Rainbow Deep Reinforcement Learning Agent for Improved Solution of The Traffic Congestion

1st Mahmoud Abo Ayaad

*Electrical Engineering Department Shoubra Faculty of Engineering* Benha University, Cairo, Egypt mahmoud.taha17@feng.bu.edu.eg

2nd Ahmed Zakey

*Electrical Engineering Department Shoubra Faculty of Engineering* Benha University, Cairo, Egypt ahmed.zaky@feng.bu.edu.eg

3rd Abdulwahab Al-Sammak *Electrical Engineering Department Shoubra Faculty of Engineering* Benha University, Cairo, Egypt [asammak@feng.bu.edu.eg](mailto:asammak@feng.bu.edu.eg)

***Abstract*—While traffic congestion hits severely the world economy, adaptive traffic signal systems would efficiently provide potential solutions. In this paper, we propose a deep reinforce- ment learning system to control the signal lights in an isolated intersection. The proposed system uses a deep convolutional neural network to extract the crucial features from the en- vironment state that is described by raw traffic information; i.e., vehicles positions, speeds, and waiting times. Besides, the system utilizes a multi-objective reward and the Rainbow agent which provides further space of enhancements to the conventional Deep Q-Networks agent. Extensive experiments illustrate that our proposed deep framework outperforms the baseline under a number of settings and traffic measures, including trip time, waiting time, fuel consumption, and stability.**

***Index Terms*—Adaptive traffic signals, Rainbow, DQN, Distri- butional RL**

1. INTRODUCTION

Traffic congestion affects significantly people every day; it wastes their times and harms their economic activities. For example, the cost of traffic congestion in the Greater Cairo Metropolitan Area (GCMA) was estimated to be five billion USD [1]. Traditionally, congestion was eliminated by infrastructure expansion, such as building more roads or adding lanes to existing ones. However, this is not often feasible because of limited space or high costs. Therefore, there is an ever-increasing effort in the area of intelligent transportation systems (ITS). For example, intelligent traffic signal controllers would minimize vehicles waiting time and fuel consumption in urban areas. Moreover, the limits of the current signal systems stimulate the demand for intelligent signals. On the one hand, the traditional pre-timed and actuated signals do not respond to traffic demands efficiently and may cause long queues [2]. On the other hand, adaptive signals like SCATS [3] and SCOOT [4] need not only high costs but also highly-skilled engineers and accurate traffic models [5] [6]. Recently, reinforcement learning has been very successful in different areas like computer games and robotics. It has also been promising in adaptive traffic signals control. Thorpe and Anderson pioneered the efforts of using RL for traffic control; the on-policy state-action-reward-state-action (SARSA) was used to control an isolated intersection [7].

©2019 IEEE

Abdulhai, Pringle, and Karakoulas presented the model-free Q-learning in traffic signal control [8]. After the astonishing success of deep reinforcement learning (DRL) in different games [9] [10][11][12], there are many attempts to utilize DRL in traffic signals.

In [13], an isolated two-phase intersection was controlled by a deep reinforcement learning agent where the Q-function was modeled by deeply stacked autoencoders, and the agent state was represented by the queueing lengths of the last four time steps. Although a minimum green time was imposed on the agent, no red-clearance time was assumed. Genders and Razavi used a deep convolutional neural network (CNN) to model the action-value function [14]. They additionally pro- posed the discrete traffic state encoding (DTSE) to construct the state. DTSE discretizes each lane into a set of equally- spaced cells such that the state included three vectors: The first vector encodes the presence/absence of vehicles in the network cells. The second represents the vehicles speeds, and the third represents the current signal lights. Although Genders and Razavi trained the agent using a deep Q-learning algorithm, the proposed algorithm did not include the target network freezing method that has been proved essential for the stability of the popular Deep Q-Networks (DQN) agent [9]. Pol and Oliehoek used the DTSE where they represented the intersection state with two-channel position image and designed a multi-objective reward function [15]. Additionally, they tuned the DQN algorithm for the traffic signal control. In [16], Mousavi, Schukat, and Howley designed a deep policy- gradient agent where a neural network was used to model the optimal policy parameters. They represented the agent state by visual snapshots from the simulator. In [17], DTSE was applied so that the input state included the vehicles positions and speed values. The Q-value function was modeled using a double dueling deep convolutional neural network, and the action space was defined by the duration of each phase in the next signal cycle. The authors limited the phases maximum duration by 60 seconds.

In this paper, we propose to solve the adaptive traffic signal problem using a deep RL agent where we use the discrete traffic state encoding such that the state includes vehicles positions, speeds, and waiting times information. Moreover, we tune a multi-objective reward and adopt the Rainbow

Accepted in the 7*th* International Japan-Africa Conference on Electronics, Communications, and Computations (JAC-ECC 2019)

agent that combines many extensions to DQN, including distributional RL, multi-step learning, and prioritized replay memory.

The remainder of this paper is organized as follows. Section II describes the proposed system. In section III, the experi- mental results are reported. Finally, section IV provides the

*Distributional RL:* The idea is that we address the full rewards distribution instead of the expected value. Bellemare, Dabney, and Munos [20] proposed to model this distribution using a distribution with discrete support, **z**, that is a set of

atoms {*zi* = *Vmin* + *i*6*z* : 0 ≤ *i < N* }, where *N* ∈ N and

*Vmin, Vmax* ∈ R, and 6*z* = *Vmax*−*Vmin* . In distributional RL,

−

conclusion and outlines our future directions.

the atoms probabilities **p***θ* (*s, a*)

*N* 1

are approximated

using a deep

1. PROPOSED SYSTEM

The traffic signal control problem could be formulated as a Markov Decision Process, MDP, where each intersection is controlled by an RL agent. At every time step *t*, the agent observes the environment state *st*; then, takes an action *at*; finally, receives a delayed reward *rt*+1. The agent aims at learning, from this iterative process, an optimal policy, *π*∗,

that maximizes the expected discounted reward, *ttt*, which is defined as *ttt* = ∞*k*=0 *γ*(*k*)*rt*+*k*+1 where *γ* is the discount factor. In Q-learning method, the agent learns the policy, *π*, by

Σ

learning state-action values *qπ* (*s, a*) that estimate the expected discounted reward by following the policy *π* after taking action *a* in state *s*. In DQN, the state-action values are approximated by deep convolutional neural network, and the input state is often in the form of stacked pixel frames [9]. Moreover, neural nets are optimized by using stochastic gradient descent to minimize the loss:

2

(*r* + *γ* max *q* (*s , a*´) − *q* (*s , a* )) (1)

*t*+1 *θ*¯ *t*+1 *θ t t a*´

where *θ* is the parameters of the online network that is used to select actions, and *θ*¯ is the parameters of the target network. *θ* is updated based-on the loss, which is calculated over a randomly selected batch from a replay memory. However, *θ*¯ is not directly optimized but cloned from *θ* every K steps (target update period). Although DQN has been a great achievement, various limitations have been identified, and many extensions have been proposed to enhance its learning speed, stability, etc. We briefly mention three relevant extensions:

*Prioritized replay memory:* The RL agent samples from the replay memory according to priorities *pt* that approximate how much the agent would learn from transitions [18].

*pt* ∝ |*rt*+1 + *γ* m*a*´ax *qθ*¯(*st*+1*, a*´) − *qθ* (*st, at*)| (2)

*w*

*w* is the prioritization factor that determines how much prior- itization is used.

*Mutli-step learning:* The RL agent considers a multi-step target instead of the single step one in the loss definition:

(*rn* + *γn* max *qθ*¯(*st*+*n, a*´) − *qθ* (*st, at*))2

neural net with parameters *θ* that is optimized using stochastic gradient descent to minimize the KL-divergence between the current and the target distributions.

Recently, Hessel et al. integrated six extensions (Multi- step learning, double Q-learning, prioritized replay, dueling networks, distributional RL, and noisy nets) into the Rainbow agent [21]. Hessel et al. observed that prioritized replay, multi-step learning, and distributional RL are the most critical components in Rainbow.

We propose an adaptive signal system based on a compact Rainbow version that employs the aforementioned three ex- tensions. We take advantage of the proposed state matrix by modeling the online and target networks using CNN networks. We adopt the categorical DQN architecture [20] that is equiv- alent to the usual DQN model [9] but with *Natoms Nactions*

×

outputs to estimate the atom probabilities *pi*(*s, a*) instead of

action-values *Q*(*s, a*). The proposed Rainbow parameters are shown in Table I.

TABLE I: Settings for the baseline and the proposed system.

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Baseline Proposed agent** | |
| Agent  State matrix size State features |  | |
| DQN  168 *×* 168 *×* 2  position | Rainbow  168 *×* 168 *×* 3  position, speed, and waiting |
| N-steps | 1 | 3 |
| Update period | 1 | 4 |
| Target network freeze | 30000 | 30000 |
| Replay memory size | 50000 | 100000 |
| Experience sampling | uniform | prioritized |
| Prioritization parameter | 0 | 0.5 |
| Learning rate (*α*) | 0.00025 | 0.00025 |
| Optimizer | adam | adam |
| Batch size | 32 | 32 |
| Exploration rate (*s*) | 0.01 | 0.01 |
| Discount factor(*γ*) | 0.99 | 0.99 |
| number of atoms  *Vmax Vmin* | NA NA NA | 51  10  -10 |

*State definition:* We adopt the discrete traffic state en- coding such that each lane is divided into equally-spaced cells of length C. The parameter C should be defined carefully to make sure that each cell is large enough to accommodate one vehicle. We propose to represent the state by a three- dimensional matrix where each cell includes the following: (i) boolean position to represent the existence of a vehicle in that

*t a*´

*n*−1

*r* = Σ *γ rt*+*k*+1

(3)

cell; (ii) the vehicle speed normalized to the maximum allowed

*n k*

*t*

*k*=0

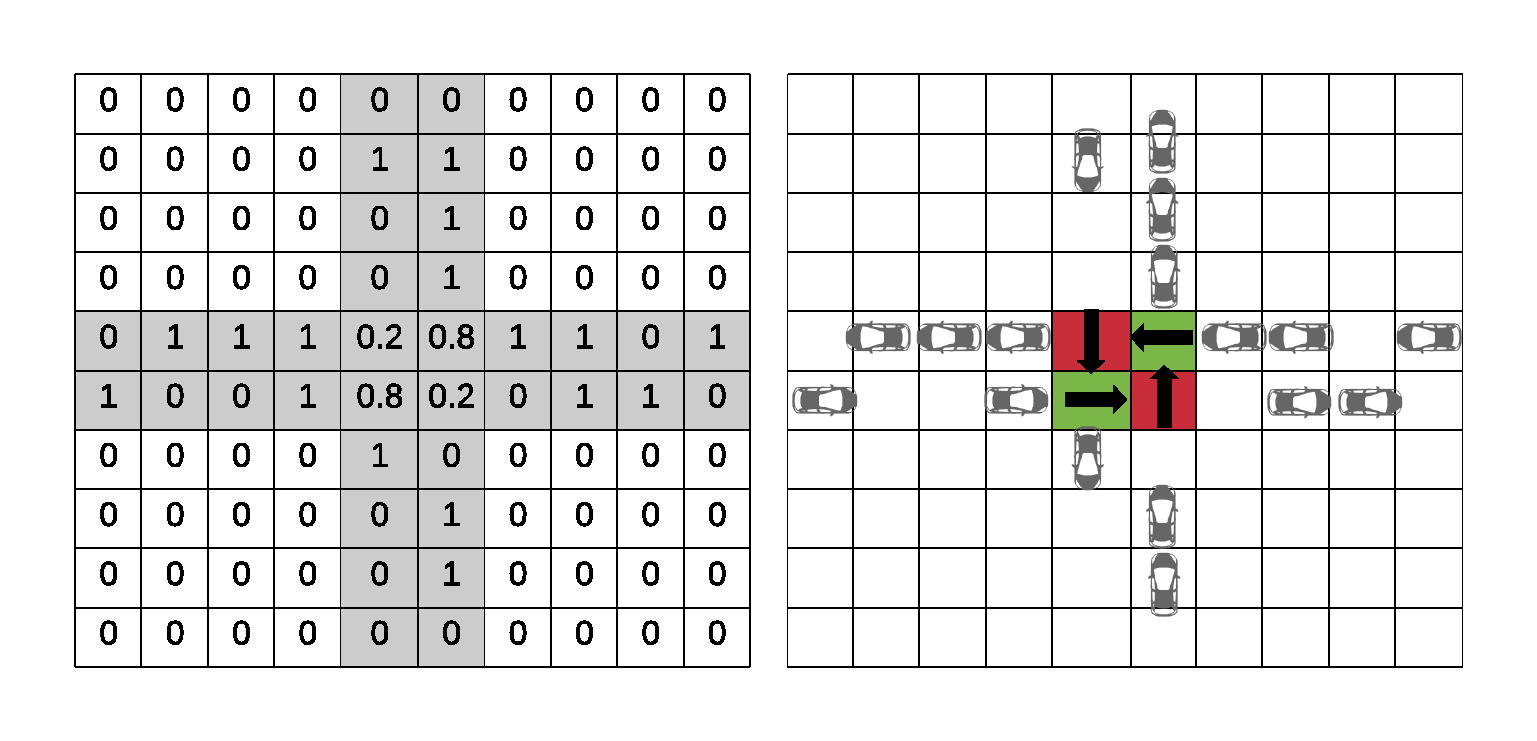
The number of steps *n* is a hyper-parameter that could increase the learning speed if a suitable value is chosen [19].

speed in the lane; and (iii) the vehicle waiting time normalized to the maximum waiting time in the network. The addition of speed is an extension beyond [15], and the waiting time is an extension beyond [15] [14][17]. Fig. 1 shows the first dimension of the proposed state where each vehicle is mapped

to a specific cell that contains the value “1”. Besides, the running signal phase of the incoming lanes is encoded using four cells. The other two channels are the same except that they represent the speeds and waiting times of the corresponding vehicles. We encode the traffic phases in the first dimension only. Consequently, the proposed state size is *M M* 3, and *M* is proposed as 168. *M* is double the value of the Atari agent [9], and it has been found that it is sufficient for 500*m* lanes where each lane around the intersection is discretized into 84 cells [15]. For longer lanes, we could consider the 500*m* as the area of interest to the agent.

× ×

Fig. 1: Simplified traffic situation and the corresponding posi- tions channel (current traffic lights are encoded in the center of the matrix).



*Action definition:* We define the action as the signal phase index that would be applied by the traffic controller into the intersection. Table II shows the phases and the corresponding actions of a conventional four phases intersection. It illustrates that the proposed agent do not map the yellow phases to actions; thus, if the agent chooses a different action than the current one, it would first insert the convenient transition phase for one second. Although we propose the minimum green time as one second, we do not enforce a constraint on the maximum time. While this configuration ensures a better green time flexibility, the agent may continuously flicker between actions. Therefore, a specific penalty would be added in the reward function to prevent such situations.

TABLE II: Traffic phases and the associated actions.

|  |  |  |  |
| --- | --- | --- | --- |
| **Phase** | **Right-of-way** | **Action** | **Safety transition phase** |
| GrGr  rGrG | N-S, S-N  E-W, W-E | 1  2 | yryr  ryry |

*Reward function:* The definition of the traffic controller objective is a critical task as there is no obvious perfect goal. For example, the agent may learn to minimize the queuing length; however, it would leave a few vehicles for a long period. We adopt a multi-objective reward function as follows:

Σ

Σ

aforementioned flickering issue, *e* is the number of vehicles that experienced an emergency stop in the last time step, *di* is the speed delay of vehicle *i* in the last time step that

*speed*

is defined as *d* = 1 − , and *wi* is the waiting

*allowed speed*

time of vehicle *i* normalized to the maximum value in the

network. Although this reward function is an extension to Pol and Oliehoek reward [15], the proposed function is different in two main points: (i) Pol and Oliehoek considered *wi* as

0*.*5 for single second of waiting and 1*.*0 for two or more;

hence, the agent would not identify the difference between two waiting seconds and 100s. Therefore, they added a penalty to teleports count that often indicates long waiting time in SUMO simulator. On the other hand, we do not clip *wi* but normalize it to the maximum waiting time in the network. Consequently, we removed the teleports term from the proposed reward. (ii) We normalize *e*, *d*, and *w* terms by *N* . As a result, the reward value is inherently clipped to the closed interval [-1.0, 0.0].

1. EXPERIMENTS AND RESULT ANALYSIS

In general, all the experiments are conducted on a standard traffic simulator; Simulation of Urban Mobility (SUMO) [22]. SUMO is a widely-used open source simulator that supports the Microscopic model and runs in either console or graphical modes. SUMO provides a powerful interface (TRACI) to monitor and control the simulation from external programs.

*Traffic scenario and implementation details:* The traffic scenario reported by Pol and Oliehoek in NIPS 2016 [15], which we considered as the baseline, is used and could be described as follows. The network consists of a single intersection with four edges where each edge has one incoming and one outgoing lane. All lanes are 500 meters long; and all turns, at the intersection, are prohibited. The intersection is controlled by a four phases traffic signal where turns are prohibited; i.e., only straight-going phases are allowed. Each phase indicates the states, red (r), green (G) or yellow (y) , of the approaching lanes. Table II summarizes the four phases; two phases are mapped to the agent actions; the others are considered as safety transition phases.

{ }

Since turns at the intersection are prohibited, there are four possible routes for any vehicle trip (North-South, East-West and vice versa). SUMO accepts the traffic demand via routes file, that defines the departure time and the complete path for every vehicle in the simulation. We generated a set of routes files; each represents an episode and simulates one traffic hour. The vehicles are inserted according to uniform distribution with probability 0.1 to insert vehicles every second at any possible route.

*Proposed agent evaluation:* To evaluate the performance of the proposed agent, we carried out a comparative assess- ment against the DQN agent [15]. We utilized a high-quality

DQN implementation released by Google and did our best to

*Rt*+1 = −0*.*2∗*c*−

* 1. ∗ *e N*

−0*.*3∗

*N*

*i*=1

*N*

*di*

−0*.*3∗

*i*=1 *wi N*

(4)

replicate the baseline. The parameters of baseline and proposed agents are shown in Table I. We trained both agents on the

where *N* is the total number of vehicles in the last time step, *c* is a boolean flag that equals true if the agent changed the phase in the last time step; this penalty is used to prevent the

*N*

aforementioned intersection using a set of traffic episodes generated using a defined seed. A variety of measures are considered during comparison; e.g., average vehicle trip time

and average vehicle waiting time. Comparing the rewards is not feasible because the agents do not share the same reward function. Fig. 2 shows the behavior of the proposed agent compared to the baseline; as seen, our agent is superior in terms of not only the achieved vehicle waiting time but also the stability at training. Table III summarizes various measures from the last ten training episodes (*s* = 0*.*01); it demonstrates that the proposed agent is better in terms of trip time and fuel consumption. Although we removed the explicit teleports penalty from the reward, the learned policy does not cause teleports.

time, and trip time. Finally, further research can be identified as: (i) testing the agent in more realistic traffic scenarios; (ii) taking advantage of the state matrix sparsity by employing sparse convolutional neural networks; and finally (iii) extend- ing the proposed agent to multi-agent scenarios.

REFERENCES

* + 1. World Bank, “Cairo traffic congestion study : Final report,” World Bank, Tech. Rep., May 2013. [Online]. Available: https:

//openknowledge.worldbank.org/handle/10986/18735

* + 1. L. Zhang, H. Li, and P. D. Prevedouros, “Signal control for oversaturated intersections using fuzzy logic,” in *Transportation and Development*

This superiority of the proposed system compared to the

*Innovative Best Practices 2008*, 2008, pp. 179–184.

* + 1. P. Lowrie, “Scats, sydney co-ordinated adaptive traffic system: A traffic

baseline could be accounted for different reasons: First, the reported instability issue of the usual DQN algorithm in the traffic environment [15]. Second, CNN networks could find better high-level features from the prosperous state. Third, the normalized reward may result in smooth updates to the neural networks. Finally, the proposed Rainbow agent has shown a superior performance to the vanilla DQN in many environments [21].

Fig. 2: Behavior of the proposed agent compared to the baseline at training.

60

Baseline Agent

Proposed Agent

50

40

Waiting Time(s)

30

20

10

0

0.0 200.0K 400.0K 600.0K 800.0K 1.0M

Training Steps

TABLE III: Performance comparison of the baseline and the proposed system.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Metric** | **Baseline** | | **Proposed system** | |
| mean std | | mean std | |
| Vehicle trip time (s) Vehicle waiting time (s) Total teleports  Total fuel consumption (l) |  | | | |
| 209  12.8  0  164 | 51  2.46  0  37 | 101  2.1  0  90 | 0.57  0.14  0  3 |

1. CONCLUSIONS AND FUTURE DIRECTIONS

In this paper, the traffic signal control problem is modeled

responsive method of controlling urban traffic,” Roads and Traffic Authority NSW, Tech. Rep., 1990.

* + 1. P. Hunt, D. Robertson, R. Bretherton, and R. Winton, “Scoot-a traffic responsive method of coordinating signals,” Tech. Rep., 1981.
    2. K. Fehon, “Adaptive traffic signals are we missing the boat,” *ITE District*, vol. 6, 2004.
    3. R. Cunningham, A. Garg, V. Cahill *et al.*, “A collaborative rein- forcement learning approach to urban traffic control optimization,” in *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, 2008. WI-IAT’08.*, vol. 2. IEEE, 2008, pp. 560–566.
    4. T. L. Thorpe and C. W. Anderson, “Traffic light control using sarsa with three state representations,” Technical report, Citeseer, Tech. Rep., 1996.
    5. B. Abdulhai, R. Pringle, and G. J. Karakoulas, “Reinforcement learning for true adaptive traffic signal control,” *Journal of Transportation Engineering*, vol. 129, no. 3, pp. 278–285, 2003.
    6. V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, “Human-level control through deep reinforcement learning,” *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.
    7. D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam,

M. Lanctot *et al.*, “Mastering the game of Go with deep neural networks and tree search,” *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

* + 1. D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang,
       1. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton *et al.*, “Mastering the game of Go without human knowledge,” *Nature*, vol. 550, no. 7676, p. 354, 2017.
    2. D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez,

M. Lanctot, L. Sifre, D. Kumaran, T. Graepel *et al.*, “Mastering chess and shogi by self-play with a general reinforcement learning algorithm,” *arXiv preprint arXiv:1712.01815*, 2017.

* + 1. L. Li, Y. Lv, and F.-Y. Wang, “Traffic signal timing via deep reinforce- ment learning,” *IEEE/CAA Journal of Automatica Sinica*, vol. 3, no. 3, pp. 247–254, 2016.
    2. W. Genders and S. Razavi, “Using a deep reinforcement learning agent for traffic signal control,” *arXiv preprint arXiv:1611.01142*, 2016.
    3. E. Van der Pol and F. A. Oliehoek, “Coordinated deep reinforcement learners for traffic light control,” *Proceedings of Learning, Inference and Control of Multi-Agent Systems (at NIPS 2016)*, 2016.
    4. 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.
    5. X. Liang, X. Du, G. Wang, and Z. Han, “Deep reinforcement learn- ing for traffic light control in vehicular networks,” *arXiv preprint arXiv:1803.11115*, 2018.
    6. T. Schaul, J. Quan, I. Antonoglou, and D. Silver, “Prioritized experience replay,” *arXiv preprint arXiv:1511.05952*, 2015.
    7. R. S. Sutton, “Learning to predict by the methods of temporal differ-

as a Markov decision process, and the deep reinforcement learning is proposed as a potential solution. The proposed agent utilizes a rich state definition, a multi-objective reward, as well as the extended Rainbow agent as compared to the conventional DQN. Experiments demonstrated the superiority of the proposed agent over the baseline in terms of various metrics, including stability, fuel consumption, vehicle waiting

ences,” *Machine learning*, vol. 3, no. 1, pp. 9–44, 1988.

* + 1. M. G. Bellemare, W. Dabney, and R. Munos, “A distributional perspec- tive on reinforcement learning,” in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 449–458.
    2. M. Hessel, J. Modayil, H. Van Hasselt, T. Schaul, G. Ostrovski,

W. Dabney, D. Horgan, B. Piot, M. Azar, and D. Silver, “Rainbow: Combining improvements in deep reinforcement learning,” in *Thirty- Second AAAI Conference on Artificial Intelligence*, 2018.

* + 1. D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker, “Recent devel- opment and applications of SUMO - Simulation of Urban MObility,”

*International Journal On Advances in Systems and Measurements*, vol. 5, no. 3&4, pp. 128–138, December 2012.