2013 IEEE International Conference on Systems, Man, and Cybernetics

Intelligent Traffic Light Control of Isolated Intersections Using Machine Learning Methods

Sahar Araghi, Abbas Khosravi, Michael Johnstone, Doug Creighton
Centre for Intelligent Systems Research (CISR)
Deakin University, Geelong, Vic, 3216, Australia
{saraghi, abbas.khosravi, michael.johnstone, douglas.creighton}@deakin.edu.au

Abstract—**Traffic congestion is one of the major problems in modern cities. This study applies machine learning methods to determine green times in order to minimize in an isolated intersection. Q-learning and neural networks are applied here to set signal light times and minimize total delays. It is assumed that an intersection behaves in a similar fashion to an intelligent agent learning how to set green times in each cycle based on traffic information. Here, a comparison between Q-learning and neural network is presented. In Q-learning, considering continuous green time requires a large state space, making the learning process practically impossible. In contrast to Q-learning methods, the neural network model can easily set the appropriate green time to fit the traffic demand. The performance of the proposed neural network is compared with two traditional alternatives for controlling traffic lights. Simulation results indicate that the application of the proposed method greatly reduces the total delay in the network compared to the alternative methods.**

Keywords-component; machine learning; Q-learning; neural network; traffic controlling; single intersection

I. INTRODUCTION

Higher levels of urbanization and increases in the number of vehicles have emphasized the need for an efficient transportation system. As fully developed cities have a limited ability to reconstruct their traffic network, providing real time control is a necessary part of modern traffic control systems. Increasing the intersection capacity, decreasing delays, and guaranteeing the safety of people and vehicles are some of the goals of the signal control. This is also effective in fuel usage and emission by decreasing the stop and delay time.

Different solutions have been proposed to solve the problem of traffic congestion in urban cities. Many traffic signals are controlled using fixed-time methods [1]. The fixed-time method manages traffic lights based on previously surveyed traffic flows for different times of day and different days of the year. In this method, a predetermined time for each intersection's traffic light is considered. This method has a considerable effect in reducing travelling delay time. However, such fixed-time systems cannot be expected to cope with traffic conditions that differ from those prevailing when the intersection was surveyed. As a result, a more flexible approach is necessary [2].

The problems of most fixed-time systems show that a more responsive approach to changing traffic conditions is needed. One approach to this problem is the Sydney Coordinated Adaptive Traffic System (SCATS) [3]. It is an adaptive system which uses detectors for recognizing current traffic amounts at an intersection and extending or terminating green time based this information. This system requires no costly pre-calculation of signal timing plans. Additionally, SCATS is self-calibrating, automatically adjusting to changing traffic patterns over time [2].

Artificial intelligent and machine learning techniques have also played an important role in improving the existing traffic control technology [4]. For example, game theory [5], fuzzy logic [2], [6], neural network [2], [7], [8], and Q-learning [9], [10], [11], [12], [13], [14] are some of the learning methods presented as effective methods in reducing traffic congestion and improving traffic flow.

In this paper, a neural network is developed for controlling signal lights for an isolated intersection. The objective is to reduce the average delay time at the intersection. The reason for using a neural network is that it is capable of generating a different range of green times for adjusting the appropriate time based on traffic demands present at each traffic phase. Q-learning is a reinforcement learning method that learns the optimal strategy for signal timing without any need to model the system. Due to the deficiency of tabular Q-learning in large state spaces, we avoid various ranges of green times in applying this method.

Here, we propose a neural network method for managing traffic in a single intersection and then we present a comparison of it with Abdoos et al.'s [14] method which is also applied for a single intersection with the same conditions. In [14], Q-learning is used for controlling the traffic and fixed cycle time is considered during learning. By considering a different range of green times and avoiding the limitation of a fixed cycle time in our neural network method, better results are achieved. Finally, we implement the proposed method, Abdoos et al.'s method and fixed-time method and present a comparison. The results of experiments illustrate an improvement in reducing the average delay time of the vehicles during simulation time.

The rest of this paper is organized as follows. The related backgrounds are discussed in section II. In section III, related reinforcement learning and neural network methods for controlling traffic signal lights are presented. Our proposed method in presented in section IV. Experimental results and discussion is presented in section V and finally conclusions in section VI.

II. Background

A. Reinforcement Learning and Q-learning

Reinforcement learning is a subfield of machine learning useful in unknown environments. This learning method exploits environmental feedback as reward or punishment Fig.1. In reinforcement learning, the learning algorithm is simple and an agent can learn an optimal policy without being able to predict the impact of its actions, its immediate payoffs, or planning ahead [15].

Q-Learning is an incremental reinforcement learning method that does not need a model of the environment and can be used online [16]. The Q-Learning algorithm stores the expected reinforcement value associated with each state-action pair represented by $Q(s_t, a_t)$ in a look-up table. The goal of the Q-Learning algorithm is to choose an action to maximize discounted cumulative rewards over time. It means there is a higher value for short term rewards in comparison to distant future rewards. The model of Q-learning is based on Markov decision process formalism. The agent in state s_t performs an action a_t, and receives the reward r_t for the related state-action from the environment. The state is changed to s_{t+1} by considering probability transition T. The Q-values stored in the Q-table are updated as below:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\,(r_t + \gamma\ max_{a\,\in A}\,Q(s_{t+1}, a) - Q(s_t, a_t)\,) \qquad (1)$$

where $\alpha\ (0 \le \alpha < 1)$ is the learning rate and $\gamma\ (0 \le \gamma \le 1)$ is the discount factor. Fig. 2 explains the Q-Learning algorithm in more details. In a particular state, Q-learning helps to decide the next action based on the expected reward.

B. Neural Networks

Neural networks (NNs) were originally inspired from the way biological nervous systems work, A NN is an information processing paradigm with excellent learning and approximation capabilities. As a universal approximator, a NN can approximate any nonlinear mapping to any degree of accuracy [17]. This feature allows NNs to find and extract hidden patterns from complicated and imprecise data, something that is too complex to be noticed by either traditional data mining methods or humans. NNs have been widely used for prediction, modeling, control, and classification problems in different fields of science and engineering.

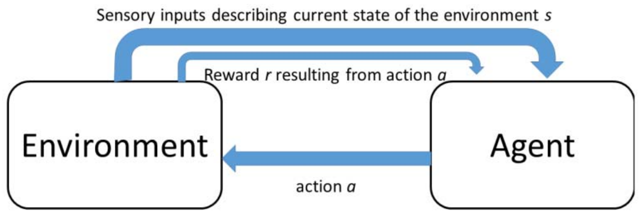
Sensory inputs describing current state of the environment s
Reward r resulting from action a
Environment
Agent
action a

Figure1. Interaction of agent and environment in reinforcement learning.

Q-Learning algorithm

```
Initialize Q(s, a) arbitrarily
Repeat (for each episode):
  Initialize s
  Repeat (for each step of episode):
    Choose a from S using policy derived from Q (e.g. ϵ-greedy)
    Take action a; observe r, s′
    $Q(s,a) \leftarrow Q(s,a) + \alpha\,[r + \gamma\ max_{a'}\,Q(s',a') - Q(s,a)]$
    $s \leftarrow s'$
  Until s is terminal
```

Figure 2. Pseudo code for Q-learning algorithm.

NNs are often trained through the minimization of an error-based cost function in the case of supervised learning. However, when desired values are unknown, NN parameters can be optimally adjusted through minimization of the cost function. Global optimization methods such as genetic algorithm [18], or simulated annealing (SA) [19] can be applied to obtain the optimal set of parameters. SA improves an initial solution iteratively by selecting another neighbor solution and doing small changes and comparing them according to the fitness function. By this strategy it finds the best solution over time and this process repeats until convergence. Transition to poorer solutions is allowed with the purpose of avoiding local optimums or flat regions. SA has a number of advantages in comparison to traditional mathematical optimization techniques. For example, it can be used for optimization of any cost function, either continuous or discontinuous. Besides, the complexity and dimensionality of the cost function does not put any restriction on SA performance in finding the globally optimal solution.

C. Reinforcement Learning and Neural Networks for Traffic Light Control

One of the most common aims in the field of traffic light signal control is to increase the number of vehicles crossing the intersection per a unit of time. For a dynamic view of traffic, a flexible approach to control traffic signals is necessary. In this regards, learning methods have emerged as useful technologies. Reinforcement learning and neural networks are two learning methods that both have made a significant improvement as a solution for this issue. Wieiring et al. [20] proposed a transition model that estimates waiting times of cars in different states. In [21], a context detection reinforcement learning method is presented. This method created a partial model of the environment based on the demand. During the time, the partial models are improved or new ones are constructed. In [18] and [19], adaptive reinforcement learning is proposed to control a model free traffic environment. In addition, an adaptive control of traffic lights is studied in [13]. In this research a function approximation method is applied as a mapping between states and signal timing. Houli et al. [24] used reinforcement learning in their proposed multi-objective control algorithm. They applied reinforcement learning to predict the overall value of the optimization objective given vehicle's states.

In [14], Abdoos et al. presented an approach for controlling signal lights in a network of 50 intersections based on Q-learning. Each intersection is considered as an agent and the entire system formed as a multi-agent system. In their method, the average queue length in approaching links determines the states of Q-learning and the number of permutations of the approaching links form the number of states. Abdoos et al, considered an intersection with four approaching links. Their state space consists of 24 states as shown in Table I. In this table l_i represent the lengths of queue in approaching link i.

As the actions of Q-learning, they considered different phase splits of the cycle time. Phase split refers to the division of the cycle time into a sequence of green signals for each group of approaching links. In addition, Abdoos et al. adjust the cycle time as a fixed value. They set a minimum green time for each phase and the cycle time is divided to a fixed minimum green time and extension time that can be assigned to each phases. The action space was defined by $<n_{ph}, t_{min}, n_{ex}, h_{ex}>$, where n_{ph}: number of phases, t_{min}: minimum green time for phases (seconds), n_{ex}: number of extensions, h_{ex}: length of each extension (seconds). The efficient cycle length δ is then calculated as follows:

$$\delta = n_{ph} \times t_{min} + n_{ex} \times h_{ex} \quad (2)$$

In addition, reward is inversely proportional to the average lengths of the queues in the approaching links, normalized to remain between 0 and 1.

On the other hand, literature is rich in application of NNs to tackle this challenging issue. For instance, for optimizing the system in [25], simultaneous perturbation stochastic approximation (SPSA) based gradient estimates are used in an NN feedback controller. In this work, a function is developed to take the current traffic information and generate the signal timings. In [26], the aforementioned NN-SPSA algorithm was used in simulation for real cities' traffic. As a result of using this method, there was 10% reduction in vehicle waiting times compared to the previously strategy. In two other studies [6], [27] multi agent systems are presented for controlling traffic lights in a network and they prefer NN in implementing their models.

Here, we will present a comparison between Abdoos's method for an isolated intersection with our proposed NN method, which is also applied to an intersection in similar conditions.

III. Proposed Neural Network Controller for Traffic Signal Timing

In order to control traffic signal lights for an isolated intersection, a learning mechanism using a feed-forward NN has been adapted. The NN used in this study consists of four input neurons, ten neurons in the hidden layer, and four neurons in the output layer. Lengths of queues are fed to the NN as independent inputs and the NN estimates the related green time for each phase. SA is applied for optimizing the weights of the NN. The average delay time over each complete simulation is considered as the cost function for optimizing weights of NN. Fig. 3 shows whole the process.

Fig.3 represents how elements of the NN learning algorithm are related to each other and how they interact with PARAMICS. In every cycle, the length of queues from PARAMICS are fed to the NN and the proposed green times for each cycle are generated and provided to PARAMICS.

TABLE I. EXAMPLE OF STATE SPACE IN [14].

State number	Link order	State number	Link order
State 1	$l_1 \geq l_2 \geq l_3 \geq l_4$	State 13	$l_2 \geq l_3 \geq l_1 \geq l_4$
State 2	$l_1 \geq l_2 \geq l_4 \geq l_3$	State 14	$l_2 \geq l_4 \geq l_1 \geq l_3$
State 3	$l_1 \geq l_3 \geq l_2 \geq l_4$	State 15	$l_3 \geq l_2 \geq l_1 \geq l_4$
State 4	$l_1 \geq l_4 \geq l_2 \geq l_3$	State 16	$l_4 \geq l_2 \geq l_1 \geq l_3$
State 5	$l_1 \geq l_3 \geq l_4 \geq l_2$	State 17	$l_3 \geq l_4 \geq l_1 \geq l_2$
State 6	$l_1 \geq l_4 \geq l_3 \geq l_2$	State 18	$l_4 \geq l_3 \geq l_1 \geq l_2$
State 7	$l_2 \geq l_1 \geq l_3 \geq l_4$	State 19	$l_2 \geq l_3 \geq l_4 \geq l_1$
State 8	$l_2 \geq l_1 \geq l_4 \geq l_3$	State 20	$l_2 \geq l_4 \geq l_3 \geq l_1$
State 9	$l_3 \geq l_1 \geq l_2 \geq l_4$	State 21	$l_3 \geq l_2 \geq l_4 \geq l_1$
State 10	$l_4 \geq l_1 \geq l_2 \geq l_3$	State 22	$l_4 \geq l_2 \geq l_3 \geq l_1$
State 11	$l_3 \geq l_1 \geq l_4 \geq l_2$	State 23	$l_3 \geq l_4 \geq l_2 \geq l_1$
State 12	$l_4 \geq l_1 \geq l_3 \geq l_2$	State 24	$l_4 \geq l_3 \geq l_2 \geq l_1$

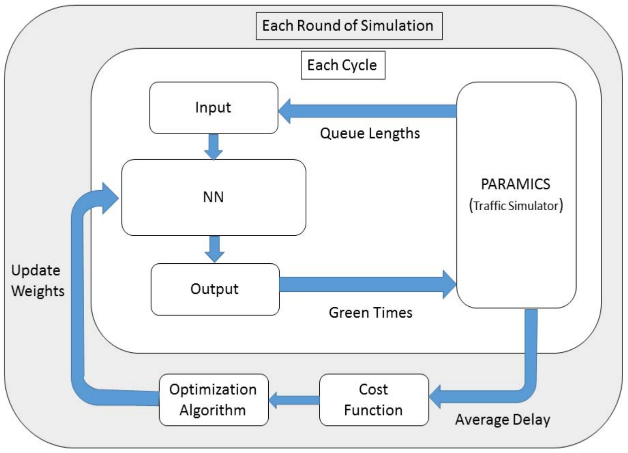

Figure3. The process of NN training. Weights of NN are updated after each round of simulation.

After each period of simulation, when a model simulation run is finished, the average delay time, from the first cycle to last cycle, is calculated and sent to Matlab as the cost function. Based on the cost function and simulated annealing methods, new weights for the NN are generated and updated. This process is repeated up to the time there is no further improvement for several iterations. Upon termination, the NN parameters are set to the optimal set of weights.

It is important to note that the NN model is indirectly trained here, because the desired targets are unknown during the training. In each iteration, a new set of parameters is developed for the NN model. The traffic simulator uses this temporary NN model as the brain for controlling traffic lights. At the end of simulation, it returns the calculated total delay as the cost function to Matlab. A decision whether to accept or discard the current solution is made in the SA method. This process allows us to optimally adjust NN parameters even if the desired targets are unavailable.

The purpose of this study is to compare the performance of NN and Q-learning in controlling traffic signal lights in an isolated intersection. To have this comparison, the Q-learning method proposed in [14] is chosen. Q-learning is a promising method for unknown environment, but it has deficiency in problems with huge state-space. In these situations, the learning process lasts for a long time, however, there may be still some states that do not occur during learning. Therefore, decision making cannot be accurate if one of these unvisited or low-visited states occurs.

In [14], a fixed cycle time is considered and also a limitation on the number of extensions of green time for each phase is defined. This strategy was useful for decreasing the number of state-space, but it can cause inflexibility in another point. It is not possible to have different range of green time for each phase due to fixed cycle time, predefined minimum green time, and limitation in the number of extensions for each phase. As per these, the proposed green time may be not completely suitable for the traffic demand.

For the proposed NN method, the green time can be an integer between zero and 100. As a result, more options exist for the green time and more accurate controlling is possible.

IV. Experiments and Discussion

For testing our experiments, an intersection with four approaching links and four phases (A, B, C, D) is considered Fig.4. The cycle time is divided between these four phases and is based on the related green time and direction of the phase, vehicles in each lane can cross the intersection. Zones are the areas that vehicles are released from them to the intersection Fig.4. Here, four different entrances to the intersection create four zones.

The simulation is modeled in PARAMICS v6. Both Q-learning and NN methods are implemented in Matlab R2011b.

In each cycle the length of queues in four approaching links are given to the NN as inputs and the appropriate green time is generated. The parameters used by SA are as follows: the initial temperature is set to 10; a geometric cooling schedule is implemented by a cooling factor of 0.9; the optimization terminates when there is no further improvement for 20 consecutive iterations.

For implementing Q-learning based on [14], 19 fixed actions have been considered. In the mentioned study, it was assumed that the cycle time is fixed and the minimum green time is set to 13 seconds. The possible green times were {13, 23, 33}. All combinations of these three numbers were not considered in the action list due to limitations forced by the fixed cycle time of 100 seconds, and also the number of extensions for each phase, which is\ set to maximum two times and each time 10 seconds. In our NN implementation, the length of a cycle is variable and is obtained from the sum of green times of all phases plus eight. The later one comes from two second safety time required after each phase. For instance, the cycle time for the set {13, 13, 23, 13} is 70 and for {33, 33, 33, 13} is 120. This safety time is also considered in [14]. As per the discussion here, the NN controller puts much less constraints on how green times are adjusted than does the Q-learning method.

In Q-learning, the action selection is ε-greedy and we apply the settings is used in [14]. ε was set to 0.9 in experiments, that means the best action is chosen 10% of times and an action is selected randomly in 90% of cases. $\gamma = 0.9$ and $\alpha = 0.1$ are the other parameters used during learning for Q-learning method. As previously mentioned, the reward is inversely proportional to the average lengths of the queues in the approaching links.

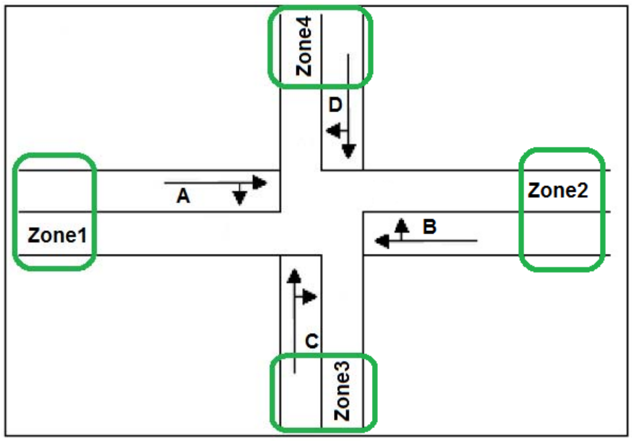

Figure 4. An intersection with four possible phases: A, B, C, D. The cycle time is divided between these four phases and based on the related green time and direction of the phase, vehicles in each lane can cross the intersection. Four zones are also specified in the figure. Zone1, Zone2, Zone3, and Zone4.

The fixed-time method is the third method chosen for comparison. In this method all phases in each cycle have a fixed value and this fixed value is set to 23 during experiments. The value 23 is selected to be closer to the proposed green time values by Abdoos et al. [14].

Two scenarios are considered for evaluating the performance of the proposed method. For the first, 4,990 vehicles enter to the intersection from all four zones, and 2,440 in the second scenario. In both cases, the simulation time is set to 10 hours. Table II and III show the details of simulation parameters respectively. To make results statistically meaningful and minimize effects of random number generators in simulations, all experiments are repeated ten times and average results are reported. During the evaluation, ten different seeds are set in traffic simulator for each scenario, and all three methods are evaluated in similar conditions.

Table II shows the demands used for the first experiment in testing performance of the three methods: fixed-time method, Abdoos method [14] and the proposed NN method. The test is performed after completing learning for Abdoos and NN methods. The same settings in the traffic simulator are applied to all three methods. During the first experiment, 4,990 cars are entered to the intersection in 10 hours. The fixed green time for all phases in fixed-time method is set to 23 seconds, which is the average of three proposed green time numbers by Abdoos (13, 23, 33).

Fig.5 represents the average delay time for each method during these 10 hours. Cumulative average delay in each cycle of ten hours of simulation for fixed time, Abdoos's, and NN methods is 51,023 seconds, 55,161 seconds, and 28,367, respectively. These figures indicate that application of the proposed method greatly improves the traffic control process by 44.40% and 48.57% compared to fixed time and Abdoos's methods respectively.

TABLE II. DEMAND OF CARS FOR EACH ZONE IN FIRST EXPERIMENTS.

	Zone1	Zone2	Zone3	Zone4	Total
Zone1	-	1,000	900	0	1,900
Zone2	400	-	0	100	500
Zone3	0	1,000	-	300	1,300
Zone4	500	0	700	-	1,200
Total	900	2,000	1,600	400	4,990

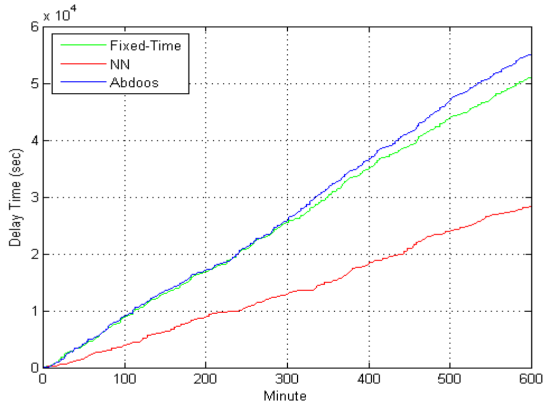

Figure 5. Comparison of the cumulative delay time of fixed-time, Abdoos, and NN methods for scenario one. Cumulative average delay time in seconds during 10 hours (600 minutes) for each method in the first experiments is presented . The results are average of ten times test for each method.

During the second experiment, we configure an intersection with 2,440 cars in 10 hours. Table III shows the demands in this experiment. As before, the performance of the three methods is quantitatively measured and examined. The same settings are considered for the fixed-time method and simulations are run for 10 hours. Fig.6 represents a comparison of the average delay time for each method during these 10 hours. Cumulative average delay in this simulation for the fixed time, Abdoos's, and proposed method are 33,561 seconds, 34,481 seconds, and 18,683 seconds, respectively. These results are again the average of ten simulation runs. Similar to the previous experiment, NN outperforms the fixed-time and Abdoos's methods in terms of more optimally controlling traffic lights by 44.33% and 45.81% respectively. Table IV shows reviews the results of both experiments.

Abdoos's method is useful for many cases as results demonstrated in [14] confirm this. However, considering some special green times reduces its flexibility and optimality for various demands of vehicles, as observed in the experiments presented here. The proposed NN method has the ability to generate different values for green time. Therefore it is more appropriate for real traffic with time variable demands of cars. In addition, the experiments show the NN controller performs much better the fixed-time method. During experiments, the fixed-time method has less mean average delay for defined demands compared to proposed method in [14]. However, better results are related to the time applied for the fixed-time method and this time has a close relation to the traffic demand. Considering some fixed numbers as green time causes a limitation for allocating the best appropriate green time for related traffic demand.

TABLE III. DEMAND OF CARS FOR EACH ZONE IN SECOND EXPERIMENTS.

	Zone1	Zone2	Zone3	Zone4	Total
Zone1	-	400	200	0	600
Zone2	400	-	0	180	580
Zone3	0	300	-	320	620
Zone4	340	0	300	-	640
Total	740	700	500	500	2,440

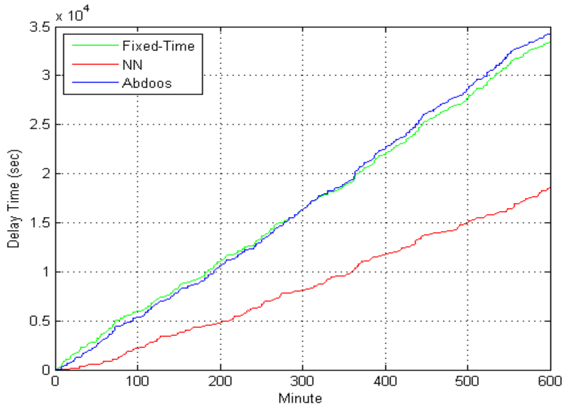

Figure 6. Comparison of the results of fixed-time, Abdoos and NN for the second scenario. Cumulative average delay time in seconds during 10 hours (600 minutes) for each method in second experiments is presented. The results are average of ten times test for each method.

TABLE IV. REVIEW OF RESULTS FOR TWO DEMANDS IN 10 HOURS OF SIMULATION.

Number of Cars in 10 hours	**Improvement of NN against fixed-time and Abdoos (Average of ten times test for each method)**	
	Fixed-method	***Abdoos***
4,900	44.40%	48.57%
2,440	44.33%	45.81%

V. CONCLUSION

In this study, we implement a NN model for controlling traffic signal lights for an isolated intersection. The performance of this method was compared with the Q-learning method presented in [14] and the fixed-time method. The experiment results indicate that the NN performance is superior. One of the main reasons for this superiority is considering flexible green time and cycle times against some special ones proposed in [14].

The simulation results illustrate that for some situations that Abdoos's method does not generate appropriate results, the NN proposes better options for controlling intersection green times.

In future work we will extend the NN method for a multi-agent network and measure its performance in a multi-intersection model.

References

[1] 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 and Technology*, vol. 12, no. 3, pp. 29–35, 2011.

[2] Balaji, "Distributed multi-agent based traffic management system," University of Madras, Singapore, 2011.

[3] Choy Min Chee, "Cooperative, hybrid multi-agent systems for distributed, real-time traffic signal control.," NUS, Singapore, 2005.

[4] A. L. C. Bazzan, "A Distributed Approach for Coordination of Traffic Signal Agents," *Auton Agent Multi-Agent Syst*, vol. 10, no. 1, pp. 131–164, Jan. 2005.

[5] I. Alvarez, A. Poznyak, and A. Malo, "Urban traffic control problem a game theory approach," in *47th IEEE Conference on Decision and Control,*, Cancun, Mexico, 2008, pp. 2168–2172.

[6] P. G. Balaji and D. Srinivasan, "Multi-Agent System in Urban Traffic Signal Control," *IEEE Computational Intelligence Magazine*, vol. 5, no. 4, pp. 43–51, 2010.

[7] L. Yang and W. D. Dai, "An Approach for Short Term Traffic Flow Forecasting Based on Genetic Neural Network," *Advanced Materials Research*, vol. 671–674, pp. 2866–2869, Mar. 2013.

[8] Q. Xia and L. Shi, "An Optimized Design Scheme for Roundabout Traffic Management," *Applied Mechanics and Materials*, vol. 253–255, pp. 1884–1889, Dec. 2012.

[9] P. Chanloha, W. Usaha, J. Chinrungrueng, and C. Aswakul, "Performance Comparison between Queueing Theoretical Optimality and Q-Learning Approach for Intersection Traffic Signal Control," in *Fourth International Conference on Computational Intelligence, Modelling and Simulation (CIMSiM)*,2012, pp. 172–177.

[10] B. Abdulhai, R. Pringle, and G. Karakoulas, "Reinforcement Learning for True Adaptive Traffic Signal Control," *Journal of Transportation Engineering*, vol. 129, no. 3, pp. 278–285, 2003.

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

[12] E. Bingham, "Reinforcement learning in neurofuzzy traffic signal control," *European Journal of Operational Research*, vol. 131, no. 2, pp. 232–241, Jun. 2001.

[13] L. A. Prashanth and S. Bhatnagar, "Reinforcement Learning With Function Approximation for Traffic Signal Control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 2, pp. 412–421, 2011.

[14] M. Abdoos, N. Mozayani, and A. L. C. Bazzan, "Traffic light control in non-stationary environments based on multi agent Q-learning," in *14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, Washington, DC, USA., 2011, pp. 1580–1585.

[15] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Mach Learn*, vol. 8, no. 3–4, pp. 279–292, May 1992.

[16] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press, 1998.

[17] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

[18] T. M. Mitchell, *Machine Learning*, 1st ed. McGraw-Hill Science/Engineering/Math, 1997.

[19] A. Dekkers and E. Aarts, "Global optimization and simulated annealing," *Mathematical Programming*, vol. 50, no. 1–3, pp. 367–393, Mar. 1991.

[20] M. Wiering, J. Vreeken, J. van Veenen, and A. Koopman, "Simulation and optimization of traffic in a city," in *IEEE Intelligent Vehicles Symposium*, 2004, pp. 453–458.

[21] B. C. da Silva, E. W. Basso, F. S. Perotto, A. L. C. Bazzan, and P. M. Engel, "Improving reinforcement learning with context detection," in *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, New York, NY, USA, 2006, pp. 810–812.

[22] K. Wen, S. Qu, and Y. Zhang, "A stochastic adaptive control model for isolated intersections," in *IEEE International Conference on Robotics and Biomimetics*, 2007, pp. 2256–2260.

[23] Y. Dai, D. Zhao, and J. Yi, "A comparative study of urban traffic signal control with reinforcement learning and Adaptive Dynamic Programming," in *The International Joint Conference on Neural Networks (IJCNN)*, 2010, pp. 1–7.

[24] D. Houli, L. Zhiheng, and Z. Yi, "Multiobjective Reinforcement Learning for Traffic Signal Control Using Vehicular Ad Hoc Network," *EURASIP Journal on Advances in Signal Processing*, no. 1, pp. 1687–6180, Sep. 2010.

[25] J. C. Spall and D. C. Chin, "Traffic _Responsive Signal Timing for System-Wide Traffic Control," vol. 5, no. 3/4, pp. 153–163, 1997.

[26] D. C. Chin, J. C. Spall, and R. H. Smith, "Evaluation of system-wide traffic signal control using stochastic optimization and neural networks," in *American Control Conference,*, 1999, vol. 3, pp. 2188–2194 vol.3.

[27] D. Srinivasan, M. C. Choy, and R. L. Cheu, "Neural Networks for Real-Time Traffic Signal Control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no. 3, pp. 261–272, 2006.