could significantly reduce traffic congestion and overall travel time in a network. Adaptive traffic signal controller (ATSC) based on multi-agent systems using state-action-reward-state-action (SARSA (λ)) are well-known state-of-the-art models to manage autonomous vehicles within urban areas. However, this study found inefficient weights updating mechanisms of the conventional SARSA (λ) models. Therefore, it proposes a Gaussian function to regulate the eligibility trace vector's decay mechanism effectively. On the other hand, an efficient understanding of the state of the traffic environment is crucial for an agent to take optimal actions. The conventional models feed the state values to the agents through the MinMax normalization technique, which sometimes shows less efficiency and robustness. So, this study suggests the MaxAbs scaled state values instead of MinMax to address the problem. Furthermore, the combination of the A-star routing algorithm and proposed model demonstrated a good increase in performance relatively to the conventional SARSA (λ)-based routing algorithms. The proposed model and the baselines were implemented in a microscopic traffic simulation environment using the SUMO package over a complex real-world-like 21-intersections network to evaluate their performance. The results showed a reduction of the vehicle's average total waiting time and total stops by a mean value of 59.9% and 17.55% compared to the considered baselines. Also, the A-star combined with the proposed controller outperformed the conventional approaches by increasing the vehicle's average trip speed by 3.4%.

KEYWORDS

adaptive traffic signal controller, agent-based learning, average waiting time, eligibility traces, gaussian distance function, SARSA (λ)

1 | INTRODUCTION

Intelligent transportation systems (ITS) have evolved as a critical component of traffic management solutions, with researchers working to improve their accuracy and efficiency. One of the essential aspects of ITS is intersection traffic signal control. Effective intersection traffic signal controlling schemes, especially in over-saturated conditions, require actions based on the dynamic traffic factors of the corresponding and neighbouring junctions and proper policy implementation. Several researchers proposed different techniques to address these problems. For example, the fixed time (FT) controllers, which use previous data to determine the optimal timing of traffic lights, are the most extensively used technique. However, this strategy cannot meet current traffic-stochastic needs or deal with unanticipated traffic circumstances (Osorio & Wang, 2017). Because of the limitations of FT controllers, Webster's technique was developed, which uses inductive detectors to observe actual traffic conditions and efficiently extend or terminate the green signal time by detecting the spacing between vehicles. However, it needs to pay attention to the accumulated information of the traffic states, resulting in a lack of performance in terms of accuracy and efficiency (Eriskin et al., 2017).

The adaptive traffic signal control (ATSC) technique appeared in this context. The Sydney coordinated adaptive traffic system (SCAT; Sims & Dobinson, 1980) and split, cycle, and offset optimisation technique (SCOOT; Hunt, 1981) both use adaptive systems to overcome the drawbacks of previous methods by collecting real-time traffic flow data at each intersection to control traffic signal timing effectively. The SCAT systems count vehicles at each stop line, whereas SCOOT uses an upstream network of advanced detectors to obtain traffic data. Using these detectors, SCOOT gives a higher resolution of current traffic conditions, such as traffic flow and the number of automobiles in the queue before they reach the stop line. The SCAT and SCOOT use centralized control techniques, with systems running locally, and intersection coordination is achieved through communication among neighbours. When a junction releases multiple vehicles, it notifies the next intersection of the time and quantity of vehicles to expect. However, the effectiveness of such approaches is highly dependent on the position and reliability of the detector.

Deep-learning (DL) models recently exhibited their worthiness in self-ATSC schemes, demonstrating significantly improved accuracy and resilience (Bouktif et al., 2021). It is commonly known that DL models can extract essential information from raw data. On the other hand, reinforcement Learning (RL) data differs significantly from traditional DL datasets, which contain massive amounts of hand-labelled data from a defined, independent distribution. Additionally, RL data only has noisy, sparse and delayed rewards. The data is also interconnected, and as the agent learns, the data distribution changes. Then, there is the question of whether or not deep reinforcement learning (DRL) can serve the same purpose. Is it possible for an agent to learn from raw data with many dimensions? Q-Learning models are most suited to mitigate these concerns where the agent learns from several recent encounters.

ATSC using state-action-reward-state-action (SARSA (λ)) algorithm has been applied by numerous researchers (Alegre et al., 2022; Aziz et al., 2018; Ziemke et al., 2021), and (Kekuda et al., 2021). Aziz et al. (2018) introduced an online SARSA (λ) model and demonstrated its goodness concerning the off-line SARSA (λ) based models. However, the state space is generally huge in the case of traffic, and the proposed model failed to deal with the enormous dimensionality problem of the traffic states. The authors of Ziemke et al. (2021) applied the SARSA (λ) with Fourier basis function approximation technique to address the problem and demonstrated a tremendous improvement in performance for a single intersection compared to the FT controller and traditional SARSA (λ) based models. Kekuda et al. (2021) proposed an n-step SARSA model for a 4-junction network considering four types of vehicles. The proposed algorithm helped the agent predict circumstances in the far future. Nevertheless, it suffered from a convergence problem resulting in learning a substandard policy. Furthermore, a more robust controlling scheme must consider multiple intersections; hence, in Alegre et al. (2022), the authors employed a similar approach in a city-wise traffic network to address this problem by considering a city network with a 22 signalized intersection to demonstrate its effectiveness. A parameter λ valued in the (0–1) range provides the decaying of the eligibility trace vector. This parameter influenced weight updates, resulting in an inefficient updating process.

The new model proposed in this article introduces a Gaussian decaying mechanism to address the aforementioned problem. In addition, the previous models employed MinMax (Patro & Sahu, 2015) normalization on the traffic state values, resulting in a lack of resilience. Because it may normalize within the $(-1,1)$ range with the arrival of any negative state-space component, this study justifies the usefulness of the MaxAbs (Ogasawara et al., 2010) technique instead of the conventional approaches to address the problem and enhance further the performance.

Routing algorithms aim to maximize the likelihood of arriving on time when travelling between two locations within a specific time budget. Compared to traditional algorithms, the A-star and Dijkstra routing algorithms, although old, can significantly boost the chance of on-time arrival (Niknami & Samaranayake, 2016). This article proposes a SARSA (λ)-based multi-agent TD-Learning ATSC on handling autonomous vehicles operating on A-star and Dijkstra routing algorithms in an urban environment. Compared to conventional SARSA (λ) and FT controller-based techniques, combining the A-star routing algorithm and the proposed multi-agent TD-Learning algorithm revealed a significant performance improvement. The main contributions of this study are as follows:

- It proposes a Gaussian approach to regulate the decaying mechanism of the eligibility trace vector for enhancing weights updating efficiency.
- It also implements a real-world-like simulation environment by considering seven classes of vehicles: car, bus, truck, delivery vehicle, motorcycle, bicycle, and an emergency vehicle, with varying characteristics to justify the model's performances.
- It exhibits the worthiness of MaxAbs normalization techniques instead of conventional MinMax to obtain the traffic state values.

- It also demonstrates a further enhancement of autonomous vehicle trip performance by combining the A-star routing algorithm with the currently proposed traffic signal controlling approach.

The rest of this article is organized as follows: Section 2 presents a summary of selected state-of-the-art works along with their performances and limitations; the proposed model formulation is given in Section 3; the experimental setup and results are presented in Section 4; Section 5 is devoted to discuss the overall performance and feasibility of the proposed model in real-world like environments, and finally, conclusions are drawn in Section 6.

2 | RELATED WORKS

The ATSC problems can be solved using different techniques. For example, vision-based algorithms can potentially address tasks in a manageable way. However, such models require complete traffic scene understanding, which needs more improvements to provide optimal outcomes (Muhammad et al., 2020, 2022). Hence, the multi-agent RL techniques are the current state-of-the-art in managing vehicles within road networks. Many existing traffic-control systems require a preset model of the traffic environment to ensure optimal performance. The relationship between actions, states, and the environment is learned by interaction with the environment in Q-Learning, which does not require a predefined environment model. One of the benefits of RL algorithms is that they are adaptable and may respond to dynamic sensory inputs and a dynamically changing environment through continuous learning and adaptation. The one-step Q-learning algorithm is adaptable to inline real-time learning since it updates the Q-estimates at short intervals in conjunction with each action.

Q-Learning is also an off-policy algorithm because it learns significant knowledge while experimenting with behaviours that may be suboptimal later. With promising results, Abdulhai et al. (Abdulhai et al., 2003) were among the first to use Q-Learning for traffic signal control in congested intersections. Using the green light district (GLD) simulator, Wiering et al. (Wiering et al., 2004) proposed an adaptive optimisation approach based on RL. The authors compared it to nonadaptive controllers, exhibiting incredible performance enhancement, particularly for light and medium traffic states. However, for over-saturated conditions, the results also showed a performance degradation. In the work of Wunderlich et al. (2008), a longest-queue-first maximal-weight-matching (LQF-MWM) method demonstrated its goodness in ensuring optimal performance. It outperformed existing baselines in high-load scenarios to maximize throughput. It can be noted that these works are limited to one or two isolated intersections and hence, lack robustness.

Arel et al. (2010) presented a multi-agent RL technique to study a five-junction traffic network, where an autonomous intelligent agent managed each intersection. The experimental results indicated the advantages of multi-agent-RL-based control over LQF-based methods. Prashanth and Bhatnagar (2011) provided another approach in order to improve the performance by incorporating multiple timescales stochastic approximation in a policy gradient actor-critic algorithm, which outperformed typical Q-learning algorithms. A reasonably extensive network was modelled using multi-agent systems by Abdoos et al. (2013), which explored Q-learning and holonic Q-learning approaches to control traffic signals. The experimental results exhibited an improved performance of holonic Q-learning in limiting over-saturation, reducing average latency, and increasing throughput.

Coordination among agents is essential to enhance robustness, particularly for over-saturated intersections. Aziz et al. (2018) investigated information sharing among all signal controllers by presenting an R-Markov Average-Reward-Technique-based RL (RMART) algorithm that outperformed in over-crowded scenarios and dramatically reduced traffic emissions. Also, Li and Xu (2021) introduced the information-sharing deep deterministic policy gradient (KS-DDPG) algorithm, demonstrating substantial efficiency in controlling large-scale networks and coping with traffic flow variations when considering knowledge sharing among agents. The need for further enhancement in robustness and accuracy prompted (Wang et al., 2021) to develop a cooperative group-based multi-agent reinforcement learning-ATSC (CGB-MATSC) framework, which reduced the average waiting time by 42.08% compared to the FT controller. Therewithal, an n-step SARSA algorithm using linear function approximation method exhibited a 5.5% reduction in average queue length compared to LQF-based approaches (Kekuda et al., 2021). However, the proposed model suffered from a serious convergence problem resulting in learning a substandard policy. The literature provides some solutions to the problem through traffic phase duration-aware approaches. For example, Zhao et al. (2022) presented a traffic intensity and phase duration-aware RL approach, which showed the potentiality to address these drawbacks.

Yen et al. (2020) proposed a deep duelling SARSA model based on a reward function defined by the ratio of the throughput and average delay, which outperformed the DQN and Deep SARSA-based algorithms. In Antes et al. (2022), the main focus was improving the agent's decision-making accuracy by proposing a hierarchical information-sharing scheme based on SARSA. The experimental results shown improved performance compared to the FT and convention SARSA models. Besides, automatic goal generation models provide promising outcomes in dealing with the ever-changing traffic environments and unseen traffic events to enhance ATSC robustness (Ghanadbashi & Golpayegani, 2022).

The methods aforementioned still require improvements in terms of robustness and accuracy, particularly for over-saturated conditions and with the arrival of non-recurrent events like accidents and other emergencies. The main focus of the proposed model was to achieve further performance improvements of the SARSA (λ) by implementing a Gaussian decaying mechanism of the eligibility trace vector for addressing efficient weights updating problems. Also, to increase the agent's decision-making abilities, the MaxAbs scaling of state values was incorporated in the algorithm.

3 | METHODOLOGY

An agent performs specific actions in the environment, so the state it is in will change accordingly. Also, every action taken will get a reward with the evolution of states. The agent will learn about the environment through the process and understand what action will lead to good or bad rewards. A pre-programmed traffic signal control, such as an FT controller, changes the signalling state according to a predefined time budget. On the other hand, an RL-based traffic signal controller learns the signalling scheme through its experiences by taking actions and getting rewards.

If S, a, R, and γ represent state, action, rewards, and discount factor, which determines the number of rewards an RL agent would get in the far future compared to the near future, respectively, then, according to the Bellman equation (Modares et al., 2014), the value of a state, $V(S)$, can be calculated as:

$$V(S) = \max_a (R(S,a) + \gamma V(S')), \tag{1}$$

where $V(S')$ represents the value of following S' state. An agent can take many actions, but only the optimal action is to be considered; hence, *max* is used. When an agent takes action for a given state, it will get a reward, which can be positive, negative or zero, and get into a new S' state. So, three separate Bellman equations will be built for three possible actions, that is, turning the signal red, green, or yellow.

There are two kinds of search: deterministic and non-deterministic. In a deterministic search, if an agent is decided to go to a particular state, it will do it with 100% of probability. While the non-deterministic search is a stochastic process, there are generally multiple options (Sutton & Barto, 2018); for example, if it wants to change the traffic state into green, there would be 10% of probability for red or yellow, and depending on the problems, the randomness could be different, which resembles reality in a much comprehensive manner. This study constructed the Q-learning agents to perform the non-deterministic search mechanism. The plan is simply like an artificial intelligence treasure map problem. However, the non-deterministic search mechanism has no predefined plan because the agent cannot know what will happen next. Hence, instead of a plan, generally, the policy is best suited for the current purposes. The Q-learning agents will learn the best policy to deal with an environment throughout the training process.

3.1 | Markov decision process

A Markov Process can be described as a property where the conditional probability distribution of a future state, that is, the action to be taken by the agent, only depends on its present state, not the sequence of events that preceded it. On the other hand, a Markov decision process (MDP) is a mathematical framework that the agents can use to understand what to do in a stochastic environment. Mathematically, the MDP is just an add-on in the Bellman equation (Otterlo & Wiering, 2012). Equation (1) is for a deterministic search where the value of the future S' state is certain. However, in a stochastic process, S' cannot be fully determined, and hence, Equation 1 needs some modification:

$$V(S) = \max_a (R(S,a) + \gamma (P_1{}^* V(S_{1'}) + P_2{}^* V(S_{2'}) + \ldots.)), \tag{2}$$

where $P_1, P_2, \ldots.$ represent the probability of getting into $V(S_{1'}), V(S_{2'}), \ldots.$ states, respectively. To be more formal, Equation (2) can be written as:

$$V(S) = \max_a \left(R(S,a) + \gamma \sum_{S'} P(S,a,S') V(S') \right). \tag{3}$$

Equation (3) is the framework that the agents will use to solve the non-deterministic search problems where random events, which cannot be controlled, are happening.

3.2 | Q-Learning

Instead of looking at the values of each state, the agent can consider the values of each action. Hence, $Q(s,a)$ values come into play, where Q represents the quality of action rather than the value of the state. Because actions lead to states, there is some relationship between the values of the states and Q-value functions. When an agent takes action, it will get a $R(S,a)$ reward and will end up in a new state, which is stochastic and can be quantified by the expected $\gamma\sum_{S'} P(S,a,S')V(S')$ value function. From Equation (3), which represents the $V(S)$ value function of a S state, $Q(S,a)$ can be formulated both as (van Hasselt et al., 2016):

$$Q(S,a) = R(S,a) + \gamma\sum_{S'} P(S,a,S')V(S'), \tag{4}$$

$$Q(S,a) = R(S,a) + \gamma\sum_{S'} P(S,a,S')\max_{a} Q(S',a'). \tag{5}$$

3.2.1 | Temporal difference

Temporal distance (TD), which may be defined as the difference in Q values at different times, allows an agent to calculate Q values in a non-deterministic environment. From the Bellman equation of Q values (Equation 5), the value of performing a particular action in a specific state by an agent depends on the rewards after completing the action and the summation of the expected values of the states it may end-up. Before taking action, assume that Q value is $Q(S,a)$. After an action is taken, the Q value can be calculated using Equation (5). TD is defined by the difference between these two values and can be formulated as (Taylor et al., 2006):

$$TD(a,S) = R(S,a) + \gamma\sum_{S'} P(S,a,S')\max_{a} Q(S',a') - Q(S,a). \tag{6}$$

Is there any difference between Equation (5) and (6)? Ideally, both equations should be the same. Nevertheless, the fact is that $Q(S,a)$ is previously known, and the remaining one is calculated after taking action. So, there is no guarantee that both equations are the same because of the randomness of the environment. The agent learns through these differences as can be formulated by:

$$Q_t(S,a) = Q_{t-1}(S,a) + \alpha TD_t(a,S), \tag{7}$$

where α is the learning rate between 0 (zero) and 1 (one). Equation (7) is the heart of the Q-learning algorithm, which indicates how the Q values are updated. The algorithm is said to be converged when the TD starts becoming closer and closer to 0 (zero).

3.3 | Proposed algorithm

The proposed algorithm is an improvement of the SARSA-TD Q-learning algorithm, mainly in terms of policy, where the target policy is the same as the behaviour policy. Two successive state-action pairs and the agent's immediate reward while transitioning states determine the new Q value. In traditional Q-learning, the target policy is always greedy; in the proposed method, the target policy is selected based on the behaviour policy of a certain state. Hence, Q values are calculated according to:

$$Q(S,a) = R(S,a) + \gamma\sum_{S'} P(S,a,S')Q(S',a'). \tag{8}$$

The policy assessment mechanism is $TD(\lambda)$, which is easily transformable into a control method. Regarding learning, predicting the expected return is based on the state and the action rather than just the status. More precisely, rather than learning an estimation of the state-value function, v, the model learns an estimate of the action-value function, q. Also, the policy is based on action-value estimations rather than a fixed policy that generates the behaviour. Because these estimations tend to improve with time, the policy also does. An exploration technique usually obtains precise estimates for all state-action values. This research employed an $\varepsilon - greedy$ behaviour policy. Algorithm 1 explains the proposed model, where ψ is an action-feature vector, and $\theta^t\psi$ is a linear estimate of q_π at t time step. A common way to derive a ψ action-feature vector from a state-feature vector involves an action-feature vector of $n|A|$ size, where n is the number of state features and $|A|$ is the number of actions. Each action corresponds to a block of n features in this action-feature vector. The features in ψ correspond to an action taken according to the values of the state features; the features corresponding to other actions have a value of 0 (zero).

After completion of an action, the weight is updated following Equation (9). It uses the gradient descent with linear function approximation rule, which in this case is a 7^{th} order Fourier approximation, to update θ weight according to:

ALGORITHM 1 **Proposed algorithm** (λ)

```
Require: Input $(\alpha, \lambda, \gamma, \theta_{init})$
   $\theta \leftarrow \theta_{init}$;
   $Loop\ (over\ episodes)$;
   $Obtain\ initial\ state\ S$
   $Select\ action\ A\ based\ on\ state\ S$
   $\psi \leftarrow features\ corresponding\ to\ S, A$
   $e \leftarrow Gaussian\ Random; Q_{old} \leftarrow 0$
   while terminal state has not been reached do
      $take\ action\ A, observe\ next\ state\ S'\ and\ reward\ R$
      $select\ action\ A'\ based\ on\ state\ S'$
      $\psi' \leftarrow features\ corresponding\ to\ S', A'$
   $(when\ S'\ is\ terminal\ state, \psi' \leftarrow 0)$
      $Q \leftarrow \theta^t \psi$
      $Q' \leftarrow \theta^t \psi'$
      $TD \leftarrow R + \gamma Q' - Q$
      for A in range action-dimension do
         if A == action then
            $e \leftarrow \gamma\lambda e + \psi - \alpha\gamma\lambda \cdot (e^T \psi)\psi$
            $\theta \leftarrow \theta + \alpha(TD + Q - Q_{old})e - \alpha(Q - Q_{old})\psi$
         else
            $e \leftarrow \gamma\lambda e$
            $e \leftarrow exp(-A^2/1000)e$
            $\theta \leftarrow \theta + \alpha(TD + Q - Q_{old})e$
         end if
      end for
      $Q_{old} \leftarrow Q'$
      $\psi \leftarrow \psi'; A \leftarrow A'$
end while.
```

$$\theta = \theta + \alpha(TD + Q - Q_{old})e - \alpha(Q - Q_{old})\psi, \tag{9}$$

where e is the eligibility trace vector, generally used to serve the credit assignment problem, and is updated using:

$$e = \gamma\lambda e + \psi - \alpha\gamma\lambda \cdot (e^T \psi)\psi. \tag{10}$$

Each weight update includes data from previously visited states attributed to the values acquired by e vector, and λ controls its decaying mechanism. This study proposes a Gaussian distance decaying mechanism for decaying the eligibility traces vector according to:

$$e = exp\left(-A^2/1000\right)e. \tag{11}$$

Figure 1 illustrates the difference between the conventional and proposed eligibility traces decaying mechanism. The convention approaches use a λ parameter within the $(0, 1)$ range to control the decaying operation, which follows sharp decay and rapid saturation with time. This effect causes a less effective weight-updating process. On the contrary, the proposed Gaussian decaying technique is much more productive in controlling the weight-updating process because it can efficiently regulate the decaying mechanism.

The flowchart of the proposed model is illustrated in Figure 2 to understand its working mechanism better. The state space comprises the information on the current signalling phase and its elapsed time divided by the maximum green time, vehicle lane density, and vehicle queue

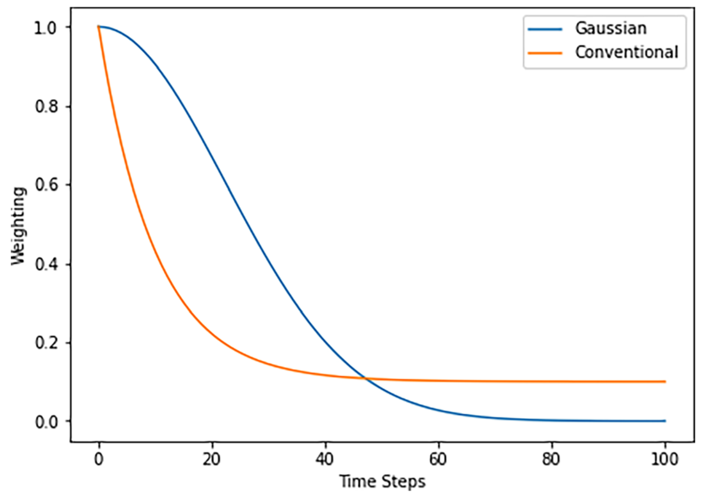


FIGURE 1 Simplified illustration of the proposed Gaussian eligibility trace decaying mechanism

length. The agents observe this information as a vector in every time step. The state-space vector is normalized before feeding to the agents. Conventional SARSA (λ) uses MinMax normalization technique to standardize those values using (Al Shalabi et al., 2006):

$$MinMax_X = \frac{X - X_{min.}}{X_{max.} - X_{min.}}. \tag{12}$$

However, one of the major disadvantages of MinMax is that it normalizes features within $(-1,1)$ if the state space vector component contains any negative feature value. Hence, this study introduces the MaxAbs normalization approach to address the problem. Each feature in MaxAbs is scaled using its maximum value as (Al Shalabi et al., 2006):

$$MaxAbs_X = \frac{X}{|X|_{max.}}. \tag{13}$$

The action space comprises three discrete actions: green, red and yellow. However, an agent can change the current active signalling phase if and only the elapsed time is equal to or greater than the minimum green time. After several trial-error tests, the optimal minimum green time was found to be 5 units in the SUMO package. Figure 3 illustrates the phase-changing scheme of an arbitrary intersection between 1000 and 1230 s of simulation. The *x*-axis and *y*-axis depict the time (s) and signalling phases of different intersection lanes, respectively. The red colour indicates that the vehicles need to stop. Yellow means they are ready to move ahead, while green allows them to advance.

The reward function is defined as the change of cumulative delay, which is calculated by summing the individual delay of the total incoming vehicles at a certain time and has the form:

$$Reward = Delay_t - Delay_{(t+1)}. \tag{14}$$

3.4 | Dijkstra algorithm

Dijkstra's algorithm (Dijkstra et al., 1959) is famous for finding the shortest paths between nodes in a network. It fixes a single node as the ‘source’ node and finds the shortest paths to all other nodes in the network, producing the shortest path tree. For example, suppose the network nodes and edges represent cities and driving distances between pairs of cities connected by a direct road. In that case, Dijkstra's algorithm helps find the shortest route between one city and all other cities. The working mechanism of Dijkstra is quite simple: it picks the unvisited vertex with the lowest distance and calculates the distance through it to each unvisited neighbour. It then updates the neighbour's distance if smaller and

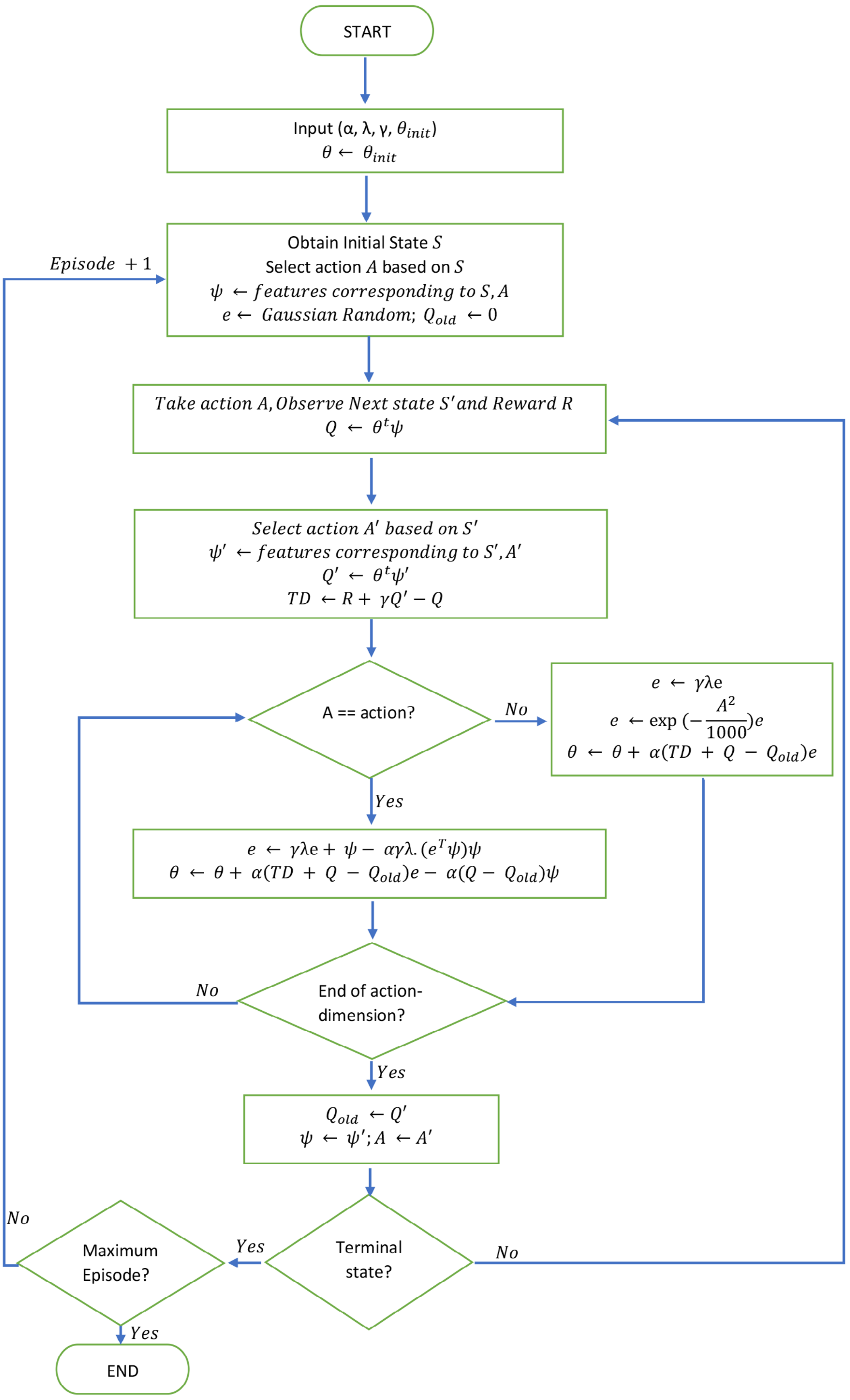

START
Input (α, λ, γ, θ_{init})
$\theta \leftarrow \theta_{init}$
Obtain Initial State S
Select action A based on S
$\psi \leftarrow features\ corresponding\ to\ S, A$
$e \leftarrow Gaussian\ Random;\ Q_{old} \leftarrow 0$
$Episode\ +1$
$Take\ action\ A, Observe\ Next\ state\ S'\ and\ Reward\ R$
$Q \leftarrow \theta^t \psi$
$Select\ action\ A'\ based\ on\ S'$
$\psi' \leftarrow features\ corresponding\ to\ S', A'$
$Q' \leftarrow \theta^t \psi'$
$TD \leftarrow R + \gamma Q' - Q$
A == action?
No
$e \leftarrow \gamma\lambda e$
$e \leftarrow \exp(-\frac{A^2}{1000})e$
$\theta \leftarrow \theta + \alpha(TD + Q - Q_{old})e$
Yes
$e \leftarrow \gamma\lambda e + \psi - \alpha\gamma\lambda.(e^T\psi)\psi$
$\theta \leftarrow \theta + \alpha(TD + Q - Q_{old})e - \alpha(Q - Q_{old})\psi$
No
End of action-dimension?
Yes
$Q_{old} \leftarrow Q'$
$\psi \leftarrow \psi'; A \leftarrow A'$
No
Maximum Episode?
Yes
Terminal state?
No
Yes
END

FIGURE 2 Illustration of the working mechanism of the proposed model

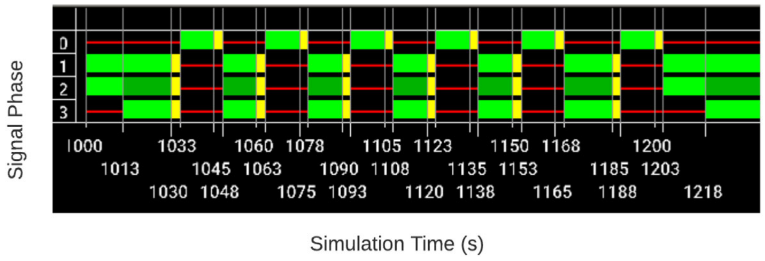


FIGURE 3 Illustration of the signal phase transition of an arbitrary intersection

marks it as visited when done with neighbours (Wang et al., 2011). This study uses a built-in Dijkstra algorithm within the SUMO package (Lopez et al., 2018) to find the shortest path from a single source vertex to all other vertices in the network.

The algorithm's major disadvantage is that it does a blind search, thereby consuming time and wasting necessary resources. Another disadvantage is that it cannot handle negative edges. It leads to acyclic graphs and most often cannot obtain the right shortest path (Darwish et al., 2019).

3.5 | A-star algorithm

The A-star algorithm, initially designed to find the least-cost paths for a predefined source-destination pair, effectively finds the optimal paths for any problem satisfying the conditions of the cost algebra. It works based on heuristic methods to achieve the optimal path, calculates the cost to travel to the neighbouring nodes and chooses the node with the lowest cost. If $f(n)$ denotes the cost, A-star chooses the node with the lowest $f(n)$ value, where n denotes the neighbouring nodes. The calculation of the value can be done as (Edelkamp et al., 2005):

$$f(n) = g(n) + h(n), \tag{15}$$

where $g(n)$ is the shortest path's value from the source to n^{th} node and $h(n)$ represents the heuristic approximation of the node's value. It is optimally efficient within a fixed cost budget (Paul et al., 2011). This study uses a built-in algorithm within the SUMO package to find the ultimate path from a single source vertex to all other vertices in the network.

4 | EXPERIMENTS

The proposed algorithm was trained for 100 epochs, and each took around 5 min on a system with an Intel® Core™ i7-10750H processor at 2.6 GHz, 16 GB of RAM, without any graphical processing units (GPU), and Ubuntu as the operating system. The Python open-source PyTorch machine learning library was used as the coding platform. The code implementation comprised three main steps. The first step covers the building of the SUMO environment. The ‘sumo-rl’ package (Alegre, 2019) using SUMO ‘Traffic Control Interface (TraCI)’ and ‘OpenAI Gym’ (Brockman et al., 2016) were used to import it. The network (with extension ‘.net.xml’) and route files (with extension’.rou.xml’) were carefully built to load the environment on the simulator. Seven classes of vehicles with different characteristics were incorporated within the environment. Table 1 summarizes their main characteristics. SpeedFactor depicts each vehicle's speed distribution, that is, the normal distribution of mean and standard deviation, in the context of their legal speed limits.

The second step was devoted to modelling the agents. This study implemented a multi-agent learning approach where one independent agent handles each intersection. Every agent is identified with agent ids, that is, traffic-signal-ids. The action and observation space is a mapping from agent traffic-signal-ids to their individual spaces. The agents typically choose the course of action with the highest Q-value, but also explore with a fixed low likelihood. In fact, they must select an alternative course of action if the elapsed time is greater than $max\_green = 50$ in the last signal phase. If not, the ε-greedy scheme is followed. The reward function is the difference between consecutive waiting times. After several trail-error tests, the following hyper-parameters provided the best overall performance: $num\_seconds = 3600$, $yellow\_time = 3$, $min\_green = 5$, $max\_green = 50$, $\alpha = 0.000000001$, $\gamma = 0.95$, $\varepsilon = 0.05$, and $\lambda = 0.1$. Here, α, γ, and ε represent the learning rate, discount factor, and exploration rate, respectively. These values were chosen because they are more common in the literature and provided the best outcomes after performing extensive experiments. However, the utilized hyper-parameter tuning scheme is time-consuming and difficult. Also, it requires considerable computational resources. So far, the research community in RL fields put less emphasis on developing adaptive tuning schemes, that is, only a few related research articles were found in the literature, for example, in Kiran and Ozyildirim (2022), which should be addressed robustly. The second step concluded with implementing the baseline model suggested by van Seijin et al. (2016) and Kekuda et al. (2021) with identical parameters and network configurations.

Finally, the third step was dedicated to visualization purposes. Python 'Matplotlib','Seaborn','Numpy','Pandas', and 'Glob' packages were used to serve the purpose. A predefined moving average function of variable window size using standard deviation and mean values of metric components was implemented to build the graphs. This approach increased the smoothness of the graphs to make them easily understandable to evaluate the algorithms' performances.

4.1 | Road network

This study addressed two road network scenarios: a simple one with two intersections (Figure 4), designated as scenario-1, and a complex real-world like 21-intersections network from Ingolstadt, a city of Bavaria in Germany (Figure 5), designated as scenario-2. To make the simulation similar to real-world traffic environments, buses, trucks, emergency, delivery vehicles, motorcycles, and bicycles were included along with passenger cars in both scenarios. The vehicle flow assimilates the morning peak hours of regular working days of a week, where a considerable influx is observed. The simulation took 3700 s on the SUMO package for each scenario. Scenario-2 contains the road network with a total lane length of 152.45km, including 381 intersections. It generated 4212 vehicles within the network, a bulk of which was produced during the start of the simulation to resemble a morning peak hour case. This study compared the proposed algorithm with conventional SARSA (λ) based on MinMax, MaxAbs, and Standard normalization techniques. These baselines were also simulated using the same scenarios with identical hyperparameters.

TABLE 1 Vehicle classes and characteristics of the simulation environment ($V_{max}(m/s)$, $a_{max}(m/s^2)$ and $norm(Mean,SD)$ represent the maximum speed, acceleration and normal distribution of the mean and standard deviation of the vehicles speed, respectively)

VClass	Numbers	$V_{max}(m/s)$	$a_{max}(m/s^2)$	Speed factor ($norm(Mean,SD)$)
Delivery	684	55.56	2.6	$norm(1.0,0.05)$
Passenger	3219	55.56	2.6	$norm(1.0,0.1)$
Bus	54	27.78	1.2	$norm(1.0,0.0)$
Truck	93	36.11	1.3	$norm(1.0,0.05)$
Motorcycle	100	55.56	6.0	$norm(1.0,0.1)$
Bicycle	51	5.56	1.2	$norm(1.0,0.1)$
Emergency	11	55.56	2.6	$norm(1.0,0.0)$

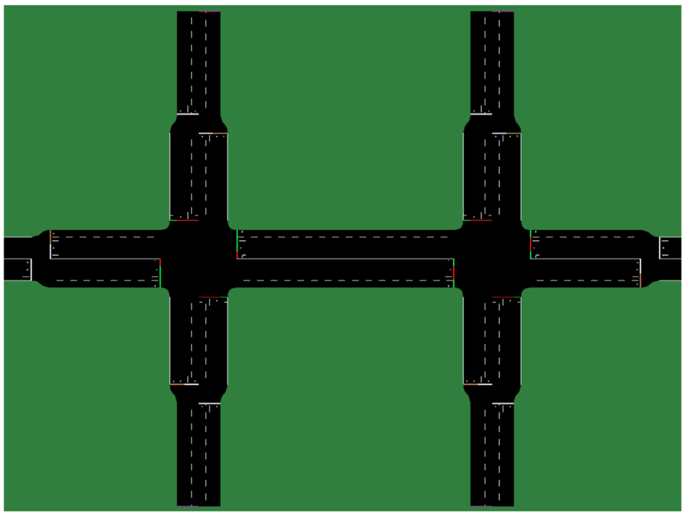


FIGURE 4 Illustration of scenario-1

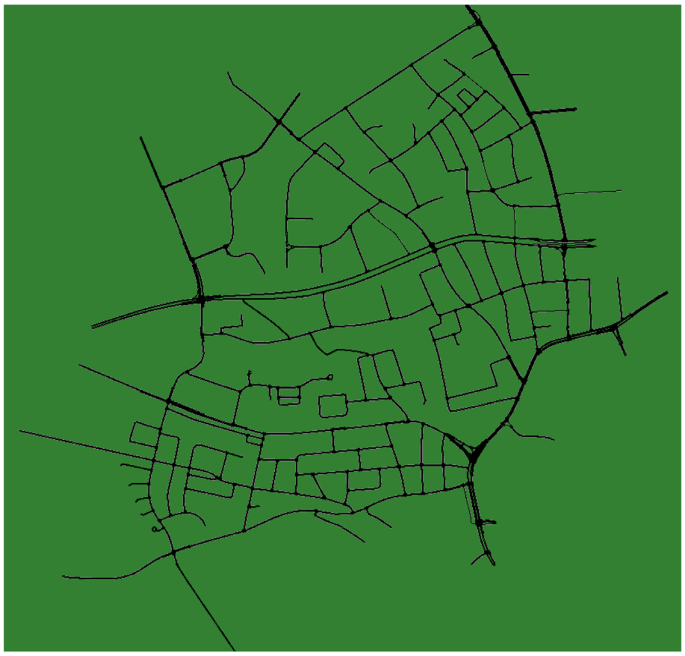


FIGURE 5 Illustration of scenario-2: It is a 21-intersections road network from Ingolstadt, a city of Bavaria, in Germany

4.2 | Results

The performance of the proposed algorithm was evaluated based on state-of-the-art metrics, mainly total waiting time (TOT), average delay time, and the average number of stops (TOS).

4.2.1 | Total waiting time

TOT during one signal cycle can be expressed as the sum of two components:

$$W = W_1 + W_2, \tag{16}$$

where W_1 and W_2 represent the total waiting time experienced in the red phase and green phase, respectively, and are defined as:

$$W_1 = \int_0^{(c-g)} [Q(0) + A(t)]dt, \tag{17}$$

and

$$W_2 = \int_{(c-g)}^{c} Q(t)dt, \tag{18}$$

where c = signal cycle, g = effective green signal time, $Q(t)$ = vehicle queue at t time, and $A(t)$ = cumulative arrivals at t.

4.2.2 | Average delay per vehicle

If d, c, g, x, and q represent the average delay per vehicle (s), cycle length (s), effective green time (s), degree of saturation, that is, flow to capacity ratio, and arrival rate (veh/s), respectively, then the average delay per vehicle can be formulated as (Bilgram et al., 2021):

$$d = \frac{c(1-g/c)^2}{2[1-(g/c)x]} + \frac{x^2}{2q(1-x)} - 0.65\left(\frac{c}{q^2}\right)^{1/3} x^{2+5(g/c)}. \tag{19}$$

This study obtained the experimental results after training the proposed algorithm and the baselines under consideration up to 100 epochs based on the above-mentioned experimental setup and hyper-parameter tuning schemes. In scenario-1 of a two intersection road network, the proposed algorithm achieved an average TOT and TOS of only 117.55*s* and 5.12, which represent 9.97% and 8.41% of improvement relative to the conventional SARSA (λ), respectively. Table 2 and Figure 6 let one realize the performance of the proposed model on a simple road network like the one of scenario-1. The shaded region over the plots shown in this figure represents the standard deviation of TOS values. The moving average window of 1 (one) size helped to smooth the lines of the plots.

Since the proposed model showed its goodness on a small demo network compared to the baseline, examining its performance on a real-world-like network is necessary to strengthen its worthiness. Most of the models presented in the literature are evaluated on smaller networks with few vehicle classes leading to a lack of interpretation ability. One of the goals of this study was to address this shortcoming by evaluating the model in a city-wide network which closely resembles a real-world network.

So, for the complex road network of scenario-2, the proposed model also exhibited its worthiness in achieving an average TOT and TOS of 396.04 s and 7.40, respectively. In terms of statistics, the results showed 39.27%–80.52% and 2.12%–32.97% improvements by the proposed algorithm relative to the conventional SARSA (λ), SARSA (λ)-MinMax and SARSA (λ)-SD-based traffic signal controllers. The dominant outputs of the agent built in the previous steps arose from the efficient weight update method and MaxAbs normalization on observation space components to feed the agents. Table 3 details the proposed model's performance and the of baselines under consideration.

Figures 7 and 8 illustrate the proposed model's performance compared with the other baselines regarding average TOS and TOT, respectively. Although the SARSA (λ)-SD-based traffic signal controller showed the worst result among all the under-comparison models in average TOT,

TABLE 2 Performance comparison of the proposed algorithm relative to the baselines on scenario-1 (* represents the best result; the SARSA (λ) was implemented according to Alegre et al., 2022)

Models	Average TOT (s)	Average TOS
SARSA(λ)	130.57	5.59
Proposed	117.55*	5.12*

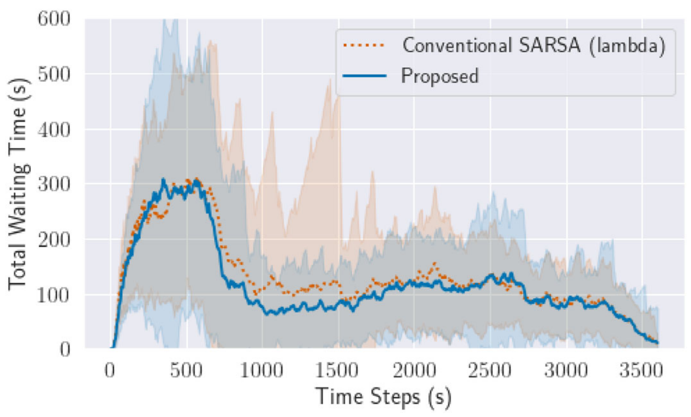


FIGURE 6 Average total waiting time for the experiment on scenario-1

TABLE 3 Performance comparison of the proposed algorithm relative to the baselines on scenario-2 (* represents the best result)

Models	Average TOT (s)	Average TOS
SARSA(λ)	652.8	8.1
SARSA(λ) − *MaxAbs*	999.85	11.04
SARSA(λ) − *SD*	2033.68	7.56
Proposed	396.04*	7.40*

it exhibited a close-ever performance on average TOS with the proposed algorithm. Similarly, the shaded region over the plots shown in those figure represents the standard deviation of the TOS and TOT values. The moving average window of 1 (one) size helped to smooth the lines of the plots. During the morning rush hours, the road networks generally undergo the highest pressure of vehicles, and the network route file was carefully constructed to mimic this usual trend. When the ATSC works well, the generated vehicles can easily pass through the intersections and exit the network according to their route definition. Otherwise, they would get stuck on the road resulting in congestion. The initial peaks of the figures suggest that the ATSC is dealing with morning rush hours, and the proposed model exhibits superior performance in handling the peak hours than the considered baselines. It can be observed from Figure 8 that the TOT curve continues to rise instead of falling to the horizontal line for the SARSA (λ)-SD-based controller. It suggests the inefficiency of the ATSC based on that particular model to help the vehicles pass through the intersections resulting in congestion, that is, higher waiting time.

The reward function was the difference between the cumulative vehicle delays of the intersections, and its distribution was also analysed. The maximum, minimum and standard deviation of the cumulative rewards of the proposed model was 24.02, −24.02 and 0.31, respectively. On the other hand, for the conventional SARSA (λ), their values were 13.53, −20.12 and 0.24, respectively. These values accounted for the model's superior performance relative to its baseline.

This study also examined the performance of two routing algorithms: Dijkstra and A-star. The average speed (m/s) of the vehicles following the A-star routing algorithm with the proposed traffic signal controller (Astar_Proposed) was $5.96m/s$, which over-performed A-star routing algorithm with SARSA (λ)-based traffic signal control (A-star_SARSA (λ)) and Dijkstra routing algorithm with SARSA (λ)-based traffic signal control (Djk_SARSA (λ)) by 17.1% and 30.7%, respectively. Moreover, the average time loss (s) per vehicle obtained by Astar_Proposed was equal to $57.36\,s$, which, relatively to A-star_SARSA (λ) and Djk_SARSA (λ), was 88.26% and 75.15% less, respectively. Table 4 allows one to clearly perceive the superior performance of the routing algorithms in the proposed signal controlling scheme.

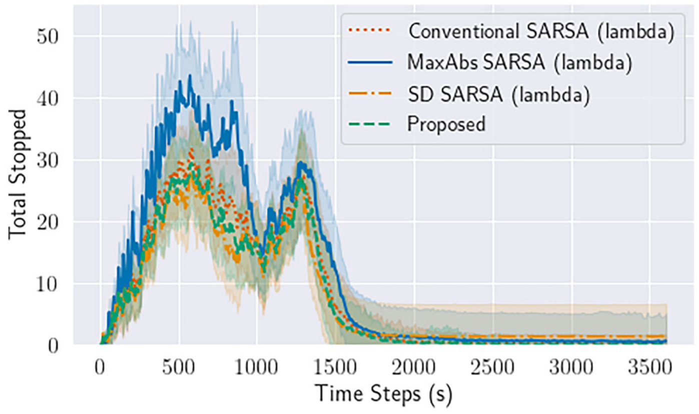


FIGURE 7 Average TOS for the experiment on scenario-2

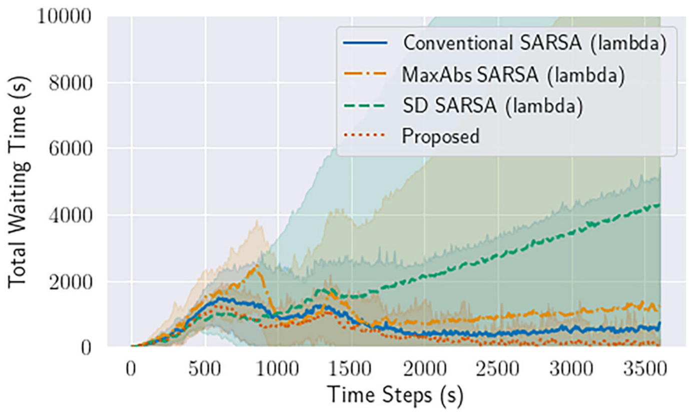


FIGURE 8 Average total waiting time for the experiment on scenario-2

TABLE 4 Performance comparison of the three studied routing algorithms (* represents the best result)

Models	Average speed (m/s)	Average time loss (s)
Dijkstra_SARSA (λ)	4.13	132.51
Astar_SARSA (λ)	4.94	145.63
Astar_Proposed	5.96*	57.36*

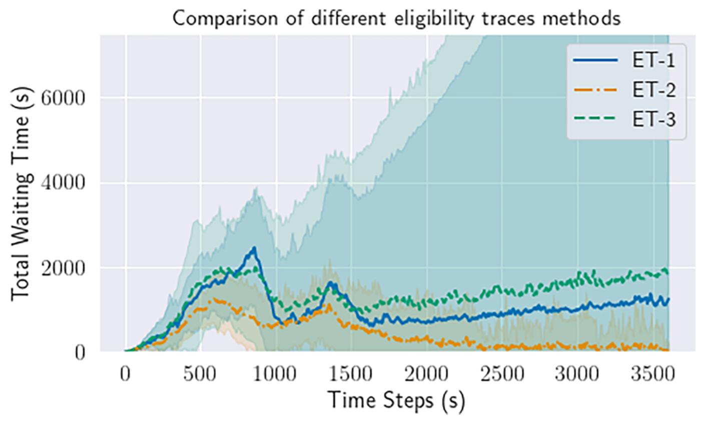


FIGURE 9 Different eligibility trace vector initialisation (ET-1, ET-2 and ET-3 represent the initialisation of the eligibility trace by a null, Gaussian random and unit vector, respectively)

5 | DISCUSSION

The currently employed traffic signal controllers in Europe, like the FT controller, cycle in sequential order all of the time and do not rely on any detection, irrespective of the kind of intersection. Hence, the signals follow the cycle even if no vehicles are present. For light traffic states, it might work well. However, such controllers are less effective when traffic varies significantly during the day. In this context, the DRL-based controller, particularly SARSA (λ), can potentially address these problems. Researchers are finding ways to improve its effectiveness and efficiency. This study aims to further enhance its worthiness by introducing a Gaussian function for effectively decaying the eligibility trace vector. Because updating the weights also depends on the eligibility trace vector, the study demonstrated the proposed decaying mechanism to be more efficient than the conventional approach.

Also, conventional SARSA (λ) uses the MinMax normalization technique to scale the observation space vector between $[0,1]$. However, MinMax possesses one demerit: it might scale the observation space vector by $(-1,1)$ if any of the components have a negative value. This research introduces the MaxAbs scaling technique to solve such a problem, demonstrating its goodness in improving the model's performance.

The proposed model employs state-of-the-art average TOT and TOS for its evaluation purpose. Its performances demonstrated promising outcomes in a small road network and a complex real-world-like network with many signalling intersections. The results proved its worthiness by achieving 39.27%–80.52% and 2.12%–32.97% of improvements compared to conventional SARSA (λ), SARSA (λ)-MinMax and SARSA (λ)-SD-based traffic signal controllers. The proposed algorithm can deal with the traffic dynamics more effectively, resulting in a reasonable performance improvement compared to the studied baselines. Besides, it can efficiently adapt to new and unseen traffic scenarios in a more manageable way. However, during the study, it was also observed that the proposed algorithm works less effectively when the number of vehicles increases abruptly.

Furthermore, this study also found that initialising the eligibility trace vector often plays a vital role in achieving optimal performance. Three different initialisation approaches were investigated in this regard: initialisation by a null vector, unit vector and Gaussian random (Lifshits, 1995) initialisation. The latter follows a distribution of 0 (zero) mean and a standard deviation of 1 (one), and it provided the best results, as shown in Figure 9.

The ATSC performance of the proposed model was compared with recently published algorithms to justify its potentiality. A SARSA (λ) model based on linear function approximation proposed in (Alegre et al., 2022) was implemented on the same environment and network configurations. The currently proposed model outperformed the SARSA (λ) model by achieving an average TOT of 396.04 s, which is 39.33% higher than it. The road network used in this research contained seven types of vehicles: Here, not only cars, but also buses, trucks, emergency vehicles, motorcycles, and bicycles with different speed and acceleration parameters were considered to resemble a real-world road network.

TABLE 5 Performance comparison of the two routing algorithms working on the proposed traffic signal controller (* represents the best result)

Metrics	Dijkstra_Proposed	Astar_Proposed
Avg. trip duration (s)	261.03	243.98*
Avg. trip waiting time (s)	75.16	68.81*
Avg. trip time loss (s)	136.46	124.38*
Avg. trip depart delay (s)	103.41	21.02*
Avg. trip speed (m/s)	5.76	5.96*

For the autonomous vehicle routing, A-star provided better performance than Dijkstra's algorithm by reducing the average trip duration, waiting time, time loss, departure delay, and trip speed to a good margin, as indicated in Table 5. The A-star routing algorithm can find the most suitable routes in any circumstance. Additionally, when A-star operated on the proposed traffic signal controlling algorithm, the average trip speed was equal to 5.96 m/s, which is 3.36% more than the same routing algorithm running on SARSA (λ) controller.

This research used TraCI using a built-in TCP-based client/server architecture which provides access to the SUMO simulator. Here, SUMO works as a server and the controller as a client. The TraCI allowed the controller to retrieve the simulated traffic environment's values and manipulate their behaviour online. The proposed algorithm is cost-effective, that is, it does not require a cloud server for its training, yet provides interesting results. Also, the agents were modelled as individual agents, and their collaboration will be considered in future work.

The proposed model is a promising solution for improving the ATSC mechanisms of a city-wide road traffic network. However, its training process in a real-world environment would be expensive and unsafe. Hence, a microscopic simulation environment, that is, SUMO, was used to develop the algorithm. This study incorporated seven different types of vehicles within the simulation environment to mimic a real-world network. However, it faces a common simulation challenge to real-world deployment, referring to Sim2Real transfer gap (Miao et al., 2023). This discrepancy commonly results from the simulation-optimisation bias, where the controller takes advantage of simulator flaws and overrates simulator performance compared to the target domain. Different approaches can be followed to overcome these issues. For example, the zero-shot transfer learning approach proposed in (Voogd et al., 2022) and the executable digital twin method suggested by Allamaa et al. (2022) can be explored.

6 | CONCLUSION

This article presented a city-wide DRL-based ATSC for autonomous vehicles. The proposed model was evaluated on a city-wide road network composed of seven classes of vehicles with varying speed and acceleration characteristics to resemble a real-world network. The introduction of the proposed Gaussian decaying approach of the eligibility trace vector for efficient weight updating exhibited dominant performance compared to the conventional methods. In addition, using MaxAbs normalization of the elements of the observation space vector addressed the drawbacks of the traditional MinMax technique. State-of-the-art metrics, mainly TOT and TOS, were used to comprehensively compare its performance against conventional SARSA (λ), SARSA (λ)-MinMax and SARSA (λ)-SD-based traffic signal controllers. The results demonstrated a reduction of average TOT and TOS by a mean value of 59.9% and 17.55% relative to mentioned baselines. This study also evaluated the performance of two routing algorithms: Dijkstra and A-star algorithms, on the proposed traffic signal controller. A-star algorithm performed better than Dijkstra's algorithm by minimizing the average trip time loss and increasing the average trip speed to 124.38 s and $5.96m/s$, which led to 8.9% and 3.4% of improvement, respectively, when compared to its counterparts. Hence, this study verified that the autonomous vehicles' trip performance could be increased by combining the A-star routing algorithm and the proposed traffic signal controller. In the future, more traffic signalling phases must be considered, and pedestrian crossings along the intersection will be included. Also, it requires the investigation of a Deep Q-network instead of TD learning for further performance enhancement.

AUTHOR CONTRIBUTIONS

João Manuel R. S. Tavares: Conceptualization and supervision. **Selim Reza:** Investigation, data collection and code implementation, formal analysis, and original draft preparation. **J. J. M. Machado, Marta Campos Ferreira and João Manuel R. S. Tavares:** Writing review and editing.

ACKNOWLEDGEMENTS

The authors would like to thank Fundação para a Ciência e Tecnologia (FCT) for the PhD grant (reference 2022.12391.BD) awarded to the first author, which this work is a part of.

CONFLICT OF INTEREST STATEMENT

The authors declare no conflict of interest.

DATA AVAILABILITY STATEMENT

Data sharing is not applicable to this article as no new data were created or analyzed in this study.

ORCID

Selim Reza https://orcid.org/0000-0002-2877-2980
Marta Campos Ferreira https://orcid.org/0000-0001-9505-5730
J. J. M. Machado https://orcid.org/0000-0002-1094-0114
João Manuel R. S. Tavares https://orcid.org/0000-0001-7603-6526

REFERENCES

Abdoos, M., Mozayani, N., & Bazzan, A. L. (2013). Holonic multi-agent system for traffic signals control. *Engineering Applications of Artificial Intelligence*, *26*(5–6), 1575–1587.

Abdulhai, B., Pringle, R., & Karakoulas, G. J. (2003). Reinforcement learning for true adaptive traffic signal control. *Journal of Transportation Engineering*, *129*(3), 278–285.

Al Shalabi, L., Shaaban, Z., & Kasasbeh, B. (2006). Data mining: A preprocessing engine. *Journal of Computer Science*, *2*(9), 735–739.

Alegre, L. N. (2019). *SUMO-RL*. GitHub https://github.com/LucasAlegre/sumo-rl

Alegre, L. N., Ziemke, T., & Bazzan, A. L. (2022). Using reinforcement learning to control traffic signals in a real-world scenario: An approach based on linear function approximation. *IEEE Transactions on Intelligent Transportation Systems*, *23*(7), 9126–9135.

Allamaa, J. P., Patrinos, P., Van der Auweraer, H., & Son, T. D. (2022). Sim2real for autonomous vehicle control using executable digital twin. *IFAC-PapersOnLine*, *55*(24), 385–391.

Antes, T. O., Bazzan, A. L., & Tavares, A. R. (2022). Information upwards, recommendation downwards: Reinforcement learning with hierarchy for traffic signal control. *Procedia Computer Science*, *201*, 24–31.

Arel, I., Liu, C., Urbanik, T., & Kohls, A. G. (2010). Reinforcement learning-based multi-agent system for network traffic signal control. *IET Intelligent Transport Systems*, *4*(2), 128–135.

Aziz, H. A., Zhu, F., & Ukkusuri, S. V. (2018). Learning-based traffic signal control algorithms with neighborhood information sharing: An application for sustainable mobility. *Journal of Intelligent Transportation Systems*, *22*(1), 40–52.

Bilgram, A., Ernstsen, E., Greve, P., Lahrmann, H., Larsen, K. G., Muniz, M., et al. (2021). Online and proactive vehicle rerouting with Uppaal Stratego. *Transportation Research Record*, *2675*(11), 13–22.

Bouktif, S., Cheniki, A., & Ouni, A. (2021). Traffic signal control using hybrid action space deep reinforcement learning. *Sensors*, *21*(7), 2302.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., et al. (2016). OpenAI Gym. *arXiv*, 1–4.

Darwish, S. M., Elmasry, A., & Ibrahim, S. H. (2019). Optimal shortest path in mobile ad-hoc network based on fruit fly optimization algorithm. In: *International Conference on Advanced Machine Learning Technologies and Applications* (pp. 91–101). Springer.

Dijkstra, E. W., et al. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, *1*(1), 269–271.

Edelkamp, S., Jabbar, S., & Lluch-Lafuente, A. (2005). Cost-algebraic heuristic search. In: *AAAI* (pp. 1362–1367).

Eriskin, E., Karahancer, S., Terzi, S., & Saltan, M. (2017). Optimization of traffic signal timing at oversaturated intersections using elimination pairing system. *Procedia Engineering*, *187*, 295–300.

Ghanadbashi, S., & Golpayegani, F. (2022). Using ontology to guide reinforcement learning agents in unseen situations: A traffic signal control system case study. *Applied Intelligence*, *52*(2), 1808–1824.

Hunt, P. (1981). SCOOT—A traffic responsive method of coordinating signals. *Transport and Road Research Laborathory LR Report*, *1014*, 1–41.

Kekuda, A., Anirudh, R., & Krishnan, M. (2021). Reinforcement learning based intelligent traffic signal control using n-step SARSA. In: *2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS)* (pp. 379–384). IEEE.

Kiran, M., & Ozyildirim, M. (2022). Hyperparameter tuning for deep reinforcement learning applications. *arXiv*, 1–11.

Li, C., & Xu, P. (2021). Application on traffic flow prediction of machine learning in intelligent transportation. *Neural Computing and Applications*, *33*(2), 613–624.

Lifshits, M. A. (1995). *Gaussian random functions* (Vol. 322). Springer Science & Business Media.

Lopez, P. A., Behrisch, M., Bieker-Walz, L., Erdmann, J., Flötteröd, Y. P., Hilbrich, R., et al. (2018). Microscopic traffic simulation using SUMO. In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)* (pp. 2575–2582).

Miao, Q., Lv, Y., Huang, M., Wang, X., & Wang, F. Y. (2023). Parallel learning: Overview and perspective for computational learning across Syn2Real and Sim2Real. *IEEE/CAA Journal of Automatica Sinica*, *10*(3), 603–631.

Modares, H., Lewis, F. L., & Naghibi-Sistani, M. B. (2014). Integral reinforcement learning and experience replay for adaptive optimal control of partially-unknown constrained-input continuous-time systems. *Automatica*, *50*(1), 193–202.

Muhammad, K., Hussain, T., Ullah, H., Del Ser, J., Rezaei, M., Kumar, N., Hijji, M., Bellavista, P., & de Albuquerque, V.H.C. (2022). Vision-based semantic segmentation in scene understanding for autonomous driving: Recent achievements, challenges, and outlooks. *IEEE Transactions on Intelligent Transportation Systems*, *23*, 22694–22715.

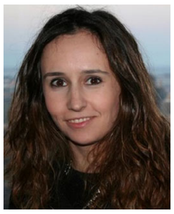
Muhammad, K., Ullah, A., Lloret, J., Del Ser, J., & de Albuquerque, V. H. C. (2020). Deep learning for safe autonomous driving: Current challenges and future directions. *IEEE Transactions on Intelligent Transportation Systems*, *22*(7), 4316–4336.

Niknami, M., & Samaranayake, S. (2016). Tractable pathfinding for the stochastic on-time arrival problem. In: *International Symposium on Experimental Algorithms* (pp. 231–245). Springer.

Ogasawara, E., Martinez, L. C., De Oliveira, D., Zimbrão, G., Pappa, G. L., & Mattoso, M. (2010). Adaptive normalization: A novel data normalization approach for non-stationary time series. In: *The 2010 International Joint Conference on Neural Networks (IJCNN)* (pp. 1–8). IEEE.
Osorio, C., & Wang, C. (2017). On the analytical approximation of joint aggregate queue-length distributions for traffic networks: A stationary finite capacity Markovian network approach. *Transportation Research Part B: Methodological*, *95*, 305–339.
Otterlo, M. V., & Wiering, M. (2012). Reinforcement learning and markov decision processes. In: *Reinforcement Learning* (pp. 3–42). Springer.
Patro, S., & Sahu, K. K. (2015). Normalization: A preprocessing stage. *arXiv*, 1–4.
Paul, B., Ibrahim, M., Bikas, M., & Naser, A. (2011). Vanet routing protocols: Pros and cons. *International Journal of Computer Applications*, *20*(3), 28–34.
Prashanth, L., & Bhatnagar, S. (2011). Reinforcement learning with average cost for adaptive control of traffic lights at intersections. In: *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)* (pp. 1640–1645). IEEE.
Sims, A. G., & Dobinson, K. W. (1980). The Sydney coordinated adaptive traffic (SCAT) system philosophy and benefits. *IEEE Transactions on Vehicular Technology*, *29*(2), 130–137.
Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT Press.
Taylor, M. E., Whiteson, S., & Stone, P. (2006). Comparing evolutionary and temporal difference methods in a reinforcement learning domain. In: *Proceedings of the 8th annual conference on Genetic and evolutionary computation* (pp. 1321–1328).
van Hasselt, H., Guez, A., & Silver, D. (2016). Deep reinforcement learning with double Q-learning. *Proceedings of the AAAI Conference on Artificial Intelligence*. https://ojs.aaai.org/index.php/AAAI/article/view/10295
Van Seijen, H., Mahmood, A. R., Pilarski, P. M., Machado, M. C., & Sutton, R. S. (2016). True online temporal-difference learning. *The Journal of Machine Learning Research*, *17*(1), 5057–5096.
Voogd, K., Allamaa, J. P., Alonso-Mora, J., & Son, T. D. (2022). Reinforcement learning from simulation to real world autonomous driving using digital twin. *arXiv*, 1–6.
Wang, H., Yu, Y., & Yuan, Q. (2011). Application of Dijkstra algorithm in robot path-planning. In: *2011 Second International Conference on Mechanic Automation and Control Engineering* (pp. 1067–1069). IEEE.
Wang, T., Cao, J., & Hussain, A. (2021). Adaptive traffic signal control for large-scale scenario with cooperative group-based multi-agent reinforcement learning. *Transportation Research Part C: Emerging Technologies*, *154*, 125–146.
Wiering, M., Vreeken, J., van Veenen, J., & Koopman, A. (2004). Simulation and optimization of traffic in a city. *IEEE Intelligent Vehicles Symposium*, *2004*, 453–458.
Wunderlich, R., Liu, C., Elhanany, I., & UrbanikII, T. (2008). A novel signal-scheduling algorithm with quality-of-service provisioning for an isolated intersection. *IEEE Transactions on Intelligent Transportation Systems*, *9*(3), 536–547.
Yen, C. C., Ghosal, D., Zhang, M., & Chuah, C. N. (2020). A deep on-policy learning agent for traffic signal control of multiple intersections. In: *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)* (pp. 1–6). IEEE.
Zhao, W., Ye, Y., Ding, J., Wang, T., Wei, T., & Chen, M. (2022). Ipdalight: Intensity-and phase duration-aware traffic signal control based on reinforcement learning. *Journal of Systems Architecture*, *123*, 102374.
Ziemke, T., Alegre, L. N., & Bazzan, A. L. (2021). Reinforcement learning vs. rule-based adaptive traffic signal control: A Fourier basis linear function approximation for traffic signal control. *AI Communications*, *34*(1), 89–103.

AUTHOR BIOGRAPHIES

Selim Reza is currently doing his PhD at Faculdade de Engenharia da Universidade do Porto (FEUP), Portugal, under the supervision of Professor João Manuel R. S. Tavares. He completed his BSc and MS from the Department of Electrical and Electronic Engineering, University of Dhaka, Bangladesh. His current research includes the application of Deep Neural Networks in urban safety, particularly in the Intelligent Transportation Systems. He has previously published five articles in different Web of Science index journals as well.



Marta Campos Ferreira is a researcher and invited assistant professor at Faculty of Engineering of University of Porto. She holds a PhD in Transportation Systems from the Faculty of Engineering of University of Porto (MIT Portugal Program), a MSc in Service Engineering and Management from the Faculty of Engineering of University of Porto and a Lic. in Economics from the Faculty of Economics of University of Porto. She is the Co-Founder & Co-Editor of the Topical Collection “Research and Entrepreneurship: Making the Leap from Research to Business” with SN Applied Sciences and Associate Editor of the *International Journal of Management and Decision Making*. She has been involved in several R&D projects in areas such as technology enabled services, transport and mobility. Her current research interests include service design, human computer interaction, data science, knowledge extraction, sustainable mobility and intelligent transport systems.

José Joaquim da Mota Machado obtained an MSc in Mechanical Engineering from the University of Porto, Portugal (2012). In 2019, he received a PhD in Mechanical Engineering in Adhesively Bonded Automotive Structures at the University of Porto (in partnership with Nagase ChemteX and Aston Martin Lagonda). From 2019 to 2020, he joined the Tokyo Institute of Technology for a post-doctoral position. Between 2012 and 2016, he worked as a Mechanical Engineer in the field of Electric Motors for hazard areas, Oil and Gas industry, and Maintenance of Mining Equipment. From 2016 to 2019, he was a researcher at the Institute of Science and Innovation in Mechanical and Industrial Engineering (INEGI). He was an Invited Assistant Professor in the Department of Mechanical Engineering (DEMec) of the Faculty of Engineering of the University of Porto (FEUP) between 2018 and 2019. He joined as an assistant professor in October 2021 in the same department. Since 2017, he has been co-supervisor of several MSc theses. He is co-author of more than 40 articles in national and international journals. In 2020, he started tasks as a guest editor of several special issues of international journals. Also, he has enrolled in several research projects, both as a researcher and as a scientific coordinator. His main research areas include design and manufacturing, computational vision and new product development.

João Manuel R. S. Tavares graduated in Mechanical Engineering at the Universidade do Porto, Portugal in 1992. He also earned his MSc degree and PhD degree in Electrical and Computer Engineering from the Universidade do Porto in 1995 and 2001, and attained his Habilitation in Mechanical Engineering in 2015 from the same University. He is a senior researcher at the Instituto de Ciência e Inovação em Engenharia Mecânica e Engenharia Industrial (INEGI) and full professor at the Department of Mechanical Engineering (DEMec) of the Faculdade de Engenharia da Universidade do Porto (FEUP). João Tavares is co-editor of more than 85 books, co-author of more than 50 book chapters, 650 articles in international and national journals and conferences and 3 international and 3 national patents. He has been a committee member of several international and national journals and conferences is co-founder and co-editor of the book series "Lecture Notes in Computational Vision and Biomechanics" published by Springer, founder and Editor-in-Chief of the journal "Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization" published by Taylor & Francis, Editor-in-Chief of the journal "Computer Methods in Biomechanics and Biomedical Engineering" published by Taylor & Francis, and co-founder and co-chair of the international conference series: International Symposium on Computational Modeling of Objects Presented in Images, ECCOMAS Thematic Conference on Computational Vision and Medical Image Processing, International Conference on Computational and Experimental Biomedical Sciences and International Conference on Biodental Engineering. Additionally, he has been (co-)supervisor of several MSc and PhD thesis and supervisor of several post-doc projects, and has participated in many scientific projects both as researcher and as scientific coordinator. His main research areas include computational vision, medical imaging, biomechanics, scientific visualization and new product development (more information can be found at: www.fe.up.pt/~tavares).

How to cite this article: Reza, S., Ferreira, M. C., Machado, J. J. M., & Tavares, J. M. R. S. (2025). A citywide TD-learning based intelligent traffic signal control for autonomous vehicles: Performance evaluation using SUMO. *Expert Systems*, *42*(1), e13301. https://doi.org/10.1111/exsy.13301