COMP9444 Project Summary

Traffic Light Control Using a Deep Learning Agent

z5360587, Aryaan Bhatia

z5308363, Kane Jackson

z5420215, Zachary Velyvis

z5362255, Arjun Malik

z5455287, Pratik Kamath

I. Introduction

The exponential increase in the number of motor vehicles that drive at any given point has generated a need for an efficient traffic management solution. Poor traffic management can lead to a drop in passenger and driver safety and also a significant increase in congestion across roads. Currently implemented solutions for the phasing of traffic lights is considered to lack the flexibility required in the changing ecosystem of the roads. They therefore fail when placed under unforeseen or challenging situations such as peak hour traffic or a challenging intersection.

This report aims to provide a solution to the problem of rigid traffic control systems by implementing a dynamic control system that takes into account many environment features to decide the next phase of the light system. The system will be implemented as a model that is trained by Deep Reinforcement Learning to make the best possible decision.

By employing the deep learning techniques, this project presents a flexible and dynamic solution to the growing problem of traffic congestion in urban environments. The solution discussed in this paper can be implemented in intricate urban environments where complex decisions must be made about the phasing of the traffic lights and conditions.

II. Related Work

Some work has been done on this subject in the past that can be summarised into 4 main papers each of which attempts to address one main facet of the problem. Paper 1, (SUMO-Based Traffic Simulation (Lopez et al., 2018), described the environment set up for the simulation of traffic control problem which dove into the viability for using SUMO to train traffic light systems and its limitations. The main thing to learn from the paper is that however, while SUMO provides a realistic environment for testing, it does not inherently include adaptive traffic control mechanisms or real-time learning agents.

In paper 2, Human-Level Control in DRL (Mnih et al., 2015), The main thing that is to be understood is that Deep Q-Network (DQN) is a viable way of training and maintaining human level control in highly complex environments such as video games or highly dynamic environments such as the traffic light system in many urban cities.

Paper 3, Intelligent Traffic Signal Control at Urban Intersections (Guo et al., 2019), talks about the use of a DQN network to monitor the queue length across a defined intersection. The DQN model implemented is shown to beat the other benchmarks mentioned in the paper.

The limitations as seen in these articles is that only a rudimentary DQN system is implemented and therefore with hyper parameter tuning and the use of a double DQN, the results may be further improved.

III. Methods

The state space in any reinforcement algorithm is defined as the total number of states that the system is able to be in at any given time during the training period. The state space for this specific problem is extremely large as there are many variables that must be taken into account, in addition to the changing of the light phases. This therefore makes a Q-table solution succumb to the curse of dimensionality. In order to solve this issue, a Q-network solution is used that does not store all of the reward values in a table but approximates them using a neural network. This is the reason why we opted to utilise a Deep Q Network for the solution of this problem.

We initially wanted to set up three baseline models that would give us the groundwork of which to base the quality of our own proposed solution. In order to set these up, use of the stable-baselines3 package was utilised. The inbuilt DQN and Double DQN models provided a sufficient launching point for our own proposed solution to be measured off of. The double DQN system was included in the baseline models as that is what we chose as our final solution. This is because the double DQN solution was able to address the DQN's vulnerability to overfitting and providing us with a more stable convergence.

When utilising the baseline models The double DQN was able to perform better after 500 training episodes and converge at a lower final waiting time.

While the baselines performed well, we decided to write our own DQN and double DQN models in order for us to be able to become more selective with the reward function that will be implemented that chooses the next step. It was observed that the baseline reward function reset the waiting times to zero as soon as some movements occurred. This is a large problem as the final solution for these baseline models "flickered" the lights from yellow to red to green in order to drastically reduce their definition of a wait time but not actually produce the best possible solution.

We then built a new reward function that also takes into account the number of cars that move through the light at the green phase in order to make sure the flickering does not occur.

IV. Experimental Setup

For the simulation of the environment we utilised an app called SUMO (Simulation of Urban MObility). We have used SUMO as it allowed us to control the vehicle flow, intersection dynamics and create a customisable road setup at an intersection. Our intersection, built using NetSim, is an 8 lane highway intersection with 8 different light phases and 1000 vehicles traversing a randomised route and moving through the intersection at random times. The reinforcement learning agent can switch to a green phase, which must last a minimum of 5 seconds, and a yellow phase, whose time is fixed at 3 seconds.

The key outputs that the SUMO APIs provide include traffic light times, lane conditions, vehicle speeds, and their relative location. All of these parameters are then fed into the reward function that then calculates the next best action that the system to take. The reward function was carefully designed to balance multiple objectives: rewarding the percentage of queued cars cleared during a phase (scaled exponentially), penalising excessive queueing (prioritising heavily queued lanes), and discouraging frequent phase switches to ensure smoother traffic flow.

The reward function:

$$R = S + e^{4p} - e^{0.2W}$$

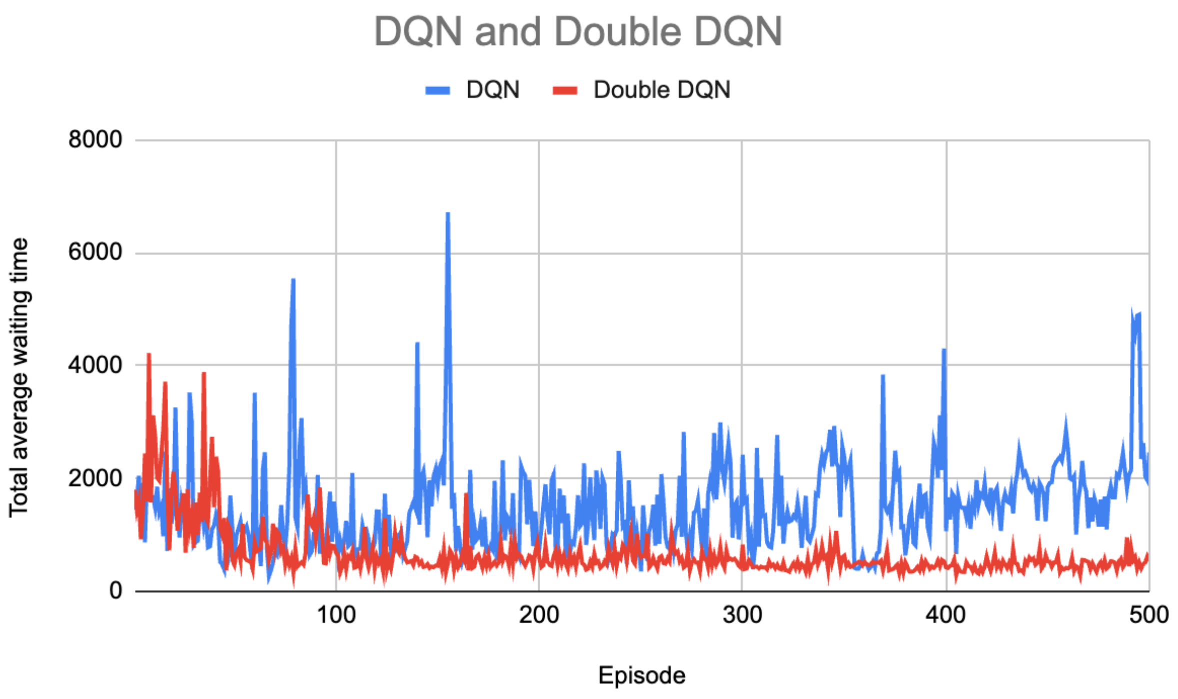
S is the static action indicator. If S = 1, the current action is the same as the previous one and S = 0 if the current action is different to the previous one. P is the percentage of vehicles that are served in the current phase. The higher the number of vehicles that are served, the higher the reward. W is the average waiting time across all lanes and penalises the reward function. As can be seen in the equation, if W is high, the reward function is penalised as opposed to a smaller waiting time where the reward is penalised less.

The key parameter that was utilised in the evaluation of the results was the average wait time of the cars in the system. We decided that this was an appropriate metric to measure the accuracy of our model because this directly measures the information that is addressed in our problem statement. Our problem statement defined a need to increase the efficiency of traffic flow and therefore a reduction in wait times for cars in a queue or stopped at a red light is a clear indicator of this.

The key hyperparameters that were tuned in this process were the number of episodes that the model was trained on and the exploration rate of the reinforcement learning agent. After utilising manual fine tuning techniques we found that the models usually converged around the 600 episode mark, and the exploration rate being higher, forced the model to converge later but achieve a better result as opposed to a lower exploration rate and higher percentage of exploitation.

V. Results

The key results are contained in the following visualisation, where our single DQN model is compared with our final tuned double-DQN model.



Our results demonstrate a promising approach to a real world implementation for traffic light control, however for deployment in the real world the learning problem becomes considerably more complex. Many real world conditions that are difficult to take into account *must* be accounted for including pedestrian activity, outlier behaviour where drivers have highly compromised adherence to expected actions (driving under the influence, medical episodes etc), unexpected swarm behaviour during extreme weather events and so on. For each of these situations the collection of data our models rely on would be a difficult problem in the real world. The most promising solution would be a powerful CNN that can interpret a variety of situations dynamically, and training this neural net to learn this task would itself be a demanding endeavour. There would be costs associated with building and training this network, along with the requirement of computer vision hardware and infrastructure installation requiring logistics and financial arbitration. The integration of existing data from induction loops is worth considering, however this would provide limited information which may not be of use.

Architectures of the neural networks themselves were not significantly distinct; based on existing literature the primary differentiator is our reward function. Baselines utilised a rudimentary reward function as follows.

$$r_t = D_{a_t} - D_{a_{t+1}}$$

Conversely our reward function utilised exponentiation to make the reward signal clearer to the network in order to achieve stronger convergence on the solution.

$$R = S + e^{4p} - e^{0.2W}$$

Where p is the percentage of traffic served, and w is the number of cars in the lanes of the intersection.

Our solution’s primary advantage over existing models manifested in considerably faster training, and demonstrated rapid convergence to the solution without exhibiting flickering or similar artefacts. While the state of the art demonstrates superior performance, a component of this is unnatural movement of vehicles which synthetically reduces total average waiting time, however these movements do not translate to realistic conditions. Our model achieved comparable wait times with more realistic behaviour and demonstrated a stable total average waiting time illustrating robust adaptation to unpredictable conditions.

VI. Conclusions

Key strengths include faster convergence than existing models, paired with rapid training time. This allows for rapid iteration on our architecture in order to achieve further refined performance and indicates great promise for scaling to a larger traffic system. Further, the efficient architecture of our model is cheap to train, requiring minutes and personal computers as opposed to enterprise level hardware and hours. In addition our architecture is simple and relatively easy to interpret, including our reward function. Finally, our solution operates well with realistic vehicle behaviour thus priming our model for development and eventually real world deployment.

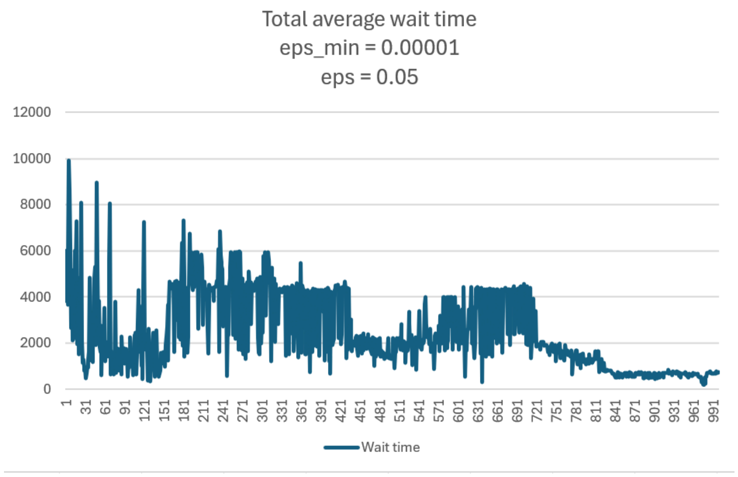
Limitations of our model include simplifying assumptions regarding pedestrian traffic and relatively outlier free data. In order to deploy this model in the real world, the implementation of a powerful CNN, and perhaps even a transformer with reasoning ability would be effective in accounting for the extremely volatile nature of traffic conditions in the real world. Such a model would also cope with weather conditions and outlier behaviour as detailed in section V.

References

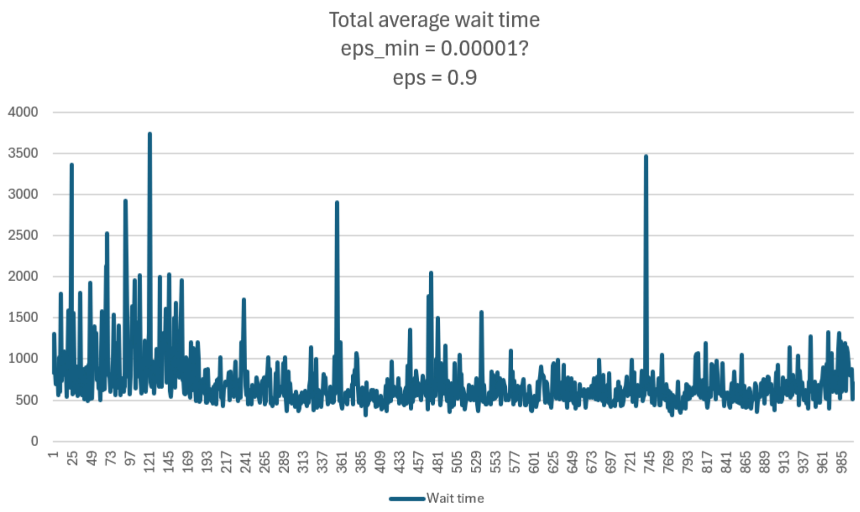
1. stable-baselines3.readthedocs.io. (n.d.). Stable-Baselines3 Docs - Reliable Reinforcement Learning Implementations — Stable Baselines3 1.2.0a2 documentation. [online] Available at: https://stable-baselines3.readthedocs.io/en/master/.
2. Guo, M., Wang, P., Chan, C. Y., & Askary, S. (2019, October). A reinforcement learning approach for intelligent traffic signal control at urban intersections. In 2019 IEEE Intelligent Transportation Systems Conference (ITSC) (pp. 4242-4247). IEEE
3. Van der Pol, Elise, and Frans A. Oliehoek. "Coordinated deep reinforcement learning for traffic light control." NeurIPS Workshop on Modeling and Decision-making in the Spatiotemporal Domain. 2016.
4. Mnih, Volodymyr, et al. "Human-level control through deep reinforcement learning." Nature 518.7540 (2015): 529533.
5. Lopez, P. A., Behrisch, M., Bieker-Walz, L., Erdmann, J., F̧lttȩrd, Y. P., Hilbrich, R., ... & Wießner, E. (2018, November). Microscopic traffic simulation using sumo. In 2018 21st international conference on intelligent transportation systems (ITSC) (pp. 2575-2582). IEEE.

Appendix

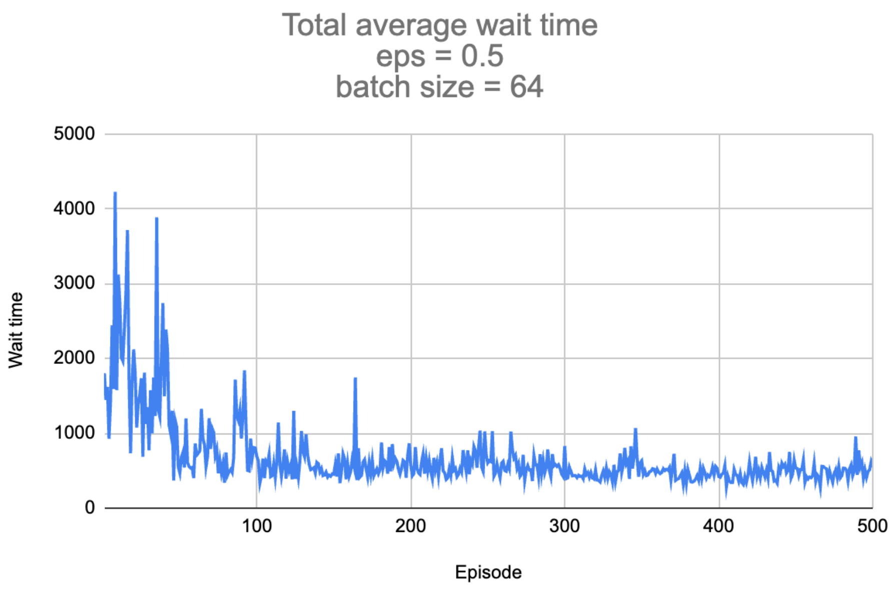
1.1 Our model double DQN (Look for hyper parameter changes)



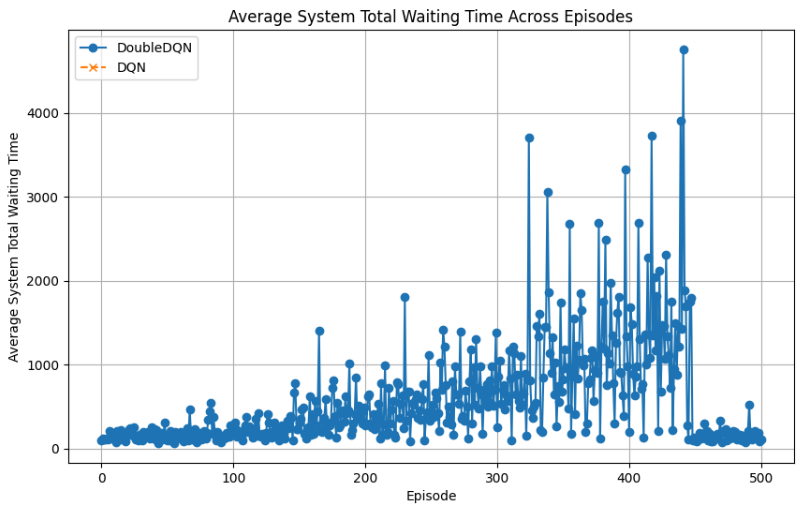
1.2 Our model double DQN (Look for hyper parameter changes)



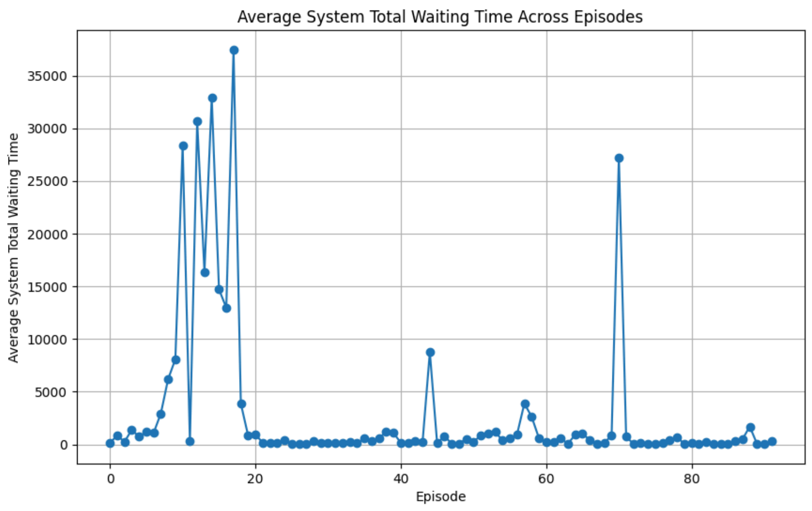
1.3 Our model double DQN final model results



2.1 Baseline model double DQN



2.2 Baseline model single DQN


Average System Total Waiting Time Across Episodes
Average System Total Waiting Time
Episode