

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  and Ciprian Alecsandru * 

Department of Building, Civil and Environmental Engineering, Concordia University,
Montreal, QC H3G 1M8, Canada; xingnan.zhou@mail.concordia.ca

* Correspondence: ciprian.alecsandru@concordia.ca

Abstract

Accurate trajectory prediction is critical for autonomous driving safety and energy-efficient motion planning in sustainable urban mobility. This study isolates the effect of local lane graph conditioning by proposing a *waterflow* method that extracts an ego-centric lane topology via breadth-first traversal of the HD map (For convenience, all acronyms used throughout this manuscript (e.g., HD, LSTM, ADE, FDE, minADE, minFDE, BFS, GNN, SOTA) are collected in the Abbreviations section at the end of the paper), fusing lane features into trajectory encoders through cross-attention. The evaluation is deliberately scoped to ego-vehicle prediction at signal-controlled intersections: we evaluate across two architectures (LSTM and Transformer), two horizons (3 s and 8 s), and both single- and multi-modal ($K = 6$) settings on 89,258 such scenarios from the Waymo Open Motion Dataset, so that “generality” refers to consistency across architectures, horizons, and output settings within this scope, rather than across prediction tasks. Lane conditioning consistently improves accuracy: +9.3% ADE at 3 s ($p = 0.007$, 3 seeds), +26.6% minADE at 8 s ($K = 6$, $p = 0.003$, 3 seeds), and +26.8% ADE for the Transformer ($p = 0.030$, 3 seeds)—with only ~8% additional parameters for the LSTM. A controlled full-graph ablation (nearest 64 lanes) shows a consistent but not statistically significant trend favouring topologically guided local selection over brute-force spatial proximity (+11.4% minADE, $p = 0.063$). Error decomposition reveals balanced lateral (+26.5%) and longitudinal (+25.4%) improvements, and a per-maneuver analysis over all 13,388 validation scenarios shows the largest gains for turning maneuvers. The lane-conditioned model (<700,000 parameters) runs in 0.8 ms per prediction on a desktop GPU (0.7 ms single-threaded CPU) with below 30 MB peak memory and an estimated 51 mJ per prediction, suggesting feasibility for resource-constrained deployment, pending validation on production automotive hardware.



Academic Editor: Jinlei Zhang, Wei Ma

Received: 8 July 2026

Revised: 30 July 2026

Accepted: 7 August 2026

Published:

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

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. The distinction from prior map-aware architectures is structural: LaneGCN [11] and its successors use the lane graph as the *primary backbone*—the entire network is designed around lane-graph representations, and adopting them requires committing to that architecture. In contrast, we treat the lane graph as an *auxiliary conditioning signal* delivered by a modular plug-in (<50,000 parameters) that attaches to an existing trajectory encoder through cross-attention, leaving the backbone untouched. This modular, pluggable design is the organizing thread of the paper: because the module is backbone-agnostic, we can evaluate the same lane conditioning mechanism across fundamentally different architectures. 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 ego-vehicle trajectory prediction at signal-controlled intersections?* Throughout this paper, *general* and *architecture-agnostic* refer to the consistency of the improvement across backbone architectures (recurrent and attention-based), prediction horizons, and output settings *within this scope*—not to generalization across prediction tasks, agent types, or roadway environments, which we discuss as limitations in Section 6.

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 examine the importance of *local* lane selection through a controlled ablation against a full-graph baseline (nearest 64 lanes with full adjacency). The waterflow method shows a consistent trend toward lower error than the full graph (+11.4% minADE

at 8 s, $p = 0.063$; +3.8% ADE at 3 s, $p = 0.087$), although these differences do not reach statistical significance at $\alpha = 0.05$; we additionally characterize the connectivity-density differences between the two graph constructions (Section 3.2.3).

3. We demonstrate that lane conditioning provides a **consistent, architecture-agnostic improvement** within the ego-vehicle, signalized-intersection setting studied here, validated with three random seeds in every headline setting: +9.3% ADE reduction for LSTM at 3 s ($p = 0.007$), +26.6% minADE reduction for multi-modal LSTM at 8 s ($p = 0.003$), and +26.8% ADE reduction for Transformer at 8 s ($p = 0.030$).
4. We show that the benefit of lane conditioning **increases with prediction horizon** (from +9.3% at 3 s to +26.6% at 8 s), confirming that lane structure becomes increasingly valuable as kinematic extrapolation degrades over longer horizons.
5. 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%).
6. We report per-maneuver and per-scenario analyses over all 13,388 validation scenarios—including median, typical, and failure cases—together with measured inference latency, memory, and energy figures, so that both the accuracy claims and the efficiency claims are grounded in distributional and measured evidence rather than selected examples.

Our results point toward computationally efficient prediction systems: the full lane-conditioned model (<700,000 parameters) runs in under a millisecond per prediction on desktop hardware (Section 6.6), suggesting that meaningful accuracy gains can be obtained without the computational footprint of state-of-the-art systems—a direction aligned with sustainable autonomous driving, pending validation on production automotive 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. For a broad survey of interaction-aware prediction and decision-making methods for autonomous vehicles in mixed traffic—covering LSTM- and Transformer-based trajectory and intention prediction alongside game-theoretic and reinforcement-learning approaches—we refer readers to the recent review by Yang et al. [22]; the present study can be read as isolating one specific design factor (local lane topology conditioning) within the prediction component of such systems.

Graph-based extensions to recurrent models have been proposed to capture dynamic agent interactions. Chandra et al. [23] employed graph-LSTMs with spectral clustering, while Li et al. [24] introduced EvolveGraph for dynamic relational reasoning. Mo et al. [25] 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 [26] have achieved state-of-the-art performance on large-scale motion forecasting benchmarks. Scene Transformer [27] proposed a unified multi-agent prediction architecture. Wayformer [28] 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. [29] 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 [30] along lane connectivity structures. LaneRCNN [31], from the same group, extended this with distributed lane-centric representations and actor-lane interaction graphs. TNT [12] and DenseTNT [32] leveraged lane centerlines as target candidates. LaPred [33] 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 [34] 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. The essential distinction between this work and LaneGCN-style architectures lies in the *role* assigned to the lane graph. In LaneGCN, the lane graph is the primary backbone: map nodes carry the central representation, actor features are fused into the lane-node space, and the architecture cannot be separated from its map encoder. In our design, the lane graph is an *auxiliary conditioning signal*: a lightweight, self-contained module (lane MLP, message passing [35], and cross-attention pooling [36]) that plugs into an otherwise unchanged trajectory encoder. Table 1 summarizes this comparison. The practical consequence is composability—the same module attaches to a recurrent or an attention-based backbone without architectural surgery—which is precisely what enables the controlled cross-architecture study in this paper. 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.

Table 1. Role of the lane graph in representative map-aware methods vs. this work.

Method	Role of Lane Graph	Backbone Coupling	Lane Component Size
LaneGCN [11]	Primary backbone	Inseparable	Entire model
VectorNet [10]	Unified polyline encoding	Inseparable	Entire model
PGP [14]	Traversal-based goal generation	Drives decoding	Core component
Ours	Auxiliary conditioning signal	Pluggable (cross-attention)	<50 K params (~8%)

An important design question is the *scope* of lane information: should models consume all nearby lanes or a curated local subset? LaneGCN [11] uses a large neighborhood (hundreds of polyline segments), while PGP [14] traverses lane-graph paths as discrete goal candidates. Our work directly addresses this question through a controlled ablation comparing full-graph (nearest 64 lanes) against waterflow-selected local lanes (16 lanes via

3-hop BFS), using the same model architecture and training pipeline to isolate the effect of graph scope.

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. [37] introduced Social GAN, using generative adversarial training to produce diverse trajectory samples. Winner-takes-all (WTA) training [38] provides a simpler alternative, assigning each ground truth to the closest prediction mode and backpropagating through that mode only. Salzman et al. [39] 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 align our evaluation protocol with the minADE/minFDE metrics standard in this literature, while noting that our task scope differs from the official benchmark (Section 5.10).

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 a sampling rate of 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. The 5 m value corresponds to roughly $1.4 \times$ a standard 3.5 m lane width, covering in-lane driving as well as lateral offsets during lane changes. The results are insensitive to this choice: across all 89,258 scenarios, the distance from the ego position to the nearest lane centerline has a median of 0.22 m and a 99th percentile of 1.99 m, so a valid ego lane is identified directly in 99.95% of scenarios at 5 m, and the identification

rate varies by less than one percentage point as the threshold is swept from 2 m to 10 m (99.01–99.99%; Supplementary Table S2). Moreover, because the ego lane is defined as the minimum-distance lane whenever one lies within the threshold—and the fallback likewise selects the nearest lane—the *identity* of the selected lane is invariant to the threshold; the parameter only determines how often the BFS expansion is replaced by the nearest-lane fallback (0.05% of scenarios at 5 m).

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 $\mathbf{m}$

```

1  $\ell_0 \leftarrow \arg \min_{\ell \in \mathcal{M}} \text{dist}(\text{centerline}(\ell), \mathbf{p}_{\text{ego}})$ ; // Ego lane
2 if  $\text{dist}(\text{centerline}(\ell_0), \mathbf{p}_{\text{ego}}) > 5 \text{ m}$  then
3    $\ell_0 \leftarrow$  nearest lane by Euclidean distance; // Fallback
4 end
5  $\mathcal{Q} \leftarrow \{(\ell_0, 0)\}$ ; // Queue: (lane, hop count)
6  $\mathcal{V} \leftarrow \{\ell_0\}$ ; // Visited set
7 while  $\mathcal{Q} \neq \emptyset$  and  $|\mathcal{V}| < L_{\max}$  do
8    $(\ell, h) \leftarrow \mathcal{Q}.\text{dequeue}()$ ;
9   if  $h < h_{\max}$  then
10    foreach  $\ell' \in \text{Succ}(\ell) \cup \text{LeftAdj}(\ell) \cup \text{RightAdj}(\ell)$  do
11      if  $\ell' \notin \mathcal{V}$  and  $|\mathcal{V}| < L_{\max}$  then
12         $\mathcal{V} \leftarrow \mathcal{V} \cup \{\ell'\}$ ;
13         $\mathcal{Q}.\text{enqueue}((\ell', h + 1))$ ;
14      end
15    end
16  end
17 end
18 foreach  $\ell \in \mathcal{V}$  do
19    $\mathbf{f}_\ell \leftarrow [\mathbf{c}_\ell \parallel \mathbf{d}_\ell \parallel s_\ell \parallel \mathbf{b}_\ell]$ ; // Equation (1)
20 end
21 Construct  $\mathbf{A}$  from lane connectivity within  $\mathcal{V}$ ; set  $\mathbf{m}$ ;
22 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.

3.2.2. Lane Feature Representation

Each lane feature vector comprises:

$$\mathbf{f}_\ell = [\mathbf{c}_\ell \parallel \mathbf{d}_\ell \parallel s_\ell \parallel \mathbf{b}_\ell], \quad (1)$$

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.

Waterflow Graph: Progressive Lane Discovery via BFS from Ego Vehicle

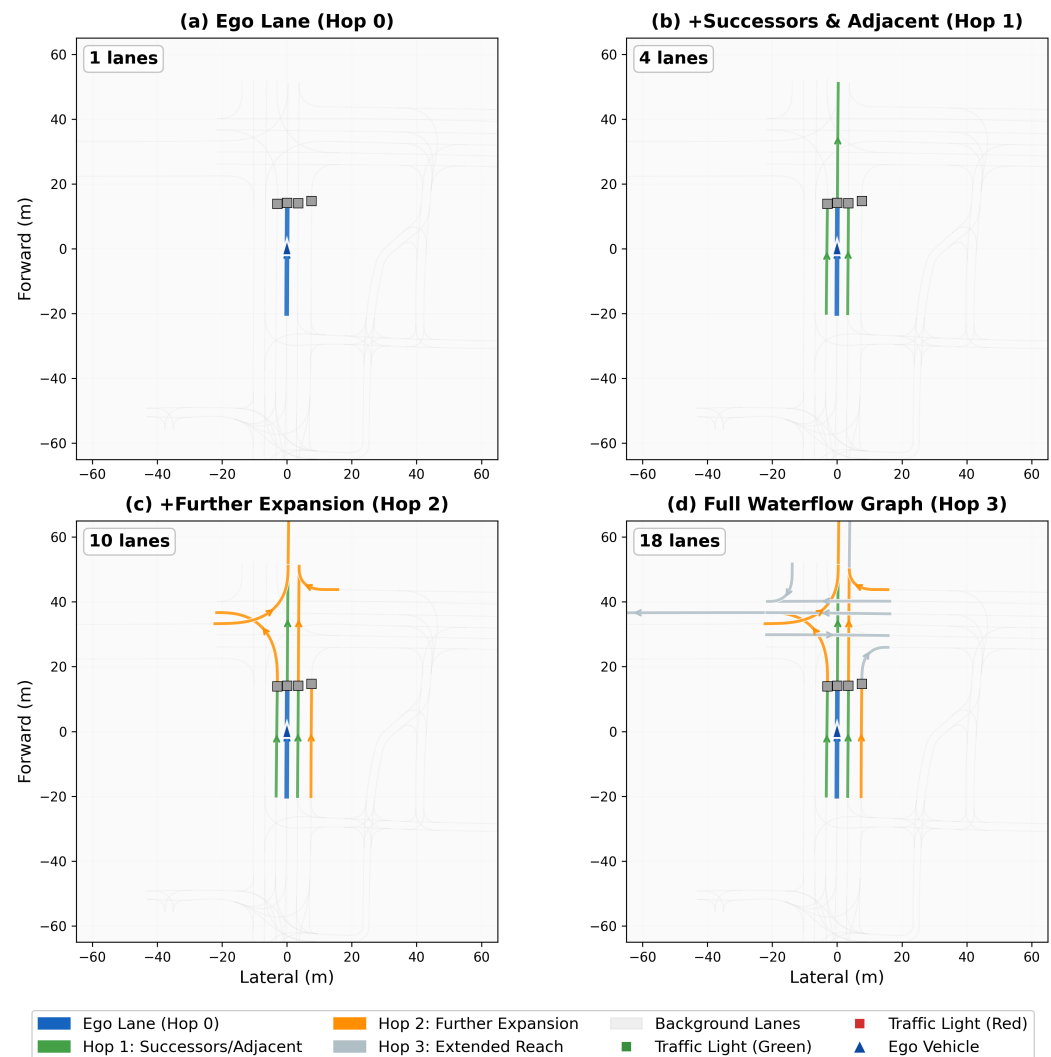


Figure 1. Waterflow lane graph extraction via breadth-first expansion. Starting from the ego lane (Hop 0), the graph progressively incorporates successor and adjacent lanes through 3 hops, capturing the local road topology relevant to trajectory prediction. This intersection example discovers 18 lanes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

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.

3.2.3. Full-Graph Baseline

To evaluate the importance of topological lane selection, we construct a full-graph baseline that bypasses the BFS traversal. Instead of Algorithm 1, all lanes within the scene are sorted by Euclidean distance to the ego position, and the nearest $L_{\max}^{\text{full}} = 64$ lanes are selected. The adjacency matrix is built from *all* successor, left-adjacent, and right-adjacent pairs among the selected lanes, rather than only edges discovered during BFS.

The lane feature representation (Equation (1)), message passing, and cross-attention fusion remain identical.

It is important to note that the two graph constructions differ in *two* respects simultaneously: (i) the lane selection strategy (topological BFS vs. Euclidean nearest- N) and (ii) the adjacency construction (edges discovered during BFS traversal vs. all connectivity pairs among the selected lanes). The resulting graphs consequently differ markedly in connectivity: on average, the waterflow graph contains 14.3 edges over 15.4 lanes (graph density 0.14), whereas the full graph contains 135.4 edges over 63.4 lanes (density 0.07, but $9.5\times$ more edges in absolute terms). A comparison between the two default configurations therefore cannot, by itself, attribute performance differences to lane selection alone. Section 5.4 disentangles these two factors explicitly with a 2×2 decoupling ablation that crosses both selection strategies with both adjacency constructions.

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.



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. **Arrows indicate the direction of data flow between modules.**

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 $d_h = 128$

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

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

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

3.4.3. Fusion and CV-Residual Decoder

The ego hidden state and neighbor context are concatenated and fused

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

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

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

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. [40] 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 [35] and cross-attention pooling [36]. The ego encoder, neighbor encoder, and CV-residual decoder remain 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

$$\mathbf{l}_\ell^{(k+1)} = \text{ReLU}(\mathbf{W}^{(k)} [\mathbf{l}_\ell^{(k)} \parallel (\tilde{\mathbf{A}} \mathbf{L}^{(k)})_\ell]), \quad (6)$$

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

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

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

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

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 [26]. 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.

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

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

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

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

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.

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

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

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

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

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-s 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

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

where SmoothL1 denotes the Huber loss [41]. The superscript asterisk marks ground-truth and winner quantities: k^* is the index of the mode closest to the ground truth, and $\mathbf{y}_t^*$ and $\mathbf{Y}^*$ denote the ground-truth position at timestep t and the full ground-truth trajectory, respectively.

3.9. Model Complexity

Table 2 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 $d_{\text{model}} = 128$. All models remain under 700,000 parameters—orders of magnitude smaller than state-of-the-art methods.

Table 2. 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 (Waymo LLC, Mountain View, CA, USA) [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 1000 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 75,870 training and 13,388 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-s 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-s 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 [42] with gradient clipping at norm 1.0. LSTM models use a 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 [43] over 100 epochs with early stopping (patience 20). The batch size is 128.

For both 3-s and 8-s LSTM experiments, we train with three random seeds (7, 42, 123) and report mean $\pm$ standard deviation with paired *t*-test *p*-values. The Transformer experiments at 8 s were initially run with a single seed due to computational constraints; for this revision, we trained the two additional seeds (7, 123) for both Transformer variants, so that all headline results are now validated across the same three seeds. Only the single-modal LSTM rows at 8 s in the cross-architecture comparison of Section 5.6 remain seed-42-only, as they serve as context for the cross-architecture comparison; the LSTM effect itself is seed-validated at both horizons in its own settings. For the full-graph ablation (Section 5.3), we train three additional seeds with identical hyperparameters (batch size 128, learning rate 10^{-3} , 100 epochs) using $L_{\max} = 64$ lanes. Model selection uses the best validation metric over training, applied uniformly to every run and seed.

All experiments are conducted on a single NVIDIA GeForce RTX 4090 GPU with 24 GB VRAM (NVIDIA Corporation, Santa Clara, CA, USA).

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@*d*m)**: 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-S Prediction: Multi-Seed Validation

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

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

Model	ADE@3 s (m)	Best Seed	<i>p</i> -Value
LSTM Baseline	0.559 ± 0.007	0.552	—
LSTM Lane-Cond.	0.507 ± 0.011	0.496	0.0071
Improvement	+9.3%		

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), and +8.1% (seed 123).

5.2. 8-S Multi-Modal Results

The 8-s 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 4 presents the results.

Table 4. Multi-modal prediction ($K = 6$) at 8 s (3 seeds, 100 epochs). Values are mean $\pm$ std.

Model	minADE (m)	minFDE (m)
LSTM Baseline	1.868 ± 0.042	5.047 ± 0.106
LSTM Lane-Cond.	1.371 ± 0.081	3.403 ± 0.242
Improvement	+26.6% ($p = 0.003$)	+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. Graph Scope Ablation: Full Graph vs. Waterflow

To validate the importance of *local* lane selection, we compare the waterflow method (3-hop BFS, $L_{\max} = 16$) against a full-graph baseline (nearest 64 lanes, full adjacency) described in Section 3.2.3. Both use the same LaneEncoder architecture and training hyperparameters; only the graph construction differs. Table 5 presents the results.

Table 5. Graph scope ablation: Baseline vs. Full Graph (nearest 64 lanes) vs. Waterflow (3-hop BFS, 16 lanes). All models use identical architecture and training; only graph construction differs. Values are mean $\pm$ std over 3 seeds.

Graph Scope	8 s Multi-Modal ($K = 6$)		3 s Single-Modal	
	minADE (m)	minFDE (m)	ADE (m)	FDE (m)
No Lane (Baseline)	1.868 ± 0.042	5.047 ± 0.106	0.559 ± 0.007	1.582 ± 0.015
Full Graph (64 lanes)	1.547 ± 0.055	4.041 ± 0.180	0.527 ± 0.006	1.451 ± 0.021
Waterflow (16 lanes)	1.371 ± 0.081	3.403 ± 0.242	0.507 ± 0.011	1.383 ± 0.031
Full Graph vs. BL	+17.2% ($p = 0.003$)	+19.9%	+5.7% ($p = 0.007$)	+8.3%
Waterflow vs. BL	+26.6% ($p = 0.002$)	+32.6%	+9.3% ($p = 0.005$)	+12.6%
Waterflow vs. Full	+11.4% ($p = 0.063$)	+15.8%	+3.8% ($p = 0.087$)	+4.7%

At the 8-s horizon, the full-graph baseline substantially outperforms the no-lane baseline (+17.2% minADE, $p = 0.003$), confirming that lane information is valuable regardless of selection strategy. The waterflow method shows an additional +11.4% mean improvement over the full graph despite using $4\times$ fewer lanes; however, this difference does not reach statistical significance across three seeds ($p = 0.063$). The same pattern appears at 3 s (Full Graph +5.7% vs. Waterflow +9.3%; difference $p = 0.087$, not significant).

These results support one firm conclusion and one trend: (1) *any* lane information helps—both lane-conditioned variants significantly outperform the no-lane baseline ($p < 0.01$); and (2) there is a consistent but not statistically significant trend favouring topologically guided local selection over brute-force spatial proximity, observed at both horizons and in both ADE and FDE. Possible mechanisms behind this trend—signal dilu-

tion in message passing, attention distraction, and topological relevance—are discussed as hypotheses in Section 6.3. Moreover, as noted in Section 3.2.3, the two default configurations differ in both lane selection and adjacency construction; the next subsection disentangles these two factors.

5.4. Disentangling Lane Selection from Graph Connectivity

As acknowledged in Section 3.2.3, the waterflow and full-graph configurations differ simultaneously in lane selection (BFS vs. nearest- N) and adjacency construction (BFS-traversal edges vs. all connectivity pairs), so the comparison in Section 5.3 cannot by itself attribute the observed difference to either factor. To disentangle them, we train two additional configurations that cross the two factors: (i) *BFS-16 + all pairs*, which keeps the waterflow lane selection but builds the adjacency from all successor/left/right pairs among the selected lanes (raising the average edge count from 14.3 to 29.3); and (ii) *nearest-64 + BFS edges*, which keeps the full-graph selection but retains only edges discovered by a BFS traversal from the ego lane restricted to the selected set (reducing the average edge count from 135.4 to 42.8). Together with the two default configurations, this forms a complete 2×2 design over {selection} $\times$ {adjacency}, trained with identical hyperparameters (seed 42, $K = 6, 8$ s). Table 6 presents the results.

Table 6. Decoupling lane selection from adjacency construction: 2×2 ablation (seed 42, $K = 6, 8$ s; best validation epoch). “Edges” is the average number of adjacency edges per scenario (Section 3.2.3). The no-lane baseline achieves minADE 1.839 (seed 42). Diagonal cells are the two default configurations from Section 5.3.

Lane Selection	BFS-Traversal Edges		All Connectivity Pairs	
	minADE (m)	Edges	minADE (m)	Edges
BFS-16 (waterflow)	1.337	14.3	1.372	29.3
Nearest-64 (full graph)	1.485	42.8	1.608	135.4

Three findings emerge. First, **lane selection is the dominant factor**: switching from nearest-64 to BFS-16 selection reduces minADE by 0.236 m under all-pairs adjacency and by 0.148 m under BFS-edge adjacency (a mean effect of 0.192 m, accounting for roughly 70% of the 0.271 m gap between the two default configurations under a Shapley-style attribution over the two change orders). Second, **adjacency construction contributes a secondary effect** (mean 0.079 m, roughly 30%) that is clearly present on the large graph but not resolvable on the small one: sparsifying the adjacency to BFS-traversal edges helps substantially on the 64-lane graph ($1.608 \rightarrow 1.485$, -0.123 m, exceeding the typical across-seed variability of ~ 0.08 m for this model class) but only marginally on the 16-lane graph ($1.372 \rightarrow 1.337$, -0.035 m, within seed-level noise). This asymmetry is consistent with the signal-dilution hypothesis of Section 6.3: dense connectivity is most harmful when many selected lanes are irrelevant, as message passing then mixes ego-relevant features with distant-lane noise. Third, *every* cell outperforms the no-lane baseline (1.839), reaffirming that lane information helps under all four constructions. In summary, the reviewer-identified confound is real—connectivity density does contribute—but the majority of the waterflow advantage is attributable to topological lane selection rather than to the sparser adjacency, supporting the paper’s central interpretation. We note that this 2×2 analysis uses a single seed (42), trading seed replication for design coverage within our compute budget; multi-seed replication of the decoupling is listed as future work.

Finally, we comment on *geometric* alternatives to BFS-based selection, such as radius-based or direction-aware filtering. The property that makes topological traversal effective is that lane-graph connectivity encodes *drivability*: a BFS from the ego lane naturally excludes

lanes that are physically near but unreachable in the driving sense—opposing-direction lanes, lanes across a physical divider, or lanes on a grade-separated crossing—which no distance radius can express. A direction-aware filter (discarding lanes whose heading opposes the ego’s) approximates only one component of this property. Because the nearest- N baseline in our ablation is precisely the limiting case of radius-based selection, and because every topologically selected configuration outperformed it (Table 7), we expect geometric filters to fall between these two families; training them explicitly is left to future work.

Table 7. Sensitivity of the multi-modal lane-conditioned LSTM to the waterflow hyperparameters (seed 42, $K = 6, 8$ s; best validation epoch). Graph statistics averaged over the same 300-scenario sample as Section 3.2.3. Reference points: no-lane baseline 1.839; full graph (nearest 64, all pairs) 1.608.

Sweep	Configuration	Avg. Lanes	Avg. Edges	minADE (m)
Hop depth ($L_{\max} = 16$)	$h_{\max} = 1$	5.0	4.0	1.375
	$h_{\max} = 2$	12.4	11.4	1.358
	$h_{\max} = 3$ (default)	15.4	14.3	1.337
	$h_{\max} = 4$	15.7	14.7	1.369
Lane budget ($h_{\max} = 3$)	$L_{\max} = 8$	7.9	6.9	1.427
	$L_{\max} = 16$ (default)	15.4	14.3	1.337
	$L_{\max} = 32$	23.7	22.7	1.376

5.5. Sensitivity to Waterflow Hyperparameters

The waterflow construction has two hyperparameters: the BFS depth $h_{\max}$ and the lane budget $L_{\max}$. To test whether the results of Sections 5.3 and 5.4 depend on an accidental choice of these values, we swept both around the default configuration ($h_{\max} = 3$, $L_{\max} = 16$): hop depths $h_{\max} \in \{1, 2, 4\}$ at fixed $L_{\max} = 16$, and lane budgets $L_{\max} \in \{8, 32\}$ at fixed $h_{\max} = 3$, each trained with identical hyperparameters (seed 42, $K = 6, 8$ s). Table 7 presents the results.

Four observations follow. First, **the default configuration achieves the best point estimate in both sweeps**, indicating that the settings used throughout the paper are not an accidental choice. Second, the answers to the natural saturation questions are direct: $L_{\max} = 16$ behaves as a saturation point—halving the budget to 8 lanes costs 0.090 m (the largest degradation in either sweep, exceeding the across-seed variability of ~ 0.08 m for this model class), while doubling it to 32 lanes does not help (1.376, slightly worse and within seed-level noise); similarly, deepening the BFS beyond 3 hops adds almost no new lanes ($15.4 \rightarrow 15.7$ on average) and no accuracy benefit. Third, **the method is robust to these hyperparameters**: apart from the deliberately starved $L_{\max} = 8$ configuration, all variants lie within 0.039 m of the default—less than half the across-seed variability—and even the 1-hop graph with only five lanes on average retains most of the lane-conditioning benefit (1.375 vs. the no-lane baseline’s 1.839). Fourth, and most importantly for the questions raised in Sections 5.3 and 5.4, **every waterflow configuration in the sweep (1.337–1.427) outperforms every full-graph configuration (1.485–1.608)**: the advantage of topologically guided local selection is a property of the construction family, not of one tuned hyperparameter setting. We note for completeness that the narrowest cross-family margin (0.058 m, between the deliberately starved $L_{\max} = 8$ variant and the best full-graph variant) is itself within seed-level variability, whereas the default configuration’s margin over the best full-graph variant (0.148 m) is not. As with the decoupling ablation, this sweep uses a single seed (42); given that most within-family differences are below seed-level variability, we refrain from ranking the waterflow variants beyond noting the shallow optimum at the default.

5.6. Architecture-Agnostic Benefit

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

Table 8. Single-modal prediction at 8 s across architectures (100 epochs). Transformer results: mean $\pm$ std over 3 seeds (7, 42, 123), best validation epoch per seed; LSTM rows: seed 42 (the LSTM effect is seed-validated in its own settings, Tables 3 and 4).

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	4.340 $\pm$ 0.452	12.543 $\pm$ 1.153	0.740 $\pm$ 0.102	—
TF Lane-Cond.	3.175 $\pm$ 0.140	8.744 $\pm$ 0.286	0.611 $\pm$ 0.047	+26.8% ($p = 0.030$)

[†] Single seed (42).

Lane conditioning improves *both* architectures: +18.7% for LSTM (seed 42) and +26.8% for the Transformer, where the Transformer result is validated across three seeds (4.340 $\pm$ 0.452 m $\rightarrow$ 3.175 $\pm$ 0.140 m, paired t -test $p = 0.030$; per-seed improvements +25.1%, +32.5%, +22.0%). Descriptively, the across-seed spread is also smaller for the lane-conditioned model (std 0.452 vs. 0.140); with only three seeds, this variance difference is not statistically distinguishable ($F(2, 2) = 10.4$, $p = 0.088$), so we report it as an observation rather than a finding. 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 [20,29].

5.7. Horizon-Dependent Improvement

Table 9 summarizes the improvement from lane conditioning across all experimental settings.

Table 9. 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	+26.6% ($p = 0.003$)
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, 3 seeds	8 s	ADE	+26.8% ($p = 0.030$)

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-s horizon, showing how the gap between baseline and lane-conditioned models widens progressively. Figure 4 further decomposes the 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.

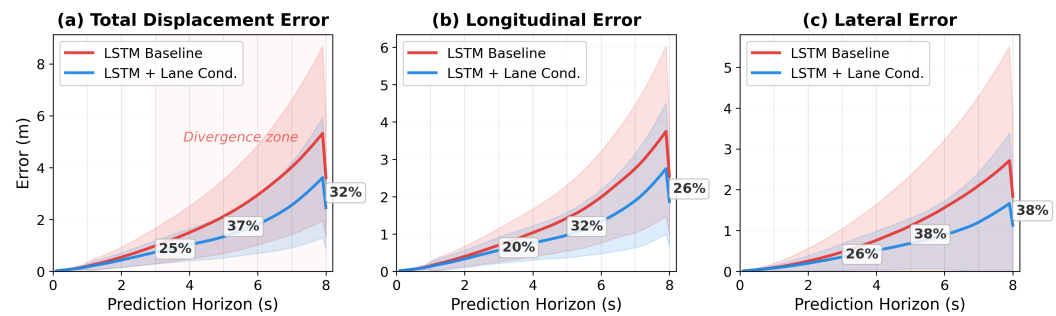
Error Growth Over Prediction Horizon: Baseline vs Lane-Conditioned (8s, K=6)

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.

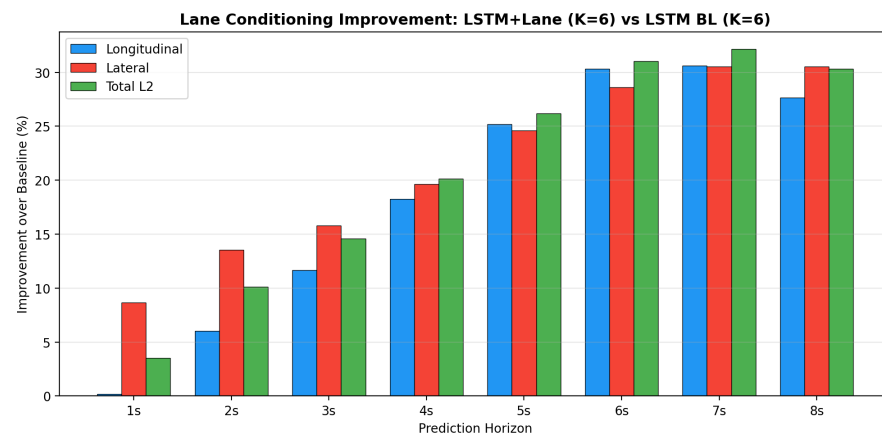


Figure 4. Percentage improvement from lane conditioning decomposed by prediction horizon and error component ($K = 6$). Longitudinal (blue), lateral (red), and total L2 (green) improvements all grow monotonically from 1 s to 7 s, with lateral improvement slightly dominant at shorter horizons.

5.8. Error Decomposition

Table 10 decomposes errors into lateral and longitudinal components for the multi-modal LSTM models at 8 s.

Table 10. 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 greatest improvement in endpoint lateral error (+30.5%). This is particularly important for intersection safety, as 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.

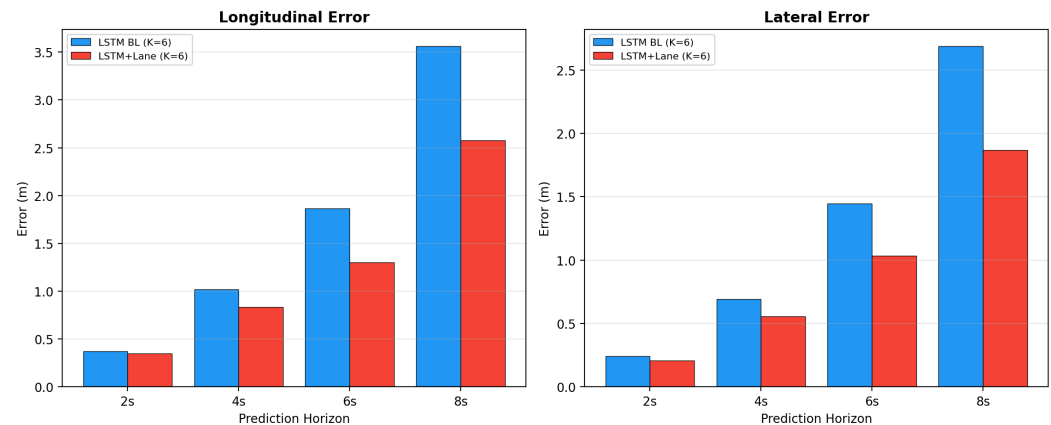


Figure 5. Absolute longitudinal (left) and lateral (right) errors at multiple horizons. The lane-conditioned model (red) consistently reduces both error components, with the gap growing at longer horizons.

5.9. Distribution of Improvement Across Scenarios and Maneuver Types

A single qualitative example cannot establish that an improvement is representative. We therefore analyze the per-scenario improvement of the lane-conditioned model over the baseline across *all* 13,388 validation scenarios (seed-42 checkpoints, $K = 6, 8$ s), stratified by maneuver type. Each scenario is classified from its ground-truth future trajectory into straight, left turn, right turn, slight turn/lane change, or stationary (path length < 3 m over 8 s; predominantly vehicles queued at red lights, which constitute 30.3% of validation scenarios in this signalized-intersection subset). Table 11 and Figure 6 summarize the results.

Table 11. Per-maneuver breakdown of the lane conditioning improvement over all 13,388 validation scenarios ($K = 6, 8$ s, seed-42 checkpoints). p -values from per-scenario paired t -tests.

Maneuver	n	Baseline minADE (m)	Lane-Cond. minADE (m)	Δ Means	Median Per-Scen. Δ	p
Straight	7580	1.977	1.513	+23.5%	+19.4%	<0.001
Left turn	893	5.512	3.498	+36.5%	+38.2%	<0.001
Right turn	484	5.083	4.058	+20.1%	+25.1%	<0.001
Slight turn/lane change	369	5.127	2.672	+47.9%	+44.9%	<0.001
Stationary (queued)	4062	0.091	0.087	+3.6%	n/a	0.027
All scenarios	13388	1.839	1.337	+27.3%	+7.8%	<0.001
Moving scenarios only	9326	2.601	1.881	+27.7%	+22.6%	<0.001

Note: Relative improvements are not meaningful for stationary scenarios, where both models achieve near-zero absolute error (<0.10 m).

Three observations follow. First, the improvement is broad-based rather than driven by outliers: among the 9326 moving scenarios, 67.2% improve under lane conditioning, with a median per-scenario improvement of +22.6%, and every moving maneuver class shows a positive median (Figure 6a). Second, the gain is largest precisely where lane structure is most informative: turning and lane-change maneuvers (+36.5% for left turns, +47.9% for slight turns/lane changes) benefit more than straight-ahead driving (+23.5%), consistent with the mechanism proposed in Section 6. Third, stationary scenarios—30.3% of the validation set—are essentially unaffected (both models < 0.10 m error), which explains why the all-scenario median (+7.8%) is much lower than the moving-scenario median: relative improvements on near-zero errors are dominated by noise.

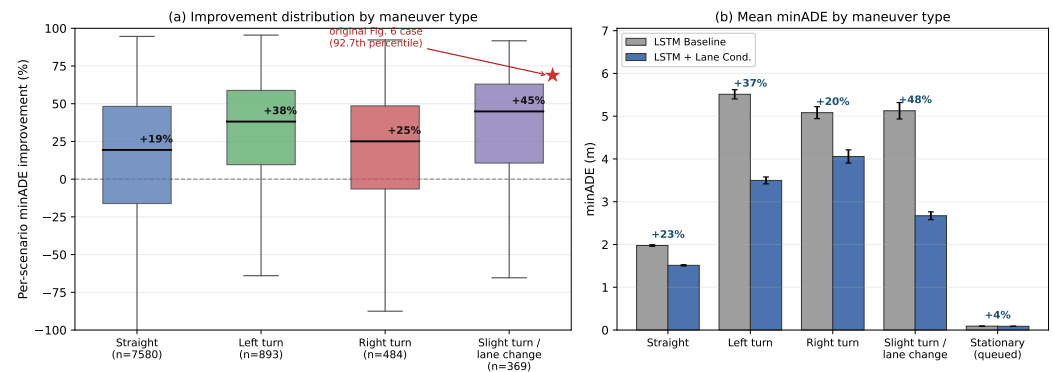


Figure 6. (a) Distribution of per-scenario minADE improvement by maneuver type (boxes: interquartile range; whiskers: 5th–95th percentile; outliers omitted for readability). Medians are positive for every moving maneuver class. The red star marks the single example presented in the originally submitted version of this manuscript (92.7th percentile of the improvement distribution). (b) Mean minADE per maneuver type. Turning maneuvers benefit most; stationary (queued) scenarios are essentially unaffected at near-zero absolute error.

Figure 7 presents qualitative examples selected by their position in this distribution rather than by visual impact: a median case (+29.3%, 65th percentile of all scenarios), a typical strong case (+53.6%, 83rd percentile), and a failure case (−78.2%, 18th percentile) in which lane conditioning *hurts*. For transparency, we note that the single example shown in the originally submitted version of this manuscript (baseline minADE 5.68 m vs. lane-conditioned 1.77 m, +68.9%) lies at the 92.7th percentile of the improvement distribution—an unusually favourable case, not a typical one. In the failure case (bottom row), the ego vehicle creeps slowly toward the stop line and proceeds straight; the baseline correctly concentrates its modes near the observed heading, whereas the lane-conditioned model distributes its modes across the intersection’s turning connections suggested by the lane graph, worsening minADE from 6.45 m to 11.49 m. This failure mode—over-dispersion of prediction modes along topologically available but unused lane branches when kinematic evidence is weak—is the characteristic downside of conditioning on lane structure, and suggests that confidence calibration between kinematic and structural cues is a worthwhile direction for future work.

5.10. Contextual Reference to Published Waymo Baselines

To situate the absolute magnitude of our errors within the literature, the Supplementary Material (Table S1) lists published Waymo LSTM baselines [17] alongside our models. **This is not a fair comparison, and readers should not infer the superiority or inferiority of any method from it.** The task settings differ fundamentally: our models perform ego-vehicle self-prediction (one prediction per scene for the SDC, which is always a vehicle) on a custom 85/15 split of 89,258 signal-controlled scenarios drawn from the training partition, whereas the Waymo baselines perform marginal prediction for up to eight heterogeneous agents per scene on the official validation set across all scenario types. We therefore make only a deliberately weak, qualitative observation: our lane-conditioned LSTM attains errors of the same order of magnitude as published LSTM baselines that consume substantially richer input features, which is consistent with—but does not prove—the view that local lane structure carries substantial predictive information for ego-vehicle prediction at signalized intersections. Rigorous benchmarking would require multi-agent marginal prediction on the official evaluation protocol, which we identify as future work, as is integrating lane conditioning into state-of-the-art architectures such as MTR [15] or MTR++ [44], which achieve substantially lower errors through rich features, dense map encodings, and iterative refinement.

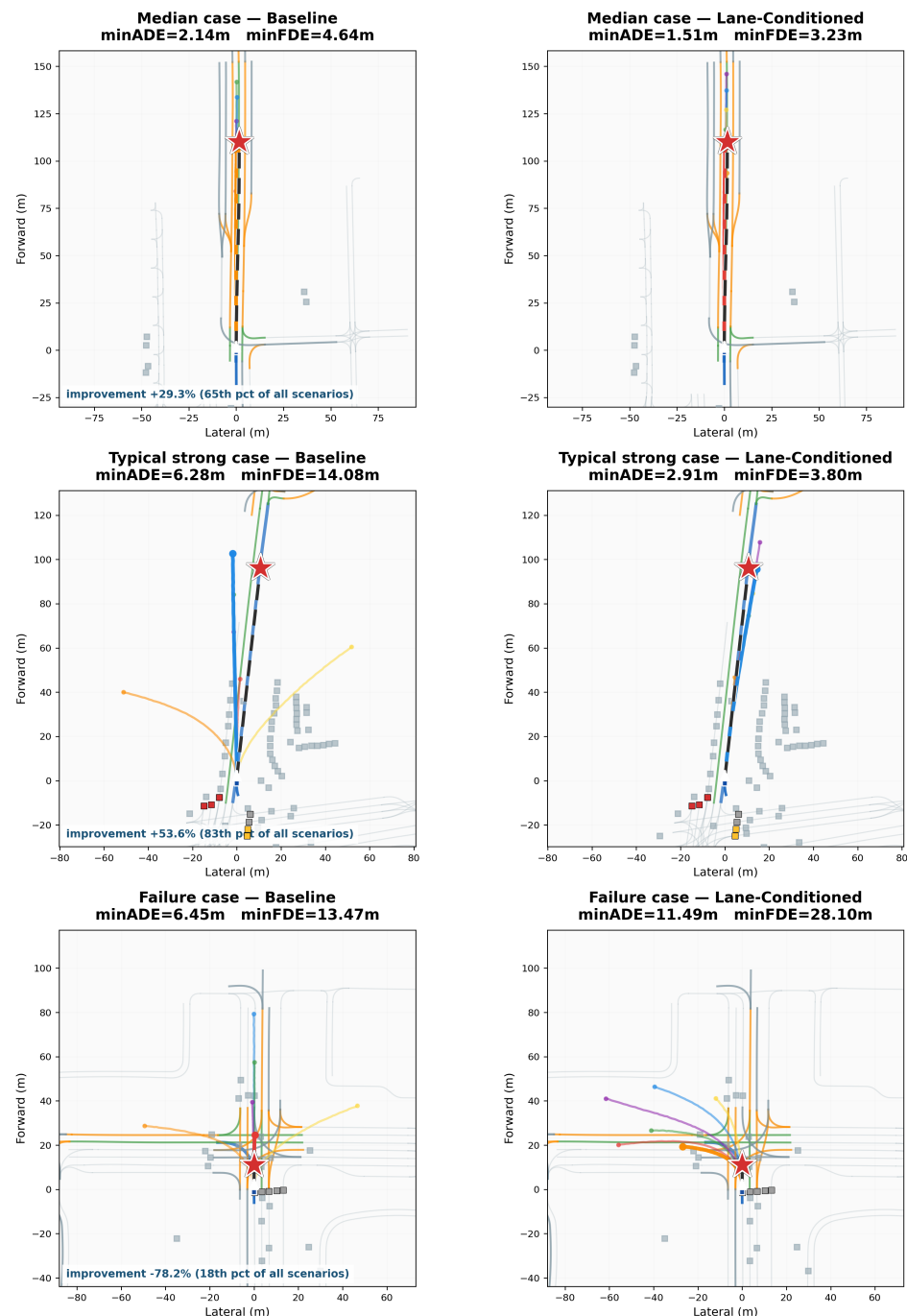


Figure 7. Qualitative comparison at three percentiles of the improvement distribution ($K = 6$ modes, 8 s prediction; (left): LSTM baseline, (right): LSTM + lane conditioning). **Top:** median case (+29.3% minADE improvement, 65th percentile of all scenarios—the median of moving scenarios). **Middle:** typical strong case (+53.6%, 83rd percentile). **Bottom:** failure case (−78.2%, 18th percentile), where the lane-conditioned model over-disperses its modes along available turning branches while the vehicle creeps straight through the intersection. Colored lines show predicted branches; the black dashed line shows ground truth; the red star marks the ground-truth endpoint.

6. Discussion

6.1. Why Lane Conditioning Helps More at Longer Horizons

The improvement increases from +9.3% at 3 s to +26.6% 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 the 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 improvement observed for both LSTM (+18.7%) and Transformer (+26.8%, $p = 0.030$ across three seeds) backbones indicates 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 benefit substantially, suggesting that even simple lane fusion mechanisms are effective. With the Transformer now validated across the same three seeds as the LSTM settings, the architecture-agnostic claim rests on seed-validated evidence in all three headline configurations.

The Transformer shows a larger relative improvement (+27% 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 6.7, the across-seed spread of the Transformer is descriptively smaller under lane conditioning (std 0.452 vs. 0.140; not statistically distinguishable at $n = 3$), which is directionally consistent with a mild implicit regularization effect.

6.3. Why Local Lane Selection May Outperform the Full Graph

The full-graph ablation suggests that *more lanes are not always better*. Despite receiving $4\times$ more lane information (64 vs. 16 lanes), the full-graph model trails the waterflow model by 11.4% at 8 s—a consistent trend, though one that does not reach statistical significance across three seeds ($p = 0.063$; Section 5.3). The interpretations below should accordingly be read as hypotheses that are compatible with the observed trend, not as demonstrated mechanisms. Three such mechanisms could contribute:

1. **Signal dilution in message passing.** GNN message passing on a 64-node graph with ~ 106 edges distributes information more diffusely than a compact 16-node graph with ~ 15 edges. Ego-relevant structural cues (e.g., successor lane curvature) are diluted by messages from distant, irrelevant lanes.
2. **Attention distraction.** Cross-attention must distribute its weights over $4\times$ more key-value pairs. Distant lanes that happen to have similar geometry to the ego lane can attract spurious attention, diverting capacity from the most informative local lanes.
3. **Topological relevance.** The waterflow BFS naturally selects lanes along the ego vehicle's plausible future paths—successors for straight-ahead and adjacent lanes for lane changes. Euclidean proximity, used by the full graph, does not distinguish between a nearby lane *ahead* of the vehicle (highly relevant) and a nearby lane on a *parallel road behind* the vehicle (irrelevant).

This trend parallels a broader principle in representation learning: curated, task-relevant features often outperform exhaustive feature sets [9]. The waterflow graph implements a *domain-specific inductive bias* that leverages lane connectivity structure to select the most decision-relevant subset of the map. The decoupling ablation in Section 5.4 provides a more direct handle on which of the two construction factors—lane selection or adjacency density—drives the observed difference.

6.4. 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 specific design principle within a controlled scope.** We observe a consistent 19–27% improvement across both backbone architectures, seed-validated with statistical significance in every headline setting ($p < 0.01$ for the LSTM configurations, $p = 0.030$ for the Transformer). This finding has direct practical implications: trajectory prediction systems using position-based features at signal-controlled intersections 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++ [44] 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, and—as discussed in Section 5.10—attain errors of the same order of magnitude as published LSTM baselines that use far richer inputs, under task settings that are not directly comparable. Within its scope, this supports local lane conditioning as a promising design principle for resource-constrained prediction systems.

6.5. 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 [45]. 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 CO₂ emissions by up to 40% compared to free-flow conditions. We do not directly measure downstream planning or emission outcomes. In principle, a 42.7% reduction in miss rate at the 5 m threshold could reduce false-positive emergency braking events and improve the reliability of cooperative maneuver planning, indirectly contributing to smoother traffic flow and lower fuel consumption; however, these downstream effects would need to be verified through integration with planning modules. A complete sustainability assessment would require such integration together with real-world or simulation-based driving experiments, which we leave for future work.

The lightweight nature of our lane conditioning module (<700,000 total parameters) could contribute to more efficient prediction systems, though concrete sustainability impacts would need to be evaluated through real-world deployment studies. 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) is directionally aligned with UN SDG 9 (Industry, Innovation and Infras-

structure), though quantifying the actual energy savings would require deployment-level benchmarking on target edge platforms.

6.6. 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 [46]. State-of-the-art methods such as MTR++ [44] employ architectures with ~ 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.

Parameter counts and training times alone, however, cannot substantiate claims about real-time deployment or onboard energy consumption. We therefore measured inference latency, memory, throughput, and energy per prediction for all model variants; Table 12 reports the results. Measurements were taken on a desktop workstation (NVIDIA GeForce RTX 4090 GPU; Intel Core i7-13700KF CPU (Intel Corporation, Santa Clara, CA, USA); PyTorch v2.7.1 (Meta Platforms, Inc., Menlo Park, CA, USA); CUDA 12.6; FP32; no TensorRT or other inference optimizations): batch-size-1 latency over 500 timed runs after 50 warm-up runs (with device synchronization), single-threaded CPU latency over 200 runs, peak GPU memory via the CUDA allocator, and GPU energy estimated by sampling board power at 100 ms intervals (nvidia-smi) during sustained batch-size-1 inference, subtracting idle power (6.4 W), and multiplying by the mean per-frame latency.

Table 12. Measured inference efficiency (batch size 1 unless noted). GPU: RTX 4090; CPU: i7-13700KF, single thread. Energy is a desktop-GPU estimate: (load – idle) power $\times$ mean latency. Waterflow lane graph preprocessing adds a mean of 2.0 ms (95th percentile 3.1 ms) on a single CPU thread for the lane-conditioned models.

Model (8 s)	Params	MACs/Frame	GPU (ms)	CPU (ms)	Peak GPU Mem (MB)	Energy (mJ/Frame)
LSTM Baseline ($K = 6$)	628,582	5.76 M	0.54	0.60	28.9	41.9
LSTM Lane-Cond. ($K = 6$)	675,750	6.26 M	0.80	0.74	29.0	50.8
Transformer Baseline	456,576	4.45 M	0.52	0.52	28.2	31.2
Transformer Lane-Cond.	608,640	5.82 M	0.91	0.80	28.7	56.0
MTR++ [44] (reported)	~ 15 M	—	—	—	—	—

Three points stand out. First, all variants run in under 1 ms per prediction on the GPU and under 1 ms single-threaded on a desktop CPU—with lane graph preprocessing included, the full lane-conditioned pipeline completes in under 3 ms on a single CPU thread, a $30\times$ margin against a 10 Hz (100 ms) planning cycle even without GPU hardware. Second, lane conditioning increases energy per prediction by roughly 9–25 mJ on this hardware (e.g., 41.9 to 50.8 mJ for the multi-modal LSTM)—the accuracy gains of Section 5 are not free, but their energy cost is small in absolute terms. Third, peak GPU memory remains below 30 MB for all variants, orders of magnitude below the capacity of automotive-grade accelerators. Training costs are correspondingly modest: every result in this paper was produced on a single consumer GPU, with each 100-epoch run requiring between 1.5 and 3 GPU-hours (measured wall-clock across all runs), rather than the multi-GPU cluster budgets typical of state-of-the-art systems.

We state the scope of these measurements plainly: they are desktop-hardware proxy measurements, not automotive measurements. Board-level power sampling on a discrete GPU is not equivalent to the power behavior of an automotive SoC (e.g., NVIDIA Orin or Xavier), and we did not measure a production perception-to-planning pipeline. The numbers above therefore *suggest feasibility* of deployment on resource-constrained plat-

forms rather than demonstrate it; on-vehicle validation, including hardware-level power instrumentation on automotive SoCs, remains future work.

6.7. 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 pronounced for the Transformer architecture, and the three-seed replication added in this revision allows us to characterize it precisely, including which parts of our original single-seed observation did and did not replicate. Across all three seeds, the Transformer *baseline* reaches its best validation ADE before the end of training and then degrades (best epochs 59, 29, and 89 for seeds 7, 42, and 123, with subsequent degradations of +4.0%, +26.7%, and +1.8% respectively), whereas the *lane-conditioned* Transformer improves essentially monotonically on every seed, reaching its best ADE at epochs 94–99 with at most 0.5% subsequent degradation. Two conclusions follow. First, the qualitative pattern—baseline degrades, lane-conditioned model does not—replicates on all three seeds; descriptively, the across-seed standard deviation of the final result is also smaller under lane conditioning (0.452 vs. 0.140), although with three seeds this variance difference is not statistically distinguishable ($F(2, 2) = 10.4$, $p = 0.088$). Second, the *catastrophic* severity we originally reported (26.7% degradation) occurred only on seed 42; on the other seeds, the baseline degradation is mild. We therefore interpret lane conditioning as providing a modest but consistent *implicit regularization* and training-stabilization effect [47]—an instance of the relational inductive biases described by Battaglia et al. [9]—while explicitly retracting the implication that catastrophic overfitting is the typical baseline behavior: severe overfitting onset is initialization-dependent, and lane conditioning’s clearest benefit is protection against that worst case. The practical benefit remains meaningful: more robust training with reduced sensitivity to initialization lowers the amount of computation wasted on failed or degraded runs, aligning with Green AI principles [46].

6.8. Limitations

This study has several limitations, most of which stem from deliberately narrowing the experimental scope to isolate the lane conditioning effect; the price of this internal control is external validity.

1. **Scope of the evaluation.** We evaluate a single task configuration: ego-vehicle (SDC) self-prediction at signal-controlled intersections, on a custom 85/15 random split (seed 42) of 89,258 scenarios rather than the official WOMB partitions. All conclusions—including “architecture-agnostic” and “general”—are claims about consistency *within* this scope. Whether the benefit transfers to multi-agent marginal prediction, to pedestrians and cyclists, to unsignalized intersections, highways, or roundabouts, and to the official evaluation protocol remains untested.
2. **Residual single-seed entries.** All headline results (LSTM at 3 s and 8 s $K = 6$; Transformer at 8 s) are now validated across three seeds. However, several secondary entries remain single-seed: the single-modal LSTM rows at 8 s in Table 8, the miss-rate figure (MR@5 m), the error decomposition (Table 10), the per-maneuver analysis (Section 5.9), and the decoupling ablation (Section 5.4) all use seed-42 checkpoints, trading seed replication for breadth of analysis within our compute budget.
3. **Coverage of the graph design space.** The decoupling ablation (Section 5.4) and the hyperparameter sweeps (Section 5.5) cover the selection/adjacency factors, hop depths 1–4, and lane budgets 8–32 for the waterflow family, but each with a single

seed (42); since most within-family differences fall below across-seed variability, the fine ranking among waterflow variants should not be over-interpreted. Alternative selection strategies—radius-based or direction-aware geometric filtering—were not trained; we argue in Section 5.4 that topological connectivity subsumes the main property such filters approximate (excluding physically near but drivably unreachable lanes), but an empirical comparison remains future work.

4. **Input minimalism.** Our models use only 2D position inputs; incorporating velocity, heading, and bounding-box features would likely improve absolute performance, though it would complicate attribution of improvements to lane conditioning.
5. **Efficiency measurements are desktop proxies.** Latency, memory, and energy figures (Section 6.6) were measured on a desktop GPU/CPU without inference optimization; they suggest, but do not demonstrate, deployability on automotive hardware.
6. **Gap to state-of-the-art.** The remaining gap to methods like MTR++ 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 those systems.

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 as follows:

1. **Consistent improvement across architectures (within scope).** Lane conditioning improves both LSTM (+9.3% to +26.6%, $p < 0.01$ across three seeds at both horizons) and Transformer (+26.8%, $p = 0.030$ across three seeds) backbones for ego-vehicle prediction at signal-controlled intersections.
2. **Horizon-dependent benefit.** Improvement increases from +9.3% at 3 s to +26.6% 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. **Lane structure as a compact input.** Our lane-conditioned LSTM with $K = 6$ modes achieves mean minADE = 1.37 ± 0.08 m for ego-vehicle prediction using only 2D positions plus 16 local lane features—errors of the same order of magnitude as published LSTM baselines that consume substantially richer inputs, although the task settings are not comparable and no equivalence is claimed (Section 5.10).
5. **A trend favouring local selection over brute-force.** A full-graph baseline using the nearest 64 lanes improves over the no-lane baseline (+17.2% minADE, $p = 0.003$), and the waterflow method with $4\times$ fewer lanes shows a further +11.4% mean improvement—a consistent trend across horizons and metrics that does not, however, reach statistical significance ($p = 0.063$).

These results support local lane graph conditioning as a lightweight, modular, and pluggable component for ego-vehicle trajectory prediction at signalized intersections—one that attaches to an existing recurrent or attention-based encoder without architectural changes, at $\sim 8\%$ parameter overhead and millisecond-scale inference cost. Establishing the same effect beyond this scope is the natural next step. Future work will explore: (1) multi-seed validation of the local-vs.-full-graph trend and of the decoupling ablation; (2) integration of the lane conditioning module into state-of-the-art architectures such as MTR to assess whether improvements transfer at higher performance levels; (3) joint multi-agent prediction leveraging shared lane graph representations, together with more diverse scenario types and the official WOMB evaluation protocol; (4) cross-dataset generalization to Argoverse [48] and nuScenes [49]; and (5) quantifying the downstream impact of im-

proved prediction accuracy on autonomous vehicle energy efficiency and safety outcomes, including power measurements on automotive-grade hardware.

Supplementary Materials: The following supporting information can be downloaded at: <https://www.mdpi.com/article/10.3390/su1010000/s1>: Table S1: published Waymo LSTM baselines and our models at the 8 s horizon (vehicle minADE, meters; presented for order-of-magnitude context only, as the task settings are not comparable; see Section 5.10); Table S2: direct ego-lane identification rate as a function of the distance threshold, over all 89,258 signal-controlled scenarios (Section 3.2).

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.

Informed Consent Statement: Not applicable.

Data Availability Statement: The trajectory prediction models and training code developed in this study are publicly available at <https://github.com/Jynxzzz/lane-graph-conditioning> (accessed on 5 August 2026). The Waymo Open Motion Dataset used for training and evaluation is publicly available at <https://waymo.com/open/> (accessed on 5 August 2026; dataset download requires free registration) under the Waymo Dataset License Agreement.

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
GNN	Graph Neural Network
HD	High Definition
LSTM	Long Short-Term Memory
minADE	Minimum Average Displacement Error (over K modes)
minFDE	Minimum Final Displacement Error (over K modes)
MLP	Multi-Layer Perceptron
MR	Miss Rate
SDC	Self-Driving Car
SDG	Sustainable Development Goal
SOTA	State Of The Art
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): Washington, DC, USA, 2010.
2. Barth, M.; Boriboonsomsin, K. Traffic Congestion and Greenhouse Gases. *ACCESS Mag.* **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. *Environ. Sci. Technol.* **2018**, *52*, 11449–11465.
4. Contreras-Castillo, J.; Zeadally, S.; Guerrero-Ibáñez, J.A. Internet of Vehicles: Architecture, Protocols, and Security. *IEEE Internet Things J.* **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 Trans. Intell. Veh.* **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 Trans. Intell. Transp. Syst.* **2022**, *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. *Accid. Anal. Prev.* **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.; et al. Relational Inductive Biases, Deep Learning, and Graph Networks. *arXiv* **2018**, arXiv:1806.01261.
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. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 11525–11533. <https://doi.org/10.1109/CVPR42600.2020.01154>.
11. Liang, M.; Yang, B.; Hu, R.; Chen, Y.; Liao, R.; Feng, S.; Urtasun, R. Learning Lane Graph Representations for Motion Forecasting. In Proceedings of the European Conference on Computer Vision (ECCV); Springer: Berlin/Heidelberg, Germany, 2020; pp. 541–556. https://doi.org/10.1007/978-3-030-58536-5_32.
12. Zhao, H.; Gao, J.; Lan, T.; Sun, C.; Sapp, B.; Varadarajan, B.; Shen, Y.; Shen, Y.; Chai, Y.; Schmid, C.; et al. TNT: Target-driven Trajectory Prediction. In Proceedings of the Conference on Robot Learning (CoRL), London, UK, 8–11 November 2021; pp. 895–904.
13. Zhou, Z.; Ye, L.; Wang, J.; Wu, K.; Lu, K. HiVT: Hierarchical Vector Transformer for Multi-Agent Motion Prediction. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; pp. 8813–8823. <https://doi.org/10.1109/CVPR52688.2022.00862>.
14. Deo, N.; Wolff, E.M.; Beijbom, O. Multimodal Trajectory Prediction Conditioned on Lane-Graph Traversals. In Proceedings of the Conference on Robot Learning (CoRL), London, UK, 8–11 November 2021; pp. 203–212.
15. Shi, S.; Jiang, L.; Dai, D.; Schiele, B. Motion Transformer with Global Intention Localization and Local Movement Refinement. In Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), New Orleans, LA, USA, 28 November–9 December 2022; Volume 35, pp. 6531–6543.
16. Zhou, Z.; Wang, J.; Li, Y.H.; Huang, Y.K. Query-Centric Trajectory Prediction. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 17–24 June 2023; pp. 17863–17873. <https://doi.org/10.1109/CVPR52729.2023.01713>.
17. Ettinger, S.; Cheng, S.; Caine, B.; Liu, C.; Zhao, H.; Pradhan, S.; Chai, Y.; Sapp, B.; Qi, C.R.; Zhou, Y.; et al. Large Scale Interactive Motion Forecasting for Autonomous Driving: The Waymo Open Motion Dataset. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, 10–17 October 2021; pp. 9690–9699. <https://doi.org/10.1109/ICCV48922.2021.00957>.
18. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780.
19. Alahi, A.; Goel, K.; Ramanathan, V.; Robicquet, A.; Fei-Fei, L.; Savarese, S. Social LSTM: Human Trajectory Prediction in Crowded Spaces. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 961–971. <https://doi.org/10.1109/CVPR.2016.110>.
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. In Proceedings of the 2018 IEEE Intelligent Vehicles Symposium (IV); IEEE: New York, NY, USA, 2018; pp. 1672–1678.
21. Deo, N.; Trivedi, M.M. Convolutional Social Pooling for Vehicle Trajectory Prediction. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Salt Lake City, UT, USA, 18–22 June 2018; pp. 1468–1476.
22. Yang, Y.; Negash, N.M.; Yang, J. Recent Advances in Interactive Driving of Autonomous Vehicles: Comprehensive Review of Approaches. *Automot. Innov.* **2025**, *8*, 304–334. <https://doi.org/10.1007/s42154-024-00332-w>.
23. Chandra, R.; Guan, T.; Panuganti, S.; Mittal, T.; Bhattacharya, U.; Bera, A.; Manocha, D. Forecasting Trajectory and Behavior of Road-Agents Using Spectral Clustering in Graph-LSTMs. *IEEE Robot. Autom. Lett.* **2020**, *5*, 4882–4890.
24. Li, J.; Yang, F.; Tomizuka, M.; Choi, C. EvolveGraph: Multi-Agent Trajectory Prediction with Dynamic Relational Reasoning. In Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), Virtual, 6–12 December 2020; Volume 33, pp. 19783–19794.

25. Mo, X.; Xing, Y.; Lv, C. Graph and Recurrent Neural Network-Based Vehicle Trajectory Prediction for Highway Driving. In *Proceedings of the 2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*; IEEE: New York, NY, USA, 2021; pp. 1934–1939. <https://doi.org/10.1109/ITSC48978.2021.9564929>.
26. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention Is All You Need. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, Long Beach, CA, USA, 4–9 December 2017; Volume 30.
27. Ngiam, J.; Caine, B.; Vasudevan, V.; Zhang, Z.; Chiang, H.T.L.; Ling, J.; Roelofs, R.; Bewley, A.; Liu, C.; Venugopal, A.; et al. Scene Transformer: A Unified Architecture for Predicting Multiple Agent Trajectories. In *Proceedings of the International Conference on Learning Representations (ICLR)*, Virtual, 25–29 April 2022.
28. Nayakanti, N.; Al-Rfou, R.; Zhou, A.; Goel, K.; Refaat, K.S.; Sapp, B. Wayformer: Motion Forecasting via Simple & Efficient Attention Networks. In *Proceedings of the 2023 IEEE International Conference on Robotics and Automation (ICRA)*; IEEE: New York, NY, USA, 2023; pp. 2980–2987.
29. Zeng, A.; Chen, M.; Zhang, L.; Xu, Q. Are Transformers Effective for Time Series Forecasting? In *Proceedings of the AAAI Conference on Artificial Intelligence*, Washington, DC, USA, 7–14 February 2023; Volume 37, pp. 11121–11128.
30. Kipf, T.N.; Welling, M. Semi-Supervised Classification with Graph Convolutional Networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, Toulon, France, 24–26 April 2017.
31. Zeng, W.; Liang, M.; Liao, R.; Urtasun, R. LaneRCNN: Distributed Representations for Graph-Centric Motion Forecasting. In *Proceedings of the IEEE/RISJ International Conference on Intelligent Robots and Systems (IROS)*; IEEE: New York, NY, USA, 2021; pp. 532–539.
32. Gu, J.; Sun, C.; Zhao, H. DenseTNT: End-to-End Trajectory Prediction from Dense Goal Sets. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, Montreal, QC, Canada, 10–17 October 2021; pp. 15303–15312.
33. 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. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Nashville, TN, USA, 20–25 June 2021; pp. 14636–14645.
34. Wang, M.; Zhu, X.; Yu, C.; Li, W.; Ma, Y.; Jin, R.; Ren, X.; Ren, D.; Wang, M.; Yang, W. GANet: Goal Area Network for Motion Forecasting. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*; IEEE: New York, NY, USA, 2023; pp. 1609–1615.
35. Gilmer, J.; Schoenholz, S.S.; Riley, P.F.; Vinyals, O.; Dahl, G.E. Neural Message Passing for Quantum Chemistry. In *Proceedings of the International Conference on Machine Learning (ICML)*, Sydney, Australia, 6–11 August 2017; pp. 1263–1272.
36. Bahdanau, D.; Cho, K.; Bengio, Y. Neural Machine Translation by Jointly Learning to Align and Translate. In *Proceedings of the International Conference on Learning Representations (ICLR)*, San Diego, CA, USA, 7–9 May 2015.
37. Gupta, A.; Johnson, J.; Fei-Fei, L.; Savarese, S.; Alahi, A. Social GAN: Socially Acceptable Trajectories with Generative Adversarial Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Salt Lake City, UT, USA, 18–23 June 2018; pp. 2255–2264. <https://doi.org/10.1109/CVPR.2018.00240>.
38. Lee, S.; Purushwalkam, S.; Cogswell, M.; Ranjan, V.; Crandall, D.; Batra, D. Stochastic Multiple Choice Learning for Training Diverse Deep Ensembles. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, Barcelona, Spain, 9–10 December 2016; Volume 29.
39. Salzmann, T.; Ivanovic, B.; Chakravarty, P.; Pavone, M. Trajectron++: Dynamically-Feasible Trajectory Forecasting with Heterogeneous Data. In *Proceedings of the European Conference on Computer Vision (ECCV)*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 683–700.
40. Schöller, C.; Aravantinos, V.; Lay, F.; Knoll, A. What the Constant Velocity Model Can Teach Us About Pedestrian Motion Prediction. *IEEE Robot. Autom. Lett.* **2020**, *5*, 1696–1703.
41. Huber, P.J. Robust Estimation of a Location Parameter. *Ann. Math. Stat.* **1964**, *35*, 73–101.
42. Loshchilov, I.; Hutter, F. Decoupled Weight Decay Regularization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, New Orleans, LA, USA, 6–9 May 2019.
43. Loshchilov, I.; Hutter, F. SGDR: Stochastic Gradient Descent with Warm Restarts. In *Proceedings of the International Conference on Learning Representations (ICLR)*, Toulon, France, 24–26 April 2017.
44. Shi, S.; Jiang, L.; Dai, D.; Schiele, B. MTR++: Multi-Agent Motion Prediction with Symmetric Scene Modeling and Guided Intention Querying. *IEEE Trans. Pattern Anal. Mach. Intell.* **2024**, *46*, 3955–3971. <https://doi.org/10.1109/TPAMI.2024.3352811>.
45. World Health Organization. *Global Status Report on Road Safety 2023*; Technical report; World Health Organization: Geneva, Switzerland, 2023.
46. Schwartz, R.; Dodge, J.; Smith, N.A.; Etzioni, O. Green AI. *Commun. ACM* **2020**, *63*, 54–63.
47. Neyshabur, B.; Tomioka, R.; Srebro, N. In Search of the Real Inductive Bias: On the Role of Implicit Regularization in Deep Learning. In *Proceedings of the ICLR 2015 Workshop Track*, San Diego, CA, USA, 7–8 May 2015.

48. Chang, M.F.; Lambert, J.; Sangkloy, P.; Singh, J.; Bak, S.; Hartnett, A.; Wang, D.; Carr, P.; Lucey, S.; Ramanan, D.; et al. Argoverse: 3D Tracking and Forecasting with Rich Maps. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 8748–8757.
49. Caesar, H.; Bankiti, V.; Lang, A.H.; Vora, S.; Liong, V.E.; Xu, Q.; Krishnan, A.; Pan, Y.; Baldan, G.; Beijbom, O. nuScenes: A Multimodal Dataset for Autonomous Driving. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Virtual, 16–18 June 2020; pp. 11618–11628.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.