

Towards Complete Input Representations for Vehicle Trajectory Prediction Models

Marcel Milich^{1,2}, Trung-Kien Tran¹

¹*Bosch Center for Artificial Intelligence*

²*Institute for Artificial Intelligence, University of Stuttgart*

Abstract

Predicting accurate trajectories for traffic participants is a crucial task in the autonomous driving field. Especially in complex scenarios like urban intersections we need to anticipate intentions and positions of surrounding agents in order to plan a safe and smooth maneuver for an ego-vehicle. Trajectory prediction remains a challenging task due to a diverse set of influencing factors like the topology of the street, traffic rules and physical dynamics but also stochastic intentions of the agents and their mutual influences. The performance of prediction models depends on both the input representation, i.e. the *encoding* of the scene, and the prediction, i.e. the *decoding* of trajectories. Recent works have mainly focused on the *decoding* part. Conversely, in this work we investigate the input representations of trajectory prediction models. They can roughly be divided into raster- and vector-based methods. Raster-based methods represent the traffic scene as bird's-eye-view images whereas vector-based methods represent all elements as sets of points. We explain the pros and cons of both methods and carry out preliminary experiments to enhance the representations by 1) enriching the over-simplistic vector representation and 2) combining both representations. Our finding results on the popular NuScenes benchmark indicate that very simple representations are actually sufficient for state-of-the-art results. This is in contrast to real-world driving where more information influences driver decisions. Consequently, it might be the call for more diverse and representative datasets and/or more expressive models.

Keywords

Vehicle Trajectory Prediction, Autonomous Driving, Graph Neural Networks

1. Introduction

Predicting accurate trajectories of traffic participants is a crucial task in the autonomous driving field. Especially in complex scenarios like urban intersections we need to anticipate intentions and positions of surrounding agents in order to plan a safe and smooth maneuver for an ego-vehicle. Vehicle trajectories are influenced by the topology of the street, traffic rules and physical dynamics but also by the unknown (and thus stochastic) intentions of the agents and their mutual interactions. Through this combination of different factors vehicle trajectory prediction remains a challenging task. Almost monthly new entries occur on the leaderboards of the major benchmarks like NuScenes¹. A strong focus in recent years has been on the *decoding* of the trajectories: Given an embedded agent, generate a diverse set of high quality trajectories. However, less focus is on the *representation* and *embedding* of the traffic scene. Often approaches from existing work are being used without developing them further [1]. Regarding the input

Edge AI meets Swarm Intelligence Technical Workshop, September 18, 2024, Dubrovnik, Croatia

✉ marcel.milich@de.bosch.com (M. Milich); trungkien.tran@bosch.com (T. Tran)



© 2024 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

¹<https://www.nuscenes.org/>

representation, one can roughly divide vehicle trajectory prediction models into two categories: Raster-based and vector-based methods.

In raster-based methods the traffic scene (trajectories, HD map,...) is represented in form of bird's-eye-view images (see figure 1) from which CNN-based methods compute hidden features for the vehicles and their interactions [2, 3, 4]. Images deliver a very complete and detailed representation of the traffic scene. Further CNNs are good at capturing local dependencies and bring other advantages like invariance to translation, equivariance to rotation and the possibility to use strong off-the-shelf vision models. However, they struggle at capturing global interactions, neglect geometric dependencies such as connectivities along lanes and are expensive [5, 6, 7]. That's why most of today's methods use vectorized representations of the trajectories and HD maps [8, 9, 10]. The idea is to represent all elements as sets or sequences of points. Trajectories and traffic elements are defined by polylines that are closely approximated by sets of points. The point sets are typically encoded with some RNN-architecture to initialize embeddings for either a GNN- or Transformer-based encoder. For GNNs one defines a graph over the point sets. Traffic elements and vehicles form nodes in the graph and edges model the respective interactions. Most graph representations only consider the centerline information in a so called *lane graph*. Vectorized representations are very efficient, good at capturing geometric structures and the mainstream input representation of SOTA models. However, they are typically rather simple (e.g. only the lane centerlines), neglecting potentially valuable information of the traffic scene.

For this reason we investigate the following question: **1)** Can a more complex vectorized representation improve the performance of a SOTA trajectory prediction model? Further, both methods have their pros and cons. It could be beneficial to combine them. There exist approaches in this direction [11, 12]. However, none of them utilize the lane graph representation, which many SOTA trajectory prediction models rely on. Thus, **2)** Can a joint rasterized and vectorized lane graph representation combine the best of both worlds and prove beneficial for the performance?

2. Preliminaries

2.1. Trajectory Prediction

Let $t_c, T \in \mathbb{N}$, we are given a time horizon $1, \dots, t_c, \dots, T$, with current time t_c , past $[1, t_c]$ and future $[t_c + 1, T]$. Further we have a set of N agents (cars, trucks, motorcycles, pedestrians) represented by 2D coordinates $x_1^a, \dots, x_T^a \in \mathbb{R}^2$, $a \in N$. We write $X_{1:T} \in \mathbb{R}^{N \times T \times 2}$ for all trajectories of all agents in matrix form. Lastly there is map information M representing street elements (lanes, crosswalks, traffic lights ...), usually given in form of an HD map.

The task is: Given the past trajectories $X_{1:t_c} \in \mathbb{R}^{N \times t_c \times 2}$ and M , predict the future trajectory $X_{t_c+1:T}^a$ of one specific agent $a \in N$ or the future trajectories of all agents $X_{t_c+1:T}$. Since there is not only one sensible continuation of a traffic scene (rather a probability distribution) it is standard to define a number of modes $K \in \mathbb{N}$ (e.g. $K = 5, 10$) and predict K futures.

Given a multi-modal prediction $\hat{X}_{t_c+1:T}^{k,a}$ for agent a and $k = 1, \dots, K$ the most common

evaluation metric is the *minimum average displacement error*:

$$\text{minADE}_K = \min_{k=1,\dots,K} \sum_{t=t_c+1}^T \frac{1}{C} \left\| \hat{x}_t^{a,k} - x_t^a \right\|, \quad (1)$$

where $C = T - (t_c + 1)$.

2.2. Raster Representation

In a raster representation a stack of bird's-eye-view images is used as input and has the form $I \in \mathbb{R}^{H \times W \times xC}$, where H and W are height and width of the image and C is the number of channels. Each channel can represent a different type of input, e.g. agent trajectories, walkways, drivable area or RGB values.

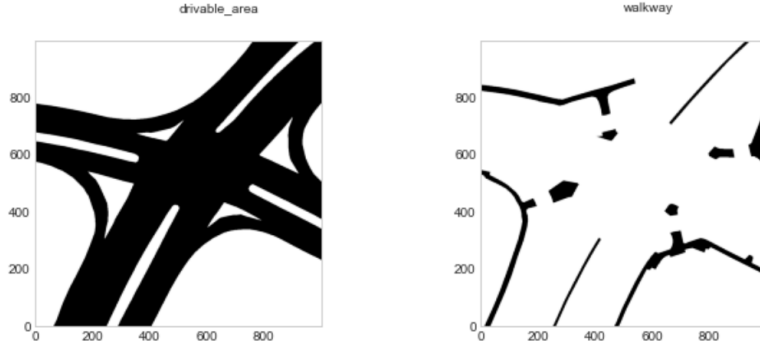


Figure 1: Two rasterized input channels from an intersection in the NuScenes dataset.

2.3. Vectorized Representation and Lane Graph

Road elements and trajectories are represented as a set or sequence of points (see Figure 2). E.g. let $x_1, \dots, x_{10} \in \mathbb{R}^2$ be the coordinates of the centerline of a segment of some lane. The segment has as initial representation the set (or sequence) $\{x_1, \dots, x_{10}\}$ plus some attributes (e.g. number of outgoing lanes, number of neighboring lanes, ...). A popular approach in recent years is the lane graph: Each lane segment defines a node in a graph. Node attributes include (an embedding of) the respective vector sets, and edges are defined by the flow of the traffic, i.e. a lane segment is naturally connected to its successor and neighboring segments [10, 5, 8].

3. Experiments

Figure 2 illustrates the simplification from the complete input to a vectorized representation which motivated our first question: Can a more detailed vectorized representation improve the performance of a SOTA model?

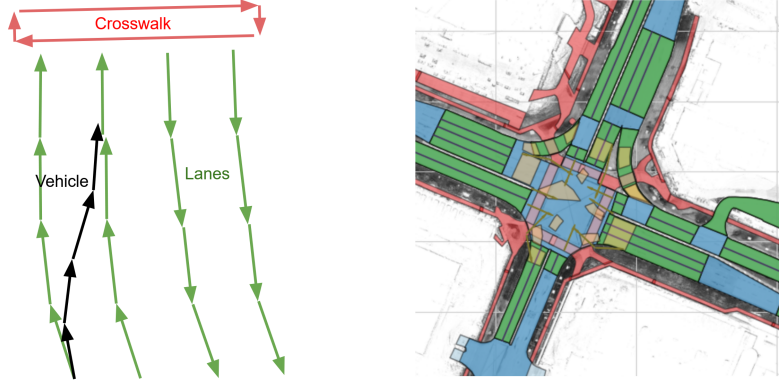


Figure 2: Vectorized representation on the left. As comparison all layers of a NuScenes map on the right side.

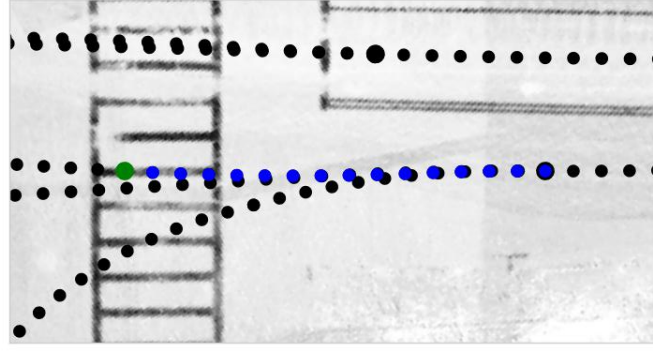


Figure 3: RGB image of one lane node, the point set of the node is colored green (start point) and blue.

To this end we consider PGP [8]: PGP is a competitive model on the NuScenes leaderboard², comes with a well written code base³ and employs a lane graph. The lane graph consists only of lane centerlines, i.e. nodes are centerline segments. The node attributes indicate whether a centerline intersects with a pedestrian crossing or a stop line. Edges are of type successor and neighbor.

Without going into detail, to compute embeddings of lanes and agents PGP encodes the vector sets of lane nodes and the trajectories with GRUs followed by some attention and message passing mechanism. For decoding, candidate paths are sampled from the lane graph and refined to a final output trajectory of an agent.

We extend PGP's lane graph by adding nodes for geometric objects other than the centerlines of the lanes (PGP+polygons). In particular, we add most of the polygonal elements depicted in

²<https://eval.ai/web/challenges/challenge-page/591/leaderboard/1659>

³<https://github.com/nachiket92/PGP>

Figure 2 on the right, including stop areas (yellow), ped crossings (light red), entire lanes (green) and road segments (blue). The representation now contains the exact extent of the elements, e.g. also the lane width and exact positions of ped crossings etc. We embed the polygonal structures (represented as sets of points) with a GNN. This embedding is the feature for the additional node in the lane graph which is connected to other nodes according to the topology of the street.

For the second question of a lane graph with a joint rasterized representation we simply add a raster feature to each of the lane nodes (PGP+raster). The raster feature is just an image of a rectangular area containing the respective centerline points and oriented in the according direction (see Figure 3). We use a pretrained ResNet-50 to embed the images and concatenate them to the embeddings of the lane nodes.

We train both approaches from scratch, optimize PGP’s hyperparameters and evaluate on the NuScenes prediction split.

Table 1

Lower is better. Performance on the NuScenes prediction dataset.

Method	minADE ₅
PGP	1.27
PGP+polygons	1.32
PGP+raster	1.36

As we can see in table 1 both PGP+polygons and PGP+raster worsen the performance. Even though they have additional information the generalization is worse. It seems sufficient to have a simplistic representation of the map to achieve good results on NuScenes prediction which is also reflected by the online leaderboard. In most cases the centerline information is all one needs to predict accurate trajectories. Even if the additional information could help in some edge cases, it generally seems to confuse the model. For PGP+polygons it could be that the deeper and more complex graph impairs the message passing which is a known problem of GNNs [13]. PGP+raster actually made the model strongly overfit. We had to reduce the embedding dimensions and add dropout on the raster feature to get the score from table 1.

4. Conclusion

In this work we tried to extend the sparse input representation for the trajectory prediction model PGP. Our preliminary experiments confirm that a simplistic representation is favorable to achieve SOTA results on NuScenes and other public trajectory prediction benchmarks. As human drivers we know this is not the whole picture. For example no traffic light state, speed limit or lane width information are considered in PGP and most models of the NuScenes leaderboard. All factors that strongly influence a driver’s decisions. In other words, the problem could be the dataset. Hence, even though the methods did not improve the performance on

the NuScenes benchmark it is worthwhile to further investigate how to add more valuable knowledge into trajectory prediction models. For this purpose one probably needs a more diverse dataset containing additional information and covering specific scenarios (like "green light turning red"), where this information is relevant, in a more extensive way. In future work we plan to test our hypothesis on other models than PGP. A Promising direction is also the connection of geometric trajectory prediction models and large language models that potentially can add reasoning abilities to the models which could help to fully leverage a more detailed input representation.

Acknowledgments

This work was funded by the European project under the grant agreement No. 101092908 (SmartEdge).

References

- [1] J. Gao, C. Sun, H. Zhao, Y. Shen, D. Anguelov, C. Li, C. Schmid, Vectornet: Encoding hd maps and agent dynamics from vectorized representation, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 11522–11530. doi:10.1109/CVPR42600.2020.01154.
- [2] T. Gilles, S. Sabatini, D. V. Tsishkou, B. Stanciulescu, F. Moutarde, Gohome: Graph-oriented heatmap output for future motion estimation, 2022 International Conference on Robotics and Automation (ICRA) (2021) 9107–9114. URL: <https://api.semanticscholar.org/CorpusID:237263205>.
- [3] M. Schäfer, K. Zhao, M. Bühren, A. Kummert, Context-aware scene prediction network (caspnet), in: 2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC), 2022, pp. 3970–3977. doi:10.1109/ITSC55140.2022.9921850.
- [4] N. Djuric, V. Radosavljevic, H. Cui, T. Nguyen, F.-C. Chou, T.-H. Lin, N. Singh, J. Schneider, Uncertainty-aware short-term motion prediction of traffic actors for autonomous driving, in: 2020 IEEE Winter Conference on Applications of Computer Vision (WACV), 2020, pp. 2084–2093. doi:10.1109/WACV45572.2020.9093332.
- [5] W. Zeng, M. Liang, R. Liao, R. Urtasun, Lanercnn: Distributed representations for graph-centric motion forecasting, 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2021) 532–539. URL: <https://api.semanticscholar.org/CorpusID:231632882>.
- [6] Z. Wang, J. Guo, Z. Hu, H. Zhang, J. Zhang, J. Pu, Lane transformer: A high-efficiency trajectory prediction model, IEEE Open Journal of Intelligent Transportation Systems 4 (2023) 2–13. doi:10.1109/OJITS.2023.3233952.
- [7] J. Lu, W. Zhan, M. Tomizuka, Y. Hu, Generalizability analysis of graph-based trajectory predictor with vectorized representation, in: 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2022, pp. 13430–13437. doi:10.1109/IROS47612.2022.9981096.
- [8] N. Deo, E. Wolff, O. Beijbom, Multimodal trajectory prediction conditioned on lane-graph

- traversals, in: A. Faust, D. Hsu, G. Neumann (Eds.), *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, PMLR, 2022, pp. 203–212. URL: <https://proceedings.mlr.press/v164/deo22a.html>.
- [9] M. Liu, H. Cheng, L. Chen, H. Broszio, J. Li, R. Zhao, M. Sester, M. Y. Yang, Laformer: Trajectory prediction for autonomous driving with lane-aware scene constraints, in: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2024, pp. 2039–2049.
 - [10] M. Liang, B. Yang, R. Hu, Y. Chen, R. Liao, S. Feng, R. Urtasun, Learning lane graph representations for motion forecasting, in: A. Vedaldi, H. Bischof, T. Brox, J.-M. Frahm (Eds.), *Computer Vision – ECCV 2020*, Springer International Publishing, Cham, 2020, pp. 541–556.
 - [11] S. Kumar, Y. Gu, J. Hoang, G. C. Haynes, M. Marchetti-Bowick, Interaction-based trajectory prediction over a hybrid traffic graph, in: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 5530–5535. doi:10.1109/IROS51168.2021.9636143.
 - [12] T. Salzmann, B. Ivanovic, P. Chakravarty, M. Pavone, Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data, in: A. Vedaldi, H. Bischof, T. Brox, J.-M. Frahm (Eds.), *Computer Vision – ECCV 2020*, Springer International Publishing, Cham, 2020, pp. 683–700.
 - [13] T. K. Rusch, M. M. Bronstein, S. Mishra, A survey on oversmoothing in graph neural networks, 2023. URL: <https://arxiv.org/abs/2303.10993>. arXiv:2303.10993.