Fine-grained Temporal Learning in Traffic Flow Forecasting: The Power of Intraday Patterns

Shen Zhou
zhoushen@njnu.edu.cn
Nanjing Normal University
Nanjing, Jiangsu, China

Longwei Li
lilongwei@njnu.edu.cn
Nanjing Normal University
Nanjing, Jiangsu, China

Yehua Sheng*
shengyehua@njnu.edu.cn
Nanjing Normal University
Nanjing, Jiangsu, China

Abstract

Traffic flow forecasting is a core technology in Intelligent Transportation Systems (ITS). The fundamental challenge lies in effectively modeling the complex spatio-temporal dependencies in traffic data. Recent studies have shown that concise deep learning architectures with effective feature representations can achieve comparable performance to complex models. However, existing methods primarily rely on static temporal embeddings, making it difficult to fully capture the dynamic patterns of traffic flow variations throughout the day, which limits prediction accuracy. To address this issue, we propose a novel spatio-temporal modeling approach (ST-FTL) that focuses on fine-grained temporal learning to capture the dynamic patterns of traffic flow variations throughout the day. Specifically, we design a new fine-grained intraday pattern encoder that enables adaptive modeling of intraday temporal patterns through dynamic weight matrices. This encoder adopts a multi-level cascaded structure, enhancing feature expressiveness while maintaining linear computational complexity. Experiments on four widely-used public traffic datasets (PEMS03, PEMS04, PEMS07, and PEMS08) demonstrate that our proposed model outperforms existing baseline methods while maintaining significantly lower computational cost than existing attention-based methods.

CCS Concepts

• **Information systems → Data mining**.

Keywords

traffic flow forecasting, spatio-temporal modeling, intraday pattern learning, dynamic temporal feature, deep learning

ACM Reference Format:
Shen Zhou, Longwei Li, and Yehua Sheng. 2025. Fine-grained Temporal Learning in Traffic Flow Forecasting: The Power of Intraday Patterns. In *WEBST '25: International Workshop on Spatio-Temporal Data Mining from the Web, April 28–29, 2025, Sydney, Australia.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/XXXXXX.XXXXXX

*Corresponding author: Yehua Sheng (shengyehua@njnu.edu.cn)

ACM ISBN 978-1-4503-XXXX-X/2025/04
https://doi.org/10.1145/XXXXXX.XXXXXX

1 Introduction

Traffic flow forecasting, as a core component of Intelligent Transportation Systems (ITS), plays a crucial role in urban planning, traffic management, and mobility services. Accurate traffic flow forecasting can help alleviate traffic congestion, optimize route planning, and improve transportation efficiency [1, 5, 29]. However, due to the complexity and dynamics of transportation systems, achieving accurate traffic flow forecasting remains a significant challenge [3, 4, 12, 24].

Recent years have witnessed remarkable progress in deep learning approaches for traffic flow forecasting. As research deepens, the evolution of forecasting models has gone through several significant stages: Early recurrent neural networks achieved success in temporal feature extraction due to their sequence modeling capabilities [7, 17, 23, 26]; The introduction of spatio-temporal Graph Neural Networks (STGNNs) further enhanced the modeling of spatial dependencies [2, 14, 18, 21, 22, 25], though their complex structures and high computational overhead limited practical applications; Recently emerging Transformer architectures have provided new insights for spatio-temporal feature modeling through self-attention mechanisms [11, 15, 16, 27, 28]. However, these methods often pursue performance improvements through increasingly complex model architectures, potentially compromising their practicality. This trend of "more complex models are better" raises a critical question: Is there a more concise yet efficient solution?

Recently, Shao et al. proposed the spatio-temporal Identity (STID) method, which achieved comparable performance to complex models using only Multi-Layer Perceptrons (MLPs) by introducing spatial and temporal embeddings [19]. This finding suggests that effective feature representations might be more crucial than complex model architectures. However, STID relies on static temporal embeddings, making it difficult to fully capture the dynamic patterns of traffic flow variations throughout the day, which limits prediction accuracy. As shown in Figure 1, by comparing the daily traffic flow variations of typical nodes in PEMS04, PEMS07, and PEMS08 datasets, we can observe distinct temporal dynamic characteristics in traffic data. The flow volumes vary significantly across different time periods. These complex temporal variation patterns have a crucial impact on prediction accuracy and require fine-grained modeling approaches.

Based on this observation, we propose a novel spatio-temporal modeling approach (ST-FTL) that focuses on fine-grained temporal learning to capture the dynamic patterns of traffic flow variations throughout the day. Building upon STID's effective spatial-temporal identity framework, we enhance the temporal modeling capability through dedicated intraday pattern learning mechanisms. The main contributions of this paper are as follows:

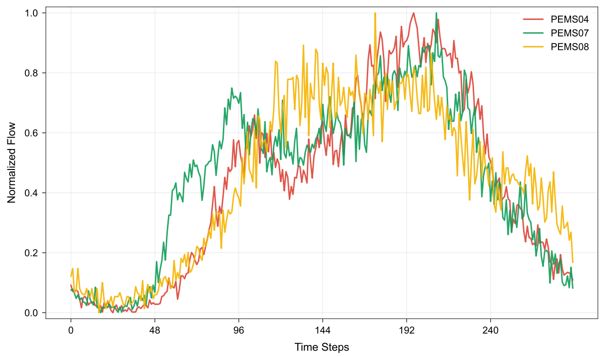

Figure 1: Daily traffic flow variations of typical nodes across different datasets. The x-axis represents time steps (5-minute intervals, 288 steps in total), and the y-axis represents normalized traffic flow.

- We propose an Intraday Pattern Block (IPB), a novel temporal learning module that enables adaptive modeling of intraday temporal patterns through dynamic weight matrices.
- We design a multi-level cascaded Intraday Pattern Encoder (IPE) based on the combination of MLP layers and IPB. This design achieves effective fine-grained temporal learning while maintaining model simplicity.
- We develop ST-FTL by integrating spatio-temporal identity embeddings with the proposed temporal learning mechanisms. Extensive experiments on four real-world datasets demonstrate that our model outperforms state-of-the-art methods while maintaining low computational overhead.

2 Methodology

2.1 Problem Definition

Consider a traffic network consisting of N nodes, where each node monitors and records traffic states in real-time. We denote the traffic flow data of N nodes over T discrete time steps as tensor $\mathbf{X} \in \mathbb{R}^{T\times N}$. The goal of traffic flow forecasting is to construct a mapping function f that utilizes observations from the past P time steps $\mathbf{X}_{input} \in \mathbb{R}^{P\times N}$ to predict traffic flow for the future F time steps $\mathbf{Y} \in \mathbb{R}^{F\times N}$.

Following STID's design, we incorporate two temporal semantic features in the input: Time in Day (TiD) and Day in Week (DiW) [19]. For a given input data $\mathbf{X}_{input}$, the corresponding temporal features can be represented as two normalized vectors $\mathbf{T}_{tid} \in \mathbb{R}^P$ and $\mathbf{T}_{diw} \in \mathbb{R}^P$, where the p-th element corresponds to the temporal information at time step $t-P+p$. Specifically, each element in $\mathbf{T}_{tid}$ represents the relative temporal position within a day, while each element in $\mathbf{T}_{diw}$ represents the relative day position within a week. For example, with a 5-minute sampling interval resulting in 288 time steps per day, the relative position of a time step within a day can be represented as $k/288$, where $k \in \{0, 1, \ldots, 287\}$; similarly, the relative position of a day within a week can be represented as $d/7$, where $d \in \{0, 1, \ldots, 6\}$.

Therefore, the traffic flow forecasting task can be formalized as the following mapping problem:

$$\mathbf{Y} = f(\mathbf{X}_{input}, \mathbf{T}_{tid}, \mathbf{T}_{diw}) \tag{1}$$

2.2 Model Architecture

ST-FTL is designed to effectively capture spatio-temporal dependencies in traffic flow data. The model consists of three key components: an embedding module, an IPE, and a regression layer. Figure 2 illustrates the overall architecture of the model.

2.2.1 *Embedding Module.*

The model builds upon the effective embedding strategy adopted by STID [19]. The embedding module aims to transform raw input data into high-dimensional representations while incorporating spatial and temporal identity information. For time series i at time step t, given input $\mathbf{X}^i_{t-P+1:t} \in \mathbb{R}^P$, we first transform it into an initial embedding $\mathbf{H}^i_t \in \mathbb{R}^D$ through a fully connected layer, where D is the hidden dimension:

$$\mathbf{H}^i_t = \mathrm{FC}_{embedding}(\mathbf{X}^i_{t-P+1:t}) \tag{2}$$

To capture spatial and temporal identity information, we define three learnable embedding matrices: spatial identity matrix $\mathbf{E} \in \mathbb{R}^{N\times D}$, TiD identity matrix $\mathbf{E}_{tid} \in \mathbb{R}^{N_d\times D}$, and DiW identity matrix $\mathbf{E}_{diw} \in \mathbb{R}^{N_w\times D}$. Here, N is the number of time series, N_d is the number of time steps in a day, and $N_w = 7$ represents the number of days in a week.

Following STID's design, we use the temporal index of the last time step in the input sequence as the temporal information for the entire time window [19]. Specifically, for input sequence $\mathbf{X}^i_{t-P+1:t} \in \mathbb{R}^P$, we extract the last element of its corresponding temporal information vectors $\mathbf{T}_{tid} \in \mathbb{R}^P$ and $\mathbf{T}_{diw} \in \mathbb{R}^P$:

$$\mathbf{T}^t_{tid} = \mathbf{T}_{tid}[P], \quad \mathbf{T}^t_{diw} = \mathbf{T}_{diw}[P] \tag{3}$$

Then, these temporal information are transformed into corresponding indices through linear mapping for extracting temporal features from the temporal embedding matrices:

$$\mathbf{E}^t_{tid} = \mathbf{E}_{tid}[\mathbf{T}^t_{tid} \cdot N_d], \quad \mathbf{E}^t_{diw} = \mathbf{E}_{diw}[\mathbf{T}^t_{diw} \cdot N_w] \tag{4}$$

For spatial embedding, we extract the corresponding embedding vector $\mathbf{E}^i = \mathbf{E}[i]$ from the spatial identity matrix for each node. Finally, by concatenating the time series embedding, spatial identity embedding, and temporal identity embeddings, we obtain the comprehensive representation for node i at time step t:

$$\mathbf{Z}^i_t = [\mathbf{H}^i_t; \mathbf{E}^i; \mathbf{E}^t_{tid}; \mathbf{E}^t_{diw}] \in \mathbb{R}^{4D} \tag{5}$$

2.2.2 *Intraday Pattern Encoder.*

The IPE serves as a novel deep learning architecture designed to effectively enhance dynamic modeling of intraday temporal patterns. The encoder extracts and reinforces temporal features through L cascaded layers, where each layer contains two core components: an MLP for feature extraction and an IPB for dynamic modeling of intraday temporal patterns.

Given the output $\mathbf{Z}^i_t \in \mathbb{R}^{4D}$ from the embedding module, the encoder extracts deep features through layer-by-layer processing. For the l-th layer, the feature transformation process can be formalized as:

$$(\mathbf{Z}^i_t)^{l+1} = \mathbf{IPB}_l(\mathbf{MLP}_l((\mathbf{Z}^i_t)^l), \mathbf{T}^t_{tid}) \tag{6}$$

where the MLP adopts a residual structure [9, 10]:

$$\mathbf{MLP}_l((\mathbf{Z}^i_t)^l) = \mathrm{FC}^l_2(\mathbf{Dropout}(\mathbf{GELU}(\mathrm{FC}^l_1((\mathbf{Z}^i_t)^l)))) + (\mathbf{Z}^i_t)^l \tag{7}$$

In each encoding layer, we design a dedicated Intraday Pattern Block that enables adaptive modeling of intraday temporal patterns through dynamic weight matrices. Specifically, for the l-th layer, we define learnable parameter matrix $\mathbf{W}^l_{tid} \in \mathbb{R}^{N_d\times 4D\times 4D}$ and

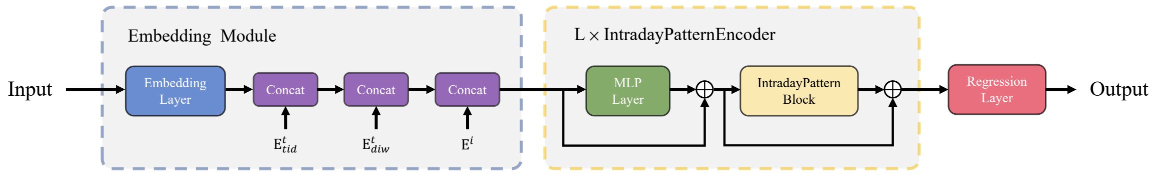

Figure 2: Architecture of the ST-FTL model. The model comprises three main components: (1) an embedding module that integrates spatial embeddings, TiD embeddings, and DiW embeddings; (2) L layers of IPE, each containing an MLP layer and an IPB with residual connections; (3) a regression layer for final prediction.

corresponding bias vector $\mathbf{b}^l_{tid} \in \mathbb{R}^{N_d \times 4D}$. This design enables each layer to independently learn temporal dynamic features at different levels of abstraction.

Given the temporal index, the block first performs dynamic feature transformation:

$$\mathbf{X}^l_{tid} = \text{LayerNorm}(\mathbf{W}^l_{tid}[\mathbf{T}^t_{tid}](\mathbf{Z}^i_t)^l + \mathbf{b}^l_{tid}[\mathbf{T}^t_{tid}]) \tag{8}$$

Then, through nonlinear transformation and residual connection [9, 10], the output of this layer is obtained:

$$(\mathbf{Z}^i_t)^{l+1} = \mathbf{Dropout}(\mathbf{GELU}(\mathbf{X}^l_{tid})) + (\mathbf{Z}^i_t)^l \tag{9}$$

2.2.3 Regression Layer and Model Training. After processing through L layers of the intraday pattern encoder, the learned features need to be mapped to the prediction space. To this end, we design a fully connected regression layer that transforms the final feature representation $(\mathbf{Z}^i_t)^L \in \mathbb{R}^{4D}$ into predictions for the future F time steps:

$$\hat{\mathbf{Y}}^i_{t+1:t+F} = \text{FC}_{regression}((\mathbf{Z}^i_t)^L) \tag{10}$$

where $\hat{\mathbf{Y}}^i_{t+1:t+F} \in \mathbb{R}^F$ represents the model's predictions for future time series.

To evaluate prediction accuracy and guide model training, we adopt Mean Absolute Error (MAE) as the loss function:

$$\mathcal{L}(\hat{\mathbf{Y}}, \mathbf{Y}) = \frac{1}{NF}\sum_{i=1}^{N}\sum_{j=1}^{F}|\hat{\mathbf{Y}}^i_j - \mathbf{Y}^i_j| \tag{11}$$

The model is trained using backpropagation and gradient descent methods by minimizing the loss function $\mathcal{L}$ to optimize all learnable parameters.

3 Experiments

3.1 Experimental Setup

Datasets. We evaluate ST-FTL on four widely-used public traffic flow datasets: PEMS03, PEMS04, PEMS07, and PEMS08, which are collected from the California Transportation Performance Measurement System (PeMS) with 5-minute sampling intervals [20]. The detailed statistics of these datasets are shown in Table 1. All datasets are preprocessed using Z-Score normalization.

Implementation Details. The experiments are conducted on an NVIDIA RTX 3090 GPU. Following STAEformer [15], we divide each dataset into training, validation, and test sets with a ratio of 6:2:2. The model takes 12 historical time steps (one hour) as input to predict the next 12 time steps. We evaluate the model performance using MAE, RMSE, and MAPE metrics.

Model Configurations. We employ a 3-layer IPE with a hidden dimension of 32. For training, we use the Adam optimizer [13] with an initial learning rate of 0.002 and weight decay of 0.0001. The learning rate follows a multi-step decay strategy, reducing by 50% at epochs 1, 25, 50, 75, 100, and 125. The batch size is set to 32, and the model is trained for 150 epochs.

Table 1: Dataset Statistics

Dataset	#Nodes	#Timesteps	Sample Rate	Time Span
PEMS03	358	26,209	5min	05/2012-07/2012
PEMS04	307	16,992	5min	01/2018-02/2018
PEMS07	883	28,224	5min	05/2017-08/2017
PEMS08	170	17,856	5min	07/2016-08/2016

Baselines. We compare our method with several representative baseline models, including: (1) time series prediction model HI [6]; (2) GNN-based models DCRNN [14], STGCN [25], GWNET [22], AGCRN [2], GTS [18], and MTGNN [21]; (3) Transformer-based models PDFormer [11] and STAEformer [15]; (4) other enhancement methods STNorm [8] and STID [19]. The performance of baseline methods is directly cited from STAEformer [15].

3.2 Performance Evaluation

Following mainstream evaluation practices, we systematically evaluate the average performance of all models across 12 prediction time steps. Table 2 presents the performance comparison between ST-FTL and baseline models on four public datasets. The best results are shown in bold, while the second-best results are underlined.

The experimental results demonstrate that ST-FTL outperforms existing baseline models across different datasets. On PEMS03 and PEMS04 datasets, ST-FTL achieves leading performance on key metrics. On PEMS07 dataset, ST-FTL's advantages are more pronounced, achieving the best results across all evaluation metrics. The consistent performance on PEMS08 dataset further validates the model's robustness and generalization ability.

3.3 Efficiency Analysis

To comprehensively evaluate the practicality of ST-FTL, we analyze its computational efficiency compared with other models. Table 3 shows the training time comparison among ST-FTL, the current best-performing STAEformer, and the baseline STID model across different datasets. All experiments are conducted under identical hardware conditions (NVIDIA RTX 3090 GPU).

The results show that ST-FTL maintains moderate computational cost while achieving competitive prediction performance. On PEMS07 dataset, ST-FTL requires only 8.9% of STAEformer's

Table 2: Performance comparison with baseline models on PEMS03, PEMS04, PEMS07, and PEMS08 datasets. The evaluation metrics include MAE, RMSE, and MAPE. Best results are shown in bold.

Dataset	PEMS03			PEMS04			PEMS07			PEMS08		
Metric	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
HI	32.62	49.89	30.60%	42.35	61.66	29.92%	49.03	71.18	22.75%	36.66	50.45	21.63%
GWNet	**14.59**	25.24	15.52%	18.53	<u>29.92</u>	12.89%	20.47	33.47	8.61%	14.40	23.39	9.21%
DCRNN	15.54	27.18	15.62%	19.63	31.26	13.59%	21.16	34.14	9.02%	15.22	24.17	10.21%
AGCRN	15.24	26.65	15.89%	19.38	31.25	13.40%	20.57	34.40	8.74%	15.32	24.41	10.03%
STGCN	15.83	27.51	16.13%	19.57	31.38	13.44%	21.74	35.27	9.24%	16.08	25.39	10.60%
MTGNN	14.85	<u>25.23</u>	<u>14.55%</u>	19.17	31.70	13.37%	20.89	34.06	9.00%	15.18	24.24	10.20%
GTS	15.41	26.15	15.39%	20.96	32.95	14.66%	22.15	35.10	9.38%	16.49	26.08	10.54%
STNorm	15.32	25.93	**14.37%**	18.96	30.98	12.69%	20.50	34.66	8.75%	15.41	24.77	9.76%
STID	15.33	27.40	16.40%	18.38	29.95	12.04%	19.61	32.79	8.30%	14.21	23.28	9.27%
PDFormer	14.94	25.39	15.82%	18.36	30.03	<u>12.00%</u>	19.97	32.95	8.55%	13.58	23.41	9.05%
STAEformer	15.35	27.55	15.18%	<u>18.22</u>	30.18	**11.98%**	<u>19.14</u>	<u>32.60</u>	<u>8.01%</u>	<u>13.46</u>	<u>23.25</u>	<u>8.88%</u>
ST-FTL	<u>14.73</u>	**24.39**	15.48%	**18.09**	**29.77**	12.19%	**18.82**	**32.14**	**7.88%**	**13.17**	**22.99**	**8.71%**

Table 3: Training Time per Epoch (seconds) for Different Models across Datasets

Dataset	PEMS03	PEMS04	PEMS07	PEMS08
STAEformer	151.61	85.00	531.40	49.13
STID	4.12	2.35	9.22	2.05
ST-FTL	18.88	10.71	47.03	5.80

training time. While ST-FTL introduces additional computations compared to STID due to its intraday pattern learning components, it maintains significantly lower computational overhead than attention-based methods.

The computational efficiency of ST-FTL primarily benefits from two aspects: first, it inherits STID's concise architectural design; second, IPB achieves adaptive modeling of intraday temporal patterns with minimal computational overhead through its dynamic weight matrices.

4 Conclusion

This paper proposes ST-FTL, a traffic flow forecasting model focusing on fine-grained temporal learning. The model introduces a novel IPB that achieves adaptive modeling of intraday temporal patterns through dynamic weight matrices. Based on this module, the model employs a multi-level cascaded encoder structure (IPE) for deep feature extraction.

Experiments on multiple real-world traffic datasets demonstrate the effectiveness of ST-FTL. The model achieves competitive prediction performance while maintaining a concise architecture design.

The significance of this research lies in revealing the importance of fine-grained temporal learning in traffic flow forecasting, providing new insights for model design. Future research directions include extending this approach to other spatio-temporal prediction tasks.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 42471463).

References

[1] Md Ashifuddin Mondal and Zeenat Rehena. 2019. Intelligent Traffic Congestion Classification System Using Artificial Neural Network. In *Companion proceedings of the 2019 world wide web conference*. 110–116.
[2] Lei Bai, Lina Yao, Can Li, Xianzhi Wang, and Can Wang. 2020. Adaptive Graph Convolutional Recurrent Network for Traffic Forecasting. In *Advances in Neural Information Processing Systems*, Vol. 33. 17804–17815.
[3] Azzedine Boukerche, Yanjie Tao, and Peng Sun. 2020. Artificial Intelligence-Based Vehicular Traffic Flow Prediction Methods for Supporting Intelligent Transportation Systems. *Computer Networks* 182 (Dec. 2020), 107484.
[4] Azzedine Boukerche and Jiahao Wang. 2020. Machine Learning-based Traffic Prediction Models for Intelligent Transportation Systems. *Computer Networks* 181 (Nov. 2020), 107530.
[5] Zhi (Aaron) Cheng, Min-Seok Pang, and Paul A. Pavlou. 2020. Mitigating Traffic Congestion: The Role of Intelligent Transportation Systems. *Information Systems Research* 31, 3 (Sept. 2020), 653–674.
[6] Yue Cui, Jiandong Xie, and Kai Zheng. 2021. Historical Inertia: A Neglected but Powerful Baseline for Long Sequence Time-series Forecasting. In *Proceedings of the 30th ACM international conference on information & knowledge management*. 2965–2969.
[7] Zhiyong Cui, Ruimin Ke, Ziyuan Pu, and Yinhai Wang. 2020. Stacked Bidirectional and Unidirectional LSTM Recurrent Neural Network for Forecasting Network-Wide Traffic State with Missing Values. *Transportation Research Part C: Emerging Technologies* 118 (Sept. 2020), 102674.
[8] Jinliang Deng, Xiusi Chen, Renhe Jiang, Xuan Song, and Ivor W. Tsang. 2021. 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*. 269–278.
[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.
[10] Dan Hendrycks and Kevin Gimpel. 2023. Gaussian Error Linear Units (GELUs). arXiv:1606.08415
[11] Jiawei Jiang, Chengkai Han, Wayne Xin Zhao, and Jingyuan Wang. 2023. PDFormer: Propagation Delay-Aware Dynamic Long-Range Transformer for Traffic Flow Prediction. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 37. 4365–4373.
[12] Sepideh Kaffash, An Truong Nguyen, and Joe Zhu. 2021. Big Data Algorithms and Applications in Intelligent Transportation System: A Review and Bibliometric Analysis. *International Journal of Production Economics* 231 (Jan. 2021), 107868.
[13] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations*.

[14] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. 2018. Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting. In *International Conference on Learning Representations*.

[15] Hangchen Liu, Zheng Dong, Renhe Jiang, Jiewen Deng, Jinliang Deng, Quanjun Chen, and Xuan Song. 2023. Spatio-Temporal Adaptive Embedding Makes Vanilla Transformer SOTA for Traffic Forecasting. In *Proceedings of the 32nd ACM international conference on information and knowledge management*. 4125–4129.

[16] Shizhan Liu, Hang Yu, Cong Liao, Jianguo Li, Weiyao Lin, Alex X. Liu, and Schahram Dustdar. 2022. Pyraformer: Low-Complexity Pyramidal Attention for Long-Range Time Series Modeling and Forecasting. In *International Conference on Learning Representations*.

[17] Zhongjian Lv, Jiajie Xu, Kai Zheng, Hongzhi Yin, Pengpeng Zhao, and Xiaofang Zhou. 2018. LC-RNN: A Deep Learning Model for Traffic Speed Prediction. In *IJCAI*. 3470–3476.

[18] Chao Shang, Jie Chen, and Jinbo Bi. 2021. Discrete Graph Structure Learning for Forecasting Multiple Time Series. In *International Conference on Learning Representations*.

[19] Zezhi Shao, Zhao Zhang, Fei Wang, Wei Wei, and Yongjun Xu. 2022. Spatial-Temporal Identity: A Simple yet Effective Baseline for Multivariate Time Series Forecasting. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 4454–4458.

[20] Chao Song, Youfang Lin, Shengnan Guo, and Huaiyu Wan. 2020. Spatial-Temporal Synchronous Graph Convolutional Networks: A New Framework for Spatial-Temporal Network Data Forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 34. 914–921.

[21] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, Xiaojun Chang, and Chengqi Zhang. 2020. Connecting the Dots: Multivariate Time Series Forecasting with Graph Neural Networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*. 753–763.

[22] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. 2019. Graph Wavenet for Deep Spatial-Temporal Graph Modeling. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. AAAI Press, 1907–1913.

[23] Bailin Yang, Shulin Sun, Jianyuan Li, Xianxuan Lin, and Yan Tian. 2019. Traffic Flow Prediction Using LSTM with Feature Enhancement. *Neurocomputing* 332 (March 2019), 320–327.

[24] Xueyan Yin, Genze Wu, Jinze Wei, Yanming Shen, Heng Qi, and Baocai Yin. 2022. Deep Learning on Traffic Prediction: Methods, Analysis, and Future Directions. *IEEE Transactions on Intelligent Transportation Systems* 23, 6 (June 2022), 4927–4943.

[25] Bing Yu, Haoteng Yin, and Zhanxing Zhu. 2018. Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*. 3634–3640.

[26] Haifeng Zheng, Feng Lin, Xinxin Feng, and Youjia Chen. 2021. A Hybrid Deep Learning Model With Attention-Based Conv-LSTM Networks for Short-Term Traffic Flow Prediction. *IEEE Transactions on Intelligent Transportation Systems* 22, 11 (Nov. 2021), 6910–6920.

[27] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. 2021. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 35. 11106–11115.

[28] Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. 2022. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *International conference on machine learning*. 27268–27286.

[29] Li Zhu, Fei Richard Yu, Yige Wang, Bin Ning, and Tao Tang. 2019. Big Data Analytics in Intelligent Transportation Systems: A Survey. *IEEE Transactions on Intelligent Transportation Systems* 20, 1 (Jan. 2019), 383–398.