Australasian Transport Research Forum 2021 Proceedings
8-10 December, Brisbane, Australia
Publication website: http://www.atrf.info

A Comparison Study of Different Data Resolutions for Deep Reinforcement Learning Based Adaptive Traffic Signal Control System

Mobin Yazdani, Majid Sarvi, Saeed Asadi Bagloee[1]

[1]Transport technologies group, Faculty of Engineering and Information Technology, University of Melbourne, Victoria

Email for correspondence: myazdani@student.unimelb.edu.au

Abstract

By recent advances in technology and Artificial Intelligence (AI), traffic signal control systems are preferably designed to have intelligence rather than rule-based structure. Deep Reinforcement Learning (RL) as a solution for sequential decision-making problems has been extensively used for adaptive traffic signal control (ATSC) systems. The deep RL-based ATSC systems have shown promising results versus current actuated (rule-based) ATSC systems. The conducted studies have employed different data resolutions either collected in vehicular network (e.g., location and speed of individual vehicles) or from camera devices (e.g., queue length, density, average speed) for proposed models. However, the impact of different data resolutions on deep RL-based ATSC systems training performance has not been studied yet. In this study, we compare the three different data resolutions in terms of computation time, training stability and results for variety of performance measurements. The Double Deep Q-Network (DDQN) algorithm is utilized as our intelligent agent. To test and evaluate the different data resolutions, a real isolated intersection is modelled in a simulation environment with real traffic volume demand. The experimental results have shown that vehicular network high resolution data can only contribute to a slight improvement versus camera data in terms of reduction in travel time, queue length etc. at the expense of more computation time in training models. Also, the camera data is more accessible compared to vehicular network data which needs sensors on plenty of vehicles in network. Hence, we recommend using camera data which provides aggregated but adequate data for deep RL-based ATSC models.

1.Introduction

Global population is increasing and Australian cities are more urbanized and congested (Audit, 2019). Traffic congestion caueses significant costs in fuel consumption, emissions and traffic delays. One effective solution to address this issue is to design an efficient traffic signal control system. The system can switch and execute the traffic lights for flexible durations so to reduce the delay of transportation modes. Hence, traffic signal control systems play a pivotal rule to reduce the traffic congestion costs.

With the advent of technology, intelligent transport system (ITS) devices such as camera, bluetooth, radar etc have provided practitioners with high resolution data to improve transport systems (Emami et al., 2020). Furthermore, the advanced Machine Learning (ML) algorithms have emerged as a powerful tool to improve systems's performance by intelligent agents instead of classical methods or human intervention (LeCun et al., 2015, Mnih et al., 2015).

Hence, ML-based methods have attracted attentions for transport applications. For traffic signal systems, deep reinforcement learning (RL) is a solution to enable smart decision-making of traffic lights selection only based on data in place. In fact, the traffic data collected by ITS devices are fed to a deep neural network (DNN) which outputs traffic signal action values. The traffic signal action selection is then taken by RL algorithm. Once the action is executed, RL agent receives a scalar value (i.e., reward) that indicates how good was the selected action. As ultimate goal in transportion is to reduce traffic congestion, the reward is defined to minimize waiting time. By continuous interaction of deep RL agent with traffic environment, the agent learns the optimal policy in action selection which leads to less traffic congestion.

The conducted studies in literature have shown superior performance of deep RL-based ATSC over rule-based traffic signal control methods. The different variety of traffic data have been used for proposed deep RL-based models. The traffic data are either assumed to be collected in vehicular network (Wu et al., 2020) or from camera devices installed at intersections (Jeon et al., 2018). The vehicular network data provides accurate information of individual vehicles such as location and speed within the road while camera data can provide aggregated data such as queue length, density and average speed of each incoming lane to the intersection. In literature, vehicular network data has been extensively used as it brings information-dense data samples for deep RL agent training. For real world implementation, however, it is critical to examine if the traffic data is available for proposed model. Also, using the higher resolution data should be justified considering the installation costs, amount of improvements and computation costs. In this study, we compare and investigate the impact of different data resolutions on deep RL-based ATSC performance. To represent the real world situation, we test and evaluate the deep RL-based ATSC models in a microsimulation model of an intersection in Melbourne, Australia. The experimental results have shown that vehicular network data can only contribute to slight improvement versus camera data at the expense of more training costs. Also, according to the easier accessibility of practitioners to camera data versus vehicular network data, camera data is more preferable. Hence, we recommend using camera data such as queue length and density for deep RL-based ATSC model.

2. Related works

Early studies in literature have applied RL on their ATSC problem (Abdoos et al., 2011, Prashanth and Bhatnagar, 2010, El-Tantawy and Abdulhai, 2010, Balaji et al., 2010, Arel et al., 2010, Grégoire et al., 2007, Wiering, Gao et al., 2017). The RL can learn from data collected at intersection to train a smart traffic signal. However, the data used in these studies were coarse and abstract because of RL agent's inability to capture problems with high dimensions. By proposing Deep Q-Network (DQN) algorithm (Mnih et al., 2015) RL agents were enabled to learn from high-dimensional data. Since then, the research conducted in ATSC literature aimed at using data with more information such as queue length of the waiting vehicles, density or speed and location of individual vehicles as presented in Table1. Despite employing data with different range of resolutions, the extent of improvements has not been investigated. In this study we examine the effect of having different data resolutions in terms of computation costs, training stability and improvements for traffic performance measurements.

Table 1: Different data resolutions used in literature

Study	Data resolution	Number of intersections
(Genders and Razavi, 2016)	Speed and location of individual vehicles	1
(Van der Pol and Oliehoek, 2016)	Location of individual vehicles for state definition and speed of individual vehicles for reward	1,2,3,4
(Gao et al., 2017)	Speed and location of individual vehicles	1
(Mousavi et al., 2017)	Speed and location of individual vehicles	1
(Jeon et al., 2018)	The image of the intersection	1
(Wei et al., 2018)	Queue length, number of vehicles, updated waiting time of vehicles, vehicle's position	1
(Genders, 2019)	Queue length and density	1
(Genders and Razavi, 2019)	Queue length and density	1
(Liang et al., 2019)	Speed and location of individual vehicles	1
(Tan et al., 2019)	Queue length	24 (traffic grid)
(Wei et al., 2019)	Total number of vehicles and segment wise distribution of vehicles on each lane	4 (Arterial)
(Zheng et al., 2019)	Total number of vehicles on each lane	4 (Arterial)
(Wu et al., 2020)	Speed and location of individual vehicles, queue length and number of pedestrians	2, 6
(Wang et al., 2021)	Congestion level of the intersection	36

3. Deep Reinforcement Learning

Reinforcement learning (RL) is one of the machine learning paradigms that enables an agent to learn a specific task. The main components of RL are state (s), action (a), reward (r). In each time step t, the intelligent agent takes action a_t in state s_t, receives reward r_t and ends up in next state s_{t+1}. The r is a feedback signal for RL agent to learn the goal of the task. To explore the solution space, random actions are mostly taken at the beginning of the learning process. As RL agent is goal-oriented, unwanted actions are penalized with a reward. RL aims at maximizing the return $G_t \doteq \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$ which is the cumulative of discounted rewards. The future rewards are discounted with parameter $\gamma \in [0,1)$ so to consider the importance of immediate rewards over future rewards. The most popular RL algorithm is Q-learning algorithm that each state-action pair value (Q-value) is the expected return under policy π as $Q_\pi(s,a) \doteq \mathbb{E}_\pi[G_t | s_t = s, a_t = a]$. Based on Bellman optimality equation, the optimal Q-value under policy π is the one with best (i.e., maximum) value (Sutton and Barto, 2018). By continuous interaction of RL agent in the defined state, the RL can finally learn to take the optimal actions. To solve the problem with higher number of state-action pairs, the deep learning variant of Q-learning algorithm, Deep Q-Network, proposed by Mnih et al. (2015). In

this model, the deep neural network is used as a function approximator for Q-value estimations. The Q-function with optimal policy π^* is calculated as in equation1.

$$Q^{\pi^*}(s,a;\theta) = \mathbb{E}_{\pi}[r_{t+1} + \gamma \max_{a} Q(s_{t+1},a;\theta)\,|s_t = s, a_t = a] \qquad \text{equation1}$$

Where, θ is the deep neural network parameter.

4. Deep RL components

In this study, the deep RL-based model performance is investigated with three different data resolutions as follows:

- ATSC-QLAS: lane queue length (QL) and link average speed (AS)
- ATSC-QLDS: lane queue length (QL) and lane density (DS)
- ATSC-DTSE: location and speed data for each individual vehicle popular as discrete traffic state encoding (DTSE)

Our case study is a real four-leg-four-lane intersection at Rathdowne-Elgin junction in Melbourne, Australia. The simulation model has 4 zones {N,E,S,W} for origin and destination demands through which the vehicles are generated with dynamic assignment model. In this section the state, action and reward definition for models are explained.

4.1. state definition

4.1.1. ATSC-QLAS model

The state definition for this model is queue length of the incoming lanes, average speed of incoming link and the current traffic signal state. Hence, the state space is $S \in \mathbb{R}^{i*j} \times \mathbb{R}^{j} \times \mathbb{T}^{n}$. Where, i and j are the number of intersection lanes and links respectively and, n is the number of agent actions. It is noted that queued vehicles are vehicles with speed less than 5 km/hr.

4.1.2. ATSC-QLDS model

The state definition for this model is queue length and density of vehicles in incoming lanes and, the current traffic signal state. Hence, the state space is $S \in (\mathbb{R} \times \mathbb{R})^{i \times j} \times \mathbb{T}^{n}$.

4.1.3. ATSC-DTSE model

In discrete traffic state encoding state definition, the individual speed and vehicles data is collected from incoming lanes. Each lane is discretized to multiple cells that fits the vehicles data as illustrated in Figure 1. The vehicle's average length in simulation is 5 meters so the cell length is selected 6 meters. The state space for this model is $S \in (\mathbb{B} \times \mathbb{R})^{(i*j)*k} \times \mathbb{T}^{n}$. Where, k is the number of cells along the link.

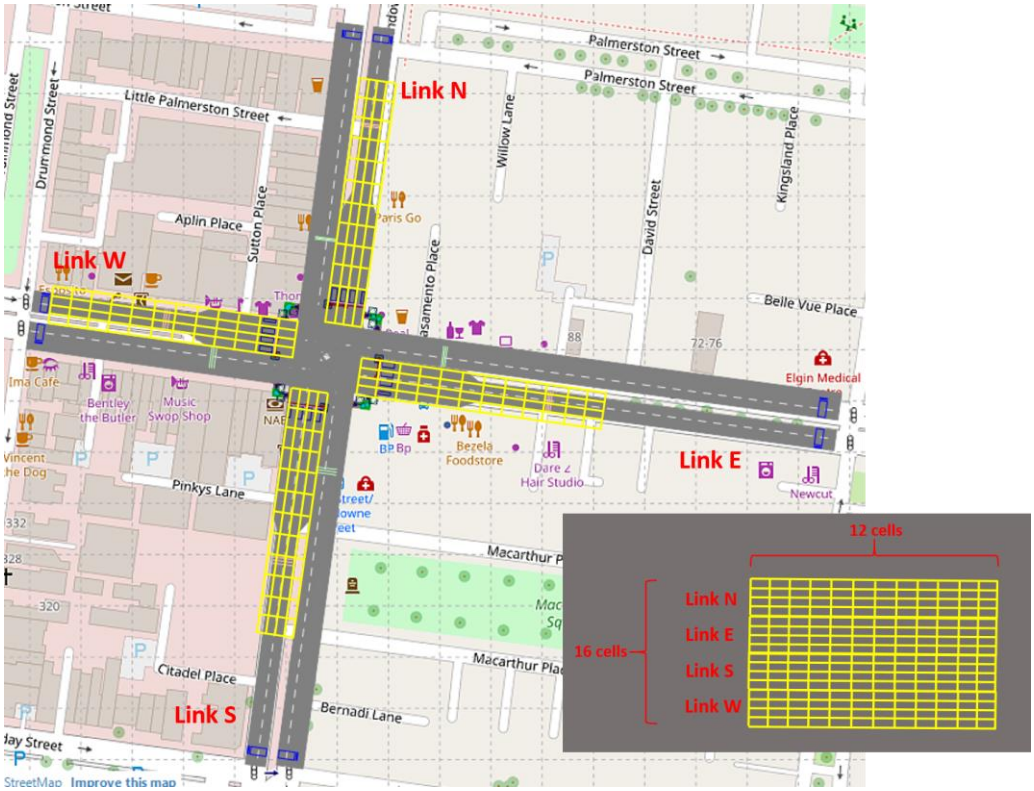

Figure 1: Discrete traffic state encoding for an intersection

4.2. action definition

The traffic light has two actions $a = \{NS, EW\}$. The RL agent either executes green light for North-South (NS stage) direction or East-West (EW stage) direction. Each action (i.e., stage) has two phases each with 5 seconds green time following the 3 seconds amber and 2 seconds red light as presented in Figure 2. The first phase includes through and left turn movement with permissive right turn movement. The next phase is a protected right turn movement, and it is only activated for transition to next action.

Figure 2: traffic signal phases of simulated intersection

4.3. Reward definition

4.3.1. ATSC-QLAS model

The reward definition must be complied with the resolution of available data. Hence, the reward function for this model is to minimize the queue length at the intersection. A vehicle is in queue if it has speed less than 5 km/hr. In each time step, the queue length data for each lane is collected as $r_t = -(q_{i,j})$. The reward for each action is the summation of rewards during the course of action execution.

4.3.2. ATSC-QLDS model

The reward function for this model is similar to ATSC-QLAS which is penalizing the queued vehicles.

4.3.3. ATSC-DTSE model

As vehicles speed and location data is available, reward function can be more accurate. The reward is to minimize the waiting time for each individual vehicle. In this study we assume that vehicles with speed less than 5 km/hr are in the queue and they are experiencing delay. To include the vehicles in range within [0,5) we define the reward function for each vehicle at time step t as $r_t = -(1 - 0.2v_t)$. Where vehicles with speed $v = 0\ km/hr$ have rewards -1 and vehicles with speed $v = 5\ km/hr$ have reward 0. The vehicles waiting time within this range are considered with a linear function.

5. Deep RL model

In this section the model specification and deep neural network architectures are explained.

5.1. Double Deep Q-Network algorithm (DDQN)

In this study the Double Deep Q-Network (DDQN) algorithm is used for training the agent. The algorithm leverages two networks, Primary network (θ^P) and Target network (θ^T) to update the Q-values. First, the Primary network get the states as input and outputs the Q-value of corresponding actions. As such, after each action execution, the experience memory tuple (s_t, a_t, r_t, s_{t+1}) is generated and is saved in a buffer memory with size $\mathcal{B}$. To slow down the learning fluctuations, Target network (θ^T) is used for target value estimation. The Target neural network is not trained. First, the (state, target) pairs are used for training the primary network and Target weights are then updated by soft update rate β as $\theta^T = \beta\theta^P + (1-\beta)\theta^T$. To address the correlation between consecutive memory tuples, the experience tuples are uniformly sampled with minibatch size $\mathcal{M}$ from a buffer memory β. The DDQN algorithm can address the maximization bias of the DQN algorithm as follows:

$$Q^{\pi^*}(s,a;\theta^P) = \mathbb{E}_\pi[r_{t+1} + \gamma Q\left(s_{t+1}, \underset{a_t}{\operatorname{argmax}}\, Q(s_{t+1}, a_t; \theta^P)\,; \theta^T\right) | s_t = s, a_t = a] \qquad \text{equation 2}$$

In this study, the action selection for DDQN agent is based on decaying ϵ-greedy strategy where enables agent to do the exploration-exploitation process. To clarify, the agent mostly takes random actions with probability ϵ_{init} in the beginning of training episodes. The ϵ is

decreased with decay ratio η until it reaches minimum epsilon ϵ_{min}. To learn the task, DDQN agent run is repeated for N number of episodes. In each episode (e) the model has a warm up period (W) before starting the process. The whole algorithm is presented in table 3. The DDQN ATSC model hyperparameters are as Table 2.

Table 2: DDQN ATSC model hyperparameters

Variable	Hyperparameters	Value
γ	Discounted factor	0.95
α	Learning rate	0.0001
β	Target update rate	0.001
$\mathcal{B}$	Buffer memory size	200
$\mathcal{M}$	Mini batch size	32
ϵ_{min}	Minimum epsilon	0.01
ϵ_{init}	Initial epsilon	0.9
η	Decay ratio	0.05

5.2. Deep neural network architecture

5.2.1. ATSC-QLAS model

This model has a deep neural network with 4 layers. The first layer is a vector with size 16 for queue length input and another vector with size 4 for average speed input. The input data is fed to 3 fully connected layers with 64, 64 and 32 neurons respectively. The ReLU activation function is also applied after each layer. The last layer is the RL actions which is 2 in this study.

5.2.2. ATSC-QLDS model

This model has a deep neural network with 3 layers. The first layer is a vector with size 16 for queue length input and another vector with size 16 for density input. The input data is fed to 2 fully connected layers with 128 and 64 neurons respectively. The ReLU activation function is also applied after each layer. The last layer is the RL actions.

5.2.3. ATSC-DTSE model

In this model, the state inputs are fed to the CNN with 5 layers. The first layer consists of two image-like input (speed and location) both with 16×12 dimensions. It is convolved with 16 filters each with 4×4 size and stride of 2. Similarly, the next layer is convolution but with 32 filters each with 2×2 size and stride of 1. Then the flattened images are fed to a fully connected layers first with 128 neurons and second with 64 neurons. It is noted that the ReLU activation function is applied after each layer. The last layer is the output of the deep CNN network with the number of RL actions. In this study, RL has 2 actions to select.
For all models, the RMSprop optimizer is used for deep neural networks (Tieleman and Hinton, 2012) to minimize the loss function mean squared error (MSE).

Algorithm 1 Double Deep Q-Network Pseudocode for Adaptive Traffic Signal Control

Initialize: Primary Network $Q_P(s,a)$ with random weights θ^P

Initialize: Target Network $Q_T(s,a)$ with random weights θ^T

Initialize: Experience replay buffer memory $\mathcal{B}$ for DDQN agent

for $e = 1,2,\dots,N$ **do**
  for $step = 1,2,\dots,W$ **do**
    Initialize traffic state S_t matrices
    Start Exploration-Exploitation strategy
    decay = $N \times \eta$
    $\epsilon = 1 - e/\text{decay}$
    $\epsilon \leftarrow \epsilon \times \epsilon_{init}$
    $\epsilon \leftarrow clip(\epsilon, \epsilon_{min}, \epsilon_{init})$
    Action selection for observed traffic state S_t

$$Q_{\pi*} = \begin{cases} \underset{a}{\text{argmax}}\, Q_n & \text{if } \epsilon_{\text{random}} > \epsilon \\ \text{select random actions} & \text{otherwise} \end{cases}$$

    Execute action a_t in s_t, receives reward r_t and ends up in s_{t+1}
    store the experience tuple (s_t, a_t, r_t, s_{t+1}) in buffer memory $\mathcal{B}$
    derive $\mathcal{M}$ random minibatch tuple samples from memory $\mathcal{B}$

$$\text{Set } target(s,a) = \begin{cases} reward & \text{for terminal } s_t \\ \text{equation2} & \text{for non} - \text{terminal } s_t \end{cases}$$

    Update Target network as $\theta^T = \beta\theta^P + (1-\beta)\theta^T$
    Update Primary network by minimizing loss function MSE
  end for
end for

6. Numerical case-study

In this study, the Vissim Component Object Model (COM) is used to read, evaluate and change the simulation objects with python.

6.1. Simulation setting and parameters

The DDQN ATSC model for each data resolution is executed for 1000 episodes. Each episode is 1800 second which is the two consecutive 15 minutes volume data collected at the intersection for morning rush hour. The origin-destination demand matrices are as follows:

Table 3-1: Demand matrix number 1

Date: 08-01-2019 , time: 08:00-08:15				
	N	**E**	**S**	**W**
N	0	53	217	13
E	6	0	46	120
S	37	12	0	15
W	10	49	11	0

Table 3-2: Demand matrix number 2

Date: 08-01-2019 , time: 08:15-08:30				
	N	**E**	**S**	**W**
N	0	67	261	14
E	5	0	54	113
S	40	12	0	16
W	10	69	12	0

6.2. Experimental results

To assess the performance of the deep RL models, data measurement and queue estimators are added in Vissim simulation model. The experimental results are set out both in tables and plots. The Table 4 represents the mean value the last 100 episodes with the 95% confidence interval. To better visualization, the plots are a simple moving average over 50 episodes for the whole training episodes.

6.2.1. cumulative reward

The reward for ATSC-DTSE is the summation of $r_t = -(1 - 0.2v_t)$ for each time step during the action execution which consists of vehicles individual speed data. However, the ATSC-QLDS and ATSC-QLAS reward is the summation of queued vehicles $r_t = -(q_{i,j})$. As the ATSC-DTSE data resolution is higher than ATSC-QLAS, the rewards are more accurate for RL agent to take the optimal actions.

6.2.2. Performance measures

According to Table1, the ATSC-DTSC is superior in all performance measures. The ATSC-DTSE outperforms ATSC-QLAS (with queue length and average speed data) for more than 4%. However, the improvement over the ATSC-QLDS is less than 0.7% which is quite negligible. Also, the ATSC-DTSE computation time is 30% more than ATSC-QLSD model training time.

Table 4: Performance metrices results of last 100 training episodes with 95% confidence interval

model	ATSC-QLAS (M1)	ATSC-QLDS (M2)	ATSC-DTSE (M3)	M3 Improvement over M1 (%)	M3 Improvement over M2 (%)
Cumulative reward	-22156.50 ± 163.56	-21352.31 ± 165.08	**-20691.06 ± 276.89**	6.61	3.1
Average travel time (s)	31.60 ± 0.13	30.90 ± 0.15	**30.76 ± 0.20**	2.66	0.45
Average delay (s)	21.56 ± 0.13	20.84 ± 0.15	**20.71 ± 0.20**	3.94	0.62
Average queue length (m)	4.93 ± 0.04	4.73 ± 0.05	**4.70 ± 0.07**	4.67	0.63
Average queue length maximum (m)	35.63 ± 0.45	**34.80 ± 0.45**	34.90 ± 0.68	2.05	-0.29
Average number of queue stops	72.99 ± 0.51	71.13 ± 0.37	**69.46 ± 0.43**	4.84	2.35

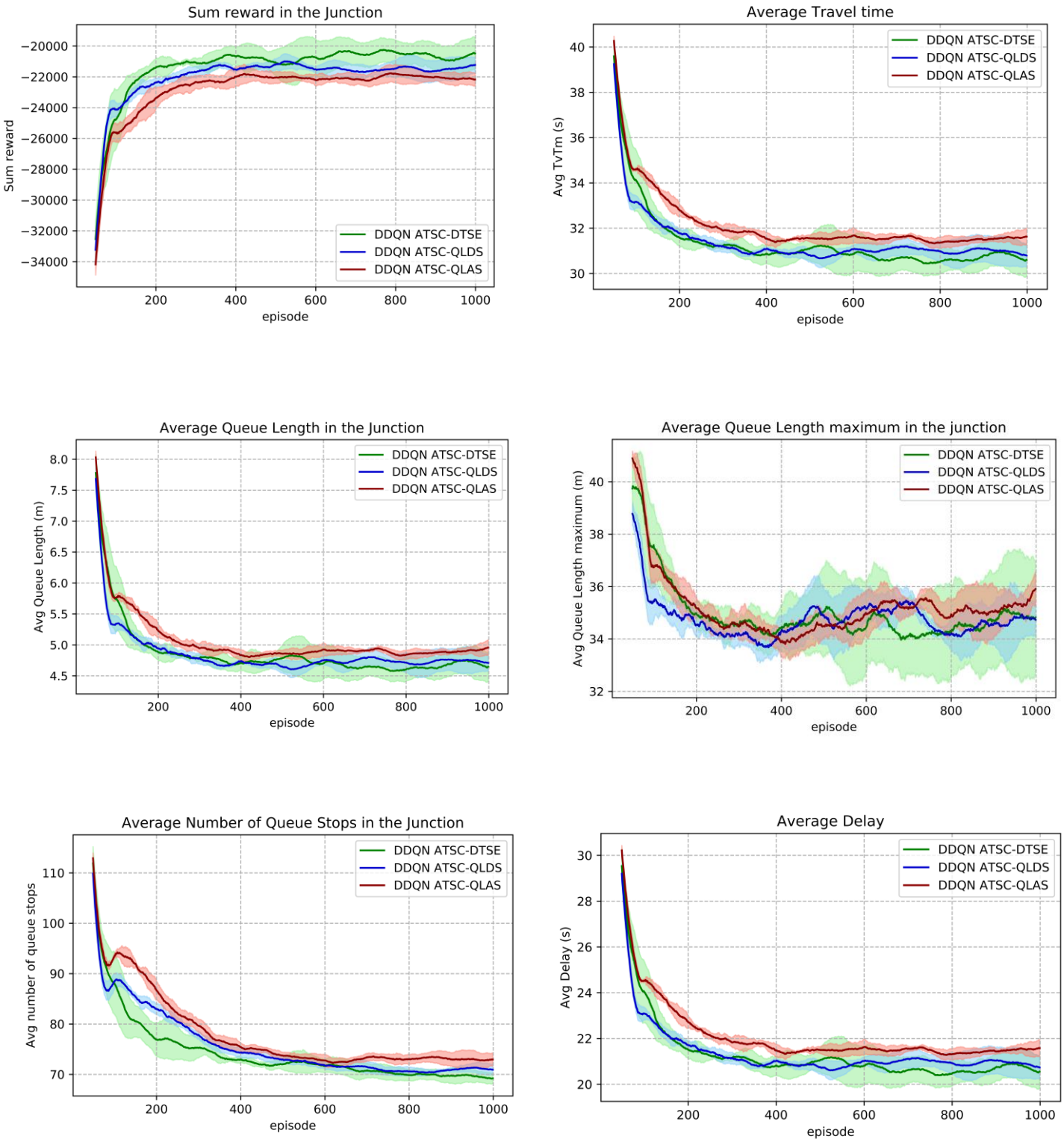

Figure 3: experimental results are the simple moving average over 50 episodes

7. Discussion

The simulation experiments have shown that ATSC-QLDS model (with queue length and density data) and ATSC-DTSE model (with information-dense data) have almost similar results in terms of travel time and delay reduction, queue length shortening. The ATSC-DTSE data is only available in a vehicular network which is hard and expensive when it comes to real-world implementation. The ATSC-QLDS data, in turn, can be collected by cameras at the intersection. Thus, it is recommended to use queue length and density data for the real-world applications of deep RL-based ATSC.

8. Conclusion

In this paper, we evaluate the performance of deep RL-based ATSC model with three different types of data resolutions. The first data is vehicles queue length of each incoming lane and vehicles average speed of the incoming links (ATSC-QLAS model). The second data is the vehicles queue length and density for incoming lanes which is provided by camera at the intersection (ATSC-QLDS model). The third and highest data resolution is speed and location of each individual vehicle in a vehicular network (ATSC-DTSE). The comprehensive experiments on an isolated intersection with DDQN algorithm agent has shown the superiority of ATSC-DTSE in all performance measures (over 4% versus ATSC-QLAS and less than 0.7% versus ATSC-QLDS). However, the ATSC-DTSE training time is 30% more than ATSC-QLDS. Also, the ATSC-QLDS model camera data is more accessible for real-world implementation while ATSC-DTSE must be collected in a vehicular network with high communication rates between vehicles. Hence, it is recommended to use the vehicles queue length and density data instead of information-dense speed and location data of each individual vehicle.

9. Reference

ABDOOS, M., MOZAYANI, N. & BAZZAN, A. L. Traffic light control in non-stationary environments based on multi agent Q-learning. 14th International IEEE conference on intelligent transportation systems (ITSC), 2011. 1580-1585.

AREL, I., LIU, C., URBANIK, T. & KOHLS, A. 2010. Reinforcement learning-based multi-agent system for network traffic signal control. *IET Intelligent Transport Systems,* 4**,** 128-135.

AUDIT, T. A. I. 2019. Urban Transport Crowding and Congestion.

BALAJI, P., GERMAN, X. & SRINIVASAN, D. 2010. Urban traffic signal control using reinforcement learning agents. *IET Intelligent Transport Systems,* 4**,** 177-188.

EL-TANTAWY, S. & ABDULHAI, B. An agent-based learning towards decentralized and coordinated traffic signal control. 13th International IEEE Conference on Intelligent Transportation Systems, 2010. 665-670.

EMAMI, A., SARVI, M. & ASADI BAGLOEE, S. 2020. A review of the critical elements and development of real-world connected vehicle testbeds around the world. *Transportation Letters***,** 1-26.

GAO, J., SHEN, Y., LIU, J., ITO, M. & SHIRATORI, N. 2017. Adaptive traffic signal control: Deep reinforcement learning algorithm with experience replay and target network. *arXiv preprint arXiv:.02755*.

GENDERS, W. 2019. Policy Analysis of Adaptive Traffic Signal Control Using Reinforcement Learning. *Journal of Computing in Civil Engineering,* 34**,** 04019046.

GENDERS, W. & RAZAVI, S. 2016. Using a deep reinforcement learning agent for traffic signal control. *arXiv preprint arXiv:.01142*.

GENDERS, W. & RAZAVI, S. 2019. Asynchronous n-step Q-learning adaptive traffic signal control. *Journal of Intelligent Transportation Systems,* 23**,** 319-331.

GRÉGOIRE, P.-L., DESJARDINS, C., LAUMÔNIER, J. & CHAIB-DRAA, B. Urban traffic control based on learning agents. IEEE Intelligent Transportation Systems Conference, 2007. 916-921.

JEON, H., LEE, J. & SOHN, K. 2018. Artificial intelligence for traffic signal control based solely on video images. *Journal of Intelligent Transportation Systems,* 22**,** 433-445.

LECUN, Y., BENGIO, Y. & HINTON, G. J. N. 2015. Deep learning. 521**,** 436-444.

LIANG, X., DU, X., WANG, G. & HAN, Z. 2019. A Deep Reinforcement Learning Network for Traffic Light Cycle Control. *IEEE Transactions on Vehicular Technology,* 68**,** 1243-1253.

MNIH, V., KAVUKCUOGLU, K., SILVER, D., RUSU, A. A., VENESS, J., BELLEMARE, M. G., GRAVES, A., RIEDMILLER, M., FIDJELAND, A. K. & OSTROVSKI, G. 2015. Human-level control through deep reinforcement learning. *Nature,* 518**,** 529.

MOUSAVI, S. S., SCHUKAT, M. & HOWLEY, E. 2017. Traffic light control using deep policy-gradient and value-function-based reinforcement learning. *IET Intelligent Transport Systems,* 11**,** 417-423.

PRASHANTH, L. & BHATNAGAR, S. 2010. Reinforcement learning with function approximation for traffic signal control. *IEEE Transactions on Intelligent Transportation Systems,* 12**,** 412-421.

SUTTON, R. S. & BARTO, A. G. 2018. *Reinforcement learning: An introduction*.

TAN, T., BAO, F., DENG, Y., JIN, A., DAI, Q. & WANG, J. 2019. Cooperative deep reinforcement learning for large-scale traffic grid signal control *IEEE transactions on cybernetics,* ePub.

TIELEMAN, T. & HINTON, G. 2012. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*.

VAN DER POL, E. & OLIEHOEK, F. A. 2016. Coordinated deep reinforcement learners for traffic light control. *Proceedings of Learning, Inference Control of Multi-Agent Systems*.

WANG, T., CAO, J. & HUSSAIN, A. 2021. Adaptive Traffic Signal Control for large-scale scenario with Cooperative Group-based Multi-agent reinforcement learning. *Transportation research part C: emerging technologies,* 125**,** 103046.

WEI, H., CHEN, C., WU, K., ZHENG, G., YU, Z., GAYAH, V. & LI, Z. 2019. Deep Reinforcement Learning for Traffic Signal Control along Arterials.

WEI, H., ZHENG, G., YAO, H. & LI, Z. Intellilight: A reinforcement learning approach for intelligent traffic light control. Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018. 2496-2505.

WIERING, M. Multi-agent reinforcement learning for traffic light control. Machine Learning: Proceedings of the Seventeenth International Conference (ICML'2000). 1151-1158.

WU, T., ZHOU, P., LIU, K., YUAN, Y., WANG, X., HUANG, H. & WU, D. O. 2020. Multi-agent deep reinforcement learning for urban traffic light control in vehicular networks. *IEEE Transactions on Vehicular Technology,* 69**,** 8243-8256.

ZHENG, G., ZANG, X., XU, N., WEI, H., YU, Z., GAYAH, V., XU, K. & LI, Z. 2019. Diagnosing Reinforcement Learning for Traffic Signal Control. *arXiv preprint arXiv:.04716*.