ADVERSARIAL DRIVING POLICY LEARNING BY MISUNDERSTANDING THE TRAFFIC FLOW

Anonymous authors
Paper under double-blind review

ABSTRACT

Acquiring driving policies that can transfer to unseen environments is essential for driving in dense traffic flows. Adversarial training is a promising path to improve robustness under disturbances. Most prior works leverage few agents to induce driving policy's failures. However, we argue that directly implementing this training framework into dense traffic flow degrades transferability in unseen environments. In this paper, we propose a novel robust policy training framework that is capable of applying adversarial training based on a coordinated traffic flow. We start by building up a coordinated traffic flow where agents are allowed to communicate Social Value Orientations (SVOs). Adversary emerges when the traffic flow misunderstands the SVO of driving agent. We utilize this property to formulate a minimax optimization problem where the driving policy maximizes its own reward and a spurious adversarial policy minimizes it. Experiments demonstrate that our adversarial training framework significantly improves zero-shot transfer performance of the driving policy in dense traffic flows compared to existing algorithms.

1 INTRODUCTION

Policy learning in dense traffic flows is a progressively active area for both academia and industry community in autonomous driving (Dosovitskiy et al., 2017; Suo et al., 2021). Since training driving policy in real world is costly, researchers aim to build dense traffic flows in simulation as an alternative (Cai et al., 2020; Pal et al., 2020; Wu et al., 2021). Peng et al. (2021) develops a traffic flow that exhibits altruistic behaviors and training driving policy in such coordinated flow also performs well. However, the internal dynamics of different traffic flows are varied, making it difficult to train driving policy in one flow and transfer it into unseen traffic patterns. Hence, it is indispensable to develop robust driving policies that can transfer among different traffic flows.

An appealing technical route to improve the robustness of driving policy is adversarial attack (Pinto et al., 2017), which models differences between training and evaluating environments as extra disturbances towards driving policy (Wachi, 2019; Chen et al., 2021; Liu et al., 2021; Huang et al., 2022). To exert disturbances on driving policy, these works leverage few agents to deliberately induce driving policy's failures. Although working well in sparse traffic situations, this pipeline cannot extend to dense traffic flows. On the one hand, increasing the number of attacking agents makes adversarial attacks easier, yet it is harder for the driving policy to resist such strong disturbances, which severely harms policy learning. On the other hand, attacking agents mainly concentrate on producing adversarial behaviors towards driving policy, while overlooking the modeling of altruistic behaviors among them. Therefore, the key is to construct a coordinated traffic flow which still generates adversarial behaviors.

We develop a coordinated traffic flow with communication and propose a misunderstanding-based adversarial training pipeline based on this flow. Specifically, for building a coordinated traffic flow, we introduce the concept of Social Value Orientation (SVO) (Liebrand, 1984) in social psychology which balances egoistic and altruistic behaviors for each agent. SVO can be regarded as the hidden information of one agent, which typically cannot be accessed by other agents. However, in this paper, we allow agents in our traffic flow to communicate genuine SVOs with each other. Since the traffic flow is served as a testbed for training and evaluating driving policies, the coordination mechanism within the traffic flow is invisible to driving policies.

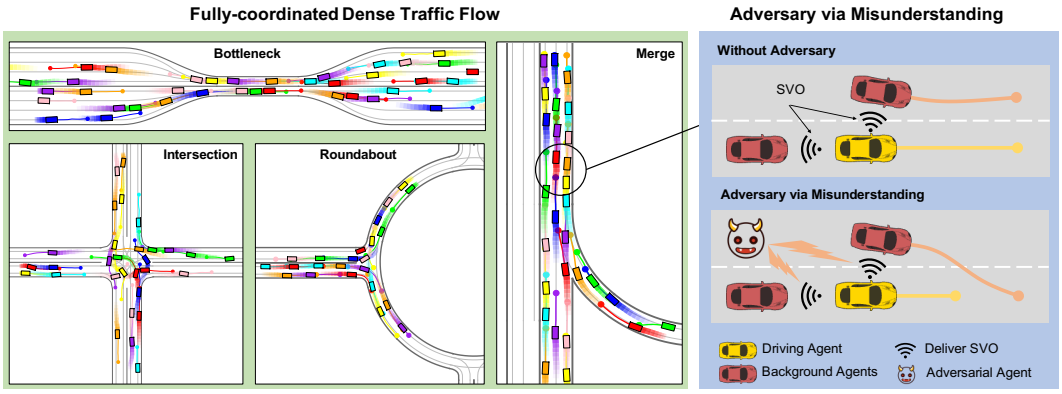

Figure 1: **Overview of our training framework.** *Left*: We build up a coordinated traffic flow in which agents communicate SVOs to coordinate with each other. *Right*: By disturbing the SVO of driving agent, our traffic flow exhibits adversarial behaviors towards the driving policy.

In other words, when placing a driving policy to interact with the traffic flow, the traffic flow requires receiving driving policy's SVO while the driving policy is unaware of traffic flows' SVOs. This property offers a neat approach to induce misunderstandings between driving policy and our traffic flow, making it adversarial towards driving policy. We reserve a spurious adversarial agent to disturb the SVO delivery from the driving agent to other agents and formulate a minimax optimization problem where the driving policy maximizes its own reward while the spurious adversarial policy minimizes it, as shown in Figure 1.

Contributions. We propose a novel adversarial training framework based on a coordinated traffic flow to obtain driving policies that can transfer across various traffic flows. We develop a coordinated traffic flow where agents exhibit egoistic, prosocial, and altruistic behaviors based on communicating SVOs with each other. Based on this traffic flow, we apply adversarial driving policy training by adversarially misunderstanding the traffic flow, which is disturbed to produce improper coordinated behaviors towards driving policy. We investigate characteristics of several traffic flows in four challenging scenarios and carry out comprehensive comparative studies to evaluate the robustness of driving policy. Results show that our traffic flow achieves the highest success rate and the proposed adversarial training pipeline significantly improves the transferability of driving policy compared to existing algorithms.

2 Related Work

Dense traffic flows. Prior works explore different methodologies to simulate dense traffic flows including rule design (Behrisch et al., 2011; Dosovitskiy et al., 2017; Cai et al., 2020; Zhou et al., 2021), Imitation Learning (IL) (Zhao et al., 2020; Gu et al., 2021; Wang et al., 2022), and Multi-Agent Reinforcement Learning (MARL) (Pal et al., 2020; Palanisamy, 2020; Wu et al., 2021). IL naturally leverages numerous human expert data but suffers from severe distribution shift and poor closed-loop performance even in simple scenarios. Most rule- and MARL-based algorithms aim to simulate individual behaviors of distinct agents, which overlooks complex interactions among agents. Similar to our work, Peng et al. (2021) also builds a coordinated traffic flow based on SVO. However, agents in their traffic flow have no access to other agents' SVOs, leading to conservative behaviors.

Adversarial attack. A common way to acquire robust policy is applying Robust Adversarial Reinforcement Learning (RARL) (Pinto et al., 2017; Pan et al., 2019; Vinitsky et al., 2020; Oikarinen et al., 2021). Researchers in autonomous driving also follow this pipeline (Wachi, 2019; Ding et al., 2020; Chen et al., 2021; Sharif & Marijan, 2021; Huang et al., 2022). Adversarial policies in Ding et al. (2020); Chen et al. (2021); Sharif & Marijan (2021); Huang et al. (2022) are optimized to collide with driving agent, while Wachi (2019); Huang et al. (2022) attempt to expel ego agent from drivable areas. Using attacking agents to interfere with driving policy deliberately, such pipeline

provides large adversarial disturbance for driving policy. However, excessively concerning rarely happened scenarios harms the robustness of driving policy in unseen environments since it fails to capture simpler yet non-trivial interactive patterns. In this work, we apply adversarial training framework on a coordinated traffic flow with communication to solve this problem.

3 Traffic Simulation Construction

3.1 Problem Setting

Partially Observable Stochastic Game (POSG). Traffic simulation systems are typically formulated as a POSG (Oliehoek & Amato, 2016). Formally, POSG is a tuple $G = \langle \mathcal{I}, \mathcal{S}, \mathcal{A}, P, \mathcal{R}, \rho_0, \mathcal{O}, n, \gamma, T \rangle$. n is the number of agents. $\mathcal{I}$ denotes the set $\{0, 1, \dots, n-1\}$. $\mathcal{S}$ is the state space. $\mathcal{A}$ is the joint action space of n agents and $\mathcal{A} = \times_{i \in \mathcal{I}} \mathcal{A}_i$. $P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ [1] is the state transition probability. $\mathcal{R} = \{R_0, R_1, \dots, R_{n-1}\}$ denotes the set of agent-specific reward functions and $R_i : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is bounded for all $i \in \mathcal{I}$. Note that each agent i receives distinct reward from its own reward function $r_i = R_i(s, a)$. $\rho_0 \in \Delta(\mathcal{S})$ is the initial state distribution. $\mathcal{O}$ is the joint observation space and $\mathcal{O} = \times_{i \in \mathcal{I}} \mathcal{O}_i$. $\gamma \in (0, 1]$ is the discount factor, and T is the time horizon. In POSG, each agent i maximizes its own expected cumulative reward via policy $\beta_i : \mathcal{O}_i \to \Delta(\mathcal{A}_i)$. When n becomes large, it is time-and-space consuming to optimize a set of policies $\mathcal{B} = \{\beta_0, \beta_1, \dots, \beta_{n-1}\}$. To solve this problem, we simply adopt parameter sharing strategy (Terry et al., 2020), i.e., $\beta_i = \beta$, with the help of neural network which has powerful representation ability.

Incorporating Social Value Orientation (SVO). From the perspective of social psychology, agents should consider surrounding agents' rewards to achieve coordinated driving. Following Schwarting et al. (2019); Buckman et al. (2019); Peng et al. (2021), we introduce the concept of SVO to model coordinated behaviors among agents and build up coordinated traffic simulation. By incorporating SVO, each agent i maximizes reward with the consideration of other surrounding agents:

$$R'_i = \cos(c_i) R_i + \sin(c_i) R_{S_i} \tag{1}$$

where $R_{S_i} = \sum_{j \in \mathcal{I}_{S_i}} R_j / |\mathcal{I}_{S_i}|$, $\mathcal{I}_{S_i}$ is the set of surrounding agents w.r.t. agent i. $c_i \in [0, \frac{\pi}{2}]$ is the SVO of agent i and kept fixed during each episode. Given equation 1, we formulate a SVO-embedded POSG $G' = \langle \mathcal{I}, \mathcal{S}, \mathcal{A}, P, \mathcal{R}', \mathcal{C}, \rho_0, \mathcal{O}, n, \gamma, T \rangle$. $\mathcal{R}' = \{R'_0, R'_1, \dots, R'_{n-1}\}$ denotes the set of SVO-embedded reward functions. $\mathcal{C} = \{c_0, c_1, \dots, c_{n-1}\}$ is set of all SVOs.

Problem formulation. As one can see, SVO determines the trade-off between egoistic and altruistic behaviors. For each agent, it is necessary to recognize SVOs of itself and other agents, which provides the ability to infer other agents' reward structures. Therefore, we design policy as $\beta : \mathcal{O}_i \times \mathcal{C}_i \times (\times_{j \in \mathcal{I}_{S_i}} \mathcal{C}_j) \to \Delta(\mathcal{A}_i)$. And we use a single policy β to solve n optimization objectives of SVO-embedded POSG:

$$\max_{\beta} \mathbb{E}_{s_t \sim P_\beta, a_t \sim \beta} [\sum_{t=0}^{T} \gamma^t R'_i(s_t, a_t, c_i)], \qquad i \in \mathcal{I} \quad c_i \in \mathcal{C}_i \tag{2}$$

3.2 Key Components

State space. Agents driving in dense traffic flow need to continually interact with surrounding agents. Besides, road structures also influence agents' decisions. Therefore, state space $\mathcal{S}$ needs to cover a collection of static and dynamic elements. The set of static elements E^s include lane centerlines, sidelines, agents' global paths, i.e., $E^s = \{centerline, sideline, path\}$. The set of dynamic elements E^d include current and historical poses and velocities (trajectories) of all agents, i.e., $E^d = \{trajectory_0, trajectory_1, \dots, trajectory_{n-1}\}$. We utilize vectorized representation based on Gao et al. (2020), which is computation- and memory-efficient. In our work, elements in E^s and E^d are sets of points containing corresponding features. Specifically, static element $e^s_i = \{v_0, v_1, \dots, v_j, \dots\}, i \in E^s$. $v_j = [p_j, c_i, i, j]$ where $p_j = (x, y, \theta)$ is the pose of point j in element i and c_i is the lane width of element i. For points in dynamic elements, $p_j = (x, y, \theta, v)$ and c_i denotes the SVO of agent i.

[1] $\Delta(\mathcal{X})$ denotes the set of probability distribution over set $\mathcal{X}$.

Observation space. In POSG, each agent could only receive perceptual information locally, we use L_2 norm to define locality, i.e., agent i could only receive points (x_e, y_e) that $\|(x_i, y_i) - (x_e, y_e)\|_2 \leq d$, in which (x_i, y_i) is the current location of agent i.

Design of $\mathcal{R}$. The goal of each agent in dense traffic flow is homogeneous, for instance, all agents want to successfully finish the task as fast as possible. Besides, since each agent receives $\mathcal{O}_i$, designing R_i upon $\mathcal{O}_i$ rather than $\mathcal{S}$ benefits policy training. In our work, we use self-motivated reward $R_i : \mathcal{O}_i \times \mathcal{A}_i \to \mathbb{R}$ and $R_i = R$ for all $i \in \mathcal{I}$. However, designing self-motivated reward function still remains an open problem. Designing fine-grained dense reward accelerates training procedure but relies too heavily on human knowledge, while training with coarse sparse reward requires much more data. To combine both benefits, we design a near-sparse reward function containing a dense reward for incentive driving fast and a sparse reward for penalizing catastrophic failures. Catastrophic failures include collision with other agents, deviation from drivable area, driving too far from global path, and crashing into wrong lane. Coordinated behaviors could be produced by incorporating SVO.

Policy training. We apply Independent Policy Learning (IPL) (Tan, 1993) to solve Equation 2. Although IPL is prone to generate egoistic suboptimal behaviors, we could alleviate this problem by incorporating SVO, which forces the algorithm to consider other agents' goals.

3.3 Policy Architecture

To better extract static and dynamic features and capture relations among them, we utilize a hierarchical feature extraction framework. We use DeepSet (Zaheer et al., 2017) to aggregate homogeneous information within dynamic and static elements, followed by Multi-Head Attention (MHA) (Vaswani et al., 2017) to further extract heterogeneous information among different elements.

Homogeneous feature aggregation. Consider the elements set $e \subset E, E = \left\{E^s, E^d\right\}$, and the function processing on the set needs to retain the adjacency between elements and permutation-invariant to the order of objects in the element. Based on theorem 2 in (Zaheer et al., 2017), the propagation function f is defined as:

$$f(e) = \rho\left(\sum_{v \in e} \phi(v)\right) \tag{3}$$

And we obtain the element level features $l_e = f(e)$, where e is the input elements set, the nodes $v \in e$ transformed into a representation $\phi(v)$. The sum of representations is processed using the ρ network defined by Multi-Layer Perception (MLP) network. In our implementation, DeepSet can extract polyline-level features while not introducing too many parameters.

Heterogeneous feature aggregation. The static element level features $l_e^s = [l_{e_0}^s, l_{e_1}^s, \ldots, l_{e_j}^s, \ldots]$ and the dynamic element features $l_e^d = [l_{e_0}^d, l_{e_1}^d, \ldots, l_{e_j}^d, \ldots]$ go through a MHA layer which takes into account their inter-relations to output the final action for the agents. Given arbitrary feature matrices w, z and their linear projections w_Q, w_K, w_V and z_Q, z_k, z_v, the $SelfAttn(w)$ and $CrossAttn(w, z)$ are defined as:

$$\begin{aligned} SelfAttn(w) &= \frac{Softmax\left(w_Q w_K^T\right)}{\sqrt{d_k}} w_V \\ CrossAttn(w, z) &= \frac{Softmax\left(w_Q z_K^T\right)}{\sqrt{d_k}} z_V \end{aligned} \tag{4}$$

where $\sqrt{d_k}$ is the dimension of the key vectors. We leverage the one-layer cross-attention network to model the interaction between dynamic and static segments. The dynamic elements features l_e^d and static elements features l_e^s are fused by the $SelfAttn$ and $CrossAttn$ operation:

$$\begin{aligned} l_o^d &= SelfAttn(l_e^d) + CrossAttn(l_e^d, l_e^s) \\ l_o^s &= SelfAttn(l_e^s) + CrossAttn(l_e^s, l_e^d) \end{aligned} \tag{5}$$

$l_o = \left\{l_o^d, l_o^s\right\}$ is the final output of MHA. We then decode the agents' action from l_o:

$$a = \varphi\left(l_o\right) \tag{6}$$

where $\varphi(\cdot)$ is the action decoder, and $a \in \mathcal{A}$. For simplicity, we use an MLP as the decoder function.

Algorithm 1: Misunderstanding-based Adversarial Reinforcement Learning

Input: SVO-embedded POMDP M containing traffic flow policy β
Output: Driving policy π, adversarial policy π_c
Initialize: Learnable parameters θ_π, θ_{π_c}
for $n = 1, 2, \ldots, N$ **do**
  /* Stage 1: Given π_c optimize π */
  for $n_1 = 1, 2, \ldots, N_1$ **do**
    Collect a set of transition tuples $\{(o, a, o', r)\}$ trajectories by rolling out π and π_c on M;
    Optimize parameters θ_π of π using any RL algorithms;
  /* Stage 2: Given π optimize π_c */
  for $n_2 = 1, 2, \ldots, N_2$ **do**
    Collect a set of transition tuples $\{(o, a, c_\beta, o', -r)\}$ trajectories by rolling out π and π_c on M;
    Optimize parameters θ_{π_c} of π_c using any RL algorithms;

4 Misunderstanding-based Adversarial Learning

4.1 Problem Setting

For single-agent driving task, we formulate SVO-embedded Partially Observable Markov Decision Process (POMDP) as $M = \langle \mathcal{S}, \mathcal{A}, P, \mathcal{R}, \mathcal{C}, \rho_0, \mathcal{O}, \gamma, T, \beta \rangle$. Note that β is the policy that controls the traffic simulation and affects state transition probability P. Following the tradition, we define policy $\pi : \mathcal{O} \to \Delta(\mathcal{A})$ to solve M. $c_\beta \in \mathcal{C} = [0, \frac{\pi}{2}]$ is the SVO of driving policy which is taken by β. The genuine SVO of driving agent c_π is always 0 since existing single-agent algorithms are fully self-interested. Adversary emerges when c_β and c_π differ:

$$\max_{\pi} \min_{c_\beta} \mathbb{E}_{s_t \sim P_{\beta, c_\beta}, a_t \sim \pi} [\sum_{t=0}^{T} \gamma^t R(s_t, a_t)] \tag{7}$$

In section 3.1 and equation 7, the SVOs are invariant during one episode for the reason of stabilizing training. However, in adversarial training, we aim to destabilize policy training. Therefore, we introduce a spurious policy $\pi_c : \mathcal{O} \to \Delta(\mathcal{C})$ to produce c_β which is allowed to change across time steps, changing equation 7 into:

$$\max_{\pi} \min_{\pi_c} \mathbb{E}_{c_{\beta,t} \sim \pi_c, s_t \sim P_{\beta, c_{\beta,t}}, a_t \sim \pi} [\sum_{t=0}^{T} \gamma^t R(s_t, a_t)] \tag{8}$$

Note that equation 8 relates to three policies including driving policy π, background policy β, and spurious adversarial policy π_c. Since driving policy maximizes its own reward, it is egoistic from the perspective of social psychology. Background policy controls the whole traffic flow to exhibit egoistic and altruistic behaviors. The spurious policy is the only one that aims to generate adversarial behaviors by minimizing driving policy's reward.

We highlight that agents in our traffic flow try to coordinate with each other, which is a fundamental difference compared to previous attacking agents. Instead of deliberately inducing failures of π, we keep β non-adversarial and leverage an extra π_c to disturb the SVO of π taken by β.

4.2 Adversarial Policy Training

Algorithm 1 outlines our training framework. Given background policy β, we alternatively optimize both driving policy π and adversarial policy π_c. The parameters θ_π of π and θ_{π_c} of π_c are randomly initialized before training. In each of N iterations, we first optimize θ_π and keep θ_{π_c} fixed, followed by optimizing θ_{π_c} and keeping θ_π fixed.

In Stage 1, we iterate N_1 times to optimize driving policy π. By sampling POMDP M, we collect a set transition tuples $\{(o, a, o', r)\}$, where $o, o' \in \mathcal{O}$, $a \in \mathcal{A}$, and $r = R(o, a)$. We then apply standard RL algorithms to optimize θ_π. In Stage 2, we iterate N_2 times to optimize adversary policy π_c. Similar to Stage 1, we sample M and get another set of transition tuples and apply

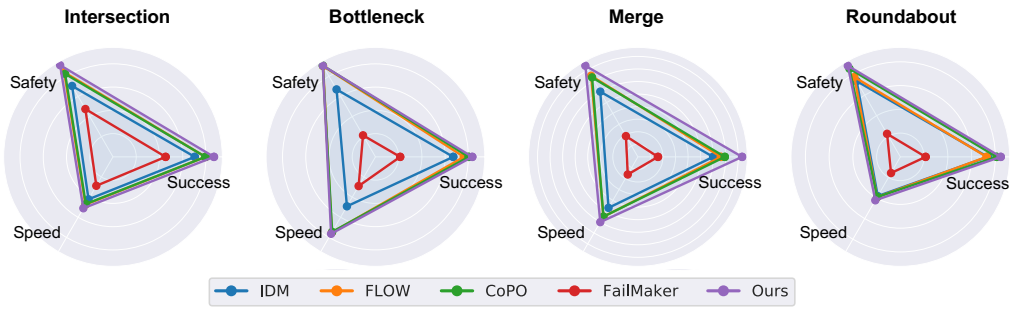

Figure 2: **Performance of different traffic flows.** The radar graphs demonstrate three essential features of different traffic flows. Safety is calculated by taking the complement of catastrophic failures.

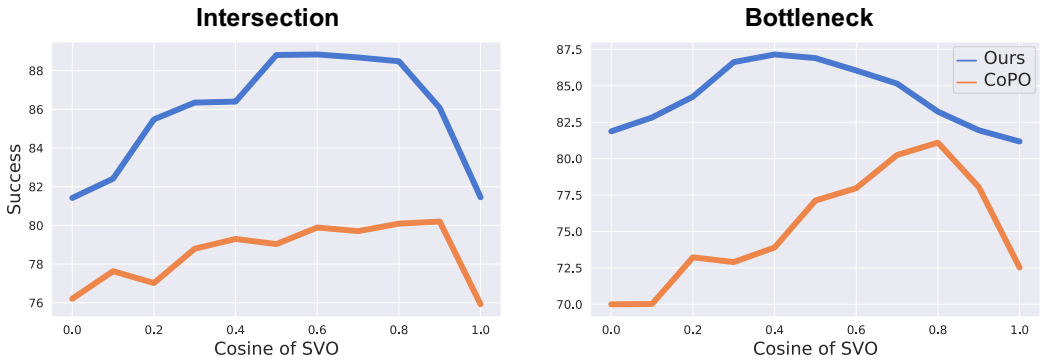

Figure 3: **Success rates of CoPO and Ours.** The figure reports the percentage of success rates in `intersection` and `bottleneck`. We assign a fixed SVO from 0 to $\frac{\pi}{2}$ at regular intervals. All agents in traffic flows are given the same SVO. We evaluate each SVO for 200 episodes.

RL algorithms. Differently, we reverse the sign of r since adversary policy aims to decrease the performance of driving policy. Note that the action of adversarial policy is c_β. a is the action of driving policy which is used to compute r. This alternating procedure is repeated for N iterations.

5 Results

In this section, we pursue to answer three seminal questions. (1) Can our proposed traffic simulation produce more coordinated behaviors? (2) Does the spurious adversarial policy degrade driving policy's performance? (3) Does our adversarial training framework improve driving policy's zero-shot transfer ability? Before discussing these questions, we first explain some preliminary details.

Settings. We evaluate our proposed method using our internal driving simulator which supports various maps and scenarios. Similar to Peng et al. (2021), we select several highly interactive scenarios including `intersection`, `bottleneck`, `merge`, and `roundabout`. During training, we randomly place 8 to 20 vehicles in each scenario at the beginning of each episode. After training, we randomly place 20 vehicles and evaluate all relevant methods.

Metrics. We consider three kinds of widely-accepted and general metrics. Firstly, success rate of the whole traffic simulation (Success). Secondly, catastrophic failure rates of the whole traffic simulation. Catastrophic failures include collision between agents (Collision), deviation from drivable area (Off Road), driving too far from global path (Off Route), and crashing into wrong lane (Wrong Lane). Third, driving efficiency is represented by average speed of the whole traffic simulation (Efficiency). As for single-agent training, these metrics are calculated from the perspective of ego agent.

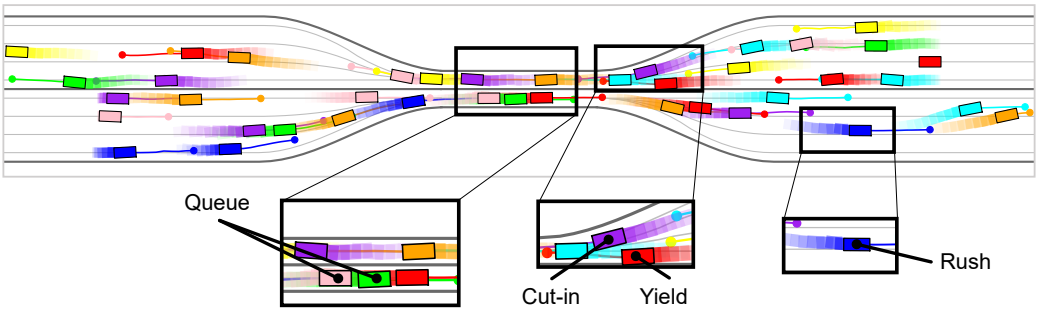

Figure 4: **Coordinated behaviors in `bottleneck`.** The figure highlights that our traffic flow with communication produces diverse coordinated behaviors such as queueing at the narrow crossing, rushing at open areas and yielding to avoid crashes.

Traffic flows. We implement four representative traffic flows to carry out comparative studies on different traffic flows and training pipelines. (1) Intelligent Driver Model (IDM) (Treiber & Kesting, 2013) is a rule-based controller which uses one single differential equation to model longitudinal movements for all agents. Each agent strictly follows its global path. (2) FLOW (Wu et al., 2021) is a MARL-based method where each agent aims to maximize its own reward. (3) CoPO (Peng et al., 2021) is a MARL-based method that also incorporates SVO. CoPO has two stages. First, similar to our traffic flow, CoPO applies IPL and SVO to train a background policy. Then, CoPO additionally trains a meta-controller to select all agents' SVOs so that the success rate of the whole population is maximized. Note that agents in CoPO have no access to other agents' SVOs. (4) FailMaker (Wachi, 2019) is a method to generate attacking behaviors. Attacking agents are rewarded if they successfully induce driving agent' catastrophic failures while avoiding personal failures except for collision.

5.1 Performance of Traffic Flows

We demonstrate the performance of different traffic flows. Figure 2 and Figure 3 show quantitative results of different traffic flows. As one can see, our proposed traffic flow achieves the highest success rates and average speeds across all scenarios. Compared with CoPO, agents in our traffic flow can recognize other agents' SVOs and produce coordinated behaviors, therefore achieving collaboration and high efficiency of the whole system. FailMaker achieves the lowest success rate and highest collision rate (lowest safety) due to its adversarial nature. Note that in `merge`, our traffic flow outperforms other methods by a large margin. The reason is that the initial poses of all agents in `merge` are much closer than these in other scenarios, which makes it harder for the agents to coordinate. Qualitative results are shown in Figure 4. See Appendix A.2 for more results.

5.2 Adversary on Coordinated Traffic Flow

Our robust policy learning framework applies adversarial training on a coordinated traffic flow. In this part, we demonstrate that our method successfully degrades driving policy's performance and has the ability to impede the driving agent.

Performance. We first train driving policy using vanilla RL in four non-adversarial traffic flows, including IDM, FLOW, CoPO, and Ours without Adversary (Ours wo Adv) and deploy these well-trained driving policies into our coordinated adversarial traffic flow. Results are shown in Table 1. Data in parentheses is the change of performances under the spurious adversarial policy. Success rates and speeds of all driving policies decrease. This reveals that our spurious adversarial policy could impede the efficiency of driving policy. Note that catastrophic failures of driving policy still increase under adversary in our traffic flow. The reason is that although highly coordinated, our traffic flow cannot eliminate catastrophic failures of the whole traffic system. And it is not unallowable for the traffic flow to incur driving policy's catastrophic failures due to the optimization objective in Equation 8.

Table 1: **Effect of adversary based on our coordinated traffic flow.** The table reports the percentage of different metrics in `intersection` and `bottleneck`. Results in parentheses indicate the performance change under adversary. Results marked in red indicate the performance degradation under adversary while results in blue indicate the performance increase. A "†" indicates our proposed traffic flow.

Methods	Intersection					
	Success (↑)	Collision (↓)	Off Road (↓)	Off Route (↓)	Wrong Lane (↓)	Speed (↑)
IDM	77.0 (-2.5)	10.0 (+4.5)	11.0 (-3.5)	1.0 (+1.5)	0.0 (+0.0)	47.1 (-1.2)
FLOW	84.0 (-1.5)	13.5 (+1.5)	1.0 (+1.0)	0.5 (-0.5)	0.5 (-0.5)	50.1 (-0.5)
CoPO	81.5 (-3.0)	14.5 (+4.0)	1.0 (+0.5)	2.0 (-1.0)	0.0 (+0.0)	48.8 (-1.3)
Ours wo Adv†	87.0 (-2.0)	7.0 (+2.0)	1.5 (+1.5)	1.5 (-1.0)	0.5 (+0.0)	51.9 (-1.1)
Methods	**Bottleneck**					
	Success (↑)	Collision (↓)	Off Road (↓)	Off Route (↓)	Wrong Lane (↓)	Speed (↑)
IDM	52.5 (-2.0)	26.0 (+1.0)	21.0 (+1.0)	0.5 (+0.0)	0.0 (+0.0)	58.3 (-1.3)
FLOW	75.5 (-2.0)	19.5 (+4.5)	4.5 (-2.5)	0.5 (+0.0)	0.0 (+0.0)	74.6 (-0.8)
CoPO	71.5 (-2.0)	13.5 (-2.5)	15.5 (+3.5)	1.0 (+0.0)	0.0 (+0.0)	70.9 (-1.0)
Ours wo Adv†	91.0 (-6.0)	4.0 (+3.0)	5.0 (+3.0)	0.0 (+0.0)	0.0 (+0.0)	81.3 (-3.0)

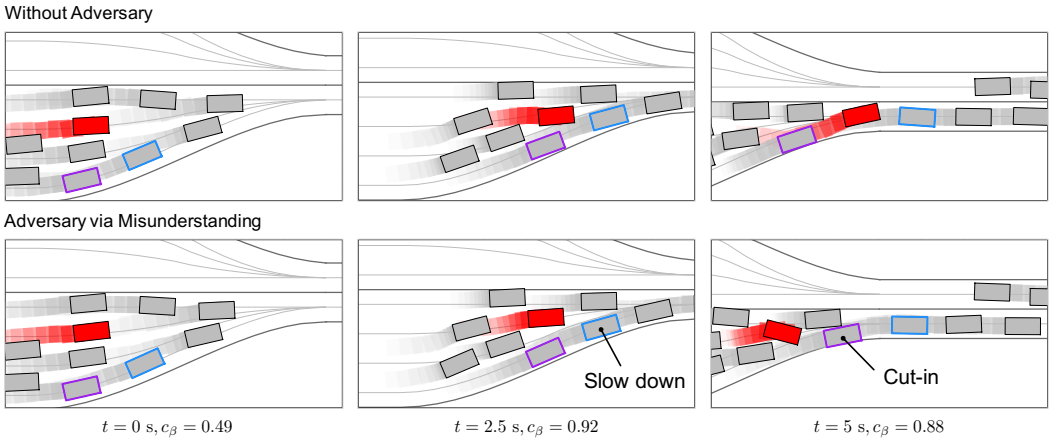

Figure 5: **Adversarial behaviors towards driving agent generated by coordinated traffic flow.** Driving policy controls the red vehicle. Vehicles with blue and purple boxes are two background agents that exhibit adversarial behaviors towards driving policy while maintaining coordination.

Adversarial behaviors on coordinated traffic flow. Figure 5 demonstrates some qualitative results on how our traffic flow impedes driving policy to finish its own task. Driving policy controls the red car. At $t = 0$s, the driving policy aims to pass through the bottleneck efficiently and keeps high speed. When driving policy approaches the bottleneck, where coordination frequently happens, the agent with blue box slows down ($t = 2.5$s) to regulate the speed of driving policy and pass through the bottleneck. After that, the agent with purple box cut in the agent ahead of driving agent ($t = 5.0$s). These agents impede the driving policy and degrade its efficiency and explicit coordinated behaviors among each other.

5.3 Zero-shot Transfer to Different Traffic Flows of Driving Policy

We use vanilla RL to train driving policies in IDM, FLOW, CoPO, and Ours respectively (denoted as VRL/IDM, VRL/FLOW, VRL/CoPO, and VRL/Ours). We reimplement the adversarial training framework in FailMaker in our setting (denoted as RARL/FailMaker). We denote our method as RARL/Ours.

To evaluate the robustness in unseen environments, we deploy all driving policies in all traffic flows and obtain Figure 6. Typically, evaluating driving policy in its training traffic flow achieves the

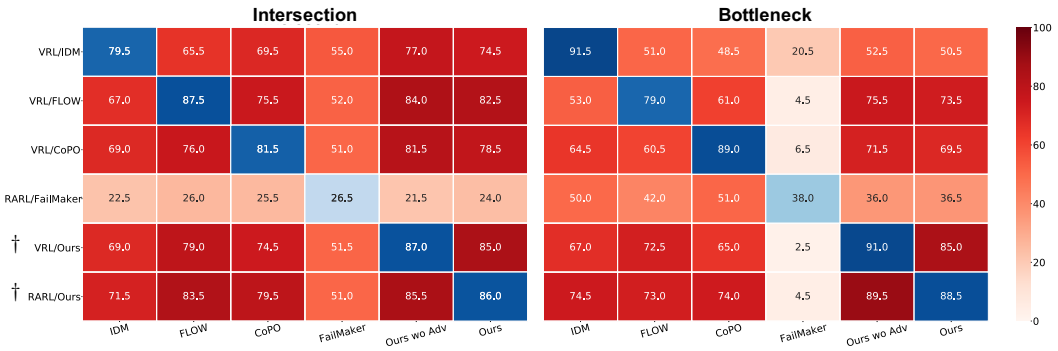

Figure 6: **Zero-shot transfer performance in `intersection` and `bottleneck`**. The heatmap reports the percentage of success rate for different methods in different traffic flows. Deeper color represents higher success rate. Primary diagonals indicates that training and evaluating environments are the same. A "†" indicates methods trained in our traffic flow.

highest success rate since the evaluate environment is Independent and Identically Distributed (IID) with training environment.

RARL/FailMaker shows the worst zero-shot transfer performance since background agents in FailMaker deliberately induce catastrophic failures of driving policy. This means that RARL/FailMaker has no way to see non-adversarial traffic behaviors. Therefore, although robustness under adversarial attack is improved, RARL/FailMaker is fragile to unseen traffic patterns. Based on this observation, VRL/IDM, VRL/FLOW, VRL/CoPO, and VRL/Ours demonstrate superior zero-shot transfer performances in non-adversarial traffic flows compared to RARL/FailMaker.

Comparing VRL/Ours and RARL/Ours, one can see that injecting adversaries in dense traffic flow significantly improves robustness in unseen non-adversarial environments. Note that all methods except RARL/FailMaker act poorly in FailMaker since it is extremely easy for background agents in FailMaker to attack driving policies, no matter how driving agents act shrewdly. See Appendix A.3 for more results.

6 Conclusion

In this paper, we propose a novel adversarial training framework based on a coordinated traffic flow with communication. Driving policies trained with this framework exhibit robust behaviors across various traffic flows. We report characteristics of several traffic flows in scenarios including `intersection`, `bottleneck`, `merge`, and `roundabout`. We carry out numerous comparative studies to evaluate the transferability of driving policy. Results show that our traffic flow achieves the highest success rate and adversarial learning on our traffic flow significantly improves driving policy's zero-shot transfer performance compared to existing algorithms.

References

Michael Behrisch, Laura Bieker, Jakob Erdmann, and Daniel Krajzewicz. Sumo–simulation of urban mobility: an overview. In *Proceedings of SIMUL 2011, The Third International Conference on Advances in System Simulation*. ThinkMind, 2011.

Noam Buckman, Alyssa Pierson, Wilko Schwarting, Sertac Karaman, and Daniela Rus. Sharing is caring: Socially-compliant autonomous intersection negotiation. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6136–6143. IEEE, 2019.

Panpan Cai, Yiyuan Lee, Yuanfu Luo, and David Hsu. Summit: A simulator for urban driving in massive mixed traffic. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4023–4029. IEEE, 2020.

Baiming Chen, Xiang Chen, Qiong Wu, and Liang Li. Adversarial evaluation of autonomous vehicles in lane-change scenarios. *IEEE Transactions on Intelligent Transportation Systems*, 2021.

Wenhao Ding, Baiming Chen, Minjun Xu, and Ding Zhao. Learning to collide: An adaptive safety-critical scenarios generating method. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2243–2250. IEEE, 2020.

Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. In *Conference on robot learning*, pp. 1–16. PMLR, 2017.

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*, pp. 11525–11533, 2020.

Junru Gu, Chen Sun, and Hang Zhao. Densetnt: End-to-end trajectory prediction from dense goal sets. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 15303–15312, 2021.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. PMLR, 2018.

Peide Huang, Mengdi Xu, Fei Fang, and Ding Zhao. Robust reinforcement learning as a stackelberg game via adaptively-regularized adversarial training. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pp. 3099–3106. International Joint Conferences on Artificial Intelligence Organization, 2022.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR (Poster)*, 2015.

Wim BG Liebrand. The effect of social motives, communication and group size on behaviour in an n-person multi-stage mixed-motive game. *European journal of social psychology*, 14(3):239–264, 1984.

Weilin Liu, Ye Mu, Chao Yu, Xuefei Ning, Zhong Cao, Yi Wu, Shuang Liang, Huazhong Yang, and Yu Wang. Multi-agent vulnerability discovery for autonomous driving with hazard arbitration reward. *arXiv preprint arXiv:2112.06185*, 2021.

Tuomas Oikarinen, Wang Zhang, Alexandre Megretski, Luca Daniel, and Tsui-Wei Weng. Robust deep reinforcement learning through adversarial loss. *Advances in Neural Information Processing Systems*, 34:26156–26167, 2021.

Frans A Oliehoek and Christopher Amato. *A concise introduction to decentralized POMDPs*. Springer, 2016.

Avik Pal, Jonah Philion, Yuan-Hong Liao, and Sanja Fidler. Emergent road rules in multi-agent driving environments. In *International Conference on Learning Representations*, 2020.

Praveen Palanisamy. Multi-agent connected autonomous driving using deep reinforcement learning. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–7. IEEE, 2020.

Xinlei Pan, Daniel Seita, Yang Gao, and John Canny. Risk averse robust adversarial reinforcement learning. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 8522–8528. IEEE, 2019.

Zhenghao Peng, Quanyi Li, Ka Ming Hui, Chunxiao Liu, and Bolei Zhou. Learning to simulate self-driven particles system with coordinated policy optimization. *Advances in Neural Information Processing Systems*, 34:10784–10797, 2021.

Lerrel Pinto, James Davidson, Rahul Sukthankar, and Abhinav Gupta. Robust adversarial reinforcement learning. In *International Conference on Machine Learning*, pp. 2817–2826. PMLR, 2017.

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.

Aizaz Sharif and Dusica Marijan. Evaluating the robustness of deep reinforcement learning for autonomous and adversarial policies in a multi-agent urban driving environment. *arXiv preprint arXiv:2112.11947*, 2021.

Simon Suo, Sebastian Regalado, Sergio Casas, and Raquel Urtasun. Trafficsim: Learning to simulate realistic multi-agent behaviors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10400–10409, 2021.

Ming Tan. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the tenth international conference on machine learning*, pp. 330–337, 1993.

Justin K Terry, Nathaniel Grammel, Ananth Hari, Luis Santos, and Benjamin Black. Revisiting parameter sharing in multi-agent deep reinforcement learning. *arXiv preprint arXiv:2005.13625*, 2020.

Martin Treiber and Arne Kesting. Traffic flow dynamics. *Traffic Flow Dynamics: Data, Models and Simulation, Springer-Verlag Berlin Heidelberg*, pp. 983–1000, 2013.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Eugene Vinitsky, Yuqing Du, Kanaad Parvate, Kathy Jang, Pieter Abbeel, and Alexandre Bayen. Robust reinforcement learning using adversarial populations. *arXiv preprint arXiv:2008.01825*, 2020.

Akifumi Wachi. Failure-scenario maker for rule-based agent using multi-agent adversarial reinforcement learning and its application to autonomous driving. In *International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence, 2019.

Jingke Wang, Tengju Ye, Ziqing Gu, and Junbo Chen. Ltp: Lane-based trajectory prediction for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 17134–17142, 2022.

Cathy Wu, Abdul Rahman Kreidieh, Kanaad Parvate, Eugene Vinitsky, and Alexandre M Bayen. Flow: A modular learning framework for mixed autonomy traffic. *IEEE Transactions on Robotics*, 2021.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. Deep sets. *Advances in neural information processing systems*, 30, 2017.

Hang Zhao, Jiyang Gao, Tian Lan, Chen Sun, Benjamin Sapp, Balakrishnan Varadarajan, Yue Shen, Yi Shen, Yuning Chai, Cordelia Schmid, et al. Tnt: Target-driven trajectory prediction. *arXiv preprint arXiv:2008.08294*, 2020.

Ming Zhou, Jun Luo, Julian Villella, Yaodong Yang, David Rusu, Jiayu Miao, Weinan Zhang, Montgomery Alban, IMAN FADAKAR, Zheng Chen, et al. Smarts: An open-source scalable multi-agent rl training school for autonomous driving. In *Conference on Robot Learning*, pp. 264–285. PMLR, 2021.