Attention-Based Spatial-Temporal Graph Neural Network With Long-Term Dependencies for Traffic Speed Prediction

Quoc-Viet Nguyen, Chun-Yu Tai, Khanh-Duy Nguyen, Min-Te Sun, *Member, IEEE*, Wu-Yuin Hwang, Kazuya Sakai, *Member, IEEE*, and Wei-Shinn Ku, *Senior Member, IEEE*

***Abstract*— Urbanization, characterized by the continuous growth of population and density in urban areas, has led to the expansion and increased complexity of transportation networks, exacerbating traffic congestion. Accurate traffic speed prediction is crucial for effective traffic network management and planning. As the complexity of real-world road networks increases, integrating spatial and temporal information for accurate traffic speed prediction has become a challenging research task. This paper proposes a novel approach by introducing a spatial-temporal graph neural network (STGNN)-based model to enhance the accuracy of traffic speed prediction. By employing an attention-based STGNN, we effectively capture the complex relationships among road segments in real-world scenarios. We utilize the Huber loss as the training objective to improve prediction accuracy. Furthermore, we present an architecture that incorporates Root Mean Square Layer Normalization into the Transformer and integrates the Spatial-Temporal Attention Wavenet (STAWnet) into the model backbone, enabling faster training while maintaining model stability. We evaluate the proposed model using five real-world traffic speed benchmark datasets. The experimental results demonstrate that our method achieves superior performance compared to state-of-the-art traffic speed prediction approaches.**

***Index Terms*— Traffic speed prediction, attention-based spatial-temporal GNN, road segment, integrate spatial and temporal data.**

I. Introduction

As URBAN areas expand, traffic congestion has become a major challenge in large cities. A recent survey [1] estimates that traffic congestion cost Australian capital cities approximately $16.5 billion in 2015, with projections rising to $30 billion by 2030. Traditional mitigation methods, such as road construction, are often costly and difficult to implement. Accurate traffic speed prediction enables governments to optimize route planning and traffic control, thereby reducing congestion and air pollution. Hence, developing efficient and practical traffic speed prediction methods is crucial for modern Intelligent Transportation Systems (ITS) [2]. Integrating spatial and temporal information has become increasingly important for enhancing traffic speed prediction. Traditional models based solely on temporal data struggle to capture complex real-world conditions, particularly the subtle spatial dependencies across locations. As a result, RNN-based approaches [3], [4] often fail to fully model the interplay between spatial and temporal patterns, leading to suboptimal performance. Unlike standard time-series tasks, traffic speed at one location can be significantly influenced by conditions at other locations due to the interconnected nature of road networks. Therefore, our goal is to develop a model capable of learning both spatial and temporal dependencies from multivariate traffic time series. STGNNs [5] address this by representing road segments as nodes and their connections as edges. However, traditional STGNNs face limitations in computational efficiency and in capturing global dependencies, especially with long input sequences. We identify four key research challenges in traffic speed prediction:

1) **Long-Term Information Features**: Most STGNNs focus on short-term data (typically within an hour), ignoring potential similarities or inverse patterns across longer sequences. This neglect hinders the model's ability to capture underlying dependencies and makes it less robust to noise. Incorporating long-term features can lead to more stable and accurate predictions.
2) **Computational Complexity**: As input sequence length increases, computational cost grows linearly or quadratically. This significantly impacts training and real-time inference, limiting practical deployment in systems that require rapid response.
3) **Temporal Dependencies**: Real-world traffic conditions exhibit periodic fluctuations (rush hour, daily cycles, seasonal trends), leading to dynamic relationships between traffic data sequences of varying lengths. Effectively capturing these temporal dependencies is crucial for

Received 30 December 2024; revised 13 June 2025 and 15 July 2025; accepted 16 July 2025. Date of publication 31 July 2025; date of current version 3 November 2025. This work was supported in part by the National Science and Technology Council, Taiwan, under Grant NSTC 114-2218-E-008-008 and Grant NSTC 114-2221-E-008-092-MY2. The Associate Editor for this article was R. C. Carlson. *(Corresponding author: Min-Te Sun.)*

Quoc-Viet Nguyen, Khanh-Duy Nguyen, and Min-Te Sun are with the Department of Computer Science and Information Engineering, National Central University, Taoyuan 320, Taiwan (e-mail: vietnq@uit.edu.vn; nkduy@tvu.edu.vn; msun@csie.ncu.edu.tw).

Chun-Yu Tai is with the Department of Computer Science and Information Engineering, National Central University, Taoyuan 320, Taiwan, and also with BenQ Materials, Taoyuan 333403, Taiwan (e-mail: drew88995@gmail.com).

Wu-Yuin Hwang is with the Department of Computer Science and Information Engineering, College of Science and Engineering, National Dong Hwa University, Hualien 974301, Taiwan, and also with the Graduate Institute of Network Learning Technology, National Central University, Taoyuan 320, Taiwan (e-mail: wyhwang1206@gmail.com).

Kazuya Sakai is with the Department of Electrical Engineering and Computer Science, Tokyo Metropolitan University, Hino, Tokyo 191-0065, Japan (e-mail: ksakai@tmu.ac.jp).

Wei-Shinn Ku is with the Department of Computer Science and Software Engineering, Auburn University, Auburn, AL 36849 USA (e-mail: weishinn@auburn.edu).

Digital Object Identifier 10.1109/TITS.2025.3591264

accurate next-step prediction. Traditional traffic speed forecasting methods face challenges in this regard.

4) **Hidden Spatial Relationships**: The influence of neighboring locations changes dynamically under different traffic conditions. For example, nearby segments have a stronger impact during heavy congestion, while distant points like intersections or major arteries may also significantly affect traffic flow. Simple distance or connectivity metrics are insufficient to capture these complex relationships, as they require higher-level modeling. Traditional methods using static topologies fail to reflect such dynamic spatial dependencies.

To address these challenges, we propose a two-stage architecture consisting of a Long-Term Feature Extraction Stage and a Traffic Speed Prediction Stage. The first stage uses a Transformer-based unsupervised model trained with a masked autoencoding strategy to efficiently learn long-term traffic patterns. In the second stage, traffic speed is predicted using an STGNN enhanced with an attention mechanism to capture complex spatial-temporal dependencies. To improve efficiency, RMSNorm [6] is applied to reduce computational costs, and Huber Loss [7] is used during training to mitigate the impact of noise and outliers in the data. In summary, the contributions of this research are listed as follows:

- We propose a two-stage STGNN-based model for traffic speed prediction, which consists of a long-term feature extraction module and an attention-based STGNN for capturing complex spatial-temporal dependencies.
- We introduce an architecture that incorporates Root Mean Square Layer Normalization [6] into the Transformer [8] and integrates the Spatial-Temporal Attention Wavenet (STAWnet) [9] into the model backbone, allowing faster computation time while preserving model stability.
- We utilized Huber Loss [7] for training to mitigate the impact of noise and outliers on traffic speed data.
- We conducted extensive experiments on five real-world traffic benchmark datasets (METR-LA, PEMS-BAY, PEMS03, PEMS04, and PEMS08). The results indicate that our method outperforms existing state-of-the-art approaches.

II. Related Work

A. Temporal Models

Long Short-Term Memory (LSTM) networks [10] have become prominent in early traffic prediction research for their ability to capture long-term dependencies via gated mechanisms. The work [11] first applied LSTM to traffic speed prediction using 5-minute interval data from the California Department of Transportation Performance Measurement System (PeMS), achieving superior RMSE performance. Extending this work, Tian et al. [12] added temporal smoothness constraints to address missing data. Chen et al. [13] combined LSTM with the Kernel Bayes Rule (KBR) to model dependencies between speed counts and traffic speeds. Wang et al. [14] introduced DBL, integrating LSTM, residual connections, and deep hierarchies to enhance time-aware modeling. Shen et al. [15] proposed SeriesNet, blending LSTMs with dilated causal convolutions to reduce dimensionality and capture temporal dependencies at multiple scales.

B. Spatial-Temporal Models

Spatial-temporal models are widely used in traffic prediction due to the complex and nonlinear nature of traffic speed patterns. Traditional RNN-based methods often struggle with mid- to long-term forecasting, as they fail to capture intricate spatial and temporal dependencies. To address this, Yu et al. [16] propose STGCN, combining Graph CNNs and Gated CNNs to learn multi-scale spatiotemporal features. Ma et al. [17] use CNNs and Bi-directional LSTM (BiLSTMs) to extract spatial and temporal patterns from traffic images and time series. Li et al. [18] introduce an encoder-decoder framework for general spatiotemporal forecasting. Casas et al. [19] leverage GNNs to incorporate vehicle interactions into traffic prediction. Wu et al. [20] present a novel WaveNet architecture with GCNs and TCNs to capture spatial dependencies across time scales. Despite these advances, many RNN- and CNN-based models still face challenges in long-term forecasting, limiting their generalizability.

C. Spatial-Temporal Attention Models

The self-attention mechanism, a core component of the Transformer [8], was initially popularized in NLP and has since shown strong potential in traffic speed prediction. By considering all time steps in the input sequence, it effectively captures long-range dependencies and the dynamic nature of traffic, while its parallel processing allows efficient training on large datasets. Zhang et al. [21] combined attention mechanisms with Gated Recurrent Units (GRUs) to model both long-term and contextual information. Tian and Chan [9] integrated temporal convolutions and self-attention within a WaveNet framework to model spatial-temporal dependencies. Wang et al. [5] proposed STGNN, combining GRUs and self-attention to capture short- and long-term temporal dependencies, along with spatial relationships via attention-based adjacency matrices—achieving strong results on METR-LA and PEMS-BAY. More recently, Shao et al. [22] introduced a pre-trained STGNN model that learns temporal patterns from long sequences, improving segment-level representation and prediction accuracy.

III. Preliminary

A. Transformer

Transformer [8], a neural network architecture based on self-attention mechanisms, has gained widespread popularity due to its efficient parallel processing capabilities and ability to capture long-range dependencies. The core component of a Transformer is an encoder-decoder architecture, which employs self-attention mechanisms. Each layer in the encoder consists of a Multi-Head Attention module and a Feed-Forward Network (FFN). The Multi-Head Attention module comprises multiple self-attention heads that compute different attention weights to capture various relevant information. The FFN further processes each encoded output individually before passing it as input to the next encoder or decoder. The decoder, on the contrary, utilizes two Multi-Head Attention modules and an FFN. The first Multi-Head Attention module employs a masked operation to prevent the model from

accessing future information. The 2nd Multi-Head Attention and FFN modules mirror the functionality of their counterparts in the Encoder. Finally, a Softmax layer computes the probability of the next output. An Add with Norm layer follows each Multi-Head Attention and Feed-Forward Network, as we will discuss in the following section.

1) Layer Normalization (LayerNorm): In conventional deep neural networks, Batch Normalization [23] is widely used in deep neural networks to reduce internal covariate shift [8]. However, in Transformer architectures, where inputs are high-dimensional and position-sensitive, Batch Normalization can disrupt positional information and is thus less suitable. LayerNorm [24] addresses this by computing the mean and variance across features for each individual sample, preserving positional information. Its independence from batch size and lower sensitivity to input order make LayerNorm more appropriate for Transformers.

2) Root Mean Square Layer Normalization (RMSNorm): RMSNorm is a simplified normalization method that normalizes summed inputs using only the root mean square, preserving the re-scaling invariance of LayerNorm while discarding re-centering invariance, which contributes less to model training. By removing the mean calculation, RMSNorm significantly reduces computational overhead and can serve as a drop-in replacement for LayerNorm in various architectures. In contrast to LayerNorm, RMSNorm shows better generalization, surpassing the baseline model without any normalization by 0.013%, and reduces training time by approximately 20.5% in the image classification task. Additionally, RMSNorm accelerates training by 7% to 64% compared to LayerNorm, depending on the model architecture, dataset, and hardware setup. These properties make it particularly suitable for long-sequence, resource-intensive tasks like traffic speed prediction [6].

B. Masked Autoencoder

The Masked Autoencoder (MAE) [25] is initially constructed as a self-supervised model based on the Vision Transformer (ViT), learning to reconstruct the original uncorrupted signal by masking the input information. MAE employs an asymmetric encoder-decoder architecture. The encoder linearly maps the input with positional embeddings to a latent representation space, processing only the unmasked information. The decoder, on the contrary, reconstructs the original input from these latent features and the masked tokens and is only used for the reconstruction task during pre-training. MAE enables efficient training of large models and outperforms supervised pre-training.

C. Graph for Time Series

Graph for Time Series (GTS) [26] is a time series forecasting model. GTS adopts a unilevel optimization approach to jointly learn the graph structure and prediction model parameters, circumventing the high computational cost of existing bilevel optimization methods. GTS employs a feature extractor to obtain feature vectors fed into the link predictor to generate a link probability matrix (sample graph). The input data and the previously learned link probability are fed into the combined temporal recurrent processing with the graph convolution module for learning and prediction. This unified optimization scheme integrates both the graph structure parameters and the prediction model parameters and incorporates regularization into the loss function to prevent excessive deviations from known prior structures, thereby enhancing prediction quality.

D. Spatial-Temporal Attention Wavenet

Spatial-Temporal Attention Wavenet (STAWnet) [9] is a multi-step prediction model. It aims to capture the latent spatial relationships within the data through a self-learned node embedding technique. At the heart of STAWnet lies a WaveNet architecture composed of a Gated Temporal Convolution Network (TCN) [27] and a Dynamic Attention Network (DAN) [9], enabling it to handle spatial-temporal dependencies across different time scales effectively. The Gated TCN, composed of dilated causal convolutions, outperforms traditional RNNs in handling long-range sequences. Parallel computation within the TCN effectively mitigates the gradient explosion problem. In addition, Gating Mechanisms are employed within the TCN layers to regulate information flow. Following that, STAWnet incorporates a DAN to handle dynamic variations in influence from other roads due to changing traffic conditions.

IV. Design

The proposed system architecture is illustrated in Figure 1. The system comprises two stages: the Long-term Features Stage and the Traffic Speed Prediction Stage. In the Long-term Features Stage, the model learns through a self-supervised task by reconstructing randomly masked patches from long-term traffic sequence data. This process enables the model to capture periodic patterns in long-term data, which subsequently aids in predictions in the next stage. In the Traffic Speed Prediction Stage, the long-term contextual information extracted by the model is incorporated into the graph structure learner, allowing for more accurate predictions of short-term traffic sequence data. The following sections will provide a detailed explanation of each of these building blocks in the proposed system:

A. Dataset Explanation

Following previous works [22], [28], [29], [30], we utilize five real-world traffic speed datasets.

- **METR-LA** is a traffic speed dataset collected from loop detectors located on freeways in Los Angeles County, provided by the Performance Measurement System (PeMS) of the California Department of Transportation (CalTrans) [31]. The data collection span for this dataset ranges from four to six months and includes 207 selected sensors. The traffic data is recorded every 5 minutes [18].
- **PEMS-BAY** is a traffic speed dataset collected from CalTrans [31]. It includes 325 selected sensors, and the data collection spans six months. The traffic data is recorded every 5 minutes [18].
- **PEMS03** is a traffic speed dataset [32]. It contains 358 selected sensors, and the data collection spans from

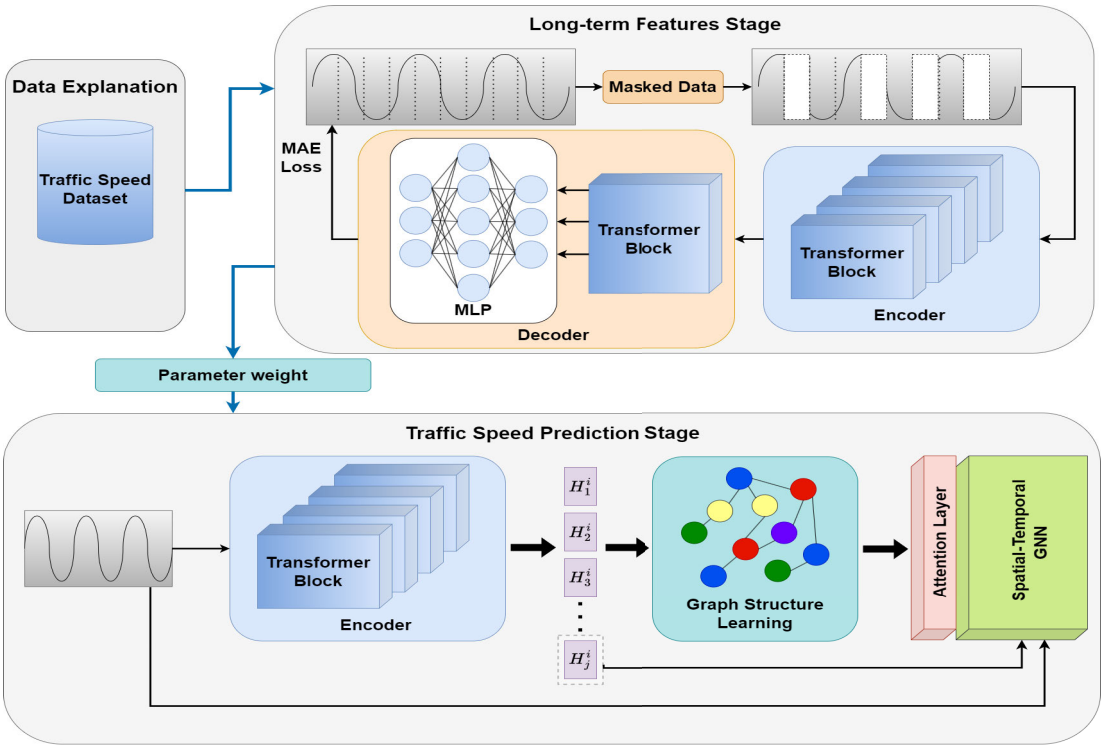

Fig. 1. Proposed system architecture.

September 1, 2018, to November 30, 2018. The traffic information is also recorded every 5 minutes.

- **PEMS04** is also a traffic speed dataset collected from CalTrans [32]. This dataset contains 307 sensors, and the data collection spans from January 1, 2018, to February 28, 2018. The traffic information is recorded every 5 minutes.
- **PEMS08** is also a traffic speed dataset collected from CalTrans [32]. This dataset contains 170 sensors, and the data collection spans from July 1, to August 31, 2016. The traffic information is recorded every 5 minutes.

These datasets consist of multiple routes, each with a number of loop detectors. We use these diverse and complex traffic speed datasets to demonstrate the accuracy and reliability of our proposed system. A loop detector records the average speed every 5 minutes. Consequently, each loop detector produces 12 records per hour, which we denote as a section. We define a patch as a collection of sections from all loop detectors within the same hour. Subsequently, we define a segment as the collection of records from all loop detectors over a time period of one week. Following the work [33], we perform Z-score normalization on the raw inputs.

B. The Long-Term Features Stage

Figure 2 illustrates the details of the Long-term Features Stage, which is designed to capture long-range temporal dependencies in traffic sequence data. This stage utilizes a masked autoencoder architecture composed primarily of Transformer blocks. It is divided into three key components: Masked Data, Encoder, and Decoder, each described below.

1) Masked Data: As illustrated in Figure 2, we denote the input sequence for node i as S_i, which is divided into P non-overlapping patches, each of length L. These input sequences are generated from the original time series data using a sliding window of length $P \times L$. The j-th patch within node i's sequence is represented as S_{ij}, where each patch $S_{ij} \in \mathbb{R}^{L \times C}$ captures local temporal information, C represents the number of input channels, and H_{ij} represents the generated representation for each S_{ij} (we set $P = 168$, and $L = 12$ follows the commonly adopted input length used in STGNN-based models [18], [20]). Subsequently, we randomly mask a subset of input patches with an optimal masking ratio of 55% creating a challenging reconstruction task for the model [25]. During the long-term feature stage, the encoder only receives the remaining 45% of unmasked patches and is trained to reconstruct the masked segments based on the visible context. This approach encourages the model to learn meaningful temporal patterns and spatial dependencies. Importantly, using patches instead of individual points provides richer semantic information, aligns with downstream STGNNs that operate on segments, and reduces the input sequence length, thereby improving training efficiency.

2) Encoder: The encoder first processes the unmasked original sequence segments through a linear layer (Input Embedding), followed by a learnable positional encoding layer to capture periodic features. The resulting sequences are then passed through Transformer layers [8] to generate output embeddings. As illustrated in Figure 3, RMSNorm [6] replaces LayerNorm to improve attention stability while reducing computational cost and memory usage. Given the more straightforward and readily interpretable nature of traffic speed data compared to language data, the Transformer Layers consist of only four layers, significantly lowering computational overhead.

3) Decoder: The Decoder takes masked segments from the encoder embeddings and reconstructs them back into their original sequence segments. It processes the entire set of patches, including both masked and unmasked tokens. After

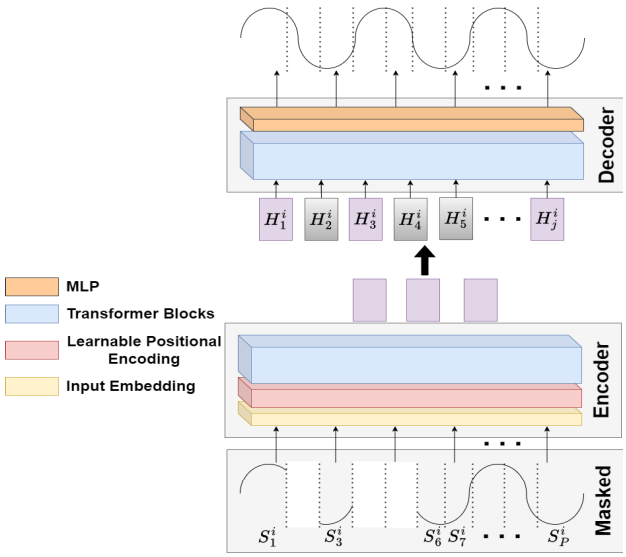

Fig. 2. Long-term features stage with masked autoencoding.

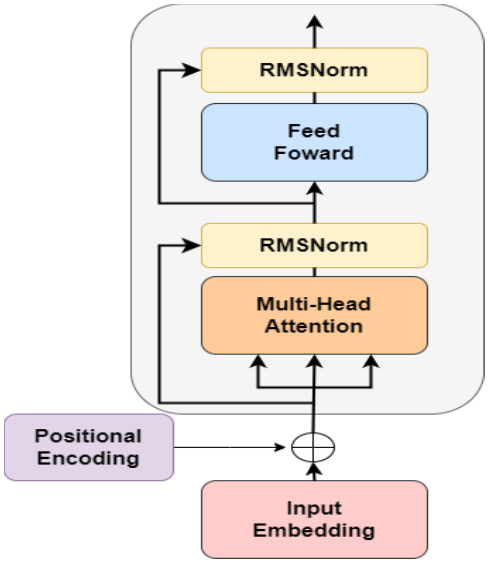

Fig. 3. The Transformer Layer with RMSNorm.

filling in the masked positions to align the number of patches with the original input length, the filled patches are fed into the Transformer Layer for information aggregation. Notably, the decoder is designed independently of the encoder and is used only during the pre-training phase for the sequence reconstruction task. Finally, several fully connected layers are used to predict future traffic speeds, formulating the task as a regression problem.

C. The Traffic Speed Prediction Stage

Figure 4 illustrates the details of the Traffic Speed Prediction Stage. This stage focuses on predicting future traffic speeds by feeding historical traffic time series data into a Transformer-based Encoder to enhance the prediction accuracy of the STGNN. The following sections delve into the three key components of this stage: the Encoder, Graph Structure Learning, and Spatial-Temporal Graph Neural Network.

1) Encoder: The encoder in the Traffic Speed Prediction Stage shares the same architecture as the Encoder in

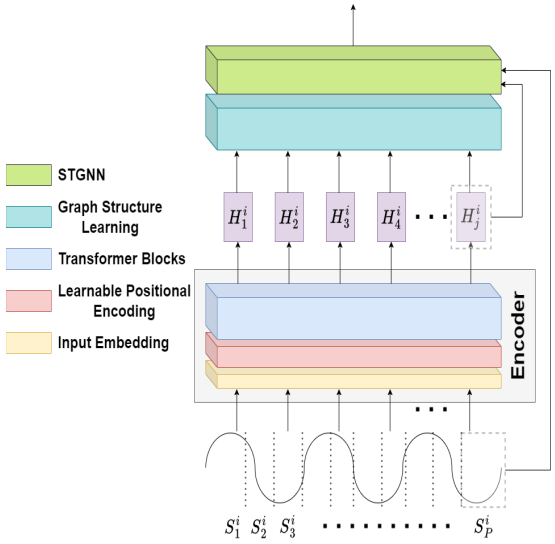

Fig. 4. Traffic speed prediction stage.

Section IV-B.2, with the primary difference being the absence of data masking in the input sequence. Additionally, this encoder leverages the long-term features learned from the Long-Term Features Stage to enhance the overall performance of the STGNN.

2) Graph Structure Learning: Graph Structure Learning is employed to capture the dependency graph among traffic time series. Regularization is utilized to extract supervised signals, enabling the computation of the K-Nearest Neighbor (KNN) graph that models interactions between time series and infers dependencies based on the hidden representations from the encoder output. To make the sampling process differentiable, the Gumbel-Softmax technique [34] is applied for reparameterization. Unlike many existing STGNNs that rely on predefined graphs, this approach learns the graph structure adaptively from the data. However, STGNN and graph structure learning are tightly coupled, and the lack of supervised loss in the graph structure learning process results in complex optimization. To mitigate this issue, we sample training data through a Transformer-based encoder, which helps decouple the learning process and facilitates more stable optimization. In the prior works [22], [26], a method for learning discrete sparse graphs is proposed. Following this, we represent the dependency relationships among all loop detectors by training a binary graph structure matrix $A \in \{0, 1\}^{n \times n}$, where n denotes the number of all loop detectors. The matrix A is directly parameterized as Θ_{ij}, and graph regularization is applied to provide supervisory signals for optimizing the graph structure. The KNN graph $\hat{A}$ is computed for hidden representation interactions, and its sparsity is controlled by setting $k \in N^+$. The cross-entropy between Θ and the KNN graph $\hat{A}$ is used as graph structure regularization, as shown in Equation 1.

$$L_{\text{Graphreg}} = \sum_{ij} -\hat{A}_{ij} \log \Theta'_{ij} - (1 - \hat{A}_{ij}) \log(1 - \Theta'_{ij}) \quad (1)$$

Since the normalized probability function is applied, we have $\Theta'_{ij} = \text{softmax}(\Theta_{ij}) \in \mathbb{R}$. Additionally, a vector g represents independent and identically distributed samples drawn from the Gumbel(0, 1) distribution. The temperature coefficient τ,

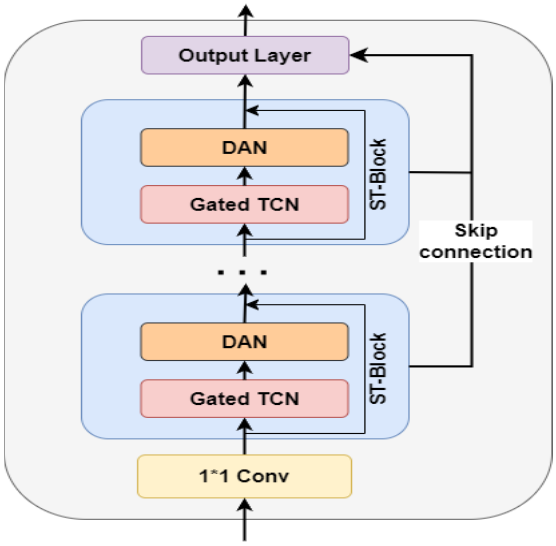

Fig. 5. The STAWnet architecture.

a tunable hyperparameter in the Gumbel-softmax function, controls the model's preference for different classes. Graph structure learning concludes with the Gumbel-Softmax parameterization, which transforms the discrete graph structure into a differentiable form. This allows the use of gradient-based optimization for training neural networks involving discrete decisions. The final dependency adjacency matrix A_{ij} is computed as: $A_{ij} = \text{softmax}\left(\frac{\Theta_{ij}+g}{\tau}\right)$

3) Spatial-Temporal Graph Neural Network (STGNN): In this paper, we choose STGNN [5] for the prediction task, as it effectively captures the complex dependencies inherent in time series data. STGNNs are well-suited for modeling spatial-temporal graph structures due to their ability to learn intricate node interactions over time and space. To enhance prediction performance, we incorporate long-term features extracted from the Long-Term Features Stage and the dependency graph obtained from Graph Structure Learning.

The masking strategy used during long-term feature extraction enables the model to learn global temporal patterns, while learnable positional embeddings, unlike traditional fixed sinusoidal ones, allow the Transformer blocks to better model periodic characteristics and long-range dependencies. The resulting contextual representations and the KNN graph $\hat{A}$ are then passed into STAWnet [9], our selected STGNN architecture, for downstream prediction. STAWnet combines Gated Temporal Convolutional Networks (TCNs) [27] with attention mechanisms to dynamically assign weights across loop detectors, effectively modeling spatial-temporal relationships. It takes as input the last patch and the learned dependency graph to generate latent hidden representations. These representations are transformed into a semantic space via a Multi-Layer Perceptron (MLP), followed by a regression layer to predict traffic speeds. Model performance is evaluated using the Mean Absolute Error (MAE) loss. The Gated TCN formula is given in Equation 2:

$$H^l_{out} = \tanh(\mathrm{W}_f * H^{l-1}_{out}) \odot \sigma(\mathrm{W}_g * H^{l-1}_{out}) \tag{2}$$

TABLE I
STATISTICS OF FIVE TRAFFIC SPEED DATASETS

Datasets	#Sensors	#Time Steps	Time Interval
METR-LA	207	5min	34272
PEMS-BAY	325	5min	52116
PEMS03	358	5min	26208
PEMS04	307	5min	16992
PEMS08	170	5min	17856

Here, H^l_{out} denotes the output of hidden representation in the l-th ST-block, H^{l-1}_{out} indicates the output of hidden representation in the $l-1$-th ST-block, W is the learnable convolution filter, $*$ denotes a convolution operator, $\odot$ denotes an element-wise multiplication operator, and σ represents the sigmoid activation function. The attention score formula is given in Equation 3:

$$\alpha_{i,j} = \frac{\exp\left(\frac{\langle W_q(h^l_i \| e_i), W_k(h^l_j \| e_j)\rangle}{\sqrt{d}}\right)}{\sum_{j \le N} \exp\left(\frac{\langle W_q(h^l_i \| e_i), W_k(h^l_j \| e_j)\rangle}{\sqrt{d}}\right)} \tag{3}$$

Here, $\alpha_{i,j}$ is the attention score indicating the importance of j-th loop detector to i-th loop detector, $\|$ denotes the concatenation operation, $\langle\rangle$ represents the inner product operation, e_i denotes the node embedding to capture hidden representation of the i-th loop detector, W_k and W_q are the key and query matrix in Dynamic Attention Network, and d indicates the dimension of $h^l_i \| e_i$. After the attention scores are obtained, the hidden state can be updated using Equation 4.

$$h'^l_i = \sum_{j \le N} \alpha_{i,j} \cdot h^l_j \tag{4}$$

4) Huber Loss: To further enhance the performance of our method, we adopt Huber Loss [7] as an alternative loss function during model pre-training. Specifically designed for regression tasks, Huber Loss combines the advantages of Mean Squared Error (MSE) and Mean Absolute Error (MAE), being sensitive to small errors like MSE while remaining robust to outliers like MAE. This makes it particularly suitable for real-world traffic speed data, which often contains noise or abrupt fluctuations due to sensor errors or external disruptions. By limiting the impact of large errors through a controlled transition between quadratic and linear behavior, Huber Loss improves model stability, convergence, and prediction accuracy during training. Additionally, prior work [32] has also demonstrated the effectiveness of adopting Huber Loss as a loss function for traffic speed prediction. The Huber Loss function is defined in Equation 5.

$$\text{Huber Loss} = \begin{cases} \frac{1}{2}(\hat{y}_i - y_i)^2 & \text{if } |\hat{y}_i - y_i| \le \delta \\ \delta \cdot (|\hat{y}_i - y_i| - \frac{1}{2}\delta) & \text{otherwise} \end{cases} \tag{5}$$

Here, δ denotes the delta hyperparameter, y_i is the actual value of the i-th sample, and $\hat{y}_i$ is the predicted value of the i-th sample.

TABLE II
PERFORMANCE ON THE **PEMS-BAY** DATASET. **BOLD** INDICATES THE BEST, AND UNDERLINE INDICATES THE SECOND BEST

Methods	Horizon 3			Horizon 6			Horizon 12		
	MAE	RMSE	MAPE (%)	MAE	RMSE	MAPE (%)	MAE	RMSE	MAPE (%)
GWNet [20]	1.30	2.73	2.71	1.63	3.73	3.73	1.99	4.60	4.71
GTS [26]	1.34	2.83	2.82	1.66	3.78	3.77	1.95	4.43	4.58
MTGNN [29]	1.32	2.79	2.77	1.65	3.74	3.69	1.94	4.49	4.53
PDFormer [30]	1.32	2.83	2.78	1.64	3.79	3.71	1.91	4.43	4.51
GMAN [28]	1.34	2.91	2.86	1.63	3.76	3.68	1.86	4.32	4.37
STEP [22]	1.26	2.73	2.59	1.55	3.58	3.43	1.79	4.20	4.18
STD-MAE [33]	**1.23**	**2.62**	**2.56**	1.53	3.53	3.42	1.77	4.20	4.17
Ours	1.24	2.65	2.59	**1.51**	**3.49**	**3.40**	**1.74**	**4.10**	**4.09**

V. PERFORMANCE

A. Datasets

We evaluate the effectiveness of our proposed model using five commonly used and publicly available real-world spatiotemporal benchmark datasets that are widely adopted in traffic speed prediction: PEMS-BAY and METR-LA [18], as well as PEMS03, PEMS04, and PEMS08 [32]. The traffic data in these datasets is recorded at 5-minute intervals. For a fair comparison, we follow the dataset splits used in previous studies [22], [28], [29], [30]. For METR-LA and PEMS-BAY, the datasets are divided into 70% for training, 20% for testing, and 10% for validation [18], [20]. For PEMS03, PEMS04, and PEMS08, we use approximately 60% for training, 20% for testing, and 20% for validation [32], [33]. The descriptions of the datasets are presented in Table I.

B. Baselines

We select several baseline methods for comparison with our proposed method. GWNet [20] is a deep learning-based approach. GTS [26], AGCRN [36], STNorm [35], GMAN [28], and MTGNN [29] are graph neural network-based methods. PDformer [30], STEP [22], and STD-MAE [33] are transformer-based models. For STEP [22] and STD-MAE [33], we pretrain the models with an input length of 2016. For the other baseline models, we obtain the results from their respective original works.

C. Evaluation Metrics

We evaluate the performance of our proposed model using three commonly used evaluation metrics for traffic speed prediction and time series forecasting: Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE) [18], [20], [22], [28].

D. Experimental Setup

Following the previous work [22], we set the number of patches to 168 for METR-LA, PEMS-BAY, PEMS03, and PEMS08 to capture one week of historical information from each dataset. For PEMS04, we set the number of patches to 336 to utilize two weeks of historical information. The number of attention heads within the Transformer blocks is set to 4, and the number of Transformer blocks in the Decoder and Encoder is set to 4 and 1, respectively. The optimization is performed using the Adam optimizer with a learning rate of 0.001 and the Masked Huber Loss. We also follow the previous work [22] to set the value of k in the KNN graph $\hat{A}$ to 10. In the implementation, we set the batch sizes for the Long-term Features Stage and the Traffic Speed Prediction Stage to 12 and 32, respectively. The number of epochs for both stages is set to 100. The optimal data masking ratio for the Long-term Features Stage is set to 55% to aid in learning long-term features. All experiments were conducted on a hardware setup featuring a 24-core 13th Gen Intel(R) Core(TM) i9-13900K CPU, an NVIDIA GeForce RTX 4090 GPU with 24GB of VRAM, and 64GB of memory. For the software configuration, we utilized Windows 11 in combination with PyTorch version 2.2.1 [37] and CUDA version 12.4. Additionally, for a fair comparison, we performed the experiments on the BasicTS platform [38].

E. Experimental Results and Analysis

This subsection presents the experimental results of our work in comparison to previous work on traffic speed prediction on five different datasets. First, we evaluate how well our model performs at different prediction time steps using the PEMS-BAY and METR-LA datasets. Specifically, Horizon 3, 6, and 12 refer to predicting traffic speeds at the 3rd, 6th, and 12th future time intervals [22], [33]. Table II reports traffic speed prediction results across different horizons on the PEMS-BAY dataset. Our method achieves the best performance for all horizons except Horizon 3, where it closely approaches the top-performing model, STD-MAE [33]. In addition, Table III shows prediction results across different horizons on the METR-LA dataset. Our method outperforms most baselines, including STEP [22], the state-of-the-art model on METR-LA dataset. For RMSE at Horizons 3 and 6, our method achieves results comparable to STEP [22], which utilized LayerNorm within the Transformer architecture. These results demonstrate the effectiveness of incorporating attention mechanisms into STGNN, with the performance gap widening as the prediction horizon increases. Second, on the PEMS03, PEMS04, and PEMS08 datasets, we compare the effectiveness of our model in terms of average MAE, RMSE, and MAPE for the entire horizons. As shown in Table IV, our method achieves the best performance in MAE and RMSE, and the second-best performance in MAPE on the PEMS03 dataset. For the PEMS04 dataset, our model also outperforms other baselines in MAE and RMSE, and achieves the second-best MAPE, demonstrating the effectiveness and generalizability of our approach across different traffic scenarios. On the PEMS08

TABLE III

PERFORMANCE ON THE **METR-LA DATASET**. **BOLD** INDICATES THE BEST, AND UNDERLINE INDICATES THE SECOND BEST

Methods	Horizon 3			Horizon 6			Horizon 12		
	MAE	RMSE	MAPE (%)	MAE	RMSE	MAPE (%)	MAE	RMSE	MAPE (%)
GWNet [20]	2.69	5.15	6.99	3.08	6.20	8.47	3.51	7.28	9.96
GTS [26]	2.67	5.27	7.21	3.04	6.25	8.41	3.46	7.31	9.98
MTGNN [29]	2.69	5.18	6.88	3.05	6.17	8.19	3.49	7.23	9.87
PDFormer [30]	2.83	5.45	7.77	3.20	6.46	9.19	3.62	7.47	10.91
GMAN [28]	2.80	5.55	7.41	3.12	6.49	8.73	3.44	7.35	10.07
STEP [22]	**2.61**	**4.98**	6.60	**2.96**	**5.97**	7.96	3.37	6.99	9.61
STD-MAE [33]	2.62	5.02	6.70	2.99	6.07	8.04	3.40	7.07	9.59
Ours	**2.61**	5.01	**6.59**	**2.96**	6.01	**7.92**	**3.35**	**6.95**	**9.58**

TABLE IV

PERFORMANCE ON **PEMS03**, **PEMS04**, AND **PEMS08** DATASETS. **BOLD** INDICATES THE BEST, AND UNDERLINE INDICATES THE SECOND BEST

Methods	PEMS03			PEMS04			PEMS08		
	MAE	RMSE	MAPE(%)	MAE	RMSE	MAPE(%)	MAE	RMSE	MAPE(%)
GWNet [20]	19.85	32.94	19.31	25.45	39.70	17.29	19.13	31.05	12.68
STNorm [35]	15.32	25.93	14.37	19.21	32.30	13.05	15.39	24.80	9.91
PDformer [30]	14.94	25.39	15.82	18.32	29.97	**12.10**	**13.58**	23.51	9.05
AGCRN [36]	15.07	26.88	15.80	19.26	31.16	12.65	15.95	25.22	10.09
STEP [22]	14.36	25.4	**14.08**	18.19	29.68	12.33	13.98	23.55	9.46
STD-MAE [33]	14.31	**25.21**	14.63	18.25	29.66	12.61	13.87	22.71	9.23
Ours	**14.18**	**25.21**	14.21	**18.01**	**29.49**	12.44	13.67	**22.49**	**8.95**

TABLE V

INFERENCE TIME PER SAMPLE COMPARISON (UNIT:MILLISECONDS). **BOLD** IS THE BEST, AND UNDERLINE INDICATES THE SECOND BEST

Datasets	**STEP [22]**	**Ours**	**STD-MAE [33]**
PEMS03	**16.35**	21.07	35.45
PEMS04	68.46	30.44	**25.89**
PEMS08	29.73	10.62	**5.42**

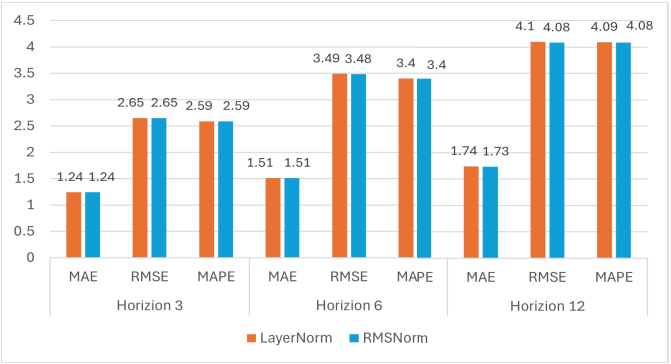

Fig. 6. Impact of LayerNorm and RMSNorm.

dataset, our model achieves the best results in RMSE and MAPE, and the second-best performance in MAE, indicating its strong ability to model spatial-temporal dependencies under varying traffic conditions. Overall, the results highlight the effectiveness of our proposed model in learning temporal patterns from long input lengths, whereas STD-MAE [33] performs well primarily with shorter input lengths.

Finally, we report the model's empirical speed. Specifically, we compare the inference time per sample of our proposed method with two state-of-the-art models for traffic speed prediction STEP [22] and STD-MAE [33] on the PEMS03, PEMS04, and PEMS08 datasets. The results, presented in Table V, show that while our method achieves the second-best inference time, it significantly outperforms both baselines across all evaluation metrics. These findings highlight that our model not only reduces computational cost but also maintains high predictive accuracy. In particular, the integration of RMSNorm enhances inference speed by simplifying the normalization process and reducing computational overhead. Despite these efficiency gains, model stability is preserved, as evidenced by consistently lower MAE, RMSE, and MAPE values across both short-term and long-term prediction horizons and across datasets.

F. Ablation Studies

In this section, we first evaluate the impact of the STAWnet backbone and Huber Loss components on model performance using the PEMS-BAY dataset. Then, we assess the impact of LayerNorm and RMSNorm on performance using the same dataset. Table VI displays the results of the model in predicting traffic speed for the PEMS-BAY dataset under different combinations of the components. STAWnet shows the most improvement across all horizons. Huber loss exhibits limited effectiveness for smaller horizons but demonstrates moderate performance gain for larger horizons. Obviously, combining the STAWnet backbone with Huber loss and RMSNorm yields more favorable results than using LayerNorm.

Finally, as shown in Figure 6, LayerNorm demonstrates good performance in terms of MAE, RMSE, and MAPE at smaller horizons. However, at larger horizons, RMSNorm achieves better results across all three metrics, highlighting its advantage in long-term prediction scenarios. These findings

TABLE VI
ABLATION STUDY ON THE PEMS-BAY DATASET. **BOLD** INDICATES THE BEST AND UNDERLINE INDICATES THE SECOND BEST

Methods	Horizon 3			Horizon 6			Horizon 12		
	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
GWNet + MAE	1.26	2.73	**2.59**	1.55	3.58	3.43	1.79	4.20	4.18
GWNet + Huber	1.26	2.68	2.63	1.55	3.55	**3.40**	1.78	4.16	4.12
STAWnet + MAE	**1.24**	**2.65**	**2.59**	**1.51**	3.49	**3.40**	1.74	4.10	4.09
STAWnet + Huber	**1.24**	**2.65**	**2.59**	**1.51**	**3.48**	**3.40**	**1.73**	**4.08**	**4.08**

are consistent with prior work [6], which shows that utilizing RMSNorm can significantly enhance model performance further supporting its effectiveness in our proposed method.

G. Discussion

Our proposed model demonstrates strong predictive performance and computational efficiency across five real-world traffic datasets. The integration of RMSNorm [6] and Huber Loss [7] in the long-term feature stage, along with the use of STAWnet [9] as the model backbone, significantly contributes to model performance and stability. Additionally, ablation studies show that using RMSNorm as a normalization method results in better inference time compared to LayerNorm across datasets. However, when compared to state-of-the-art models, our method achieves the second-best inference time while maintaining superior performance in terms of MAE, RMSE, and MAPE.

VI. CONCLUSION

In this study, we introduce a novel spatial-temporal graph neural network-based model to enhance the efficiency and accuracy of traffic speed prediction. By employing an attention-based STGNN, we effectively capture the intricate relationships between road segments in real-world traffic networks. To further improve prediction accuracy and reduce computational costs, we propose an architecture that incorporates RMSN [6] into the Transformer [8] and integrates the STAWnet [9] into the model backbone, enabling faster training while maintaining model stability. Our method is evaluated on five real-world traffic speed datasets and compared against several existing approaches. The evaluation results indicate that our model outperforms state-of-the-art methods across most scenarios. For future work, we aim to further improve prediction accuracy by incorporating additional data modalities, such as weather and landmark information, to enrich the model and enhance its predictive capabilities.

REFERENCES

[1] D. A. Tedjopurnomo, Z. Bao, B. Zheng, F. M. Choudhury, and A. K. Qin, "A survey on modern deep neural network for traffic prediction: Trends, methods and challenges," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 4, pp. 1544–1561, Apr. 2022.

[2] G. Dimitrakopoulos and P. Demestichas, "Intelligent transportation systems," *IEEE Veh. Technol. Mag.*, vol. 5, no. 1, pp. 77–84, Mar. 2010.

[3] Y. Kim, P. Wang, and L. Mihaylova, "Structural recurrent neural network for traffic speed prediction," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 5207–5211.

[4] X. Meng et al., "D-LSTM: Short-term road traffic speed prediction model based on GPS positioning data," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 3, pp. 2021–2030, Mar. 2022.

[5] X. Wang et al., "Traffic flow prediction via spatial temporal graph neural network," in *Proc. Web Conf.*, Apr. 2020, pp. 1082–1092.

[6] B. Zhang and R. Sennrich, "Root mean square layer normalization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 12381–12392.

[7] T. Hastie, R. Tibshirani, and J. H. Friedman, *The Elements of Statistical Learning*. New York, NY, USA: Springer, 2001.

[8] A. Vaswani et al., "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 5998–6008.

[9] C. Tian and W. K. Chan, "Spatial–temporal attention wavenet: A deep learning framework for traffic prediction considering spatial–temporal dependencies," *IET Intell. Transp. Syst.*, vol. 15, no. 4, pp. 549–561, Apr. 2021.

[10] A. Graves and A. Graves, "Long short-term memory," in *Supervised Sequence Labelling With Recurrent Neural Networks*. Berlin, Germany: Springer, 2012, pp. 37–45.

[11] Y. Tian and L. Pan, "Predicting short-term traffic flow by long short-term memory recurrent neural network," in *Proc. IEEE Int. Conf. Smart City/SocialCom/SustainCom (SmartCity)*, Dec. 2015, pp. 153–158.

[12] Y. Tian, K. Zhang, J. Li, X. Lin, and B. Yang, "LSTM-based traffic flow prediction with missing data," *Neurocomputing*, vol. 318, pp. 297–305, Nov. 2018.

[13] Z. Chen, J. Wen, and Y. Geng, "Predicting future traffic using hidden Markov models," in *Proc. IEEE 24th Int. Conf. Netw. Protocols (ICNP)*, Nov. 2016, pp. 1–6.

[14] J. Wang, F. Hu, and L. Li, "Deep bi-directional long short-term memory model for short-term traffic flow prediction," in *Proc. Int. Conf. Neural Inf. Process.*, Nov. 2017, pp. 306–316.

[15] Z. Shen, Y. Zhang, J. Lu, J. Xu, and G. Xiao, "SeriesNet: A generative time series forecasting model," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2018, pp. 1–8.

[16] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," 2017, *arXiv:1709.04875*.

[17] X. Ma, J. Zhang, B. Du, C. Ding, and L. Sun, "Parallel architecture of convolutional bi-directional LSTM neural networks for network-wide metro ridership prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 6, pp. 2278–2288, Jun. 2019.

[18] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," 2017, *arXiv:1707.01926*.

[19] S. Casas, C. Gulino, R. Liao, and R. Urtasun, "Spatially-aware graph neural networks for relational behavior forecasting from sensor data," 2019, *arXiv:1910.08233*.

[20] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang, "Graph WaveNet for deep spatial–temporal graph modeling," 2019, *arXiv:1906.00121*.

[21] J. Zhang, X. Shi, J. Xie, H. Ma, I. King, and D.-Y. Yeung, "GaAN: Gated attention networks for learning on large and spatiotemporal graphs," 2018, *arXiv:1803.07294*.

[22] Z. Shao, Z. Zhang, F. Wang, and Y. Xu, "Pre-training enhanced spatial–temporal graph neural network for multivariate time series forecasting," in *Proc. 28th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, Aug. 2022, pp. 1567–1577.

[23] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 448–456.

[24] J. Lei Ba, J. Ryan Kiros, and G. E. Hinton, "Layer normalization," 2016, *arXiv:1607.06450*.

[25] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 16000–16009.

[26] C. Shang, J. Chen, and J. Bi, "Discrete graph structure learning for forecasting multiple time series," 2021, *arXiv:2101.06861*.

[27] C. Lea, M. D. Flynn, R. Vidal, A. Reiter, and G. D. Hager, "Temporal convolutional networks for action segmentation and detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 156–165.

[28] C. Zheng, X. Fan, C. Wang, and J. Qi, "GMAN: A graph multi-attention network for traffic prediction," in *Proc. AAAI Conf. Artif. Intell.*, Apr. 2020, vol. 34, no. 1, pp. 1234–1241.

[29] Z. Wu, S. Pan, G. Long, J. Jiang, X. Chang, and C. Zhang, "Connecting the dots: Multivariate time series forecasting with graph neural networks," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2020, pp. 753–763.

[30] J. Jiang, C. Han, W. X. Zhao, and J. Wang, "PDFormer: Propagation delay-aware dynamic long-range transformer for traffic flow prediction," in *Proc. AAAI Conf. AI*, 2023, vol. 37, no. 4, pp. 4365–4373.

[31] C. Chen, K. Petty, A. Skabardonis, P. Varaiya, and Z. Jia, "Freeway performance measurement system: Mining loop detector data," *Transp. Res. Record: J. Transp. Res. Board*, vol. 1748, no. 1, pp. 96–102, Jan. 2001.

[32] C. Song, Y. Lin, S. Guo, and H. Wan, "Spatial–temporal synchronous graph convolutional networks: A new framework for spatial–temporal network data forecasting," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 914–921.

[33] H. Gao, R. Jiang, Z. Dong, J. Deng, Y. Ma, and X. Song, "Spatial–temporal-decoupled masked pre-training for spatiotemporal forecasting," 2024, *arXiv:2312.00516*.

[34] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," 2016, *arXiv:1611.01144*.

[35] J. Deng, X. Chen, R. Jiang, X. Song, and I. W. Tsang, "St-norm: Spatial and temporal normalization for multi-variate time series forecasting," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discov. Data Min.*, 2021, pp. 269–278.

[36] L. Bai, L. Yao, C. Li, and X. Wang, "Adaptive graph convolutional recurrent network for traffic forecasting," in *Proc. NIPS*, 2020, pp. 17804–17815.

[37] A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 8026–8037.

[38] Z. Shao et al., "Exploring progress in multivariate time series forecasting: Comprehensive benchmarking and heterogeneity analysis," *IEEE Trans. Knowl. Data Eng.*, vol. 37, no. 1, pp. 291–305, Jan. 2025.

Quoc-Viet Nguyen received the B.Sc. degree in information technology from the HCMC University of Education in 2014 and the M.Sc. degree in management information systems from the HCMC University of Technology, VNU-HCM, Vietnam, in 2020. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Information Engineering, National Central University, Taiwan. He is a Lecturer at the Faculty of Information Systems, HCMC University of Information Technology, VNU-HCM. His research interests include AI, deep learning, and data science.

Chun-Yu Tai received the master's degree in computer science and information engineering from National Central University in 2023. He is currently a Senior Engineer at BenQ Materials. His research interests include artificial intelligence, graph neural networks, and social media analysis.

Khanh-Duy Nguyen received the B.S. and M.S. degrees in information technology from Tra Vinh University, Vietnam, in 2019 and 2022, respectively. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Information Engineering, National Central University, Taiwan. His research interests include deep learning, web application development, and the Internet of Things.

Min-Te Sun (Member, IEEE) received the B.Sc. degree from National Taiwan University, the M.Sc. degree from Indiana University, Bloomington, IN, USA, and the Ph.D. degree in computer and information science from The Ohio State University. He is currently a Professor with the Department of Computer Science and Information Engineering, National Central University, Taiwan. His research interests include distributed computing and the IoT. He is a member of ACM.

Wu-Yuin Hwang is currently a Distinguished Professor with the Department of Computer Science and Information Engineering, College of Science and Engineering, National Dong Hwa University, and the Institute of Network Learning Technology, National Central University, Taiwan. His current research interests include the integration of the IoT, AI, and multimedia sensors of mobile devices for interactions among humans and all things in augmented reality contexts, such as smart agriculture, buildings, and campus.

Kazuya Sakai (Member, IEEE) received the Ph.D. degree in computer science and engineering from The Ohio State University in 2013. He is currently an Associate Professor at the Department of Electrical Engineering and Computer Science, Tokyo Metropolitan University. His research interests include information and network security, wireless and mobile computing, and distributed algorithms. He received the IEEE Computer Society Japan Chapter Young Author Award 2016. He is a member of ACM.

Wei-Shinn Ku (Senior Member, IEEE) received the M.S. degree in computer science and the M.S. degree in electrical engineering from USC in 2003 and 2006, respectively, and the Ph.D. degree in computer science from the University of Southern California (USC) in 2007. He was the Program Director of the National Science Foundation from 2019 to 2022. He is currently a Professor with the Department of Computer Science and Software Engineering at Auburn University. He has published more than 170 research papers in refereed international journals and conference proceedings. His research interests include databases, data science, mobile computing, and cybersecurity. He is a member of ACM.