Methods: RL, QLearning

Ideas:

1. Multi-crossroads, use previous traffic light situation to make decision in next environment, seems like Markov decision
2. Use context to describe to environment, make some category of the environment, like Busy, Not Busy…

Limitation: When facing emergency event, crossroads still need hand-craft. The environment

<https://pytorch.org/tutorials/intermediate/reinforcement_q_learning.html>

Pytorch tutorial

<http://incompleteideas.net/book/the-book-2nd.html>

Paper description

This paper, published in an early stage in 2018, represents one of the initial attempts to address traffic signal control problems using reinforcement learning. The primary objective is to make traffic signals, referred to as IntelliLight or DRL (Deep Reinforcement Learning), more dynamic and responsive to road conditions. It is compared with two other types of traffic signals: Fixed Time Control (FT), which follows predetermined timing schedules, and Self-Organizing Traffic Light Control (SOTL), which manually adjusts traffic signals based on real-time traffic conditions. IntelliLight, based on the Deep Q-Learning (DQN) framework, enhances DRL with the incorporation of Memory Palace and Phase Gate (definitions of which will be provided later). Experimental results show that IntelliLight, across different scenarios, outperforms FT and SOTL to some extent, demonstrating the effectiveness of reinforcement learning-based intelligent traffic signals in alleviating traffic congestion. In traditional research on reinforcement learning-based traffic signal control, two main technical challenges are encountered: How to represent the current road conditions to the model. How to model the correlation between the environment and decision-making, often related to finding an optimal value function. Apart from these two challenges, there is a third challenge not mentioned in the paper: the lack of immediacy in rewards. It is difficult to immediately determine whether the decisions made at a given moment are globally optimal. This issue is addressed within the paper but is not explicitly discussed. In this paper, the authors do not emphasize the first challenge. They believe that representing road conditions can be achieved by processing images of road surfaces using convolutional neural networks (CNN) and using these as inputs to the IntelliLight model. For the second challenge, the authors provide a deeper explanation of how they design the value function. The model is divided into two parts: the online part and the offline part. The offline part primarily collects road information for model training, while the online part includes the main tasks of gathering road information, making decisions using the model's policy and epsilon-greedy strategy, receiving feedback, and storing information. This section is further divided into two parts: further training and upgrading. The further training portion utilizes samples from two different time points (which can be multiple) for a traffic light as examples. After determining whether to change the traffic signal and receiving feedback, this data (state, action, reward) is stored. The model is then upgraded through mini-batch iterations. The upgraded model is used to make decisions for the next traffic signal based on the road conditions, and the process is repeated. This approach ensures that the traffic signal continues to improve, avoiding the impact of delayed rewards and continually optimizing the signal for the best performance. This process leads to the creation of a Value Function, which considers: The total queue length L on approaching lanes, where L is calculated as the total number of waiting vehicles on the given lane. The total delay D on approaching lanes, where lane i's delay Di is defined as in Equation 1. Di = 1 - (lane speed / speed limit) The total updated waiting time W on approaching lanes, equal to the sum of W for all vehicles on approaching lanes. The updated waiting time W for vehicle j at time t is defined as in Equation 2. Wj(t) = { Wj(t-1) + 1, vehicle < 0.10 0, vehicle ≥ 0.1 } The indicator C of traffic light switches, where C = 0 for maintaining the current state and C = 1 for changing the current state. The total number of vehicles N passing through the intersection during the time interval t after the last action a. The total travel time T of vehicles passing through the intersection during the time interval t after the last action a, defined as the total time (in minutes) that vehicles spent on approaching lanes. To estimate rewards based on the state and action, the model needs to learn a Deep Q-Network Q(s, a). However, it is challenging for the network to determine which processing method to use at different time stages. Therefore, the authors introduce a Phase Gate to assist the model in distinguishing between different phases. Phase Gate learning involves selecting the appropriate Phase Gate to activate the corresponding model based on the current state, action, reward, and other information. This helps the system adapt better to different environments and tasks, improving learning effectiveness and generalization. Additionally, the model becomes proficient at estimating rewards for frequent phase-action combinations but may ignore less frequent combinations. This can lead to suboptimal decisions in infrequent scenarios. To address this imbalance, the authors introduce Memory Palace. Different samples for various phase-action combinations are stored in separate Memory Palaces. An equal number of samples are then selected from these Memory Palaces, ensuring a balanced training process. This approach prevents interference between different phase-action combinations during training and enhances the network's ability to accurately predict rewards. Memory Palace is maintained by periodically sampling from memory and using these samples to update the network. New data samples are added to the memory, and old samples are occasionally removed to maintain it. In summary, this paper conducts experiments using six different methods: Fixed-time Control (FT): This method uses predetermined cycle and phase timing plans and is widely used when traffic flow is stable. Self-Organizing Traffic Light Control (SOTL): This method controls traffic signals based on the current traffic state, including elapsed time and the number of vehicles waiting at red lights. Specifically, the traffic lights change when the number of waiting cars exceeds a manually adjusted threshold. Deep Reinforcement Learning for Traffic Light Control (DRL): This method applies the DQN framework to select optimal traffic light configurations for intersections. It relies solely on the original traffic information as an image.

Listed some features might be used.

**Cartpole**