SLiDNet: Scale-Separated Liquid Dynamics Network for Spatio-Temporal Forecasting

Anonymous authors
Paper under double-blind review

Abstract

Spatio-temporal forecasting requires modeling both the state of each node and the rate at which that state evolves. However, existing models often rely on shared latent representations for state formation and temporal evolution. This coupling may mix signal-level differences with temporal variation and make it harder to distinguish a node's own evolution from the influence of its neighbors. To address this issue, we propose the Scale-Separated Liquid Dynamics Network (SLiDNet), which separates state formation from evolution-rate modeling. SLiDNet forms node states from scale-stable temporal content and compact window-level signal context. Robustly normalized first- and second-order temporal differences, together with recurrent context, condition node-specific Liquid time constants to adapt the evolution rate. Spatial dependencies are captured through prior-guided graph diffusion. Experiments across six real-world spatio-temporal datasets and multiple forecasting horizons show consistent improvements over competitive baselines. Controlled ablations and sensitivity analyses further support the effectiveness of the proposed scale-separated design.

1 Introduction

A central challenge in spatio-temporal forecasting is to model heterogeneous node dynamics in interconnected systems, where interacting variables can exhibit different value ranges and temporal variations, as commonly observed in intelligent transportation and distributed renewable-energy networks (Li et al., 2018; Simeunovic et al., 2022). However, existing models often use shared latent representations for both state formation and evolution rates. When these representations are tied to signal amplitude, they can conflate signal-level information with relative temporal variation. This leads to a key question: *what state should each node form, and how rapidly should that state evolve?*

This distinction is readily observed in METR-LA (Li et al., 2018), where traffic speed represents the traffic state at each sensor, while its signal level characterizes the operating level of that state and its temporal variation reflects how rapidly the state evolves. As shown in Fig. 1, nodes with similar signal levels can exhibit markedly different temporal variations, whereas nodes with different signal levels may show comparable variation patterns. Thus, the level of a node's state alone does not determine how rapidly that state evolves. These observations suggest that signal level and temporal variation provide distinct but complementary cues for state formation and evolution-rate modeling.

Existing spatio-temporal models capture heterogeneous dependencies through adaptive graphs, node-specific parameterization, and spatio-temporal embeddings (Wu et al., 2019; Bai et al., 2020; Liu et al., 2023a; Dong et al., 2024). However, state formation from amplitude-varying observations may still mix relative temporal structure with signal-level information, while normalization mainly reduces amplitude sensitivity and can leave the latter implicit (Deng et al., 2021; Kim et al., 2022; Liu et al., 2023b). A suitable state-formation mechanism should therefore capture relative temporal structure while retaining explicit signal-level context.

State formation alone does not determine how rapidly the resulting state should evolve. Continuous-time graph models describe spatio-temporal evolution through ordinary or controlled differential equations (Fang et al., 2021; Choi et al., 2022), while Liquid models provide adaptive time constants and closed-form continuous-time dynamics (Hasani et al., 2021; 2022). Yet adaptive dynamics do not by themselves specify which information should govern the evolution rate.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

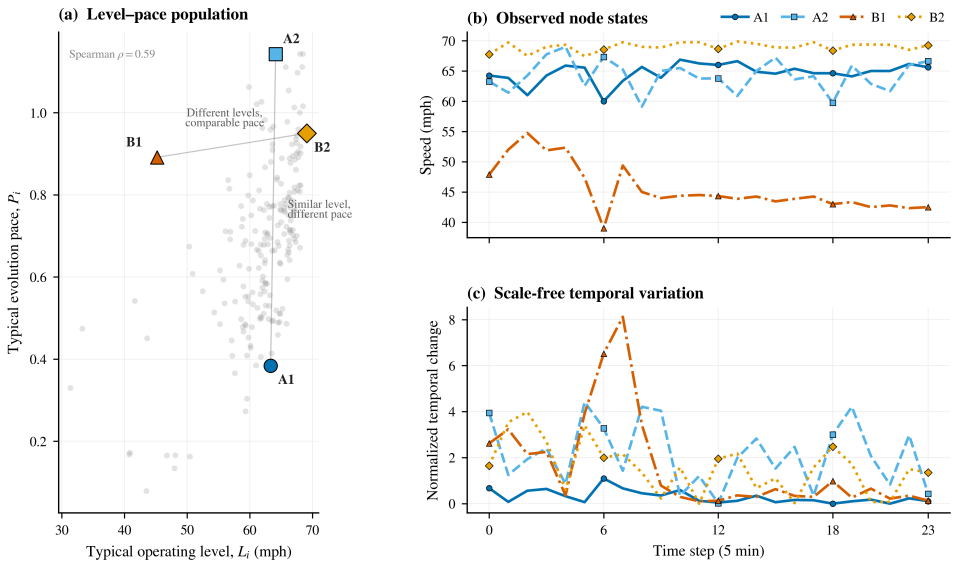

Figure 1: Signal level and temporal variation in METR-LA. (a) Node-wise distribution over the training split; A1/A2 show similar signal levels with different variations, whereas B1/B2 show different signal levels with comparable variations. (b) Raw speed trajectories. (c) Corresponding normalized temporal changes. Definitions are provided in Appendix B.

Taken together, the core challenge is *how signal-level features should inform state formation and relative temporal variation guide evolution-rate modeling*, while preserving their distinct roles within unified latent dynamics.

To address this challenge, we propose SLiDNet, a Scale-Separated Liquid Dynamics Network that separates state formation from evolution-rate modeling. It leverages a Liquid update, in which state evolution is parameterized by an update target and an adaptive time constant. SLiDNet forms the update target from scale-stable temporal content and compact signal-level context, while conditioning node-specific time constants on relative temporal variation and recurrent context. Spatial dependencies are captured by prior-guided diffusion graph convolution.

The main contributions are summarized as follows:

- We introduce a scale-separated modeling approach that separates state formation from evolution-rate modeling under heterogeneous node dynamics.
- We develop SLiDNet, a scale-separated Liquid architecture that combines scale-stable state formation with adaptive node-specific evolution-rate modeling, while incorporating prior-guided graph diffusion for spatial dependencies.
- Extensive experiments across six real-world spatio-temporal datasets and multiple forecasting horizons demonstrate consistent forecasting gains. Controlled ablations and sensitivity analyses further support the effectiveness of the proposed scale-separated design.

2 Methodology

2.1 Problem Definition

We consider graph-based spatio-temporal forecasting over N interconnected nodes. At time step t, $X_t \in \mathbb{R}^{C\times N}$ collects C input features from all nodes. Given a historical window of T steps, $X_{t-T+1:t} \in \mathbb{R}^{C\times N\times T}$, the goal is to predict the future H steps.

Spatial relations are represented by a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{A})$, where $\mathcal{V}$ and $\mathcal{E}$ denote the node and edge sets, and $\mathbf{A} \in \mathbb{R}^{N\times N}$ is the adjacency matrix.

The forecasting task is to learn a mapping $\mathcal{F}_\theta$ such that $\hat{\mathbf{Y}}_{t+1:t+H} = \mathcal{F}_\theta(X_{t-T+1:t}, \mathcal{G})$, where $\hat{\mathbf{Y}}_{t+1:t+H} \in \mathbb{R}^{N\times H}$ denotes the predicted future target values.

2.2 Overview of SLiDNet

As illustrated in Fig. 2, SLiDNet first constructs scale-stable temporal content, window-level signal context, and relative temporal variation from the historical observations. The temporal content enters four stacked SLiD blocks, with the signal context and temporal variation providing complementary conditioning for the Liquid dynamics. In each block, state formation and node-specific evolution-rate control are realized within the Liquid update. Graph diffusion then refines the Liquid features before they enter the residual and skip paths. The residual output becomes the input to the next block, while skip features from all blocks are summed and decoded into the multi-step forecast.

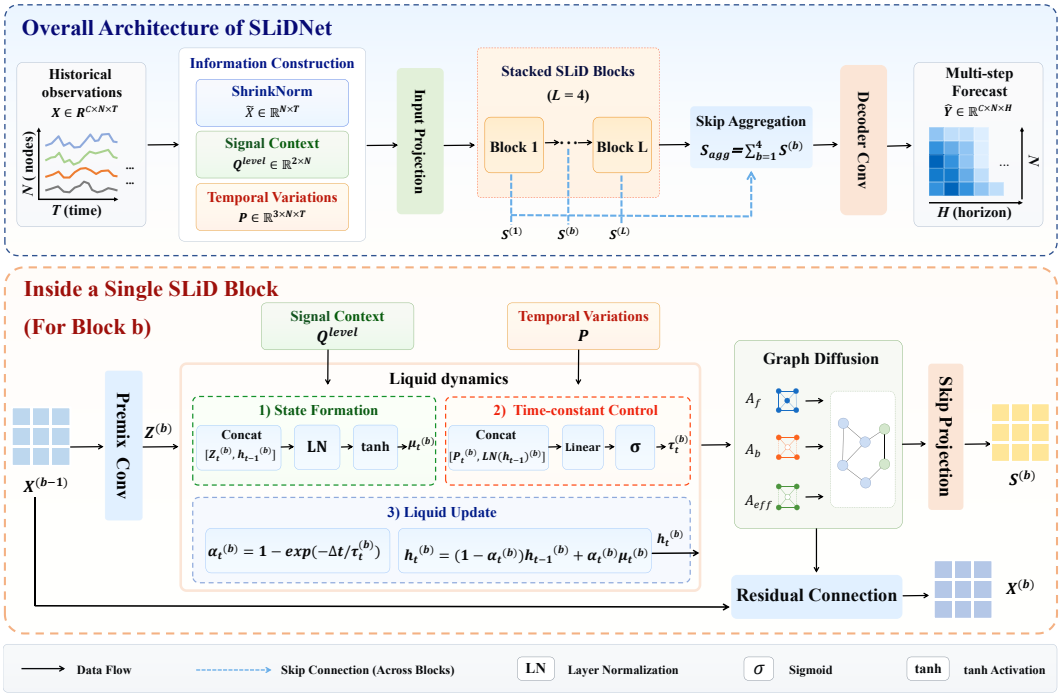

Figure 2: Framework of SLiDNet. Four stacked SLiD blocks integrate state formation, evolution-rate control, and graph diffusion, with skip features summed for multi-step forecasting.

2.3 Scale-Separated Information Construction

Before entering the stacked SLiD backbone, we construct complementary representations for state formation and evolution-rate modeling. For notational simplicity, we present the construction for a single target variable and omit its variable index.

Shrinkage-normalized content. Let $X = [x_{i,t}] \in \mathbb{R}^{N\times T}$ denote the historical observations over the current window. For each node i, reference statistics (μ_i^{tr}, v_i^{tr}) are estimated from the training data, while (μ_i^{w}, v_i^{w}) are computed from the current window. With $\lambda_T = T/(T+\kappa)$, we define the active center and scale as

$$c_i = \lambda_T \mu_i^{w} + (1-\lambda_T)\mu_i^{\mathrm{tr}}, \qquad s_i = \sqrt{\lambda_T v_i^{w} + (1-\lambda_T)v_i^{\mathrm{tr}} + \epsilon_s}, \tag{1}$$

where $\epsilon_s > 0$ ensures numerical stability. ShrinkNorm maps each observation to $\widetilde{x}_{i,t} = (x_{i,t} - c_i)/s_i$, yielding scale-stable coordinates for relative temporal structure while retaining window-specific adaptation. The reference statistics are estimated once from the training data and fixed thereafter. Detailed definitions are provided in Appendix D.1.

Window-level signal context. Different signal conditions may yield similar normalized trajectories; we therefore retain the latest observation and active scale as a compact node-wise context:

$$q_i^{\mathrm{level}} = [x_{i,T},\ \log(s_i + \epsilon_L)]^{\top}, \qquad Q^{\mathrm{level}} \in \mathbb{R}^{2\times N}. \tag{2}$$

Supplied separately from $\widetilde{X}$, this context conditions candidate-state formation in each SLiD block.

Scale-normalized temporal variation. We characterize relative temporal variation using first- and second-order differences, denoted by $\Delta x_{i,t}$ and $\Delta^2 x_{i,t}$. To compare these variations across heterogeneous signal scales, we define a robust window scale:

$$r_i = \sqrt{\mathrm{MAD}(x_i)^2 + \gamma_d \, \mathrm{MAD}(\Delta x_i)^2}. \tag{3}$$

With $\bar{r}_i = \max(r_i, \epsilon_p)$, the normalized variation representation is

$$p_{i,t} = \left[\frac{\Delta x_{i,t}}{\bar{r}_i}, \; \frac{\Delta^2 x_{i,t}}{\bar{r}_i}, \; \frac{|\Delta x_{i,t}|}{\bar{r}_i} \right]^\top. \tag{4}$$

Its three entries respectively encode signed first-order change, second-order change, and absolute first-order change relative to the node's robust window scale. Stacking them over nodes and time yields $P \in \mathbb{R}^{3\times N\times T}$. Detailed definitions and boundary handling are provided in Appendix D.2.

An input projection maps the normalized sequence $\widetilde{X}$ to the initial backbone representation $X^{(0)}$. The signal context Q^{level} and variation representation P are computed once per input window and shared across all four SLiD blocks. Their block-specific effects on candidate formation and Liquid time constants are detailed in Sec. 2.5.

2.4 Stacked SLiD Backbone

Starting from the projected features $X^{(0)}$, SLiDNet propagates them through four stacked SLiD blocks. In block b, a causal gated premix transforms $X^{(b-1)}$ into the premixed sequence $Z^{(b)}$. The Liquid operator then evolves $Z^{(b)}$ into the Liquid features $H_L^{(b)}$, conditioned on the signal context Q^{level} and variation representation P. Graph diffusion over the support set $\mathcal{A} = \{A_f, A_b, A_{\mathrm{eff}}\}$ incorporates spatial interactions, yielding the graph-corrected features $H_D^{(b)}$.

From $H_D^{(b)}$, the block generates a skip feature and a residual update:

$$S^{(b)} = W_{\mathrm{skip}}^{(b)} H_D^{(b)}, \qquad X^{(b)} = \mathrm{BN}^{(b)}\left(X^{(b-1)} + \mathrm{Dropout}\left(W_{\mathrm{res}}^{(b)} H_D^{(b)} \right) \right). \tag{5}$$

The residual output $X^{(b)}$ becomes the input to the next block, while $S^{(b)}$ is retained for skip connection. Each block has its own transformation parameters, whereas Q^{level}, P, and the graph supports are shared across depth. After four blocks, the skip features are aggregated as $S_{\mathrm{agg}} = \sum_{b=1}^{4} S^{(b)}$, and passed to the prediction head.

2.5 Scale-Separated Liquid Dynamics with Spatial Correction

We next detail the internal computation of a SLiD block, which combines *scale-separated Liquid dynamics* for node-wise temporal evolution with *prior-guided graph diffusion* for spatial correction.

Within each block, the premix uses two parallel width-2 causal temporal convolutions to generate filter and gate responses. The two branches apply $\tanh$ and sigmoid activations, and their outputs are multiplied element-wise to form $Z^{(b)}$, which is then processed recurrently by a Liquid cell. For node i at time t, its hidden state evolves over the sampling interval Δt as

$$\alpha_{i,t}^{(b)} = 1 - \exp\left(-\frac{\Delta t}{\tau_{i,t}^{(b)}} \right), \qquad h_{i,t}^{(b)} = \left(1 - \alpha_{i,t}^{(b)}\right) \odot h_{i,t-1}^{(b)} + \alpha_{i,t}^{(b)} \odot u_{i,t}^{(b)}. \tag{6}$$

Eq. (6) can be viewed as the closed-form transition of a first-order continuous-time relaxation toward the candidate state over Δt: $u_{i,t}^{(b)}$ specifies the target state, while $\tau_{i,t}^{(b)}$ determines the relaxation rate. The two draw on different conditioning signals while remaining coupled through the recurrent state.

State formation with signal context. To complement the normalized temporal content, SLiDNet applies a block-specific projection of q_i^{level} to modulate candidate formation. At recurrent step t, the resulting context offset is added to the candidate pre-activation formed from the premixed feature $z_{i,t}^{(b)}$ and previous recurrent state $h_{i,t-1}^{(b)}$:

$$\delta_i^{(b)} = \tanh\left(W_L^{(b)} q_i^{\mathrm{level}} + b_L^{(b)} \right), \qquad u_{i,t}^{(b)} = \tanh\left(W_u^{(b)} \left[z_{i,t}^{(b)} \| h_{i,t-1}^{(b)} \right] + b_u^{(b)} + \delta_i^{(b)} \right). \tag{7}$$

The symbol $\|$ denotes channel-wise concatenation. The premixed feature and recurrent state determine the base pre-activation, while $\delta_i^{(b)}$ introduces the window-level signal context. Since q_i^{level} remains fixed throughout the recurrent scan, the same context offset is applied at each time step. Each block learns its own context projection, allowing similar normalized temporal patterns to yield distinct candidate states under different signal conditions.

Node-calibrated time-scale ranges. To accommodate heterogeneous evolution rates, the hidden channels are assigned positive base lower bounds $\underline{\tau}_0$ and spans $\Delta\tau_0$, defining fast and slow base ranges. In block b, a learnable node embedding $e_i^{(b)}$ produces bounded adjustments $[\eta_i^{(b)} \| \xi_i^{(b)}] = \tanh\Big(\phi_{\text{nt}}^{(b)}(e_i^{(b)})\Big)$, where $\phi_{\text{nt}}^{(b)}$ is an affine projection. With modulation strength $s_\tau \geq 0$, the node-calibrated bounds are

$$\underline{\tau}_i^{(b)} = \underline{\tau}_0 \odot \exp\Big(s_\tau \eta_i^{(b)}\Big), \qquad \overline{\tau}_i^{(b)} = \underline{\tau}_i^{(b)} + \Delta\tau_0 \odot \exp\Big(s_\tau \xi_i^{(b)}\Big). \tag{8}$$

The exponential parameterization preserves positive lower bounds and spans. Each block learns its own embeddings and projection, while the resulting bounds are shared across input windows and recurrent steps, capturing persistent node-wise rate heterogeneity. Implementation details are provided in Appendix E.

Variation-conditioned Liquid evolution. Within these node-calibrated ranges, SLiDNet dynamically adapts the Liquid time constants. At recurrent step t, block b combines the variation feature $p_{i,t}$ with the layer-normalized recurrent state:

$$\ell_{i,t}^{(b)} = W_\tau^{(b)}\Big[p_{i,t} \| \operatorname{LN}\Big(h_{i,t-1}^{(b)}\Big)\Big] + b_\tau^{(b)}, \qquad \tau_{i,t}^{(b)} = \underline{\tau}_i^{(b)} + \Big(\overline{\tau}_i^{(b)} - \underline{\tau}_i^{(b)}\Big) \odot \sigma\Big(\ell_{i,t}^{(b)}\Big). \tag{9}$$

The variation feature encodes current temporal change, while the recurrent state supplies complementary latent context. The sigmoid term constrains each time constant to its node-specific range. For fixed Δt, smaller time constants increase $\alpha_{i,t}^{(b)}$, driving the hidden state toward its candidate more rapidly. Iterating Eq. (6) over the input window produces the hidden-state sequence. A block-specific pointwise projection then yields the Liquid representation $H_L^{(b)}$.

Prior-guided spatial correction. After node-wise Liquid evolution, SLiDNet introduces spatial dependencies through graph diffusion. Following Graph WaveNet (Wu et al., 2019), we construct an adaptive adjacency from learnable node embeddings and combine it with a sparse, row-normalized correlation prior estimated from the training split, which serves as a structural anchor:

$$A_{\text{adp}} = \operatorname{softmax}_{\text{row}}(\operatorname{ReLU}(E_1 E_2)), \qquad A_{\text{eff}} = (1-\rho)A_{\text{adp}} + \rho A_{\text{prior}}. \tag{10}$$

The coefficient $\rho \in [0,1]$ controls the prior contribution. Construction details of A_{prior} are provided in Appendix F. Combining A_{eff} with the forward and backward physical supports forms $\mathcal{A} = \{A_f, A_b, A_{\text{eff}}\}$. Given the Liquid features $H_L^{(b)}$, graph diffusion aggregates multi-hop messages over $\mathcal{A}$:

$$M^{(b)} = \phi_D^{(b)}\left[H_L^{(b)} \| \left\{A^k H_L^{(b)}\right\}_{A\in\mathcal{A},\, k=1,\dots,K}\right], \qquad \beta^{(b)} = \sigma\Big(\phi_\beta^{(b)}\Big[H_L^{(b)} \| M^{(b)}\Big]\Big), \tag{11}$$

where K denotes the diffusion order. The pointwise projections $\phi_D^{(b)}$ and $\phi_\beta^{(b)}$ generate the diffusion message and gate logits. The sigmoid gate $\beta^{(b)}$ controls the contribution of the graph message to the Liquid features, yielding $H_D^{(b)} = H_L^{(b)} + \beta^{(b)} \odot \Big(M^{(b)} - H_L^{(b)}\Big)$. The graph-corrected representation then enters the skip and residual paths in Eq. (5).

2.6 Prediction and Training

We apply a pointwise projection to the aggregated skip features S_{agg} and retain the final temporal position. A two-layer 1×1 convolutional prediction head then generates all H forecast steps in parallel, yielding $\widehat{Y}^{\text{int}}$ in shrinkage-normalized coordinates.

Training objective. Using the same active center and scale (c_i, s_i) as for the input, each internal prediction $\hat{y}^{\text{int}}_{i,k}$ is first mapped back as $c_i + s_i \hat{y}^{\text{int}}_{i,k}$, followed by inverse dataset-level scaling to recover the original data scale. This two-stage inverse transformation reverses the normalization applied to the forecasting variable. We then minimize masked MAE over valid targets:

$$\mathcal{L}_{\text{pred}} = \text{MaskedMAE}\left(\widehat{\mathcal{Y}}, \mathcal{Y}\right). \tag{12}$$

The complete forward procedure is summarized in Algorithm 1.

Algorithm 1 Scale-separated forward propagation of SLiDNet.

Require: Historical input $X_{t-T+1:t}$, graph supports $\{A_f, A_b\}$, spatial prior A_{prior}

Ensure: Multi-step forecast $\widehat{Y}_{t+1:t+H}$

1: Construct shrinkage-normalized content $\widetilde{X}$, signal context Q^{level}, and temporal variation P using Eqs. (1)–(4).
2: $X^{(0)} \leftarrow \text{InputProj}(\widetilde{X})$
3: Construct A_{eff} using Eq. (10), and set $\mathcal{A} \leftarrow \{A_f, A_b, A_{\text{eff}}\}$.
4: $S_{\text{agg}} \leftarrow 0$
5: **for** $b = 1, \ldots, 4$ **do**
6: $\quad Z^{(b)} \leftarrow \text{Premix}^{(b)}(X^{(b-1)})$
7: $\quad$ Compute node-calibrated time-scale ranges using Eq. (8).
8: $\quad$ **for** $t = 1, \ldots, T$ **do**
9: $\quad\quad$ Form candidate state $u_t^{(b)}$ from $Z_t^{(b)}$, $h_{t-1}^{(b)}$, and Q^{level} using Eq. (7).
10: $\quad\quad$ Determine $\tau_t^{(b)}$ from P_t, $h_{t-1}^{(b)}$, and the node-calibrated ranges using Eq. (9).
11: $\quad\quad$ Update the Liquid state using Eq. (6).
12: $\quad$ **end for**
13: $\quad$ Project the recurrent states to Liquid features $H_L^{(b)}$.
14: $\quad$ Refine $H_L^{(b)}$ into graph-corrected features $H_D^{(b)}$ using Eq. (11).
15: $\quad$ Compute $S^{(b)}$ and $X^{(b)}$ using Eq. (5).
16: $\quad S_{\text{agg}} \leftarrow S_{\text{agg}} + S^{(b)}$
17: **end for**
18: Decode S_{agg} into $\widehat{Y}^{\text{int}}$ and apply the inverse transforms.
19: **return** $\widehat{Y}_{t+1:t+H}$

3 Experiments

We evaluate SLiDNet from four perspectives: forecasting performance, contributions of its core mechanisms, sensitivity to key hyperparameters, and computational efficiency.

3.1 Experimental Setup

Datasets and forecasting tasks. We evaluate SLiDNet on six widely used spatio-temporal forecasting benchmarks: METR-LA and PEMS-BAY (Li et al., 2018), together with PEMS03, PEMS04, PEMS07, and PEMS08 (Song et al., 2020). These datasets contain traffic measurements sampled at five-minute intervals and cover networks ranging from 170 to 883 sensors. Detailed statistics are provided in Appendix G.1. For each dataset, observations are divided chronologically into training, validation, and test sets in a 7:1:2 ratio. We evaluate three matched history–forecast configurations, $(T, H) \in \{(3, 3), (6, 6), (12, 12)\}$, corresponding to 15-, 30-, and 60-minute histories and forecasts. Each configuration is evaluated using five random seeds $\{0, 1, 2, 3, 4\}$.

Evaluation metrics and baselines. Forecasting performance is evaluated by mean absolute error (MAE) and root mean squared error (RMSE) in the original data scale. We compare SLiDNet with seven spatio-temporal forecasting models spanning graph-convolutional, recurrent, continuous-time, Transformer-based, pivotal-node graph, and decomposition architectures: Graph WaveNet (Wu et al., 2019), AGCRN (Bai et al., 2020), STG-NCDE (Choi et al., 2022), STAEformer (Liu et al., 2023a), PDFormer (Jiang et al., 2023), STPGNN (Kong et al., 2024), and STDN (Cao et al., 2025). Brief descriptions and settings are provided in Appendix G.2.

Training and implementation. All methods use the same data splits, forecasting tasks, random seeds, base learning rate of 10^{-3}, and 50-epoch training budget. For each run, we select the checkpoint with the lowest validation masked MAE and evaluate it once on the test set. Baseline configurations otherwise follow their official implementations whenever available. SLiDNet is optimized with Adam, with the learning rate held fixed for the first 30 epochs and cosine-decayed over the remaining 20. Complete optimization and implementation settings are provided in Appendix G.3.

3.2 Forecasting Performance

To evaluate the forecasting performance of SLiDNet, we compare it with representative spatio-temporal forecasting baselines across six datasets and three forecasting horizons.

Table 1: Forecasting performance on six benchmarks. Avg. denotes the mean across three horizons. Lower is better; best and second-best results are bolded and underlined, respectively.

Dataset	Method	$H=3$		$H=6$		$H=12$		Avg.	
		MAE↓	RMSE↓	MAE↓	RMSE↓	MAE↓	RMSE↓	MAE↓	RMSE↓
METR-LA	Graph WaveNet	2.4815	4.5619	2.7248	5.2367	3.0838	6.1211	2.7634	5.3066
	AGCRN	2.5765	4.8232	2.7894	5.4018	3.1430	6.3233	2.8363	5.5161
	STG-NCDE	2.9056	5.4240	3.2343	6.2380	3.6112	6.8684	3.2504	6.1768
	STAEformer	2.5865	4.8953	2.7958	5.4859	3.0671	6.2455	2.8165	5.5422
	PDFormer	2.9564	5.3569	2.8364	5.4398	3.2181	6.3839	3.0036	5.7269
	STPGNN	2.6186	4.9174	2.8379	5.5045	3.2597	6.4767	2.9054	5.6329
	STDN	2.6251	5.6344	2.8466	5.8611	3.1533	6.7619	2.8750	6.0858
	SLiDNet	**2.4096**	**4.4156**	**2.6519**	**5.0848**	**2.9914**	**5.9932**	**2.6843**	**5.1645**
PEMS-BAY	Graph WaveNet	1.1302	2.2465	1.3388	2.8359	1.5867	3.5159	1.3519	2.8661
	AGCRN	1.1433	2.2897	1.3441	2.8653	1.5881	3.5451	1.3585	2.9000
	STG-NCDE	1.1990	2.4525	1.4773	3.2793	1.7377	3.8634	1.4713	3.1984
	STAEformer	1.1164	2.2380	1.3194	2.8259	1.5480	3.4730	1.3279	2.8456
	PDFormer	1.3780	2.7333	1.3375	2.8816	1.6181	3.6464	1.4445	3.0871
	STPGNN	1.1732	2.3702	1.4036	3.0216	1.6144	3.5932	1.3971	2.9950
	STDN	1.2633	2.4297	1.3847	2.9521	1.5976	3.6137	1.4152	2.9985
	SLiDNet	**1.0961**	**2.1672**	**1.3038**	**2.7651**	**1.5343**	**3.4360**	**1.3114**	**2.7894**
PEMS03	Graph WaveNet	13.6187	22.7323	13.9066	23.8323	14.4639	24.9987	13.9964	23.8544
	AGCRN	13.9960	23.6183	14.5694	24.7910	15.0240	25.9360	14.5298	24.7818
	STG-NCDE	15.9131	26.3775	15.2794	25.9904	15.3514	26.8946	15.5146	26.4208
	STAEformer	13.7011	22.6103	14.4219	24.2610	15.4545	26.2480	14.5258	24.3731
	PDFormer	19.9911	30.9080	13.9370	23.9780	15.0727	25.7358	16.3336	26.8739
	STPGNN	13.2848	22.8696	13.7543	24.4646	14.8227	25.1060	13.9539	24.1467
	STDN	13.7850	23.2414	14.1345	24.8822	15.2269	26.7016	14.3821	24.9417
	SLiDNet	**12.5636**	**19.6735**	**13.1431**	**20.9393**	**13.9872**	**22.5118**	**13.2313**	**21.0415**
PEMS04	Graph WaveNet	18.7437	29.7249	18.4627	29.4297	19.0349	30.4235	18.7471	29.8594
	AGCRN	18.9356	29.9239	19.3517	30.4848	19.4859	30.9031	19.2577	30.4373
	STG-NCDE	21.2595	32.9774	20.1646	31.5533	19.3818	30.7702	20.2686	31.7670
	STAEformer	18.4254	29.5790	18.8964	30.3051	19.6089	31.2989	18.9769	30.3943
	PDFormer	25.9254	39.0852	19.0830	30.5993	20.1377	32.3130	21.7154	33.9992
	STPGNN	17.5976	**28.2211**	18.0649	**29.1194**	19.1843	30.6316	18.2823	29.3240
	STDN	18.4337	28.9876	18.2082	29.8668	18.6458	31.4547	18.4292	30.1030
	SLiDNet	**17.3745**	28.2445	**17.6884**	29.1225	**18.2778**	**30.0021**	**17.7802**	**29.1230**
PEMS07	Graph WaveNet	19.6858	31.0760	20.2182	32.2330	21.2349	34.1932	20.3796	32.5007
	AGCRN	19.8945	31.2038	20.4465	32.2086	20.8992	33.5840	20.4134	32.3321
	STG-NCDE	22.9786	35.1367	22.2726	34.3764	21.0794	33.7894	22.1102	34.4342
	STAEformer	19.5826	31.4657	20.6068	33.1138	21.7377	34.9850	20.6424	33.1882
	PDFormer	28.2458	41.1485	19.3394	31.4877	20.3647	33.4331	22.6500	35.3564
	STPGNN	19.8832	31.6785	19.9671	31.9948	20.6918	33.5370	20.1807	32.4034
	STDN	20.5462	31.9305	20.2072	31.9873	20.0914	34.1420	20.2816	32.6866
	SLiDNet	**17.5818**	**28.5731**	**18.1275**	**30.0047**	**19.1489**	**32.0093**	**18.2861**	**30.1957**
PEMS08	Graph WaveNet	14.3583	22.2145	14.1968	22.3524	14.6719	23.4574	14.4090	22.6748
	AGCRN	14.7238	22.6602	15.1681	23.5155	15.3892	24.2054	15.0937	23.4604
	STG-NCDE	17.1587	25.8550	15.9737	24.7661	15.8266	24.7641	16.3197	25.1284
	STAEformer	14.2529	22.5925	14.9484	23.9521	15.7628	25.3635	14.9880	23.9694
	PDFormer	21.0766	30.5019	14.2653	22.9592	15.1735	24.6555	16.8385	26.0389
	STPGNN	14.0684	22.0401	14.2549	22.6895	15.1814	24.1603	14.5016	22.9633
	STDN	14.2709	21.8511	13.5797	21.8207	**13.8118**	23.3064	13.8875	22.3261
	SLiDNet	**13.0159**	**20.7427**	**13.2322**	**21.4769**	13.8511	**22.7606**	**13.3664**	**21.6601**

Table 1 shows that SLiDNet achieves the best horizon-averaged performance in both metrics on all six datasets. Across the 18 dataset–horizon settings, it ranks first in 17 MAE comparisons and 16 RMSE comparisons, placing second in the remaining cases. Relative to the strongest baseline for each dataset and metric, the largest reductions in horizon-averaged errors are 9.39% for MAE on PEMS07 and 11.79% for RMSE on PEMS03.

The advantage extends to the longest evaluated horizon, $H = 12$, where SLiDNet leads in RMSE on all six datasets and in MAE on five. The few exceptions across the evaluated settings involve narrow margins: STPGNN achieves slightly better RMSE on PEMS04 at $H = 3$ and $H = 6$, while STDN records a marginally lower MAE on PEMS08 at $H = 12$. These results indicate broad forecasting gains rather than improvements confined to a particular dataset or horizon. The consistency across heterogeneous datasets supports the need to distinguish state formation from evolution-rate modeling under different signal levels and temporal variations. Moreover, the sustained advantage at longer horizons is consistent with the importance of modeling how rapidly node states evolve.

3.3 Ablation Study

We conduct ablation studies on METR-LA, PEMS03, and PEMS08 under horizons $H = 3, 6, 12$, comparing SLiDNet with four variants. **w/o Reference Shrinkage** sets $\kappa = 0$, removing the training-reference shrinkage while retaining window normalization and its inverse transform. **w/o Node-Tau** sets $s_\tau = 0$, disabling node-specific time-scale range modulation. **Static Node-Calibrated Tau** replaces dynamic time-constant selection with static positions within node-calibrated ranges. **Direct Candidate Update** sets $h_t = u_t$, removing rate-controlled interpolation.

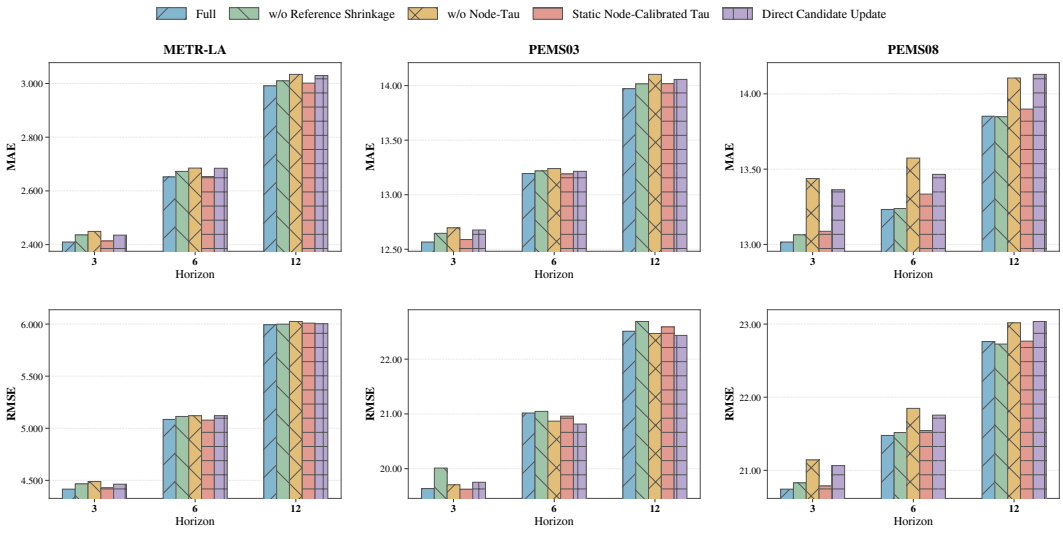

Figure 3: Ablation results on METR-LA, PEMS03, and PEMS08. The top row reports MAE and the bottom row reports RMSE. Each bar shows the mean over five random seeds.

Fig. 3 shows that the proposed designs for state modeling and evolution-rate control improve performance across most datasets, horizons, and metrics. A less consistent pattern is observed for RMSE on PEMS03, where several ablated variants achieve marginally lower RMSE despite higher MAE. Since RMSE places greater weight on large residuals, this discrepancy is consistent with changes in the tail of the error distribution rather than a systematic reversal in overall forecasting quality. Taken together, these results support the effectiveness of the proposed scale-separated design for state formation and evolution-rate control, which allows SLiDNet to separately model what state each node should form and how rapidly it should evolve under heterogeneous spatio-temporal dynamics.

3.4 Parameter Sensitivity

We further examine the sensitivity to κ and s_τ, which control reference shrinkage and node-specific time-scale modulation. As shown in Fig. 4, moderate reference shrinkage is beneficial, although the preferred strength varies across datasets. Increasing s_τ generally improves performance before

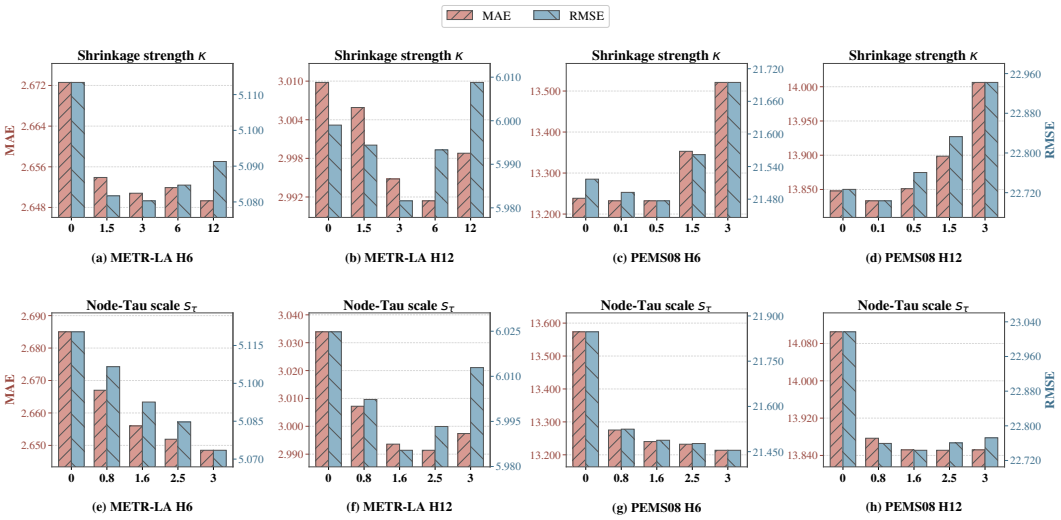

Figure 4: Sensitivity analysis of κ and s_τ on METR-LA and PEMS08.

reaching a stable region, indicating that sufficient node-specific time-scale flexibility is important while the model is not overly sensitive within a moderate range. These trends are consistent with the scale-separated design, in which state formation and evolution-rate modeling benefit from distinct forms of adaptation.

3.5 Efficiency Analysis

Table 2 reports the efficiency comparison on METR-LA and PEMS03 at horizon $H = 12$. SLiDNet uses about 0.5 M parameters and maintains moderate runtime and memory usage while achieving the best MAE and RMSE on both datasets. These results indicate that SLiDNet achieves improved forecasting performance with a moderate computational footprint.

Table 2: Efficiency comparison on METR-LA and PEMS03. Runtime is measured on a fixed synthetic workload of 50 batches with the same tensor shape as the benchmark, memory reports peak GPU memory during training/inference, while MAE and RMSE are taken from Table 1.

Dataset	Model	Params (M)	Train (s)	Infer. (s)	Memory (GiB)	MAE	RMSE
METR-LA	Graph WaveNet	**0.309**	3.62	2.16	2.33 / 1.84	3.0838	6.1211
	AGCRN	0.752	**3.01**	**1.08**	1.95 / **0.18**	3.1430	6.3233
	STG-NCDE	0.375	26.83	7.85	**0.98** / 0.56	3.6112	6.8684
	STAEformer	1.259	10.90	3.49	8.47 / 1.40	3.0671	6.2455
	PDFormer	0.531	13.90	4.63	9.37 / 1.44	3.2181	6.3839
	STPGNN	0.421	3.45	2.05	2.21 / 1.63	3.2597	6.4767
	STDN	5.971	10.94	3.34	9.79 / 1.56	3.1533	6.7619
	SLiDNet	0.496	7.00	2.21	5.10 / 0.77	**2.9914**	**5.9932**
PEMS03	Graph WaveNet	**0.312**	3.99	1.36	2.23 / 1.13	14.4639	24.9987
	AGCRN	0.753	4.78	1.71	3.35 / **0.31**	15.0240	25.9360
	STG-NCDE	0.378	44.09	12.90	**1.66** / 0.96	15.3514	26.8946
	STAEformer	1.404	21.03	6.69	18.36 / 3.65	15.4545	26.2480
	PDFormer	0.531	30.38	9.92	23.29 / 3.73	15.0727	25.7358
	STPGNN	0.516	**3.56**	**1.11**	2.79 / 0.41	14.8227	25.1060
	STDN	6.358	18.58	5.63	16.90 / 2.69	15.2269	26.7016
	SLiDNet	0.509	11.58	3.64	8.81 / 1.34	**13.9872**	**22.5118**

4 Conclusion

We presented SLiDNet, a scale-separated Liquid dynamics network that distinguishes state formation from evolution-rate control. It forms candidate states from scale-stable temporal content and signal context, while adapting node-specific evolution rates to relative temporal variation. Experiments on six spatio-temporal benchmarks, together with ablations and sensitivity analyses, support the effectiveness of the proposed design. Future work will extend SLiDNet to incomplete spatio-temporal observations, leveraging its adaptive Liquid dynamics to propagate latent states across unobserved intervals and jointly recover missing observations and forecast future dynamics.

AI USE STATEMENT

Generative AI tools were used to assist with language editing, rewriting and translation, figure and table presentation, and consistency checks. All such assistance was carried out under the authors' direction and control, and the resulting materials were reviewed and validated by the authors. The research questions, method design, experimental plans, interpretation of results, and scientific conclusions were ultimately determined by the authors.

REPRODUCIBILITY STATEMENT

The model formulation and training objective are specified in Sec. 2. Detailed definitions of the scale-separated information construction, node-calibrated time scales, and spatial prior are provided in the corresponding appendices. Dataset statistics, baseline descriptions, data splits, random seeds, optimization details, hardware environment, and model hyperparameters are provided in Appendix G. The common evaluation protocol is described in Sec. 3.1.

REFERENCES

Lei Bai, Lina Yao, Can Li, Xianzhi Wang, and Can Wang. Adaptive graph convolutional recurrent network for traffic forecasting. In *Advances in Neural Information Processing Systems*, volume 33, pp. 17804–17815, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/ce1aad92b939420fc17005e5461e6f48-Abstract.html.

Lingxiao Cao, Bin Wang, Guiyuan Jiang, Yanwei Yu, and Junyu Dong. Spatiotemporal-aware trend-seasonality decomposition network for traffic flow forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(11):11463–11471, 2025. doi: 10.1609/aaai.v39i11.33247.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K. Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, volume 31, pp. 6571–6583. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/hash/69386f6bb1dfed68692a24c8686939b9-Abstract.html.

Jeongwhan Choi, Hwangyong Choi, Jeehyun Hwang, and Noseong Park. Graph neural controlled differential equations for traffic forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(6):6367–6374, 2022. doi: 10.1609/aaai.v36i6.20587.

Jinliang Deng, Xiusi Chen, Renhe Jiang, Xuan Song, and Ivor W. Tsang. ST-Norm: Spatial and temporal normalization for multi-variate time series forecasting. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 269–278, 2021. doi: 10.1145/3447548.3467330.

Zheng Dong, Renhe Jiang, Haotian Gao, Hangchen Liu, Jinliang Deng, Qingsong Wen, and Xuan Song. Heterogeneity-informed meta-parameter learning for spatiotemporal time series forecasting. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 631–641, 2024. doi: 10.1145/3637528.3671961.

Zheng Fang, Qingqing Long, Guojie Song, and Kunqing Xie. Spatial-temporal graph ODE networks for traffic flow forecasting. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 364–373, 2021. doi: 10.1145/3447548.3467430.

Ramin Hasani, Mathias Lechner, Alexander Amini, Daniela Rus, and Radu Grosu. Liquid time-constant networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(9):7657–7666, 2021. doi: 10.1609/aaai.v35i9.16936.

Ramin Hasani, Mathias Lechner, Alexander Amini, Lucas Liebenwein, Aaron Ray, Max Tschaikowski, Gerald Teschl, and Daniela Rus. Closed-form continuous-time neural networks. *Nature Machine Intelligence*, 4(11):992–1003, 2022. doi: 10.1038/s42256-022-00556-7.

Jiawei Jiang, Chengkai Han, Wayne Xin Zhao, and Jingyuan Wang. PDFormer: Propagation delay-aware dynamic long-range Transformer for traffic flow prediction. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(4):4365–4373, 2023. doi: 10.1609/aaai.v37i4.25556.

Patrick Kidger, James Morrill, James Foster, and Terry Lyons. Neural controlled differential equations for irregular time series. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 6696–6707. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/hash/4a5876b450b45371f6cfe5047ac8cd45-Abstract.html.

Taesung Kim, Jinhee Kim, Yunwon Tae, Cheonbok Park, Jang-Ho Choi, and Jaegul Choo. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=cGDAkQo1C0p.

Weiyang Kong, Ziyu Guo, and Yubao Liu. Spatio-temporal pivotal graph neural networks for traffic flow forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(8):8627–8635, 2024. doi: 10.1609/aaai.v38i8.28707.

Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=SJiHXGWAZ.

Hangchen Liu, Zheng Dong, Renhe Jiang, Jiewen Deng, Jinliang Deng, Quanjun Chen, and Xuan Song. Spatio-temporal adaptive embedding makes vanilla Transformer SOTA for traffic forecasting. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pp. 4125–4129, 2023a. doi: 10.1145/3583780.3615160.

Zhiding Liu, Mingyue Cheng, Zhi Li, Zhenya Huang, Qi Liu, Yanhu Xie, and Enhong Chen. Adaptive normalization for non-stationary time series forecasting: A temporal slice perspective. In *Advances in Neural Information Processing Systems*, volume 36, pp. 14273–14292. Curran Associates, Inc., 2023b. doi: 10.52202/075280-0628.

Jelena Simeunovic, Baptiste Schubnel, Pierre-Jean Alet, and Rafael E. Carrillo. Spatio-temporal graph neural networks for multi-site PV power forecasting. *IEEE Transactions on Sustainable Energy*, 13(2):1210–1220, 2022. doi: 10.1109/TSTE.2021.3125200.

Chao Song, Youfang Lin, Shengnan Guo, and Huaiyu Wan. Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(01):914–921, 2020. doi: 10.1609/aaai.v34i01.5438.

Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. Graph WaveNet for deep spatial-temporal graph modeling. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, pp. 1907–1913, 2019. doi: 10.24963/ijcai.2019/264.

Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, Xiaojun Chang, and Chengqi Zhang. Connecting the dots: Multivariate time series forecasting with graph neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 753–763, 2020. doi: 10.1145/3394486.3403118.

Bing Yu, Haoteng Yin, and Zhanxing Zhu. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, pp. 3634–3640, 2018. doi: 10.24963/ijcai.2018/505.

A Related Work

We situate SLiDNet within graph-based forecasting, heterogeneous representations, and continuous-time dynamics. These perspectives motivate its focus on the information used to form node states and control their evolution rates.

Graph-based spatio-temporal forecasting. Graph forecasting methods combine temporal modeling with information exchange between related locations. DCRNN integrates bidirectional diffusion with recurrent sequence modeling, while STGCN combines graph and temporal convolutions (Li et al., 2018; Yu et al., 2018). Beyond a predefined graph, Graph WaveNet learns an adaptive adjacency from node embeddings and combines diffusion graph convolution with dilated causal convolution (Wu et al., 2019). MTGNN learns relations between variables and uses mix-hop propagation with temporal convolution (Wu et al., 2020). STPGNN emphasizes highly connected pivotal nodes when organizing spatial interactions (Kong et al., 2024). These approaches establish spatial structure and message propagation as tools for forecasting. SLiDNet builds on this foundation through physical and adaptive diffusion supports. Its temporal representation is formed by a node-wise Liquid recurrence before graph messages are incorporated through a gated correction. A training-derived correlation prior provides an anchor for the adaptive support within this stage.

Heterogeneity and adaptive representations. Heterogeneity can be represented through node-dependent parameters, learned embeddings, or context-dependent interactions. AGCRN uses node embeddings for both adaptive graph learning and node-adaptive parameter generation (Bai et al., 2020). HimNet uses meta-parameter learning to capture spatial, temporal, and spatio-temporal heterogeneity (Dong et al., 2024). In attention-based models, STAEformer introduces spatio-temporal adaptive embeddings into a Transformer, while PDFormer combines geographic and semantic attention masks with propagation-delay modeling (Liu et al., 2023a; Jiang et al., 2023). These methods express heterogeneity through the representations and transformations used for forecasting. SLiDNet focuses on how heterogeneous information assumes distinct roles within the recurrent transition. Persistent node differences calibrate admissible time-scale ranges through learned node embeddings. Window-level signal context conditions candidate formation, while current temporal variation helps select a time constant within those ranges. This organization distinguishes persistent node characteristics from window-specific conditions and temporal change, while the recurrent state connects their effects during evolution.

Continuous-time and Liquid dynamics. Continuous-time models parameterize latent-state evolution through differential equations. Neural ODEs parameterize hidden-state derivatives, while neural controlled differential equations use interpolated observations to drive latent dynamics (Chen et al., 2018; Kidger et al., 2020). For spatio-temporal forecasting, STGODE develops a tensor-based graph ordinary differential equation, while STG-NCDE couples temporal and spatial neural controlled differential equations (Fang et al., 2021; Choi et al., 2022). Liquid Time-Constant networks introduce input- and state-dependent effective time constants into recurrent dynamics (Hasani et al., 2021). Closed-form continuous-time networks derive an approximate closed-form solution to Liquid Time-Constant dynamics, enabling temporal modeling without numerical integration (Hasani et al., 2022). These works provide mechanisms for continuous-time evolution and adaptive response rates. Building on this perspective, SLiDNet focuses on the conditioning information that determines the update target and evolution rate. Its Liquid update is a first-order relaxation transition evaluated at each observed step. Normalized temporal content and window-level signal context condition the candidate state. Scale-normalized temporal variation and the layer-normalized recurrent hidden state select a time constant within learned node-specific bounds. The hidden state also enters candidate formation, connecting both conditioning paths through recurrence. The resulting transition distinguishes what state to form from how rapidly the recurrent state should approach it.

B Signal Level and Temporal Variation Analysis

The analysis in Fig. 1 is conducted using only the training split of METR-LA. For node i, its typical signal level is defined as

$$L_i = \mathrm{median}_t \, x_{i,t}, \tag{13}$$

where $x_{i,t}$ denotes the observed speed of node i at time t.

Its typical temporal variation is represented by

$$V_i = \mathrm{median}_t \frac{|x_{i,t} - x_{i,t-1}|}{\mathrm{MAD}(x_i) + \epsilon}, \tag{14}$$

where $\mathrm{MAD}(x_i)$ denotes the median absolute deviation of the observations at node i, and ϵ is a small constant for numerical stability. The statistic V_i is used only as a node-level proxy for the motivation analysis. It is distinct from the variation representation P used by SLiDNet, which incorporates both first- and second-order temporal changes.

C Notation

Table 3 summarizes the main notation used throughout the paper.

Table 3: Main notation used in SLiDNet.

Symbol	Description
N	Number of graph nodes.
C	Number of input features.
T	Length of the historical input window.
H	Forecasting horizon.
d	Hidden dimension of the Liquid state.
$X_{t-T+1:t}$	Historical input sequence.
$\widehat{Y}_{t+1:t+H}$	Predicted future sequence.
Q^{level}	Window-level signal context used for state formation.
P	Scale-normalized temporal-variation representation.
κ	Reference-shrinkage strength in ShrinkNorm.
$X^{(b)}$	Backbone features after SLiD block b.
$Z^{(b)}$	Premixed features of SLiD block b.
A_{adp}	Adaptive adjacency learned from node embeddings.
A_{prior}	Sparse correlation prior constructed from training data.
A_{eff}	Effective adjacency combining adaptive and prior relations.
$H_L^{(b)}$	Liquid features produced by block b.
$H_D^{(b)}$	Graph-corrected features produced by block b.
s_τ	Modulation strength for node-specific time-scale calibration.
$h_{i,t}^{(b)},\ u_{i,t}^{(b)}$	Liquid hidden state and candidate state of node i at step t in block b.
$\tau_{i,t}^{(b)},\ \alpha_{i,t}^{(b)}$	Liquid time constant and corresponding update coefficient.
$\underline{\tau}_i^{(b)},\ \overline{\tau}_i^{(b)}$	Node-calibrated lower and upper time-scale bounds.

D Details of Scale-Separated Information Construction

We provide additional details for the information construction in Sec. 2.3. All quantities below are defined after dataset-level standardization; pre-ShrinkNorm therefore refers to the coordinates before the internal ShrinkNorm transform rather than the original physical scale. Only the forecasting variable undergoes this internal transform, while auxiliary input features remain unchanged.

D.1 ShrinkNorm and Signal Context

Shrinkage statistics. Let $\{X^{(m)}\}_{m=1}^{M}$ denote the M training input windows for one target variable, where $X^{(m)} = [x_{i,t}^{(m)}] \in \mathbb{R}^{N\times T}$. For each node i, the training reference statistics are computed

over all sample–time positions:

$$\mu_i^{\mathrm{tr}} = \frac{1}{MT}\sum_{m=1}^{M}\sum_{t=1}^{T} x_{i,t}^{(m)}, \qquad v_i^{\mathrm{tr}} = \frac{1}{MT}\sum_{m=1}^{M}\sum_{t=1}^{T}\left(x_{i,t}^{(m)} - \mu_i^{\mathrm{tr}}\right)^2. \tag{15}$$

For a current input window $X = [x_{i,t}]$, the window statistics are

$$\mu_i^{w} = \frac{1}{T}\sum_{t=1}^{T} x_{i,t}, \qquad v_i^{w} = \frac{1}{T}\sum_{t=1}^{T}\left(x_{i,t} - \mu_i^{w}\right)^2. \tag{16}$$

Both variances use the population form. The training reference statistics are estimated once from the training split and remain fixed, whereas (μ_i^w, v_i^w) are recomputed for each input window. Predictions in the internal coordinates are mapped back using the same active statistics as $c_i + s_i\hat{y}_{i,k}^{\mathrm{int}}$, where k indexes the forecast step.

Signal context. The signal context in Eq. 2 is computed from the latest pre-ShrinkNorm observation and the active scale s_i.

D.2 Temporal-Variation Construction

For a finite sequence z, we define the median absolute deviation as

$$\mathrm{MAD}(z) = \mathrm{median}_k\left|z_k - \mathrm{median}_j(z_j)\right|. \tag{17}$$

For node i, let $x_i = (x_{i,1}, \ldots, x_{i,T})$. The valid first- and second-order temporal differences are

$$\Delta x_{i,t} = x_{i,t} - x_{i,t-1}, \qquad t = 2, \ldots, T, \tag{18}$$

$$\Delta^2 x_{i,t} = x_{i,t} - 2x_{i,t-1} + x_{i,t-2}, \qquad t = 3, \ldots, T. \tag{19}$$

Accordingly, $\mathrm{MAD}(\Delta x_i)$ in Eq. 3 is computed over the $T-1$ valid first-order differences before boundary padding. The first-order difference is zero-padded at the first input step, while the second-order difference is zero-padded at the first two steps. If $r_i \le \epsilon_p$, the variation representation is set to zero for numerical stability.

E Node-Calibrated Time Scales

The first $\lfloor d/2 \rfloor$ hidden channels use the base range $[0.1, 2.0]$, and the remaining channels use $[2.0, 12.0]$. For each channel j, the lower endpoint defines $\tau_{0,j}$, and the difference between endpoints defines $\Delta\tau_{0,j}$. These are the ranges before node calibration; the calibrated fast and slow ranges need not remain disjoint. Time constants are measured in sampling-step units, with $\Delta t = 1$.

Each block maintains its own node embeddings $e_i^{(b)} \in \mathbb{R}^{16}$ and affine projection $\phi_{\mathrm{nt}}^{(b)} : \mathbb{R}^{16} \to \mathbb{R}^{2d}$. The projected output is passed through $\tanh$ and split into $\eta_i^{(b)}$ and $\xi_i^{(b)}$, which modulate the base lower bound and span, respectively, as defined in Eq. (8). These embeddings are separate from those used to construct the adaptive adjacency. An elementwise floor of 10^{-4} is applied when evaluating inverse time constants for the state update.

F Spatial Prior Construction

For each dataset and (T, H) setting, we construct A_{prior} using only the training input windows $\{X^{(m)}\}_{m=1}^{M}$. Pairwise Pearson correlations are computed from the observations at the final input position:

$$R_{ij} = \begin{cases} \max\Big(0,\ \mathrm{corr}\left(\{x_{i,T}^{(m)}\}_{m=1}^{M}, \{x_{j,T}^{(m)}\}_{m=1}^{M}\right)\Big), & i \neq j, \\ 0, & i = j. \end{cases} \tag{20}$$

Non-finite observations are replaced by zero before correlation estimation, and non-finite correlations are set to zero.

Let $\mathcal{N}_i$ denote the indices of up to eight largest positive entries in row i of R. We define

$$\widetilde{A}_{\mathrm{prior},ij} = \begin{cases} 1, & i = j, \\ R_{ij}, & j \in \mathcal{N}_i, \\ 0, & \text{otherwise}, \end{cases} \qquad A_{\mathrm{prior},ij} = \frac{\widetilde{A}_{\mathrm{prior},ij}}{\sum_{k=1}^{N} \widetilde{A}_{\mathrm{prior},ik}}. \tag{21}$$

Neighbor selection is performed independently for each row without symmetrization. The resulting A_{prior} is shared across random seeds and all SLiD blocks and remains fixed during training and evaluation. Neither validation nor test observations are used in its construction. The prior contributes to the model only through the mixture in Eq. (10).

G Additional Experimental Details

G.1 Dataset statistics.

Table 4 summarizes the forecasting variable, observation period, and graph size of each benchmark. All datasets are sampled at five-minute intervals.

Table 4: Statistics of the six spatio-temporal forecasting benchmarks.

Dataset	Forecast variable	Observation period	# Nodes
METR-LA	Speed	Mar. 1–Jun. 30, 2012	207
PEMS-BAY	Speed	Jan. 1–May 31, 2017	325
PEMS03	Flow	Sep. 1–Nov. 30, 2018	358
PEMS04	Flow	Jan. 1–Feb. 28, 2018	307
PEMS07	Flow	May 1–Aug. 31, 2017	883
PEMS08	Flow	Jul. 1–Aug. 31, 2016	170

G.2 Baseline Descriptions

We compare SLiDNet with seven representative spatio-temporal forecasting baselines spanning convolutional, recurrent, continuous-time, Transformer-based, pivotal-node graph, and decomposition-based architectures. Their main modeling strategies are summarized below.

- **Graph WaveNet** (Wu et al., 2019) combines stacked dilated causal convolutions with diffusion graph convolution. It further learns an adaptive adjacency matrix from node embeddings to capture latent spatial relations.
- **AGCRN** (Bai et al., 2020) integrates node-adaptive parameter generation with data-adaptive graph learning in a recurrent graph-convolutional architecture, enabling node-specific temporal modeling without requiring a predefined graph.
- **STG-NCDE** (Choi et al., 2022) uses two neural controlled differential equations to model temporal and spatial dynamics, respectively, and couples them within a continuous-time forecasting framework.
- **PDFormer** (Jiang et al., 2023) employs dynamic spatial self-attention constrained by geographic and semantic graph masks to capture local and long-range dependencies. A delay-aware feature transformation further represents propagation delays across locations.
- **STAEformer** (Liu et al., 2023a) augments a vanilla Transformer with learned spatio-temporal adaptive embeddings that encode intrinsic spatial and temporal relations together with chronological information.
- **STPGNN** (Kong et al., 2024) identifies highly connected pivotal nodes and applies pivotal graph convolution to model dependencies centered on them. Its parallel design separately processes pivotal and non-pivotal nodes.

- **STDN** (Cao et al., 2025) learns dynamic graph structures and spatio-temporal embeddings, and then decomposes node representations into trend-cyclical and seasonal components for encoder–decoder forecasting.

G.3 Training and Implementation Details

The datasets are split into training, validation, and test sets with a ratio of 7:1:2. The forecasting variable is standardized using the mean and standard deviation computed from the training set. We evaluate $(T, H) \in \{(3, 3), (6, 6), (12, 12)\}$, corresponding to 15-, 30-, and 60-minute histories and forecasts.

All methods use the same data splits, forecasting tasks, random seeds, base learning rate of 10^{-3}, and 50 training epochs. Baseline-specific model configurations and training procedures follow their official implementations. Each configuration is evaluated with five random seeds $\{0, 1, 2, 3, 4\}$. The checkpoint with the lowest validation masked MAE is selected for test evaluation.

All experiments are run in FP32 precision using PyTorch 2.4.0 with cuDNN autotuning enabled on NVIDIA A800 80GB GPUs.

SLiDNet is optimized with Adam using a batch size of 64, weight decay 10^{-4}, and dropout 0.3. The learning rate is kept at 10^{-3} for the first 30 epochs and cosine-decayed over the remaining 20 epochs. The minimum learning-rate ratio is 0.10 for $H \in \{3, 6\}$ and 0.02 for $H = 12$.

For the model architecture, SLiDNet uses four stacked SLiD blocks with 64 residual channels, 256 skip channels, a temporal kernel size of 2, and a prediction head with 512 end channels. ShrinkNorm uses $\epsilon_s = 10^{-5}$, with $\kappa = 6$ for METR-LA and PEMS-BAY and $\kappa = 0.5$ for PEMS03, PEMS04, PEMS07, and PEMS08. We set $\epsilon_L = 10^{-5}$, $\gamma_d = 1$, and $\epsilon_p = 10^{-5}$. For time-scale calibration, the node embedding dimension is 16; the first $\lfloor d/2 \rfloor$ hidden channels use the base range $[0.1, 2.0]$, while the remaining channels use $[2.0, 12.0]$. We set $s_\tau = 2.5$ and $\rho = 0.05$. The spatial prior is constructed as described in Appendix F.