Deep Q-Network Model for Intelligent Traffic Light

Daffa Haritsa Maulana
School of Electrical Engineering and Informatics
Institut Teknologi Bandung
Bandung, Indonesia
Email: 13218054@std.stei.itb.ac.id

***Abstract*—The problem of controlling traffic lights at intersections has begun to be regulated using artificial intelligence (AI) or machine learning. One of the most widely implemented methods is by using Q-Learning. However, this method has some limitations, such as the inflexibility of the number of states and the need for appropriate policies in order for the system to be converged. For these reasons, in this paper, we propose a method to control the traffic light cycle using Deep Q-Network (DQN), a combination of Reinforcement Learning (RL) and Neural Networks (NN). The proposed method can eliminate the state limitation problem which becomes a problem for Reinforcement Learning. This proposed method is compared to a conventional traffic light control system in their queue length performance.**

***Index Terms*—Reinforcement learning, traffic light control system, Deep Q-Network, artificial intelligence, Machine Learning, Neural Network.**

I. Introduction

With the fourth revolution of the industry, many tasks are very difficult for humans to do in a fast and optimal time. The uses of artificial intelligence and machine learning have proven to be able to find various kinds of problems in many industries, including smart navigation, crowd detection, and traffic light control system. Most traffic light control systems still use the conventional system, namely by determining a fixed time, thus it does not depend on the real conditions of the intersection [1]. The system is considered not adaptive thus it would be difficult to be solved with this system if there is an unusual traffic condition.

Several methods have been proposed to improve the conventional traffic light control system more adaptive. One of the most popular algorithms is using Q-learning [1], [2], [9]. This algorithm was chosen because it can provide appropriate actions according to the congestion that occurs. However, Q-learning has limitations in its actions and states. In Q-learning, the number of states is fixed so it is not possible to increase the number of states. Thus, the Q-learning algorithm cannot be used if there are additional states, and the size of the state-action space is large and complex. In addition, Q-learning also cannot be applied to the intersection network (a combination of several intersections) because of the limited state of the finite agent. Multi-Agent Reinforcement Learning (MARL) is one of the solutions to this problem. There are multiple agents that allow for better coordination in various traffic conditions. However, in fact, MARL, which is currently applied in the traffic light system, is not fully coordinated. MARL agent only coordinates with any nearby agent. Overall optimization cannot be achieved with MARL if there is no overall coordination of each agent [3], [4].

Recently many researchers have developed a combination of Q-Learning and neural networks to address the previous problems. This combination is called Deep Q-Network (DQN) [5]-[7]. With the help of a neural network, the definition of the state becomes more flexible than Q-Learning. There is no need for special policies such as e-greedy to make this system to be converged as in Q-learning. Several works of literature have succeeded in proving that the DQN algorithm can be used well at single a intersection [7].

Unfortunately, the small state-action space in systems that use DQN is due to its ability to allow for a more flexible state-action space. In calculating the weight, the neural network requires a calculation which in another work states that a sigmoid is used to calculate the weight [6]. The use of sigmoid can cause vanishing gradients [8], therefore in this paper, the use of Rectified Linear Unit (ReLU) is proposed. With some considerations of several methods to regulate the traffic light system, this paper proposes an adaptive traffic light control system using the DQN method using Simulation of Urban MObility (SUMO) simulator. For these reasons, in this paper, we propose a method to control the traffic light cycle using Deep Q-Network (DQN). The proposed method can eliminate the state limitation problem.

The next section will introduce literature regarding the contributions described in this introduction section. The third section will describe the modeling and proposed method of a DQN. The fourth section will describe the design of simulation experiments that were implemented to confirm the performance of the proposed scheme and also the explanation of the results of simulation experiments. In the last section, conclusions are drawn, and further extensions are suggested to advance the proposed methodology.

II. Background

Currently, machine learning has penetrated all fields. Machine learning is considered to be able to ease human work because it is a system that can study the environment and take a certain action. One part of machine learning is reinforcement learning. Reinforcement learning works like humans. Reinforcement learning can study and analyze an environment and then it will determine the action for that state. When the action is considered good, the agent of reinforcement learning will get a reward. On the other hand, if the action is bad for the environment, the agent will be given a negative punishment or reward. Later, the agent will carry out learning iterations in certain policies to get as many rewards as possible.

A. Conventional Traffic Light Control

Conventional traffic light arrangements are currently usually divided into 2 systems. The first is the pre-timed setting [6][7][8], namely the traffic cycle timing which is determined based on the habits at the intersection without considering the real-time fluctuations in the traffic. The second is vehicle-actuated[9][10] in which the traffic light change cycle is regulated based on real-time information. This method is very suitable to be used for conditions with high traffic jams that occur randomly. However, this method relies heavily on man-made policies for changing traffic light cycles by considering only the current state of affairs without considering the long-term effects of congestion. Therefore, this method cannot be used optimally in global congestion (a fairly large area) [1].

B. Reinforcement Learning

Reinforcement learning itself is a type of machine learning based on state and action. Reinforcement Learning is machine learning that has an agent that can learn on its own. The agent will then analyze a certain environment. The results of the analysis are made into a state by the agent. From that state, the agent will determine the appropriate action so that it gets as much reward as possible from the action taken [11].

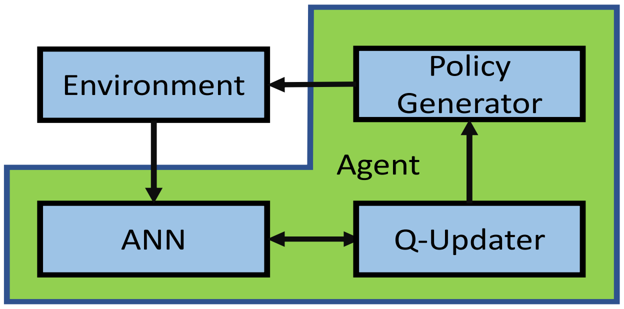

Fig. 1. *Reinforcement Learning* diagram

C. Artificial Neural Network

Artificial neural networks algorithm is one of the machine learning compliments which has many advantages. Neural Network is inspired by human biological neural networks. As like neural networks in the human body, neural networks in machine learning has their own neurons. Artificial neural networks (ANN) are simulated neural networks that mimicking the way that biological neurons.

As shown in Figure 2, ANN consists of three parts layers of neurons, the first part is called the input layer (L_i). The input layer usually consists of a layer that contains the states of the environment. The second part is called hidden layers (L_{hn}). In this part, there could be more than a layer, or even so many layers. And the last part is the output layer (L_o),

TABLE I
NUMBER OF NEURONS EACH LAYER

Layers	L_i	L_{h1}	L_{h2}	L_{h3}	L_{h4}	L_{h5}	L_o
# Neurons	80	500	500	500	500	500	4

which usually contains the actions of the agent. Table I shows the number of neurons in each layer for this work.

Each node or artificial neuron is connected to another and has an associated weight and threshold/activation function. If the output of each neuron exceeds the specified threshold value, once that node is activated then sending data to the next layer.

Neural networks rely on their own datasets they have learned to act and improve their accuracy over time and episodes. However, once this algorithm is fine-tuned for accuracy, they are so powerful tools to help a lot of human tasks due to its ability that allows us to classify big data at a high velocity.

The use of neural networks in DQN replaces the role of the Q-table in selecting the action. In conventional reinforcement learning, the choice of action is seen based on the largest Q-value owned by one of the action options. In the DQN algorithm, the action is selected through a process from the input layer to the hidden layer, until it comes out of the output layer.

The other advantage of DQN is when we converge the infinite state space (and/or action space) then it becomes impossible to use a Q-table, we need to use function approximation to generalize over states. This is can be solved by using neural networks due to their powerful abilities. Deep Q-networks usually don't usually take state and action pair as input, but take it as a representation of the state-action space.

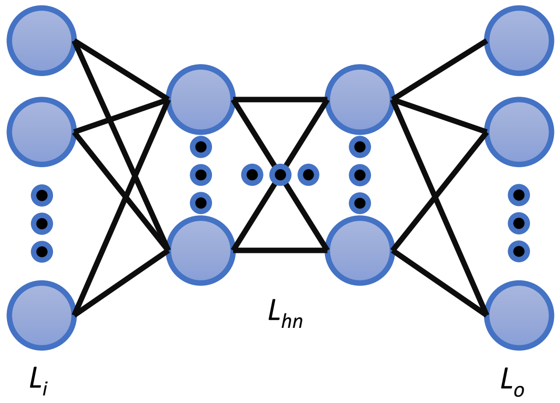

Fig. 2. Neural Network Diagram

From Fig 3, we know that we can approximate the neural networks equation. The equation of the neural network is intended to approximate for humans how the neural networks determine their choice of action. The equation is as follows:

$$Z_i = Bias + W_1X_1 + W_2X_2 + \ldots + W_nX_n \tag{1}$$

In equation (1), the parameters are :

- Z_i: Outputs from a certain layer
- W_i: Weight of a neuron
- X_i: Inputs for a certain layer
- $Bias$: Intercept or W0

D. Q-Learning Bellman Equation

When the action is selected, the state will change. By changing the state, the Q-value will automatically change. However, the Q-Value in a state can also change, this depends on the iterations that have been carried out by an agent. Changes in Q-Value at a certain state are based on an equation called the Bellman Equation [12].

$$\begin{aligned} Q_{new}(s_t, a_t) &= (1-\alpha)Q(s_t, a_t) \\ &+ \alpha(r_t + \gamma max_a Q(s_{t+1}, a_t)) \end{aligned} \quad (2)$$

In this equation, the Q-Value value is updated based on several parameters, including :

- s_t: Current State, which is the condition of the environment just before an action is performed
- a_t: Action taken on the current state
- α: Learning rate of a reinforcement learning system shows how fast an agent replaces old information with new information
- γ: Discountfactor, calculates how important it is for an agent to consider long-term rewards compared to instantaneous rewards
- r_t: The value of the current reward.

III. System Modelling

With its capabilities and characteristics, the Reinforcement Learning algorithm can be used at an intersection as a traffic light control system. We selected the 4-arm intersection type as the type of intersection to be tested with our method. In the RL part, the algorithm used is the Bellman equation. State of the RL is obtained by combining the congestion level on each arm with the number of vehicles from its lanes (each arm has four lanes). As discussed in the previous section, the reward is feedback for the RL agent whether the output is good or not. In this case, the reward is defined as the difference in waiting time between the new action and the previous action. The waiting time itself is the waiting time of a vehicle which is defined as the time period when a vehicle is considered in a waiting status (vehicle with speed=0m/s).

Then RL is connected to the neural network. The neural network used is Artificial Neural Network with layer type is Dense or feedforward. Consists of three parts, namely the input layer which contains the state and is connected to the hidden layers. Hidden layers can vary the number of layers and also the number of neurons from each layer. The output layer consists of four actions with the following conditions:

- a straight phase
- a left-turning phase (only if there is a dedicated left-turn lane)
- a straight phase for the direction orthogonal to the first one
- a left-turning phase for the direction orthogonal to the first one (only if there is a dedicated left-turn lane)

This system is implemented through a simulation of the SUMO application, which is an application that can simulate traffic engineering in real-time. With the SUMO application, the value of waiting time for each vehicle and on each arm can be obtained. In addition to the queuing time, the SUMO application can also calculate the queue length on each arm with the configuration of the length of each vehicle assuming the same all.

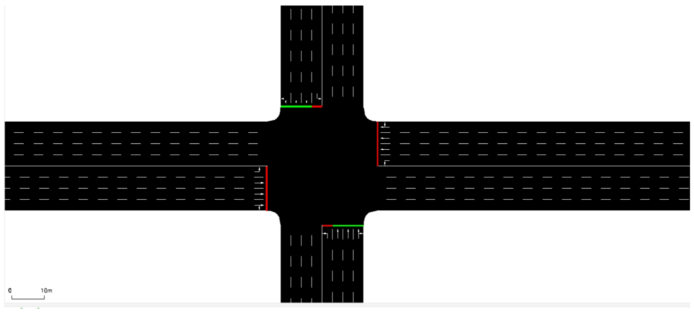

Fig. 3. intersection Model [12]

IV. Experimental Results

The configuration used in this experimental results is we use a total of seven layers neural network. The first and the last layers act as input layers as well as output layers, while the other five layers are the hidden layers. Each hidden layer has the same number of neurons, which is 500 neurons per layer. In feed-forward NN usually, the learning process requires the declaration of an error function. In this experiment, we choose to use the mean squared error.

$$E(X) = \frac{1}{2N} \sum_{i=1}^{N} (g(W_i X_i + bias) - Z_i)^2 \quad (3)$$

In equation (3), the parameters are:

- $E(X)$: Mean squared error function,
- N: Total number of state-action pair,
- g: Activation function,
- W_i: Weight of a neuron,
- X_i: Inputs for a certain layer,
- Z_i: Outputs from a certain layer,
- $Bias$: Intercept or W_0.

From Equation 3, there is what is called an activation function, which in short is the threshold of the layer to send data to the next layer. We chose to use ReLU, the main purpose is to avoid vanishing gradient points that can occur in other types of activation functions such as sigmoid. In addition, ReLU can make the learning process faster as follows;

$$Z_i \leq 0 \quad then \quad g(Z) = 0, \quad (4)$$

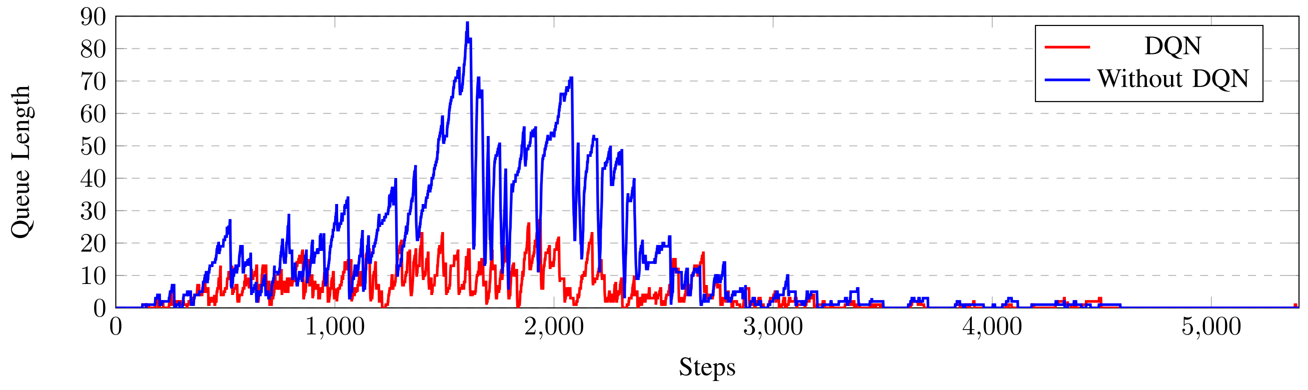

Fig. 4. DQN performance on single intersection

TABLE II
MAXIMUM NUMBER OF QUEUES AND ITS AVERAGES

Method	Maximum	Average	x
Without DQN	88	12.887	x
DQN	27	4.077	x

and for the value of

$$Z_i > 0 \quad then \quad g(Z) = Z. \tag{5}$$

After the setup for the experiment is being completed, the results of the comparison between the performance of the DQN algorithm against traffic lights are obtained without using the DQN algorithm. With the same cycle of 15 seconds for the green light and 3 seconds for the yellow light, 500 episodes of training were conducted on an environment intersection in the SUMO application. With the same environment intersection, the results of the comparison of queue lengths between the two systems are obtained.

In Figure 5 and Table II, we can see the performance of queue length between the DQN algorithm and those without the DQN algorithm. In a system that does not use DQN, the action is chosen randomly with a green cycle of 15 seconds and a yellow for 3 seconds. While the DQN algorithm uses RL and ANN to choose the best action. With the DQN algorithm, the maximum queue length that occurs in each cycle is the total maximum queue length is only 27 vehicles. Meanwhile, without the use of DQN, the condition of the intersection becomes very bad with the maximum queue length that occurs reaching 88 total vehicles.

In addition to the maximum queue length, in Figure 5. The average queue length on a system without DQN shows a much higher average queue length than the system with DQN algorithm installed.

V. CONCLUSIONS

The problem of controlling traffic lights at intersections has begun to be regulated using artificial intelligence (AI) or machine learning. One of the most widely implemented methods is by using Q-Learning. However, this method has some limitations, such as the inflexibility of the number of states and the need for appropriate policies in order for the system to be converged. For these reasons, in this paper, we propose a method to control the traffic light cycle using Deep Q-Network (DQN), a combination of Reinforcement Learning (RL) and Neural Networks (NN). The proposed method can eliminate the state limitation problem which becomes a problem for Reinforcement Learning. This proposed method is compared to conventional traffic light control system in their queue length performance. The DQN algorithm shows better performance over the conventional control system.

REFERENCES

[1] H. Wei dan H. Yao, “IntelliLight : A Reinforcement Learning Approach for Intelligent Traffic Light Control,” Research Track Paper, London, 2018.

[2] P. Mannion, J. Duggan, and E. Howley, “An experimental review of reinforcement learning algorithms for adaptive traffic signal control,” in Autonomic Road Transport Support Systems. Cham, Birkhäuser, 2016, pp. 47–66.

[3] S. El-Tantawy, B. Abdulhai, and H. Abdelgawad, “Multiagent reinforcement learning for integrated network of adaptive traffic signal controllers (MARLIN-ATSC): Methodology and large-scale application on downtown Toronto,” IEEE Trans. Intell. Transp. Syst., vol. 14, no. 3, pp. 1140–1150, Sep. 2013.

[4] L.Kuyer et al., “Multiagent reinforcement learning for urban traffic control using coordination graphs,” in Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Berlin, Heidelberg, Springer, 2008.

[5] J. Lee, J. Chung and K. Sohn, ”Reinforcement Learning for Joint Control of Traffic Signals in a Transportation Network,” in IEEE Transactions on Vehicular Technology, vol. 69, no. 2, pp. 1375-1387, Feb. 2020, doi: 10.1109/TVT.2019.2962514.

[6] L. Li, Y. Lv and F. Wang, ”Traffic signal timing via deep reinforcement learning,” in IEEE/CAA Journal of Automatica Sinica, vol. 3, no. 3, pp. 247-254, 10 July 2016, doi: 10.1109/JAS.2016.7508798.

[7] H. Jeon, J. Lee, and K. Sohn, “Artificial intelligence for traffic signal control based solely on video images,” J. Intell. Transp. Syst., vol. 22, no. 5, pp. 433–445, 2018.

[8] M. Rosyidi, S. Bismantoko and T. Widodo, ”Reinforcement Learning with the Classical Q-Learning Algorithm for Optimizing Single Intersection Performance,” 2020 International Conference on Data Analytics for Business and Industry: Way Towards a Sustainable Economy (ICDABI), 2020, pp. 1-5, doi: 10.1109/ICDABI51230.2020.9325657.

[9] Seung-Bae Cools, Carlos Gershenson, and Bart D’Hooghe. 2013. Self-organizing traffic lights: A realistic simulation. In Advances in applied self-organizing systems. Springer, 45–55.

[10] Isaac Porche and Stéphane Lafortune. 1999. Adaptive look-ahead optimization of traffic signals. Journal of Intelligent Transportation System 4, 3-4 (1999), 209–254

[11] University of the Ryukyus Faculty of Engineering, “The 24th LSI 2021 Design Contest in Okinawa,” University of the Ryukyus Faculty, 2021.Available: http://www.lsi-contest.com/2021/shiyou3-1e.html. [Accessed 7 October 2021]

[12] AmenRa (2019) drl-traffic-lights-control. https://github.com/AmenRa/drl-traffic-lights-control.git