ULTRA: A reinforcement learning generalization benchmark for autonomous driving

Mohamed Elsayed*
Noah's Ark Lab
Huawei Technologies Canada, Ltd
Edmonton, Canada
mohamed.elsayed2@huawei.com

Kimia Hassanzadeh*
Noah's Ark Lab
Huawei Technologies Canada, Ltd
Edmonton, Canada
kimia.hassanzadeh@huawei.com

Nhat M. Nguyen*
Noah's Ark Lab
Huawei Technologies Canada, Ltd
Edmonton, Canada
minh.nhat.nguyen@huawei.com

Montgomery Alban
Noah's Ark Lab
Huawei Technologies Canada, Ltd
Toronto, Canada
montgomery.alban@huawei.com

Xiru Zhu
McGill University
Department of Computer Science
xiru.zhu@mail.mcgill.ca

Daniel Graves
Noah's Ark Lab
Huawei Technologies Canada, Ltd
Edmonton, Canada
daniel.graves@huawei.com

Jun Luo
Noah's Ark Lab
Huawei Technologies Canada, Ltd
Toronto, Canada
jun.luo1@huawei.com

Abstract

The unprotected left turn is one of the most difficult problems in real-world autonomous driving. Its difficulty is due to the diverse and hard-to-predict interactions among possibly many road participants in the absence of traffic lights. To help address this challenge, we developed a new benchmark called ULTRA (Unprotected Left Turn for Robust Agents). ULTRA offers controllable diversity and a way to measure the generalization performance of agents. It is also readily expandable as more scenarios and behavior models are developed and incorporated. Unlike prior benchmarks, ULTRA is explicitly focused on providing a rich diversity of interaction scenarios. In this way, it challenges the RL community to develop algorithms and driving policies that generalize better and are thereby more suitable for real-world autonomous driving. Our code is available at https://github.com/huawei-noah/ULTRA

*These authors contributed equally to this work

Machine Learning for Autonomous Driving Workshop at the 34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

1 Introduction

Autonomous driving is challenging partly due to the diversity of dynamic interactions among vehicles [26, 39]. One of the most challenging problems is the handling of intersections [26] where most accidents occur [9]. A particularly difficult case in point is the unprotected left turn [34, 26, 9] where an autonomous vehicle must predict the best time to cross traffic in possibly multiple other directions by simultaneously taking into account how others vehicles may behave and how itself may drive. Moreover, the type of social vehicles (i.e. vehicles other than the autonomous vehicle being considered) plays a significant role in crashes at intersections due to differences in turning behavior between heavy and light vehicles [34].

There are many ways to address the unprotected left turn challenge, including predicting the trajectories of other vehicles [24, 11], predicting the intentions of other vehicles [29, 16], or modeling the uncertainty in risk estimates [16]. Reinforcement learning (RL) is also a popular approach to the unprotected left turn problem [10, 5, 30, 32, 3, 15] partly due to its demonstrated capacity for learning good policies when handling complex situations such as those in Go [28] and Starcraft II [33]. However, when RL is applied to the real world, several challenges arise; partial observability of state, high diversity, poor generalization [14, 7, 40], safety criticalness, and non-stationarity.

Our objective is to develop a benchmark that is useful for both the autonomous driving and the RL communities. This benchmark evaluates the ability of agents to generalize by presenting them with many different unprotected left turn scenarios that feature a high level of interaction diversity and partial observability in a safety-critical setting. We call our benchmark *Unprotected Left Turn for Robust Agents*, or *ULTRA* for short. We understand robustness informally as generalizable performance, reliability, and safety in the presence of scenario diversity and limited observability and expect good RL generalization to be a key source of such robustness.

Many forms of diversity exist in driving: (1) visual diversity, (2) behavioral diversity of other drivers and pedestrians [26], (3) situational diversity [9] [39], and (4) vehicle diversity [34]. Instead of trying to be all encompassing, ULTRA focuses on providing scenarios that test the agent's ability to robustly handle behavioral diversity, situational diversity, and vehicle diversity. While there are some excellent autonomous driving simulation platforms that emphasize visual fidelity, such as [8], we have chosen to build ULTRA on top of a recent autonomous driving simulator SMARTS [1] that focuses on behavioral diversity of road participants.

Furthermore, while the unprotected left turn problem is in many ways a multi-agent problem, real-world autonomous driving does not fit nicely into standard multi-agent problem formulations. Firstly, it is neither purely competitive nor purely cooperative. Secondly, most vehicles lack the ability to communicate and coordinate with each other in a standard way. Thirdly, other drivers cannot be assumed to be acting optimally as is common in multi-agent RL [4] or even maximizing the same reward. Leaving that issue of multi-agent formulation for future work, we instead make the simplifying assumption that in any given scenario there is only one "ego agent" that is learning among "social agents" with mature policies that are no longer changing. However, note that this setting is compatible with the ego agent learning to model the social agents based on the observable behavior of the social vehicles under their control. Thus, ULTRA casts the multi-agent situation into an observability issue and assumes that the primary source of partial observability is that the decision making processes of the social agents are not directly accessible.

ULTRA turns the unprotected left turn—a most challenging problem from real-world autonomous driving—into a generalization benchmark in machine learning for autonomous driving (ML4AD). We hope that ULTRA will make it easier for ML4AD researchers to systematically and accurately assess solutions for complex interaction, and thereby help to stimulate new researches that will eventually enable autonomous vehicles to fluidly handle dynamic real-world interactions.

2 Benchmark

The desiderata for the ULTRA benchmark is that it must (1) support many varied scenarios, (2) evaluate robustness to variability, and (3) provide a graph-based observation. Fig. 1 shows the ULTRA environment developed in SMARTS [1]. The goal for the ego vehicle (red) is to move from south to west to reach the goal while respecting the road rules and interacting with other social vehicles (yellow).

Robustness to variability is evaluated similar to [22, 38] by defining a task as a set of training scenarios and testing scenarios where the agent learns in the training scenarios and is evaluated in the testing scenarios. A scenario is a combination of map and randomly generated vehicle flow patterns that describe social vehicle routes to populate, their entry and exit conditions, the types of vehicles, types of behaviors, and traffic densities. The map includes intersection type, number of lanes, and vehicle speeds. Fig. 2 highlights the scenario diversity in ULTRA.

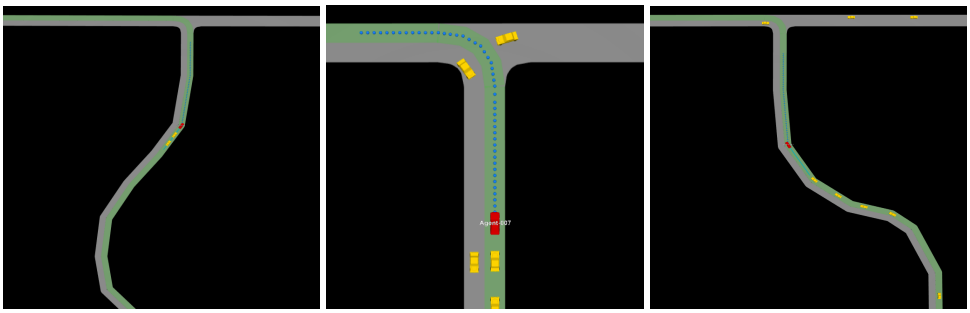


Figure 1: ULTRA environment in SMARTS (red is the ego vehicle whereas yellow is the social vehicle). The ego vehicle must turn left to reach the goal located in the west part of the route.

2.1 Variability

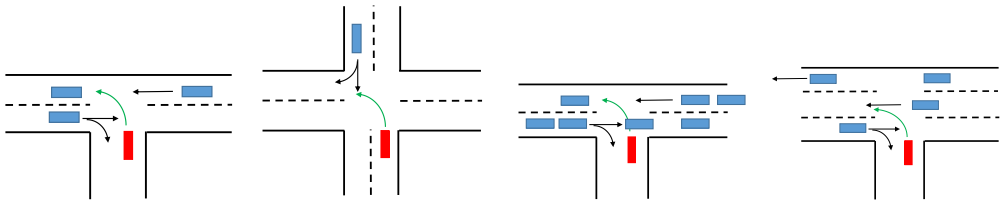
The unprotected left turn environment must provide controllable diversity of social agent behaviors, different traffic densities, different types of intersections, and different types of vehicles.

Controllable Diversity: Diversity of the social agents is a critical requirement for this benchmark. The diversity must be controllable such that the scenario is repeatable across training runs regardless of any other details in implementation. This is accomplished by generating a configurable number of randomized scenarios and forming tasks around these scenarios. Scenarios are selected randomly during training; the result is a diverse yet deterministic training environment. The diversity is achieved through a variety of social behaviors, traffic flow rates, and number of lanes. These different social behaviors, traffic flow rates and number of lanes are discussed in more detail in Appendix 6.1 and 6.2.

Controllable traffic density: A vehicle is generated according to its emission probability which is a configurable parameter that controls the traffic density in each scenario. In this paper, we use three traffic density levels, low, mid, and high. These density levels are defined in Appendix 6.2.

Different Types of Intersections: There are 2 basic intersection types currently supported by ULTRA and consist of (1) the T intersection, and (2) the cross intersection as shown in Fig. 2a and 2b. For each intersection, the number of lanes ranges between 2 to 6 as well 3 different speed limit variations consisting of 50 km/h, 70 km/h and 100 km/h. The combination of different intersection designs and speed limits results in a total of 30 types of intersection scenarios.

Different Types of Vehicles: There are 5 basic types of vehicles currently supported by ULTRA which include cars, buses, trailers, coaches, and trucks. Each vehicle type has different physical sizes, dynamics profiles, and behaviors which is discussed in Appendix 6.1.



(a) T Intersection (b) Cross Intersection (c) High Traffic (d) Multi-lane

Figure 2: Scenario diversity in ULTRA.

2.2 Observation and Action Spaces

The ego agent is the learning agent. It receives actions for throttle, brake, and steering angle where the steering of the vehicle is provided by an Ackermann steering mechanism. The observation space consists of the ego information, the social agents, and the information from the map. The ego information includes its speed, current steering angle. The information from the map includes the distance from the center of the route on the map, the position of the goal relative to the ego position, several future route waypoints, the angle deviation from the desired route, and the speed limit. This is a highly simplified representation since the agent is not provided with details like traffic lights or the number of lanes, although we plan to add these in the future. The number of waypoints is configurable and defines how much of the map is visible to the agent.

The social agent's information is a variable length list consisting of information from each social agent within a radius of the ego vehicle. Each social agent information includes speed, heading relative to the ego heading, and position relative to the ego position. The information about social vehicles is designed to be organized into a graph structure. While it is clearly challenging to encode the graph information, we describe some example encoders in the Appendix 6.8.

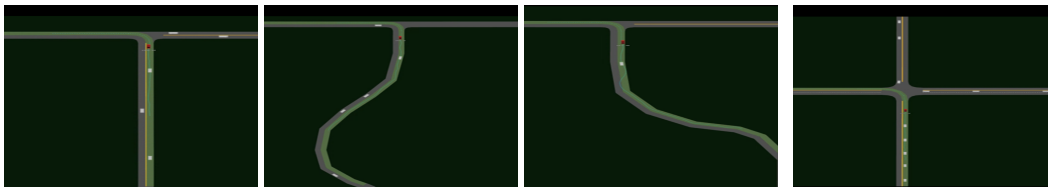
Graph Structured Observations: Given a recent explosion of graph-based deep learning [36], this benchmark provides an opportunity to train RL agents that observe the world as a graph with a set of vertices given as the set of social agents in the vicinity of the ego agent. The vertices are attributed with relative position, velocity, heading, and other characteristics of the vehicles. The design of the adjacency matrix is left to the researcher since this is an open area of research in autonomous driving. A common approach computes the adjacency matrix based on relative distance. In some cases, an adjacency matrix is not required; for example, PointNet can be used to process the social agents as a list of points. The graph of social agents is dynamic in the sense that the number of vertices changes over time. Our justification for this representation is that there are very few benchmarks for graph-based RL and yet this is a common representation in autonomous driving [24, 11].

Partial Observability: While the position, velocity, and heading of the social agents are known, their behavior is not known and thus this benchmark presents a partial observability challenge for learning policies. The true state is not known to the RL agent. Formally, this is commonly characterized by a partially observable Markov decision process (POMDP) [37, 16, 27]. The only observable attributes of the social agents are position relative to ego, velocity, heading and vehicle class label (e.g. car, truck, bus, etc). While we acknowledge that this is a simplification of autonomous driving where other characteristics of the vehicle, including the driver's visual characteristics (e.g. sex and age of the driver), and color of the vehicle can be correlated with different driving behaviors, these characteristics are difficult to model accurately in simulation and beyond the scope of this benchmark. We believe the varied vehicle types and behaviors provides enough diversity to challenge RL agents to learn robust policies under uncertainty, particularly in novel situations on the road.

2.3 Evaluation metrics

After designing the tasks and generating scenarios, the agent is presented with a random scenarios either from the training set during training or testing set during evaluation. Metrics are recorded to evaluate performance and some are used directly in the reward function. These metrics are typically computed over the length of an episode where an episode ends when the vehicle (1) travels off road, (2) travels off route, (3) reaches the goal, or (4) times out before any of the other events occur. Each episode is summarized according to the following metrics: (1) travelled off road, (2) travelled off route, (3) reached the goal location, (4) timed out before reaching the goal location, (5) the episode length, (6) the average speed, (7) the distance travelled, (8) the average distance from the center of the route, (9) the number of safety violations, and (10) final distance to the goal location. More details about the definition of these metrics are given in Appendix 6.7.

Generalization and Benchmarking: This platform can create benchmarks to evaluate generalization. Generalization is described using a similar method to ProcGen [6]. First, the task is designed according to training set and testing set configurations), then a number of scenarios is generated for each set. The level of performance on the testing set compared to the training set indicates the level of generalization to the new scenarios. However, to design a benchmark, one needs to fix the number of scenarios in each set. We can say an algorithm A generalizes better than algorithm B to the testing set if the difference in performance between training and testing is lower for the algorithm.



(a) Training (b) Evaluation

Figure 3: Task-1 Intersection designs where the training set consists of variations of the T-intersection and the testing set consists of variations of the cross intersection

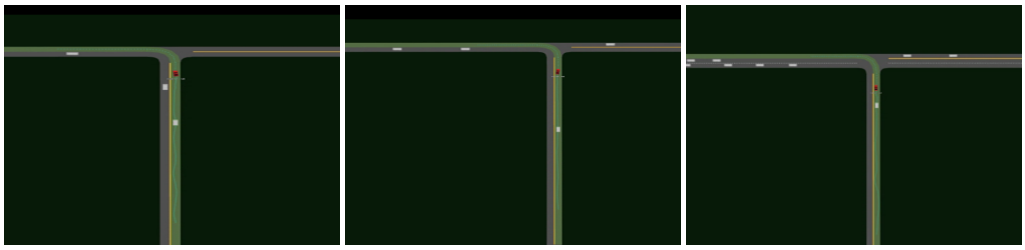


Figure 4: Task-2: Traffic densities are depicted for low, medium, and high traffic where the training set and testing set consist of different distributions of each traffic density.

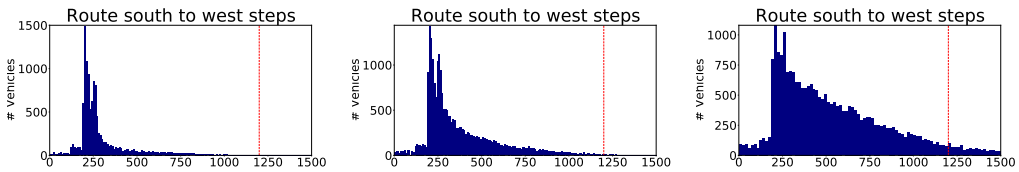
Scenario Solvability: The scenarios are generated such that many of the scenarios can be solved. Our definition of "solved" is where the number of steps in an episode is less than a fixed threshold 90% of the time according to a pre-defined behavior policy. The heavier traffic densities can result in occasional deadlocks and thus special care was taken to understand when these occur and how to mitigate them. The scenarios and variations in social agent behaviors and traffic densities were tuned to minimize the risk of a scenario being unsolvable. The detailed analysis for solvability is described in subsection 3.1.

3 Benchmark Tasks

We define two tasks to evaluate generalization for different intersection designs and different traffic densities. In Task-1 the agent is trained on straight and curvy t-intersections (Fig. 3a) and is evaluated on cross-intersections (Fig. 3b). This task is designed to be easy as the majority of traffic distribution in both train and test is low-density (61%) and the rest is mid-density (33%), high-density (3%) and no traffic (3%). Task-2 evaluates generalization to different traffic flows including traffic flows heavier than the training data and lighter than the training data. All tasks involve different lane configurations at 3 speed limits (50 km/h, 70 km/h and 100 km/h) (Fig. 2d). The minor road is always 2 lanes at 50 km/h. More details for each task are given in Appendix 6.3. The training and evaluation results for task 1 are given in Fig. 8 and 9. Task 2 evaluates the generalization among different distributions of traffic densities depicted in Fig. 4 where the low-high task that tests generalization from lower to higher traffic densities is given in Table 1. The training set and testing set of the low-high task is reversed to create the high-low task that tests generalization of traffic densities in the reverse direction.

Table 1: Task-2 traffic density distributions for the training set and testing set

	Low Traffic	**Medium Traffic**	**High Traffic**	**No traffic**
Training	61%	33%	3%	3%
Testing	3%	33%	61%	3%



(a) Low (b) Medium (c) High

Figure 5: Histogram of number of time steps needed for a vehicle from ego lane to exit the scenarios for different traffic densities

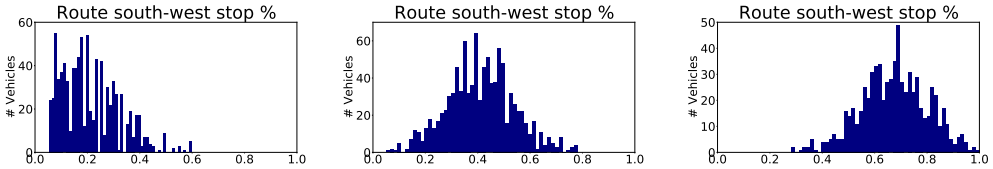
3.1 Analysis of the T-Intersection Scenarios

The T-intersection is using is all of our training sets and many of our testing sets and thus we conduct a thorough analysis of the T-Intersection here. Our analysis focused on characterizing the solvability and diversity of the T-intersection problem.

Solvablity: To ensure that a learning agent can reach the goal in a reasonable amount of time, we designed the traffic densities and behavior distributions such as that a scenario can be solved in a reasonable amount of time for low, medium, and high traffic. We defined solvability as the expectation that a social vehicle from the ego lane controlled by a hand-crafted baseline policy can successfully exit the scenario within 1200 time steps more than 90% of the time. The hand-crafted policy is designed with SUMO [21]. Because the hand-crafted policy is not-optimal, the policies learned by RL should learn to solve the problem in similar or fewer steps. In other words, the solvability of the hand-crafted SUMO policy serves as a lower bound on the solvability of the scenario.

To obtain the solvability for a traffic density, 1000 randomly generated scenarios of that density were ran for 6000 time steps (distributed as 333, 333, and 334 for 50 km/h, 70 km/h, and 100 km/h speed limits respectively). The number of time steps needed for vehicles starting in the ego lane to successfully turn left were collected. The solvability for a given density is determined as the percentage of vehicles that successfully turn left within 1200 time steps. For all of our traffic densities, the solvability obtained by the SUMO policy calculated were all above 90%. This means that the solvability by an optimal policy would be above 90% as well. As the tasks are composed of scenarios that belong to these traffic densities, their solvability would be above 90% as well, satisfying our design requirements. This is shown in Fig. 5 where we illustrate a histogram of the number of total number steps for a vehicle from ego lane needed to finish the simulation for each traffic density. We can see that there is still a difference between the histograms of the densities, suggesting multiple levels of difficulty. Further information about the relation to routes or behaviors is provided in Fig. 12, 13, and 14.

Diversity: Next, we characterized the diversity of the T-intersection scenarios. The reason was to understand the diversity of the low, medium, and high traffic flow definitions. The objective was to determine suitable traffic and behavior definitions that resulted in distinctly different amounts of interaction to justify and characterize the traffic labels "low", "medium" and "high". Specifically, we characterized the percentage of all vehicles that stopped for other vehicles at least once as a surrogate metric for the diversity of interactions. The stop percentage for the south-west route measures the percentage of vehicles that stopped for any number of time steps either at the intersection or due to traffic jamming. To generate the plot, we run 1000 randomly generate scenarios the same way as in section 3.1. The distribution of the maximum number of vehicles in each route is given in table 4. For all tasks and levels, the same behavioral distribution is used which is defined in table 5. The number of vehicles that stop at least once is non-trivial for all densities. There are three different modes; they correspond to the three traffic densities in the histograms, where the amount of interactions increases progressively from low to high.



(a) Low traffic (b) Medium traffic (c) High traffic

Figure 6: Interaction amount for different traffic densities

4 Related Works

Benchmarks: There are several benchmarks in RL that tailor to evaluating the generalization of policies [6, 13, 38, 22]. The challenge they address is a well-known problem that RL overfits to the training environment [35, 41, 14, 7, 40]. To address this issue, benchmarks need to (1) test an agent's ability to better generalize [7] and (2) evaluate periodically during training [22].

One recent benchmark for testing generalization is ProcGen [6]. ProcGen aims to evaluate the generalization of the agent to level diversity through procedurally generated worlds. They find that 10,000 different training levels are needed to see generalization on unseen training levels. While the observation of the agent is image-based, it's unclear if algorithms that generalize well in ProcGen will generalize to autonomous driving where there is a far more diverse collection of possible scenarios and behaviors of other agents.

Another generalization challenge in RL is due to physical variations of the environment as found in [13]. The benchmark can be characterized as a multi-task, transfer, or lifelong learning benchmark that focuses on evaluating the suitability of a policy to different physical variations of the same task. As an example, physical variations include the strength of a joint or the mass of the robot. While this challenge can be important in robotics, including autonomous driving, due to the different size, mass, and dynamics of the vehicles, the social diversity in intersections remains the most challenging problem in autonomous driving [26].

Robotic manipulation is a popular real-world application of RL [38]. MetaWorld [38] is a benchmark consisting of 45 training tasks and 5 testing tasks that are created to evaluate multi-task and meta-learning performance. This benchmark addresses the challenge of generalizing over the goal-space, i.e. learning to solve new problems more quickly through meta-learning and few shot learning. While MetaWorld could be useful in autonomous driving to adapt prior learned skills to new skills, it addresses a different problem than the diversity challenge encountered in autonomous driving.

There are a few simulators and benchmarks in autonomous driving worth mentioning. CARLA is an open-source simulator with a growing number of users and tasks [8]. However, while CARLA focuses on visual fidelity based on game engine technology, the graphics computing overhead is substantial. Another autonomous driving environment called Highway-Env implemented an unprotected left turn scenario [19]. However, the unprotected left turn scenario lacks diversity particularly in the behavior of social agents and types of intersections. Another simulator is SMARTS [1] which is a new open-source autonomous driving simulator designed for learning policies for multi-agent RL.

Unfortunately, there are no standard benchmarks for evaluating the performance of RL agents in autonomous driving that addresses the problem of generalization given the vast diversity of scenarios and agent behaviors. ULTRA fills this need.

Unprotected Left Turn: There are many useful surveys of RL applied to autonomous driving [31, 2, 18]. This is beyond the scope of this paper. Instead, we will focus on the applications of RL and related methods to the unprotected left turn problem. Most methods recognize the partial observability of the intention of other drivers. One approach is to learn a hidden Markov model to learn their intention [29]. Another method is to predict the intention of other drivers in a partially observable Markov decision process (POMDP) motion planner [37]. In [16], the intention is predicted with a POMDP motion planner to provide risk-bound guarantees. A hierarchical approach is proposed in [27] where a candidate path generator chooses trajectories and a low-level POMDP planner for execution that ensures safe behaviors.

Modeling partial observability with uncertainty and risk is also common. A risk-sensitive reinforcement learning algorithm called worst case policy gradients is developed to avoid uncertain futures [32]. Their approach is based on distributional reinforcement learning where they compute the conditional value-at-risk measure. They tested on an unprotected left turn scenario in CARLA. An RL algorithm is developed that provides probabilistic guarantees for intersection scenarios [3]. In [15], Bayesian RL is used to estimate uncertainty and make better tactical-decisions in intersections.

Several other methods apply deep RL to the unprotected left turn problem. [10] built a custom simulator for the unprotected left turn where they applied DRL to choose between stop and go actions. [20] applies game theory using leader-follow game pairs to tackle uncontrolled intersections. [5] combines RL and imitation learning in a hierarchical way where RL chooses policies learned by imitation learning to tackle near accident driving scenarios including an unprotected left turn scenario. Structured control nets for deep reinforcement learning [30] is proposed for the unprotected left turn scenario by combining both linear and nonlinear control.

5 Conclusions

We introduce ULTRA as a benchmark for evaluating the generalization of policies, particularly but not exclusively those learned by reinforcement learning, to different scenarios and social behaviors in autonomous driving. We have chosen to focus on the unprotected left turn problem as the underlying problem in our benchmark because it is narrow enough for RL to be easily applied but broad enough for real-world diversity to be systematically incorporated and evaluated. A unique feature of ULTRA is that it offers a great deal of diversity supported by the expressive APIs of the underlying multi-agent simulation platform SMARTS [1]. Our initial version supports many different kinds of intersections, different speed limits, different traffic density levels, different social agent policies, and different social vehicle types. With variegated scenarios systematically available through standard training and evaluation interfaces, ULTRA allows us to evaluate RL generalization in a way that is both rigorous in machine learning terms and relevant to real-world autonomous driving.

Moving forward, we hope to expand the benchmark to include more scenarios, vehicle types, and intersection types (such as roundabouts). We also plan to leverage the "social agent zoo" of SMARTS, which is supposed to provide many diverse and realistic behavior models of road users, for generating even more diverse and realistic interactions for ULTRA. Additionally, we are also considering different levels of difficulty that are tailored to different levels of computational resources available for training and testing. In addition, we provide observations in the form of an interaction graph representing the relations among social agents. Unlike the most challenging environments today that focus on image-based observations, ULTRA presents a challenge to learn policies from graph-based observations. Finally, while we have chosen to present ULTRA as a generalization challenge for single-agent RL, we do hope that the proposed solutions will learn to model the behavior of other vehicles either implicitly or explicitly. And we plan to explicitly treat the multi-agent aspect of the unprotected left turn in future work.

Overall, ULTRA channels the unprotected left turn problem—one of the most difficult challenges in autonomous driving—into a machine learning benchmark and thereby makes real-world autonomous driving more tangible to the broad machine learning and RL research community. On the other hand, because the unprotected left turn is far from adequately solved in autonomous driving, we also hope that research stimulated by ULTRA will benefit the autonomous driving community who cares deeply about the real-world applicability of machine learning solutions.

References

[1] Smarts: An open-source scalable multi-agent rl training school for autonomous driving. 2020.

[2] Szilárd Aradi. Survey of deep reinforcement learning for motion planning of autonomous vehicles. *ArXiv*, abs/2001.11231, 2020.

[3] Maxime Bouton, Jesper Karlsson, Alireza Nakhaei, Kikuo Fujimura, Mykel Kochenderfer, and Jana Tumova. Reinforcement learning with probabilistic guarantees for autonomous driving. 04 2019.

[4] Lucian Busoniu, Robert Babuska, and Bart De Schutter. Multi-agent reinforcement learning: A survey. pages 1 – 6, 01 2007.

[5] Zhangjie Cao, Erdem Biyik, Woodrow Wang, Allan Raventos, Adrien Gaidon, Guy Rosman, and Dorsa Sadigh. Reinforcement learning based control of imitative policies for near-accident driving. In *Robotics: Science and Systems XVI*, Jul 2020.

[6] Karl Cobbe, Christopher Hesse, Jacob Hilton, and John Schulman. Leveraging procedural generation to benchmark reinforcement learning. *CoRR*, abs/1912.01588, 2019.

[7] Karl Cobbe, Oleg Klimov, Christopher Hesse, Taehoon Kim, and John Schulman. Quantifying generalization in reinforcement learning. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 1282–1289. PMLR, 2019.

[8] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017.

[9] Francesca M. Favarò, Nazanin Nader, Sky O. Eurich, Michelle Tripp, and Naresh Varadaraju. Examining accident reports involving autonomous vehicles in california. *PLOS ONE*, 12(9):1–20, 09 2017.

[10] Andreas Folkers, Matthias Rick, and Christof Büskens. Controlling an autonomous vehicle with deep reinforcement learning. 06 2019.

[11] Jiyang Gao, Chen Sun, Hang Zhao, Yi Shen, Dragomir Anguelov, Congcong Li, and Cordelia Schmid. Vectornet: Encoding hd maps and agent dynamics from vectorized representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[12] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft actor-critic algorithms and applications, 2018.

[13] Peter Henderson, Wei-Di Chang, Florian Shkurti, Johanna Hansen, David Meger, and Gregory Dudek. Benchmark environments for multitask learning in continuous domains. *Lifelong Learning workshop at ICML*, 08 2017.

[14] Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters, 2017. cite arxiv:1709.06560Comment: Accepted to the Thirthy-Second AAAI Conference On Artificial Intelligence (AAAI), 2018.

[15] Carl-Johan Hoel, Tommy Tram, and J. Sjoberg. Reinforcement learning with uncertainty estimation for tactical decision-making in intersections. 06 2020.

[16] Xin Huang, Sungkweon Hong, Andreas Hofmann, and Brian Williams. Online risk-bounded motion planning for autonomous vehicles in dynamic environments. 04 2019.

[17] Jun Jin, Nhat M. Nguyen, Nazmus Sakib, Daniel Graves, Hengshuai Yao, and Martin Jagersand. Mapless navigation among dynamics with social-safety-awareness: a reinforcement learning approach from 2d laser scans, 2019.

[18] B Ravi Kiran, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A. Al Sallab, Senthil Yogamani, and Patrick Pérez. Deep reinforcement learning for autonomous driving: A survey. 2020.

[19] Edouard Leurent. An environment for autonomous driving decision-making. `https://github.com/eleurent/highway-env`, 2018.

[20] Nan Li, Yu Yao, Ilya Kolmanovsky, Ella Atkins, and Anouck Girard. Game-theoretic modeling of multi-vehicle interactions at uncontrolled intersections. 04 2019.

[21] Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie Wießner. Microscopic traffic simulation using sumo. In *The 21st IEEE International Conference on Intelligent Transportation Systems*. IEEE, 2018.

[22] Marlos C. Machado, Marc G. Bellemare, Erik Talvitie, Joel Veness, Matthew Hausknecht, and Michael Bowling. Revisiting the arcade learning environment: Evaluation protocols and open problems for general agents. *J. Artif. Int. Res.*, 61(1):523–562, January 2018.

[23] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation, 2016.

[24] Nicholas Rhinehart, Rowan McAllister, Kris Kitani, and Sergey Levine. Precog: Prediction conditioned on goals in visual multi-agent settings. *International Conference on Computer Vision*, pages 2821–2830, 2019.

[25] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

[26] Wilko Schwarting, Alyssa Pierson, Javier Alonso-Mora, Sertac Karaman, and Daniela Rus. Social behavior for autonomous vehicles. *Proceedings of the National Academy of Sciences*, 116(50):24972–24978, 2019.

[27] K. Shu, H. Yu, X. Chen, L. Chen, Q. Wang, L. Li, and D. Cao. Autonomous driving at intersections: A critical-turning-point approach for left turns. 03 2020.

[28] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, jan 2016.

[29] Weilong Song, Guangming Xiong, and Huiyan Chen. Intention-aware autonomous driving decision-making in an uncontrolled intersection. *Mathematical Problems in Engineering*, 2016:1025349, Apr 2016.

[30] Mario Srouji, Jian Zhang, and Ruslan Salakhutdinov. Structured control nets for deep reinforcement learning. 02 2018.

[31] Victor Talpaert, Ibrahim Sobh, Bangalore Kiran, Patrick Mannion, Senthil Yogamani, Ahmad Sallab, and Patrick Perez. Exploring applications of deep reinforcement learning for real-world autonomous driving systems. 01 2019.

[32] Yichuan Charlie Tang, Jian Zhang, and Ruslan Salakhutdinov. Worst cases policy gradients. *3rd Conference on Robotic Learning*, 2019.

[33] Oriol Vinyals, Igor Babuschkin, Wojciech Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David Choi, Richard Powell, Timo Ewalds, Petko Georgiev, Junhyuk Oh, Dan Horgan, Manuel Kroiss, Ivo Danihelka, Aja Huang, Laurent Sifre, Trevor Cai, John Agapiou, Max Jaderberg, and David Silver. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575, 11 2019.

[34] Xinpeng Wang, Ding Zhao, Huei Peng, and David Leblanc. Analysis of unprotected intersection left-turn conflicts based on naturalistic driving data. pages 218–223, 06 2017.

[35] Shimon Whiteson, Brian Tanner, Matthew E. Taylor, and Peter Stone. Protecting against evaluation overfitting in empirical reinforcement learning. *2011 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, pages 120–127, 2011.

[36] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip Yu. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, PP:1–21, 03 2020.

[37] Ming-Yuan Yu, Ram Vasudevan, and Matthew Johnson-Roberson. Occlusion-aware risk assessment for autonomous driving in urban environments. *IEEE Robotics and Automation Letters*, 4(2):2235–2241, 2019.

[38] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. *CoRL*, 2019.

[39] Wei Zhan, Liting Sun, Di Wang, Haojie Shi, Aubrey Clausse, Maximilian Naumann, Julius Kümmerle, Hendrik Königshof, Christoph Stiller, Arnaud de La Fortelle, and Masayoshi Tomizuka. INTERACTION Dataset: An INTERnational, Adversarial and Cooperative moTION Dataset in Interactive Driving Scenarios with Semantic Maps. *arXiv:1910.03088 [cs, eess]*, September 2019.

[40] Chiyuan Zhang, Oriol Vinyals, Rémi Munos, and Samy Bengio. A study on overfitting in deep reinforcement learning. *CoRR*, abs/1804.06893, 2018.

[41] Chenyang Zhao, Olivier Sigaud, Freek Stulp, and Timothy M. Hospedales. Investigating generalisation in continuous deep reinforcement learning. *CoRR*, abs/1902.07015, 2019.

6 Appendix

6.1 Social Behaviors and Sizes

Diversity is a key element in ULTRA. We define different social behaviors that ranges from small to big vehicles and from cautious to aggressive behaviors. The definition of these behaviors and sizes are given in Tables 2 and 3. Each behavior is defined by acceleration, deceleration, minimum gap, impatience in junction, and lane change cooperation. The minimum gap is the minimum distance between the vehicle and the one in front of it. Impatience in junction is the willingness to wait at the intersection for other vehicles. Lane change cooperation is the willingness to help other vehicles change lane if needed. These parameters are defined in SUMO [21] as the underlying behavioral model provider. Later, we will add other models such as data-driven ones.

Table 2: Behavior definitions in SUMO

Behavior	**Accel.** (m/s^2)	**Decel.** (m/s^2)	**Min gap** (m)	**Impatience**	**Lane change cooperation**
Normal	3.0	7.5	2.5	Low	High
Aggressive	15.0	10.0	0.01	High	Low
Slow	3.0	10.0	5.5	Low	High
Blocker	1.0	10.0	5.0	Low	High
Crusher	20.0	2.0	0.0	High	Low
Bus	1.2	4.0	2.5	High	Low
Coach	2.0	4.0	2.5	High	Low
Trailer	1.1	4.0	2.5	High	Low
Truck	1.3	4.0	2.5	High	Low

Table 3: Vehicles' sizes

Vehicle	**Length** (m)	**Width** (m)	**Height** (m)
Car	4.3	1.8	1.5
Bus	12.0	2.5	3.4
Coach	14.0	2.6	4.0
Trailer	18.75	2.55	4.0
Truck	7.1	2.4	2.4

6.2 Road distributions and densities

To generate vehicles in the environment, we use emission probability, the expected to denote the probability of a vehicle spawning at timestep t, modeled by $(1 - emission)^t$. From this, we can obtain the spawn time of each vehicle on the route. The emission probability is applied independently to each lane; the number of vehicles scales with lane number. Thus, difficulty of a scenario scales with lane number as well. The emission probability for the low density, medium density, and high density are 0.02 (Fig. 6a), 0.04 (Fig. 6b), and 0.06 (Fig. 6c) respectively. To avoid situations where the ego agent is spawned before most social vehicles, leading to low amount of traffic interactions, each scenario is warmed up with all vehicles controlled by SUMO for fixed amount of time. A social vehicle moving in the south to west route is then hijacked to become the ego agent. This step makes sure that there is enough time for vehicles on other routes to be populated, increasing the probability of them having interactions with ego in the junction.

Table 4: Road distribution for different densities

Route	**Vehicles number**
West-south	100
South-west	100
South-east	100
East-south	100
East-west	1000
West-east	1000

Table 5: Social behavior distributions

Behavior	**Probability**
Normal	85%
Aggressive	5%
Blocker	5%
Cautious	5%

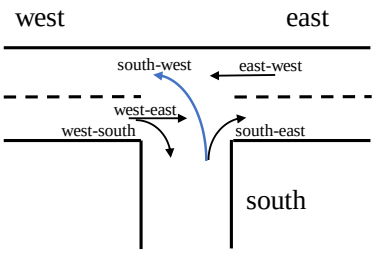


Figure 7: Routes directions

6.3 Tasks and Difficulty Levels

In ULTRA, we design two tasks. First, transfer from t-intersection to cross-intersection. Second, transfer from one traffic density into another (from high traffic to low traffic and the opposite). The purpose of the first task is to evaluate the generalization on another type of intersection whereas the purpose of the second task is to evaluate generalization to heavier traffics. Each task contains a set of training scenarios and a set of evaluation scenarios.

In each task, we define two level for each task. For task 1, we have 10000 scenarios in the training set and 200 in the testing set. The training set is defined as 50% 2-lane and 50% 3-lanes t-intersection configurations. For both configurations, they contain 21% low-density with a 50 km/h speed limit, 21% low-density with a 70 km/h speed limit, 20% low-density with a 100 km/h speed limit, 12% mid-density with a 50 km/h speed limit, 12% mid-density with a 70 km/h speed limit, 12% mid-density with a 100 km/h speed limit, 1% high-density with a 50 km/h speed limit, 1% high-density with a 50 km/h speed limit, and 1% high-density with a 100 km/h speed limit. The testing scenarios are designed to test generalization to the cross intersection type. The configurations and distributions are the same as the ones in the training set but with cross-intersections instead of t-intersections.

The hard level in task 1 consists of the same number of training and testing scenarios with the same lane configurations percentages. However, the specifications of the two configurations are different. The agent is trained on heavier traffic. For both configurations, they contain 1% low-density with a 50km/h speed limit, 1% low-density with a 70km/h speed limit, 1% low-density with a 100km/h

speed limit, 12% mid-density with a 50km/h speed limit, 12% mid-density with a 70km/h speed limit, 12% mid-density with a 100km/h speed limit, 21% high-density with a 50km/h speed limit, 21% high-density with a 50km/h speed limit, and 20% high-density with a 100km/h speed limit. The testing scenarios are different only in the intersection type the same as the testing scenarios in the easy level.

On the other hand, we have *high-to-low* and *low-to-high* levels for the second task. *low-to-high* represents training on low density and evaluating on high density, and vice versa with *high-to-low*. We use the same number of training and testing scenarios as in task 1. For *low-to-high*, the training set is defined as 50% 2-lane and 50% 3-lanes t-intersection configurations. The training set contains 21% low-density with a 50 km/h speed limit, 21% low-density with a 70 km/h speed limit, 20% low-density with a 100 km/h speed limit, 12% mid-density with a 50 km/h speed limit, 12% mid-density with a 70 km/h speed limit, 12% mid-density with a 100 km/h speed limit, 1% high-density with a 50 km/h speed limit, 1% high-density with a 50 km/h speed limit, and 1% high-density with a 100 km/h speed limit. The testing scenarios are designed to test generalization to the heavier traffics. The configurations and distributions are the same is defined as 50% 2-lane and 50% 3-lanes t-intersection configurations. The testing set contains 21% high-density with a 50 km/h speed limit, 21% high-density with a 70 km/h speed limit, 20% high-density with a 100 km/h speed limit, 12% mid-density with a 50 km/h speed limit, 12% mid-density with a 70 km/h speed limit, 12% mid-density with a 100 km/h speed limit, 1% low-density with a 50 km/h speed limit, 1% low-density with a 50 km/h speed limit, and 1% low-density with a 100 km/h speed limit. Results of this level in task 2 are shown in Fig, 10.

For *high-to-low*, it is the almost the same as *low-to-high* but all high and low densities are switched. The purpose of this level is to evaluate generalization to low traffics after training on high traffics. Results of this level in task 2 are shown in Fig. 11.

6.4 Algorithms and Hyperparameters

In our experiments, we report the results obtained by two algorithms, PPO [25] and SAC [12]. The hyperparameters of each are given by table 6.

Table 6: Algorithms Hyperparameters

PPO		**SAC**	
Batch size	2048	Replay buffer size	10^6
Adam stepsize	0.001	Adam stepsize	0.0003
Num. epochs	10	target smoothing coefficient (τ)	0.005
Minibatch size	128	target update interval	1
γ	0.99	γ	0.99
λ	0.96	gradient steps	1
number of hidden layers	2	number of hidden layers	2
number of hidden units per layer	256	number of hidden units per layer	256

6.5 Training and Evaluation Results

In benchmark experiments, we use two state-of-the-art algorithms PPO [25] and SAC [12] described in appendix table 6. The tuning of hyper-parameters is completed using no-traffic (having no social vehicles) and low-traffic scenarios. Episode termination events are collision, going off-road, timing out, and reaching the goal. The maximum steps for the time-out event is set to 1200 and each time-step corresponds to 0.1 seconds in real-time. The evaluation is run every 10000 observation iterations on a separate set of scenarios and results are averaged over 200 different fixed seeds. Reward signal and evaluation metric are discussed in details in section 6.7. Environment setup is consistent for all tasks but scenarios are adjusted accordingly. Benchmark tasks are designed to challenge the agent's

generalization skills by changing the density of traffics and types of intersections. For task definitions and training results refer to Appendix 6.3.

We show the training and evaluation plots for our two tasks. The plots show collisions, distance to road center, distance travelled, reaching goal signal, timeouts, ego-safety violations, social-safety violations, episode reward, distance to goal, speed violations, off-roads, and speed. All of these data are plotted as a function of the time steps. We train the agent in each experiment for 1M steps. The training data are plotted and filtered by a moving average filter with a window of 100 steps. These plots are shown in Fig. 8, 9, 10, and 11.

6.6 Other statistics

Further statistics about our scenarios to understand the collective behavior by agents with the high diversity offered by our framework are provided in Fig. 12, 13, and 14. For each traffic density, the following statistics are collected and plotted in a histogram form.

- **Steps taken by each vehicle type to exit the episode.** This depends on the behaviors of different vehicle types. It allows us to partly see how different the vehicle types are. (Fig. 12a, 13a, and 14a)
- **Steps taken by vehicle on each route to exit the episode.** This depend on the distributions of vehicle on each route and the interaction between the routes. It allows us to grasp how dense traffic is and how difficult it is to navigate traffic in the intersection. (Fig. 12b, 13b, and 14b)
- **Percentage of vehicles that got stopped at any point during its time in the episode, aggregated by route.** A vehicle is defined as being stopped at a time step if its speed at that time step is 0. This quantity is a surrogate for the number of meaningful interactions between vehicles in an episode for each route. In the simulation, a social vehicle only stop due to yielding or avoiding collision with another vehicle. (Fig. 6)

6.7 Rewards and Evaluation Metrics

The agent interacts with the environment for 600 time steps where it gets rewarded based on the reward function R_t given by Eq.1 after getting an observation O_t which presents partial information about the environment state S_t.

$$\begin{aligned} R_{t+1}(S_t) = r_{\text{collision}} + r_{\text{off road}} + r_{\text{reached goal}} + r_{\text{dist to center}} \\ + r_{\text{angle error}} + r_{\text{ego safety}} + r_{\text{social safety}} + r_{\text{environment}} \end{aligned} \tag{1}$$

with

$$r_{\text{collision}} = -10.0 \text{ if ego collides with another vehicle} \tag{2}$$

$$r_{\text{off road}} = -1.0 \text{ if ego goes out of the road} \tag{3}$$

$$r_{\text{reached goal}} = 100.0 \text{ if ego reach the desired destination} \tag{4}$$

$$r_{\text{dist to center}} = -0.002 * \min(4, d_t^{\text{way point}}) \text{ if ego reach the desired destination} \tag{5}$$

$$r_{\text{dist to center}} = -0.005 * \max(0, \cos\theta_t) \tag{6}$$

$$r_{\text{ego safety}} = -0.02 \text{ if ego safety is violated} \tag{7}$$

$$r_{\text{social safety}} = -0.02 \text{ if social safety is violated} \tag{8}$$

$$r_{\text{environment}} = \frac{r_{\text{SMARTS}}}{10} \tag{9}$$

where θ_t is the difference between heading of the ego and the lane heading at the current time step, v_t is the speed of ego at the current time step, $d_t^{\text{way point}}$ is the distance from the ego to the middle of the road at the current time step. r_{SMARTS} is the step reward that is coming from SMARTS.

The reward is designed to encourage behaviors that will follow the road rules and respect other vehicles while maintaining safety. $r_{\text{collision}}$ penalizes the agent for colliding with any social vehicle. $r_{\text{off road}}$ penalizes the agent for going off road. $r_{\text{reached goal}}$ rewards the agent after reaching the desired

goal position. $r_{\text{dist to center}}$ penalizes the agent for going away from the center of the road. $r_{\text{ego safety}}$ and $r_{\text{social safety}}$ are designed to make the agent maintain safety. The methods to detect collision, wrong way, off road, reached goal and to compute ego speed, difference between ego heading and lane heading, location of center of the road are provided by SMARTS. The criteria used for determining whether ego safety or social safety are violated are given in [17] where $t_c = 1s$ and $d_{min} = 1m$ for both the ego and social vehicles. Finally, $r_{\text{environment}}$ is the scaled down reward from SMARTS, which rewards the agent according to the distance travelled.

6.8 Social Vehicles Encoders

For handling the graph structured information about social vehicles, three encoders are used. Low level features of social vehicles observed from the environment are relative position with respect to the ego vehicle, the relative heading, and speed. These encoders can handle any number of social vehicles. The output of each encoder is a fixed length vector that is concatenated to other observation elements.

6.8.1 Direct encoder

In this encoding scheme, each social vehicle that exist in the scenario is represented by their own low level features. Instead of considering all social vehicles, we only select a fixed number of social vehicles that are prioritized based on these criteria: (1) highest priority are nearest social vehicles in front and rear of ego in its current lane, (2) social vehicles on other lane are prioritized by their distance to ego, nearer vehicle has higher priority. We then select the top prioritized vehicle as output from the direct encoder. In case there are less vehicles than the number needed, fake vehicles with fixed low level features are padded in. Additionally, a permutation of the vehicles are performed to make downstream task resilient to effect of permutation.

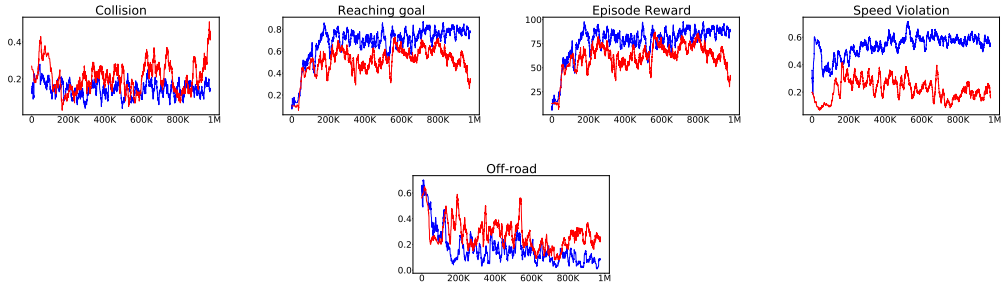
6.8.2 Shared-FC

At each time step, the agent can observe a variable number of social vehicles. For each social vehicle, its features are fed into a shared fully connected network between all of the observed social vehicles. The encoding of each vehicle information is concatenated. To maintain a fixed length vector, we have a limit on the number of vehicles that can be detected.

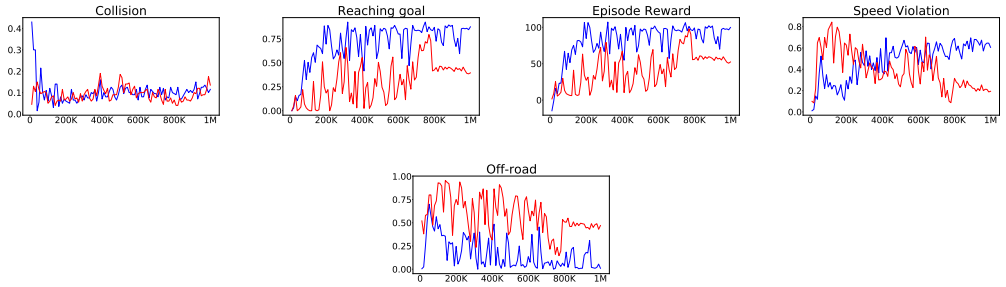
6.8.3 PointNet

PointNet [23] takes a batch of points (i.e. a point cloud) with variable batch size as input and output their global feature, which can then be used for downstream tasks. The points are first passed independently through a number of shared fully connected layers, resulting in a batch of point features with the same length. Then, the batch of point features are aggregated into a two-dimension matrix and a maxpooling is applied along the batch dimension to give the final fixed length feature.

In order to make the representation of the point cloud invariant to geometric transformations in the original point space as well as in the intermediate feature spaces, transformation matrices are applied to the original coordinates of the points and their intermediate features before each shared fully connected layer to undo the effects of such transformations. Each transformation matrix is predicted from the batch using a similar idea as above: the point coordinates/features are passed independently through multiple shared fully connected layers, followed by by a max pooling to get the feature of the point cloud, this feature is then passed through a couple of fully connected layers to regress the transformation matrix of the batch. To keep optimization easy during training, the transformation matrices are constrained to be close to be orthogonal via an auxiliary loss objective.

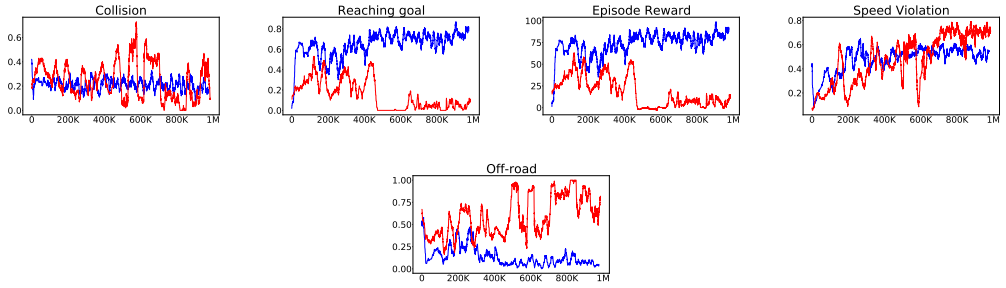


(a) Training

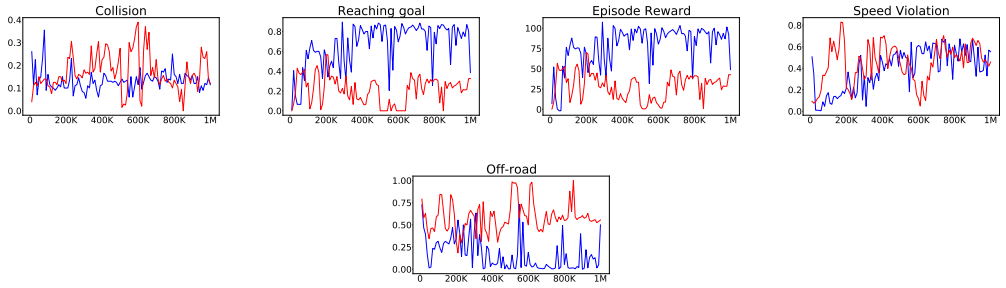


(b) Evaluation

Figure 8: Training and evaluation for task-1 easy (generalization performance to cross intersection) for PPO (red) and SAC (blue)

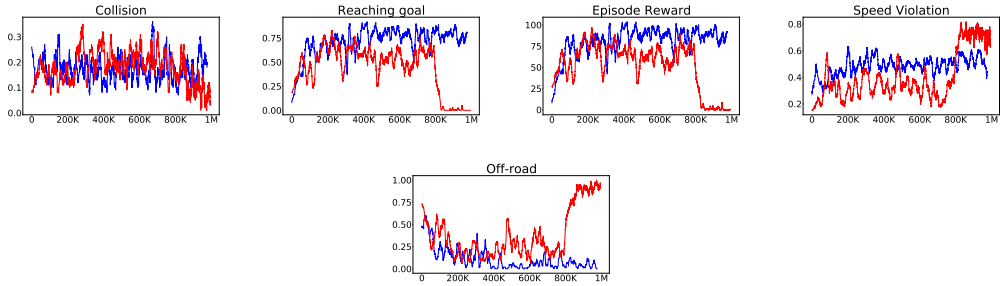


(a) Training

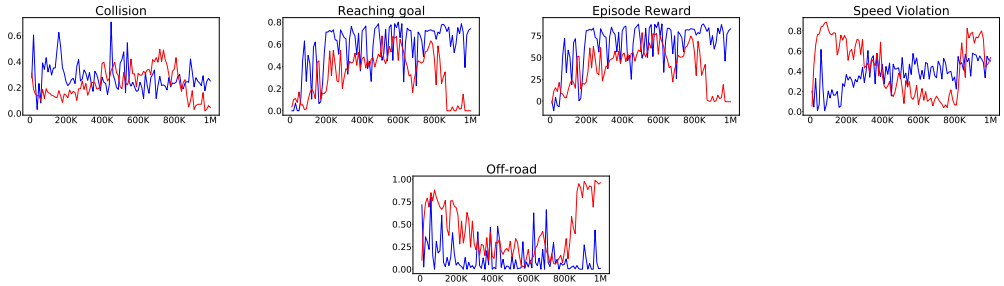


(b) Evaluation

Figure 9: Training and evaluation for task-1 hard (generalization performance to cross intersection) for PPO (red) and SAC (blue)

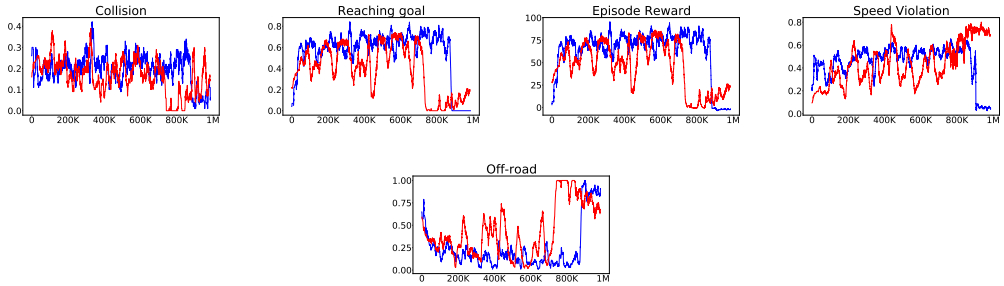


(a) Training

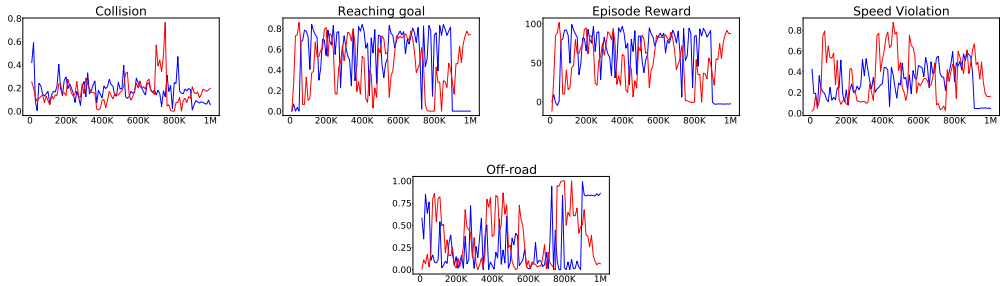


(b) Evaluation

Figure 10: Training and evaluation for task-2 low-to-high (generalization performance to heavier traffic) for PPO (red) and SAC (blue)

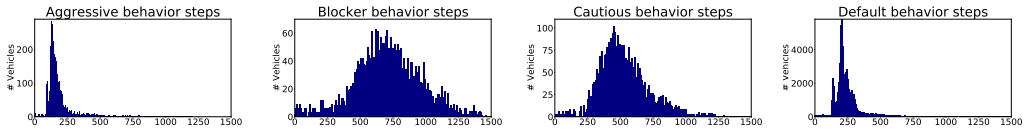


(a) Training

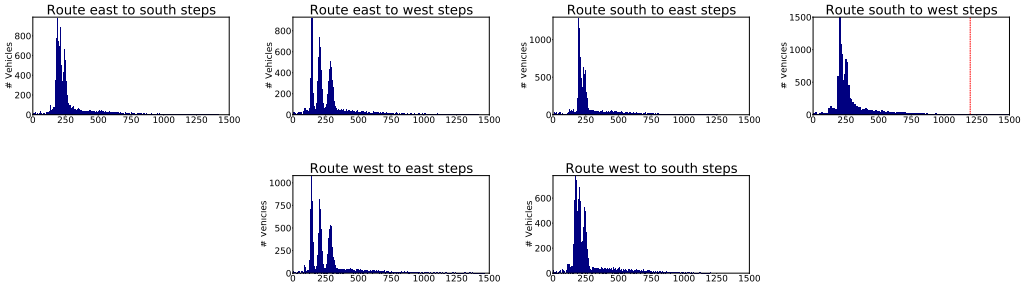


(b) Evaluation

Figure 11: Training and evaluation for task-2 high-to-low (generalization performance to heavier traffic) for PPO (red) and SAC (blue)

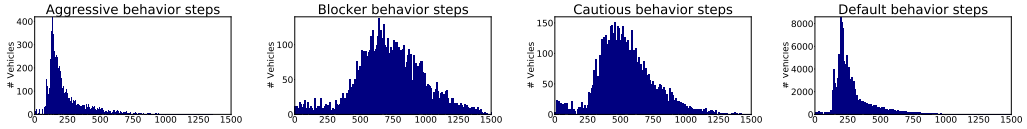


(a) By behavior

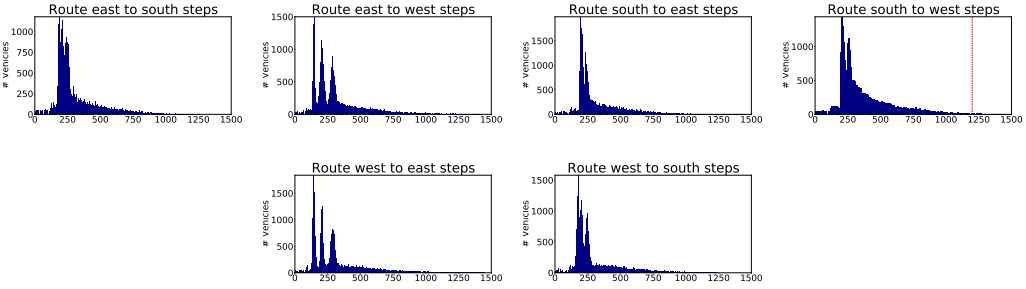


(b) By route

Figure 12: Steps needed to finish simulation in low traffics

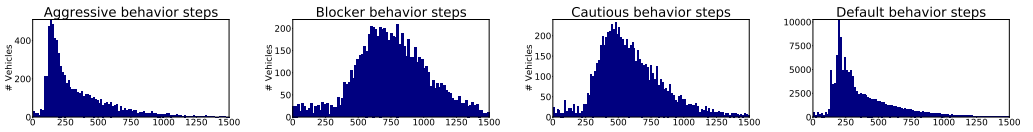


(a) By behavior

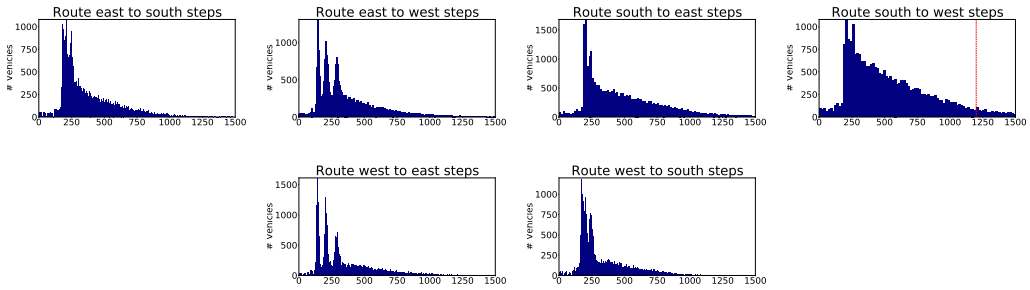


(b) By route

Figure 13: Steps needed to finish simulation in medium traffics



(a) By behavior



(b) By route

Figure 14: Steps needed to finish simulation in high traffics