Relation-based Motion Prediction using Traffic Scene Graphs

Maximilian Zipfl[1], Felix Hertlein[1], Achim Rettinger[2], Steffen Thoma[1], Lavdim Halilaj[3], Juergen Luettin[3], Stefan Schmid[3], Cory Henson[4]

Abstract— **Representing relevant information of a traffic scene and understanding its environment is crucial for the success of autonomous driving. Modeling the surrounding of an autonomous car using semantic relations, i.e., how different traffic participants relate in the context of traffic rule based behaviors, is hardly been considered in previous work. This stems from the fact that these relations are hard to extract from real-world traffic scenes. In this work, we model traffic scenes in a form of spatial semantic scene graphs for various different predictions about the traffic participants, e.g., acceleration and deceleration. Our learning and inference approach uses Graph Neural Networks (GNNs) and shows that incorporating explicit information about the spatial semantic relations between traffic participants improves the predicdtion results. Specifically, the acceleration prediction of traffic participants is improved by up to 12% compared to the baselines, which do not exploit this explicit information. Furthermore, by including additional information about previous scenes, we achieve 73% improvements.**

I. Introduction

In road traffic, there exist many influencing factors important to safely drive from one point to another. While a number of these factors can be considered static, for example the road infrastructure, the behavior of the various traffic participants changes dynamically over time. Therefore, the behavior of nearby traffic participants is important in determining the appropriate action of an autonomous vehicle, e.g. steering or braking. Traffic participants might react differently depending on their particular situation, so a model about their intentions, indicating whether they will brake soon, would provide added value. To obtain that, in addition to the position of traffic participants, knowledge about how they relate to other traffic participants is crucial. This is particularly relevant for autonomous driving when the autonomous agent needs to detect and evaluate the current traffic situation (scene) in order to generate a trajectory. Human drivers derive an understanding of the underlying traffic situation directly from their perceived environment and their knowledge about traffic rules, as well as experience from previous social interactions and behaviors. In this work, we incorporate explicit spatial information about relations among the various traffic participants, and clearly demonstrate the importance of this information. For autonomous agents, it is desirable to have an explicit description of the road environment or driving context in order to evaluate the influences of individual factors and ensure the safety requirements of the autonomous vehicle. To this end, we use semantic scene graphs as a way to describe the driving context of an autonomous agent, including the relation to nearby traffic participants independently of scene types (constellation of traffic participants) or road geometry. This description is represented in a form that can be used as contextual information for the motion planning of an autonomous agent.

This paper is structured as follows: In Section II we provide a review of existing traffic scenario descriptions and related work in the context of motion prediction. Section III describes a short overlook over the used scene representation model and how certain graph's attributes are derived. In Section IV, our implementation in regard to the used datasets is discussed. Moreover, two net architectures are proposed on how to extract information from a scene graph. The results of the models are shown and evaluated in Section V. Finally, in Section VI we conclude this contribution.

II. State of the Art

A. Scene and Scenario Descriptions

In recent years, knowledge graphs have found their way into a wide variety of domains. They are used to store knowledge in a structured and extensible graph representation and enable agents to query them for all kinds of information. Ulbrich et al. [1] present an approach for representing and modeling context and environment in driving scenarios. It comprises various layers for describing information about lane, traffic rules, participants, and the overall situation. Henson et al. [2] describes a semantic model with the most important concepts in a driving scene such as *sequence*, *scene*, *participant* and their inter-relationships. A knowledge-graph based approach for representing and fusing heterogeneous types of information of traffic scenes is presented in [3]. The integrated knowledge is then used along with graph neural networks for classification of driving situations.

B. Motion Prediction

A general survey about motion prediction for automated driving can be found in [4]. A prominent method that reasons jointly about the 3D scene layout of intersections as well as the location and orientation of objects in the scene is proposed in [5]. An approach that includes high-level semantic information in a spatial grid, combined with CNN to model complex scene context for trajectory prediction, is described in [6]. Casas et al. [7] present IntentNet, a one-stage detector

[1]FZI Research Center for Information Technology, Karlsruhe, Germany {zipfl, hertlein, thoma}@fzi.de
[2]Trier University, Trier, Germany rettinger@uni-trier.de
[3]Bosch Corporate Research, Renningen, Germany {lavdim.halilaj, juergen.luettin, stefan.schmid}@de.bosch.com
[3]Bosch Research and Technology Center, Pittsburgh, PA, USA cory.henson@us.bosch.com

and forecaster based on 3D point clouds of a LiDAR sensor and dynamic maps of the environment. MutliPath [8] uses state-sequence anchors that correspond to model of trajectory distributions. At inference, the model predicts a discrete distribution over the anchors and regresses offsets from anchor waypoints along with uncertainties. These yield a Gaussian mixture at each time step. Zhao et al. [9] present the Target-driveN Trajectory (TNT) framework, that consists of three steps: predict the agent's potential target states into the future, generate trajectory state sequences conditioned on targets, estimate trajectory likelihoods and final compact trajectory predictions.

C. Graph-based Motion Predictions

A survey of deep learning-based vehicle behavior prediction for automated driving is provided by [10]. Diel et al. [11] compare the trajectory prediction performance of Graph Convolutional Networks (GCN) [12] with Graph Attention Networks (GAT) [13] and propose modifications to the task of vehicle behavior prediction. An attention-based spatio-temporal Graph Neural Networks (GNN) [14] for pedestrian trajectory prediction is proposed in [15]–[17]. Li et al. [18] present a graph based interaction-aware trajectory prediction (GRIP) approach. This models the interaction between the vehicles using a GCN and graph operations. Mo et al. [19] describe a GNN-RNN based approach for trajectory prediction, where the vehicles are modelled by an RNN and the interactions between them as a directed graph using a GNN. The output of the model is further processed by a Long Short Term Memory (LSTM) [20]. An enhanced version [21] uses both fixed and dynamic graphs for trajectory prediction. Frenet coordinate systems are well investigated for motion planning and trajectory generation [22]. The Frenet representation is adopted for multi-agent interaction aware trajectory prediction by a GNN-based solution [23]. Instead of a Cartesian coordinate system and global context images, it uses the pairwise relations between agents and pairwise context information. Fang et al. [24] propose an approach for long-term behavior prediction using ontology-based reasoning that considers interactions between traffic participants. Their likelihood behavior is inferred by a Markov logic network. Gao et al. [25] presented VectorNet, a vectorized definition of the scene where unified representations are learned from their vectorized form. The graphic extent of a road feature can be a point, a polygon, or a curve in geographic coordinates. A GNN is then used to incorporate the set of vectors, where each vector is treated as a node in the graph. Li et al. [26] present a spatio-temporal graph dual-attention network for multi-agent prediction and tracking. It incorporates relational inductive biases, a kinematic constraint layer and leverages both trajectory and scene context information.

III. Scene Representation

In this chapter, we describe the procedure for creating the scene description for a concrete application example. To analyze realistic traffic scenes and the resulting behaviors, we use openly available datasets, which we describe in detail in Section IV-A.

As the foundation for representing relations between traffic participants and describing the underlying traffic scenes, we apply the *Semantic Scene Model* (SSM), which is based on our previous work [27]. This model establishes relations between the traffic participants in the scene based on the given road topology. The traffic scene is mapped to a graph, in which the traffic participants are described by nodes and their relations by edges.

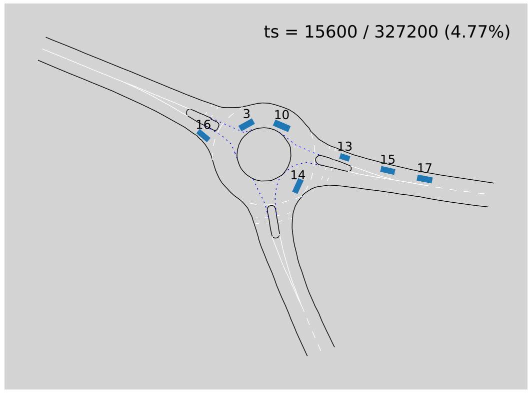

Fig. 1: Traffic scene seen from bird's eye view.

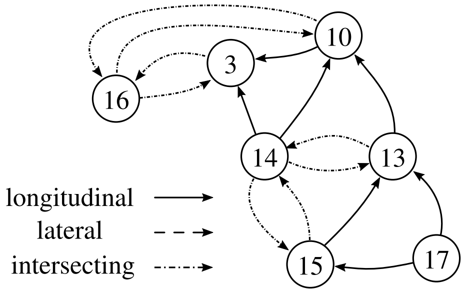

Fig. 2: Scene graph topology of the traffic scene of Figure 1.

Figure 1 illustrates an exemplary traffic scene from the INTERACTION dataset [28] and its corresponding topology of the derived scene graph is depicted in Figure 2. Each of the seven traffic participants is represented as a node. Each node v contains information about its classification (car, pedestrian, truck, ...) and its velocity. Edges between the nodes are of three different types: longitudinal, lateral and intersecting. This can be seen for example at the nodes or vehicles 10, 13, 14 and 15. If we consider the road layout in Figure 1, vehicles 13 and 15 follow vehicle 10. Vehicle 14 also follows vehicle 10, but at the same time, vehicle 13 and vehicle 15 have an intersecting relation with vehicle 14, since they drive on two roads that will merge eventually. This situation can be illustrated well by the generic graph description (Figure 2).

When converting the scene into the scene graph, the individual traffic participants are assigned to lanes. This allows the traffic scene to be viewed in the *Frenet* space, with the roads being the curves. In addition to the classification of the edge type, its probability and the distance along the roadway, between the two respective entities, is also included as an edge attribute (see table 1). When assigning every entity to a lane, the distance to the centerline of the lane and the angular difference between the entity's pose and the lane is calculated. These two values can be used to calculate the assignment probability $P(i)$ of an entity i to a lane. We define the probability $P(e_{ij})$ of an edge e_{ij} as follows:

$$P(e_{ij}) = P(i) \cdot P(j). \tag{1}$$

A more detailed overview of the calculation of the probability of each entity, especially with respect to the assignment to multiple lanes, is given in our previous work [27].

An example of the distance attributes, a schematic traffic scene and the corresponding edges is presented in Figure 3. If the relation between two entities is longitudinal (`lon`), the distance along the center line of the road to the projection of the entity is calculated, so that a lateral shift of the entity does not affect the distance (see $d_{2,4}$). The principle remains the same for merging lanes. Here, the distance along the road is also determined. The point where the two roads merge is called p_{int}. This means that the distance between vehicle (3) and vehicle (4) is noted as $d_{3,p}+d_{p,4}$. For two parallel roads, the distance between the two entities is determined similarly to longitudinal calculation. In this example, vehicle (2) is projected to the same lane as vehicle (1). Then the distance between both traffic participants is measured (compare $d_{1,2}$). For intersecting entities (`int`), only the distance from the tail-entity to the intersection point (here p_{int}) is considered (see $d_{2,p}, d_{3,p}$).

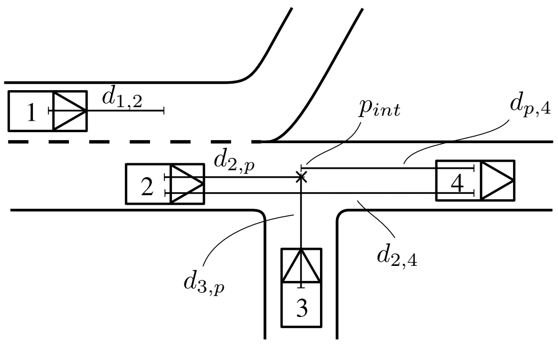

Fig. 3: Schematic traffic with distances between traffic participants

edge	edge probability	classification	distance
$e_{12} = (1, 2)$	$P(e_{12})$	`lat`	$d_{1,2}$
$e_{21} = (2, 1)$	$P(e_{21})$	`lat`	$-d_{1,2}$
$e_{23} = (2, 3)$	$P(e_{23})$	`int`	$d_{2,p}$
$e_{32} = (3, 2)$	$P(e_{32})$	`int`	$d_{3,p}$
$e_{24} = (2, 4)$	$P(e_{24})$	`lon`	$d_{2,4}$
$e_{34} = (3, 4)$	$P(e_{34})$	`lon`	$d_{3,p} + d_{p,4}$

Tab. 1: Structure of the edge attributes of the scene described in Figure 3.

IV. Implementation and Training

A. Dataset

Two different datasets are investigated: first, the PandaSet dataset [29], which provides short sequences (8s). Driving scenes are captured by a car equipped with multiple cameras and sensors. The captured scenes are divided into single images that depict the state of a scene at a specific point in time. The dataset comprises complex scenarios typically happening in urban areas like dense traffic and construction sites, as well as a variety of times of day and lighting conditions.

Secondly, the INTERACTION dataset [28], which is recorded by a drone at intersections and roundabouts. This dataset focuses on the behavior of vehicles on roads in different countries. In addition to the object tracks, the corresponding roads are provided as HD[1] maps. Both datasets are recorded at a frequency of $10\,Hz$.

B. Preprocessing

We initially convert both datasets to the same data format. The used output data format is the same as in the INTERACTION dataset. Here, all tracks of each traffic participant for each point in time are stored in a csv[2]-file. Each state of each traffic participant is uniquely defined by the track-id and the timestamp. In addition, the pose, the classification $\{Car, Pedestrian, Truck, ...\}$, the velocity vector and, in the PandaSet, the motion-state $\{Parked, Stopped, Driving\}$ are stored.

The focus of the PandaSet dataset is on object detection using Lidar sensors. Nevertheless, object lists with the individual tracked entities with their pose and classification can be read out. An important input variable is the velocity of the individual traffic participants. In the PandaSet dataset, this is calculated retrospectively based on the change in position over time, followed by a low pass filter to compensate measurement inaccuracies. The original dataset contains partially duplicate annotations for a given entity at a given timestamp. In order to remove the duplicates, we represent each entity by a rotated rectangular polygon and calculate *intersection over union* (IoU) between possible duplicates. Starting from an IoU of 0.2 and above, we consider those two to as conflicting. To identify the set of duplicates, we interpret conflicts as edges and involved entities as nodes in a conflict graph G_{conf}. G_{conf} is used to search for a maximum independent set of nodes such that all nodes are not connected, and delete all other entities. Furthermore, we removed all vehicles farther away than 80 meters from the ego vehicle since the measuring inaccuracy for far vehicles lead to extreme velocities and accelerations.

C. Net Architecture

For modelling the traffic scenes, we use a graph representation $G = (V, E)$ on which we can leverage a message passing approach. This allows for displaying

[1]high-definition
[2]comma-separated values

specific traffic participants and their relation in a machine-readable format. For further calculations, the graph is described by three data blocks. The node attributes are described in a $n \times |v|$ matrix, where n describes the number of nodes in the graph G and $|v|$ the number of attributes of a node $v \in V$. The edge information is stored in a $m \times |e|$ matrix, where m represents the number of edges and $|e|$ represents the number of edge attributes. The graph topology information, which nodes i, j are connected by which edge $e_{ij} \in E$, is stored in a $2 \times m$ matrix in the COO[3]-format.

Single Step Network

In this process, the main goal is to capture the behavior of a traffic participant based on its immediate (dynamic) environment. A graph convolution approach is used to learn the representation of the participant's environment, where each participant is denoted by a node. Thus, nodes are updated depending on the outgoing edges and the corresponding neighbors.

In general, the graph convolution approach consists of two steps, the message step (Equation (2)) and the update step (Equation (3)). In Figure 4, a schematic illustration of our graph convolution architecture is depicted. Our message passing approach is based on the work of Gilmer et al. [30].

$$m_i^{s+1} = \frac{1}{|N(i)|} \sum_{j \in N(i)} h_j^s \cdot \theta_{edge}(e_{ij}) \tag{2}$$

$$h_i^{s+1} = h_i^s \cdot \Theta + m_i^{s+1} \tag{3}$$

A message m_i^{s+1} for each node i is generated regarding the neighbor node's state h_j^s and the corresponding edge attributes e_{ij}, for all neighbors $j \in N(i)$. This is done by the propagation module P (see Figure 4) in which all outgoing edge attributes $h_{e_{ij}}$ of the root node i are extracted with the corresponding initial attributes h_j^0 of the neighbor node j.

The edge attributes of e_{ij} are fed through a neural network θ_{edge}, which consists of two fully connected layers and an interposed activation function. Subsequently, the current hidden state of the root node is updated by a learnable weight matrix Θ. By adding this state and the message m_i^{s+1} generated in the previous step, the state is updated to the new hidden state h_i^{s+1}.

Our network is constructed in such a way that several of these message passing steps ($0 \leq s \leq S$) can be connected in series. This propagates the information of the individual nodes further into the graph. In this work, the GNN architecture contains only one step of message passing ($s = S$) to demonstrate the principle function of the model and to keep it as simple as possible.

After the message passing step, each node in the graph has a hidden state depending on its neighbors and the corresponding relations. Finally, each h_i^S is mapped to a single floating point number which represents the predicted acceleration a_i via a neural network $\phi(\cdot)$, which consists of two fully connected layers with an interposed activation function.

[3]Coordinate list

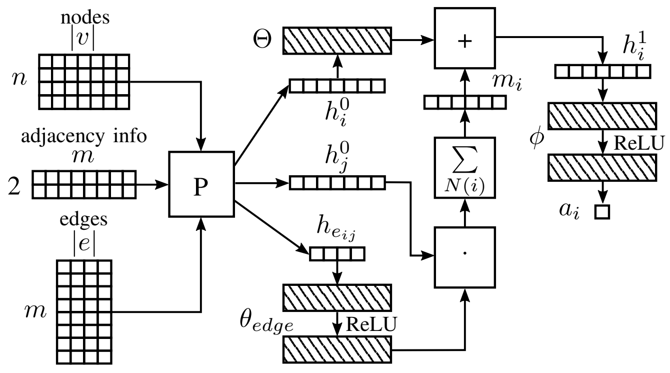

Fig. 4: Graph convolution architecture

$$a_i = \phi(h_i^S) \tag{4}$$

In our implementation θ_{edge} is an MLP with a hidden size of 32. Hidden states h have the size of 64. ϕ is also an MLP with a hidden size of 128. All MLPs have an interposed ReLU activation function.

All hyperparameters were optimized manually and empirically.

Recurrent Network

In addition to considering only a single traffic scene, we examine the influence of a temporal sequence of scenes on motion prediction. To capture the temporal information, we use an LSTM architecture (see Figure 5). Our temporal architecture is based on the works of Taheri et al. [31].

A sequence can contain up to T previous scenes. Hereby, T is practically only limited by the dataset and the scenes occurring in it. It should be noted that only the acceleration of the traffic participants that are in the scene at the starting point $t = 0$ are predicted. Traffic participants that are in the sequence but leave the scene earlier are still considered for the calculation of the historical states.

At first, we use the graph convolution $GConv$ architecture described above to generate a hidden state for each traffic participant ($H^1(t) = \{h_0^1(t), ..., h_n^1(t)\}$) with respect to its neighbors at a given time t (compare Figure 4 and Equation (2), Equation (3)). These states are fed into an LSTM block [20], which updates the LSTM encoding H^{LSTM}. This step is then repeated for every scene in the sequence to continuously update the LSTM encoding. The final encoding $H^{LSTM}(t)$ is then mapped to the corresponding acceleration values $a = \{a_0, a_1, ..., a_n\}$ for each node using ϕ.

V. Experiments

In this section, we evaluate the results of the two proposed network architectures. With the experiments, we want to show that the scene graphs can help to provide contextual information of a traffic scene in respect to behavioral prediction. For each entity i in $G(t)$, the acceleration a_i is set

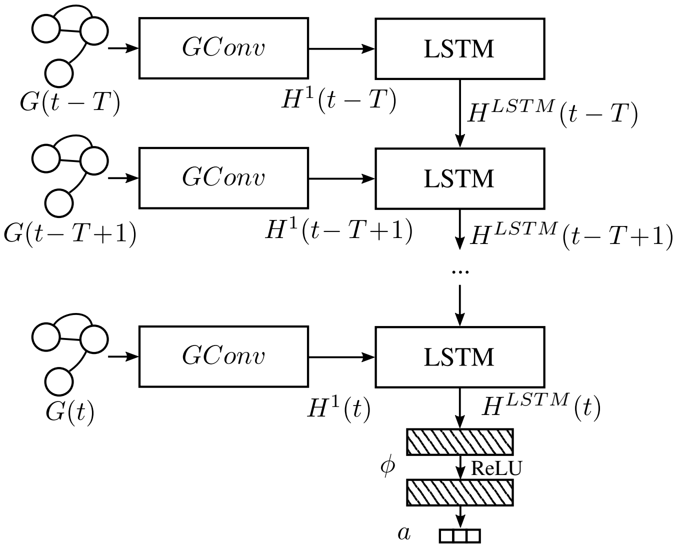

Fig. 5: Recurrent graph neural network architecture

as the respective label. The acceleration is calculated using the ground truth trajectories' velocities vel_i of entity i.

$$a_i^{label}(t) = \frac{vel_i(t+\delta_t) - vel_i(t)}{\delta_t} \tag{5}$$

It should be noted that acceleration prediction is only a single use case of the proposed approach. It intends to demonstrate the applicability and to verify the functionality. In our experiments, we set δ_t to 10 time steps, which corresponds to a time period of $1s$ in the datasets used. We evaluate our model, how well acceleration can be imitated using two different motion datasets.

We compare our model against a set of simple baselines for two reasons: First, we want to measure the benefit of a spatial semantic scene graph and second, to the best of our knowledge, there are no comparable approaches based on our input data. Many recent approaches use a rasterized bird's-eye image of the scenes [8], [32] or at least the coordinates of traffic participants as input. The work of Ma et al. [23] is still the closest related to our approach, but it also uses the image information for the edge attributes. Therefore, two straightforward baselines were used: Baseline *Mean* emulates a model that always outputs the mean value over the whole test dataset. Baseline *Zero* emulates a model that always outputs zero as the acceleration value, regardless of the input.

A. Training

We trained our model using Adam [33] with an initial learning rate of 10^{-3} and a batch size of 1. As a learning rate scheduler, we employed the *ReduceLROnPlateau* method with a factor of 0.1 and a patience value of 10. The number of epochs is set to a maximum of 200 with an early stopping mechanism after 25 epochs without validation improvement. In our experiments, the trainings stopped at roughly 75 epochs, depending on the model and dataset. Furthermore, we clipped the gradients to a global gradient norm of 1.

Our model is trained on the INTERACTION dataset with 13603 graphs, corresponding to 59972 entities. The model is subsequently tested with 11871 unseen graphs or behavior of 57413 traffic participants. The training set of PandaSet dataset consists of 4977 graphs and the testing set consists of 2923 graphs.

During the training phase, we used the loss defined in Equation (6) for the INTERACTION dataset and the loss defined in Equation (7) for the PandaSet dataset.

$$Loss = \frac{1}{n}\sum_{i\in V}\left|a_i - a_i^{label}\right| \tag{6}$$

$$Loss_{pandaset} = \left|a_0 - a_0^{label}\right| \tag{7}$$

Due to the inaccurate vehicle tracking described in Section IV-B, only the acceleration of the measurement vehicle a_0 is predicted for the PandaSet dataset and also used for training.

B. Results

The results are calculated in Table 2 for each model as linear error (L1-loss) or as squared error (MSE-loss). Compared to the baseline, the Single Step model (see Figure 4) is about 20% better at predicting accelerations. The Single Step model, with no edge data used, serves as an ablation. Only the node information is fed into the neural network, and all edge information is set to zero. Compared to the ablation model, our model performed about 12% better. The temporal network architecture performs significantly better. Compared to the baseline, better results are achieved between 70% and 73%, depending on the length of the sequence under consideration. Sequences with 5, 10 and 15 scenes are considered. The longer the time history, the better the acceleration of the individual traffic participants can be predicted. However, in our experiments there are only minimal changes in the errors, when considering 10 or 15 scenes per sequence.

To make the results comparable, we assume the calculated acceleration to be constant for future time steps and calculate a trajectory based on the ground truth path of a traffic participant. This would describe the deviation of the driven distance, which is comparable to the *Final Displacement Error* (FDE) of a trajectory comparison, as follows:

$$\text{FDE}_t = \left|a_i - a_i^{label}\right|\frac{t^2}{2} \tag{8}$$

With our Recurrent15 model of the INTERACTION dataset, this would give a $\text{FDE}_3 = 0.765\,m$. However, it should be kept in mind that so far our model only predicts the longitudinal motion component (accelerating/breaking) of a traffic participant and neglects the lateral component (steering).

If we compare the predictive performance of our model with those of state-of-the-art models [34], we perform in the lower range of common predictors. However, there are obvious reasons for that since we optimize only for an acceleration value and not for a future trajectory, i.e., the future course of the road is not included in the model. The curvature of the road, regulating road markings, such as a stop line, is also not considered yet. Instead, we show that relational information in a traffic scene helps to enable more meaningful predictions in general. Our approach complements the previous work and does not intend to replace it.

Model	Dataset	L1	MSE
Single Step	INTERACTION	0.494	0.451
Single Step no edge data	INTERACTION	0.552	0.538
Recurrent5	INTERACTION	0.271	0.14
Recurrent10	INTERACTION	0.188	0.098
Recurrent15	INTERACTION	**0.170**	**0.085**
Baseline Mean	INTERACTION	0.654	0.740
Baseline Zero	INTERACTION	0.599	0.622
Single Step	PandaSet	0.332	0.378
Recurrent5	PandaSet	0.259	0.294
Recurrent10	PandaSet	0.178	0.241
Recurrent15	PandaSet	**0.158**	**0.206**
Baseline Mean	PandaSet	0.347	0.489
Baseline Zero	PandaSet	0.34	0.489

Tab. 2: Evaluation results of the tested models

The relative trend of the prediction models between the INTERACTION and the PandaSet dataset is similar (see Table 2). However, comparatively inferior results are achieved with the PandaSet. One possible explanation is that the PandaSet dataset contains about 20 times less training samples than the INTERACTION dataset because in the latter, a future acceleration is estimated for each traffic participant and not only for the ego vehicle. In addition, the PandaSet contains significantly larger numbers of participants per traffic scenes. On average, it contains about 24 traffic participants, each with 12 relations to their neighbors. In the INTERACTION dataset, there are on average about 5 vehicles in the scene and each object has about 2 relations. This reduces noise and allows the model to focus more precisely on individual traffic participants.

VI. Conclusion

In this work, we present two models that are capable of predicting the movement of traffic participants based on spatial semantic scene graphs. By means of the scene graphs, semantic relationships among traffic participants are encoded. Through the ablative analysis, it is shown that these play an important role in the behavior of the traffic participants. In addition, the temporal model empirically shows that the preceding scenes provide important contextual information for the prediction task, and lead to significantly better results. With the inclusion of previous scenes, up to 46% better results are achieved for the PandaSet and up to 73% better results are achieved for the INTERACTION dataset compared to the baseline.

In the future, more semantic relation types will be investigated. While the presented model is able to represent various types of traffic participants and their different types of relations, e.g., pedestrians are not included yet, and only spatial relations are considered in our experiments. In the upcoming work, the influence of other traffic participants will be investigated. In addition, the used scene graph is limited to a few attributes, which will be extended in the future by further semantic information, possibly from external sources.

Moreover, we will also investigate to what extent the contextual information of the scene extracted by the presented model can be used to improve a state-of-the-art image-based trajectory predictor.

References

[1] S. Ulbrich, T. Nothdurft, M. Maurer, and P. Hecker, "Graph-based context representation, environment modeling and information aggregation for automated driving," in *IEEE Intelligent Vehicles Symposium Proceedings*, 2014.

[2] C. Henson, S. Schmid, A. T. Tran, and A. Karatzoglou, "Using a knowledge graph of scenes to enable search of autonomous driving data," in *Inter. Semantic Web Conf. ISWC - Satellite Tracks*, 2019.

[3] L. Halilaj, I. Dindorkar, J. Lüttin, and S. Rothermel, "A knowledge graph-based approach for situation comprehension in driving scenarios," in *Extended Semantic Web Conference ESWC*, 2021.

[4] F. Leon and M. Gavrilescu, "A review of tracking, prediction and decision making methods for autonomous driving," *ArXiv*, vol. abs/1909.07707, 2019.

[5] A. Geiger, M. Lauer, C. Wojek, C. Stiller, and R. Urtasun, "3d traffic scene understanding from movable platforms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2014.

[6] J. Hong, B. Sapp, and J. Philbin, "Rules of the road: Predicting driving behavior with a convolutional model of semantic interactions," *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8446–8454, 2019.

[7] S. Casas, W. Luo, and R. Urtasun, "Intentnet: Learning to predict intention from raw sensor data," *ArXiv*, vol. abs/2101.07907, 2018.

[8] Y. Chai, B. Sapp, M. Bansal, and D. Anguelov, "Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction," in *CoRL*, 2019.

[9] H. Zhao, J. Gao, T. Lan, C. Sun, B. Sapp, B. Varadarajan, Y. Shen, Y. Shen, Y. Chai, C. Schmid, C. Li, and D. Anguelov, "Tnt: Target-driven trajectory prediction," in *CoRL*, 2020.

[10] S. Mozaffari, O. Y. Al-Jarrah, M. Dianati, P. Jennings, and A. Mouzakitis, "Deep learning-based vehicle behaviour prediction for autonomous driving applications: A review," *ArXiv*, vol. abs/1912.11676, 2019.

[11] F. Diehl, T. Brunner, M. Truong-Le, and A. Knoll, "Graph neural networks for modelling traffic participant interaction," *IEEE IV*, pp. 695–701, 2019.

[12] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *ICLR*, 2017.

[13] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *6th International Conference on Learning Representations, ICLR*, 2018.

[14] F. Scarselli, M. Gori, A. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Transactions on Neural Networks*, vol. 20, pp. 61–80, 2009.

[15] H. Zhou, D. Ren, H. Xia, M. Fan, X. Yang, and H. Huang, "Ast-gnn: An attention-based spatio-temporal graph neural network for interaction-aware pedestrian trajectory prediction," *Neurocomputing*, vol. 445, pp. 298–308, 2021.

[16] C. Yu, X. Ma, J. Ren, H. Zhao, and S. Yi, "Spatio-temporal graph transformer networks for pedestrian trajectory prediction," in *ECCV*, 2020.

[17] S. Haddad, M. Wu, H. Wei, and S. K. Lam, "Situation-aware pedestrian trajectory prediction with spatio-temporal attention model," *ArXiv*, vol. abs/1902.05437, 2019.

[18] X. Li, X. Ying, and M. C. Chuah, "Grip: Graph-based interaction-aware trajectory prediction," *IEEE Intelligent Transportation Systems Conference (ITSC)*, pp. 3960–3966, 2019.

[19] X. Mo, Y. Xing, and C. Lv, "Graph and recurrent neural network-based vehicle trajectory prediction for highway driving," *IEEE Inter. Intell. Transportation Systems Conference (ITSC)*, pp. 1934–1939, 2021.

[20] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997. [Online]. Available: https://direct.mit.edu/neco/article/9/8/1735-1780/6109

[21] X. Li, X. Ying, and M. C. Chuah, "Grip++: Enhanced graph-based interaction-aware trajectory prediction for autonomous driving," *arXiv: Computer Vision and Pattern Recognition*, 2019.

[22] M. Werling, J. Ziegler, S. Kammel, and S. Thrun, "Optimal trajectory generation for dynamic street scenarios in a frenét frame," *IEEE Inter. Conference on Robotics and Automation*, pp. 987–993, 2010.

[23] H. Ma, Y. Sun, J. Li, and M. Tomizuka, "Multi-agent driving behavior prediction across different scenarios with self-supervised domain knowledge," in *IEEE International Intelligent Transportation Systems Conference (ITSC)*, 2021, pp. 3122–3129.

[24] F. Fang, S. Yamaguchi, and A. Khiat, "Ontology-based reasoning approach for long-term behavior prediction of road users," *ITSC*, 2019.

[25] 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," *IEEE/CVF Conf. on Comp. Vision and Pattern Recognition (CVPR)*, pp. 11 522–11 530, 2020.

[26] J. Li, H. Ma, Z. Zhang, J. Li, and M. Tomizuka, "Spatio-temporal graph dual-attention network for multi-agent prediction and tracking," *ArXiv*, vol. abs/2102.09117, 2021.

[27] M. Zipfl and J. M. Zöllner, "Towards traffic scene description: The semantic scene graph," *CoRR*, 2021. [Online]. Available: https://arxiv.org/abs/2111.10196

[28] 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.

[29] P. Xiao, Z. Shao, S. Hao, Z. Zhang, X. Chai, J. Jiao, Z. Li, J. Wu, K. Sun, K. Jiang *et al.*, "Pandaset: Advanced sensor suite dataset for autonomous driving," in *IEEE International Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2021, pp. 3095–3101.

[30] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural Message Passing for Quantum Chemistry," *arXiv:1704.01212 [cs]*, Jun. 2017, arXiv: 1704.01212. [Online]. Available: http://arxiv.org/abs/1704.01212

[31] A. Taheri and T. Berger-Wolf, "Predictive temporal embedding of dynamic graphs," in *Proceedings of the IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. ACM, Aug. 2019, pp. 57–64.

[32] M. Bansal, A. Krizhevsky, and A. Ogale, "ChauffeurNet: Learning to Drive by Imitating the Best and Synthesizing the Worst," *arXiv:1812.03079 [cs]*, Dec. 2018, arXiv: 1812.03079. [Online]. Available: http://arxiv.org/abs/1812.03079

[33] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[34] [Online]. Available: http://challenge.interaction-dataset.com/leader-board