# DIDER: Discovering Interpretable Dynamically Evolving Relations

Enna Sachdeva[1] and Chiho Choi[1]

*Abstract*— **Effective understanding of dynamically evolving multiagent interactions is crucial to capturing the underlying behavior of agents in social systems. It is usually challenging to observe these interactions directly, and therefore modeling the latent interactions is essential for realizing the complex behaviors. Recent work on Dynamic Neural Relational Inference (DNRI) captures explicit inter-agent interactions at every step. However, prediction at every step results in noisy interactions and lacks intrinsic interpretability without post-hoc inspection. Moreover, it requires access to ground truth annotations to analyze the predicted interactions, which are hard to obtain. This paper introduces DIDER, Discovering Interpretable Dynamically Evolving Relations, a generic end-to-end interaction modeling framework with intrinsic interpretability. DIDER discovers an interpretable sequence of inter-agent interactions by disentangling the task of latent interaction prediction into sub-interaction prediction and duration estimation. By imposing the consistency of a sub-interaction type over an extended time duration, the proposed framework achieves intrinsic interpretability without requiring any post-hoc inspection. We evaluate DIDER on both synthetic and real-world datasets. The experimental results demonstrate that modeling disentangled and interpretable dynamic relations improves performance on trajectory forecasting tasks.**

## I. INTRODUCTION

Real-world applications such as autonomous driving, mobile robot navigation, and air-traffic management involve multiagent interactions for joint behavior prediction and complex decision making. Modeling these interactions is crucial to understanding the underlying dynamic behavior of the agents. For instance, the future behavior (yielding or right of way) of a vehicle approaching an intersection is influenced by another approaching vehicle. However, it is challenging to model these interagent interactions, as we often do not know about the ground truth interactions between agents.

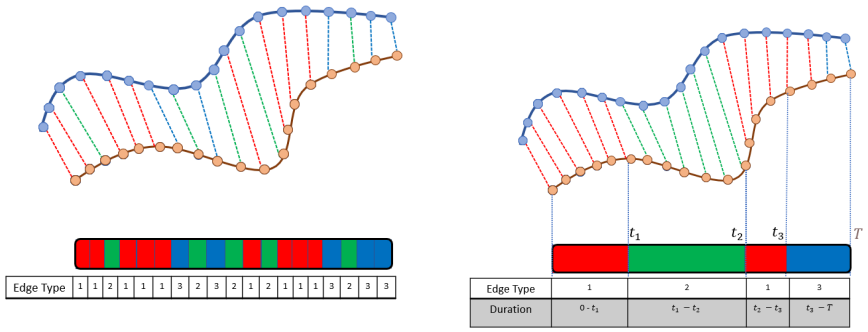
In recent years, there has been a considerable amount of work towards explicit modeling of multiagent interactions from raw trajectories[1], [2], [3], [4]. These methods leverage graph neural networks to model relational structures with multiple interaction types. It was firstly introduced in Neural Relational Inference (NRI) [1], which infers a static relational graph between multiple agents while simultaneously modeling the dynamics of the interacting system. However, most real-world social systems involve dynamically evolving multiagent interactions. To address this gap of modeling dynamic interactions, Dynamic NRI [3] and Evolvegraph [2] were proposed. These models discover unseen interactions between agents at every step to improve the performance of trajectory forecasting tasks. While per-step prediction provides dynamic interactions, it results in a noisy sequence of non-interpretable interactions. These methods usually require post-hoc analysis to interpret these noisy interactions, which could be ambiguous or falsely interpreted by humans.

In most real-world situations, agents interact with each other in a sequence of sub-interactions for an extended period to jointly execute the downstream task. This is similar to how humans usually tend to break down the long duration of task into a sequence of sub-tasks [5], [6]. For instance, in a lane-changing scenario, a new-follower vehicle (in the new lane) may be required to execute a *yielding* policy to yield to the lane-changing car, followed by a *car-following* policy by maintaining a safe distance from the leader car, to navigate safely. We argue that to discover an interpretable sequence of disentangled sub-interactions, one should account for the extended duration of sub-interaction between agents. This may be achieved by incorporating an additional module to predict the time duration of inter-agent sub-interactions. This additional time duration predictor module adds an interpretability constraint to the model and generate an intrinsically interpretable [7] sequence of sub-interactions. Additionally, these extracted sequences of disentangled sub-interactions facilitate the generation of new and diverse scenarios by combining these interactions in various ways [8],[9].

This paper introduces DIDER- Discovering Interpretable Dynamically Evolving Relations, an unsupervised learning framework for discovering interpretable dynamic multiagent interactions from observations. It leverages VAE-framework to discover interpretable dynamic temporal interactions while simultaneously learning the dynamic model of the system. We incorporate intrinsic interpretability into the model by decoupling the interaction prediction task into sub-interaction and duration predictions. The key contributions of this work are summarized as follows:

- We propose an end-to-end explicit interaction modeling framework, with intrinsic interpretability, by disentangling dynamic interaction prediction into sub-interaction prediction and duration prediction, as shown in Fig. 1.
- The proposed model uses trajectory prediction as a surrogate task for learning interpretable dynamically evolving interactions. By predicting each sub-interaction's start and end time, the model provides better interpretability of the latent interactions while improving the performance on downstream trajectory prediction task.
- The proposed model is a generic framework for modeling dynamic interactions and is flexible to be incorporated into any existing VAE-like relational inference framework to improve interactions interpretability and trajectory prediction performance.

[1]Honda Research Institute, CA, USA
esachdeva@honda-ri.com, cchoi@honda-ri.com




(a) Interactions Prediction by DNRI

(b) Interactions prediction by DIDER

Fig. 1: Overview of the interaction prediction of DIDER as compared to DNRI. a) DNRI predicts interaction at every time step t, which results in noisy and non-interpretable interactions. b) DIDER disentangles the task of interaction prediction into sub-interaction prediction and duration estimation, which provides interpretable dynamic interactions. By learning duration along with interaction/edge type, DIDER guides the model to learn consistent sub-interactions for extended duration of time.

- We evaluate the performance of the proposed framework on both simulated and realistic trajectory forecasting tasks and visualize the predicted interactions to elucidate the interpretability of the predicted interactions.

We use the terms *relations*, *interactions*, and *edges* interchangeably in this paper.

## II. Related Work

### A. Interpretable Motion Prediction Frameworks

In recent years, interpretability has been considered an important factor in developing motion prediction frameworks. Recently, Brewitt et al. introduced GRIT [10], a goal recognition framework with interpretable decision trees on vehicle trajectory dataset. The encoding of discrete latent space in CVAE framework in motion prediction frameworks like Trajectron [11] and Trajectron ++ [12] aids interpretability of the learned latent space. Parth et al. [13] combines rule-based models with neural network-based models to predict interpretable high-level intents as well as scene-specific residuals. These methods encode intrinsic interpretability into the model to make black-box models more transparent with or without domain knowledge. However, these methods do not explicitly model the interactions between agents for the motion prediction task.

### B. Interpretable Multiagent Interactions

Several works have realized interaction modeling and relational reasoning using Graph Neural Networks [1], [3], [4], [2], [14], [15], [16]. They introduce nodes to represent the interactive agents and edges to represent their interaction types. By explicitly modeling the dynamic interaction graphs, they learn the dynamic model of the system. While these methods model the agents' underlying static and dynamic interactions, their interpretability was not explored until recently. Recent work on Grounded Relational Inference (GRI) [17] sets as a stepping stone towards generating interpretable and grounded inter-agent interaction graphs. GRI learns the reward functions for various semantically meaningful interactions between agents while simultaneously modeling system dynamics by formulating the problem as Inverse Reinforcement Learning. Another very recent work by Lingfeng and Chen et al. [18] leverages pseudo labels to enforce the model to learn interpretable interactions. Further, GRIN [19] disentangles inter-agent interactions from agents' intentions for better interpretability over inferred *static* interactions. [20] partially addresses the interpretability of predicted interactions using a supervised learning framework by generating a simple labeling function to annotate the ground truth interactions between agents. While all these methods primarily focus on improving the interpretability of interactions using domain knowledge, they assume interactions to be static across time. We are the first to address discovering disentangled and interpretable sequences of dynamic interactions from multiagent observations to the best of our knowledge.

### C. Trajectory Segmentation

Trajectory segmentation has been well studied in the literature to facilitate learning localized control policies and combinatorial generalization to unseen scenarios. There exist several unsupervised learning frameworks to decompose trajectories into various skills [21], [22], [23]. These methods aim to decompose trajectories into a sequence of subgoals or skills. TSC-DL [24] segments trajectories into locally-similar contiguous sections but requires prior knowledge of the number of segments. A similar approach CompILE [25], segments a trajectory in an unsupervised framework but lacks interpretability in latent space and learning length-independent skills. Recent work on SKID [26] operates on trajectories with a varying and unknown number of skills per trajectory. Most of the work in this direction addresses single-agent trajectory segmentation to various subgoals or skills. However, none of these methods aim to segment latent multiagent interactions in an unsupervised manner.

## III. Background

### A. Dynamic Neural Relational Inference

Dynamic Neural Relational Inference (DNRI) [3] models the dynamic evolving relation types between agents

by predicting $z^t_{i,j}$ at each time step. It simultaneously learns the dynamic model of the interacting system. DNRI formulates this problem using a Conditional Variational Autoencoder (CVAE) framework. Consider a set of $N$ agents with their trajectories (of duration $T$) denoted as: $x_1^{1:T}, x_2^{1:T}, ....., x_N^{1:T}$, it predicts the trajectories using relational embeddings. The interactions between entities are represented by $z^t_{i,j} \epsilon \{1, 2, ..., e\}$ for every pair of entities $(i, j)$ at time step $t$, where $e$ denotes the number of possible interaction types between entities.

LSTMs are used to model the dynamic prior $p_\phi(z|x)$ and encoder $q_\phi(z|x)$. The encoder at each timestep is conditioned on a full trajectory, while the prior is conditioned on the observation and relation prediction from previous steps:

$$p_\phi(z|x) := \prod_{t=1}^{T} p_\phi(z^t|x^{1:t}, z^{1:t-1}),$$
$$q_\phi(z|x) := \prod_{t=1}^{T} q_\phi(z^t|x^{1:T}). \tag{1}$$

The decoder then predicts the future states of the entities $x$. The decoder is the formulation conditioning on the dynamic $z_t$ sampled from the encoder at every time step $t$.

$$p_\theta(x|z) := \prod_{t=1}^{T} p_\theta(x^{t+1}|x^{1:t}, z^{1:t}). \tag{2}$$

$\theta$, $\phi$ are the trainable parameters of probability distributions, which are optimized by maximizing the following evidence lower bound (ELBO).

$$\mathcal{L}(\phi, \theta) = \mathbb{E}_{q_\phi}(z|x)[\log p_\theta(x|z)] - KL[q_\phi(z|x)||p_\phi(z|x)] \tag{3}$$

### B. Trajectory Segmentation using SKID

Our method is loosely inspired by SKID [26], an unsupervised framework to segment the trajectories into re-occurring patterns (skills) from unlabelled demonstrations. SKID frames the problem using VAEs, with latent space $z = \{z_d, z_s\}$ describing the properties of a segment, where $z_d$ and $z_s$ represents duration of the skill and skill type.

SKID models the skill duration $z_d$ with a Gaussian distribution given a prior i.e. $z_d \sim \mathcal{N}(\mu_d, \sigma_d^2)$. Each skill duration $z_d$ is obtained using the remainder of the trajectory, i.e., the part of the trajectory that has not been explained by all previous $z_d$. This extracted sub-trajectory $\tau$ is then used for learning the skill type $z_s$. Assuming that a trajectory consists of $N$ segments, this iterative process is repeated $N$ times until it reaches the last time step of the trajectory. The learning is done by jointly optimizing the generative model and the inference network by maximizing the evidence lower bound. SKID utilizes full trajectories for learning skills and duration, making it suitable for offline settings.

## IV. MODEL DESIGN

Our objective is to discover an interpretable sequence of sub-interactions among agents from their observations in an online setting. We achieve this by disentangling the task of predicting dynamic interactions into two parts -sub-interaction prediction and duration estimation, both of which are unobserved. Thus, we model this problem using the Conditional Variational Autoencoders (CVAE) framework with two latent variables. The discrete and continuous latent variables $z_e$ and $z_d$ represent agents' interaction (edge) type and the corresponding time duration. Our model learns an unknown number of varying lengths of sub-interactions from the observations. The formulation involves the simultaneous prediction of time duration and interaction type between agents. This requires a novel encoder and prior models, as motivated by previous work on sequential segmentation modeling [3], [26], [27]. The model is trained by maximizing ELBO. The architecture of DIDER is shown in Fig. 2.

In Dynamic NRI, the prior and encoder predict interaction at every time step $t$, by capturing past and past + future instances of trajectories, respectively. In contrast, DIDER firstly predicts the time duration of an interaction type with *Duration encoder* using the last segment of past trajectories. The time duration sampled from the duration prior is then used by Edge prior and Edge encoder to learn an interaction type corresponding to the specific segment of the trajectory.

### A. Prior and Encoder

To model evolving sequence of sub-interactions, we learn prior probabilities on the edge duration $z_d$ and edge types $z_e$ conditioned on the past. Similar to NRI and DNRI, the input at each time step is passed through a Graph neural network to produce edge embeddings, as follows:

$$h^t_{i,1} = f_{emb}(x^t_i), \tag{4}$$

$$v \to e : h^t_{(i,j),1} = f^1_e([h^t_{i,1}, h^t_{j,1}]), \tag{5}$$

$$e \to v : h^t_{(i,j),2} = f^1_v(\Sigma_{i \neq j} h^t_{(i,j),1}), \tag{6}$$

$$v \to e : h^t_{(i,j),emb} = f^2_{emb}([h^t_{i,2}, h^t_{j,2}]). \tag{7}$$

This architecture implements a form of neural message passing in a graph where vertices $v$ represents entities $i$, and edges $e$ represents the relations between entities pairs $(i, j)$. $f_{emb}$, $f^1_e$ and $f^1_v$ are MLPs. The embeddings $h^t_{(i,j),1}$ only depends on $x_i$ and $x_j$, while $h^t_{(i,j),2}$ uses information from the whole graph. We refer to [1], [3] for details. This neural message passing architecture outputs a per time step edge embedding $h^t_{(i,j),emb}$, which is fed into the forward and reverse LSTM networks to model the probabilities over edge duration and edge types.

$$h^t_{(i,j),prior} = LSTM_{forward}(h^t_{(i,j),emb}, h^{t-1}_{(i,j),prior}), \tag{8}$$

$$h^t_{(i,j),reverse} = LSTM_{reverse}(h^t_{(i,j),emb}, h^{t+1}_{(i,j),reverse}). \tag{9}$$

*1) Duration Encoder:* The edge duration $z^k_d$ is modeled as a continuous latent variable, and determines the duration ($d_k$) of an interaction type for $k^{th}$ segment of an edge. With the initial burn-in period (observation period with groundtruth trajectory) of $T_{obs}$, it models the probability distribution of the duration ($d_1$) of first segment ($k = 1$) of an interaction as $p_{\phi_d}(z^1_d|x^{1:T_{obs}})$, where $t_1 = 0$, $d_0 = T_{obs}$, and $t_k$ and

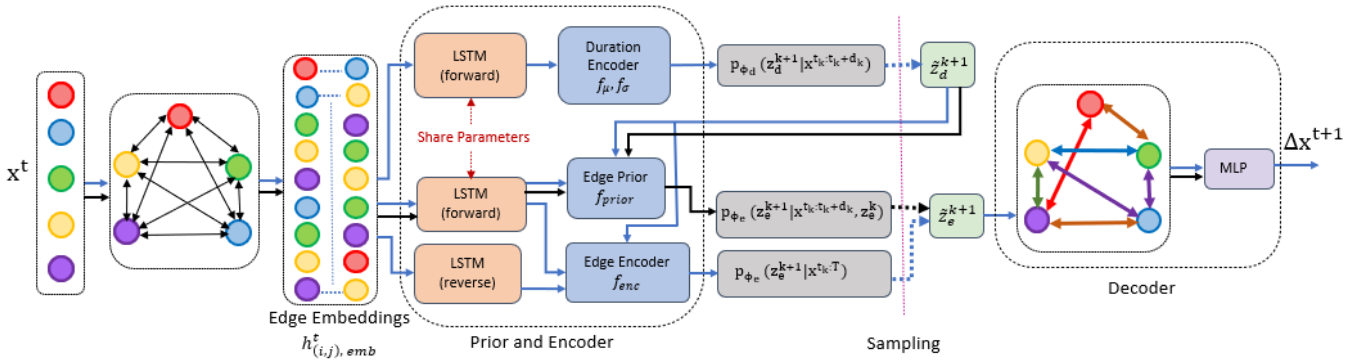



Fig. 2: **Architecture of DIDER:** The input trajectories are fed to a fully-connected GNN to produce edge embeddings at every time step. These are aggregated using forward and reverse LSTM to encode the past and future trajectories. The duration prior and the edge prior are computed as a function of the past trajectory, and the edge encoding is computed as a function of both past and future. The edge types are sampled from the edge encoder during training and edge prior during inference. The decoder predicts the state of the entities at the next time step. The path shown by solid and dotted blue and black arrows corresponds to training and evaluation.

$t_k + d_k$ represent the start time and the end time of $k^{th}$ segment. $T_{remaining}$ is the duration of the remainder of the trajectory, which has not been utilized for duration estimation of previous sub-interactions. It is represented as $T - t_k - d_k$.

*a) Parametization of Continuous latent Variables:* We parametrize $p_{\phi_d}(z_d^k|x)$ by a Gaussian distribution, i.e $p_{\phi_d}(z_d^k|x) = \mathcal{N}(\mu_k, \sigma_k^2)$, where $\mu_k$, and $\sigma_k^2$ are parameterized by neural networks. Let the prior be a Gaussian distribution with $p(z_d) = \mathcal{N}(\mu_0, \sigma_0^2)$. We use the reparametrization trick for the Gaussian distributed $z_d$ to sample the time duration factor $z_d^k = \mu_k + \sigma_k \epsilon$, where $\epsilon$ is an auxiliary noise variable $\epsilon \sim \mathcal{N}(0, 1)$. Then the time duration of the $k^{th}$ segment is estimated as:

$$\begin{aligned} \mu_{k+1} &= \tanh(f_\mu(h^{t_k+d_k}_{(i,j),prior})), \\ \sigma_{k+1} &= \text{sigmoid}(f_\sigma(h^{t_k+d_k}_{(i,j),prior})), \\ p_{\phi_d}(z_d^{k+1}|x^{t_k:t_k+d_k}) &= \mathcal{N}(\mu_{k+1}, \sigma^2_{k+1}), \\ z_d^{k+1} &= \mu_{k+1} + \sigma_{k+1}\epsilon, \\ d_{k+1} &= z_d^{k+1} \cdot T_{remaining}, \end{aligned} \tag{10}$$

where $f_\mu$, $f_\sigma$ are realized using MLPs.

*2) Edge Prior and Edge Encoder:* The edge prior probabilities over edge types are modeled in an autoregressive manner. For each segment duration $d_{k+1}$ sampled from the duration encoder, the prior probabilities over edge types are conditioned on the relation type predicted in the previous segment ($z_e^k$) as well as the sequence of observations in that segment, as following-

$$p_{\phi_e}(z^{k+1}_{(i,j)}|x^{t_k:t_k+d_k}, z_e^k) := \text{softmax}(f_{prior}(h^{t_k+d_k}_{(i,j),prior})), \tag{11}$$

$$p_{\phi_e}(z_e|x) := \prod_{k=1}^{K} p_{\phi_e}(z_e^{k+1}|x^{t_k:t_k+d_k}, z_e^k). \tag{12}$$

We encode the dependence of previous $z_e^k$ to next $z_e^{k+1}$ in the hidden state $h^{t_k+d_k}_{(i,j),prior}$.

During training, the encoder computes the approximate distribution of edge types for every segment by using the information of the whole sequence (past segment and future). The true posterior over the latent space is a function of the future states of the observed variable [28]. Therefore, similar to DNRI, we use a reverse LSTM to capture future states of the sequence. The relational embedding $h^t_{(i,j),emb}$ is passed through a reverse LSTM and then concatenated with the results of forward LSTM to estimate posterior as follows:

$$q_{\phi_e}(z^{k+1}_{(i,j)}|x) := \text{softmax}(f_{enc}([h^{t_k+d_k}_{(i,j),reverse}, h^{t_k+d_k}_{(i,j),prior}]). \tag{13}$$

The encoder approximates distribution of interactions for each segment as follows:

$$q_{\phi_e}(z_e|x) := \prod_{k=1}^{K} p_{\phi_e}(z_e^{k+1}|x^{t_k:T}). \tag{14}$$

The encoder and prior models share the parameters, so we use $\phi_e$ to refer to the parameters of both of these models.

*a) Parameterization of Discrete latent Variables:* We parameterize discrete categorical distribution with a continuous approximation function (*i.e.*, softmax) to obtain probability distribution over each edge type [29], [30]. The sampling is done via reparametrization by first sampling a vector $g$ of independent and identically distributed samples drawn from Gumbel (0, 1) and computing the following,

$$z_{e_{(i,j)}} = \text{softmax}(h(i,j) + g/\tau), \tag{15}$$

where $\tau$ is the softmax temperature which controls the sample smoothness.

*3) Generic framework of Edge prior and Edge encoder:* We investigate the general formulation of edge prior and encoder modules, discussed in Eqn. 12 and 14. For a particular case where the number of segments $K$ is hardcoded as equal to time horizon $T$, corresponding to $d_k = 1$, the formulation

of edge prior and encoder is factorized as:

$$p_{\phi_e}(z_e|x) := \prod_{t=1}^{T} p_{\phi_e}(z_e^{t+1}|x^{1:t}, z_e^{1:t-1}), \tag{16}$$

$$q_{\phi_e}(z_e|x) := \prod_{t=1}^{T} p_{\phi_e}(z_e^{t+1}|x). \tag{17}$$

This specific case corresponds to the encoder and prior formulation of DNRI. Therefore, DIDER provides a generic framework with a time duration encoder, which provides additional flexibility for improving the interpretability of models which follows a VAE-like framework, such as DNRI.

### B. Decoder

Decoder is used to predict the trajectory given the observations of the entities, and the sampled relation types at every time step $t$. Similar to NRI and DNRI, we use an autoregressive model, which factorizes as follows:

$$p_\theta(x|z_e) := \prod_{t=1}^{T} p_\theta(x^{t+1}|x^{1:t}, z_e^{1:t}). \tag{18}$$

We always provide the ground truth states to the decoder during training for the same reason suggested in DNRI.

### C. Training and Inference

We jointly train the generative and inference model parameters $\theta$, $\phi_d$ and $\phi_e$, by maximizing the ELBO:

$$\begin{aligned}\mathcal{L}(\theta, \phi_d, \phi_e) = \mathbb{E}_{q_\phi}(z_e|x)[log p_{\theta_d}(x|z_e)] \\ -\beta_d(KL(q_{\phi_d}(z_d|x)||p(z_d)) - C_d) \\ -\beta_e(KL(q_{\phi_e}(z_e|x)||p_{\phi_e}(z_e|x)) - C_e).\end{aligned} \tag{19}$$

where $\beta_d$, $\beta_e$ are constant scaling factors, and $C_d$, $C_e$ are the information capacity terms. The first term aims at reconstructing the data, while the KL divergence forces the model to stay close to the given prior. Further, to enforce disentanglement, we use $\beta$-VAE formulation, as introduced in [31]. Similar to SKID, we add capacity terms to the ELBO as proposed in [32], [33]. Since DIDER discovers a sequence of sub-interactions for each edge individually, it adds a complexity of $O(n^2T)$, where $n$ is the number of agents.

## V. Experiments

We demonstrate the performance of DIDER on the synthetic dataset, basketball dataset, and inD dataset [34]. The synthetic data was generated similar to as generated in DNRI [3]. We mainly compare the performance with DNRI and SKID, which are mostly related to us. Originally, SKID is not designed for segmenting interactions from multiagent observations, but we are adopting their framework to learn to segment interactions from multiagent interactions. Further, SKID uses complete trajectory information for predicting the segment duration, making it suitable for offline setting. As we consider trajectory prediction as a surrogate task, we aim to perform disentangled interaction segmentation in an online setting. Therefore, we do not assume access to future observations during evaluation. We include an ablation study, where we use SKID-based *Duration Encoder* for predicting the interaction duration in DIDER by providing complete trajectory information. We use the following labels for our proposed framework, baseline method, and ablation study-

- DNRI: Dynamic Neural Relational Inference [3]
- DIDER (Ours): Our proposed framework
- DIDER (with SKID): Our framework, with *Duration Encoder* formulation adapted from SKID [26]

For all experiments, the first edge type is hardcoded to represent *No-Interaction*. We conduct four statistically independent runs with random seeds from 1 to 4 and report the mean and standard deviation. The plots present the average performance with the shaded region showing a 95% confidence interval. To demonstrate the interpretability of discovered interactions, we either compute edge accuracy, visualize the interaction transitions across time, or both.

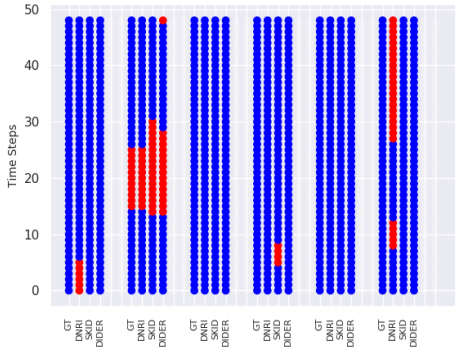



Fig. 3: Visualization of latent interactions inferred by DNRI, DIDER (with SKID), and DIDER (Ours) on synthetic physics simulations data with 3 particles. There exists 6 interactions for 3 particles' environment. Blue edge represents *Non-Interacting* edge type, and red represents *Interacting* edge type. GT denotes GroundTruth edges.

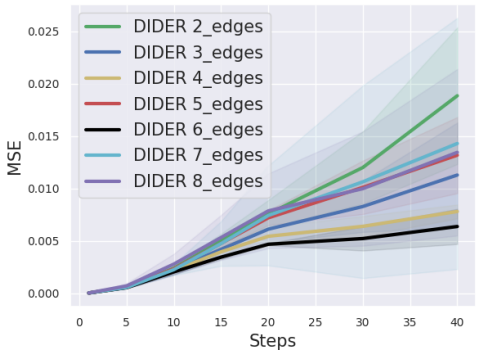



Fig. 4: Ablation study on DIDER with inD data, to determine the optimal number of edge types based on MSE loss of downstream trajectory prediction task

### A. Synthetic Physics Simulation

To evaluate the performance of DIDER and showcase its ability to discover dynamic relations, we use physical simulation systems, *i.e.*, moving particles with dynamic relations between them [3]. As the dataset is synthetically

TABLE I: MSE and Edge Accuracy of Synthetic Data with 3 Particles

| Models | MSE for various prediction horizons | | | Edge accuracy (in %) | |
|---|---|---|---|---|---|
| | 1 | 15 | 25 | *No Interaction* edge | *Interaction* edge |
| DNRI | $(8.16 \pm 1.43) \times 10^{-7}$ | $(3.19 \pm 1.79) \times 10^{-3}$ | $(4.18 \pm 2.25) \times 10^{-3}$ | $85.76 \pm 0.40$ | $58 \pm 9.80$ |
| DIDER (with SKID) | $(7.62 \pm 2.45) \times 10^{-7}$ | $(1.63 \pm 0.38) \times 10^{-3}$ | $(2.20 \pm 0.50) \times 10^{-3}$ | $87.90 \pm 1.10$ | $\mathbf{94.08 \pm 0.50}$ |
| DIDER (Ours) | $\mathbf{(4.71 \pm 0.80) \times 10^{-7}}$ | $\mathbf{(1.13 \pm 0.17) \times 10^{-3}}$ | $\mathbf{(1.51 \pm 0.17) \times 10^{-3}}$ | $\mathbf{87.84 \pm 0.60}$ | $92.50 \pm 1.08$ |

TABLE II: MSE of Trajectory Prediction on Basketball Dataset with 5 agents

| Models | MSE for various prediction horizons | | |
|---|---|---|---|
| | 1 | 5 | 9 |
| DNRI | $(7.90 \pm 0.58) \times 10^{-5}$ | $(5.98 \pm 0.12) \times 10^{-4}$ | $(2.42 \pm 0.10) \times 10^{-3}$ |
| DIDER (with SKID) | $\mathbf{(7.35 \pm 0.07) \times 10^{-5}}$ | $(5.79 \pm 0.20) \times 10^{-4}$ | $(2.38 \pm 0.06) \times 10^{-3}$ |
| DIDER (Ours) | $(7.44 \pm 0.12) \times 10^{-5}$ | $\mathbf{(5.65 \pm 0.08) \times 10^{-4}}$ | $\mathbf{(2.34 \pm 0.03) \times 10^{-3}}$ |

TABLE III: MSE and Consistency Analysis of inD dataset evaluated on 50 timestep

| Models | MSE for various prediction horizons | | | Consistency Check (max % samples/edge type) | |
|---|---|---|---|---|---|
| | 1 | 20 | 40 | Parked Vehicles | Non-Intersecting trajectories |
| DNRI | $(3.40 \pm 0.43) \times 10^{-5}$ | $(8.5 \pm 1.60) \times 10^{-3}$ | $(32.20 \pm 8.40) \times 10^{-3}$ | $63.10 \pm 2.20$/1 | $46.31 \pm 1.53$/1 |
| DIDER (with SKID) | $\mathbf{(1.54 \pm 0.06) \times 10^{-5}}$ | $(4.50 \pm 0.20) \times 10^{-3}$ | $(5.35 \pm 1.25) \times 10^{-3}$ | $96.22 \pm 0.52$/4 | $63.24 \pm 0.23$/1 |
| DIDER (Ours) | $(1.63 \pm 0.08) \times 10^{-5}$ | $\mathbf{(4.49 \pm 0.28) \times 10^{-3}}$ | $\mathbf{(5.27 \pm 1.64) \times 10^{-3}}$ | $\mathbf{99.89 \pm 0.41}$/3 | $\mathbf{64.86 \pm 0.24}$/1 |

TABLE IV: MSE of inD dataset evaluated for 200 timesteps

| Models | MSE for various prediction horizons | | | |
|---|---|---|---|---|
| | 1 | 40 | 120 | 190 |
| DNRI | $(5.67 \pm 0.42) \times 10^{-5}$ | $(7.70 \pm 2.90) \times 10^{-2}$ | $2.37 \pm 1.52$ | $17.87 \pm 11.76$ |
| DIDER (with SKID) | $(3.28 \pm 1.49) \times 10^{-5}$ | $(2.49 \pm 0.70) \times 10^{-2}$ | $\mathbf{0.34 \pm 0.26}$ | $3.97 \pm 5.58$ |
| DIDER (Ours) | $\mathbf{(2.37 \pm 0.11) \times 10^{-5}}$ | $\mathbf{(2.38 \pm 0.50) \times 10^{-2}}$ | $0.35 \pm 0.33$ | $\mathbf{1.96 \pm 2.18}$ |

generated, the ground truth interactions are known in prior as: *Interaction* and *No-Interaction*. The dataset consists of three particles: where two particles move with constant velocity, and the third is initialized with a random velocity. The third one is pushed away by other particles whenever the distance separating them is less than 1, and its edge type changes from *No-Interaction* to *Interaction*. We generated 40k samples with the trajectory length of 50 steps.

During the evaluation, we provide the ground truth position and velocity corresponding to the first 5 steps, and the models predict the remainder of the trajectory. Since we have access to the ground truth interaction type for this dataset, we compare the edge accuracy corresponding to both the edge types. Our findings are summarized in Table I. Results demonstrate that DIDER (Ours) outperforms DNRI and DIDER (with SKID) in MSE and accuracy. We further show that DIDER (Ours) shows comparable performance with DIDER (with SKID) for *Interaction* edge type accuracy.

We also provide the visualization of 6 edges corresponding to 3 particle simulations data predicted by all methods in Fig 3. DIDER (with SKID) and DIDER (Ours) are very close to ground truth edge types compared to DNRI. It also shows that the DIDER can predict interactions having only one edge type as *Non-Interacting* type across time.

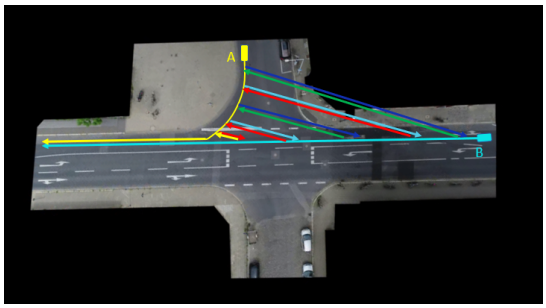
### *B. BasketBall Dataset*

The Basketball dataset [35] consists of trajectories (positions and velocities) of 5 players. The data is preprocessed into 49 frames which span approximately 8 seconds of play [3]. We train the model with initial 40 frames and are tasked to predict the remaining trajectories. The model uses two edge types as *No-Interacting* and *Interacting*. The results are summarized in Table II. DIDER (Ours) and DIDER (with SKID) outperform DNRI. It demonstrates the efficacy of our proposed disentangled framework against the step-wise counterpart. Further, DIDER (Ours) outperforms DIDER (with SKID) for longer horizon predictions, i.e., for 5 steps and 9 steps. We do not have ground truth interactions and are unaware of human-defined semantics of interaction types for the basketball dataset, unlike the road user traffic dataset, where the semantics of interactions are typically defined as yielding, following, passing-by, etc. [9]. Therefore, we only compare the performance using MSE.

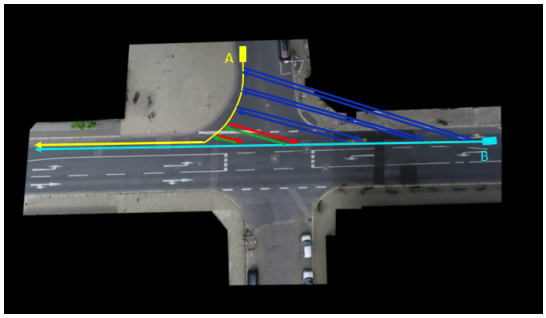
### *C. inD Dataset*

inD (Intersection Drone dataset) is a naturalistic German intersections-based road user trajectory dataset. This dataset is suitable for our problem setting, as it incorporates various temporal dynamic interactions between agents, such as *car-following*, *yielding*, *passing-by*, *cutting-in*, *lane-changing* etc. However, per-step annotation of dynamic interactions between agents is a challenging and time-consuming task and may introduce annotator's subjectivity and bias. In this experiment, we aim to discover the latent interpretable sequence of dynamic interactions from agents' observations, also referred to as traffic-primitives in [9].

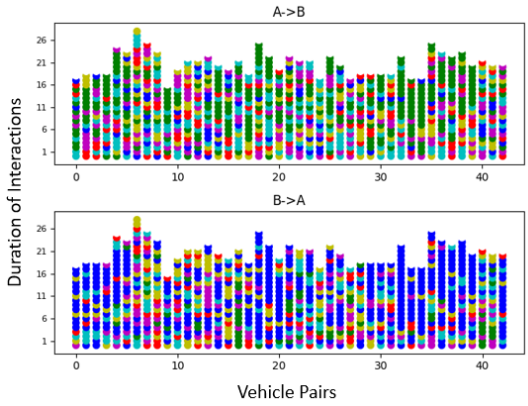
The dataset consists of 33 recordings, and we use a split of 19, 7, and 7 for the train, validation, and test dataset,



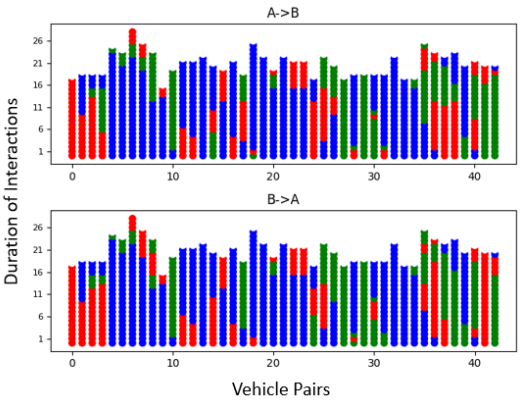

(a) Noisy per-step interaction prediction by DNRI




(b) Segments of sub-interactions prediction by DIDER




(c) Visualization of yielding trajectories with DNRI




(d) Visualization of yielding trajectories with DIDER

Fig. 5: Qualitative Results on inD dataset: (a, b) Evolution of dynamic interactions between 2 cars approaching an intersection, where yellow car (A) yields for the straight going cyan car (B). The segments of interactions from $A \rightarrow B$ and $B \rightarrow A$ discovered by DIDER is shown by different colors. Based on these segments of interactions, we may infer that red represents *yielding* interaction, and green represents *passing-by* interaction, while blue is already hardcoded as *no-interaction*. (c, d) Interactions segments corresponding to different pairs of cars involved in similar intersection at different times.

respectively, as used in DNRI [3]. During testing, we divide the trajectories into sequences of 50 steps. Contrary to the previous dataset we used in this paper; the inD dataset has a varying number of agents at every time step. Therefore, for each agent present in the sequence, we provide the model with its ground truth position and velocity for the first 5 time steps, and the model forecasts the remainder of the trajectory.

As we do not have ground truth interactions for this dataset, we cannot directly measure the accuracy of each edge type, to quantify the interpretability. Moreover, it is challenging to determine the number of edge types $(e)$ for this dataset. Therefore, we conduct an ablation study with DIDER using various edge types, from 2 to 8 and select the one with the least MSE for the downstream trajectory prediction task. Fig. 4 shows the results of the ablation study with different edge types, and we choose $e = 6$ for all methods. The comparison of prediction task results with different methods is shown in Table III. We further evaluate the performance of these methods on trajectories with longer sequences to highlight the difference in the performance of DIDER (with SKID) and DIDER (Ours). As the SKID-based framework uses an entire future sequence of trajectories to estimate the time duration of the current edge type, it adds noise into the estimation, which results in an inaccurate prediction of segment duration as sequence length increases. To demonstrate this, we train the models on inD dataset with longer sequences, where we divide trajectories into sequences of 200 steps and provide the ground truth for the first 5 steps. Table IV shows that DIDER (Ours) outperforms DIDER (with SKID) for longer sequences.

We further investigate the interpretability of these models on different traffic scenarios extracted from the dataset. For a sanity check, we firstly perform consistency analysis on the predicted interaction types between non-stationary vehicles, which seem to have no interactions based on our understanding of the scenarios. We extract these pairs of vehicles based on their location at the intersection, if they are moving in opposite directions lanes and their trajectories do not intersect at any time. The results show that DNRI, DIDER (with SKID) and DIDER (Ours) cluster $46.31\%$, $63.24\%$ and $64.86\%$, respectively, of such interactions under edge type 1, which is hard-coded to represent *No-interaction*. Similar analysis of interactions between parked cars in the dataset shows that DNRI, DIDER (with SKID), and DIDER (Ours) cluster a maximum number of such interactions under edge types 1, 4, and 3, respectively, with their percentage as $63.10\%$, $96.22\%$, and $99.89\%$, though we don't have the semantic meaning of these edge types.

We further visualize the interactions corresponding to pair of cars, where one vehicle yields for the other vehicle in the scene, as shown in Fig. 5a. The distribution of edge types shows that DIDER predicts a sequence of sub-interactions that are consistent across different pairs of cars, as shown in Fig. 5d, while DNRI predicts noisy interactions, as shown in Fig. 5c. The difference in the sequence of sub-interactions across various pairs of cars is due to the different times they approach the intersection.

## VI. CONCLUSION

This paper introduces DIDER, a generic, intrinsically interpretable, and unsupervised framework that learns disentangled and interpretable inter-agent relations from unlabeled raw observations. It targets a class of tasks where agents interact with each other in a sequence of temporal sub-interactions. DIDER achieves this by using discrete and continuous latent space to disentangle dynamic interactions prediction into sub-interaction and duration prediction. We demonstrate that DIDER achieves interpretable relations, as well as better trajectory prediction performance, as compared to models that predict interactions at every time step.

An interesting future direction is to investigate the problem with a more complex autonomous driving dataset with infrastructure, context, and heterogeneous agents. Further, we would study how the learned causal graphs can provide human interpretable explanations for predictions.

## REFERENCES


[1] T. Kipf, E. Fetaya, K.-C. Wang, M. Welling, and R. Zemel, "Neural relational inference for interacting systems," in *International Conference on Machine Learning*. PMLR, 2018, pp. 2688–2697.

[2] J. Li, F. Yang, M. Tomizuka, and C. Choi, "Evolvegraph: Multi-agent trajectory prediction with dynamic relational reasoning," *Advances in neural information processing systems*, 2020.

[3] C. Graber and A. Schwing, "Dynamic neural relational inference for forecasting trajectories," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020.

[4] D. Gong, F. Z. Zhang, J. Q. Shi, and A. van den Hengel, "Memory-augmented dynamic neural relational inference," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[5] S. Enna, S. Khadka, S. Majumdar, and K. Tumer, "Dynamic skill selection for learning joint actions," in *Proceedings of the International Conference on Autonomous Agents and MultiAgent Systems*, 2021.

[6] E. Sachdeva, S. Khadka, S. Majumdar, and K. Tumer, "Maedys: multiagent evolution via dynamic skill selection," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2021.

[7] M. Li, S. Chen, Y. Shen, G. Liu, I. W. Tsang, and Y. Zhang, "Online multi-agent forecasting with interpretable collaborative graph neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[8] W. Wang and D. Zhao, "Extracting traffic primitives directly from naturalistically logged data for self-driving applications," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 1223–1229, 2018.

[9] W. Zhang, W. Wang, J. Zhu, and D. Zhao, "Multi-vehicle interaction scenarios generation with interpretable traffic primitives and gaussian process regression," in *IEEE Intelligent Vehicles Symposium*, 2020.

[10] C. Brewitt, B. Gyevnar, S. Garcin, and S. V. Albrecht, "Grit: Fast, interpretable, and verifiable goal recognition with learned decision trees for autonomous driving," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021.

[11] B. Ivanovic and M. Pavone, "The trajectron: Probabilistic multi-agent trajectory modeling with dynamic spatiotemporal graphs," in *IEEE/CVF International Conference on Computer Vision*, 2019.

[12] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone, "Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data," in *European Conference on Computer Vision*, 2020.

[13] P. Kothari, B. Sifringer, and A. Alahi, "Interpretable social anchors for human trajectory forecasting in crowds," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2021.

[14] F. Alet, E. Weng, T. Lozano-Pérez, and L. P. Kaelbling, "Neural relational inference with fast modular meta-learning," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[15] R. Xiao, M. K. Singh, and R. Yu, "Dynamic relational inference in multi-agent trajectories," *arXiv preprint arXiv:2007.13524*, 2020.

[16] S. Löwe, D. Madras, R. Zemel, and M. Welling, "Amortized causal discovery: Learning to infer causal graphs from time-series data," *arXiv preprint arXiv:2006.10833*, 2020.

[17] C. Tang, N. Srishankar, S. Martin, and M. Tomizuka, "Grounded relational inference: domain knowledge driven explainable autonomous driving," *arXiv preprint arXiv:2102.11905*, 2021.

[18] L. Sun, C. Tang, Y. Niu, E. Sachdeva, C. Cho, T. Misu, M. Tomizuka, and W. Zhan, "Domain knowledge driven pseudo labels for interpretable goal-conditioned interactive trajectory prediction," *arXiv preprint arXiv:2203.15112*, 2022.

[19] L. Li, J. Yao, L. Wenliang, T. He, T. Xiao, J. Yan, D. Wipf, and Z. Zhang, "Grin: Generative relation and intention network for multi-agent trajectory prediction," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[20] 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.

[21] T. Shankar and A. Gupta, "Learning robot skills with temporal variational inference," in *International Conference on Machine Learning*. PMLR, 2020, pp. 8624–8633.

[22] S. Krishnan, A. Garg, S. Patil, C. Lea, G. Hager, P. Abbeel, and K. Goldberg, "Transition state clustering: Unsupervised surgical trajectory segmentation for robot learning," *The International Journal of Robotics Research*, vol. 36, no. 13-14, pp. 1595–1618, 2017.

[23] F. Despinoy, D. Bouget, G. Forestier, C. Penet, N. Zemiti, P. Poignet, and P. Jannin, "Unsupervised trajectory segmentation for surgical gesture recognition in robotic training," *IEEE Transactions on Biomedical Engineering*, vol. 63, no. 6, pp. 1280–1291, 2015.

[24] A. Murali, A. Garg, S. Krishnan, F. T. Pokorny, P. Abbeel, T. Darrell, and K. Goldberg, "Tsc-dl: Unsupervised trajectory segmentation of multi-modal surgical demonstrations with deep learning," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2016.

[25] T. Kipf, Y. Li, H. Dai, V. Zambaldi, A. Sanchez-Gonzalez, E. Grefenstette, P. Kohli, and P. Battaglia, "Compile: Compositional imitation learning and execution," in *International Conference on Machine Learning*. PMLR, 2019, pp. 3418–3428.

[26] D. Tanneberg, K. Ploeger, E. Rueckert, and J. Peters, "Skid raw: Skill discovery from raw trajectories," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4696–4703, 2021.

[27] K. Gregor, G. Papamakarios, F. Besse, L. Buesing, and T. Weber, "Temporal difference variational auto-encoder," *arXiv preprint arXiv:1806.03107*, 2018.

[28] M. Fraccaro, S. K. Sønderby, U. Paquet, and O. Winther, "Sequential neural models with stochastic layers," *Advances in neural information processing systems*, vol. 29, 2016.

[29] C. J. Maddison, A. Mnih, and Y. W. Teh, "The concrete distribution: A continuous relaxation of discrete random variables," *arXiv preprint arXiv:1611.00712*, 2016.

[30] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," *arXiv preprint arXiv:1611.01144*, 2016.

[31] 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.

[32] E. Dupont, "Learning disentangled joint continuous and discrete representations," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[33] C. P. Burgess, I. Higgins, A. Pal, L. Matthey, N. Watters, G. Desjardins, and A. Lerchner, "Understanding disentangling in beta-vae," *arXiv preprint arXiv:1804.03599*, 2018.

[34] J. Bock, R. Krajewski, T. Moers, S. Runde, L. Vater, and L. Eckstein, "The ind dataset: A drone dataset of naturalistic road user trajectories at german intersections," in *IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2020, pp. 1929–1934.

[35] Y. Yue, P. Lucey, P. Carr, A. Bialkowski, and I. Matthews, "Learning fine-grained spatial models for dynamic sports play prediction," in *IEEE international conference on data mining*. IEEE, 2014.