ISSN 2442-6571

Traffic light optimization (TLO) using reinforcement learning for automated transport systems

Mohammad Mehedi Hassan [a,1,*], Stephen Karungaru [a,2], Kenji Terada [a,3]

[a] Computer Science and Mathematical Science Program, Tokushima University, Minamijosanjimacho 2-Chōme, Tokushima, Japan
[1] metsys19@gmail.com; [2] karungaru@tokushima-u.ac.jp; [3] terada@is.tokushima-u.ac.jp
* corresponding author

ARTICLE INFO

Article history
Received July 11, 2024
Revised February 11, 2025
Accepted February 13, 2025
Available online February 28, 2025

Keywords
Intelligent transportation systems
Machine learning
Reinforcement learning
Traffic management
Simulation and modeling

ABSTRACT

Current traffic light systems follow predefined timing sequences, causing the light to turn green even when no cars are waiting, while the side road with waiting vehicles may still face a red light. Reinforcement learning can help by training an intelligent model to analyze real-time traffic conditions and dynamically adjust signal lights based on actual demand and necessity. If the traffic light becomes intelligent and autonomous, then it can significantly reduce the time wasted on commuting due to previously determined traffic light timing sequences. In our previous work, we used fuzzy logic to control the traffic light where the time was fixed but, in this paper, the waiting time becomes a variable that changes depending on other road variables like vehicles, pedestrians, and times. Moreover, we trained an agent in this work using reinforcement learning to optimize the traffic flow in junctions with traffic lights. The trained agent worked using the greedy method to improve traffic flow to maximize the rewards by changing the signals appropriately. We have two states and there are only two actions to take for the agent. The results of the training of the model are promising. In normal situations, the average waiting time was 9.16 seconds. After applying our fuzzy rules, the average waiting time was reduced to 0.26 seconds, and after applying reinforcement learning, it was 0.12 seconds in a simulator. The average waiting time was reduced by 97-98%. These models have the potential to improve real-world traffic efficiency by approximately 67-68%.

1. Introduction

We are all too familiar with the daily struggles of commuting, especially during rush hour when reaching our destination on time seems like an insurmountable challenge. Even in the absence of accidents, strikes, or inclement weather, traffic congestion can add significant amounts of time and resources to our daily commute. INRIX, a transport consulting firm, estimates that major cities worldwide will lose billions of dollars annually due to traffic congestion [1]. Traffic congestion remains a significant challenge in urban areas worldwide, leading to increased travel times, fuel consumption, and environmental pollution. As cities grow and vehicle ownership rises, traditional [2] traffic management systems struggle to adapt to fluctuating demand and unexpected disruptions. Conventional traffic light control relies on fixed-time or rule-based adjustments [3]. This kind of system often fails to optimize flow in real-time, especially during peak hours or in response to incidents. Factors such as pedestrian movement, public transport coordination, and emergency vehicle prioritization further complicate traffic control. Current systems' inefficiencies affect economic productivity and increase carbon emissions and commuter frustration [4]. Given the rapid advancements in artificial intelligence and sensor technology, optimizing traffic light operations through intelligent and adaptive algorithms is urgently necessary to improve urban mobility, reduce congestion, and create more sustainable transportation networks. Improving existing traffic management systems is urgent, particularly in traffic light operations. Wasted time waiting for traffic lights to turn green is a common frustration for drivers, especially when there is little to no traffic between them [5]. The resulting irritation and anxiety can lead to reckless driving and accidents [6]. In addition to lost time waiting for traffic lights, traffic jams, and congestion are major sources of frustration for commuters. Providing real-time traffic information to motorists can help alleviate some of these issues. Although local radio stations provide traffic information, it is not always up-to-date and may not apply to all drivers [7]. Therefore, improved methods for regulating traffic lights and providing real-time, location-specific traffic information are urgently needed—traditional traffic light systems, such as fixed-timing and sensor-based adaptive systems. Current traffic systems often struggle with inefficiencies, including poor responsiveness to fluctuating traffic conditions. Fixed-time traffic signals are simple, cost-effective, and easy-to-implement systems. These systems don't adjust to real-time traffic changes; they run on pre-programmed cycles. The systems have drawbacks such as poor scalability, expensive installation and maintenance, and an inability to forecast future traffic patterns accurately. Sensor-based traffic systems are vulnerable to false detections due to environmental conditions because they depend on sensor accuracy. On the other hand, reinforcement learning (RL)-based systems [8] learn optimal traffic light control policies through trial and error. The systems adapt dynamically based on current and predicted traffic conditions. RL overcomes traditional limitations by continuously improving and interacting with the environment. It provides scalability, real-time adaptation, and cost efficiency. It can reduce long-term infrastructure costs by minimizing the need for additional sensors.

Automated or self-driving cars [9]–[12] are on the horizon, we are almost already there, but designing intelligent roads or traffic lights is still far. Traffic lights [13] have a significant role in the traffic flow and safety measures of the roads. Road networks must have traffic lights since they are essential to controlling traffic flow. They are intended to regulate which stream of traffic has priority by alternately stopping and allowing cars to pass through a junction. Long wait times and congestion can result from inefficient traffic signals, which can lower the effectiveness of the road system. As an illustration, improper traffic light timing might result in a backlog of vehicles, which can prolong travel times and cause delays. However, when traffic lights are improved, they can greatly increase traffic flow by minimizing delays and reducing congestion. The timing of traffic lights can be adjusted using various methods, such as using computer vision to count vehicles and implementing intelligent systems that use machine learning algorithms to predict traffic patterns and adjust the lights in real time. The placement and design of the lights should also be considered in addition to maximizing the timing of traffic lights. For instance, some junctions may benefit from roundabouts or four-way stops instead of using a traffic signal. Overall, effective traffic light design and optimization may significantly improve traffic flow, decrease delays, increase safety, and minimize negative environmental impacts.

A lot of similar research has already been done. In our previous works [14], [15], we proposed fuzzy logic [16] to optimize the traffic light by getting information on the vehicles, pedestrians, and lane lines. Other research intrigued us to research intelligent traffic control systems. One work [17] proposed a method using three onboard cameras to support the driver with Advanced Safety Vehicle (ASV) Technology, where the authors are trying to develop a safe driving support system. This research paper from 2018 utilizes three different cameras to detect various elements on the road. One camera tracks the driver's appearance, while the other two track front-end pedestrians, running lanes, and approaching vehicles and pedestrians. The HOG feature is employed to detect pedestrians and vehicles, while edge detection and RANSAC are used for detecting lanes. Using information from all three cameras, the system determines whether or not to alert the driver based on the situation at hand. Furthermore, the onboard camera system can detect road signs and make automatic decisions in real-time. To improve detection accuracy, the authors used genetic algorithms to enhance the matching process and narrow the search range by color extraction. Additionally, neural networks were used to confirm the results of the template-matching process. Specifically, this research focused on detecting the stop sign, which is known to cause accidents due to driver negligence [18]. Radio Frequency Identification (RFID) technology has been utilized [19] in several research studies exploring intelligent traffic control systems. In the paper, the authors describe how they installed RFID readers to track vehicles and collect their electronic product code (EPC) data via the RFID tags attached to the vehicles. This data is then used for decision-making and traffic control purposes within the system.

A novel method [20] for a large-scale traffic signal re-timing system that uses vehicle trajectories as input, reducing congestion and energy consumption without relying on vehicle detectors, is proposed. The system uses a probabilistic time-space diagram to reconstruct traffic states and update parameters. A real-world test in Birmingham, Michigan, showed a 20% decrease in delay and a 30% reduction in stops at signalized intersections. This scalable, sustainable, and efficient solution could be applied to all fixed-time signalized intersections. Changing the traffic signal by giving priority to emergency vehicles based on the traffic density is proposed by Vani *et al.* [21]. In this paper, the authors utilized cameras in traffic systems to monitor and quantify the volume of vehicles, allowing for the determination of traffic density. Additionally, the camera system was designed to recognize emergency vehicles with sirens and grant them priority by adjusting the traffic light to green. To process the images, the authors employed a masking algorithm that focuses on the relevant portion of the image while ignoring extraneous details. Furthermore, the authors used Visual Basic programming to regulate the traffic light duration based on the number of vehicles on the road; Bhilawade and Ragha [22] propose a method to detect road accidents and violations of vehicle movement using sensors and embedded technology. The goal is to minimize time wasted in heavy traffic jams, reduce waiting time at road junctions, assist emergency vehicles in navigating traffic, and detect stolen vehicles. The results of the research were published in a journal in 2018. The system employs a combination of RFID and GPS-based automatic lane clearance to prioritize emergency vehicles. By integrating sensors and technology, the system can quickly determine the presence of traffic congestion and make real-time decisions to optimize traffic flow.

Another study [23], presents a new method called Active Control of Traffic Signals (ACTS) using the Non-Dominated Sorting Genetic Algorithm. The model considers multiple objectives, such as minimizing average delay time and vehicle stops per cycle. The method reduces average vehicle delay by almost half compared to the current solution, promoting faster traffic flow and reducing congestion. To ensure normal traffic operation and reduce congestion in urban intersections, Zhang *et al.* [24] propose a modified Webster function for signal timing at intersections based on signal cycle and green light duration. The method reduces intersection delay by 15.64% using a modified genetic algorithm. Another work, Wang *et al.* [25], proposes a traffic light timing optimization method called EP-D3QN, which uses double dueling deep Q-network, MaxPressure, and Self-organizing traffic lights (SOTL) to control traffic flows. The method dynamically adjusts traffic light duration based on rules and lane pressure. Each intersection corresponds to an agent, with traffic light phases varying between 0-60 seconds. The agent's reward is the difference between the waiting time of all vehicles in two consecutive signal cycles. Experimental results show EP-D3QN improves traffic efficiency in light and heavy traffic flow scenarios.

Many previous studies have utilized onboard cameras or focused on specific areas, such as detecting humans, lanes, or vehicles. Some research has been based on sensors, while others have been based on image processing. Current traffic light optimization faces several critical gaps. The current system lacks real-time adaptability, inefficiency in multi-intersection coordination, and limited use of predictive analytics. It has high infrastructure and maintenance costs and an inability to handle unpredictable events. Traditional sensor-based systems struggle to adapt to unexpected congestion, accidents, or special events. while reinforcement learning and AI-driven models can proactively forecast traffic trends and adjust signal timing. Traffic light systems are also isolated from other smart transportation infrastructures, such as connected vehicles, GPS data, or public transit systems. Traditional traffic lights do not consider fuel consumption and emissions when optimizing signal timings, resulting in energy inefficiency and environmental impact. However, this particular study is focused on using fuzzy logic and reinforcement learning [25] methods to train an agent to control the traffic light and experiment in a simulator. Our research focuses on optimizing traffic flow by analyzing vehicle density at connected intersections. We count the number of vehicles and calculate their waiting times to minimize delays. For instance, if one road has no vehicles while another has waiting vehicles, the traffic signal dynamically adjusts to prioritize the waiting traffic. Additionally, the signal changes to prevent excessive delays if the waiting time exceeds a certain threshold. This approach enables the agent to be aware of its environment and make real-time decisions, similar to how a human would assess traffic conditions. We utilize cameras for situational awareness, employing computer vision to observe traffic flow. A trained RL-based agent then processes this information and optimally adjusts the traffic signals to improve efficiency. The primary goal of this research is to train an agent to maximize the traffic flow in minimum time and develop an intelligent system that can efficiently control traffic lights based on real-time traffic conditions. This is an extended version of our previous research, exploring how computer vision can enhance the current traffic light management systems. Our approach integrates a trained agent with computer vision to assess the environment, making real-time decisions based on the recognition of vehicles, pedestrians, and lane markings. This system optimizes traffic flow by adapting to varying conditions. As autonomous vehicles become more prevalent, our system could facilitate communication between vehicles and traffic infrastructure (V2I) [26], allowing for smarter traffic management. Additionally, the system could prioritize pedestrian safety by adjusting signal timings based on specific situations. For example, if an elderly person is crossing the street slowly, the system can extend the signal duration to accommodate them. The system can suggest alternative routes in an accident or roadblock by leveraging infrastructure-to-infrastructure (I2I) communication. By maximizing traffic flow, our system has the potential to significantly reduce congestion, lower pollution levels, and decrease driver frustration, ultimately contributing to safer and more efficient roadways.

2. Method

2.1. Overview of Research Design and Proposed Methods

In our previous works [14], [15], we proposed a method where high-resolution cameras were installed at various junctions to collect real-time traffic data, which included vehicles, pedestrians, and lanes. The collected data was analyzed to identify unnecessary waiting times. Next, a set of self-made fuzzy rules was developed to optimize the signal lights. The rules were then applied in a simulator to determine the optimization results. The results were satisfactory.

In this research, we proposed to train an intelligent agent to maximize the traffic flow in a minimum time. Here we divided our work into two parts, one is previously introduced Fuzzy logic but with an updated waiting time equation where the waiting time depends on the variables, and the second part is the reinforcement learning method to train an agent. In both cases, we used the simulator Simulation of Urban Mobility (SUMO) [27]. Simulation of Urban Mobility (SUMO) is an open-source, multi-modal traffic simulation software designed to model and analyze road traffic, public transportation, and pedestrian mobility in urban environments. SUMO enables the simulation of large-scale traffic networks, incorporating elements such as vehicles, traffic lights, pedestrians, and public transit systems. It allows for modeling microscopic (individual vehicle) and macroscopic (aggregate flow) traffic models, providing insights into traffic flow, congestion patterns, emissions, and the impact of various traffic management strategies. SUMO supports dynamic simulation, enabling real-time adjustments based on external data, and offers flexibility for evaluating smart traffic control systems and autonomous vehicle integration. Additionally, SUMO interfaces with various external tools and data sources, making it a widely used platform for intelligent transportation systems (ITS) and urban mobility solutions. The challenges of implementing fuzzy logic in real-time and conducting experiments in sensitive areas of road safety have led us to use simulators to obtain better results and understand the optimal traffic light situations for future real-time implementation.

In our setup, we simulated a junction consisting of two roads (ROAD A and ROAD B), as shown in Fig. 1 using the SUMO network format (net.xml). The simulation was conducted with a specific number of vehicles, defined through vehicle demand and traffic flow parameters (e.g., maxSpeed, departureTime, and route). We utilized SUMO's Traffic Light Control (TLC) model to simulate the signal phases and timings, which were adjusted under different scenarios. In the first scenario, we applied a general situation with a fixed signal phase, using default parameters for signal timings (e.g., greenTime, yellowTime, and redTime) without modifying the traffic light phases.

In the second scenario, we implemented a dynamic control approach based on the traffic conditions. The traffic light phases were modified according to the number of vehicles (vehicleCount) on each road and their waiting times (waitingTime). We dynamically calculated the green light duration using parameters like the total waiting time of vehicles at the junction and traffic flow, enabling better road capacity utilization.

We introduced a reinforcement learning agent for the third scenario that controlled the traffic light phases. The agent was trained using a reward function designed to maximize traffic flow (e.g., by maximizing the throughput rate or minimizing vehicle waiting times) based on state-action pairs. The agent adjusted the traffic light phases, learning optimal traffic control strategies through trial and error, thus improving the system's overall efficiency. Key parameters for the agent's learning process included learning rate (alpha), discount factor (gamma), and exploration-exploitation balance (epsilon). This method taught the agent to optimize traffic flow based on real-time conditions.

2.2. Experimenting with Fuzzy Logic

To optimize the traffic lights, we applied our self-made fuzzy logic in a scenario, where there is only one junction with two roads, shown in Fig. 1. We used the simulator SUMO. A SUMO network file describes the roads and intersections that the simulated cars run along or through on the traffic-related portion of a map. A directed graph with a rough scale is a SUMO network. Nodes, which are sometimes referred to as "junctions" in context, are used to depict intersections, while "edges" are typically represented by roads or streets. While Netgenerate provides a new map with straightforward shapes, Netconvert assists in converting various formats into SUMO network files. In this file's map, sumo executes its simulation. Most other SUMO programs read these files to produce or import data that has to be mapped onto a road network. The study created a network of two roadways so that it could be tested in a simulator.

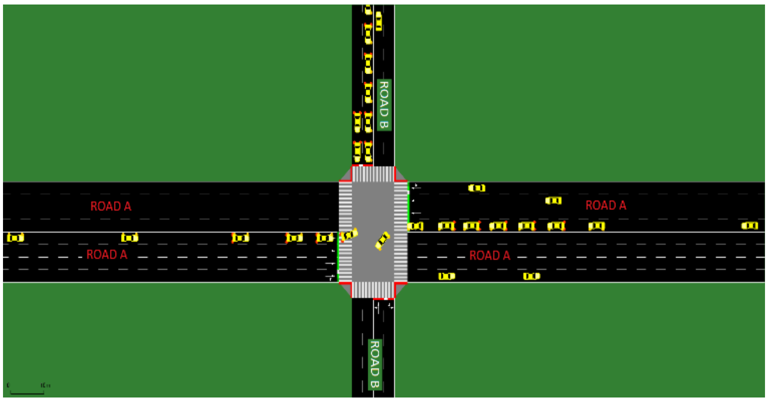

Fig. 1. Two roads Junction

We defined them as main road A, and side road as B. Additionally, the study implemented pre-determined timing sequences for the signal lights to replicate real-life conditions. This means that the timing of the signals was fixed, regardless of the presence or absence of vehicles. The signal lights followed predetermined instructions, mimicking a normal situation. The traffic light program allowed for the customization of signal durations and states. The duration specified how long a signal light would remain before transitioning to another, while the state determined the specific configuration of the signal lights. This approach enabled the researchers to analyze the timing required to manage traffic flow in the network effectively. After gathering information about the system without any optimization rules, the study proceeded to implement optimization rules or Fuzzy logic to investigate the waiting time of vehicles at the junction. Specifically, we used fuzzy inference systems (FIS) to dynamically adjust the signal phases based on two main inputs: traffic flow and vehicle waiting time. The inputs were fuzzified using membership functions that described different traffic conditions such as 'Low', 'Medium', and 'High' for both vehicles count and waiting time. These fuzzy rules were integrated into the SUMO simulation by modifying the traffic light controller (TLC) settings, allowing real-time changes in the signal phases based on the fuzzy logic output. The system continuously evaluated the input conditions and applied the appropriate fuzzy rule to determine the control action (i.e., the green, yellow, or red phases). The computation of the fuzzy logic was carried out within the simulation framework using the fuzzy control algorithm, which processed the inputs, applied the rules, and then de-fuzzified the result to determine the actual signal phase duration.

The rules for traffic light optimization are:

Rule 1: $V_B > 2V_A$; Here, V_B is the total vehicle Count for Road B; V_A is the total vehicle count for Road A; If the B road's vehicles count double that of road A, then the signal light will change for road B.

Rule 2: $Wb > Tw$; Here, W_b are Vehicles Waiting Time for Road B; T_w is a waiting time; If road B's vehicle's waiting time is more than the T_w then the signal light will change.

Rule 3: $E_b > 0\&\&E_a = 0$; Here, E_b is an emergency Vehicles Count for Road B; E_b is Emergency Vehicles Count for Road A; If there are no emergency vehicles on Road A and there is one on Road B, then the signal light will change.

Rule 4: $P_b > 4\&\&\ W_p > T_w$; Here, P_b are the pedestrians Count for Road B; W_p is a waiting time for Pedestrians; and $T_w\ is\ a\ waiting\ time$; The last rule for traffic light optimization is that if the pedestrians' counts are equal or more than four in the crosswalk and their waiting time is more than T_w then the signal light will change.

To calculate the waiting time (T_w), assuming that the traffic signal allocates equal time to both roads, the waiting time for side road B vehicles can be estimated as follows;

$$T_{clearA} = d_A/V_s * V_A \tag{1}$$

$$T_{clearB} = d_B/V_s * V_B \tag{2}$$

here, T_{clearA}, T_{clearB} are the clearing times for road A and B. d_A, and d_B are the distance to cover for the road A, and B. V_s is the average speed of the vehicles for each road and V_A, and V_B are the total vehicle counts for both roads. To calculate the waiting time, we also need to calculate the flow rate for each road.

Calculating the flow rate for road B:

$$R_B = V_B/T_{clearB} \tag{3}$$

Calculating the flow rate for road A:

$$R_A = V_A/T_{clearA} \tag{4}$$

Calculating the total flow rate:

$$R_A = V_A/T_{clearA} \tag{5}$$

Calculating the time allocated for road A:

$$T_A = R_A/T_{total} * cycleTime \tag{6}$$

Calculating the time allocated for road B:

$$T_B = R_B/T_{total} * cycleTime \tag{7}$$

Yellow Light's Duration Calculation:

$$T_{yellow} = round\left(\frac{V_{max}}{A_{dec}}\right); \tag{8}$$

Example: If the maximum speed of a car or the lane is 60km/h, which is equal to 16.67m/s. The commonly seen decelerating acceleration for vehicles varies depending on the context and conditions but is typically around 3.5 to 4.5 m/s^2 [28], [29]. Studies have shown that drivers generally decelerate at a rate of approximately 0.3 g (2.94 m/s^2) to 0.4 g (3.92 m/s^2) when approaching an intersection or stop sign under normal conditions. This rate is often selected for driver comfort and control during deceleration. Let's assume the decelerating acceleration is 4.5m/s2 then the duration of Tyellow is 3.7 ≈ 4 seconds.

Calculating average waiting time for side road B vehicles during each cycle:

$$T_w = T_A + T_{clearA} + T_{yellow} - T_{clearB} \quad (9)$$

2.3. Using an Agent to control Traffic Light

Even though we got some better results using fuzzy logic experiments, it has some drawbacks. For example, scaling fuzzy logic controllers to larger, more complex traffic networks can be challenging and may lead to computational inefficiencies. It needs ongoing maintenance and adjustment because it cannot evolve like reinforcement learning. In highly dynamic or nonlinear traffic environments, traditional control approaches like fuzzy logic may face challenges in effectively optimizing traffic signal timings. Fuzzy logic relies on expert knowledge and predefined rules to make decisions, which may not adequately capture traffic patterns' complex and evolving nature. In contrast, reinforcement learning (RL) offers a more adaptive and data-driven approach. RL algorithms learn optimal control policies through trial and error interactions with the environment, allowing them to adapt to changing traffic conditions and learn complex strategies for traffic signal control. By continuously learning from feedback, RL algorithms can dynamically adjust signal timings based on real-time traffic conditions, potentially leading to better performance in highly dynamic or nonlinear traffic environments compared to traditional fuzzy logic control. Thus, we adapted another experiment with reinforcement learning to train an agent to control the traffic light.

Fig. 2 shows the proposed idea for the agent to control the traffic light. The objective of the trained agent is to maximize the flow of traffic in a minimum time. The states are divided into two categories: the moving state S_m and the stopped state Ss. Because either vehicles are moving or stopped in a signal or may be due to accidents or other problems. The possible actions that could be taken for traffic signals are green light Ag or red light A_r, We did not consider the yellow light because it is the preprocess of red light. The rewards are given for each transition from one state to another while taking a particular action.

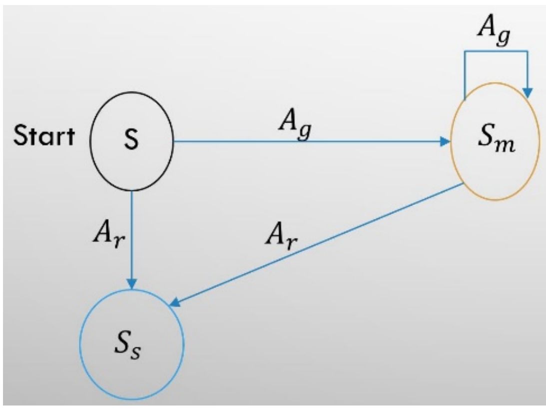

Fig. 2. Proposed idea for training an agent. Two probable states (Moving state, Stopped state) and two actions (Green Light, Red Light)

Formally, the Markov decision process (MDP) [30] can be defined as follows:

1. State Space: $S = S_m, S_s$
2. Action Space: $A = A_g, A_r$

3. Transition Probability: $P(s, a, s') = P(s' \mid s, a))$
   - When taking action A_g, the probability of transitioning from S_s to S_m is high, and the probability of staying in S_s is low.
   - When taking action A_r, the probability of staying in S_s is high, and the probability of transitioning from S_m to S_s is low
4. Reward Function: $R(s, a, s') = r$
   - When transitioning from S_m to S_s while taking action A_r, the reward is negative, as it causes traffic to stop.
   - When transitioning from S_s to S_m while taking action A_g, the reward is positive, as it causes traffic to flow.
   - Every other transition is associated with a reward of zero (*0*).

Firstly we used the Deep Q-Network (DQN) [31] to train an agent to maximize the traffic flow. The DQN algorithm uses a neural network, known as the Q-network, to approximate the Q-function. The same basic idea of estimating and updating Q-values for state-action pairings serves as the basis for DQN. Through repeated updates, DQN tries to find the optimal course of action. The DQN algorithm depends on the Bellman equation [32], which combines the immediate reward and the discounted maximum Q-value of the next state to represent the optimum Q-value.

DQN, a deep learning extension of Q-learning, utilizes a neural network to approximate the Q-values, allowing it to handle high-dimensional state spaces more effectively than traditional Q-learning. This "model-free" approach can deal with problems that involve uncertain transitions and rewards without the need for an environmental model.

In a traffic environment, the state space can be defined by variables representing the traffic lights, where each variable indicates whether the corresponding direction has a red or green light. The DQN agent interacts with this environment through steps and resets, updating the state according to the chosen action and receiving rewards based on the traffic conditions. The reward system might grant positive rewards for time steps without traffic jams and negative rewards for traffic jams.

The agent's behavior is governed by the Q-network, learning rate, discount factor, and exploration rate. Actions are chosen either by exploring with a certain probability or by selecting the action with the highest Q-value in the current state. The learning process involves updating the Q-value for the current state-action pair using the reward and the maximum Q-value for the next state, guided by the Bellman equation. The equation 10 [33], represents the Q-value equation.

$$Q(s, a) = E\left[R + \gamma \underset{a'}{max} Q(s', a') \mid s, a\right] \tag{10}$$

Here, $Q(s, a)$ represents the Q-value of the state-action pair (s, a), which is the expected cumulative reward for taking action a in state s; R is the immediate reward obtained after taking action a in state s; γ (gamma) is the discount factor, which determines the relative importance of future rewards compared to immediate rewards. It is a value between 0 and 1; $max_{a'} Q(s', a')$ represents the maximum Q-value over all possible actions a' in the next state s'.

The predicted cumulative rewards for each potential action in a given condition are represented by the Q-function. Inputting the state, the network generates a Q-value for each action. The agent selects the action that possesses the highest Q-value. We created a replay memory buffer to store the agent's experiences in each episode. Each experience is defined as a tuple *state*, *action*, *reward*, *next_state*, and *done*. Here, the state represents the current state, and action is the action taken; two actions can be taken in our research: reward is the immediate reward received, next_state is the resulting state and *done* indicates if the episode terminated. The Q-network created a neural network with weights that take the state as input and output Q-values for all possible actions that can be taken. To improve the stability, a target network was used to compute the target Q-values during training and updated frequently. For each episode, the agent chooses an action according to the greedy policy to increase the rewards, the action with the highest Q-value. Then, the action is executed in the environment, the next state is observed and rewarded, and the experiences are stored. Then, combining the experience replay and a target network to find the problems of sample correlation and non-stationary targets, respectively. For effectiveness, the target network provides more stable target Q-values for training. Through this process, the DQN algorithm learned an optimal policy for the conjunction to maximize the traffic flow by maximizing the cumulative rewards over time, even in complex environments with high-dimensional state spaces. Fig. 3 shows the architecture of the proposed DQN. This architecture consists of four layers.

- Input Layer: Receives the state representation as input. The size of this layer is determined by the problem's state space (state_size).
- Hidden Layer 1: The first hidden layer of neurons. Each neuron in this layer performs a linear transformation on the input data followed by a nonlinear activation function (e.g., ReLU).
- Hidden Layer 2: The second hidden layer, similar to Hidden Layer 1. It further processes the information extracted from the previous layer.
- Output Layer: Produces the Q-values for each action in the action space (action_size). The output of this layer is used to select the action with the highest Q-value

At the beginning of the training, the model was working well, but as the number of epochs increases, the loss starts to increase at a significant rate, showing signs of over-fitting to the training data becomes evident in the model. It is becoming too complex and is fitting the noise in the data rather than the underlying patterns. Even though after 20,000 epochs, the loss starts to decline again, which indicates that the model is starting to generalize better to new, unseen data

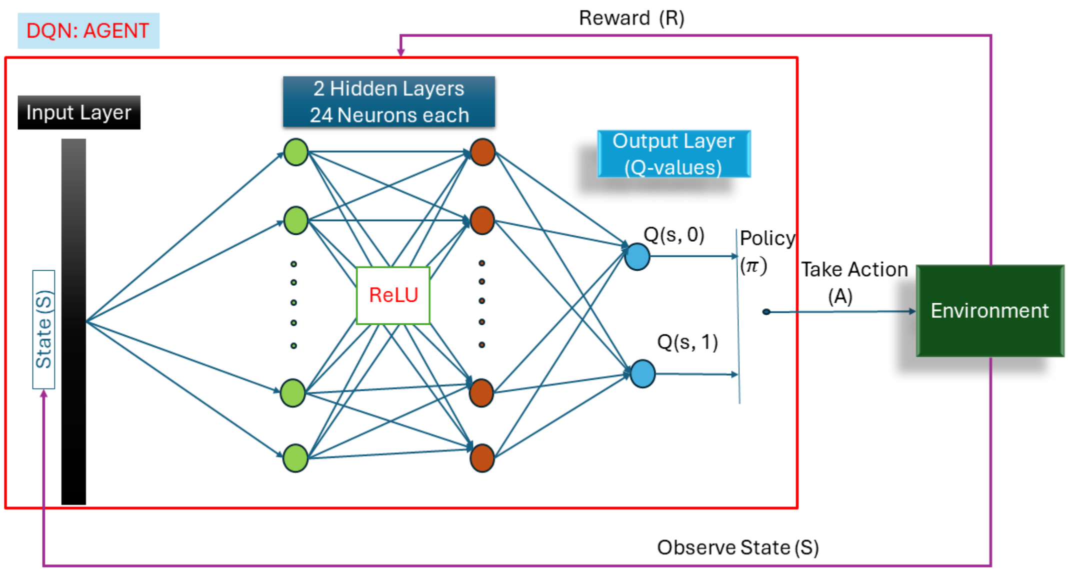

Fig. 3. The architecture of the proposed Deep Q-Network. The input is the state size and the output layer consists of two possible actions (Red light or Green Light)

3. Results and Discussion

Our results are divided into two sections, reflecting our experimental methodology. The first section presents the outcomes from the experiment using Fuzzy Logic, and the second section details the results from the experiment using reinforcement learning.

To validate our methods, we performed a series of experiments comparing the performance of the proposed reinforcement learning-based traffic control system with traditional traffic signal control strategies, including fixed-time and fuzzy logic-based controllers. The comparison was based on several performance metrics, such as traffic flow, waiting time, and congestion levels. Additionally, we tested the system under various traffic densities to evaluate its adaptability in different real-world scenarios. The results showed a significant improvement in traffic flow and a reduction in waiting times when using our reinforcement learning approach, validating its effectiveness in managing traffic. To test the system's robustness, we simulated a range of traffic scenarios, including variations in vehicle arrival rates. The system was subjected to these dynamic conditions to evaluate how well it maintained optimal traffic flow and adjusted the signal timings accordingly.

We also conducted a sensitivity analysis to assess how sensitive the model's performance was to changes in key parameters, such as the reward function, learning rate, and discount factor. The analysis revealed that while the agent's performance was generally stable across a range of parameter values, certain configurations led to improved learning efficiency and faster convergence. For instance, a moderate learning rate and discount factor resulted in a good balance between exploration and exploitation, enhancing the model's ability to adapt to dynamic traffic conditions.

3.1. Results with the Fuzzy Logic

The results we obtained from our previous experiment [15] are presented in "Fig. 4". The graph depicts the total and average waiting times for traffic under two different scenarios: a normal case and an optimized case using fuzzy logic. The x-axis represents the simulation steps, while the y-axis measures the waiting time in seconds. The solid red line indicates the total waiting time in the normal scenario, showing significant fluctuations and peaks, which suggest periods of high congestion. The average waiting time for the normal scenario, represented by the red dashed line, is approximately 9.16 seconds. This relatively high average waiting time highlights the inefficiency in traffic flow management in the normal scenario.

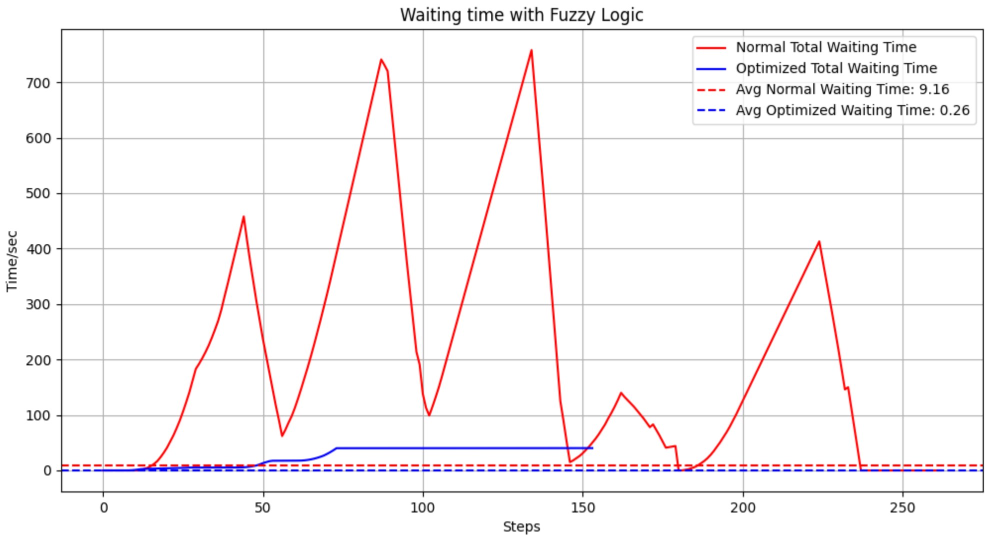

Fig. 4. Comparison between normal situation and optimized with the fuzzy logic situation

Conversely, the solid blue line represents the total waiting time in the optimized scenario. This line remains significantly lower and more stable compared to the normal case, indicating a more efficient traffic flow with fewer instances of congestion. The average waiting time for the optimized scenario, marked by the blue dashed line, is around 0.26 seconds, which is notably lower than in the normal case. This stark contrast in average waiting times between the two scenarios underscores the effectiveness of fuzzy logic optimization in minimizing traffic waiting times and improving overall traffic management efficiency.

In summary, the graph demonstrates that the optimized scenario using fuzzy logic substantially reduces both total and average waiting times, leading to a more efficient and smoother traffic flow compared to the normal scenario.

3.2. Results with the DQN algorithm

Fig. 5 shows the graph of model loss (y-axis) versus epochs (x-axis) during the training of a DQN algorithm. At the beginning of the graph, the loss is almost 0, which indicates that the model is performing well on the training data. However, as the number of epochs increases, the loss starts to increase at a significant rate, peaking at around 15,000 epochs. This suggests that the model is starting to over-fit the training data, which is becoming too complex and fitting the noise in the data rather than the underlying patterns. After 20,000 epochs, the loss starts to decline again, which could indicate that the model is starting to generalize better to new, unseen data.

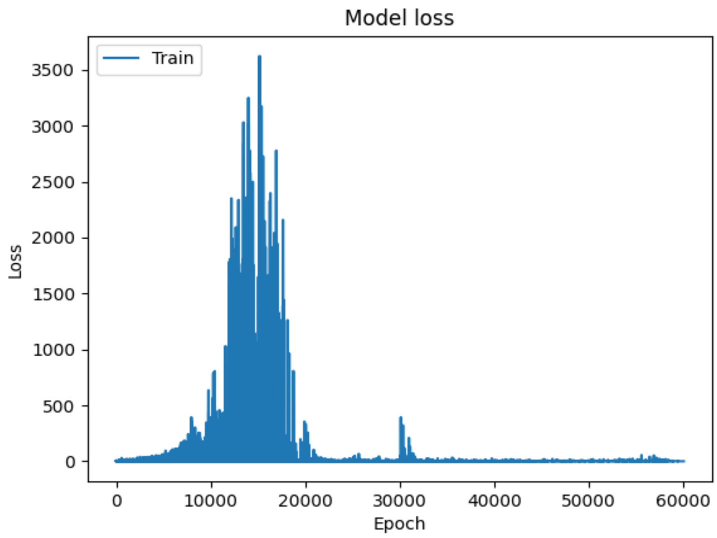

Fig. 5. Model Loss for the DQN algorithm

The Fig. 6 shows the confusion matrix for the DQN algorithm. The x-axis represents the predicted labels (predicted state of the traffic). The y-axis represents the true labels (actual state of the traffic). The values in the matrix are normalized, meaning they represent proportions rather than raw counts.

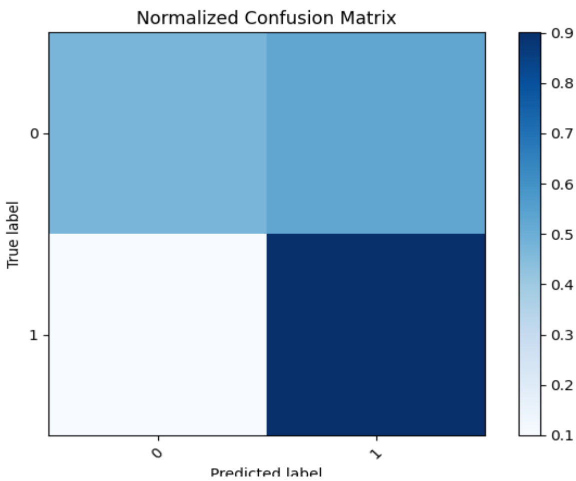

Fig. 6. Confusion Matrix for the DQN algorithm

The Confusion Matrix represents:

- True Positive (Bottom-right, [1,1]): This represents the proportion of instances where the traffic was actually moving, and the algorithm correctly predicted it as moving. The value is close to 0.9, indicating that the algorithm is very good at recognizing when traffic is moving.

- True Negative (Top-left, [0,0]): This represents the proportion of instances where the traffic was actually stopped, and the algorithm correctly predicted it as stopped. The value is slightly above 0.7, indicating that the algorithm has a decent performance in recognizing when traffic is stopped.
- False Positive (Top-right, [0,1]): This represents the proportion of instances where the traffic was stopped, but the algorithm incorrectly predicted it as moving. The value is just below 0.3, suggesting there are some cases where the algorithm incorrectly predicts the traffic as moving when it is stopped.
- False Negative (Bottom-left, [1,0]): This represents the proportion of instances where the traffic was moving, but the algorithm incorrectly predicted it as stopped. The value is very low, indicating that it is rare for the algorithm to predict the traffic as stopped when it is moving incorrectly.

The analysis of these metrics implies several conclusions. A high true positive rate indicates that the algorithm effectively predicts when traffic moves, which is crucial for minimizing congestion and optimizing traffic flow. The moderate true negative rate shows the algorithm's capability to correctly identify stopped traffic, which is important for recognizing when to change the light to green to alleviate traffic jams. The low false negative rate is a positive aspect as it avoids unnecessary stops that could cause delays. However, the presence of a false positive rate indicates instances where the traffic is stopped, but the algorithm predicts it as moving, potentially leading to slight inefficiencies by delaying the switch to a green light. Overall, the confusion matrix suggests that the DQN algorithm is quite effective in maximizing traffic flow by accurately predicting traffic states and adjusting the traffic lights accordingly. Nevertheless, there is room for improvement in reducing false positives to enhance further the algorithm's efficiency in managing stopped traffic situations. This could be achieved by fine-tuning the model, increasing the training data, or incorporating additional features into the state representation.

Following the completion of our training, we proceeded to apply our model for traffic light control, which yielded noteworthy outcomes. Vehicles experienced significantly reduced wait times, with an average waiting time of merely 0.12 seconds. In contrast, in a normal situation where traffic light sequences are predetermined, the average waiting time was 9.16 seconds. The Fig. 7 illustrates a graphical representation of these two scenarios and their corresponding waiting times. The red line signifies traffic lights controlled by our agent, while the blue line represents traffic lights operating based on predetermined timing sequences. It is important to note that achieving these results in real-time scenarios may pose challenges; nonetheless, this experimentation underscores the potential for optimizing traffic light control through reinforcement learning techniques.

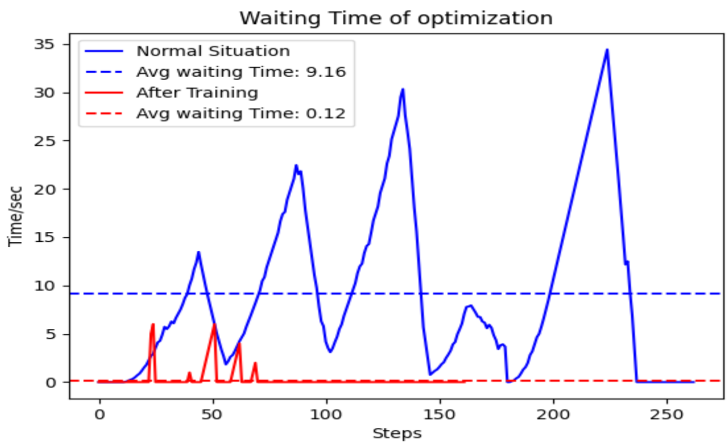

Fig. 7. Comparison between normal and trained agents in traffic light control

3.3. Potential Real-World Impact and Summary of the Results

When considering the application of the experiment conducted in the SUMO simulator to a real-time scenario, several factors must be considered. Here are some points to consider for estimating the potential percentage effect.

- Model Accuracy: The accuracy of the SUMO model in representing real-world traffic conditions is crucial. If the model accurately captures the dynamics of real traffic, the results in the real world could closely match the simulation. However, any discrepancies in the model could reduce the effectiveness.
- Environmental Variability: Real-world environments are more variable and less predictable than simulations. Weather conditions, driver behavior, accidents, and road construction can significantly impact traffic flow. These variables are often simplified or omitted in simulations.
- System Latency and Responsiveness: The real-time traffic management system's latency and responsiveness could differ from the simulation. Delays in sensor data processing or traffic signal updates could affect performance.
- Compliance and Enforcement: The effectiveness of traffic optimization algorithms depends on drivers' compliance with traffic signals. In real-world scenarios, driver behavior is less predictable, and non-compliance could reduce the system's effectiveness.
- Scalability: The system's scalability from a limited scope simulation to a real-world application with a much larger and more complex network can introduce additional challenges and potential inefficiencies.

Given these factors, it is reasonable to expect that the efficiency improvements seen in the simulation will be reduced when implemented in the real world.

We calculated the potential real-world impact with Equation (11).

$$R_i = S_i * A_f \tag{11}$$

Where R_i for estimated Real-World Improvement; S_i for simulation Improvement, which is the percentage improvement observed in the simulation; and A_f is an adjustment factor, which is a factor less than 1 to account for real-world inefficiencies.

$$S_i = \left(\frac{Normal\ Avg.WaitingTime(Simulation) - OptimizedAvg.WaitingTime(Simulation)}{Normal\ Avg.WaitingTime(Simulation)}\right) \times 100 \tag{12}$$

$$S_i = \left(\frac{9.16-0.12}{9.16}\right) \times 100 \approx 98.69\%$$

Assume an adjustment factor of 0.7 (which accounts for a 30% reduction in effectiveness due to real-world factors: $Ri =\ 98.69 \times 0.7\ 69.08 \approx\ 69.08\%$ or 69%). The overall results are shown in Table 1.

Table 1. Summary of the results of different experiments

Summary of the Experiment Results		
Situations in a Simulator	***Average Waiting Time in Simulator (seconds)***	***Potential Real-World Impact (Improvement %)***
Normal Situation	9.16	-
Optimized with Fuzzy Logic	0.26	68%
Optimized with Reinforcement Learning	0.12	69%

3.4. Summary of the results

The total number of vehicles processed in all three cases is 70; the numbers are fixed to understand different situations better. To clear all the vehicles in the simulator, the three situations took Normal Situation: 263 seconds, Fuzzy Logic Based Situation: 156 seconds, and RL-based Situation: 150 seconds. We calculated the traffic flow rate with Equation (13). The summary of the results is shown in Table 2.

$$Traffic_{Flow} = \frac{Total\ Vehicles\ Processed}{Simulation\ Duration} \times 3600 \tag{13}$$

Table 2. The summary of the results

Method	Average Waiting Time (seconds)	Total Time (seconds)	Total Vehicles Processed	Traffic Flow (vehicles/hour)
Normal	9.16	263	70	958.17
Fuzzy Logic	0.26	156	70	1615.38
RL-based Control	0.12	150	70	1680

We visualize these results with a graph, shown in Fig. 8, illustrating the effectiveness of different traffic light control methods in optimizing traffic flow over the span of one hour. The Reinforcement Learning (RL)-based approach demonstrates the highest efficiency, allowing a maximum traffic flow of 1,680 vehicles per hour. Following closely behind, the Fuzzy Logic-based method achieves a slightly lower traffic flow, managing 1,615 vehicles per hour. In contrast, the traditional traffic light control system, which operates based on predefined timing sequences without any adaptive adjustments, results in the lowest traffic flow, averaging only 958.17 vehicles per hour. This comparison highlights the superiority of adaptive and intelligent traffic control systems in managing congestion and improving overall traffic efficiency.

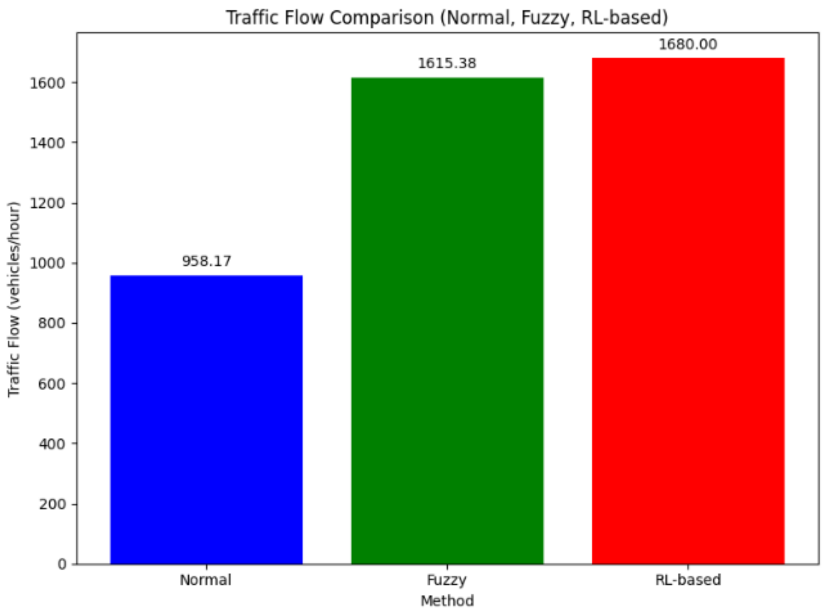

Fig. 8. Traffic flow comparison among normal situation, Fuzz logic based situaion, and RL-based Situation

3.5. Discussion

In a simulator like SUMO, experiments can be controlled and optimized to yield ideal results. However, predicting the exact average waiting time in the real world based solely on the simulation's average waiting time in SUMO (Simulation of Urban MObility) is challenging because of the numerous variables and complexities involved in real-world traffic, such as weather, accidents, human behavior, and infrastructure limitations. For instance, the simulation may rely on certain assumptions, such as idealized traffic patterns, uniform driver behavior, or perfect sensor accuracy, which may not hold true in real-world scenarios. These assumptions can lead to discrepancies between simulated and real-world results. Additionally, the simulation environment may have limitations, such as simplified road networks, lack of real-time dynamic traffic updates, or insufficient modeling of external factors like weather conditions or emergencies. Potential sources of error, such as computational inaccuracies or incomplete data inputs, should also be acknowledged, as they can impact the reliability of the simulation results. While simulation might work well for a small or controlled environment, scaling it to larger, more complex urban networks requires addressing issues like increased computational load, system latency, and the integration of diverse traffic management systems. Additionally, system latency is another critical factor, as real-world traffic systems demand near-instantaneous responses to dynamic conditions, which may be difficult to achieve with current technology. Compliance issues also arise, as real-world deployments must adhere to regulatory standards and ensure interoperability with existing infrastructure and vehicles. Furthermore, the success of real-world implementations heavily depends on the development and deployment of robust V2I and V2V [34] communication systems, which are essential for enabling real-time data exchange and coordination between vehicles and infrastructure. These systems must be reliable, secure, and capable of handling high volumes of data under varying conditions. Real-world implementation requires robust vehicle-to-infrastructure (V2I) and vehicle-to-vehicle (V2V) [34] communication systems, which are not yet fully developed. Scaling from a simulation to a real-world network also introduces challenges, including managing diverse traffic flows and unpredictable driver behaviors. But, the simulation's average waiting time can provide a reference point or baseline for understanding potential performance. Careful consideration of these factors and robust testing and incremental deployments can help bridge the gap between simulation and real-world performance. The improvements seen in simulations provide a strong foundation and justification for further research and real-world trials, potentially leading to more efficient and intelligent traffic management systems. For future research, We aim to integrate computer vision and agent control systems. The trained agent would manage traffic lights based on live video feeds, observing both vehicles and pedestrians on the road. Additionally, we plan to incorporate emergency vehicles and account for the random behaviors of different vehicles. We also intend to simulate real-world traffic conditions using synthesized and simulator data to understand the agent's behavior better and improve its decision-making in varied traffic scenarios. This approach could significantly enhance traffic light optimization systems.

4. Conclusion

In our previous works, we focused on real-world scenarios, but due to limitations, we changed our research into a simulator where we applied our fuzzy logic to control the traffic lights, which improved the waiting time from 9.16 to 0.26 seconds. However, fuzzy logic cannot adapt to the environment; it only focuses on some logic and heavily depends on the data. Thus, we changed our approach and applied an RL-based method where we trained a model to control the traffic lights based on the environment. We trained an agent to control the traffic light to maximize the traffic flow in a minimum time. We used the reinforcement learning algorithm, DQN to train the agent to learn with a greedy policy where the agent tries to maximize the reward by taking actions. We trained our model based on one junction with two roads. There were only two states, and only two actions could be taken. Trained agents trained in reinforcement learning algorithms showed promising results. In this case, we see better results; it reduces waiting time from 9.16 seconds to 0.12 seconds. By enabling traffic lights to learn and adapt to dynamic road conditions, RL-based systems can offer more efficient and flexible traffic management than traditional, pre-programmed systems. These RL agents can continuously adjust traffic signal timings based on real-time data, such as vehicle density, pedestrian movement, and the presence of emergency vehicles, ensuring that traffic flow remains optimal and reduces congestion. One of the key benefits of using RL for traffic light optimization is its ability to handle complex, ever-changing traffic patterns. RL in traffic light systems could significantly reduce waiting times for vehicles and pedestrians, improving overall traffic efficiency. This reduction in waiting times can ease congestion and reduce fuel consumption and emissions, contributing to more sustainable urban transportation. Furthermore, by optimizing traffic signal timings, RL could help mitigate common urban problems such as traffic bottlenecks, inefficient intersection management, and the risk of accidents. The strength of the methodology lies in its use of state-of-the-art technology, which has the potential to reduce traffic congestion in real-time significantly. The system's ability to adapt dynamically and optimize traffic light control through reinforcement learning represents a promising advancement in traffic management. However, a limitation of the current approach is that it remains unrealistic for immediate deployment in real-world settings due to the complexities involved, such as the need for robust infrastructure and real-time data collection. Despite this, expanding our research to create a real-world simulator could offer valuable insights, allowing the agent to learn more effectively through trial and error. Furthermore, by combining the reinforcement learning agent with computer vision capabilities, the system could utilize camera footage to analyze traffic situations more accurately. This integration would enable the agent to make more informed decisions based on the current road conditions, improving the adaptability and performance of the system in real-world scenarios. Acknowledging these strengths and limitations helps in recognizing the potential for future research improvements, which could lead to more practical and effective traffic light optimization solutions. The trained model successfully controls the traffic light in a minimum time but has some problems due to greedy policy, which is inappropriate for the real world; thus, we used mathematical equations to adjust the time to calculate the improvement of real-world impact. In our future work, we will experiment with more data from the real world to estimate the authentic impact that can improve traffic light management systems. The study explores reinforcement learning's potential in optimizing traffic light systems to reduce congestion and improve traffic flow. It shows promising results in reducing vehicle waiting times in simulations. However, real-world implementation presents challenges like advanced communication systems and traffic pattern variability. Future research should explore integrating computer vision with reinforcement learning for more accurate decision-making. While the current study lays a strong foundation for future work in traffic management, addressing the limitations and expanding on the findings could lead to significant improvements in urban transportation systems. Future investigations should focus on resolving real-world implementation challenges, improving system adaptability, and exploring new ways to integrate emerging technologies for more efficient traffic control.

Declarations

Author contribution. Mohammad Mehedi Hassan: Conceptualization, Methodology, Software, Writing—Original Draft Preparation. Stephen Karungaru: Investigation, Data Curation, Visualization, Supervision, Project Administration, Writing—Review & Editing. Kenji Terada: Resources, validation—Review & Editing. All authors have read and agreed to the published version of the manuscript.

Funding statement. This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors

Conflict of interest. The authors declare no conflict of interest.

Additional information. The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

References

[1] B. Pishue, "INRIX Global Traffic Scorecard Reveals Return to Office Drives Congestion and Trips to Downtown," *INRIX*, 2025. [Online]. Available at: https://inrix.com/blog/inrix-global-traffic-scorecard-reveals-return-to-office-drives-congestion-and-trips-to-downtown/.

[2] S. South, "How Do Traffic Light Control Systems Actually Work?," *Signal Equipment Company South*, 2021. [Online]. Available at: https://secosouth.net/2021/10/how-do-traffic-light-control-systems-actually-work/.

[3] F. I. Ariwibowo and D. P. Ismi, "Transforming traffic surveillance: a YOLO-based approach to detecting helmetless riders through CCTV," *Sci. Inf. Technol. Lett.*, vol. 3, no. 1, pp. 45–58, May 2022, doi: 10.31763/sitech.v3i1.1216.

[4] X. Wang, X. Dong, Z. Zhang, and Y. Wang, "Transportation carbon reduction technologies: A review of fundamentals, application, and performance," *J. Traffic Transp. Eng. (English Ed.*, vol. 11, no. 6, pp. 1340–1377, Dec. 2024, doi: 10.1016/j.jtte.2024.11.001.

[5] M. Ayoubi, H. Aman, R. Akbari, and H. Lodin, "AI-Enabled Traffic Light Control System: An Efficient Model to Manage the Traffic at Intersections using Computer Vision," *Int. J. Integr. Sci. Technol.*, vol. 2, no. 8, pp. 767–794, Sep. 2024, doi: 10.59890/ijist.v2i8.2438.

[6] H. M. Zinzow and S. M. Jeffirs, "Driving Aggression and Anxiety: Intersections, Assessment, and Interventions," *J. Clin. Psychol.*, vol. 74, no. 1, pp. 43–82, Jan. 2018, doi: 10.1002/jclp.22494.

[7] P. M. Simon, "Does Radio Need Weather and Traffic Reports Anymore?," *Barrett Media*, 2024. [Online]. Available at: https://barrettmedia.com/2024/08/05/does-radio-need-weather-and-traffic-reports-anymore/.

[8] S. M. Masfequier Rahman Swapno, S. Nuruzzaman Nobel, R. A C, M. Babul Islam, R. Haque, and M. M. Rahman, "Traffic Light Control Using Reinforcement Learning," in *2024 International Conference on*

Integrated Circuits and Communication Systems (ICICACS), Feb. 2024, pp. 1–7, doi: 10.1109/ICICACS60521.2024.10498933.

[9] R. Hussain and S. Zeadally, "Autonomous Cars: Research Results, Issues, and Future Challenges," *IEEE Commun. Surv. Tutorials*, vol. 21, no. 2, pp. 1275–1313, 2019, doi: 10.1109/COMST.2018.2869360.

[10] C. Badue *et al.*, "Self-driving cars: A survey," *Expert Syst. Appl.*, vol. 165, p. 113816, Mar. 2021, doi: 10.1016/j.eswa.2020.113816.

[11] H. A. Ignatious, H.-E.- Sayed, and M. Khan, "An overview of sensors in Autonomous Vehicles," *Procedia Comput. Sci.*, vol. 198, pp. 736–741, Jan. 2022, doi: 10.1016/j.procs.2021.12.315.

[12] S. McLachlan, E. Kyrimi, K. Dube, N. Fenton, and B. Schafer, "The Self-Driving Car: Crossroads at the Bleeding Edge of Artificial Intelligence and Law," *arXiv Comput. Languag*, pp. 1–27, 2022, [Online]. Available at: http://arxiv.org/abs/2202.02734.

[13] A. M. de Souza, C. A. R. L. Brennand, R. S. Yokoyama, E. A. Donato, E. R. M. Madeira, and L. A. Villas, "Traffic management systems: A classification, review, challenges, and future perspectives," *Int. J. Distrib. Sens. Networks*, vol. 13, no. 4, p. 155014771668361, Apr. 2017, doi: 10.1177/1550147716683612.

[14] M. Mehedi Hassan, S. Karungaru, and K. Terada, "An intelligent traffic control systems using on-road cameras," in *Fifteenth International Conference on Quality Control by Artificial Vision*, Jul. 2021, vol. 11794, p. 49, doi: 10.1117/12.2589192.

[15] M. M. Hassan, S. Karungaru, and K. Terada, "Application of On-Road Surveillance Cameras for Traffic Light Optimization," in *2022 Joint 12th International Conference on Soft Computing and Intelligent Systems and 23rd International Symposium on Advanced Intelligent Systems (SCIS&ISIS)*, Nov. 2022, pp. 1–6, doi: 10.1109/SCISISIS55246.2022.10001932.

[16] L. A. Zadeh, "Fuzzy Logic," in *Encyclopedia of Complexity and Systems Science Series ((ECSSS))*, Springer, New York, NY, 2023, pp. 19–49, doi: 10.1007/978-1-0716-2628-3_234.

[17] S. Karungaru, "Advanced Safety Vehicle (ASV) Technology Driver Support System Monitor using Three Onboard Camera," *J. Inst. Ind. Appl. Eng.*, vol. 6, no. 1, pp. 21–28, Jan. 2018, doi: 10.12792/JIIAE.6.21.

[18] S. Karungaru, H. Nakano, and M. Fukumi, "Road Traffic Signs Recognition Using Genetic Algorithms and Neural Networks," *Int. J. Mach. Learn. Comput.*, pp. 313–317, 2013, doi: 10.7763/IJMLC.2013.V3.329.

[19] A. Chattaraj, S. Bansal, and A. Chandra, "An intelligent traffic control system using RFID," *IEEE Potentials*, vol. 28, no. 3, pp. 40–43, May 2009, doi: 10.1109/MPOT.2009.932094.

[20] X. Wang *et al.*, "Traffic light optimization with low penetration rate vehicle trajectory data," *Nat. Commun.*, vol. 15, no. 1, p. 1306, Feb. 2024, doi: 10.1038/s41467-024-45427-4.

[21] R. Vani, N. Thendral, J. C. Kavitha, and N. P. G. Bhavani, "Intelligent Traffic Control System with Priority to Emergency Vehicles," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 455, no. 1, p. 012023, Dec. 2018, doi: 10.1088/1757-899X/455/1/012023.

[22] V. Bhilawade and L. K. Ragha, "Intelligent Traffic Control System," *Int. J. Sci. Res. Publ.*, vol. 8, no. 2, pp. 571–574, 2018, [Online]. Available at: https://www.ijsrp.org/research-paper-0218.php?rp=P747241.

[23] S. S. Leal and P. E. M. de Almeida, "Traffic light optimization using non-dominated sorting genetic algorithm (NSGA2)," *Sci. Rep.*, vol. 13, no. 1, p. 15550, Sep. 2023, doi: 10.1038/s41598-023-38884-2.

[24] H. Zhang *et al.*, "Traffic Light Optimization Based on Modified Webster Function," *J. Adv. Transp.*, vol. 2021, no. 1, pp. 1–10, Aug. 2021, doi: 10.1155/2021/3328202.

[25] B. Wang, Z. He, J. Sheng, and Y. Chen, "Deep Reinforcement Learning for Traffic Light Timing Optimization," *Processes*, vol. 10, no. 11, p. 2458, Nov. 2022, doi: 10.3390/pr10112458.

[26] F. Outay, F. Kamoun, F. Kaisser, D. Alterri, and A. Yasar, "V2V and V2I Communications for Traffic Safety and CO2 Emission Reduction: A Performance Evaluation," *Procedia Comput. Sci.*, vol. 151, pp. 353–360, Jan. 2019, doi: 10.1016/j.procs.2019.04.049.

[27] R. Monga and D. Mehta, "Sumo (Simulation of Urban Mobility) and OSM (Open Street Map) Implementation," in *2022 11th International Conference on System Modeling & Advancement in Research Trends (SMART)*, Dec. 2022, pp. 534–538, doi: 10.1109/SMART55829.2022.10046720.

[28] C. S. Bari, Y. V. Navandar, and A. Dhamaniya, "Analysis of Vehicle Specific Acceleration and Deceleration Characteristics at Toll Plazas in India," *Transp. Dev. Econ.*, vol. 7, no. 1, p. 4, Apr. 2021, doi: 10.1007/s40890-021-00115-6.

[29] D. Danaher, S. McDonough, D. Donaldson, and R. Cochran, "A Naturalistic Study of Vehicle Acceleration and Deceleration at an Intersection," *J.S. Held Insights*, 2022. [Online]. Available at: https://www.jsheld.com/insights/articles/a-naturalistic-study-of-vehicle-acceleration-and-deceleration-at-an-intersection.

[30] E. Altman, *Constrained Markov Decision Processes*. Boca Raton: Routledge, pp. 1-257, 2021, doi: 10.1201/9781315140223.

[31] J. F. Hernandez-Garcia and R. S. Sutton, "Understanding Multi-Step Deep Reinforcement Learning: A Systematic Study of the DQN Target," *arxiv Artif. Intell.*, no. Nips, pp. 1–12, 2019, [Online]. Available at: https://arxiv.org/abs/1901.07510.

[32] J. TORRES, "The Bellman Equation," *Towards Data Science*, 2020. [Online]. Available at: https://towardsdatascience.com/the-bellman-equation-59258a0d3fa7/.

[33] B. Jang, M. Kim, G. Harerimana, and J. W. Kim, "Q-Learning Algorithms: A Comprehensive Classification and Applications," *IEEE Access*, vol. 7, pp. 133653–133667, 2019, doi: 10.1109/ACCESS.2019.2941229.

[34] A. R. Khan *et al.*, "DSRC Technology in Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) IoT System for Intelligent Transportation System (ITS): A Review," in *Lecture Notes in Electrical Engineering*, vol. 730, Springer, Singapore, 2022, pp. 97–106, doi: 10.1007/978-981-33-4597-3_10.