



# Exploring the Road Graph in Trajectory Forecasting for Autonomous Driving

Rémy Sun, Diane Lingrand, Frédéric Precioso  
Université Côte d’Azur, Inria, CNRS, I3S, Maasai, Nice, France  
{firstname.lastname}@univ-cotedazur.fr

## Abstract

*As Deep Learning tackles complex tasks like trajectory forecasting in autonomous vehicles, a number of new challenges emerge. In particular, autonomous driving requires accounting for vast a priori knowledge in the form of HDMaps. Graph representations have emerged as the most convenient representation for this complex information. Nevertheless, little work has gone into studying how this road graph should be constructed and how it influences forecasting solutions. In this paper, we explore the impact of spatial resolution, the graph’s relation to trajectory outputs and how knowledge can be embedded into the graph. To this end, we propose thorough experiments for 2 graph-based frameworks (PGP [6], LAformer [14]) over the nuScenes [1] dataset, with additional experiments on the LaneGCN [13] framework and Argoverse 1.1 [2] dataset.*

## 1. Introduction

The last decade has seen Deep Learning revolutionize how we approach fundamental problems like Computer Vision [11] and Natural Language Processing [24]. As such, it is perhaps no surprise that neural architectures are now spearheading a new generation of techniques that will directly impact our daily lives (e.g. Chatbots [3], Fraud detection systems [19] and Autonomous Vehicles [12]). Some, like Autonomous Vehicles, are even now motivating new inquiries into difficult questions [12, 7].

One of the core issues of Autonomous Vehicles, trajectory forecasting [20, 7], has recently emerged as a difficult and complex task for neural networks. At its core, this task aims to forecast possible trajectories of pedestrians or vehicles. In its most common iterations, it takes as inputs both bounding boxes of active agents (pedestrians, cars, bikes, ...) pre-extracted from sensor data and known map information about the roads [12].

Interestingly, trajectory forecasting techniques have proven heavily reliant on HDMaps [15] that encode knowl-

Figure 1. **Investigating the Road Graph.** We propose to investigate 3 aspects of the Road Graph used in trajectory forecasting: its resolution, how it can correct predictions, and how persistent knowledge can be embedded in it.

edge about the network of roads, signs, and objects, the agents might interact with. HDMaps are notable both due to the high complexity of the information they encode, and due to the fact they require encoding prior human knowledge into a neural system which is a notoriously difficult problem. Therefore, the question of how to encode this information has been the focus of much debate in the literature [20, 7, 13].

Graph (or vectorial) representations of the road network have slowly emerged as the consensus solution over the last few years [7, 13]. Contrarily to previous solutions like rasterized representations [20], road graphs depicting the relation between the various lanes and agents offer a flexible and powerful input that can be directly fed to a graph neural network [25]. Better, the road graph structure can be leveraged by teaching the network to traverse the graph before yielding a continuous trajectory prediction [6].

It is therefore surprising that the particulars of this road graph have seen very little study. While recent works propose general heuristics regarding the optimal spatial resolution used by the graph [17] or the information to encode

in the nodes [6], we can find no specific work directly exploring these questions. Indeed, most existing work merely takes advantage of the graph structure to acquire good features and instead focuses on feature extraction [6, 14] or decoding schemes [10].

In this paper, we explore 3 possible ways this road graph can interact with trajectory forecasting solutions: the impact of spatial resolution, the graph’s relation to trajectory outputs and how persistent or static knowledge can be embedded into the graph. The first is a straightforward exploration of how the number of nodes in the road graph directly impacts the model’s understanding of distances and of the world. The second aims to study how relating the output to the graph can allow for even better trajectories. The last point explores how the model can acquire “persistent” knowledge about the characteristics of cities and lanes.

**Contributions** We propose these 4 contributions:

- We investigate the spatial resolution of the road graph in relation to trajectory forecasting.
- We check how examining output predictions with regards to the graph can help further improve the final trajectories given by models.
- We explore a few ways to embed more persistent knowledge into the graph that can recur across multiple scenarios (e.g. knowledge on a city).
- We evaluate our propositions experimentally on two well-established graph based frameworks (PGP [6], LAformer [14]) over the nuScenes [1] dataset, with additional experiments on the LaneGCN [13] framework and the Argoverse 1.1 [2] dataset.

## 2. Context

Before elaborating in more details regarding the construction of the Road Graph in Sec. 3, we start by offering here some brief context on trajectory forecasting and HDMap representations.

**Trajectory forecasting** Trajectory forecasting [20, 7] - ie, the task of predicting possible trajectories for a given agent (pedestrian or vehicle) - is in itself a fairly new problem for autonomous driving with the recent push for fully automated vehicles [9]. As opposed to more traditional autonomous driving problems like planning [21], forecasting is inherently a multimodal problem that requires outputting a vast spectrum of possible solutions to model most likely possibilities (see Fig. 2). The exact solutions developed are beyond the scope of this paper but they all require combining traditional perceptual inputs (detected agent trajectories [6] or raw records [8]) with symbolic knowledge bases (HDMaps).

Figure 2. **Trajectory forecasting** takes complex inputs (both perceptual detections and symbolic knowledge) to predict possible trajectories for a given vehicle of interest.

**Map representation** In order to help navigate the complexities of the real world, known information about the road network is typically embedded in rich HDMaps provided to the forecasting systems [9]. This map information is however difficult to feed directly to a deep architecture. It must therefore be processed into a suitable representation. While converting the map to an image - a process known as rasterization [20] - is a convenient solution that has been used by early solutions, such representations lose much in translation [7].

As such, a trend has emerged to modelize the map as a set of vectorial (or graphical) entities with different relationships to each other [7, 13]. VectorNet [7] first introduced this idea in 2020 by modeling each lane in a map as a node and learning an embedding of the map through an attention mechanism between the nodes. This representation has since been refined into a more detailed road graph that mimics the road network by LaneGCN [13].

The information contained in HDMaps has proven crucial to the functioning of modern trajectory forecasting solutions. Indeed, a number of graph based methods like PGP [6] and LAformer [14] explicitly reason on the road graph and therefore cannot be used without map information. Even for a non graph based method (e.g. mmTransformer [16]), [15] shows a 10% drop in performance for most commonly used metrics. There is therefore a pressing need to better understand how Road Graphs leverage HDMaps information to improve in current Deep trajectory forecasting systems.

## 3. Meet the road graph

As this study focuses on exploring the impact of the Road Graph on trajectory forecasting methods, we provide here a brief introduction both of the graph proper (Sec. 3.1) and of how it is leveraged in current methods (Sec. 3.2).

Figure 3. **Construction of a typical Road Graph.** We discretize lane centerlines into segments before converting the segments into nodes. The nodes are then connected to reflect the HDMap topology.

### 3.1. Building a road graph

The Road Graph is meant to encode all the complex information encoded in HDMaps in a way readily understandable to a neural network. As such, quite a bit of preprocessing must go into its construction and a number of choices must be made by the implementer as to what needs to be encoded and how. For simplicity's sake, we mostly discuss here the road graph as it is proposed in methods that rely heavily on it like PGP [6].

**Notations** In the following, we use common notations in the literature that we introduce now. We refer to lanes centerlines - ie, the line at the center of a drivable lane - by a sequence of vectors  $C_i = \{C_i^0, \dots, C_i^l\}$ . We denote agents by sequences of descriptive vectors  $A_i = \{A_i^{-T_0}, \dots, A_i^0\}$  describing its past history for some time horizon  $T_0$ . We characterize the road graph  $G = (V, E)$  by its set of nodes  $V$  and its the set of edges in the graph  $E$ .

At a very high level, the road graph encodes the centerlines  $C$  as nodes  $V$ , and includes in  $E$  edges that indicate whether it is possible to navigate from one node to the other according to the HDMap. Agents  $A$  can be added to the nodes of this graph and linked to lane nodes they are close to spatially. This exact representation is used in some works of the literature like the seminal VectorNet [7], but it lacks in detail and does not quite reflect the topology of the problem [13].

In practice, many modern methods **discretize lane centerlines**  $C$  into segments such that a centerline  $C_i = \{\hat{C}_{i,0}, \dots, \hat{C}_{i,s}\}$  is comprised of  $s$  sub-centerline segments  $\hat{C}_{i,j} = \{\hat{C}_{i,j}^0, \dots, \hat{C}_{i,j}^l\}$  (see Fig. 3). From this, we can **extract nodes** - one node per segment - such that a node in the road graph denotes a spatial location in the world, stretching across a few meters on a lane. These segments  $\hat{C}_{i,j}$

typically get represented as a sequence of vectors (to then be encoded by some neural layer like a GRU [4]), with each vector representing a point on the segment. What each of those "point" vector  $\hat{C}_{i,j}^k$  contain depends on the method, but a typical implementation encodes spatial coordinates  $x, y$ , yaw angle as well as some semantic flags indicating whether a crosswalk or stop sign is near.

We then **connect lane segment nodes**  $V$  by edges to reflect the road network on the HDMap. Typically, the set of edge  $E$  contains both edges connecting successive segments in a given lane (" $E_{succ}$ ") but also connections between segment nodes belonging to different lanes (" $E_{prox}$ "). The generally adopted heuristic is to connect two nodes if a vehicle could legally move from one node to the other. Concretely, this could mean connecting two segment nodes that are within a certain radius of each other spatially and have similar yaw orientations (to avoid allowing impossible U-turns).

The end result is shown on Fig. 3, where the original network of HDMap lane centerlines is faithfully represented by the constructed road graph. This graph provides a lightweight representation that can easily be processed by standard neural architectures like GNNs. Contrarily to rasterized representations, it allows for a clear and unambiguous depictions of complex situations where lanes intersect and overlap (e.g the blue and violet lanes do not obscure other lanes). As such, it has emerged over the last few years as the de facto solution for HDMap representations in Autonomous Driving trajectory forecasting.

While this fundamental definition of the Road Graph does not include agents, agents  $A$  of course play a role in most methods and can be regarded as additional nodes in a heterogeneous graph. In such cases, these additional agent nodes are connected to segment nodes that come within a certain radius of the agent node. This can be explicitly formulated as such [14] or processed as a separate type of

Figure 4. **Graph traversal for trajectory forecasting.** After encoding a meaningful road graph, methods like PGP [6] can sample likely traversal on the graph. These traversals can then be converted to real world trajectories.

graph relation [6].

### 3.2. Using a road graph in practice

To better understand the utility of the now defined Road Graph representation, we quickly illustrate here how it can be leveraged by forecasting solutions in the literature.

**Baseline use: GNN features** At the very least, the graph structure of the Road Graph can be processed by Graph Neural Networks [25] to yield relevant features [7, 13]. These features can then be used to enrich agent representations or be directly fed to a trajectory generator. This single improvement has led to significant improvements in performance over previous rasterized representations [13].

**Reasoning on the graph: PGP** The Road Graph can also be leveraged further by directly reasoning on the graph to guide the target agent’s movement over the lanes. A good example of this is provided by the PGP framework [6] which provides a solid foundation still used to this day by very recent works [5, 14]. As such, we provide in Fig. 4 and the rest of this section a brief summary of the graph-traversal mechanism in the PGP framework.

The PGP framework can be broken down to three main components as shown on Fig. 4: graph encoding, graph traversal and trajectory decoding. The first step - graph encoding - simply refines the feature representation of the segment nodes through a combination of GRU encoders and GNN layers. The last step roughly translates a traversal on the road graph into a complete trajectory in real world coordinates.

Direct reasoning on the graph takes place through the second step of PGP: graph traversal. For each node on the graph, the model learns to predict transition probabilities to



Figure 5. **1st Exploration: Graph Resolution.** We study the importance of the graph resolution on prediction results.

adjacent nodes. Given these transition probabilities, PGP samples possible graph traversal. These traversals can then be translated to a trajectory in the third step. As such, the method explicitly takes advantage of the road graph’s structure to learn plausible predictions. We therefore propose in this paper to further study how changes to this road graph can directly impact results.

## 4. Contribution: Exploring the Road Graph

We investigate in this study three possible ways the road graph can directly influence the predicted trajectory. First, we look into how the graph resolution influences the prediction in Sec. 4.1. Afterwards, we attempt to contextualize the proposed trajectories with the road graph to improve performances in Sec. 4.2. Finally, we discuss how persistent knowledge could be embedded to further enhance the Road Graph in Sec. 4.3.

### 4.1. Spatial resolution in the Road Graph

The most natural questions when considering the road graph pertain to its construction: How much do the lengths of the segment nodes and connecting edges matter? At a very high level, this problem is analogous to resolution in images. The shorter the distances involved, the more detailed the representation is. This comes with both advantages and disadvantages. On the one hand, performance in Computer Vision tends to improve with higher resolutions. On the other, this comes with both higher dimensionality and more intensive computations [22]. While this is sometimes considered in ablative studies [18], we propose specifically investigating the question as formulated in Fig. 5.

This analogy to Computer Vision however fails to account for a particular aspect of Autonomous Driving: Vehicle Movement. Indeed, the end goal is to describe the movement of vehicles moving at varying speeds. In principle, long road segments are perfectly appropriate to describe the

Figure 6. **2nd Exploration: Trajectory correction.** We further refine final predictions using the graph.

movement of fast agents whereas slow ones would require shorter segments and connections. It is therefore necessary to carefully study the interaction between vehicle movement, prediction model performance and graph resolution in our experiments.

If resolution is to be linked to speed, it introduces the difficult question of how models can accommodate graphs of various resolutions. Changing the graph resolution between one image and the other leads to representing the world very differently within a dataset, confusing the model (as is well known in Computer Vision [23]). Technically, the increasingly distant coordinates of segment nodes should indicate the changes to the model but it is not clear this would be enough. This should therefore be accounted for when designing a model to deal with such situations.

#### 4.2. Relating predictions back to the Road Graph

A natural second question to consider relates to how the final continuous predictions relate to the Road Graph. A typical framework goes as follows: the world is discretized into a Road Graph, feature extraction or even reasoning takes place on the graph, and a continuous prediction is then output. Conspicuously, once we output a trajectory, no further use is made of the graph. We propose to look into this matter more closely here.

Interestingly, a number of work have recently taken to refining initial trajectory prediction by predicting corrections to be made to the original estimate [10]. Unfortunately, these attempts have until now been made largely independently from the Road Graph. While some works do incorporate information on the Road Graph [14], the role of the Road Graph in the refinement process is never focused on. As such, it is important to study this refinement mechanism further.

We propose to examine the importance of information on the Road Graph to the trajectory refinement process. Be-

Figure 7. **3rd Exploration: Persistent node features.** We study how “persistent” node features can be learned to characterize permanent objects like cities and lanes.

yond simply verifying that adding Road Graph information to the process is beneficial, there is a wealth of unexplored questions regarding the type of Road Graph information that should be taken into account. As such, we explore the different types of information that can be embedded into the process like the features of the graph itself or information on the spatial structure tied to the graph (e.g. the coordinates of the nodes involved in a graph traversal).

#### 4.3. Embedding persistent knowledge

Finally, we tackle the complex question of how persistent knowledge can be learned and embedded in the Road Graph. Can we learn that a road is slow? Fast? Can we learn how people drive in different parts of the world? This is a fairly under explored part of the literature as of yet, one which has significant long term ramifications for the future of autonomous driving.

A first simple consideration at this level is to tell the network which city the situation takes place in. Indeed, this is interesting as a number of driving rules are local to various places in the world. For instance, the nuScenes [1] dataset collects scenarios from two cities: Singapore and Boston. Driving habits between the two cities differ significantly due to a number of considerations ranging from the typical width of the lanes to the driving directions (left in Singapore, right in Boston). To leverage this, we propose adding a simple embedding to nodes in the lane graph depending on the city of the scenario.

A more ambitious project would be to learn characteristics at the lane level: each lane has its own characteristics such as the different driver speeds and behavior. This can be due to a number of real world factors. For instance, within the same city, it is not rare to find both very narrow streets



Method	PGP		LAformer	
	MinADE	MissRate	MinADE	MissRate
20 meter lane segments	1.27	0.51	1.26	0.41
15 meter lane segments	1.26	0.51	1.20	0.37
10 meter lane segments	1.25	0.52	1.19	0.37
5 meter lane segments <sup>†</sup>	1.26	0.52	1.20	0.36
Dynamic resolution (5 to 15 meter segments)	1.28	0.51	N/A	N/A

Table 1. **Influence of resolution on nuScenes.** Results using PGP and LAformer over the nuScenes dataset illustrate how resolution affects model performance. Default settings: PGP, <sup>†</sup>LAformer

and wide boulevard lanes, with the resulting differences in driver behavior. We explore this line of thought by reprising the previous embedding mechanism. In other words, for each lane in a city, we create a unique embedding that gets added to any road graph node belonging to that lane. The goal behind this mechanism is to acquire unique signatures for each lane that can be continually updated as part of a flexible knowledge base in an autonomous driving system.

## 5. Experiments

We investigate here the three lines of inquiry discussed in Sec. 4. After briefly discussing the experimental setting, we provide our results and conclusions regarding the impact of graph resolution (Sec. 5.1), the relation between the predicted trajectories and the graph coordinates (Sec. 5.2), as well as our efforts to embed persistent knowledge in the road graph (Sec. 5.3).

**Setting** To ensure the validity of our observations, we check the impact of the Road Graph across three graph based trajectory forecasting systems: PGP [6], LAformer [14] and (to a lesser extent) LaneGCN [13]. These three methods cover different levels of graph involvement in forecasting. The PGP method is completely reliant on the Road Graph to sample graph traversals, and is still used to this day as a backbone in some recent methods. LAformer is the current State of the Art solution on nuScenes (alongside FRM [17]), and reasons on the graph to find the most likely lane segment used by the target vehicle. Finally, LaneGCN is a seminal work that solely uses the Road Graph to extract meaningful features. All experiments are run by modifying official code repositories<sup>1</sup> for the three methods studied.

We base our conclusions on the well established nuScenes [1] dataset for the most part, with a complementary experiment on the Argoverse 1 [2] dataset to shed further light on results in Sec. 5.3. The nuScenes dataset pro-

Method	PGP (MinADE)		
	$p < 75$	$75 < p < 150$	$150 < p$
20m segments	1.14	1.35	1.33
15m segments	1.17	1.34	1.27
10m segments	1.14	1.33	1.28
5m segments	1.14	1.34	1.32
Dynamic resolution	1.15	1.37	1.32

Table 2. **Detailed influence of resolution** on PGP over nuScenes MinADE for different types of vehicle motions.  $p$  is the trajectory length in meters. Default setting:

vides high quality annotation made by humans of 5 hours of driving footage (2Hz) in 2 cities (Singapore and Boston). As such, it has been a staple of trajectory forecasting efforts since its introduction in 2021. The dataset provides as inputs logs of the past two seconds of driving for each agent and HDMaps. The expected output is a set of predictions matching the next 6 seconds of the target vehicle trajectory.

In line with the literature, we choose to evaluate the MinADE and Miss Rate metrics at K on the nuScenes val set. For the MinADE metric, given the K likeliest predicted trajectories, we compute each trajectory the Average Deviation Error to the ground truth (“ADE”) and keep the best score out of the K predictions (“Min”). Similarly, the Miss Rate (at K) indicates whether at least one of the K likeliest trajectories remained within 2 meters of the Ground Truth. Following nuScenes convention, we show metrics at K=5 predictions, and  $K=\{1,6\}$  for Argoverse 1.

### 5.1. Spatial resolution in the Road Graph

In accordance with the discussion conducted in Sec. 4.1, we explore how “resolution” (ie, the length of the lane segment nodes) impacts the performance of models. Our general experimental results are summarized in Tab. 1. Tab. 2 provides additional details to better examine the effect of changing resolution.

<sup>1</sup>PGP: [gi thub. com/nachi ket92/PGP](https://github.com/nachi-ket92/PGP)  
LAformer: [gi thub. com/mengmengli i u1998/LAformer](https://github.com/mengmengli1998/LAformer)  
LaneGCN: [gi thub. com/uber - research/LaneGCN](https://github.com/uber-research/LaneGCN)

Method	PGP		LAformer	
	MinADE	Miss Rate	MinADE	Miss Rate
Refine (with no graph involvement)	1.28	0.53	1.24	0.37
Refine (with graph features)	1.26	0.52	1.20	0.36
Refine (with relevant (x,y) node coordinates)	1.25	0.52	N/A	N/A

Table 3. **Influence of graph knowledge on trajectory refinement.** Results using PGP and LAformer on the nuScenes dataset indicate clear performance improvement from referring to the road graph.

**Experiment** We train models with different road graph resolutions (5, 10, 15 and 20 meters) and check the impact on the resulting performance. For PGP, we perform an additional experiment where we adapt the resolution (in a range between 5 to 15 meters) of the graph depending on the speed of the vehicle at the start of the prediction: slow target vehicles navigate high resolution graphs (short segments) and faster vehicles work with lower resolutions (longer segments). It should be noted that LAformer (resp. PGP) natively uses 5 (resp. 20) meter lane segments.

Tab. 1 shows that lower resolutions (e.g. 20 meter segments) tend to perform worse. In particular, LAformer performs very poorly with 20 meter segments. Interestingly, the best performance seems to be obtained with 10 meter lane segment nodes which is not the resolution used by either of those methods natively. The slightly worse performance with 5 meter segments shows that increasing the resolution does not necessarily improve results, and that other factors must be considered.

To better understand the effect of resolutions, we check in Tab. 2 the minADE (at  $K=5$ ) of PGP when predicting samples where the vehicle moves little (less than 75 meter long trajectory), moves moderately (less than 150 meter long trajectory, but more than 75) or moves a lot (more than 150 meter long trajectory). The resolution seems to have fairly little importance when predicting trajectories that span less than 150 meters. Indeed, trajectories longer than 150 meters account for most of the difference in performance between resolutions. In line with expectations, 5 meter segments struggle with long trajectories as they depict shorter spatial locations. This reasoning fails to explain why 20 meter segments perform worse with longer trajectories. A possible explanation might be that lower resolution induces more inaccuracies, and that these inaccuracies compound the longer the trajectory is.

Finally, our attempt to dynamically tie the resolution to initial vehicle speed produces very poor results across the board. This does not seem to simply be an issue of the method striking a compromise between the performance of high and low resolution models. Indeed, the performance over moderately long trajectories appears far worse than

that obtained by any single resolution model. As such, this could be caused by the network having difficulty interpreting different network resolutions.

## 5.2. Relating predictions back to the Road Graph

As discussed in Sec. 4.2, we try to link predicted trajectories in “the real world” back to the structured graph representation to see how this affects performance. Our general experimental results are summarized in Tab. 3.

**Experiments** We add a refinement head (3 layer CNN) to PGP, and reprise the native refinement module of LAformer. Refinement heads are fed as input the predicted scratch trajectories to refine along with some other features. These other features do not traditionally include graph information. We check two ways of injecting graph knowledge: an attention-based summary of relevant graph features, or the (x,y) coordinates of the samples graph traversal (in PGP).

As shown in Tab. 3, including graph knowledge of any kind significantly improves the MinADE. Simply adding a refinement head fails to improve performance for PGP (and might actually deteriorate it). This result stands in sharp contrast to the performance of both variants of the graph based refinement models we propose. Interestingly, introducing the (x,y) coordinates of the sampled graph traversal seems to yield even better results than using an attention based summary of graph features as is done natively in LAformer. This indicates that not all the spatial information contained in the graph gets translated to the predicted trajectory. Further work is therefore needed on the matter.

## 5.3. Embedding persistent knowledge

Building on the ideas developed in Sec. 4.3, attempt to inject some “persistent” knowledge in the graph. Ideally, this should provide us with a way to link many different scenarios that happen in the same city, district or even large avenue. Our general experimental results are summarized in Tab. 4 and Tab. 5.

**Experiments** We tackle two types of persistent knowledge here: city-wide knowledge and lane-wide knowl-



Method	LaneGCN				LAformer			
	K=1		K=6		K=1		K=6	
	MinADE	Miss Rate	MinADE	Miss Rate	MinADE	Miss Rate	MinADE	Miss Rate
No persistent feature	1.35	0.49	0.72	0.10	1.23	0.46	0.65	0.09
Lane embedding	1.34	0.49	0.71	0.10	1.22	0.45	0.65	0.09

Table 4. **Influence of lane embeddings.** Results of LaneGCN and LAformer on the Argoverse 1 dataset show consistent benefits of lane embeddings.

Method	PGP	
	MinADE	Miss Rate
No persistent feature	1.27	0.51
City features	1.28	0.52
City features w/ proper flip	1.26	0.51

Table 5. **Influence of city features.** Results on the nuScenes Dataset using the PGP framework indicate city features improve performance.

edge. For the first, we conduct a preliminary experiment on nuScenes using the PGP backbone: we add city identifying binary features to the nodes of the road graph to indicate whether the scene takes place in Singapore or Boston. As to the second, we catalogue all the known lanes in the Argoverse 1 dataset. For each lane segment node, we add a unique learnable embedding characterizing the lane from which it comes. These experiments on lane embeddings are performed on the Argoverse 1 dataset as the nuScenes dataset proved too small. Since the PGP framework translates poorly to the Argoverse 1 format, we use LAformer and LaneGCN as baseline methods.

Preliminary experiments on the city features seem to indicate some improvement as indicated by Tab. 5. Interestingly, this improvement only manifests when accounting for the influence of horizontal flip augmentation: city features need to be switched off when the sample is flipped. This seems to indicate the city features help the model learn whether one needs to drive on the left (Singapore) or on the right (Boston). When the horizontal augmentations is performed indiscriminately, the model learns to be invariant to this information and therefore the city features are useless.

Tab. 4 shows consistent improvements for both LaneGCN and LAformer on the Argoverse 1 dataset. Given the minimal cost of adding lane embeddings to graph nodes, it seems worthwhile to further explore the idea. Greater improvements could be obtained by better focusing which lane embeddings should be updated. As it stands, every lane embedding gets updated when training on a scenario even if a lane has nothing to do with the prediction of interest.

Interestingly, this improvement seems to require a large amount of data. Preliminary experiments on the nuScenes dataset - where a lane might appear in at most a few hundred scenarios - invariably lead to clear overfitting. From that, we can glean two conclusions: lane embeddings provide a lot of expressive power to models, and only well traveled roads should be considered for real-time updates of lane embeddings. These results show that adding persistent knowledge may improve MinADE in some cases and merits further study.

## 6. Conclusion

In this paper, we explore three questions regarding the use of road graphs in trajectory forecasting for autonomous vehicles. What is the importance of the graph resolution? Can we further improve predictions by relating them back to the road graph? How should we embed learnable “persistent” knowledge in the graph?

After formally introducing the road graph and its role for autonomous vehicle trajectory forecasting, we develop these three general questions and identify key points to elucidate in our experiments. Through experiments on the well-known nuScenes dataset and through 2 recent forecasting methods (PGP and LAformer), we manage to provide some insights on our questions: graph resolution seems to mostly affect predictions of longer trajectories, graph information (node coordinates in particular) is very valuable when refining trajectories, and it seems possible to learn some “persistent” node features that can be shared across multiple scenarios.

Through this exploratory study, we have striven to shed some light on the road graph that lies at the center of many modern trajectory forecasting methods. Nevertheless, we have focused on three specific facets of the issue without considering the many others that would affect model performance. The properties of the road graph remain to be explored. We hope that our work can lead to further investigation both of new questions and of the interrogations that remain on the ones we have studied.

**Acknowledgements** This work was realized by the MultiTrans project funded by the Agence Nationale de la Recherche under grant reference ANR-21-CE23-0032. The authors are grateful to the OPAL infrastructure from Université Côte d’Azur for providing resources and support.

## References

- [1] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuScenes: A multimodal dataset for autonomous driving. *arXiv preprint arXiv:1903.11027*, 2019. 1, 2, 5, 6
- [2] Ming-Fang Chang, John W Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, and James Hays. Argoverse: 3d tracking and forecasting with rich maps. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1, 2, 6
- [3] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. PaLM: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022. 1
- [4] Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NeurIPS 2014 Deep Learning and Representation Learning Workshop*, 2014. 3
- [5] Daniel Dauner, Marcel Hallgarten, Andreas Geiger, and Kashyap Chitta. Parting with misconceptions about learning-based vehicle motion planning. *ArXiv preprint*, 2023. 4
- [6] Nachiket Deo, Eric Wolff, and Oscar Beijbom. Multimodal trajectory prediction conditioned on lane-graph traversals. In *Conference on Robot Learning (CoRL)*, 2021. 1, 2, 3, 4, 6
- [7] Jiyang Gao, Chen Sun, Hang Zhao, Yi Shen, Dragomir Anguelov, Congcong Li, and Cordelia Schmid. VectorNet: Encoding HD Maps and Agent Dynamics From Vectorized Representation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 1, 2, 3, 4
- [8] Junru Gu, Chenxu Hu, Tianyuan Zhang, Xuanyao Chen, Yilun Wang, Yue Wang, and Hang Zhao. Vip3d: End-to-end visual trajectory prediction via 3d agent queries. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2
- [9] Mahir Gulzar, Yar Muhammad, and Naveed Muhammad. A survey on motion prediction of pedestrians and vehicles for autonomous driving. *IEEE Access*, vol. 9:137957–137969, 2021. 2
- [10] Xiaosong Jia, Li Chen, Penghao Wu, Jia Zeng, Junchi Yan, Hongyang Li, and Yu Qiao. Towards capturing the temporal dynamics for trajectory prediction: a coarse-to-fine approach. In *Conference on Robot Learning (CoRL)*, 2022. 2, 5
- [11] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2012. 1
- [12] Sampo Kuutti, Richard Bowden, Yaochu Jin, Phil Barber, and Saber Fallah. A survey of deep learning applications to autonomous vehicle control. *IEEE Transactions on Intelligent Transportation Systems*, 2020. 1
- [13] Ming Liang, Bin Yang, Rui Hu, Yun Chen, Renjie Liao, Song Feng, and Raquel Urtasun. Learning lane graph representations for motion forecasting. In *European Conference on Computer Vision (ECCV)*, 2020. 1, 2, 3, 4, 6
- [14] Mengmeng Liu, Hao Cheng, Lin Chen, Hellward Broszio, Jiangtao Li, Runjiang Zhao, Monika Sester, and Michael Ying Yang. LAformer: Trajectory Prediction for Autonomous Driving with Lane-Aware Scene Constraints. *ArXiv preprint arXiv:2302.13933*, 2023. 1, 2, 3, 4, 5, 6
- [15] Yicheng Liu, Yuan Yuantian, Yue Wang, Yilun Wang, and Hang Zhao. Vectormapnet: End-to-end vectorized hd map learning. In *International Conference on Machine Learning (ICML)*, 2023. 1, 2
- [16] Yicheng Liu, Jinghui Zhang, Liangji Fang, Qinghong Jiang, and Bolei Zhou. Multimodal motion prediction with stacked transformers. *Computer Vision and Pattern Recognition (CVPR)*, 2021. 2
- [17] Daehee Park, Hobin Ryu, Yunseo Yang, Jegyeong Cho, Jiwon Kim, and Kuk-Jin Yoon. Leveraging future relationship reasoning for vehicle trajectory prediction (frm). In *International Conference on Learning Representations (ICLR)*, 2023. 1, 6
- [18] Liad Pollak Zuckerman, Eyal Naor, George Pisha, Shai Bagon, and Michal Irani. Across scales & across dimensions: Temporal super-resolution using deep internal learning. In *European Conference on Computer Vision (ECCV)*. Springer, 2020. 4
- [19] Abhimanyu Roy, Jingyi Sun, Robert Mahoney, Loreto Alonzi, Stephen Adams, and Peter Beling. Deep learning detecting fraud in credit card transactions. In *Systems and Information Engineering Design Symposium (SIEDS)*, Charlottesville, VA, USA, 2018. 1
- [20] Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In *European Conference on Computer Vision (ECCV)*, 2020. 1, 2
- [21] Wilko Schwarting, Javier Alonso-Mora, and Daniela Rus. Planning and decision-making for autonomous vehicles. *Annual Review of Control, Robotics, and Autonomous Systems*, 2018. 2
- [22] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks, 2020. 4
- [23] Hugo Touvron, Andrea Vedaldi, Matthijs Douze, and Hervé Jégou. Fixing the train-test resolution discrepancy, 2022. 5
- [24] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. 1
- [25] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks. *International Conference on Learning Representations (ICLR)*, 2018. 1, 4