Smart Traffic Light System Using Machine Learning

Mohamad Belal Natafgi
Department of Electrical and Computer Engineering
American University of Beirut
Beirut, Lebanon
mmn32@mail.aub.edu

Mohamad Osman
Department of Electrical and Computer Engineering
American University of Beirut
Beirut, Lebanon
mno04@mail.aub.edu

Asser Sleiman Haidar
Department of Electrical and Computer Engineering
American University of Beirut
Beirut, Lebanon
ars17@mail.aub.edu

Lama Hamandi
Department of Electrical and Computer Engineering
American University of Beirut
Beirut, Lebanon
lh13@aub.edu.lb

***Abstract*—In Lebanon, traffic problems are a major concern for the population. The rising number of cars that exceeds the capacity of the roads, the inefficiency of public transportation infrastructures and the non-adaptive traffic light systems are contributors to the traffic crisis. Most roads in Lebanon suffer from traffic jams due to the traditional static green and red times allocations that are inconsiderate to the current state of the traffic. A solution to this problem is a system that adapts to the variations of the traffic dynamically and updates the traffic signal phases accordingly. In this paper, an adaptive traffic light system is implemented using reinforcement learning and tested using real data from Lebanese traffic. For training and testing the system, a software simulation tool is used. This tool can simulate the traffic intersection and allows the neural network to interact with it. Compared with the actual traffic light system, the proposed model displayed a reduction in average queue lengths by 62.82% and in average queuing time by 56.37%.**

Keywords—machine learning, reinforcement learning, adaptive traffic light, single-agent model, multi-agent model, traffic simulation, neural network

I. INTRODUCTION

In Lebanon, traffic congestion is a recurring daily problem that affects all classes of society with inhabitants of cities being the most severely affected. According to Urban Transport Development Project, a study showed that people in Lebanon spend around 720 hours annually in their cars stuck in traffic jams [1]. Moreover, traffic jams lead to numerous issues such as: tardiness of citizens to their duties due to wasted time spent on the road, increase in the consumption of fuel which leads to a higher pollution footprint and higher transportation costs, and increase in the rate of accidents and crashes. While there are many causes for such issues, such as the rising number of vehicles and the inefficient public transportation and infrastructure, this paper is focusing on the problems caused by the traffic light systems. The majority of the traffic lights implemented in Lebanon are pre-timed which means that their green and red timers are preconfigured to a static value causing congestion in most roads. To remedy the inefficiency of the current traffic light systems, an adaptive traffic light system is proposed. This system takes into consideration the state of the traffic at a given time before deciding which phase should be applied to the intersection.

Section II highlights the actual implemented traffic light systems as well as adaptive traffic light systems described in the literature, Section III discusses the proposed system and its implementation, Section IV describes the training process of the system, Section V compares the results of the proposed system with the results of the current system, and Section VI shows future work and how the system could be expanded further.

II. LITERATURE REVIEW

According to the Traffic Management Center (TMC) in Lebanon, there are three types of traffic light systems: pre-timed, semi-actuated and fully-actuated. Pre-timed systems use traffic light signals that have fixed phase timers. A traffic intersection phase is a specific combination of green and red lights that allow some lanes in the traffic intersection passage while denying it to others to prevent accidents. Semi- actuated systems use traffic lights that are implemented in the following way: whenever a vehicle reaches the intersection, the light is turned into green and then back into red when the vehicle crosses that intersection. Note that in the semi-actuated systems, only one road is given passage upon the arrival of a vehicle whereas all other roads of the intersection are pre-timed. In the fully-actuated systems, all roads of the intersection are given passage similarly to the semi-actuated system. Since the sensors used in the semi/fully-actuated systems to extract data are not reliable and need constant maintenance, 90% of the systems are pre-timed. These systems, however, have some disadvantages. For instance, in pre-timed systems, the performance is very poor when the traffic is irregular and therefore it requires human intervention. In fully-actuated systems, heavy loaded roads cannot be handled. Finally, since the semi-actuated system employs techniques from fully-actuated and pre-timed systems, it has the disadvantages of both.

Adaptive traffic lights using reinforcement learning is a subject that has undergone heavy research. A state description, which is how the state of the environment is communicated to the agent, needs to be defined in a reinforcement learning algorithm. The agent in reinforcement learning is the neural network. The majority of systems that were proposed used the length of the queues in lanes to describe the current state of the traffic light [2, 3, 4]. Section III lists the issues that can arise when using only the queue lengths to describe the state of the traffic intersection.

In another research, a discrete traffic state encoding (DTSE) was proposed as a method to describe the state of the environment [5]. DTSE attempts to discretize the state

978-1-5386-4509-0/18/$31.00 ©2018 IEEE

of the environment to better represent it by dividing the lanes into cells and measuring vehicle presence and velocity vectors for each cell. DTSE requires measurements for individual vehicles in the lane. The velocity of every vehicle as well as its location is required to accurately describe the state of the traffic intersection. The research mentions that current technological advancements might allow the extraction of such measurement however that claim was not confirmed through testing.

In an adaptive algorithm, the agent receives a reward upon completing an action. The reward indicates to the agent how correct its action was. The reward that was used in previous systems was based on either the change in queue lengths [3] or the change in delays [6]. Using a reward that is based on one component might result in undesired behavior which will be discussed in Section III.

The systems proposed in the literature relied on a multi-agent model [2, 4]. The systems attempted to optimize the behavior of several adjacent intersections working concurrently by assigning an agent to each intersection. However, a multi-agent model means that each intersection changes its phase according to what it deems optimal for it individually. The behavior of the group of agents might not be optimized for all the traffic lights. The multi-agent model eliminates an advantage of the pre-timed traffic light which is the coordination of a green wave. A green wave is the event where the driver of a vehicle traverses several traffic intersections in sequence while encountering a green light at each intersection. In a multi-agent model, the coordination of a green wave is not possible since every intersection is governed independently by an agent.

III. Proposed System

A. Overall Design

The proposed system is an adaptive traffic light that is based on a neural network trained using reinforcement learning. The traffic intersection is assumed to be isolated. An isolated traffic intersection is sufficiently distanced from other nearby traffic intersections so that a traffic congestion in other traffic intersections does not affect it. The overall flow of the proposed system is illustrated in Fig. 1. The sensors installed on the roads detect passing vehicles. The controller uses the sensors data to calculate the queue lengths and time delays in those roads as will be explained in section III.B. Then, it saves the count of vehicles that have been detected in a database every hour. The data in the database can be later reviewed to study the intersection and to help simulate the environment for training the neural network. The queue lengths and the queuing times, i.e. the maximum wait times of vehicles on each lane, are fed as input to the model as well as the current phase of the traffic light. The output of the model will determine the optimal phase to set the traffic light to.

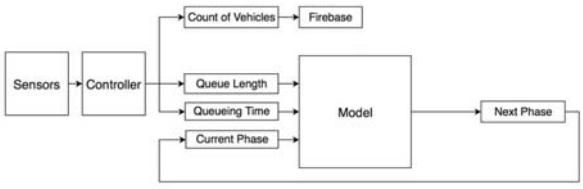

Fig. 1. Proposed System Design.

B. Sensors and Controller

It is proposed to install ultrasonic sensors on the incoming roads in such a way that each lane in those roads has a dedicated sensor for vehicle detection. The placement of the sensors is shown in Fig. 2.

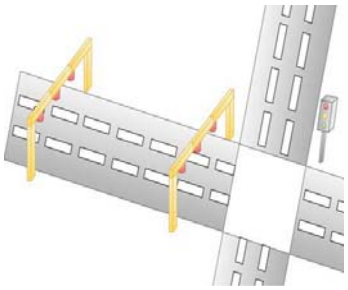

Fig. 2. Placement of proposed system's sensors on a road.

When the ultrasonic waves collide with an obstacle, they are reflected to the receiver of the sensor. The distance between the transmitter and the obstacle is then recorded. When a vehicle passes, the wave travels a shorter distance and thus the vehicle is detected. The controller is responsible for interpreting the sensor measurements and mapping them to vehicle detection, queue length and queueing time measurements. For each road, the sensors are installed at 2 locations: at the ending of the road near the intersection, called exit point, and at a location earlier in the road, called the entry point, where representative queue length can be obtained for that road. It is assumed that a vehicle cannot change its lane between the entry point and exit point. A queue data structure is used in conjunction with the sensors to determine the queue lengths and queuing times for each lane. Each vehicle detected by the entry point sensors is added to the queue data structure and its entry time, queue length at entry, and velocity are recorded. When it reaches and passes the exit point, it is removed from the queue. The velocity is calculated by the average length of the car divided by the time it took to cross the sensor. The queue length is determined by the number of elements in the queue data structure. The formula for calculating the queuing time is the following:

$$t-(d-q\_entry)/v \quad (1)$$

where t is the time the vehicle spends between the two points, determined by the entry and exit times. Given that it retains its velocity v, the time the vehicle spends in the queue can be determined; d is the distance between the entry and exit points and q_entry is the length of the queue at entry. The time the vehicle spends between the two points can be divided into two components: the first component is the time it takes to reach the queue and the second is the time it waits in the queue. The travel time to reach the queue should be deducted from the total time spent between the points to determine the queuing time. The travel time to reach the queue can be determined by:

$$(d-q\_entry)/v \quad (2)$$

C. Model

Reinforcement learning [7] is a training process that tunes a neural network to improve its prformance. Reinforcement learning functions by letting an agent interact with an environment and assigning to the agent a reward based on the action that it made. The agent attempts to take actions that will maximize the reward that it receives. The type of reinforcement learning used to train the network is Q-learning. In Q-learning the agent takes into account both immediate and future rewards that it will receive after making an action. The output of the neural network in Q-learning is the reward that it expects to receive for each possible action. The agent in this study is the neural network. The environment will be a traffic intersection that the agent interacts with. The action that the agent is allowed to make is changing the current phase of the traffic light. The output of the neural network is the reward that it expects to receive for each phase that it can choose to assign to the traffic intersection. The phase that will yield the maximum reward is chosen.

Before training, the behavior of the neural network is random and very distant from the required behavior, therefore it cannot be trained directly on a real-life traffic intersection. The random assignment of the weights is required by the optimizer Gradient Descent. Gradient Descent is a greedy algorithm that attempts to reach the nearest local minimum of the error function. If we choose fixed initial weights, we introduce a bias and the Gradient Descent algorithm will then always reach the same local minimum. Choosing random initial weights allows us to train the network multiple times and choose the best local minimum reached, thus eliminating the bias. During the training phase the environment is a traffic intersection that is replicated using a traffic simulator. After the training is over the environment that the neural network interacts with will be a real traffic intersection. The state of the environment needs to be communicated to the agent so that it can take an appropriate action. The state of the traffic intersection is a vector that is given as input to the neural network. The values used to indicate the current state of the traffic intersection are the following: the current phase of the traffic intersection, the length of the queue in meters on each lane in all incoming roads of the intersection, and the maximum time in seconds a vehicle has waited in each lane in the intersection which is the waiting time of the first vehicle in the queue since it is the vehicle that has waited the most.

D. Reward

Following an action, the agent receives a reward that reflects the result of the action on the environment. The reward that is given to the neural network reflects how it improved or worsened the state of traffic. A decrease in the length of a queue in a lane is rewarded positively and a lengthening of the queue is rewarded negatively. On the other hand, a decrease in the maximum wait time in a lane is rewarded positively while an increase is rewarded negatively. Combining a reward that is based on two components allows the agent to avoid undesired behavior that can occur when using only one parameter for the reward. If the reward given is solely based on the decrease in queue length and the wait time is not taken into consideration, a situation can occur where one vehicle that is waiting on a lane can wait indefinitely and vehicles on other crowded lanes are allowed passage. Alternatively, if only the waiting time is used to calculate the reward, the agent will prioritize lanes where vehicles have waited for a longer period of time regardless of the number of vehicles in each lane which is undesirable. The reward that the neural network receives after changing the phase is the sum of the rewards that are calculated for each lane. The reward for each lane is calculated by comparing the length of the queue and the queuing time for that lane both before and after changing the phase.

E. Parameters to Tune

The parameters that can be tuned to get a better performance of the model are: the discount factor, the exploration rate, and the phase duration.

In Q-learning, the agent takes into account future and immediate rewards that it can receive when calculating the reward associated with an action. The immediate reward is the reward that the agent receives upon completing an action. The future reward is the reward that it can expect to receive for the action that follows the current action. The discount factor is a number that ranges between 0 and 1 that modifies the weight that is given to future rewards. A discount factor of 0 will result in an agent that only makes actions that will yield high immediate rewards, while a discount factor of 1 will result in an agent that assigns as much importance to future rewards as immediate rewards.

The exploration rate is a number that ranges between 0 and 1 that indicates how often the agent explores. An exploration is defined as forcing the agent to make a random action and observe the reward for that action. In this work, exploration means that the agent sets a random phase to the traffic intersection regardless what the current state of traffic is. The phase duration is the time in seconds that the agent waits between two actions. The neural network checks the current state of the traffic light and chooses the phases that it expects will yield the highest reward. The agent then waits the phase duration and then it receives the reward associated with the last action and chooses a new phase for the traffic light.

IV. Training The Model

To test the proposed system, the machine-learning model should be trained first using traffic simulation software. The training period of the model was accomplished using the traffic simulator SUMO [9]. SUMO is a free and open source simulation software for traffic systems. It provides the developer with complete control over the environment such as the vehicles, traffic light signal phases, network shape and infrastructure, etc. through an API called TRACI. Moreover, it provides many types of simulation data that are saved after the simulation has ended. The variables of interest are the queue length and queueing time which, fortunately, can be provided by SUMO. The fundamental parts for making a simulation are: building the network and generating the route file. The environment that the model interacts with is the network that was designed in NETEDIT, which is the visual network editor used by SUMO, the traffic light signals, and the vehicles in that network. To simulate real-life scenarios, the case study considered is the intersection found in Ain Mreisse, Beirut, Lebanon. A network similar to the real intersection is drawn using NETEDIT as shown in Fig. 3. Moreover, data is obtained from the Traffic

Management Center (TMC) in Beirut for that intersection for a period of two months, which was used to generate the route file that SUMO needs to simulate cars circulating in the network.

Fig. 3. SUMO in action using the network drawn similar to Ain Mreisse's intersection.

The lanes that are considered in this intersection are only the incoming lanes in the network. The lanes are labeled from DT01 to DT08. Moving clockwise to label the lanes, the first incoming road is the one on the right which holds lanes DT01, DT02, DT03 and DT04. The second incoming road is the one on the bottom which holds lanes DT05 and DT06. Finally, the last incoming road, which is located on the left, holds the lanes DT07 and DT08. There are three possible phases in this intersection. The output of the model is the number indicating the phase that the model should choose depending on the state of the traffic. Fig. 4 shows the three possible phases where green lines indicate lanes that are allowed passage and red lines indicate lanes that are prohibited passage during that phase. After every phase duration, the model chooses the next phase. On the contrary, a static traffic light cycles between those phases disregarding the state of the traffic.

.

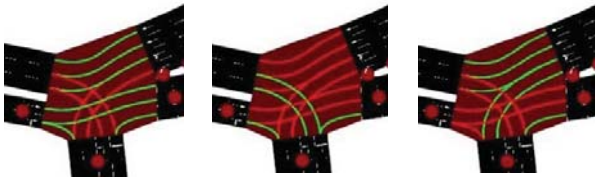

Fig. 4. A possible set of phases for the intersection.

SUMO runs in simulation steps, where each step is worth 1 second in the simulation environment. The phase duration, therefore, determines how many simulation steps SUMO can run before letting the model take a new action. A phase duration of 10 seconds means that SUMO runs 10 simulation steps before giving the model permission to predict the next optimal phase of the traffic light. For training, a phase duration of 10 seconds is used. In section V, the effect of phase duration on the queue lengths and queueing times of the vehicles is analyzed.

To make everything work, multiple modules were developed. The first module is responsible for generating the route file, which represents the distribution of vehicle across the network, relative to the input file, which is the file obtained from TMC. Moreover, live simulation data were extracted from SUMO corresponding to the current state of the traffic, i.e. the queue length and queueing time for each inflow lane in the network. Therefore, a web server was developed to listen for data streams from the simulator. Finally, all the running threads were synchronized: SUMO's thread running the simulation, the webserver's thread listening for live data and parsing it, and finally the machine-learning model's thread responsible for training and predicting the optimal phase given the current state of the traffic.

The training was done by simulating 2 weeks' worth of data. The data, shown in Table I that the proposed system works on, is formatted as a table where the first column is the index of the row representing the date followed by the hour. The remaining columns, titled as the names of the lanes in the network, represent the count of vehicles that have passed in that intersection on that lane in real life during that hour. Being formatted this way allowed us to programmatically generate the route file which simulates real-life scenarios as long as the columns' names match the names of the inflow lanes in NETEDIT. The rest of the data was available to be used for testing the model. To make things simpler for the user of the system, a graphical user interface was developed to allow the user to input a file formatted as explained previously, and either test or train the model. Training the model through software simulation is indeed needed since the model should be mature enough before it is applied on the real intersection.

TABLE I. SAMPLE OF DATA GIVEN BY TMC.

HOUR	FLOW							
	DT01	*DT02*	*DT03*	*DT04*	*DT05*	*DT06*	*DT07*	*DT08*
01/12/17 11	205	1012	221	256	92	254	1079	747
01/12/17 12	810	1175	251	297	115	292	1019	611
01/12/17 13	1184	1246	238	307	42	224	1056	543
01/12/17 14	1034	981	149	113	93	125	1009	456
01/12/17 15	287	973	207	225	67	183	1040	250
01/12/17 16	299	1230	298	426	73	233	1120	413
01/12/17 17	494	1933	385	479	87	262	1369	728
01/12/17 18	561	1694	443	465	60	249	1349	505
01/12/17 19	210	1264	353	482	50	222	1244	222
01/12/17 20	278	1324	353	483	40	217	1122	482

V. TESTING AND RESULTS

The first series of tests that were conducted aimed to determine what phase duration should be used. The metrics that were used to measure the performance of the model are the maximum length of the queue of the roads, the maximum time delay of the roads, the average length of the queues of all roads, and the average time delays of all roads. A smaller value of the phase duration means that the model can adapt faster to changes in the state of the traffic. However, the phase duration cannot be too small since the cars would not have enough time to cross the traffic light due to a very short green light. The performance of the model is measured on the same route file but using different phase durations. As expected, the results of the

tests, presented in Fig. 5, showed that the performance worsened when the phase duration is increased, i.e. the queue lengths and queueing time increased.

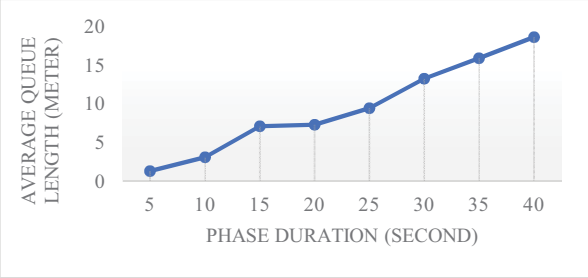

Fig. 5. Effect of phase duration on average queue length of roads

The performance of the proposed model is compared to the performance of the static traffic light that the TMC uses for most intersections. The site of the traffic light in Ain Mreisse is considered where the sequence of phases that the traffic light follows as well as the duration of each phase are determined manually. The network in SUMO is configured so that it follows the same phase sequence and phase duration as the static traffic light. To compare the performance of the static traffic light to the proposed model, two simulations are run with the same route file. In the first simulation the static configuration was used. In the second, the model adaptively modifies the phase of the traffic light according to the proposed model.

The six test scenarios proposed, presented in Table II, were run on two days. The first day was Monday December 4 2017 and the second was Sunday December 3 2017. Monday represents a working day, during which the traffic light is subjected to heavy load. Sunday represents a holiday which means that the traffic load is lighter than other days. Four scenarios consisted of one-hour simulation at different times of day: 7, 12, 17 and 22. These scenarios measured the performance of the system for different traffic loads. A fifth scenario was a 24-hour simulation that tested the performance of the system over a full day. The sixth scenario was that of a queue accumulation, the phase is fixed for 200 seconds to allow queues to accumulate in the network and the system is required to eliminate the queues as promptly as possible.

The proposed model proved to be more efficient than the current system in all six scenarios with noticeable improvements in the metrics. The average improvements of the maximum queue lengths, average queue lengths, maximum time delays and average time delays were 68.25%, 62.82%, 60.19% and 56.37% respectively. The most notable improvements were observed in simulations that were run at 5pm. Those improvements were 89.64%, 81.99, 75.65 and 67.44 respectively for the above-mentioned metrics. When studying the data, it is found that 5pm was the time at which the traffic light witnessed the most traffic. This shows that the adaptive model performs noticeably better when the traffic load is high. This model also showed great improvements in simulations over a full day. This is an important finding since it shows that the model performs better than the static traffic light in a realistic scenario. Therefore, should the system be implemented on a traffic light, the daily performance of the model is expected to be better than the performance of a static traffic light.

The performance of the proposed system was compared to the performance of two other proposed systems that were implemented using Q-learning. The first proposed system [10] defined the state of the environment using a grid. The intersection is divided into cells and the network is fed the grid describing the environment. This system showed an improvement of 25.7% in waiting time over the fixed time traffic lights. Not only is this system outperformed by the system proposed in this paper which showed an average improvement in time delay of 56.37%, but also it was not tested on real-life scenarios and traffic data as it was in the latter. The second proposed system [11] used real-life traffic data to train and measure the performance of the model. The data that was used to train

TABLE II. COMPARISON BETWEEN THE CURRENT SYSTEM AND THE PROPOSED SYSTEM

Time	# Vehicles Entering Intersection	Static System				Suggested Adaptive System				Improvement			
		Max Queue Length in a Road (meters)	*Average Queue Length in Roads (meters)*	*Max Time Delay in a Road (seconds)*	*Average Time Delay in Roads (seconds)*	*Max Queue Length in a Road (meters)*	*Average Queue Length in Roads (meters)*	*Max Time Delay in a Road (seconds)*	*Average Time Delay in Roads (seconds)*	*Max Queue Length in a Road (%)*	*Average Queue Length in Roads (%)*	*Max Time Delay in a Road (%)*	*Average Time Delay in Roads (%)*
Monday 04/12/2017 Heavy Day Load													
07	3265	8.65	7.23	15.7	11.35	4.87	3.64	5.68	4.23	43.70	49.65	63.82	62.73
12	4505	12.04	9.75	18.57	12.4	5.35	4.8	6.31	4.67	55.56	50.77	66.02	62.34
17	6168	113.07	42.64	28.67	14.99	11.71	7.68	6.98	4.88	89.64	81.99	75.65	67.44
22	3340	12.62	7.9	16.32	10.89	4.16	3.26	9.29	5.53	67.04	58.73	43.08	49.22
Full Day	73195	43.97	18.38	16.97	11.11	5.55	4.08	6.28	4.24	87.38	77.80	62.99	61.84
17 w/ Accumulation	6168	116.25	46.37	30.97	17.15	21.69	14.54	11.68	8.17	81.34	68.64	62.29	52.36
Sunday 03/12/2017 Light Day Load													
07	1053	2	1.7	7.4	4.93	1.13	0.98	2.54	1.72	43.50	42.35	65.68	65.11
12	3745	8.5	7.29	16.12	11.3	3.86	3.4	9.14	5.69	54.59	53.36	43.30	49.65
17	4448	112.11	40.54	28.97	13.73	12.11	8.31	10.97	6.64	89.20	79.50	62.13	51.64
22	3424	9.14	7.04	14.6	11.06	5.52	3.37	4.21	3.48	39.61	52.13	71.16	68.54
Full Day	62567	49.51	19.33	17.73	10.47	4.54	3.6	6.7	4.19	90.83	81.38	62.21	59.98
17 w/ Accumulation	4448	117.3	44.45	31.63	15.76	27.43	18.88	17.73	11.72	76.62	57.53	43.95	25.63
										Average of Improvements			
										68.25	62.82	60.19	56.37

the model was retrieved from traffic surveillance cameras. The improvements that this system displayed over the fixed-time system was 47.61% reduction in queue length with the queue length being 10.238 meters in that system and the queue length in the fixed-time system being 19.542 meters. Moreover, the improvement brought by this system over the fixed-time system in terms of time delay averaged in 19.16% where the average time delay of the proposed system was 2.73 seconds and the average time delay of the fixed-time system was 3.377 seconds. This system [11] is outperformed by the system proposed in this paper in both queue length (62.82%) and time delay (56.37%) improvements.

VI. Future Work

The proposed system is a system that assumes that the traffic intersection is isolated. The behavior of the system might not be optimal in a case where the intersection is not isolated. The inputs to the neural network do not include the lengths of queues and queuing times in lanes in outgoing roads. Therefore, in the case where outgoing roads are in a congested state, the vehicles leaving the intersection will have to wait in a queue in the outgoing road. In such a case the performance of this system is suboptimal.

The proposed system can be modified so that it can be applied to non-isolated intersections. Currently the inputs to the neural network are the queuing times and the queue lengths for one intersection, and the outputs are the rewards associated with each phase possible for the traffic intersection. The proposed model can be expanded to interact with several adjacent intersections at the same time. The environment will consist of several intersections that are located in a specific region instead of one isolated intersection. The model will be given the length of the queues and the queuing times for all the lanes in all the intersections in the region. The neural network currently chooses a phase for each intersection; therefore, the expanded neural network will choose a combination of phases for the group of intersections that it manages. The output of the network will be the rewards associated with each combination of phases instead of the reward of one phase. The size of the input and output of the neural network will increase when the number of intersections increases. The increase in the size of the input is linear while the increase in the size of the output is exponential. The neural network needs to be modified to account for the increase in size of the input and output and the training period required will be longer.

Acknowledgment

We would like to acknowledge the great assistance of the traffic management center in Lebanon and their respectable welcoming. They provided us with information about the traffic in Lebanon. They also provided us with the traffic data that we used to simulate the environment and train our neural network.

References

[1] T. A. W. http://www.thearabweekly.com/, “Traffic congestion adds to Lebanon,” The Arab Weekly. [Online]. Available: http://www.thearabweekly.com/Society/7233/Traffic-congestionadds-to-Lebanon%E2%80%99s-many-woes. [Accessed: 06-Dec-2017].

[2] M. Wieringet., “Multi-agent reinforcement learning for traffic lightcontrol,” inICML, 2000, pp. 1151–1158.

[3] Y. K. Chin, N. Bolong, A. Kiring, S. S. Yang, and K. T. K. Teo, “Q-learning based traffic optimization in management of signal timing plan,”International Journal of Simulation, Systems, Science & Technology,vol. 12, no. 3, pp. 29–35, 2011.

[4] I. Arel, C. Liu, T. Urbanik, and A. Kohls, “Reinforcement learning-basedmulti-agent system for network traffic signal control,”IET Intelligent Transport Systems, vol. 4, no. 2, pp. 128–135, 2010.

[5] W. Genders and S. Razavi, “Using Deep Reinforcement Learning Agent for Traffic Signal Control,” arXiv:1611.01142, vol. 1, Nov. 2016.

[6] S. El-Tantawy and B. Abdulhai, “Multi-Agent Reinforcement Learning for Integrated Network of Adaptive Traffic Signal Controllers (MARLIN-ATSC),” 2012 15th International IEEE Conference on Intelligent Transportation Systems, Sep. 2013.

[7] S. Leishman, “Guest Post (Part I): Demystifying Deep Reinforcement Learning,” Intel AI, 17-Aug-2018. [Online]. Available: https://ai.intel.com/demystifying-deep-reinforcement-learning/ [Accessed 9 Mar. 2018].

[8] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, “Human-level control through deep reinforcement learning,” Nature, vol. 518, no. 7540, pp. 529–533, Feb. 2015.

[9] Sumo.sourceforge.net. (2018). SUMO_User_Documentation - SUMO - Simulation of Urban Mobility. [online] Available at: http://sumo.sourceforge.net/userdoc/ [Accessed 8 Feb. 2018].

[10] X. Liang, X. Du, G. Wang, and Z. Han, “Deep Reinforcement Learning for Traffic Light Control in Vehicular Networks,” arXiv:1803.11115, vol. 1, Mar. 2018.

[11] H. Wei, G. Zheng, H. Yao, and Z. Li, “IntelliLight,” Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining - KDD 18, 2018.