Offline Reinforcement Learning for Autonomous Driving

EECS 545 Term Project, Winter 2024

Shitanshu Bhushan
University of Michigan
sbhushan@umich.edu

Kuang-Yang Cheng
University of Michigan
ykycheng@umich.edu

Yuxing Liu
University of Michigan
dylanliu@umich.edu

Aakash Puntambekar
University of Michigan
apuntamb@umich.edu

Abstract

Learning human driving behaviors and patterns is crucial for real-world autonomous driving applications. The two main challenges are (1) predicting the future trajectory of the self-driving car and (2) simulating the driver's behavior. To learn these complex tasks, we need to have accurate and representative data. In this project, one key insight is that we utilize a dataset with highly interactive driving scenarios. This leads to our primary objective of learning from human behavior. We investigate the use of offline model-based reinforcement learning (RL) as a different method for making trajectory decisions. First, we introduced offline RL methods and construct an offline RL dataset for the roundabout trajectory decisions on the real world driving dataset INTERACTION. Then, we design a reward function satisfying demands of the roundabout scenario. Further, we deploy conservative Q-learning algorithm and analyze its performance under test and augmented data. Finally, the discussions and conclusions are provided to analyzed the future goal and challenge for offline RL in autonomous driving.

1 Introduction

Autonomous driving has recently received a high degree of public attention, not all positive. Recent incidents involving General Motors' Cruise division and Tesla's autopilot system have been highlighted in the media and have shaken public confidence in autonomous technologies [(Domonoske, 2024)(Krisher, 2023)]. However, we believe that market demand for autonomous driving could potentially be very high if the technology is able to win over public opinion. Although passenger cars receive the bulk of attention, other sectors could also benefit from the technology. For example, over-the-road trucks that are autonomous would be able to meet tighter delivery deadlines as there is no human driver needing periodic breaks. While non-road and non-land vehicles stand to benefit from this technology as well (indeed the concept of autopilot has existed in aviation for decades), on-road vehicles pose challenges in route planning and obstacle avoidance that are unique. In order to build public confidence in on-road autonomous technologies, many avenues must be explored. We propose exploring offline reinforcement learning (RL) from human feedback as one of these avenues.

2 Proposed method

2.1 High Level Approach

At a high level, our approach will be to train an agent using a RL algorithm using some fraction of this data set. As per the paradigm of RL, we will assign a reward based on the similarity of the actions taken by our autonomous agent to the actions a human would take given the same scene. The remainder of the data will then be used to test our agent. Based on the INTERACTION dataset [(Zhan et al., 2019)], which contains data recorded from drones and traffic cameras, we can utilize the pre-recorded complex driving behaviors, such as negotiations and aggressive decisions, to perform offline RL for cloning behavior or predicting motion. Initially, we will focus on training a solitary conservative Q-learning (CQL) agent. Subsequently, we intend to expand our scope by training multiple CQL agents. During the inference phase, we'll aggregate the q-values of these agents for a specific state-action pair by computing their mean. We can then compare the results of both to see if our ensemble performs better. We will also then upload the results of our agent on the INTERACTION-Dataset-based PREdicTion (INTERPRET) Challenge to see how well our agent performs against existing methods.

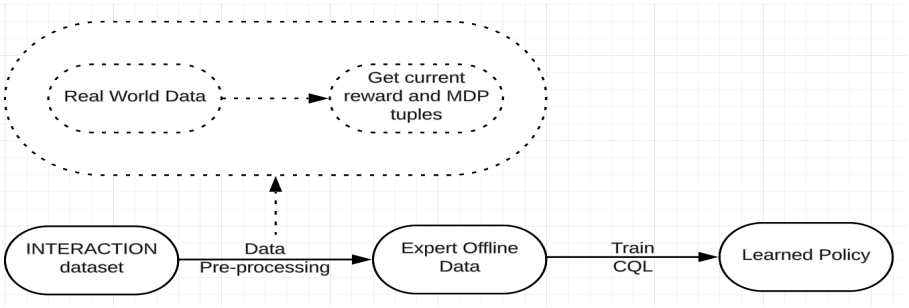

Figure 1: Pipeline of Method

2.2 Problem Formulation

RL is a Markov Decision Process (MDP), specified by the tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, T, r, \mu_0, \gamma)$. $\mathcal{S}$ defines the state space, $\mathcal{A}$ denotes the action space. $T(\mathbf{s}_{t+1} \mid \mathbf{s}_t, \mathbf{a}_t)$ is a conditional probability that describes the dynamics of the system and $r(\mathbf{s}, \mathbf{a})$ represents the reward function. $\mu_0(s)$ denotes the initial state distribution. $\gamma \in (0, 1)$ denotes the discount factor. We wish to learn a policy that maximizes return, denoted by

$$J(\pi) := \mathbb{E}_{(\mathbf{s}_t, \mathbf{a}_t) \sim \pi} \left[\sum_{t=0}^{\infty} \gamma^t r(\mathbf{s}_t, \mathbf{a}_t) \right]$$

We must find this policy while only having access to an offline dataset of transitions collected using a behavior policy $\pi_\beta, \mathcal{D} = \left\{ \left(\mathbf{s}_t^i, \mathbf{a}_t^i, \mathbf{s}_{t+1}^i, r_t^i \right) \right\}$.

2.3 Reward Function for Trajectory Decisions

As described previously in this report, our chosen data set, INTERACTION [(Zhan et al., 2019)], is comprised of tabulated vehicle position data as a function of time. These data have been collected during times at which the vehicle is being driven by a human rather than by an autonomous agent. For our purposes, we seek to define a reward function based on the deviation of our RL agent's trajectory from a human's decisions given the same scenario. This reward function will be used in our Q-learning implementation and will contain several tunable hyperparameters.

We begin by describing the intuition behind our desired reward function. Consider a table of data that shows the trajectory of a vehicle as a function of time. Note that in the INTERACTION dataset [(Zhan et al., 2019)], the data is comprised of not just the vehicle we wish to train as an autonomous agent, but rather also the objects around the this subject vehicle, therefore we have available to us a rich set of data containing high-resolution (in time) recorded positions. Our reward function should be capable of rewarding the agent's capability to match both a human's position and velocity decisions, while penalizing heavily any collisions. Mathematically, we express this as follows:

$$R = R_p + R_v + R_C \tag{1}$$

where R is the total reward, R_p is the reward for the position match, R_v is the result for the velocity match and R_C is a penalty (that is, negative reward) for a collision with another object.

To expand each of the terms in Equation ((1)) further, we first need to define the notion of future trajectory length. Recall that the goal here is for an autonomous agent to plan a trajectory given a certain scenario, which is an instantaneous snapshot of the positions of the agent and its surrounding objects. Thus, a tunable hyperparameter is the number of timesteps S that will be planned.

We now define each of the terms individually as follows:

$$R_p = r_p \frac{1}{||p - \hat{p}||^2} \tag{2}$$

$$R_v = r_v \frac{1}{||v - \hat{v}||^2} \tag{3}$$

$$R_C = r_C \sum_{s=1}^{S} C \tag{4}$$

In equations ((2)) - ((4)), p is a matrix in $\mathbb{R}^{Sx2}$ consisting of x and y position coordinates for the next S timesteps, v is an analagous vector for velocity and C is a count over the next S for whether the route planned by our agent coincides with any other tracked objects in the scene at the same time (indicating a collision). $r_p > 0$, $r_v > 0$ and $r_C < 0$ are tunable hyperparameters that will affect the training of the RL agent.

Substituting the details above into Equation ((1)), we obtain:

$$R = r_p \frac{1}{||p - \hat{p}||^2} + r_v \frac{1}{||v - \hat{v}||^2} + r_C \sum_{s=1}^{S} C \tag{5}$$

We intend to use this approach for our reward function and report the deviation of our agent's trajectory plan from that undertaken in the next S timesteps by a human. Depending on our results, the R_C term may not be necessary, as data about collisions is implicitly contained in the human decisions in the data set. However, if necessary, it can be defined as an indicator function that is 1 when our agent is within a certain distance from another object in the scene, and 0 if not. Or, mathematically expressed:

$$C = \mathbf{1}[||p_o - \hat{p}||^2 \leq d_t] \tag{6}$$

where p_o is the position of any other object in the scene and d_t is some minimum distance from it in 2-dimensional coordinates, another tunable hyperparameter.

2.4 Conservative Q-learning

We have Q function, which is the expected value of the total reward we will get if we start from state s_t and then take action a_t,

$$Q^\theta(\mathbf{s}_t, \mathbf{a}_t) = E\left[\sum_{t'=t}^{T} \gamma^{t'-t} r(\mathbf{s}_t, \mathbf{a}_t)\right] \tag{7}$$

Q functions follow Bellman equations,

$$Q^\theta(\mathbf{s}_t, \mathbf{a}_t) = r(\mathbf{s}_t, \mathbf{a}_t) + \gamma E\left[Q^\theta(\mathbf{s}_{t+1}, \mathbf{a}_{t+1})\right] \tag{8}$$

From the above, an optimal policy can be obtained by

$$\pi(a_t, s_t) = argmax_{a_t} Q^\theta(s_t, a_t) \tag{9}$$

This is the basis of Q-learning and the objective for training the Q function would be:

$$\min_{Q^\theta} E_{(\mathbf{s},\mathbf{a})\sim\pi_\beta(\mathbf{s},\mathbf{a})}\left[\left(Q^\theta(\mathbf{s}_t, \mathbf{a}_t) - \left(r(\mathbf{s}_t, \mathbf{a}_t) + \gamma E\left[Q^\theta(\mathbf{s}_{t+1}, \mathbf{a}_{t+1})\right]\right)\right)^2\right] \tag{10}$$

In an offline learning setting, Q-learning faces an issue called distribution shift: while our function approximator is trained under one distribution, it will be evaluated on a different distribution[(Levine et al., 2020)]. In an online setting, this would be less of an issue as we could just learn from this out of distribution state but in an offline only setting, we cannot do the same and must entirely rely on static dataset.

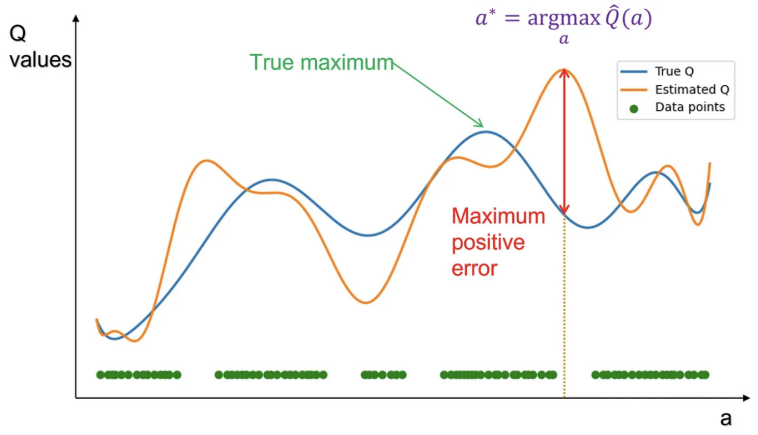

Figure 2: Distribution Shift in Q-learning[(Kapoutsis, 2022)]

From the figure we can see that, as in Q-learning we select π_{new} to maximise the expected value of Q, we end up with an erroneous over-estimation for these out of distribution data.

This gives us the motivation for CQL[(Kumar et al., 2020)], keep our standard Bellman error minimization term and add another term that explicitly seeks to find actions with high Q values and then minimize their value. We also add another term to make sure we don't end up being too conservative. This gives us the following objective:

$$\begin{aligned}\hat{Q}^{\pi} = \arg\min_{Q}\max_{\mu}\alpha E_{\mathbf{s}\sim D,\mathbf{a}\sim\mu(\mathbf{a}|\mathbf{s})}[Q(\mathbf{s},\mathbf{a})] - \alpha E_{(\mathbf{s},\mathbf{a})\sim D}[Q(\mathbf{s},\mathbf{a})] \\ + E_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim D}\left[\left(Q(\mathbf{s},\mathbf{a}) - \left(r(\mathbf{s},\mathbf{a}) + E_{\pi}\left[Q\left(\mathbf{s}',\mathbf{a}'\right)\right]\right)\right)^2\right]\end{aligned} \tag{11}$$

The third term here is our standard Bellman error. The first term always pushes down Q values to prevent overestimation. The second term here is used to counterbalance by maximizing the Q-values of the actions in the dataset.

3 Related work

3.1 Offline Reinforcement Learning

There is an increasing interest in the development and enhancement of RL algorithms for offline scenarios. One set of techniques for Offline RL is dedicated to enhancing the stability of off-policy Q-learning by mitigating the tendency to overestimate the Q-function in areas beyond the range of the dataset [(Kostrikov et al., 2021),(Kumar et al., 2020)].

Another category of approaches is centered around doing offline RL in a model-based environment, employing a learning process akin to the one described in [(Janner et al., 2019)]. At a macroscopic level, these methods initially optimize a model $f\left(s_t, a_t\right)$ to accurately anticipate the transition dynamics of the environment. Next, they train a policy by conducting autoregressive rollouts of the dynamics model using actions sampled from the policy that is being optimized. However, like the Q-learning algorithms, the model frequently fails to accurately predict outcomes in areas of the state space that are not inside the data distribution. Hence, recent studies, as those mentioned in the range [(Kidambi et al., 2020),(Yu et al., 2020)], have developed dynamics models that take into account uncertainty. These models incorporate a penalty in the reward function or termination function based on the level of uncertainty in the state estimation during policy optimization.

Our approach makes use of the INTERACTION data set, for which there is limited prior art. Our works differs from the prior works on this data set by investigating offline RL in a roundabout scenario, while also making use of ensemble Q-learning.

3.2 Offline RL dataset for autonomous driving

In recent years, many corporations and research entities have started to publicly release their autonomous driving datasets [(Chang et al., 2019) (Ettinger et al., 2021)]. However, identifying the most valuable datasets can be difficult as it typically involves extensive online research. Fields like motion prediction, imitation learning, representation learning, behavior modeling, and algorithm testing rely on the availability of high-quality motion datasets. Ideal datasets should cover interactive driving situations across diverse driving cultures and behaviors. Instead of using popular autonomous driving datasets like Argoverse [(Chang et al., 2019)] and Waymo Open Dataset [(Ettinger et al., 2021)], we use the real-world driving trajectory dataset INTERACTION [(Zhan et al., 2019)].

INTERACTION dataset [(Zhan et al., 2019)] is a dataset we chose to use in this project which features a diverse range of driving scenarios captured by drones, showcasing high levels of interaction. It provides different interactive scenarios including: roundaboud with 10479 vehicle data, unsignalized intersection with 14876 vehicle data, merging and lane change with 10933 vehicle data, and signalized intersection with 3775 vehicle data.

3.3 Similarities and Difference to Existing Work

In the landscape of autonomous vehicle (AV) research, our work situates itself amidst the growing exploration of offline reinforcement learning (RL) for motion planning. Drawing on the foundation of using a dynamics model trained on historical driving data, our research aligns with contemporary approaches in offline model-based RL. This methodology, leveraging the INTERACTION dataset, aims to refine motion planning policies by utilizing rich, real-world driving scenarios without necessitating further online data collection that poses safety risks.

Despite these commonalities with existing research, our project carves out its distinct niche through several key innovations and focus areas:

Addressing Distributional Shift and Model Generalization: Our work places a significant emphasis on overcoming the challenges inherent in offline RL, particularly the issues of distributional shift and model generalization. By integrating domain-specific insights into the dynamics model, we endeavor to steer the

policy optimization toward solutions that are not only safe but also robustly generalizable, reducing the peril of extrapolating from the training data.

Innovative Multi-Agent Training Environment: We extend the conventional framework of offline model-based RL with a novel multi-agent training environment. This environment simulates intricate, interactive driving scenarios by incorporating multiple agents, each following potentially distinct policies. This advancement not only augments the realism and relevance of the trained policies but also facilitates the exploration of both cooperative and competitive dynamics within AV motion planning, a dimension less explored in existing literature.

Utilization of the INTERACTION Dataset: Contrary to common practice where popular datasets like Argoverse or the Waymo Open Dataset are employed, our research utilizes the INTERACTION dataset. This choice is strategic, aiming to harness the dataset's comprehensive coverage of diverse, interactive driving situations across various driving cultures and behaviors, specifically focusing on complex scenarios such as roundabouts, which have seen limited exploration in existing CQL-based studies.

Unique Approach to Safety and Interaction: Aligning with the feedback from our team, our approach innovatively applies ensemble CQL, differing from existing practices by focusing on roundabout navigation. Furthermore, our custom reward function, which emphasizes maintaining a safe distance over direct collision avoidance, presents a nuanced method of enhancing safety, reflective of the proactive measures integral to real-world driving.

In summary, while our project is grounded in the principles of offline CQL RL, it diverges through a concerted effort to tackle pivotal issues like safety, interaction, and the complexity of real-world driving scenarios. Our approach, characterized by methodological and domain-specific advancements, contributes a novel perspective to the ongoing discourse on AV motion planning, promising to address some of the field's most pressing challenges.

4 Experimental results

4.1 Offline training dataset: data preprocessing

To learn real human driving behavior by applying offline RL, we have chosen to use the real-world driving dataset INTERACTION [(Zhan et al., 2019)] which has real driving trajectories with motion information, vehicle shape and time from complex urban scenarios. It is widely used for trajectory prediction research and applications. In this project, we considier the validation of decision and planning in the DR_USA_Roundabout_FT mapas show in Fig.3. We will create an offline training dataset for MDP from this dataset.

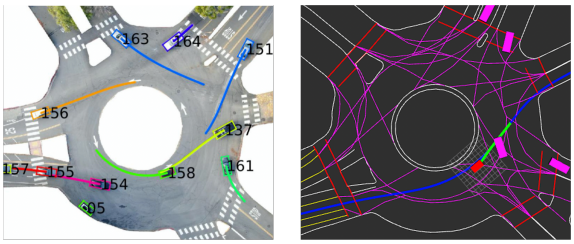

Figure 3: Example of the detection and tracking results in highly interactive roundabout driving scenario in the INTERACTION dataset. [(Zhan et al., 2019)]

Fisrt, we represent the current state of the ego vehicle using a 36-dimensional vector, denoted as $s = [s_{vehicle}, s_{ego}, s_1, s_2, s_3]$. The state vector s is segmented into three primary setctions, as shown in TABLE 1. $s_{vehicle}$ indicates the presence of the ego vehicle and three surrounding interactive vehicles, streamlining the computation process. s_{ego} encapsulates features of the ego vehicle, such as its dimensions, current and previous speed values, present coordinates, and projected coordinates for the next moment, assuming the ego vehicle continues on its current path and speed. The features of the three interactive vehicles represented by $s_1, ..., s_3$ also include the same information as s_{ego}. The subsequent state, s', is determined using the same methodology.

The following rules are the filter we use to determine interactive vehicles:

- A vehicle within a radius of 20 meters from the ego vehicle.
- A vehicle positioned no more than 10 meters to the rear of the ego vehicle.
- The nearest three vehicles are designated as interactive vehicles.

For the Action, we consider the ego vehicle's speed in the next time step as its action a. Because the trajectories form INTERACTION [(Zhan et al., 2019)] are human driving's data. Next, we will build collision data for the offline dataset we created to augment the data.

Table 1: State vector

Component	Description
$s_{vehicle}$	ego vehicle and interactive vehicles
s_{ego}	ego vehicle's features
$s_k(k = 1, 2, 3)$	interactive vehicle's features

4.2 Current Challenges

We are in the process of identifying a suitable environment to train our agent. Initially, we explored a simulator based on INTERACTION [(Zhan et al., 2019)] introduced in SHAIL [(Jamgochian et al., 2022)], but encountered challenges during the compilation process. Additionally, the simulator's reward function differs significantly from the one we intend to use.

Given these circumstances, our current objective is twofold. Firstly, we aim to find an existing environment that we can modify to align with our desired state and action spaces, as well as our custom reward function. If such an environment proves elusive, our contingency plan is to develop our own environment using the OpenAI Gym API.

By following this approach, we strive to ensure that our agent's training environment accurately reflects the requirements and specifications of our RL problem, ultimately enhancing the agent's performance and effectiveness.

5 Future Milestones

Our future milestones and schedule is given concisely in the Gantt chart below.

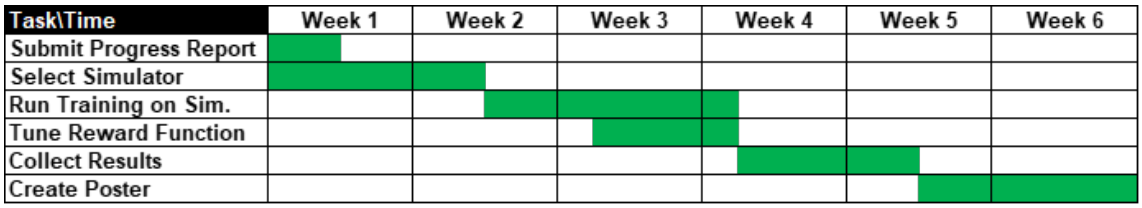

Figure 4: Schedule of milestones - note that week 1 corresponds to the week of March 11, 2024.

6 Conclusion

In this project, we propose an offline RL method for autonomous driving using real-world interactive driving data. We apply CQL and design a reward function to model human driving behavior. We have finished data preprocessing and are now working on data augmentation, particularly with collision data. Currently, we are developing a simulated environment to train the agent. However, we've encountered some challenges that we're actively resolving.

7 Author Contribution

Aakash - Developed reward function and its mathematical formulation and did background research on autonomous vehicle industry current events.
Shitanshu - Did research on approach to take and problem formulation with focus on Q-learning and CQL. Worked with Yuxing to research and set up training environment.
Kuang-Yang - Did Research on datasets, analyzed the data on INTERACTION dataset, designed state vector and processed the dataset for offline RL use.
Yuxing - Reviewed and summarized related works by other people so far on offline reinforcement learning. Worked with Shitanshu to research and set up training environment.

All co-authors were involved in writing this report. All co-authors equally contributed to this project.

References

Ming-Fang Chang, John Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, and James Hays. 2019. Argoverse: 3d tracking and forecasting with rich maps.

Camila Domonoske. 2024. "gm's driverless car company cruise is under investigation by several agencies" national public radio: Wbex chicago, january 25, 2024. https://www.npr.org/2024/01/25/1226953350/gm-driverless-cruise-investigation-federal-agencies.

Scott Ettinger, Shuyang Cheng, Benjamin Caine, Chenxi Liu, Hang Zhao, Sabeek Pradhan, Yuning Chai, Ben Sapp, Charles Qi, Yin Zhou, Zoey Yang, Aurelien Chouard, Pei Sun, Jiquan Ngiam, Vijay Vasudevan, Alexander McCauley, Jonathon Shlens, and Dragomir Anguelov. 2021. Large scale interactive motion forecasting for autonomous driving : The waymo open motion dataset.

Arec Jamgochian, Etienne Buehrle, Johannes Fischer, and Mykel J. Kochenderfer. 2022. Shail: Safety-aware hierarchical adversarial imitation learning for autonomous driving in urban environments.

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. 2019. When to trust your model: Model-based policy optimization.

Athanasios Kapoutsis. 2022. The monster of distribution shift in offline rl and how to pacify it (https://medium.com/@athanasios.kapoutsis/the-monster-of-distribution-shift-in-offline-rl-and-how-to-pacify-it-4ea9a5db043).

Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. 2020. Model-based offline reinforcement learning.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. 2021. Offline reinforcement learning with implicit q-learning.

Tom Krisher. 2023. Tesla's recall of 2 million vehicles to fix its autopilot system uses technology that may not work.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. 2020. Conservative q-learning for offline reinforcement learning.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. 2020. Offline reinforcement learning: Tutorial, review, and perspectives on open problems.

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. 2020. Mopo: Model-based offline policy optimization.

Wei Zhan, Liting Sun, Di Wang, Haojie Shi, Aubrey Clausse, Maximilian Naumann, Julius Kummerle, Hendrik Konigshof, Christoph Stiller, Arnaud de La Fortelle, and Masayoshi Tomizuka. 2019. Interaction dataset: An international, adversarial and cooperative motion dataset in interactive driving scenarios with semantic maps.