Reinforcement Learning with a Deep Q-Network for Traffic Signal Control

Mathew K. John
UMass Amherst
Amherst, MA 01003
mkjohn@umass.edu

Ryan Fletcher
UMass Amherst
Amherst, MA 01003
rafletcher@umass.edu

Justin Fallo
UMass Amherst
Amherst, MA 01003
jfallo@umass.edu

Abstract

Current traffic control systems struggle to handle the intensity of traffic in urban areas. Decreasing road congestion by introducing novel methods for traffic control systems would reduce greenhouse gas emissions, the commute time of drivers, and the consumption of resources by vehicles. The paper explores an extension to an approach that uses a deep Q-network (DQN) for traffic signal control. The extensions implement multi-agent reinforcement learning (MARL) to consider the traffic patterns of a targeted region and determines the optimal traffic signal plan accordingly. The extension attempts to make the previous approach super-Turing. The final report will reach conclusions about these extensions and the original approach from previous literature.

1 Introduction

1.1 Background and purpose

The problem of road congestion in urban areas presents disadvantages for drivers and negatively impacts the environment. Road congestion leads to longer commute times for drivers, increasing the amount of fuel consumed by their vehicles. The emission of greenhouse gases such as carbon dioxide and nitrous oxide from these vehicles is a leading cause of global warming. Currently, most traffic signal control systems use fixed time signal plans because such plans are inexpensive and easy to implement. Fixed time signal plans use pre-set signal cycles that are determined by previous traffic data. Thus, traffic signal systems that use fixed time signal plans cannot determine the best signaling for traffic demand at a particular time and location. The paper introduces an approach to control traffic signals based on the traffic demand in a targeted region and live changes in traffic patterns to optimize traffic flow.

1.2 Research scope and methodology

The initial intent of the project was to compare traffic data obtained from three different approaches to traffic signal control: the Fixed Time Signal Control approach, a previous approach discussed in Section 2, and the proposed bio-plausible approach, with the networks of the latter two trained to convergence. The implemented methodology generally followed the initial intent of the project, but some alterations were made to improve results. Steps were taken to enhance the realism of the traffic simulation in order to make our results more reliable, but in doing so, our hardware became unable to support the computational requirements of the network. Thus, we had to decrease the scope of the training periods to account for computational limits. To determine the efficacy of each approach, current results compare the various control schemes at each episode. Current results also demonstrate reliable learning by comparing two episodes with the same traffic pattern for one of the network control schemes. The final results will compare the Fixed Time Signal Control approach to the

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

network control schemes trained on two or three episodes of different traffic patterns to demonstrate the presence of super-Turing computation. The previous literature suggests that the convergence of these network schemes will take 50 to 60 epochs of 1.2 simulated hours of training but our hardware limits us to two or three episodes of four simulated hours. Thus we will extrapolate from the obtained results as well as possible.

2 Literature review

2.1 Deep Q-Network

RL is a machine learning technique that involves three components: actions, reward, and observation. Actions are what an agent can do, the reward reflects the success of an agent's action, and the observation represents the state of the environment. RL is used to solve sequential decision-making problems by learning how to behave in a given situation to maximize the reward. One of the predominant RL algorithms is a DQN. Q-learning is learning algorithm that has a Q function that estimates a Q value for each state–action pair where the Q value determines whether to perform a specific action in a specific state. A DQN uses a deep neural network to estimate the Q function and compute the Q value so that it can solve problems with large state and action spaces. A DQN uses experience replay and target network techniques. Experience replay is a method in which an agent stores a sample of their experience with the environment in memory and randomly extracts it for learning. By removing high correlation of the training data, DQNs suppress overfitting and enable stable learning. Target network techniques use the target network to eliminate learning instability when using one network.

2.2 Prior traffic signal control studies using RL

DQN-based traffic signal control models Park et al. developed two traffic signal control models using a DQN: an isolated intersection traffic signal control model and coordinated intersection traffic signal control model. The performance was evaluated by comparing the developed models to an optimal fixed-time signal model. A simulation environment was constructed using Vissim and the COM interface. The environment was so simple that applying the developed traffic signal control model to a real site could prove difficult. Thus, the model was limited by a lack of practical experience. The traffic signal control method for the models provides a signal timing plan based on traffic demand for the next cycle. At the end of a cycle, the traffic signal control model selects the optimal signal timing plan for the next cycle. Traffic demand for a cycle was described by the maximum queue length which is the maximum number of waiting vehicles in the cycle. For the RL algorithm, the state was described by the maximum queue length, the action was described by the signal timing plan selected for the next cycle, and the reward was described by the average stop delay. Thus, the models learned by selecting actions that minimized the average stop delay per cycle. Evaluation and validation showed that performance was superior to that of the optimal fixed-time signal plan in terms of average delay, average travel speed, and average number of stops for both models.

RL benchmarks for traffic signal control Ault and Sharon propose a toolkit for developing and comparing RL based traffic signal controllers that includes implementation of RL algorithms for signal control and benchmark control problems based on realistic traffic scenarios. The toolkit allows for comparison of RL based signal controllers while providing benchmarks for future comparisons. The paper compares the relative performance of current RL algorithms on the Cologne and Ingolstadt road networks. These road networks were chosen because they are well-accepted by the transportation community and include a congested downtown zone with multiple signalized intersections. The paper suggests that a deep Q-learning approach is best for more realistic signal layouts.

3 Development of a traffic signal control model based on a DQN

3.1 Selection and implementation of reinforcement-learning algorithms

The RL algorithm we selected was a DQN because of the success of previous literature that used a DQN. We also implemented a MARL algorithm. RL algorithms allow for learning based on experience collected during training. Thus, the model can adapt to different scenarios to determine

the optimal traffic signal plan. Because the data generated during traffic signal control simulation is large and storage is limited, experience replay cannot handle an enormous number of steps. Experience replay has been accused of being biologically implausible because how the brain handles mass amounts of data is unknown. The replay process of the hippocampus is more generative and less exact. To improve performance of the DQN, we employed experience replay and target network techniques. A double-ended queue of 4-tuples with maximum capacity 10,000 was used to implement the former where each 4-tuple contained an action, reward, state, and subsequent state to represent a RL step. This implementation conserved memory and allowed for the safe storage and removal of new step reports and old 4-tuples, respectively. 128 samples were randomly selected at each step for learning. We implemented a MARL algorithm so that agents could cooperate by sharing information about the impedance of vehicles at their traffic signal. This allowed for agents to make decisions within the context of the entire transportation network. Two MARL algorithms were developed. The first made use of a single DQN for all agents. The second, which we named Cooperative DQN, assigned each agent a DQN and the agents cooperated with each other. The structures of these algorithms can be seen in Fig. 1. The rewards received by each agent indicated the maximum impedance weighted by the number of emergency stops made by vehicles in the controlled incoming lanes.

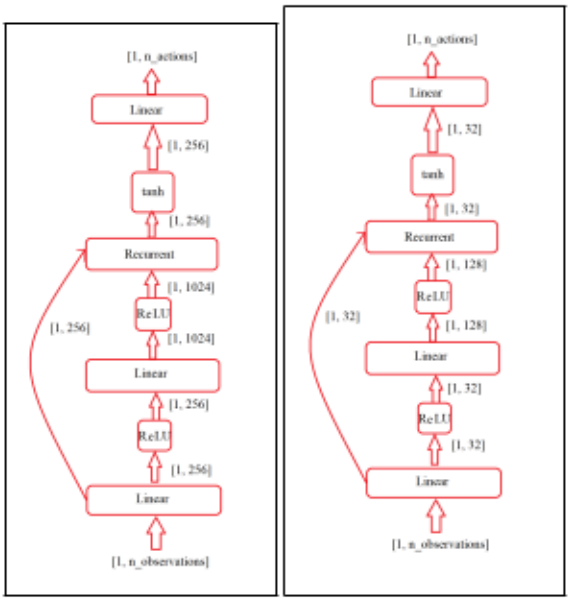

Figure 1: Single DQN (Left) and Cooperative DQN (Right)

3.2 Development of simulation environment framework

Park et al. claim that RL agents must observe a simulated traffic environment to decide which action they should take and that agents get a reward from the environment too. To implement a simulated traffic environment we used the microscopic traffic model, SUMO which stands for Simulation of Urban Mobility. The SUMO binary would simulate a microscopic traffic flow on the Ingolstadt network that our reinforcement agents could observe by interfacing TraCI (Traffic Control Interface) through Python. SUMO is a microscopic traffic simulation because it can reckon microscopic variables of vehicles each second or millisecond. Microscopic vehicle variables include speed, position, acceleration, CO2 emissions, CO emissions, NOx emissions, particulate matter emissions, hydrocarbon emissions, noise emissions and fuel consumption, In addition to using the PyTorch package, we used the Sumo-RL package and TraCI package to develop a TSC model based on RL. SUMO has various car following models, junction models and lane change models. According to the German Aerospace Center, an example of a car-following model in SUMO is the Krauss model that assigns each driver a speed depending on the minimum time the driver desires to travel before reaching rear bumper of the vehicle in front of them travelling in the same direction the driver is travelling. The variety of model types makes simulation more realistic because it portrays the driving behavior of drivers more accurately. The PettingZoo package was used implement MARL to control

all the traffic lights in the virtual transportation network because it had more than one controlled junction.

3.3 Development of a coordinated intersection traffic signal control model using a DQN

Model development and learning The test bed we used to conduct TSC experiments using single and coordinated RL agents equipped with DQNs was a traffic scenario based on the German city of Ingolstadt. The virtual transportation network in our simulation environment contained 23339 junctions, 122877 edges, 185052 lane connections, 17 roundabouts and 1484 traffic lights. However, only 12138 of the junctions and 27754 of the edges were loaded when debugging network attributes. Not all the intersections in the traffic scenario were signalized, some of the intersections in the simulation were uncontrolled. Figure **??** displays a rendering of the Ingolstadt transportation network in SUMO GUI. Our DQN based TSC models learned on the transportation network for exactly one episode.

4 Evaluation and validation of the developed traffic signal control model

4.1 Traffic signal control model evaluation and validation method

Selecting measures of effectiveness To evaluate the performance of the traffic signal control models we developed using DQNs, we selected 3 different measures of effectiveness (MOEs): system total waiting time, system average vehicle speed and total CO2 emissions. The MOEs are calculated at each step. The first two measures of effectiveness we selected were based on prior work in the field of traffic signal control and RL. The total waiting time is defined to be the total amount of time vehicles spent waiting in the transportation network measured in seconds. The average vehicle speed is defined to be the average speed of all the vehicles in the transportation network measured in meters per second. The total CO2 emissions is the mass in kilograms of CO2 emitted by all the vehicles in the transportation network.

Methodology of evaluating and validating the developed traffic signal control model To evaluate the performance of the DQN based traffic signal control models, the passive control group we used is the fixed traffic signal control model. The two single DQN based TSC models and the two cooperative DQN based TSC models were compared to the FTSC model. Also, to evaluate the performance of the DQN based traffic signal control models that used compressed observation matrices, the passive control groups we used are the DQN based traffic signal control models that used the default uncompressed observation function. The single DQN based TSC model that used a compressed observation matrix was compared with a single DQN based TSC model that used the default uncompressed observation function. The cooperative DQN based TSC model that used a compressed observation matrix was compared with a cooperative DQN based TSC model that used the default uncompressed observation function. Moreover, to compare the performance of the single DQN based TSC models with cooperative DQN based TSC models, the comparison groups we used were the single DQN based TSC models. The system total waiting time, system mean speed and total CO2 emissions of the cooperative DQN based TSC models were compared with the single DQN based TSC models that had the same observation function.

4.2 Evaluation and validation of coordinated intersection traffic signal control model

Estimation, optimization and emulation methodology To evaluate the performance of the DQN based TSC models, we required a virtual environment created based on real-world traffic data and geographic data. The real-world traffic conditions and traffic light programs of the German city of Ingolstadt were emulated using estimations of the traffic conditions in the city. According to Harth and others, conducting traffic signal control experiments in the real world is expensive and consumes great amounts of time. Furthermore, real world traffic signal control experiments can be dangerous and their results may not be reproducible. In comparison, traffic signal control simulation is cheap, quick, safe and the results of experiments can be replicated. In addition to improving the validity of a traffic simulation, using real-world traffic conditions and real world geographic maps allows us to compare the results of real-world traffic signal control with simulated results. The traffic data scenario we used in our research paper was created in the SAVe;, SAVeNoW and KIVI research

projects. OpenStreetMap was used to model a virtual transportation network based on the city of Ingolstadt. Real word traffic data from the city of Ingolstadt was transferred to SUMO. Although Ingolstadt uses actuated traffic signal control and prioritizes public transportation, the average green time of the traffic signals was used to create hourly fixed time traffic signal control programs for 75 of the traffic lights. The static programs of the rest of the traffic lights were manually created to handle traffic demand. Traffic flow through the virtual network was calibrated using regional traffic statistics in Ingolstadt. Factors like the number of inhabitants in Ingolstadt and their preferences were considered when generating the traffic flow. Langer and others write that a genetic algorithm was employed to optimize 6 parameters of the car-following model and the junction model of the SUMO environment. The Krauss car-following model was determined to be the optimal car-following model in SUMO for the Ingolstadt transportation network.

Traffic scenarios simulated We simulated a 4 hour long traffic scenario from 6 AM to 10 AM on Wednesday, September 16 2020 in Ingolstadt for each of the TSC models. The simulated network involved multi-modal transport like passenger vehicles, heavy-vehicles and bicycles. The simulation was slightly influenced by the COVID restrictions on September 16, 2020 in Ingolstadt. Although there are sidewalks, pedestrians won't be able to walk over crossings between several streets of the network. Therefore, the traffic scenario can't be used to simulate pedestrians crossing streets because the network is disconnected and we didn't simulate pedestrian movement.

Results The developed TSC models were evaluated by comparing the selected MOEs with the FTSC model and by comparing the selected MOEs among the developed TSC models. When comparing the simulation results of FTSC with the DQN based TSC models, it was evident that the FTSC model outperforms the DQN based TSC models in terms of total waiting time, mean speed and CO2 emissions.

In general, the fixed traffic signal control model had a lower system total waiting time than the developed TSC models that used RL. FTSC outperformed the single DQN based TSC model that used the compressed observation function 99.30555556% of the steps in terms of total waiting time. FTSC outperformed the single DQN based TSC Model that used the default observation function 99.13194444% of the steps in terms of total waiting time. FTSC outperformed the CDQN based TSC model that used the compressed observation function 99.82638889% of the steps in terms of total waiting time. FTSC outperformed the CDQN based TSC model that used the default observation function 99.82638889% of the steps in terms of total waiting time.

Comparing the FTSC model with the developed TSC models reveals that on average, the FTSC model allowed vehicles to travel more quickly than the developed TSC models. FTSC outperformed the single DQN based TSC model that used the compressed observation function 98.09027778% of the steps in terms of mean speed. FTSC outperformed the single DQN based TSC Model that used the default observation function 97.22222222% of the steps in terms of mean speed. FTSC outperformed the CDQN based TSC model that used the compressed observation function 99.30555556% of the steps in terms of mean speed. FTSC outperformed the CDQN based TSC model that used the default observation function 99.30555556% of the steps in terms of mean speed.

The results of our experiment shed insight into the environmental impact of the DQN based TSC models. FTSC outperformed the single DQN based TSC model that used the compressed observation function 89.58333333% of the steps in terms of CO2 emissions. FTSC outperformed the single DQN based TSC Model that used the default observation function 87.5% of the steps in terms of CO2 emissions. FTSC outperformed the CDQN based TSC model that used the compressed observation function 89.93055556% of the steps in terms of CO2 emissions. FTSC outperformed the CDQN based TSC model that used the default observation function 91.66666667% of the steps in terms of CO2 emissions.

In comparison with the single DQN that used the compressed observation function, the single DQN that used the default observation function was better 94.27083333% of the steps in terms of system total waiting time. Moreover, in 59.02777778% of the steps, the single DQN that used the default observation function was better than the single DQN that used the compressed observation function in terms of CO2 emissions. In addition, the single DQN using the default observation function outperformed the single DQN that used the compressed observation function 65.79861111% of the steps in terms of system mean speed.

In terms of system total waiting time, the CDQN that used the compressed observation function outperformed the SDQN that used the compressed observation function 91.14583333% of the steps. Moreover, the CDQN that used the default observation function outperformed the SDQN that used the default observation function 67.88194444% of the steps in terms of system total waiting time. In terms of system mean speed, the SDQN that used the default observation function outperformed the CDQN that used the default observation function 66.31944444% of the steps. The SDQN that used the default observation function produced less total CO2 emissions than the CDQN that used the default observation function 59.375% of the steps.

The median system mean speed of vehicles in the single DQN based TSC model that used the default observation function was the highest among all the TSC models employing RL. Moreover, the model had the least standard deviation among the RL algorithms too and it was the most stable model.

The CDQN based TSC model that used the default observation function had the least system total waiting time among all the RL models and it had the least standard deviation too. Thus, the model was the most stable TSC model that used RL.

Table 1: System Total Waiting Time of TSC Models

	System Total Waiting Time (s)			
Model	Median	SD	Diff. Median	Diff. SD
FTSC	151780	102164.8037	-	-
SDQN Compressed Observation	284412.5	214982.8619	132632.5	112818.0582
SDQN Default Observation	234875	190313.8928	83095	88149.08908
CDQN Compressed Observation	222007.5	200683.2117	70227.5	98518.40796
CDQN Default Observation	197495	183908.169	45715	81743.36529

The total CO2 emissions of vehicles in the single DQN based TSC model that used the default observation function was the least among all the TSC models employing RL. Moreover, the model had the least median CO2 emissions among the RL algorithms. However, the CDQN based TSC model that used the compressed observation function had a lower standard deviation than the single DQN based TSC model that used the default observation function. Therefore, the CDQN based TSC model that used the compressed observation function was more stable than the single DQN based TSC model that used the default observation function.

We trained the single DQN with the compressed observation function for an extra episode and compared the results we obtained with the FTSC model and the results of the same TSC model during episode 1. The FTSC model still outperforms the single DQN based TSC model that uses the compressed observation function in terms of the median system total waiting time, system mean speed and total CO2 emissions. The median system total waiting time in the single DQN based TSC model that uses the compressed observation function improved in episode 2 and at the present rate of the decrease in the median system total waiting time per episode, assuming a negative linear trend, only 4 more episodes will be necessary to obtain a median system total waiting time in par with the FTSC model. The median system mean speed also improved in episode 2, but the total CO2 emissions became worse. Episode 2 involved a decrease in the standard deviation of the median system total waiting time, median system mean speed and total CO2 emissions. Therefore, the single DQN based TSC model with the compressed observation function is becoming more stable in episode 2.

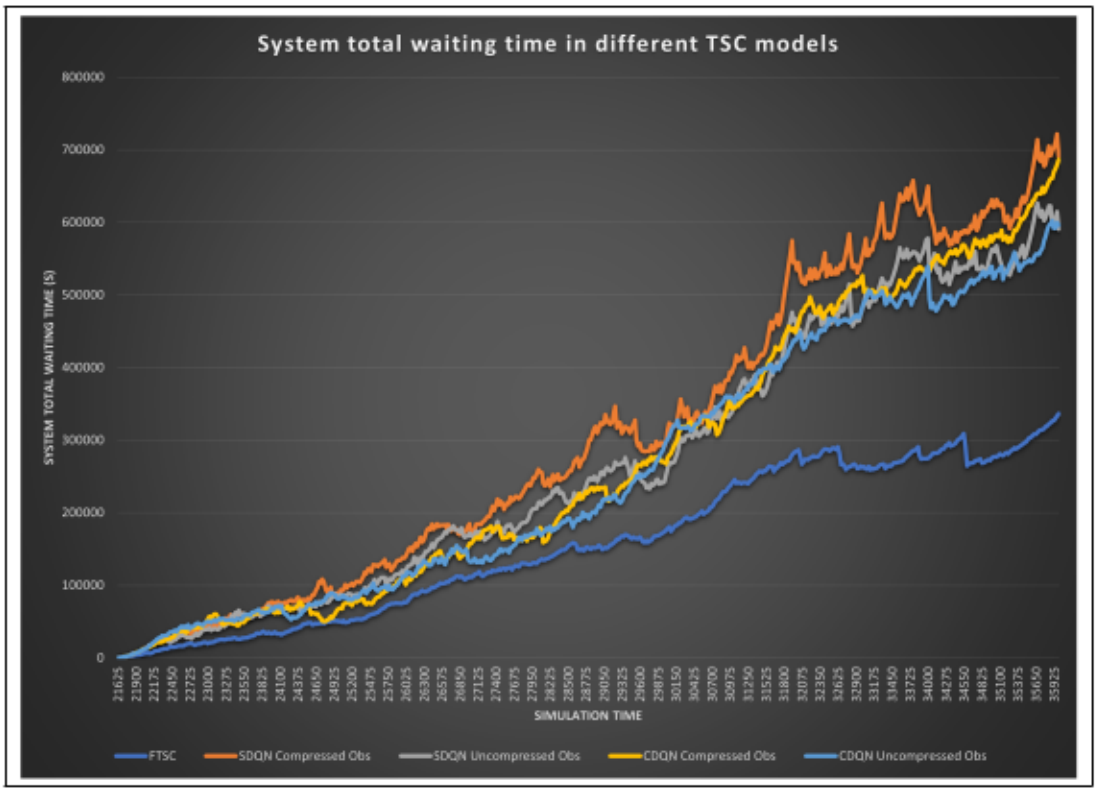

Figure 2: System total waiting time in different TSC models in 1 episode

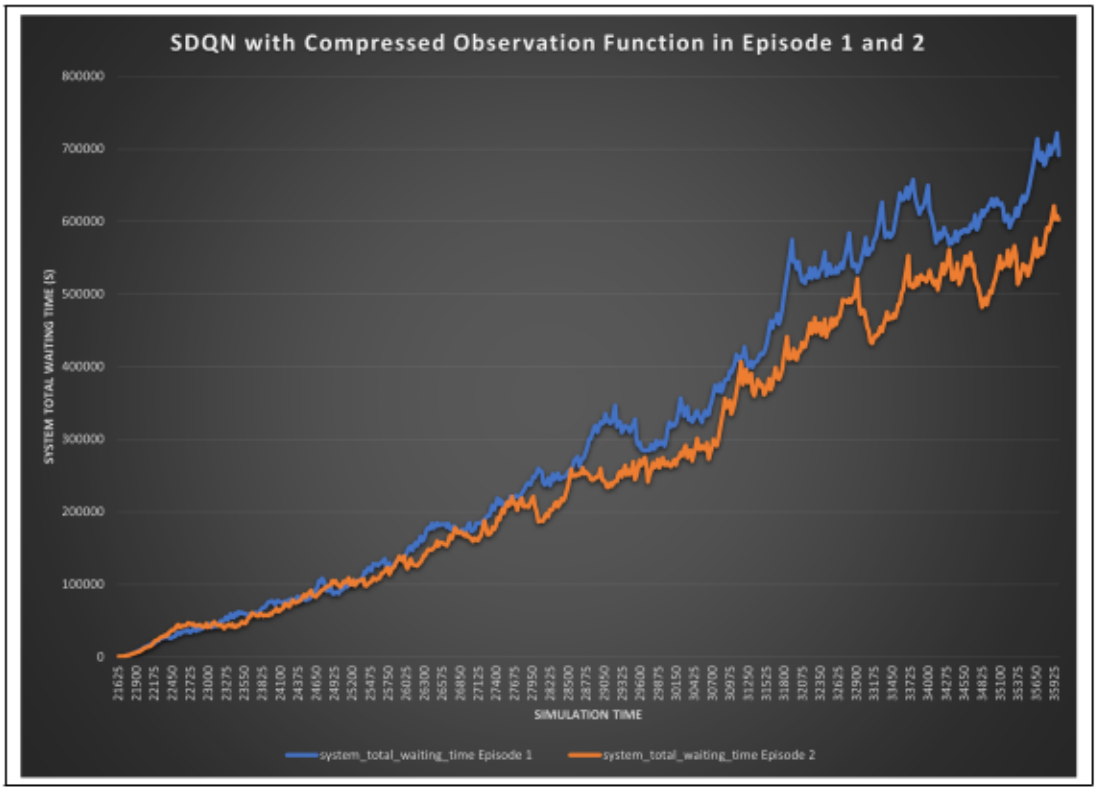

Figure 3: System total waiting time in episode 1 and episode 2 of training the SDQN with the compressed observation function

5 Conclusions and future studies

In our research paper, we focused on researching a solution to the ever-increasing issue of traffic congestion in urban areas using a DQN based RL algorithm. We selected a DQN based RL algorithm because it had been extensively used in previous studies to optimize traffic flow. To conduct our experments, we created a microscopic simulation environment using SUMO and PettingZoo and used TraCI to interface with the environment. Subsequently, we conducted RL experiments on different types of DQN based TSC models using a real-world traffic scenario modelled on the traffic conditions of Wednesday, September 16th, 2020 in Ingolstadt, Germany. Excluding the single DQN based TSC model that used the compressed observation function, each RL experiment was carried out for only a single episode because of hardware resource limitations. The single DQN based TSC model that used the compressed observation function was trained for two episodes. The performance of the DQN based traffic signal control models was overshadowed by FTSC in terms of system total waiting time, system mean speed and total CO2 emissions. However, the single DQN based TSC model exhibited an improvement in the median system total waiting time in episode 2.

Future work could be directed at eliminating biases in our research study. Our research was restricted by hardware limitations and we could not run the DQN experiments over more than 2 episodes or run the experiments with different random seeds. Therefore, we were not able to test hypotheses using a Wilcoxon signed rank test. When simulating the traffic scenario, we increased the reaction time of the drivers and the drivers didn't have enough time to slow down before reaching a vehicle the drivers were following. Therefore, the simulation involved a high number of collisions which wasn't consistent with the real-world traffic scenario in Ingolstadt. This issue could be resolved by keeping the traffic scenario parameter intact and running the experiment again. Another issue related to our work was that the traffic scenario we simulated was influenced by the COVID pandemic. Traffic flow may have been affected by changes in preferences of drivers caused by restrictions in public places during the pandemic. Further research may be done in this field to identify ways to mitigate the influence of the COVID pandemic on the traffic simulation. Moreover, research still needs to be done to incorporate pedestrians and public transport into the traffic scenario and to improve bicycle routing. Also, the structure of the DQNs we used in our experiments could use convolution layers to improve the performance of the RL algorithm.

References

[1] S. Park, E. Han, S. Park, H. Jeong, and I. Yun, "Deep Q-network-based traffic signal control models," PLOS ONE, vol. 16, no. 9, p. e0256405, Sep. 2021, doi: https://doi.org/10.1371/journal.pone.0256405.

[2] J. Ault and G. Sharon, "RL Benchmarks for Traffic Signal Control." Accessed: May 06, 2023. [Online]. Available: https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/file/f0935e4cd5920aa6c7c996a5ee53a70f-Paper-round1.pdf

[3] J. Lee, J. Chung, and K. Sohn, "RL for Joint Control of Traffic Signals in a Transportation Network," IEEE Transactions on Vehicular Technology, vol. 69, no. 2, pp. 1375–1387, Feb. 2020, doi: https://doi.org/10.1109/TVT.2019.2962514.

[4] S. Liu, G. Wu, and M. Barth, "A Complete State Transition-Based Traffic Signal Control Using Deep RL," IEEE Xplore, Apr. 01, 2022. https://ieeexplore.ieee.org/document/9794168/ (accessed May 06, 2023).

[5] M. Harth, M. Langer, and K. Bogenberger. "Automated Calibration of Traffic Demand and Traffic Lights in SUMO Using Real-World Observations". SUMO Conference Proceedings, vol. 2, June 2022, pp. 133-48, doi:10.52825/scp.v2i.120.

[6] M. Langer, M. Harth, L. Preitschaft, R. Kates, and K. Bogenberger, "Calibration and Assessment of Urban Microscopic Traffic Simulation as an Environment for Testing of Automated Driving," IEEE Xplore, Sep. 01, 2021. https://ieeexplore.ieee.org/document/9564743 (accessed May 06, 2023).