Traffic signal control using reinforcement learning based on the teacher-student framework

Junxiu Liu[1], Sheng Qin[1], Min Su[1*], Yuling Luo[1*], Shunsheng Zhang[1], Yanhu Wang[1], Su Yang[2]

[1]Guangxi Key Laboratory of Brain-inspired Computing and Intelligent Chips, School of Electronic and Information Engineering, Guangxi Normal University

[2]Department of Computer Science, Swansea University, Swansea, UK

Email: junxiu6@gxnu.edu.cn, qs_qinsheng@163.com, *{sumin0303, yuling0616}@gxnu.edu.cn,shunszhang@gxnu.edu.cn, 1695017072@qq.com, su.yang@swansea.ac.uk

Abstract: Reinforcement Learning (RL) is an effective method for adaptive traffic signals control. As one type of RL, the teacher-student framework has been found helpful in improving the model performance for different application fields (such as robot control, game, hybrid intelligence), but it is rarely applied for traffic control due to that the hyper-parameters and the number of state-action pairs experienced are difficult to determine. In this work, the teacher-student framework is used for traffic signal control, where only a single reward function is designed to guide the student agent and by using this method the number of hyper-parameters and the model complexity are reduced. Specifically, the teacher agent uses an importance function to evaluate and guide the student, where the importance function combines with environment reward to form a synthetic reward for the student agent. Experimental results under different traffic environments show that the proposed method achieves the expected performance enhancement and is better than most of the state-of-the-art RL-based traffic signal control methods.

Keywords: reinforcement learning; adaptive traffic signal control; teacher-student framework

1. Introduction

Adaptive Traffic Signal Control (ATSC) is a useful method for reducing traffic congestion. Previous studies had tried many methods for ATSC, such as heuristics (Cools et al., 2013; Wunderlich et al., 2008), fuzzy logic (Gokulan & Srinivasan, 2010; Qiao et al., 2011), and Reinforcement Learning (RL) (Liu et al., 2018; Mousavi et al., 2017a; Wei et al., 2018; Zheng, Xiong, et al., 2019). In particular, the performance of RL had been found superior to traditional transportation methods (Wei, Chen, et al., 2019), due to that RL is able to learn the knowledge of the traffic by interacting with the environment. In the approach of (Hester et al., 2018), an RL approach, namely Deep Q-learning from Demonstrations (DQfD), that can learn from demonstration data was proposed in the gaming field. The DQfD can use small sets of demonstration data to massively accelerate the learning process. Based on the DQfD, a method of learning to control light by demonstrations was proposed by (Xiong et al., 2019), in which demonstration data were collected by a traditional transportation method. However, many RL methods outperform traditional transportation methods (Wei et al., 2018; Zheng, Xiong, et al., 2019; Zheng, Zang, et al., 2019). In intuition, an RL agent is guided by a pre-trained RL agent that may be better than a traditional transportation method, which can be realized by the teacher-student framework (Torrey & Taylor, 2013).

In (Torrey & Taylor, 2013), the teacher agent (i.e., the pre-trained RL agent) gives action advices to the student agent (i.e., the new RL agent) on a budget, which introduced four methods, including early advising, importance advising, mistake correcting and predictive advising. These four methods are heuristic. The approach of (Zimmer et al., 2016) built a sequential decision-making problem to describe the teacher-student framework. In the hybrid intelligence systems, teachers always give advice to students on a budget, leading to unrealistic attentions and communication

demands. Thus, an interactive teaching strategy is introduced in (Kamar, 2016), which does not require teacher continuously monitor students, but needs to verify students' decisions when students seek advice. These methods used the teacher-student framework with single teacher model, and furthermore the approach of (Zhan et al., 2016) introduced a multiple teacher advice model. In order to reduce communication requirements, an ad hoc advisor-advisee model was proposed in a multi-agent environment. The ad hoc advisor-advisee relations were set according to the confidence between each agent (Da Silva et al., 2017).

Methods proposed by (Cruz et al., 2018; Torrey & Taylor, 2013; Zimmer et al., 2016) depend on a state importance function $I = max_a Q(s, a) - min_a Q(s, a)$. If I reaches a threshold value (a hyper-parameter), the teacher gives advice to the student. Note that this hyper-parameter of threshold value is set manually. The value of the hyper-parameter is difficult to determine through stable and credible methods, as it requires lots of experience or experimental verification. According to function I, the hyper-parameter i is dependent on the scale of the value function. In other words, the hyper-parameter i is dependent on the scale of the reward, which increases the difficulty of determining the hyper-parameter, especially in an environment with a complex reward function. In addition, the models proposed by (Da Silva et al., 2017; Zimmer et al., 2016) depend on the number of state-action pairs, which makes the model inapplicable in large state space environments, such as traffic signal control system.

In order to avoid cumbersome determination of the hyper-parameter i and count of the number of state-action pairs, an RL approach based on the teacher-student framework is proposed and applied to the ATSC in this work. The proposed approach does not limit the number of guidance (i.e., without any budget). The teacher agent guides the student agent by evaluating the importance of its action, rather than providing suggested actions. Specifically, the teacher agent is considered as a senior expert on the controlled system (i.e., the ATSC). The teacher agent evaluates the action of the student agent by an importance function, which is a part of the reward function of the student agent. The advantage of this approach is that it is no longer troubled by the hyper-parameter i. The proposed approach is evaluated in a traffic network, by using datasets with a certain difficulty level of the traffic environment. Numerical experiments confirm that the proposed approach outperforms several RL-based ATSC approaches. The main contributions of this work are: (1) A teacher-student framework with an importance function is applied to the ATSC, which can effectively reduce the traffic congestion. (2) The reward setting of proposed approach is explored and results show a better performance is obtained by using different reward setting rather than the same reward setting. (3) The performances of the proposed approach are evaluated by comparing with other traffic light control methods, and the results show that the proposed approach achieves a better performance.

The rest of this paper is organised as follows. Section 2 provides related works in the field of traffic light control. Section 3 presents the background of the teacher-student framework and RL. The proposed approach is introduced in Section 4 . The experimental results for evaluating and analyzing the performance of the proposed

approach is provided in Section 5. Finally, this work is concluded in Section 0.

2. Related Works

2.1. The teacher-student framework

A teacher-student framework for reinforcement learning was introduced in (Torrey & Taylor, 2013), which studies how the teacher suggests actions to the student. Based on this approach, the study of (Zimmer et al., 2016) further improved the performance, where the teacher learns how to teach the student by fitting a teaching policy, and the student uses an approach called max update to estimate the value function. The work of (Cruz et al., 2018) introduced the teacher-student framework to the robotic control filed, and summarized that a good teacher should have a fairly distributed experience. It shows that a pre-trained agent obtained high accumulative reward maybe is not a good teacher. An interactive teaching strategy was introduced by (Kamar, 2016) in hybrid intelligence systems, where the teacher does not suggest action to students proactively, but answer students when students consult. Furthermore, a multiagent advising framework was proposed in a shared environment (Da Silva et al., 2017), which allows agents advise each other. The advantage of this multiagent advising framework is that agents can accelerate learning, even if all agents do not have prior knowledge at the beginning.

2.2. Traffic signal control based on reinforcement learning

Traditionally, traffic signal control approaches can be classified into different categories, such as the fixed-time method (Miller, 1963), longest-queue-first method (Wunderlich et al., 2008), and self-organizing method (Cools et al., 2013). The fixed-time method controls the traffic signals by a fixed cycle that is about the green, red, and yellow light, in which the light duration of each colour is pre-set. The longest-queue-first method allows the directions with the longest queue length to have a green signal. The self-organising method needs a professional and experienced operator to set the parameters of the control program, which is inconvenient. According to the results in the approach of (Wei et al., 2018), these methods have better performance in the traffic environments with low traffic dynamics than the highly dynamic and complex traffic environments.

Given the shortcoming of the traditional methods, RL is applied to ATSC to deal with dynamic and complex traffic scenarios. RL learns to control the traffic signals by interacting with the traffic environment. The tabular Q-learning method was used to control traffic signals on isolated intersections in early works (Abdulhai et al., 2003; Balaji et al., 2010). However, the curse of dimensionality is an unavoidable defect of the tabular Q-learning. In recent years, the deep neural network is used to approximate

the value-function or the policy of the RL agent (Mnih et al., 2015), thereby eliminating this defect. The performances of deep policy-gradient methods and value-function-based methods on traffic environments were verified in (Mousavi et al., 2017b). The control logic of the DRL model was analyzed in literature (Wei et al., 2018), and several synthetic datasets and real-world datasets were used to verify the performances of the DRL model on static and dynamic traffic scenarios. In addition, multi-agent RL methods were used to control large-scale traffic signals. The shortcoming of multi-agent RL methods is the action space grows exponentially with the number of intersections (Tan et al., 2020a). Thus, decentralized training, limited communication, and hierarchical structure were used to overcome this shortcoming. An actor-critic model combines with long-short term memory, neighborhood fingerprints and a distance factor showed good performance when controlling large-scale traffic signals (Chu et al., 2020). A novel structure that combines graph attention network and communication was proposed in (Wei, Xu, et al., 2019), which used the attention mechanism to understand the importance of different parts of the communication message and realize cooperation between neighboring agents. The Nash-A2C and A3C were proposed by (Wu et al., 2022), which were used to construct the distributed internet of things computing architecture of urban traffic. Results showed that this computing architecture can effectively reduce the congestion of urban traffic.

All these RL methods learn traffic signals control from zero, which is inefficient. In (Hester et al., 2018), an RL approach learned from demonstration data was proposed in the game field. Another work (Xiong et al., 2019) based on (Hester et al., 2018) further learned the traffic signals control by using demonstration data obtained by a traditional traffic signals control method. The performance of the RL model was improved by using a traditional method to demonstrate. Previous work (Chen et al., 2020; Liu et al., 2018; Mousavi et al., 2017a; Tan et al., 2020a; Wei et al., 2018; Zheng, Xiong, et al., 2019; Zheng, Zang, et al., 2019) presented that the performance of the RL model is better than traditional methods. Thus, this work explores that using a well-trained RL agent to guide a new one.

Table I. Typical works of the RL-based TSC methods.

Ref	Scale	Model	State	Action	Reward
(Abdulhai et al., 2003)	Single	Q-learning	Queue length, phase time	Binary phase	Total delay
(Balaji et al., 2010)	Multiple	Q-learning	Occupancy, queue length,	Green time	History value,

			vehicle count		Q value
(Mousavi et al., 2017a)	Single	Deep Q-network, policy-gradient	Image representation	Green phase	Delay
(Wei et al., 2018)	Single	Deep Q-network	Queue length, number of vehicles, waiting time, vehicles' position	Binary phase	Queue length, delay, waiting time, light switches, vehicle number, travel time
(Tan et al., 2020a)	Multiple	Deep Q-network, Wolpertinger Architecture	Queue length	Green phase	Queue length, moving vehicle number
(Chu et al., 2020)	Multiple	Actor-critic	Vehicle number, cumulative delay	Green phase	Queue length, delay
(Wei, Xu, et al., 2019)	Multiple	CoLight	Vehicle number, phase	Green phase	Queue length
(Wu et al., 2022)	Multiple	Nash-A2C, Nash-A3C	Vehicle queue	Traffic phases	Waiting time

3. Background

3.1. Double Deep Q-Network

RL is always presented as a markov decision process, which is completely described by a 4-tuple (S,A,P,R). Suppose that an RL agent performs a task in an environment E. In this 4-tuple, S is a state set representing the situation of the environment E, A is an action set included the RL agent can execute, P is a dynamic transition distribution, and R is a reward set that is used to illustrate the quality of the student's executed action. At each time step t, an RL agent interacts with the environment E. The agent observes a state $s_t \in S$, and then executes an action a_t according to a policy π. The policy π is a rule, which maps s_t to a_t. The environment transfers the state s_t to a next state s_{t+1} under the effect of action a_t, feeds a reward $r_t \in R$ to the agent. The goal of the agent is to maximize a long-term return $R_t = \sum_{t=\tau}^{T} \gamma^{t-\tau} r_t$, where $\gamma \in [0, 1]$ is a discount factor. The expected total return is represented as its value function $Q(s_t,a_t) = \mathbb{E}[R_t|s = s_t, a = a_t]$, which estimates the value of action a_t under a given state s_t.

Deep Q-Network (DQN) is a classical RL algorithm, which approximates the value function $Q(s_t,a_t)$ by an evaluated network and a target network (Mnih et al., 2015). The DQN is based on the Q-learning algorithm, which estimates the value function as

$$Q(s_t,a_t) = Q(s_t,a_t) + \alpha\left(r_t + \gamma \max_a Q(s_{t+1},a) - Q(s_t,a_t)\right), \quad (1)$$

where $\max_a Q(s_{t+1},a)$ represents the maximal value under the state s_{t+1}. The loss function of the DQN updated the parameters of the evaluated network can be presented as

$$L(\Theta) = \mathbb{E}\left[\left(r_t + \gamma \max_a Q^t(s_{t+1},a|\Theta^-) - Q^e(s_t,a_t|\Theta)\right)^2\right], \quad (2)$$

where $Q^t(s_{t+1}, a|\Theta^-)$ represents the estimated value by the target network, $Q^e(s_t, a_t|\Theta)$ represents the estimated value by the evaluated network, Θ and Θ^- are the parameters of the target network and the evaluated network, respectively. The parameters Θ^- are only updated with the parameters Θ every C trained cycles, where C is constant.

Another key technique of the DQN is the experience replay. The experience replay collects the transition $[s_t, a_t, r_t, s_{t+1}]$ into a replay buffer M, where d indicates whether an episode ends (if the episode ends, $d = 1$; the episode does not end, $d = 0$). A mini-batch of experiences is randomly sampled from the buffer M when the evaluated network is trained.

A possible issue of DQN is that it may overestimate the action values, and the Double DQN (DDQN) can overcome this problem (Durugkar et al., 2016). In double Q-learning, the overestimation is reduced by using two Q functions (Q^A and Q^B) to estimate the action values (Van Hasselt, 2010). It can be presented as

$$Q^A(s_t, a_t) = Q^A(s_t, a_t) + \alpha\left(r_t + \gamma \max_a Q^B(s_{t+1}, a) - Q^A(s_t, a_t)\right), \quad (3)$$

$$Q^B(s_t, a_t) = Q^B(s_t, a_t) + \alpha\left(r_t + \gamma \max_a Q^A(s_{t+1}, a) - Q^B(s_t, a_t)\right),$$

One of the $Q^B(s_t, a_t)$ and $Q^A(s_t, a_t)$ is randomly selected to estimate the action value at time t. The DDQN uses a novel method to implement double Q-learning, i.e., it skillfully uses the target network to estimate action values. The function that estimates the action values only uses a network (i.e., the evaluated network). When the network is updated, the action selected on s_{t+1} is the greedy action of the evaluated network, rather than the Q_B or the target network. The loss function of the DDQN can be written as

$$L(\Theta) = \mathbb{E}\left[\left(Y^{DD} - Q^{e}(s, a|\Theta)\right)^2\right], \quad (4)$$

where $Y^{DD} = r_t + \gamma Q^t\left(s_{t+1}, \underset{a}{argmax}\, Q^e(s_{t+1}, a|\Theta)|\Theta^-\right)$ is a target value.

3.2. The teacher-student framework on a budget

In (Torrey & Taylor, 2013), a pre-trained RL agent is considered as a teacher with a fixed policy π_t, and a new RL agent with the same task as the teacher is considered as a student. The student learns a policy to complete the task with teacher's advice on a budget n. Based on the different way for teaching student, four algorithms were proposed, which are early advising, importance advising, mistake correcting and predictive advising, respectively. The early advising algorithm as shown in Algorithm 1, the teacher observes student's state, and advises an action according to π_t in first n states (i.e., the budget n). The Algorithm 2 describes the importance advising algorithm, which uses an importance function (Clouse, 1996) to identify the states needed advice. The importance function can be presented as

$$I(s) = \max_a Q(s, a) - \min_a Q(s, a). \quad (5)$$

The function $I(s)$ is computed by the value function of the teacher rather than the value function of the student, and is used to approximate a student's confidence in the state s. The mistake correcting algorithm (see Algorithm 3) adds a process of judging mistakes by comparing the student's announced action and the teacher's action. The predictive advising algorithm further improves model by using a prediction model, and the prediction model is trained using the states a student encounters and the actions it takes.

Algorithm 1. Early Advising

---

teacher's policy π_t, a budget n;

for each student state s do

if $n > 0$ then

$n \leftarrow n - 1$;

Advise $\pi_t(s)$;

Algorithm 2. Importance Advising

---

teacher's policy π_t, a budget n, importance threshold t;

for each student state s do

if $n > 0$ and $I(s) \geq t$ then

$n \leftarrow n - 1$;

Advise

Algorithm 3. Mistake Correcting

---

teacher's policy π_t, a budget n, importance threshold t;

for each student state s do

Observe student's announced action a;

if $n > 0$ and $I(s) \geq t$ and $a \neq \pi_t(s)$ then

$n \leftarrow n - 1$;

Advise $\pi_t(s)$;

Algorithm 4.Predictive Advising

teacher's policy π_t, a budget n, importance threshold t;

for each student state s do

Predict student's intended action a;

if $n > 0$ and $I(s) \geq t$ and $a \neq \pi_t(s)$ then

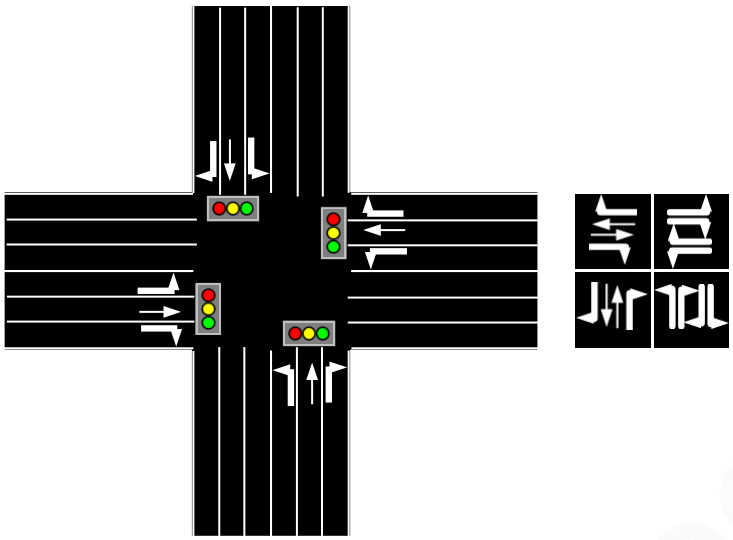
$n \leftarrow n - 1$;

Advise $\pi_t(s)$;

4. Proposed framework

4.1. Problem definition

The ATSC problem can be presented as a Markov Game (Tan et al., 2020b). Each intersection in a traffic environment is controlled by an RL agent in a discrete-time system. A group of phases is defined to control the movement of traffic flow, which is shown in Figure 1. The information of the intersection can be observed by the corresponding agent at time t, which is a state s_t. Then the agent executes an action a_t under the state s_t. A transition dynamic happens $s_{t+1} \backsim p(\cdot|s_t, a_t)$ and a reward r_t is received. In this work, the state s_t contains the number of vehicles on entering lanes, the queue length of entering lanes, and a one-hot vector about the current phase. The action a_t is a phase. The reward r_{env} comes from the environment, which is defined as the sum of all queue lengths in all entering lanes.



(a) Intersection (b)Four phases

Figure 1. The illustration of an intersection with four phases.

4.2. Importance function and guidance reward function

In the teacher-student framework, the hyper-parameter n is used to limit the number of states that need guidance, and another threshold hyper-parameter i is used to indicate that an advice action can be given by the teacher (Torrey & Taylor, 2013). As it was pointed out in Section 1, the thresholds n and i increase the complexity of the teacher-student framework. In addition, the models proposed by (Da Silva et al., 2017; Zimmer et al., 2016) need to count the number of state-action pairs, which makes the model inapplicable in large state space environments. In order to avoid these troubles, a method is suggested to reduce this difficulty in this work. The importance of the action that is selected by the student (the student action) is measured by using an importance function. The importance function can be represented as

$$I(a_t^{stu}) = \frac{Q_{tea}^{e}(s_t, a_t^{stu}) - \mathbb{E}[Q_{tea}^{e}]}{\max_{a} Q_{tea}^{e}(s_t, a) - \mathbb{E}[Q_{tea}^{e}]} \quad (6)$$

where a_t^{stu} is the student action, and Q_{tea}^{e} represents the action values of the teacher that is estimated by the evaluated network of the teacher model. The numerator and denominator of the equation are the advantage function (Wang et al., 2016) of the student action a_t^{stu} and the greedy action of the teacher, respectively. The advantage function means a relative measure of the importance of each action. The denominator

presents the relative measure of the most important action of the teacher, which is a baseline to measure the importance of the student action. The numerator presents the relative measure of the importance of the student action, which is based on the teacher action value.

After the importance of the action is obtained, a guidance reward function is given by

$$r_{gui} = I(a_t^{stu}) + r_{env}. \quad (7)$$

The first part of this equation is the importance of the student action, it is used to guide the agent when the guidance reward r_{gui} is fed to the agent. The second part is the external reward, which is used to reflect the information of the traffic environment.

4.3. Methodology

The proposed approach is based on the teacher-student framework, which is shown in Figure 2. To elaborate the process, a state s_t is observed by the teacher and student at time t. The action applied to the environment is selected by the student. Meanwhile, the action selected is sent to the teacher. An external reward r_{env} is provided by the traffic environment, and a teacher reward I is calculated by the importance function. A synthetic reward r_{gui} is fed to the student finally.

The teacher is firstly pre-trained in a separate traffic scenario, it is then reused in other traffic scenarios. The student with randomly initialized parameters is guided by the pre-trained teacher. The policies of the student and teacher are not the same. In the pre-train stage, the policy of the teacher is the ε-greedy policy (Watkins, 1989). In the reuse stage, the teacher adopts the greedy policy, as the teacher in this case is considered an expert. The policy of the student is also the ε-greedy policy. The major difference between the ε-greedy policy and the greedy policy is that the ε-greedy policy can explore by random action with a certain probability. The different traffic scenarios have different transition dynamics, although the teacher is considered an expert, it is necessary for the teacher to be further trained to improve the teaching effect. Thus, the teacher is further trained based on the experiences of the student in the guidance process.

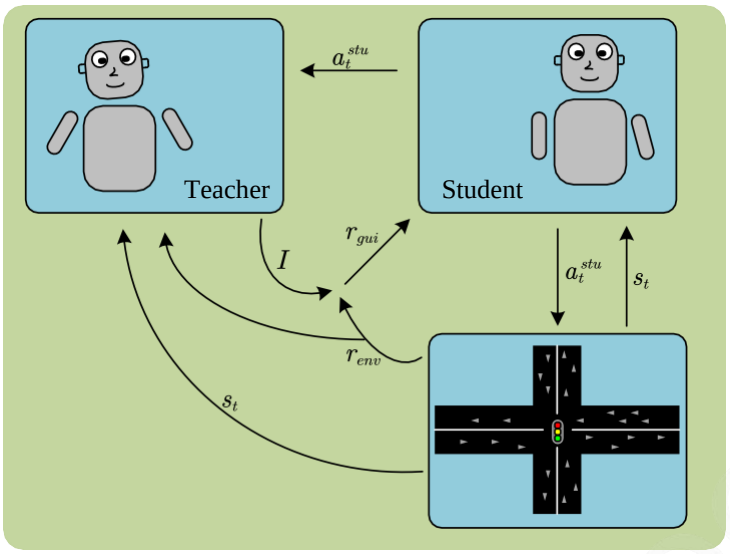


Figure 2. The interaction system of teacher and student.

Both the student and teacher are re-trained/trained by the DDQN model, the overall architecture of the model is shown in Figure 3. As introduced in Section 4.1, as the state is a vector, the adopted evaluation network is a multilayer perceptron with an input layer, two hidden layers, and an output layer. The number of neurons in the input layer is the same as the length of the state vector. The number of neurons in the two hidden layers is 100 and 50, respectively. The number of neurons in the output layers is the same as the size of the action space, i.e., it is 4 in this work. The parameters of the evaluated network are optimized by the Adam method (Kingma & Ba, 2014). The updated way of the target network is soft-update (Lillicrap et al., 2015), which can be given by

$$\Theta^{-} = (1 - \tau)\Theta^{-} + \tau\theta, \quad (8)$$

where τ is a constant coefficient, and $\tau \ll 1$.

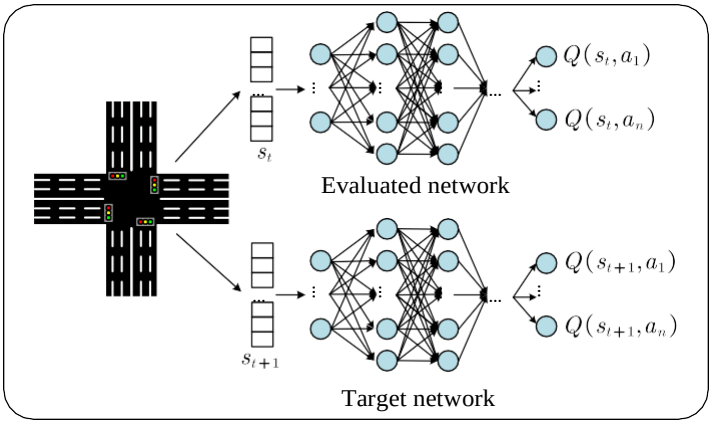


Figure 3. The evaluated network and target network.

5. Results

5.1. Simulation platform, datasets and parameters setting

The proposed approach is tested and verified in a traffic grid. The traffic grid is constructed by CityFlow (Zhang et al., 2019). CityFlow is an open-source traffic simulation platform, and it supports large-scale city traffic signal control. The traffic grid consists of 4×4 intersections. Each intersection is set to be a four-way intersection, with four 300-meters long road segments. Four synthetic traffic datasets are applied to the traffic grid, as shown in Table II. In the synthetic datasets, two types of vehicles' arrival rates are provided, i.e. Flat (0.3 variances) and Peak (0.6 variances) patterns. Furthermore, the proposed approach is tested and verified in three real-world traffic datasets and corresponding traffic grids, including Jinan, Hangzhou, and New York (Wei, Xu, et al., 2019b). The details about three real-world traffic datasets are listed in Table III. The parameters of each agent are shown in Table IV. The following subsections show explore the impacts of the reward toward the student and teacher agents to the proposed system (Section 5.2 & 5.3), as well as a comprehensive comparison with the state-of-the-arts works (Section 5.4).

Table II. The synthetic traffic datasets (Chen et al., 2020).

Config	Demand pattern	Arrival rate (vehicles/s)
1	Flat	0.388
2	Peak	

3	Flat	0.416
4	Peak	

Table III. Data statistics of three real-world traffic datasets (Wei, Xu, et al., 2019b).

Dataset	intersections	Arrival rate (vehicles/300s)			
		Mean	Std	Max	Min
$D_{NewYork}$	196	240.79	10.08	274	216
$D_{Hangzhou}$	16	526.63	86.70	676	256
D_{Jinan}	12	250.70	38.21	335	208

Table IV. The parameters of each RL agent.

Parameter	Value
Batch size	64
Learning rate	0.001
Memory length	3600
ϵ for exploration (student)	0.01
Target network update τ	0.05

5.2. Impact of the same external rewards

In this experiment, the teacher and student have the same way to understand the knowledge of the traffic environment. Under this setting, the teacher and student have the same external reward r_{env}, which shows the effect of RL policy. The difference is that the real reward of the student is r_{gui}. The r_{env} is the average queue length in this experiment. The teacher is pre-trained on Config 3 when it is applied to Config 1, 2, and 4, and it is pre-trained on Config 2 when it is applied to Config 3. The average travel time of the proposed approach and DDQN is listed in Table V. The standard deviation (std) of the average travel time is listed in Table VI. The proposed approach is trained 100 episodes. DDQN-100 represents that the training time of the DDQN agent is 100 episodes, and DDQN-200 represents that the training time of the DDQN agent is 200 episodes, which simulate that an agent is pre-trained and is loaded to continue train. For the DDQN-200, the teacher is also trained during guiding the student.

The results of the average travel time show that the proposed approach has a lower average travel time than DDQN when both approaches are trained same time, i.e., the proposed approach is better than DDQN without guidance. Due to that the teacher is also trained when it guides the student, the training time of DDQN increases to 200 episodes. Although the gap between the proposed approach and DDQN reduces from 3.6% to 1.2% when the DDQN is trained with more episodes, the average travel time of the proposed approach is smaller than DDQN. This result illustrates that an agent with expert guidance and self-exploration can reduce training time compared with an agent with only self-exploration. In addition, the results of the std show that the proposed approach is more stable than DDQN no matter how long DDQN training is under most scenes. The average waiting times of the proposed approach and DDQN on four configurations are provided in Table VII. As same as the average travel times, the average waiting times of the proposed approach is smaller than that of DDQN-100. As discussion in (Pol & Oliehoek, 2016), the average waiting time is a proxy for the average travel time. Thus, the same conclusion can be obtained from both, i.e. the proposed approach is more effective than DQN to reduce travel time for the same training episodes. Although the average travel time and waiting time are not part of reward function, the queue length is also a proxy for the average travel time (Zheng, Zang, et al., 2019). Figure 4 shows the convergence speed of the proposed approach and DDQN, where the performance improvement of the student model is limited when teachers and students understand the traffic environment in the same way. This is due to that the final reward of student agent varies linearly in most scenes, while the teacher agent has different policies comparing to student agent when encountering some special scenes after pre-trained.

Table V. The average travel time of the proposed approach and the DDQN method under the same external reward. DDQN-100 represents that a DDQN agent is trained 100

episodes, DDQN-200 represents that a DDQN agent is trained 200 episodes. (unit: second)

Method	Config1	Config2	Config3	Config4
This work	**279.78**	**294.33**	**297.01**	**322.35**
DDQN-100	297.05	305.07	313.12	340.54
DDQN-200	285.19	298.04	303.05	329.53

Table VI. The std of the travel time of the proposed approach and the DDQN method under the same external reward. DDQN-100 represents that a DDQN agent is trained 100 episodes, DDQN-200 represents that a DDQN agent is trained 200 episodes. (unit: second)

Method	Config1	Config2	Config3	Config4
This work	**1.03**	**1.08**	6.23	**2.27**
DDQN -100	1.71	1.47	**1.07**	4.13
DDQN -200	1.91	1.63	1.99	3.34

Table VII. The average waiting time of the proposed approach and the DDQN method under the same external reward. (unit: second)

Method	Config1	Config2	Config3	Config4
This work	**162.64**	**179.79**	**180.19**	**198.13**
DDQN -100	181.57	191.83	198.27	206.99

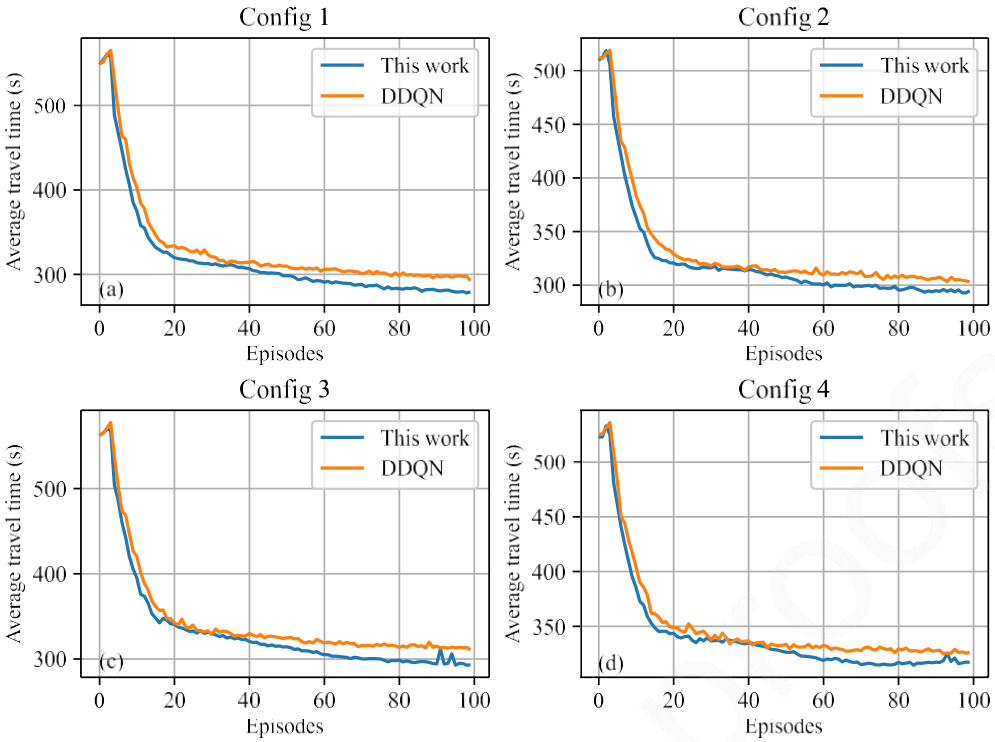


Figure 4. Convergence speed of the proposed approach and DDQN during training.

5.3. Impact of the different external rewards

In this experiment, the teacher and student have different external reward r_{env}. The r_{env} of the student is the pressure reward (Wei, Chen, et al., 2019), and the r_{env} of the teacher is the queue length reward (Zheng, Zang, et al., 2019). Different external rewards for teachers and students means that they have different ways to understand the knowledge of the traffic environment. This also means that teachers guide students from a novel perspective. The average travel time of the proposed approach and DDQN is listed in Table VIII. The proposed approach is trained 100 episodes in this experiment. The implication of the DDQN-100 and DDQN-200 are the same as Section 5.2.

The proposed approach has a lower average travel time than DDQN in all four traffic scenarios. Specifically, compared with DDQN-100, the proposed approach reduces the average travel time by 27.91%, 16.97%, 24.99%, and 15.73% in Config 1, 2, 3 and 4, respectively. Compared with DDQN-200, the proposed approach reduces the average travel time by 16.61%, 10.26%, 14.43%, and 8.21% in Config 1, 2, 3 and 4, respectively. Although the DDQN is trained more times, its performance is still worse than the proposed approach. The std of the average travel time is listed in Table IX, and the proposed approach has a lower std than DDQN. The result shows that the proposed approach can improve the stability of the system. The average waiting times is listed in Table X. Compared with the DDQN-100, the proposed approach reduces the average waiting time from 39.83% to 60.72%. In Figure 5, the convergence speed of the proposed approach and DDQN is shown, where the proposed approach has a faster convergence speed (the curve of the proposed method is below DDQN) and is more

stable than the DDQN under four traffic scenarios (the proposed approach has fewer spikes than DDQN). The DDQN and the student have the same r_{env}, but the student has a teacher guiding which shows the importance of the teacher. Compared with the coverage speed of experiment 1, the coverage speed gap between the proposed approach and DDQN increases. This result shows that the use of different external rewards for teachers and students can improve the coverage speed of the student.

Table VIII. The average travel time of the proposed approach and the DDQN method under the different external reward. DDQN-100 represents that a DDQN agent is trained 100 episodes, DDQN-200 represents that a DDQN agent is trained 200 episodes. (unit: second)

Method	Config1	Config2	Config3	Config4
This work	**281.75**	**291.71**	**293.60**	**314.74**
DDQN-100	360.38	341.26	366.98	364.25
DDQN-200	328.55	321.64	335.97	340.59

Table IX. The std of the travel time of the proposed approach and the DDQN method under the different external reward. DDQN-100 represents that a DDQN agent is trained 100 episodes, DDQN-200 represents that a DDQN agent is trained 200 episodes. (unit: second)

Method	Config1	Config2	Config3	Config4
This work	**1.13**	**0.97**	**1.04**	**1.51**
DDQN-100	2.64	3.48	4.65	4.39
DDQN-200	1.53	2.28	1.61	1.58

Table X. The average waiting time of the proposed approach and the DDQN method under the different external reward. (unit: second)

Method	Config1	Config2	Config3	Config4

This work	**169.52**	**174.18**	**174.11**	**191.25**
DDQN -100	272.46	243.55	278.05	268.14

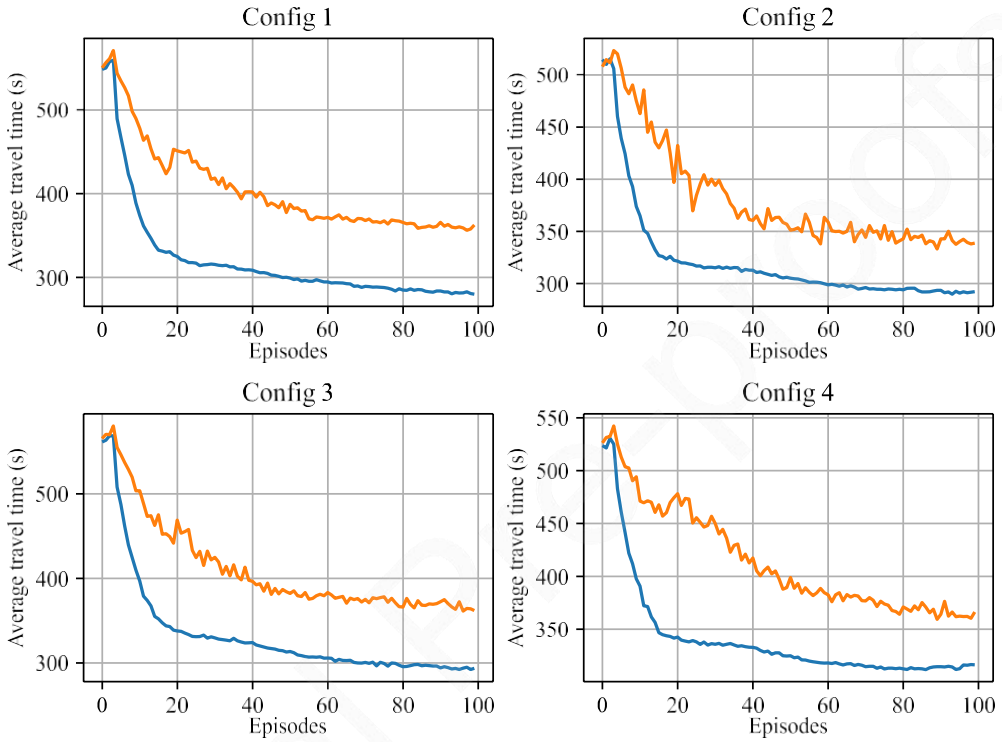


Figure 5. Convergence speed of the proposed approach and DDQN during training.

5.4. Performance comparison with existing methods

In this experiment, the reward of the teacher and student is the same as the Section 5.3. The baseline methods are as follows:

- the FixedTime method (F, 2008): This method controls traffic signals by a fixed cycle with a preset green ratio split among all phases.
- the MaxPressure method (Varaiya, 2013): A concept, named pressure, is defined as the difference in the number of vehicles between incoming lanes and outcoming lanes. The maxPressure method selects a phase with the max pressure to ensure the throughput.
- the GRL method (Pol & Oliehoek, 2016): The GRL method based on deep Q-learning is a coordinated deep reinforcement algorithm, which combines with transfer planning and max-plus coordination algorithm.

- the GCN method (Nishi et al., 2018): This method uses a graph convolutional neural network to generate geometric features about traffic information, and it also is an RL-based model.

- the NeighborRL method (Arel et al., 2010): This model uses its own and its neighbors' observation as a state, which is a multi-agent deep q-learning algorithm.

- the PressLight method (Wei, Chen, et al., 2019): The PressLight is an RL-based method, which uses pressure as its reward to unite its neighbors.

- the FRAP method (Zheng, Xiong, et al., 2019): A state-of-the-art RL-based traffic signal control method, which uses a novel network structure to capture the phase competition relation between different traffic movements.

- the MPLight method (Chen et al., 2020): It is based on the FRAP model and PressLight model, and is a decentralized deep RL method for large-scale traffic signal control.

In all baseline methods, FixedTime and MaxPressure are conventional transportation methods, and others are RL-based methods. The average travel time of the proposed approach and other baseline methods is listed in Table XI. The FixedTime uses a pre-determined plan for cycle length and phase time to control traffic signals, which lacks flexibility. Thus, it has a higher average travel time than the proposed approach. The MaxPressure controls larger-scale traffic signals by selecting the phase with maximal pressure. As discussed in (Wei, Chen, et al., 2019), The MaxPressure is often implemented in a greedy manner, which leads to a local optimum. Due to it, the performance of MaxPressure is much worse than the proposed approach.

These results show that the proposed approach is better than conventional transportation methods. Compared with the existing RL methods, the proposed approach has been found more effective than GRL, GCN, NeighborRL, and PressLight in all traffic scenarios. The reward function of PressLight is as same as the r_{env} of the proposed approach. The difference between PressLight and the proposed approach is that the proposed approach has a teacher agent and a student agent with pressure reward, but PressLight only has an agent with pressure reward. These results show that the teacher's guidance can effectively improve the performance of the RL model. Compared to the MPLight, the performance of the proposed approach is more robust. In other words, the proposed approach has smaller performance changes in different traffic environments than the MPLight (the maximal performance gap of MPLight is 34.5%, the maximal performance gap of the proposed approach is 11.6%). This result illustrates that the proposed approach has wider traffic environment applicability than MPLight. Similar conclusions can be drawn when it is compared to FRAP (the maximal performance gap of FRAP exceeds 100%).

Table XI. The average travel time of the proposed approach and other baseline methods.

(unit: second)

Method	Config1	Config2	Config3	Config4
FixedTime (F, 2008)	573.13	564.02	536.04	563.06
MaxPressure (Varaiya, 2013)	361.17	402.72	360.05	406.45
GRL (Pol & Oliehoek, 2016)	735.38	758.58	771.05	721.37
GCN (Nishi et al., 2018)	516.65	523.79	646.24	585.91
NeighborRL (Arel et al., 2010)	690.87	687.27	781.24	791.44
PressLight (Wei, Chen, et al., 2019)	354.94	353.46	348.21	398.21
FRAP (Zheng, Xiong, et al., 2019)	340.44	298.55	361.36	598.52
MPLight (Chen et al., 2020)	309.33	262.50	281.34	353.13
This work	281.75	291.71	293.60	314.74

5.5. Performance comparison with existing methods in real-world traffic data

In this experiment, all approaches are tested in three real-world traffic datasets. Some of these approaches are introduced in Section 5.4, and others are introduced as follows:

CoLight-node (Wei, Xu, et al., 2019b): This approach uses graph attentional networks to facilitate communication, and the neighbourhood scope of an agent is constructed through the smallest hop count between two nodes (i.e. node distance).

CoLight (Wei, Xu, et al., 2019b): This approach is similar to the CoLight-node, but the neighbourhood scope of an agent is constructed through the geo-distance between two intersections' geo-locations.

Individual RL (Wei et al., 2018): This approach is based on DQN algorithm, which

uses a phase-gated model and memory palace structure to improve the performance of model.

OneModel (Chu et al., 2020b): This approach is an actor-critic algorithm, which uses a long-short term memory network to memorize short history. It obtains neighbor policies to extend the knowledge of an agent.

All experimental results are listed in Table XII. On the real-world traffic dataset $D_{NewYork}$, the proposed approach has a better performance than other methods. The best one of the conventional transportation methods is MaxPressure, whose average travel time is 1633.41 seconds. Compared with the MaxPressure, the proposed approach reduces the average travel time by 39.5%. The Individual RL cannot scale up to 196 intersections in New York's road network. The CoLight has a higher travel time than
the proposed approach. For the datasets $D_{Hangzhou}$ and D_{Jinan}, the CoLight has lower travel times than the proposed approach. From the analysis of the data statistics of three real-world traffic datasets, the $D_{NewYork}$ has a lower std than other traffic datasets, which makes the proposed approach better than CoLight. In other words, the proposed approach is more suitable for handling a gentler traffic environment. While the CoLight uses the node distance to construct the neighbourhood scope of an agent, it has a higher travel time than the proposed approach for some travel patterns. Compared with other RL approaches (except CoLight and CoLight-node), the proposed approach reduces the average travel time by at least 6.2%. The reason behind this result is that the teacher-student framework gives the proposed method more experiences for dealing with traffic scenarios.

Table XII. The average travel time of the proposed approach and other methods in three real-world traffic data (Wei, Xu, et al., 2019b). (unit: second)

Method	$D_{NewYork}$	$D_{Hangzhou}$	D_{Jinan}
FixedTime (F, 2008)	1950.27	728.79	869.85
MaxPressure (Varaiya, 2013)	1633.41	422.15	361.33
GRL (Pol & Oliehoek, 2016)	2187.12	1582.26	1210.70
GCN (Nishi et al., 2018)	1876.37	768.43	625.32
NeighborRL (Arel et al., 2010)	2280.92	1053.45	1168.32

Individual RL (Wei et al., 2018)	-	345.00	325.56
OneModel (Chu et al., 2020b)	1973.11	394.56	728.63
CoLight-node (Wei, Xu, et al., 2019b)	1493.37	331.50	340.70
CoLight (Wei, Xu, et al., 2019b)	1459.28	297.26	291.14
This work	987.92	319.40	305.48

5.6. Discussion

In Section 5.2, 5.3 and 5.4, results show the teacher's guidance can improve the performance of student, even if the teacher and student have same external rewards. The effect of the teacher's guidance is obvious when the teacher and student have different external rewards. The DDQN has been trained for more episodes, but its performance is not better than the proposed approach. These results show that the importance function can guide the student to learn a more effective policy on traffic signal control. Compared to the studies of (Cruz et al., 2018; Torrey & Taylor, 2013; Zimmer et al., 2016), the proposed approach does not need additional hyper-parameters or to count the number of state-action pairs, which makes it very easy to use. Furthermore, for teacher-student framework the student has more perspectives to understand the world. The difference between the proposed method and a linear reward function is that the linear reward function is extremely sensitive to coefficients according to (Wei, Chen, et al., 2019).

As Table XIII shown, the average travel time of the proposed approach under the same external reward is lower than the one under the different external reward in Config 2, Config 3, and Config 4. This result illustrates that the teacher and student with the different external reward is better for the teacher to guide student. For different external reward setting, the teacher provides a novel view to the student, and the student may explore other policy space. Compared with transfer learning, the proposed approach has lower average travel times (see Table XIII). Transfer learning makes the agent keep its knowledge about other traffic scenarios at the beginning, but the knowledge is covered after being trained for many times. The proposed approach keeps the knowledge about other traffic scenarios by using a teacher agent and improves its ability through further training.

Table XIII. The average travel time of the proposed approach and transfer learning.

(unit: second)

Method	Config1	Config2	Config3	Config4
Same external reward	**279.78**	294.33	297.01	322.35
Different external reward	281.75	**291.71**	**293.60**	**314.74**
Transfer learning	283.66	297.01	296.31	317.46

In addition, the acceleration of training is not obvious at the beginning, as the student is initialized randomly. Another limitation (Cruz et al., 2018; Kamar, 2016; Torrey & Taylor, 2013; Zhan et al., 2016; Zimmer et al., 2016) is that the teacher needs to communicate with the student in every interaction, which leads to high communication costs.

6. Conclusion and Future Work

In this work, a novel RL approach based on the teacher-student framework is introduced to reduce the complexities caused by the additional parameters of the importance advising model (e.g., determining key parameter values empirically), hence further to improving its robustness. The proposed approach is based on the DDQN model, which uses an importance function to represent the guidance of the teacher agent. The importance function is combined with the external rewards from the environment to improve the performance of the proposed approach. The effect of the synthetic reward is verified in different external reward settings. Results show that the proposed approach has a greater improvement than the based model when the teacher and student have different external rewards. In different external rewards settings, the proposed approach reduces the average travel time by at least 5.53% compared with baseline DDQN. Results also show that the proposed approach is more suitable for traffic signal control than conventional transportation methods and most of RL-based methods. The proposed approach has a ~11.5% reduction in the average travel time compared with the MaxPressure, and ~8.5% reduction in the average travel time compared with the PressLight which has the same reward function as the external reward of the student agent. It is worth to note that the proposed approach only discusses the performance impact of different external reward settings within the teacher-student framework, which only improves the performance and learning speed of the student agent.

As described in Section 5.6, the random initialization of student agent constrains the training time reduction. In the future, the efficiency of the interactive experience

between the agents and environment can be further improved to help reduce the training time. In addition, the coordination of multi-agents in the teacher-student framework is also a direction of further study.

Acknowledgements

This research is supported by the National Natural Science Foundation of China under Grant 61976063, the Guangxi Natural Science Foundation under Grant 2022GXNSFFA035028, research fund of Guangxi Normal University under Grant 2021JC006, the AI+Education research project of Guangxi Humanities Society Science Development Research Center under Grant ZXZJ202205.

References

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

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

Balaji, P. G., German, X., & Srinivasan, D. (2010). Urban traffic signal control using reinforcement learning agents. *IET Intelligent Transport Systems*, *4*(3), 177–188.

Chen, C., Wei, H., Xu, N., Zheng, G., Yang, M., Xiong, Y., Xu, K., & Li, Z. (2020). Toward a thousand lights: decentralized deep reinforcement learning for large-scale traffic signal control. *Proceedings of the AAAI Conference on Artificial Intelligence*, 3414–3421.

Chu, T., Wang, J., Codeca, L., & Li, Z. (2020). Multi-agent deep reinforcement learning for large-scale traffic signal control. *IEEE Transactions on Intelligent Transportation Systems*, *21*(3), 1086–1095.

Clouse, J. A. (1996). *On integrating apprentice learning and reinforcement learning*. University of Massachusetts Amherst.

Cools, S.-B., Gershenson, C., & D'Hooghe, B. (2013). Self-organizing traffic lights: a realistic simulation. In *Advanced Information and Knowledge Processing* (pp. 45–55).

Cruz, F., Magg, S., Nagai, Y., & Wermter, S. (2018). Improving interactive reinforcement learning: What makes a good teacher? *Connection Science*, *30*(3),

306–325.

Da Silva, F. L., Glatt, R., & Costa, A. H. R. (2017). Simultaneously learning and advising in multiagent reinforcement learning. *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, 1100–1108.

Durugkar, I. P., Rosenbaum, C., Dernbach, S., & Mahadevan, S. (2016). Deep Reinforcement Learning with Double Q-Learning. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI)* (pp. 2094–2100).

F, K. Ge. (2008). Traffic signal timing manual. *United States. Federal Highway Administration*.

Gokulan, B. P., & Srinivasan, D. (2010). Distributed geometric fuzzy multiagent urban traffic signal control. *IEEE Transactions on Intelligent Transportation Systems* , *11*(3), 714–727.

Hester, T., Schaul, T., Sendonaris, A., Vecerik, M., Piot, B., Osband, I., Pietquin, O., Horgan, D., Dulac-Arnold, G., Lanctot, M., Quan, J., Agapiou, J., Leibo, J. Z., & Gruslys, A. (2018). Deep q-learning from demonstrations. *AAAI Conference on Artificial Intelligence*, 3223–3230.

Kamar, E. (2016). Directions in hybrid intelligence: Complementing AI systems with human intelligence. *IJCAI International Joint Conference on Artificial Intelligence*, 4070–4073.

Kingma, D., & Ba, J. (2014). Adam: a method for stochastic optimization. *ArXiv Preprint ArXiv:1412.6980*.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., & Wierstra, D. (2015). Continuous control with deep reinforcement learning. *ArXiv Preprint ArXiv:1509.02971*.

Liu, X.-Y., Ding, Z., Borst, S., & Walid, A. (2018). Deep reinforcement learning for intelligent transportation systems. *32nd Conference on Neural Information Processing Systems(NIPS)*, 1–8.

Miller, A. J. (1963). Settings for fixed-cycle traffic signals. *Journal of the Operational Research Society*, *14*(4), 373–386.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., & Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529–533.

Mousavi, S. S., Schukat, M., & Howley, E. (2017a). Traffic light control using deep policy-gradient and value-function-based reinforcement learning. *IET Intelligent Transport Systems*, *11*(7), 417–423.

Mousavi, S. S., Schukat, M., & Howley, E. (2017b). Traffic light control using deep policy-gradient and value-function-based reinforcement learning. *IET Intelligent Transport Systems*, *11*(7), 417–423.

Nishi, T., Otaki, K., Hayakawa, K., & Yoshimura, T. (2018). Traffic signal control based on reinforcement learning with graph convolutional neural nets. *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, 877–883.

Pol, E. van der, & Oliehoek, F. A. (2016). Coordinated deep reinforcement learners for traffic light control. *Proceedings of Learning, Inference and Control of Multi-Agent Systems*.

Qiao, J., Yang, N., & Gao, J. (2011). Two-stage fuzzy logic controller for signalized intersection. *IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans*, *41*(1), 178–184.

Tan, T., Bao, F., Deng, Y., Jin, A., Dai, Q., & Wang, J. (2020a). Cooperative deep reinforcement learning for large-scale traffic grid signal control. *IEEE Transactions on Cybernetics*, *50*(6), 2687–2700.

Tan, T., Bao, F., Deng, Y., Jin, A., Dai, Q., & Wang, J. (2020b). Cooperative deep reinforcement learning for large-scale traffic grid signal control. *IEEE Transactions on Cybernetics*, *50*(6), 2687–2700.

Torrey, L., & Taylor, M. E. (2013). Teaching on a budget: agents advising agents in reinforcement learning. *12th International Conference on Autonomous Agents and Multiagent Systems(AAMAS)*, 1053–1060.

Van Hasselt, H. (2010). Double Q-learning. *Advances in Neural Information Processing Systems*, 2613–2621.

Varaiya, P. (2013). Max pressure control of a network of signalized intersections. *Transportation Research Part C: Emerging Technologies*, *36*, 177–195.

Wang, Z., Schaul, T., Hessel, M., Van Hasselt, H., Lanctot, M., & De Frcitas, N. (2016). Dueling Network Architectures for Deep Reinforcement Learning. *33rd International Conference on Machine Learning(ICML)*, 2939–2947.

Watkins, C. J. C. H. (1989). Learning from delayed rewards. *PhD Thesis, Cambridge University*.

Wei, H., Chen, C., Zheng, G., Wu, K., Gayah, V., Xu, K., & Li, Z. (2019). Presslight: Learning max pressure control to coordinate traffic signals in arterial network.

Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 1290–1298.

Wei, H., Xu, N., Zhang, H., Zheng, G., Zang, X., Chen, C., Zhang, W., Zhu, Y., Xu, K., & Li, Z. (2019). CoLight:learning network-level cooperation for traffic signal control. *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 1913–1922.

Wei, H., Zheng, G., Yao, H., & Li, Z. (2018). IntelliLight: a reinforcement learning approach for intelligent traffic light control. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2496–2505.

Wu, Q., Wu, J., Shen, J., Du, B., Telikani, A., Fahmideh, M., & Liang, C. (2022). Distributed agent-based deep reinforcement learning for large scale traffic signal control. *Knowledge-Based Systems*, *241*, 108304.

Wunderlich, R., Liu, C., Elhanany, I., & Urbanik, T. (2008). A novel signal-scheduling algorithm with quality-of-service provisioning for an isolated intersection. *IEEE Transactions on Intelligent Transportation Systems*, *9*(3), 536–547.

Xiong, Y., Xu, K., Zheng, G., & Li, Z. (2019). Learning traffic signal control from demonstrations. *International Conference on Information and Knowledge Management, Proceedings*, 2289–2292.

Zhan, Y., Ammar, H. B., & Taylor, M. E. (2016). Theoretically-grounded policy advice from multiple teachers in reinforcement learning settings with applications to negative transfer. *IJCAI International Joint Conference on Artificial Intelligence*, 2315–2321.

Zhang, H., Ding, Y., Zhang, W., Feng, S., Zhu, Y., Yu, Y., Li, Z., Liu, C., Zhou, Z., & Jin, H. (2019). CityFlow: A multi-agent reinforcement learning environment for large scale city traffic scenario. *Proceedings of the World Wide Web Conference*, 3620–3624.

Zheng, G., Xiong, Y., Zang, X., Feng, J., Wei, H., Zhang, H., Li, Y., Xu, K., & Li, Z. (2019). Learning phase competition for traffic signal control. *International Conference on Information and Knowledge Management, Proceedings*, 1963–1972.

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:1905.04716*.

Zimmer, M., Viappiani, P., Weng, P., Zimmer, M., Viappiani, P., Weng, P., & Framework, T. (2016). Teacher-Student Framework : a Reinforcement Learning Approach. *AAMAS Workshop Autonomous Robots and Multirobot Systems*, 1–17.