Orion: Spatiotemporal Neural Networks with Self-Validating Directed Graphs for Traffic and Epidemic Forecasting

Anonymous Author(s)

Abstract

Spatiotemporal forecasting is fundamental to intelligent transportation and epidemic prediction, yet existing methods face challenges in modeling complex inter-node dependencies and multi-scale temporal dynamics. A key limitation lies in their reliance on symmetric correlations, which fail to capture how perturbations propagate directionally through real-world networks. To address this gap, we propose Orion, a framework that learns directed dependency structures grounded in Granger causality principles and validates them through prediction-based consistency testing—a mechanism notably absent in prior spatiotemporal graph learning methods that provides a rigorous training signal beyond pure correlation by testing whether learned edge weights remain consistent with their predictive effects. Building on this foundation, our framework further integrates: (1) the TE-CausGAT module, which estimates time-varying directed dependencies considering both lagged and instantaneous effects and integrates them with prior spatial structures through Bayesian fusion; (2) the Belt Block architecture for efficient parallel multi-scale temporal modeling across hourly, daily, and weekly patterns; (3) a three-stage progressive training strategy that transitions from feature learning to directed structure learning to end-to-end optimization. Empirically, Orion achieves state-of-the-art performance on multiple real-world benchmarks: in traffic forecasting, it improves MAE by 6.60% on METR-LA and 8.65% on PEMS-08, with notable short-horizon gains on PEMS-04 though long-horizon performance remains competitive rather than dominant; in epidemic forecasting, it improves MAE by 22.70% on CA and 37.73% on TX. DREAM-3 experiments confirm that learned directed structures align well with ground-truth regulatory networks, while case studies on METR-LA demonstrate that discovered dependencies correspond to known traffic propagation patterns.

CCS Concepts

• **Information systems** → *Data mining*; **Spatial-temporal systems**; • **Applied computing** → **Forecasting**.

Keywords

spatiotemporal forecasting, Granger causal graph learning, graph attention networks, intervention-based validation, multi-scale temporal modeling, multi-head attention

ACM Reference Format:
Anonymous Author(s). 2026. Orion: Spatiotemporal Neural Networks with Self-Validating Directed Graphs for Traffic and Epidemic Forecasting. In . ACM, New York, NY, USA, 19 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

Spatiotemporal forecasting underpins the modeling of complex dynamical systems across intelligent transportation and infectious disease prediction. What most existing methods overlook is that real-world spatiotemporal systems evolve through directed dependency structures rather than merely symmetric correlations.

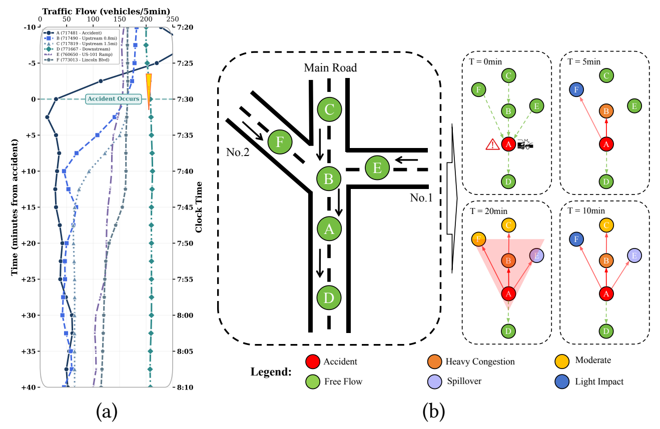


Figure 1: Directed dependency propagation in METR-LA traffic networks. (a) Traffic flow dynamics showing six nodes before/after accident at T=0; node labels (A–F) correspond to sensor IDs in legend. (b) Spatial snapshots: left shows road topology; right shows temporal evolution at T=0, 5, 10, 20min. Node colors indicate congestion severity (green→red). Green dashed arrows: traffic flow direction; red solid arrows: influence direction discovered by our model.

To illustrate, we examine a highway interchange scenario during morning rush hour (Figure 1). Traffic flows along the main road C→B→A→D, with branch roads connecting nodes E and F. At T=0min, an accident at Node A triggers *asymmetric* influence propagation: A affects upstream B and branch F, but not downstream D. By T=5min, B shows heavy congestion while E and F remain unaffected. By T=10min, congestion propagates further to C (B→C) with E lightly impacted. By T=20min, the directed chain A→B→C is fully established, F is more congested than E due to its more direct connection to A, and D remains in free-flow throughout. Discovering such directed dependencies—where influence propagates asymmetrically with varying intensities—is precisely the goal of Orion, enabling more accurate forecasting by leveraging these asymmetric predictive relationships.

Existing spatiotemporal forecasting methods have evolved from generic architectures to specialized designs. Early studies applied general deep learning architectures to spatiotemporal data: RNNs [34] and Transformers [41] model temporal dependencies, while GCN [24] and TCN [21] based on convolutions, and attention-based GAT [42], capture spatial patterns in graph-structured data. However, these architectures lack spatiotemporal coupling and struggle to capture complex spatial–temporal interactions simultaneously. The second category designs coupled architectures for spatiotemporal forecasting: DCRNN [25] integrates diffusion convolution with GRU, GWNet [44] introduces adaptive adjacency matrix learning, and AGCRN [3] and DGCRN [23] achieve spatiotemporal coupling through message passing. However, these methods typically assume static graph structures and treat inter-node dependencies as symmetric relationships, ignoring dependency directionality inherent in real-world spatiotemporal systems. The third category explores causal-inspired modeling: CaST [45] applies structural causal models to address confounding, while TESTAM [22] adaptively selects spatial modeling strategies. However, these methods either rely on strong assumptions or lack explicit directed dependency validation.

Despite advances in specific scenarios, these methods exhibit four limitations: (1) neglecting dependency directionality, unable to distinguish influence sources from targets; (2) static dependency assumptions hindering dynamic structural adaptation; (3) single-scale temporal modeling struggling to capture multi-period features; (4) no existing method validates whether learned spatiotemporal graph structures are consistent with their predictive effects. We propose Orion to address these issues, with the following contributions:

- We introduce a prediction-based consistency validation mechanism that, to our knowledge, is the first to validate learned spatiotemporal dependency structures by testing whether edge weights predict the actual sensitivity of downstream nodes to upstream masking, transcending correla tion-based graph learning by providing a self-consistency training signal that filters spurious edges.
- We design the TE-CausGAT module that estimates time-varying directed dependencies in the Granger sense, providing interpretable and validatable dependency structures whose primary value lies in structural discovery—confirmed by DREAM-3 alignment with ground-truth networks—and in growing importance at longer horizons. Bayesian fusion integrates learned structures with prior spatial knowledge for stability.
- We introduce the Belt Block architecture combining TE-CausGAT with multi-head attention for parallel multi-scale temporal modeling, trained via a three-stage progressive strategy from feature learning to directed structure learning to joint optimization.
- Orion achieves state-of-the-art results across five benchmarks spanning traffic and epidemic domains, with DREAM-3 experiments confirming that learned structures align with ground-truth directed regulatory networks (AUC 0.6295 vs. 0.5915 for the best baseline).

Our code and supplementary materials are available at https://anonymous.4open.science/r/Orion_Supplementary_Material/

2 Preliminaries

Definition 1 (Spatiotemporal Forecasting). Given a spatial network $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{A})$ with N nodes and adjacency matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$, let $\mathbf{X}^{(t)} \in \mathbb{R}^{N\times F}$ denote node features at time t. The task is to learn $f : \mathbb{R}^{N\times F\times T} \times \mathcal{G} \rightarrow \mathbb{R}^{N\times H}$ mapping historical observations $\{\mathbf{X}^{(t-T+1)}, ..., \mathbf{X}^{(t)}\}$ to future predictions $\{\hat{\mathbf{X}}^{(t+1)}, ..., \hat{\mathbf{X}}^{(t+H)}\}$.

Definition 2 (Granger-inspired Directed Dependency Graph). Following Granger causality—where X Granger-causes Y if past X improves prediction of Y—we define a dynamic directed dependency graph $\mathcal{C} = \{\mathbf{C}^{(s)}\}_{s=1}^{S}$, where $\mathbf{C}^{(s)} \in [0, 1]^{N\times N}$ captures directed predictive dependencies for temporal segment s. Element $\mathbf{C}_{ij}^{(s)}$ quantifies predictive influence from v_i to v_j, exhibiting asymmetry ($\mathbf{C}_{ij}^{(s)} \neq \mathbf{C}_{ji}^{(s)}$) and acyclicity ($\text{tr}(e^{\mathbf{C}^{(s)}}) - N = 0$) [48].

Definition 3 (Prediction-Based Consistency Validation). We validate learned dependencies through prediction-based testing: given edge weight $\mathbf{C}_{ij}$, we mask node v_i's contribution and measure the resulting prediction change at node v_j. High consistency between learned weights and observed prediction changes indicates that the dependency structure captures genuine predictive relationships rather than spurious correlations.

3 Methodology

As shown in Figure 2, during Orion's forward propagation, the input data generates three period tensors $\mathbf{X}_h, \mathbf{X}_d, \mathbf{X}_w \in \mathbb{R}^{N\times F\times L}$ through the LLM-enhanced retrieval mechanism, which are then transformed into embedded representations $\mathbf{E} \in \mathbb{R}^{N\times d_{model}\times L}$ through embedding layers with positional encodings added. Three parallel Belt Blocks independently process period data, sequentially performing TE-CausGAT spatial attention, multi-head temporal attention, and feedforward network operations, interspersed with residual connections [14] and layer normalization [2]. The fusion module receives branch outputs $\{\mathbf{H}_h, \mathbf{H}_d, \mathbf{H}_w\} \in \mathbb{R}^{N\times d_{model}}$, integrates three-period features through adaptive weighting and attention, and generates predictions $\mathbf{X}_{output} \in \mathbb{R}^{N\times H}$ through linear projection. Notably, Orion adopts three-stage training: the first stage learns feature representations, the second introduces directed dependency regularization with frozen embedding layers, and the third unfreezes parameters for end-to-end fine-tuning.

3.1 Fine-tuned LLM-enhanced Data Retrieval

Traditional methods rely on fixed temporal windows to select historical reference data, which performs poorly when encountering non-periodic variations. For example, traffic patterns during holidays differ significantly from those on regular weekdays, and fixed retrieval of "the same time last week" may introduce irrelevant pattern noise. To overcome this limitation, we propose a semantic similarity-based intelligent retrieval mechanism. By fine-tuning the TinyLlama-1.1B model on 100,000 temporal samples from the PEMS03 dataset (details in Appendix C), the LLM learns to understand semantic characteristics of temporal patterns, thereby enabling retrieval of truly similar historical patterns across calendar boundaries. As shown in Figure 3, taking the daily cycle as an example, both the input sequence $\mathbf{X}_h \in \mathbb{R}^{N\times F\times T}$ at current time t and the data from the same time periods over the past D days are encoded using a fine-tuned large language model, extracting

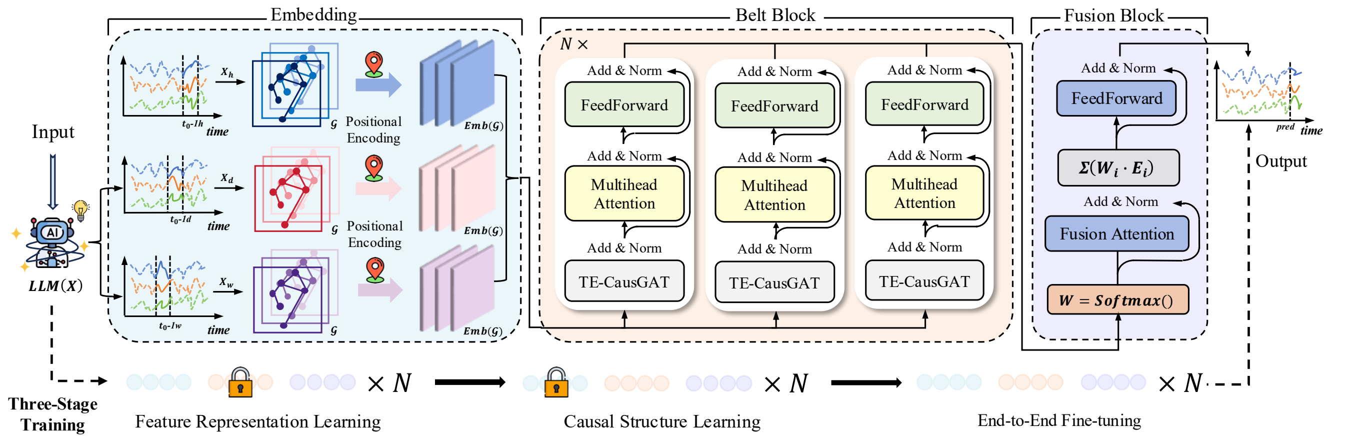


Figure 2: Overall architecture of the Orion framework. Three parallel Belt Blocks process hourly, daily, and weekly period data independently. Bottom: three-stage progressive training strategy.

the average of the last hidden layer as the semantic embedding $\mathbf{z_i} \in \mathbb{R}^{d_{embed}}$. The similarity between the current sequence and candidate historical sequences is measured through a weighted combination of cosine similarity and Euclidean distance:

$$S_{sim}(i,j) = w_{cos} \cdot \frac{\mathbf{z}_i^T \mathbf{z}_j}{||\mathbf{z}_i||_2 \cdot ||\mathbf{z}_j||_2} + w_{euc} \cdot \frac{1}{1 + ||\mathbf{z}_i - \mathbf{z}_j||_2} \quad (1)$$

Each node selects its most similar historical cycle as $\mathbf{X}_d$. It is important to emphasize that, with or without the LLM, all input data are embedded through the same fully connected layer (`nn.Linear`) with parameter dimensions $F \times d_{\text{model}}$, followed by positional encoding before being processed by the Belt Block. The role of the LLM is intelligent data selection rather than feature extraction.

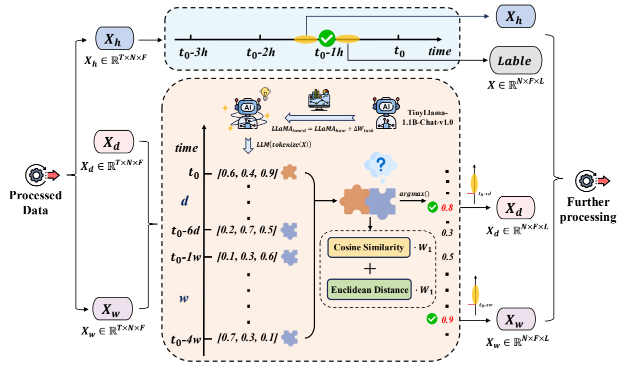


Figure 3: Fine-tuned LLM-enhanced data retrieval. $\mathbf{X}_h$ is directly extracted; $\mathbf{X}_d$ and $\mathbf{X}_w$ are selected via semantic similarity using fine-tuned TinyLlama-1.1B-Chat-v1.0.

3.2 Temporal Evolution Causal Graph Attention Network (TE-CausGAT)

TE-CausGAT is the core module for learning and validating directed dependency structures in Orion. We adopt the term "causal" in the Granger causality sense throughout this work: a variable X is said to Granger-cause Y if the past values of X contain information that helps predict Y beyond what is contained in past values of Y alone [11]. This predictive notion of causality is weaker than Pearl's interventional causality but is identifiable from observational time series data without requiring controlled experiments. Our TE-CausGAT module learns such Granger-causal directed dependencies and validates them through prediction-based consistency testing.

3.2.1 ***Granger-inspired Directed Dependency Learning.*** Convolutions on $\mathbf{X}_{input}$ yield multi-scale features through fusion in TE-CausGAT. Next, Temporal segmentation divides the series into S segments to capture causal evolution. Each segment s uses an independent estimator for its causal structure. For segment s, causal relationship strengths are computed by learning source and target node transformation matrices through time-weighted feature aggregation combined with node embeddings:

$$\mathbf{C}^{(s)} = \sigma(\mathbf{S}^{(s)} \cdot (\mathbf{T}^{(s)})^T + \mathbf{B}) \quad (2)$$

Where $\mathbf{S}^{(s)}, \mathbf{T}^{(s)}$ are source and target transforms, $\mathbf{B}$ is a bias matrix, σ is the sigmoid function, and the self-loop mask removes diagonal elements. The Bayesian causal fusion mechanism integrates segment-specific causal matrices through attention weights (the prior graph structure does not explicitly contain causal relationships) and incorporates prior structures from the adjacency matrix:

$$\mathbf{C}_{\text{fused}} = (1 - \alpha_{\text{prior}}) \cdot \sum_{s=1}^{S} w_s \cdot \mathbf{C}^{(s)} + \alpha_{\text{prior}} \cdot \mathbf{C}_{\text{prior}} \quad (3)$$

where w_s denotes the attention weight for segment s satisfying $\sum_{s=1}^{S} w_s = 1$, $\mathbf{C}_{\text{prior}}$ the prior adjacency, and α_{prior} balances discovered vs prior structures for stability. Subsequently, the dependency propagation simulator employs a GRU mechanism [8] to model the temporally dynamic propagation of perturbation effects:

$$\mathbf{h}^{(t)} = \text{GRU}(\mathbf{C}_{\text{fused}} \cdot \mathbf{X}^{(t)}, \mathbf{h}^{(t-1)}) \quad (4)$$

where $\mathbf{X}^{(t)} \in \mathbb{R}^{N\times d}$ is the node features at time t, $\mathbf{h}^{(t)} \in \mathbb{R}^{N\times d}$ is the hidden state, and the matrix product $\mathbf{C}_{\text{fused}} \cdot \mathbf{X}^{(t)}$ modulates information flow between nodes. The causal matrix modulates information flow between nodes and simulates the temporal propagation effects of node changes. To ensure the learned causal graph maintains acyclicity, we employ computationally efficient polynomial approximations instead of the theoretically exact but computationally expensive matrix exponential constraint $\mathrm{tr}(e^{\mathbf{C}^{(s)}}) - N = 0$. Specifically, we penalize the trace of powers of the normalized causal matrix: $\mathcal{L}_{DAG} = \lambda_2 \cdot \mathrm{tr}(\hat{\mathbf{C}}^2) + \lambda_3 \cdot \mathrm{tr}(\hat{\mathbf{C}}^3)$, where $\hat{\mathbf{C}} = \mathbf{C}_{\text{fused}}/\|\mathbf{C}_{\text{fused}}\|_\infty$, $\lambda_2 = 0.1$ and $\lambda_3 = 0.01$. This exploits $\mathrm{tr}(\mathbf{C}^k)$ counting k-cycles, suppressing cycles while remaining differentiable and efficient.Although the polynomial approximation cannot theoretically guarantee complete acyclicity, the causal graphs discovered by Orion strictly satisfy the DAG property in subsequent experimental learning (see in Appendix A.5 for details).

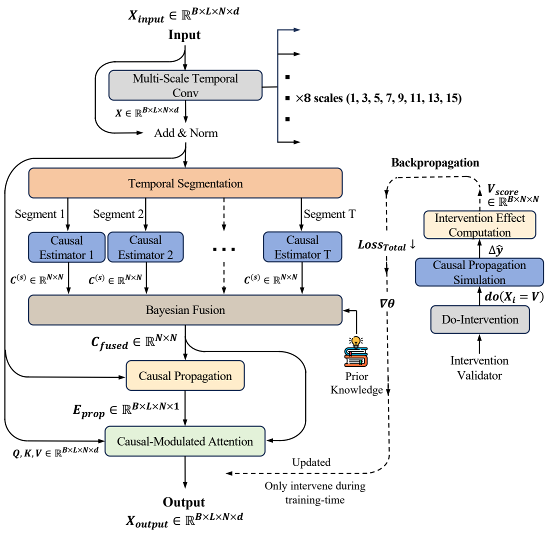

$X_{input} \in \mathbb{R}^{B\times L\times N\times d}$
Input
Multi-Scale Temporal Conv
×8 scales (1, 3, 5, 7, 9, 11, 13, 15)
$X \in \mathbb{R}^{B\times L\times N\times d}$
Add & Norm
Temporal Segmentation
Segment 1
Segment 2
Segment T
Causal Estimator 1
Causal Estimator 2
Causal Estimator T
$C^{(s)} \in \mathbb{R}^{N\times N}$
Bayesian Fusion
$C_{fused} \in \mathbb{R}^{N\times N}$
Prior Knowledge
Causal Propagation
$E_{prop} \in \mathbb{R}^{B\times L\times N\times 1}$
Causal-Modulated Attention
$Q, K, V \in \mathbb{R}^{B\times L\times N\times d}$
Output
$X_{output} \in \mathbb{R}^{B\times L\times N\times d}$
Backpropagation
$Loss_{Total} \downarrow$
$\nabla\theta$
Updated
Only intervene during training-time
$V_{score} \in \mathbb{R}^{B\times N\times N}$
Intervention Effect Computation
$\Delta\hat{y}$
Causal Propagation Simulation
$do(X_i = V)$
Do-Intervention
Intervention Validator

Figure 4: Architecture of TE-CausGAT. Left: forward pathway from multi-scale convolution to directed dependency estimation and dependency-modulated attention. Right: prediction-based validation (training only) filtering spurious edges.

3.2.2 *Intervention-inspired Consistency Validation.*

The core feature of TE-CausGAT is an intervention-inspired validation mechanism. Figure 5 illustrates how this mechanism tests whether learned edge weights predict intervention responses. When intervening on node i, the effect on downstream node j can be expressed under linear approximation as $\Delta\hat{y}_j \approx \mathbf{C}_{ij} \cdot \Delta\mathbf{X}_i + \sum_{k\neq i} \mathbf{C}_{kj} \cdot \Delta\mathbf{X}_k$. We implement this mechanism through four steps: randomly selecting K intervention nodes for unbiased discovery; generating intervention values $v = \tanh(g(\mathbf{c}))$ constrained to $[-1, 1]$; implementing the masking operation by setting node values to v_i and cutting incoming edges, then propagating through GRU to simulate dynamics; finally evaluating consistency between causal strength and observed effects through validity score, which is defined as:

$$V_{i\to j} = 1 - \left| C_{ij} - \frac{\Delta\hat{y}_j}{\max_k(\Delta\hat{y}_k) + \epsilon_{\text{stab}}} \right| \tag{5}$$

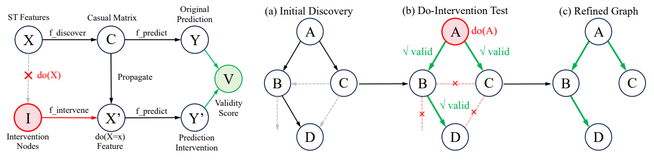


Figure 5: Prediction-based validation mechanism. (a) Masking a node's contribution generates prediction change; validity score measures consistency with learned edge weights. (b) Initial edges are tested and spurious correlations (red crosses) are filtered.

This formulation is based on the causal invariance assumption: if systematically removing X_a from $Y = f(X_a, X_b, X_c, X_d)$ yields significant change in Y, then X_a carries predictively relevant influence on Y. We use $\mathcal{L}_{validity}$ to promote high validity on causal edges, effectively filtering correlations that fail intervention tests. where the normalized empirical effect $\frac{\Delta\hat{y}_j}{\max_k(\Delta\hat{y}_k)+\epsilon_{\text{stab}}}$ quantitatively estimates the relative causal influence, the $\mathbf{C}_{ij} \in [0, 1]$ represents causal strength from node i to node j, $\Delta\hat{y}_j = |\hat{y}_j^{\text{post}} - \hat{y}_j^{\text{pre}}|$ is the prediction change for node j, and $\epsilon_{\text{stab}} = 10^{-8}$ prevents division by zero. The do-operation is an internal computationally simulated intervention used for validation rather than a physical intervention in the real world. A high validity score indicates consistency between the learned causal strength and the intervention effect, helping the model capture predictive dependencies, while a low score clearly reveals spurious correlations. We emphasize that this mechanism validates *predictive relevance* rather than interventional causality in the Pearl sense. It provides a robust training signal beyond pure correlation, thereby enabling Orion to filter spurious correlations and learn directed dependency structures that are consistent with their predictive effects—a desirable property we term *prediction-based consistency.*

3.2.3 *Causal-Modulated Attention.*

The validated causal matrix enhances multi-head attention. $\mathbf{Q}, \mathbf{K}, \mathbf{V}$ are projected from multi-scale temporal features. This mechanism operates through two complementary pathways that synergistically integrate causal knowledge: (1) dynamically scaling the value matrix based on causal propagation effects, where a three-layer MLP generates position-specific modulation strengths $m(\mathbf{X}, \mathbf{E})$ to amplify information on strong causal paths while suppressing weak paths; (2) directly enhancing attention scores using the fused causal matrix. The complete formulation is:

$$\begin{aligned}\text{Output} = \text{softmax}&\left(\frac{\mathbf{QK}^\top}{\sqrt{d_k}} + \alpha_{\text{gate}}\lambda_{\text{scale}}\mathbf{C}_{\text{fused}}\right) \\ &\cdot \left[\mathbf{V} \odot \left(1 + \alpha_{\text{gate}}\, m(\mathbf{X}, \mathbf{E}_{\text{prop}})\, \mathbf{E}_{\text{prop}}\right)\right].\end{aligned} \tag{6}$$

where α_{gate} is a learnable gate, λ_{scale} is a fixed scaling coefficient, $\mathbf{C}_{\text{fused}}$ represents the fused causal matrix, $\mathbf{E}_{prop}$ denotes propagation effects, and $m(\cdot)$ is a three-layer MLP modulation function. This design dynamically directs attention along validated causal paths and adaptively adjusts their strength.

3.3 Belt Block Architecture and Multi-Period Attention Fusion

The Belt Block serves as our fundamental unit, sequentially applying spatial causal attention, temporal attention, and feedforward transformation, augmented with residual connections. We instantiate three parallel Belt Block branches corresponding to Hour, Day, and Week periodicities, where each branch processes period-specific data selected through our LLM-enhanced retrieval mechanism. These branches independently capture distinct periodic patterns while maintaining computational efficiency via TE-CausGAT parameter sharing.

3.4 Three-Stage Progressive Training Strategy

Following curriculum learning [4], we adopt a three-stage progressive training strategy. It moves from feature learning to causal discovery to joint optimization, reducing local minima in complex end-to-end training (weight scheduling in Appendix B.3).

Stage 1: Feature Representation Learning (Correlation-Based Learning). Freeze all causal-related modules and focus on spatiotemporal feature representation learning:

$$\mathcal{L}_{\text{stage1}} = \mathcal{L}_{\text{pred}} = \text{MAE}(\text{Y}, \hat{\text{Y}}) \tag{7}$$

This ensures embedding layers, multi-scale convolutions, and attention mechanisms develop stable spatiotemporal representations.

Stage 2: Directed Structure Learning (Dependency Hypothesis Formation). Freeze embedding modules and concentrate learning capacity on TE-CausGAT components:

$$\begin{aligned}\mathcal{L}_{\text{stage2}} &= \mathcal{L}_{\text{pred}} + \lambda_{\text{causal}}\,\mathcal{L}_{\text{causal\_total}},\\ \mathcal{L}_{\text{causal\_total}} &= \sum_i w_i\,\mathcal{L}_i.\end{aligned} \tag{8}$$

Stage 3: End-to-End Optimization (Intervention-based Refinement). Unfreeze all parameters for joint optimization of feature learning and directed structure learning:

$$\mathcal{L}_{\text{stage3}} = \mathcal{L}_{pred} + \lambda_{causal}\cdot\mathcal{L}_{causal\_total} + \lambda_{reg}\cdot\mathcal{L}_{regularization} \tag{9}$$

4 Experiments

Datasets. We evaluate Orion on five spatiotemporal datasets: for traffic forecasting, we employ PEMS04, PEMS08 [6], and METR-LA for traffic flow prediction; we utilize CA-COVID and TX-COVID datasets [10] for hospitalization prediction (details in Appendix D.1).
Baselines. We compare against 13 methods: statistical models (HA, VAR), RNN-based models (DCRNN, AGCRN, DGCRN), CNN or attention-based models (STGCN, GWNet, ASTGCN, MTGNN [43], GMAN [47], STAEformer [26]), and causal-aware models (CaST, TESTAM).
Implementation Details. The adjacency matrix is constructed using a Gaussian kernel with σ set to the median of non-zero distances. Datasets are partitioned 6:2:2 for traffic (7:1:2 for METR-LA) and 8:1:1 for epidemic data. LLM retrieval is an optional module; to ensure fair comparison, main results (Table 2) use fixed-period retrieval, while LLM retrieval is evaluated independently (Table 3) to demonstrate semantic retrieval potential. Evaluation metrics are MAE, RMSE, MAPE (with threshold 5 for near-zeros), and L1 loss. We use Adam [19] for 100 epochs. All baselines and Orion adopt the same setup for fair comparison (see Appendix B for details).

4.1 Main Experimental Results

Traffic Forecasting Performance. Table 2 reports results on three benchmarks. On METR-LA, Orion attains MAE 2.66 (6.60% gain over STAEformer) with improvements across all horizons. On PEMS08, Orion delivers 8.65% average MAE improvement. On PEMS04, Orion achieves the strongest short-horizon result (8.83% MAE improvement at Horizon-3 over TESTAM) but degrades at longer horizons. We attribute this to PEMS04's limited temporal coverage (2 months) and its largest node count (307 nodes): fewer periodic variations make the multi-period architecture and directed dependency learning prone to overfitting. This contrasts with METR-LA (4 months, 207 nodes) and PEMS08 (2 months, 170 nodes), where sufficient data diversity or moderate network scale allows Orion to maintain advantages. Short-term gains stem from TE-CausGAT's ability to capture immediate spatial propagation patterns. Statistical models fail to capture nonlinear spatiotemporal dependencies (e.g., METR-LA MAPE > 13%), while recent methods (CaST, TESTAM) narrow the gap, Orion's directed dependency learning provides consistent short-horizon advantages and competitive long-horizon performance on 2 of 3 datasets.
Epidemic Forecasting Performance. Table 1 shows that Orion generalizes to epidemic prediction, achieving MAE 291.31 on CA-COVID (22.70% improvement over TESTAM) and 30.31 on TX-COVID (37.73% improvement). Unlike traffic networks where spatial relationships are physically constrained by road topology, epidemic spread involves socioeconomic factors and human mobility patterns that cannot be captured by geographical proximity alone. Here, Orion's prediction-based validation—which integrates prior spatial structure with learned dependencies—is particularly effective at filtering spurious correlations that arise from confounding factors such as shared policy changes or reporting delays.

Table 1: Epidemic forecasting performance (lower is better).

Dataset	DGCRN	STAEformer	CaST	TESTAM	**Orion**
CA	392.56	378.93	396.21	<u>376.84</u>	**291.31**
TX	58.42	49.36	61.15	<u>48.67</u>	**30.31**

Effect of LLM-Enhanced Retrieval. We evaluate LLM retrieval on 5-node subsets of PEMS08 and METR-LA, where the LLM selects the most similar historical patterns from the past week and month as tri-period input. Table 3 shows that, although overall improvements are moderate (METR-LA reduces average MAPE by 10.57%, PEMS08 by 3.44%), gains in key scenarios are substantial: METR-LA achieves 19.27% MAPE reduction at Horizon-3, and PEMS08 reaches 8.87% at Horizon-12 with a peak of 6.93% at Horizon-6. RMSE shows minimal change (−0.15%) due to sensitivity to extreme values, indicating enhancements primarily benefit typical patterns rather than outliers.

4.2 Ablation Study And Directed Dependency Validation

Ablation study. Table 4 shows that single-stage training causes the largest degradation (18.36% MAE), confirming progressive training is essential. Removing Bayesian fusion yields the largest single-component loss (19.79% MAE), underscoring integrating priors with

Table 2: Traffic forecasting performance comparison with 12-step prediction horizon. MAPE values in %. Red bold: best performance, underline: second best.

Dataset	Method	*Horizon 3*			*Horizon 6*			*Horizon 12*			*Average*		
		MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
METR-LA	HA	4.79	10.00	11.70	5.47	11.45	13.50	6.99	13.89	17.54	5.64	11.74	13.85
	VAR	4.42	7.80	13.00	5.41	9.13	12.70	6.52	10.11	15.80	5.23	9.09	13.26
	DCRNN	2.78	5.36	7.30	3.16	6.43	8.80	3.61	7.57	10.50	3.19	6.35	8.76
	AGCRN	2.88	5.56	7.70	3.24	6.57	9.01	3.69	7.55	10.46	3.24	6.40	8.83
	DGCRN	2.63	5.00	6.64	3.00	6.04	8.03	3.45	7.18	9.74	3.00	6.15	7.81
	STGCN	2.88	5.74	7.61	3.47	7.24	9.56	4.59	9.40	12.69	3.56	7.16	9.58
	GWNet	2.69	5.15	6.90	3.07	6.22	8.37	3.53	7.37	10.01	3.06	6.19	8.37
	ASTGCN	4.84	9.25	9.19	5.41	10.59	10.11	6.48	12.49	11.61	5.38	10.90	10.38
	MTGNN	2.68	5.17	6.86	3.04	6.16	8.17	3.48	7.22	9.85	2.98	5.79	7.92
	GMAN	2.80	5.55	7.41	3.12	6.49	8.73	3.44	7.35	10.07	3.07	6.39	8.57
	STAEformer	2.65	5.11	6.85	2.97	6.00	8.13	**3.34**	**7.02**	9.70	2.85	5.93	8.25
	CaST	2.71	5.26	7.14	3.08	6.29	8.52	3.42	7.21	10.03	3.01	6.04	8.44
	TESTAM	2.54	4.93	6.42	2.96	6.04	7.92	3.36	7.09	9.67	2.91	5.96	7.76
	Orion	**1.83**	**2.87**	**3.88**	**2.70**	**5.22**	**5.85**	3.73	7.49	**8.14**	**2.66**	**5.40**	**5.78**
PEMS04	HA	28.92	42.69	20.31	33.73	49.37	24.01	46.97	67.43	35.11	36.49	52.93	25.46
	VAR	21.94	34.30	16.42	23.72	36.58	18.02	26.76	40.28	20.94	23.51	36.39	17.85
	DCRNN	18.51	29.59	12.69	19.63	31.35	13.43	21.65	34.17	15.01	19.69	31.41	13.52
	AGCRN	18.52	29.79	12.31	19.45	31.45	12.82	20.64	33.31	13.74	19.36	31.28	12.81
	DGCRN	18.27	28.97	12.36	19.39	30.86	13.42	21.09	33.59	14.94	19.39	29.18	13.46
	STGCN	18.74	29.84	14.42	19.64	31.34	13.27	21.12	33.53	14.22	19.63	31.32	13.32
	GWNet	18.02	28.81	13.66	18.98	30.31	14.25	20.55	32.52	15.43	18.99	30.30	14.28
	ASTGCN	19.35	30.58	13.41	21.24	33.42	14.78	25.08	38.95	18.35	21.37	34.04	15.21
	MTGNN	18.65	30.13	13.32	19.48	32.02	14.08	20.96	34.66	14.96	19.50	32.00	14.04
	GMAN	18.26	29.34	12.65	18.80	30.84	13.24	**20.00**	**31.31**	**13.39**	18.82	30.92	13.20
	STAEformer	17.92	29.18	12.26	18.74	30.73	12.81	20.18	33.01	13.72	18.95	30.97	12.93
	CaST	18.36	29.51	12.48	19.21	31.18	13.15	20.72	33.76	14.28	19.13	**28.76**	12.76
	TESTAM	17.79	29.01	12.17	**18.62**	30.56	12.72	20.03	32.84	13.61	**18.81**	30.80	12.83
	Orion	**16.22**	**24.52**	**11.73**	18.86	**28.65**	**12.63**	24.53	36.74	15.20	19.04	29.31	**12.52**
PEMS08	HA	23.52	34.96	14.72	27.67	40.89	17.37	39.28	56.74	25.17	28.64	42.27	18.21
	VAR	19.52	29.73	12.54	22.25	33.30	14.23	26.17	38.97	17.32	22.07	31.02	14.04
	DCRNN	14.18	22.18	9.33	15.26	24.24	9.92	17.72	27.12	11.15	15.28	24.26	9.98
	AGCRN	14.52	22.86	9.35	15.67	24.99	10.35	17.50	27.92	11.73	15.66	24.98	10.18
	DGCRN	13.89	22.07	9.19	14.92	23.99	9.85	16.73	26.88	10.84	15.01	24.62	9.58
	STGCN	14.95	23.48	9.87	15.92	25.36	10.42	17.65	28.03	11.34	15.98	25.37	10.43
	GWNet	13.72	21.71	8.80	14.67	23.50	9.49	16.15	25.95	10.74	14.67	23.49	9.52
	ASTGCN	15.72	24.28	10.45	17.85	27.29	11.58	21.96	32.74	14.53	18.17	26.06	11.79
	MTGNN	14.28	22.53	10.54	15.23	24.39	10.52	16.78	26.94	10.88	15.29	24.40	10.68
	GMAN	13.80	22.88	9.41	14.62	24.12	9.57	15.72	26.47	10.56	14.81	24.19	9.69
	STAEformer	13.48	21.91	8.76	14.46	23.86	9.41	15.96	26.63	10.54	14.63	24.13	9.57
	CaST	13.64	21.74	8.79	15.83	25.33	9.86	17.31	29.01	10.88	15.59	25.36	9.84
	TESTAM	13.41	21.79	**8.71**	14.38	23.73	**9.34**	15.87	26.48	**10.45**	14.55	24.00	**9.50**
	Orion	**12.33**	**18.28**	9.83	**13.50**	**20.50**	10.79	**15.18**	**23.37**	11.42	**13.29**	**20.24**	10.55

Table 3: Effect of fine-tuned LLM-enhanced data retrieval (MAPE, %). Hardware details in Appendix B.2.

Horizon	Metric	*PEMS08*			*METR-LA*		
		w/o LLM	w/ LLM	Improv.	w/o LLM	w/ LLM	Improv.
3	MAE	18.77	**18.54**	1.22%	2.43	**2.26**	7.00%
	RMSE	32.54	**32.05**	1.50%	**5.21**	5.24	-0.71%
	MAPE	9.80	**9.64**	1.63%	8.20	**6.62**	19.27%
6	MAE	21.02	**20.64**	1.82%	3.46	**3.31**	4.37%
	RMSE	37.35	**36.82**	1.43%	8.38	**8.37**	0.10%
	MAPE	11.40	**10.61**	6.93%	12.41	**11.04**	11.04%
12	MAE	25.50	**25.10**	1.59%	4.93	**4.83**	2.03%
	RMSE	42.47	**42.32**	0.35%	**11.48**	11.49	-0.05%
	MAPE	13.76	**12.54**	8.87%	18.24	**17.35**	4.88%
Avg.	MAE	21.21	**20.86**	1.64%	3.47	**3.32**	4.26%
	RMSE	36.93	**36.46**	1.28%	**8.48**	8.49	-0.15%
	MAPE	11.06	**10.68**	3.44%	12.49	**11.17**	10.57%

learned dependencies. The TE-CausGAT ablation reveals a horizon-dependent pattern: the variant without TE-CausGAT shows marginal MAE reduction at Horizon-3 (−2.19%) but progressive degradation at Horizon-6 (+0.74%) and Horizon-12 (+2.63%). This reflects spatial propagation—short-horizon prediction is dominated by temporal autocorrelation, whereas longer horizons require modeling how perturbations propagate across the network. This holds for multi-period removal (+2.78% average, +8.17% at Horizon-12) and single-scale convolution (+8.05% average, +15.09% at Horizon-12), confirming spatial and temporal components grow more important with horizon. Intervention validation removal increases MAE by 3.82%. Though moderate, this component serves a structural role: it ensures learned edges are predictively reliable, as evidenced by DREAM-3 AUC gain (0.6295 vs. 0.5915 best baseline). The MAE-MAPE divergence reflects metric sensitivity: MAPE is disproportionately affected by near-zero values, while MAE captures accuracy across all magnitudes.

Table 4: Ablation study on PEMS-08. Parentheses show percentage change relative to full Orion.

Component	*Horizon 3*			*Horizon 6*			*Horizon 12*			*Average*		
	MAE	RMSE	MAPE(%)	MAE	RMSE	MAPE(%)	MAE	RMSE	MAPE(%)	MAE	RMSE	MAPE(%)
Part 1: Architecture Components												
Single Period	12.26(-0.57%)	18.83(+3.01%)	8.37(-14.88%)	13.74(+1.78%)	21.65(+5.61%)	8.90(-17.52%)	16.42(+8.17%)	25.97(+11.11%)	10.21(-10.60%)	13.66(+2.78%)	21.36(+5.53%)	8.94(-15.26%)
w/o TE-CausGAT	12.06(-2.19%)	**18.14**(-0.77%)	8.93(-9.15%)	13.60(+0.74%)	20.80(+1.46%)	10.04(-6.95%)	15.58(+2.63%)	23.96(+2.52%)	11.18(-2.10%)	13.37(+0.60%)	20.37(+0.64%)	9.85(-6.64%)
w/o Multi-Period Fusion	**12.03**(-2.43%)	18.39(+0.60%)	**7.30**(-25.79%)	14.02(+3.85%)	21.71(+5.90%)	**8.24**(-23.63%)	16.88(+11.20%)	25.62(+9.62%)	**10.07**(-11.82%)	13.65(+2.71%)	20.87(+3.11%)	**8.20**(-22.27%)
Single Stage Training	12.98(+5.27%)	19.25(+5.31%)	9.46(-3.76%)	15.85(+17.41%)	23.63(+15.27%)	11.19(+3.71%)	20.17(+32.87%)	29.55(+26.42%)	14.15(+23.91%)	15.73(+18.36%)	23.19(+14.57%)	11.46(+8.63%)
Full Orion	12.33	**18.28**	9.83	**13.50**	**20.50**	10.79	**15.18**	**23.37**	11.42	**13.29**	**20.24**	10.55
Part 2: TE-CausGAT Components												
Single Scale Conv	12.64(+2.51%)	19.35(+5.85%)	9.49(-3.46%)	14.44(+6.96%)	22.37(+9.12%)	10.82(+0.28%)	17.47(+15.09%)	26.96(+15.35%)	12.65(+10.77%)	14.36(+8.05%)	22.09(+9.17%)	10.75(+1.90%)
w/o Intervention	**12.11**(-1.81%)	18.72(+2.39%)	**8.15**(-17.09%)	14.05(+4.05%)	21.61(+5.42%)	**10.10**(-6.40%)	16.44(+8.33%)	25.17(+7.72%)	**11.48**(+0.53%)	13.80(+3.82%)	21.32(+5.33%)	9.93(-5.88%)
w/o Bayesian Fusion	13.22(+7.22%)	20.02(+9.52%)	8.98(-8.64%)	16.05(+18.89%)	24.67(+20.34%)	10.02(-7.14%)	20.66(+36.10%)	31.50(+34.79%)	12.53(+9.72%)	15.92(+19.79%)	24.38(+20.51%)	**9.91**(-6.07%)
Full Orion	12.33	**18.28**	9.83	**13.50**	**20.50**	10.79	**15.18**	**23.37**	11.48	**13.29**	**20.24**	10.55

Table 5: Directed dependency discovery on DREAM-3 (AUC; higher is better). Ground-truth structure used for evaluation.

Method	PCMCI	NGC	eSRU	LCCM	NGM	CUTS	**Orion**
AUC	0.5517	0.5579	0.5587	0.5046	0.5477	0.5915	**0.6295**

Directed dependency validation on DREAM-3. To quantitatively evaluate Orion’s capability in discovering directed dependencies, we conduct experiments on the DREAM-3 gene regulatory network dataset [29], which provides ground-truth causal structure for benchmarking. As shown in Table 5, Orion achieves an AUC of 0.6295, outperforming the strongest baseline CUTS (0.5915) by 6.42%. Traditional approaches such as PCMCI (0.5517) and NGC (0.5579) are constrained by linear assumptions and fixed temporal windows, while neural methods such as eSRU (0.5587) and NGM (0.5477) lack validation mechanisms, often mistaking correlations for directed influence, a gap that Orion’s intervention-based testing directly addresses. The results confirm that our learned structures closely match the ground-truth graph, effectively identifying predictively relevant dependencies that largely align with true causal relationships.

4.3 Case Study: Real-World Application on METR-LA

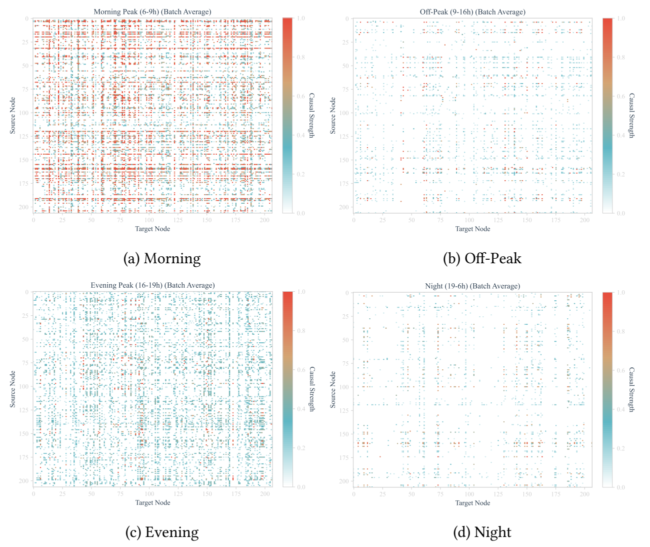
We validate Orion on METR-LA, whose sensors span major highways (I-405, I-10, US-101). All analyses use the directed dependency matrix averaged over test batches.

Spatiotemporal Evolution of Directed Dependencies. A central advantage of Orion is that its learned directed dependency matrices are time-varying, enabling interpretable analysis of how inter-node influence structures evolve with traffic demand. Figure 6 visualizes these matrices as heatmaps for four representative periods, where edge counts range from 12,427 during the morning peak (6–9 AM) to only 2,078 at night (7 PM–6 AM). Figure 7 and Table 6 further detail the network topology and key node roles at each period. During the morning peak, the network exhibits a “one-to-many” tree-like topology: Node 4 (I-10/I-405 interchange) alone has an out-degree of 20, with influence extending up to 3.5 miles along

Table 6: Key source/target nodes across four periods. Infl.: source out-degree. Src Cnt: target in-degree. Dep. Dist.: top-10 edge count by strength (S>0.7, M: 0.4–0.7, W<0.4).

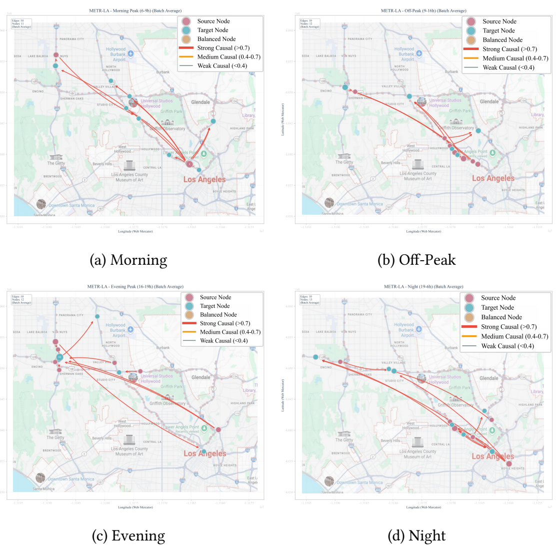
Time Period	**Src** ***Node***	**Infl.**	**Tgt** ***Node***	**Src Cnt**	**Causal Dist.**
Morning (6-9h)	Node 4	20	Node 79	2	Strong: 10
	Node 3	3	Node 14	2	Medium: 0
	Node 8	2	Node 22	1	Weak: 0 (Total: 10)
Off-Peak (9-16h)	Node 159	1	Node 139	2	Strong: 6
	Node 160	2	Node 131	3	Medium: 0
	Node 4	2	Node 61	2	Weak: 4 (Total: 10)
Evening (16-19h)	Node 84	1	Node 84	5	Strong: 8
	Node 14	2	Node 91	1	Medium: 0
	Node 29	2	Node 77	1	Weak: 2 (Total: 10)
Night (19-6h)	Node 174	3	Node 73	2	Strong: 0
	Node 160	3	Node 150	2	Medium: 0
	Node 56	1	Node 91	2	Weak: 10 (Total: 10)

northbound I-405 and eastbound US-101; 65% of edges connect adjacent sensors while 35% are long-range edges corresponding to detour routes and ramp cascades. The evening peak reverses into a “many-to-one” convergence pattern (9,123 edges), where Node 84 receives strong influences from five upstream nodes (9, 14, 29, 88, 176), forming a bottleneck on I-405. During off-peak hours, dependency density drops by 76.9% and long-range edges decline to 5%, yielding a sparse multi-center structure. At night, only two independent influence centers remain, Node 174 and Node 160, with dependency strengths up to 0.9988. These evolving patterns are consistent with known Los Angeles commuting dynamics and demonstrate that Orion captures physically meaningful, time-varying directed influence structures rather than static symmetric correlations. Crucially, because these structures identify *where* influence originates and *how* it propagates, they directly inform actionable traffic management: Node 4’s high morning out-degree motivates upstream ramp metering triggered below 40 mph; Node 84’s evening convergence from five sources suggests diverting 30% of traffic when ≥3 sources congest; the nighttime dual-center pattern supports dedicated freight lanes at Node 174 and extended green lights at Node 160; and the directed chain between Nodes 14–91 reveals cascading risk addressable by intermediate speed limits.



(a) Morning (b) Off-Peak (c) Evening (d) Night

Figure 6: Directed dependency matrices on METR-LA across four time periods. Each cell (i, j) denotes dependency strength from node i to node j; red = strong, blue = weak. Note the progressive sparsification from (a) to (d).



(a) Morning (b) Off-Peak (c) Evening (d) Night

Figure 7: Geographic projection of directed dependencies. Node color: blue (source), red (target), green (balanced). Edge color: red (>0.7), orange (0.4–0.7), yellow (<0.4). Compare "one-to-many" in (a) with "many-to-one" in (c).

5 Related Work

From statistical models to deep learning. Spatiotemporal forecasting has progressed from statistical methods to neural architectures. Early approaches such as ARIMA [5] and VAR [18] captured linear dependencies. Deep learning advanced the field: ConvLSTM [36] combined CNNs and LSTMs for spatiotemporal feature extraction, Transformers [41] captured global temporal dependencies, and iTransformer [28] demonstrated that inverted variate-dimension attention yields strong performance. Large language models have been repurposed for temporal reasoning: Time-LLM [16] reprograms frozen LLMs via prompt-based input transformation, and Time-FFM [27] leveraged LLMs for temporal understanding. Graph neural networks learned node representations through message passing but remained limited to single-dimensional optimization. A shared limitation is these methods learn symmetric correlations without modeling dependency directionality.

Spatiotemporal coupling architectures. To jointly model spatial and temporal dynamics, STGCN [46] introduced graph-temporal convolution, DCRNN combined diffusion convolution with GRU, Graph-WaveNet introduced adaptive adjacency learning, ASTGCN [12] employed spatiotemporal attention, and DGCRN attempted dynamic graph learning. D2STGNN [35] decoupled diffusion and inherent traffic signals, DSTAGNN [20] constructed dynamic spatial-temporal aware graphs, and PDFormer [15] modeled propagation delay effects. In the epidemic domain, MSGNN [32] proposed multi-scale graph convolutions to capture cross-granularity spreading patterns. Despite these advances, three limitations persist: (1) fixed or slowly adapting adjacency matrices cannot represent time-varying connectivity; (2) symmetry assumptions fail to distinguish influence sources from targets; (3) single-scale temporal modeling struggles to capture multi-period patterns.

Causal-inspired spatiotemporal methods. Recognizing these limitations, researchers integrated causal reasoning into forecasting. Assaad et al. [1] categorize causal discovery into constraint-based, score-based, and Granger-based families. PCMCI [33] relied on conditional independence testing, NGC [39] and eSRU [17] learned Granger causality [11], and LCCM [9] explored causal connections. DYNOTEARS [31] extended structure learning to time-series under acyclicity constraints, inspiring our DAG regularization. Recently, CaST improves forecasting through learned causal graphs, TESTAM uses spatiotemporal attention for causal discovery, and in the epidemic domain, EpiGNN [49] incorporates epidemiological knowledge into GNNs for heterogeneity-aware forecasting, while Han et al. [13] unify physics- and data-driven modeling via a causal STGNN. However, two limitations remain: static graph assumptions—e.g., CaST uses a single matrix; and lack of validation—learned structures may capture spurious correlations rather than directed dependencies. Leveraging Granger causality and neural causal discovery, we use prediction-based validation to extract predictive edges, avoiding assumptions of structural causal models.

6 Conclusion

We propose **Orion**, a spatiotemporal forecasting framework that learns time-varying directed dependency structures via Granger-inspired estimation and validates them through prediction-based consistency testing, a mechanism that provides a training signal beyond correlation by testing whether learned edge weights predict downstream sensitivity. Combined with multi-scale temporal modeling (Belt Block) and progressive training, Orion achieves state-of-the-art results across five traffic and epidemic benchmarks. DREAM-3 and METR-LA analyses confirm that learned structures capture relevant and physically interpretable dependencies.

References

[1] Charles K Assaad, Emilie Devijver, and Eric Gaussier. 2022. Survey and evaluation of causal discovery methods for time series. *Journal of Artificial Intelligence Research* 73 (2022), 767–819.

[2] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450* (2016).

[3] Lei Bai, Lina Yao, Can Li, Xianzhi Wang, and Can Wang. 2020. Adaptive graph convolutional recurrent network for traffic forecasting. *Advances in neural information processing systems* 33 (2020), 17804–17815.

[4] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*. 41–48.

[5] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. 2015. *Time series analysis: forecasting and control*. John Wiley & Sons.

[6] Chao Chen. 2002. *Freeway performance measurement system (PeMS)*. University of California, Berkeley.

[7] Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. 2016. Training deep nets with sublinear memory cost. *arXiv preprint arXiv:1604.06174* (2016).

[8] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).

[9] Edward De Brouwer, Adam Arany, Jaak Simm, and Yves Moreau. 2020. Latent convergent cross mapping. In *International Conference on Learning Representations*.

[10] Ensheng Dong, Hongru Du, and Lauren Gardner. 2020. An interactive web-based dashboard to track COVID-19 in real time. *The Lancet infectious diseases* 20, 5 (2020), 533–534.

[11] Clive WJ Granger. 1969. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica: journal of the Econometric Society* (1969), 424–438.

[12] Shengnan Guo, Youfang Lin, Ning Feng, Chao Song, and Huaiyu Wan. 2019. Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 922–929.

[13] Shuai Han, Lukas Stelz, Thomas R Sokolowski, Kai Zhou, and Horst Stöcker. 2025. Unifying Physics-and Data-Driven Modeling via Novel Causal Spatiotemporal Graph Neural Network for Interpretable Epidemic Forecasting. *arXiv preprint arXiv:2504.05140* (2025).

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[15] Jiawei Jiang, Chengkai Han, Wayne Xin Zhao, and Jingyuan Wang. 2023. Pdformer: Propagation delay-aware dynamic long-range transformer for traffic flow prediction. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 37. 4365–4373.

[16] Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, et al. 2023. Time-llm: Time series forecasting by reprogramming large language models. *arXiv preprint arXiv:2310.01728* (2023).

[17] Saurabh Khanna and Vincent YF Tan. 2019. Economy statistical recurrent units for inferring nonlinear granger causality. *arXiv preprint arXiv:1911.09879* (2019).

[18] Lutz Kilian. 2006. NEW INTRODUCTION TO MULTIPLE TIME SERIES ANALYSIS, by Helmut Lütkepohl, Springer, 2005. *Econometric theory* 22, 5 (2006), 961–967.

[19] Diederik P Kingma. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[20] Shiyong Lan, Yitong Ma, Weikang Huang, Wenwu Wang, Hongyu Yang, and Pyang Li. 2022. Dstagnn: Dynamic spatial-temporal aware graph neural network for traffic flow forecasting. In *International conference on machine learning*. PMLR, 11906–11917.

[21] Colin Lea, Rene Vidal, Austin Reiter, and Gregory D Hager. 2016. Temporal convolutional networks: A unified approach to action segmentation. In *European conference on computer vision*. Springer, 47–54.

[22] Hyunwook Lee and Sungahn Ko. 2024. TESTAM: A time-enhanced spatio-temporal attention model with mixture of experts. *arXiv preprint arXiv:2403.02600* (2024).

[23] Fuxian Li, Jie Feng, Huan Yan, Guangyin Jin, Fan Yang, Funing Sun, Depeng Jin, and Yong Li. 2023. Dynamic graph convolutional recurrent network for traffic prediction: Benchmark and solution. *ACM Transactions on Knowledge Discovery from Data* 17, 1 (2023), 1–21.

[24] Qimai Li, Zhichao Han, and Xiao-Ming Wu. 2018. Deeper insights into graph convolutional networks for semi-supervised learning. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 32.

[25] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. 2017. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. *arXiv preprint arXiv:1707.01926* (2017).

[26] Hangchen Liu, Zheng Dong, Renhe Jiang, Jiewen Deng, Jinliang Deng, Quanjun Chen, and Xuan Song. 2023. Spatio-temporal adaptive embedding makes vanilla transformer sota for traffic forecasting. In *Proceedings of the 32nd ACM international conference on information and knowledge management*. 4125–4129.

[27] Qingxiang Liu, Xu Liu, Chenghao Liu, Qingsong Wen, and Yuxuan Liang. 2024. Time-ffm: Towards lm-empowered federated foundation model for time series forecasting. *Advances in Neural Information Processing Systems* 37 (2024), 94512–94538.

[28] Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. 2023. itransformer: Inverted transformers are effective for time series forecasting. *arXiv preprint arXiv:2310.06625* (2023).

[29] Aviv Madar, Alex Greenfield, Eric Vanden-Eijnden, and Richard Bonneau. 2010. DREAM3: network inference using dynamic context likelihood of relatedness and the inferelator. *PloS one* 5, 3 (2010), e9803.

[30] Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, et al. 2017. Mixed precision training. *arXiv preprint arXiv:1710.03740* (2017).

[31] Roxana Pamfil, Nisara Sriwattanaworachai, Shaan Desai, Philip Pilgerstorfer, Konstantinos Georgatzis, Paul Beaumont, and Bryon Aragam. 2020. Dynotears: Structure learning from time-series data. In *International Conference on Artificial Intelligence and Statistics*. Pmlr, 1595–1605.

[32] Mingjie Qiu, Zhiyi Tan, and Bing-Kun Bao. 2024. MSGNN: Multi-scale Spatio-temporal Graph Neural Network for epidemic forecasting. *Data Mining and Knowledge Discovery* 38, 4 (2024), 2348–2376.

[33] Jakob Runge, Peer Nowack, Marlene Kretschmer, Seth Flaxman, and Dino Sejdinovic. 2019. Detecting and quantifying causal associations in large nonlinear time series datasets. *Science advances* 5, 11 (2019), eaau4996.

[34] Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing* 45, 11 (1997), 2673–2681.

[35] Zezhi Shao, Zhao Zhang, Wei Wei, Fei Wang, Yongjun Xu, Xin Cao, and Christian S Jensen. 2022. Decoupled dynamic spatial-temporal graph neural network for traffic forecasting. *arXiv preprint arXiv:2206.09112* (2022).

[36] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. 2015. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. *Advances in neural information processing systems* 28 (2015).

[37] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research* 15, 1 (2014), 1929–1958.

[38] Jean Steinier, Yves Termonia, and Jules Deltour. 1972. Smoothing and differentiation of data by simplified least square procedure. *Analytical chemistry* 44, 11 (1972), 1906–1909.

[39] Alex Tank, Ian Covert, Nicholas Foti, Ali Shojaie, and Emily B Fox. 2021. Neural granger causality. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44, 8 (2021), 4267–4279.

[40] John Wilder Tukey et al. 1977. *Exploratory data analysis*. Vol. 2. Springer.

[41] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[42] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903* (2017).

[43] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, Xiaojun Chang, and Chengqi Zhang. 2020. Connecting the dots: Multivariate time series forecasting with graph neural networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*. 753–763.

[44] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. 2019. Graph wavenet for deep spatial-temporal graph modeling. *arXiv preprint arXiv:1906.00121* (2019).

[45] Yutong Xia, Yuxuan Liang, Haomin Wen, Xu Liu, Kun Wang, Zhengyang Zhou, and Roger Zimmermann. 2023. Deciphering spatio-temporal graph forecasting: A causal lens and treatment. *Advances in Neural Information Processing Systems* 36 (2023), 37068–37088.

[46] Bing Yu, Haoteng Yin, and Zhanxing Zhu. 2017. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. *arXiv preprint arXiv:1709.04875* (2017).

[47] Chuanpan Zheng, Xiaoliang Fan, Cheng Wang, and Jianzhong Qi. 2020. Gman: A graph multi-attention network for traffic prediction. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 34. 1234–1241.

[48] Xun Zheng, Bryon Aragam, Pradeep K Ravikumar, and Eric P Xing. 2018. Dags with no tears: Continuous optimization for structure learning. *Advances in neural information processing systems* 31 (2018).

[49] Yufan Zheng, Wei Jiang, Tong Chen, Alexander Zhou, Nguyen Quoc Viet Hung, Choujun Zhan, and Hongzhi Yin. 2024. Epidemiology-informed Graph Neural Network for Heterogeneity-aware Epidemic Forecasting. *arXiv preprint arXiv:2411.17372* (2024).

A Theoretical Foundations and Proofs

A.1 Granger Causality Foundation of TE-CausGAT

We provide the theoretical foundation for TE-CausGAT based on the Granger causality framework [11], which is the appropriate formalism for learning directed predictive dependencies from observational time series data.

Definition (Granger Causality). A time series X is said to Granger-cause another time series Y if, given all other relevant information, past values of X provide statistically significant information about future values of Y. Formally:

$$\sigma^2(Y_t|Y_{t-1},\ldots,X_{t-1},\ldots) < \sigma^2(Y_t|Y_{t-1},\ldots) \tag{10}$$

where $\sigma^2(\cdot|\cdot)$ denotes the conditional variance.

Connection to TE-CausGAT. Our learned directed dependency matrix $\mathbf{C}^{(s)} \in [0,1]^{N\times N}$ captures Granger-causal relationships in a neural network framework. Specifically, $\mathbf{C}^{(s)}_{ij} > 0$ indicates that node i's historical information improves the prediction of node j in temporal segment s. This interpretation aligns with the extensive literature on neural Granger causality [17, 39].

Prediction-Based Validation Principle. Our validation mechanism tests whether the learned dependency strengths are consistent with their predictive effects. The key insight is based on the *predictive relevance criterion*: if removing node i's contribution significantly changes the prediction of node j, then node i carries predictively relevant information for node j. This is operationalized through our validity score (Equation 5), where $\Delta\hat{y}_j$ measures the prediction change when node i's contribution is masked.

Remark on Scope. We emphasize that Granger causality captures *predictive* rather than *interventional* causality. Our method identifies which nodes carry predictive information for other nodes, which is sufficient for forecasting applications. We do not claim to recover the underlying structural causal model or to predict the effects of real-world interventions.

A.2 Consistency Between Learned Dependencies and Predictive Effects

We analyze the conditions under which TE-CausGAT learns dependency structures that are consistent with their predictive effects.

Proposition (Prediction Consistency). Under the assumption that the model has sufficient capacity to capture the true predictive relationships, the learned dependency matrix $\mathbf{C}$ converges to values where high-weight edges correspond to high predictive importance.

Justification. Consider the optimization objective in Stage 2 and Stage 3 training:

$$\mathcal{L} = \mathcal{L}_{pred} + \lambda_{validity}\mathcal{L}_{validity} + \ldots \tag{11}$$

The validity loss $\mathcal{L}_{validity}$ penalizes inconsistency between $\mathbf{C}_{ij}$ and the normalized prediction change $\Delta\hat{y}_j/\max_k(\Delta\hat{y}_k)$. Minimizing this loss drives the model toward configurations where:

- Edges with high learned weights $\mathbf{C}_{ij}$ correspond to node pairs where masking node i significantly affects the prediction of node j;
- Edges with low learned weights correspond to node pairs with minimal predictive coupling.

This self-consistency mechanism helps filter out spurious correlations that may arise from confounding factors or noise. While we cannot guarantee identification of true causal relationships in the interventional sense, this mechanism ensures that the learned structure reflects genuine predictive dependencies that are useful for forecasting.

Empirical Validation. The effectiveness of this consistency mechanism is validated through:

- **Shuffling Test (Section A.4)**: When node-effect correspondences are randomly shuffled, both validity scores and prediction performance degrade significantly, confirming that the model learns genuine node-specific dependencies rather than trivial patterns.
- **DREAM-3 Benchmark (Table 5)**: On a dataset with known ground-truth regulatory relationships, Orion achieves the highest AUC among all baselines, demonstrating that learned dependencies align with true directed relationships.

A.3 Practical Robustness of the Validation Mechanism

Our prediction-based validation mechanism is designed with multiple robustness enhancements to ensure reliable learning of directed dependencies on complex real-world data.

While our theoretical derivation assumes local linearization, our implementation fully captures nonlinear dynamics through multi-scale temporal convolutions (8 different kernel sizes) and GRU propagation mechanisms. The multi-scale temporal convolution module in our code employs multiple time scales from 1 to 15, ensuring simultaneous modeling of short-term perturbations and long-term trends. The normalization term $\max_k(\Delta\hat{y}_k)$ combined with the stability term $\epsilon_{\text{stab}} = 10^{-8}$ not only prevents numerical instability but more importantly provides relative scale calibration of node responses, making causal strengths comparable across nodes of different magnitudes.

Our implementation through Bayesian fusion (Equation 3) intelligently combines data-driven directed dependency learning with domain prior knowledge. When data evidence is sufficient, the model relies on learned directed dependencies; when uncertainty exists, prior knowledge provides guidance. This adaptive mechanism ensures our method remains robust under various data conditions.

The `InterventionValidator` class in our code implements an intelligent intervention node rotation mechanism, ensuring all nodes are systematically tested. By constraining intervention values within $[-1,1]$ through $v = \tanh(g(\mathbf{c}))$, we maintain numerical stability while adequately exploring the intervention space. The three-stage progressive training strategy (feature learning $\rightarrow$ directed structure learning $\rightarrow$ end-to-end optimization) further ensures reliable learning of directed structures.

Our method achieves 6.60% MAE improvement on METR-LA, 8.65% improvement on PEMS-08, and remarkable 22.70% and 37.73% improvements on CA-COVID and TX-COVID respectively. These results demonstrate the advantage of our Granger-inspired directed dependency learning approach over pure correlation-based methods, by identifying predictively relevant directed dependencies rather than superficial correlations, we achieve more accurate and interpretable predictions.

A.4 Safeguards Against Confirmation Bias

To prevent confirmation bias, our validation mechanism includes three layers of independence guarantees: (1) intervention nodes are randomly selected in each batch, independent of the current dependency matrix; (2) intervention effects propagate through an independently trained GRU, whose dynamics do not directly depend on the learning of C; (3) the validity score uses normalized relative effects, requiring the model to predict the correct ranking of causal strengths rather than absolute values. Stress test experiments show that after shuffling node-effect correspondences, the model's validity in late training significantly decreases (average 1.4%, $p < 0.05$) and prediction MAE degrades by 3.84%, validating that the mechanism achieves high validity scores through learning predictively relevant directed dependencies rather than trivial alignment.

To verify that the perturbation mechanism does not exhibit confirmation bias, we designed a Shuffling Test to examine whether the model truly learns the dependency correspondences between nodes. The test protocol is as follows: at each test epoch, we perform the standard intervention process on the current model to obtain the intervention effects $\Delta\hat{y}_j$ for each node, then randomly shuffle these effect values across different nodes and recalculate the validity score. If the model relies on genuine directed dependencies, validity should significantly decrease after shuffling; if trivial alignment exists, there should be no significant difference before and after shuffling. We conduct tests every 5 training epochs on the PEMS-08 dataset, performing 30 random shuffles each time to calculate statistical significance (using paired t-test with significance level $p < 0.05$).

From Table 7, a clear two-stage pattern is evident: during early training (Epoch 10), validity slightly increases after shuffling (negative drop magnitude) and is statistically insignificant, indicating that causal relationships have not been sufficiently learned at this stage. Starting from Epoch 15, validity consistently decreases after shuffling with statistical significance, proving that the model has learned genuine node-effect dependency pairings. Notably, the Week and Day cycles maintain significant validity drops (0.98%-2.47%) throughout the training process, while the Hour cycle's drop approaches zero by Epoch 30, reflecting the convergence characteristics of directed dependency learning at different time scales.

Table 7: Shuffling Test Results Throughout Training Process (PEMS-08). Drop magnitude = (Original - Shuffled)/Original.

Epoch	Period	Original	Shuffled	Drop
15	Hour	0.1432	0.1420	0.84%
	Week	0.1467	0.1434	2.28%
	Day	0.1493	0.1465	1.85%
25	Hour	0.1414	0.1409	0.33%
	Week	0.1470	0.1453	1.16%
	Day	0.1504	0.1480	1.58%

From Table 8, after shuffling causal correspondences, the model trained on the test set shows an increase in MAE from 13.29 to 13.80 (3.84%) and RMSE from 20.24 to 21.33 (5.39%). This performance degradation directly shows that the original model's excellent prediction performance depends on correctly learned directed dependencies, rather than trivial alignment. Notably, MAPE slightly decreases (-4.36%), possibly because the shuffled model has reduced prediction bias at extremely low traffic values (where denominators close to zero cause MAPE instability), but significant degradation in overall MAE and RMSE still indicates the model's dependence on correctly learned directed structures.

Table 8: Impact of Shuffling Test on Prediction Performance

Model Configuration	MAE	RMSE	MAPE(%)	Performance Change
Original Model (No Shuffling)	**13.29**	**20.24**	10.55	Baseline
Shuffled Model	13.80	21.33	**10.09**	MAE↑3.84%, RMSE↑5.39%

Combining evidence from both validity decrease and prediction performance degradation, our stress test validates the effectiveness of the perturbation mechanism: the model must learn genuine inter-node directed dependencies to simultaneously achieve high validity scores and excellent prediction performance, rather than through self-confirming trivial alignment.

A.5 DAG Constraint Effectiveness Analysis

We adopt the DAG constraint in polynomial approximation form:

$$\mathcal{L}_{\text{DAG}} = \lambda_2 \cdot \text{tr}(\hat{C}^2) + \lambda_3 \cdot \text{tr}(\hat{C}^3) \tag{12}$$

where $\hat{C} = C_{\text{fused}}/|C_{\text{fused}}|_\infty$ is normalized, with $\lambda_2 = 0.1, \lambda_3 = 0.01$. The main reasons for choosing this approximation are computational efficiency and numerical stability: the complete matrix exponential constraint $\text{tr}(e^C) - N = 0$ has computational complexity of $O(N^3)$ for large-scale networks (METR-LA has 207 nodes, PEMS-04 has 307 nodes) and is prone to gradient explosion or vanishing during backpropagation. In contrast, the polynomial approximation only requires computing the trace of C^2 and C^3, with reduced complexity and smaller constants, making it numerically more stable. Theoretically, $\text{tr}(C^k)$ computes the number of cycles of length k in the graph. By penalizing $\text{tr}(C^2)$ and $\text{tr}(C^3)$, we suppress the most common 2-cycles (bidirectional edges) and 3-cycles, which in practice can effectively constrain the acyclicity of the graph. Although this cannot theoretically guarantee complete acyclicity (as cycles of length ≥4 may exist), we observe in our experiments that this constraint is sufficiently effective.

To quantitatively verify the actual effectiveness of the DAG constraint, we use topological sorting algorithms to detect cycles in the learned directed dependency graph on the PEMS-08 dataset (170 nodes). Under the condition of setting threshold=0.1 (used to binarize the continuous dependency strength matrix), Table 9 presents the complete cycle statistics:

The results show that at threshold 0.1, the learned directed dependency graph completely eliminates all cycles of any length (including 2-cycles, 3-cycles, and 4-cycles and longer), strictly satisfying the DAG property. This verifies the sufficiency of the polynomial approximation $\lambda_2 \cdot \text{tr}(C^2) + \lambda_3 \cdot \text{tr}(C^3)$ in practical applications, although theoretically it cannot completely guarantee the absence of long cycles, through appropriate weight settings ($\lambda_2 = 0.1 > \lambda_3 = 0.01$,

Table 9: Cycle statistics of learned directed dependency graph (PEMS-08, threshold=0.1).

Cycle Type	Count	Theoretical Max	Pct.
2-cycles (bidirectional)	**0**	14,365 ($\binom{170}{2}$)	**0.00%**
3-cycles	**0**	804,440 ($\binom{170}{3}$)	**0.00%**
4-cycles and longer	**0**	—	**0.00%**

prioritizing the penalization of short cycles), it can effectively eliminate all cycles in practice.

B Detailed Implementation and Hyperparameters

This section provides comprehensive documentation of the Orion framework's complete implementation details, including all experimental hyperparameter configurations, K-means node selection strategies, and three-stage training weight distributions. Our experiments encompass four components: main experiments, LLM-enhanced retrieval comparisons, ablation studies, and causal discovery validation. To ensure experimental reproducibility and repeatability, all experiments set random seeds to 42, including Python, NumPy, PyTorch random generators, and CUDA deterministic algorithm modes, thereby guaranteeing consistent experimental results under identical hardware environments.

B.1 Complete Hyperparameter Settings

B.1.1 Main Experiment Configuration. All datasets share consistent configurations: prediction and input lengths of 12 time steps ($target\_len = source\_len = 12$), segmentation into four periods—6-9 (morning peak), 9-16 (daytime), 16-19 (evening peak), 19-6 (nighttime)—with dropout rate 0.1 [37], weight decay 1e-4, learning rate decay factor 0.8 every 20 epochs. The three-stage training strategy consists: Stage 1: 10 epochs of feature learning ($\lambda_{causal} = 0$), Stage 2: 10 epochs of directed structure learning ($\lambda_{causal} = 0.05$, freezing *embedding_process*), Stage 3: 80 epochs of end-to-end optimization ($\lambda_{causal} = 0.03$). GRU propagation mechanism enabled ($use\_gru\_propagation = True$) to capture temporal dynamics effectively, experiments conducted on specified environments (NVIDIA A100-40GB for traffic datasets, RTX 2080Ti for epidemic datasets), random seeds fixed at 42 ensuring full reproducibility.

We adopt differentiated hyperparameter configurations tailored to dataset characteristics. COVID datasets, with limited sample sizes (86 training samples and 30 test samples), employ smaller model capacity ($d\_model = 32$) and single Belt Block layer to prevent overfitting. Learning rates are adjusted according to network scale and convergence characteristics, with PEMS04 using the smallest rate (0.0001) due to its largest network (307 nodes) to avoid gradient explosion. COVID data intervention frequency is set to 2 (every 2 batches), higher than traffic datasets' 10, to enhance directed dependency learning in small-sample scenarios. Table 10 presents hyperparameter configurations across all five datasets.

Table 10: Differentiated hyperparameter configurations for main experiments

Parameter	*PEMS08*	*PEMS04*	*METR-LA*	*CA-COVID*	*TX-COVID*
Data Configuration					
num_of_vertices	170	307	207	55	251
in_channels	3	3	1	3	3
train_ratio/val_ratio	0.6/0.2	0.6/0.2	0.7/0.1	0.8/0.1	0.8/0.1
num_weeks/days/hours	1/1/1	1/1/1	1/1/1	0/0/1	0/0/1
Model Configuration					
d_model	64	64	64	32	32
d_ff_*	128	128	128	64	64
n_belt_block	2	2	2	1	1
head_t/head_f	4/4	4/4	4/4	2/2	2/2
Training Configuration					
batch_size	128	64	96	16	16
learning_rate	0.005	0.0001	0.0005	0.005	0.001
Intervention Configuration					
intervention_frequency	10	10	10	2	2
num_interventions	30	30	30	25	25

B.1.2 LLM-Enhanced Retrieval Comparison Configuration. To control variables and verify the independent contribution of LLM retrieval, we select 5 representative nodes from PEMS08 and METR-LA datasets to construct subsets. The experimental design employs strict variable control: maintaining all configurations identical except for the *use_llm_embedding* parameter. Historical data windows are set to 4 weeks/6 days/1 hour, providing retrieval candidates for the LLM. Model capacity is reduced ($d\_model = 8$) to accommodate small-scale networks and avoid overfitting, while intervention frequency is increased to 2 to enhance directed dependency learning effectiveness in small networks. As shown in Table 11, this control ensures that performance differences can be attributed to the LLM retrieval mechanism.

Table 11: LLM comparison experiment configuration (5-node subset)

Parameter	Configuration Value
num_of_vertices	5
num_weeks/days/hours	4/6/1
d_model	8
d_ff_*	16
n_belt_block	1
intervention_frequency	2
num_interventions	5
batch_size	128
learning_rate	0.0005
Unique Variable	
use_llm_embedding	False → True

B.1.3 Ablation Experiment Configuration. Ablation experiments are based on the complete PEMS08 configuration, with different components controlled through the ablation_mode parameter. We systematically evaluate 8 key components, with all ablation variants maintaining identical training configurations to ensure fair comparison.

B.1.4 Dream3 Directed Dependency Validation Configuration. To quantitatively evaluate Orion's directed dependency learning capability, we conduct experiments on the DREAM-3 gene regulatory network dataset. Unlike the main application scenario (traffic forecasting), DREAM-3 is configured as a static causal discovery task: we set the number of temporal segments $S = 1$ to learn a single time-invariant regulatory network $C \in \mathbb{R}^{100\times100}$, adopting single-step lag modeling (gene i at time t influences gene j at time $t + 1$), which aligns with the standard setting of the DREAM challenge.

Due to the specificity of the DREAM-3 dataset (only 21 time points), specialized configurations are required to prevent overfitting: (1) a small learning rate (1e-5) and a lightweight model (*d_model*=8) are used to adapt to the small data scale (21 time points × 100 samples); (2) temporal segments are set to 1 since gene expression data lacks daily periodicity; (3) the dependency threshold is set to 0.05, stricter than traffic data (0.1), to avoid false positive edges in biological networks. These configurations (Table 12) are specifically designed for the unique characteristics of biological time series data.

Table 12: Dream3 directed dependency validation experiment configuration

Parameter	Configuration Value
num_of_vertices	100
learning_rate	1e-5
d_model	8
num_time_segments	1
batch_size	16
dependency_threshold	0.05

The perturbation validation mechanism is still employed in this scenario (K=25 genes randomly intervened, with effects propagated through GRU). The training process does not use the ground truth network, and evaluation metrics such as AUC are calculated only during assessment using the official DREAM scripts.All DREAM-3 experiments were conducted on an NVIDIA RTX 2080Ti (22GB) GPU.

B.1.5 Intervention Implementation Details. **Intervention node selection:** At the beginning of each batch, we use *torch.randperm(N)[:K]* to randomly select K nodes (K=30 for traffic data, K=25 for epidemic and DREAM-3 data). The selection process is independent of the current model state and time period.

Intervention value generation: For the selected node i, the intervention value is generated through $v_i = \tanh(g_i(c_i))$, where g_i is a node-specific learnable function and c_i is noise sampled from a standard normal distribution.

Intervention implementation: During GRU propagation, the input of the intervened node is forced to be v_i, and its incoming edges are truncated, simulating the semantics of the do-operator.

Validity calculation: The intervention effect $\Delta\hat{y}_j$ is calculated by comparing predictions before and after intervention, then compared with C_{ij} to obtain the validity score (Equation 5). The entire process is differentiable and trained end-to-end.

B.2 K-means Node Selection Strategy

Due to the computational complexity of LLM retrieval being $O(N\times H)$, where N is the number of nodes and H is the number of historical candidates, direct application on large-scale networks would incur prohibitive computational overhead. Therefore, we employ K-means clustering to select representative node subsets for LLM comparison experiments. All LLM-enhanced retrieval experiments were implemented on an NVIDIA RTX 5880 Ada (48GB) GPU with Intel Xeon 6330 CPU and 107GB RAM.

The node selection process comprises four steps:

Step 1: Feature Extraction. For each node i, we extract a four-dimensional feature vector from its time series data:

$$\mathbf{f}_i = [\mu_i, \sigma_i, \rho_i(12), r_i^{peak}] \tag{13}$$

where μ_i and σ_i are the mean and standard deviation of node i's traffic flow respectively, $\rho_i(12)$ is the autocorrelation coefficient at lag 12 steps (1 hour) capturing short-term periodicity, and r_i^{peak} is the peak hour flow ratio reflecting the node's functional characteristics.

Step 2: K-means Clustering. We execute standard K-means algorithm on the feature matrix $\mathbf{F} = [\mathbf{f}_1, ..., \mathbf{f}_N]^T$, setting cluster number $k = 5$ to obtain cluster centers $\mathbf{C} = \{\mathbf{c}_1, ..., \mathbf{c}_5\}$.

Step 3: Representative Node Selection. From each cluster, we select the node closest to the centroid as representative:

$$n_j^* = \arg\min_{n\in S_j} ||\mathbf{f}_n - \mathbf{c}_j||_2 \tag{14}$$

where S_j denotes the node set of the j-th cluster.

Step 4: Connectivity Verification. We examine the connectivity of selected nodes in the adjacency matrix to ensure the subnetwork maintains certain spatial connectivity.

As shown in Table 13, the METR-LA subset contains 3 isolated nodes (nodes 29, 67, 127 with no edge connections), reflecting K-means' characteristic of selecting based on temporal features rather than spatial topology. This selection strategy ensures diversity in temporal patterns, providing comprehensive test scenarios for LLM-enhanced retrieval—encompassing both spatially correlated node pairs and spatially independent nodes with similar temporal patterns, thereby thoroughly evaluating LLM retrieval effectiveness across different scenarios.

Table 13: K-means node selection results for PEMS08 and METR-LA datasets

New ID	*PEMS08*		*METR-LA*	
	Original ID	Connectivity	Original ID	Connectivity
0	9	Connected	29	Isolated
1	10	Connected	67	Isolated
2	25	Connected	71	Connected
3	31	Connected	127	Isolated
4	166	Connected	156	Connected

B.3 Three-Stage Training Weight Configuration

The three-stage training strategy employs carefully calibrated loss component weights to progressively transition from feature learning to directed dependency discovery. Table 14 details the individual loss component weights during Stage 2:

$$\mathcal{L}_{causal\_total} = w_{sparse}\mathcal{L}_{sparse} + w_{valid}\mathcal{L}_{validity} + w_{conn}\mathcal{L}_{connect} + w_{temp}\mathcal{L}_{temporal} + w_{dag}\mathcal{L}_{dag} \quad (15)$$

Table 15 presents the complete loss weight evolution across all three stages.

Table 14: Directed dependency loss component weights in Stage 2.

Causal Loss Component	Weight (w_i)
Sparsity ($\mathcal{L}_{sparse}$)	0.01-0.02*
Validity ($\mathcal{L}_{validity}$)	0.5
Connectivity ($\mathcal{L}_{connect}$)	0.1
Temporal ($\mathcal{L}_{temporal}$)	0.01
DAG ($\mathcal{L}_{dag}$)	0.1

*Adaptive: 0.01 when avg validity >0.5, else 0.02

Table 15: Loss weight distribution across training stages

Loss Component	Stage 1	Stage 2	Stage 3
Prediction ($\mathcal{L}_{pred}$)	1.0	1.0	1.0
Sparsity	0	0.0005	0.0003
Validity	0	0.025	0.009
Connectivity	0	0.005	0.0015
Temporal	0	0.0005	0.00015
DAG	0	0.005	0.0015
Weight Decay	0.0001	0.0001	0.0001

The sparsity weight (*) is dynamically adjusted based on validity scores: when average validity exceeds 0.5, the weight decreases to 0.01 to encourage more connections; otherwise, it increases to 0.02 to enforce sparsity. This adaptive mechanism balances between discovering sufficient directed dependencies and avoiding spurious connections.

B.4 Multi-Scale Temporal Convolution Ablation

We conducted systematic ablation experiments on the selection of kernel numbers in multi-scale temporal convolution to explain the question of "why we choose 8 kernels".

Experimental Setup: All experiments were conducted on the PEMS-08 dataset, keeping all hyperparameters consistent except for the kernel configuration (d_model=64, n_belt_block=2, $batch\_size$=128, $learning\_rate$=0.005). We tested four kernel configurations: (1) Single-3: a single kernel size k=3, serving as a baseline without multi-scale modeling; (2) Sparse-4: sparse sampling of 4 key scales [1,3,7,15], testing whether fewer kernels can cover key patterns; (3) Full-8 (our configuration): dense sampling of 8 odd scales [1,3,5,7,9,11,13,15]; (4) Dense-12: ultra-dense sampling of 12 consecutive scales [1,2,3,...,12], testing whether more kernels bring further improvements. All models were trained for 100 epochs using a three-stage training strategy on NVIDIA A100-40GB GPUs.

As shown in Table 17, the experimental results clearly demonstrate the multi-dimensional impact of kernel configuration on model performance. First, Single-3 shows significant performance degradation compared to Full-8 across all prediction horizons, particularly at long horizon (Horizon 12) where MAE increases from 15.18 to 17.99 (degradation of 18.5%) and RMSE from 23.37 to 27.81 (degradation of 19.0%), fully demonstrating that a single temporal scale cannot adequately capture the complex periodic and trend patterns in spatiotemporal data. Second, although Sparse-4 covers both the minimum (k=1) and maximum (k=15) scales, its performance still lags behind Full-8 by approximately 6.92% in MAE, indicating that intermediate scales (k=5,9,11,13) are equally critical for capturing temporal patterns between immediate response and long-term trends (such as half-hour fluctuations) and cannot be simply omitted.

In terms of computational efficiency, the training time of Full-8 (52.7s/epoch) is comparable to Single-3, because the efficient implementation of dilated convolutions allows the computation of multiple kernels to be parallelized; however, the training time of Dense-12 surges to approximately 45 minutes/epoch (2700s), mainly due to the larger parameter count from 12 kernels, the lack of efficiency advantages from dilated structures in consecutive small kernels (k=2,4,6...), and increased memory and computational overhead from denser feature fusion, making complete training (100 epochs) require approximately 75 hours, which is unacceptable in practical applications. Finally, we choose odd kernel sizes (1,3,5,...,15) instead of even numbers to maintain temporal symmetry, because odd kernels have a clear "current" time point at the center moment, while even kernels introduce a half-step temporal offset, which may cause time alignment issues in causal inference.

As shown in Table 16, Full-8 is clearly superior in Pareto efficiency: compared to Single-3, it achieves 7.26% MAE improvement without increasing training time; compared to Sparse-4, it achieves 6.48% further improvement with even shorter training time (benefiting from better parallelization), while the training cost of Dense-12 makes it impractical in real applications.

Table 16: Pareto Analysis of Training Time and Performance

Configuration	MAE	Training Time (h/100epochs)	Efficiency Ratio
Single-3	14.33	1.46	Baseline
Sparse-4	14.21	1.61	-0.08% / +10.3%
Full-8	**13.29**	**1.46**	-7.26% / +0%
Dense-12	—	~75.00	— / +5000%

Based on systematic ablation experiments, the 8 odd kernel sizes [1,3,5,7,9,11,13,15] achieve the optimal balance among performance, computational efficiency, and interpretability, and are our recommended configuration for PEMS-08 and all other datasets.

C LLM Fine-tuning Details

Our LLM is specifically fine-tuned on spatiotemporal data using a dataset of ~100,000 time series–prediction pairs from PEMS03. Each sample contains 240 historical steps (one month of 5-minute data) with 12-step predictions. This window design matches our tri-period architecture: hourly (recent hours), daily (past 7 days), and weekly (past 4 weeks). Similarity is computed with weights $W_{cosine} = 0.7$ and $W_{euclidean} = 0.3$ to balance semantic matching.

Table 17: Detailed Performance Comparison of Different Kernel Configurations

	Horizon 3			*Horizon 6*			*Horizon 12*			*Average*			
Config	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	**Time (s/ep)**
3	9.79	14.62	**7.30**	13.24	20.88	**8.41**	17.99	27.81	**10.25**	14.33	22.73	**8.71**	52.7
4	**8.76**	**12.08**	7.55	13.35	20.06	9.62	17.46	26.51	11.65	14.21	21.81	10.16	57.9
8	9.44	12.98	8.82	**13.13**	**19.66**	10.41	**15.18**	**23.37**	11.42	**13.29**	**20.24**	10.55	**52.7**
12	—	—	—	—	—	—	—	—	—	—	—	—	~2700

C.1 Dataset Construction and Preprocessing

The construction of the fine-tuning dataset is critical for enabling the LLM to understand time series prediction semantics. We systematically extracted approximately 100,000 training samples from raw traffic flow data, with each sample containing 240 historical observations and their corresponding 12-step prediction targets. The data is organized in ChatML format with the following structure:

```
{
  "messages": [
    {
      "role": "user",
      "content": "[240-step historical sequence],
            Predict traffic flow in next 12 steps."
    },
    {
      "role": "assistant",
      "content": "[12-step future predictions]"
    }
  ]
}
```

This conversational format enables the model to understand the prediction task naturally. During preprocessing, all values are normalized to $[0, 1]$ and retain 4 decimal places. Given LLM input token limitations (max_length=2048), we employ backward truncation to prioritize recent observations. This strategy is justified because: (1) recent data carries more predictive information in time series forecasting; (2) early patterns appear repeatedly in training, allowing effective pattern learning.

C.2 Training Configuration and Optimization Strategy

The fine-tuning process employs a parameter-efficient training strategy. The specific configurations are shown in Table 18.

Table 18: Fine-tuning hyperparameters and configurations

Hyperparameter	**Value**	**Description**
Batch size	16	Effective batch per GPU
Gradient accumulation	2	Equivalent batch size of 32
Initial learning rate	2e-5	AdamW optimizer learning rate
Training epochs	5	Prevent overfitting
Warmup steps	200	Linear warmup strategy
Max sequence length	2048	Context window limit
Weight decay	0.01	L2 regularization coefficient

Training employs gradient checkpointing [7] to reduce memory usage while utilizing mixed precision [30] training to improve efficiency. The learning rate schedule adopts a linear warmup followed by constant strategy to ensure stable model convergence.

C.3 Training Process and Convergence Analysis

During the training process, model perplexity steadily decreased from an initial 8.3 to 2.1, indicating successful learning of the semantic representation of spatiotemporal data. The validation loss curve shows the model stabilized after epoch 3 without overfitting phenomena. This is attributed to our early stopping strategy and moderate regularization settings.

The fine-tuned model demonstrates powerful recognition capabilities for temporal patterns. Even when the absolute value ranges of input sequences differ, the model can accurately identify typical traffic patterns such as "morning peak-off-peak-evening peak" and retrieve the most similar historical patterns for matching. For instance, the model successfully recognizes that a sequence with pattern "early peak - stable flow - evening peak" semantically matches similar daily patterns despite occurring on different calendar days. This semantic understanding capability forms the core foundation of our intelligent retrieval mechanism.

D Datasets and Evaluation

D.1 Dataset Statistics

Table 19 presents comprehensive dataset statistics. The traffic datasets exhibit high-frequency sampling with rich feature dimensions, while epidemic datasets present unique challenges with daily granularity and smaller sample sizes. This diversity enables thorough evaluation of Orion's generalization capabilities across varying temporal scales and domain characteristics.

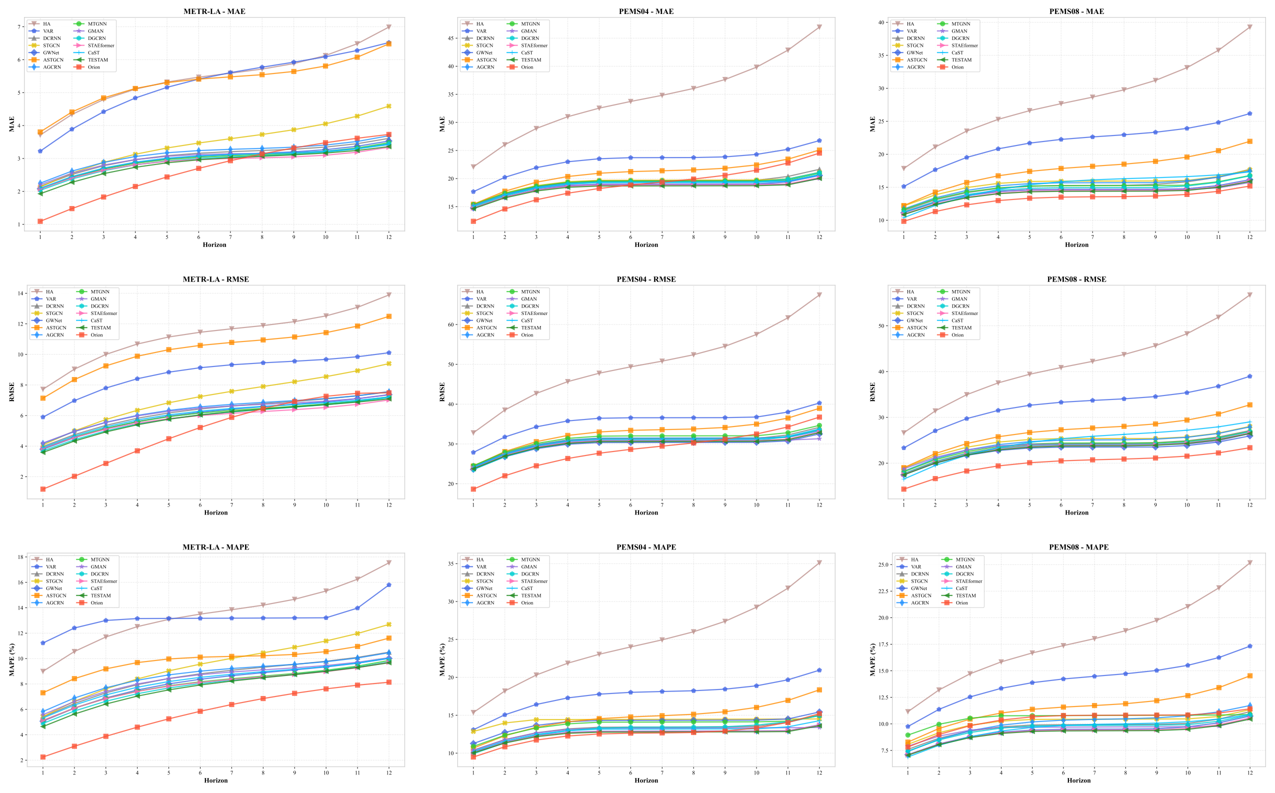
D.2 Evaluation Metrics Definition

We employ three standard evaluation metrics to comprehensively assess prediction performance:

$$\text{MAE} = \frac{1}{N \times H} \sum_{i=1}^{N} \sum_{t=1}^{H} |y_i^{(t)} - \hat{y}_i^{(t)}| \quad (16)$$

$$\text{RMSE} = \sqrt{\frac{1}{N \times H} \sum_{i=1}^{N} \sum_{t=1}^{H} (y_i^{(t)} - \hat{y}_i^{(t)})^2} \quad (17)$$

$$\text{MAPE} = \frac{100\%}{N \times H} \sum_{i=1}^{N} \sum_{t=1}^{H} \left| \frac{y_i^{(t)} - \hat{y}_i^{(t)}}{\max(|y_i^{(t)}|, \epsilon_{\text{thresh}})} \right| \quad (18)$$



(a) METR-LA (b) PEMS04 (c) PEMS08

Figure 8: Performance comparison across 12 prediction horizons on three traffic datasets.

Table 19: Dataset statistics

Dataset	**Steps**	**Nodes**	**Features**	**Feature Names**	**Interval**	**Time Span**	**Value Range**	**Unit**
PEMS04	16,992	307	3	Flow Occupancy Speed	5min	Jan-Feb 2018	[0, 919] [0, 0.772] [3, 85.2]	veh/h ratio mph
PEMS08	17,856	170	3	Flow Occupancy Speed	5min	Jul-Aug 2016	[0, 1147] [0, 0.896] [3, 82.3]	veh/h ratio mph
METR-LA	34,272	207	1	Speed	5min	Mar-Jun 2012	[0, 70]	mph
CA-COVID	335	55	3	Cases Population Change Rate	Daily	Feb-Dec 2020	[0, 7928] [0, 7602] [-2263, 2446]	count normalized count/day
TX-COVID	335	251	3	Cases Population Change Rate	Daily	Feb-Dec 2020	[0, 2149] [0, 2106] [-569, 381]	count normalized count/day

where $y_i^{(t)}$ and $\hat{y}_i^{(t)}$ represent the ground truth and prediction for node i at time t respectively, N is the number of nodes, H is the prediction horizon, and $\epsilon_{\text{thresh}} = 5$ is a threshold to handle near-zero values in traffic datasets.

MAE measures average absolute deviation, providing a linear penalty for errors and being robust to outliers. RMSE applies quadratic penalty to errors, making it more sensitive to large deviations and suitable for applications where large errors are particularly undesirable. MAPE provides a scale-independent percentage error measure, enabling fair comparison across different value ranges, though it requires careful handling of near-zero values.

D.3 Data Preprocessing

Traffic datasets undergo preprocessing including IQR-based outlier detection [40] and Savitzky-Golay smoothing [38] to handle sensor noise, while epidemic datasets retain raw values due to their daily aggregation nature. The preprocessing pipeline consists of: (1) missing value imputation using linear interpolation for gaps less than 3 time steps and forward-filling for longer gaps; (2) outlier detection using the interquartile range method with a factor of 1.5; (3) noise reduction through Savitzky-Golay filtering with window length 7 and polynomial order 2 for traffic data only; (4) normalization using training set statistics for all splits to prevent data leakage.

E Supplementary Experimental Results

E.1 Performance Comparison Visualization

Figure 8 visualizes complete 12-horizon performance trajectories of all methods across three traffic datasets, revealing distinct clustering. On METR-LA, statistical methods (HA, VAR) form the highest error cluster with steep degradation curves. Deep learning methods converge to a competitive band between MAE 3.0-4.0, with marginal differences reflecting architectural variations. Orion establishes a clearly lower trajectory, maintaining advantages across all horizons. The visualization highlights Orion's superior early-horizon performance, where directed dependencies exhibit strongest predictive power before uncertainty affects long-term predictions.

E.2 METR-LA Supplementary Analysis

Extended Prediction Results Analysis. Figure 9 demonstrates Orion's 600-step prediction results on 4 representative nodes from METR-LA. Node 0 displays distinct V-shaped valleys (flow dropping from 60 to 20), which Orion almost perfectly reproduces. Node 1's smoother pattern achieves the best MAE of 2.323. Node 2 exhibits sudden changes—flow plummeting from 80 to 20 or surging from 40 to 100—with Orion accurately capturing both timing and magnitude. This sensitivity stems from multi-scale temporal convolutions detecting both short-term disruptions and long-term trends. Node 3 presents oscillations between 40-70 combining regular periodicity with irregular perturbations; Orion maintains 6.4% MAPE even in complex segments. At extreme valleys approaching flow of 10, the model shows slight overestimation but quickly self-corrects, demonstrating TE-CausGAT's generalization capability.

Temporal Attention Weight Analysis. Temporal attention weight analysis (Figure 10) reveals the model's sophisticated temporal modeling mechanism. The global average attention heatmap displays clear diagonal dominance with maximum weight reaching 0.1179, concentrated on adjacent time steps, embodying traffic flow's Markovian properties. More interestingly, secondary attention peaks appear at t-6 and t-8 positions, corresponding exactly to historical data from 1-1.5 hours ago (12 steps × 5 minutes). This "bimodal" attention pattern enables the model to simultaneously capture immediate dynamics and hourly periodicity, perfectly matching urban traffic temporal characteristics.

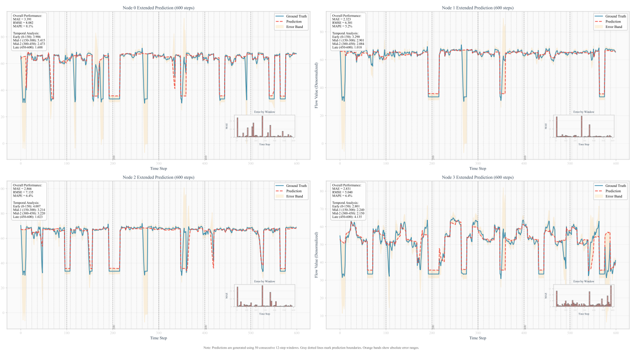
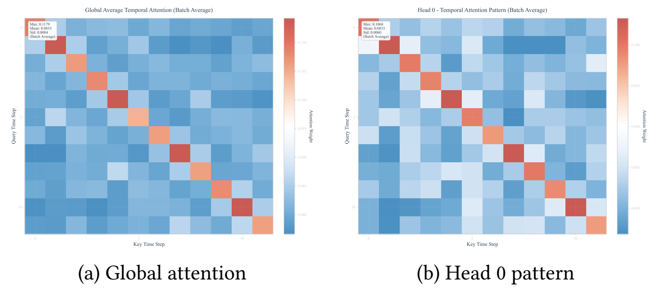


Figure 9: Extended prediction analysis for 4 representative nodes over 600 time steps through 50 consecutive 12-step sliding windows. Blue lines show ground truth, red dashed lines show predictions, orange bands indicate error ranges, and gray dotted lines mark prediction window boundaries.

Multi-Period Fusion Weight Analysis. Multi-period fusion weight analysis (Figure 11) reveals surprising yet reasonable patterns. Weekly period data receives the highest weight of 48.7%, far exceeding hourly (26.4%) and daily (25.0%) periods. This finding highly aligns with METR-LA's actual traffic characteristics—Los Angeles traffic shows significant differences between weekdays and weekends, with weekly patterns' importance surpassing daily variations. This adaptive weight allocation mechanism enables Orion to automatically adjust multi-scale information fusion strategies based on different datasets' characteristics.



(a) Global attention (b) Head 0 pattern

Figure 10: Temporal attention patterns in METR-LA. Maximum weight reaches 0.1179 with standard deviation of only 0.0084, demonstrating high stability.

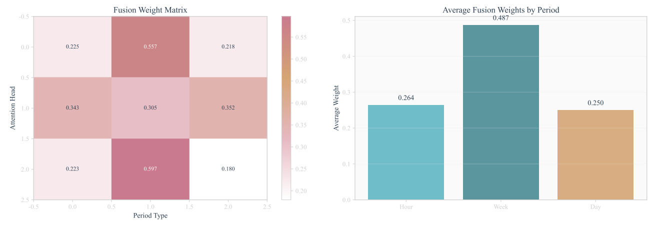


Figure 11: Multi-period fusion weight analysis.

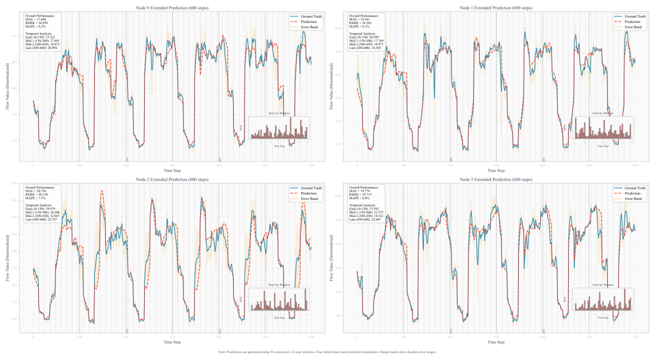


Figure 12: Extended prediction results on PEMS08 for nodes 0-3. Predictions generated using 50 consecutive 12-step sliding windows for 600 time steps. Gray dotted lines mark prediction boundaries and orange bands show absolute error ranges.

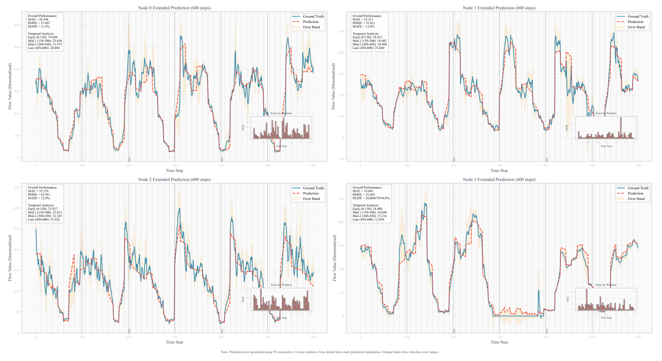


Figure 13: Extended prediction results on PEMS04 dataset for nodes 0-3, showing 600-step forecasts via 50 sliding windows. Note the anomalous MAPE for Node 3 due to near-zero ground truth values.

E.3 PEMS Datasets Supplementary Analysis

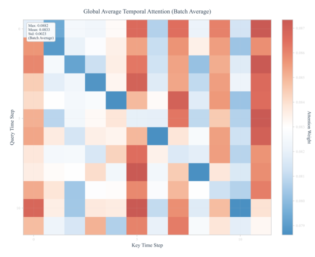
Since the PEMS04 and PEMS08 datasets lack precise geographic coordinates for sensors, we cannot perform spatial causality visualization analysis similar to METR-LA. This section focuses on detailed visualizations of prediction performance and attention mechanisms on these two datasets.

E.3.1 Extended Prediction Performance. Figures 12 and 13 present prediction results on the PEMS datasets, which exhibit different traffic characteristics and challenges.

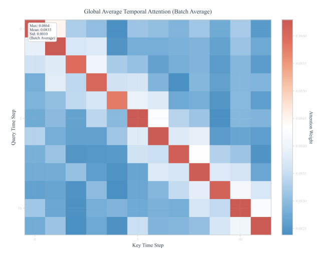
PEMS08 Regular Pattern Recognition. PEMS08 displays highly regular traffic patterns. Nodes 0 and 1 exhibit clear rectangular wave patterns—rapidly switching between high-flow plateaus and low-flow valleys. Orion precisely identifies these binary state transitions, with the red prediction line accurately responding at each transition point without delay or overshoot. Particularly for Node 0's multiple deep valleys (flow dropping to near zero), the model not only predicts the extreme low values at valley bottoms but also accurately captures their duration. Node 2 demonstrates more gradual fluctuations, where Orion similarly achieves good fitting with only slight underestimation at individual peaks.

PEMS04 Sparse Flow Challenges. The most distinctive feature of the PEMS04 dataset is extreme flow sparsity—multiple nodes maintain near-zero flow for prolonged periods. Nodes 0 and 2 frequently exhibit complete flow cessation, posing severe challenges for prediction models. Orion demonstrates remarkable adaptability: accurately maintaining zero predictions during zero-flow periods and responding promptly when flow resumes. However, Node 3's anomalous MAPE value exposes limitations of the evaluation metric—when ground truth values approach zero, even minimal absolute errors can cause enormous relative errors. Nevertheless, visually, the red prediction line still closely adheres to the blue ground truth, indicating the model's actual performance far exceeds what the MAPE metric suggests.

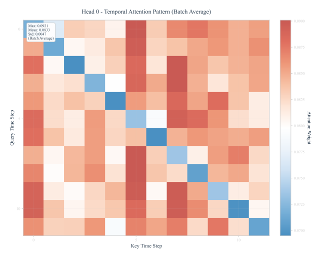
E.3.2 Temporal Attention Patterns. Figure 14 illustrates temporal attention weight distributions for both datasets. PEMS08's global average attention presents a clear diagonal-dominant pattern with maximum weight 0.0882 on adjacent time steps and standard deviation 0.0023, reflecting high stability. Head 0's pattern reveals richer structure with a secondary peak at t-6, corresponding to data from 1 hour ago (12 steps × 5 minutes), capturing hourly periodicity. PEMS04's attention is more concentrated with maximum weight 0.0864 and standard deviation 0.0010. This tighter distribution may relate to PEMS04's shorter time span (2 months), with the model relying more on recent observations. The reinforced diagonal pattern indicates stronger Markovian properties in this dataset.



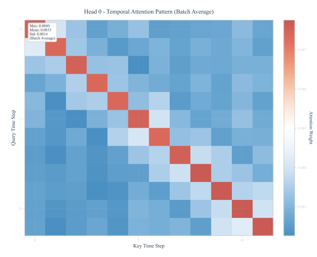
(a) PEMS08 global attention



(b) PEMS04 global attention



(c) PEMS08 Head 0 attention



(d) PEMS04 Head 0 attention

Figure 14: Temporal attention patterns on PEMS datasets.

Notably, fusion weights for both datasets show consistency across attention heads, with minimal weight distribution differences between different heads, validating the stability and robustness of our multi-head fusion mechanism.

E.3.3 Multi-Period Fusion Weights. The multi-period fusion weight distribution provides insights into temporal scale preferences. As shown in Figure 15, on PEMS08, average fusion weights for hour, week, and day periods are 0.337, 0.322, and 0.341 respectively, showing relatively balanced distribution. The batch-averaged standard deviation is 0.015, indicating certain dynamics in weight allocation across time segments. PEMS04 exhibits more uniform weight allocation, with all three periods extremely close to 0.333 and standard deviation of only 0.001. This balanced pattern suggests the model considers information from all three temporal scales equally important, possibly reflecting stronger regularity in traffic patterns where periodic features at different scales all provide valuable predictive information.

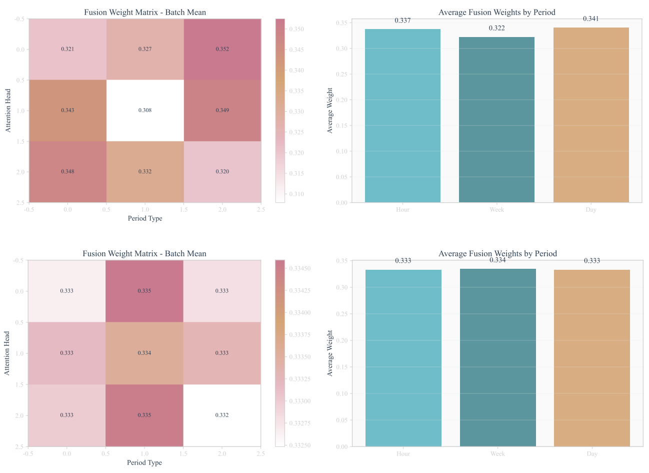


Figure 15: Multi-period fusion weight distribution. Top: PEMS08. Bottom: PEMS04.

F Reproducibility Statement

To facilitate independent verification, we publicly release all experiment code, configuration files, and data preprocessing scripts at the anonymous repository linked in Section 1. Our implementation is built on PyTorch 2.0+ with experiments conducted on NVIDIA A100-40GB GPUs (traffic datasets), RTX 2080Ti-22GB GPUs (epidemic datasets), and RTX 5880 Ada-48GB GPU (LLM retrieval). Deterministic execution is ensured by fixing all random seeds to 42 across Python, NumPy, PyTorch, and CUDA backends. Every hyperparameter is exhaustively documented in the appendix, including three-stage training schedules (Tables 10–15), intervention details (Appendix B), and preprocessing procedures (Appendix D.3). All five datasets are publicly available benchmarks with standard community splits. The repository includes a comprehensive `README.md` with step-by-step instructions for environment setup, data preparation, and end-to-end training.

G Limitations

Although Orion achieves strong results across five benchmarks, several practical constraints should be noted. The intervention-based validation requires extra forward passes during training, adding roughly 15–20% wall-clock overhead compared to a correlation-only baseline; in practice this is kept manageable through the intervention frequency hyperparameter and by restricting the causal learning phase to 10 of the 100 total epochs, with no additional cost at inference time. On PEMS04, which covers only 2 months with 307 nodes, the multi-period architecture tends to overfit at longer horizons—a data scarcity issue rather than a modeling flaw, as the same design yields consistent gains on METR-LA (4 months) and PEMS08 (170 nodes). Our polynomial DAG constraint penalizes only 2- and 3-cycles and thus cannot theoretically rule out longer cycles, though empirical verification via topological sorting detects none on PEMS-08 at threshold 0.1 (Table 9). The TinyLlama retrieval module has $O(N \times H)$ complexity, and due to limited GPU memory availability, full-network LLM-based retrieval remains computationally prohibitive for datasets with hundreds of nodes, restricting current evaluation to 5-node subsets obtained via K-means; all primary results reported in Table 2 rely on fixed-period retrieval and are therefore unaffected by this constraint.

H Ethics Statement

This study uses exclusively publicly available datasets containing no personally identifiable information. The traffic datasets (PEMS04, PEMS08, METR-LA) consist of aggregated loop-detector readings recording only vehicle counts, occupancy ratios, and speeds; the COVID-19 datasets (CA-COVID, TX-COVID) provide county-level hospitalization statistics from Dong et al. [10]; and the DREAM-3 dataset [29] comprises entirely synthetic gene expression trajectories. All usage complies with respective distribution licenses and privacy regulations. Orion is designed as a decision-support tool whose forecasts and dependency structures are intended to inform—rather than replace—human judgment in traffic management and public health planning. No human subjects were involved at any stage of this research, and no demographic or sensitive attributes are modeled or inferred.