A Method for Road Accident Prevention in Smart Cities based on Deep Reinforcement Learning

Giuseppe Crincoli[1], Fabiana Fierro[2], Giacomo Iadarola[1][a], Piera Elena La Rocca[1], Fabio Martinelli[1], Francesco Mercaldo[3,1][b] and Antonella Santone[3]

[1]*Institute of Informatics and Telematics, National Research Council of Italy (CNR), Pisa, Italy*

[2]*Spike Reply, Milan, Italy*

[3]*University of Molise, Campobasso, Italy*

Keywords: Machine Learning, Deep Reinforcement Learning, Smart Cities, Automotive, Artificial Intelligence.

Abstract: Autonomous vehicles play a key role in the smart cities vision: they bring benefits and innovation, but also safety threats, especially if they suffer from vulnerabilities that can be easily exploited. In this paper, we propose a method that exploits *Deep Reinforcement Learning* to train autonomous vehicles with the purpose of preventing road accidents. The experimental results demonstrated that a single self-driving vehicle can help to optimise traffic flows and mitigate the number of collisions that would occur if there were no self-driving vehicles in the road network. Our results proved that the training progress is able to reduce the collision frequency from 1 collision every 32.40 hours to 1 collision every 53.55 hours, demonstrating the effectiveness of deep reinforcement learning in road accident prevention in smart cities.

1 INTRODUCTION

Smart city is an expression that recently has become fashionable and it refers to the collective ideal of the city of the future. The notion of smart city is related to an urban area where the use of digital technologies and technological innovation optimizes and improves the infrastructures and services provided to citizens. In the smart cities context, the so-called autonomous vehicles (i.e., AVs) play a crucial role. As a matter of fact, AVs are expected to be among the first robotic systems to enter and widely affect existing societal systems (Wu et al., 2017a).

On the other hand, cybersecurity threats are emerging hand in hand with technological advancement, and they may affect the rapid diffusion of AVs. Therefore, improving safety and reliability in AVs is growing interest from both the industrial and the academic research community, considering that the global market for AVs is expected to rapidly grow in the next decade (Statista.com, 2021; Martinelli et al., 2021b).

As pointed out by Kathoun et al. (Khatoun and Zeadally, 2017), the smart city architecture and technological innovation necessarily leads to open cybersecurity scenarios that involve new threats and challenges for security experts (Martinelli et al., 2021a; Martinelli et al., 2018a). The wide adoption of Information and Communications Technologies (ICT) systems into vehicles has opened new possibilities for attackers to access and attack cars. Nowadays, we use smartphones to control the radio/GPS tracker, these features can be exploited to control cars passing through ICT systems such as Android applications. Since it is possible to remotely control the behaviour of a vehicle, we should design AV systems that can automatically react to these attacks that cause deviations from the expected driving behaviour.

In the light of the above considerations, this paper proposes a method aimed to reduce the likelihood of road accidents using vehicular network data to smooth traffic flows and control the behaviour of the drivers. In detail, we are interested in understanding whether a *Deep Reinforcement Learning* (*DRL*) trained vehicle inserted in the road network can prevent road accidents. We assume that 'vehicles under cyberattacks' have unwanted aggressive and unsafe driving. The study poses the following research ques-

[a] https://orcid.org/0000-0001-7060-6233

[b] https://orcid.org/0000-0002-9425-1657

Crincoli, G., Fierro, F., Iadarola, G., Rocca, P., Martinelli, F., Mercaldo, F. and Santone, A.
A Method for Road Accident Prevention in Smart Cities based on Deep Reinforcement Learning.
DOI: 10.5220/0011146500003283
In *Proceedings of the 19th International Conference on Security and Cryptography (SECRYPT 2022)*, pages 513-518
ISBN: 978-989-758-590-6; ISSN: 2184-7711

tion:

- *RQ*: *Is it possible to increase driving safety by introducing in a road network a vehicle trained by DRL?*

The paper proceeds as follows: in Section 2, we present the proposed method; in Section 3, we discuss the results of the experimental evaluation; current state of the art is discussed in Section 4; finally, in last section conclusion and future research plans are provided.

2 A METHOD FOR PREVENTING ROAD ACCIDENTS

In this paper, we propose a method that, using *DRL*, is dedicated to prevent traffic accidents in smart cities when some running vehicles are under cyberattacks. Figure 1 illustrates the workflow of the proposed method. More in detail, the security analyst uses Flow to implement its *DRL* algorithm and interfaces with the *TraCI* APIs that are used to interact - both inbound and outbound - with the traffic managed by the *SUMO* simulator.

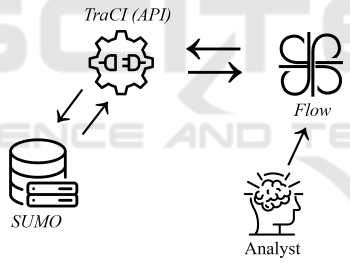

Figure 1: The proposed method for avoiding traffic incidents in smart cities.

We applied the proposed method in a simulated scenario where vehicles are driven on several lanes, taking into account, in addition to sudden accelerations, the cases in which the sudden lane-change may cause inconvenience or collisions, since lane-changing is considered one of the main causes of road accidents (Wu et al., 2017b). The rationale behind the proposed method is related to accident prevention in the AVs context, by considering the application of *DRL* to train vehicles on how to behave to avoid sudden collisions. To demonstrate the effectiveness of the proposed method, we considered a simulated environment involving several vehicles, in which it is possible to intervene on the simulation causing vehicles to make real-time decisions. Attacks are configured in aggressive driving caused, purposely, by human decisions. As an evidence, there is the possibility to externally set the trajectory and the speed of the individual vehicles and above all the possibility of changing vehicles' speed during the simulation. This applies both to the self-driving vehicle trained with *DRL* and to vehicles under attack. Furthermore, it is worth noting that our methodology does not take into account a specific cyberattack on the vehicles, but instead it is a countermeasure to any possible one that interferes with the usual driving style. Indeed, we assume that an attacked vehicle will start acting aggressively (such as invading lines), and then we study how all the other vehicles in the simulation can react to this behaviour to mitigate the collisions and damages.

Our method uses *SUMO* as a server that is running on a remote port waiting for clients to connect. A client interacts with the server to cause actions on the entities in the simulation. The *TraCI* APIs are used to intervene at run-time on the running simulation (e.g. to make a car accelerate, brake, change lane, etc.). The analyst or security expert interfaces with Flow to implement its *DRL* methodology, allowing them to interact with traffic (Figure 1). At run-time, depending on the effects of the actions performed by the vehicles, each action of a vehicle is given a reward when it is considered positive, otherwise a penalty when the action is considered negative. It's important stressing that *SUMO* creates a model that is collision-free by nature and this makes it impossible to simulate road accidents. Thus, in order to let vehicles learn that actions that lead to collisions are negative, we had to reduce to 0 the minimum safety distance between vehicles to let them crash. This change within the simulation increased the probability of causing cars' collisions.

Interacting with the simulation is the way to cause actions that can lead to *rewards* and *penalties*. For the sake of clarity, consider that suddenly increasing vehicles' speeds and/or making them lane-change can probably let vehicles collide. These actions can be unwanted if they are the result of vehicular cyberattacks on the cars within the simulation. In that scenario, the vehicles that performed bad actions that caused collisions receive a penalty aimed at discouraging them from repeating those actions again. After performing several simulations, it was clear that - at each simulation cycle - the execution of the algorithm improved the performance of the vehicles even if some cyberattacks were performed on some vehicles making them perform some dangerous actions, since generally all the vehicles are able to learn from good and wrong actions performed in the past.

In order to design a method to prevent road accidents we developed a network configuration customizing the original single-lane ring network de-

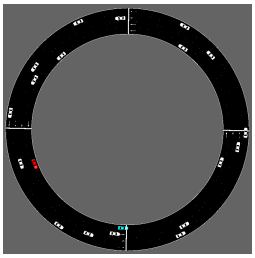

Figure 2: Our four-lanes network (19 "human" vehicles and 1 DRL vehicle).

scribed in the paper by Vinitsky *et al.* (Wu et al., 2017b) and that is one of the benchmarks designed in flow. The original *ring network* provided by *Flow* (Wu et al., 2017b) simply consists of a circular lane with a fixed length. Our personalized ring network extended the original single-lane environment by adding three lanes: we created a four-lane ring road network that is a natural extension to problems involving a single-lane ring in order to consider a more realistic multi-lane scenario. Basically, it is a loop road generated by specifying the number of lanes and a ring circumference, in this case personalized starting from the original. In our four-lanes ring network were inserted 20 vehicles (19 vehicles driven by humans and 1 guided by *DRL*) in order to cause cyberattacks on 5 different vehicles (25% of the total number of vehicles).

Once the network is created, the initial positioning of the vehicles is random on multiple lanes. Each vehicle has a starting speed of 60 km/h. Each vehicle is 5 meters long and follows the *Intelligent Driver Model* (*IDM*) (Sutton and Barto, 2018) dynamics with parameters specified by Martin Treiber and Arne Kesting (Treiber and Kesting, 2013). In traffic flow modeling, the *IDM* is a time-continuous car-following model for the simulation of freeways and urban traffic. The proposed network also relies on additional parameters. Despite the default timestep[1] duration is 1 second for *SUMO*, in our case each step lasts 0.1 seconds, as this is the default in *Flow*. Though the speed limit is 36.1 m/s (maximal limit of Italian highways, just as baseline), the vehicles can reach very high velocities because of their maximum possible accelerations and decelerations that are set to 30 m/s^2 that is the highest acceleration possible as it is reached by a supercar[2]. Vehicles should be allowed to reach high accelerations in order to increase the possibilities to let them be victims of cyberattacks consisting of sudden increasing/decreasing a vehicle's acceleration. Also, in order to give the model the possibility to cause collisions when hazard, dangerous and unsafe maneuvers are performed, we had to set the *minGap*[3] = 0 and the *minGapLat*[4] = 0. At this point, vehicles were free to crash - both latitudinally and longitudinally - because they were not blocked anymore by *SUMO* at certain distances. In fact, since *SUMO* is a collision-free model, if it is not forced to reduce the minimum gap between vehicles to 0, vehicles cannot crash, despite the presence of reckless vehicles and maneuvers. Not only *SUMO*, but also *Flow* supplement its car following models with safe driving rules that prevent the inherently unstable car following models from crashing (Wu et al., 2017b).

As *SUMO* terminates its experiments when a collision occurs, *Flow* provides a more conservative course of action that is a *fail-safe mechanism*, called *the final position rule*. The *final position rule* forces each vehicle to keep a velocity such that if the preceding vehicle suddenly starts braking with max deceleration a, then even if the following vehicle has a delay τ, it can still slow down such that it comes to rest at the final position of the rear bumper of the preceding vehicle (Wu et al., 2017b). Based on *SUMO*, *Flow* leverages various safety features from *SUMO* in order to prevent longitudinal and lateral collisions. These fail-safes serve as bounds on the accelerations and lane-changes human and autonomous vehicles may perform and may be relaxed on any set of vehicles in the network to allow for the prospect of more aggressive actions to take place, as we did in our case study.

The action space containing the set of actions a vehicle is allowed to perform is made of accelerations and lane-changing. In fact, for tasks with k autonomous vehicles, the action space is a set of accelerations $c \in \mathbb{Q}^k$ and lane-changing decisions $d \in [-1, 1]$. This is because when following a predefined route, a vehicle performs *longitudinal* and *lateral* actions, where *longitudinal* is meant accelerations and *lateral* is meant lane-changing. The lane-changing values are rounded to the nearest integer in (-1, 0, 1) denoting lane-change left, don't lane-change and lane-change right, respectively; this keeps the action space representation continuous. In our work, we inserted both accelerations and lane-changes in our action space because we built a multi-lane ring network where, in addition to accelerations, lane-changes are allowed.

The observation space may be any set of state information the user wishes to provide to the agent to let the agent fully or partially know the state of the environment. In our experimental work, the *DRL* vehicle

[1] A timestep is the sampling frequency

[2] Ferrari Daytona SP3 0 to 100 km/h in 2.85 seconds

[3] By minGap we mean the distance between 2 vehicles longitudinally

[4] By minGapLat we mean the distance between 2 vehicles latitudinally

follows the *partially observed* setting, that is observing only the velocity of the autonomous vehicle, the velocity of its preceding vehicle and its relative position in respect to the preceding vehicle. At the state of the art, the reward function can be any function of vehicle speed, position, fuel consumption, acceleration, distance elapsed, etc. (Wu et al., 2017b; Martinelli et al., 2018b). As researchers working on a new scenario, we desired a different reward function from the predefined ones. After a long phase of trial and error, we came up with a final reward function that mixes different parameters we want to keep an eye on. Our system-level objective was mitigating the occurrences of collisions when cyberattacks happen in the network and we followed the road of creating cyberattacks while letting the autonomous vehicle (*AV*) learn how to mitigate collisions. More specifically, our *reward function* gives a positive reward if the velocity of the vehicle is near the averages velocities of the whole set of vehicles and if it is under the target velocity, while a penalty if the vehicle is accelerating and lane-changing too much.

In our experimental analysis, we tried to improve the traffic scenario when some cyberattacks are performed on some vehicles in the ring network. After different simulations involving different cyberattacks, we used *Flow* to implement a *DRL* algorithm able to stabilize the traffic flows and reduce the number of collisions in time, despite several cyberattacks happening at each step. We freely release for research purposes the Python scripts we developed and the SUMO scenario [5].

3 EXPERIMENTAL ANALYSIS

Below we provide the answers for RQ.

In Figure 3 we can see that after 3.500.000 timesteps (0.1 s) corresponding to 97.22 hours, only 3 collisions happened. For this reason, the collision frequency is 1 collision every 32.40 hours.

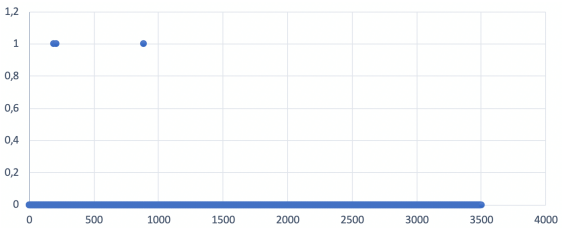

Figure 3: *RQ*. 3.500.000 timesteps of 0.1 s resulted in 3 collisions.

[5]https://mega.nz/file/wNkmXIiY#oiPCV2ZUCQSmu5tcgez2JnWU8TA89GPzNanM0owx7fM

In Figure 4 we can see that after 7.000.000 timesteps (0.1 s) corresponding to 194.44 hours, only 5 collisions happened. For this reason, the collision frequency is 1 collision every 38.89 hours.

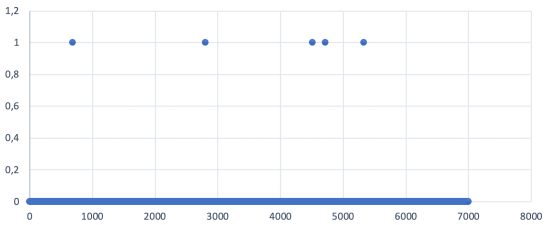

Figure 4: *RQ*. 7.000.000 timesteps of 0.1 s resulted in 5 collisions.

In Figure 5 we can see that after 10.594.600 timesteps (0.1 s) corresponding to 294.29 hours, only 6 collisions happened. For this reason, the collision frequency is 1 collision every 49.04 hours.

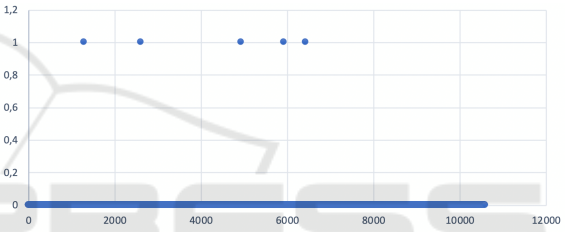

Figure 5: *RQ*: 10.594.6005 timesteps of 0.1 s resulted in 6 collisions.

Similarly to the previous trial, in this case in Figure 6 we can see that after 10.615.500 timesteps (0.1 s) corresponding to 294.88 hours, only 6 collisions happened. For this reason, the collision frequency is 1 collision every 49.15 hours.

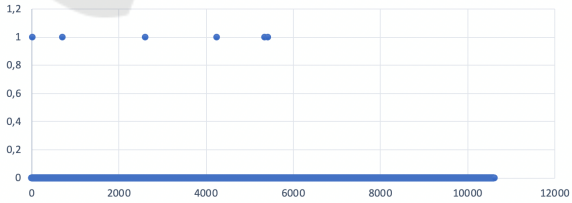

Figure 6: *RQ*: 10.615.500 timesteps of 0.1 s resulted in 6 collisions.

In Figure 7 we can see that after 21.659.500 timesteps (0.1 s) corresponding to 601.65 hours, only 10 collisions happened. For this reason, the collision frequency is 1 collision every 60.16 hours.

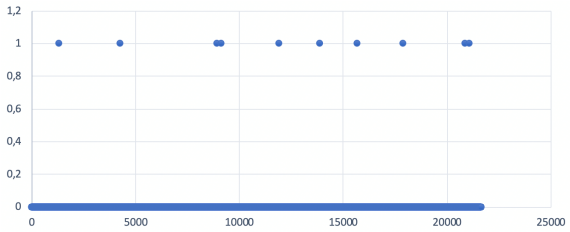

Figure 7: *RQ*: 21.659.500 timesteps of 0.1 s resulted in 10 collisions.

In Figure 8 we can see that after 32.770.035 timesteps (0.1 s) corresponding to 910.28 hours, 17 collisions happened. For this reason, the collision frequency is 1 collision every 53.55 hours. The overall collision frequency is good, but in this case we noticed a *collisions phase* after a *no collisions phase* from timestep 15.000 because of the worsening of the mean reward reached by the model.

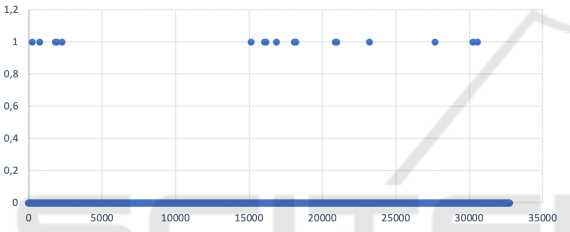

Figure 8: *RQ*: 32.770.035 timesteps of 0.1 s resulted in 17 collisions.

Table 1: Results related to RQ (19 human vehicles and 1 DRL vehicle).

Trial	Steps(0.1s)	Hours	Collision	Collision frequency
1	3.500.000	97.22 h	3	1 co1/32.40 h
2	7.000.000	194.44 h	5	1 co1/38.89 h
3	10.594.600	294.29 h	6	1 co1/49.04 h
4	10.615.500	294.88 h	6	1 col/49.15 h
5	21.659.500	601.65 h	10	1 col/60.16 h
6	32.770.035	910.28 h	17	1 co1/53.55 h

In table 1, results for each simulation are summarized, showing that - with the training progress - the collision frequency was reduced from 1 collision every 32.40 hours to 1 collision every 53.65 hours.

3.1 Discussion

Our results showed that just 1 *DRL* vehicle can help to optimize traffic flows and mitigate the number of collisions that would happen if no autonomous vehicles were inserted in the network.

Answering RQ, we showed that the collision frequency decreases while the number of simulation steps is increasing: this means that the *DRL* agent learns from the environment while bad behaviours happen on some vehicles. In fact, observing the results related to the training iterations, we can see that collision frequencies range from 1 collision every 32.40 hours (3.500.000 steps) to 1 collision every 53.55 hours (32.770.035 steps). The fifth trial related to 21.659.500 steps that resulted in a collision frequency of 1 collision every 60.16 hours is a good result but can be seen as an outlier because it represents the only case in which the next trial (trained on more steps) performs worse. Generally, the model exhibits a consistent and expected behaviour over time.

4 RELATED WORK

In this section, related works are discussed.

The statistics by the National Safety Council (NSC) of the United States reported that more than 40.000 roadway fatalities happened in 2017 (BOMEY, 2018). The primary reason for these road tragedies is the human factor, where reckless driving is the most considerable one (Saiprasert and Pattara-Atikom, 2013), (Yu et al., 2016).

In 2015, two cybersecurity researchers published a report (Miller and Valasek, 2015) showing how someone can wirelessly control a Jeep Cherokee after shattering the vehicle's Uconnect system.

In 2011, researchers at the *University of California at San Diego* and at the *University of Washington* found ways into a Chevy Impala's innards that included everything, e.g OnStar connection to a hacked smartphone connected to its infotainment system via Bluetooth (Drozhzhin, 2015).

Martinez et al. (Sikander and Anwar, 2018) showed that the driving state such as velocity and acceleration can be used to determine the driving performance. Based on the corrected results, the system may provide either passively or actively corrective feedback to the driver. A sort of feedback mechanism is involved also in *DRL* through the reward function and the progressive learning by the model. In our case, relying on *DRL*, feedback is implicit and automatic: the vehicles in the network are able to adapt to what's happening, so reckless behaviours are mitigated by the *DRL* vehicles themselves.

5 CONCLUSION AND FUTURE WORKS

In this paper, we propose the application of *DRL* in the context of autonomous driving. At each step of our experiment, we cyberattacked some vehicles in

the network to let them behave in an unexpected and erroneous way, while monitoring if an autonomous vehicle trained with a customized *DRL* algorithm was able to minimize the number of road accidents (collisions) over time.

The experimental analysis results showed that only 1 *DRL* vehicle can help to optimize traffic flows in order to mitigate the number of collisions that realistically would happen if no autonomous vehicles were inserted in the network. We showed that the collision frequency decreases while the number of steps increases: this means that the *DRL* agent is able to learn from the environment despite the presence of bad behaviours.

As future work, we plan to add an increasing number of *DRL* vehicles in the network with the aim to create more complex scenarios to evaluate the performances of the proposed *DRL* model based on a reward function that penalizes strong accelerations and lane-changes, while rewards velocities under a target velocity and near to the mean velocity of all the vehicles in the network. Also, we would like to execute a *no-DRL* scenario with the same network configuration in order to compare *DRL* performance with *no-DRL* performance.

ACKNOWLEDGEMENTS

This work has been funded by Spike Reply and partially supported by MIUR - SecureOpenNets, EU SPARTA, CyberSANE and E-CORRIDOR projects.

REFERENCES

BOMEY, N. (2018). U.S. vehicle deaths topped 40.000 in 2017, National Safety Council estimates. https://eu.usatoday.com/story/money/cars/2018/02/15/national-safety-council-traffic-deaths/340012002/. Accessed: Nov-2021.

Drozhzhin, A. (2015). Black hat usa 2015: The full story of how that jeep was hacked. https://www.kaspersky.com/blog/blackhat-jeep-cherokee-hack-explained/9493/. Accessed: Nov-2021.

Khatoun, R. and Zeadally, S. (2017). Cybersecurity and privacy solutions in smart cities. *IEEE Communications Magazine*, 55(3):51–59.

Martinelli, F., Marulli, F., Mercaldo, F., and Santone, A. (2021a). Neural networks for driver behavior analysis. *Electronics*, 10(3):342.

Martinelli, F., Mercaldo, F., Nardone, V., Orlando, A., and Santone, A. (2018a). Cluster analysis for driver aggressiveness identification. In *ICISSP*, pages 562–569.

Martinelli, F., Mercaldo, F., Nardone, V., Orlando, A., and Santone, A. (2018b). Who's driving my car? a machine learning based approach to driver identification. In *ICISSP*, pages 367–372.

Martinelli, F., Mercaldo, F., Nardone, V., and Santone, A. (2021b). Driver identification through formal methods. *IEEE Transactions on Intelligent Transportation Systems*.

Miller, C. and Valasek, C. (2015). Remote exploitation of an unaltered passenger vehicle. *Black Hat USA*, 2015(S 91).

Saiprasert, C. and Pattara-Atikom, W. (2013). Smartphone enabled dangerous driving report system. In *2013 46th Hawaii International Conference on System Sciences*, pages 1231–1237. IEEE.

Sikander, G. and Anwar, S. (2018). Driver fatigue detection systems: A review. *IEEE Transactions on Intelligent Transportation Systems*, 20(6):2339–2352.

Statista.com (2021). Projected size of the global autonomous car market from 2019 to 2023. https://www.statista.com/statistics/428692/projected-size-of-global-autonomous-vehicle-market-by-vehicle-type/. Accessed: Nov-2021.

Sutton, R. S. and Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press.

Treiber, M. and Kesting, A. (2013). Traffic flow dynamics. *Traffic Flow Dynamics: Data, Models and Simulation, Springer-Verlag Berlin Heidelberg*.

Wu, C., Kreidieh, A., Parvate, K., Vinitsky, E., and Bayen, A. M. (2017a). Flow: A modular learning framework for autonomy in traffic. *arXiv preprint arXiv:1710.05465*.

Wu, C., Kreidieh, A., Parvate, K., Vinitsky, E., and Bayen, A. M. (2017b). Flow: Architecture and benchmarking for reinforcement learning in traffic control. *arXiv preprint arXiv:1710.05465*, page 10.

Yu, J., Chen, Z., Zhu, Y., Chen, Y., Kong, L., and Li, M. (2016). Fine-grained abnormal driving behaviors detection and identification with smartphones. *IEEE transactions on mobile computing*, 16(8):2198–2212.