EULER FREQUENCY MODULATION FOR NON-STATIONARY SPATIOTEMPORAL GRAPH TRAFFIC FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Short-term traffic forecasting is essential for urban transportation management. While traffic flow exhibits regular day-to-day periodic patterns, spatiotemporal fluctuations make accurate prediction challenging. Conventional traffic flow prediction models have difficulty capturing these variations, particularly when spatiotemporal dependencies become highly non-stationary. To address this issue, this study proposes Euler Frequency Spatiotemporal Graph Neural Network(EFSG), a spatiotemporal model that uses Euler Frequency Modulation (EFM), which applies FFT on each short input window and jointly edits amplitude and phase for every sample, node, and channel. EFSG further builds a spectrum-aware dynamic graph linking nodes with similar spectral patterns beyond physical adjacency, and a gated fusion layer adaptively combines local and frequency-domain features. Ablation studies reveal that EFM and spectral propagation function as a coupled unit driving performance gains. Experiments on four real-world datasets show that EFSG achieves a 19.9% reduction in MAE on NYC-Bike pick-up and a 60% reduction in GPU memory consumption on PEMS04 compared with TEDDCF, while converging within 20 epochs.

1 INTRODUCTION

Short-term traffic forecasting is vital for intelligent transportation systems. It supports traffic control, route planning, and daily urban management. In recent years, spatio-temporal graph neural networks (STGNNs) have become the main tools for this task (Li & Zhu, 2021; Wu et al., 2021; Li et al., 2023). They combine graph convolution with temporal modules to learn spatial links and time patterns at the same time. To raise accuracy, many recent models grow very complex. They stack deep residual blocks (Shao et al., 2022b), multi-scale attention (Bai et al., 2021), or high-order differential equations (Choi et al., 2022; Fang et al., 2021). These designs often need large GPU memory, train slowly, and take too much time to run.

Beyond model size, two practical issues remain. First, real traffic peaks change in strength and time. Accidents or sudden demand can make a peak stronger, weaker, earlier, or later. Standard temporal convolutions and attention treat time steps as fixed, so they struggle with these shifts. Second, most models rely on a fixed road graph or a single learned adjacency. They miss dynamic ties between nodes that share similar fluctuation patterns but are not direct neighbors on the map.

To address these issues, we propose the Euler Frequency Spatiotemporal Graph network (EFSG). EFSG starts from a light temporal convolution backbone and a simple adaptive graph. It then applies Euler Frequency Modulation (EFM) to adjust the amplitude and phase of each node in the frequency domain. This lets the model change peak strength and peak timing in a direct way. After modulation, EFSG builds a dynamic graph from spectrum similarity, so nodes with like fluctuation patterns can exchange information even when they are far apart on the road network. A gating step finally fuses the local and frequency paths before multi-step prediction.

The main contributions are as follows:

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.

- We propose EFSG, a memory-efficient model that uses Euler's formula to modulate per-frequency magnitude and phase in an input-dependent manner over samples, nodes, and channels.

- Experiments on four real-world datasets show strong accuracy and efficiency. EFSG reaches new best MAE results on the four datasets. On PEMS04 it cuts GPU memory by 60% against TEDDCF and finishes training within 20 epochs.

2 Related Work

Early traffic flow prediction mainly used statistical methods to capture linear patterns in historical data. For example, some studies modeled traffic flow as a seasonal ARIMA process and discussed its theoretical basis and empirical results (Williams & Hoel, 2003). An integrated Granger causality test and vector autoregression (VAR) method was also proposed for network traffic prediction (Lu et al., 2016). Although these methods laid a useful foundation, they usually treated the traffic network as a set of isolated nodes and ignored the complex spatial dependencies in the road network. As data size and complexity grew, deep learning methods gradually replaced traditional approaches (Li & Zhu, 2021). In the temporal dimension, researchers commonly use recurrent neural networks (RNNs) (Chauhan & Kumar, 2024), temporal convolutional networks (TCNs) (Wu et al., 2020), and attention mechanisms (Bai et al., 2021; Guo et al., 2021) to extract hidden time patterns. In the spatial dimension, Graph Neural Networks (GNNs) have become the standard way to capture road network topology (Wu et al., 2021). Early work mainly explored static spatio-temporal fusion graphs (Li & Zhu, 2021; Yu et al., 2017), while later studies shifted toward learning adaptive graph structures (Zhang et al., 2020). Typical models introduce spatial modeling through Chebyshev polynomial approximation and diffusion convolution (Li et al., 2017), and later frameworks propose adaptive adjacency matrices to learn potential spatial correlations (Wu et al., 2019). However, these models still find it challenging to handle non-stationary peaks whose strength and arrival time change quickly, and their graphs mainly follow road distance or learned spatial links rather than spectral similarity.

To further improve performance, researchers have turned to physical equations and frequency analysis. Some studies combine physical equations to strengthen the physical awareness of the model, such as graph neural controlled differential equations (Choi et al., 2022) and space-time graph ODE networks (Fang et al., 2021). However, many of these physics-aware models need heavy pre-processing and bring high computational cost, which limits real-world use. The clear periodic nature of traffic naturally motivates frequency-domain analysis. While classical approaches used discrete Fourier transforms only as a post-hoc feature extractor, recent advances have begun to explore frequency-domain operations as efficient substitutes for complex attention mechanisms in time-series forecasting (Zhou et al., 2022). Wavelet-based and FFT-based methods have also been introduced to explicitly separate traffic signals into trend and event-driven components (Fang et al., 2023; Xu et al., 2025). Hosseini et al. (2026) extends neural operators to the graph Fourier domain for long-range dependency modeling, while Teng et al. (2025)and Ren et al. (2025) introduce dynamic frequency-filtering and spectral gating to separate multi-scale periodic patterns. Meanwhile, advanced frameworks explore hierarchical frequency decomposition in HiFiNet (Ma et al., 2026), combine graph Fourier representations with state-space models in STGFMamba (Han et al., 2026). However, most of these frequency-based methods still build spatial graphs from physical distance or simple k-nearest neighbors, so they miss nodes that are far on the map but share similar temporal spectral patterns. They also rarely offer a light way to edit both peak strength and peak phase at each node. To overcome these limitations, this research proposes EFSG for short-term traffic forecasting under non-stationary conditions. The model builds on temporal convolutions and uses Euler frequency modulation to adjust peak intensity and phase at each node. It also forms spectrum-aware links among locations that share similar fluctuation patterns, so mutual influence is not limited to physical adjacency.

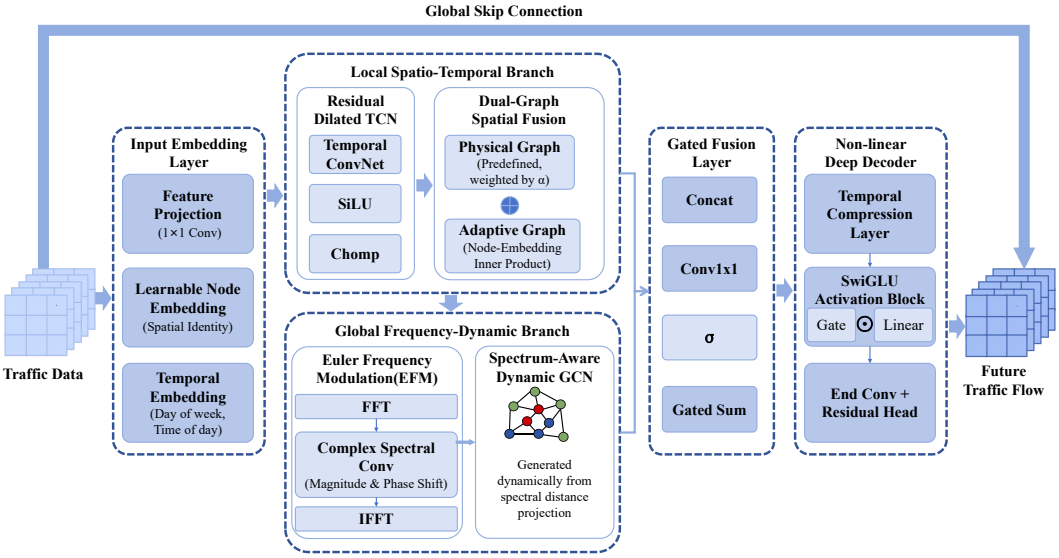


Figure 1: The overall architecture of EFSG.

3 METHODOLOGY

3.1 OVERALL ARCHITECTURE

This paper proposes EFSG, a spatiotemporal forecasting model that joins temporal convolution, adaptive graph learning, Euler frequency modulation, and spectrum-aware dynamic graph propagation. The model takes historical traffic series from all nodes as input. The data flow follows Figure 1 from left to right. On the left of Figure 1, the input embedding layer first projects the raw features with a 1×1 convolution, then adds a learnable node embedding for spatial identity and a temporal embedding for day of week and time of day. These embedded features enter the local spatio-temporal branch. In this branch, a residual dilated TCN extracts short-term temporal patterns, and a dual-graph spatial fusion mixes a predefined physical graph with a learnable adaptive graph built from node-embedding inner products. Building on the local representation, the global frequency-dynamic branch applies Euler frequency modulation. Each signal is transformed by FFT, adjusted in magnitude and phase by a complex spectral convolution, and mapped back by IFFT. A spectrum-aware dynamic GCN then builds edges from spectral distance and aggregates messages across nodes that share similar frequency patterns. A gated fusion layer combines the local path and the frequency path with a learned gate. The fused features pass through a non-linear deep decoder that uses temporal compression and a SwiGLU block, followed by an end convolution with a residual head. A global skip connection runs from the raw input through a 1×1 convolution and is added to the final prediction so that multi-step future traffic flow keeps a direct residual link to history.

The input and output of the model are

$$\mathbf{X} \in \mathbb{R}^{B \times C_{\text{in}} \times N \times T}, \tag{1}$$

$$\hat{\mathbf{Y}} \in \mathbb{R}^{B \times C_{\text{out}} \times N \times T'}, \tag{2}$$

where B is the batch size, C_{in} is the number of input channels, C_{out} is the number of output channels, N is the number of traffic nodes, T is the history length, and T' is the forecast length. The element $\mathbf{X}_{b,c,n,t}$ is the observed value of variable c at node n and time t in sample b. Similarly, the element $\hat{\mathbf{Y}}_{b,c,n,\tau}$ is the forecast at future step τ.

In the experiments, both the history length and the forecast length are set to 12:

$$T = 12, \qquad T' = 12. \tag{3}$$

When the sampling interval is 5 minutes, the history and the forecast each cover one hour. In this study, for PEMS03 and PEMS04, the input contains three channels (one traffic variable, day-of-week index, and time-of-day index), and the output predicts only that traffic variable, so

$$C_{\text{in}} = 3, \qquad C_{\text{out}} = 1. \tag{4}$$

For NYC-BIKE and NYC-TAXI, the input contains four channels (pick-up rate, drop-off rate, day-of-week index, and time-of-day index), while the output predicts only pick-up and drop-off rates, so

$$C_{\text{in}} = 4, \qquad C_{\text{out}} = 2. \tag{5}$$

3.2 Temporal Feature Extraction and Adaptive Graph

This subsection corresponds to the multi-view embedding layer and local spatio-temporal branch in Figure 1. We first keep only the target traffic channels and map them with a 1×1 convolution. A learnable node embedding is added to give each location a stable identity. Time-of-day and day-of-week indices are turned into embeddings and concatenated with the traffic features:

$$\mathbf{H}_x = \text{Conv}_{1\times1}(\mathbf{X}_{\text{feat}}) + \mathbf{E}_{\text{node}}, \tag{6}$$

$$\mathbf{E}_t = \mathbf{E}_{\text{tod}}(s_t) + \mathbf{E}_{\text{dow}}(d_t), \tag{7}$$

$$\mathbf{H}_0 = \text{Concat}(\mathbf{H}_x,\, \mathbf{E}_t) \in \mathbb{R}^{B\times 2C\times N\times T}. \tag{8}$$

Here $\mathbf{X}_{\text{feat}}$ denotes the traffic channels, s_t is the time-of-day index, d_t is the day-of-week index, C is the hidden size, and $\mathbf{E}_{\text{node}} \in \mathbb{R}^{1\times C\times N\times 1}$ denotes a learnable spatial embedding that provides time-invariant geographical representations for each node.

Taking this initial representation $\mathbf{H}_0$ as the starting point ($l = 0$), dilated temporal convolutions then extract multi-scale patterns along the time axis only. At layer l the dilation is $d_l = 2^l$. Each dilated conv is followed by a chomp operation to keep the sequence length equal to T. Each layer uses a residual link, SiLU activation, and dropout:

$$\mathbf{H}_l = \mathbf{H}_{l-1} + \text{Dropout}\Big(\sigma\big(\text{Conv}^{(d_l)}_{1\times K}(\mathbf{H}_{l-1})\big)\Big), \tag{9}$$

where σ is SiLU and K is the kernel size (set to 3), and $\mathbf{H}_{l-1} = \mathbf{H}_0$ when $l = 1$. After three layers we obtain the local temporal feature $\mathbf{H}_{\text{tcn}} \in \mathbb{R}^{B\times 2C\times N\times T}$.

Next we build a learnable adaptive graph from two node embedding matrices $\mathbf{E}_1 \in \mathbb{R}^{N\times 32}$ and $\mathbf{E}_2 \in \mathbb{R}^{32\times N}$:

$$\mathbf{A}_{\text{adp}} = \text{softmax}\Big(\frac{\mathbf{E}_1\mathbf{E}_2}{\sqrt{32}}\Big). \tag{10}$$

If a real road adjacency $\mathbf{A}_{\text{real}}$ is available, we mix it with a learnable weight α:

$$\mathbf{A} = \frac{\text{ReLU}(\mathbf{A}_{\text{adp}} + \alpha\mathbf{A}_{\text{real}})}{\text{ReLU}(\mathbf{A}_{\text{adp}} + \alpha\mathbf{A}_{\text{real}})\mathbf{1} + \varepsilon}. \tag{11}$$

Otherwise we simply use $\mathbf{A} = \mathbf{A}_{\text{adp}}$. Graph message passing is applied and controlled by a scalar gate g:

$$\mathbf{H}_{\text{g}} = g \cdot (\mathbf{A}\mathbf{H}_{\text{tcn}}). \tag{12}$$

The local branch is finally formed by residual fusion and RMS normalization:

$$\mathbf{H}_{\text{local}} = \text{RMSNorm}\left(\mathbf{H}_{\text{tcn}} + \mathbf{H}_{\text{g}} + \mathbf{H}_0\right). \tag{13}$$

This branch captures regular day-to-day patterns and fixed spatial structure. Peak shifts and long-range links are handled next.

3.3 Euler Frequency Modulation and Spectrum-Aware Propagation

The key mechanism of this module lies in using Euler's formula to modulate per-frequency magnitude and phase in an input-dependent manner over samples, nodes, and channels. The Global Frequency-Dynamic Branch in Figure 1 implements the two blocks below: Euler frequency modulation (EFM) and the spectrum-aware dynamic GCN. Traffic peaks often change in strength and arrive earlier or later. We adjust these two factors in the frequency domain using Euler's formula

in polar form $x = |x|\, e^{j\theta}$, where modifying the magnitude scales peak intensity and changing the phase shifts peak timing.

Euler Frequency Modulation (left block of the Global Frequency-Dynamic Branch in Figure 1). Given $\mathbf{H}_{\text{local}}$, we apply a real Fast Fourier Transform (rFFT) along the time axis to obtain the spectrum $\mathbf{F} \in \mathbb{C}^{B\times 2C\times N\times F}$ (where $F = T/2 + 1$). To dynamically modulate every frequency, we concatenate the real and imaginary parts of $\mathbf{F}$ and do the convolution along the frequency axis. Here, $\mathrm{Re}(\mathbf{F})$ and $\mathrm{Im}(\mathbf{F})$ represent the real and imaginary parts of the complex spectrum $\mathbf{F}$, respectively. The output is split into two halves, the first half is mapped by a sigmoid to a magnitude modulation gate $\mathbf{G}_m$, and the second half is mapped by $\tanh$ (scaled by π) to a dynamic phase shift $\mathbf{\Phi}_{\text{dyn}}$:

$$[\mathbf{M},\, \mathbf{\Delta}_{\text{dyn}}] = \mathrm{Conv}_{1\times 5}\left([\mathrm{Re}(\mathbf{F}),\, \mathrm{Im}(\mathbf{F})]\right), \tag{14}$$

$$\mathbf{G}_m = \mathrm{sigmoid}(\mathbf{M}), \qquad \mathbf{\Phi}_{\text{dyn}} = \pi \cdot \tanh(\mathbf{\Delta}_{\text{dyn}}). \tag{15}$$

Combining these with a learnable base phase $\mathbf{\Delta}_\theta$ (specific to each channel and frequency), the modulated spectrum is formulated as:

$$\tilde{\mathbf{F}} = \left(|\mathbf{F}| \odot \mathbf{G}_m\right) \exp\Big(j\big(\angle\mathbf{F} + \mathbf{\Delta}_\theta + \mathbf{\Phi}_{\text{dyn}}\big)\Big). \tag{16}$$

In short, Eq. equation 16 enables the model to jointly adjust peak strength and shift peak timing for every sample, channel, and node. An inverse real FFT returns the signal to the time domain, followed by a 1×1 convolution (channel mix) and a residual connection:

$$\mathbf{H}_{\text{eul}} = \mathrm{Conv}_{1\times 1}\left(\mathrm{irFFT}(\tilde{\mathbf{F}})\right) + \mathbf{H}_{\text{local}}. \tag{17}$$

$\mathbf{H}_{\text{eul}} \in \mathbb{R}^{B\times 2C\times N\times T}$ is the Euler-modulated feature. It has the same shape as $\mathbf{H}_{\text{local}}$ and is the input to the next block.

Spectrum-Aware Graph Propagation (right block of the Global Frequency-Dynamic Branch in Figure 1). After Euler modulation, nodes sharing similar spectral patterns interact through a spectrum-aware graph.We again apply rFFT, but this time on $\mathbf{H}_{\text{eul}}$. We take the mean magnitude over channels, project it to a 32-dimensional vector, and ℓ_2-normalize each node embedding:

$$\mathbf{S} = \mathrm{Normalize}\Big(\mathrm{Linear}\left(\mathrm{mean}_c\, |\,\mathrm{rFFT}(\mathbf{H}_{\text{eul}})|\right)\Big) \in \mathbb{R}^{B\times N\times 32}, \tag{18}$$

Here $\mathrm{rFFT}(\mathbf{H}_{\text{eul}})$ is the spectrum of the Euler output, $|\cdot|$ is the magnitude, mean_c averages over the channel axis, The spectrum amplitude vector of each node is linearly mapped to 32 dimensions, and $\mathrm{Normalize}$ is ℓ_2 normalization. The row $\mathbf{S}_{bi} \in \mathbb{R}^{32}$ is the spectral embedding of node i in batch b. We measure the similarity between two nodes by computing the Euclidean distance between their spectral embeddings, then convert this distance into an affinity weight using a Gaussian kernel:

$$D_{bij} = \|\mathbf{S}_{bi} - \mathbf{S}_{bj}\|_2, \qquad A^{\text{spec}}_{bij} = \exp\Big(-\frac{D^2_{bij}}{2\sigma_b^2}\Big). \tag{19}$$

D_{bij} is the Euclidean distance between the spectral embeddings of node i and node j in batch b. A^{spec}_{bij} is the soft affinity weight. σ_b is the standard deviation of all pairwise Euclidean distances inside batch b (plus a small constant for numerical stability). A smaller Euclidean distance gives a larger affinity weight, meaning nodes with similar spectral patterns become neighbors in this dynamic graph.

To keep the graph sparse and cheap, we keep only the top-k neighbors of each node and apply degree normalization:

$$\tilde{\mathbf{A}}^{\text{spec}} = \mathrm{TopK}(\mathbf{A}^{\text{spec}}, k), \qquad \hat{\mathbf{A}}^{\text{spec}} = \mathbf{D}^{-\frac{1}{2}}\tilde{\mathbf{A}}^{\text{spec}}\mathbf{D}^{-\frac{1}{2}}. \tag{20}$$

$\tilde{\mathbf{A}}^{\text{spec}}$ is the sparse affinity matrix after Top-k. $\mathbf{D}$ is the degree matrix of $\tilde{\mathbf{A}}^{\text{spec}}$. $\hat{\mathbf{A}}^{\text{spec}}$ is the normalized graph used for message passing.

Graph convolution mixes features of spectrally similar nodes, then a residual link and LayerNorm produce the frequency-branch feature:

$$\mathbf{H}_{\text{freq}} = \mathrm{LayerNorm}\left(\mathrm{Conv}_{1\times 1}(\hat{\mathbf{A}}^{\text{spec}}\mathbf{H}_{\text{eul}}) + \mathbf{H}_{\text{eul}}\right). \tag{21}$$

$\hat{\mathbf{A}}^{\text{spec}}\mathbf{H}_{\text{eul}}$ aggregates neighbor features of $\mathbf{H}_{\text{eul}}$. The 1×1 convolution is a channel-wise linear map. $\mathbf{H}_{\text{freq}}$ has the same shape as $\mathbf{H}_{\text{eul}}$.

Table 1: The details for the datasets.

Datasets	Nodes	Samples	Granularity	Time range
PEMS03	358	26208	5 min	2018/9/1–2018/11/30
PEMS04	307	16992	5 min	2018/1/1–2018/2/28
NYCTaxi	266	4368	30 min	2016/4/1–2016/6/30
NYCBike	250	4368	30 min	2016/4/1–2016/6/30

3.4 Gated Fusion and Decoder

We then add the original local feature and apply RMS normalization:

$$\mathbf{H}_{\text{freq}} = \text{RMSNorm}(\mathbf{H}_{\text{freq}} + \mathbf{H}_{\text{local}}). \tag{22}$$

This step explicitly reinjects the local representation into the frequency branch to preserve fine-grained patterns.

A gating mechanism (Gated Fusion Layer in Figure 1) blends the local branch and the frequency branch:

$$\mathbf{G} = \text{sigmoid}\left(\text{Conv}_{1\times1}([\mathbf{H}_{\text{local}}, \mathbf{H}_{\text{freq}}])\right), \tag{23}$$

$$\mathbf{H}_{\text{final}} = \mathbf{G} \odot \mathbf{H}_{\text{local}} + (1 - \mathbf{G}) \odot \mathbf{H}_{\text{freq}}. \tag{24}$$

$[\mathbf{H}_{\text{local}}, \mathbf{H}_{\text{freq}}]$ means concatenation along the channel axis. $\mathbf{G} \in (0, 1)$ is a soft gate of the same shape as the features. When $\mathbf{G}$ is close to 1, the model trusts the local branch more; when it is close to 0, it trusts the frequency branch more. $\mathbf{H}_{\text{final}}$ is the fused representation used by the decoder.

The decoder (Non-linear Deep Decodr in Figure 1) first compresses the time axis with a convolution over the full window, then applies a SwiGLU block, and finally maps to T' future steps:

$$\mathbf{Y}_{\text{deep}} = \text{Conv}_{\text{end}}\big(\text{SwiGLU}(\text{Conv}_T(\mathbf{H}_{\text{final}}))\big). \tag{25}$$

Conv_T reduces the time length from T to 1. SwiGLU is a gated nonlinear transform. Conv_{end} produces the deep forecast tensor $\mathbf{Y}_{\text{deep}}$.

To ensure training stability, we incorporate two skip paths (top residual arrow in Figure 1) consisting of a learned mapping from the history window and a weighted copy of the last observed step:

$$\hat{\mathbf{Y}} = \mathbf{Y}_{\text{deep}} + \text{Skip}(\mathbf{X}_{\text{feat}}) + \beta \cdot \mathbf{X}_{\text{feat}}[:, :, :, -1:]. \tag{26}$$

$\mathbf{X}_{\text{feat}}$ is the raw target history (e.g., the observed traffic channels). Skip is a convolution that maps the history window of length T to the forecast horizon of length T'. β is a learnable scalar. $\mathbf{X}_{\text{feat}}[:, :, :, -1:]$ is the last time step, repeated as a simple identity baseline. The final prediction $\hat{\mathbf{Y}}$ is reshaped to $\mathbb{R}^{B\times C_{\text{out}}\times N\times T'}$.

The whole pipeline stays light because temporal convolution is cheap, the adaptive spatial graph is shared across time, Euler modulation works on a short FFT of length $T = 12$, and the spectrum graph keeps only k neighbors per node. This design yields strong accuracy with low GPU memory and fast convergence within 20 epochs.

4 Experiments

4.1 Datasets and Experimental Settings

We evaluate EFSG on four datasets (Table 1), two highway flow datasets (PEMS03/04(Song et al., 2020)) and two NYC demand datasets (NYCBike/NYCTaxi (Ye et al., 2021)), the latter featuring dual-dimensional "Pick-up" and "Drop-off" data.

To reduce the influence of random initialization, we trained with five random seeds and report the average performance. All datasets are split 6:2:2 for training, validation, and testing, using 12 historical steps to predict the next 12. Experiments are conducted on an NVIDIA RTX 4090 GPU using PyTorch 2.5.1. Hyper-parameters (e.g., channels, learning rate, and neighbors k) are optimized

Table 2: Baseline comparison results of PEMS datasets. Best results are in **bold**.

Model	PEMS03			PEMS04		
	MAE	RMSE	MAPE	MAE	RMSE	MAPE
HA	30.08	46.22	28.64%	38.51	55.75	28.21%
SVR	21.97	35.29	21.51%	28.70	44.56	19.20%
GMAN	16.52	27.18	17.36%	18.84	30.75	13.25%
ASTGCN	17.41	29.57	17.32%	21.80	34.36	14.89%
ASTGNN	14.78	25.01	14.79%	18.67	30.94	12.46%
PDFormer	14.94	25.39	15.82%	18.32	29.97	12.10%
MAGCRN	15.10	26.28	**14.08%**	19.04	31.20	12.45%
TEDDCF	15.11	26.19	15.84%	18.18	29.68	12.08%
STGCN	15.65	27.31	15.39%	19.57	31.38	13.44%
DCRNN	15.53	27.18	15.62%	19.63	31.26	13.59%
STFGNN	16.77	28.34	16.30%	19.83	31.88	13.02%
STG-NCDE	15.57	27.09	15.06%	19.21	31.05	12.76%
DSTAGNN	15.57	27.21	14.68%	19.27	31.21	12.86%
STIDGCN	14.76	24.59	15.28%	18.16	29.77	12.24%
STID	15.33	27.40	16.40%	18.38	29.95	12.04%
STAEformer	15.35	27.55	15.18%	18.22	30.18	**11.98%**
EFSG	**14.59**	**23.59**	15.33%	**18.01**	**29.60**	12.34%

via Optuna with the TPE sampler. We use the Adam optimizer with Masked MAE loss and a OneCycleLR scheduler (20% warm-up). Training is capped at 20 epochs.

To quantitatively evaluate the forecasting accuracy of our proposed EFSG, we employ three standard metrics, Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE). Lower values in these metrics signify superior predictive performance. We evaluate the performance of EFSG by comparing it against several baseline models, ranging from classical statistical methods to the latest state-of-the-art (SOTA) deep learning frameworks. To balance clarity and space, we organize the baselines into three representative groups: (1) *Traditional and early spatio-temporal models*, including HA(Sun et al., 2014), SVR(Drucker et al., 1996), STGCN(Yu et al., 2017), DCRNN(Li et al., 2017), ASTGCN(Guo et al., 2019), and STSGCN(Song et al., 2020); (2) *Adaptive, dynamic, and fusion-based graph networks*, which learn latent spatial dependencies and cross-domain correlations, such as AGCRN(Bai et al., 2020), MTGNN(Wu et al., 2020), DGCRN(Li et al., 2023), STFGNN(Li & Zhu, 2021), DSTAGNN(Lan et al., 2022), STID(Shao et al., 2022a), STG-NCDE(Choi et al., 2022), STFGCN(Li et al., 2024), D^2STGNN(Shao et al., 2022b), STIDGCN(Liu & Zhang, 2024), and MAGCRN(Zeb et al., 2025); and (3) *Attention-based, Transformer, and frequency/decomposition-driven methods*, which model long-range patterns and complex spectral dynamics, consisting of GMAN(Zheng et al., 2020), ASTGNN(Guo et al., 2021), PDFormer(Jiang et al., 2023), STAEformer(Liu et al., 2023), STWave(Fang et al., 2023), and TEDDCF(Xu et al., 2025).

4.2 Overall Performance

The comprehensive performance of all models on the PEMS and NYC datasets is summarized in Table 2 and Table 3. Across all experimental scenarios, EFSG consistently achieves competitive or superior results. The analysis of these results leads to the following observations.

As shown in Table 2, EFSG sets new optimal MAE records on PEMS03 and PEMS04. For instance, on PEMS03, EFSG achieves an MAE of 14.59 and RMSE of 23.59, outperforming the signal-decoupling model TEDDCF (15.11/26.19). In Table 3, EFSG demonstrates a significant leap in performance on the NYC datasets. Specifically, on the NYCBike Pick-up task, EFSG yields a 19.9% reduction in MAE (from 2.01 to 1.61) compared to the previous SOTA, TEDDCF.

4.3 Ablation Study

We run ablations on NYC-Taxi under the same training setting. Each row changes only one design choice. **Full** is the complete model. Table 4 groups related variants together.

Table 3: Baseline comparison results of NYC datasets. Best results are in **bold**.

Model	NYCBike Drop-off			NYCBike Pick-up			NYCTaxi Drop-off			NYCTaxi Pick-up		
	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
DCRNN	1.96	2.94	51.42%	2.09	3.30	54.22%	5.19	9.63	37.78%	5.40	9.71	36.09%
STGCN	2.01	3.07	50.45%	2.08	3.31	53.12%	5.38	9.60	39.12%	5.71	10.22	36.51%
ASTGCN	2.79	4.20	69.88%	2.76	4.45	64.23%	6.98	14.70	45.48%	7.43	13.84	47.96%
AGCRN	2.09	3.19	51.91%	2.16	3.46	56.35%	5.45	9.56	40.67%	5.79	10.11	40.40%
MTGNN	1.94	2.91	50.47%	2.03	3.19	53.73%	5.02	8.76	37.62%	5.39	9.41	37.21%
GMAN	2.09	3.00	54.82%	2.20	3.35	57.34%	5.09	8.95	35.00%	5.48	9.62	36.11%
STSGCN	2.73	4.50	57.89%	2.36	3.73	58.17%	5.62	10.21	37.92%	6.19	11.14	39.67%
ASTGNN	2.24	3.35	57.21%	2.37	3.67	60.08%	6.28	12.00	49.78%	5.90	10.71	40.15%
DGCRN	1.96	2.93	51.99%	2.06	3.21	54.06%	5.14	9.39	35.09%	5.44	9.82	35.78%
STG-NCDE	2.28	3.42	60.96%	2.15	3.97	55.49%	5.38	9.74	40.45%	6.24	11.25	43.20%
D^2STGNN	1.92	2.90	51.94%	2.02	3.18	53.60%	5.01	8.74	35.81%	5.34	9.23	35.96%
STWave	1.93	2.93	49.86%	2.30	3.80	59.78%	5.11	9.30	36.77%	5.28	9.19	35.80%
STFGCN	2.42	3.87	55.21%	2.46	4.19	58.52%	5.61	10.12	43.16%	5.80	10.28	43.72%
TEDDCF	1.89	**2.80**	**49.74%**	2.01	**3.12**	**52.41%**	4.99	8.66	**34.77%**	5.26	9.16	35.58%
EFSG	**1.53**	2.90	54.84%	**1.61**	3.19	56.94%	**4.78**	**8.60**	35.31%	**4.90**	**8.81**	**34.07%**

Table 4: Ablation results on NYC-Taxi.

Model	Drop-off			Pick-up		
	MAE	RMSE	MAPE	MAE	RMSE	MAPE
Full	4.78	8.60	35.31%	4.90	8.81	34.07%
w/o_Freq	4.87	8.77	35.88%	4.99	8.98	34.72%
w/o_Euler	4.80	8.61	35.42%	4.92	8.84	34.24%
w/o_dyn_graph	4.78	8.59	35.16%	4.91	8.83	34.15%
phase_only	4.79	8.57	35.21%	4.90	8.80	33.98%
fixed_highpass	4.81	8.63	35.52%	4.92	8.83	34.29%
inverse_phase	4.78	8.60	35.24%	4.91	8.85	34.10%
w/o_TCN	4.81	8.61	35.88%	4.92	8.81	34.64%
w/o_GC	4.79	8.60	35.19%	4.91	8.83	34.07%
w/o_Skip	4.78	8.60	35.05%	4.91	8.84	33.97%

Frequency branch vs. single parts. w/o_Freq removes the whole frequency branch, including Euler modulation and the spectrum-based dynamic graph. This variant hurts most, Drop-off MAE rises from 4.78 to 4.87, and Pick-up MAE from 4.90 to 4.99. In contrast, removing only Euler (w/o_Euler) or only the dynamic graph (w/o_dyn_graph) causes much smaller changes (Pick-up MAE 4.92 and 4.91, respectively). So each piece looks mild if tested alone, but dropping the full frequency path is clearly worse than those single drops. This pattern suggests that the gains are *not* a simple sum of independent parts. Euler editing and spectrum-graph mixing work better *together* inside the same frequency branch. The branch matters as a coupled unit, not as two isolated add-ons.

Euler modulation vs. other frequency controls. We also replace Euler editing with simpler alternatives, while keeping the rest of the model unchanged. phase_only updates phase and freezes amplitude; it stays close to **Full**, which means the learned phase shift carries most of the Euler gain (peak timing). fixed_highpass uses a fixed high-pass rule instead of learned amplitude–phase control, and is consistently worse than **Full**. inverse_phase flips the phase direction and fails to match **Full** on Pick-up (MAE 4.91 vs. 4.90, RMSE 8.85 vs. 8.81). Thus, hand-fixed filtering or a wrong phase rule cannot replace Euler modulation, the model needs *learnable* and *correctly oriented* phase (and amplitude) editing in the frequency domain.

Other backbone modules. w/o_TCN, w/o_GC, and w/o_Skip change the scores only slightly. They still help training stability and local feature extraction, but their effect is smaller than keeping the frequency branch intact.

Table 5: Peak GPU (GB) on PEMS04. Best results are in **bold**.

Method	AGCRN	DSTAGNN	D^2STGNN	STWave	TEDDCF	EFSG
Peak GPU (GB)	5.4	17.7	19.4	12.3	8.3	**3.3**

Table 6: MAE at 12 prediction horizons on PEMS04. Best results are in **bold**.

Model	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10	H11	H12
TEDDCF	16.72	17.15	17.49	17.76	17.99	18.19	18.39	18.61	18.76	18.94	19.12	19.50
DSTAGNN	17.40	17.94	18.36	18.68	18.97	19.26	19.56	19.80	20.00	20.21	20.53	21.03
EFSG	**16.44**	**16.93**	**17.32**	**17.61**	**17.85**	**18.05**	**18.25**	**18.41**	**18.58**	**18.74**	**18.91**	**19.13**

Overall, on NYC-Taxi, the main accuracy comes from the frequency path. Euler modulation is the key operator inside that path, and it beats simple substitutes. The spectrum dynamic graph alone is a small factor, but together with Euler it supports the stronger `w/o_Freq` result. The remaining modules provide smaller, complementary support.

4.4 Efficiency and Multi-Horizon Performance

A practical traffic model should stay light on GPU memory and keep errors under control when the forecast window grows. We check both points on PEMS04 under the same training setting.

Table 5 reports the peak GPU memory during training. EFSG uses 3.3 GB and is the lowest in the group. It is lower than AGCRN (5.4 GB) and TEDDCF (8.3 GB), and much lower than STWave (12.3 GB), DSTAGNN (17.7 GB), and D^2STGNN (19.4 GB). Against D^2STGNN the cut is about 83%. This gap fits the design of EFSG. The model reuses one adaptive spatial graph, runs an FFT of length $T = 12$, and builds a sparse spectrum graph. It does not stack many heavy spatio-temporal blocks or keep large intermediate states, so the memory footprint stays small.

Table 6 lists MAE for each step from $H1$ to $H12$ on PEMS04. Error rises for every model as the horizon grows, which is expected because uncertainty builds up over time. EFSG stays best at all 12 steps. From $H1$ to $H12$ its MAE moves from 16.44 to 19.13 ($\Delta = 2.69$). That rise is smaller than DSTAGNN (17.40 to 21.03, $\Delta = 3.63$).

Taken together, the two tables show that EFSG cuts peak memory by a large margin and still gives the best MAE at every horizon on PEMS04.

5 Conclusions

In this paper we proposed EFSG for short-term traffic forecasting under non-stationary conditions. The model builds on temporal convolutions and uses Euler frequency modulation to adjust peak intensity and phase at each node. It also forms spectrum-aware links among locations that share similar fluctuation patterns so mutual influence is not limited to physical adjacency. Experiments on highway traffic and travel demand datasets support this design with clear gains. On NYC-Bike pick-up EFSG cuts MAE by 19.9% against TEDDCF. On PEMS04 it lowers peak GPU memory by 60% relative to the same baseline stays best at every horizon from one to twelve steps and keeps a slower error rise than heavier graph models as the window grows. Training converges within 20 epochs. Ablation on NYC-Taxi further shows that removing the full frequency branch hurts most that Euler modulation is the main operator inside that branch and that fixed high-pass filtering or inverted phase cannot replace learnable and correctly oriented amplitude-phase editing. Taken together these results show that Euler frequency modulation with spectrum-aware propagation is the core stack behind the gains. EFSG improves non-stationary peak timing and intensity modeling and it offers a effective route for spatiotemporal traffic forecasting in this setting.

AI use statement

In this work, we used generative AI tools for polishing the language and improving the presentation of the manuscript, and for assisting with LaTeX typesetting and table formatting.

Reproducibility Statement

The full source code of EFSG (including all ablation variants, data preprocessing scripts, and training configurations) is submitted as supplementary material. Dataset download links, preprocessing steps, and directory structure are documented in `README.md`. Evaluation metrics follow standard definitions and are implemented in `util.py`.

References

Jie Bai, Jiarui Zhu, Yujiao Song, Liang Zhao, Zhiling Hou, Ruochen Du, and Haoyang Li. A3T-GCN: Attention temporal graph convolutional network for traffic forecasting. *ISPRS International Journal of Geo-Information*, 10(7):485, 2021.

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 (NeurIPS)*, volume 33, pp. 17804–17815, 2020.

Narendra Singh Chauhan and Naresh Kumar. Confined attention mechanism enabled recurrent neural network framework to improve traffic flow prediction. *Engineering Applications of Artificial Intelligence*, 136:108791, 2024.

Jeongwhan Choi, Hwangyong Choi, Jeehyun Hwang, and Noseong Park. Graph neural controlled differential equations for traffic forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 6367–6374, 2022.

Harris Drucker, Christopher J. C. Burges, Linda Kaufman, Alexander Smola, and Vladimir Vapnik. Support vector regression machines. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 9, 1996.

Yuchen Fang, Yanjun Qin, Haiyang Luo, Fang Zhao, Bingbing Xu, Liang Zeng, and Chenxing Wang. When spatio-temporal meet wavelets: Disentangled traffic forecasting via efficient spectral graph attention networks. In *39th IEEE International Conference on Data Engineering (ICDE)*, pp. 517–529, 2023.

Zheng Fang, Qing 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 and Data Mining*, pp. 364–373, 2021.

Shengnan Guo, Youfang Lin, Ning Feng, Chao Song, and Huaiyu Wan. Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 922–929, 2019.

Shengnan Guo, Youfang Lin, Huaiyu Wan, Xiucheng Li, and Gao Cong. Learning dynamics and heterogeneity of spatial-temporal graph data for traffic forecasting. *IEEE Transactions on Knowledge and Data Engineering*, 34(11):5415–5428, 2021.

Yu Han, Xinyuan Zhou, Ruiyi Lu, Zhiang Hou, Yao Ren, Wenwu Wang, and Shiyong Lan. Stgf-mamba: Spatio-temporal graph fourier-enhanced mamba for traffic prediction. *Information Sciences*, 752:123601, 2026. ISSN 0020-0255. doi: 10.1016/j.ins.2026.123601. URL `https://www.sciencedirect.com/science/article/pii/S0020025526005323`.

S. M. Hosseini, S. M. Rahmatinia, and S. A. Hosseini-Seno. Integrated spatio-temporal modeling with hybrid graph convolutions and the graph fourier neural operator for traffic prediction. *Scientific Reports*, 16:12945, 2026. doi: 10.1038/s41598-026-38563-y.

Jiawei Jiang, Chengkai Han, Wayne Xin Zhao, and Jingyuan Wang. PDFormer: Propagation delay-aware dynamic long-range transformer for traffic flow prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 4365–4373, 2023.

Shiyong Lan, Yitong Ma, Weikang Huang, Wenwu Wang, Hao Yang, and Pyang Li. DSTAGNN: Dynamic spatial-temporal aware graph neural network for traffic flow forecasting. In *International Conference on Machine Learning (ICML)*, pp. 11906–11917, 2022.

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

H. Li, J. Liu, S. Han, J. Zhou, T. Zhang, and C. L. P. Chen. STFGCN: Spatial–temporal fusion graph convolutional network for traffic prediction. *Expert Systems with Applications*, 255:124648, 2024.

Mengzhang Li and Zhanxing Zhu. Spatial-temporal fusion graph neural networks for traffic flow forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 4189–4196, 2021.

Yaguang Li, Ruining Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. *arXiv preprint arXiv:1707.01926*, 2017.

Ao Liu and Yuzhou Zhang. Spatial–temporal dynamic graph convolutional network with interactive learning for traffic forecasting. *IEEE Transactions on Intelligent Transportation Systems*, 25(7): 7645–7660, 2024.

Hangchen Liu, Zheng Dong, Renhe Jiang, Jiewen Deng, Jinliang Deng, Quanjun Chen, and Xuan Song. Staeformer: 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.

Zonghui Lu, Chao Zhou, Jin Wu, Hong Jiang, and Shuguang Cui. Integrating granger causality and vector auto-regression for traffic prediction of large-scale WLANs. *KSII Transactions on Internet and Information Systems*, 10(1):136–151, 2016.

Jingtian Ma, Jingyuan Wang, and Leong Hou U. Hierarchical frequency-decomposition graph neural networks for road network representation learning. In *Proceedings of the Fortieth AAAI Conference on Artificial Intelligence (AAAI-26)*, 2026. doi: 10.48448/hgyw-v321.

Yao Ren, Wujiang Zhu, Shiyong Lan, Xinyuan Zhou, Hongyu Yang, and Zhiang Hou. Fe-hgat: Frequency-enhanced hybrid graph attention network for traffic prediction. In *2025 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 5173–5178, 2025. doi: 10.1109/SMC58881.2025.11342712.

Zezhi Shao, Zhao Zhang, Fei Wang, Wei Wei, and Yongjun Xu. Spatial-temporal identity: A simple yet effective baseline for multivariate time series forecasting. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management (CIKM)*, 2022a.

Zezhi Shao, Zhao Zhang, Wei Wei, Fei Wang, Yongjun Xu, Xin Cao, and Christian S. Jensen. Decoupled dynamic spatial-temporal graph neural network for traffic forecasting. *Proceedings of the VLDB Endowment*, 15(11):2733–2746, 2022b.

Chao Song, Youfang Lin, Shengnan Guo, and Huaiyu Wan. Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 914–921, 2020.

Yuxing Sun, Guang Zhang, and Hong Yin. Passenger flow prediction of subway transfer stations based on nonparametric regression model. *Discrete Dynamics in Nature and Society*, pp. 397154, 2014.

Guoqing Teng, Han Wu, Hao Wu, Jiahao Cao, and Meng Zhao. Frequency-aware and interactive spatial-temporal graph convolutional network for traffic flow prediction. *Applied Sciences*, 15 (20):11254, 2025. ISSN 2076-3417. doi: 10.3390/app152011254. URL https://www.mdpi.com/2076-3417/15/20/11254.

Billy M. Williams and Lester A. Hoel. Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results. *Journal of Transportation Engineering*, 129(6):664–672, 2003.

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

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.

Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S. Yu. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1):4–24, 2021.

Y. Xu, L. Wei, and L. Feng. A spatial-temporal trend-event decoupling dual-channel framework for traffic flow prediction. *Expert Systems with Applications*, 287, 2025.

Junchen Ye, Leilei Sun, Bowen Du, Yanjie Fu, and Hui Xiong. Coupled layer-wise graph convolution for transportation demand prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 4617–4625, 2021.

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

Ahmad Zeb, Junyu Zheng, Yuan Ye, Jiacheng Chen, Shanghui Zhang, Xinyu Wei, and James J. Q. Yu. Traffic forecasting with meta attentive graph convolutional recurrent network. *Expert Systems with Applications*, 275:128073, 2025.

Qi Zhang, Jianlong Chang, Gaofeng Meng, Shiming Xiang, and Chunhong Pan. Spatio-temporal graph structure learning for traffic forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 1177–1185, 2020.

Chuanpan Zheng, Xiaoliang Fan, Cheng Wang, and Jianzhong Qi. GMAN: A graph multi-attention network for traffic prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 1234–1241, 2020.

Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, volume 162 of *Proceedings of Machine Learning Research*, pp. 27268–27286, 2022.

A HYPERPARAMETERS

Regarding Optuna, hyperparameters were selected in a coarse-to-fine manner: broad ranges were explored first, then repeatedly narrowed around promising settings. For each dataset, we conducted at least 10 coarse-to-fine tuning rounds. Across the full tuning process, we recorded 193 Optuna trials in total. The final search spaces used in the last-stage studies were as follows.

- **PEMS03:** channels $\in \{80, 96\}$; lr $\in [1.1 \times 10^{-3},\ 2 \times 10^{-3}]$ (log); weight_decay $\in [1 \times 10^{-5},\ 5 \times 10^{-4}]$ (log); dropout $\in \{0.10, 0.20\}$.
- **PEMS04:** channels $\in \{80, 96\}$; lr $\in [1 \times 10^{-3},\ 2 \times 10^{-3}]$ (log); weight_decay $\in [3 \times 10^{-5},\ 6 \times 10^{-5}]$ (log); dropout $\in \{0.30, 0.35, 0.40\}$.
- **NYC-Bike / NYC-Taxi:** channels $\in \{64, 96, 128\}$; lr $\in [1 \times 10^{-3},\ 3.8 \times 10^{-3}]$ (log); weight_decay $\in [7 \times 10^{-6},\ 1 \times 10^{-3}]$ (log); dropout $\in \{0.05, 0.10, \ldots, 0.30\}$ (step 0.05); $k \in \{8, 10, 15\}$; gc_gate_init $\in \{0.1, 0.2, 0.3\}$; graph_alpha_init $\in \{0.01, 0.1, 0.3\}$.

B PARAMETER SENSITIVITY ANALYSIS

To systematically evaluate the robustness of our proposed EFSG, we conducted parameter sensitivity experiments across four datasets. We investigated three crucial hyperparameters: initial channel dimensions (Dim), the number of TCN layers, and the sparsity threshold in the spectrum-aware graph (Top-K). The variations in MAE and RMSE are illustrated in Figure 2.

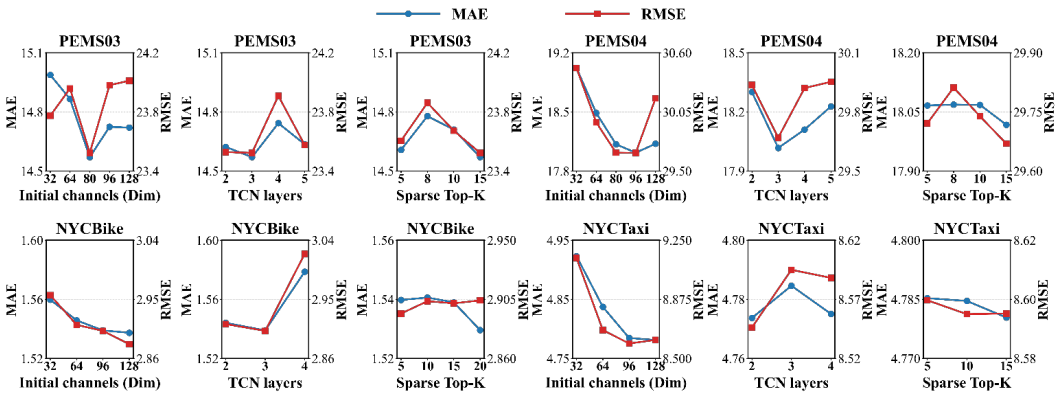


Figure 2: Parameter sensitivity analysis of the proposed EFSG model on PEMS03, PEMS04, NYCBike, and NYCTaxi datasets. The subplots demonstrate the impact of the initial channels (Dim), the number of TCN layers, and the Sparse Top-K value on MAE and RMSE.

Initial Channel Dimension (Dim) defines the representation capacity of the embeddings. As shown in Figure 2, on the highway datasets, the prediction errors form a distinct U-shape curve. The optimal performance is achieved at Dim=80 for PEMS03 (MAE: 14.59) and Dim=96 for PEMS04 (MAE: 18.01). Increasing the dimension beyond these thresholds (e.g., to 128) results in a slight performance degradation, indicating that excessive dimensions introduce redundant parameters and increase the risk of overfitting. Conversely, for the traffic demand datasets (NYCBike and NYCTaxi), the errors continue to descend smoothly, reaching the optimum at Dim=128. This suggests that complex urban demand patterns require higher-dimensional spaces for adequate representation.

Number of TCN layers dictates the temporal receptive field. For the PEMS03 and PEMS04 datasets, the absolute minimum of the error is achieved at 3 layers. Two layers provide an insufficient receptive field to capture long-term periodicities, whereas deeper networks (4 or 5 layers) cause sharp increases in error due to the over-smoothing issue and the ingestion of irrelevant distant-past noise. Interestingly, on the NYCTaxi dataset, the model peaks at 2 layers (MAE: 4.77), indicating that taxi demand relies more heavily on recent, short-term temporal dynamics rather than distant historical states.

Spectrum-Aware Graph Sparsity (Top-K) controls the density of the dynamically generated frequency-domain graph. Across PEMS03 and PEMS04, as K increases from 5 to 15, the error experiences slight initial fluctuations before dropping to its lowest point at K=15 (e.g., MAE: 14.57 for PEMS03). A smaller K (e.g., 5) arbitrarily truncates valuable global frequency correlations, starving the message-passing mechanism. However, for NYCBike, setting K=20 yields the best result (MAE: 1.53), implying that urban bicycle mobility heavily relies on wider, global spatial resonances across the city.