MDPI

Article

Local Lane Graph Conditioning as a General Inductive Bias for Trajectory Prediction: A Multi-Architecture Study on the Waymo Open Motion Dataset

Xingnan Zhou [1] and Ciprian Alecsandru [1,*]

[1] Department of Building, Civil and Environmental Engineering, Concordia University, Montreal, QC H3G 1M8, Canada

* Correspondence: ciprian.alecsandru@concordia.ca (C.A.)

Academic Editors: Jinlei Zhang and Wei Ma
Version February 12, 2026 submitted to Sustainability

Abstract: Accurate trajectory prediction is critical for both autonomous driving safety and energy-efficient motion planning in sustainable urban mobility systems. While state-of-the-art methods employ complex architectures with hundreds of input features, the contribution of individual components—particularly local road topology—remains difficult to isolate. This study investigates whether local lane graph conditioning provides a *general, architecture-agnostic* improvement, using only minimal position inputs and a lightweight lane encoder. We propose a *waterflow* lane graph extraction method that constructs an ego-centric lane topology through breadth-first traversal of the HD map, fusing lane features into trajectory encoders via cross-attention. We evaluate across two encoder architectures (LSTM and Transformer), two prediction horizons (3 s and 8 s), and both single- and multi-modal (K=6) settings on the Waymo Open Motion Dataset (89,258 signal-controlled intersection scenarios). Lane graph conditioning consistently improves accuracy: +9.3% ADE at 3 s (LSTM, p=0.007), +26.7% minADE at 8 s (LSTM, K=6, p=0.003), and +32.0% ADE at 8 s (Transformer)—with approximately 8% additional parameters for LSTM. Error decomposition reveals improvements in both lateral (+26.5%) and longitudinal (+25.4%) components, with endpoint lateral error showing the largest reduction (+30.5%). Our lane-conditioned LSTM achieves a mean minADE of 1.37 ± 0.08 m at 8 s for ego-vehicle prediction, numerically comparable to the Waymo official LSTM baseline (1.34 m, vehicle class) that uses the full feature set—though we note that our ego-only single-agent task is simpler than Waymo's multi-agent marginal prediction setting. These findings support local lane graph conditioning as a lightweight, general-purpose module for trajectory prediction in safety-critical urban environments.

Keywords: trajectory prediction; lane graph; autonomous driving; Waymo Open Motion Dataset; LSTM; Transformer; multi-modal prediction; traffic safety; sustainable transportation

1. Introduction

Urban intersections represent one of the most safety-critical environments in road transportation networks. According to the National Highway Traffic Safety Administration (NHTSA), approximately 40% of all police-reported crashes in the United States occur at or near intersections [1], resulting in significant human, economic, and environmental costs. Each collision-induced traffic disruption propagates congestion through the surrounding network, increasing vehicle idle times, fuel consumption, and greenhouse gas emissions [2]. For connected and automated vehicles, Taiebat et al. [3] identify trajectory prediction as a key enabling technology with cascading implications for

energy efficiency, emissions reduction, and traffic safety. Consequently, improving the ability of intelligent transportation systems to anticipate vehicle movements is a key pathway toward more sustainable urban mobility [4,5].

Trajectory prediction—the task of forecasting future positions of traffic agents given their observed motion history and environmental context—is a foundational capability for autonomous vehicles and advanced driver-assistance systems [6,7]. Accurate trajectory forecasts enable proactive safety interventions such as emergency braking and cooperative maneuver planning, reducing collision risk and its downstream sustainability impacts [8].

A key insight driving recent progress is that road map information—particularly lane geometry and connectivity—provides strong *relational inductive biases* [9] for predicting where vehicles are likely to travel [10–12]. Methods such as LaneGCN [11] use lane graphs as the primary backbone representation, while HiVT [13] and PGP [14] hierarchically fuse lane topology with agent dynamics. State-of-the-art methods such as MTR [15] and QCNet [16] achieve remarkable performance by jointly encoding rich agent features (velocity, acceleration, heading, bounding box dimensions) with dense map representations using large Transformer architectures. However, the entanglement of multiple input features and architectural innovations makes it difficult to isolate the specific contribution of lane topology information.

This paper addresses this gap through a controlled study that isolates the effect of local lane graph conditioning. Unlike LaneGCN [11], which builds the entire architecture around lane graph representations, we treat lane conditioning as a *modular plug-in* and evaluate it across multiple backbone architectures to assess its generality. We deliberately employ simple backbones—an LSTM encoder-decoder and a vanilla Transformer encoder—with minimal input features (2D position only), so that any observed improvement can be directly attributed to the lane conditioning module. Our key research question is: *Does local lane graph conditioning provide a consistent, architecture-agnostic improvement for trajectory prediction?*

We answer this question affirmatively through comprehensive experiments on the Waymo Open Motion Dataset [17], evaluating across two architectures, two prediction horizons, and both single-modal and multi-modal prediction settings. Our contributions are as follows:

1. We propose a *waterflow* lane graph extraction method that constructs a local, ego-centric lane topology through breadth-first traversal of the HD map, and a lightweight lane encoder with graph message passing and cross-attention fusion.
2. We demonstrate that lane conditioning provides a **consistent, architecture-agnostic improvement**: +9.3% ADE reduction for LSTM at 3 s (p=0.007, 3 seeds), +26.7% minADE reduction for multi-modal LSTM at 8 s (p=0.003, 3 seeds), and +32.0% ADE reduction for Transformer at 8 s (seed 42).
3. We show that the benefit of lane conditioning **increases with prediction horizon** (from +9.3% at 3 s to +26.7% at 8 s), confirming that lane structure becomes increasingly valuable as kinematic extrapolation degrades over longer horizons.
4. Through error decomposition analysis, we reveal that lane conditioning improves both lateral (+26.5%) and longitudinal (+25.4%) error components, with the strongest improvement at trajectory endpoints (lateral FDE: +30.5%).
5. We show that our lane-conditioned LSTM with K=6 modes achieves a mean minADE of 1.37 ± 0.08 m at 8 s for ego-vehicle prediction, **numerically comparable to the Waymo official LSTM baseline** (1.34 m, vehicle class) [17] that uses the full feature set—while using only 2D position inputs plus local lane features. We note that our ego-only single-agent task is not directly comparable to Waymo's multi-agent marginal prediction setting (see Section 5.7).

By demonstrating that a lightweight lane conditioning module (<700,000 parameters) can match the accuracy of models requiring rich hand-engineered features, our results point toward more computationally efficient prediction systems—a direct contribution to sustainable autonomous driving

through reduced onboard energy consumption and accessible deployment on resource-constrained platforms.

2. Related Work

2.1. Recurrent Approaches to Trajectory Prediction

The application of LSTM networks [18] to trajectory prediction was pioneered by Alahi et al. [19], who introduced Social LSTM with a social pooling mechanism for pedestrian interactions. Park et al. [20] adapted the encoder-decoder LSTM architecture for vehicle trajectory prediction, demonstrating that sequence-to-sequence models effectively capture temporal dynamics. Deo and Trivedi [21] proposed convolutional social pooling for highway lane-change prediction. These methods established the core encoder-decoder paradigm upon which our LSTM baseline builds.

Graph-based extensions to recurrent models have been proposed to capture dynamic agent interactions. Chandra et al. [22] employed graph-LSTMs with spectral clustering, while Li et al. [23] introduced EvolveGraph for dynamic relational reasoning. Mo et al. [24] combined graph neural networks with recurrent architectures for highway prediction. These works demonstrate the value of structured relational reasoning but primarily target highway scenarios.

2.2. Transformer-Based Approaches

Transformer architectures [25] have achieved state-of-the-art performance on large-scale motion forecasting benchmarks. Scene Transformer [26] proposed a unified multi-agent prediction architecture. Wayformer [27] demonstrated efficient attention-based forecasting. MTR [15] introduced motion transformers with global intention localization, and QCNet [16] proposed query-centric prediction. HiVT [13] uses a hierarchical architecture that combines local agent–lane interactions with global scene-level attention. These methods typically combine Transformer attention with rich input features and map representations, making it difficult to attribute improvements to specific components. Notably, Zeng et al. [28] showed that simple models can outperform Transformers for time series forecasting, suggesting that the Transformer's advantage depends on the availability of rich contextual information. Our work complements these efforts by isolating the lane conditioning component within a simple Transformer encoder.

2.3. Map-Aware and Lane-Conditioned Methods

The integration of HD map information has emerged as a critical factor in prediction performance. VectorNet [10] proposed a unified vectorized representation for agent trajectories and map elements. LaneGCN [11] introduced lane graph representations with graph convolutions [29] along lane connectivity structures. LaneRCNN [30], from the same group, extended this with distributed lane-centric representations and actor–lane interaction graphs. TNT [12] and DenseTNT [31] leveraged lane centerlines as target candidates. LaPred [32] explicitly conditioned predictions on lane-level features. PGP [14] conditions multi-modal predictions on discrete lane-graph traversals, treating lane connectivity as a tree of possible goals—an approach philosophically similar to our waterflow extraction but using graph traversals for mode generation rather than feature conditioning. GANet [33] uses lane-level goal areas for multi-modal forecasting, demonstrating that lane structure constrains plausible endpoints.

While these methods demonstrate the value of lane information, they employ complex architectures where the lane component is deeply integrated with the rest of the model, making it difficult to isolate the lane contribution. In particular, LaneGCN builds the entire architecture around the lane graph as the primary backbone, whereas we treat lane conditioning as a *modular plug-in* that can be attached to arbitrary backbone architectures. This distinction is important: our controlled study tests whether lane conditioning generalizes across fundamentally different encoder architectures (recurrent and attention-based), using a message passing formulation [34] for lane feature propagation

and cross-attention [35] for fusion. By analogy with the relational inductive bias framework of Battaglia et al. [9], our lane graph provides a structured prior that constrains the model's hypothesis space without dictating the overall architecture.

2.4. Multi-Modal Prediction

Vehicle trajectories at intersections are inherently multi-modal, as drivers may turn left, go straight, or turn right at the same intersection. Gupta et al. [36] introduced Social GAN, using generative adversarial training to produce diverse trajectory samples. Winner-takes-all (WTA) training [37] provides a simpler alternative, assigning each ground truth to the closest prediction mode and backpropagating through that mode only. Salzmann et al. [38] used conditional variational autoencoders in Trajectron++ for probabilistic multi-modal prediction. The Waymo Motion Prediction Challenge [17] evaluates methods using minADE and minFDE (minimum over K modes), making multi-modal prediction essential for benchmark evaluation. We adopt WTA training with K=6 modes to enable direct comparison with Waymo benchmarks.

3. Methodology

This section describes the problem formulation, the waterflow lane graph, and the model architectures.

3.1. Problem Formulation

Given the observed trajectory of an ego vehicle over $T_{\text{obs}} = 11$ timesteps (1.1 s at 10 Hz), the observed trajectories of up to $N = 10$ neighboring agents within a 30 m radius, and optionally a local lane graph, the goal is to predict the ego vehicle's future trajectory. We evaluate two prediction horizons: $T_{\text{pred}} = 30$ (3.0 s) and $T_{\text{pred}} = 80$ (8.0 s). All positions are in a bird's-eye-view coordinate frame centered on the ego vehicle's last observed position and aligned with its heading direction.

In the single-modal setting, the model outputs one predicted trajectory $\hat{\mathbf{Y}} \in \mathbb{R}^{T_{\text{pred}} \times 2}$. In the multi-modal setting, the model outputs $K = 6$ trajectory hypotheses $\{\hat{\mathbf{Y}}_k\}_{k=1}^{K}$ with associated confidence scores $\{c_k\}_{k=1}^{K}$, where $\sum_k c_k = 1$.

3.2. Waterflow Lane Graph Extraction

To incorporate local road topology, we extract a structured lane graph from the HD map. We term this the *waterflow* graph because the extraction models the directional propagation of traffic flow potential from the ego vehicle outward through the lane connectivity network. Unlike standard undirected graph expansions, the waterflow traversal respects the directionality of lane successors (forward connectivity) while also capturing lateral alternatives, mirroring how traffic flow possibilities radiate outward from a vehicle's current position.

3.2.1. Graph Construction

The extraction proceeds in four stages:

1. **Ego Lane Identification.** The lane whose centerline passes closest to the vehicle's last observed position (within 5 m) is selected as the ego lane. If no lane centerline lies within this threshold, the system falls back to the nearest lane by Euclidean distance. In our signal-controlled subset, valid ego lanes are identified in >98% of scenarios, ensuring the lane conditioning module is active in the vast majority of cases.
2. **Breadth-First Expansion.** Starting from the ego lane, a 3-hop BFS traverses successor lanes (forward connectivity), left-adjacent lanes, and right-adjacent lanes.
3. **Truncation.** The subgraph is truncated to $L_{\max} = 16$ lanes, prioritized by topological proximity.
4. **Feature Extraction.** For each lane ℓ, a feature vector $\mathbf{f}_\ell \in \mathbb{R}^{26}$ is computed.

Algorithm 1 formalizes this procedure. The BFS traversal uses a visited set to naturally handle cycles (e.g., roundabouts), and the $L_{\max}$ bound in the outer loop ensures that the 16-lane limit takes precedence over the 3-hop limit when many adjacent lanes are present at wide intersections. Ties among equidistant lanes are broken by BFS visit order (first-in, first-out).

Algorithm 1: Waterflow Lane Graph Extraction

Input: HD map $\mathcal{M}$, ego position $\mathbf{p}_{\text{ego}}$, $h_{\max}$=3, $L_{\max}$=16
Output: Lane features $\mathbf{F} \in \mathbb{R}^{L_{\max} \times 26}$, adjacency $\mathbf{A} \in \{0,1\}^{L_{\max} \times L_{\max}}$, mask $\mathbf{m}$

$\ell_0 \leftarrow \arg\min_{\ell \in \mathcal{M}} \text{dist}(\text{centerline}(\ell), \mathbf{p}_{\text{ego}})$; // Ego lane
if $\text{dist}(\text{centerline}(\ell_0), \mathbf{p}_{\text{ego}}) > 5\,\text{m}$ **then**
  $\ell_0 \leftarrow$ nearest lane by Euclidean distance ; // Fallback
end
$\mathcal{Q} \leftarrow \{(\ell_0, 0)\}$; // Queue: (lane, hop count)
$\mathcal{V} \leftarrow \{\ell_0\}$; // Visited set
while $\mathcal{Q} \neq \emptyset$ ***and*** $|\mathcal{V}| < L_{\max}$ **do**
  $(\ell, h) \leftarrow \mathcal{Q}$.dequeue();
  if $h < h_{\max}$ **then**
    foreach $\ell' \in \text{Succ}(\ell) \cup \text{LeftAdj}(\ell) \cup \text{RightAdj}(\ell)$ **do**
      if $\ell' \notin \mathcal{V}$ ***and*** $|\mathcal{V}| < L_{\max}$ **then**
        $\mathcal{V} \leftarrow \mathcal{V} \cup \{\ell'\}$;
        $\mathcal{Q}$.enqueue$((\ell', h+1))$;
      end
    end
  end
end
foreach $\ell \in \mathcal{V}$ **do**
  $\mathbf{f}_\ell \leftarrow [\mathbf{c}_\ell \,\|\, \mathbf{d}_\ell \,\|\, s_\ell \,\|\, \mathbf{b}_\ell]$; // Eq. (1)
end
Construct $\mathbf{A}$ from lane connectivity within $\mathcal{V}$; set $\mathbf{m}$;
return $\mathbf{F}$, $\mathbf{A}$, $\mathbf{m}$

Figure 1 illustrates the progressive expansion of the waterflow graph from the ego lane through three hops, showing how local lane topology is incrementally captured.

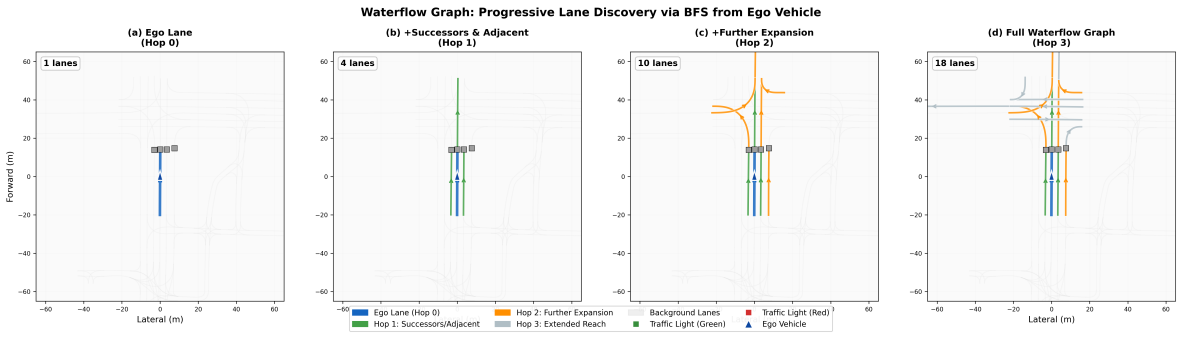

Figure 1. Waterflow lane graph extraction via breadth-first expansion. Starting from the ego lane (Hop 0), the graph progressively incorporates successor and adjacent lanes through 3 hops, capturing the local road topology relevant to trajectory prediction. This intersection example discovers 18 lanes within 3 hops; the algorithm retains the first $L_{\max}$=16 by BFS visit order.

3.2.2. Lane Feature Representation

Each lane feature vector comprises:

$$\mathbf{f}_\ell = [\mathbf{c}_\ell \,\|\, \mathbf{d}_\ell \,\|\, s_\ell \,\|\, \mathbf{b}_\ell], \tag{1}$$

where $\mathbf{c}_\ell \in \mathbb{R}^{20}$ contains the flattened (x, y) coordinates of 10 resampled centerline points in the ego-centric frame, $\mathbf{d}_\ell \in \mathbb{R}^2$ is the normalized direction vector, $s_\ell \in \mathbb{R}$ is the normalized lane length, and $\mathbf{b}_\ell \in \{0,1\}^3$ encodes three boolean flags: ego lane, traffic signal controlled, and stop sign present.

The lane adjacency matrix $\mathbf{A} \in \{0,1\}^{L_{\max} \times L_{\max}}$ encodes undirected connectivity, and a validity mask $\mathbf{m} \in \{0,1\}^{L_{\max}}$ indicates which slots contain valid lanes.

3.3. Architecture Overview

Figure 2 provides an overview of the four model variants evaluated in this study. All models share the same neighbor encoder, fusion layer, and CV-residual decoder; only the ego encoder (LSTM vs. Transformer) and the optional lane conditioning module differ. The lane conditioning module (highlighted in green) is a modular plug-in that can be attached to either backbone architecture, enabling a controlled ablation of its effect.

3.4. Model A: LSTM Baseline

The baseline model follows a standard encoder-decoder architecture with three components.

3.4.1. Ego Trajectory Encoder

The ego history is projected to a 64-dimensional embedding and processed by a 2-layer LSTM with hidden dimension $d_h = 128$:

$$\mathbf{e}_t = \mathrm{ReLU}(\mathbf{W}_e \mathbf{x}_t + \mathbf{b}_e), \quad \mathbf{h}_t, \mathbf{c}_t = \mathrm{LSTM}(\mathbf{e}_t, \mathbf{h}_{t-1}, \mathbf{c}_{t-1}). \tag{2}$$

3.4.2. Neighbor Context Encoder

Each neighbor's trajectory is independently encoded using a smaller 1-layer LSTM (hidden dimension 64). Neighbor representations are aggregated via masked max-pooling:

$$\mathbf{n}_{\text{ctx}} = \max_{i:m_i=1} (\mathrm{LSTM}_{\text{nbr}}(\mathbf{X}_i^{\text{nbr}})) \in \mathbb{R}^{64}. \tag{3}$$

3.4.3. Fusion and CV-Residual Decoder

The ego hidden state and neighbor context are concatenated and fused:

$$\mathbf{z} = \mathrm{ReLU}(\mathbf{W}_f[\mathbf{h}_{T_{\text{obs}}} \,\|\, \mathbf{n}_{\text{ctx}}] + \mathbf{b}_f) \in \mathbb{R}^{128}. \tag{4}$$

A three-layer MLP maps $\mathbf{z}$ to displacement residuals $\Delta\mathbf{Y} \in \mathbb{R}^{T_{\text{pred}} \times 2}$, added to a constant-velocity baseline to form a CV-residual prediction:

$$\hat{\mathbf{y}}_t = (\mathbf{p}_{\text{last}} + \mathbf{v}_{\text{last}} \cdot t) + \Delta\mathbf{y}_t. \tag{5}$$

Here $\mathbf{v}_{\text{last}} = \mathbf{p}_{T_{\text{obs}}} - \mathbf{p}_{T_{\text{obs}}-1}$ is the velocity estimated from the last two observed positions. This CV-residual formulation provides a strong inductive bias for approximately linear motion segments, following the insight of Schöller et al. [39] that constant-velocity prediction serves as a surprisingly strong baseline.

3.5. Model B: Lane-Conditioned LSTM

The lane-conditioned model extends the LSTM baseline by incorporating local lane graph information through two additional components: a lane encoder with graph message passing [34] and cross-attention pooling [35]. The ego encoder, neighbor encoder, and CV-residual decoder remain

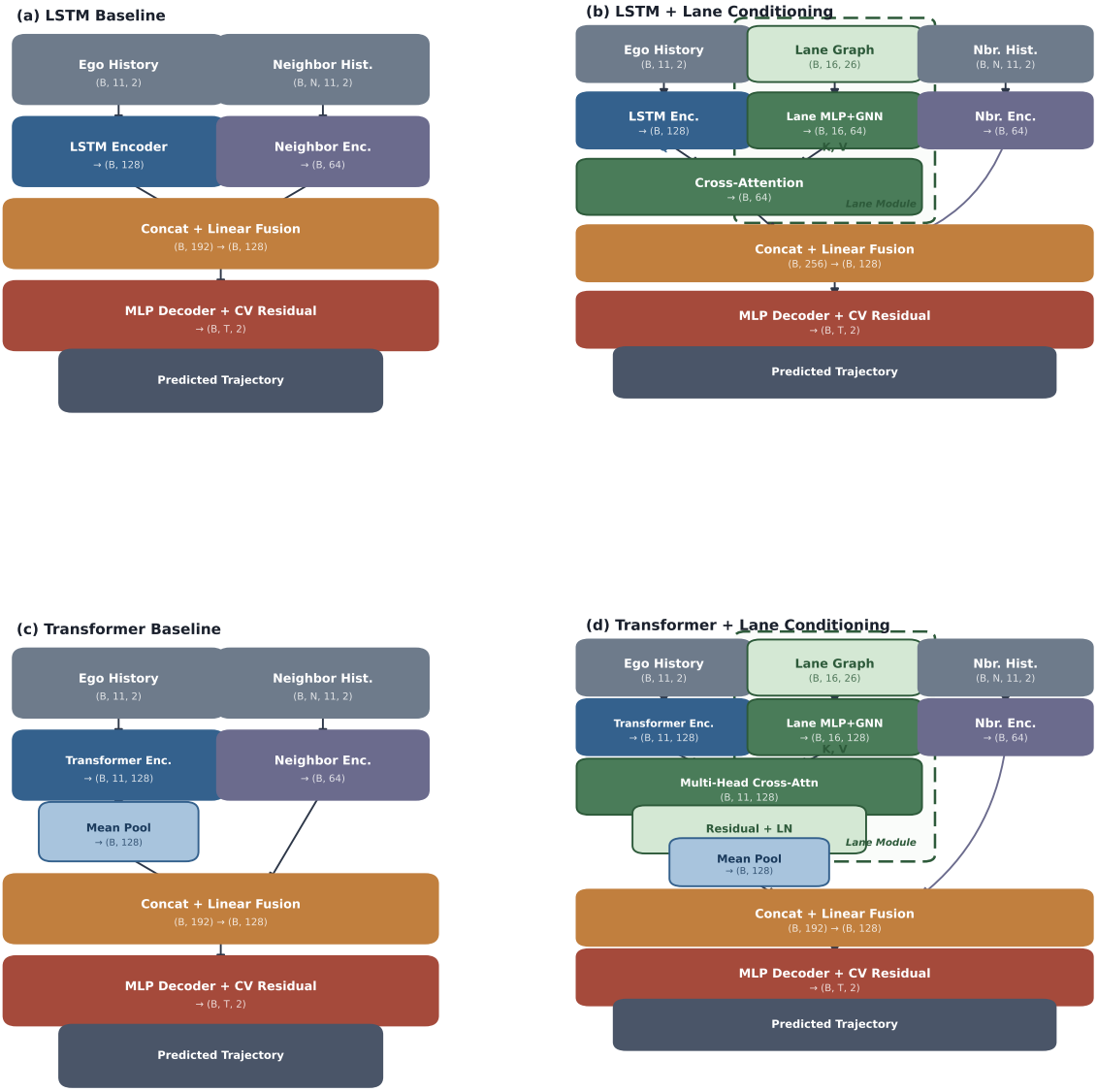

Figure 2. Architecture overview of the four model variants. **Left column**: baselines without lane information. **Right column**: lane-conditioned variants with the lane conditioning module (green dashed box) added as a modular plug-in. The LSTM variants (top row) use single-vector cross-attention pooling, while the Transformer variants (bottom row) use multi-head cross-attention over the full ego sequence to capture timestep-level lane–trajectory interactions.

identical, enabling a controlled ablation where any performance difference is directly attributable to the lane conditioning module.

3.5.1. Lane Encoder with Message Passing

Each lane feature vector is embedded into a 64-dimensional representation through a two-layer MLP. Two rounds of message passing propagate structural information along the lane graph:

$$\mathbf{l}_\ell^{(k+1)} = \mathrm{ReLU}\Big(\mathbf{W}^{(k)}[\mathbf{l}_\ell^{(k)} \,\|\, (\tilde{\mathbf{A}}\mathbf{L}^{(k)})_\ell]\Big), \tag{6}$$

where $\tilde{\mathbf{A}} = \mathbf{D}^{-1}\mathbf{A}$ is the degree-normalized adjacency matrix.

3.5.2. Cross-Attention Pooling

Lane embeddings are aggregated using cross-attention with the ego hidden state as the query:

$$\alpha_\ell = \frac{\exp(\mathbf{q}^\top \mathbf{k}_\ell / \sqrt{d_k})}{\sum_{j:m_j=1} \exp(\mathbf{q}^\top \mathbf{k}_j / \sqrt{d_k})}, \quad \mathbf{l}_{\text{ctx}} = \sum_\ell \alpha_\ell \mathbf{v}_\ell \in \mathbb{R}^{64}. \tag{7}$$

The lane context is concatenated with the ego and neighbor representations before fusion:

$$\mathbf{z} = \mathrm{ReLU}(\mathbf{W}_f[\mathbf{h}_{T_{\text{obs}}} \,\|\, \mathbf{l}_{\text{ctx}} \,\|\, \mathbf{n}_{\text{ctx}}]) \in \mathbb{R}^{128}. \tag{8}$$

3.6. Model C: Transformer Baseline

To test the generality of lane conditioning across architecturally distinct backbones, we replace the LSTM ego encoder with a Transformer encoder [25]. This allows us to determine whether the benefits of lane conditioning are specific to the LSTM's sequential inductive bias or transfer to the Transformer's attention-based paradigm.

3.6.1. Transformer Ego Encoder

The ego history is projected to $d_{\text{model}} = 128$ dimensions, combined with learnable positional embeddings, and processed by a 2-layer Transformer encoder with 4 attention heads:

$$\mathbf{E} = \mathrm{TransformerEncoder}(\mathbf{W}_p \mathbf{X}^{\text{ego}} + \mathbf{P}_{\text{pos}}) \in \mathbb{R}^{T_{\text{obs}} \times d_{\text{model}}}. \tag{9}$$

Unlike the LSTM, which compresses the sequence into a single hidden vector, the Transformer produces a full sequence representation. The ego representation is obtained by mean pooling:

$$\mathbf{h}_{\text{ego}} = \frac{1}{T_{\text{obs}}} \sum_{t=1}^{T_{\text{obs}}} \mathbf{E}_t \in \mathbb{R}^{d_{\text{model}}}. \tag{10}$$

The neighbor encoder, fusion, and CV-residual decoder are identical to the LSTM baseline.

3.7. Model D: Lane-Conditioned Transformer

The key advantage of the Transformer for lane conditioning is that it produces a *full sequence representation* $\mathbf{E} \in \mathbb{R}^{T_{\text{obs}} \times d_{\text{model}}}$, enabling multi-head cross-attention between all ego timesteps and lane embeddings simultaneously. This contrasts with the LSTM variant (Model B), which compresses the trajectory into a single vector before attending to lane features.

3.7.1. Multi-Head Cross-Attention

Lane features are projected to $d_{\text{model}} = 128$ dimensions via a two-layer MLP, followed by graph message passing. The ego sequence queries the lane embeddings through multi-head cross-attention:

$$\mathbf{C} = \text{MultiHeadAttn}(\mathbf{Q}{=}\mathbf{E},\ \mathbf{K}{=}\mathbf{L},\ \mathbf{V}{=}\mathbf{L}) \in \mathbb{R}^{T_{\text{obs}} \times d_{\text{model}}}, \tag{11}$$

with 4 attention heads. A residual connection and layer normalization produce the lane-conditioned sequence:

$$\tilde{\mathbf{E}} = \text{LayerNorm}(\mathbf{E} + \mathbf{C}). \tag{12}$$

The ego representation is obtained by mean pooling over $\tilde{\mathbf{E}}$, followed by the same neighbor fusion and decoder.

3.8. Multi-Modal Extension

At urban intersections, vehicles face multiple plausible futures (e.g., turning left, going straight, turning right). To capture this multi-modality, we extend the LSTM models with $K = 6$ prediction heads for the 8-second horizon; the Transformer experiments at 8 s use single-modal prediction. Each mode has an independent MLP decoder producing a trajectory with CV-residual decoding (Equation (5)). A shared confidence head maps the fused representation to mode probabilities via softmax.

We use a WTA loss where only the mode closest to the ground truth receives gradient:

$$k^* = \arg\min_k \frac{1}{T} \sum_t \|\hat{\mathbf{y}}_{k,t} - \mathbf{y}_t^*\|_2, \quad \mathcal{L} = \text{SmoothL1}(\hat{\mathbf{Y}}_{k^*}, \mathbf{Y}^*) - \log c_{k^*}, \tag{13}$$

where SmoothL1 denotes the Huber loss [40].

3.9. Model Complexity

Table 1 summarizes the parameter counts for all model variants. Lane conditioning adds fewer than 50,000 parameters (approximately 8%) to the LSTM backbone, from the lane MLP projection, message passing weights, and cross-attention parameters. The Transformer lane-conditioned variant shows a larger overhead (+33.3%) because the lane encoder dimension matches the Transformer's wider d_{model}=128. All models remain under 700,000 parameters—orders of magnitude smaller than state-of-the-art methods.

Table 1. Model parameter counts.

Model	Parameters	Overhead
LSTM Baseline (single)	582,562	Ref.
LSTM Lane-Cond. (single)	629,698	+8.1%
LSTM Lane-Cond. (K=6)	679,618	+16.7%*
Transformer Baseline	456,576	Ref.
Transformer Lane-Cond.	608,640	+33.3%

*Overhead relative to LSTM Baseline (single).

4. Experimental Setup

4.1. Dataset and Preprocessing

We use the Waymo Open Motion Dataset (WOMD) v1.1.0 [17], one of the largest public motion forecasting benchmarks, with a training set of approximately 487,000 scenarios spanning diverse urban environments. Each scenario covers 9.1 s at 10 Hz (91 frames), providing both agent trajectories and HD map information including lane centerlines, connectivity, traffic signals, and stop signs.

We process a representative sample of 123,031 scenarios from 252 of the 1,000 training shards (approximately 25% of the full training set), following the dataset's original random ordering. From

these, we select 89,258 signal-controlled scenarios (72.5% of the processed sample) where structured lane graphs with successor and adjacent connectivity are available. We focus on signal-controlled intersections for three reasons: (a) these urban environments exhibit the richest lane topology, with multi-way turning movements that the waterflow graph is designed to capture; (b) multi-modal trajectory behavior is most prevalent at signalized intersections, where vehicles may turn left, proceed straight, or turn right; and (c) intersection safety is a primary motivation for this work, as approximately 40% of crashes occur at or near intersections [1]. A 15% random split (seed 42) yields 75,869 training and 13,389 validation scenarios. Because we use a custom subset and split rather than the official WOMD train/val/test partition, absolute numbers should not be directly compared with leaderboard results. However, all internal comparisons (baseline vs. lane-conditioned) use identical data splits, ensuring fair evaluation of the lane conditioning effect.

The prediction target is the **ego vehicle (self-driving car, SDC) trajectory only**: each sample produces one trajectory prediction for the autonomous vehicle, with up to 10 nearby agents used as context (neighbor history). This is a single-agent self-prediction task rather than multi-agent marginal prediction. For 3-second prediction, each scenario yields up to six samples using anchor frames at indices $\{10, 20, 30, 40, 50, 60\}$, resulting in approximately 450,000 training samples. For 8-second prediction, only one anchor (index 10) is valid due to the 9.1 s scenario length, yielding approximately 75,000 training samples. All trajectories are transformed to an ego-centric bird's-eye-view coordinate system aligned with the ego vehicle's heading. Data augmentation includes random 360° rotation of the entire scene (trajectories, neighbors, and lane features), which prevents the model from memorizing absolute orientations.

4.2. Training Details

All models are trained using AdamW [41] with gradient clipping at norm 1.0. LSTM models use learning rate 10^{-3} with weight decay 10^{-4}. Transformer models use 5×10^{-4} with weight decay 10^{-2} and 5 epochs of linear warmup. All models use cosine annealing [42] over 100 epochs with early stopping (patience 20). Batch size is 128.

For both 3-second and 8-second experiments, we train with three random seeds (7, 42, 123) and report mean $\pm$ standard deviation with paired t-test p-values. Transformer experiments at 8 s use seed 42 due to computational constraints.

All experiments are conducted on a single NVIDIA RTX 4090 GPU with 24 GB VRAM.

4.3. Evaluation Metrics

- **ADE**: mean L_2 distance over all future timesteps.
- **FDE**: L_2 distance at the last predicted timestep.
- **minADE / minFDE**: minimum ADE / FDE over K modes.
- **Miss Rate (MR@dm)**: fraction where best-mode endpoint error exceeds d meters.

For error decomposition, we project errors onto the heading direction (longitudinal) and perpendicular axis (lateral). Throughout this paper, improvements are reported as percentage reductions in error (e.g., "+9.3%" denotes a 9.3% lower ADE).

5. Results

5.1. 3-Second Prediction: Multi-Seed Validation

We first validate the lane conditioning effect at the shorter 3-second horizon using three random seeds to establish statistical significance. Table 2 presents the results.

Lane conditioning achieves a statistically significant 9.3% ADE reduction ($p = 0.0071$, paired t-test). The improvement is consistent across all three seeds: +10.9% (seed 7), +9.1% (seed 42), +8.1% (seed 123).

Table 2. Single-modal prediction at 3 s (89K scenes, 100 epochs, 3 seeds).

Model	ADE@3 s (m)	Best Seed	p-value
LSTM Baseline	0.559 ± 0.007	0.552	—
LSTM Lane-Cond.	$\mathbf{0.507 \pm 0.011}$	**0.496**	**0.0071**
Improvement	+9.3%		

5.2. 8-Second Multi-Modal Results

The 8-second multi-modal setting (K=6) represents the primary benchmark comparison. Note that we predict only the ego vehicle (SDC) trajectory per scene, whereas the Waymo Motion Prediction Challenge evaluates marginal predictions for up to 8 agents per scene (vehicles, pedestrians, and cyclists). Table 3 presents the results.

Table 3. Multi-modal prediction (K=6) at 8 s (3 seeds, 100 epochs). Values are mean ± std.

Model	minADE (m)	minFDE (m)
LSTM Baseline	1.868 ± 0.042	5.047 ± 0.106
LSTM Lane-Cond.	$\mathbf{1.371 \pm 0.081}$	$\mathbf{3.403 \pm 0.242}$
Improvement	+26.7% (p=0.003)	+32.6% (p=0.004)

The improvement at 8 s is substantially larger than at 3 s and statistically significant across three seeds ($p < 0.005$ for both metrics). The minFDE improvement of 32.6% confirms that lane structure becomes *more* valuable at longer horizons, where endpoint accuracy matters most.

5.3. Architecture-Agnostic Benefit

Table 4 compares the effect of lane conditioning across LSTM and Transformer architectures at 8 s (single-modal).

Table 4. Single-modal prediction at 8 s across architectures (seed 42, 100 epochs).

Model	ADE (m)	FDE (m)	ADE@3 s (m)	LC Improv.
LSTM Baseline	3.781	11.244	0.553	—
LSTM Lane-Cond.	**3.075**	**8.688**	**0.516**	+18.7%
TF Baseline	4.859	13.875	0.828	—
TF Lane-Cond.	**3.303**	**8.956**	**0.663**	+32.0%

Lane conditioning provides a consistent improvement for *both* architectures: +18.7% for LSTM and +32.0% for Transformer. The LSTM outperforms the Transformer in absolute terms, which is expected with only 11 input timesteps ($T_{\text{obs}} = 11$): the LSTM's recurrent inductive bias for short sequential processing likely outweighs the Transformer's general-purpose attention at this limited sequence length, consistent with findings that simple models can match or outperform Transformers for short time series [20,28].

5.4. Horizon-Dependent Improvement

Table 5 summarizes the improvement from lane conditioning across all experimental settings.

The improvement increases consistently from 3 s to 8 s and from ADE to FDE to miss rate, revealing that lane conditioning most strongly benefits trajectory *endpoints*—consistent with the intuition that lane structure constrains where vehicles can plausibly end up.

Figure 3 visualizes the error growth over the 8-second horizon, showing how the gap between baseline and lane-conditioned models widens progressively. Figure 4 further decomposes the

Table 5. Lane conditioning improvement across horizons and settings.

Setting	Horizon	Metric	Improvement
LSTM, single, 3 seeds	3 s	ADE	+9.3% (p=0.007)
LSTM, single	8 s	ADE	+18.7%
LSTM, K=6, 3 seeds	8 s	minADE	+26.7% (p=0.003)
LSTM, K=6, 3 seeds	8 s	minFDE	+32.6% (p=0.004)
LSTM, K=6, seed 42	8 s	MR@5 m	+42.7%
Transformer, single	8 s	ADE	+32.0%

improvement by horizon and error axis, confirming that both longitudinal and lateral components benefit, with the improvement growing monotonically from 1 s to 7 s before a slight plateau at 8 s.

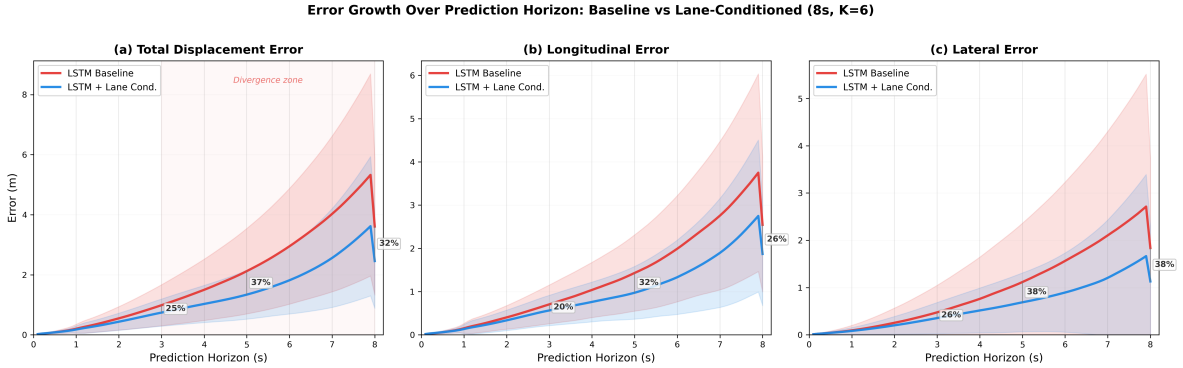

Figure 3. Error growth over prediction horizon for LSTM baseline vs. lane-conditioned model (K=6, 8 s). Left: total L2 error. Center: longitudinal error. Right: lateral error. The gap between models widens with horizon, confirming that lane conditioning becomes increasingly valuable for longer-term prediction.

5.5. Error Decomposition

Table 6 decomposes errors into lateral and longitudinal components for the multi-modal LSTM models at 8 s.

Table 6. Lateral / longitudinal error decomposition (K=6, 8 s, seed 42).

Component	Baseline (m)	Lane-Cond. (m)	Improvement
Avg. Longitudinal	1.238	0.924	+25.4%
Avg. Lateral	0.919	0.675	+26.5%
Endpoint Longitudinal	3.561	2.577	+27.6%
Endpoint Lateral	2.687	1.867	+**30.5**%

Both components improve substantially, with the strongest improvement in endpoint lateral error (+30.5%). This is particularly important for intersection safety: lateral errors correspond to lane departures and potential conflicts with adjacent traffic. The balanced improvement across both axes indicates that lane conditioning provides a comprehensive geometric prior.

Figure 5 provides a visual complement, showing the absolute longitudinal and lateral errors at 2 s, 4 s, 6 s, and 8 s horizons for both models.

5.6. Qualitative Analysis

Figure 6 presents a representative side-by-side comparison of trajectory predictions from the baseline and lane-conditioned LSTM models (K=6, 8 s prediction) at an intersection with a curved lane. In the baseline panel (left), predicted modes diverge in multiple directions without regard for

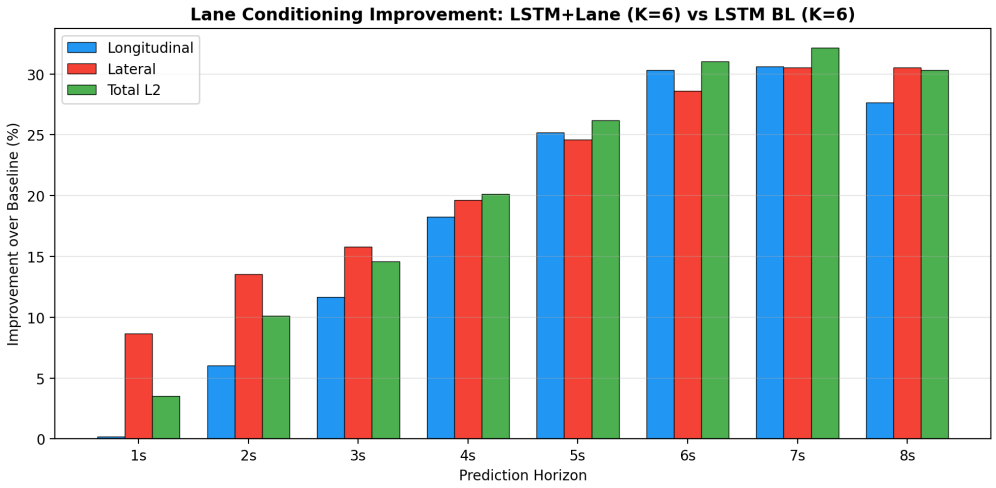

Figure 4. Percentage improvement from lane conditioning decomposed by prediction horizon and error component (K=6). Longitudinal (blue), lateral (red), and total L2 (green) improvements all grow monotonically from 1 s to 7 s, with lateral improvement slightly dominant at shorter horizons.

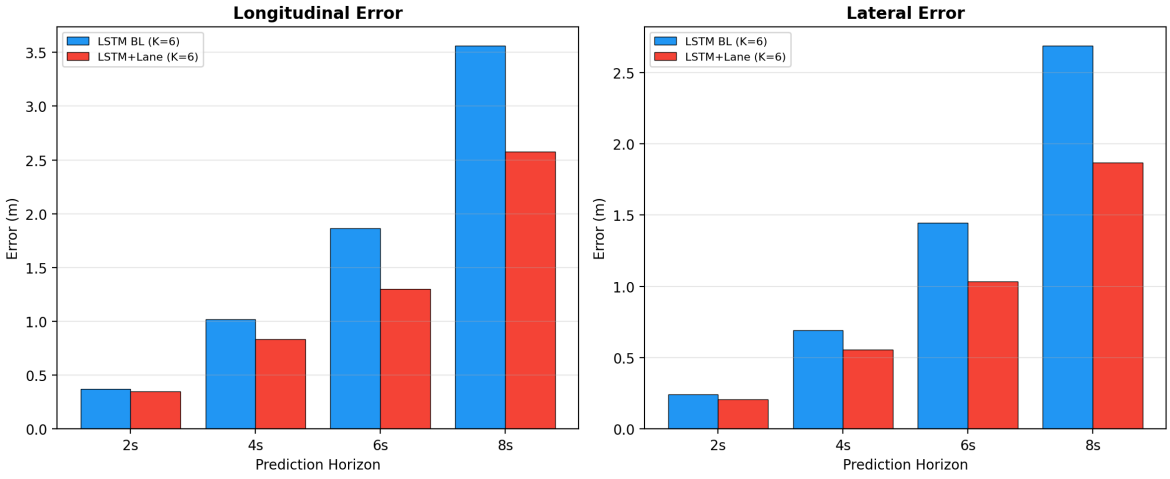

Figure 5. Absolute longitudinal (left) and lateral (right) errors at multiple horizons. The lane-conditioned model (red) consistently reduces both error components, with the gap growing at longer horizons.

road structure, resulting in a minADE of 5.68 m and minFDE of 12.79 m. In the lane-conditioned panel (right), the same scene yields tightly clustered predictions that closely follow the curved lane structure, achieving a minADE of 1.77 m and minFDE of 3.34 m—a 68.9% ADE improvement. The waterflow lane graph is visible in both panels: the ego lane (blue), successor lanes (green), and adjacent lanes (amber). The lane-conditioned model's predictions align closely with these structural cues, demonstrating how lane features constrain predictions to geometrically plausible trajectories even at complex intersections. While this is a single representative example, the quantitative improvements across 13,389 validation scenarios (Tables 2–5) confirm the generality of this effect.

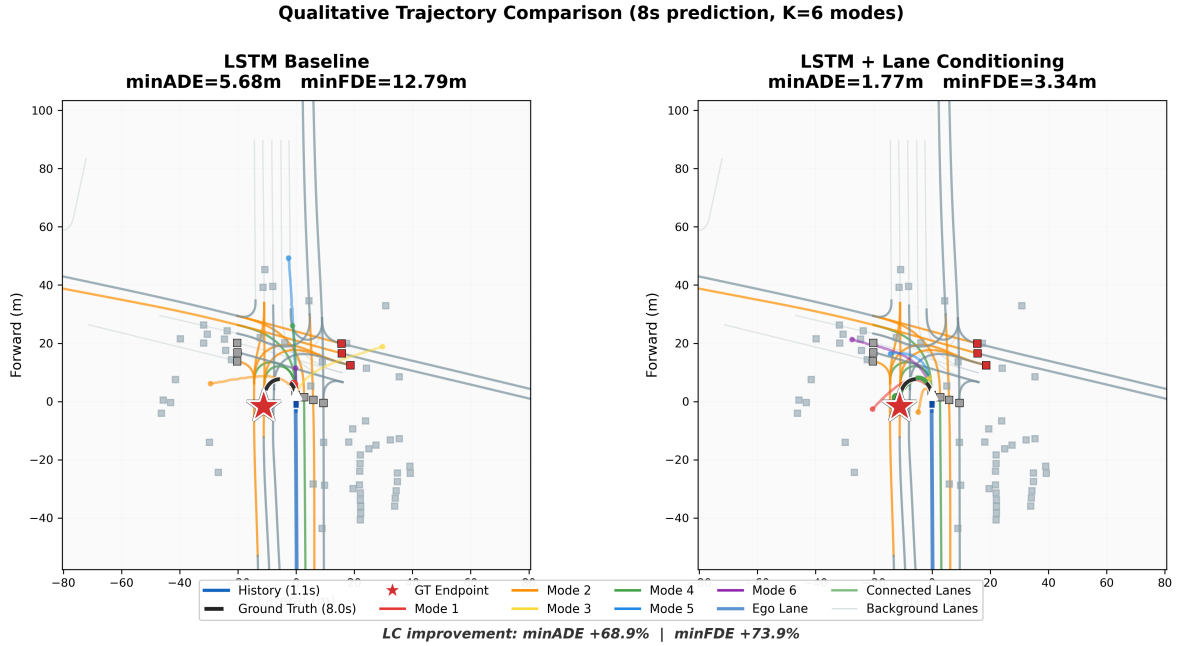

Figure 6. Qualitative trajectory comparison: LSTM Baseline (left) vs. LSTM + Lane Conditioning (right) at an intersection (K=6 modes, 8 s prediction). Colored lines show predicted modes; black dashed line shows ground truth; red star marks the ground-truth endpoint. The lane-conditioned model produces tightly clustered predictions along the curved lane, while baseline modes scatter widely across the scene.

5.7. Comparison with Waymo Baselines

Table 7 contextualizes our results within published Waymo benchmarks. An important caveat applies: our model performs *ego-vehicle self-prediction* (one prediction per scene for the SDC, which is always a vehicle), whereas the Waymo Motion Prediction Challenge evaluates *marginal prediction* for up to 8 diverse agents per scene across three types (vehicle, pedestrian, cyclist). The Waymo vehicle-class metric averages over all tracked vehicles, including parked, turning, and lane-changing agents, which exhibit more diverse behaviors than the ego vehicle. This task difference means our numbers are not directly comparable; the comparison is presented for reference only.

Despite the task difference, the comparison provides a useful reference point. Our lane-conditioned model (1.37 ± 0.08 m across three seeds) achieves a minADE numerically comparable to the Waymo official LSTM using the *full feature set* (1.34 m) (agent state + road graph + traffic signals + high-order interactions), while using only 2D position inputs plus local lane features. This suggests that **local lane structure provides substantial predictive information comparable to hand-engineered kinematic features**, at least for ego-vehicle prediction. We additionally note that the Waymo baselines are evaluated on the official validation set, while our results use a custom 85/15 split of 89,258 signal-controlled scenarios from the training partition; this further limits direct numerical comparison. Extending the model to multi-agent marginal prediction and evaluating on the official validation set

Table 7. Reference comparison with published Waymo benchmarks at 8 s (vehicle minADE, meters). Task differences are discussed in the text.

Method	minADE@8 s	Input Features	Pred. Scope
Waymo LSTM†	2.63	agent state (pos, vel, bbox)	multi-agent
Waymo LSTM + rg + ts + hi†	1.34	agent state + map + signals + interactions	multi-agent
Our LSTM Baseline (K=6)	1.87 ± 0.04	position + neighbors	ego only
Our LSTM-LC (K=6)	$\mathbf{1.37 \pm 0.08}$	**position + lane graph**	ego only

†Waymo baselines from [17], Table 2 (vehicle class, standard val set).
rg = road graph, ts = traffic signals, hi = high-order interactions.

would enable rigorous benchmarking. Integrating lane conditioning into state-of-the-art architectures such as MTR [15] or MTR++ [43], which achieve substantially lower errors through rich features, dense map encodings, and iterative refinement, is a promising direction for future work.

6. Discussion

6.1. Why Lane Conditioning Helps More at Longer Horizons

The improvement increases from +9.3% at 3 s to +26.7% at 8 s. This can be explained by the interplay of two information sources:

1. **Kinematic signal**: The vehicle's history provides strong short-term predictions through velocity and acceleration. This signal decays rapidly—by 8 s, it is essentially uninformative about lane occupancy.
2. **Structural signal**: Lane geometry constrains physically plausible future positions. This constraint is invariant to prediction horizon.

As kinematic information degrades, lane structure becomes the dominant useful signal, explaining the increasing benefit at longer horizons.

6.2. Architecture-Agnostic Nature

The consistent improvement across LSTM (+18.7%) and Transformer (+32.0%) provides strong evidence that lane conditioning is not architecture-specific. The LSTM uses single-vector cross-attention, while the Transformer uses full-sequence multi-head cross-attention. Both yield improvements exceeding 18%, suggesting that even simple lane fusion mechanisms are effective.

The Transformer shows a larger relative improvement (+32% vs. +19%), possibly because full-sequence cross-attention enables richer lane-trajectory correspondences—associating early positions with upstream lanes and later positions with downstream lanes. Additionally, as discussed in Section 6.6, lane conditioning provides an implicit regularization effect that is particularly pronounced for the Transformer, further contributing to the larger improvement.

6.3. On the Gap to State-of-the-Art Methods

We emphasize that **the goal of this study is not to achieve state-of-the-art absolute performance, but to validate a general design principle**. Specifically, we demonstrate that local lane graph conditioning provides a consistent 19–32% improvement *regardless of the backbone architecture*, validated with statistical significance across multiple random seeds ($p < 0.01$). This finding has direct practical implications: any existing or future trajectory prediction system can potentially benefit from adding a lightweight lane conditioning module.

To contextualize the gap to state-of-the-art: methods such as MTR [15] and MTR++ [43] employ (i) rich input features (velocity, acceleration, heading, bounding box, traffic light states), (ii) dense global map encodings with hundreds of polylines, (iii) multi-scale attention with millions of parameters,

and (iv) iterative refinement decoding. Our models use only 2D position with fewer than 700,000 parameters. The fact that our lane-conditioned model achieves a comparable minADE (1.37 ± 0.08 m) to the Waymo official LSTM baseline [17] (1.34 m) using the *full feature set*—despite using only position inputs plus local lane features—suggests that local lane structure provides substantial predictive information for ego-vehicle prediction. While the task difference (ego-only vs. multi-agent, see Section 5.7) prevents claiming equivalence, this result supports local lane conditioning as a promising design principle that generalizes across architectures, horizons, and deployment constraints.

6.4. Implications for Safety and Sustainable Urban Mobility

The World Health Organization reports that road traffic injuries remain a leading cause of death globally, with intersection-related crashes constituting a disproportionate share [44]. The miss rate reduction from 33.9% to 19.4% at the 5 m threshold is particularly significant for intersection safety. A 33.9% miss rate means one in three trajectory predictions ends more than 5 m (approximately two lane widths) from the true endpoint—a potentially dangerous error for planning and collision avoidance systems. Reducing this to 19.4% substantially improves the reliability of downstream safety applications, directly contributing to UN Sustainable Development Goal 3 (Good Health and Well-Being) through reduced road fatalities, and to the goal of reducing the 40% of crashes that occur at intersections [1].

From a sustainability perspective, improved trajectory prediction is expected to have cascading benefits for urban mobility, supporting UN SDG 11 (Sustainable Cities and Communities). Taiebat et al. [3] identify prediction-enabled cooperative driving as one of the key mechanisms through which connected and automated vehicles can reduce energy consumption and emissions. Barth and Boriboonsomsin [2] estimate that congestion-related stop-and-go driving can increase CO_2 emissions by up to 40% compared to free-flow conditions. While we do not directly measure downstream planning or emission outcomes, a 42.7% reduction in miss rate at the 5 m threshold is expected to reduce false-positive emergency braking events and improve the reliability of cooperative maneuver planning, indirectly contributing to smoother traffic flow and lower fuel consumption. These benefits remain indirect: a complete sustainability assessment would require integration with downstream planning modules and real-world or simulation-based driving experiments, which we leave for future work.

Moreover, the lightweight nature of our lane conditioning module ($<$700,000 total parameters) is amenable to deployment on resource-constrained edge devices rather than power-hungry cloud infrastructure. For fleet-scale deployment of autonomous vehicles, the per-vehicle computational cost becomes a significant factor in the total environmental footprint. A model that achieves competitive accuracy with orders-of-magnitude fewer parameters than SOTA methods (which employ millions of parameters) aligns with UN SDG 9 (Industry, Innovation and Infrastructure) by facilitating affordable, energy-efficient prediction systems suitable for urban environments.

6.5. Computational Efficiency and Green AI

Beyond prediction accuracy, the computational footprint of trajectory prediction models has direct sustainability implications aligned with the principles of Green AI [45]. State-of-the-art methods such as MTR++ [43] employ architectures with $\sim$15 million parameters, requiring substantial computational resources for both training and inference. For electric autonomous vehicles (EVs), the energy consumed by onboard computation directly reduces driving range—a critical concern for sustainable deployment at scale.

Our approach offers a favorable accuracy-efficiency trade-off. The lane conditioning module adds fewer than 50,000 parameters (approximately 8% overhead for LSTM) while providing 19–32% accuracy improvement. The total model size ($<$700,000 parameters, approximately 20$\times$ smaller than SOTA) enables real-time inference on resource-constrained edge devices without dedicated GPU hardware. In our experiments, all models were trained on a single consumer-grade GPU (NVIDIA

RTX 4090), with each 100-epoch training run completing in approximately 8–12 GPU-hours—a fraction of the computational budget required for SOTA systems that train on multi-GPU clusters. This makes the approach particularly suitable for large-scale fleet deployment, where per-vehicle computational cost is a significant factor in the total environmental footprint of autonomous driving operations.

6.6. Training Dynamics and Implicit Regularization

An important practical finding is that lane-conditioned models require longer training. Baseline models converge within 30–50 epochs, while lane-conditioned models continue improving until 80–100 epochs. Short training runs can be misleading, showing no benefit or even degradation from lane conditioning.

This effect is most striking for the Transformer architecture. The Transformer baseline reaches its best validation ADE at epoch 29 (4.859 m) but then *overfits progressively*, degrading to 6.154 m by epoch 99—a 26.6% increase in error. In contrast, the lane-conditioned Transformer improves monotonically throughout training, reaching its best ADE at epoch 94 (3.282 m) with no sign of overfitting. This suggests that lane conditioning acts as an *implicit regularizer* [46]: the structural prior from lane features constrains the model's hypothesis space—an instance of the relational inductive biases described by Battaglia et al. [9]—preventing the Transformer's flexible attention mechanism from memorizing training-set artifacts. This regularization benefit is practically significant—it reduces the need for extensive hyperparameter tuning (e.g., dropout, weight decay) and makes training more robust, which in turn reduces wasted computation from failed experiments, aligning with Green AI principles [45].

6.7. Limitations

This study has several limitations. First, while both 3-second and 8-second LSTM results are validated across three seeds ($p < 0.01$), the Transformer experiments at 8 s use a single seed due to computational constraints; additional seeds would strengthen the cross-architecture conclusions. Second, our models use only 2D position inputs; incorporating velocity, heading, and bounding box features would likely improve absolute performance, though this would complicate the isolation of the lane conditioning effect. Third, the waterflow lane graph is limited to 16 lanes within a 3-hop neighborhood, which may be insufficient for very large or complex intersection topologies. Fourth, we evaluate only ego-vehicle (SDC) self-prediction—a single vehicle-type agent per scene—rather than the multi-agent marginal prediction over vehicles, pedestrians, and cyclists used in the Waymo Motion Prediction Challenge; extending to the full multi-agent setting would provide a more complete and directly comparable assessment. Finally, while our lane conditioning module achieves substantial relative improvements, the remaining gap to state-of-the-art methods indicates that lane conditioning alone does not substitute for the full suite of innovations (rich input features, dense global map encoding, iterative refinement) employed in methods like MTR++.

7. Conclusions

This paper presented a controlled study of local lane graph conditioning across two encoder architectures, two prediction horizons, and both single-modal and multi-modal settings on the Waymo Open Motion Dataset. The key findings are:

1. **Consistent, architecture-agnostic improvement.** Lane conditioning improves both LSTM (+9.3% to +26.7%, $p < 0.01$ across three seeds at both horizons) and Transformer (+32.0%, seed 42) backbones.
2. **Horizon-dependent benefit.** Improvement increases from +9.3% at 3 s to +26.7% at 8 s, confirming that lane structure becomes more valuable as kinematic extrapolation degrades.
3. **Balanced error reduction.** Both lateral (+26.5%) and longitudinal (+25.4%) errors improve, with endpoint lateral error showing the strongest reduction (+30.5%).

4. **Feature substitution.** Our lane-conditioned LSTM with K=6 modes achieves mean minADE = 1.37 ± 0.08 m for ego-vehicle prediction, numerically comparable to the Waymo official LSTM (1.34 m, vehicle class) using the full feature set—though the task scope differs (ego-only vs. multi-agent).

These results establish local lane graph conditioning as a lightweight, general-purpose module for trajectory prediction. Future work will explore: (1) integration of the lane conditioning module into state-of-the-art architectures such as MTR to assess whether improvements transfer at higher performance levels; (2) joint multi-agent prediction leveraging shared lane graph representations; (3) cross-dataset generalization to Argoverse [47] and nuScenes [48]; and (4) quantifying the downstream impact of improved prediction accuracy on autonomous vehicle energy efficiency and safety outcomes.

Author Contributions: Conceptualization, X.Z. and C.A.; methodology, X.Z.; software, X.Z.; validation, X.Z.; formal analysis, X.Z.; investigation, X.Z.; resources, C.A.; data curation, X.Z.; writing—original draft preparation, X.Z.; writing—review and editing, X.Z. and C.A.; visualization, X.Z.; supervision, C.A.; project administration, C.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The trajectory prediction models and training code developed in this study are publicly available at https://github.com/Jynxzzz/scenario-dreamer-jynxzzz. The Waymo Open Motion Dataset used for training and evaluation is publicly available at https://waymo.com/open/data/motion/ under the Waymo Dataset License Agreement.

Informed Consent Statement: Not applicable.

Acknowledgments: The authors acknowledge the use of the Waymo Open Motion Dataset for the experiments presented in this work. Computational resources were provided by Concordia University.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

ADE	Average Displacement Error
BEV	Bird's-Eye View
BFS	Breadth-First Search
CV	Constant Velocity
EV	Electric Vehicle
FDE	Final Displacement Error
HD	High Definition
LSTM	Long Short-Term Memory
MLP	Multi-Layer Perceptron
MR	Miss Rate
SDC	Self-Driving Car
SDG	Sustainable Development Goal
TF	Transformer
WOMD	Waymo Open Motion Dataset
WTA	Winner-Takes-All

References

1. Choi, E.H. Crash Factors in Intersection-Related Crashes: An On-Scene Perspective. Technical Report DOT HS 811 366, National Highway Traffic Safety Administration (NHTSA), 2010.
2. Barth, M.; Boriboonsomsin, K. Traffic Congestion and Greenhouse Gases. *ACCESS Magazine* **2009**, *1*, 2–9.
3. Taiebat, M.; Brown, A.L.; Safford, H.R.; Qu, S.; Xu, M. A Review on Energy, Environmental, and Sustainability Implications of Connected and Automated Vehicles. *Environmental Science & Technology* **2018**, *52*, 11449–11465.

4. Contreras-Castillo, J.; Zeadally, S.; Guerrero-Ibáñez, J.A. Internet of Vehicles: Architecture, Protocols, and Security. *IEEE Internet of Things Journal* **2018**, *5*, 3701–3709.
5. Guerrero-Ibáñez, J.; Zeadally, S.; Contreras-Castillo, J. Sensor Technologies for Intelligent Transportation Systems. *Sensors* **2018**, *18*, 1212.
6. Huang, Y.; Du, J.; Yang, Z.; Zhou, Z.; Zhang, L.; Chen, H. A Survey on Trajectory-Prediction Methods for Autonomous Driving. *IEEE Transactions on Intelligent Vehicles* **2022**, *7*, 652–674.
7. Mozaffari, S.; Al-Jarrah, O.Y.; Dianati, M.; Jennings, P.; Mouzakitis, A. Deep Learning-Based Vehicle Behaviour Prediction for Autonomous Driving. *IEEE Transactions on Intelligent Transportation Systems* **2020**, *23*, 33–47.
8. Wang, C.; Xu, C.; Xia, J.; Qian, Z.; Lu, L. A Review of Surrogate Safety Measures and Their Applications in Connected and Automated Vehicles Safety Modeling. *Accident Analysis & Prevention* **2021**, *157*, 106157.
9. Battaglia, P.W.; Hamrick, J.B.; Bapst, V.; Sanchez-Gonzalez, A.; Zambaldi, V.; Malinowski, M.; Tacchetti, A.; Raposo, D.; Santoro, A.; Faulkner, R.; others. Relational Inductive Biases, Deep Learning, and Graph Networks. *arXiv preprint arXiv:1806.01261* **2018**.
10. Gao, J.; Sun, C.; Zhao, H.; Shen, Y.; Anguelov, D.; Li, C.; Schmid, C. VectorNet: Encoding HD Maps and Agent Dynamics from Vectorized Representation. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 11525–11533.
11. Liang, M.; Yang, B.; Hu, R.; Chen, Y.; Liao, R.; Feng, S.; Urtasun, R. Learning Lane Graph Representations for Motion Forecasting. European Conference on Computer Vision (ECCV). Springer, 2020, pp. 541–556.
12. Zhao, H.; Gao, J.; Lan, T.; Sun, C.; Sapp, B.; Varadarajan, B.; Shen, Y.; Shen, Y.; Chai, Y.; Schmid, C.; others. TNT: Target-driven Trajectory Prediction. Conference on Robot Learning (CoRL). PMLR, 2021, pp. 895–904.
13. Zhou, Z.; Ye, L.; Wang, J.; Wu, K.; Lu, K. HiVT: Hierarchical Vector Transformer for Multi-Agent Motion Prediction. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022, pp. 8823–8833.
14. Deo, N.; Wolff, E.; Beijbom, O. Multimodal Trajectory Prediction Conditioned on Lane-Graph Traversals. Conference on Robot Learning (CoRL). PMLR, 2022, pp. 203–212.
15. Shi, S.; Jiang, L.; Dai, D.; Schiele, B. Motion Transformer with Global Intention Localization and Local Movement Refinement. Advances in Neural Information Processing Systems (NeurIPS), 2022, Vol. 35, pp. 6531–6543.
16. Zhou, Z.; Wang, J.; Li, Y.H.; Huang, Y.K. Query-Centric Trajectory Prediction. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2023, pp. 6184–6193.
17. Ettinger, S.; Cheng, S.; Caine, B.; Liu, C.; Zhao, H.; Pradhan, S.; Chai, Y.; Sapp, B.; Qi, C.R.; Zhou, Y.; Yang, Z.; Chou, A.; Sun, P.; Ngiam, J.; Vasudevan, V.; McCauley, A.; Shlens, J.; Anguelov, D. Large Scale Interactive Motion Forecasting for Autonomous Driving: The Waymo Open Motion Dataset. Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2021, pp. 9710–9719.
18. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Computation* **1997**, *9*, 1735–1780.
19. Alahi, A.; Goel, K.; Raber, V.; Sadeghian, A.; Fei-Fei, L.; Savarese, S. Social LSTM: Human Trajectory Prediction in Crowded Spaces. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 961–971.
20. Park, S.H.; Kim, B.; Kang, C.M.; Chung, C.C.; Choi, J.W. Sequence-to-Sequence Prediction of Vehicle Trajectory via LSTM Encoder-Decoder Architecture. 2018 IEEE Intelligent Vehicles Symposium (IV). IEEE, 2018, pp. 1672–1678.
21. Deo, N.; Trivedi, M.M. Convolutional Social Pooling for Vehicle Trajectory Prediction. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2018, pp. 1468–1476.
22. Chandra, R.; Guan, T.; Panber, S.; Manocha, D. Forecasting Trajectory and Behavior of Road-Agents Using Spectral Clustering in Graph-LSTMs. *IEEE Robotics and Automation Letters* **2020**, *5*, 4882–4890.
23. Li, J.; Yang, F.; Tomizuka, M.; Choi, C. EvolveGraph: Multi-Agent Trajectory Prediction with Dynamic Relational Reasoning. Advances in Neural Information Processing Systems (NeurIPS), 2020, Vol. 33, pp. 19783–19794.
24. Mo, X.; Huang, Z.; Xing, Y.; Lv, C. Graph and Recurrent Neural Network-Based Vehicle Trajectory Prediction for Highway Driving. *IEEE Transactions on Intelligent Transportation Systems* **2022**, *23*, 17534–17547.

25. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention Is All You Need. Advances in Neural Information Processing Systems (NeurIPS), 2017, Vol. 30.
26. Ngiam, J.; Caine, B.; Vasudevan, V.; Zhang, Z.; Chiu, H.L.; Pierce, A.; Truong, Y.; Dao, T.D.; Sapp, B.; Qi, C.; others. Scene Transformer: A Unified Architecture for Predicting Multiple Agent Trajectories. International Conference on Learning Representations (ICLR), 2022.
27. Nayakanti, N.; Al-Rfou, R.; Zhou, A.; Goel, K.; Refaat, K.S.; Sapp, B. Wayformer: Motion Forecasting via Simple & Efficient Attention Networks. 2023 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2023, pp. 2187–2193.
28. Zeng, A.; Chen, M.; Zhang, L.; Xu, Q. Are Transformers Effective for Time Series Forecasting? Proceedings of the AAAI Conference on Artificial Intelligence, 2023, Vol. 37, pp. 11121–11128.
29. Kipf, T.N.; Welling, M. Semi-Supervised Classification with Graph Convolutional Networks. International Conference on Learning Representations (ICLR), 2017.
30. Zeng, W.; Liang, M.; Liao, R.; Urtasun, R. LaneRCNN: Distributed Representations for Graph-Centric Motion Forecasting. Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2021, pp. 532–539.
31. Gu, J.; Sun, C.; Zhao, H. DenseTNT: End-to-End Trajectory Prediction from Dense Goal Sets. Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2021, pp. 15303–15312.
32. Kim, B.; Park, S.H.; Lee, S.; Khoshimjonov, E.; Kum, D.; Kim, J.; Kim, J.S.; Choi, J.W. LaPred: Lane-Aware Prediction of Multi-Modal Future Trajectories of Dynamic Agents. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021, pp. 14636–14645.
33. Wang, M.; Zhu, X.; Yu, C.; Li, W.; Ma, Y.; Jin, R.; Ren, X.; Li, D.; Yin, M.; Wang, W. GANet: Goal Area Network for Motion Forecasting. Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2023, pp. 10784–10790.
34. Gilmer, J.; Schoenholz, S.S.; Riley, P.F.; Vinyals, O.; Dahl, G.E. Neural Message Passing for Quantum Chemistry. International Conference on Machine Learning (ICML). PMLR, 2017, pp. 1263–1272.
35. Bahdanau, D.; Cho, K.; Bengio, Y. Neural Machine Translation by Jointly Learning to Align and Translate. International Conference on Learning Representations (ICLR), 2015.
36. Gupta, A.; Johnson, J.; Fei-Fei, L.; Savarese, S.; Alahi, A. Social GAN: Socially Acceptable Trajectories with Generative Adversarial Networks. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 2255–2264.
37. Lee, S.; Purushwalkam, S.; Cogswell, M.; Crandall, D.; Batra, D. Stochastic Multiple Choice Learning for Training Diverse Deep Ensembles. Advances in Neural Information Processing Systems (NeurIPS), 2016, Vol. 29.
38. Salzmann, T.; Ivanovic, B.; Chakravarty, P.; Pavone, M. Trajectron++: Dynamically-Feasible Trajectory Forecasting with Heterogeneous Data. European Conference on Computer Vision (ECCV). Springer, 2020, pp. 683–700.
39. Schöller, C.; Aravantinos, V.; Lay, F.; Knoll, A. What the Constant Velocity Model Can Teach Us About Pedestrian Motion Prediction. *IEEE Robotics and Automation Letters* **2020**, *5*, 1696–1703.
40. Huber, P.J. Robust Estimation of a Location Parameter. *The Annals of Mathematical Statistics* **1964**, *35*, 73–101.
41. Loshchilov, I.; Hutter, F. Decoupled Weight Decay Regularization. International Conference on Learning Representations (ICLR), 2019.
42. Loshchilov, I.; Hutter, F. SGDR: Stochastic Gradient Descent with Warm Restarts. International Conference on Learning Representations (ICLR), 2017.
43. Shi, S.; Jiang, L.; Dai, D.; Schiele, B. MTR++: Multi-Agent Motion Prediction with Symmetric Scene Modeling and Pair-Wise Context. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **2024**, *46*, 3543–3558.
44. World Health Organization. Global Status Report on Road Safety 2023. Technical report, World Health Organization, Geneva, 2023.
45. Schwartz, R.; Dodge, J.; Smith, N.A.; Etzioni, O. Green AI. *Communications of the ACM* **2020**, *63*, 54–63.
46. Neyshabur, B.; Tomioka, R.; Srebro, N. In Search of the Real Inductive Bias: On the Role of Implicit Regularization in Deep Learning. ICLR 2015 Workshop Track, 2015.

639 47. Chang, M.F.; Lambert, J.; Sangkloy, P.; Singh, J.; Bak, S.; Hartnett, A.; Wang, D.; Carr, P.; Lucey, S.; Ramanan,
640 D.; others. Argoverse: 3D Tracking and Forecasting with Rich Maps. Proceedings of the IEEE/CVF
641 Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 8748–8757.
642 48. Caesar, H.; Bankiti, V.; Lang, A.H.; Vora, S.; Liong, V.E.; Xu, Q.; Krishnan, A.; Pan, Y.; Baldan, G.; Beijbom,
643 O. nuScenes: A Multimodal Dataset for Autonomous Driving. Proceedings of the IEEE/CVF Conference
644 on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 11621–11631.