# Optimazation of Traffic Light Controlling with Reinforcement Learning

(Dated: November 24, 2017)

**Abstract:** Ensuring the efficiency of transportation systems is a priority for modern society. Technological advances enable transportation system to collect huge sets of varied data on an unprecedented scale. Recent studies in combining deep neural network architectures with reinforcement learning techniques have demonstrated promising potential results in solving complex control problems with high dimensional state and action spaces. Inspired by these studies, we applied the similar idea to build a neural network architecture implemented by the reinforcement learning, where a new reward function for the traffic light control problem is implemented. Through experiment in SUMO traffic simulator, we prove that this approach could improve the efficiency of transportation system compared to the traditional controlling policy.

## 1. INTRODUCTION

With regard to the increasing population, and subsequent vehicle ownership, demand of road infrastructure is greatly growing and is sometimes beyond its capacity, resulting congestion. To address traffic congestion problem, we proposed a technological solution to improve road condition at crossroad by intelligently controlling traffic signal.

Currently there are three types of traffic controlling system. The first is pre-time signal, where a fixed time is determined for all green phases according to historical traffic demand, without considering concurrent changes of traffic demand. The second is vehicle-actuated signal control, where information related to traffic demand is detected and provided by inductive loop detectors equipped in traffic intersections to help make decisions. The third is adaptive signal control, signal control policy is managed and updated automatically according to the current intersection condition[1]. In this study, we focus on adaptive signal control where Convolutional Neural Network and Reinforcement Learning are applied[2][3]. Our experiment is based on traffic environment generated by SUMO[4], an open source traffic simulator which provides control over automobile and traffic signal transition.

## 2. TASK DEFINITION

Task in this study is to develop a single-crossing traffic signal control system which improves the traffic efficiency at single intersection. The indicator of the traffic condition is relative speed and the total waiting time of all vehicles, which would be further elaborated in subsection *Transition*.

## 3. INFRASTRUCTURE

### 3.1. SUMO

To provide testing environment for our project, traffic simulator *Simulation of Urban MObility*(SUMO) is used as simulator, which provides following features and functionalities.

SUMO is an open source, highly portable, microscopic and continuous road traffic simulation package designed to handle large road networks. SUMO is licensed under the Eclipse Public License V2. Both command line interface and graphical user interface are provided for displaying traffic control system and traffic flows with individual vehicles. To benefit the simplicity of obtaining the learning outcomes and demonstration, a common intersection with traffic lights will be implemented on SUMO.

### 3.2. Realization of Online Simulation

While processing models, *TraCI*, a real time interface for SUMO, communicates the simulator and model. *TraCI* python API is used to retrieve parameters including the vehicle position, accumulating waiting time, and speed factor of individual agents, based on which the phase of traffic light can be set by *TraCI* back into the simulator according to the result given by the network learning.

### 3.3. Data Collection

Input route schedule of vehicles in our experiment is generated by simulator during training process, which requires no pre-data-collection process.

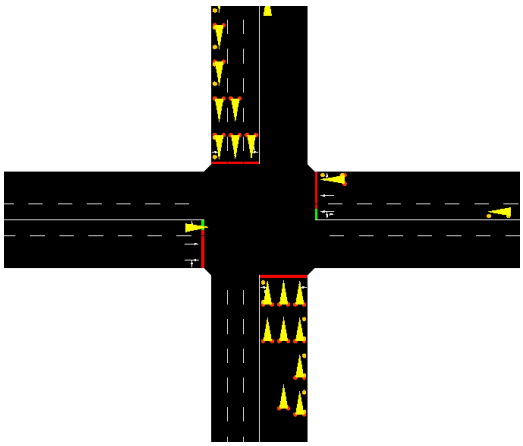

FIG. 1: Visualization of the model

## 4. REINFORCEMENT LEARNING FOR TRAFFIC LIGHT CONTROL

Reinforcement Learning framework is applied to the problem of selecting optimal light configurations for traffic intersections. The SUMO traffic simulator[5] is used to run experiments. For details on the experimental setup, see[6].

### 4.1. Model

We describe the traffic light control problem using the components of a Markov Decision Process, where an agent responds to an environmental state $s \in S$, takes one of the possible actions $a \in A$, with some transition probability ends up in state $s'$ and receive reward signal $r(s, a, s')$.

In the model, the traffic intersection is simplified into a simple crossroad as shown in the FIG.1.The model is simplified as:

**1**. The intersection is a crossroad consisting of 4 roads with the same length.
**2**. The traffic light phase in the roads of south and north directions are always the same.The traffic light phase in the roads of east and west directions are always the same.
**3**. Once there is green signal in south-north direction, the signal in east-west direction must be red. The constraint is also applicable to south-north if the signal in east-west direction is green.
**4**. There are totally 3 lanes directed from end of the road to the intersection.
**5**. The vehicles heading towards the junction in lanes from leftmost to rightmost can only turn left, go straight, turn right respectively.
**6**. Theres no crashes among vehicles.
**7**. All the vehicles are the same, including size, highest speed, acceleration, reaction time, etc.

TABLE I: Set of Actions

| action index | from East or West | | from Nouth or South | |
|---|---|---|---|---|
| | left | straight and right | left | straight and right |
| 0 | √ | √ | × | × |
| 1 | √ | × | × | × |
| 2 | × | × | √ | √ |
| 3 | × | × | √ | × |

**8**. Each vehicle follows the traffic light and never change its lane.
**9**. The generation of the vehicles follows Poisson Distribution, helping the simulation closer to the reality.

#### *4.1.1. State Representation*

Each vehicle, on its predetermined route, in this model, has its position and velocity, . The global state is simplified as a 12 by 10 matrix, where 12 rows represents 12 route(4 incoming direction times 3 goal directions) and 10 columns represents the position(incoming lane divided to 10 segments by a 50 m interval), each matrix elements are integer indicating number of vehicles within corresponding lane segment.

#### *4.1.2. Actions*

The actions are defined as the changing of traffic light phase. There are 4 possible phases of the traffic light as shown in TABLE.1.

The duration of each phase must be equal to or longer than 10 seconds. To implement this, the transition is set to be: if the chosen phase is different from the current phase, change to that phase and simulate for 10 seconds to get the resulted state before the next action. If the chosen phase is the same as the current phase, simulate for 1 second before the next action.

#### *4.1.3. Transition*

Changing the traffic light phase too frequently is not applicable in reality. So the transition rule is based on an restriction on the phase change.

The duration of each phase must be equal to or longer than 10 seconds. To implement this, the transition is set to be: if the chosen phase is different from the current phase, change to that phase and simulate for 10 seconds to get the resulted state before the next action. If the chosen phase is the same as the current phase, simulate for 1 second before the next action.

#### 4.1.4. Reward Specification

A good policy should be able to reduce travel time, or in other words, waiting time. However, the average travel time of a vehicle is not available until it has completed it route, which leads to the problem of extremely delayed rewards. Therefore, we introduce the speed factor $f$

$$f = \frac{\text{vehicle speed}}{\text{maximum speed}} \tag{1}$$

and the waiting time $w$

$$w = \text{accumulated waiting time of vehicle.} \tag{2}$$

To recognize the case where long queue is formed on lane while some of the other lanes are empty, the boundary waiting time of a longer accumulated waiting time should have a higher weight, penalizing the policy that makes a few vehicles waiting too long. Thus the final reward function $r_t$ at each time step is given by averaging over all $N$ vehicles in the scenery

$$r_t = \sum_{i}^{N} -0.02 * w_i + f_i \tag{3}$$

where $i$ is the index of vehicles.

### 4.2. Approach

#### 4.2.1. Approach Comparison

Current researches have proposed multiple possible solutions on applying AI on urban traffic light controlling system, which differ from network input, network structure and training methods.

**Network input.** For all existing research, input of network represents current road condition, yet it can be varied in forms. Conventional approach represents state with a feature vector, including current traffic volume and time occupancy. With the introduction of new network structure, states are represented in forms of two-dimensional matrix or image, which provide more implicit information for network to explore.

**Network structure and training method.** At the early age of AI in traffic signal, controlling system is composed of multiple network with distinctive functionality, including classification network and prediction network for each category. Input is first classified by classification network then allocated to corresponding prediction network to estimate its value. The system shows the improvement it made on traffic condition, yet also have limitation on state representation and numerous labor requirement during training process. Recent years, illuminated by vigorous research and achievement in deep learning and reinforcement learning, new network structure and training approach has been proposed to address intelligent traffic signal control problem. In a recently published paper, system is trained with reinforcement learning methodology, with the network compound replaced by a single convolutional neural network. The network serves as value-function approximator that estimates Q-value for each state action pair $(s, a)$.

**Other techniques.** However, instability and partial overestimation are unavoidable if Deep Q-Network is applied. To make up for these disadvantages, the following approaches are adopted in our system.

**Experience replay.** One mature approach to address instability of reinforcement learning is experience replay. With training each experience for only a single time, network performance vibrates partially due to the non-uniform contribution brought by current and past experience to the learning outcome. With experience replay, when new $(s, a)$ pair is generated by simulator, it is stored in experiment buffer for later training, rather than be fed into network immediately. Learning and acting are logically separated and each experience probably is brought to training process for more than one time, In this way, not only less training experience is required, but also training data is more like identical independent distribution, indicated by work done by *Deepmind*[7].

**Target network.** Target network is another practice to improve the stability of Deep Q-Network. Instead of using online network for both training and predicting process, we introduce target network, which is a duplicate version of online one. During the training, only online network parameter is updated. Target network is used to predict target value for each state. Its parameters remain unchanged and are only updated every n step from the online network, where n can be set by programmers. Without target network, the target value calculated turn out to be unstable, since online network value alters every training step, which will further lead to unreliable value estimation, and destabilize whole system by turning it into a feedback loops between target and estimated Q-values.

**Dueling network.** Dueling network is based on the concept of the decomposition from Q-value to value function $V(s)$ and advantage function $A(s, a)$. Instead of using single fully-connected layer to predict Q-value of each $(s, a)$unitarily, dueling network separate the last layer of Convolutional Neural Network(CNN) into two fully connect layer, learning $V(s)$ and $A(s, a)$ separately. This turns out to be efficient for states where action does not make too much difference[7].

TABLE II: Sequence of phases and duration of baseline

| action index | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| duration(in second) | 15 | 15 | 15 | 15 |

## 5. EXPERIMENT

### 5.1. Baseline and Oracle

#### *5.1.1. Baseline*

In the baseline model, the rules defined into the traffic light phase control system is determined and unchangeable. The change of the traffic light phases repeatedly as shown in TABLE.2.

This policy gives a result of average reward $E_{[r_t]} = -66.1$ in the test case.

#### *5.1.2. Oracle*

The oracle policy, if exist, is highly sensitive to the characteristic of the incoming traffic flow. But from the simulation result we can estimate its outcome to be $E_{[r_t]} \sim -10$.

### 5.2. Experiment Using Deep Q-Learning

We trained the model using Deep Q-Learning for 100 epochs and each epoch consists of 10,000 steps. At the very beginning, the model is gradually learning the Q-values where the reward is moving back and forth. When the training epoch accumulates, the reward comes relatively stabilized at above the baseline model except for two occasions. We shall see a decent improvement after 30 epochs and a wide gap of the reward between the start and the end of the training.

However, the network might be overfitted after the beginning episodes since the routes information of vehicles are regenerated randomly after an interval of a particular number of episodes. The beginning episodes before the first regenerating might have some characteristics caused by their identical route schedule. And some of these characteristics are not desired, as they are the result of a single randomization.

This effect also slowed down the stabilization in the later episodes, since the network always tends to fit the new characteristics of the route.

### 5.3. Result and Comparison

At the beginning stage, the model is not performing well and during a majority of time, the reward is lower than the baseline result. As training goes on after epoch 30, the model continues to outperform than the baseline except for two occasions. This reflects by the implementation of the Deep Q-Learning model, the AI-driven actions could improve the rewards comparing with the baseline actions, signaling the average waiting time can be effectively lowered.

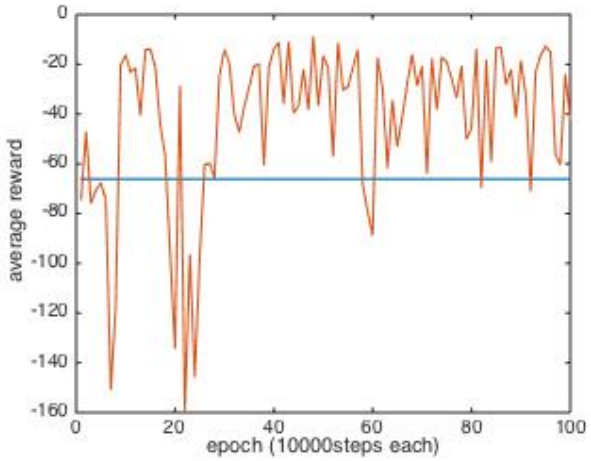


FIG. 2: Average reward received during training

TABLE III: Result comparison

| | Baseline | RL |
|---|---|---|
| average reward(over $10^4$ steps) | $-66.1$ | $-35.3 \pm 6.6$ |

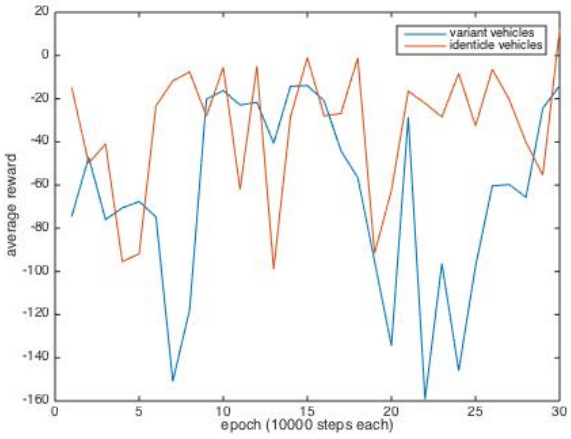


FIG. 3: Visualization of the model

## 6. ERROR ANALYSIS

The model is originally designed to deal with cars with identical features. To test its potential capacity, another experiment, with various type of vehicles with different length, acceleration and other features, for example, trucks, buses, and motorbikes, was carried out by running the same model.

However, the result has a high fluctuation does not perform as well as on identical cars. which shows that the learned model is not adaptive to different vehicle types. This might be caused by that the current model can not properly capture the influence on traffic liquidity by the variant properties of vehicles, which requires further work.

## 7. DISCUSSION AND CONCLUSION

This paper presents the optimization of the traffic light control system with applying SUMO (Simulation of Urban MObility) as the simulator and Deep Q-Learning as the main learning algorithm. The approach yields a more efficient traffic light policies comparing to the baseline model, reflecting the AI-driven traffic light system is capable for improving the current system and reducing the unnecessary waiting time encountered by vehicles significantly. Nevertheless, the reward of the model is still oscillating in a slight extent while the the number of epochs accumulates, and this can be attributed by the limited time and number of epochs being trained which can be further enhanced by the number of training epochs. Further works in this area can include a variety types of vehicles numbers of traffic lights, pedestrian and realistic vehicle behaviors such as no emergency stop.

## 8. REFERENCE

1. S. El-Tantawy, B. Abdulhai, and H. Abdelgawad. Multiagent reinforcement learning for integrated network of adaptive traffc signal controllers (marlin-atsc): methodology and large-scale application on downtown toronto. IEEE Transactions on Intelligent Transportation Systems, 14(3):1140-1150, 2013.

2. Daniel Krajzewicz, Jakob Erdmann, Michael Behrisch, and Laura Bieker. Recent development and applications of SUMOsimulation of urban mobility. International Journal On Advances in Systems and Measurements, 5(3&4), 2012.

3. V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602, 2013.

4. V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. Nature, 518(7540):529-533, 2015.

5. Daniel Krajzewicz, Jakob Erdmann, Michael Behrisch, and Laura Bieker. Recent development and applications of sumosimulation of urban mobility. International Journal On Advances in Systems and Measurements, 5(3&4), 2012.

6. Elise van der Pol. Deep reinforcement learning for coordination in traffic light control. Masters thesis, University of Amsterdam, 2016.

7. Ziyu Wang, Tom Schaul, Matteo Hessel, Hado van Hasselt, Marc Lanctot, Nando de Freitas. Dueling Network Architectures for Deep Reinforcement Learning Google DeepMind, London, UK 2016.