# Urban traffic signal control based on Reinforcement Learning

Zhenhao Zhou

Machine Learning (CS 419/519)


When I was young , my dad was traffic police who in charge of traffic lights. So I was always wondering how can he set the interval time for traffic lights to make sure people do not need waste time in road. The interval time of traffic lights is fixed, so it is only suitable for a period of time. Lot of times, you need to wait for green light, even through there is no car in the other direction of the crossroads.

Traffic jam is a big issue in urban area. With the rapid development of the economy, urban traffic congestion has become increasingly serious. Especially in some big cities, traffic jams and traffic accidents are commonplace. Traffic congestion not only affects the normal operation of the city, but also reduces people's daily work. Efficiency and quality of life. The traditional means of solving traffic congestion problems is to increase infrastructure investment, expansion and new

roads. However, the space available for roads in cities is becoming more and more limited. Deep Reinforcement Learning for Traffic Light Control in Vehicular Networks[1], this essay give a really good solution to efficient control urban traffic lights signal.

Using deep Q-learning to built intelligent traffic lights system is really good idea. Every “Agent” which can modify the interval to maximize traffic efficiency based on changes in the environment dynamic and autonomous.

The vehicle detector transmits the detected road information to the agent, and the adjacent agent also provides the information with its own road information; the agent decides action based on the received information and relevant empirical knowledge. After a certain time interval(like 5s). The vehicle detector transmits the intersection information to agent again, and calculates a Reward value feedback to the agent, and the agent corrects the Q-value according to the reward, then Make decisions based on traffic status again. The flow diagram as shown in Figure 1.

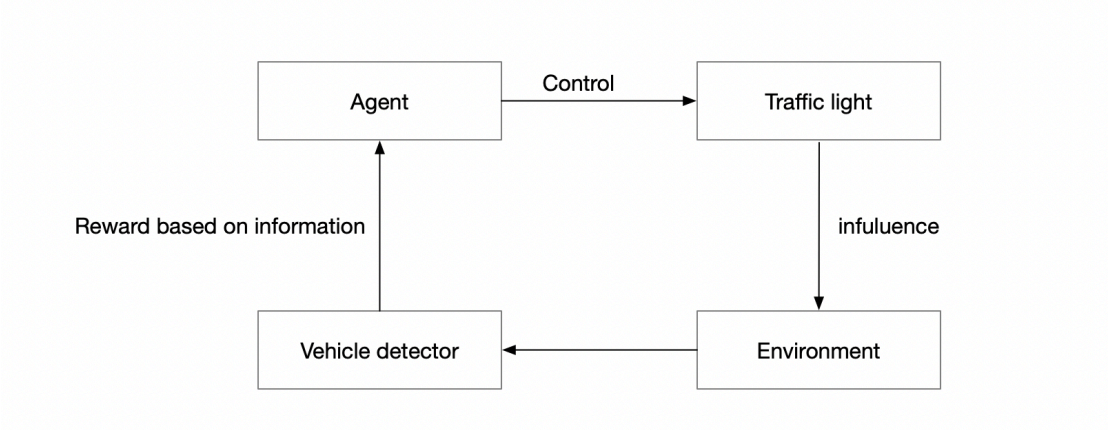



**Figure 1 the flow chart for intelligent traffic system**

< position, speed > : vehicles' position and speed can be obtained [2]. Then the traffic light can extract a virtual snapshot image of the current intersection. The whole intersection is divided into same-size small square-shape grids. The length of grids, c, should guarantee that no two vehicles can be held in the same grid and one entire vehicle can be put into a grid to reduce computation. The position dimension is a binary value, which denotes whether there is a vehicle in the grid. If there is a vehicle in a grid, the value in the grid is 1; otherwise, it is 0. The speed dimension is an integer value, denoting the vehicle's current speed in m/s.

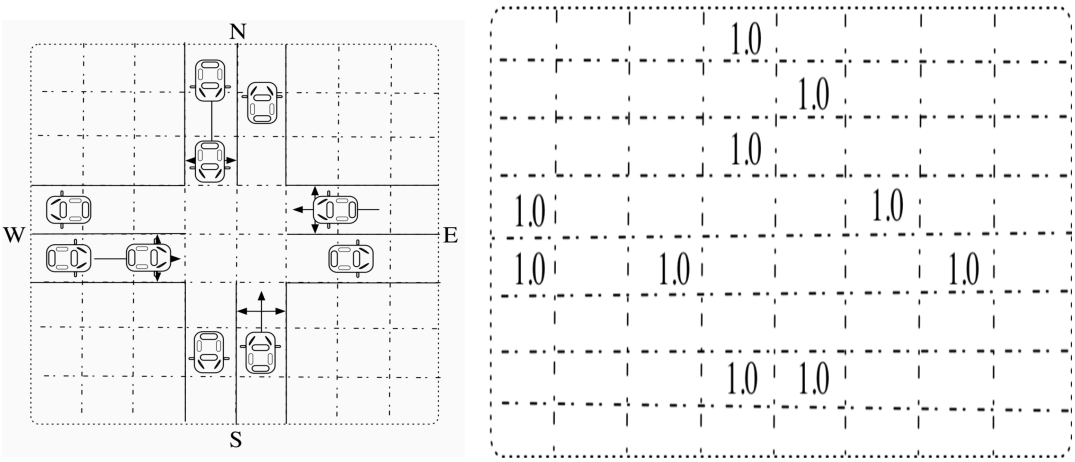



**Figure 2: the modeling of state**

There are four phases, north-south green, north-turningRight&South-turningLeft green, east-west green, and east-turningLeft&west-turningRight green. So use a four-tuple < t1 , t2 , t3 , t4 > denote the duration of the four phases. Each time we can add 5 second to keep this phrase, or delete 5 second to decrease this pares duration. So it has 8 actions.

Action 1: $t_1$+5.

Action 2: $t_1$-5.

Action 3: $t_2$-5.

Action 4: $t_2$+5.

Action 5: $t_3$+5.

Action 6: $t_3$-5.

Action 7: $t_4$-5.

Action 8: $t_4$-5.

Action 9: no change

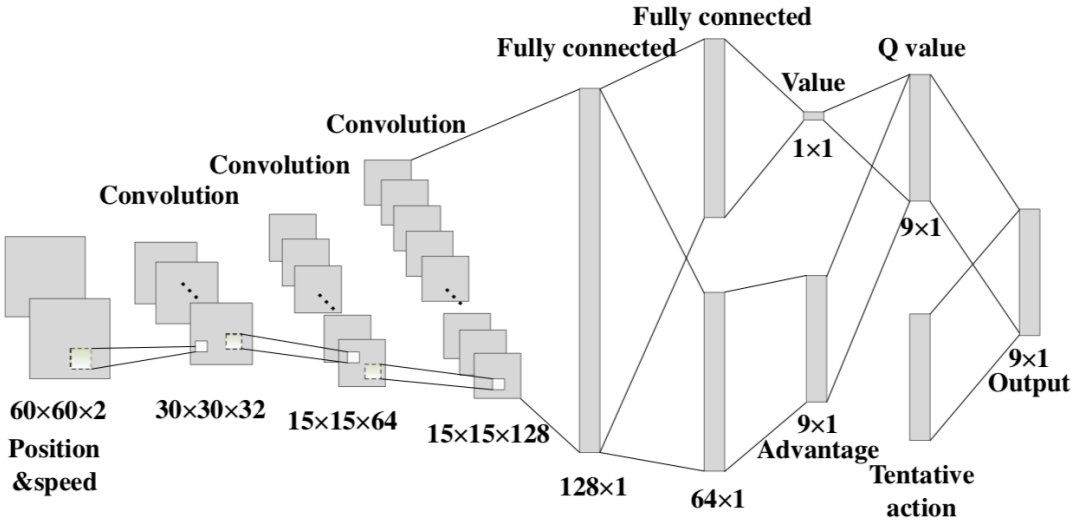
The traffic light can select an action in a similar way as the previous procedure. In addition, we set the max legal duration of a phase as 60 seconds and the minimal as 0 second.

In reinforcement learning, the reward is really important. It will teach program whether it is good or bad, so the program can update based on rewards. Thus it is really important for this algorithm. It will directly affect the performance of this system. The program aim is improve the efficiency of intersection. So we use the sum of waiting time to measure whether it is efficient. Thus , calculate the change of sum of waiting time between before take this action with after take this action.

The reward is equal to the increment in cumulative waiting time(sum of all vehicles'waiting time) between before taking the action and after the action.

In this program, it is not based on normal Q-learning. Deep Q-learning is applied ,which stores the data obtained by the system exploration environment, and then randomly samples the parameters to update the parameters of the neural network.

There are two main methods to store Q-value: one is Q-value table method, which is mainly applicable to the simple case with limited state; the other is the neural network storage method, which is applicable to the very many state, because the neural network has strong storage capacity, and can use the generalization ability of the neural network to extrapolate the state that has not been learned. For the traffic signal control system, because there are too many state in traffic environment, neural networks is best way to store Q-values. So we will change Q(s,a) to Q(s,a; θ ) so that we can generate Q-value by neural network, and you also do not need afraid of too many state.




Dueling Networt[3] is also applied to built network. Q(s,a; θ)=V(s; θ)+A(s,a; θ) Dueling network make Q(s,a; θ) divide to V(s) which means the overall expected rewards by taking probabilistic actions in the future steps and A(s,a; θ) which means how important an action is to the value function among all actions. The dueling architecture is shown to effectively improve the performance in reinforcement

learning. I think the reasons is make agent can learn more useful information and ignore useless information.

Deep Q-learning have a advantage that it can learn the previous experience, because Q learning is an off-policy learning method, it can learn the current experience, and also learn the experience of the past, even learning others. So random experience before enter the target network will make the neural network more efficient. Choose experience which stored in the playback memory network, when training, randomly take out some (minibatch) to train can lead to faster learning and to better final policy The motivation for Experience Replay is: 1. Deep neural network as a supervised learning model, requiring data to satisfy independent and identical distribution, 2. the sample obtained by Q Learning algorithm is related before. In order to break the correlation between data, the Experience Replay method breaks this association by means of storage-sampling.

Then we input the randomly select tuple to target network which can provide stable update in each iteration, same architecture as the primary neural network but different parameters, is usually employed to generate the target value. Using the Mean Square Error (MSE) to be loss function to update network.

But deep Q-learning have the problem that the max operation causes the estimated value function to be larger than the true value of the value function. If the value of each point of the value function is overestimated by the same magnitude, the action corresponding to the largest value function is found, then the optimal strategy is Stay the same. But in practice, the overestimation is not uniform, so overestimation of the

value function will affect the final policy decision, resulting in the final strategy is not optimal. How we can solve this problem?

So we use Double Q-learning algorithm[4] to solve this problem. Basically Double Q-learning algorithm's architecture is same with deep Q-learning algorithm which have two network. But Double Q-learning is no longer directly input the maximum Q value in the target Q network, but find the action corresponding to the maximum Q value in the current Q network, and then use this action to update in target network. The algorithm also employ the Elision-greedy algorithm to balance the exploration and exploitation in choosing actions same as we learned in class.

In the real life, Deep learning often requires a lot of time and machine resources to train, which is also a major reason for the development of deep learning algorithms. So we need optimization method to reduce time and make model coverage fast. In this program ,they choose "Adam" （adaptive moment estimation） optimization algorithm[5]. Adam is a optimization algorithm that can replace the traditional stochastic gradient descent process. It can iteratively update the neural network weights based on the training data. The Adam algorithm is different from the traditional random gradient drop. The stochastic gradient descent maintains a single learning rate (ie, alpha) to update all weights, and the learning rate does not change during the training process. Adam calculates independent adaptive learning rate for different parameters by calculating the first moment estimation and second moment estimation of the gradient.

To summaries, the process is like input the current state, then go through the neural network to choose the action which can get most rewarding. Then current state and action along with the next state and received reward are stored into the memory as a four-tuple (s, a, r, $s'$) which will be selected randomly to be a set to update neural network's weighs. The tuples which are stored in memory will be randomly selected then input into target network to update weight. Using the double DQN and dueling DQN to reduce the possible overestimation and improve performance.

After building model, using "SUMO"[6] which is an open source, highly portable, microscopic and continuous traffic simulation package designed to handle large road networks to provides real-time traffic simulation. The model is trained in iteration , one iteration is one hours. The reward is accumulated in an episode. The goal in our network is to maximize the reward in the one-hour episode by modifying the traffic signals' time duration. The simulation results are the average values of the nearest 100 iterations.

In conclusion, I think it is good example which combine neural network with reinforcement learning. The reinforcement interacts with the environment to get rewards from actions. Its goal is to take the action to maximize the numerical rewards in the long run so that it can learn knowledge from environment and change the light depends on current environment, which will improve the efficiency of road. Neural network let program have ability that can process numerous state and also give the ability that model can learn the situation that model never meet before.

[1] Xiaoyuan Liang, Xusheng Du, Student Member, IEEE, Guiling Wang, Member, IEEE, and Zhu Han Fellow, IEEE, Deep Reinforcement Learning for Traffic Light Control in Vehicular Networks, March 2018

[2] W. Genders and S. Razavi, “Using a deep reinforcement learning agent for traffic signal control,” arXiv preprint arXiv:1611.01142, November 2016.

[3] Wang Z, de Freitas N, Lanctot M. Dueling network architectures for deep reinforcement learning[J]. arXiv preprint arXiv:1511.06581, 2015.

[4] Hado van Hasselt , Arthur Guez, and David Silver, Deep Reinforcement Learning with Double Q-Learning. Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)

[5] Diederik P.Kingma , Jimmy Ba, Adam: A Method for Stochastic Optimization . a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015

[6] Krajzewicz, Daniel und Hertkorn, Georg und Rössel, C. und Wagner, Peter (2002) , *SUMO (Simulation of Urban MObility) - an open-source traffic simulation.* the 4th Middle East Symposium on Simulation and Modelling (MESM20002), Seiten 183-187. 4th Middle East Symposium on Simulation and Modelling, 2002-09, Sharjah (United Arab Emirates). ISBN 90-77039-09-0.