2021 IEEE International Conference on Robotics and Automation (ICRA 2021)
May 31 - June 4, 2021, Xi'an, China

IDE-Net: Interactive Driving Event and Pattern Extraction from Human Data

Xiaosong Jia, Liting Sun, Masayoshi Tomizuka, and Wei Zhan

***Abstract*— Autonomous vehicles (AVs) need to share the road with multiple, heterogeneous road users in a variety of driving scenarios. It is overwhelming and unnecessary to carefully interact with all observed agents, and AVs need to determine *whether* and *when* to interact with each surrounding agent. In order to facilitate the design and testing of prediction and planning modules of AVs, in-depth understanding of interactive behavior is expected with proper representation, and events in behavior data need to be extracted and categorized automatically. Answers to *what* are the essential patterns of interactions are also crucial for these motivations in addition to answering *whether* and *when*. Thus, learning to extract interactive driving events and patterns from human data for tackling the *whether-when-what* tasks is of critical importance for AVs. There is, however, no clear definition and taxonomy of interactive behavior, and most of the existing works are based on either manual labelling or hand-crafted rules and features. In this paper, we propose the Interactive Driving event and pattern Extraction Network (IDE-Net), which is a deep learning framework to automatically extract interaction events and patterns directly from vehicle trajectories. In IDE-Net, we leverage the power of multi-task learning and proposed three auxiliary tasks to assist the pattern extraction in an unsupervised fashion. We also design a unique spatial-temporal block to encode the trajectory data. Experimental results on the INTERACTION dataset verified the effectiveness of such designs in terms of better generalizability and effective pattern extraction. We find three interpretable patterns of interactions, bringing insights for driver behavior representation, modeling and comprehension. Both objective and subjective evaluation metrics are adopted in our analysis of the learned patterns.**

I. Introduction

A. Motivation

In various driving scenarios, autonomous vehicles need to interact with other road users such as vehicles, pedestrians and cyclists. It is computationally overwhelming and unnecessary for autonomous vehicles to pay equal attention to all detected entities simultaneously. Therefore, it is desired if we can identify *whether* each object may potentially interact with the ego vehicle, and *when* would be the starting and ending points of the interaction period, so that limited resources can be better allocated to tackle high-priority entities timely.

Meanwhile, predicting complex human driving behavior and designing human-like behaviors for autonomous vehicles are always challenging due to our insufficient understanding of interactive human behavior. Motions of interactive agents are intrinsically high dimensional. Desirable answers to *what* essential representation (with patterns) of such motions can be extracted or learned will significantly facilitate prediction and behavior modeling [**?**], [1], [2], imitation learning [3], as well as decision and planning [4], [**?**] in terms of performances and computational efficiency with in-depth comprehension in a much lower dimensional space.

Moreover, with more and more trajectory data obtained from bird's-eye view [5] and ego-vehicle perspective [6], it is desired to learn to automatically extract interactive driving events (*whether* and *when*) and properly tag/categorize them (*what*) from large amounts of trajectory data in order to efficiently train and test behavior-related algorithms, as well as to generate scenarios and behavior for testing.

B. Related Works

1) Agents prioritization and scenario representation: In order to rank the priority of surrounding entities of the ego vehicle in a driving scene, a comprehensive scene and motion representation framework was proposed in [7] to construct semantic graph for interactive vehicle prediction based on prior knowledge regarding the static and dynamic scenes. The representation is highly interpretable and reusable, but more extension is required to explicitly learn interactive patterns from data. [8] learned the relevance rank and extracted interactions by utilizing the reaction of autonomous vehicles' (AVs) planner for an agent. However, it requires a planner and is limited to capturing interactions only with AVs. IDE-Net does not have these constraints. Though it has supervised auxiliary *whether* and *when* task, they are based on rules rather than human labeling. Additionally, the answers to *whether*, *when* and *what* for extracting interactions were not explicitly provided in these works.

Research on unsupervised representation learning was also presented recently to learn to identify driving modes in car-following scenarios [9] or extract motion primitives for vehicle encountering [10]. In [11] the authors proposed a data-driven, non-parametric method for identifying underlying interactions in the form of stochastic velocity fields. The extracted patterns were expected to be more interpretable and reusable, and it is desired to train and test the models via driving data with complete information of surrounding entities in various highly interactive scenarios. In this work, we explicitly provide the answers to *whether*, *when* and *what* for extracting interactive events and patterns with better interpretability and reusability based on highly interactive driving data with complete surrounding information.

2) Interactive pattern extraction: Research efforts have been devoted to modeling interactions of agents. [12] constructed a relation network as an augmentation to other modules to handle relational reasoning. [13] proposed a

X. Jia, L. Sun, M. Tomizuka and W. Zhan are with the Department of Mechanical Engineering, University of California, Berkeley, Berkeley, California, U.S.A. Corresponding email: wzhan@berkeley.edu

hierarchical Bayesian model to perceive interactions from movements of shapes. [14] constructed a relation-aware framework to infer relational information from the interactions, and [15] utilized real traffic data to learn the interaction policies of drivers based on game theory. The aforementioned works are based on either supervised learning of predefined interaction types or implicit modeling of interactions. In this paper, we explicitly extract interactive driving patterns with unsupervised learning to grant insight on how drivers tackle complicated driving scenarios and provide more interpretable representation for downstream modules.

Explicit inference over the potential interactions were performed in [16], [17]. Neural Relational Inference (NRI) [16] proposed a novel framework to output explicit interaction types of entities, and [17] further expanded the model to enable the combination of interactions types at the same time step. The interaction type between two objects is static, that is, given trajectories of two objects, NRI only provides one interaction type at the current time step. The model has to be run over and over again by step to predict the interaction types in a period. Such independence among steps might yield frequent switches of interaction types, which is unrealistic according to the findings in [2], [15]. Additionally, the framework determines the interaction type of a single time step based on the historical trajectories only, but the interaction types can be better inferred via utilizing the entire interaction trajectories (i.e., including both historical and future trajectories, similar as bi-LSTM outperforming LSTM). Our framework dynamically assigns interaction types for each time-step according to an overview of the complete trajectory to overcome the issues.

C. Contribution

Our contributions can be summarized as follows:

- We propose the **I**nteractive **D**riving event and pattern **E**xtraction **Net**work (IDE-Net) which can extract interactive events and patterns based on trajectory data of naturalistic human driving behavior.
- We propose a multi-task learning framework in IDE-Net to leverage the power of connected tasks in learning.
- We propose a spatial-temporal block with mixed LSTM and Transformer modules to encode trajectories, which achieves the best performance in experiments.
- We find three explainable interaction patterns by IDE-Net, which brings insight for driver behavior modeling. We did both objective and subjective evaluation to analyze the learned patterns. We also show that the proposed model could generalize well across scenarios with different road conditions and coordinate system.

II. Problem Formulation

In this work, we aim to design a learning framework which can automatically extract different interaction patterns in human driving behaviors by observing their joint trajectories. In a two-agent setting, a pair of joint trajectories from the two vehicles defines one sample. Suppose that the joint trajectories are of length T, i.e., T time steps. Then a sample contains (D^1, D^2) where $D^1 = \{\boldsymbol{d}_1^1, \boldsymbol{d}_2^1, ..., \boldsymbol{d}_T^1\}$ and $D^2 = \{\boldsymbol{d}_1^2, \boldsymbol{d}_2^2, ..., \boldsymbol{d}_T^2\}$ are, respectively, the trajectory data of vehicle 1 and vehicle 2. We also denote $D_t = \{\boldsymbol{d}_t^1, \boldsymbol{d}_t^2\}$ as the set of two vehicles' features at time step t. With these definitions, our goal is build a learning structure (such as a deep neural network) which takes in (D^1, D^2) and output interaction patterns at each time step, i.e., $l_{\text{type}} = \{l_{1,\text{type}}, l_{2,\text{type}}, ..., l_{T,\text{type}}\}$. No-interaction is also counted as one special interaction type.

Since there are no golden criteria regarding the interaction patterns, the proposed task is thus an unsupervised learning task without direct-accessible labels. In Section III, we will introduce some unique network structures which helps us achieve such a goal.

III. Model Architecture

A. Trajectory Encoding via Mixed LSTM and Transformer

Note that interactive driving patterns emerge from the joint trajectories of two interacting vehicles. To encode such spatial-temporal signals, we propose a spatial-temporal block (SB-T), as shown in the yellow box in Fig. 1. Each ST-B contains one spatial block and one temporal block, sequentially connected. For the spatial block, Transformer layers [18] are introduced to obtain agent-permutation-invariant representation of the joint trajectories while fusing the relational information of two agents at each time step[1]. Such fused representation will be further encoded by the temporal block that contains several LSTM layers. The relationship between the input and output of the ST-B can be represented by:

$$[Z^{v_1}, Z^{v_2}] = f_{\text{st}}([X^{v_1}, X^{v_2}]), \tag{1}$$

where $[X^{v_1}, X^{v_2}]$ is the input representation of the joint trajectories, and $[Z^{v_1}, Z^{v_2}]$ are correspondingly the output representation. Note that temporally, the lengths of the input and output equal. Therefore, several ST-Bs can be stacked to encode complex dynamics of the joint trajectories.

B. Learning via Auxiliary Tasks

To tackle the unsupervised "*what*" task, we design three auxiliary self/- supervised tasks in the IDE-Net to help extract interaction patterns. As shown in Fig. 1, the three tasks include: 1) an upstream "*whether*" task to predict whether interaction occurs between two vehicles, 2) another upstream "*when*" task to predict at which time steps such interaction occurs, and 3) a downstream trajectory prediction (TP) task to predict the joint future trajectories (L2 loss is used) based on historical observations and the predicted interaction probabilities from the "*whether*", "*when*" and "*what*" tasks. We call the "*whether*" and "*when*" tasks as upstream tasks because interaction patterns will exist only when interaction happens, and the features used to encode the upstream tasks should be considered in the pattern extraction task. Therefore, we design a shared "Interaction Feature Extractor" for all the three "*whether*", "*when*" and "*what*" tasks in IDE-Net. Similarly, we call the TP task as a downstream task because different interaction patterns

[1]Transformer is an attention-based order-in variable set operation in deep learning. It can effectively fuse information in set in a pairwise way and has obtained huge success in the NLP field [18].

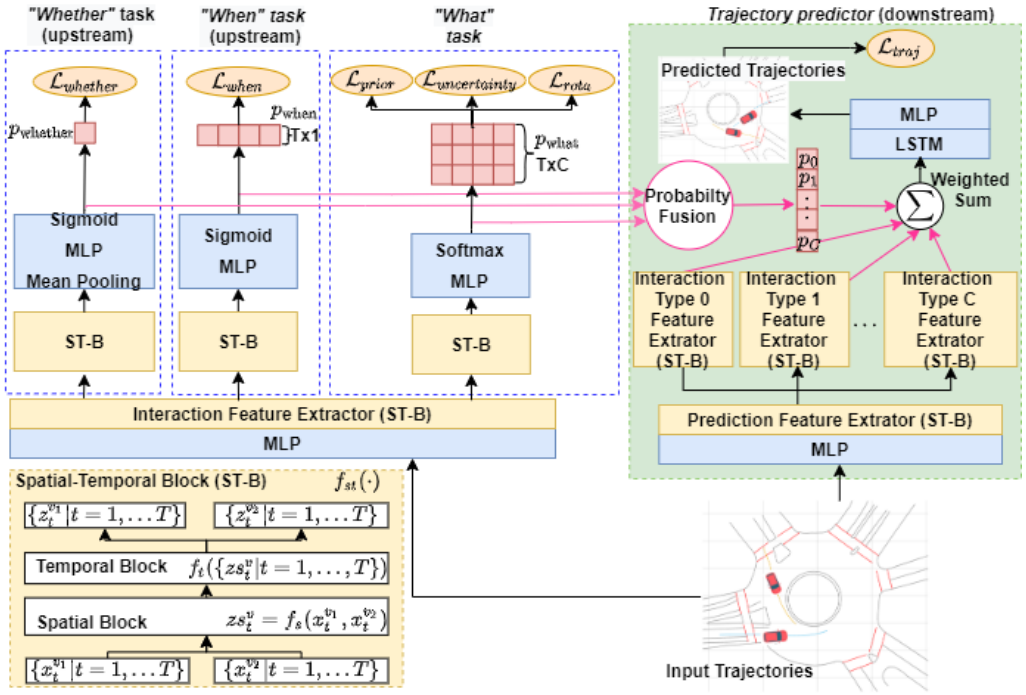


Fig. 1: The overview structure of the proposed IDE-Net. Its main task is to extract interactive driving patterns from human data, i.e., the unsupervised "*what*" task. To assist such an unsupervised learning task, we designed the IDE-Net to contain three more auxiliary tasks: a supervised "*whether*" task to predict interaction happens between the two vehicles, a supervised "*when*" task to predict the interacting time period, and finally a self-supervised trajectory prediction (TP) task (green box). The "*whether*" and "*when*" tasks are upstream tasks, thus they can utilize easy-to-obtain labels to extract low-level features that are shared with the "*what*" task. The TP task is a downstream task, which allows the IDE-Net to leverage the influence from the interaction types to the joint trajectories to better extract interaction patterns. For all the four tasks, we use several Spatial-Temporal Blocks (ST-B) $f_{st}(\cdot)$ to encode the spatial-temporal trajectories. MLP means multilayer perceptron.

will influence the final trajectories of the vehicles, namely, the joint trajectories are effective indicators of interaction types. Via the "probability fusion" block and the weighted feature extractors in the TP task in the IDE-Net (Fig. 1), we allow such information flow. Hence, IDE-Net leverages the power of relatively easy but connected upstream and downstream tasks to extract unknown interaction patterns. Both the "*whether*" and "*when*" tasks are supervised tasks where the ground-truths are manually labeled via rules. The TP task is self-supervised since all the ground-truth future joint trajectories can be directly observed from data itself.

C. *Unsupervised* What *Task - Interaction Pattern Extraction*

In the *What* task, we assume that there are C types of interaction patterns between two agents, and let network to predict the probability of each type at each time step. Although the three auxiliary tasks help reduce the difficulty of the "*What*" task, we cannot rely on those tasks due to the mode collapse problem. Specifically, the mode collapse problem means that the model would always tend to output only one kind of interaction type because the neural network's easiest strategy to minimize the trajectory prediction loss is to output only one type and optimize its corresponding parameters in the Trajectory Predictor, which is not what we want. Therefore, in IDE-Net, we introduce three additional loss to encourage it to extract meaningful interaction patterns, namely, the prior loss $\mathcal{L}_{\text{prior}}$, uncertainty loss $\mathcal{L}_{\text{uncertainty}}$, and rotation-invariant loss $\mathcal{L}_{\text{rota}}$, as shown in Fig. 1.

a) Prior Loss.: To alleviate mode collapse, inspired by the solution of mode collapse for GAN[19], we punish those extreme prediction outputs that are significantly deviating from uniform prior distribution, i.e., a prior loss $\mathcal{L}_{\text{prior}}$ is given as the entropy of the interaction type distribution:

$$\mathcal{L}_{\text{prior}} = \sum^{C} p_c \log p_c, \text{ with } p_c = \frac{1}{NT} \sum^{N} \sum^{T} p^t_{\text{what},n,c}. \tag{2}$$

Considering the intuition that some interaction types may be more common than the others, we do not hope the output distribution to be constrained as uniform distribution. Thus, we assign the prior loss with a relatively small weight so that the output distribution is not constrained too much but large enough to avoid the extreme output.

b) Uncertainty Loss.: If only with prior loss, the model finds another way to cheat: it tends to output equal confidences for all kinds of interactions at all time-steps of all samples, which can minimize the prior loss but is not what we want as well. Therefore, to encourage the IDE-Net to output certain prediction at each time step for some interaction type c, an uncertainty loss is introduced to punish

outputs with high entropy at each time step:

$$\mathcal{L}_{\text{uncertainty}} = \frac{1}{NT}\sum^{N}\sum^{T}\sum^{C} -p^{t}_{\text{what},n,c}\log p^{t}_{\text{what},n,c}. \quad (3)$$

By using the uncertainty term $\mathcal{L}_{\text{uncertainty}}$, the model would tend to output $\{[1,0,0],[0,1,0],[0,0,1]\}$ instead of simply {[0.333,0.333,0.333], [0.333,0.333,0.333], [0.333,0.333,0.333]}.

c) Rotation-Invariant Loss.: Inspired by recent success of contrasting learning [20], [21], we would like to make the prediction of interaction types rotation-invariant.

For each sample, we randomly rotate it twice with different angles and punish inconsistent outputs:

$$\mathcal{L}_{\text{rota-Invariant}} = \frac{1}{NTC}\sum^{N}\sum^{T}\sum^{C}(p^{t,1}_{\text{what},n,c} - p^{t,2}_{\text{what},n,c})^2, \quad (4)$$

where $p^{t,1}_{\text{what},n,c}$ and $p^{t,2}_{\text{what},n,c}$ represent the confidences of interaction type c at time-step t predicted based on the same sample n with two rotation angles respectively.

IV. Experiments

A. Datasets and experiment settings

Trajectory datasets facilitating meaningful interaction extraction research should contain the following aspects: 1) a variety of driving scenarios with complex driving behaviors, such as roundabouts, unsignalized intersections, merging and lane change, etc.; 2) densely interactive driving behavior with considerable numbers of interaction pairs; 3) HD map with semantic information; 4) complete information (without occlusions) of the surrounding entities which may impact the behavior of the investigated objects. The INTERACTION dataset [5] meets all the aforementioned requirements to the best of our knowledge, and our experiments were performed on it. We randomly selected 15657 pairs of vehicles from five different driving scenarios as the training set, and another 1740 pairs as the test set. We empirically set the number of interaction types as threeand it serves as a hyper-parameter of our model. The gap between two time-steps is 0.2 second and all samples are less than 100 time-steps. The prediction horizon is 5 time-steps so that the model could focus on the instantaneous effect of interactions types of each time-step

B. Data Preprocessing

Note that in structured driving environments, vehicle trajectories differ significantly among different HD maps. Our goal is to extract the interactive driving events and patterns between agents in a generic fashion over all such scenarios, i.e., excluding the influences of coordinate system or orientation[22], [23]. Therefore, we proposed three data preprocessing methods to reduce such influences from road structures:

- Coordinate Invariance (CI): CI converts global coordinates to relative coordinates by setting the origin of coordinate system as the center position of the two agents' initial positions.
- Orientation Invariance (OI): OI helps to make features not sensitive to absolute orientations by data augmentation with random rotation at each batch.
- Rotation Normalization (RN): RN normalizes the coordinates in different scenarios according to scale of the augmented data in OI. To make the rotation valid and keep the covariance of the input feature to be 1, it applies the same scale factor $\sqrt{E\frac{(X)^2+(Y)^2}{2}}$ to both X and Y coordinate.

C. Performance of the "When" and "Whether" Tasks

a) Manipulated Factors.: We manipulated a single factor: the structure of the ST-B in IDE-Net. Three conditions were compared: 1) LSTM layers only, 2) Transformer layers only, and 3) the proposed mixed LSTM and Transformer in Section III-A. In setting 2), we extended the Transformer to handle temporal signals by adding the position embedding as used in [18].

b) Dependent Variables.: For the "*whether*" task, we measured the prediction accuracy. For the "*when*" task, we measured the intersection-over-union (IoU) accuracy and adopt **IoU 0.6 accuracy** as correct prediction, namely, if IoU of a prediction is larger than 0.6, it counts as an accurate prediction. IoU is defined as:

$$\text{IoU} = \frac{[\hat{l}_{\text{start}},\hat{l}_{\text{end}}]\cap[l_{\text{start}},l_{\text{end}}]}{[\hat{l}_{\text{start}},\hat{l}_{\text{end}}]\cup[l_{\text{start}},l_{\text{end}}]}, \quad (5)$$

where $\hat{l}_{\text{start}}$ and $\hat{l}_{\text{end}}$ are, respectively, the predicted starting and ending frames of interactions.

c) Hypothesis.: We hypothesize that the proposed mixed LSTM and Transformer ST-B structure can achieve better prediction accuracy in "*whether*" and "*when*" tasks compared to the other two settings.

d) Results.: We tested the prediction performances on five different scenarios (FT, GL, MA, SR, EP) in the INTERACTION dataset [5] with diverse number of samples ranging from 9000 to 100. We tested the performance under three different training settings: 1) *Single* means training and testing both only with the current one scenario, 2) *Transfer* means training with all the other scenarios and testing on the current one, and 3) *Finetune* means training with all the other scenarios and then finetuning with the current one.

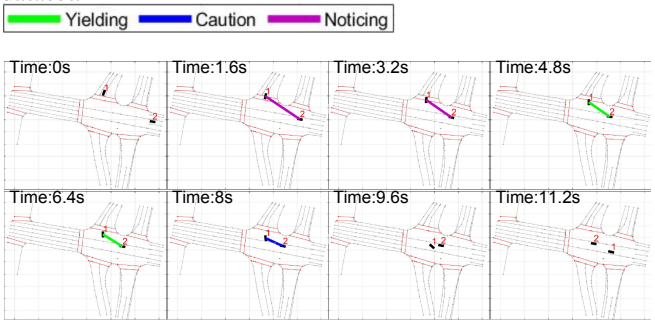
The results are shown in Table I. We can conclude that: 1) "*whether*" task was much simpler than "*when*" task since even training without samples of the current scenario in the dataset, it was still able to achieve decent results (> 0.7); 2) regarding the prediction performance, Mixed $>$ Pure LSTM $>$ Pure Transformer, which shows the effectiveness of Transformer for set of data and LSTM for sequence of data; and 3) finetuning always had the best performance among the three settings. It proved that better generalization ability can be achieved by taking into account more samples from other driving scenarios which may have completely different road structures. Therefore, for all the following experiments, we used the proposed mixed ST-B as it achieved the best performance among the three compared structures (point 2)).

D. Results of the "What" Task

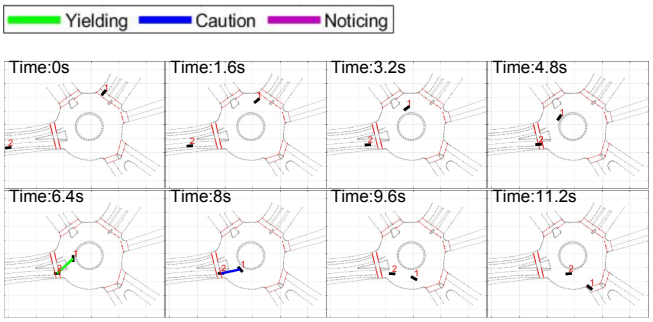
1) Learned Patterns: From the interaction types given by the IDE-Net, we conclude the three following patterns (interaction type):

Method	Setting	FT		GL		MA		SR		EP	
		When	Whether	When	Whether	When	Whether	When	Whether	When	Whether
LSTM	Single	0.818	0.876	0.914	0.923	0.783	0.862	0.500	0.891	0.904	0.889
	Transfer	0.491	0.798	0.599	0.724	0.635	0.793	0.615	0.864	0.735	0.865
	Finetune	0.837	**0.912**	0.937	0.924	0.815	**0.871**	0.750	0.923	0.923	**0.944**
Transformer	Single	0.740	0.877	0.862	0.884	0.727	0.839	0.147	0.897	0.333	0.846
	Transfer	0.433	0.777	0.528	0.718	0.535	0.760	0.473	0.884	0.612	0.841
	Finetune	0.751	0.882	0.883	0.901	0.741	0.876	0.559	0.910	0.778	0.923
Mixed	Single	0.848	0.911	0.918	0.922	0.839	0.859	0.706	0.936	0.907	0.892
	Transfer	0.542	0.812	0.681	0.692	0.634	0.802	0.637	0.879	0.653	0.833
	Finetune	**0.878**	0.909	**0.946**	**0.929**	**0.853**	0.869	**0.882**	**0.949**	**0.930**	**0.944**

TABLE I: Comparison of three ST-B settings. FT, GL, MA, SR and EP are different scenarios in the INTERACTION dataset.



(a) Each set of images represents the interaction state at different time-steps in an order of left to right and up to down. Noticing (Purple)-Yielding (Green)-Caution (Blue). For simplicity, we denote one vehicle as $v1$ and the other as $v2$. The interaction began when the two vehicles noticed each other (Purple). Then $v2$ yielded and the $v1$ kept going (Green). Finally, $v2$ started to proceed slowly after $v1$ passed and the interaction ended.



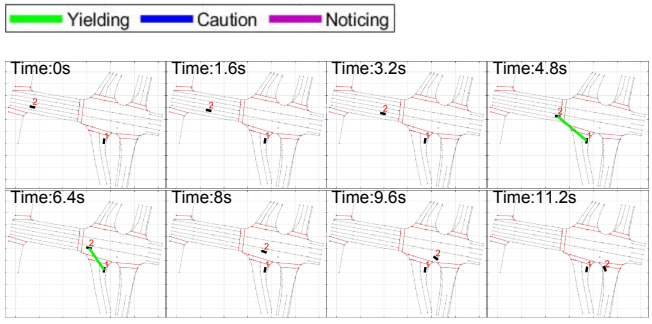
(b) Yielding (Green)-Caution (Blue). Before $v1$ arrived, the $v2$ was waiting at the stop sign. There was no Noticing part and $v1$ just passed while $v2$ kept yielding (Green). Once $v1$ passed $v1$, $v2$ proceeded immediately and followed $v1$ closely (Blue). Finally, the interaction ended.

Fig. 2: Examples of interaction patterns.

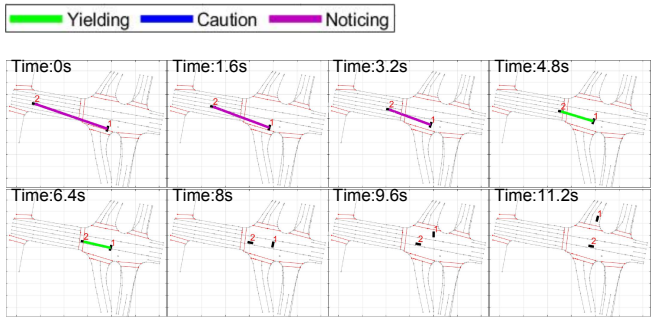
- **Noticing**: two vehicles observe each other and realize that there would be overlaps between their paths. They slow down to avoid collisions, although the speeds may still be relatively high.
- **Yielding**: one vehicle tends or decides to yield.
- **Caution**: two vehicles are relatively close to each other; one moving and the other may also be moving but cautiously.

Figures 2 and 3 demonstrate cases of the learned interaction types at different time steps of interactive trajectory pairs. We uniformly sampled 8 frames to represent each example video. In each subfigure, the line between the two vehicles represents that they are interacting and different colors represent different interaction types (Noticing-Purple, Yielding-Green, Caution-Blue).

2) Quantitative Evaluation: To quantify the performance of the pattern extraction task (*what* task), we adopted two



(a) Yielding (Green). $v1$ stopped before $v2$ arrived and thus there was no noticing part. v1 proceeded when $v2$ passed for a while. There was no potential collision (no caution part). This example shows that the model was able to capture patterns of vehicle motions separately.



(b) Noticing (Purple)-Yielding (Green). Both vehicles noticed each other at a very early stage and slowed down. $v2$ yielded until $v1$ passed. There was not caution part because their directions were perpendicular and once one of the vehicles passed, it was impossible to collide and thus there was no need to be overly cautious.

Fig. 3: Examples of interaction patterns.

metrics: 1) a cross-validation metric which compares the distribution difference between the extracted pattern types in the training and test sets; and 2) a user study which compares the human labeled interaction patterns with the extracted ones via the IDE-Net.

Type	Noticing	Yielding	Caution
Train	0.469	0.208	0.322
Test	0.452	0.229	0.322

(a) Ratios of each interaction type on training and testing sets

VTA	MTA	VAA	MAA
92.5%	76.0%	100.0%	90.5%

(b) Results of user study

TABLE II: Results with both the cross-validation metric and the user study

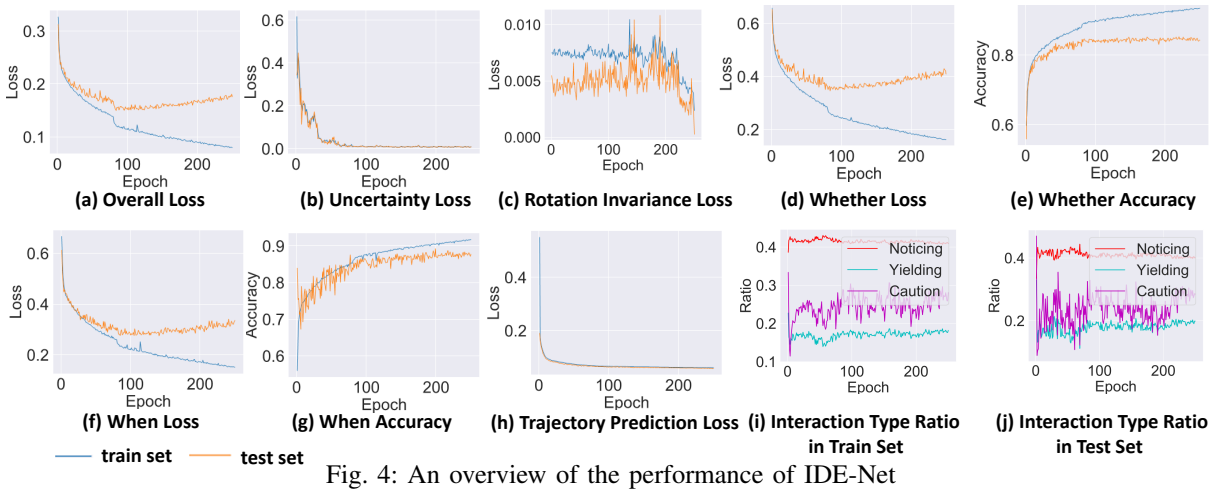


Fig. 4: An overview of the performance of IDE-Net

a) Results with the Cross-Validation Metric.: The percentages of the extracted interaction patterns are shown in Fig. 4 (i)-(j) and Table II (a). The distribution of the three interaction patterns on the training and test sets were quite similar, which indicates good generalization capability of the proposed IDE-Net, i.e., it did not overfit the training set but indeed learned to extract different interaction patterns of vehicles.

b) User Study: We also conducted a user study to provide a subjective metric to evaluate the performance of IDE-Net. We recruited 10 users with driving experiences, 5 males and 5 females. We visualized the interactive trajectories as well as the extracted interaction types together as videos and showed each user 12 videos (each scenario has at least 2 videos). We asked the users to label 1) one of the three interaction types by themselves, and 2) whether they agree with the predicted interaction types. The procedure of the user study is as follows:

1) Initial setting: We told users the meaning of the three interaction types: Noticing (N), Yielding(Y), and Caution(C). Each user watched all 12 videos in a random order.
2) Experiment I: We showed users videos only with marks indicating whether the two vehicles were interacting at each time step. We asked users to provide the interaction types and their order.
3) Experiment II: We showed them videos with marks indicating the interaction types at each time step. We asked users whether they agree with the types labeled by IDE-Net.

Metric: In Experiment I, we considered the interactions types labeled by the model were aligned with user inputs if both the interaction types and their order were the same. We calculate two metrics based on it: 1) for each video, we averaged the human labels by voting (Voting Type Accuracy - VTA); and 2) we took the mean accuracy over all videos and all users (Mean Type Accuracy - MTA). In Experiment II, similarly, we defined two metrics: *Voting Agreed Accuracy - VAA* and *Mean Agreed Accuracy - MAA*, which were obtained via the similar process as the *Voting Type Accuracy* and *Mean Type Accuracy* in Experiment I.

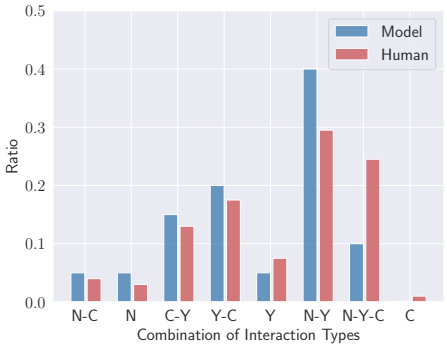


Fig. 5: The ratio of permutations labeled by the model and by human.

	Noticing	Yielding	Caution
Model	0.60	0.90	0.50
Human	0.61	0.92	0.60

TABLE III: Interaction types' ratio of occurrence in all videos.

The results are shown in Table II(b). The interaction types labeled by the users aligned quite well with the results obtained by the IDE-Net. This shows that IDE-Net was able to extract the generic interaction patterns and the extracted patterns are quite interpretable.

E. Analysis and Discussions

The performance of IDE-Net over all losses is summarized in Fig. 4(a)-(h). We can conclude that although the supervised "*whether*" and "*when*" tasks show a sign of over-fitting, the uncertain loss kept decreasing and no over-fitting occurred for the trajectory prediction task. Moreover, the consistent decreasing in uncertain loss means that the model became more and more certain about the interaction types of each time step. We also counted the time steps when the confidence over a specific interaction type was significantly larger than the other two, i.e., the confidence was greater than 0.9. Then we divided it by the overall length of the interaction window. We find that the ratio was more than **99.5%**, which indicates that the proposed model was able to find the significant differences among interaction types.

We also calculated the frequency of different interaction

types. As shown in Table III, the results given by IDE-Net were quite close to that given by human, although human tended to label more interaction types. It means that IDE-Net was relatively conservative.

To explore the co-appearance relationship of different interaction types, we define the interaction types and the order of their appearance as a permutation. For instance, in a video, if the two vehicles are labeled as first Noticing, then Yielding, and finally Caution, the permutation of this label is then "N-Y-C". We calculated the ratio of permutations labeled by the model and by users as shown in Fig. 5. Human behaved quite similar to the model except that human tend to give more "N-Y-C". "N-Y-C" was preferred probably due to the biased default in human, i.e., all three interaction types should appear and show up in sequence.

F. Differences among Learned Interaction Patterns

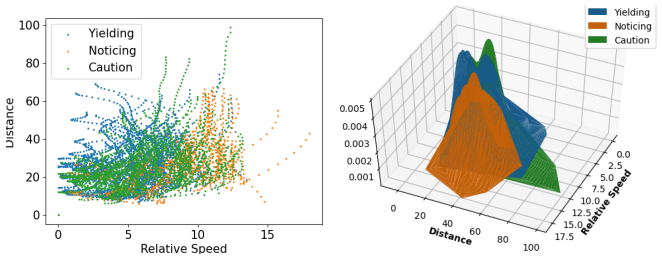


Fig. 6: Orange-Noticing, Blue-Yielding, Green-Caution. Visualization on relative distance - relative speed space with each color representing an interaction type (subfigure left). Its probability density distribution estimated by Gaussian kernel density estimation (subfigure right).

There may be concerns that the interaction types and their corresponding features for trajectory prediction were very similar to each other. Thus, in this section, we checked the differences among the three learned interaction patterns. We visualize the interaction types of each time-step in a feature space including the relative distance and relative speed between two vehicles. In Fig. 6 (left), their distributions were different. The characteristics of three interaction types are as follows: "Noticing" (Orange) happens when the two vehicles are driving towards a potentially conflicting area but still at relatively larger distances compared to "Yielding" (Blue) and relatively larger speeds compared to "Caution" (Green); "Yielding" features smaller relative distances; and "Caution" features the smallest relative speed, indicating negotiation.

To have a better illustration of the differences of their distributions, we conducted the kernel density estimation to estimate the probability density function for time steps in relative distance - relative speed space. Gaussian kernel was used. As shown in Fig. 6 (right), the three interaction types has different peaks, which shows that our model can capture the difference of different interactions patterns considering both relative speed and distance.

V. Ablation Study

IDE-Net aims to extract interpretable interaction types between vehicles. Towards the goal, the proposed "*whether-when-what*" paradigm has two benefits. First, the three tasks are intrinsically hierarchical with semantic meanings and increasing complexity. The existence of upstream tasks can improve the performance of downstream tasks. Second, the feature-sharing structure leverages the power of multi-task learning to boost both the learning performance and efficiency. We conducted ablation studies to support these two statements.

Whether	When	What	Whether Acc	When Acc	Best Epoch
✓			0.893	-	264
	✓		0.181	0.843	279
✓	✓		0.898	0.867	263
✓	✓	✓	0.909	0.878	291

TABLE IV: Ablation study for *whether* and *when* tasks under different multi-task settings. Note that with *when* task only, if the output sequence contains no "1" labels, the output of *whether* task is assigned as "0".

We conducted several comparison experiments: 1) only *whether* task, 2) only *when* task, 3) only *whether* and *when* tasks, and 4) our approach. Under each experiment setting, we calculate the accuracy (Acc) of the *whether* and *when* tasks. The goal of this set of comparison experiment is to demonstrate the power of feature-sharing structure in improving the learning efficiency and performance.

Results are shown in Tab. IV. Under the multi-task learning framework, performance of both *whether* and *when* tasks were improved remarkably. Note that the Whether Acc was bad with *when* task only. The reason is that compared to the binary classification in "whether" task, it is much harder for *when* task to output all 0s for the non-interactive trajectories due to the probabilistic nature of deep learning. Additionally, the parallel training of multiple tasks did not make a significant difference in terms of converging time (best epoch) compared to training those tasks separately. Therefore, the multi-task learning framework, which was able to avoid the burden of learning different low-layer features for each task, is efficient.

To further investigate the necessity of *whether* and *when* task, we have conducted another user study on the output from the model without them. The users cannot distinguish the extracted interaction patterns. This proved that the proposed multi-task structure can improve the interpretability of the learned patterns.

VI. Different Mask Settings for Modules

To avoid data leakage in the downstream trajectory prediction task, we employ causal mask in the trajectory predictor. Causal mask is initially defined in NLP and in our task, it means the module would only use features until time-step t to predict the future motion after time-step t. However, in the upstream tasks and *what* task, all blocks do not have causal mask, which means that each time step can access the features of all the other time steps. There are two reasons to not have ca usual mask in those modules. First, interactions are over a period of time and it is hard to decide the interaction type of a single time step without the entire trajectory. Second, with the proposed structure, only interaction types of each time step are transmitted to

the trajectory predictor. If this leakage facilitates trajectory prediction, the predicted interaction types can capture their impact on vehicle motions, which is the goal of our model.

VII. Results of Trajectory Prediction Task

To further demonstrate the effectiveness of the interaction types for downstream task - trajectory prediction, we compared IDE-Net's interaction-aware LSTM trajectory predictor with single LSTM trajectory predictor and the Constance Velocity Model. The results are in Tab. V, where ADE represents Average Distance Error between the ground-truth and the predicted trajectories and Final Distance Error (FDE) between the ground-truth and the predicted trajectories. The results show that when taking interaction into consideration, the prediction results could be improved.

	ADE	**FDE**
Constant Velocity	1.194	3.274
LSTM w/o interaction module	0.255	0.900
LSTM w interaction module	**0.239**	**0.784**

TABLE V: Comparison of trajectory prediction task.

VIII. Conclusion

In this paper, we proposed a framework to automatically extract interactive driving events and patterns from driving data without manual labelling. Experiments on the INTERACTION dataset across different driving scenarios were performed. The results showed that through the introduction of the Spatial-Temporal Block, the mutual attention and influence between trajectories of agents were effectively extracted, as evaluated via both objective and subjective metrics. Moreover, we found that the extracted three interaction types are quite interpretable with significantly different motion patterns. Such a framework can greatly boost research related to interactive behaviors in autonomous driving, e.g., interactive behavior analysis and interactive prediction/planning, since it can be used to as a tool to extract not only the interactive pairs but also the interaction patterns from enormous of detected trajectories.

There are still limitations about this work. Currently, the proposed framework focuses on two-agent settingsHowever, interactions can be potentially among multi-agents (more than two), and we will extend towards that direction in our future work.

References

[1] L. Sun, W. Zhan, Y. Hu, and M. Tomizuka, "Interpretable modelling of driving behaviors in interactive driving scenarios based on cumulative prospect theory," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2019, pp. 4329–4335.

[2] Z. Wu, L. Sun, W. Zhan, C. Yang, and M. Tomizuka, "Efficient sampling-based maximum entropy inverse reinforcement learning with application to autonomous driving," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5355–5362, 2020.

[3] N. Rhinehart, R. McAllister, and S. Levine, "Deep Imitative Models for Flexible Inference, Planning, and Control," in *2019 International Conference on Learning Representations (ICLR)*, 2019.

[4] W. Zeng, W. Luo, S. Suo, A. Sadat, B. Yang, S. Casas, and R. Urtasun, "End-to-end interpretable neural motion planner," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[5] W. Zhan, L. Sun, D. Wang, H. Shi, A. Clausse, M. Naumann, J. Kümmerle, H. Königshof, C. Stiller, A. de La Fortelle, and M. Tomizuka, "INTERACTION Dataset: An INTERnational, Adversarial and Cooperative moTION Dataset in Interactive Driving Scenarios with Semantic Maps," *arXiv:1910.03088 [cs, eess]*, 2019.

[6] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, V. Vasudevan, W. Han, J. Ngiam, H. Zhao, A. Timofeev, S. Ettinger, M. Krivokon, A. Gao, A. Joshi, Y. Zhang, J. Shlens, Z. Chen, and D. Anguelov, "Scalability in perception for autonomous driving: Waymo open dataset," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[7] Y. Hu, W. Zhan, and M. Tomizuka, "Scenario-transferable semantic graph reasoning for interaction-aware probabilistic prediction," *arXiv preprint arXiv:2004.03053*, 2020.

[8] K. S. Refaat, K. Ding, N. Ponomareva, and S. Ross, "Agent prioritization for autonomous navigation," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 2060–2067.

[9] Q. Lin, Y. Zhang, S. Verwer, and J. Wang, "MOHA: A Multi-Mode Hybrid Automaton Model for Learning Car-Following Behaviors," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–8, 2018.

[10] W. Wang, W. Zhang, and D. Zhao, "Understanding V2V Driving Scenarios through Traffic Primitives," *arXiv:1807.10422 [cs, stat]*, Jul. 2018, arXiv: 1807.10422.

[11] Y. Guo, V. V. Kalidindi, M. Arief, W. Wang, J. Zhu, H. Peng, and D. Zhao, "Modeling multi-vehicle interaction scenarios using gaussian random field," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2019, pp. 3974–3980.

[12] A. Santoro, D. Raposo, D. G. Barrett, M. Malinowski, R. Pascanu, P. Battaglia, and T. Lillicrap, "A simple neural network module for relational reasoning," in *Advances in neural information processing systems*, 2017, pp. 4967–4976.

[13] T. Shu, Y. Peng, L. Fan, H. Lu, and S.-C. Zhu, "Perception of human interaction based on motion trajectories: From aerial videos to decontextualized animations," *Topics in cognitive science*, vol. 10, no. 1, pp. 225–241, 2018.

[14] C. Choi and B. Dariush, "Learning to infer relations for future trajectory forecast," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 0–0.

[15] L. Sun*, M. Cai*, W. Zhan, and M. Tomizuka, "A game-theoretic policy-aware interaction strategy with validation on real traffic data," in *2020 IEEE Conference on Intelligence Robots and Systems (IROS)*. IEEE, 2020.

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

[17] E. Webb, B. Day, H. Andres-Terre, and P. Lió, "Factorised neural relational inference for multi-interaction systems," *arXiv preprint arXiv:1905.08721*, 2019.

[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[19] T. Salimans, I. J. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," *CoRR*, vol. abs/1606.03498, 2016. [Online]. Available: http://arxiv.org/abs/1606.03498

[20] X. Chen, H. Fan, R. Girshick, and K. He, "Improved baselines with momentum contrastive learning," *arXiv preprint arXiv:2003.04297*, 2020.

[21] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. Hinton, "Big self-supervised models are strong semi-supervised learners," *arXiv preprint arXiv:2006.10029*, 2020.

[22] S. Casas, C. Gulino, R. Liao, and R. Urtasun, "Spatially-aware graph neural networks for relational behavior forecasting from sensor data," *arXiv preprint arXiv:1910.08233*, 2019.

[23] J. Gao, C. Sun, H. Zhao, Y. Shen, D. Anguelov, C. Li, and C. Schmid, "Vectornet: Encoding hd maps and agent dynamics from vectorized representation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 525–11 533.