Explainable Autonomous Driving with Grounded Relational Inference

Chen Tang[1,2], **Nishan Srishankar**[1,3], **Sujitha Martin**[1], **and Masayoshi Tomizuka**[2]

[1]Honda Research Institute, CA, USA
[2]Department of Mechanical Engineering, University of California Berkeley, CA, USA
[3]Department of Robotics Engineering, Worcester Polytechnic Institute, MA, USA
chen_tang@berkeley.edu, nsrishankar@wpi.edu
smartin@honda-ri.com, tomizuka@berkeley.edu

Abstract

Explainability is essential for systems interacting with humans and other objects during operation (e.g., autonomous vehicles). Humans need to understand and anticipate the actions taken by the autonomous systems for trustful and safe cooperation. In this work, we aim to enable model explainability at the design stage by incorporating expert domain knowledge into the model and propose Grounded Relational Inference (GRI). It models an interactive system's underlying dynamics by inferring an interaction graph representing the agents' relations. We ensure an interpretable interaction graph by grounding the relational latent space into semantic behaviors defined with expert domain knowledge. We demonstrate that it can model simple interactive traffic scenarios under both simulation and real-world settings, and generate interpretable graphs explaining the vehicle's behavior by their interactions.

1 Introduction

Deep learning has been utilized to address various autonomous driving problems [1, 2, 3]. However, deep neural networks lack the transparency that helps people understand their underlying mechanism. It is a crucial drawback for safety-critical applications with humans involved (e.g., autonomous vehicles). Humans need to understand and anticipate the actions taken by the autonomous systems for trustful and safe cooperation. In response to this problem, the concept of explainable AI (XAI) was introduced. It refers to machine learning techniques that provide details and reasons that make a model's mechanism easy to understand [4]. Most of the existing works for deep learning models focus on post-hoc explanations [4]. They enhance model explainability by unraveling the underlying mechanisms after training: Vision-based approaches illustrate which segments of the input image affect the outputs, with visual attention [5] or deconvolution [6]; Interaction-aware models identify the agents that are critical to the decision-making procedure, with social attention in social LSTM [7, 8] or graph attention in graph neural networks (GNN) [9, 10, 11, 12].

Although promising, post-hoc explanations could be ambiguous and falsely interpreted by humans because of the non-interpretable nature of deep neural networks. Unless the model is interpretable by design, it is deceiving to claim that the generated post-hoc explanation can capture the model's underlying mechanism. In this work, we aim to improve interpretability at the design stage and develop a model that can generate interpretable explanations clearly defined in human domain knowledge and operate as the explanations suggest. We consider the problem of interactive system modeling—which is the foundation behind interaction-aware prediction and control models for autonomous vehicles—and follow the practice in Neural Relational Inference (NRI) [12] to model an

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

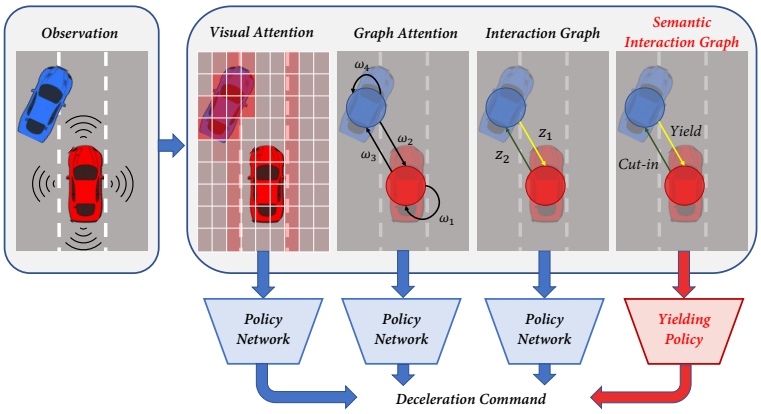

Figure 1: A motivating lane-changing scenario where we ask different models to control the red vehicle. In contrast to other frameworks, our proposed GRI model infers the interaction graph on a latent space grounded in yielding and cutting-in behaviors. It learns control policies that generate behaviors consistent with their definitions in domain knowledge (e.g. traffic rules) and executes the corresponding polices according to the inferred edge types.

interactive system by explicitly inferring its inherent interactions. Similar to NRI, our model outputs an interaction graph with discrete edge variables corresponding to a cluster of pairwise interactions between the agents. However, unlike NRI, which learns latent space in an unsupervised manner, we aim to ground it in a set of interactive behaviors defined with expert domain knowledge.

As a running example, consider the scenario depicted in Fig. 1, where we ask different models to control the red vehicle. Attention mechanisms can indicate the critical pixels or agents, but they cannot recognize different effects—the two cars are mutually important but affect each other in distinct ways. The NRI model can distinguish between different interactive behaviors. Still, the latent space does not have explicit semantic meaning. In contrast, our model should determine the interaction graph based on a latent space grounded in these two types of interactions. It learns control policies that generate behaviors consistent with their definitions in domain knowledge (e.g., traffic rules) and executes the corresponding policies according to the inferred edge types. Therefore, we ensure a semantic interaction graph, which illustrates the model's understanding of the scenario and explain the action it takes.

A straightforward way to enable semantic relations is using supervision. Interaction labels can be either obtained from human experts [13] or heuristic labeling functions [14]. However, accurate labels are practically prohibitive because human intentions are intricate and unobservable. Inaccurate labels could introduce bias and limit model capacity. Moreover, it is unclear if the model can understand the semantic meaning behind the labels and synthesize the right behaviors. Instead, we recast relational inference into an inverse reinforcement learning (IRL) problem and introduce structured reward functions to ground the latent space. Concretely, the system is modeled as a multi-agent Markov decision process (MDP), where the agents share a reward function that depends on the relational latent space. We design structured reward functions based on expert domain knowledge to explicitly define the interactive behaviors corresponding to the latent space. Compared to direct supervision, we merely specify the function space of the reward for each type of interaction, but leave the reward parameters and interaction graph—namely which reward function each agent follows—to be learned from data without supervision signals.

To solve the formulated IRL problem, we propose Grounded Relational Inference (GRI). It has a variational-autoencoder-like (VAE) GNN in NRI [12] as the backbone model. Additionally, we incorporate the structured reward functions into the model as a decoder. A variational extension of the adversarial inverse reinforcement learning (AIRL) algorithm is derived to train all the modules simultaneously. Experiments show that GRI can model simple interactive traffic scenarios under simulation and real-world settings, and generate interpretable graphs explaining the vehicle's behavior by their interactions. Moreover, the interpretable latent space enables humans to govern the model and ensure safety under unfamiliar situations. We believe that GRI is a critical step towards the next level of explainable models for autonomous driving and other multi-agent systems.

2 Related Work

Our model combines graph neural networks and adversarial inverse reinforcement learning for interactive system modeling. This section gives a concise literature review on these two topics and summarizes the existing works closely related to ours.

Interaction modeling using GNN. GNN has been widely applied for interactive system modeling [11, 15, 16]. VAIN [9] applied attention in multi-agent modeling. The attention map unravels the interior interaction structure to some extent. An approach closely related to ours is NRI [12], which modeled the interaction structure explicitly with discrete relational latent space. We explain the difference between NRI and our proposed method in Sec. 1 and 5. Another related approach in the autonomous driving domain is [14], which also modeled interactive driving behavior with semantically meaningful interactions but in a supervised manner.

Adversarial inverse reinforcement learning. Our work is related to two types of IRL approaches, multi-agent and latent AIRL algorithms. Yu et al. [17] proposed a multi-agent AIRL framework for Markov games under correlated equilibrium. The PS-GAIL algorithm [18] considered a multi-agent environment in the driving domain and extended GAIL [19] to model the interactive behaviors. In [20], they augmented the reward in PS-GAIL as a principle manner to specify prior knowledge, which shares the same spirit with our method. Latent AIRL models integrate a VAE into either the discriminator or generator for different purposes. Wang et al. [21] conditioned the discriminator on the embeddings generated by a VAE trained separately using behavior cloning. VDB [22] constrained information contained in the discriminator's internal representation to balance the training procedure for adversarial learning algorithms. The PEMIRL framework [23] encoded the demonstration into a contextual latent space to achieve meta-IRL. Though studied in a different context, it is conceptually similar to our framework as both its generator and discriminator depend on the latent variables.

3 Background: Neural Relational Inference

In this section, we briefly introduce NRI to get the readers familiar with the relational inference problem and related terminologies that will appear throughout the paper. In NRI, Kipf et al. [12] represent an interactive system as a complete bi-directed graph $\mathcal{G}_{\text{scene}} = (\mathcal{V}, \mathcal{E})$ with vertices $\mathcal{V} = \{v_i\}_{i=1}^{N}$ and edges $\mathcal{E} = \{e_{i,j} = (v_i, v_j) \mid i \neq j\}$,[1] where each vertex corresponds to an object. The NRI model is formalized as a VAE with a GNN encoder, $q_\phi(\mathbf{z}|\mathbf{x})$, which infers the underlying interactions, and a GNN decoder, $p_\eta(\mathbf{x}|\mathbf{z})$, which synthesizes the system dynamics given the interactions.

Specifically, the model takes a state trajectory as input, denoted by $\mathbf{x} = \left(\mathbf{x}^0, \ldots, \mathbf{x}^{T-1}\right)$ where $\mathbf{x}^t = \{\mathbf{x}_1^t, \ldots, \mathbf{x}_N^t\}$.[2] The vector $\mathbf{x}_i^t \in \mathbb{R}^n$ denotes the state vector of object v_i at time t. The encoder operates over $\mathcal{G}_{\text{scene}}$, with $\mathbf{x}_i$ as the node feature of v_i. It infers the posterior distribution of the edge type $z_{i,j}$ for all the edges, collected into a single vector $\mathbf{z}$. The decoder operates over an interaction graph $\mathcal{G}_{\text{interact}}$—which is constructed by assigning sampled $\mathbf{z}$ to the edges of $\mathcal{G}_{\text{scene}}$—together with the initial state. Based on them, the decoder reconstructs $\mathbf{x}$. The model is trained by maximizing the evidence lower bound (ELBO).

4 Problem Formulation

As stated before, we ground the relational latent space in GRI by reformulating the relational inference problem into an IRL problem. We start with modeling the interactive system as a multi-agent MDP with graph representation. As in NRI, the system has an underlying interaction graph $\mathcal{G}_{\text{interact}}$. The discrete latent variable $z_{i,j}$ takes a value from $0, 1, \ldots, K-1$. It indicates the type of relation between v_i and v_j in respect to its effect on v_j. Additionally, we assume the objects of the system are homogeneous intelligent agents who make decisions based on their interactions with others.

Concretely, each agent is modeled with identical state space $\mathcal{X}$, action space $\mathcal{A}$, transition operator $\mathcal{T}$ and reward function r. At time step t, the reward of agent v_j depends on the states and actions of

[1] The edge $e_{i,j}$ refers to the one pointing from v_i to v_j.

[2] Alternatively, the input sequence can be decomposed into $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_N)$ where $\mathbf{x}_i = \left\{\mathbf{x}_i^0, \ldots, \mathbf{x}_i^{T-1}\right\}$.

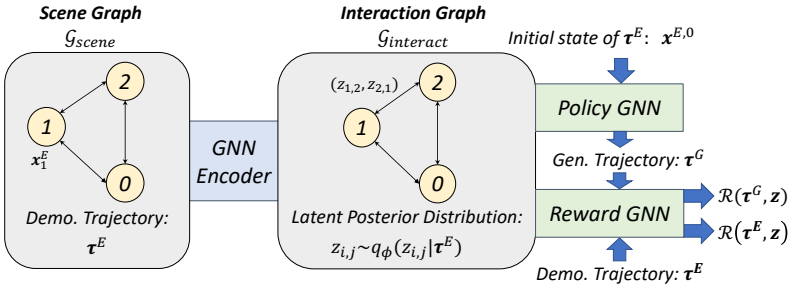

Figure 2: Architecture of grounded relational inference model.

itself and the pairwise interactions between itself and all its neighbors:

$$r_{\theta,\psi}(v_j^t, \mathbf{z}_j) = r_\theta^n(\mathbf{x}_j^t, \mathbf{a}_j^t) + \sum_{i \in \mathcal{N}_j} \sum_{k=1}^{K} \mathbf{1}(z_{i,j} = k) r_{\psi_k}^{e,k}(\mathbf{x}_i^t, \mathbf{a}_i^t, \mathbf{x}_j^t, \mathbf{a}_j^t), \tag{1}$$

where $\mathcal{N}_j$ is the set of v_j's neighbouring nodes, $\mathbf{z}_j$ is the vector collecting $\{z_{i,j}\}_{i \in \mathcal{N}_j}$, r_θ^n is the node reward function, and $r_{\psi_k}^{e,k}$ is the edge reward function for the k^{th} type of interaction. We utilize expert domain knowledge to design $r_{\psi_k}^{e,k}$, so that the corresponding interactive behavior emerges by maximizing the rewards. Particularly, the edge reward equals to zero for $k = 0$, indicating the action taken by v_j does not depend on its interaction with v_i.

We assume the agents act cooperatively to maximize the cumulative reward of the system:

$$\mathcal{R}_{\theta,\psi}(\boldsymbol{\tau}, \mathbf{z}) = \sum_{t=0}^{T-1} \mathbf{r}_{\theta,\psi}\left(\mathbf{x}^t, \mathbf{a}^t, \mathbf{z}\right) = \sum_{t=0}^{T-1} \sum_{j=1}^{N} r_{\theta,\psi}\left(v_j^t, \mathbf{z}_j\right),$$

with a joint policy denoted by $\boldsymbol{\pi}_\eta\left(\mathbf{a}^t | \mathbf{x}^t, \mathbf{z}\right)$. The cooperative assumption is not necessarily valid for generic multi-agent systems [17], but it simplifies the training procedure significantly. We will leave the extension of the proposed method to non-cooperative interactive systems as a future work.

Given a demonstration dataset, we aim to infer the underlying reward function and policy. Different from a typical IRL problem, both $r_{\theta,\psi}$ and π_η depend on $\mathbf{z}$. Therefore, we need to infer the distribution $p(\mathbf{z}|\boldsymbol{\tau})$ to solve the IRL problem.

5 Grounded Relational Inference

We now present the GRI model to solve the IRL problem. The model consists of three modules modeled by message-passing GNNs [24]: an encoder infers the posterior distribution of edge types, a policy decoder generates control actions conditioned on the edge variables sampled from the posterior distribution, and a reward decoder models the rewards conditioned on the inferred edge types.

5.1 Architecture

The overall model structure is illustrated in Fig. 2. Given a demonstration trajectory $\boldsymbol{\tau}^{\mathrm{E}} \in \mathcal{D}^{\mathrm{E}}$, the encoder operates over $\mathcal{G}_{\text{scene}}$ and infers the distribution $p(\mathbf{z}|\boldsymbol{\tau}^{\mathrm{E}})$ as $q_\phi(\mathbf{z}|\boldsymbol{\tau}^{\mathrm{E}})$. The policy decoder operates over a $\mathcal{G}_{\text{interact}}$ sampled from the inferred $q_\phi(\mathbf{z}|\boldsymbol{\tau}^{\mathrm{E}})$ and models the policy $\boldsymbol{\pi}_\eta\left(\mathbf{a}^t|\mathbf{x}^t, \mathbf{z}\right)$. Given the initial state of $\boldsymbol{\tau}^{\mathrm{E}}$, we sample a trajectory $\boldsymbol{\tau}^{\mathrm{G}}$ by sequentially sampling $\mathbf{a}^t$ from $\boldsymbol{\pi}_\eta\left(\mathbf{a}^t|\mathbf{x}^t, \mathbf{z}\right)$ and propagating the state. The state is propagated with either the transition operator $\mathcal{T}$ if given, or a simulating environment if $\mathcal{T}$ is not accessible. Since these two modules are essentially the same as in NRI, we omit the detailed model structures here and include them in Appx. 8.1.

Afterwards, we use the reward decoder to compute the cumulative rewards of $\boldsymbol{\tau}^{\mathrm{G}}$ and $\boldsymbol{\tau}^{\mathrm{E}}$ conditioned on the sampled $\mathcal{G}_{\text{interact}}$. The reward decoder is in the form of Eqn. (1). Additionally, we augment the functions r_θ^n and $r_{\phi_k}^{e,k}$ with MLP shaping terms to mitigate the reward shaping effect [25], resulting

$$\max_{\eta} \min_{\theta,\omega,\psi,\xi,\phi} = \mathbb{E}_{\boldsymbol{\tau}^{\mathrm{E}}\sim\boldsymbol{\pi}^{\mathrm{E}}(\boldsymbol{\tau})}\Bigg\{\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\boldsymbol{\tau}^{\mathrm{E}})}\Bigg[-\sum_{t=0}^{T-1}\log \mathcal{D}_{\theta,\omega,\psi,\xi,\eta}(\mathbf{x}^{\mathrm{E},t},\mathbf{a}^{\mathrm{E},t},\mathbf{x}^{\mathrm{E},t+1},\mathbf{z}) \\ - \mathbb{E}_{\boldsymbol{\tau}^{\mathrm{G}}\sim\boldsymbol{\pi}_\eta(\boldsymbol{\tau}|\mathbf{z})}\sum_{t=0}^{T-1}\log\left(1-\mathcal{D}_{\theta,\omega,\psi,\xi,\eta}(\mathbf{x}^{\mathrm{G},t},\mathbf{a}^{\mathrm{G},t},\mathbf{x}^{\mathrm{G},t+1},\mathbf{z})\right)\Bigg]\Bigg\}, \quad (2)$$
$$\text{s.t.}\quad \mathbb{E}_{\boldsymbol{\tau}^{\mathrm{E}}\sim\boldsymbol{\pi}^{\mathrm{E}}(\boldsymbol{\tau})}\left\{D_{\mathrm{KL}}\left[q_\phi\left(\mathbf{z}|\boldsymbol{\tau}^{\mathrm{E}}\right))||p(\mathbf{z})\right]\right\} \leqslant I_c,$$

in:

$$f^n_{\theta,\omega}(\mathbf{x}^t_j,\mathbf{a}^t_j,\mathbf{x}^{t+1}_j) = r^n_\theta(\mathbf{x}^t_j,\mathbf{a}^t_j) + h^n_\omega(\mathbf{x}^{t+1}_j) - h^n_\omega(\mathbf{x}^t_j),$$
$$f^{e,k}_{\psi_k,\xi_k}(\mathbf{x}^t_i,\mathbf{a}^t_i,\mathbf{x}^{t+1}_i,\mathbf{x}^t_j,\mathbf{a}^t_j,\mathbf{x}^{t+1}_j) = r^{e,k}_{\psi_k}(\mathbf{x}^t_i,\mathbf{a}^t_i,\mathbf{x}^t_j,\mathbf{a}^t_j) + h^{e,k}_{\xi_k}(\mathbf{x}^{t+1}_i,\mathbf{x}^{t+1}_j) - h^{e,k}_{\xi_k}(\mathbf{x}^t_i,\mathbf{x}^t_j).$$

The shaped reward function, $\mathbf{f}_{\theta,\omega,\psi,\xi}\left(\mathbf{x}^t,\mathbf{a}^t,\mathbf{x}^{t+1},\mathbf{z}\right)$, together with the policy model, defines the discriminator which distinguishes $\boldsymbol{\tau}^{\mathrm{G}}$ from $\boldsymbol{\tau}^{\mathrm{E}}$:

$$\mathcal{D}_{\theta,\omega,\psi,\xi,\eta}(\mathbf{x}^t,\mathbf{a}^t,\mathbf{x}^{t+1},\mathbf{z}) = \frac{\exp\left\{\mathbf{f}_{\theta,\omega,\psi,\xi}\left(\mathbf{x}^t,\mathbf{a}^t,\mathbf{x}^{t+1},\mathbf{z}\right)\right\}}{\exp\left\{\mathbf{f}_{\theta,\omega,\psi,\xi}\left(\mathbf{x}^t,\mathbf{a}^t,\mathbf{x}^{t+1},\mathbf{z}\right)\right\} + \boldsymbol{\pi}_\eta\left(\mathbf{a}^t|\mathbf{x}^t,\mathbf{z}\right)}.$$

5.2 Training

We aim to train the three modules simultaneously. Consequently, we incorporate the encoder model $q_\phi\left(\mathbf{z}|\boldsymbol{\tau}^{\mathrm{E}}\right)$ into the objective function of AIRL, resulting in the optimization problem (2). The encoder is integrated into the minimization problem because the reward function has a direct dependence on the latent space. The model is then trained by solving problem (2) in an adversarial scheme: we alternate between training the encoder and reward for the minimization problem and training the policy for the maximization problem. The constraint enforces an upper bound I_c on the KL-divergence between $q_\phi\left(\mathbf{z}|\boldsymbol{\tau}^{\mathrm{E}}\right)$ and the prior $p(\mathbf{z})$. A sparse prior is chosen to encourage sparsity in $\mathcal{G}_{\mathrm{interact}}$. It has the similar regularization effect as the D_{KL} term in ELBO. We borrow its format from variational discriminator bottleneck (VDB) [22]. Although having different motivation, we adopt it because the constrained problem can be relaxed by introducing a Lagrange multiplier β. During training, β is updated through dual gradient descent as follows:

$$\beta \leftarrow \max\left(0, \alpha_\beta\left(\mathbb{E}\left\{D_{\mathrm{KL}}\left[q_\phi\left(\mathbf{z}|\boldsymbol{\tau}^{\mathrm{E}}\right))||p(\mathbf{z})\right]\right\} - I_c\right)\right). \quad (3)$$

We find the adaptation scheme particularly advantageous as the model can focus on inferring $\mathbf{z}$ for reward learning after satisfying the sparsity constraint, because the magnitude of β decreases towards zero once the constraint is satisfied. However, it is worth noting that our framework does not rely on the bottleneck constraint to induce an interpretable latent space as in [26]. In contrast, GRI relies on the structured reward functions to ground the latent space into semantic interactive behaviors. The bottleneck serves as a regularization to find out the minimal interaction graph to represent the interactions. In fact, we show in the experiments that the constraint itself is not sufficient to induce a sparse and interpretable interaction graph, by training baseline NRI models with the same constraint and weight update scheme.

In general, when the dynamics $\mathcal{T}$ is unknown or non-differentiable, maximum entropy RL algorithms [27] are adopted to optimize the policy. We assume known and differentiable dynamics in this work, which is a reasonable assumption for the investigated scenarios. It allows us to directly backpropagate through the trajectory for gradient estimation, which simplifies the training procedure.

6 Experiments

We evaluated the proposed GRI model on a synthetic dataset as well as a naturalistic traffic dataset. The synthetic data were generated using policy models trained given the ground-truth reward function and interaction graph. We intend to verify if GRI can induce an interpretable relational latent space and infer the underlying relations precisely. The naturalistic traffic data were extracted from the

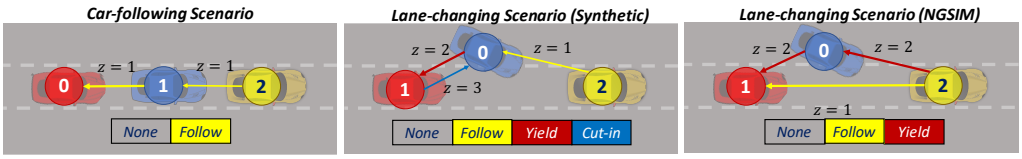

Figure 3: Synthetic and naturalistic traffic scenarios.

NGSIM dataset. We aim to validate if GRI can model real-world traffic scenarios effectively with the grounded interpretable latent space. Unlike synthetic agents, we do not have the privilege to access the graph governing human drivers' interactions. Instead, we constructed hypothetical graphs after analyzing the segmented data. The hypotheses reflect humans' understanding of the traffic scenarios. We would like to see if GRI can model real-world interactive systems in the same way as humans. We claim the model interpretable if the inferred interaction graphs are consistent with the hypotheses.

To evaluate a trained model, we sample a $\boldsymbol{\tau}^{\text{exp}}$ from the test dataset and extract the maximum posterior probability (MAP) estimate of edge variables, $\hat{\mathbf{z}}$, from $q_\phi(\mathbf{z}|\boldsymbol{\tau}^{\text{exp}})$. Afterward, we obtain a single sample of trajectories $\hat{\boldsymbol{\tau}}$ by executing the mean value of the policy outputs. The root mean square errors (RMSE) of states and the accuracy of $\mathcal{G}_{\text{interact}}$ are selected as the evaluation metrics, which are computed based on $\hat{\mathbf{z}}$, $\hat{\boldsymbol{\tau}}$, $\boldsymbol{\tau}^{\text{exp}}$, and the ground truth or hypothetical latent variables denoted by $\mathbf{z}^{\text{exp}}$:

$$\text{RMSE}_\epsilon = \sqrt{\frac{1}{(N-1)T}\sum_{j=1}^{N}\sum_{t=0}^{T-1}(\epsilon_j^{\text{exp},t} - \hat{\epsilon}_j^t)^2}, \quad \text{Accuracy} = \frac{\sum_{i=1}^{N}\sum_{j=1,j\neq i}^{N}\mathbf{1}(z_{i,j}^{\text{exp}} = \hat{z}_{i,j})}{N(N-1)}.$$

If multiple edge types exist, we test all the possible permutations of edge types and report the one with the highest graph accuracy for NRI.

6.1 Synthetic Scenes

We designed two synthetic scenarios, car-following (CF) and lane-changing (LC). The two scenes and their underlying interaction graphs are illustrated in Fig. 3. In both scenarios, we have a leading vehicle whose behavior does not depend on the others, and its trajectory is given without the need for reconstruction. We assume it runs at a constant velocity. The other vehicles interact with each other and the leading one in different ways. In CF, we model the system with two types of edges: $z_{i,j} = 1$ means that Vehicle j follows Vehicle i; $z_{i,j} = 0$ means that Vehicle j does not interact with Vehicle i. In LC, two additional edge types are introduced: $z_{i,j} = 2$ means that Vehicle j yields to Vehicle i; $z_{i,j} = 3$ means that Vehicle j cuts in front of Vehicle i. The MDPs for the tested scenarios, including the dynamics models and the designed structured reward functions, are summarized in Appx. 8.2.

Results. After training the expert policy, we generated the demonstration dataset by randomly sampling all the vehicles' initial states. For each scenario, we trained a GRI model with the policy decoder (6)-(8) introduced in Appx. 8.1. As a baseline, we trained a NRI model with the same encoder and policy decoder. The same adaptation scheme in Eqn. (3) was applied to enforce sparsity.

The results are summarized in Table 1. The NRI model can reconstruct the trajectories with smaller errors. However, it does not recover the ground-truth $\mathcal{G}_{\text{interact}}$ precisely, which indicates a relational latent space that is different from the one underlying the demonstration; Therefore, the edge variables cannot be interpreted as those semantically meaningful behaviors. In contrast, our GRI model interprets the interactions in the same way as the domain knowledge and recovers the interaction graph with high accuracies.

To further evaluate the models' explainability, we computed the empirical distribution of the estimated edge variables $\hat{z}$ over the test dataset (Fig. 4). The distribution concentrates on a single interaction graph for both models in both scenarios—as opposed to the case on the naturalistic traffic dataset, which we will mention later—because the synthetic agents have the same interaction patterns over all the samples. In CF, the interaction graph generated by the NRI model has two additional edges compared to the ground-truth one: $z_{2,0} = 1$ and $z_{0,1} = 1$. It is relatively reasonable to have $z_{2,0} = 1$ because Vehicle 2 indirectly affects Vehicle 0. On the other hand, $z_{0,1} = 1$ is not consistent with the inherent causality and cannot be interpreted as car-following as the other edges. In LC, the NRI model

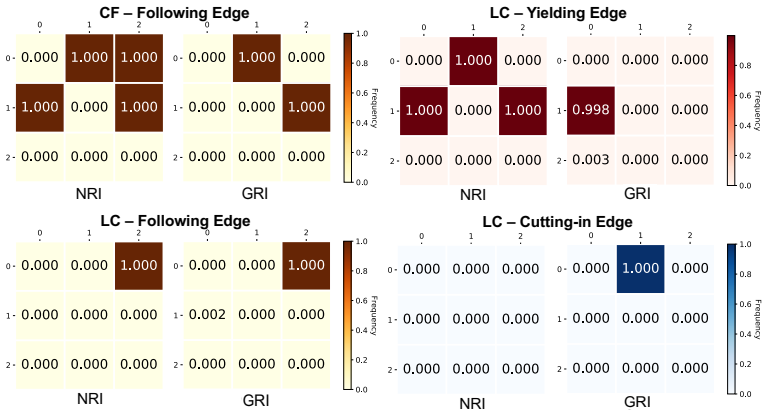

Figure 4: The empirical distribution of estimated edge variables $\hat{z}$ over the test dataset in the synthetic scenarios (CF: car following, LC: lane changing). We summarize the results in multiple adjacency matrices corresponding to different edge types. In the adjacency matrix corresponding to the k^{th} type of interaction, the element $A_{i,j}$ indicates the relative frequency of $z_{j,i} = k$, where $z_{j,i}$ is the latent variable for the edge from node j to node i.

treats the edges $e_{1,0}$, $e_{0,1}$, and $e_{2,1}$ the same, which makes it challenging to interpret the semantic meaning behind because they represent different interactive behaviors in the expert demonstration.

Some may argue that predictive accuracy is more critical than explainability when evaluating the models. We want to emphasize that explainability is at least an equally important metric, especially for human-robot interaction. With GRI, the safety driver or passengers of an autonomous vehicle can override the inferred $\mathcal{G}_{\text{interact}}$ if the model misunderstands the scenario. We show in Appx. 8.4 that, because of the semantic meaningful latent space and policy, the GRI policy can induce behaviors suggested by the enforced graph even in novel circumstances. It implies a safe and reliable way to operate the autonomous vehicle in unfamiliar situations through trustful cooperation with humans, which cannot be achieved by an inexplicable model. Such kind of safety assurance is essential when deploying autonomous vehicles in real-world environments.

6.2 Naturalistic Traffic Scenes

To evaluate the proposed method in real-world traffic scenarios, we investigated the same scenarios as in the synthetic case, car-following, and lane-changing. We segmented data from the Highway-101 and I-80 datasets of NGSIM. Afterward, we further screened the data to select those interactive samples and ensure that no erratic swerving or multiple lane changes occur. The hypotheses for the two scenarios are depicted in Fig. 3. The one for CF is identical to the ground-truth interaction graph for the synthetic agents. However, we proposed a different hypothesis for LC. We excluded the cutting-in relation to reduce the number of edge types and therefore simplify the training procedure. Moreover, we defined the interactions according to the vehicles' lateral position. We say that a vehicle yields to its preceding vehicle if they drive in neighboring lanes, whereas it follows the preceding one

Table 1: Performance Comparison on Synthetic Dataset

(a) Car-Following Scenario ($T = 20, \Delta t = 0.2s$)

Model	$\text{RMSE}_x(\text{m})$	$\text{RMSE}_v(\text{m/s})$	$\text{Accuracy}(\%)$
GRI	0.241 ± 0.125	0.174 ± 0.068	$\mathbf{100.00 \pm 0.00}$
NRI	$\mathbf{0.047 \pm 0.024}$	$\mathbf{0.056 \pm 0.0148}$	66.70 ± 0.00

(b) Lane-Changing Scenario ($T = 30, \Delta t = 0.2s$)

Model	$\text{RMSE}_x(\text{m})$	$\text{RMSE}_y(\text{m})$	$\text{RMSE}_v(\text{m/s})$	$\text{Accuracy}(\%)$
GRI	0.636 ± 0.297	0.273 ± 0.060	0.344 ± 0.135	$\mathbf{99.95 \pm 0.01}$
NRI	$\mathbf{0.116 \pm 0.046}$	$\mathbf{0.197 \pm 0.049}$	$\mathbf{0.084 \pm 0.022}$	66.70 ± 0.00

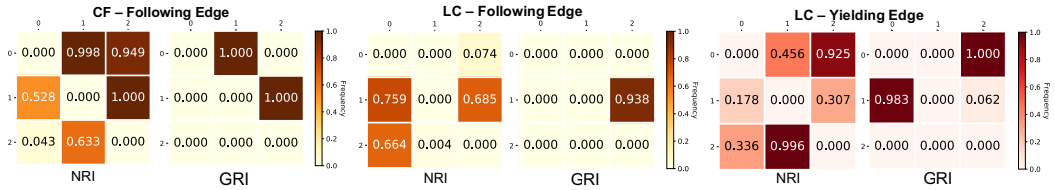

Figure 5: The empirical distribution of estimated edge variables $\hat{z}$ over the test dataset in the naturalistic traffic scenarios. See Fig.4 on how to interpret the results.

if they are in the same lane. We omit the detailed description of the dynamics models and reward functions and summarize them in Appx. 8.3 due to the limited space.

Results. For each scenario, we trained a GRI model with the recurrent policy decoder (9)-(12) in Appx. 8.1. As in the synthetic scene, we trained a NRI model with the same encoder, policy decoder, and the adaptation scheme as a baseline approach.

The results are summarized in Table 2. In both scenarios, the NRI model slightly outperforms our model in trajectory reconstruction, but the RMSEs of the two models are comparable. Meanwhile, GRI dominates NRI in graph accuracy. We visualize the interaction graphs in Fig. 5. One interesting observation is that the graphs inferred by NRI have more edges in general. We want to emphasize that both models are trained under the same sparsity constraint. The results imply that we can guide the model to explore a clean and sparse representation of interactions by incorporating relevant domain knowledge. Moreover, the NRI model assigns the same edge type to both edges between a pair of agents. It makes the graphs less interpretable because the vehicles ought to affect each other in different ways. On the other hand, even if different from the hypotheses, our GRI model tends to infer sparse graphs with directional edges.

Table 2: Performance Comparison on Naturalistic Traffic Dataset

(a) Car-Following Scenario ($T = 30, \Delta t = 0.2s$)

Model	$\text{RMSE}_x(\text{m})$	$\text{RMSE}_v(\text{m/s})$	$\text{Accuracy}(\%)$
GRI	1.700 ± 1.005	0.721 ± 0.363	$\mathbf{100.00 \pm 0.00}$
NRI	$\mathbf{1.436 \pm 0.880}$	$\mathbf{0.650 \pm 0.328}$	64.09 ± 0.08

(b) Lane-Changing Scenario ($T = 40, \Delta t = 0.2s$)

Model	$\text{RMSE}_x(\text{m})$	$\text{RMSE}_y(\text{m})$	$\text{RMSE}_v(\text{m/s})$	$\text{Accuracy}(\%)$
GRI	7.118 ± 3.647	0.764 ± 0.336	4.320 ± 2.392	$\mathbf{98.55 \pm 0.06}$
NRI	$\mathbf{6.532 \pm 3.822}$	$\mathbf{0.330 \pm 0.181}$	$\mathbf{4.291 \pm 2.544}$	28.98 ± 0.08

7 Discussion and Conclusion

In this work, we propose Grounded Relational Inference (GRI), which models an interactive system's underlying dynamics by inferring the agents' semantic relations. By incorporating structured reward functions, we ground the relational latent space into semantically meaningful behaviors defined with expert domain knowledge. Therefore, we assure an interpretable interaction graph at the design stage. We demonstrate that it can model simple traffic scenarios under both simulation and real-world settings, and generate interpretable graphs explaining the vehicle's behavior by their interactions.

Although we limit our experimental study to the autonomous driving domain, the model itself is formulated without specifying the context. As long as proper domain knowledge is available, the proposed method can be extended naturally to other fields (e.g., human-robot interaction). However, there are several technical gaps we need to bridge before extending the current framework to more complicated traffic scenarios and interactive systems in other fields. One gap between the current model and these practical modules is graph dynamics. Throughout the paper, we assume a static interaction graph over the time horizon. We will investigate how to incorporate dynamic graph modeling into the current framework. Another gap is the cooperative assumption, which we would like to remove in the future so that the framework can be generalized to non-cooperative scenarios.

References

[1] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, *et al.*, "End to end learning for self-driving cars," *arXiv preprint arXiv:1604.07316*, 2016.

[2] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3d object detection network for autonomous driving," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1907–1915, 2017.

[3] C. Tang, Z. Xu, and M. Tomizuka, "Disturbance-Observer-Based Tracking Controller for Neural Network Driving Policy Transfer," *IEEE Transactions on Intelligent Transportation Systems*, 2019.

[4] A. B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-López, D. Molina, R. Benjamins, *et al.*, "Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai," *Information Fusion*, vol. 58, pp. 82–115, 2020.

[5] J. Kim and J. Canny, "Interpretable learning for self-driving cars by visualizing causal attention," in *Proceedings of the IEEE international conference on computer vision (ICCV)*, pp. 2942–2950, 2017.

[6] M. Bojarski, A. Choromanska, K. Choromanski, B. Firner, L. J. Ackel, U. Muller, P. Yeres, and K. Zieba, "Visualbackprop: Efficient visualization of cnns for autonomous driving," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1–8, IEEE, 2018.

[7] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social lstm: Human trajectory prediction in crowded spaces," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 961–971, 2016.

[8] A. Vemula, K. Muelling, and J. Oh, "Social attention: Modeling attention in human crowds," in *2018 IEEE international Conference on Robotics and Automation (ICRA)*, pp. 1–7, IEEE, 2018.

[9] Y. Hoshen, "Vain: Attentional multi-agent predictive modeling," in *Advances in Neural Information Processing Systems (NIPS)*, pp. 2701–2711, 2017.

[10] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.

[11] S. Sukhbaatar, A. Szlam, and R. Fergus, "Learning multiagent communication with backpropagation," in *Advances in Neural Information Processing Systems (NIPS)*, pp. 2244–2252, 2016.

[12] T. Kipf, E. Fetaya, K.-C. Wang, M. Welling, and R. Zemel, "Neural relational inference for interacting systems," *arXiv preprint arXiv:1802.04687*, 2018.

[13] L. Sun, W. Zhan, and M. Tomizuka, "Probabilistic prediction of interactive driving behavior via hierarchical inverse reinforcement learning," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pp. 2111–2117, IEEE, 2018.

[14] D. Lee, Y. Gu, J. Hoang, and M. Marchetti-Bowick, "Joint interaction and trajectory prediction for autonomous driving using graph neural networks," *arXiv preprint arXiv:1912.07882*, 2019.

[15] S. Van Steenkiste, M. Chang, K. Greff, and J. Schmidhuber, "Relational neural expectation maximization: Unsupervised discovery of objects and their interactions," *arXiv preprint arXiv:1802.10353*, 2018.

[16] P. Battaglia, R. Pascanu, M. Lai, D. J. Rezende, *et al.*, "Interaction networks for learning about objects, relations and physics," in *Advances in neural information processing systems*, pp. 4502–4510, 2016.

[17] L. Yu, J. Song, and S. Ermon, "Multi-agent adversarial inverse reinforcement learning," *arXiv preprint arXiv:1907.13220*, 2019.

[18] R. P. Bhattacharyya, D. J. Phillips, B. Wulfe, J. Morton, A. Kuefler, and M. J. Kochenderfer, "Multi-agent imitation learning for driving simulation," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1534–1539, IEEE, 2018.

[19] J. Ho and S. Ermon, "Generative adversarial imitation learning," in *Advances in neural information processing systems*, pp. 4565–4573, 2016.

[20] R. P. Bhattacharyya, D. J. Phillips, C. Liu, J. K. Gupta, K. Driggs-Campbell, and M. J. Kochenderfer, "Simulating emergent properties of human driving behavior using multi-agent reward augmented imitation learning," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 789–795, IEEE, 2019.

[21] Z. Wang, J. S. Merel, S. E. Reed, N. de Freitas, G. Wayne, and N. Heess, "Robust imitation of diverse behaviors," in *Advances in Neural Information Processing Systems*, pp. 5320–5329, 2017.

[22] X. B. Peng, A. Kanazawa, S. Toyer, P. Abbeel, and S. Levine, "Variational discriminator bottleneck: Improving imitation learning, inverse rl, and gans by constraining information flow," *arXiv preprint arXiv:1810.00821*, 2018.

[23] L. Yu, T. Yu, C. Finn, and S. Ermon, "Meta-inverse reinforcement learning with probabilistic context variables," in *Advances in Neural Information Processing Systems*, pp. 11772–11783, 2019.

[24] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1263–1272, JMLR. org, 2017.

[25] J. Fu, K. Luo, and S. Levine, "Learning robust rewards with adversarial inverse reinforcement learning," *arXiv preprint arXiv:1710.11248*, 2017.

[26] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, "beta-vae: Learning basic visual concepts with a constrained variational framework," 2016.

[27] S. Levine, "Reinforcement learning and control as probabilistic inference: Tutorial and review," *arXiv preprint arXiv:1805.00909*, 2018.

[28] A. Kesting, M. Treiber, and D. Helbing, "Enhanced intelligent driver model to access the impact of driving strategies on traffic capacity," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 368, no. 1928, pp. 4585–4605, 2010.

[29] M. Treiber, A. Hennecke, and D. Helbing, "Congested traffic states in empirical observations and microscopic simulations," *Physical review E*, vol. 62, no. 2, p. 1805, 2000.

8 Appendix

8.1 Graph Neural Network Model Details

In terms of model structure, both the encoder and policy decoder are built based on node-to-node message-passing [24], consisting of a node-to-edge message-passing and an edge-to-node message-passing:

$$v \to e: \quad \mathbf{h}_{i,j}^{l} = f_e^{l}(\mathbf{h}_i^{l}, \mathbf{h}_j^{l}, \mathbf{x}_{i,j}), \tag{4}$$

$$e \to v: \quad \mathbf{h}_j^{l+1} = f_v^{l}(\sum\nolimits_{i \in \mathcal{N}_j} \mathbf{h}_{i,j}^{l}, \mathbf{x}_j), \tag{5}$$

where $\mathbf{h}_i^l$ is the embedded hidden state of node v_i in the l^{th} layer and $\mathbf{h}_{i,j}^l$ is the embedded hidden state of the edge $e_{i,j}$. The features $\mathbf{x}_i$ and $\mathbf{x}_{i,j}$ are assigned to the node v_i and the edge $e_{i,j}$ respectively as inputs. $\mathcal{N}_j$ denotes the set of the indices of v_i's neighbouring nodes connected by an incoming edge. The functions f_e^l and f_v^l are neural networks for edges and nodes respectively, shared across the graph within the l^{th} layer of node-to-node massage-passing.

GNN Encoder. The GNN encoder is essentially the same as in NRI. It models the posterior distribution as $q_\phi(\mathbf{z}|\boldsymbol{\tau})$ with the following operations:

$$\begin{aligned}
\mathbf{h}_j^1 &= f_{\text{emb}}(\mathbf{x}_j), \\
v \to e: \quad \mathbf{h}_{i,j}^1 &= f_e^1(\mathbf{h}_i^1, \mathbf{h}_j^1), \\
e \to v: \quad \mathbf{h}_j^2 &= f_v^1\left(\sum\nolimits_{i \neq j} \mathbf{h}_{i,j}^1\right), \\
v \to e: \quad \mathbf{h}_{i,j}^2 &= f_e^2(\mathbf{h}_i^2, \mathbf{h}_j^2), \\
q_\phi(\mathbf{z}_{i,j}|\boldsymbol{\tau}) &= \text{softmax}\left(\mathbf{h}_{i,j}^2\right),
\end{aligned}$$

where f_e^1, f_v^1 and f_e^2 are fully-connected networks (MLP) and f_{emb} is a 1D convolutional networks (CNN) with attentive pooling.

GNN Policy Decoder. The policy operates over $\mathcal{G}_{\text{interact}}$ and models the distribution $\boldsymbol{\pi}_\eta\left(\mathbf{a}^t|\mathbf{x}^t, \mathbf{z}\right)$, which can be factorized with $\pi_\eta\left(\mathbf{a}_j^t|\mathbf{x}^t, \mathbf{z}\right)$. We model π_η as a Gaussian distribution with the mean value parameterized by the following GNN:

$$v \to e: \quad \tilde{\mathbf{h}}_{i,j}^t = \sum_{k=0}^{K} \mathbf{1}(z_{i,j} = k) \tilde{f}_e^k(\mathbf{x}_i^t, \mathbf{x}_j^t), \tag{6}$$

$$e \to v: \quad \mu_j^t = \tilde{f}_v\left(\sum\nolimits_{i \neq j} \tilde{\mathbf{h}}_{i,j}^t\right), \tag{7}$$

$$\pi_\eta\left(\mathbf{a}_j^t|\mathbf{x}^t, \mathbf{z}\right) = \mathcal{N}(\boldsymbol{\mu}_j^t, \sigma^2\mathbf{I}). \tag{8}$$

Alternatively, the model capacity is improved by using a recurrent policy $\pi_\eta\left(\mathbf{a}_j^t|\mathbf{x}^t, \ldots, \mathbf{x}^1, \mathbf{z}\right)$; Namely, the agents take actions according to the historical trajectories of the system. We follow the practice in [12] and add a GRU unit to obtain the following recurrent model:

$$v \to e: \quad \tilde{\mathbf{h}}_{i,j}^t = \sum_{k=0}^{K} \mathbf{1}(z_{i,j} = k) \tilde{f}_e^k\left(\tilde{\mathbf{h}}_i^t, \tilde{\mathbf{h}}_j^t\right), \tag{9}$$

$$e \to v: \quad \tilde{\mathbf{h}}_j^{t+1} = \text{GRU}\left(\sum\nolimits_{i \neq j} \tilde{\mathbf{h}}_{i,j}^t, \mathbf{x}_j^t, \tilde{\mathbf{h}}_j^t\right), \tag{10}$$

$$\mu_j^t = f_{\text{out}}\left(\tilde{\mathbf{h}}_j^{t+1}\right), \tag{11}$$

$$\pi_\eta\left(\mathbf{a}_j^t|\mathbf{x}^t, \ldots, \mathbf{x}^1, \mathbf{z}\right) = \mathcal{N}(\boldsymbol{\mu}_j^t, \sigma^2\mathbf{I}), \tag{12}$$

where $\tilde{\mathbf{h}}_i^t$ is the recurrent hidden state encoding the historical information up to the time step $t-1$.

8.2 Synthetic Scene MDPs Specification

The MDPs for the synthetic scenarios are specified as follows. In CF, since the vehicles mainly interact in longitudinal direction, we model their dynamics with 1D point-mass model. Specifically, the dynamics is governed by the following equations:

$$x_j^{t+1} = x_j^t + v_j^t \Delta t + \frac{1}{2} a_j^t \Delta t^2, \tag{13}$$

$$v_j^{t+1} = v_j^t + a_j^t \Delta t, \tag{14}$$

$$a_j^{t+1} = a_j^t + \delta a_j^t \Delta t, \tag{15}$$

where x_j^t is the longitudinal coordinate, v_j^t is the velocity, a_j^t is the acceleration, δa_j^t is the jerk, and Δt is the sampling time. In LC, we consider both longitudinal and lateral motions and model the vehicles as Dubin's cars:

$$\begin{aligned}
x_j^{t+1} &= x_j^t + v_j^t \cos(\theta_j^t) \Delta t, \\
y_j^{t+1} &= y_j^t + v_j^t \sin(\theta_j^t) \Delta t, \\
v_j^{t+1} &= v_j^t + a_j^t \Delta t, \\
\theta_j^{t+1} &= \theta_j^t + \omega_j^t \Delta t, \\
a_j^{t+1} &= a_j^t + \delta a_j^t \Delta t, \\
\omega_j^{t+1} &= \omega_j^t + \delta \omega_j^t \Delta t,
\end{aligned}$$

where y_j^t is the lateral coordinate, θ_j^t is the yaw angle, ω_j^t is the yaw rate, $\delta \omega_j^t$ is the yaw acceleration, and the remaining terms are the same as in (13)-(15).

The structured reward functions were designed based on expert domain knowledge (e.g. transportation studies [28, 29]). We mainly referred to [13] in this paper. For the car-following behavior, its reward function is defined as follows:

$$\begin{aligned}
r_{\psi_1}^{e,1}\left(\mathbf{x}_i^t, \mathbf{x}_j^t\right) = &- \left(1 + \exp(\psi_{1,0})\right) g_{\text{IDM}}(\mathbf{x}_i^t, \mathbf{x}_j^t) \\
&- \left(1 + \exp(\psi_{1,1})\right) g_{\text{dist}}(\mathbf{x}_i^t, \mathbf{x}_j^t) \\
&- \left(1 + \exp(\psi_{1,2})\right) g_{\text{lat}}(\mathbf{x}_i^t, \mathbf{x}_j^t),
\end{aligned}$$

where the features are defined as:

$$g_{\text{IDM}}(\mathbf{x}_i^t, \mathbf{x}_j^t) = \left(\max\left(x_i^t - x_j^t, 0\right) - \Delta x_{i,j}^{\text{IDM},t}\right)^2, \tag{16}$$

$$g_{\text{dist}}(\mathbf{x}_i^t, \mathbf{x}_j^t) = \exp\left(-\frac{\left(\max\left(x_i^t - x_j^t, 0\right)\right)^2}{\zeta^2}\right), \tag{17}$$

$$g_{\text{lat}}(\mathbf{x}_i^t, \mathbf{x}_j^t) = \left(y_j^t - g_{\text{center}}(y_i^t)\right)^2.$$

The feature g_{IDM} suggests a spatial headway $\Delta x_{i,j}^{\text{IDM},t}$ derived from the intelligent driver model (IDM) [28]. The feature f_{dist} ensures a minimum collision-free distance. We penalize the following vehicle for surpassing the preceding one with the help of $x_{i,j}^{\text{IDM},t}$ in Eqn. (16) and Eqn. (17). The last feature g_{lat} exists only in LC. It regulates the following vehicle to stay in the same lane as the preceding one with the help of g_{center}, which determines the lateral coordinate of the corresponding centerline based on the position of the preceding vehicle.

The reward function for yielding is defined as:

$$\begin{aligned}
r_{\psi_2}^{e,2}\left(\mathbf{x}_i^t, \mathbf{x}_j^t\right) = &- \left(1 + \exp(\psi_{2,0})\right) g_{\text{yield}}(\mathbf{x}_i^t, \mathbf{x}_j^t) \\
&- \left(1 + \exp(\psi_{2,1})\right) g_{\text{dist}}(\mathbf{x}_i^t, \mathbf{x}_j^t).
\end{aligned}$$

The feature g_{dist} is defined in Eqn. (17). The other feature g_{yield} suggests a spatial headway appropriate for yielding:

$$\begin{aligned}
g_{\text{yield}}(\mathbf{x}_i^t, \mathbf{x}_j^t) = &\mathbf{1}\left(g_{\text{center}}(y_j^t) = g_{\text{center}}(y_i^t)\right) g_{\text{IDM}}(\mathbf{x}_i^t, \mathbf{x}_j^t) \\
&+ \mathbf{1}\left(g_{\text{center}}(y_j^t) \neq g_{\text{center}}(y_i^t)\right) g_{\text{goal}}(\mathbf{x}_i^t, \mathbf{x}_j^t), \\
g_{\text{goal}}(\mathbf{x}_i^t, \mathbf{x}_j^t) = &\left(\max\left(x_i^t - x_j^t - \Delta x^{\text{yield}}, 0\right)\right)^2.
\end{aligned} \tag{18}$$

The suggested headway is set to be a constant value, Δx^{yield}, when the other vehicle is merging, and switches to $\Delta x_{i,j}^{\text{IDM},t}$ once the merging vehicle enters into the same lane, where its behavior becomes consistent with car following.

The reward function for cutting-in is quite similar:

$$\begin{aligned} r_{\psi_3}^{e,3}\left(\mathbf{x}_i^t, \mathbf{x}_j^t\right) = &- \left(1+\exp(\psi_{3,0})\right) g_{\text{goal}}(\mathbf{x}_j^t, \mathbf{x}_i^t) \\ &- \left(1+\exp(\psi_{3,1})\right) g_{\text{dist}}(\mathbf{x}_j^t, \mathbf{x}_i^t), \end{aligned}$$

where the features are defined as in Eqn. (17) and Eqn. (18), but with the input arguments switched, because the merging vehicle should stay in front of the yielding one.

Apart from the edge rewards, all the agents share the same node reward function. The following one is adopted for LC:

$$\begin{aligned} r_\theta^n(\mathbf{x}_j^t, \mathbf{a}_j^t) = &- \left(1+\exp(\theta_0)\right) f_v(\mathbf{x}_j^t) \\ &- \left(1+\exp(\theta_{1:3})\right)^\intercal f_{\text{state}}(\mathbf{x}_j^t) \\ &- \left(1+\exp(\theta_{4:5})\right)^\intercal f_{\text{action}}(\mathbf{a}_j^t) \\ &- \left(1+\exp(\theta_6)\right) f_{\text{lane}}(\mathbf{x}_j^t), \end{aligned}$$

where f_{state} and f_{action} take the element-wise square of $\left[a_j^t\ \theta_j^t\ \omega_j^t\right]$ and $\left[\delta a_j^t\ \delta\omega_j^t\right]$ respectively. The feature f_v is the squared error between v_j^t and the speed limit v_{lim}. The last term f_{lane} penalizes the vehicle for staying close to the lane boundaries. For CF, we simply remove those terms that are irrelevant in 1D motion.

In all the reward functions, the parameters collected in ψ and θ are unknown during training and inferred by GRI. We take the exponents of them and add one to the results. It enforces the model to use the features when modeling the corresponding interactions.

8.3 Naturalistic Traffic Scene MDPs Specification

The node dynamics is the same as in the synthetic scene for CF. For LC, since we did not have accurate heading information, we adopted 2D point-mass model instead. Since the behavior of human drivers is much more complicated than the synthetic agents, we designed reward functions with larger model capacity using neural networks. In CF, the reward functions are defined as follows:

$$\begin{aligned} r_{\psi_1}^{e,1}\left(\mathbf{x}_i^t, \mathbf{x}_j^t\right) = &- \left(1+\exp(\psi_{1,0})\right) g_{\text{v}}^{\text{NN}}(\mathbf{x}_i^t, \mathbf{x}_j^t) \\ &- \left(1+\exp(\psi_{1,1})\right) g_{\text{s}}^{\text{NN}}(\mathbf{x}_i^t, \mathbf{x}_j^t), \\ r_\theta^n\left(\mathbf{x}_j^t, \mathbf{a}_j^t\right) = &- \left(1+\exp(\theta_0)\right) f_{\text{v}}^{\text{NN}}(\mathbf{x}_j^t) \\ &- \left(1+\exp(\theta_1)\right) f_{\text{acc}}(\mathbf{x}_j^t) \\ &- \left(1+\exp(\theta_2)\right) f_{\text{jerk}}(\mathbf{x}_j^t, \mathbf{a}_j^t), \end{aligned}$$

where the features are defined as:

$$\begin{aligned} f_{\text{v}}^{\text{NN}}(\mathbf{x}_j^t) &= \left(v_j^t - h_1(\mathbf{x}_j^t)\right)^2, \\ g_{\text{v}}^{\text{NN}}(\mathbf{x}_i^t, \mathbf{x}_j^t) &= \left(v_j^t - h_2(\mathbf{x}_i^t, \mathbf{x}_j^t)\right)^2, \\ g_{\text{s}}^{\text{NN}}(\mathbf{x}_i^t, \mathbf{x}_j^t) &= \text{ReLU}\big(h_3\left(\mathbf{x}_i^t, \mathbf{x}_j^t\right) - x_i^t + x_j^t\big)^2. \end{aligned}$$

The features f_{acc} and f_{jerk} penalize the squared magnitude of acceleration and jerk. The functions h_1, h_2 and h_3 are neural networks with ReLU output activation. The feature g_{s}^{NN} is the critical component which shapes the car-following behavior. It learns a non-negative reference headway and penalizes the following vehicle for violating it. The feature g_{v}^{NN} and f_{v}^{NN} suggest reference velocities considering interaction and merely itself respectively.

In LC, the edge reward function for car-following and the node reward function are similar to those in CF, with additional terms for lateral position, velocity and acceleration. To design the yielding reward, we define a collision point of two vehicles based on their states. We approximate the vehicles' trajectories as piecewise-linear between sequential timesteps, and compute the collision point as the

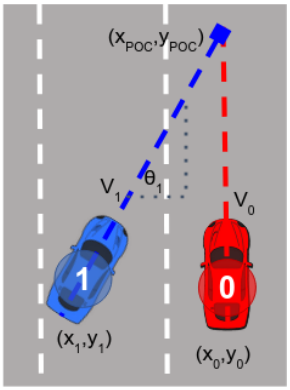

Figure 6: Collision point diagram. At every timestep, we calculate the agents' heading vectors by approximating the motion as linear. We define the intersection between these vectors as the collision point, where two agents would collide if neither of them yields.

intersection between their trajectories (Fig. 6). We threshold the point if it exceeds a hard-coded range of interest (e.g. if it is behind the vehicle or their distance is greater than certain value). Afterwards, we define the distance-to-collision (d_{poc}) as the longitudinal distance from the vehicle to the collision point, and the time-to-collision (T_{col}) as the time to reach the collision point calculated by dividing d_{poc} with the velocity of the vehicle. Then the yielding reward function is defined as follows:

$$\begin{aligned} r_{\psi_2}^{e,2}\left(\mathbf{x}_i^t, \mathbf{x}_j^t\right) = &- \left(1+\exp(\psi_{2,0})\right) g_{\text{spatial}}^{\text{NN}}(\mathbf{x}_i^t, \mathbf{x}_j^t) \\ &- \left(1+\exp(\psi_{2,1})\right) g_{\text{time}}^{\text{NN}}(\mathbf{x}_i^t, \mathbf{x}_j^t), \end{aligned}$$

where

$$\begin{aligned} g_{\text{spatial}}^{\text{NN}}(\mathbf{x}_i^t, \mathbf{x}_j^t) &= \text{ReLU}\big((x_j - x_{poc}) - h_{\text{d}_{\text{poc}}}(\mathbf{x}_i^t, \mathbf{x}_j^t)\big)^2, \\ g_{\text{time}}^{\text{NN}}(\mathbf{x}_i^t, \mathbf{x}_j^t) &= \text{ReLU}\big(h_{\text{T}_{\text{col}}}(\mathbf{x}_i^t, \mathbf{x}_j^t) - (T_{col_i} - T_{col_j})\big)^2. \end{aligned}$$

The functions $h_{\text{d}_{\text{poc}}}$ and $h_{\text{T}_{\text{col}}}$ are neural networks with ReLU output activation. The g_{spatial} term learns a spatial aspect of the yield behavior and compares the agent's distance from the estimated collision-point with the NN-learned *safe* reference within which the LC maneuver can be done. The second term g_{time} adds a temporal aspect to yield and compares a learned *safe* headway time and to the difference in time-to-collision for the two vehicles. The intuition behind is to ensure that the vehicles do not occupy the same position at the same time.

8.4 Out-of-distribution Experiment

In this section, we report an experiment that illustrates GRI's advantages because of its semantic meaningful latent space and policy model. In the synthetic scenarios, we shifted the distribution of initial states: In CF, the longitudinal distance between two vehicles was sampled from $\text{unif}(2,4)$ instead of $\text{unif}(4,8)$; In LC, the same distance was sampled from $\text{unif}(8,12)$ instead of $\text{unif}(6,8)$. Afterward, we enforced the ground-truth $\mathcal{G}_{\text{interact}}$, and ran the policy decoders to generate the trajectories for comparison with the ones generated by the expert. The results are summarized in Table 3. The GRI policy induces relatively consistent behaviors with the expert, whereas the trajectories generated by the NRI policy deviate significantly from the expert ones, especially in LC. The experiment is analogous to the case where the autonomous vehicle encounters an unfamiliar situation. The results show the advantages of GRI under such circumstances. The safety driver or passengers can override the inferred $\mathcal{G}_{\text{interact}}$ to ensure safety if the model misunderstands the scenario. Because of the semantic meaningful latent space and policy model, the desired behavior will emerge as expected. In contrast, an inexplicable model does not enable the same safety assurance because it models the interactions with a cluster of uninterpretable behaviors. Fig. 7 visualizes an exaggerated but illustrative example to demonstrate this point. We deliberately placed the leading car behind the following one at the initial timestep and enforced the same $\mathcal{G}_{\text{interact}}$ as usual. The GRI policy still prompts the car-following behavior: It slows down the vehicle until the leading one surpasses it. On the other hand, the NRI policy does not behave as $\mathcal{G}_{\text{interact}}$ suggests.

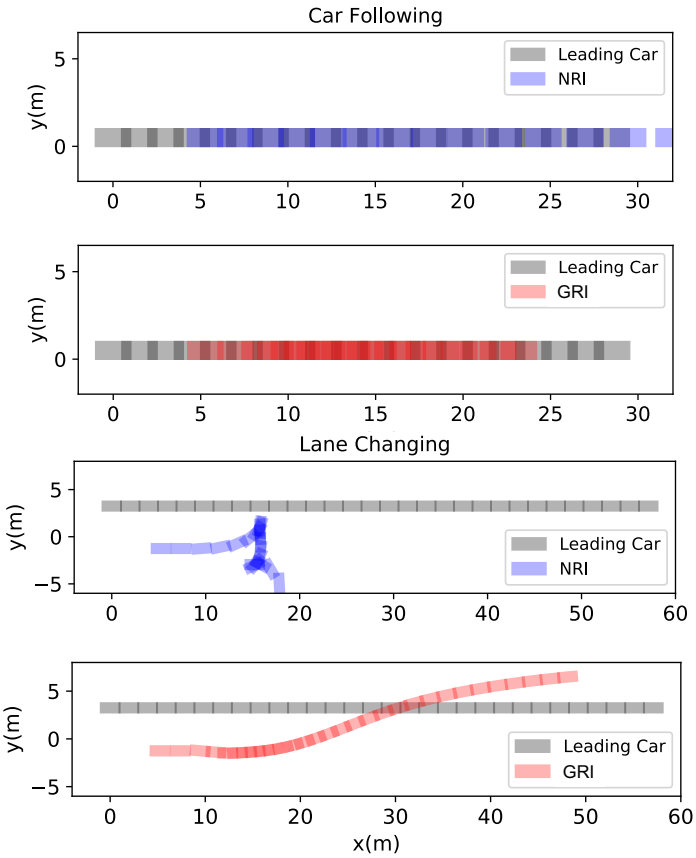

Figure 7: Examples where the leading car is placed behind the following one at the initial timestep. The GRI policy still prompts the car-following behavior, but the NRI policy does not behave as $\mathcal{G}_{\text{interact}}$ suggests.

Table 3: Performance Comparison on Shifted Synthetic Dataset

(a) Car-Following Scenario ($T = 20, \Delta t = 0.2s$)

Model	$\text{RMSE}_x(\text{m})$	$\text{RMSE}_v(\text{m/s})$
GRI (Policy)	$\mathbf{0.907 \pm 0.523}$	$\mathbf{0.727 \pm 0.243}$
NRI (Policy)	1.745 ± 0.427	0.866 ± 0.180

(b) Lane-Changing Scenario ($T = 30, \Delta t = 0.2s$)

Model	$\text{RMSE}_x(\text{m})$	$\text{RMSE}_y(\text{m})$	$\text{RMSE}_v(\text{m/s})$
GRI (Policy)	$\mathbf{0.689 \pm 0.281}$	$\mathbf{1.107 \pm 0.551}$	$\mathbf{0.310 \pm 0.083}$
NRI (Policy)	11.858 ± 2.926	1.475 ± 0.522	3.776 ± 0.753