2022 IEEE International Conference on Robotics and Automation (ICRA)
May 23-27, 2022. Philadelphia, PA, USA

A Deep Concept Graph Network for Interaction-Aware Trajectory Prediction

Yutong Ban[1,5*], Xiao Li[1*], Guy Rosman[3], Igor Gilitschenski[3,4], Ozanan Meireles[5], Sertac Karaman[2] and Daniela Rus[1]

***Abstract*— Temporal patterns (how vehicles behave in our observed past) underline our reasoning of how people drive on the road, and can explain why we make certain predictions about interactions among road agents. In this paper we propose the *ConceptNet* trajectory predictor - a novel prediction framework that is able to incorporate agent interactions as explicit edges in a temporal knowledge graph. We demonstrate the sample efficiency and the overall accuracy of the proposed approach, and show that using the graphical structure to explicitly model interactions enables better detection of agent interactions and improved trajectory predictions on a large real-world driving dataset.**

I. Introduction

Predicting the behavior of human road agents remains a challenge. This problem is further complicated by the large number of agents acting at a vehicle's vicinity, and the large set of possible actions they could take, separately or jointly, over the prediction horizon. We often use patterns to reason about human behavior on the road – such patterns include different multi-agent interactions [1], [2], [3], maneuvers [4], rules[5], [6], and other semantics [7], [8], [9], [10], [11], [12]. These patterns can overlap in a myriad of ways and involve different number of elements from the scene – consider in Fig. 1 the interaction of the left turning ego vehicle (red) with an oncoming vehicle (highlighted blue vehicle) that it has to yield to. The grey dotted lines represent past trajectories, yellow represents future. The lines connecting the ego vehicle to other vehicles represent neighbors that the ego vehicle is paying attention to. Such an example requires the ego vehicle to reason about the scene and the interactions among its surrounding vehicles in order to make a reasonable prediction of other vehicles' behaviors.

One way to capture knowledge about the world involves *knowledge graphs* [13]. In knowledge graphs, knowledge is represented as a set of entities and relations with specific semantics. This allows a general representation about the world and entities in it. However, often these graphs are applied to datasets of static knowledge such as texts, rather than temporal and dynamic knowledge such as observations of real-life interactions.

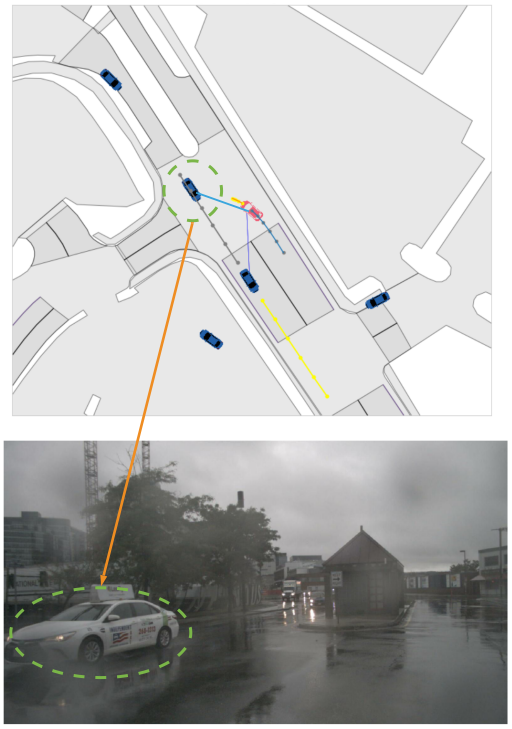

Fig. 1. **An example scenario** in which the left turning ego vehicle (red) with an oncoming vehicle (blue vehicle highlighted in orange circle) that it has to yield to. The grey dotted lines represent past trajectories, yellow represents future. The lines connecting the ego vehicle to other vehicles represent neighbors that the ego vehicle is paying attention to. Such an example requires the ego vehicle to reason about the scene and the interactions among its surrounding vehicles in order to make a reasonable prediction of other vehicles' behaviors. The camera image is in the ego vehicle's view.

We use a knowledge graph [14] with a spatio-temporal graph structure that estimates an interaction state associated with each (directed) pair of agents in the graph. This allows us to embed relations within the graph, and afford reasoning about road agent trajectories. We use the temporal graph in both the analysis and synthesis parts of an encoder-decoder based trajectory predictor. The graph allows us to place a structural prior on the predictions that can take high-level interaction semantics into account. That structure allows for accurate and data efficient representations learning for the prediction task, as well as seamless merging of

Toyota Research Institute provided funds to support this work.
* denotes equal contribution.
[1] Computer Science and Artificial Intelligence Lab, Massachusetts Institute of Technology {yban, xiaoli, rus}@mit.edu
[2] Laboratory for Information and Decision Systems, Massachusetts Institute of Technology sertac@mit.edu
[3] Toyota Research Institute {guy.rosman}@tri.global
[4] Department of Computer Science, University of Toronto gilitschenski@cs.toronto.edu
[5] Massachusetts General Hospital ozmeireles@mgh.harvard.edu

discrete detections and continuous predictions in the same representation.

Our Contributions are as follow:

- We propose the *ConceptNet* predictor - a trajectory prediction framework for capturing high-order interaction concepts (e.g. lead-follow, yielding, etc) via a temporal knowledge graph;
- We show how to leverage this framework within a encoder-decoder based trajectory predictor;
- We demonstrate how incorporating explicit high-level concepts (interactions in the case of autonomous driving, eg. concepts like Car 2 yields to Car 1) within the predictor allows for better prediction. and show the trade-off of labeled and unlabeled data on large driving datasets.

II. Related Works

Trajectory prediction has become a key area of research and heavily impacts autonomous driving. Within trajectory prediction, several approaches have been used to capture structural priors in the agent behaviors which includes maneuvers [4], rules [5], [6], multi-modality [15] and dynamic models [16].

Interactions with neighboring agents [1], [3], [10] is another important aspect that affect driver behaviors. Graphs is a natural means of modeling agent interactions and have been widely used in trajectory prediction. In computer vision and audio processing field, traditional Bayesian graphs have been adopted to model the human behaviour [17], [18], [19], [20], [21], where each node represents an individual person. In addition to that, the authors of [22] use a set of nodes to represent spatial coordinates of road-agents and weighted undirected edges to connect two agents if they are within a thresholded distance. In [16], the authors use nodes to model various agent types (car, bus, pedestrians, etc) and directed edges to model their influences to each other (directed edge takes into account perception distance). This type of node/edge presentation is common in the trajectory forecasting literature. The graph network in our work is different in that our edges represent *explicit interactions* between two vehicle and is able to intake auxiliary guidance (interaction labels) during training. During inference, out predictor operates on two levels - first it predicts the high-level interaction semantics between any two vehicles in the graph and then this information is used to generate trajectory predictions. We show that this explicit modeling of interactions along with interaction labels during training improves the predictor's performance as well as explainability.

More broadly, graphical knowledge representation has been extensively studied in both the knowledge representation community [13], [23] and more recently in a growing community centered around graph neural networks [24], [25], [26], [27], [28], [29]. It has also been applied to the inductive physics fields for learning the concepts in physics [30], [31], [32]. Most of the works in knowledge graphs involves static information, and does not rely on sensor data streams, nor inference and predictions from these streams. The spatial-temporal graph neural network is first applied to surgical analysis [33]. In this work we improve the model and convert it for use in vehicle trajectory predictions.

III. Concept Network

In this section, firstly, we formulate the trajectory prediction problem and introduce a single agent prediction model. Then we present in detail the proposed ConceptNet for multi-agent trajectory and interaction predictions.

A. Problem Formulation

We focus on the trajectory prediction problem in driving. It is defined as given a set of past trajectories of the driving agents, a predictor tries to estimate the distribution the future trajectories. In a driving scene, assume there are N agents. We observe the past T_P steps of trajectories, which are represented by $\{\mathbf{S}_t\}_{t=T_P}^{0}$. At each time step, $\mathbf{S}_t = [s_t^0 \dots s_t^N]$. We include in $\mathbf{S}_t$ all agents' positions, as well as map information. Our goal is to correctly predict the distribution of $\{\mathbf{S}_t\}_{t=0}^{T_F}$, within the next T_F frame of these agents. In addition to predicting the agents' positions, we would also like to predict the interactions between the entities. Each interaction in the scene is represented by a semantic label and let $\mathbb{L}$ be a set of L labels that describe semantics at different time points in the future. (e.g. "vehicle A is yielding to vehicle B at 0.5 seconds"). In a labeled trajectory prediction problem, we must predict the semantic labels correctly, in addition to predicting the trajectories.

B. Single Agent Predictor

Sequential encoder-decoder frameworks are well explored in the single agent prediction task. In such a framework, the encoder employs past information and the decoder tries to emit the future predictions. In a sequential data prediction problem, long-short term memory (LSTM) units are commonly used. In this case, the agent is represented by a sequence-to-sequence model described below.

Encoder The encoder is an LSTM, which takes the past vehicle trajectories $\mathbf{S}$ and velocities as inputs and aggregates all the past information into the last encoder hidden state. An agent-centered raster map is fed into a map encoder for map features. The map feature is further concatenated with the last encoder hidden state. A fully-connected layer is applied project the concatenated feature to dimension M.

Decoder The decoder is also an LSTM. It takes the concatenated feature to initialize its hidden state. At each time step, the decoder emits a two-dimensional position of the agent in an autoregressive fashion.

C. Multi-Agent Prediction with ConceptNet

Different from a single agent predictor, we introduce a multi-agent graph neural network based predictor, modeling the agents' dynamics as well as the interactions among them. The graph neural network H is defined as:

$$H = (V, E) \tag{1}$$

where $v \in V$ are graph nodes, that represent entities (such as agents or map elements), and graph edges $e \in$

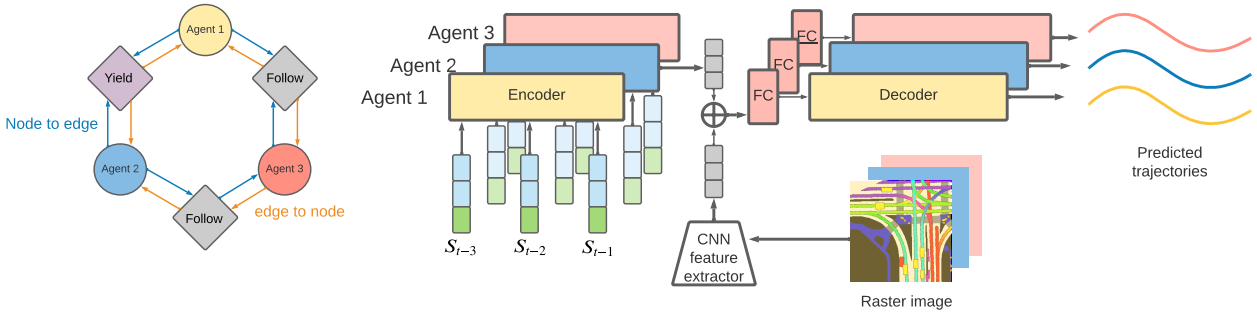

Fig. 2. **The overview of the ConceptNet trajectory predictor.** Each agent is represented by a node (circle) and each interaction is represented by an edge (diamond). Messages at first pass from node to edge to aggregate the node information to infer the interactions, then the information pass back to nodes throughout the edge-to-node message passing. Each node is an encoder-decoder predictor.

E represent relations between entities, with some defined interaction semantics (e.g. agent 1 follows agent 2). The detailed description of nodes and edges is as follows:

Nodes Each node follows a single agent encoder-decoder described in III-B. In practice, the LSTM hidden state dimension is $M = 32$ for all the nodes. They are the same for both encoders and decoders. A node LSTM should not only receive its own trajectory information as input, it should also receive the neighbour information through edges, depending on the activation of the interactions. If an interaction between the current node $V_{t,n}$ and a neighbour node $V_{t,m}$ is activated, the interaction edge should aggregate the information from both nodes and send the aggregated information back.

Edges As interactions among agents vary with time, the edge network should also be able to capture the temporal variations of the interactions. To achieve this, each edge is also designed as an LSTM encoder-decoder. Each edge has an independent hidden state, which is represented by the vector $e_k^t \in \mathbb{R}^{d_e}$ at time t. Therefore the LSTM input is a concatenation of (i) the agents' hidden states which are involved in the interaction, e.g. 'agent 1 yields agent 2' involves both 'agent 1' and 'agent 2' (ii) The relative positions between the two agents. The output of the LSTM is the probability of whether agent 1 is yielding to agent 2.

Message Passing Message passing is a key component to interaction modeling. It can be categorized into node-to-edge passing and edge-to-node passing. We first compute an edge update step, which is to aggregate the node information into interaction features:

$$e_k^{t+1/2} = \phi^E(e_k^t, v_r^t, v_s^t, u^t) \tag{2}$$

where ϕ^E is the edge LSTM, v_r^t, v_s^t are the nodes which are involved in the interaction, and u^t is the relative position between the nodes. After aggregating node information, an edge-to-node aggregation step is applied. The edge-to-node aggregation step aims to transfer the interaction information back to nodes. formally written as:

$$\begin{aligned} \bar{e}_n^{t+1/2} &= \rho^{e\to v}({E'}_n^{t+1/2}) \\ v_n^{t+1} &= \phi^V(\bar{e}_n^{t+1/2}, v_n^t) \end{aligned} \tag{3}$$

where $\rho^{e\to v}$ is a feed-forward network that maps the edge feature ${E'}_n^{t+1/2}$ to node n. Then the node LSTM ϕ^V takes the projected feature $\bar{e}_n^{t+1/2}$ as input and evolves temporally. The temporal process of the node LSTM and edge LSTM are formally written as:

$$\begin{aligned} \phi^V(\bar{e}_n^{t+1/2}, v_n^t, u^t) &= LSTM(h^{v,k}, \bar{e}_n^{t+1/2})) \\ \phi^E(e_k^t, \bar{v}_r^t, \bar{v}_s^t, u^t, I^t) &= LSTM(h^{v,k}, \bar{v}_r^t, \bar{v}_s^t)) \end{aligned} \tag{4}$$

where ϕ^V is the node LSTM.

D. Network Emissions

The model is able to generate from different hidden states both spatial and temporal emissions according to the task.

Temporal emissions are generated by an emission head from the edges LSTM hidden state, which indicates the existence of a single interactions.

Spatial emissions come from the hidden states of the nodes, which is in dimension 2, indicating the 2D position of the agents.

IV. Experiments

NuScenes Dataset. We use the NuScenes dataset [34] for training and evaluation. The dataset contains 1000 scenes of 20s each. It also includes rich semantic information including 23 object classes (pedestrian, vehicle, etc) and HD maps with 11 annotated layers (lanes, walkways, etc).

Method of Evaluation. The first metric we use is the average displacement error (**ADE**) - average L2-norm between the predicted and ground truth trajectories. ADE measures how well our model is able to generate trajectories that mimic those from the human demonstrators in the dataset. The second metric is the final displacement error (**FDE**) - the L2 distance between the final points of the prediction and ground truth ; the third is max displacement error (**MaxDist**) - the maximum point-wise L2 distance between the prediction and ground truth. This measures the largest error for each prediction and lastly the semantic accuracy (**SA**) - calculated by the number of true interactions divided by the number of predicted interactions.

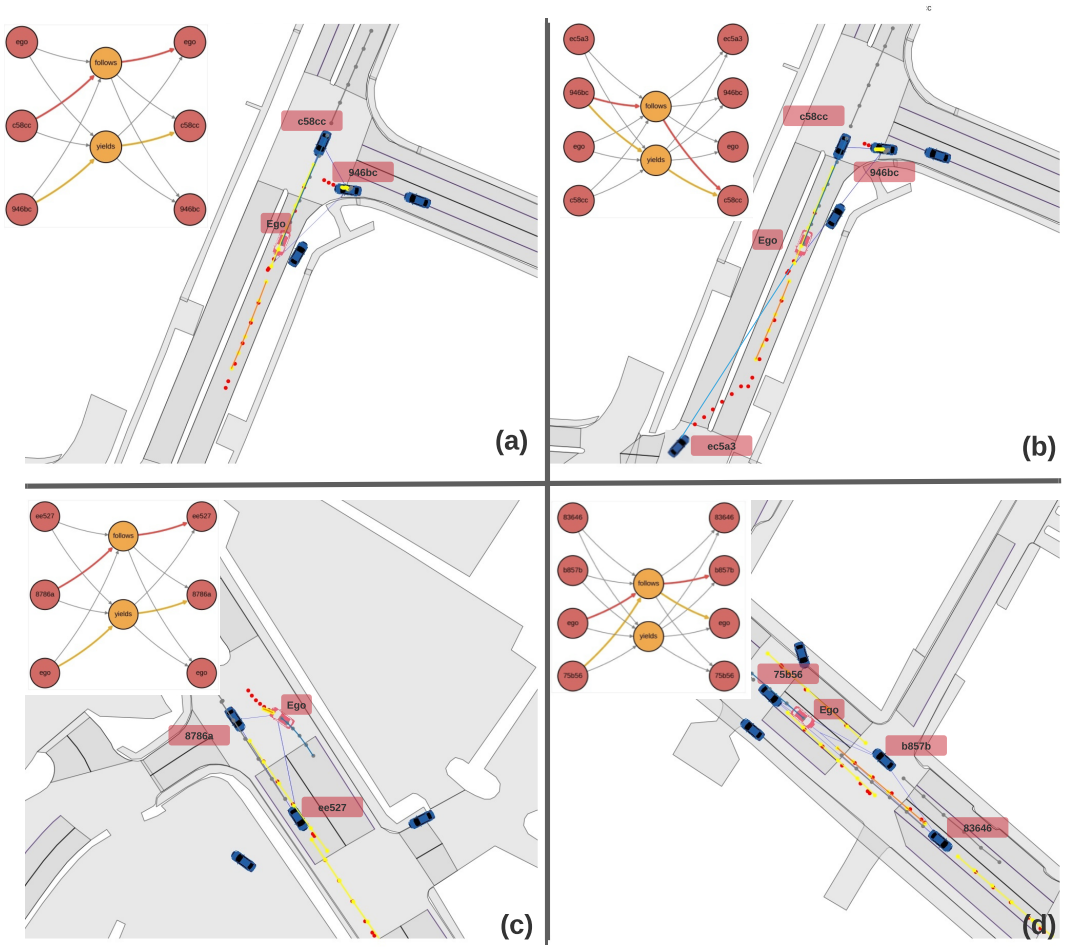

Fig. 3. **Example scene predictions.** the ego vehicle (red) along with its 4 nearest neighbor (blue) vehicles (within a range of 40 meters) are connected into a graph (connection shown as light blue lines) which is used to construct the ConceptNet. The yellow dot-dash lines are the ground truth future trajectories. The red dotted trajectories are the predictions. The graph on the top-left hand corner of each scenes illustrates the output of the ConceptNet at this instant in time. Here the ConceptNet predicts interactions between vehicles in graph through the nodes *follows* and *yields*. The edges are thickened and colored if an interaction is predicted to occur. For example, in Figure 3 (a), ConceptNet predicts that "*c58cc* follows *ego*" and "*946bc* yields *c58cc*".

TABLE I
PERFORMANCE METRIC COMPARISONS

Model	ADE (m)		FDE (m)		MaxDist (m)	
	mean	90th	mean	90th	mean	90th
CoverNet	4.01	8.34	7.97	17.87	8.17	17.89
No-ConceptNet	2.80	4.9	5.47	**10.4**	5.57	**10.41**
ConceptNet	**2.18**	**4.66**	**4.72**	11.01	**4.78**	11.01

Comparison Cases. We use the following settings for comparison and ablation - ***ConceptNet***: this corresponds to the architecture in Figure III; ***No-ConceptNet***: this corresponds to the architecture in Figure III without ConceptNet; ***Covernet*** - this is the architecture proposed in [35]. We also conduct a set of self-abalation studies which will be discussed in the results section.

Creating interaction labels. To guide the learning of high-level interactions, we need to create interaction labels from low-level data (trajectories, speed, steering, etc). In this work, we generate two types of interactions - *follow* and *yield*. Suppose we have 2 vehicles a_1 and a_2. a_1 is said to *follow* a_2 if they satisfy (a) the angle between the direction of their velocities is less than a threshold a_1 and (b) they are traveling along the same lane. a_1 is said to *yield* to a_2 if (a) a_1's speed is less than a threshold (slowing down), (b) the angle between 2 cars' velocities is larger than a threshold (not traveling along or against each other) and (c) the (extended) future trajectory of a_1 intersects that of $a2$ (both cars trying to cross the same intersection). Our graph network is not limited to the number of nodes in the graph. In practice, to facilitate the training processes the interactions of a maximum of 4 vehicles at a time, therefore, we generate *follow* and *yield* interaction labels for all (directed) pairs of vehicles within the graph.

Results and Discussion. In Figure 3, we show 4 example scenes that showcase the capabilities of our predictor. In each scene, the *ego* vehicle (red) along with its 4 nearest

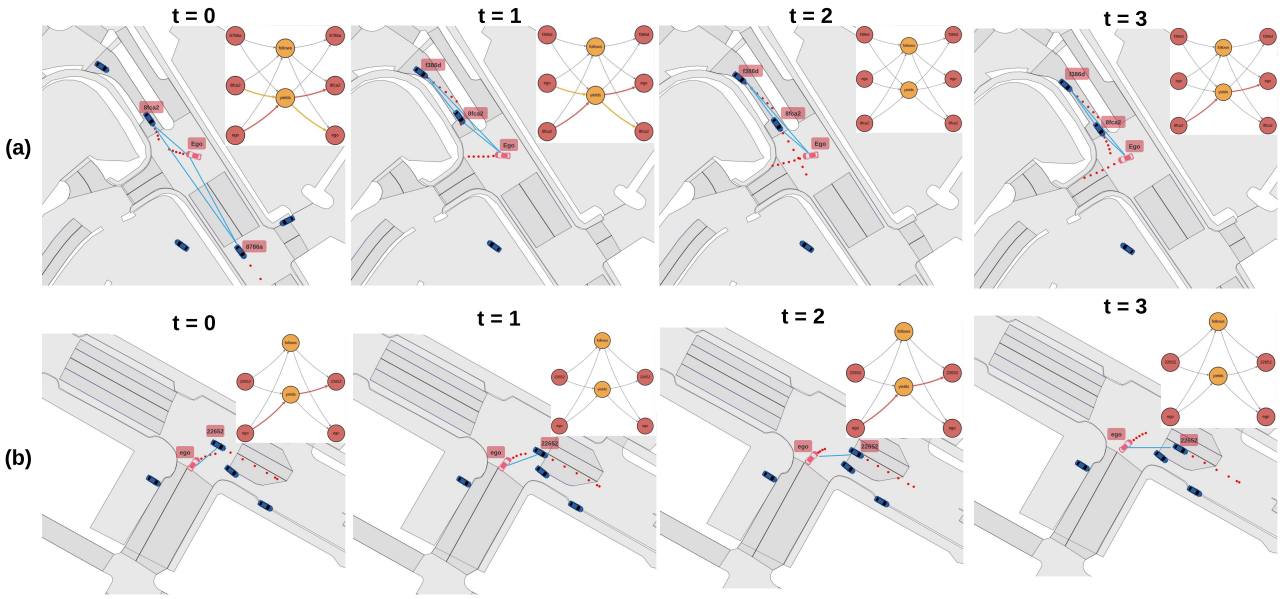

Fig. 4. **Example execution traces** at testing time. To make the figures more legible, we keep only the predictions (red dotted trajectories) and the vehicle connections (blue lines). Both traces focus on the more interesting case of yielding.

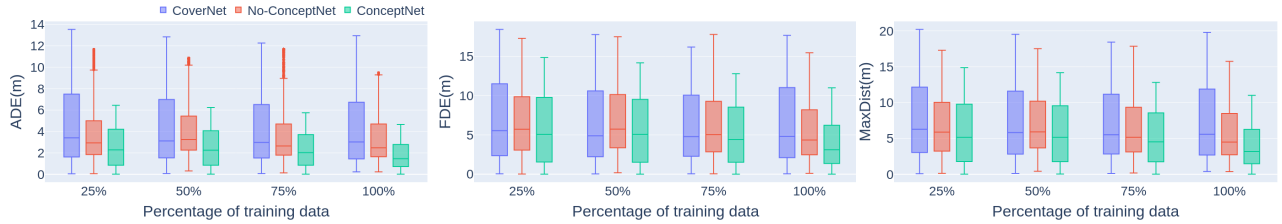

Fig. 5. **Sample efficiency study.** We trained our model along with comparison models at different fractions of the training dataset (25%, 50%, 75%, 100%). Validation set remains the same. The results are shown in Figure 5. To help with visualization, the box plots are constructed from data up to the 90th percentile of their individual distributions (major outliers are discarded).

TABLE II

ABLATION STUDY. THE PERFORMANCE OF DIFFERENT MODEL CONFIGURATIONS IS EVALUATED. BY USING ALL OF THE INTERACTION SUPERVISION LABELS, THE PROPOSED MODEL HAS ACHIEVED THE BEST PERFORMANCE.

	ADE (m)	**FDE (m)**	**Max Distance (m)**	**SA (%)**
No relations	2.8	5.4	5.5	-
No interaction labels	2.6	5.4	6.0	-
Attention on edges	2.3	5.0	5.1	-
Node-to-node	2.2	4.9	5.0	-
Yield-relation only	2.2	4.7	4.8	67.2%
Follow-relation only	2.3	5.2	5.3	55.4%
Proposed	2.1	4.8	4.8	73.4%

neighbor (blue) vehicles (within a range of 40 meters) are connected into a graph (connections shown as light blue lines) which is used to construct the ConceptNet. The yellow dot-dashed lines are the ground truth future trajectories. The red dotted trajectories are the predictions. The graphs on the top-left hand corner of each scene illustrates the output of the ConceptNet at this instant in time. Here the ConceptNet predicts interactions between vehicles through nodes *follows* and *yields*. The edges are thickened and colored if an interaction is predicted to occur. For example, in Figure 3 (a), ConceptNet predicts that "*c58cc* follows *ego*" and "*946bc* yields *c58cc*". We can see that these two interaction predictions result in the correct behaviors in trajectory terms. Agent *c58cc*'s predicted trajectory is closely in line with that of the ego vehicle. And agent *964bc*'s trajectory is predicted to slow down to let *c58cc* to pass. This set of scenes shows that our predictor is able to reasonably predict the high-level interactions between vehicles in its graph and use this prediction to aid the generation of trajectory forecasts. In Figure 3 (b), the network predicts that agent *946bc* both

follows and yields to agent *c58cc*. This is an interesting prediction given that both are true. In this case, agent *946bc* is yielding for *c58cc* to pass and it's also planning to turn left (shown by the ground truth trajectory) into the same lane as *c58cc* which takes it into a following behavior. Our predictor is able to capture both interactions.

To study the effectiveness of ConceptNet, we compare our architecture with and without ConceptNet along with an off-the-shelf predictor (CoverNet [35]). The results shown in Table I indicates that adding ConceptNet yields a significant improvement in most evaluation metrics. This is due to the fact that ConceptNet is able to consume additional priors in the form of interaction labels. Also the prediction of each one agent is conditioned on the information of all other agents in the graph (through message passing). Such consideration of surrounding vehicle behaviors is much in line with how humans drive and therefore ConceptNet is able to generate more human-like trajectories.

To study the continuous evolution of the ConceptNet during execution, Figure 4 illustrates two traces at testing time. To make the figures more legible, we keep only the predictions (red dotted trajectories) and the vehicle connections (blue lines). Both traces focus on the more interesting case of yielding. One noticeable behavior is that when ConceptNet outputs the *yield* interaction for a vehicle, the predicted trajectory for that vehicle is shortened signifying a slow down motion. This is shown in Figure 4 (a) for vehicle *8fca2* at times $t = 1$ and $t = 3$. At time $t = 2$, the network did not output the *yield* interaction for *8fca2* which results in a elongated trajectory. The same pattern is also shown in Figure 4 (b) from $t = 1$ to $t = 3$ for the *ego* vehicle. From Figure 4 (a) we can also observe that ConceptNet is responsive to the change in possible interactions among its agents. At $t = 0$ it is not clear between *ego* and *8fca2* which vehicle will be yielding so it predicted both vehicles to yield each other (the more conservative and safer prediction). As time rolls out, it becomes clear that *8fca2* is yielding to *ego* and our model is able to correctly capture this interaction through time. A caveat occurs in Figure 4 (a) $t = 1$ where ConceptNet predicts that *ego* is yielding to *8fca2* but the generated trajectory does not reflect this. In contrast, the predicted trajectory for *8fca2* under the same interaction nearly stops in front of the intersection. This is a case where the trajectory prediction is accurate but the interaction prediction is not.

It is expected that incorporating the right priors can improve the sample efficiency of prediction models. To investigate, we trained our model along with comparison models at different fractions of the training dataset (25%, 50%, 75%, 100%). Validation set remains the same. The results are shown in Figure 5. To help with visualization, the box plots are constructed from data up to the 90th percentile of their individual distributions (major outliers are discarded). From the figures we can see that ConceptNet is able to out-perform comparison cases at all fractions of the training set and is able to obtain descent accuracy at even 25% of training data (although with a larger spread).

In an abalation study, we compared different ways of predicting the scene, the results are shown in Table II. We first compare with no relations, we cut-off the message-passing of both node-to-edge and edge-to-node. With no relation information, the model achieved the worst ADE of 2.8m. We also trained the model with no interaction labels, so the edge module can only learn automatically the neighbour information without high-level semantics. Moreover, we compare the model with node-to-node message passing, without the explicit construction of the edges. We also compare the model with yield-relation-only and follow-relation-only, only one of the relation is used. We observe that the proposed model achieved the most accurate prediction trajectories (ADE 2.1m). Moreover, in terms of the semantic accuracy the proposed model carries out highest interaction score (73.4%).

In the results above, we have showed that by designing a prediction model that is able to consume high-level semantic labels, we can obtain a predictor with better prediction accuracy and explainability. It is worth noting that, the performance of the predictor depends on the quality of the interaction labels. This is because part of ConceptNet's capability is to mimic the functionality of the interaction label generator. At deployment time, it use this learned ability to generate interactions labels and use them to affect the behaviors of the predicted trajectories.

V. Conclusion

In this work, we propose the use of high-level interaction labels as auxiliary guidance to training vehicle trajectory predictors. We introduce the use of the ConceptNet as a means to explicitly model interactions and consumes the interaction labels during training. On a real-world driving dataset, we show that incorporating the ConceptNet improves the overall accuracy of the trajectory predictor with the added benefit of enhanced explainability. For future work, we will explore the incorporate vehicle-to-pedestrian interactions as well as the integration of ConceptNet predictions with differentiable planners.

References

[1] 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, 2016, pp. 961–971.

[2] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi, "Social gan: Socially acceptable trajectories with generative adversarial networks," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 2255–2264.

[3] J. Amirian, J.-B. Hayet, and J. Pettré, "Social ways: Learning multi-modal distributions of pedestrian trajectories with gans," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, 2019, pp. 0–0.

[4] N. Deo and M. M. Trivedi, "Multi-modal trajectory prediction of surrounding vehicles with maneuver based lstms," in 2018 IEEE Intelligent Vehicles Symposium (IV). IEEE, 2018, pp. 1179–1184.

[5] X. Li, G. Rosman, I. Gilitschenski, J. DeCastro, C.-I. Vasile, and S. K. D. Rus, "Differentiable logic layer for rule guided trajectory prediction," 2020.

[6] X. Li, G. Rosman, I. Gilitschenski, C.-I. Vasile, J. A. DeCastro, S. Karaman, and D. Rus, "Vehicle trajectory prediction using generative adversarial network with temporal logic syntax tree features," IEEE Robotics and Automation Letters, vol. 6, no. 2, pp. 3459–3466, 2021.

[7] M. Liang, B. Yang, R. Hu, Y. Chen, R. Liao, S. Feng, and R. Urtasun, "Learning lane graph representations for motion forecasting," in European Conference on Computer Vision. Springer, 2020, pp. 541–556.

[8] 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.

[9] N. Rhinehart, R. McAllister, K. Kitani, and S. Levine, "Precog: Prediction conditioned on goals in visual multi-agent settings," in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 2821–2830.

[10] I. Gilitschenski, G. Rosman, A. Gupta, S. Karaman, and D. Rus, "Deep context maps: Agent trajectory prediction using location-specific latent maps," IEEE Robotics and Automation Letters, vol. 5, no. 4, pp. 5097–5104, 2020.

[11] C. Tang and R. R. Salakhutdinov, "Multiple futures prediction," Advances in Neural Information Processing Systems, vol. 32, pp. 15 424–15 434, 2019.

[12] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone, "Trajectron++: Multi-agent generative trajectory forecasting with heterogeneous data for control," ArXiv, vol. abs/2001.03093, 2020.

[13] M. Nickel, K. Murphy, V. Tresp, and E. Gabrilovich, "A review of relational machine learning for knowledge graphs," Proceedings of the IEEE, vol. 104, no. 1, pp. 11–33, 2015.

[14] A. Bretto, Hypergraph Theory - An Introduction. Springer International Publishing, 2013. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01024351

[15] H. Cui, V. Radosavljevic, F.-C. Chou, T.-H. Lin, T. Nguyen, T.-K. Huang, J. Schneider, and N. Djuric, "Multimodal trajectory predictions for autonomous driving using deep convolutional networks," 2019 International Conference on Robotics and Automation (ICRA), pp. 2090–2096, 2019.

[16] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone, "Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data," in ECCV, 2020.

[17] Y. Ban, S. Ba, X. Alameda-Pineda, and R. Horaud, "Tracking multiple persons based on a variational bayesian model," in European Conference on Computer Vision. Springer, 2016, pp. 52–67.

[18] Y. Ban, L. Girin, X. Alameda-Pineda, and R. Horaud, "Exploiting the complementarity of audio and visual data in multi-speaker tracking," in Proceedings of the IEEE International Conference on Computer Vision Workshops, 2017, pp. 446–454.

[19] Y. Ban, X. Alameda-Pineda, C. Evers, and R. Horaud, "Tracking multiple audio sources with the von mises distribution and variational em," IEEE Signal Processing Letters, vol. 26, no. 6, pp. 798–802, 2019.

[20] Y. Ban, X. Alameda-Pineda, L. Girin, and R. Horaud, "Variational bayesian inference for audio-visual tracking of multiple speakers," IEEE transactions on pattern analysis and machine intelligence, vol. 43, no. 5, pp. 1761–1776, 2019.

[21] X. Li, Y. Ban, L. Girin, X. Alameda-Pineda, and R. Horaud, "Online localization and tracking of multiple moving speakers in reverberant environments," IEEE Journal of Selected Topics in Signal Processing, vol. 13, no. 1, pp. 88–103, 2019.

[22] R. Chandra, T. Guan, S. Panuganti, T. Mittal, U. Bhattacharya, A. Bera, and D. Manocha, "Forecasting trajectory and behavior of road-agents using spectral clustering in graph-lstms," IEEE Robotics and Automation Letters, vol. 5, pp. 4882–4890, 2020.

[23] B. Fatemi, P. Taslakian, D. Vazquez, and D. Poole, "Knowledge hypergraphs: Prediction beyond binary relations," in AAAI. Yokohama, Japan: International Joint Conferences on Artificial Intelligence Organization, Jul. 2020, pp. 2191–2197. [Online]. Available: https://www.ijcai.org/proceedings/2020/303

[24] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," IEEE transactions on neural networks and learning systems, vol. 32, no. 1, pp. 4–24, 2020.

[25] P. W. Battaglia, J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner, C. Gulcehre, F. Song, A. Ballard, J. Gilmer, G. Dahl, A. Vaswani, K. Allen, C. Nash, V. Langston, C. Dyer, N. Heess, D. Wierstra, P. Kohli, M. Botvinick, O. Vinyals, Y. Li, and R. Pascanu, "Relational inductive biases, deep learning, and graph networks," arXiv:1806.01261 [cs, stat], Oct. 2018, arXiv: 1806.01261. [Online]. Available: http://arxiv.org/abs/1806.01261

[26] H. Wang, F. Zhang, M. Zhang, J. Leskovec, M. Zhao, W. Li, and Z. Wang, "Knowledge-aware graph neural networks with label smoothness regularization for recommender systems," in Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining, 2019, pp. 968–977.

[27] Y. Feng, H. You, Z. Zhang, R. Ji, and Y. Gao, "Hypergraph neural networks," in AAAI, vol. 33, no. 01, 2019, pp. 3558–3565.

[28] Z. Zhang, F. Zhuang, H. Zhu, Z. Shi, H. Xiong, and Q. He, "Relational graph neural network with hierarchical attention for knowledge graph completion," in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, no. 05, 2020, pp. 9612–9619.

[29] S. Deng, H. Rangwala, and Y. Ning, "Dynamic knowledge graph based multi-event forecasting," in Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 1585–1595.

[30] K. Yi, C. Gan, Y. Li, P. Kohli, J. Wu, A. Torralba, and J. B. Tenenbaum, "Clevrer: Collision events for video representation and reasoning," ICLR, 2020.

[31] Z. Chen, J. Mao, J. Wu, K.-Y. K. Wong, J. B. Tenenbaum, and C. Gan, "Grounding physical concepts of objects and events through dynamic visual reasoning," ICLR, 2021.

[32] C. Gan, N. Wang, Y. Yang, D.-Y. Yeung, and A. G. Hauptmann, "Devnet: A deep event network for multimedia event detection and evidence recounting," in CVPR, 2015.

[33] Y. Ban, J. A. Eckhoff, T. M. Ward, D. A. Hashimoto, O. R. Meireles, D. Rus, and G. Rosman, "Concept graph neural networks for surgical video understanding," https://arxiv.org/abs/2202.13402, 2022.

[34] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," arXiv preprint arXiv:1903.11027, 2019.

[35] T. Phan-Minh, E. Grigore, F. Boulton, O. Beijbom, and E. M. Wolff, "Covernet: Multimodal behavior prediction using trajectory sets," 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 14 062–14 071, 2020.