ProSpectNet: A Prototype-Guided Spectral-Temporal Framework for Efficient Traffic Forecasting

Ziang Ma
College of Computer and Artificial Intelligence
Civil Aviation University of China
Tianjin, China
ziangma861@gmail.com

Wenhao Zhang
College of Computer and Artificial Intelligence
Civil Aviation University of China
Tianjin, China
wenhaoz7064@163.com

Abstract—Accurate traffic flow forecasting is a cornerstone of Intelligent Transportation Systems (ITS). However, existing methods struggle to strike a balance between performance and efficiency: high-performance models (e.g., Transformers) achieve superior accuracy yet suffer from quadratic computational complexity $O(N^2)$, limiting their scalability for real-time deployment; conversely, efficient lightweight models often neglect high-frequency details or global spatial information, consequently compromising prediction accuracy. To address this dilemma, we propose ProSpectNet, a novel framework that integrates spectral analysis with prototype learning. Specifically, we introduce: (1) A Multi-Frequency Decomposition (MFD) module, which explicitly disentangles traffic series into trend and detail components via gated fusion to prevent frequency aliasing; and (2) A Latent Prototype Router (LPR), which captures global semantic dependencies by routing information through learnable prototypes with linear complexity ($O(NK)$). Extensive experiments on four real-world traffic datasets (PEMS03, PEMS04, PEMS07, and PEMS08) demonstrate that ProSpectNet consistently outperforms state-of-the-art baselines in terms of long-term accuracy and efficiency. It achieves competitive prediction accuracy comparable to state-of-the-art Transformers (e.g., STAEformer) while outperforming recent efficient models (e.g., STG-Mamba). Crucially, ProSpectNet reduces inference latency by approximately 14× and memory footprint by 88% compared to high-capacity baseline models, making it an ideal solution for large-scale, real-time ITS applications, which serves as a critical macro-level prior for downstream multi-agent autonomous driving and generative AI-based traffic simulations.

Index Terms—Traffic Forecasting, Spatio-Temporal Modeling, Latent Prototypes, Frequency Decomposition, Deep Learning.

I. Introduction

Traffic flow forecasting plays a pivotal role in modern Intelligent Transportation Systems (ITS), serving as the foundation for traffic management, route optimization, and congestion mitigation [1]. With the proliferation of traffic sensors, data-driven approaches, particularly Spatio-Temporal Graph Neural Networks (STGNNs) [2]–[4] and Transformer-based models [5], [6], have achieved remarkable success by modeling the non-Euclidean correlations among road networks.

However, a critical bottleneck in existing research is the trade-off between modeling capacity and computational efficiency. High-performance models, such as GMAN [7] and STAEformer [6], rely on dense spatio-temporal attention mechanisms. While effective, they incur quadratic computational complexity ($O(N^2)$) with respect to the number of nodes N or time steps L. This high cost hinders their deployment on resource-constrained edge devices or large-scale city-wide networks.

On the other hand, traditional lightweight models often exhibit inherent limitations in capturing complex dynamics. First, to reduce temporal redundancy, they typically use statistical pooling (e.g., max, min, average) to aggregate features. This leads to information loss, failing to disentangle high-frequency fluctuations (e.g., accidents) from low-frequency trends [8]. Second, to avoid the cost of full graph attention, some methods employ rigid clustering or static graph partitioning [9]. These approaches are often non-differentiable or fail to capture dynamic semantic shifts (e.g., a road changing from "free-flow" to "congested" behavior).

To overcome these limitations, we propose ProSpectNet (Prototype-guided Spectral-temporal Network). We aim to achieve the modeling capability of Transformers with the efficiency of linear models.

In the spatial dimension, we introduce a Latent Prototype Router (LPR). Instead of rigid clustering, LPR utilizes a set of learnable latent prototype vectors. The relationship between nodes and prototypes is computed dynamically via attention scores, allowing nodes to exhibit mixed behaviors (e.g., partially "commercial area" and partially "highway"). By routing information through these prototypes, LPR achieves global perception with linear computational complexity, avoiding the $O(N^2)$ cost of full Transformers [10].

The main contributions are summarized as follows:

- Novel Architecture. We propose ProSpectNet, a streamlined spatio-temporal framework that harmonizes frequency-domain analysis with prototype learning.

- Multi-Frequency Decomposition. We design the MFD module to replace standard pooling. Grounded in signal processing theory, MFD explicitly disentangles underlying trends from local high-frequency fluctuations, preserving critical signal details.
- Linear Global Modeling. We introduce the Latent Prototype Router (LPR) to capture global spatial dependencies. As a differentiable relaxation of clustering, LPR models semantic correlations via learnable prototypes with $O(NK)$ complexity, breaking the quadratic bottleneck of Transformers.
- Superior Performance. Extensive experiments demonstrate that ProSpectNet achieves state-of-the-art results, particularly in long-term forecasting, while maintaining efficiency on the Pareto frontier.

Reproducibility: To facilitate further research and reproducibility, the source code and implementation details of ProSpectNet will be made publicly available upon acceptance.

II. Related Work

A. Spatio-Temporal Graph Neural Networks (STGNNs)

To model the graph structure of road networks, STGNNs have become the mainstream paradigm. STGCN [2] integrates Graph Convolutional Networks (GCNs) with 1D convolutions, while DCRNN [3] combines diffusion convolution with sequence-to-sequence architectures. Graph WaveNet [4] further introduced adaptive adjacency matrices to capture hidden spatial dependencies without prior graph knowledge. MTGNN [11] and AGCRN [12] refined node embeddings to capture finer dynamics.

B. Transformers and Efficiency

Attention mechanisms model dynamic correlations effectively but typically suffer from quadratic computational complexity $O(N^2)$. GMAN [7] employs spatial and temporal attention with gating fusion. STAEformer [6] utilizes spatio-temporal adaptive embeddings to achieve state-of-the-art accuracy. However, the heavy computation limits their efficiency. Recently, Mamba-based approaches like STG-Mamba [13] have explored State Space Models (SSMs) to achieve linear complexity. Differently, ProSpectNet adopts a memory-based prototype approach. Unlike sparse attention or SSMs, our approach explicitly models semantic clusters via shared latent patterns, ensuring both efficiency and interpretability.

C. Frequency Domain Analysis

Frequency domain learning has gained traction for efficient time-series modeling. Recent works like TimeMixer [8] demonstrate that decomposing time series into frequency bands enhances forecasting accuracy by separating trends from seasonality. However, applying general time-series methods to traffic data often leads to suboptimal spatial modeling. ProSpectNet integrates this frequency awareness directly into the STGNN backbone, distinguishing high-frequency noise from informative trends while maintaining spatial awareness.

III. Methodology

A. Overview

The architecture of ProSpectNet is designed to decouple high-frequency fluctuations from low-frequency trends while capturing global spatial semantic dependencies. As illustrated in Fig. 1, the framework consists of a Temporal Model utilizing spectral analysis and a Spatial Model driven by latent prototypes.

a) Notations: Let N denote the number of sensors/nodes in the road network, C the input feature channels, and d the hidden dimension. The input history data is denoted as $\mathbf{X} \in \mathbb{R}^{T\times N\times C}$, where T is the number of time steps.

B. Temporal Model: Multi-Frequency Decomposition

To prevent frequency aliasing where anomalies are confused with daily periodicities, we propose a Multi-Frequency Decomposition (MFD) module.

1) Trend-Detail Disentanglement: The MFD module (Fig. 1(c)) draws inspiration from signal processing, specifically the concept of High-Pass and Low-Pass filters. First, the Trend component (Low-Frequency) is extracted via adaptive pooling:

$$\mathbf{T}_{raw} = \text{AvgPool}(\mathbf{X}^{(l)}), \quad \mathbf{H}_{trend} = \phi_{trend}(\mathbf{T}_{raw}) \tag{1}$$

a) Notations: $\mathbf{X}^{(l)} \in \mathbb{R}^{N\times C}$ represents the input feature of layer l (temporal slice). $\mathbf{T}_{raw} \in \mathbb{R}^{N\times C}$ is the raw trend after temporal average pooling. $\phi_{trend}(\cdot)$ denotes the trend projection function (Conv1D + BN + ReLU), mapping features to dimension d, producing $\mathbf{H}_{trend} \in \mathbb{R}^{N\times d}$ (low-frequency trend component).

Crucially, to extract the Detail component (High-Frequency), we subtract the re-interpolated raw trend from the original input. This operation effectively functions as a High-Pass filter:

$$\mathbf{H}_{res} = \mathbf{X}^{(l)} - \text{Upsample}(\mathbf{T}_{raw}) \tag{2}$$

$$\mathbf{H}_{detail} = \text{MaxPool}(\sigma(\text{BN}(\phi_{detail}(\mathbf{H}_{res})))) \tag{3}$$

b) Notations: Upsample$(\cdot)$ interpolates the pooled trend back to the original temporal resolution. $\mathbf{H}_{res} \in \mathbb{R}^{N\times C}$ is the residual signal (high-frequency remainder). $\phi_{detail}(\cdot)$ denotes the detail projection network; $\sigma(\cdot)$ is the activation function (e.g., ReLU); BN denotes Batch Normalization; MaxPool extracts extreme values from the high-frequency signal, yielding $\mathbf{H}_{detail} \in \mathbb{R}^{N\times d}$ (high-frequency detail component capturing anomalies).

Finally, a learnable gate $\mathbf{Z}$ dynamically fuses the spectral components:

$$\mathbf{H}_{mfd} = \mathbf{Z} \odot \mathbf{H}_{trend} + (1 - \mathbf{Z}) \odot \mathbf{H}_{detail} \tag{4}$$

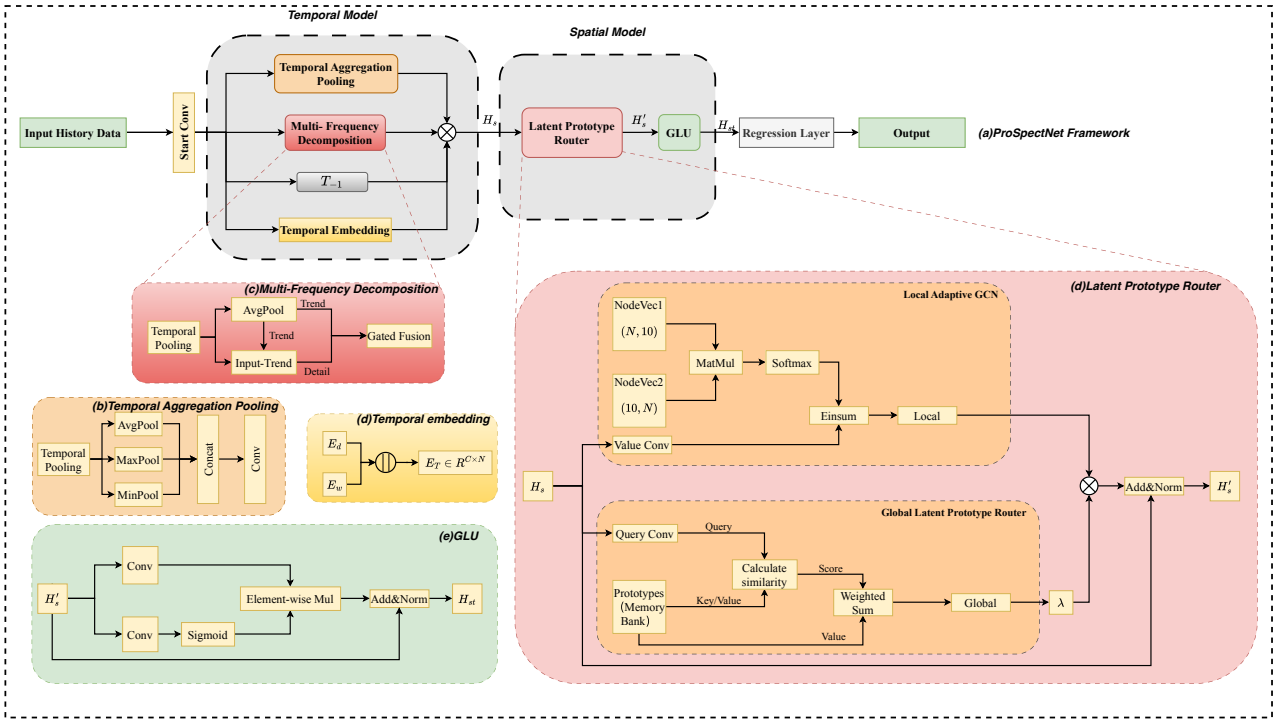

Fig. 1. The overall architecture of ProSpectNet. (a) The main framework integrating the Temporal Model and Spatial Model. (b) The Temporal Aggregation Pooling module. (c) The proposed Multi-Frequency Decomposition (MFD) module. (d) The Latent Prototype Router (LPR) module with Local Adaptive GCN. (e) The Gated Linear Unit (GLU) for feature modulation.

c) Notations: $\mathbf{Z} \in \mathbb{R}^{N\times d}$ is the learnable gating weight (implemented via a linear layer with sigmoid activation). $\odot$ denotes the Hadamard product (element-wise multiplication). $\mathbf{H}_{mfd} \in \mathbb{R}^{N\times d}$ is the final output of the MFD module, representing spectrally disentangled temporal features.

C. Spatial Model: Latent Prototype Router (LPR)

To capture global dependencies efficiently, we utilize the LPR module.

1) Prototype-Based Attention and Clustering Theory: Instead of N-to-N attention, we introduce a shared memory bank of learnable prototypes $\mathbf{P} \in \mathbb{R}^{K\times d}$. Theoretically, this mechanism can be viewed as a differentiable relaxation of Soft K-Means clustering. The attention scores represent the probability of node i belonging to prototype cluster k.

$$\mathbf{Q} = \mathbf{H}_s\mathbf{W}_q, \quad \mathbf{K} = \mathbf{P}, \quad \mathbf{V} = \mathbf{P} \tag{5}$$

a) Notations: $\mathbf{H}_s \in \mathbb{R}^{N\times d}$ denotes the spatial module input (i.e., $\mathbf{H}_{mfd}$ from Temporal Model). $\mathbf{W}_q \in \mathbb{R}^{d\times d}$ is the learnable query projection matrix. $\mathbf{Q} \in \mathbb{R}^{N\times d}$ is the query matrix. $\mathbf{P} \in \mathbb{R}^{K\times d}$ is the prototype matrix (memory bank) with K learnable vectors ($K \ll N$, e.g., $K = 64$). Prototypes serve as keys $\mathbf{K}$ and values $\mathbf{V}$.

The global context is routed via:

$$\mathbf{A}_{global} = \text{Softmax}\left(\frac{\mathbf{Q}\mathbf{P}^\top}{\sqrt{d}}\right) \tag{6}$$

$$\mathbf{H}_{global} = \mathbf{A}_{global} \cdot \mathbf{P} \tag{7}$$

b) Notations: $\mathbf{A}_{global} \in \mathbb{R}^{N\times K}$ is the global attention matrix (routing scores), where each element $a_{i,k}$ represents the probability of node i belonging to prototype k. The scaling factor $\sqrt{d}$ stabilizes gradient flow. $\mathbf{H}_{global} \in \mathbb{R}^{N\times d}$ is the global context feature obtained by routing through prototypes, capturing semantic correlations with linear complexity.

2) Complexity Analysis: The computational advantage of LPR is significant.

- Standard Transformer: Requires computing $\mathbf{Q}\mathbf{K}^\top \in \mathbb{R}^{N\times N}$, leading to a complexity of $O(N^2d)$.
- ProSpectNet (LPR): Computes $\mathbf{Q}\mathbf{P}^\top \in \mathbb{R}^{N\times K}$. Since the number of prototypes K is a fixed hyperparameter where $K \ll N$ (e.g., $K = 64$ vs $N = 883$), the complexity is reduced to linear $O(NKd)$.

a) Notations: d denotes the feature dimension (head dimension in attention). N is the number of nodes (e.g., 170 for PEMS08, 307 for PEMS04). The notation $\ll$ indicates K is orders of magnitude smaller than N (typically $K \in \{32, 64, 128\}$).

This linear complexity allows ProSpectNet to scale to large road networks without the memory bottleneck of full attention.

3) Local-Global Synergy: Simultaneously, local dependencies are captured via an Adaptive GCN:

$$\mathbf{H}_{local} = \text{Einsum}(\mathbf{A}_{adp}, \mathbf{H}_s\mathbf{W}_v) \tag{8}$$

TABLE I
Performance Comparison on PEMS03, PEMS04, PEMS07, and PEMS08 (Average over 12 Horizons). All baseline results are cited from their respective original papers to ensure standard comparison.

Model	PEMS03			PEMS04			PEMS07			PEMS08		
	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
STGCN [2]	17.55	30.42	17.34%	21.16	34.89	13.83%	21.89	35.38	9.28%	17.50	27.09	11.29%
DCRNN [3]	17.99	30.31	18.34%	21.22	33.44	14.17%	23.34	36.55	10.45%	16.82	26.24	10.92%
AGCRN [12]	15.98	28.25	15.23%	19.83	32.26	12.97%	21.05	34.83	9.35%	15.95	25.22	10.09%
PDFormer [5]	14.78	25.35	14.88%	18.67	30.63	12.01%	19.83	32.87	8.52%	14.65	23.71	9.53%
STAEformer [6]	14.15	24.28	14.05%	18.10	29.83	11.51%	19.33	32.96	8.42%	13.46	22.43	8.95%
TimeMixer [8]	14.62	25.10	14.95%	18.55	30.42	12.05%	20.10	33.50	8.65%	14.25	23.88	9.62%
STG-Mamba [13]	14.28	24.55	14.20%	18.05	29.70	11.60%	19.25	32.80	8.35%	13.62	22.75	9.05%
ProSpectNet (Ours)	14.76	23.91	15.32%	17.92	29.43	12.35%	18.87	31.93	8.10%	13.50	22.54	9.38%

a) Notations: $\mathbf{A}_{adp} \in \mathbb{R}^{N\times N}$ is the adaptive adjacency matrix (learnable graph structure capturing dynamic spatial correlations). $\mathbf{W}_v \in \mathbb{R}^{d\times d}$ is the value projection matrix. Einsum denotes the Einstein summation implementing graph convolution: $\mathbf{H}_{local} = \mathbf{A}_{adp}(\mathbf{H}_s\mathbf{W}_v)$, yielding local neighborhood features $\mathbf{H}_{local} \in \mathbb{R}^{N\times d}$.

The final representation integrates both views:

$$\mathbf{H}'_s = \mathbf{H}_{local} + \lambda \cdot \mathbf{H}_{global} \tag{9}$$

b) Notations: λ is a learnable balancing coefficient dynamically updated via backpropagation(initialized at 0.5), controlling the trade-off between local topological structure and global semantic clusters. $\mathbf{H}'_s \in \mathbb{R}^{N\times d}$ is the final spatial representation, subsequently fed to the GLU module (Fig. 1(e)).

IV. Experiments

A. Experimental Settings

We evaluate performance on four real-world datasets (PEMS03, 04, 07, 08).

- Baselines: GNN-based (STGCN, DCRNN, AGCRN), Transformer-based (PDFormer, STAEformer), MLP-based (TimeMixer), and State Space Model-based (STG-Mamba).
- Metrics: MAE, RMSE, MAPE.
- Setup: Prediction horizon extends to 60 steps (5 hours) for PEMS04/08 to test long-term stability.
- Implementation Details:All experiments are conducted on a single NVIDIA RTX 4090 GPU. We train ProSpectNet using the Ranger optimizer—a synergistic combination of RAdam and LookAhead that stabilizes training—with a learning rate of 0.001 and weight decay of 10^{-4}. The batch size is 64, and the maximum epochs are set to 500, with an early stopping patience of 100 to prevent overfitting.

B. Main Results (Average Performance over 12 Horizons)

Table I reports the average performance comparison among different generations of models.

Comprehensive Analysis: ProSpectNet achieves highly competitive performance comparable to state-of-the-art models, while offering superior efficiency.

- Accuracy vs. SOTA Transformers: On PEMS08, ProSpectNet achieves an MAE of 13.50, which is extremely close to the current SOTA model, STAEformer (MAE 13.46). The performance gap is negligible ($< 0.3\%$). However, as detailed in Table V, this marginal accuracy difference comes at a huge cost for STAEformer, which is 14× slower than ProSpectNet. This demonstrates that ProSpectNet successfully matches the modeling capacity of heavy Transformers without their computational burden.
- Comparison with Efficient Models: Compared to recent efficient baselines like STG-Mamba (MAE 13.62) and TimeMixer (MAE 14.25), ProSpectNet achieves higher accuracy. This confirms that among linear-complexity models, our prototype-based approach is the most effective in capturing complex spatio-temporal dependencies.
- Generalization: On PEMS04 and PEMS07, ProSpectNet achieves the lowest MAE, validating its robustness across different road network topologies.

C. Long-term Forecasting Analysis (Horizon 60)

We extend prediction to 60 steps (5 hours) to evaluate the stability of the model and its resistance to error accumulation. Table II reports the performance at the 60-th step.

Analysis: As the prediction horizon increases, all models suffer from error accumulation. However, ProSpectNet exhibits the best stability and slowest error degradation.

- Stability Advantage over Transformers: STAEformer shows a sharp performance degradation at Horizon 60 (MAE 18.82 on PEMS08), indicating that attention maps may become noisy over long horizons. ProSpectNet maintains a much lower MAE (16.12), validating that our MFD module effectively filters out noise that causes cumulative errors.
- Consistency compared to Efficient Models: While STG-Mamba is designed for long sequences and performs well (MAE 16.85 on PEMS08), ProSpectNet still achieves better accuracy (MAE 16.12). This suggests that for traffic data, which contains specific daily and weekly periodicities, our explicit trend-detail decomposition provides better temporal consis-

TABLE II
Long-term Performance at Horizon 60 (5 Hours). Baseline results are cited from [14], except for TimeMixer and STG-Mamba which are obtained from their official implementations.

Model	PEMS04			PEMS08		
	MAE	RMSE	MAPE(%)	MAE	RMSE	MAPE(%)
STGCN [2]	26.43	40.80	19.45	24.88	37.88	17.19
DCRNN [3]	25.39	39.92	18.10	23.12	35.22	16.35
AGCRN [12]	25.72	41.42	17.66	24.73	38.79	17.15
PDFormer [5]	23.64	37.73	15.91	19.52	31.40	14.12
STAEformer [6]	24.47	39.47	15.58	18.82	30.76	13.83
TimeMixer [8]	22.10	36.50	15.20	18.90	30.50	13.95
STG-Mamba [13]	21.05	34.20	14.60	16.85	27.90	12.10
ProSpectNet	20.44	33.67	14.30	16.12	26.86	11.33

tency than the implicit state-space mixing of Mamba. TimeMixer, lacking graph structure modeling, fails to capture long-term spatial dependencies, resulting in higher errors (MAE 18.90 on PEMS08).

D. Ablation Study

Ablation studies on PEMS08 (Table III) confirm the necessity of our design.

TABLE III
Ablation Study Results on All Datasets (Average)

Dataset	Variant	MAE	RMSE	MAPE
PEMS03	ProSpectNet (Full)	14.49	23.48	15.23%
	w/o MFD (AvgPool)	14.59	23.70	16.13%
	w/o LPR (Local only)	14.59	23.50	16.03%
PEMS04	ProSpectNet (Full)	18.04	29.64	12.40%
	w/o MFD (AvgPool)	18.09	29.64	12.45%
	w/o LPR (Local only)	18.08	29.64	12.42%
PEMS08	ProSpectNet (Full)	13.62	23.21	9.23%
	w/o MFD (AvgPool)	13.76	22.78	9.11%
	w/o LPR (Local only)	13.74	23.05	9.23%

E. Hyperparameter Sensitivity Analysis

To investigate the robustness of ProSpectNet and the influence of the latent prototype memory bank, we conduct a sensitivity analysis on the hyperparameter K (the number of prototypes). We evaluate the model's performance on the PEMS08 dataset ($N = 170$) under varying settings of $K \in \{8, 17, 32, 64\}$.

It is worth noting that to facilitate a rapid and fair relative comparison, this set of ablation experiments was uniformly conducted under a constrained training budget of 200 epochs. The results are summarized in Table IV.

TABLE IV
Sensitivity Analysis of Prototype Quantity (K) on PEMS08 (recorded at 200 epochs for relative comparison).

Number of Prototypes (K)	MAE	RMSE	MAPE
$K = 8$	14.08	23.16	9.66%
$K = 17$	14.13	23.25	9.45%
$K = 32$	14.00	23.11	9.17%
$K = 64$	13.93	22.99	9.42%

As presented in Table IV, the prototype quantity K fundamentally controls the trade-off between semantic representational capacity and computational overhead. When K is set to a relatively small value (e.g., $K = 8$ or 17), the model exhibits semantic underfitting. Due to the limited number of prototype vectors, the Latent Prototype Router (LPR) struggles to fully encapsulate the diverse, localized traffic behaviors across the 170 sensor nodes, resulting in suboptimal predictive accuracy.

Conversely, expanding the prototype memory bank to $K = 64$ provides sufficient representational capacity to disentangle distinct spatial semantic clusters, thereby achieving the optimal performance (MAE of 13.93). While these intermediate metrics have not yet fully converged to our state-of-the-art results reported in Table I (MAE 13.50 at the full 500 epochs), the relative performance trend clearly validates the structural advantage of $K = 64$.

Furthermore, we intentionally do not employ larger values such as $K = 128$. The core motivation of the LPR module is to act as an information bottleneck that extracts shared, low-dimensional semantic patterns. For a network of $N = 170$ nodes, setting a disproportionately large K (e.g., $K \approx N$) would destroy this bottleneck, introducing severe parameter redundancy and increasing the risk of overfitting. More crucially, as K approaches N, the linear computational advantage $O(NK)$ would degenerate towards the prohibitive $O(N^2)$ cost of standard full attention. Therefore, $K = 64$ strikes the strict optimal Pareto frontier, harmonizing high modeling capacity with lightweight execution.

F. Complexity and Efficiency Analysis

A critical contribution of ProSpectNet is its ability to model global dependencies with linear computational complexity, avoiding the $O(N^2)$ bottleneck of Transformers. As shown in Table V, we compared the computational costs against representative baselines from three categories: classical GNNs, heavy Transformers, and recent efficient architectures.

Quantitative Analysis: ProSpectNet achieves efficient computational performance while maintaining predictive accuracy, demonstrating comprehensive advantages in both accuracy and efficiency:

1) Vs. high-capacity Transformers: Compared to STAEformer and PDFormer, ProSpectNet provides an

TABLE V
Efficiency Comparison on PEMS08 (Batch Size=64). ProSpectNet achieves the best balance of low parameters and high speed.

Model	Params (M)	Training (s/epoch)	Inference (ms/batch)	GPU Mem (MB)
STGCN [2]	0.21	15	8.5	1100
PDFormer [5]	3.10	168	72.1	7200
STAEformer [6]	2.54	142	58.2	6800
TimeMixer [8]	0.85	18	5.2	1350
STG-Mamba [13]	1.62	35	14.5	2100
ProSpectNet	0.32	12	4.11	821

TimeMixer and STG-Mamba metrics are reproduced based on official configs.

order-of-magnitude improvement in efficiency. It is approximately 14× faster in inference (4.11 ms vs. 58.2 ms) and reduces GPU memory usage by 88% (821 MB vs. 6800 MB), making it feasible for deployment on edge devices.

2) Vs. Recent Efficient Models: Even compared to the latest efficient designs, ProSpectNet demonstrates superiority. Although STG-Mamba achieves linear complexity $O(N)$, it requires significantly more parameters (1.62 M vs. 0.32 M) and memory to maintain its state variables. ProSpectNet is 3.5× faster than Mamba. While TimeMixer is extremely fast (5.2 ms), ProSpectNet achieves even faster inference (4.11 ms) with fewer parameters, without sacrificing the spatial accuracy that TimeMixer lacks.

In summary, ProSpectNet effectively balances high performance with computational efficiency, achieving the lowest latency and memory footprint among all baselines.

G. Visualization Analysis

To intuitively understand the model's effectiveness and interpretability, we visualize the learned latent prototypes and node embeddings on the PEMS08 dataset.

Analysis of Latent Prototype Clustering: To investigate the interpretability of the proposed Latent Prototype Router, we visualize the learned prototype vectors and node embeddings in the latent space using t-SNE, as shown in Fig. 2.

Several key observations can be drawn from the visualization:

1) Semantic Disentanglement: Although the model was initialized with multiple prototypes, it automatically converged to utilizing three dominant prototypes (Proto 3, 5, and 7) to represent the entire traffic network. This indicates that the model has successfully filtered out redundant information and captured the core underlying patterns of the road network (Automatic Feature Selection).
2) Physical Interpretation: The three distinct clusters align well with the fundamental phases of traffic flow theory: free-flow, congestion, and transition states. For instance, the large cluster associated with Proto 7 likely represents nodes with stable traffic patterns (e.g., free-flow), while the smaller, distinct clusters (Proto 3 and 5) may correspond to nodes exhibiting specific congestion patterns or high variability.
3) Functional Clustering: Unlike traditional Graph Convolutional Networks (GCNs) that aggregate information based on fixed geographical adjacency, our method groups nodes based on their functional similarity in the latent space. Nodes that are geographically distant but share similar traffic behaviors are effectively clustered together.

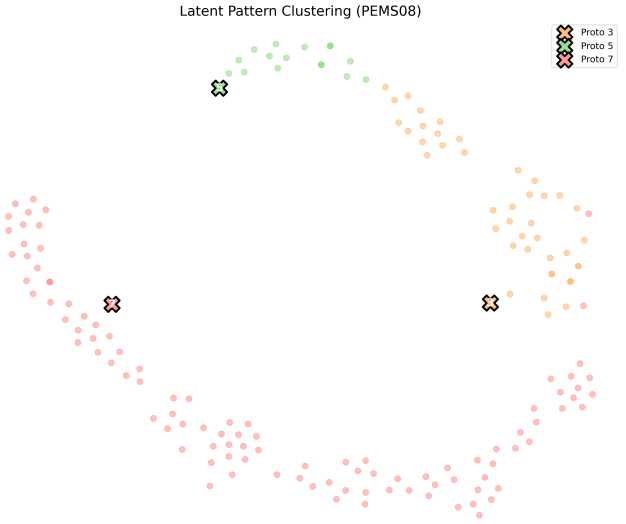

Fig. 2. Visualization of the learned latent prototypes and node embeddings on the PEMS08 dataset. The crosses (X) represent the learned latent prototypes, and the circles (•) represent traffic nodes. Nodes are colored based on their assignment to the nearest prototype. The visualization demonstrates that the proposed Latent Prototype Router successfully disentangles traffic nodes into distinct semantic clusters (e.g., Proto 3, 5, and 7) based on their functional similarities.

In summary, this visualization confirms that our proposed module effectively learns robust and interpretable representations, enabling the model to distinguish between different functional regions within the complex traffic network.

V. Limitations and Future Work

While ProSpectNet demonstrates a superior balance between performance and efficiency, we identify three avenues for further improvement:

1) Response to Non-Stationary Outliers: The current prototype learning relies on historical traffic patterns. Consequently, the model may struggle to generalize to unseen, extreme events (e.g., massive accidents or natural disasters) where traffic behaviors deviate significantly from the learned latent clusters.
2) Static Prototype Allocation: Although the attention mechanism in LPR is dynamic, the number of prototypes K is a fixed hyperparameter. An insufficient K

may cause underfitting in complex networks, while an excessive K introduces redundancy. A mechanism to dynamically expand or prune prototypes based on data complexity would be desirable.

3) Cold Start Problem: For newly installed sensors with limited historical data, the MFD module may fail to extract stable trend components, potentially affecting early-stage prediction accuracy. Although we have employed random graph scrubbing during training to mitigate partial missing data, handling absolute zero-history nodes remains a future challenge.

Future Work: We plan to extend ProSpectNet in two directions: (1) Incorporating Large Language Models (LLMs) to align textual traffic reports with numerical data, enhancing the model's understanding of extreme events; and (2) Exploring Online Learning mechanisms to update prototype vectors in real-time, allowing the system to adapt to shifting traffic distributions and mitigating the cold start issue.

VI. Conclusion

In this paper, we presented ProSpectNet, a streamlined spatio-temporal framework designed to address the critical inefficiency of existing high-performance traffic forecasting models. By integrating the Multi-Frequency Decomposition (MFD) module for signal disentanglement and the Latent Prototype Router (LPR) for global context modeling, we successfully reduced the computational complexity from quadratic to linear while maintaining fine-grained perception.

Experimental results on real-world datasets confirm that ProSpectNet effectively balances high performance with computational efficiency. While maintaining accuracy on par with heavy SOTA Transformers (e.g., STAEformer), our model achieves an order-of-magnitude improvement in speed (4.11 ms vs. 58.2 ms) and significantly lower memory usage. Furthermore, it outperforms the latest efficient baselines, such as STG-Mamba and TimeMixer, in terms of accuracy. ProSpectNet thus proves that high accuracy does not require heavy computation, establishing a new benchmark for efficient and scalable traffic forecasting in modern ITS.

References

[1] W. Jiang and J. Luo, "Deep learning for traffic forecasting: A survey," IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 9, pp. 14 069–14 097, 2022.

[2] B. Yu et al., "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," in Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI), 2018, pp. 3634–3640.

[3] Y. Li et al., "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in International Conference on Learning Representations (ICLR), 2018.

[4] Z. Wu et al., "Graph wavenet for deep spatial-temporal graph modeling," in Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI), 2019, pp. 1907–1913.

[5] J. Jiang et al., "Pdformer: Propagation delay-aware dynamic long-range transformer for traffic flow prediction," in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 37, no. 4, 2023, pp. 4365–4373.

[6] H. Liu et al., "Spatio-temporal adaptive embedding makes vanilla transformer sota for traffic forecasting," in Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM), 2023, pp. 4125–4129.

[7] C. Zheng et al., "Gman: A graph multi-attention network for traffic prediction," in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, no. 01, 2020, pp. 1234–1241.

[8] S. Wang et al., "Timemixer: Decomposable multiscale mixing for time series forecasting," in International Conference on Learning Representations (ICLR), 2024.

[9] S. Lan et al., "Decoupled spatial-temporal attention network for traffic forecasting," in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 36, no. 4, 2022, pp. 4081–4090.

[10] L. Cai et al., "Traffic transformer: Capturing the continuity and periodicity of time series for traffic forecasting," in Transactions in GIS, vol. 24, no. 3, 2020, pp. 736–755.

[11] Z. Wu et al., "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, 2020, pp. 753–763.

[12] L. Bai et al., "Adaptive graph convolutional recurrent network for traffic forecasting," in Advances in Neural Information Processing Systems (NeurIPS), vol. 33, 2020, pp. 17 804–17 815.

[13] L. Li et al., "Stg-mamba: Spatial-temporal graph learning with selective state space models," in Proceedings of the AAAI Conference on Artificial Intelligence, 2025.

[14] P. Wang, W. Zhang, L. Feng, and K. Hui, "Ta-csa: A spatio-temporal framework with temporal aggregation and clustered spatial attention for traffic forecasting," IEEE Internet of Things Journal, pp. 1–1, 2025.