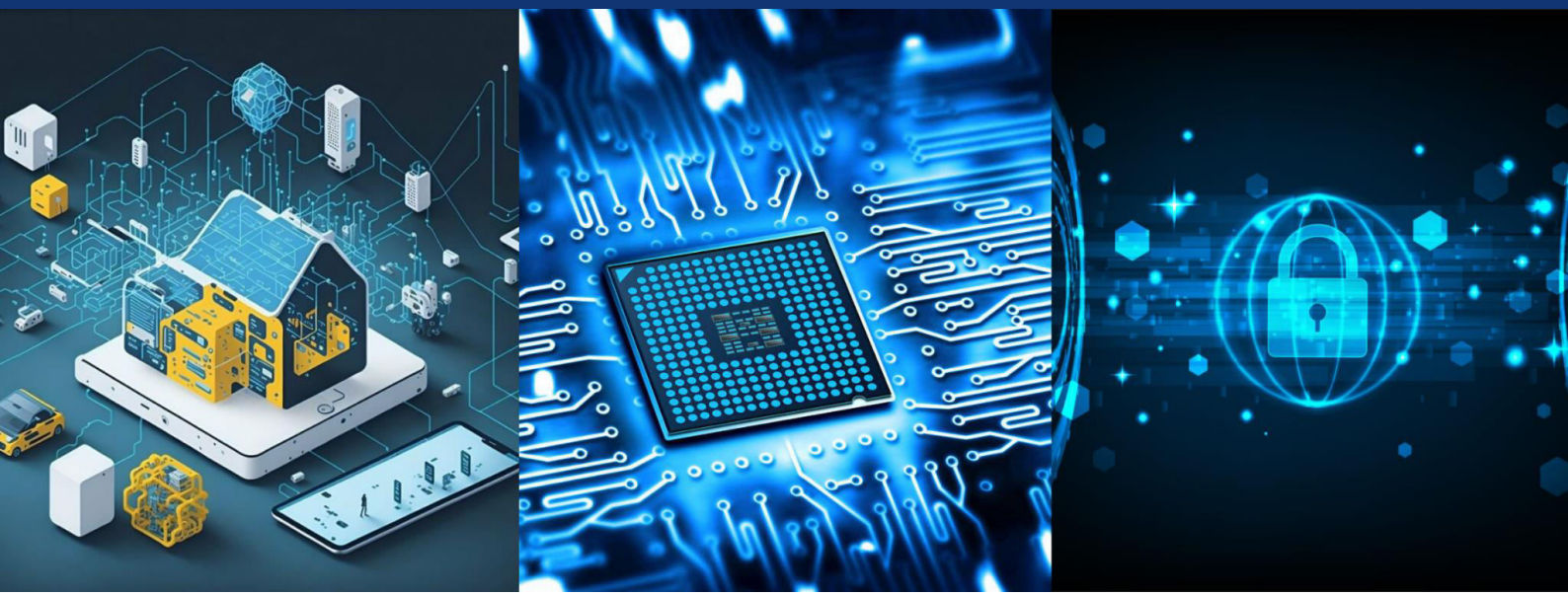
IJIRCCE

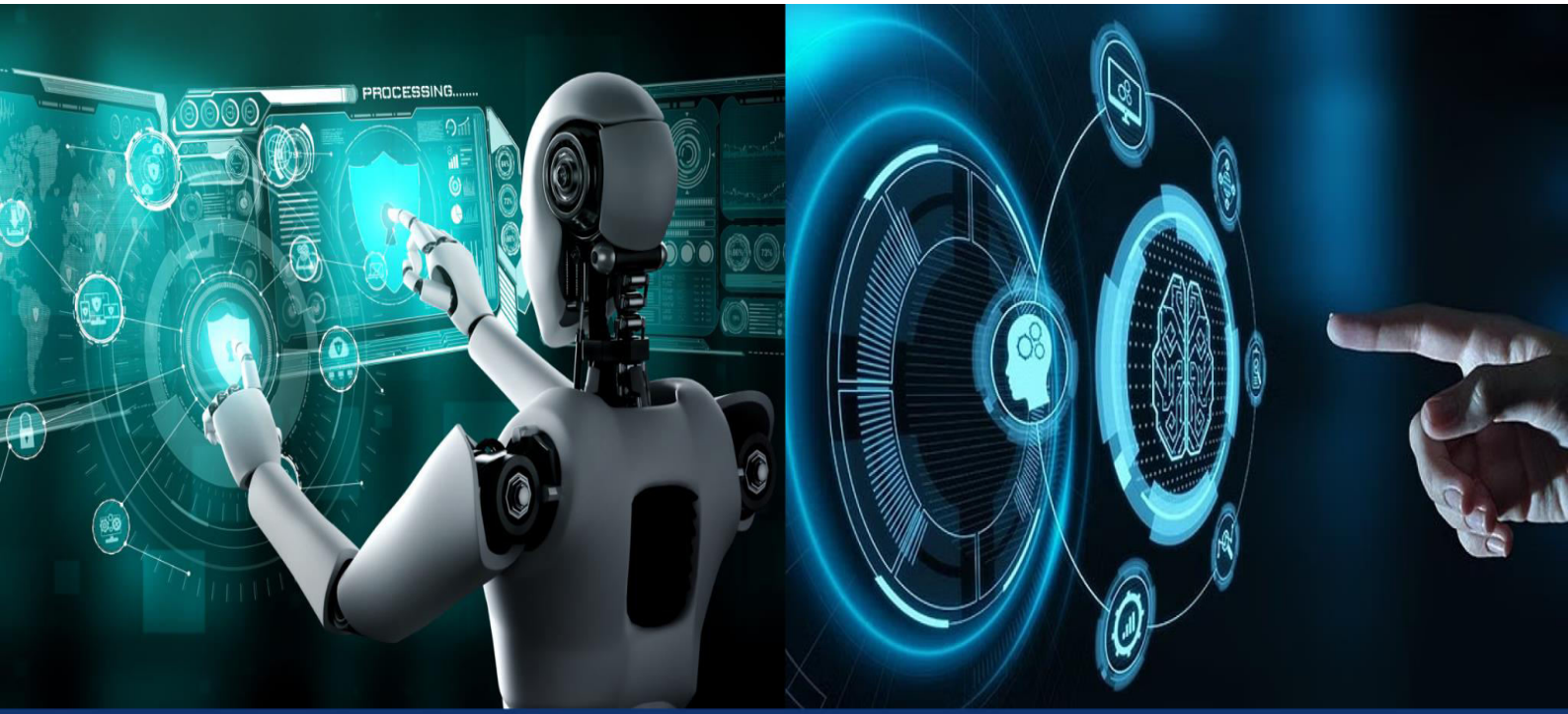
ISSN(O): 2320-9801

ISSN(P): 2320-9798



International Journal of Innovative Research in Computer and Communication Engineering

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



Impact Factor: 9.927

Volume 14, Issue 8, August 2026

© 2026 IJIRCCE | Volume 14, Issue 8, August 2026 DOI: 10.15680/IJIRCCE.2026.1408026

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

An End-to-End Off-policy Deep Reinforcement Learning Framework for Adaptive Traffic Signal Control

Syeda Rifza Fathima, Prof. Michelle D Souza

M.Tech Student, Department of Computer Science and Engineering, Maharaja Institute of Technology, Mysore, Karnataka, India

Assistant Professor, Department of Computer Science and Engineering, Maharaja Institute of Technology, Mysore, Karnataka, India

ABSTRACT: An efficient transportation system can substantially benefit our society, but road intersections have always been among them traffic bottlenecks leading to traffic congestion. Appropriate traffic signal timing adapted to real-time traffic may help mitigate such traffic congestion. However, most existing traffic signal control methods require a huge amount of road information, such as vehicle positions. In this paper, we focus on a particular road intersection and aim to minimize the average waiting time. We propose a traffic signal control (TSC) system based on an end-to-end off-policy deep reinforcement learning (deep RL) agent with background removal residual networks. The agent takes real-time images at the road intersection as input. Upon sufficient training, the agent can perform (near-) optimal traffic signaling based on real-time traffic conditions. We conduct experiments on different intersection scenarios and compare various TSC methods. The experimental results show that our end-to-end deep RL approach can adapt to the dynamic traffic based on the traffic images and outperforms other TSC methods.

KEYWORDS: Deep reinforcement learning, deep learning, artificial intelligence, traffic signal control systems, background removal residual learning

I. INTRODUCTION

An efficient transportation system with smooth traffic can benefit our society in various aspects, such as the economy, environment, and human health. Relieving city traffic congestion using various technologies has been actively studied in recent years. For example, a dynamic lane reversal routing and scheduling system has been proposed to reduce traffic congestion by dynamically reversing lane directions based on real-time traffic [1]. One critical traffic bottle neck is at the road intersections [2],at which vehicles have to stop For red lights.

Traditionally, we rely on pre-timed traffic signal control (TSC), which contains “green”, “yellow”, and “red” phases in a cycle with pre-defined durations based on human decision or historical traffic flow. However, such pre-timed control method cannot account for the dynamic traffic flow, leading to ineffective traffic control. In order to take dynamic traffic into consideration, many control approaches have been proposed to optimize the traffic timing. Due to the “curse of dimensionality” of the highly dynamic transportation system, the number of possible state-action pairs is so large that the ordinary control methods become computationally intractable. To overcome this, artificial intelligent techniques are considered for TSC.

Among these techniques, Reinforcement Learning (RL) [3], also called Approximate Dynamic Programming (ADP) [4],is primarily designed for control problems and it learns to perform the (near-) optimal action based on given observations of the system. supervised learning technique, most of the RL algorithms are model-free such that self-learning is performed without requiring additional system knowledge such as the system dynamics. RL addresses the issue of curse of dimensionality” by using neural networks to approximate the corresponding cost function of the state-action pairs. Therefore, it has become popular to study RL-based algorithms for TSC-related problems [5].

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

When RL is applied to TSC, state observations related to the roads and vehicles are essential elements for the RL agent to determine the corresponding actions. The state observations are usually real-time information related to the roads and vehicles, such as queue size, red phase timing, phase of current traffic light, physical positions of waiting vehicles at an inter-section, and discredited vehicle position matrix [6]. Thus, those RL systems found in the literature were usually built on the assumption that the detailed and accurate vehicle information is always available. Without built-in sensors, the TSC system can only rely on external on-road sensing devices to determine the states but the cost will be very high to cover every single road intersection. Although there are new vehicular communication technologies developed to enable on-road applications [7], the accuracy and the cost of sensors may induce other problems. Recently, there are many break through on computer vision and pattern recognition, such as object categorization, action recognition, and motion tracking, using advanced deep learning techniques. Recognizing vehicles in images captured by cameras is no longer a challenging task and the cost of cameras is relatively low. In fact, cameras can be used as the major sensors in transportation systems. Moreover, the handcrafted features to be recognized are mainly based on human intuition, which may not provide the right level of abstraction for control. In [8], an agent with joint perception and control system was trained by Deep Reinforcement Learning (deep RL) to achieve human level proficiency on playing video games based on the video screen only and without domain knowledge of the game. A similar deep RL algorithm can also be applied to end-to-end traffic light control system to avoid expensive sensors. Hence, a simple and unified end-to-end system with joint perception and control, that directly observes real-time road images to perform (near-) optimal actions, is of huge practical application value.

In this paper, we present a TSC system that takes real-time road intersection images as state input and performs (near-) optimal traffic signaling based on dynamic traffic. We aim to minimize the average waiting time of the vehicles at a particular intersection. To the best of our knowledge, we are among the first to propose an end-to-end off-policy deep RL agent with background removal residual networks for TSC based on real-time traffic images. We conduct experiments on different intersection scenarios and compare with various TSC methods. The experimental results show that the proposed end-to-end deep RL method can adapt to dynamic traffic based on the images and outperforms other TSC methods. The main contributions of this paper can be summarized as follows:

1. We propose an off-policy end-to-end deep RL based TSC system to control traffic signaling for dynamic traffic, aiming to minimize the average waiting time at the intersection.
2. The background removal residual networks are proposed for a Q-network so as to focus on the moving vehicles rather than other unrelated background information in the images.
3. The traffic condition is represented solely by recaptured by road surveillance cameras and other expensive on-road or built-in sensors are not required.
4. The traffic intersection model is carefully designed in compliance with safety and practical considerations.
5. Extensive experiment son different intersection scenarios are conducted to evaluate our solution with various TSC methods, and it is shown that the end-to-end deep RL method can adapt to the dynamic traffic and outperform other compared methods.

II. Related Work

In this section, we review the existing work on intelligent TSC systems using various methods.

A. Traffic Signal Control System

Intelligent TSC systems have been widely studied for decades as a possible replacement for the traditional pre-timed controller operating in fixed cycle length, fixed duration of each stage, and fixed sequence, with a fully actuated controller which adjusts the duration of each stage based on traffic demand. In [9], a dynamic programming approach was used to develop an optimal real-time demand-responsive TSC but it was computationally intensive. Building a fuzzy system for intelligent TSC was an early successful attempt [10]. [11] developed a set of fuzzy decision rules for TSC to control the traffic signal timing based on cycle time, phase split, and offset parameters. Some others focused on fuzzy-neural decision making [12], [13] for large-scale TSC, where a local agents responsible for a single intersection and the decisions are cooperatively combined by the higher-level agents coordinating multiple intersections.

A type-2 fuzzy set was introduced in [14] to develop a TSC system that can handle uncertainty in the inputs and the setoff rules of the controller. Besides fuzzy system, other computational intelligence algorithms were also considered.[15] has modeled the traffic flow and density using Cell Transmission Model [16] and has formulated the TSC problem as a mixed integer program. [17] has modeled the TSC problem with predicted traffic flow, which were decomposed into sub-problems in a hierarchical fashion. However, the prediction accuracy is one of the key elements

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

that affect the performance. Another line of study for traffic signal control is Max Pressure Control [18]. The idea of Max Pressure Control is to minimize the "pressure" of an intersection, which is defined as the number of queuing vehicles on entering lanes minus the number of queuing vehicles on exiting lanes. It has been shown that the throughput of an intersection can be maximized by using Max Pressure Control [18]. In [19], the authors, inspired by the back-pressure algorithms developed for routing and scheduling in communication networks, have proposed a multi commodity back pressure schemes for traffic signal control. In [20], the Generalized Proportional Allocation controller has been proposed to produce a similar optimal throughput solution by using the local queue length, which does not require information of the traffic flow turning into different directions. [21] has proposed a simplified model to capture the dynamics of traffic networks which was capable of performing computationally tractable and network-wide optimization of the green-splits durations at intersections.

B. Reinforcement Learning

Q-learning is a popular and commonly used algorithm in RL [22]. In general, Q-learning tries to determine or learn the Q-value of each state-action pair, and the Q-value can be considered as a cumulative utility of taking Action a under State s. After the Q-values are explored and established, the action with the highest Q-value is selected to be performed .In [23], Q-learning with lookup table representation for Q-values has been proposed for TSC. However, such lookup tables grow exponentially with the number of vehicles, road lanes, and intersections, leading to intensive computation. To deal with this "curse of dimensionality" issue, [24] has proposed a temporal-difference based RL algorithm to approximate the Q-value function with a linear approximate function while [25] has presented a function-based approximation for the Q-value function using a linear combination of tunable weight vectors and feature vectors. Multi-agent RL has also been investigated, where the main problem was decomposed into smaller sub-problems and each agent corresponded to a single road intersection (i.e., sub-problem) [26], [27]. The agents shared information, such as rewards and Q-values, to determine the optimal joint actions. In [28], the max-plus algorithm has been designed as a message passing strategy to share information between a pair of nodes in a graph. Beside rewards and Q-values, the shared information can be the controller actions and each agent converges to the best action based on its neighbor's actions [29]. The Q-value function can also be distributive updated based on the feedback of loss function from its neighbors [30].

C. Deep Learning

In recent years, there are many breakthroughs in computer vision and pattern recognition based on deep learning techniques. For example, Alex Net was a deep convolutional neural network and achieved superior performance on visual recognition tasks [31]. Another important benchmark is Res Net [32] that used a residual connection in CNN and achieved human level performance on visual recognition task. Meanwhile, the success of deep learning has motivated many fields of studies including transportation system. In fact, neural networks are not restricted to visual recognition. They can perform other tasks such as traffic data forecasting and decision making if we designate the inputs and outputs appropriately. For example, a deep neural network has been used to predict short-term traffic flow with high accuracy [33], [34]. Lane detection [35] and autonomous driving [36] have also been facilitated by deep learning. A novel deep learning model, called Multi-Scale Convolutional LSTM Network, has been proposed to predict travel demand [37] and origin destination pairs [38] by considering temporal and spatial correlations of the historical traffic data. These kinds of computer vision and pattern recognition techniques can be applied to TSC if we train the neural network to output suitable control signaling. We can see that using deep learning on intelligent transportation system promising in many unexplored research questions.

D. Deep Reinforcement Learning

Deep RL is a recently emerged field that makes significant improvement to RL by integrating deep learning architecture with RL algorithm to handle very hard control problems [39], [40].A successful example is playing video games with human level proficiency. This is a very hard control problem when considering the video screen as state observation and the console as action input [41]. One popular algorithm in deep RL is Deep Q-learning, which uses a deep neural network and an experience replay buffer for Q-value approximation[8]. The deep neural network, such as convolutional neural network (CNN), supports high dimensional approximation and out performs traditional shallow feed-forward neural networks while the experience play buffer stores the state, action, and reward tuple for later training. For TSC, researchers have used a deep stacked auto-encoder [42] and a deep CNN [43] for approximating Q-values of state-action pairs, where the states consist of vehicle position, speed, and queue length. To better represent the vehicle position on the road, some systems utilize a discretized image-like matrix, where the matrix elements indicate the existence of vehicles or vehicle speeds at the corresponding positions as the state observation. In [44], each state

© 2026 IJIRCCE | Volume 14, Issue 8, August 2026 DOI: 10.15680/IJIRCCE.2026.1408026

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

observation is composed of queue length, waiting time, traffic signal phase, number of vehicles, and the matrix representation. In [6], only the image-like matrix is required as input with the deep RL agent .However, this matrix representation may discretize the vehicle position, speed, and other hidden information, which may degrade the performance due to the loss of high precision information. For multiple intersections, a max-plus algorithm can be used to optimize the joint action over the connected road intersections [45].

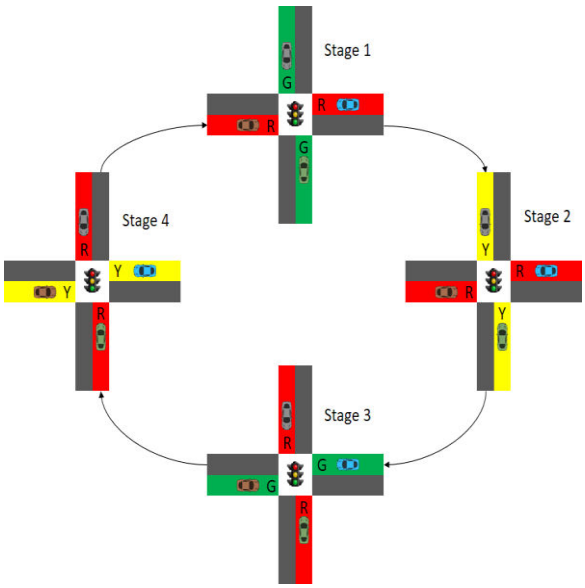


Fig.1. The four stages of traffic signaling at a four-way interaction.

Similarly, agents can cooperate by sharing the trained policy at each training episode with other agents [46]. A hierarchical structure has been designed in [47], in which the local policy and value function learned by each agent were aggregated at the centralized global agent to form the final Q-function for the entire region. These research projects provide good references on developing multi agent deep RL for a large-scale TSC system.

III. PROBLEM DEFINITION

Consider a four-way road intersection consisting of four connected road segments: **North, East, South, and West**. Each road segment allows vehicles to approach the intersection from a different direction. To ensure safe and organized vehicle movement, traffic signals are deployed at the intersection to regulate the right-of-way of vehicles. Each traffic signal operates in one of three basic states: **Green, Yellow, and Red**. During the **green phase**, vehicles approaching from the corresponding direction are permitted to enter and pass through the intersection. During the **red phase**, vehicles must stop before the stop line and wait until the signal becomes green. The **yellow phase** serves as a transition period between green and red, warning approaching vehicles that the available crossing time is ending and that the signal will soon change to red. For the proposed traffic-control framework, the four-way intersection is divided into **four coordinated traffic stages**. Each stage determines which traffic movements are allowed to proceed while conflicting movements remain stopped. A simplified stage sequence can be represented as:

1) Green for traffic from North-South and red for East-West.
2) Yellow for traffic from North-South and red for East-West.
3) Red for traffic from North-South and green for East-West.
4) Red for traffic from North-South and yellow for East-West.

These four stages are cycled in the above sequence as shown in Fig.1. In practice, it is not allowed to skip any stage and to cycle in the reversal order, but simplified models without the yellow phase can be found in [44] and [47]. In traditional traffic signaling, the duration of each stage is fixed and dynamic traffic situations are not taken into account. To adapt to dynamic traffic, an agent is needed to control the timing of the stages. Since they ellow phases inStages2

© 2026 IJIRCCE | Volume 14, Issue 8, August 2026 DOI: 10.15680/IJIRCCE.2026.1408026

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

and 4 are used to alert the changing to the red phase for a smooth transition, their duration shave to be fixed in order to allocate enough reaction time for drivers. Thus the agent mainly focuses on determining the time duration of Stages 1 and 3 based on real-time traffic.

The intersection can be modeled by a 4-tuple Markov decision process ⟨S,A,P,R⟩, where S and A are the set of states and actions, respectively. P(s′|s,a) represents the state transition probability from States s∈ S to s′ ∈ S for performing Action a∈ A.R(s,a,s′)is the received reward due to the transition from s to s′after taking a. At time t, the strategy of the agent with Action a_t given State s_t is defined by Policy π:S→A such that $\pi(s_t)=a_t$.We aim to build an agent for TSC with an optimal policy $\pi^*$to determine optimal action a^*based on dynamic traffic such that the cumulative reward (total vehicle waiting time at the intersection) is maximized (minimized):

$$\pi^*=\arg\max_{\pi}\sum_{t=0}^{\infty}\gamma^t R(s_t, \pi(s_t), s_{t+1}), \quad (1)$$

Where $0\leq\gamma^t<1$ is the discount factor to prevent an infinite cumulative reward. Details of the deep reinforcement learning algorithm are included in Section IV. The model focused on the traffic signal control will be presented in Section V.

IV. DEEP REINFORCEMENT LEARNING

For a highly dynamic transportation system, the number of possible state-action pairs is very large. A manually designed policy may not be optimal for dynamical traffic since the reward is complicated to derive manually and not all possible states can be fully covered. Therefore, we employed deep RL to address the "curse of dimensionality" and to learn the optimal policy π^*. π^* can be obtained by maximizing the cumulative reward, which is defined as Q-value (a.k.a. state-action value):

$$Q(s,a)=\mathbb{E}\left[\sum_{i=0}^{\infty}\gamma^i r_{t+i}|s_t=s,a_t=a,\pi\right], \quad (2)$$

where r_t is the reward at time t. The optimization problem can be addressed by Dynamic Programming [49].The Bellman optimality equation used to obtain the optimal Qvalue $Q_\pi*(s, a)$can be expressed as

$$Q_{\pi^*}(s,a)=\mathbb{E}\left[r_t+\gamma\max_{a'}Q(s',a')|s,a.\right] \quad (3)$$

For unknown system dynamics, the Q-value function cannot be derived analytically and it has to be empirically obtained based on the returned reward from the environment. Although one may try to store all the returned Q-values, the required memory is extremely large for high dimensional states and actions. In deep RL, a deep neural network trained with Q-value samples is used to generalize and approximate the Q-value for all state-action pairs. For an agent with a well trained Q-value network (Q-network), an action can be selected based on the highest corresponding Q-value.

The training algorithm of Q-network is given in Algorithm
1. It is an off-policy algorithm based on Rainbow [50], which includes several improvements to the Deep Q-network (DQN) algorithm [8]. According to [50] and [51], Multi-step learning [3],Prioritized Experience Replay [52],and Distributional RL
[53] are selected and implemented in our Algorithm. Rainbow and DQN are designed as on-policy deep RL algorithms, where the Q-network is updated based on its previous evaluated transition. For TSC, training of the Q-network should be done prior to implementation since it is dangerous to train the Q-network on site, an on-policy algorithm may not be suitable for our case. Thus we developed an off-policy version such that on-site training is avoided and it can learn from other policies offline.

© 2026 IJIRCCE | Volume 14, Issue 8, August 2026 DOI: 10.15680/IJIRCCE.2026.1408026

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Algorithm 1 Off-policy deep RL algorithm for TSC

Input: off-policy database D, replay buffer M, batch size of replay buffer B, minimum replay buffer value M_{min}, Discount factorγ
Output: policy π
Initialization: Q-network parameters θ
1: Sample M_{min} transitions(s_i,a_i,r_i,s_{i+1}) from D to M
2: **for** t=1 to N **do**
3: Sample transition (s_i,a_i,r_i,s_{i+1}) from D to M
4: Sample B transitions (s_j,a_j,r_j,s_{j+1}) from M according to Eq. (4)
5: Set L_n according to Eq.(5)
6: Update the Q-network parameters θ using Adam [54] optimization
7: **end for**
8: **return** Q-network parameters θ

In Algorithm 1, we first built a database for off-policy training. For each time step t, we obtained the states $_t$, action a_t, reward r_t, and next states$_{t+1}$, where the action was selected based on pre-defined policy, such as pretimed, greedy, and random policy. The transition (s_t,a_t,r_t,s_{t+1}) was then added to the off-policy database. During training, we Sampled M_{min} transitions from the off-policy database D to are play buffer M (Step1). For each training step (Step2), a transition$(s_i,a_i,r_i,s_{i+1})$was added to M(Step 3) and B transitions were sampled from the replay buffer for training (Step 4). With prioritized experience replay technique [52], valuable data were sampled more frequently for training. The sampling probability p_t for the data is proportional to the last encountered absolute temporal difference error, i.e.,

$$p_t \propto |r_{t+1}+\gamma_{t+1}\max Q(s',a')-Q(s,a)|^{\omega}, \quad (4)$$

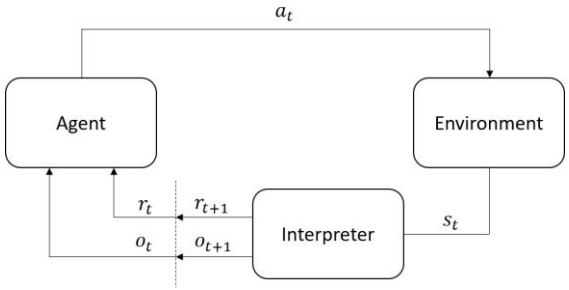


Fig.2. Interaction between the environment and the agent.

To update the parameters of theQ-network based on the loss

$$L_n=\left(\sum_{k=0}^{n-1}\gamma^k r_{t+k+1}+\gamma^n \max_{a\in A} Q'(s_{t+n},a)-Q(s,a)\right)^2 \quad (5)$$

Where Q′, called the target network, is a copy of the Q-network function(Step5):
with parameters updated once every few steps to stabilize the training of the Q-network. N is a hyper-parameter which allows the Q-network to be trained by multi-step rewards [3]. The Q-network only provides the Q-value of the best action if the expected Q-value is structured to be the output. On the other hand, the distribution of Q-value contains more information such as the second best action. Since the Q-value distribution is more suitable in our case, a s in [53], we approximated the distribution of Q-value, instead of the expected Q-value. Finally, the Q-network parameters θ were updated using Adam [54] optimization (Step6).

V. MODEL FOR TRAFFIC SIGNAL CONTROL

© 2026 IJIRCCE | Volume 14, Issue 8, August 2026 DOI: 10.15680/IJIRCCE.2026.1408026

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

In a typical deep RL model, an interpreter is required to obtain the states and rewards from the environment. This information is then passed to an agent for determining the optimal action which will influence the transition of the future states and rewards. Fig. 2 shows the interaction between the environment, the interpreter, and the agent. We followed this general model for the TSC problem and the components are discussed in this section.

A. Environment

The environment refers to the road intersection, represented by State s_t at time t. Action a_t received from the agent is executed in the environment. State s_t transits to the next state s_{t+1} based on the transition probability $P(s_{t+1}|s_t,a_t)$, where the transition probability models the unknown dynamics of the environment.

1) State and Observation: A state is a representation of the physical condition of the environment while an observation is the information received by the agent. They are the same for a fully observable system but different for a partially observable system. Vehicular information, such as vehicle position, speed, and queue length, are usually used as the observation in the literature, but such information may not be accurately measured in practice. Even though there are dedicated sensors available with high accuracy, the cost is usually too high to be implemented at every intersection. In this way, we adopt the quite cost-effective cameras as sensors and the real-time captured images of the road intersection become Observation o ∈O of the agent. For simplicity, we assumed Observation o is equal to States for the fully observable system. Both s and o are used interchangeably in this paper.

2)Reward: We aim to minimize the average waiting time of vehicles at the intersection. The average waiting time W_t at time t is defined as:

$$W_t = \frac{1}{|V|} \sum_{v} w^v, \quad t$$

Where w^v is the waiting time of vehicle vat the intersection at time t^t. W_t may become very large when many vehicles get stuck at the intersection. Hence, we applied a transformation to there ward function R_t to bound It between 0 and 1

$$R_t = \frac{1}{W_t + 1}. \qquad (7)$$

In this way, the reward function inversely depends on the waiting time and this allows the deep RL algorithm to be trained to maximize the given reward (i.e., minimize the waiting times).

Note that the reward function is used in the training phase only for minimizing the waiting time. Since the data for training were collected offline, we assumed either the vehicles are cooperators for generating the dataset or additional processes such as vehicles detection and tracking [55] can be done offline for extracting the waiting time information. After the agent is well-trained, the waiting time information as well as the reward function are not required for actual implementation. Thus, the vehicles normally do not need to transmit the waiting time and we do not need to measure the waiting time information during implementation.

B. Interpreter

The interpreter is defined as the device or process to get the observations from the states. The states S and the observations O can be different in scenarios of partially observable Markov decision process (POMDP) if the states cannot be completely observable. We avoid making the assumption that the states are completely observable since the position and view of the road surveillance camera may not be perfect for every intersection. Therefore, the input to the agent is defined as“ observations” which may be different from some literature that use the term states as the input to the agent since the states are assumed to be fully observable (i.e. state S is equal to observation O), which does not hold in our case.

Cameras and communication units are the interpreter for the TSC problem. We as summed a non-road camera isused to capture the road condition of the intersection. The camera angle should be wide enough to cover all the connected roads at the intersection. During the data collection process, we assumed the waiting vehicles are cooperative and transmit the waiting times to the agent. Note that rewards are only required in the training process and they are not required when

© 2026 IJIRCCE | Volume 14, Issue 8, August 2026 DOI: 10.15680/IJIRCCE.2026.1408026

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

the system is being operated. Therefore, our system does not require the vehicles equipped with sensing devices and communication units for collecting state and reward in practice. All regular and intelligent vehicles can benefit from our system.

C. Agent

An agent is used to determine an appropriate action based on a given observation. It is basically a deep neural network, called Q-network, with trained parameters θ, which takes an observation as input and outputs a distribution of Q-value for action. The training method of the deep neural network is discussed in Section IV.

1) Action: The agent controls the traffic signal in order to minimize the waiting time. For safety reasons, the traffic signal must follow a specific stage sequence and skipping any stage is not allowed. Moreover, the duration of those stages with yellow signal must be fixed. As a result, the agent controls the durations of those stages without yellow signal ,i.e., Stages 1 and 3. At any moment, the agent need s to decide whether it should remain in the same stage or move on to the next. This decision is stage-dependent. We defined a binary action a_t to represent this stage-dependent action. In particular, the action a_t represents which roads should be in green and red. The final decision on traffic signaling is determined from a_t based on the current stage. For example, in Stage 1, a_t=0 represents keeping in the same stage and a_t=1 represents moving on to the next stage. In contrast, the decision in Stage 3 is the opposite of that in Stage 1, where a_t= 0 represents moving onto then exist age and a_t=1 represents keeping in the same stage. The reason is that it may be easier for the agent to learn which roads should be in Red or Green. The stage remains at 1 if the agent repeatedly sends a_t=0, while it remains at 3 if the agent repeatedly sends a_t=1.Thus a_t=0 leads to Stage 1 and a_t=1 leads to Stage 3. The transitions between different traffic signal stages are given in Fig. 3.

2) Q-network: A Q-network was used to estimate the Q-value Q(s ,a) of Action a given State s. Since the deep RL algorithm is designed end-to-end, images are directly input to the Q-network. To handle the images, the Q-network should have excellent performance on image processing tasks. Res Net[32] is one of the state-of-the-art network architectures for image processing task. It performs residual learning, where a building block is defined as:

$$x_{i+1} = F_i(x_i) + x_i, \qquad (8)$$

where x_i and F_i are the input and transfer function of the i^{th} layer neural network, respectively. The bypass connection of x_i provides an identity of the input such that the deep neural network layer $F_i(\cdot)$learns the residual$(x_{i+1}-x_i)$. This addresses the degradation during training of deep neural networks. In this work, vehicles are the major elements which influence traffic signaling, and thus the network will pay attention to the vehicles instead of the background roads. The main motivation of this work is to develop an end-to-end approach for low cost hardware (sensor). To speed up the

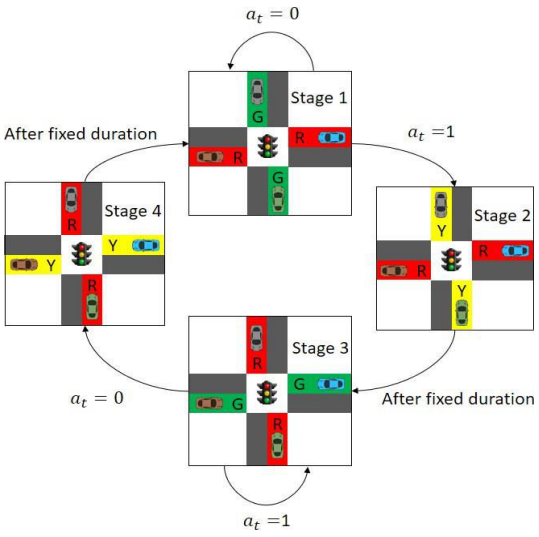


Fig.3.Transitions between different traffic signal stages.

© 2026 IJIRCCE | Volume 14, Issue 8, August 2026 DOI: 10.15680/IJIRCCE.2026.1408026

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

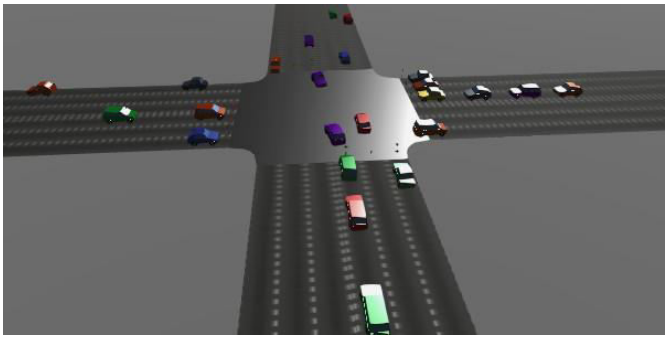


Fig.4.3Dvisualization of an intersection model.

computation, we avoid using the existing image processing techniques which are computationally intensive. Therefore, we designed a background removal Res Net (BGR Res Net) as the Q-network that removes the background automatically:

$$x_2=F_1(x_1-x_{bg})+x_1-x_{bg}, \quad (9)$$

where x_{bg} is the background image of the intersection captured without vehicles. Fig. 4 shows an example of 3D visualization of an intersection model, used as the input x_1 while Fig. 5 gives the background image x_{bg}. Fig. 6 illustrates the corresponding background removed input image, $x_1- x_{bg}$. As shown in the figures, only vehicles remain in the images, as it is easier for the agent to learn from the images without the background.

The architecture of the background removal Q-network is shown in Fig. 7. It contains convolutional layers with residual connection between the layers and outputs the distribution of Q-value of the actions based on the input observation.

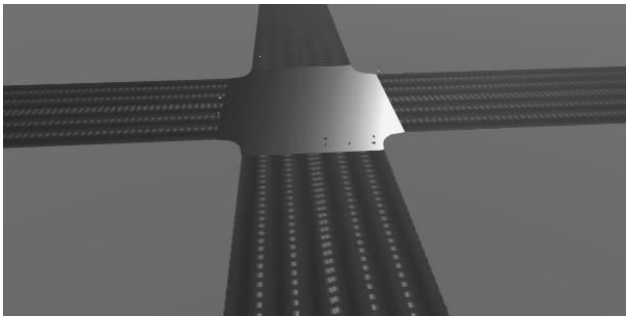


Fig.5.The back ground image of the intersection, x_bg.

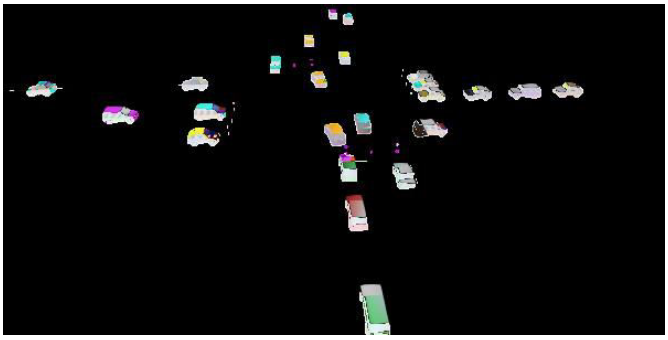


Fig.6.An example of the input image with the back ground removed, x_1-x_{bg}.

© 2026 IJIRCCE | Volume 14, Issue 8, August 2026 DOI: 10.15680/IJIRCCE.2026.1408026

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

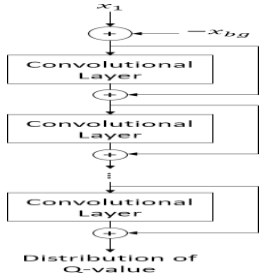


Fig.7.Q-networkarchitecture.

VI. EXPERIMENTS

A. Experimental setup

We evaluated the end-to-end deep RL algorithm for the TSC problem using traffic simulator SUMO [48] and 3D visualization framework Sum o-web 3d[56]. SUMO is a traffic simulator used to simulate intersection environment such as

TABLEI
PARAMETERS OF THE SIMULATED INTERSECTION.

Parameter	Value
Saturation flowrate	2286vehiclesperhour
Jam density	177vehiclespermileperlane
Number of lanes per approach	3

vehiculartrafficandtrafficlightlogicwhileSumo-web3d is a 3D visualization tool to visualize the vehicles and the intersection. All the traffic parameters used in the experiments follow the default settings in SUMO unless specified other-wise. Some important parameters of the simulated intersection are summarized in Table I.

Traditional pre-timed and greedy algorithms were tested in the experiments to be compared with our propose method:

1. Pre-timed: The four stages are sequentially cycled with fixed duration;
2. Max pressure control: The traffic stage of a road transits to or remains at the green phase if the “pressure” on the road is higher than that of other roads, where the “pressure” is defined as the number of queuing vehicles on entering lanes minus the number of queuing vehicles on exiting lanes;
3. Greedy on fleet size: The traffic stage of a road transits to or remains at the green phase if the number of vehicles on the road are higher than other roads;
4. Greedy on waiting time: The traffic stage of a road transits to or remains at the green phase if the total waiting time of vehicles on the road are higher than other roads. The following deep RL algorithms were also compared to evaluate the performance:
5. DQN [8]: Deep RL algorithm to train the deep Q-network;
6. C51 [53]: Deep RL algorithm to train the deep distributional Q-network;
7. Rainbow[50] Deep RL algorithm based on DQN to train the deep Q-network with multi-step learning, prioritized experience replay, and distributional RL.

For each of the deep RL algorithms, two types of deep neural networks, CNN and ResNet, were used as the Q-network. The hyper-parameters of the algorithms are shown in Table II.

We considered both real-world and synthetic scenarios in our experiments for the intersection. For the former, we adopted the traffic flow dataset of a four-way intersection in Colognecity[58]. The traffic flow from the North ,East, South, and West directions per hour of the selected intersection in Cologne city are 738, 138, 312, and 612, respectively. The total number of simulated vehicles is 1800 vehicles per hour. For the latter, the incoming traffic flows of the roads were randomly generated with different probabilities. We built four scenarios with different traffic flow probability ratios to test the effectiveness of the algorithms:

© 2026 IJIRCCE | Volume 14, Issue 8, August 2026 DOI: 10.15680/IJIRCCE.2026.1408026

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

- ScenarioS1: Incoming vehiculartrafficflowsofallroads are 500 vehicles per hour; the total number of simulated vehicles is 2000 vehicles per hour;

TABLEII

HYPER PARAMETER SETTING.

	Parameters	Setting			
Pre timed method	Cycle length	90s	70s	50s	30s
	Stage1duration	39s	29s	19s	9s
	Stage2duration	6s	6s	6s	6s
	Stage3duration	39s	29s	19s	9s
	Stage4duration	6s	6s	6s	6s
DQN[8]	Stack size	4			
	Update horizon	3			
	Update period	4			
	Target update period	2500			
	Replay memory capacity	50000			
	Batch size	32			
	Optimizer	Adam			
	Learning rate	6.25×10^{-5}			
	Discount factor	0.99			
	Q-network output	Q-values			
	Replay memory scheme	Uniform			
C51[53]	Stack size	4			
	Update horizon	3			
	Update period	4			
	Target update period	2500			
	Replay memory capacity	50000			
	Batch size	32			
	Optimizer	Adam			
	Learning rate	6.25×10^{-5}			
	Discount factor	0.99			
	Q-network output	Distribution of Q-values			
	Replay memory scheme	Uniform			
Rainbow[50]	Stack size	4			
	Update horizon	3			
	Update period	4			
	Target update period	2500			
	Replay memory capacity	50000			
	Batch size	32			
	Optimizer	Adam			
	Learning rate	6.25×10^{-5}			
	Discount factor	0.99			
	Q-network output	Distribution of Q-values			
	Replay memory scheme	Prioritized			
CNN[57]	Number of hidden layer	4			
	Filters size	32,64,64,64			
	Kernel size	8×8,4×4,3×3,3×3			
	Activation function	Re U			
ResNet[32]	Number of hidden layer	4			
	Filters size	32,64,64,64			
	Kernel size	8×8,4×4,3×3,3×3			
	Activation function	ReL U			
	Residual Connection	layer2to3,layer3to4			

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

- ScenarioS2:Incoming traffic flows from North and South are 250 vehicles per hour while those of East and West are 500 vehicles per hour; the total number of simulated vehicles is 1500 vehicles per hour;
- ScenarioS3:Incoming traffic flows from North and South are 125 vehicles per hour while those of East and West are 500 vehicles per hour; the total number of simulated vehicles is 1250 vehicles per hour;
- ScenarioS4:Incoming traffic flows from North and South are 75 vehicles per hour while those of East and West are 500 vehicles per hour; the total number of simulated vehicles is 1150 vehicles per hour.

The main differences among these four scenarios come from different traffic flow ratios. For Scenario S1, the traffic is balanced and all road segments have equal traffic flow. We reduced the traffic flow of north and south road segments step-by-step for the other three scenarios. For Scenario S4, the traffic flow ratio of North, East, South and West is 1:10:1:10, which is heavily imbalanced between North-South and East-West. We tested the TSC methods on these scenarios with different traffic flow ratios.

The waiting time of vehicles at time t was evaluated based on Eq.(6). The duration of each simulation episode was set to 10 minutes of simulation time. The off-policy database was built by simulating the above scenarios under the four control schemes(Pre-timed, Greedy on number of vehicles, Greedy on waiting time, uniformly random control) and each transition (s_i,a_i,r_i,s_{i+1}) was stored in the database.

The deep RL algorithms for TSC were implemented in the Dopamine[51]framework using Tensor flow[59]and Python. The SUMO and Sumo-web3d simulators were embedded into OpenAI Gym [60]. All the experiments were run on a GPU machine with GeForce GTX 1080 Ti.

B. Experimental results

1) Waiting time : We investigated the required waiting times of vehicles at the intersection using different TSC methods. Table III shows the total waiting times of vehicles computed by various methods for the five scenarios. The four pre-timed control methods, considered as the baseline approaches, have a relatively large waiting time for all scenarios. We can see that Rainbow with BGR Res Net performed the best among all compared methods with 783s waiting time on average for the five scenarios. There was around 39% less waiting time on the average for the two greedy methods, 29% less for the max pressure control, and 40% less for other deep RL methods. For the Cologne scenario, our BGR Re s Net outperformed other TSC methods in general. The two Greedy methods performed better when compared to most of the deep learning methods other than ours.

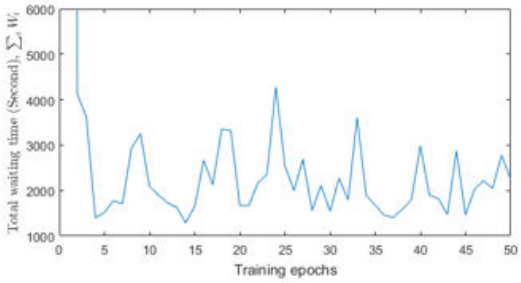
From the results of DQN and C51 using CNN and Res Net, we can see that using Res Net does not generally outperform those using CNN. The possible reasons are given as follows. Both the training algorithm and the Q-network contribute to the performance in deep RL. For the training algorithms com-pared in the experiments, the main difference is that Rainbow uses Prioritized Experience Replay [52] to sample valuable training data more frequently while DQN and C51 sample the training data uniformly. In other words, less valuable data are sampled relatively more frequently for training in non-Rainbow algorithm when compared to Rainbow. For the Q-network, Res Net is shown to ease the training with higher convergence rate and to negate the gradient vanishing problem

© 2026 IJIRCCE | Volume 14, Issue 8, August 2026 DOI: 10.15680/IJIRCCE.2026.1408026

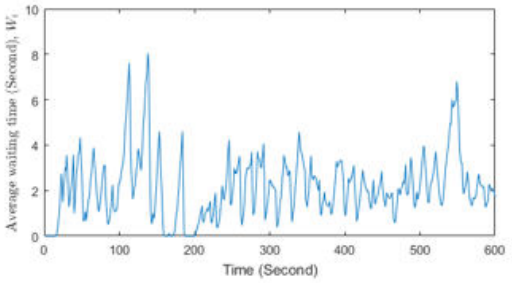
www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

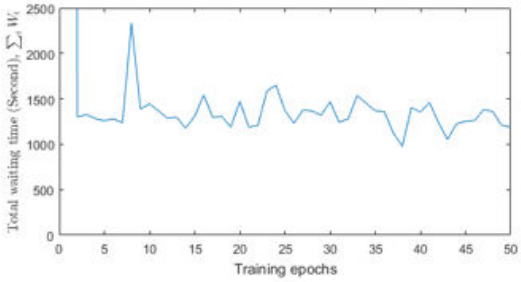
(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



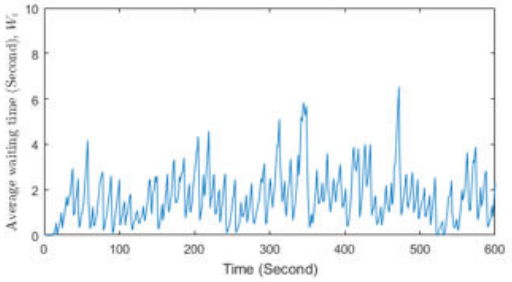
(a) Colognescenario



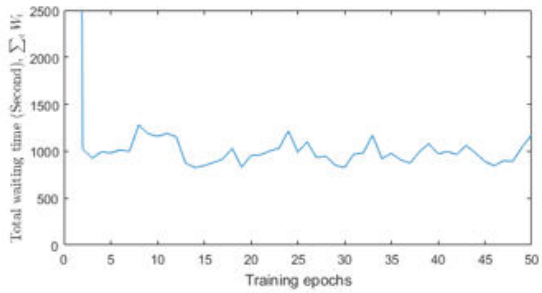
(a) Colognescenario



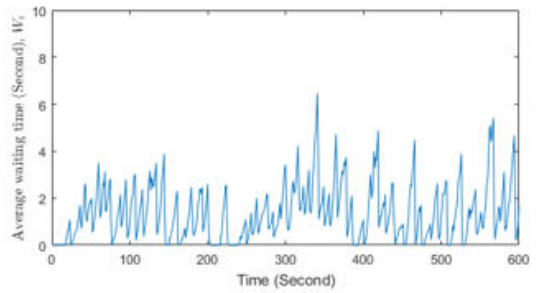
(b) ScenarioS1



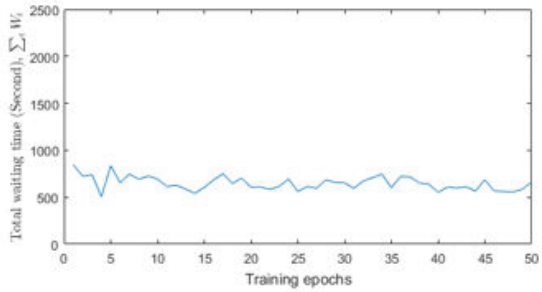
(b) ScenarioS1



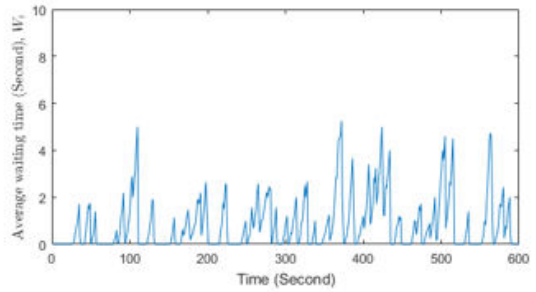
(c) ScenarioS2



(c) ScenarioS2



(d) ScenarioS3



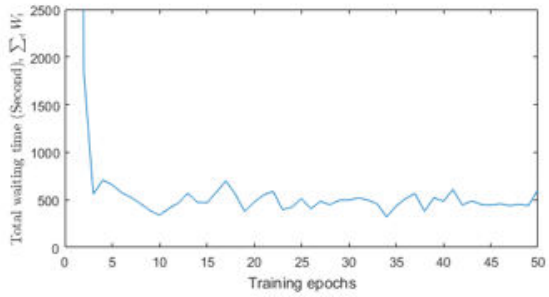
(d) ScenarioS3

© 2026 IJIRCCE | Volume 14, Issue 8, August 2026 DOI: 10.15680/IJIRCCE.2026.1408026

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

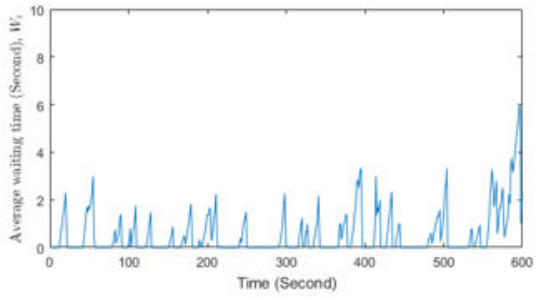
International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



(e) ScenarioS4

Fig.8.Total waiting time at different training epochs



(e) Scenario

Fig. 9. Average waiting time in each time step

VII. CONCLUSION

A transportation system without traffic congestion is desired in a smart city. In order to mitigate the traffic congestion and the average waiting time at the intersection, we proposed an end-to-end off-policy deep RL based TSC system that controls the traffic signal by observing images from road surveillance cameras. The system employed BGR Res Net as the Q-network and performs (near-)optimal traffic signaling based on dynamic traffic situation observed in the images. We conducted experiments on different intersection scenarios and various TSC methods. The experimental results show that the end-to-end off policy deep RL can adapt to the dynamic traffic based on the images and outperformed other TSC methods.

The deep RL agent was trained and tested in a simulator in this work. It is impractical to train the agent in the real world directly without pre-training in the simulator. We shall investigate in the future to transfer the trained agent to real world intersection using transfer learning.

REFERENCES

[1] K. F. Chu, A. Y. S. Lam, and V. O. K. Li, "Dynamic lane reversal routing and scheduling for connected and autonomous vehicles: Formulation and distributed algorithm," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 6, pp. 2557–2570, 2020.
[2] W. Lee, S. Tseng, J. Shieh, and H. Chen, "Discovering traffic bottlenecks in an urban network by spatiotemporal data mining on location-based services," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 4, pp. 1047–1056, Dec. 2011.
[3] R. S. Sutton, A. G. Barto et al., *Introduction to Reinforcement Learning*. MIT Press, Cambridge, 1998, vol. 135.
[4] W. B. Powell, *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. John Wiley & Sons, 2007, vol. 703.
[5] K.-L. A. Yau, J. Qadir, H. L. Khoo, M. H. Ling, and P. Komisarczuk, "A survey on reinforcement learning models and algorithms for traffic signal control," *ACM Computing Surveys (CSUR)*, vol. 50, no. 3, p. 34, 2017.
[6] X. Liang, X. Du, G. Wang, and Z. Han, "A deep reinforcement learning network for traffic light cycle control," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 2, pp. 1243–1253, Feb. 2019.
[7] K. F. Chu, E. R. Magsino, I. W. Ho, and C. K. Chau, "Index coding of point cloud-based road map data for autonomous driving," in *2017 IEEE 85th Vehicular Technology Conference (VTC Spring)*, June 2017.
[8] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, A. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.
[9] N. H. Gartner, "OPAC: A demand-responsive strategy for traffic signal control," *Transportation Research Record*, no. 906, pp. 75–81, 1983.
[10] D. Zhao, Y. Dai, and Z. Zhang, "Computational intelligence in urban traffic signal control: A survey," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 4, pp. 485–494, 2011.

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

[11] S. Chiu, "Adaptive traffic signal control using fuzzy logic," in *Proceedings of the Intelligent Vehicles Symposium*. IEEE, 1992, pp. 98–107.
[12] M. C. Choy, D. Srinivasan, and R. L. Cheu, "Cooperative, hybrid agent architecture for real-time traffic signal control," *IEEE Transactions on Systems, Man, and Cybernetics—Part A: Systems and Humans*, vol. 33, no. 5, pp. 597–607, 2003.
[13] D. Srinivasan, M. C. Choy, and R. L. Cheu, "Neural networks for real-time traffic signal control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no. 3, pp. 261–272, 2006.
[14] B. P. Gokulan and D. Srinivasan, "Distributed geometric fuzzy multi-agent urban traffic signal control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 3, pp. 714–727, 2010.
[15] H. K. Lo, "A novel traffic signal control formulation," *Transportation Research Part A: Policy and Practice*, vol. 33, no. 6, pp. 433–448, 1999.
[16] C. F. Daganzo, "The cell transmission model: A dynamic representation of highway traffic consistent with the hydrodynamic theory," *Transportation Research Part B: Methodological*, vol. 28, no. 4, pp. 269–287, 1994.
[17] P. Mirchandani and L. Head, "A real-time traffic signal control system: Architecture, algorithms, and analysis," *Transportation Research Part C: Emerging Technologies*, vol. 9, no. 6, pp. 415–432, 2001.
[18] P. Varaiya, "Max pressure control of a network of signalized intersections," *Transportation Research Part C: Emerging Technologies*, vol. 36, pp. 177–195, 2013.
[19] A. A. Zaidi, B. Kulcsár, and H. Wymeersch, "Back-pressure traffic signal control with fixed and adaptive routing for urban vehicular networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 8, pp. 2134–2143, 2016.
[20] G. Nilsson and G. Como, "A micro-simulation study of the generalized proportional allocation traffic signal control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 4, pp. 1705–1715, 2020.
[21] G. Bianchin and F. Pasqualetti, "Gramian-based optimization for the analysis and control of traffic networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 7, pp. 3013–3024, 2020.

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

INNO SPACE SJIF Scientific Journal Impact Factor

doi crossref

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

9940 572 462 6381 907 438 ijircce@gmail.com

www.ijircce.com

Scan to save the contact details