2021 IEEE Intelligent Transportation Systems Conference (ITSC)
Indianapolis, USA. September 19-21, 2021

Centralized Traffic Signal Control for Multiple Intersections based on Sequence-to-Sequence model and Attention Mechanism

Le Ma [1†], Bo Xue[1†], Jia Wu[1*]

***Abstract*— Recently, the use of deep reinforcement learning techniques (DRL) has attracted increasing interest due to its ability of dynamical traffic signal control for multiple intersections. Only a few researches use the centralized control with single-agent to intelligently control all the signals, because the major problem, i.e., the curse of dimensionality, has not been successfully solved. We propose a novel centralized control method based on sequence-to-sequence model and attention mechanism to deal with this problem. The idea is similar to the Divide and Conquer paradigm. We mitigate the difficulty of searching in the huge space by dividing the state and action space into sub-spaces through sequence-to-sequence model. In addition, we greatly facilitate the communication and cooperation of traffic signals among intersections by introducing the attention mechanism. The DRL agent is trained by an efficient off-policy learning method - Proximal Policy Optimization. To the best of our knowledge, we are the first to use sequence-to-sequence method to deal with the huge search space problem in the traffic control. The comprehensive experiments demonstrate that our method can efficiently solve the curse of dimensionalitly problem and outperforms the traditional methods and other centralized control methods based on DRL.**

***Index Term*—Traffic signal control, Deep reinforcement learning, Sequence-to-sequence model, Attention mechanism**

I. INTRODUCTION

Urban Traffic Light Control (TLC) is an important but challenging problem in real life [1]. Its goal is to maximize the efficiency of urban traffic with limited urban traffic resources and improve the throughput rate of the traffic network, reduce the average end-to-end delay. TLC has always been a hot topic of research for researchers due to the complexity of the problem. A key question in TLC is: how to cooperate traffic signals to achieve the global optimization? [2], [3].

Nowadays, most traffic signals operate according to a fixed time scheme set in advance [4]. This scheme cannot adapt to the dynamically changing traffic conditions, and cooperate with the signals at the surrounding intersections [5]. At present, many researchers have used reinforcement learning (RL) techniques to dynamically control traffic signals at intersections. These methods usually have more advantages than traditional control methods. Since RL is a learning framework that allows the agent to learn the optimal control policy based on the feedback from the environment, RL has a great potential to deal with the issues that cannot be solved by conventional techniques [6].

There are two ways to solve the TLC problem using RL technology. One way is centralized control with single-agent to control the whole grid. The agent outputs the joint action for all intersections. However, as the scale of grid increases, the centralized control suffers from the curse of dimensionality [7]. Another way is decentralized control with multi-agents. Each agent controls a single intersection and shares information with the adjacent intersections. However, training a bunch of RL agents separately requires consideration of three issues: 1) The impact of surrounding intersections on the target intersection is changing dynamically. Agents need to dynamically capture this changing traffic information and assess the impact of surrounding intersections [8]. 2) Which adjacent intersections the agent should pay more attention to. 3) The agent hardly observes the state of the whole grid.

In this work, we design a centralized method to control the grid since theoretically the centralized method can co-operate traffic signals between intersections to find a global optimal solution. Inspired by the idea of *Divide and Conquer* paradigm, we decompose the state and action space to mitigate the problem of searching the optimal control policy in the large state-action space.

As shown in the Fig. 1, at each simulation time t, the agent perceives the state of the grid $\boldsymbol{s}_t = [s_1^t, \ldots, s_N^t]$ and generates the joint action $\boldsymbol{a}_t = [a_1^t, \ldots, a_N^t]$ for all intersections. s_i^t and a_i^t denote, respectively, the state and the control action of intersection i, $i \in [1, N]$ and N is the total number of intersections.

The agent consists of two parts: *actor* (for computing an action based on a state) and *critic* (to evaluate the value of the state). Actor is a sequence-to-sequence model, which is composed of *encoder* and *decoder*.

Encoder and Decoder unroll N time-steps. At time-step i, encoder network is fed in one sub-state s_i^t. And after ingesting the input sequence $\boldsymbol{s}_t = [s_1^t, \ldots, s_N^t]$, encoder outputs a vector h that represents the state of the whole grid. Decoder network takes h as input, the encoding output by the encoding network, and outputs decision a_1^t for intersection 1 at the first time-step. After that, a_1^t is fed to the decoder network to generate the next action a_2^t for intersection 2. The process continues until N time-steps, decoder network outputs the joint action $\boldsymbol{a}_t = [a_1^t, \ldots, a_N^t]$. In short, the state is embedded into a low dimensional vector and the action space is reduced at each time-step. Therefore, it is much easier for the agent to make decision at each time-step.

Naturally, the decision made for the target intersection

*This work was supported by the National Natural Science Foundation of China (Grant 61503059). Corresponding author: Jia Wu.

†*Equal Contribution*

[1]All the authors are with the School of Information and Software Engineering, University of Electronic Science and Technology of China, ChengDu, China. 15882190474@163.com; 15608065769@163.com; jiawu@uestc.edu.cn

2021 IEEE International Intelligent Transportation Systems Conference (ITSC) | 978-1-7281-9142-3/21/$31.00 ©2021 IEEE | DOI: 10.1109/ITSC48978.2021.9564818

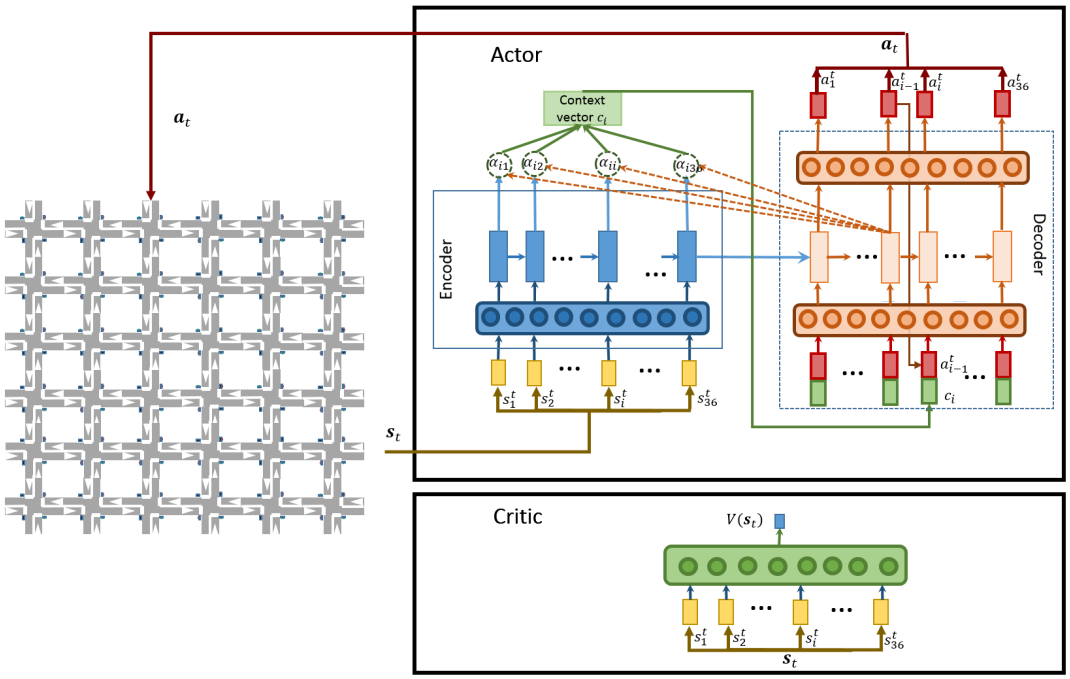

Fig. 1. Framework of the proposed model. At each simulation time t, the agent receives the state of the whole grid $\boldsymbol{s}_t = [s_1^t, \ldots, s_N^t]$ and makes a joint action $\boldsymbol{a}_t = [a_1^t, \ldots, a_N^t]$ for all intersections. Right: internal structure of the agent. The agent is composed of two models: actor and critic. Actor is composed of a sequence-to-sequence model with attention mechanism. Encoder digests the input $\boldsymbol{s}_t = [s_1^t, \ldots, s_N^t]$ and conveys it into hidden state h. Decoder receives the output of the encoder and outputs decision a_i^t for intersection i at each time-step i. The procedure repeats N times and gives a joint-action $\boldsymbol{a}_t = [a_1^t, \ldots, a_N^t]$, N is the number of intersections to control. Critic evaluates the value of the state at each intersection.

should consider the states of the surrounding intersections and explore the correlation between them. We specially design the attention mechanism to deal with this issue. Attention mechanism allows the decoder to pay attention to different parts of the source sequence at different decoding steps [9]. Thus, the agent can pay more attention to the target intersection and its surrounding intersections, which coordinates the signals between them efficiently.

To the best of our knowledge, this is the first work applying sequence-to-sequence with attention to achieve the centralized control in traffic network. Our contributions are as follows:

- **Mitigate the problem of the curse of dimensionality:** By the special design of the agent structure, our agent reads in the sub-state at each time-step and then memorizes the whole state and stores it in a low-dimension vector h. Thus, the large state space is compressed. The decoder network only selects action for one intersection at a time. The searching space of joint-action is decomposed into sub-spaces. As claimed by [10], any complex high-dimensional action can be selected incrementally, component by component, where each component's probability also depends on components already selected earlier. Both make the policy searching easier. Therefore, the problem of the curse of dimension is mitigated.
- **Coordinate the traffic signals between intersections:** To enable efficient cooperation between traffic lights, we apply attention mechanism, which can dynamically learn the influence of surrounding intersections on the target intersection. In the experimental part, we compare our method with the same sequence-to-sequence model with the attention mechanism removed, and the results show that the attention mechanism can well facilitate the information exchanging between intersections. In addition, by studying the spatial distribution of attention, the experimental results demonstrate that the attention model can learn the dynamics of the influence between intersections.
- **Conduct rich experiments to prove the proposed method:** We conduct an exhaustive experiment in a 6×6 traffic grid with 36 signals, and the maximum traffic flow rate at each lane is 800 vehicles per hour. To the best of our knowledge, none of the centralized methods with single-agent have tested on more than 16 signals [3]. The experimental results show that the proposed method can effectively solve the dimensional explosion problem, and outperforms other models.

The rest of this paper is organized as follows. Section II formally define the problem and the agent's design is shown in Section III. The experimental results are presented in Section IV. Finally, we conclude the paper in Section V.

II. PROBLEM DEFINITION

We consider a $n \times n$ traffic grid of $N = n^2$ intersections with the same layout. Each intersection has four ways and each way has one lane.The traffic signal at each intersection runs cyclically and each cycle has two traffic phases, i.e.,

North-South and East-West [11]. A DRL agent controls all the traffic signals in this grid. The agent observes the state of the whole grid and decides for all the intersections whether to keep the current phase or not. The goal of the agent is to minimize the average vehicle delay. The control problem can be formulated as Markov Decision Process (MDP) with five key elements $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma \rangle$. Next, we will detail the definitions of state $\mathcal{S}$, action $\mathcal{A}$, transition probability $\mathcal{P}$ and reward design r of our problem.

A. State

The agent observes the information of all intersections at t, $\boldsymbol{s}_t = [s_1^t, \ldots, s_i^t, \ldots, s_N^t] \in \mathcal{S}$. We define s_i^t the essential information of the state of intersection i at time t, which includes the speed ratio, the distance to the intersection of K vehicles around intersection i, the traffic density and the queue length of all lanes, and the current phase at intersection i. Specifically,

1) vr_k: the speed ratio of vehicle k on lane j, which is calculated by:
$$vr_k = \frac{v_k}{v_{max}^j} \tag{1}$$
where v_k is the speed of vehicle k around the intersection i on lane j; v_{max}^j is the maximum speed allowed on lane j.
2) dis_k: Distance from vehicle k to intersection.
3) $density_j$: Traffic density of the j^{th} lane :
$$density^j = \frac{\bar{L}_{veh}^j \times n_{veh}^j}{L^j} \tag{2}$$
where $\bar{L}_{veh}^j$ is the average length of vehicles in the j^{th} lane around the intersection; n_{veh}^j is the number of vehicles in the j^{th} lane; L^j is the length of the j^{th} lane.
4) $queue^j$: the queue length on lane j^{th} connected with the intersection, $j \in [1, 4]$;
5) p_i: the current phase of traffic signal at intersection i.

s_i^t is a $(2K + 2 \times 4 + 1)$ vector. And the state at t is a $(2K+2\times4+1)\times n^2$ vector and most of the components in this vector are real numbers. It is noted that the state space is quite large.

B. Action

$\mathcal{A}$ denote the actions space of the agent. In the traffic signal control problem, the agent decides, at each time t, a_i^t whether to keep the current phase or activate another phase for each intersection i. Thus, the action space of the agent is 2^{n^2}, which is exponentially large. In this paper, the action a_i^t is a real number. If $a_i^t < 0$, the phase in the North-South direction is activated; And if $a_i^t > 0$, the phase in the East-West direction is activated.

C. Transition probability

$\mathcal{P}$ denotes the transition probability to the next state $\boldsymbol{s}_{t+1}$, given the current state $\boldsymbol{s}_t$ and the joint action $\boldsymbol{a}_t$ of the agent at time t, $\mathcal{P}(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, \boldsymbol{a}_t) : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$. For the traffic signal control problem, due to the inherent complexity of the problem, the transition probability is unknown.

D. Reward

Agent receives a reward r_t after taking an action $\boldsymbol{a}_t$. Reward is a number that shows how good or bad the current state is, $S \times \mathcal{A} \to \mathbb{R}$. The goal of the agent is to maximize its cumulative reward, which is also called return: $\sum_{t=1}^{T} \gamma^t r_t$, where T is the total simulation time and γ is the discount factor. The design of rewards is significant and can affect the final performance to some extent [12]. In this paper, we define the reward for the whole traffic network as average vehicle delay.

III. METHOD

We use one of the modern Actor-Critic algorithms, Proximal Policy Optimization (PPO-Clip) algorithm to train the agent. In this section, we will first introduce the encoder and decoder network of the actor model, which is a sequence-to-sequence model with attention. Then, we will present the training algorithm of the agent.

A. Encoder

Encoder aims to find the encoding of the state and help decoder network to generate the corresponding decisions.

For notation convenience, we will delete the superscript t on state and action in the following.

Since the state dimension of each intersection is too large, we first compress the raw observation data s_i of each intersection into a low dimensional vector s_i' via a linear transformation:

$$s_i' = W_e s_i + b_e \tag{3}$$

where W_e and b_e are embedding parameters. And then convert vector s_i' into vector h_i^{enc}:

$$h_i^{enc} = GRU_{enc}(s_i', h_{i-1}^{enc}) \tag{4}$$

Where h_i^{enc} is a hidden state of the encoder at time-step i. Encoder network is composed of GRU.

B. Decoder

Decoder makes decisions for all intersections based on the output of encoder network. However, the decoder cannot fully pay attention to the relationship between adjacent intersections. For example, when the agent tries to generate the action for intersection i, it seems like that the agent should be looking primarily at the surrounding intersections of i. Attention mechanism makes all this work much better. It computes a set of attention weights, which indicate how much attention the agent should put to the other intersections. The specific steps are as follows:

Decoder generates the action a_i given the context vector c_i, the previously predicted action a_{i-1} and the hidden state h_{i-1}^{dec} [9]:

$$\begin{aligned} h_i^{dec} &= GRU_{dec}(h_{i-1}^{dec}, g_i) \\ g_i &= Relu(W_r[c_i; a_{i-1}] + b_r) \end{aligned} \tag{5}$$

where W_r and b_r are weight and bias parameters for linear transformation.

The context vector c_i contains information about the whole traffic network with a strong focus on the parts surrounding the i^{th} intersection. c_i depends on a sequence of hidden states of encoder $(h_1^{enc}, \ldots, h_N^{enc})$, and is calculated by:

$$c_i = \sum_{j=1}^{N} \alpha_{ij} h_j^{enc} \tag{6}$$

where α_{ij} represents the degree of influence of intersection j on intersection i, which is obtained by:

$$\alpha_{ij} = softmax(e_{ij}) = \frac{exp(e_{ij})}{\sum_{k=1}^{N} exp(e_{ik})} \tag{7}$$

$$e_{ij} = score(h_{i-1}^{dec}, h_j^{enc}) = W_s[h_{i-1}^{dec}; h_j^{enc}] + b_s \tag{8}$$

where $score$ function is simple linear transformation and W_s and b_s are weight and bias parameters. $score$ function measures how much information needed to predict the right action for intersection i based on previous hidden state h_{i-1}^{dec} and the information of intersection j. The parameters of $score$ are jointly trained with all the other components of the proposed method.

C. Training algorithm

We use PPO-Clip to update the actor and critic part [13]. Actor's loss function $\mathcal{L}_{actor}(\theta)$ is defined as:

$$\begin{aligned} \mathcal{L}_{actor}(\theta) = & \frac{1}{T} \sum_{t=1}^{T} \min \left(pr_t(\theta) A\left(\boldsymbol{s}_t, \boldsymbol{a}_t\right) , \right. \\ & \left. \operatorname{clip}\left(pr_t(\theta), 1-\epsilon, 1+\epsilon\right) A\left(\boldsymbol{s}_t, \boldsymbol{a}_t\right)\right) \end{aligned} \tag{9}$$

where $pr_t(\theta)$ denote the probability ratio $pr_t(\theta) = \frac{\pi_\theta(\boldsymbol{a}_t|\boldsymbol{s}_t)}{\pi_{\theta_{\text{old}}}(\boldsymbol{a}_t|\boldsymbol{s}_t)}$. $\pi\left(\boldsymbol{a}_t \mid \boldsymbol{s}_t\right)$ is the output of the decoder at time-step i. We adopt the Generalized Advantage Estimator [20] to estimate the discounted advantage function $A\left(\boldsymbol{s}_t, \boldsymbol{a}_t\right)$, which is calculated as following:

$$\begin{aligned} \delta_t &= r_t + \gamma V(s_{t+1}) - V(s_t) \\ A\left(\boldsymbol{s}_t, \boldsymbol{a}_t\right) &= \textstyle\sum_{t'=t}^{\infty} (\gamma\lambda)^{t'-t} \delta_t \end{aligned} \tag{10}$$

where $V(\cdot)$ is a value function (critic); λ and γ are discounted factors which adjust bias-variance tradeoff, where $0 \leq \lambda \leq 1$ and $0 \leq \gamma \leq 1$.

Critic's loss function $\mathcal{L}_{critic}(\phi)$ is defined as:

$$\mathcal{L}_{critic}(\phi) = \tfrac{1}{T} \textstyle\sum_{t=1}^{T} (G_t - V(\boldsymbol{s}_t))^2 \tag{11}$$

where G_t is the reward-to-go for the agent at i^{th} unrolling step in simulation time t. $G_t \doteq \sum_{t'=t}^{T} \gamma^{t'-t} r_t$.

After computing $\mathcal{L}_{actor}(\theta)$ and $\mathcal{L}_{critic}(\phi)$, the objective that PPO needs to maximize in each iteration is:

$$\mathcal{L}(\theta, \phi) = \mathcal{L}_{actor}(\theta) - c_1 \mathcal{L}_{critic}(\phi) + c_2 S[\pi_\theta](\boldsymbol{s}_t) \tag{12}$$

where c_1, c_2 are coefficients, and S denotes an entropy bonus [14].

TABLE I
HYPER-PARAMETERS FOR OUR METHODS

Model Parameter	Value	Model Parameter	Value
Lerning rate	1×10^{-3}	Coefficient c_1	1×10^{-5}
Episodes	436	Coefficient c_2	1×10^{-3}
SGD iteration	10	Discounted factors λ	1
Memory length	1000	Disconted factors γ	0.99

IV. EXPERIMENT

In this subsection, we perform exhaustive experiments using open source traffic simulators: Flow[1] and SUMO[2]. Flow is a traffic control benchmarking framework. It provides a set of traffic control tools for designing custom traffic schemes, and offers integration with DRL algorithm. SUMO is an open source traffic system simulation software that enables microscopic control of traffic flow.

A. Settings

The simulation environment is a 6×6 grid of 36 intersections (traffic signals). Each intersection is connected by four 300 m long lanes, and each lane is bi-directional. The traffic volume of each lane is set to obey Poisson distribution. The lane speed limit is 15 m/s. When there is no traffic conflict, vehicles are always allowed to turn right. The traffic signal at each intersection contains two phases: East-West and North-South. The phase is followed by a 3-second interval of yellow lights and 2-second all-red time to ensure the safety [2]. The time slot of simulation is 5 seconds. The critic is a fully connected network with two layers, and each layer has 128 units. The actor consists of encoder and decoder, which are GRU models with one layer and each layer has 128 units. The hyper-parameters are shown in Table I.

B. Evaluation Metric

We use the following metrics to evaluate the performance of different control schemes:

- Delay: Intersection delay is an evaluation metric that reflects the time lost by vehicles being blocked and traveling on an intersection. It specifically refers to the average queuing time of all vehicles in the queue while waiting to cross the intersection [15].
- Queue Length: The average queue length is the average length of the queue waiting to cross the intersection after the stop line at all intersections. It is important for evaluating the performance of TLC and measuring the severity of traffic congestion [16].
- Average speed: Average speed vehicles spent on approaching lanes(in seconds). A greater average speed means that vehicles can spend less time crossing the intersection [17].

C. Compared Methods

We compare our method with traditional traffic signal control method and an DRL-based traffic signal control

[1]https://flow-project.github.io/
[2]https://www.eclipse.org/sumo/

method. Each control method is carefully fine-tuned using the best hyperparameters for comparison experiments.

1) Traditional control method:
   - Fixed-time Control (FT). Fixed-time control method uses a pre-determined plan for cycle length and phase time [7]. The cycle length is set to 70 seconds, the yellow interval is set to 6 seconds, and the all-red interval is set to 2 seconds.
   - Actuated Control (AT). If the traffic signal detects continuous traffic flow, it will extend the current phase. After detecting a long enough interval between successive vehicles, it will switch to the next phase [18]. The minimum green is set to 8 seconds, the maximum green is set to 45 seconds, the yellow interval is set to 6 seconds, and the all-red interval is set to 2 seconds.
2) DRL-based methed: a centralized control approach where the agent is composed of actor and critic. The actor and critic are fully connected neural networks [19].

In addition to the baseline methods, we also consider a variation of our model without attention mechanism, and consider the effect of the input order of the intersection:

- No-Attention. The structure of this method is same as ours except that attention mechanism in agent's actor is removed.

D. Performance Comparison

First, we will verify whether the agent can learn effectively during the training process and whether the training process converge stably. We compare our method with DRL-method and our model without attention mechanism. Fig.2 presents the learning curves of the cumulative rewards in 800 flowrate. We can see that our method outperforms the other two models, both in terms of stability and the cumulative rewards. In addition, the No-Attention model has better training stability than the DRL model since the huge state and action space is greatly reduced by dividing into subspaces through sequence-to-sequence model. We note that our model converges faster and learns information about the surrounding intersections more effectively than the No-Attention model owing to its attention mechanism.

Table II, Table III and Table IV show our comparison of our model with the remaining four TLC methods in the 36-intersections. In particular, the DRL, No-Att and Ours models completed 436 episodes in the training phase. The trained model parameters were saved and tested with 1000 seconds of traffic flow data. The experimental results are analyzed below:

1) It can be seen that the FT method and AT method are less effective because the conventional TLC method cannot effectively observe the information of surrounding intersections and cannot make timely adjustments according to the traffic flow conditions.
2) Our model significantly outperforms the remaining four TLC methods in terms of average speed, delay and queue length. This is due to the fact that, firstly, our model divides the state and action space into subspaces, and searches policy in the smaller sub-space, which is much easier. Secondly, our model uses an attention mechanism to coordinate other signals based on the impact of the surrounding intersections on the target intersection.

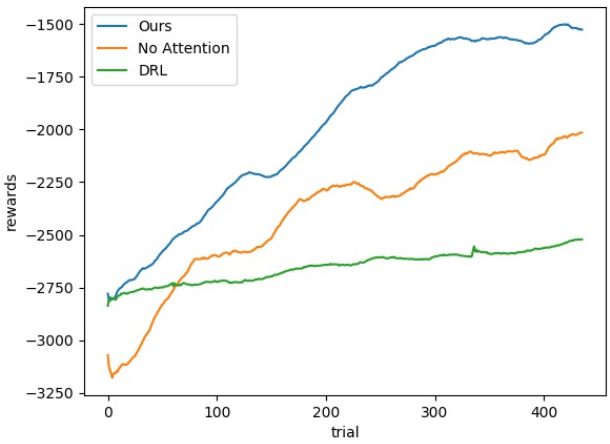

Fig. 2. Learning curves of ours model and other two RL-based methods

TABLE II

PERFORMANCE COMPARISON (THE TRAFFIC FLOW IN EACH LANE FOLLOWS A POISSON DISTRIBUTION AND THE λ IS 50)

Model Name	Average Speed	Delay	Queue
Ours	20.116±0.074	0.425±0.002	3.436±0.083
No-Att	20.069±0.091	0.426±0.002	3.501±0.153
DRL	14.628±0.069	0.581±0.002	4.645±0.196
FT	14.481±0.077	0.586±0.002	5.019±0.216
AT	14.666±0.130	0.580±0.003	4.522±0.144

TABLE III

PERFORMANCE COMPARISON (THE TRAFFIC FLOW IN EACH LANE FOLLOWS A POISSON DISTRIBUTION AND THE λ IS 100)

Model Name	Average Speed	Delay	Queue
Ours	18.137±0.191	0.481±0.005	7.335±0.151
No-Att	17.727±0.292	0.493±0.008	7.880±0.355
DRL	13.483±0.112	0.614±0.003	10.137±0.317
FT	13.515±0.073	0.613±0.002	9.994±0.159
AT	13.584±0.026	0.611±0.001	10.121±0.142

TABLE IV

PERFORMANCE COMPARISON (THE TRAFFIC FLOW IN EACH LANE FOLLOWS A POISSON DISTRIBUTION AND THE λ IS 150)

Model Name	Average Speed	Delay	Queue
Ours	14.676±0.243	0.580±0.006	13.213±0.375
No-Att	14.448±0.394	0.587±0.011	13.815±0.686
DRL	12.412±0.084	0.645±0.002	16.484±0.240
FT	12.346±0.077	0.647±0.002	16.513±0.186
AT	12.348±0.055	0.647±0.001	16.674±0.216

E. Attention Study

To better verify the effectiveness of the attention mechanism, we analyze the spatial distribution of the attention score generated during model training.

We take the 200^{th} episode of model training as an example, and output the distribution of attention within the traffic network at this time.

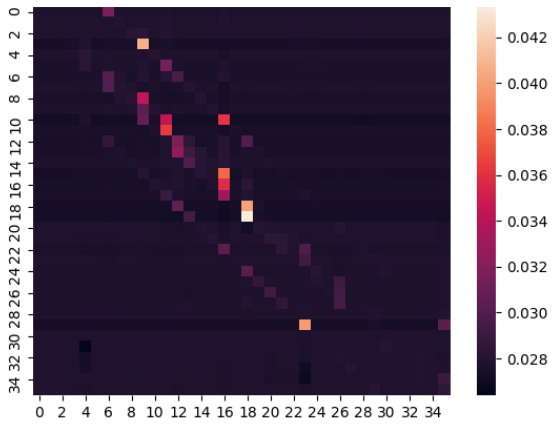

Fig. 3. Spatial difference of attention distribution in a 6-by-6 grid network. The horizontal coordinate is the id of the source intersection and the vertical coordinate is the id of the target intersection. The value in the image is the corresponding attention score of both. The lighter the color of the pixel block, the greater the influence of the target intersection on the source intersection.

From the Fig.3, it can be seen that only about 4 blocks of pixels in each column of the heatmap are lighter in color. That is, each intersection is mainly influenced by the surrounding 4 intersections. The rest of the intersections have less impact on the target intersection. This verifies the effectiveness of the attention mechanism sideways.

V. CONCLUSIONS

In this paper, we propose a novel method to control the traffic grid with single-agent. The agent is composed of a sequence-to-sequence model with attention to deal with the curse of dimensitonality. We divide the entire state space by intersections, inputs them sequentially into the encoder and memorizes the states. The decoder outputs the actions one by one for all intersections. In this way, the state space is compressed and the action space is reduced at each step. To better cooperate traffic signals, we apply attention mechanism to make the agent pay more attention on the states of surrounding intersections. We implemented exhaustive experiments in a 6×6 network. This is the first time to achieve a more satisfactory control effect in a larger road network by the centralized control with single-agent. Moreover, the effectiveness of the attention mechanism is verified by visualizing the distribution of the attention score of the road network.

We also acknowledge the limitations of our current approach and would like to point out several important future directions to make the method more applicable to real world. First, we will conduct experiment in a larger and more complex road network and use real world data for comparison and validation. Secondly, we will use more flexible attention mechanism to speed up the training. Finally, we will use more complex model to extract spatial and temporal features of the traffic grid.

REFERENCES

[1] Hua Wei et al. "Intellilight: A reinforcement learning approach for intelligent traffic light control". In:*Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2018, pp. 2496–2505.

[2] Hua Wei et al. "Colight: Learning network-level cooperation for traffic signal control". In: *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 2019, pp. 1913–1922.

[3] Ammar Haydari and Yasin Yilmaz. *"Deep reinforcement learning for intelligent transportation systems: A survey"*. In: IEEE Transactions on Intelligent Transportation Systems (2020).

[4] Alan J Miller. "Settings for fixed-cycle traffic signals". In: *Journal of the Operational Research Society* 14.4 (1963), pp. 373–386.

[5] Fo Vo Webster. *Traffic signal settings*. Tech. rep. 1958.

[6] Baher Abdulhai, Rob Pringle, and Grigoris J Karakoulas. "Reinforcement learning for true adaptive traffic signal control". In: *Journal of Transportation Engineering* 129.3 (2003), pp. 278–285.

[7] Samah El-Tantawy, Baher Abdulhai, and Hossam Abdelgawad. "Multiagent reinforcement learning for integrated network of adaptive traffic signal controllers (MARLIN-ATSC): methodology and large-scale application on downtown Toronto". In: *IEEE Transactions on Intelligent Transportation Systems* 14.3 (2013), pp. 1140–1150.

[8] Tianshu Chu et al. "Multi-agent deep reinforcement learning for large-scale traffic signal control". In: *IEEE Transactions on Intelligent Transportation Systems* 21.3 (2019), pp. 1086–1095.

[9] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate". In: *arXiv preprint arXiv:1409.0473* (2014).

[10] Juergen Schmidhuber. "Reinforcement Learning Upside Down: Don't Predict Rewards–Just Map Them to Actions". In: *arXiv preprint arXiv:*1912.02875 (2019).

[11] Hua Wei et al. "Presslight: Learning max pressure control to coordinate traffic signals in arterial network". In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2019, pp. 1290–1298.

[12] Chia-Cheng Yen et al. "A Deep On-Policy Learning Agent for Traffic Signal Control of Multiple Intersections". In: *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*. IEEE. 2020, pp. 1–6.

[13] John Schulman et al. "Proximal policy optimization algorithms". In: *arXiv preprint arXiv*:1707.06347 (2017).

[14] Ronald J Williams. "Simple statistical gradientfollowing algorithms for connectionist reinforcement learning". In: textslMachine learning 8.3-4 (1992), pp. 229–256.

[15] Jian Wu et al. "Delay-based traffic signal control for throughput optimality and fairness at an isolated intersection". In: *IEEE Transactions on Vehicular Technology* 67.2 (2017), pp. 896–909.

[16] Xiaoyuan Liang et al. "A deep reinforcement learning network for traffic light cycle control". In: *IEEE Transactions on Vehicular Technology* 68.2 (2019), pp. 1243–1253.

[17] Wade Genders and Saiedeh Razavi. "Using a deep reinforcement learning agent for traffic signal control". In: *arXiv preprint arXiv*:1611.01142 (2016).

[18] Rahmi Akcelik. *Traffic signals: capacity and timing analysis*. 1981.

[19] Elise Van der Pol and Frans A Oliehoek. "Coordinated deep reinforcement learners for traffic light control". In: *Proceedings of Learning, Inference and Control of Multi-Agent Systems (at NIPS 2016)* (2016).

[20] John Schulman et al. "High-Dimensional Continuous Control Using Generalized Advantage Estimation". In: *arXiv:1506.02438 [cs.LG]* (2015).