

LMFormer: Lane based Motion Prediction Transformer

Harsh Yadav¹ Maximilian Schäfer² Kun Zhao² Tobias Meisen¹

¹University of Wuppertal, Germany ²Aptiv Services Deutschland GmbH

harsh.yadav@uni-wuppertal.de

Abstract

Motion prediction plays an important role in autonomous driving. This study presents LMFormer, a lane-aware transformer network for trajectory prediction tasks. In contrast to previous studies, our work provides a simple mechanism to dynamically prioritize the lanes and shows that such a mechanism introduces explainability into the learning behavior of the network. Additionally, LMFormer uses the lane connection information at intersections, lane merges, and lane splits, in order to learn long-range dependency in lane structure. Moreover, we also address the issue of refining the predicted trajectories and propose an efficient method for iterative refinement through stacked transformer layers. For benchmarking, we evaluate LMFormer on the nuScenes dataset and demonstrate that it achieves SOTA performance across multiple metrics. Furthermore, the Deep Scenario dataset is used to not only illustrate cross-dataset network performance but also the unification capabilities of LMFormer to train on multiple datasets and achieve better performance.

1. Introduction

As stated by Hu et al. a typical autonomous driving pipeline involves *perception*, *prediction*, and *planning* [17] modules. Here the prediction module aims to predict the future trajectories of the agents (e.g. vehicles, bicycles, pedestrians, etc.) in an environment. The inputs to this module are the dynamic context, i.e., past trajectories of the agents, and the surrounding static context, e.g., traffic lights, road boundaries, lanes, pedestrian crossings, traffic signs, parked vehicles, construction sites, etc. The downstream planning module is tasked with identifying the optimal trajectories and generating vehicle control commands.

Recent advances in trajectory prediction have seen significant progress through the integration of machine learning approaches [8, 13, 20, 22, 29, 34, 35, 40, 43]. However, current state-of-the-art methods still face substantial challenges, leaving considerable room for improvement before trajectory prediction can be considered a solved problem. In

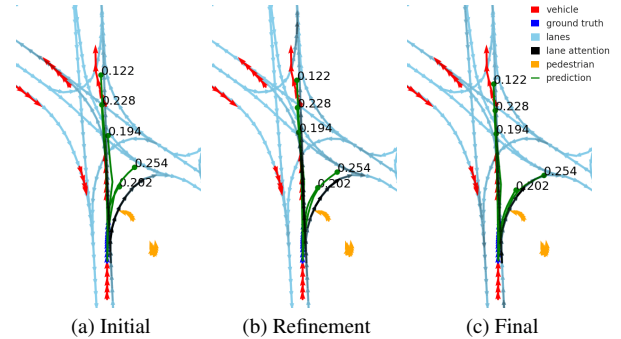


Figure 1. A schematic of predicted trajectories (green) with mode probabilities for the target agent. In (a), the initial predictions exhibit noticeable misalignment with the drivable lanes. However, through iterative refinement steps, the trajectories become increasingly structured, as seen in (b) and (c). Notably, the final predictions not only align well with the road topology but also exhibit a realistic distribution of possible future paths.

this study, we address some of these challenges and propose novel solutions to tackle them. Our investigation primarily focuses on three critical aspects:

(1) Dauner et al. [9] demonstrate that by following the road lane, their open-loop planner for the target vehicle outperforms other state-of-the-art (SOTA) methods. Since open-loop planning and marginal trajectory prediction problems are correlated, we hypothesize that a similar phenomenon should also be observed for the trajectory prediction task. Indeed, in the trajectory prediction domain, some recent studies [19, 22, 38] have shown that prediction modules designed solely based on lane information as a part of the static context can still achieve SOTA performance. However, in these studies, lane awareness is divided into two modules: the first module performs lane attention and selects the most important lanes in the scene, while the second module uses the contextual information of the selected lanes for predicting future trajectories. In contrast, inspired by Transformer architectures, we argue that lane attention should be a part of the prediction module and should dynamically prioritize lane segments critical for predicting the corresponding trajectory and vice versa.

(2) Prior studies [13, 19, 22, 29, 38, 42, 43] fail to account for lane connection information among different lanes at intersections, lane splits and lane merges. Sun et al. [35] attempt to address this limitation by generating a knowledge graph of lane segments to learn their long-range dependency. However, this approach heavily relies on map ontology, which restricts its scalability.

(3) Predicted trajectories frequently exhibit jerky behavior and poor lane alignment. Previous studies [35, 38] address this issue by generating a large number of trajectory proposals and filtering out those deemed unreasonable. Alternatively, other works [22, 43] employ two-stage prediction frameworks, where the first stage produces coarse trajectory anchors, and the second stage refines these anchors by calculating the corresponding offsets. However, both approaches are computationally expensive; the former due to the generation of redundant proposals and the latter due to the additional learnable parameters required for the refinement module. Zhou et al.’s adaptive refinement method [41] mitigates these costs by performing iterative refinement with minimal computational overhead. Nonetheless, its performance is highly sensitive to hyperparameter selection, limiting its robustness in real-world applications.

To address the aforementioned challenges, we propose **Lane based Motion Prediction Transformer (LMFormer)**, that incorporates lane-aware attention within the cross-attention layer of the prediction module. Our method dynamically prioritizes lane segments, using them as anchors to predict future trajectories. By keeping the lengths of individual lane segments sufficiently small, we effectively capture road curvature. However, this introduces the issue of long-range dependencies across lane segments. To overcome this, we construct a graph of lane segments that captures the long-range road structure. A graph neural network (GNN)-based map encoder is employed to enrich each lane segment’s representation with information from downstream segments, including lane splits and lane merges. Notably, our map encoder avoids collecting information from irrelevant or oppositely directed lane segments, as done in [21], ensuring greater computational efficiency. We further address the challenge of cost-effective refinement by computing additional refinement losses from intermittent output queries of stacked layers. To do so, we hypothesize that the output queries of the intermittent transformer layers should map to the same latent space as the final layer. Thus, intermittent output queries can also be transformed into trajectories and compared against the ground truth. It is important to note that the weights across different stacked layers are not shared. To the best of our knowledge, no existing trajectory prediction approach leverages intermittent outputs of stacked transformer layers to compute additional refinement losses.

We validate our proposed approach through extensive

experimentation on the nuScenes [1] and the DeepScenario [26] datasets, achieving SOTA performance across a range of metrics. We demonstrate that:

- The incorporation of lane awareness within the cross-attention layer of the transformer-based architecture enhances explainability in the prediction module.
- The Graph Neural Network (GNN)-based architecture effectively models long-distance dependencies across lane segments and improves the accuracy of predicted trajectories.
- The iterative refinement strategy refines predicted trajectories with minimal additional computational cost.
- LMFormer can be simultaneously trained on multiple datasets to achieve better generalization capabilities.

2. Related Work

The lane-aware trajectory prediction task primarily involves three important challenges: scene representation, learning a lane graph, and forcing the diversity in the outputs. We review some of the important related work for these strategies in this section.

2.1. Scene Representation

Earlier works [3, 8] rely on rasterized representations of the surrounding scene, where static and dynamic context information is stored across the channel dimension. These representations allow for the use of Convolutional Neural Networks (CNNs) to extract meaningful features. However, rasterized representations suffer from limited resolution and include redundant pixel information, motivating the exploration of more compact scene representations. LaneGCN [21] first introduced the idea that HD maps inherently possess a graph structure, leveraging graph convolution to encode this information. Building on this, VectorNet [13] proposed a unified vector representation for various scene elements, including lanes, crosswalks, traffic lights, and surrounding agents. Despite these innovations, vector-based approaches face a limitation: they encode scene elements from the perspective of the target vehicle, necessitating re-normalization and re-encoding at each time step, which introduces redundant computations. QCNet [43] addresses this issue by adopting a query-centric approach. It encodes each scene element in its own coordinate frame, removing the need for re-normalization and re-encoding. Additionally, QCNet employs temporal and spatial embeddings to capture relative information across scene elements effectively. In our work, we adopt the query-centric vector-based approach to encode scene elements, as it offers lower computational complexity compared to agent-centric and rasterized representations.

2.2. Lane Graph

The lane information in the static context for motion prediction tasks is typically stored as lane polylines. Each polyline is represented as $L_p^i = [P_1^i, P_2^i, \dots, P_K^i]$, where K denotes the number of control points in the polyline i , and P_k^i denotes the position of these control points in the global coordinate system. Prior studies [13, 22, 29] first transform the polylines into the target agent coordinate system before generating their corresponding encodings. On the other hand, the approaches [21, 38, 42, 43] first convert the point-based representation of the polylines to segment-based representation; $L_v^i = [V_1^i, V_2^i, \dots, V_{K-1}^i]$, where $V_k^i = [P_k^i, P_{k+1}^i]$ stores the vector information of lane segments. In this case, the lane features are generated from the vectors of individual segments. The advantage of the segment-based approach is that road curvature is explicitly encoded in the lane encodings. A potential limitation of all the aforementioned approaches is that they fail to use explicit information on lane connections at intersections, lane splits, and lane merges. Sun et al. [35] attempt to use lane connection information to generate consistent lanes. However, their method requires an extensive map ontology, the creation and maintenance of which is expensive and thus has limited scalability. Therefore, in our work, we propose a simple mechanism to use the lane connection information together with the segment-based representation. This integrated approach not only achieves SOTA performance, but also improves the explainability of the prediction module.

2.3. Multi-Modal Prediction

To safely plan paths in complex traffic scenarios, motion prediction modules must generate multiple modes, each representing a scene-consistent trajectory. The advent of vector-based scene representations has enabled researchers to explore techniques such as Variational Autoencoders (VAEs) [2, 7] and Generative Adversial Networks (GANs) [18] for producing diverse trajectory predictions. However, these approaches often suffer from mode collapse, where all predicted modes converge to a single trajectory with minor variations. In contrast, transformer architectures leveraging learnable anchors have achieved notable success in generating diverse trajectory predictions [24, 29, 42, 43]. Current SOTA methods build on transformer backbones and incorporate advanced techniques, including asked auto encoders [6, 20], knowledge graphs [35], and language models [34]. These approaches have been extensively validated on open-source datasets such as nuScenes [1], Argoverse [4], Argoverse 2 [39], and Waymo [28]. Building on this trend, UniTraj [12] introduces a unified framework that consolidates different datasets, network architectures, and evaluation criteria, enabling the development of larger and more generalized models.

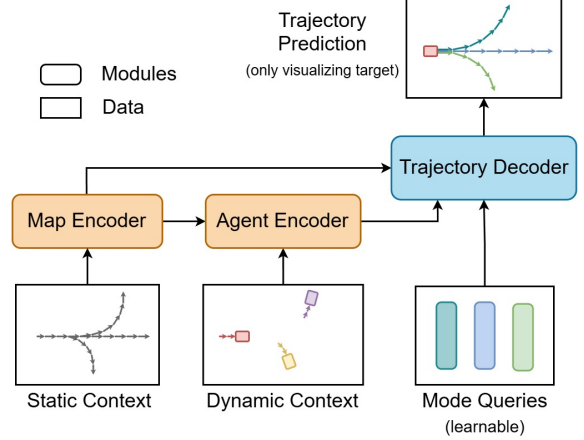


Figure 2. An Illustration of LMFormer architecture. We employ a transformer-based encoder-decoder architecture to generate multiple scene-consistent trajectories for all the dynamic agents. Notably the static context only consists of lane segments.

3. Approach

In this section, we describe the input-output formulation and the network architecture. We further provide a mathematical illustration of the loss function.

3.1. Input-Output Formulation

The input to LMFormer consists of the positions of surrounding lanes and dynamic agents in the scene. As described in section 2.1, we opt for a query-centric representation [43] of each scene element. This representation eliminates the need for redundant agent-centric encodings at each time step, thereby reducing the computational overhead associated with repeated coordinate transformations as well as encoding generation. Below, we detail how we compute the query-centric coordinates and features for both static and dynamic contexts. In addition, we also illustrate the formulation of the output trajectory in vector format.

Static Context: As described in section 2.2, the static context consists of lane polylines from which lane segments are generated. Every lane segment is characterized by its start and end positions in global coordinates. The query-centric coordinate frame for a lane segment is set as follows: we designate the segment’s start position as its origin and align the segment’s vector direction with its x-axis. As a result, the only feature preserved in this query-centric coordinate frame is the segment’s length, which we use as its sole feature in the query-centric embedding generation.

Dynamic Context: For the dynamic context, agent trajectories are provided as $\mathcal{T}_{in}^a = [P_1^a, P_2^a, \dots, P_T^a]$ with global positions P_t^a . Motion vectors $M_t^a = [P_t^a, P_{t+1}^a]$ are derived at each time step from these trajectories, encapsulating the agent’s start and end positions within that interval. The query-centric coordinate frame for a motion vector is deter-

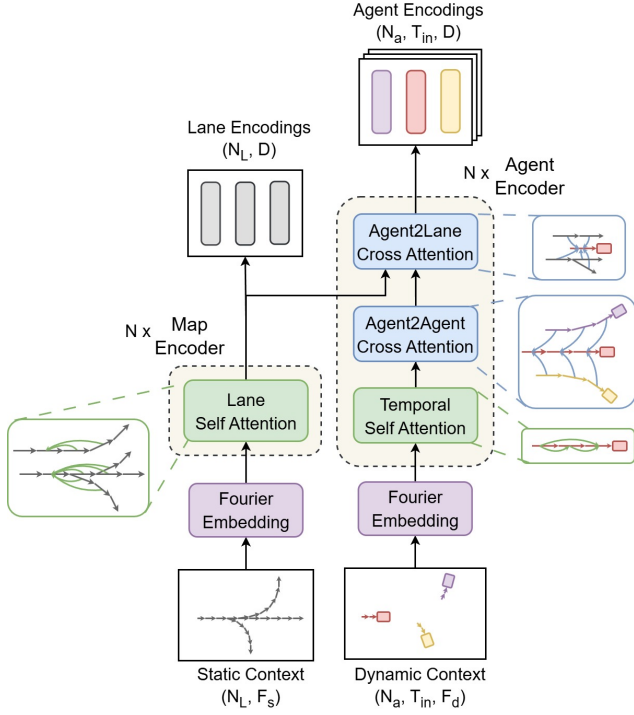


Figure 3. The encoder receives Static and Dynamic context as input and it outputs Lanes and Agents Encodings. The encoder is divided into two parts: Map Encoder and Agent Encoder. The Map Encoder models the long-range interaction among the lane segments. The Agent Encoder models the interaction of all the surrounding static and dynamic elements into each agent’s latent embedding. The attention mechanisms are illustrated by green (self-attention) and blue (cross-attention) arrows, where the arrowheads point toward the queries and the tails point away from keys/values. The encoders repeat the interaction modeling N times, to learn complex interactions, where the weights across each layer are not shared.

mined as follows: we set the motion vector’s start position as its origin and align the agent’s instantaneous direction of travel with its x-axis. Notably, the motion vector’s direction may differ from the instantaneous direction of travel, introducing an angular deviation from the x-axis. Consequently, in the query-centric representation, we encode the motion vector’s length and its orientation relative to the x-axis as its features.

Output Representation: The output can be represented as $\mathcal{T}_{out}^{a,m} = [(V_1^{a,m}, S_1^{a,m}), (V_2^{a,m}, S_2^{a,m}), \dots, (V_{T'}^{a,m}, S_{T'}^{a,m})]$, where T' is the temporal length of the future trajectories, and $V_t^{a,m} = [P_{t-1}^{a,m}, P_t^{a,m}]$ is the motion vector corresponding to agent a , mode m , and time step t , with an associated variance of $S_t^{a,m}$. The query-centric coordinate frame for these motion vectors is defined as the last observed position and the current direction of travel of the corresponding agent. The predicted future trajectories are reconstructed

from these motion vectors in the global coordinate frame, preserving mean positions and associated variances at each time step. Additionally, we predict the probability distribution over predicted trajectories, quantifying the likelihood of each mode.

3.2. Network Architecture

Our model follows a transformer-based encoder-decoder architecture, where the decoder generates scene-consistent trajectories from the latent scene encodings produced by the encoder, as shown in Figure 2.

3.2.1. Encoder

The encoder is designed to capture interactions between different scene elements. Our design leverages the fact that while the static context is independent of the dynamic context, the reverse does not hold. Consequently, we first encode the static context independently before incorporating it into the dynamic context encoding process. This necessitates two distinct encoders: a Map Encoder and an Agent Encoder, as illustrated in Figure 3. Given the high-frequency nature of motion vectors, we apply learnable Fourier Embeddings [36] to all scene elements prior to modeling interactions within the encoder.

The Map Encoder is responsible for capturing long-range interactions between lane segments. To construct this interaction graph, we exploit the fact that the most relevant contextual information for each lane segment is present in the lane segments accessible via traversing in the direction of travel. These interactions are modeled using multi-headed attention [37], where each lane segment is represented by its query-centric feature embeddings. To capture the relative positions in the multi-headed attention, the keys/values are concatenated with relative position embeddings [43]. The Lane Self-Attention module is applied iteratively N times, refining the lane encodings before passing them to the Agent Encoder and subsequent network components.

The Agent Encoder learns interactions between agents and their surrounding scene elements. It comprises three specialized attention modules:

- Temporal Self-Attention: Captures dependencies across different time steps for the same agent.
- Agent2Agent Cross-Attention: Models interactions between different agents at a given time step.
- Agent2Lane Cross-Attention: Encodes contextual interactions from lane segments (keys/values) to agents (queries).

Similar to the Map Encoder, multi-headed attention with relative position embeddings is employed in the Agent Encoder. Each interaction module is applied iteratively N times, producing Agent Encodings that serve as input for the trajectory decoder.

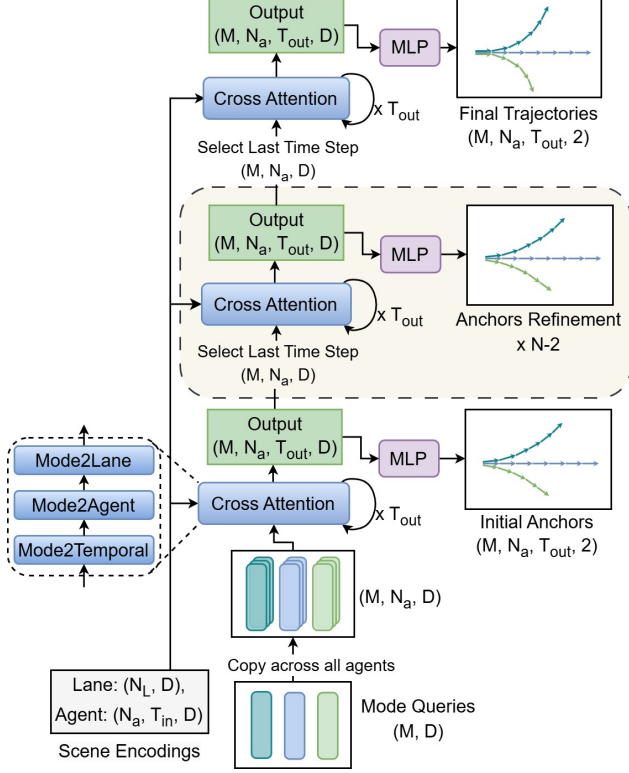


Figure 4. A depiction of the decoder architecture. The decoder performs cross-attention in between learnable mode queries and Scene Encodings (keys/values). The cross-attention layer is stacked N times and the intermittent output as well as the final output queries are transformed into trajectories with MLP. Thus we obtain N trajectories corresponding to each mode of every agent. During the training, all these N trajectories are trained against the ground truth, while during inference only the final layer output is generated. Importantly, the weights across the stacked cross-attention layers are not shared, while those in the MLP layers are.

3.2.2. Decoder

To facilitate multimodal trajectory prediction, we introduce learnable mode queries, with each query corresponding to a distinct trajectory for every agent in the scene. Figure 4 shows the architecture of the decoder.

During initial experiments, we observed that the predicted trajectories exhibited jerky motion and, in some cases, misalignment with lane structures. To mitigate these issues, we drew inspiration from the iterative refinement of bounding boxes in DAB-DETR [23], where bounding boxes are progressively refined through offset computations after each transformer output layer. However, direct adaptation of this approach is impractical for trajectory prediction, as bounding boxes in computer vision are randomly initialized, whereas trajectory prediction requires a reliable set of initial predictions. Thus, instead of applying offset-based refinement at each transformer layer, we compute entire tra-

jectories and compare them against the ground truth to derive additional refinement losses. Another important distinction from DAB-DETR is that LMFormer does not share the weights across each refinement layer, but rather uses the intermittent output from the multiple stacked layers. We show the effectiveness of the additional refinement loss in an ablation study.

Previous research [40, 43] demonstrates that recurrent temporal decoding enhances trajectory prediction quality. Therefore, we incorporate recurrent decoding ($\times T_{out}$) in cross-attention modules. Each recurrent loop updates the query position as well as the query itself, similar to CASPFormer [40]. The cross-attention module consists of three key attention mechanisms:

- **Mode2Temporal Cross Attention:** Aggregates temporal information from Agent Encodings (keys/values) to modes (queries).
- **Mode2Agent Cross Attention:** Each mode of an agent attends to the same mode of the other agent in the scene.
- **Mode2Lane Cross Attention:** Enables mode queries to incorporate static context information from Lane Encodings (keys/values).

3.3. Loss Formulation

We model the predicted trajectories using a mixture model, which is trained to maximize the likelihood of the ground truth trajectory. The optimization objective is thus formulated as follows:

$$\mathcal{L}(Y) = \sum_{m=1}^M \pi_m \prod_{t=1}^T P(Y^t | \mu_m^t, b_m^t), \quad (1)$$

where Y represents the ground truth trajectory, π_m denotes the probability of mode m , and $P(\cdot)$ follows a Laplace distribution parameterized by the mean μ_m and scale b_m , which defines the positional uncertainty of the predicted trajectories.

As shown by Rupprecht et al. [32], directly optimizing the Negative Log Likelihood (NLL) of mixture models can lead to numerical instability and mode collapse. To mitigate this, the mixture model optimization is decomposed into two objectives via separate regression and classification losses [8, 27, 40, 42]. In our approach, we adopt the Winner-Takes-All (WTA) strategy, where the regression loss optimizes only the parameters of the best-matching mode and is defined as:

$$\begin{aligned} L_{reg} &= -\log \prod_{t=1}^T P(Y^t | \mu_{m*}^t, b_{m*}^t) \\ &= -\sum_{t=1}^T \log P(Y^t | \mu_{m*}^t, b_{m*}^t), \end{aligned} \quad (2)$$

where m^* is the mode with the smallest L_2 distance to the ground truth trajectory. For the classification loss, we optimize the NLL of the likelihood function shown in equation (1) as follows:

$$L_{cls} = -\log \mathcal{L}(Y) \quad (3)$$

To enhance training stability, we decouple the optimization: the classification loss updates only the mode probabilities π_m , while the regression loss optimizes the trajectory parameters μ_m and b_m . Finally, the total training loss is defined as the sum of classification and regression losses:

$$L = \lambda L_{cls} + \sum_{n=2}^N L_{reg}, \quad (4)$$

where N denotes the number of stacked layers, and λ controls the trade-off between classification and regression losses.

4. Experiments

In this section, we present the experimental evaluation of LMFormer. We first describe the datasets and experimental setup, followed by a detailed quantitative and qualitative performance analysis. Our goal is to demonstrate the effectiveness of LMFormer on both standard benchmarks and real-world-inspired scenarios while also highlighting its generalization ability across different datasets.

4.1. Experimental Setup

Dataset: As we aim to evaluate LMFormer’s performance in structured urban environments as well as more diverse intersection-heavy scenarios, we selected the nuScenes [1] and the Deep Scenario [26] datasets for benchmarking. The nuScenes dataset comprises driving scenarios from Boston and Singapore, generated using sensor data of a moving vehicle in complex scenarios. The Deep Scenario dataset is built from recordings of stationary drones, placed vertically above complex traffic scenarios, e.g. intersections.

Metrics: We report the quantitative performance of LMFormer on the nuScenes and Deep Scenario datasets in terms of minADE_k , minFDE_k , MR_k , and OffRoadRate . minADE_k measures the average Euclidean distance between the ground truth and the best-matching trajectory among the top- k predicted trajectories. A lower value indicates better alignment with the actual observed path. This metric is particularly relevant for multi-modal prediction settings, where multiple future outcomes are possible. minFDE_k computes Euclidean distances between the endpoint of the ground truth and the best prediction among the top- k trajectories. It indicates how well the network predicts the final goal position. MR_k measures the fraction of misses, where a miss occurs when the maximum pointwise

L_2 distance between the ground truth and predicted trajectories is larger than two meters. A small value indicates that the predicted velocity profile matches the ground truth well. Finally, the OffRoadRate metric reports the fraction of predicted trajectories outside the drivable area.

Implementation Details: We train LMFormer on one Nvidia A100 GPU using AdamW Optimizer [25]. As required by nuScenes Prediction Challenge [30], the duration of the input trajectory is limited to 2 seconds and that of the output trajectory to 6 seconds, with a sampling rate of 2Hz. Also, the maximum number of modes k is set to 5, to compute minADE_5 and MR_5 for the comparison against other SOTA methods. The same configurations are extended to the Deep Scenario Dataset, as it does not specify any restrictions on the aforementioned hyperparameters.

Both static and dynamic contexts enclose an area of 150m x 100m, where the former is along the driving direction. In order to help the model generalize for both left-hand and right-hand driving conditions, the inputs are flipped along the driving direction with a probability of 50%. We also identify that increasing the number of stacked layers N beyond 3 does not improve the network performance, and hence we set $N = 3$ to keep the computational costs to a minimum. A similar observation is made for the length of a lane segment, and its optimum value is recorded at 3 meters.

4.2. Results and Discussion

We provide a comparison of LMFormer against the SOTA networks on the nuScenes Motion Prediction Challenge test split [30] in Table 1. LMFormer achieves SOTA results across minADE_5 , MR_5 and OffRoadRate . Qualitative results are depicted in Figure 1. As is evident, the trajectories undergo a substantial enhancement during the refinement process. It is noteworthy that the network is capable of making diverse predictions and aligning the predicted trajectories with a high degree of accuracy with the lanes. Furthermore, the attention map provides insight into the underlying learning behaviors of LMFormer. We observe that the network pays higher attention to the lanes, which are important

Table 1. Comparison with state-of-the-art on the nuScenes prediction challenge test split.

Method	$\text{minADE}_5 \downarrow$	$\text{MR}_5 \downarrow$	$\text{minFDE}_1 \downarrow$	$\text{OffRoad} \downarrow$
GOHOME [14]	1.42	0.57	6.99	0.04
Autobot [16]	1.37	0.62	8.19	0.02
THOMAS [15]	1.33	0.55	6.71	0.03
PGP [10]	1.27	0.52	7.17	0.03
MacFormer [11]	1.21	0.57	7.50	0.02
LAFormer [22]	1.19	0.48	6.95	0.02
Socialea [5]	1.18	0.48	6.77	0.03
FRM [31]	1.18	0.48	6.59	0.02
CASPNet_v2 [33]	1.16	0.50	6.18	0.01
CASPFormer [40]	1.15	0.48	6.70	0.01
SemanticFormerR [35]	1.14	0.50	6.27	0.03
LMFormer (ours)	1.14	0.47	6.85	0.01

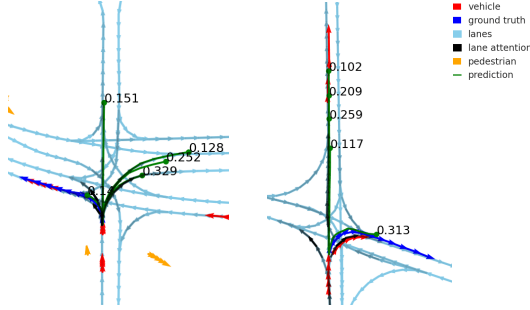


Figure 5. An illustration of predicted trajectories (green) with mode probabilities for the target agent at the final output layer. These samples are selected out of the 100 worst predictions based on minADE_5 .

for predicting its future trajectory.

It is important to note that despite achieving SOTA performance across all other metrics, LMFormer does not achieve SOTA on minFDE_1 . We suspect that this behavior emerges due to the fact that the predicted trajectories have to align with the lanes, which is rarely the case in the real world. Such lane alignment would predominantly impact the predicted position towards the end of the trajectory compared to the initial ones. Thus, the results of this effect are more pronounced in minFDE_k compared to minADE_k and MR_k .

We additionally investigate the worst reported prediction of LMFormer based on the minADE_5 metric (see Figure 5). Interestingly, even though the network generates diverse predictions, the predicted velocity for the mode following the ground truth path can be suboptimal, resulting in a bad minADE_k . Thus, we hypothesize that the performance of the motion prediction modules can be improved by separately predicting velocity profiles and future paths. The work by Sun et al. [35] also reports a similar observation, i.e., if the prediction modules have access to the ground truth velocity, the performance of the models improves significantly. Given this evidence, the direction of separation between velocity and path prediction should be explored in future work.

We also test the generalization capability of LMFormer on the Deep Scenario dataset [26]. To this end, we train the network on 55,000 training samples, generated from four different intersections’ recordings (‘Unparalleled Frankfurt’, ‘Trendy Renningen’, ‘Fortunate Karlsruhe’, ‘Stunning Stuttgart’) in the dataset. We conducted the qualitative testing for this model checkpoint on the samples of unobserved intersections during training (refer Figure 6). We observe that in certain cases, LMFormer fails to predict all plausible trajectories. We hypothesize that this limitation is due to an unbalanced data distribution, where some maneuvers occur less frequently in the training data. This suggests that

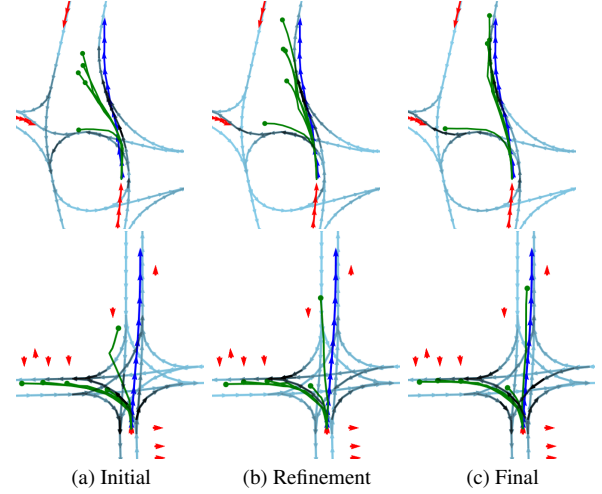


Figure 6. Visualization of predicted trajectories (green) for a target agent at an unseen intersection during training. The model, trained on the Deep Scenario dataset, refines its predictions progressively from (a) to (c), achieving better alignment with the lane structure. However, as shown in the second row, a limitation of our approach becomes evident: despite the attention maps indicating that the network should predict right-turn trajectories, it fails to generate them.

additional mechanisms, such as explicit trajectory diversity constraints or targeted data augmentation, may be required to improve the robustness of multimodal prediction, but we leave these improvements for future work.

Furthermore, a quantitative evaluation of this model checkpoint, which is trained on Deep Scenario, is done on the nuScenes validation split to assess its cross-dataset generalization capabilities. The corresponding results are presented in Table 2. It can be observed that the performance of this model checkpoint is significantly worse in comparison to the one trained on the nuScenes training set. In our analysis of this cross-dataset performance degradation, we identified a systematic underestimation of the predicted velocity in the case of Deep Scenario as compared to nuScenes. We attribute this discrepancy to the distributional differences between the datasets: Deep Scenario captures mainly slow-moving vehicles in intersection-heavy environments, while nuScenes provides a more diverse range of driving dynamics. To test this hypothesis, we train LMFormer on a combined dataset, containing training samples from both the

Table 2. LMFormer Cross Datadataset performance validated on nuScenes validation split

Training Dataset		$\text{minADE}_5 \downarrow$	$\text{MR}_5 \downarrow$	$\text{minFDE}_5 \downarrow$	OffRoad \downarrow
Deep Scenario	nuScenes				
✓	-	1.54	0.57	2.86	0.02
-	✓	1.13	0.48	2.13	0.01
✓	✓	1.10	0.46	2.07	0.01

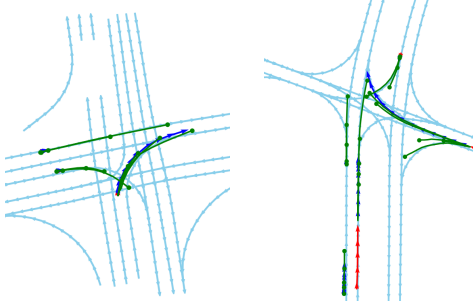


Figure 7. A depiction of marginal scene prediction. It illustrates the capability that LMFormer can be extended to multi-agent prediction.

nuScenes and Deep Scenario training sets. The corresponding qualitative results are shown in Table 2. We observe that this model checkpoint outperforms the others trained individually on Deep Scenario and nuScenes training sets. This indicates that the network can learn to generalize better with more diverse datasets and can overcome limitations present in individual datasets, as previously reported in Uni-Traj [12]. It is important to note that the network architecture as well as the number of parameters were kept constant in the cross-dataset evaluations.

In addition, we also illustrate the capability of LMFormer to predict multiple agents in parallel. Figure 7 shows a marginal scene prediction output, where the trajectories are predicted in a single forward pass for all vehicles in the scene.

4.3. Ablations

In our ablation study, we validate the two key components of our approach: (1) the refinement strategy and (2) the long-range lane encoder. To evaluate the impact of refinement, we remove the additional regression loss introduced by the refinement layers. Similarly, we remove the lane self-attention module to evaluate the role of long-range lane interactions within the Map Encoder. The results of the ablation are presented in Table 3. These results show that removing either component leads to a degradation in performance, with the degradation being slightly more pronounced when refinement losses are omitted. Overall, the ablation study confirms that both proposed components contribute significantly to SOTA performance.

Table 3. Ablation Study conducted on the nuScenes val split

Lane Self Attention	Refinement	minADE ₅ ↓	MR ₅ ↓	minFDE ₅ ↓	OffRoad↓
✓	✓	1.13	0.48	2.13	0.01
✓	-	1.16	0.51	2.20	0.01
-	✓	1.15	0.49	2.19	0.01

5. Conclusion

The paper introduces a transformer-based motion prediction module and tackles several important challenges such as trajectory refinement and lane attention. Our work establishes that lanes are the most important components in the static context for prediction tasks. Although LMFormer quantitatively achieves SOTA performance, the qualitative results show that further improvements are needed to address robustness in prediction diversity. We also note that the current evaluation benchmarks are limited in scope and could benefit from additional metrics for velocity profiles and predicted paths. Our cross-dataset evaluation highlights a potential limitation of training on intersection-centric data when generalizing to broader urban driving scenarios, but this issue could be overcome by combining samples from individual datasets.

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 multi-modal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020. 2, 3, 6
- [2] Sergio Casas, Cole Gulino, Simon Suo, Katie Luo, Renjie Liao, and Raquel Urtasun. Implicit latent variable model for scene-consistent motion forecasting. In *Computer Vision—ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIII 16*, pages 624–641. Springer, 2020. 3
- [3] Yuning Chai, Benjamin Sapp, Mayank Bansal, and Dragomir Anguelov. Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction. *CoRR*, abs/1910.05449, 2019. 2
- [4] Ming-Fang Chang, John Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, et al. Argoverse: 3d tracking and forecasting with rich maps. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8748–8757, 2019. 3
- [5] Jiuyu Chen, Zhongli Wang, Jian Wang, and Baigen Cai. Q-eonet: Implicit social modeling for trajectory prediction via experience-anchored queries. *IET Intelligent Transport Systems*, 2023. 6
- [6] Jie Cheng, Xiaodong Mei, and Ming Liu. Forecast-mae: Self-supervised pre-training for motion forecasting with masked autoencoders. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8679–8689, 2023. 3
- [7] Alexander Cui, Sergio Casas, Abbas Sadat, Renjie Liao, and Raquel Urtasun. Lookout: Diverse multi-future prediction and planning for self-driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16107–16116, 2021. 3
- [8] Henggang Cui, Vladan Radosavljevic, Fang-Chieh Chou,

- Tsung-Han Lin, Thi Nguyen, Tzu-Kuo Huang, Jeff Schneider, and Nemanja Djuric. Multimodal trajectory predictions for autonomous driving using deep convolutional networks. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 2090–2096. IEEE, 2019. 1, 2, 5
- [9] Daniel Dauner, Marcel Hallgarten, Andreas Geiger, and Kashyap Chitta. Parting with misconceptions about learning-based vehicle motion planning. In *Conference on Robot Learning*, pages 1268–1281. PMLR, 2023. 1
- [10] Nachiket Deo, Eric Wolff, and Oscar Beijbom. Multimodal trajectory prediction conditioned on lane-graph traversals. In *5th Annual Conference on Robot Learning*, 2021. 6
- [11] Chen Feng, Hangning Zhou, Huadong Lin, Zhigang Zhang, Ziyao Xu, Chi Zhang, Boyu Zhou, and Shaojie Shen. Macformer: Map-agent coupled transformer for real-time and robust trajectory prediction. *IEEE Robotics and Automation Letters*, 2023. 6
- [12] Lan Feng, Mohammadhossein Bahari, Kaouther Messaoud Ben Amor, Éloi Zablocki, Matthieu Cord, and Alexandre Alahi. Unitraj: A unified framework for scalable vehicle trajectory prediction. *arXiv preprint arXiv:2403.15098*, 2024. 3, 8
- [13] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11525–11533, 2020. 1, 2, 3
- [14] Thomas Gilles, Stefano Sabatini, Dzmitry Tsishkou, Bogdan Stanculescu, and Fabien Moutarde. Gohome: Graph-oriented heatmap output for future motion estimation. In *2022 international conference on robotics and automation (ICRA)*, pages 9107–9114. IEEE, 2022. 6
- [15] Thomas Gilles, Stefano Sabatini, Dzmitry Tsishkou, Bogdan Stanculescu, and Fabien Moutarde. THOMAS: Trajectory heatmap output with learned multi-agent sampling. In *International Conference on Learning Representations*, 2022. 6
- [16] Roger Girgis, Florian Golemo, Felipe Codevilla, Jim Aldon D’Souza, Samira Ebrahimi Kahou, Felix Heide, and Christopher J. Pal. Latent variable nested set transformers & autobots. *CoRR*, abs/2104.00563, 2021. 6
- [17] Yihan Hu, Jiazhi Yang, Li Chen, Keyu Li, Chonghao Sima, Xizhou Zhu, Siqi Chai, Senyao Du, Tianwei Lin, Wenhai Wang, et al. Planning-oriented autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17853–17862, 2023. 1
- [18] Xin Huang, Stephen G McGill, Jonathan A DeCastro, Luke Fletcher, John J Leonard, Brian C Williams, and Guy Rosman. Diversitygan: Diversity-aware vehicle motion prediction via latent semantic sampling. *IEEE Robotics and Automation Letters*, 5(4):5089–5096, 2020. 3
- [19] ByeoungDo Kim, Seong Hyeon Park, Seokhwan Lee, Elbek Khoshimjonov, Dongsuk Kum, Junsoo Kim, Jeong Soo Kim, and Jun Won Choi. Lapred: Lane-aware prediction of multimodal future trajectories of dynamic agents. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14636–14645, 2021. 1, 2
- [20] Zhiqian Lan, Yuxuan Jiang, Yao Mu, Chen Chen, and Shengbo Eben Li. Sept: Towards efficient scene representation learning for motion prediction. In *The Twelfth International Conference on Learning Representations*, 2023. 1, 3
- [21] Ming Liang, Bin Yang, Rui Hu, Yun Chen, Renjie Liao, Song Feng, and Raquel Urtasun. Learning lane graph representations for motion forecasting. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 541–556. Springer, 2020. 2, 3
- [22] 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. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2039–2049, 2024. 1, 2, 3, 6
- [23] Shilong Liu, Feng Li, Hao Zhang, Xiao Yang, Xianbiao Qi, Hang Su, Jun Zhu, and Lei Zhang. DAB-DETR: Dynamic anchor boxes are better queries for DETR. In *International Conference on Learning Representations*, 2022. 5
- [24] Yicheng Liu, Jinghuai Zhang, Liangji Fang, Qinhong Jiang, and Bolei Zhou. Multimodal motion prediction with stacked transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7577–7586, 2021. 3
- [25] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. 6
- [26] Chengjie Lu, Tao Yue, and Shaikat Ali. Deepscenario: An open driving scenario dataset for autonomous driving system testing. In *2023 IEEE/ACM 20th International Conference on Mining Software Repositories (MSR)*, pages 52–56. IEEE, 2023. 2, 6, 7
- [27] Osama Makansi, Eddy Ilg, Ozgun Cicek, and Thomas Brox. Overcoming limitations of mixture density networks: A sampling and fitting framework for multimodal future prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7144–7153, 2019. 5
- [28] Jieru Mei, Alex Zihao Zhu, Xinchun Yan, Hang Yan, Siyuan Qiao, Liang-Chieh Chen, and Henrik Kretzschmar. Waymo open dataset: Panoramic video panoptic segmentation. In *European Conference on Computer Vision*, pages 53–72. Springer, 2022. 3
- [29] Jiquan Ngiam, Vijay Vasudevan, Benjamin Caine, Zhengdong Zhang, Hao-Tien Lewis Chiang, Jeffrey Ling, Rebecca Roelofs, Alex Bewley, Chenxi Liu, Ashish Venugopal, et al. Scene transformer: A unified architecture for predicting future trajectories of multiple agents. In *International Conference on Learning Representations*, 2021. 1, 2, 3
- [30] nuScenes Prediction Challenge by Motional. eval.ai/web/challenges/challenge-page/591/. accessed: 20th Feb 2025. 6
- [31] Daehee Park, Hobin Ryu, Yunseo Yang, Jegyeong Cho, Jiwon Kim, and Kuk-Jin Yoon. Leveraging future relationship reasoning for vehicle trajectory prediction. In *The*

Eleventh International Conference on Learning Representations, 2023. [6](#)

of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 17863–17873, 2023. [1](#), [2](#), [3](#), [4](#), [5](#)

- [32] Christian Rupprecht, Iro Laina, Robert DiPietro, Maximilian Baust, Federico Tombari, Nassir Navab, and Gregory D Hager. Learning in an uncertain world: Representing ambiguity through multiple hypotheses. In *Proceedings of the IEEE international conference on computer vision*, pages 3591–3600, 2017. [5](#)
- [33] Maximilian Schäfer, Kun Zhao, and Anton Kummert. Caspnet++: Joint multi-agent motion prediction. *arXiv preprint arXiv:2308.07751*, 2023. [6](#)
- [34] Ari Seff, Brian Cera, Dian Chen, Mason Ng, Aurick Zhou, Nigamaa Nayakanti, Khaled S Refaat, Rami Al-Rfou, and Benjamin Sapp. Motionlm: Multi-agent motion forecasting as language modeling. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8579–8590, 2023. [1](#), [3](#)
- [35] Zhigang Sun, Zixu Wang, Lavdim Halilaj, and Juergen Luetin. Semanticformer: Holistic and semantic traffic scene representation for trajectory prediction using knowledge graphs. *arXiv preprint arXiv:2404.19379*, 2024. [1](#), [2](#), [3](#), [6](#), [7](#)
- [36] Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in neural information processing systems*, 33:7537–7547, 2020. [4](#)
- [37] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. [4](#)
- [38] Jingke Wang, Tengju Ye, Ziqing Gu, and Junbo Chen. Ltp: Lane-based trajectory prediction for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17134–17142, 2022. [1](#), [2](#), [3](#)
- [39] Benjamin Wilson, William Qi, Tanmay Agarwal, John Lambert, Jagjeet Singh, Siddhesh Khandelwal, Bowen Pan, Ratnesh Kumar, Andrew Hartnett, Jhony Kaesemodel Pontes, et al. Argoverse 2: Next generation datasets for self-driving perception and forecasting. *arXiv preprint arXiv:2301.00493*, 2023. [3](#)
- [40] Harsh Yadav, Maximilian Schaefer, Kun Zhao, and Tobias Meisen. Caspformer: Trajectory prediction from bev images with deformable attention. In *International Conference on Pattern Recognition*, pages 420–434. Springer, 2025. [1](#), [5](#), [6](#)
- [41] Yang Zhou, Hao Shao, Letian Wang, Steven L Waslander, Hongsheng Li, and Yu Liu. Smartrefine: A scenario-adaptive refinement framework for efficient motion prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15281–15290, 2024. [2](#)
- [42] Zikang Zhou, Luyao Ye, Jianping Wang, Kui Wu, and Kejie Lu. Hivt: Hierarchical vector transformer for multi-agent motion prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8823–8833, 2022. [2](#), [3](#), [5](#)
- [43] Zikang Zhou, Jianping Wang, Yung-Hui Li, and Yu-Kai Huang. Query-centric trajectory prediction. In *Proceedings*