2019 IEEE Intelligent Transportation Systems Conference (ITSC)
Auckland, NZ, October 27-30, 2019

Reinforcement Learning with Explainability for Traffic Signal Control

Stefano Giovanni Rizzo[1*], Giovanna Vantini[1] and Sanjay Chawla[1]

Abstract— **Deep reinforcement learning has recently provided promising results on the traffic light control optimization problem, by training neural network agents to select the traffic light phase. These agents learn complex models by optimizing a simple objective, such as the average traffic speed, but are considered opaque when it comes to explaining their decisions. Nevertheless, explanations are required in transferring this technology in the real world, especially in complex scenarios with nontrivial phases, such as in the case of signalized roundabouts with entry and circulatory traffic lights. In this paper, after training a Policy Gradient agent on a signalized roundabout with 11 phases and real traffic data, we analyze the relation between the agent phase preferences and the actual traffic, and we assess the agent capability of reacting to the current detectors state. Then, we estimate the effect of the road detectors state on the agent selected phases, through the SHAP model-agnostic technique, using Shapley values recovered from a linear explanation model. The results show that it is possible to extract meaningful explanations on the decision taken by a complex policy, in relation to both the traffic volumes and the lanes occupancy.**

I. INTRODUCTION

In recent years, the notion of eXplainable Artificial Intelligence (XAI) has become more and more relevant, in the effort of producing more explainable models, while maintaining a high level of performance (e.g. accuracy prediction), enabling users to understand and trust A.I. partners [1]. One of the main factor for this tension between performance and explainability is the increasing *opacity* of machine learning models, and in particular neural networks, defined by complex structures and large amounts of parameters values.

The problem is particularly crucial in Reinforcement Learning (RL), where the models are usually trained to fit, and often overfit [2], data generated from thousands or millions of episodes, thus becoming at the same time very complex but weak to unseen scenarios. The resulting trained agents are then expected to take decisions in the real-world autonomously, without human supervision.

Losing explanation capability in favor of RL performances can be acceptable in some cases, such as when video games [3] or board games [4] are played by RL agents to beat worldwide human champions. In many other real-world applications – medicine, legal, finance and transportation in particular – some guarantees on the reasoning behind these complex models are demanded in order to gain the trust of the stakeholders. Another reason to stress the importance of explainability in RL for real-world domains is that a real context is simulated by means of a digital environment, with the result of oversimplifying complex dynamics or introducing artificial phenomena.

In the context of traffic light control optimization, among the others, machine learning based solutions should provide explanations about their decisions, and these explanations must be assessed by domain experts before deployment. Traffic light control is considered a successful application of RL techniques, where the agent is trained to switch the traffic light phase with the goal of reducing travel times and emissions. The combination of reinforcement learning algorithms and deep learning, such as Deep Q-learning and Deep Policy Gradient, has been recently applied in a successful attempt of engineering optimal reactive policies for signalized intersection [5], [6], [7], [8], [9], [10]. Despite recent works also highlighted the need for interpreting the agent dynamics [8], no attempt has been made to interpret the decision making of a RL agent for traffic light control.

At the same time, model explanation methods such as LIME (Local Interpretable Model-Agnostic Explanations)[11] and SHAP (Shapley Additive exPlanation) [12] are gaining popularity in applied machine learning for providing explanations that can be verified by domain experts. SHAP values, in particular, unifies the other features attribution models under a single solution, satisfying several desirable properties of features attribution [12]. To the best of our knowledge, these methods have not been considered for the interpretation of RL agents decisions.

In this paper, we analyze the relations between the traffic flows and the trained RL agent, and we propose SHAP values as an additive attribution method to interpret the logic behind the agent decisions. After implementing a Policy Gradient neural network for induction loop data, we study the relations between the input state of a simulated road network and the RL agent output. We consider as a case study the complex environment of a signalized roundabout with four entry arms of three lanes, traffic lights on both incoming sections and circulatory sections and eleven possible phases from which the agent can select the next move.

We analyze the consistency of choices made by the agent and highlight unintended effects of the policy in a simulated environment. We show that Shapley values have the potential to capture the effect of traffic volumes in each specific lane, and also to highlight overfitting phenomena for training on the same scenario.

II. RELATED WORK

The optimization of traffic light control strategy has a long history in RL, with one of the earliest work applying SARSA on the traffic light control problem dating more than

[1] Qatar Computing Research Institute (QCRI), Doha, Qatar
* strizzo@hbku.edu.qa

two decades ago [13]. Reinforcement Learning approaches model the signalized vehicular network as a Markov Decision Process (MDP). The state of the MDP is the representation of the traffic situation, such as the discretized positions of vehicles [14], [15] or ranges of traffic volume [16]. More often the traffic state is represented as the occupancy of discretized cells in a grid [5], [7] or the vehicles properties in each grid's cell [9].

Recent approaches use neural networks as function approximators to estimate the state-action value (Deep Q-learning [17], [18], [7]) or to learn directly a policy from state to actions (Deep Policy Gradient [6]). Deep learning approaches allowed the use of more complex states representation, such as camera view from above the intersection or, more precisely, a screenshot of the simulator [6], [8].

Most of these works only evaluate the agent performance, however the importance to study the learned policies has been emphasized in [8] and it has been addressed on a high-level by comparing the traffic and the selected phase over time in a 2-phases intersection.

In modeling the dynamics of roundabout traffic in relation with circulatory and incoming lanes, a conflict matrix and queue theory have been proposed [19] for 1-lane and 2-lanes roundabouts. Attempts at optimizing timings in signalized roundabouts have also been made, using pre-defined models [20], [21], automatically alternating between yield control and full signalization [22] and applying the Cross-entropy method for Reinforcement Learning [23].

III. METHOD

A. Environment modeling

The environment is a real signalized roundabout, with traffic lights on both circulatory and entry lanes. We represent this signalized vehicular network as a Markov Decision Process (MDP) with states, actions and rewards.

1) States: The state in our environment is defined as the set of car counts coming from the detection loops installed on the network. The detectors loops are installed on both approaching and circulatory lanes (see Figure 1). For approaching lanes, two lines of detectors are installed, one line closer to the roundabout entrance, one several meters farther (e.g. 150m before the entry). For circulatory lanes, two lines of detectors are installed for each direction, one line before the exit arm, one line before the entry arm. A line of detectors has one detector for each lane. The state space is discrete given that, during any time step, the count of cars from a detector is a natural number between 0 and the capacity of the lane per time step.

2) Actions: A discrete set of actions allows to control the flow by selecting the phase of the traffic lights. In the proposed environment the agent can select any phase at each time step without particular order.

We define one phase for each entry arm, and one phase for each pair of opposite entry arms, assuming they result in less conflicting routes, where two entry arms have both the green light and all other entry traffic lights are red. We also provide, for the previous pairs, the combinations of phases

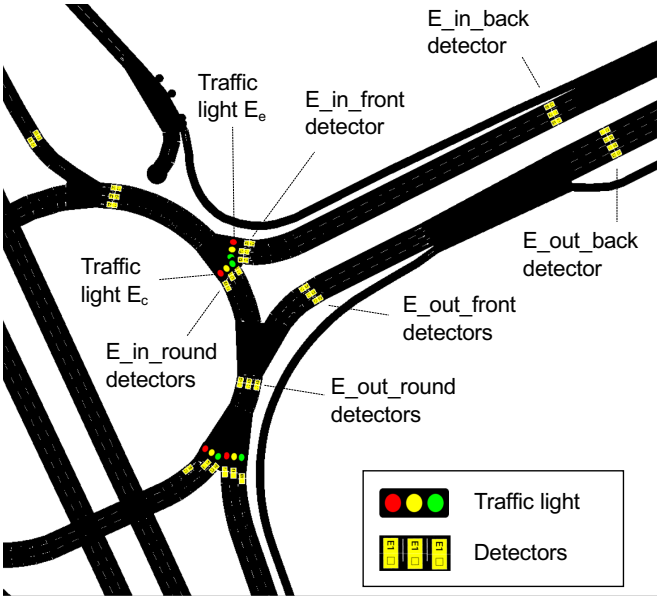

Fig. 1. Illustration of the detectors and traffic lights for the East section of the roundabout.

for which one entry arm of the pair has green and the other has an intermittent amber (i.e. approaching cars must yield). Finally, an additional *all-red* phase is provided to allow the discharge of the circulatory lanes when congested.

In Table I we list the set of 11 actions, each one corresponding to a traffic light phase, for our four-way roundabout. In order to safely switching between any pair of phases, we create an amber light phase transition, hidden from the agent, for each possible pair of distinct phases. Conversely, no actual change is made on the phases if the action selected is equal to the previous.

Action	Entry lanes	Circulatory lanes
a_1	$G(N_e); R(W_e, S_e, E_e)$	$G(W_c, S_c, E_c); R(N_c)$
a_2	$G(W_e); R(S_e, E_e, N_e)$	$G(S_c, E_c, N_c); R(W_c)$
a_3	$G(S_e); R(E_e, N_e, W_e)$	$G(E_c, N_c, W_c); R(S_c)$
a_4	$G(E_e); R(N_e, W_e, S_e)$	$G(N_c, W_c, S_c); R(N_c)$
a_5	$G(N_e, S_e); R(W_e, E_e)$	$G(W_c, E_c); R(N_c, S_c)$
a_6	$G(W_e, E_e); R(N_e, S_e)$	$G(N_c, S_c); R(W_c, E_c)$
a_7	$G(N_e), A(S_e); R(W_e, E_e)$	$G(W_c, E_c); A(S_c); R(N_c)$
a_8	$G(S_e), A(N_e); R(W_e, E_e)$	$G(W_c, E_c); A(N_c); R(S_c)$
a_9	$G(W_e), A(E_e); R(N_e, S_e)$	$G(N_c, S_c); A(E_c); R(W_c)$
a_{10}	$G(E_e), A(W_e); R(N_e, S_e)$	$G(N_c, S_c); A(W_c); R(E_c)$
a_{11}	$R(N_e, W_e, S_e, E_e)$	$G(N_c, W_c, S_c, E_c)$

TABLE I

PHASES ACTIONS FOR A FOUR-WAY ROUNDABOUT.

3) Reward: the goal of traffic light optimization in a saturated network has a dual objective: maximizing the throughput of the intersection and avoid bad traffic conditions such as long queues that may interfere with other junctions. We trade-off between these two objectives by considering the capacity in the reward function, while avoiding traffic jams using the final states defined below. We define the reward function, as the number of vehicles departed from the roundabout at time t.

4) Final states: In the episodic setting, the return G_t is defined as the sum of the rewards $R_{t+1} + R_{t+2} + \ldots + R_T$, with T being the time of termination. Thus, the expected return increases with T and each episode is a repeated attempt at avoiding the episodic condition. We design traffic jam related final states so that the optimal policy is forced to avoid them in order to maximize the accumulated reward. A secondary effect a faster convergence in Monte Carlo setting, in which the return G_t at a time step t is computed with a complete rollout of the episode because it will generate shorter bad episodes. We consider as traffic jam the presence of a queue so long that interfere with another junction.

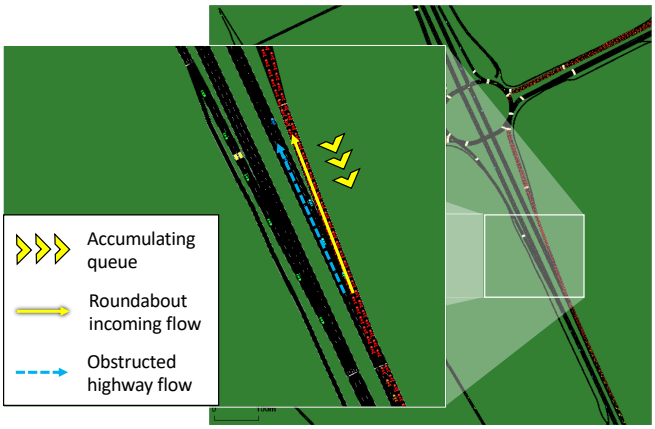

Fig. 2. Vehicles entering the roundabout interferes with the highway, obstructing its flow. In our environment, this scenario is identified as a final state, thus terminating the episode and the accumulation of rewards.

As an example, Figure 2 shows the queue from the entry arm of the roundabout blocking the traffic flow of the highway, despite the two being grade-separated.

B. Policy Gradient with time advantage

The goal of reinforcement learning is to learn an optimal policy $\pi_*(a|s)$, that is, a mapping from states to probabilities of selecting each action, that maximizes the accumulated reward. At each step, the policy takes the state of the environment as input and yield a probability distribution over actions. Following this distribution, the agent selects randomly an action, so that most probable actions, according to the policy output, have more likelihood of being picked. In policy gradient methods a parameterized policy $\pi(a|s,\theta)$ is learned in order to maximize some performance measure $J(\theta)$ based on the accumulated reward. The optimal policy in these methods is approximated by the iterative update

$$\theta_{t+1} = \theta_t + \alpha \nabla J(\theta_t). \tag{1}$$

We prefer policy optimization over dynamic programming methods, as it learns directly a transition model to maximize the reward. The policy $\pi(a|s,\theta)$ can be any parameterized function as long as the partial derivatives with respect to parameter vector θ exist and are finite for all the states and actions. In practice, the policy must not be deterministic in order to obtain an exploratory behavior, therefore an exponential soft-max distribution is used

$$\pi(a|s,\theta) = \frac{e^{h(s,a,\theta)}}{\sum_b e^{h(s,b,\theta)}} \tag{2}$$

where $h(s,a,\theta)$ expresses the numerical preferences for each state and action and θ can be the weight vector from a linear regression or a deep artificial neural network.

A crucial aspect of Policy Gradient methods is dealing with the variance of the gradients in different states and episodes [24]. This can negatively affect the training dynamic, therefore it is very common to subtract a *baseline* from the accumulated reward in order to normalize the gradient scale in the optimization step. In the episodic case, the number of remaining steps until the final state T can also be an important source of variance. We apply a B_t return baseline at time t, computed as the moving average of the discounted return at time t, G_t. The update rule of the REINFORCE using the time baseline is

$$\theta_{t+1} = \theta_t + \alpha(G_t - B_t)\frac{\nabla\pi(A_t|St,\theta_t))}{\pi(A_t|S_t,\theta_t)} \tag{3}$$

We define a Monte-Carlo Policy Gradient algorithm with time baseline using the above defined update rule and an exponentially weighted moving average (EWMA) for updating the baseline B_t in every episode.

C. Shapley Values

In cooperative game theory, *Shapley values* are a solution concept to the problem of calculating the payoff of every single player of a coalition of players, when there is a degree of correlation between different players' contributions. By considering the marginal contribution in each possible subset of the coalition, Shapley values decompose the total values in every single player's payoff, providing the only solution that satisfies the properties of efficiency, linearity, symmetry and null player [25]. This approach has been successfully applied to compute feature values importance for linear models in the presence of multicollinearity. The feature values importance are named *Shapley regression values*.

Given a feature value i, its Shapley regression value is the difference between a model $f_{S\cup\{i\}}(x_{S\cup\{i\}})$ trained with i, and a model $f_S(x_S)$ trained without i, averaged over all possible subsets S of features values F.

$$\phi_i = \sum_{S\subseteq F\setminus\{i\}} \frac{|S|!(|F|-|S|-1)!}{|F|!}[f_{S\cup\{i\}}(x_{S\cup\{i\}}) - f_S(x_S)] \tag{4}$$

The Shapley value decomposition estimates the most likely contribution of each individual feature value pair to the fit of the model[26]. It has been shown that Shapley values are the only possible explanation model, that satisfies all desirable properties of feature attribution methods [12].

On the other hand, computing the Shapley values for each feature is challenging. LIME (Local Interpretable Model-Agnostic Explanations) [11] is an efficient additive feature attribution method that learns a linear model $g(z')$ on the simplified binary input vector z', where $g(z') = f(h_x(z'))$

and h_x is a function from the simplified input to the original input. In the case of numerical features such as in the vehicle detectors, the simplified input vector has an element for each numerical bin of each numerical feature. In order to efficiently evaluate the Shapley values while keeping the consistency with the above desirable properties, a specific instance of the LIME method consistent with the Shapley value formulation has been recently defined, named *Kernel SHAP*[12]. A SHAP (Shapley Additive exPlanation) value [12] estimates the change in prediction, from the base value $E[f(x)]$ to the instance output $f(x)$, attributed to each single feature i. For example, for the feature $i = 1$ with value $z_1 = x$ the SHAP value is $E[f(z)|z_1 = x_1] - E[f(z)]$.

IV. RESULTS

A. Experimental setup

We run the experiments on a real roundabout dataset. The dataset includes the network shape of an area of 2.4km $\times$ 3.4km, together with traffic data from the peak hour of a weekday.

Experiments are carried out using the SUMO simulation software (Simulation of Urban MObility)[1]. We implemented an Open AI gym module to create the simulation environment described in Section III-A. Because the SUMO time step represents the minimum reaction time, we set it to 0.5 seconds to reduce collisions and ensure a smooth traffic flow, while environment step is 10 seconds long so that each phase can last a multiple of 10 seconds. Each phase includes one green sub-phase, shown in Table I and a yellow sub-phase of 3 seconds. The yellow sub-phase is needed for phase transitions, however, if the agent choose an action that is equal to the last phase, then the yellow phase is skipped.

We train the Policy Gradient agent using a neural network with one hidden layer of 512 ReLu units, stopping at 15,000 episodes when the return is stabilized.

B. Traffic and phases distribution

We first analyze the relation between the simulation traffic and the learned policy $\pi_{sa}(a|s)$ at a high-level, averaged over a complete episode. In Figure 3 we compare the origin destination (OD) matrix of vehicles traveling through the roundabout, and the average phases probability of the learned RL agent. The phase probability for one of the phases a_i is calculated as

$$p_{sa}(a_i) = \sum_{t=1}^{m} \frac{\pi_{sa}(a_i|s_t)}{m}$$

Where $m = 360$ for an episode of one hour and a time step of 10 seconds, and π_{sa} is the learned policy. With an average probability of 0.41, the East-West (`G EW`) is by far the most selected phase during the simulated hour. The intuitive interpretation is that the volume traffic incoming from West and from East is heavier, as can be seen from the two corresponding rows in the OD matrix. However, this does not explain why, for example, this phase is preferred to the single East and West phases, or why the West by East phase (`G W by E`) is preferred to the East by West one (`G E by W`). The first step toward a better understanding is to consider also the destination of vehicles routes, and therefore decompose the traffic in two: traffic through each incoming traffic light (Figure 4 and traffic through each circulatory traffic light (Figure 5). In Figure 4 the traffic

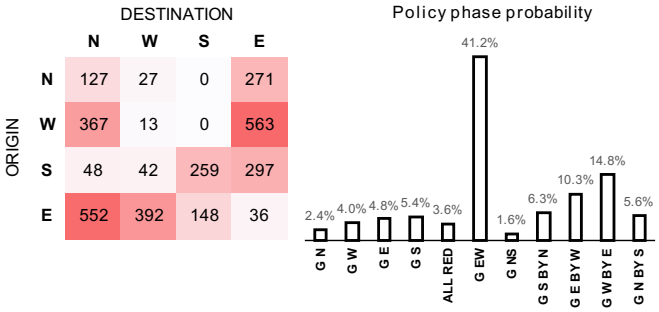

Fig. 3. OD matrix for traffic going through the roundabout and policy average probability for each phase over one hour episode.

through each entry traffic light is the sum of all the related destinations in the OD matrix. The policy probability for entry traffic light is the sum of the average probability of the phases that give green or amber (yielding) light to that traffic light. For example, the probability of the East entry traffic light is the sum of the averages probabilities for each action that has a $G(E_e)$ or a $A(E_e)$ in entry lanes as described in Table I, that is, the average of $\pi_{sa}(a_4|s) + \pi_{sa}(a_6|s) + \pi_{sa}(a_9|s) + \pi_{sa}(a_{10}|s)$ over all states s. Finally,

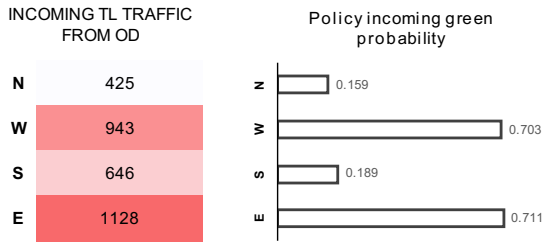

Fig. 4. Incoming vehicles at traffic light for each direction, and relative probability of green in the learned policy averaged over all time steps.

in Figure 5, the same analysis is carried out for circulatory traffic lights. In order to measure the traffic at a given circulatory traffic light, we sum all the OD matrix pairs that have a route going through that traffic light. For example, the OD routes going through the North circulatory traffic light are: `(W,W),(S,W),(S,S),(E,W),(E,S),(E,E)`. Policy circulatory green probability is again computed as the average probability of a circulatory traffic light having a green or amber light. From Figure 5 it emerges that the South circulatory traffic light faces around 60% more traffic than the other circulatory traffic lights. From the policy probabilities it results that the policy was successfully able to implicitly estimate the circulatory traffic for each direction, without having access to OD routes. It also shows that, in choosing the phase for the current time step, the agent

[1]https://sumo.dlr.de/

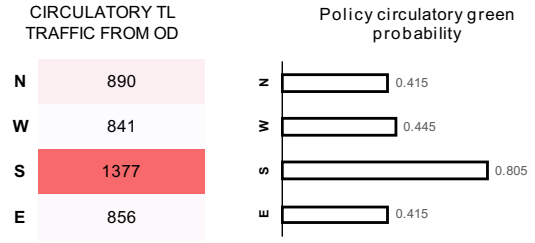

Fig. 5. Circulatory vehicles at traffic light for each direction, and relative probability of green in the learned policy averaged over all states in the episode.

considers not only the incoming traffic volume but also how this will affect the circulatory lanes in the future time steps.

C. State dependency

The above analysis considers the policy probability averaged over the entire episode and how this reflects the traffic rates of each traffic light. If this was sufficient to explain the policy effectiveness, a policy based only on actions could be defined, disregarding the current state of the detectors, obtaining the same results. Given $\pi_{sa}(a|s)$ the policy learned using Policy Gradient, we define π_a as the policy that simulates the π_{sa} probabilities in Figure 3:

$$\pi_a(a_i) = p_{sa}(a_i) = \sum_{t=1}^{m} \frac{\pi_{sa}(a_i|s_t)}{m}$$

As for π_{sa}, the agent of π_a is stochastic a selects an action following π_a probability distribution.

Policy	Reward	Arrived	Waiting time [s]	CO_2 [kg]
Fixed short phases	1577	5278	3262	6586
Fixed long phases	1811	5383	2594	5682
Action only π_a	2268	6259	2573	3974
PG policy π_{sa}	**2501**	**6598**	**728**	**3316**

TABLE II
POLICY PERFORMANCES. REPLICATING THE PG POLICY ACTION PROBABILITIES IN AN ACTION-ONLY POLICY π_a IS NOT SUFFICIENT TO REACH PG AGENT PERFORMANCE.

In Table II we show also two fixed time policies for two common cycle lengths of 120 and 170 seconds. The *Action only* π_a is the action-dependent policy defined on the average probabilities from the PG policy π_{sa}. The learned PG policy π_{sa} exhibits better values in all the measures, in particular it provides a 72% reduction of the waiting time.

D. Shapley values

In order to further interpret the decision making of the RL agent given a specific detectors state, we compute the SHAP values of the detector states in each time step using the Kernel explainer of the SHAP library [12].

In Figure 6 we picked an instance of a particularly congested time step, showing SHAP values of the selected phase (Green N by S), its opposite phase and the ALL RED phase. The chart of each phase is centered on the average output while showing how each detector state has driven the prediction from the average value to the current output value. For all phases, the front detectors played minimal to no role in the decision making, for which back detectors and round detectors seems instead to be crucial. This can be explained by the fact that front detectors are not useful to estimate an entry arm queue, but only how many vehicles were entering the roundabout from that arm during the last time step.

In the same way, North detectors are not taken in consideration, suggesting that the agent has learned that very low traffic is coming from the North arm. Depending on the context, this may be considered as a negative overfitting phenomenon: in the event of unexpected traffic (e.g. traffic from a football match from North) this agent can be suboptimal.

In the SHAP values of the Green N by S phase, as expected, the high values of the lanes detectors from the South arm are contributing to the phase selection, as well as the low values in the East and West lanes. On the contrary, for the phase Green E by W all the South detectors are on the right side, decreasing the probability of giving green to East and West arms.

For both Green N by S and Green E by W phases, the role of detectors from different lanes of the same incoming arm is difficult to interpret. This becomes more clear from the values of the detectors in the ALL RED phase. This phase goal is to free up the circulatory lanes in view of an incoming wave of traffic. In this phase, the back detectors in lane 0 and lane 2 from the South arm (`S_in_back_0`, `S_in_back_2`) have the highest absolute values, but they are driving the probability of ALL RED in opposite direction. The `S_in_back_0` detector, placed on the external lane of the South arm (i.e. the lane on the far right) is a predictor for eastbound traffic, entering and quickly leaving the roundabout at the first exit without the need of traveling around the roundabout. On the other hand, `S_in_back_2` is a predictor of traffic incoming from South and traveling all around the roundabout until the third or fourth exit, for which the occupancy of circulatory lane is crucial.

V. CONCLUSIONS AND FUTURE WORK

In this paper, we have shown how it is possible to derive explanations from a neural network agent for traffic light control. In particular, we considered the complex scenario of a signalized roundabout having eleven phases, eight traffic lights in circulatory and entry lanes and heavily congested traffic, with a neural network agent trained using Policy Gradient. The road network and the simulated traffic are real and recorded in the peak hour of a business day.

We have first analyzed the probability distribution of the learned policy for each phase, with respect to the volumes of the traffic routes. We found that the probability of a traffic light having a green or amber phase is reasonably proportional to the volume of traffic it faces during the simulation both for entry lanes and circulatory lanes. We show that this is not sufficient to explain the agent effectiveness, by replicating the same probabilities in an action-only policy.

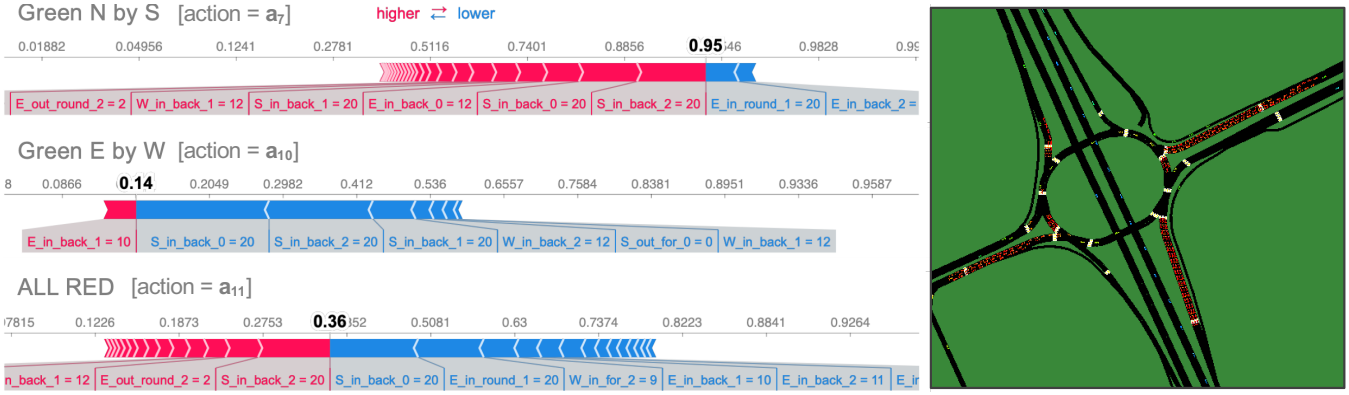

Fig. 6. Time step $t = 200$ at minute `00:33:20`, during a traffic peak, showing SHAPE values of two opposite phases (Green N by S and Green E by W) and the ALL RED phase.

Then, by computing the Shapley values of each detector we measured how the detector value contributed (positively or negatively) to the choice of each traffic light phase. These preliminary results shown how the agent learned to react differently based on each specific lane's traffic, by implicitly predicting the route of the traffic and thus its future circulatory occupancy. We were also able to show some evidence of consistency in the agent logic, analyzing the detectors Shapley values for contrasting phases. However, the analysis also highlighted a bias toward the traffic seen in the training (overfitting), and some relations between detectors values and phase choices could not be intuitively explained.

As a future line of research, we plan to directly train simpler, explainable models and study the tradeoff of accuracy in comparison with a complex Deep Learning model.

References

[1] D. Gunning, "Explainable artificial intelligence (xai)," *Defense Advanced Research Projects Agency (DARPA), nd Web*, 2017.

[2] C. Zhang, O. Vinyals, R. Munos, and S. Bengio, "A study on overfitting in deep reinforcement learning," *arXiv preprint arXiv:1804.06893*, 2018.

[3] O. Vinyals, I. Babuschkin, J. Chung, M. Mathieu, M. Jaderberg, W. M. Czarnecki, *et al.*, "Alphastar: Mastering the real-time strategy game starcraft ii," https://deepmind.com/blog/alphastar-mastering-real-time-strategy-game-starcraft-ii/, 2019.

[4] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, *et al.*, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, p. 484, 2016.

[5] L. Kuyer, S. Whiteson, B. Bakker, and N. Vlassis, "Multiagent reinforcement learning for urban traffic control using coordination graphs," in *Machine Learning and Knowledge Discovery in Databases*, W. Daelemans, B. Goethals, and K. Morik, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 656–671.

[6] S. S. Mousavi, M. Schukat, and E. Howley, "Traffic light control using deep policy-gradient and value-function-based reinforcement learning," *IET Intelligent Transport Systems*, vol. 11, no. 7, pp. 417–423, 2017.

[7] J. Gao, Y. Shen, J. Liu, M. Ito, and N. Shiratori, "Adaptive traffic signal control: Deep reinforcement learning algorithm with experience replay and target network," *arXiv preprint arXiv:1705.02755*, 2017.

[8] H. Wei, G. Zheng, H. Yao, and Z. Li, "Intellilight: A reinforcement learning approach for intelligent traffic light control," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, ser. KDD '18. New York, NY, USA: ACM, 2018, pp. 2496–2505.

[9] X. Liang, X. Du, G. Wang, and Z. Han, "A deep q learning network for traffic lights' cycle control in vehicular networks," *IEEE Transactions on Vehicular Technology*, pp. 1–1, 2019.

[10] K.-L. A. Yau, J. Qadir, H. L. Khoo, M. H. Ling, and P. Komisarczuk, "A survey on reinforcement learning models and algorithms for traffic signal control," *ACM Computing Surveys (CSUR)*, vol. 50, no. 3, p. 34, 2017.

[11] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should i trust you?: Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. ACM, 2016, pp. 1135–1144.

[12] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Advances in Neural Information Processing Systems*, 2017, pp. 4765–4774.

[13] T. L. Thorpe, "Vehicle traffic light control using sarsa," [Online]. Available: citeseer.ist.psu.edu/thorpe97vehicle.html, Tech. Rep., 1997.

[14] M. Wiering, "Multi-agent reinforcement learning for trac light control," in *Proceedings of the Seventeenth International Conference on Machine Learning*, 2000, pp. 1151–1158.

[15] L. Chun-Gui, W. Meng, S. Zi-Gaung, L. Fei-Ying, and Z. Zeng-Fang, "Urban traffic signal learning control using fuzzy actor-critic methods," in *Natural Computation, 2009. ICNC'09. Fifth International Conference on*, vol. 1. IEEE, 2009, pp. 368–372.

[16] P. Balaji, X. German, and D. Srinivasan, "Urban traffic signal control using reinforcement learning agents," *IET Intelligent Transport Systems*, vol. 4, no. 3, pp. 177–188, 2010.

[17] W. Genders and S. Razavi, "Using a deep reinforcement learning agent for traffic signal control," *arXiv preprint arXiv:1611.01142*, 2016.

[18] L. Li, Y. Lv, and F.-Y. Wang, "Traffic signal timing via deep reinforcement learning," *IEEE/CAA Journal of Automatica Sinica*, vol. 3, no. 3, pp. 247–254, 2016.

[19] Z. Qu, Y. Duan, H. Hu, and X. Song, "Capacity and delay estimation for roundabouts using conflict theory," *The Scientific World Journal*, vol. 2014, 2014.

[20] W. Ma, Y. Liu, L. Head, and X. Yang, "Integrated optimization of lane markings and timings for signalized roundabouts," *Transportation research part C: emerging technologies*, vol. 36, pp. 307–323, 2013.

[21] Xiaoguang Yang, Xiugang Li, and Kun Xue, "A new traffic-signal control for modern roundabouts: method and application," *IEEE Transactions on Intelligent Transportation Systems*, vol. 5, no. 4, pp. 282–287, Dec 2004.

[22] H. Xu, K. Zhang, and D. Zhang, "Multi-level traffic control at large four-leg roundabouts," *Journal of Advanced Transportation*, vol. 50, no. 6, pp. 988–1007, 2016.

[23] M. Maher, "The optimization of signal settings on a signalized roundabout using the cross-entropy method," *Computer-Aided Civil and Infrastructure Engineering*, vol. 23, no. 2, pp. 76–85, 2008.

[24] R. S. Sutton and A. G. Barto, *Introduction to reinforcement learning*. MIT press Cambridge, 1998, vol. 135.

[25] H. P. Young, "Monotonic solutions of cooperative games," *International Journal of Game Theory*, vol. 14, no. 2, pp. 65–72, 1985.

[26] S. K. Mishra, "Shapley value regression and the resolution of multicollinearity," *Available at SSRN 2797224*, 2016.