# Deep Q Learning with LSTM for Traffic Light Control

Chung-Jae Choe, Seungho Baek, Bongyoung Woon, and Seung-Hyun Kong*, *Senior Member, IEEE*

***Abstract*—Most Conventional traffic light control (TLC) techniques do not provide enough efficiency to control dynamic traffic situations in real-time. Recently, DQN (Deep Q Network) algorithm is considered for TLC at the intersection because of its optimization technique for complex problems, where key features of the intersection traffic, such as vehicle positions and velocities, are obtained from the intersection by the camera installed at well above the ground. However, the general DQN-based TLC algorithms have failed to utilize the fact that vehicle trajectories are continuous, which can be very useful in sensing real-time traffic. To utilize the continuous vehicle motion for TLC improvement, we propose DRQN-TLC (Deep Recurrent Q Network for TLC) algorithm that is based on LSTM (Long-Short Term Memory) with DQN. The superior performance of the proposed algorithm is demonstrated with the simulation; the proposed algorithm reduces the average traveling time by 23% and the overall vehicle waiting time by 10% when compared with the general DQN-based TLC algorithm.**

## I. INTRODUCTION

Traffic congestion causes social costs such as time spent on the road, unnecessary fuel consumption, environmental pollution. As the traffic congestion cost increases worldwide [1], there is a strong demand for efficient traffic congestion control. Traffic light control (TLC) technique is widely used as one of the various techniques to reduce the traffic congestion. Studies [2]–[6] introduced in the literature have calculated the cycle length of the traffic signal phase in advance based on the accumulated traffic data. This method may not effectively cope with dynamic traffic flows in real-time, making TLC inefficient and leading to greater traffic congestion in some cases. To resolve this problem, real-time TLC techniques are studied in order to operate robustly in dynamic traffic flows. Deep reinforcement learning technique, such as DQN (Deep Q Network) [7], has gained an increasing attention as an optimization technique for complex problems such as TLC. In the TLC with reinforcement learning, the traffic light acts as an agent and interacts with the intersection environment.

In real intersection environments, various sensors and appropriate sensing algorithms are required for TLC agent to acquire precise state information in real-time. However, in the process of sensing, there is a probability that the state observation is incomplete or incorrect due to the error of the sensor and the sensing algorithm. A recent study in TLC [8]–[10] based on the CNN (Convolutional Neural Network)-DQN algorithm assumes perfect sensing and extracts the main features of the state without any error. However, this assumption may not be applied to practice, because sensing can never be perfect due to the errors. Therefore, it is necessary to cope with the imperfection of the observed state caused by the sensing error, so that the agent can select more reliable and efficient actions. In addition, since the input and output are handled independently in general DNN (Deep Neural Network), the TLC techniques [11], [12] can have degraded performance because of the states with the causal relationship over a long time in the intersection.

Although CNN and general DNN have limited applicability to the TLC, we need a neural network for state-action function approximation, because there are countless states in dynamic intersection environments. In this paper, we use a recurrent network as a function approximator to recognize the causal relationship between the observed states in the intersection, since the recurrent network can process sequential information. As shown in Fig. 1, a vehicle trajectory in the intersection is characterized by a certain order and persistence that can be predicted in general. Therefore, when the recurrent network is used to process the state, it can understand the continuous vehicle motion with the observed state at every time-steps. In addition, the recurrent network can sufficiently compensate for the imperfection of the state due to the sensing noise and errors inevitably occurring in real-world environments.

In this paper, we propose DRQN-TLC (Deep Recurrent Q Network for Traffic Light Control), which exploits LSTM [13], a recurrent network, and DQN. The proposed DRQN-TLC algorithm can analyze the sequential change and find the vehicle trajectory and motion in the observed states of the intersection. In this paper, we assume a single camera detector that can be used as a traffic monitoring camera in practice to observe the state of the intersection. The state, action, and reward are defined as the key elements of reinforcement learning. To define the state, we use three kinds of information to express key features of the intersection environment: <P, V, S>, where P, V, and S represent the vehicle presence, vehicle velocity, and current traffic signal phase state, respectively. Action is defined as <Keep, Change>, which is to express whether to 'keep' the current traffic signal phase or 'change' to the next traffic signal phase. Reward is defined as the sum of the presence values (0 or 1) of all vehicles within and around the intersection. As an outcome of the reinforcement learning, the agent learns the optimal action policy that minimizes the total travel time and waiting time of the vehicles.

This paper is organized as follows. In Section II, we provide a literature review of previous studies in TLC. In Section III, we define the proposed algorithm model for deep reinforcement learning. Section IV introduces the proposed DRQN-TLC. The proposed algorithm is evaluated in Section

* Corresponding author.

The Authors are with the CCS Graduate School of Green Transportation, Korea Advanced Institute of Science and Technology, Daejeon 305-701, South Korea (e-mail: cjchoe12@kaist.ac.kr, bsh0749@kaist.ac.kr, wbyoung@kaist.ac.kr, skong@kaist.ac.kr).

This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(2018020032)

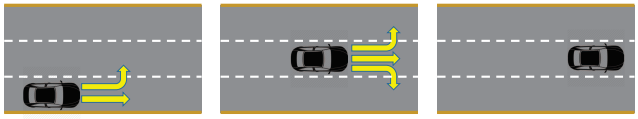


Fig. 1: Trajectory of a vehicle.

V with simulations. And the conclusion is drawn in Section VI with a discussion of future works.

## II. RELATED WORKS

TLC has been a field of research since 1980s. The initial study was to find the optimum cycle length of traffic lights to reduce traffic congestion based on the traffic data observed over a period of time, which are traffic volumes of roads, average vehicular speeds of roads, and etc. This approach can be categorized as Adaptive Traffic Control Systems (ATCS). More than 20 technologies, for example, SCOOT [2], SCATS [3], OPAC [4], RHODES [5], ACS-Lite [6], have been developed by 2010 [14]. ATCS analyzes collected traffic data such as vehicular density for several minutes and set the cycle length of the traffic light. Since the traffic light is not controlled simultaneously, there is a time lag between the data collection and the TLC operation.

In order to resolve this problem, real-time TLC has emerged as a major topic in the field. It has focused on controlling the traffic light in real time by setting each intersection as a decentralized agent. SURTRAC [15] is a decentralized schedule-driven method using forward dynamic programming. To ensure real-time performance, large clusters are formed to reduce the state space. Al Islam and Hajbabaie [16] proposed a TLC using Mixed Integer Linear Programming (MILP), which assumes a specific situation such as no yellow/red clearance, no phase min/max requirement and etc.

With the introduction of deep learning model for reinforcement learning in 2015 [17], DQN has begun to be used as optimization method in complex environments. In TLC, which deals with complex traffic environments, reinforcement learning can effectively reduce traffic congestion. Compared with the macroscopic multi-agent reinforcement learning studies [18]–[20], which control the multiple intersections, microscopic studies have been conducted actively to view the intersection as a decentralized agent, which makes TLC easy in real-time.

Wade [8] applied the DQN by limiting the state of the intersection environment to the discrete position and the corresponding speed. However, the action that the agent can choose is simplified to four traffic signal phases, and it can be changed randomly. It may not be applicable in the real environments because the actual traffic signal phases change in a certain order. Mousavi [9] used deep policy-gradient and value-function based reinforcement learning, where image snapshot is used as a state. This approach is only applicable to simulations with snapshots, so it is difficult to apply in real-world intersection.

Du et al. [10] studied TLC using 3DQN (Double, Dueling, and Deep Q network) algorithms. However, the MDP(Markov Decision Process) in [10] requires nine actions even in a simple lane intersection environment, which may lead multiple actions for a complex intersection. There is a potential disadvantage that it does not converge to the optimal action policy in reinforcement learning, if there are too many actions. In addition, the up and down of the traffic signal phase duration is set in units of 5 seconds, and the action is performed once every 5 seconds to avoid a sudden change, which is inefficient in real-time.

In this paper, we set the state, action, and reward that enable convergence to the optimal action policy even in a more complex environments than environments considered in the previous studies. Instead of exploiting CNN-based approach, we propose a new technique direction by applying LSTM so that the algorithm can understand the certain vehicle motion in the state at each time-step. In section IV, we will show that it allows faster convergence to the optimal action policy and better learning performance than general DQN-based TLC algorithm.

## III. ALGORITHM MODEL

In this section, we define the state, action, and reward used in the proposed DRQN-TLC algorithm. The key parameters for the reinforcement learning are described in detail.

### A. State

Previous studies for TLC using DQN algorithm have obtained information of intersection using image snapshot from simulator; An image snapshot is used to determine the vehicle's occupancy [8]–[10], or the position and velocity of vehicles are achieved through a grid structure that divides each traffic lane into cells of a pre-defined size. However, image snapshots cannot be applied in real intersection environments. Since various features such as position and velocity of each vehicle are necessary to describe the intersection traffic situation, various sensing methods are required to obtain such information in real intersection environments. Typically, in order to detect traffic in a real intersection environment from the viewpoint of a traffic light, a camera and various sensors can be fused for robust detection [21]. In addition, the key features of the intersection environment can be extracted by the sensing method using the latest machine learning detection technique for the camera image [22].

In the real-world environments, a sensing system extracts key information of intersection by fusing with traffic camera or other sensors. The traffic camera at a height of the traffic light might observe objects within a certain range. In the simulation, we use traffic camera to derive vehicle presence and velocity information at the intersection; sensing range is 30m from the center of the intersection. There are 60 (5x3x4) detecting areas, the five areas in each of three lanes in North, South, East, and West directions.

The size of each detecting area for the traffic camera processing is equal to the maximum pre-defined vehicle length

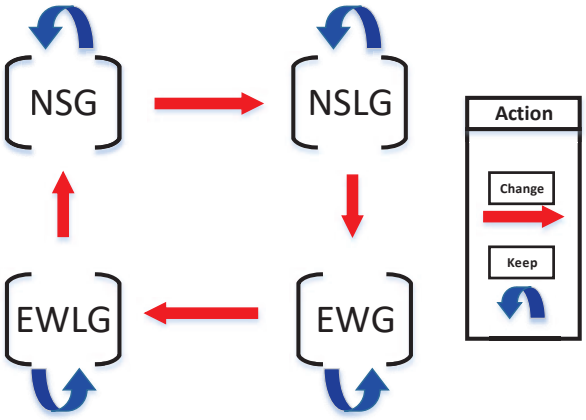



Fig. 2: Sequence of traffic signal phase change via the actions.

(5m) and the interval between detecting areas is set to the predefined maximum gap (2m) with the preceding vehicle. The presence of vehicles on each detecting area is denoted as <0 or 1>, and the vehicle velocity is expressed as a normalized value, <velocity of the current vehicle / maximum speed limit in the lane>. Also, it is necessary to perceive the current traffic signal phase among the 8 possible traffic signal phases because of our action definition, keep or change the signal phase. As a conclusion, state is a vector of <P (vehicle presence), V (normalized velocity of the vehicle), S (current traffic signal phase)>.

### B. Action

In reinforcement learning, the agent selects an optimal action according to the observed state. In the proposed algorithm, <keep, change> is defined as action. For the safety of the drivers, the traffic signal phase changes according to a predefined sequence used in the real-world TLC system. Agent determines whether to keep the current traffic signal phase or to change to the next phase based on the current state.

We set minimum keep duration as $N_{keep}$ seconds ($N_{keep}$ = 3 sec in the simulation) for each traffic signal phase. That is, it maintains the current traffic signal phase at least Nkeep seconds even the change-action is selected. It is considered for fairness; there must be no vehicle waiting too much in the intersection. Furthermore, we set maximum keep duration as $N_{straight}$ ($N_{straight}$ = 60 sec in the simulation) for straight traffic signal phase and $N_{left}$ ($N_{left}$ = 20 sec in the simulation) for left-turn traffic signal phase in order to prevent the infinite increase of the duration of any traffic signal phase. For the driver's safety, there must be yellow/red clearance (3 sec in the simulation) between green traffic signal phases. It provides enough time to vehicles approaching to the intersection for deceleration. The sequence of traffic signal phase is shown in Fig. 2

### C. Reward

It is one of the most important factors in the reinforcement learning to select the reward with appropriate range. It can be set as plus-reward and minus-reward according to agent's action result. The agent learns the optimal action policy maximizing cumulative rewards from the feedback of the evaluation on the action conducted by the agent at every time-step.

Previous studies for TLC using DQN algorithm have also devised appropriate reward definitions. For example, cumulative delay of all vehicles during the simulation [8] and staying time of all vehicles during green traffic signal phase [11] can be considered as reward. In this paper, we set a reward as a minus reward, sum of the vehicle's presence value (0 or 1) in the 60 detecting areas obtained by the traffic camera at each time-step as shown in equation (1). When the total sum of the presence values is reduced, the traveling time and the waiting time of the vehicles decrease. Reward is always a negative value.

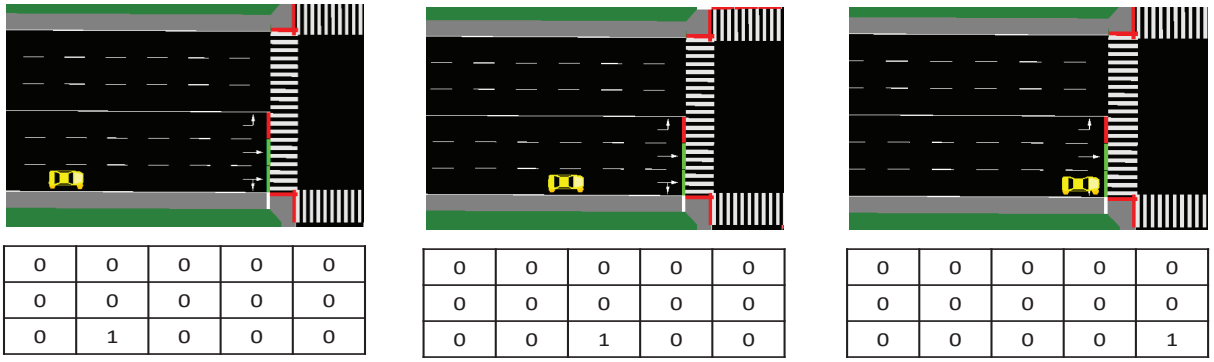
$$r_t = -\sum_{i=1}^{60} p_{i,t} \tag{1}$$

## IV. DEEP RECURRENT Q NETWORK FOR TRAFFIC LIGHT CONTROL

In the studies using DQN algorithm for TLC, CNN is used for state approximation [8]–[10]. The authors assume perfect sensing (i.e., no sensor error or sensing algorithm error) and extract the main features of the observed state in the intersection using CNN. This assumption is impractical as there often occur various sensor errors and algorithmic errors in real environments. To mitigate the effect of these errors, it is necessary to compensate the state error caused by the sensing, so that the agent can recognize the state correctly and select appropriate actions.

The fact that the state feature shown in the Fig. 3 (e.g. vehicle presence) and other state features (e.g. velocity of the vehicle and lane occupied by the vehicle) have appeared in the previous steps is because of a continuous vehicle motion along a trajectory and those features will also appear in the next state. In order to exploit the continuous flow (i.e., vehicle motions) of traffic, it is found useful in this paper to process the state using a recurrent network that can recognize linked information over time. In addition, there is a risk of obtaining incorrect state information due to the errors in the sensing in practice. However, learning through the recurrent network can cope with the risk and be robust to those errors for TLC, and we utilize recurrent network in the proposed algorithm.

As shown in Fig. 4, the network of DRQN-TLC is a combination of MLP (Multi-Layer Perceptron) structure with three hidden layers and LSTM. MLP consists of three hidden layers with 512, 256, and 128 output dimensions respectively, and ReLU (Rectified Linear Unit) is used for an activation function. The output of the MLP is the input to the LSTM. The state S = <P, V, L> observed from the intersection is the input




(a) Trajectory and presence value at t-2. (b) Trajectory and presence value at t-1. (c) Trajectory and presence value at t.

Fig. 3: Trajectory of the vehicle at the intersection with presence value.

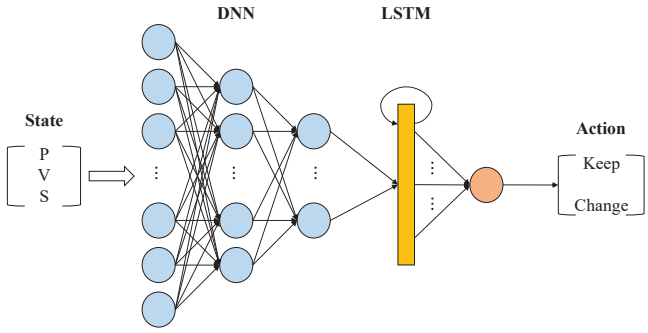



Fig. 4: The architecture of DRQN-TLC.

of the proposed algorithm, and the output vector is the number of actions to approximate the Q-function. In addition, in order to solve the problem of correlation between samples and non-stationary target, which is a fundamental problem occurring in DQN, we exploit the learning method of experience reply and target network [17].

## V. EVALUATION

This section describes the simulation environment for the proposed DRQN-TLC algorithm performance verification. We show the performance of the proposed algorithm by improvement of the occupancy rate of vehicles, waiting time of vehicles, and average running time of all vehicles.

### A. Simulation enviroment

To implement the proposed algorithm, we utilize the Keras and Tensorflow libraries to build a deep learning network, and the OpenAI-Gym library to build reinforcement learning environments. Simulation of Urban Mobility (SUMO), which is an open traffic simulation software that simulates traffic conditions at an intersection, is used to verify the performance of the proposed algorithm. SUMO provides several convenient functions for traffic signal control through the TraCI (Traffic Control Interface) module. Table. I summarizes the intersection environment and vehicle parameters considered in this simulation.

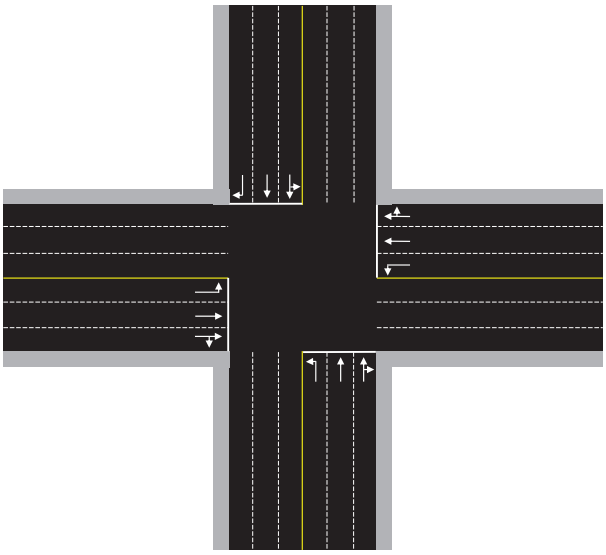


Fig. 5: The intersection area [320m x 160m] for the simulation.

*1) Intersection environment:* The intersection area considered in this paper is of size 320m x 160m as shown in Fig. 5 The length of East-West road is 160m and North-South road is 80m. Each road from East/West/North/South comprises three lanes and each lane allows left turn only, straight ahead only, and jointly serving right turn and straight, respectively. The maximum allowed speed of all lanes is limited to 13.89 m/s (i.e. about 50 km/h).

*2) Traffic flow generation:* Generation of vehicles entering the intersection is determined according to the random process as follows; first, a uniform random number in the range of 0 to 1 is generated at every 1 second. When the random number is 0.5 or more, a vehicle is generated with a probability of 0.1 for each North-South lane and 0.2 for each of the East-West lane. In contrast, when it is less than 0.5, the vehicle is created with a probability of 0.2 for each North-South lane and a probability of 0.1 for East-West lane. The departure and destination (i.e.

arrival lane) of the vehicle are the default random value provided by SUMO. Depending on the probability, the flow of the vehicle is renewed for every episode. Traffic flow of each episode are different. Each episode corresponds to 1000 time-steps. In this study, we set up to follow the Krauss-car following model [23], which avoids collisions between vehicles and allows safe speeds.

*3) Considering real-world vehicular characteristics:* Unlike most vehicle models of the same size and characteristics used for the previous studies, vehicles of diffrent size travel simultaneously with their own characteristics on the actual road. A driver, who cares about safety, keeps a wide distance from the front car and runs at the appropriate speed or low speed. Drivers preferring fast move keep a short distance, and may travel at a maximum speed. In this paper, similar to the reality, uniformly random vehicle characteristics are considered within range: length of vehicles 3∼4.5m, minimum gap between vehicles 1∼2m, maximum speed (40∼50km/h), acceleration (0.5∼1m/s$^2$), and deceleration (3.5∼5m/s$^2$).

TABLE I: Key parameters of the simulation.

| Parameter | Value(range) |
|---|---|
| Road Length | 160m(East-West), 80m(North-South) |
| Available Route | Straight, Right-turn, Left-turn |
| Maximum Speed at Road | 50Km/h |
| Length of Vehicles | 3∼4.5m |
| Minimum Gap between Vehicles | 1∼2m |
| Max Speed | 40∼50km/h |
| Acceleration | 0.5∼1m/s$^2$ |
| Deceleration | 3.5∼5m/s$^2$ |
| Traffic Flow Generation | 0.1/sec or 0.2/sec |

## B. Simulation result

In order to verify the performance of the proposed DRQN-TLC algorithm, we analyze the results of the simulation with three indicators: cumulative reward, average traveling time of each vehicle, and overall waiting time of vehicles. We compared with the general DQN-based TLC algorithm that does not combine with LTSM to verify the performance of the proposed algorithm. All parameters used in the simulation are the same in both cases.

*1) Cumulative reward:* The reward is defined as the total sum of the vehicles in the current intersection environment, that is, the sum of the presene values of all vehicles within the camera observation range at each time-step. Reward uses a cumulative reward that is accumulated until the end of an episode. Reward is always negative and learning of the algorithm proceeds with the goal of converging to zero as close as possible. The change in reward obtained by simulations can be confirmed by Fig. 6. The average of 50 episodes is represented by one epoch, which represents the horizontal-axis of Fig. 6. The red dotted line shows the change in reward of the proposed algorithm in this paper, and the blue line shows the change in reward of the general DQN-based TLC algorithm. The red dotted line changes steadily to the starting point of about 12 epoch, but the blue line still shows unstable change.

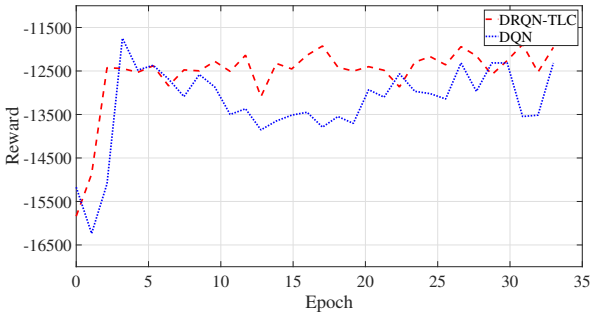



Fig. 6: Cumulative reward during all the training epochs.

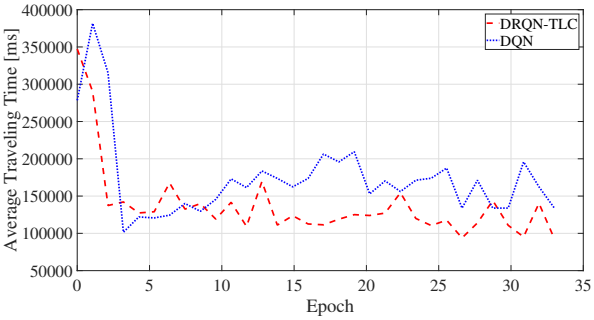



Fig. 7: The average traveling time during all the training epochs.

In addition, we can see that the red dotted line converges to a higher reward than the blue line in overall change trend. The reward of the proposed algorithm based on the last epoch is about -12000 and the reward of the typical DQN-based algorithm is about -12400. This indicates that the proposed algorithm optimizes TLC more quickly and stably than the general DQN-based TLC algorithm.

*2) Average traveling time:* The performance of the proposed algorithm is analyzed through, the average time for each vehicle to arrive at its destination, which is found in Fig. 7. The red dotted line in Fig. 7 shows the average traveling time change of the vehicles as a result of applying the proposed algorithm. The blue line shows the average traveling time of the vehicles using the general DQN-based TLC algorithm. As shown in Fig. 7, It is confirmed that the proposed algorithm outperforms the general DQN-based TLC algorithm. Here, we can see that the average traveling time of vehicles converges quickly and stably. Also, the general DQN-based TLC algorithm shows that the average traveling time of the vehicles reaches the steady state at about 180,000 ms. However, the average traveling time of the proposed algorithm reaches about 140,000 ms at steady state, which is the result of 23% reduction. This result shows that our algorithm can effectively reduce the average traveling time of vehicles compared to the general DQN-based TLC algorithm.

*3) Overall waiting time:* We analyze the performance of the proposed algorithm using the time that the vehicles wait to pass the intersection. Fig. 8 shows the change in the sum

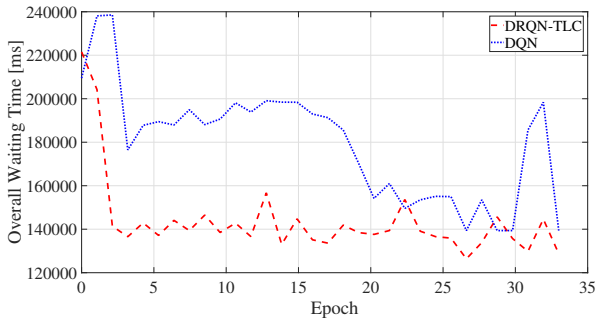



Fig. 8: The overall waiting time during all the training epochs.

of the waiting times of all vehicles in each simulation. The red dotted line shows overall waiting time of the proposed algorithm. The blue line shows overall waiting time of the general DQN-based TLC algorithm. Similar to the average traveling time, the proposed algorithm outperforms the general DQN-based TLC algorithm in the viewpoint of the overall waiting time of all vehicles. Based on the last epoch in Fig. 8, overall waiting time of the vehicles is about 125,000 ms for the proposed algorithm, and that for the general DQN-based TLC algorithm is about 140,000 ms. This demonstrates that when the proposed algorithm is applied, the overall waiting time of the vehicles is reduced by 10% compared with the recent techniques.

## VI. CONCLUSIONS

In this paper, DRQN-TLC (Deep Recurrent Q Network for Traffic Light Control) algorithm has been proposed to solve the traffic congestion problem in the intersection. The superior performance of the proposed algorithm has been demonstrated with the simulation. As a result, the proposed algorithm reduces the overall waiting time of the incoming vehicles to the intersection by 23% and reduces the average traveling time of vehicles running through the intersection by 10% when compared with general DQN-based TLC algorithm. For the future work, we plan to apply empirical sensing errors or imperfect sensing results in order to develop robust TLC algorithm useful in practice.

## REFERENCES

[1] L. Mussone, S. Grant-Muller, and J. Laird, "Sensitivity analysis of traffic congestion costs in a network under a charging policy," *Case Studies on Transport Policy*, vol. 3, no. 1, pp. 44–54, 2015.

[2] P. Hunt, D. Robertson, R. Bretherton, and M. C. Royle, "The scoot on-line traffic signal optimisation technique," *Traffic Engineering & Control*, vol. 23, no. 4, 1982.

[3] P. Lowrie, "The sydney cooridinated adaptive traffic (scat) system-principles, methodology, algorithm," in *Proc. of International Conference on Road Traffic Signaling*, pp. 67–70, IEE, 1982.

[4] N. H. Gartner, *OPAC: A demand-responsive strategy for traffic signal control*. No. 906, 1983.

[5] S. Sen and K. L. Head, "Controlled optimization of phases at an intersection," *Transportation science*, vol. 31, no. 1, pp. 5–17, 1997.

[6] F. Luyanda, D. Gettman, L. Head, S. Shelby, D. Bullock, and P. Mirchandani, "Acs-lite algorithmic architecture: applying adaptive control system technology to closed-loop traffic signal control systems," *Transportation Research Record: Journal of the Transportation Research Board*, no. 1856, pp. 175–184, 2003.

[7] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[8] W. Genders and S. Razavi, "Using a deep reinforcement learning agent for traffic signal control," *arXiv preprint arXiv:1611.01142*, 2016.

[9] S. S. Mousavi, M. Schukat, and E. Howley, "Traffic light control using deep policy-gradient and value-function-based reinforcement learning," *IET Intelligent Transport Systems*, vol. 11, no. 7, pp. 417–423, 2017.

[10] X. Liang, X. Du, G. Wang, and Z. Han, "Deep reinforcement learning for traffic light control in vehicular networks," *arXiv preprint arXiv:1803.11115*, 2018.

[11] J. Gao, Y. Shen, J. Liu, M. Ito, and N. Shiratori, "Adaptive traffic signal control: Deep reinforcement learning algorithm with experience replay and target network," *arXiv preprint arXiv:1705.02755*, 2017.

[12] L. Li, Y. Lv, and F.-Y. Wang, "Traffic signal timing via deep reinforcement learning," *IEEE/CAA Journal of Automatica Sinica*, vol. 3, no. 3, pp. 247–254, 2016.

[13] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[14] A. Stevanovic, *Adaptive traffic control systems: domestic and foreign state of practice*. No. Project 20-5 (Topic 40-03), 2010.

[15] S. F. Smith, G. J. Barlow, X.-F. Xie, and Z. B. Rubinstein, "Surtrac: Scalable urban traffic control," 2013.

[16] S. B. Al Islam and A. Hajbabaie, "Distributed coordinated signal timing optimization in connected transportation networks," *Transportation Research Part C: Emerging Technologies*, vol. 80, pp. 272–285, 2017.

[17] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[18] L. Prashanth and S. Bhatnagar, "Reinforcement learning with average cost for adaptive control of traffic lights at intersections," in *Intelligent Transportation Systems (ITSC), 2011 14th International IEEE Conference on*, pp. 1640–1645, IEEE, 2011.

[19] K. Prabuchandran, H. K. AN, and S. Bhatnagar, "Multi-agent reinforcement learning for traffic signal control," in *Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on*, pp. 2529–2534, IEEE, 2014.

[20] Y. Liu, L. Liu, and W.-P. Chen, "Intelligent traffic light control using distributed multi-agent q learning," *arXiv preprint arXiv:1711.10941*, 2017.

[21] S. Sivaraman and M. M. Trivedi, "Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 4, pp. 1773–1795, 2013.

[22] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.

[23] S. Krauß, P. Wagner, and C. Gawron, "Metastable states in a microscopic model of traffic flow," *Physical Review E*, vol. 55, no. 5, p. 5597, 1997.