Technical report of City Brain Challenge

Team D

ABSTRACT

In this report we show the details about our method in City Brain Challenge. We use Reinforcement Learning to learn how to control the traffic lights in a city-scale road network. The key techniques in our methods are utilizing different features of State and designing the Reward. In addition, we also tune some hyper-parameters to get a better performance. When submitting the final solution, we synthesize many individual models and use ensemble strategy. Our method won the tenth place in City Brain Challenge in the final round.

CCS CONCEPTS

• **Information systems** → **Spatial-temporal systems**; • **Computing methodologies** → Neural networks.

KEYWORDS

City Brain Challenge, DQN

ACM Reference Format:
Team D. 2021. Technical report of City Brain Challenge. In *City Brain Challenge: How many vehicles can you serve at most with a city-scale road network, April 01–July 05, 2021.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Traffic congestion is becoming a more and more critical issue in cities. It costs tremendous amount of money in big cities every year to improve traffic construction. Reducing traffic congestion could improve city efficiency, save energy and ease people's daily life.

Smart coordinated traffic signals can be an effective way to reduce the traffic congest of city. Traditional traffic lights are controlled with pre-defined fix-time plan and cannot be dynamically adjusted according to the real traffic flow.

To make the traffic lights more intelligent, reinforcement learning technique, such as Deep Q-learning[3], have been applied and recent studies have shown inspiring result. In this RL framework, Environment is composed of traffic light phase and traffic condition. Agent is the traffic lights which can change phase and control the traffic movements. State is a feature representation of the environment. Some related works vary in state representations including handcraft features, such as queue length, average delay. Agent takes state as input and learns a model to predict whether to "keep the current phase" or "change another phase". The decision is sent to the environment and the reward (e.g., how many vehicles pass the intersection) is sent back to the agent. The agent consequently updates the model and further makes the new decision for the next timestamp based on the new state and the updated model.

One key issue of RL algorithm is how to design a reward function. Related studies have different reward designs, including average delay, the average travel time, and queue length. The work of IntelliLight[6] defined the reward as a weighted sum of six factors, including queue length, delay over all approaching lanes, updated waiting time, indicator of light, total number of vehicle and total travel time of vehicles that pass the intersection. PressLight[5] proposed a reward design inspired by max pressure(MP) theory in the transportation field, which can be proved to be maximizing the throughput of the traffic network.

In this competition, we are provided a city-scale road network and its traffic demand derived from real traffic data. We aim to coordinate the traffic signals to maximize number of vehicles served while maintaining an acceptable delay.

Traffic signals coordinate the traffic movements at the intersection and a intelligent traffic signal coordination algorithm is the key to transportation efficiency. For a four-leg intersection, 1 of 9 types of signal phases can be selected at each period of time step, serving a pair of non-conflict along with all right turning traffic movements. The action is defined as the traffic signal phase for each intersection to be selected at next 10 seconds. If an agent is switched to a different phase, there will be a 5 seconds period of 'all red' at the beginning of the next phase, which means all vehicles could not pass this intersection.

Our proposed solution is based on Deep-Q learning algorithm. In this task, given the current state of the traffic condition, the agent is supposed to select an action(which phase to change or keep the current phase) that can make the reward maximum in the long run, following the Bellman Equation(Equation 1). In this case, the action value function Q for time step t is the summation of the reward of the next time step t+1 and the maximum potential future reward. Therefore, the action can find action that are beneficial to future reward.

$$Q(s_t, a, t) = r_{a,t+1} + \gamma max Q(s_{a,t+1}, a_{t+1}, t+1) \tag{1}$$

We use some trick when training our DQN, e.g. double DQN[2], dueling DQN[4], prioritized experience replay buffer[1]. We design two different reward in qualification round and final round. We use ensemble strategy to improve our model performance. Our work mainly has the following highlights:

More Feature Engineering: Based on the original features provided, we do further feature engineering. We consider the number of vehicles in each approaching lane that can cross the intersection in a certain period of time. We suppose these vehicles accelerate and will not exceed the speed limit. Besides, we consider these vehicles that are blocked in the exit lane, because it may make the vehicles in some approaching lane cannot pass the intersection. We consider the current phase because there is a penalty for switching the phase.

KDD CUP 2021, April 01–July 05, 2021,
© 2021 Association for Computing Machinery.
ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

Reward Design: Our reward function is in the form of the difference between two step. In the final round, we use the total number of vehicles that pass the intersection during time interval and reduction of the queue length after a light phase switch.

Ensemble strategy: We use different methods and parameters to train several DQNs, and use their mean value as the final strategy to select action. This method improves greatly than using a single DQN.

The rest of this report is organized as follows. Section 2 discusses the method and training details in qualification round. Section 3 discusses the method and training details in the final round. Finally, we conclude this report in Section 4.

2 QUALIFICATION ROUND METHOD

This section we show the details in Qualification Round. The model we use is DQN, and this model is used for every agent. For each agent, the "State" is its corresponding observation and we design "Reward" to evaluate its action.

2.1 Reward

Since we have access to the information about vehicle route and travel time at speed limit, we can calculate the "delay index" of every vehicle directly. So it's intuitive to use the delay index to generate the reward. In each timestamp, we calculate the average delay index of all vehicles on approaching lanes for an agent, i.e. from lane 1 to lane 12. The average delay index in timestamp t is denoted as D_t and the reward after we choose the action for agent in t is

$$r^i = -(D^i_{t+1} - D^i_t) \tag{2}$$

The r^i only reflects the impact of individual action. To consider the whole situation and we also calculate the global reward

$$r^g = -(D^g_{t+1} - D^g_t) \tag{3}$$

where D^g_t is the average delay index of all vehicles on approaching lanes of all agents in t. The final reward for each agent is

$$r = r^i + \alpha * r^g \tag{4}$$

2.2 State

To give the agent more information about the environment, the state is designed with seven traffic features information. The traffic features are as follows:

1) average speed of the lanes: this feature is a 24-dims vector and each dimension corresponds to the average speed of the corresponding lane.
2) vehicle number of the lanes: this feature is a 24-dims vector and each dimension corresponds to the vehicle number of the corresponding lane.
3) sum of reciprocal of travel time estimated with free-flow speed (i.e. $1/TT^f_i$): this feature is a 24-dims vector and each dimension corresponds to the sum of $1/TT^f_i$ of the corresponding lane.
4) current action: one-hot vector and the dimension of 1 means current action.
5) sum of delay index of the lanes: this feature is a 24-dims vector and each dimension corresponds to the sum of delay index of the corresponding lane.
6) sum of delay index of the potential vehicles which can cross the intersection of the lanes: this feature is a 12-dims vector. We consider each vehicle can cross the intersection when the length of its route is greater than 1 and $speed\_limit \times 5$ is greater than the distance to intersection. We calculate the sum of delay index of these vehicles and only consider 12 entry lanes. So this feature is a 12-dims vector.
7) number of the potential vehicles which can cross the intersection of the lanes: this feature is a 24-dims vector and each dimension corresponds to the vehicle number that can cross the intersection of the corresponding lane.

Each type of feature is normalized and then we concat these features to form the final state. The method of normalization is dividing by the sum of each type of feature.

2.3 Training Details

This section we show the training details. The architecture of DQN model is 4 fully-connection layers. The hidden sizes are 128, 128, 64, 8. Except the final layer which has no activation function, other layers use "relu". To have a stable training process, we use the target network with the same architecture and update the target network every 20 training iterations.

We use Prioritized Experience Replay buffer, the size of buffer is 2000 and the batch size is 512. The discount rate is 0.4, the initial ϵ in epsilon-greedy exploration is 0.1 and the minimum ϵ is 0.01. The decay rate of ϵ is 0.995. We use SGD to optimize the model and the learning rate is 0.1. We found that SGD is more stable than Adam. The number of training episode is 200.

2.4 Ensemble

In order to further improve the performance, we train many models and use these models to do ensemble to get a better policy. In order to have more diversity, we set different hype-parameters, such as $\alpha = 0$ or $\alpha = 1$, and remove or add one fully-connection layer. After obtaining many models, we add all outputs of these models as the q values and choose the action which has the maximum q value. We execute action every 10 seconds.

2.5 Rules

Even we have a better performance via ensemble, the policy still has some weaknesses. So we make several rules to make up for the deficiencies of the policy.

First is action mask. The mask is only generated on a three-leg intersection. For example, if an intersection has no roads in West, as showed in Fig 1, there is no need to execute action 4 and 8, since there will no vehicle can entry or drive from West road. The action 6 and 7 can also be removed because action 1,2,3,5 have already contained all possible lane change. So when choose the action for an agent, we first remove all useless actions.

Second, we judge whether it is worth to do an action. By calculating the number of the potential vehicles which can cross the intersection of the lanes, if all entry lanes have no vehicle can cross the intersection, then we keep the same action since any actions

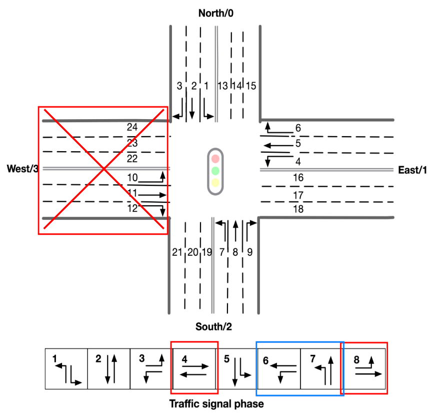

Figure 1: Example of useless action of a three-leg intersection(no West road). The action in red box can be removed completely and the action in blue box can be replaced by other actions.

will make traffic condition better for this agent. If this condition is not satisfied, we remove the action that its corresponding lanes have no vehicles can cross the intersection. The reason is the same as above.

Finally we consider whether it's worth to execute the action with maximum q value from the rest actions to change. For example, if the current action is action 4 and the policy choose action 8 to change. If the lane 10 has no vehicles can cross the intersection, we still execute the action 4 and we regard it as a useless action switch.

In every step, for each agent we choose action according to the policy and above rules. The rules have advantages and these rules form our final method in qualification round. However, these rules have no promote in final round. Maybe the reason is the policy learned in final round is more robust.

Through Ensemble and Rules, we obtained $13th$ place in the Qualification Round.

3 FINAL ROUND METHOD

In this section, the detail of the method in the final round is introduced. The change in this round is that different traffic data between training and scoring submissions and travel time at free-flow speed for every vehicle is unknown, which requires the model is robust. In this round, the model we use is still DQN but different state and reward because of the change.

3.1 Traffic Data

In this round, traffic data for scoring submissions is unknown. It is very important to generate sample traffic data for training. We generate different traffic data using different traffic settings for training. We train DQN models in the different traffic data. However, it is hard to confirm which traffic data is better with each other. Improvement on a specified traffic data may has low score when submitting.

At the same time, we only use the rules which is validated to be very useful in the qualification round and test in the traffic data given by the official in this round. In the rule, we consider the greedy method that as many vehicles as possible pass through the intersection after the agent takes phase.If an agent is switched to a different phase, there will be a 5 seconds period of 'all red' at the beginning of the next phase, which means all vehicles could not pass this intersection. We consider this in our greedy methods. We calculate the number of outgoing vehicles in the lanes every phase corresponds to for every phase. Especially, the number multiply 0.6 if phase switch is needed when taking this phase. The phase corresponding to the maximum is chosen for every agent. We consider the waiting length if phases corresponding to the maximum are more than one. We found the submitting score is always higher when the score in the traffic data given by the official in this round is higher using only rules. Therefore, we decide to use traffic data given by the official as our training data.

3.2 Reward

In this round, the "delay index" of every vehicle can't be calculated because of unknown route. The reward we used in qualification round can't be used in this round directly. Based on the knowledge of qualification round, we expect to design new reward function approximating the reward function used in the qualification phase at first. The travel time at free-flow speed and actual travel time a vehicle has traveled can be calculated in this round. We try several reward functions that use the travel time at free-flow speed and actual travel time a vehicle has traveled. However, models which use those reward functions are not robust because of the approximation, which has a high score in the training data but low score when submitting.

Therefore, we consider reward that is not directly related to travel time, such as outgoing vehicle number from incoming lanes, waiting length and so on. We think the most import factor for traffic is that as many vehicles as possible pass through the intersection at a traffic light switch. Secondly, the total delay index will be lower if lanes with longer waiting queue are passable. The longer the waiting length, the more vehicles will move on. Moreover, the average speed and the number of vehicles which are not in the waiting queue in the lane can also be considered. Therefore, we use the reward function in this round is

$$r_i = o_i + 0.05 * q_i \tag{5}$$

which o_i is the outgoing number of vehicles all incoming lanes in the i_{th} intersection, and q_i is the length of waiting queue in the lanes the phase agent take corresponds to. A vehicle is considered to wait in the queue if the speed of the vehicle is less than 0.5 and distance from starting intersection is longer than 1 meter. It is clear from this formula that the number of outgoing vehicles is more import than waiting length when the agent takes phase. 0.05 is the hype-parameter for the reward. We try different hype-parameters include 0.03,0.05 and 0.1. We found 0.1 is the worst result, which could be caused by the frequent switching between phases.

3.3 State

Features for state is not normalized compared with the qualification round. Because the most features used in this round is in the same scale. The dimension of features is 48. We mainly consider the factors may be related to reward, including the number of outgoing vehicles, waiting length, number of waiting vehicle in the outgoing lanes and current phase in the intersection. We don't punishment phase switch like using rule. Because we think the model can learn this from those feature. The following is the details about every feature.

1) Number of outgoing vehicles. This feature is a 12-dims vector about the 12 incoming lanes for the intersection and each dimension corresponds to the number of vehicles that can leave the incoming lane if the incoming lane is passable. A vehicle can leave the lane if remaining time is less than phase interval time. Remaining time is calculated by acceleration calculation formula in physics.
2) Waiting length. Waiting length refers to number of vehicles that waiting in the incoming lanes. This feature is a 12-dims vector about the 12 incoming lanes for the intersection and each dimension corresponds to the number of vehicles waiting in the incoming lane. A vehicle is waiting in the incoming lane if the speed of this vehicle is less than 0.5 and distance from starting intersection is longer than 1 meter. Especially, we consider a vehicle is waiting if remaining time for this vehicle is less than phase interval time, which ensure outgoing vehicles must be in the waiting queue.
3) Number of waiting vehicles in the outgoing lanes: this feature is a 12-dims vector about the 12 outgoing lanes for the intersection and each dimension corresponds to the number of vehicles waiting in the tail of the outgoing lane. A vehicle is at the tail of the outgoing lane may obstruct the vehicle in the incoming lane coming in this outgoing lane. A vehicle is waiting if the speed of this vehicle is less than 0.5 and distance from starting intersection is longer than 1 meter. We only consider maximum number of vehicles that can come in this outgoing lane is at the tail of this outgoing lane.
4) Current phase: this feature is a 12-dims one-hot vector about the 12 incoming lanes for the intersection and each dimension corresponds to current phase the agent takes. All right turning lanes is one because all right turning lanes are passable all the time.

Each type of feature is concatenated to form the final state.

3.4 Training Details

This section we show the training details. In this round, we use an open-source library RLlib to train DQN model. The traffic data we used is given by the official in this round. The input for the model is 48-dims features. The out is a 9-dims data, representing 9 different phase. The architecture of DQN model is the default setting, 3 fully-connection layers. The hidden sizes are 256, 256, 9. The exploration type we use is epsilon greedy with initial epsilon 0.1, final epsilon 0.02 and epsilon time step 1000. The discount rate γ is 0.5. The learning rate is 0.001. The number of training iterations is 20000. The detail of model parameter is shown in table 1.

3.5 Ensemble

In order to further improve the performance and robustness, models with different hyper parameters and architectures are trained.

Model parameter	Value
learning rate	0.001
hidden size	256,256
γ for future reward	0.5
initial ϵ for exploration	0.1
final ϵ for exploration	0.02
replay buffer size	50000
target network update frequency	500

Table 1: Details of model parameter

For the architecture, We use origin DQN and Dueling. For hype-parameters, such as γ and learning rate. We ensemble multiple models and expect to get better results. However, it is not true that the more models, the better results. Finally, we choose the combination with more diversity, which is more robust.

The ensemble model with the best score when submitting is made up of 6 models with different architectures and hype-parameters. The details of ensemble are shown in the table 2. Those models are evaluated every 20s on the traffic data given by the official in this round. The evaluation ends when total delay index is bigger than 1.40.

Model	Architecture	Number of vehicles	Delay index
1	DQN	48590	1.403846
2	DQN	48590	1.404268
3	DQN	48590	1.404330
4	DQN	48590	1.404296
5	Dueling	48590	1.403848
6	Dueling	48590	1.404877
Ensemble		49048	1.410244

Table 2: Details of ensemble in the final round

We found the rules we used in the qualification round don't work in this round. Because the model learns how to deal with situations the rules cares for well because of state and reward. The performance of the rules is not good as the model. But it is a baseline of our model and is used to validate the effect of different rewards.

4 CONCLUSION

In this competition, we use reinforcement learning approach to control the smart traffic light aiming to maximize number of vehicles served while maintaining an acceptable delay.

In qualification round, since we can get the information about vehicle route and travel time at speed limit, we use reduction of the average delay index of all vehicles on approaching lanes between two time step as our reward. Besides, we do a lot of feature engineering to represent the traffic condition. We also use ensemble strategy and get a better policy. Last but not least, we propose some effective rules to solve the bad case generated by models. Finally, our total served vehicle number is 126572 and delay is 1.511, we obtained 13th place in qualification round.

In final round, "delay index" cannot be calculated when training because of unknown route. We design a new reward function, the reward is defined as a weighted sum of the total number of vehicles pass the intersection during time interval and the length of waiting queue. We also do feature engineering and ensemble like in qualification round. And we conduct many experiments to search the best hyperparameter. Finally, our total served vehicle number is 316941 and delay is 1.402, we obtained 10th place in the final round.

REFERENCES

[1] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. 2016. Prioritized experience replay. In *Proceedings of the 4th International Conference on Learning Representations.* 322–355.

[2] Hado Van Hasselt, Arthur Guez, and David Silver. 2016. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 30.

[3] Mnih Volodymyr, Kavukcuoglu Koray, Silver David, Andrei A Rusu, Veness Joel, Marc G Bellemare, Graves Alex, Riedmiller Martin, Andreas K Fidjeland, and Ostrovski and Georg. 2019. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2019), 529–533.

[4] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Hasselt, Marc Lanctot, and Nando Freitas. 2016. Dueling network architectures for deep reinforcement learning. In *International conference on machine learning*. PMLR, 1995–2003.

[5] Hua Wei, Chacha Chen, Guanjie Zheng, Kan Wu, Kai Xu, Vikash Gayah, and Zhenhui Li. 2019. Presslight: Learning max pressure control for signalized intersections in arterial network. In *Int. Conf. Knowl. Discov. Data Min.(KDD)*. 1290–1298.

[6] Hua Wei, Guanjie Zheng, Huaxiu Yao, and Zhenhui Li. 2018. IntelliLight: A Reinforcement Learning Approach for Intelligent Traffic Light Control. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery Data Mining* (London, United Kingdom) *(KDD '18)*. Association for Computing Machinery, New York, NY, USA, 2496–2505. https://doi.org/10.1145/3219819.3220096