*Mathematics*

AIMS Mathematics, 11(5): 13837–13864.
DOI: 10.3934/math.2026570
Received: 01 April 2026
Revised: 05 May 2026
Accepted: 11 May 2026
Published: 18 May 2026

https://www.aimspress.com/journal/Math

*Research article*

# Traffic speed sparse time series prediction model integrating spatiotemporal periodic features

**Shan Jiang[1,2], Yuming Feng[1,2,*] and Jiang Xiong[3]**

[1] School of Computer Science and Engineering, Chongqing Sanxia University of Science and Technology, Wanzhou, Chongqing 404100, China
[2] Key Laboratory of Intelligent Information Processing and Control, Chongqing Sanxia University of Science and Technology, Wanzhou, Chongqing 404100, China
[3] School of Mathematics and Statistics, Chongqing Sanxia University of Science and Technology, Wanzhou, Chongqing 404100, China

* **Correspondence:** Email: ymfeng@sanxiau.edu.cn.

**Abstract:** Sparse time series forecasting (SparseTSF) is a recently proposed lightweight multi-step forecasting model with advantages such as high computational efficiency and wide adaptability. However, the SparseTSF model also has some limitations, for example, it can only downsample a single main period, making it difficult to handle multi-period data. Based on the characteristics of traffic forecasting, this paper integrates weekly and spatial feature extraction modules to extract deep features that fuse daily, weekly, and spatial features. The SparseTSF model is then optimized to construct a multi-period spatial time series forecasting (MSTSF) model. This model is applied to traffic speed forecasting scenarios, aiming to reduce prediction error and achieve a balance between performance and parameter size. Experiments on the Guangzhou traffic and Performance Measurement System (PeMS) datasets show that the MSTSF model performs well in both prediction accuracy and model efficiency. Compared with mainstream deep learning models, this model achieves lower prediction errors. The MSTSF model has advantages such as small parameter size, high iteration efficiency, and fast inference speed, making it suitable for scenarios with limited computational resources. Our model achieves better results in traffic speed forecasting.

**Keywords:** Multi-step prediction; MSTSF; SparseTSF; periodic features; spatial features; traffic speed
**Mathematics Subject Classification:** 68T07

## 1. Introduction

Traffic speed prediction stands as one of the core tasks within Intelligent Transportation Systems (ITS); accurate predictions provide critical support for traffic signal optimization, route guidance, congestion management, smart mobility services, and urban transportation planning. However, traffic speed data inherently exhibits complex spatiotemporal dependencies: in the spatial dimension, the traffic states of adjacent and functionally similar road segments within a network mutually influence one another; in the temporal dimension, speed sequences manifest both continuous short-term fluctuations and significant periodic patterns, such as daily and weekly cycles. The central challenge in the field of traffic speed prediction lies in effectively capturing the spatial dependencies of the road network while simultaneously fully leveraging the latent periodic characteristics embedded within the speed sequences. Fundamentally, traffic speed prediction entails learning an optimal "prediction function" from historical spatiotemporal data. Over the course of nearly seven decades, this field has undergone a multi-stage evolution: transitioning from traditional model paradigms to deep learning models, and ultimately entering the current era, one that strikes a balance between lightweight efficiency and fine-grained precision.

### *1.1. Traditional methods*

Early traffic forecasting models were primarily based on statistical methods, such as the autoregressive integrated moving average (ARIMA) model [1] and the vector autoregression (VAR) model [2]. Founded on linear assumptions, these models treated traffic speed as a time series, generating predictions by capturing the autocorrelation and moving average components within the sequence. While they performed well under stable traffic conditions, they were capable of fitting only stationary traffic patterns and struggled to characterize the strong nonlinearity, spatiotemporal coupling, and stochastic disturbance features inherent in traffic flow. Among these, the ARIMA model was the most widely applied; it transforms non-stationary traffic time series into stationary ones through differencing operations, utilizing a combination of autoregressive and moving average mechanisms to characterize the linear temporal dependencies of traffic flow. Traditional machine learning models, including support vector regression (SVR) [3], random forest regression (RFR) [4], and K-nearest neighbors (KNN) [5], have also been widely applied in the field of traffic forecasting. These methods transform the prediction task into a regression problem by employing feature engineering techniques involving variables such as historical speeds, time-of-day identifiers, and traffic events. Although capable of capturing nonlinear relationships, these approaches suffer from limitations, such as lengthy computation times, the need for manual feature selection, and parameter constraints, that result in suboptimal performance within big data environments. Furthermore, they remain limited in their ability to identify dynamic and complex traffic patterns, making it difficult for them to adapt to the current trend of massive, real-time traffic data growth.

### *1.2. Deep learning methods*

Leveraging its powerful capability for automatically extracting nonlinear features, deep learning has broken through the limitations of linear modeling inherent in traditional models. Recurrent neural networks (RNNs) and their variants have emerged as core tools in this domain. The long short-term

memory (LSTM) network [6], an improved version of the RNN, addresses the vanishing gradient problem by introducing "memory cells" and gating mechanisms, thereby enabling the capture of long-range dependencies within traffic time series data. Convolutional neural networks (CNNs) constitute one of the foundational deep learning models for traffic prediction; their core strength lies in their robust capability for extracting spatial features. CNNs are frequently combined with time-series models, such as LSTMs, to form hybrid CNN-LSTM models, which enable the simultaneous extraction of both spatial and temporal features. Yan et al. [7] employed a hybrid CNN-LSTM model to effectively forecast short-term visitor flow in mountainous scenic areas; at a time granularity of 15 minutes, the model achieved an $R^2$ score of 0.92, enabling precise short-term visitor flow forecasting and accurate estimation of capacity requirements for such scenic areas during holiday periods. Temporal convolutional networks (TCNs) utilize causal and dilated convolutions to capture long-range sequence dependencies and support parallel computation, typically exhibiting higher training efficiency than RNNs. A framework named PSTA-TCN has been proposed, which incorporates a parallel spatiotemporal attention mechanism to extract dynamic internal correlations; by fully leveraging parallel computing, this framework significantly reduces training time while simultaneously achieving substantial improvements in prediction accuracy [8]. The dynamic pattern-aware spatiotemporal convolutional network (DPSTCN), which integrates an attention mechanism, effectively captures temporal patterns, including local and global dependencies, dynamics, and periodicity. In the spatial dimension, the model constructs static and dynamic pattern-aware convolutions that utilize geographic and regional functional information to effectively capture complex spatial patterns, such as dynamics and heterogeneity. Evaluations conducted on four different traffic benchmark datasets demonstrate that the model achieves superior results compared to various existing methods [9].

The Transformer [10] originally developed for natural language processing (NLP), captures global dependencies through self-attention mechanisms and positional encoding. When adapted for long-term time-series forecasting, its primary advantage lies in its ability to transcend the temporal dependency constraints inherent in traditional recurrent networks, allowing the model to directly attend to historical information at any given time step. This makes it particularly well-suited for long-term forecasting tasks spanning hundreds or even thousands of time steps; notable Transformer-based models designed for multi-step long-term time-series forecasting include FEDformer and Informer. To leverage their respective strengths while mitigating their weaknesses, one of the current research trends involves integrating these models with graph neural networks (GNNs), as exemplified by GAT-Informer [11]. The GAT-Informer model was evaluated using real-world data collected in London; experimental results demonstrated that, compared to other baseline models, this model exhibits superior performance in long-term traffic flow forecasting. Liu et al. [12] proposed the IEEAFormer model, an implicit information embedding and enhanced spatial-temporal multi-Head attention transformer. This model employs a novel parallel spatial self-attention architecture capable of simultaneously capturing both long-range and short-range dependencies within the data. Validation results across four real-world traffic datasets indicate that the proposed IEEAFormer outperforms most existing models in terms of predictive performance. The N-BEATS model, eschewing reliance on RNNs or CNNs, instead utilizes fully connected layers to learn and combine basis functions, such as trends and seasonality, to generate forecasts.

Characterized by its strong interpretability, N-BEATS has been successfully applied across various specific domains, including vehicle speed prediction, traffic accident duration prediction, and

hybrid electric vehicle speed forecasting. When integrated with an optimization algorithm within the ARDE-N-BEATS framework, the model achieves a high accuracy of at least 94% in the majority of predictions. Furthermore, it outperforms existing peer methods and features the capability to automatically tune model hyperparameters, thereby reducing computational overhead by up to 78.90% [13]. Cheng et al. [14] proposed a novel hybrid forecasting model, EEMD-TiDE, which combines improved ensemble empirical mode decomposition (EEMD) with the time series dense encoder (TiDE) to enhance predictive accuracy; this model has been successfully applied to short-term passenger flow forecasting in urban rail transit systems. Zuo et al. [15] introduced a hybrid methodology for forecasting subway passenger flow. This approach first employs variational mode decomposition (VMD) to decompose time-series data into intrinsic mode functions (IMFs), and subsequently utilizes TimesNet and SARIMA to perform adaptive forecasting for each individual IMF. This hybrid methodology demonstrates exceptional performance during peak hours, reducing the mean absolute error (MAE) by 52.75% and the root mean square error (RMSE) by 50.61%, and outperforms benchmark models such as Informer and Crossformer, achieving respective reductions of 66.14% and 63.24% in MAE and RMSE. KAN [16], a novel neural network paradigm proposed in 2024 and inspired by the Kolmogorov-Arnold representation theorem, has opened new avenues for time-series analysis in terms of interpretability and continual learning, and is rapidly emerging as a research hotspot in the field of traffic prediction. Research efforts in this area primarily focus on transcending the limitations of standalone applications and advancing toward the integration of multiple technologies. The innovative spatiotemporal decomposition learning architecture, STKAN [17], serves as a prime example of this trend. Extensive experimental evaluations conducted on widely recognized benchmark datasets have convincingly demonstrated that STKAN achieves superior predictive accuracy and interpretability, while clearly delineating the key spatial groupings and significant temporal patterns that exert a substantial influence on predictions.

### *1.3. Lightweighting methods*

To address the critical pain points of deep learning models, their high computational overhead, and the difficulties associated with edge deployment, lightweight traffic prediction models have emerged. With the core design objective of dynamically balancing prediction accuracy against inference efficiency, these models employ key techniques such as model structure reconstruction, parameter pruning and compression, spatiotemporal modeling decoupling, and multi-scale feature sparse distillation. By doing so, they significantly reduce the number of model parameters, computational complexity, and inference latency, while largely preserving prediction accuracy, thereby enabling adaptation to resource-constrained scenarios such as edge computing environments.

The first category of lightweighting methods comprises models based on minimalist backbone networks. These models eschew complex graph neural networks (GNNs) or recurrent neural network (RNN) structures, opting instead to utilize minimalist yet effective networks as their core architecture. Their primary advantage lies in their exceptional computational efficiency; in terms of both training and inference speeds, they typically outperform models reliant on graph message passing or recurrent computations by a wide margin, making them ideally suited for deployment on resource-constrained devices such as roadside units. However, they face limitations regarding their ability to capture complex spatial interactions; specifically, they may exhibit weaknesses in identifying primary

dependencies and adapting to dynamic changes. Representative models in this category include M³-Net [18] and STLinear [19]. Among these, M³-Net is a graph-free model based on multi-layer perceptrons (MLPs) that employs a mixture of experts (MoE) system to capture both the homogeneous and heterogeneous characteristics of traffic data. Its advantages include low deployment costs and the ability to effectively process the complex patterns inherent in traffic data; however, its generalization capability across different cities and varying road network topologies remains to be further validated. STLinear's core mechanism is built upon purely linear layers, which implicitly learn spatial features through node embeddings. Representing a minimalist model architecture designed to optimize both efficiency and performance, STLinear boasts extremely high operational efficiency; its computational requirements are reduced by over 95% compared to mainstream spatiotemporal graph neural networks (STGNNs), while its prediction accuracy remains comparable to, or even surpasses, that of leading STGNN models [19]. Its limitations stem from its inability to explicitly learn the complex spatial dependencies dictated by road network topologies; consequently, it may lack sufficient expressive power when confronted with nonlinear, non-stationary conditional shifts, such as sudden incidents or adverse weather conditions.

The second category comprises efficient architectural designs. These approaches preserve the model's robust spatiotemporal modeling capabilities while reducing computational overhead, thereby achieving linear or near-linear complexity, through the design of more efficient spatial or temporal modules. Representative models in this category include LSTNN [20], TLAST [21], and PASTNet [22]. LSTNN decomposes time series data using down-sampling and block-sampling strategies, modeling each component independently; its advantage lies in an architecture that is both simple and effective, avoiding the redundant computations often associated with complex structures. However, its limitation is that the sampling process may result in the loss of certain short-term temporal information. TLAST introduces a spatial proxy attention (SPA) module, which efficiently captures dynamic spatial dependencies with linear complexity; this approach outperforms existing state-of-the-art baseline models while reducing memory usage by 85.21% and computational time by 75.14% [21]. The core mechanism of the PASTNet model involves extracting a limited set of "prototypes" from massive traffic datasets to serve as prior knowledge guiding the prediction process, and employing "star-shaped" operations to construct a lightweight network. The key advantage here is that these "prototype" priors enable the model to focus on common traffic patterns, thereby effectively enhancing its generalization capabilities. Experimental results on large-scale, real-world traffic datasets demonstrate that PASTNet significantly outperforms existing advanced traffic prediction models in terms of both prediction accuracy and computational efficiency [22].

The third category involves compression and decomposition, utilizing mathematical methods to "slim down" models or data and thereby reduce resource consumption. Representative models in this category include DSFormer-LRTC [23], DFGNet [24], and STH-SepNet [25]. DSFormer-LRTC employs low-rank tensor compression techniques to compress the weights within the self-attention mechanism of a Transformer; this approach reduces the parameter count by two-thirds while maintaining predictive accuracy. Its limitation, however, is that the compression may compromise the model's ability to capture global dependencies [23]. DFGNet utilizes learnable frequency-domain decomposition to decouple high-frequency and low-frequency patterns within traffic signals, employing a dual-branch network to model each component separately. A lightweight adaptive fusion module dynamically balances these two branches, resulting in enhanced interpretability and robustness, and enabling the model to handle complex traffic conditions more effectively [24]. Chen

et al. [25] proposed a novel framework known as the spatio-temporal hypergraph separation network (STH-SepNet). By decoupling the spatio-temporal modeling process, this framework simultaneously boosts both efficiency and accuracy; it leverages dynamic hypergraphs to capture higher-order dependencies, thereby accelerating training speed threefold.

Current traffic speed prediction still faces numerous challenges, including the difficulty of modeling the spatiotemporal coupling relationships within complex road networks, insufficient prediction accuracy in scenarios involving sudden events or limited data samples, the high computational demands of large-scale models, and limited cross-regional generalization capabilities. However, driven by the expanding scale of spatiotemporal data, the widespread deployment of road network sensing devices, and the continuous iteration of AI algorithms, deep learning based traffic speed prediction is rapidly evolving in several key directions: upgrading model architectures, fusing multi-source data, integrating physical mechanisms, achieving lightweight deployment for practical application, and enhancing both trustworthiness and generalization capabilities.

### *1.4. SparseTSF*

SparseTSF is a novel, extremely lightweight model centered on its "cross-period sparse forecasting" technique. This technique decouples the periodicity and trend components within time series data; by downsampling the original series, it shifts the modeling focus to forecasting cross-period trends. While effectively extracting periodic features, this approach simultaneously minimizes model complexity and parameter count. Leveraging this technique, the model achieves competitive, and in some cases, superior, forecasting performance compared to current state-of-the-art models, despite having fewer than 1000 parameters [26], and demonstrates exceptional generalization capabilities. However, the SparseTSF model does possess certain limitations; specifically, beyond basic aggregation, it lacks explicit modeling of temporal dependencies within individual subsequences. Furthermore, it is restricted to downsampling and decomposing data based on a single primary period, making it difficult to effectively handle data characterized by the interweaving of multiple distinct periods. Consequently, it is unable to capture the complex patterns arising from such multi-period interweaving or to explicitly model the spatial dependencies inherent in networks such as traffic road networks.

This study proposes the design of additional modules to enhance SparseTSF's capabilities, thereby constructing a multi-period spatial sparse time series forecasting (MSTSF) model. This model aims to address the aforementioned limitations, striking a balance between performance and parameter size while improving forecasting accuracy in scenarios involving interwoven multi-period data. By balancing efficiency with effectiveness, the model seeks to enhance its utility in application contexts characterized by limited computational resources, scarce sample sizes, or lower data quality. The fundamental principles, advantages, core innovations, and application scenarios of the SparseTSF model are illustrated in Figure 1.

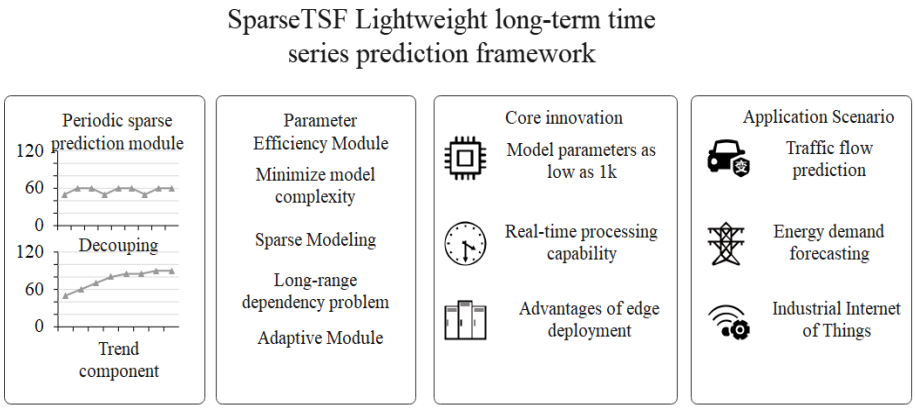


**Figure 1.** Basic principles, advantages, core innovations, and application scenarios of the SparseTSF model.

## 2. Construction of multi-period spatial traffic speed forecast model (MSTSF)

### *2.1. Analysis of the characteristics of traffic speed time series*

The traffic speed time series for road 6 in Guangzhou from August 1 to August 5, 2016 is shown in Figure 2. The data clearly exhibits significant daily periodicity; that is, the daily traffic operation patterns are highly similar.

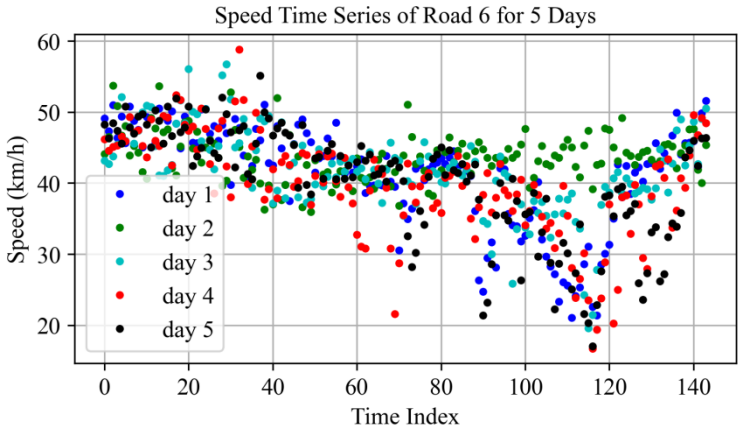


**Figure 2.** Traffic speed time series for road 6 in Guangzhou over 5 days.

Figure 3 shows the traffic speed time series of road 6 on August 1, August 8, August 15, August 22, and August 29, 2016. By comparing the data curves of these five time nodes horizontally, it can be seen that the traffic speed change trends on corresponding dates of different weeks are almost similar. It can be inferred that the traffic speed time series for this road has strong weekly periodicity feature.

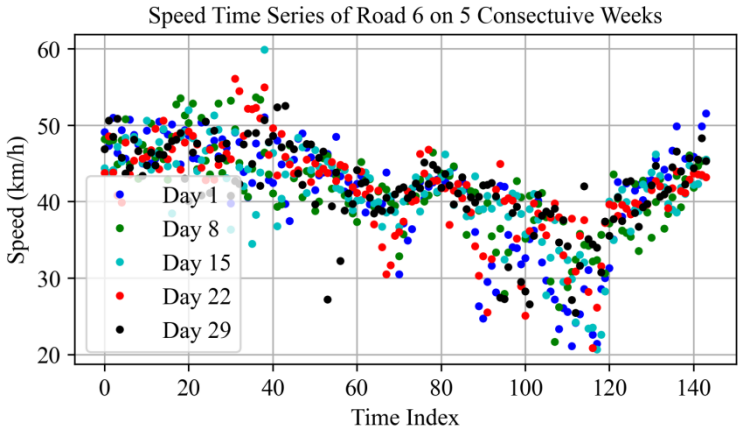


**Figure 3.** Traffic speed time series for road 6 in Guangzhou for 5 consecutive weeks.

Figure 4 shows the traffic speed time series changes of five adjacent roads on August 1, 2016. By analyzing the five curves, we see that in the same time dimension, the speed change trends of the five roads are highly coordinated demonstrating that there is a close and significant spatial correlation between the time series of adjacent sections.

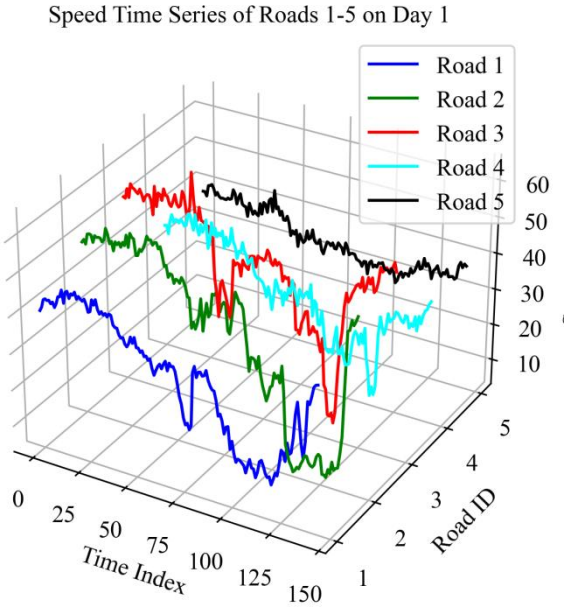


**Figure 4.** Traffic speed time series for five roads on the same day.

## *2.2. Basic ideas of model construction*

This paper selects the SparseTSF model as the basic framework for traffic speed prediction, and conducts in-depth optimization based on the unique attributes of the speed prediction task to improve the adaptability of the model to traffic scenarios. In view of the significant multi-periodic characteristics (such as daily cycles and weekly cycles) in the traffic speed time series, this paper

embeds a weekly period feature extraction module in the model structure to accurately capture the law of periodic fluctuations of traffic flow over time. At the same time, considering the correlation coupling of traffic speed in the spatial dimension, that is, the traffic status of adjacent sections often affecting each other, the spatial feature extraction module is further introduced to enhance the model's ability to describe the spatial interaction of traffic flow by considering the spatial dependency between sections. This paper constructs a multi-period spatially sparse time series prediction (MSTSF) model based on SparseTSF by integrating daily, weekly, and spatial features. Compared with the original model, the MSTSF model can more comprehensively explore the spatiotemporal evolution patterns of traffic speed by collaboratively modeling time period features and spatial correlation features, providing a more efficient and accurate model for traffic speed prediction. Figure 5 is a schematic diagram of the MSTSF model.

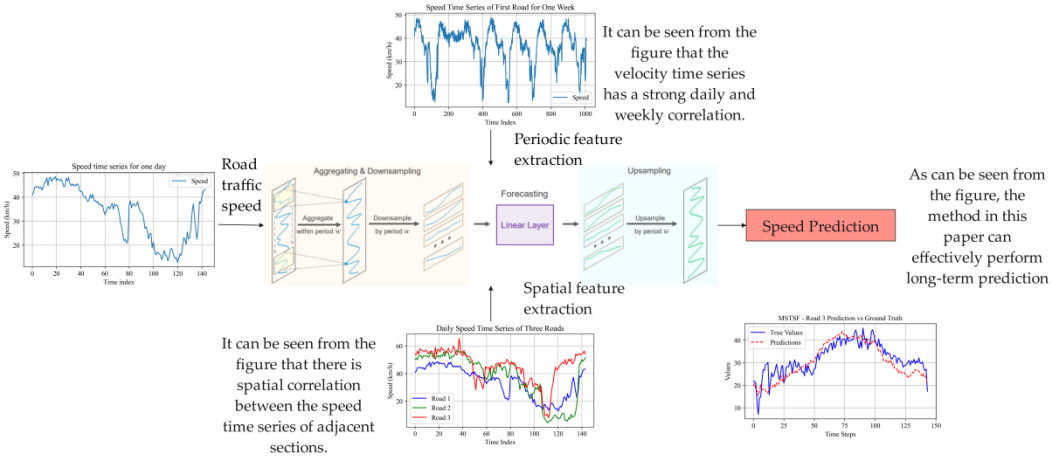


**Figure 5.** Schematic diagram of the MSTSF model.

The key innovations of the method proposed in this paper primarily include the following: (1) Addressing the dual daily and weekly periodicity inherent in traffic speeds, a dedicated weekly periodicity feature extraction module is introduced—integrating Gaussian filtering for denoising, multi-scale one-dimensional convolution, and residual connections—to resolve the issue of insufficient periodic feature representation in the original model. (2) By incorporating inverse distance weighting, dilated convolutions, and a cross-sectional feature interaction mechanism, the method overcomes the limitations of traditional models in capturing long-range and asymmetric spatial influences, thereby accurately characterizing the spatial coupling relationships between road segments. (3) The MSTSF model is built upon SparseTSF and constructed by integrating dual-periodicity and spatial features; by leveraging the collaborative modeling of spatiotemporal features and a residual enhancement mechanism, the model can significantly improve the prediction accuracy and generalization capability for multi-periodic traffic speeds.

### *2.3. Weekly feature extraction*

To effectively capture the implicit weekly periodicity of traffic speed and solve the problem of

insufficient feature representation in the original simple convolution operation, this study innovatively optimizes the weekly feature extraction module based on the SparseTSF model. The improved module integrates historical data denoising, multi-scale one-dimensional convolution, batch normalization, and residual connection mechanisms. It takes the historical traffic speed time series as input, sequentially completes data preprocessing, multi-scale periodic feature extraction, feature fusion, and feature enhancement, and finally generates deep weekly periodic features.

#### 2.3.1. Historical weekly data extraction and denoising

First, extract the traffic speed data for the target road section 7 days prior (i.e., the same time window in the previous week) from the input traffic speed time series, denoted as $X_{week} \in \mathrm{R}^{B \times L \times R}$. There, $B$ represents the batch size during model training, $L$ represents the length of the input time series (number of time points), and $R$ represents the input feature dimension (here, $R$=1, which is the traffic speed dimension; if multiple traffic parameters are fused, $R$>1).

Due to the interference of external factors (such as traffic accidents, weather changes, and equipment errors), the extracted weekly historical data contains random noise, which will affect the accuracy of periodic feature extraction. Therefore, a Gaussian filtering denoising operation is introduced to smooth the data. The specific formula is:

$$X_{week\_denoise}(b, l, r) = \sum_{k=-K}^{K} g(k) \cdot X_{week}(b, l+k, r). \tag{1}$$

In the formula, $g(k)$ is the Gaussian filter weight, which is calculated as:

$$g(\mathrm{k}) = \frac{1}{\sqrt{2\pi}\sigma_g} \exp\left(-\frac{k^2}{2\sigma_g^2}\right) \tag{2}$$

where $b \in [1, B]$, $l \in [1, L]$, $r \in [1, R]$ are the batch index, time index, and feature index, respectively. $X_{week\_denoise}$ is the vector after noise removal. For the edge time points ($l \leq K$ or $l \geq L-K$), the zero-padding method is adopted to avoid data loss.

#### 2.3.2. Multi-scale one-dimensional convolution for periodic feature extraction

To fully capture the multi-scale weekly periodic patterns (such as the periodicity of peak hours and off-peak hours in a week), the original single-scale convolution is replaced with a multi-scale one-dimensional convolution structure, which includes three parallel Conv1D layers with different convolution kernel sizes.

First, perform channel expansion on the denoised weekly data to match the input channel number of the convolution layer. The channel expansion formula is:

$$X_{week\_expand} = X_{week\_denoise} \otimes 1_{1 \times 1 \times 3} \tag{3}$$

where $X_{week\_expand} \in \mathrm{R}^{B \times L \times 3}$ is the weekly data after channel expansion; $\otimes$ is the tensor outer product; $1_{1 \times 1 \times 3}$ is a 3-dimensional all-1 tensor with a size of $1 \times 1 \times 3$.

Then, input the expanded data into three parallel Conv1D layers for convolution operation, and the calculation formulas of each layer are:

$$F_{week\_conv1}=\text{Conv1D}(X_{week\_expand};W_1,b_1)=\sum_{k=0}^{C_1\text{-}1} W_1(k,:,:)\cdot X_{week\_expand}(:,:C_1,:)+b_1 \tag{4}$$

$$F_{week\_conv2}=\text{Conv1D}(X_{week\_expand};W_2,b_2)=\sum_{k=0}^{C_2\text{-}1} W_2(k,:,:)\cdot X_{week\_expand}(:,:C_2,:)+b_2 \tag{5}$$

$$F_{week\_conv3}=\text{Conv1D}(X_{week\_expand};W_3,b_3)=\sum_{k=0}^{C_3\text{-}1} W_3(k,:,:)\cdot X_{week\_expand}(:,:C_3,:)+b_3 \tag{6}$$

where $F_{week\_conv1},F_{week\_conv2},F_{week\_conv3}$ are the calculated results of three convolutional channels. To avoid overfitting and accelerate model convergence, batch normalization (BN) is performed on the output features of each convolution layer. The BN operation formula is:

$$\widehat{F}_{week\_convi}=\frac{F_{week\_convi}-\mu_{F_i}}{\sqrt{\sigma_{F_i}^2+\epsilon}}\cdot\gamma_i+\beta_i(i=1,2,3). \tag{7}$$

$\widehat{F}_{week\_conv1},\widehat{F}_{week\_conv2},\widehat{F}_{week\_conv3}\in R^{B\times L\times 3}$ are the features after batch normalization.

Next, introduce the ReLU activation function to add non-linearity to the features and enhance the model's ability to fit complex periodic patterns. The formula is:

$$F_{week\_acti}=max(\widehat{F}_{week\_conv1}+\widehat{F}_{week\_conv2}+\widehat{F}_{week\_conv3},0), \tag{8}$$

where $F_{week\_acti}\in R^{B\times L\times 3}$ is the activated multi-scale weekly periodic feature and $\max(.,0)$ is the ReLU activation function, which sets the negative feature values to 0 and retains the positive feature values.

### 2.3.3. Multi-periodic feature fusion and residual enhancement

The daily periodic features $F_{day}\in R^{B\times L\times 3}$ are extracted from the original SparseTSF model (the extraction method is consistent with the weekly feature extraction, but the input is the daily traffic speed data of the target road section). To realize the effective fusion of weekly and daily periodic features, a weighted fusion mechanism is adopted instead of simple element-wise addition. The specific formula is:

$$F_{week\_fusion}=\alpha\cdot F_{week\_acti}+(1\text{-}\alpha)\cdot F_{day} \tag{9}$$

where $F_{week\_acti}\in R^{B\times L\times 3}$ is the activated multi-scale weekly periodic feature, and the daily periodic features $F_{day}\in R^{B\times L\times 3}$ are extracted from the original SparseTSF model. To further enhance the feature expression ability and avoid gradient disappearance, a residual connection is introduced to fuse the initially fused feature with the expanded weekly data. The specific formula is:

$$F_{week}=F_{week\_fusion}+X_{week\_expand}+\lambda\cdot\text{Dropout}(F_{week\_fusion}). \tag{10}$$

Dropout(·) is the dropout regularization operation, and the dropout rate is set to 0.2 to avoid overfitting; the dropout operation formula is:

$$\text{Dropout}(F)=F\cdot mask, \tag{11}$$

$$mask\in\{0,1\}^{B\times L\times 3}, \tag{12}$$

$$P(mask=1)=0.8. \tag{13}$$

### *2.4. Spatial feature extraction*

Considering the significant spatial correlation between adjacent road sections and the problem that the original module cannot capture the long-distance spatial interaction and asymmetric spatial influence, this study optimizes the spatial feature extraction module by integrating spatial weight assignment, dilated convolution, cross-section feature interaction, and feature normalization. The improved module takes the traffic speed data of the target road section and its adjacent sections (including upstream, downstream, and adjacent parallel sections) as input, sequentially completes spatial neighbor selection, weighted stacking, dilated convolution feature extraction, cross-section interaction, and feature fusion, and finally generates deep spatial features.

#### 2.4.1. Spatial neighbor selection and weighted data extraction

Due to the different distances between adjacent sections and the target section, their influence on the traffic speed of the target section is asymmetric. Therefore, a spatial weight calculation method based on the inverse distance is introduced to assign different weights to each adjacent section. The spatial weight formula is:

$$w_j=\frac{1}{d_j+\epsilon}/\sum_{k\in\{i-2,i-1,i+1,i+2\}}\frac{1}{d_k+\epsilon}\ (j\in\{i-2,\ i-1,\ i+1,\ i+2\}). \tag{14}$$

The target section itself is assigned the highest weight $w_i=0.5$, and the sum of the weights of the four adjacent sections is

$$0.5\ \text{(i.e., }\sum_{j\in\{i-2,i-1,i+1,i+2\}} w_j=0.5).$$

Then, the weighted traffic speed data of each section is calculated as:

$$X_j^{weight}=w_j\cdot X_j\ (j\in\{i-2,\ i-1,\ i\ ,\ i+1,\ i+2\}), \tag{15}$$

where $X_j^{weight}$ is the linearly weighted variable.

#### 2.4.2. Weighted stacking and spatial feature expansion

Vertically stack the weighted traffic speed data of the target section and its four adjacent sections along the feature dimension (axis=2) to form a preliminary spatial feature matrix. The stacking formula is:

$$X_{spatial\_stack}=\text{concat}(X_{i-2}^{weight},X_{i-1}^{weight},X_i^{weight},X_{i+1}^{weight},X_{i+2}^{weight},\text{axis}=2), \tag{16}$$

where $X_j^{weight}$ represents the linearly weighted variable, and $X_{spatial\_stack}$ represents the spatially stacked variable.

To enhance the spatial feature representation ability, perform feature expansion on the stacked matrix through a 1×1 convolution kernel (point-wise convolution), which can adjust the feature

dimension without changing the time and batch dimensions. The feature expansion formula is:

$$X_{spatial\_expand}=\text{Conv1D}(X_{spatial\_stack},W_{pw},b_{pw},\text{kernel\_size}=1), \tag{17}$$

where $X_{spatial\_expand}$ represents the variable after spatial expansion.

### 2.4.3. Dilated convolution for spatial correlation extraction

To capture the long-distance spatial correlation between non-adjacent sections (such as the correlation between the upstream parallel section i-2 and the downstream parallel section i+2), the original standard one-dimensional convolution is replaced with a dilated one-dimensional convolution (Dilated Conv1D). The dilated convolution can expand the receptive field without increasing the number of parameters, thereby capturing more distant spatial interaction information.

The calculation formulas of the two dilated convolution layers are:

$$F_{spatial\_dil1}=\sum_{k=0}^{2} W_4(k,:,:)\cdot X_{spatial\_expand}(:,:,k\times d_1:k\times d_1+2R)+b_4, \tag{18}$$

$$F_{spatial\_dil2}=\sum_{k=0}^{2} W_5(k,:,:)\cdot X_{spatial\_expand}(:,:,k\times d_2:k\times d_2+2R)+b_5. \tag{19}$$

Fuse the output features of the two dilated convolution layers through element-wise multiplication to enhance the complementarity of multi-receptive field spatial features, and then perform layer normalization (LN) to stabilize the feature distribution. The formula is:

$$F_{spatial\_conv}=\text{LN}(F_{spatial\_dil1}\odot F_{spatial\_dil2}), \tag{20}$$

where $F_{spatial\_dil1},F_{spatial\_dil2}$ represent the outputs of the first and second channels, respectively, and $\odot$ denotes the element-wise product operation. $\text{LN}(\cdot)$ is the layer normalization operation, and the formula is:

$$\text{LN}(F)=\frac{F-\mu_L}{\sqrt{\sigma_L^2+\epsilon}}\cdot\gamma_L+\beta_L. \tag{21}$$

### 2.4.4. Cross-section feature interaction and final spatial feature fusion

Introduce a cross-section feature interaction mechanism to capture the mutual influence between the target section and each adjacent section. The specific method is to calculate the correlation coefficient between the target section's processed feature $X_{processed}\in\mathrm{R}^{B\times L\times R}$ (after data preprocessing, including normalization and denoising) and the extracted spatial correlation feature $F_{spatial\_conv}$, and use the correlation coefficient as the interaction weight. The correlation coefficient calculation formula is:

$$\rho=\frac{\text{Cov}(X_{processed},F_{spatial\_conv})}{\sqrt{\text{Var}(X_{processed})\cdot\text{Var}(F_{spatial\_conv})+\epsilon}}. \tag{22}$$

Normalize the correlation coefficient to obtain the interaction weight $\omega=\frac{\rho+1}{2}$ (ensuring $\omega\in[0,1]$), then perform weighted fusion of the target section feature and the spatial correlation

feature. The final spatial feature formula is:

$$F_{spatial}=\omega\cdot X_{processed}+(1-\omega)\cdot F_{spatial\_conv}+\delta\cdot \mathrm{L2Norm}(F_{spatial\_conv}). \tag{23}$$

### *2.5. Overall model process*

The model architecture consists of three core sub-modules: a weekly feature extraction module, a spatial feature extraction module, and a prediction module. The weekly feature extraction module, based on historical data mining mechanisms, extracts the periodic features of the target road seven days prior. The spatial feature extraction module focuses on topological correlation and model building, and analyzes the multidimensional features of adjacent road segments. These two feature extraction modules work together to extract deep features of the time series, laying a solid data foundation for subsequent long-term time series prediction tasks. The prediction module, as the engine of the model, generates high-precision time series prediction results based on the extracted spatiotemporal features.

In the visualization presented in Figure 6, the red modules correspond to the weekly feature extraction modules, the orange modules represent the spatial feature extraction modules, and the green modules denote the prediction model. The flowchart in Figure 6 intuitively and logically illustrates the complete operational mechanism of the model from data input and feature extraction through to result output.

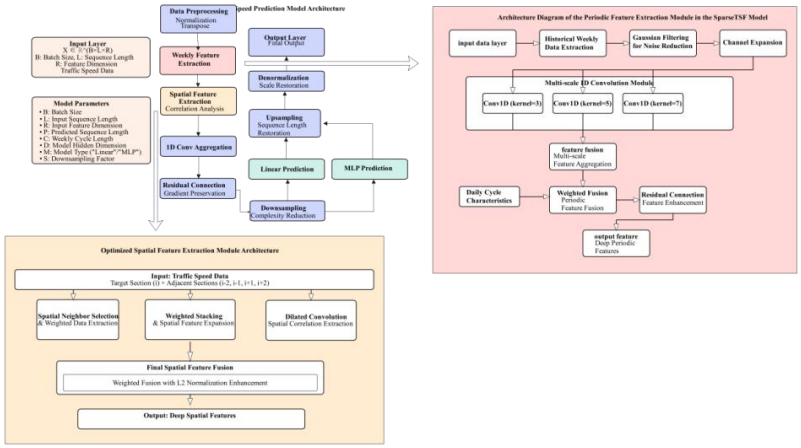


**Figure 6.** Flowchart of the MSTSF model calculation process.

### *2.6. Model calculation steps*

The calculation process of the MSTSF model includes 11 steps, covering parameter initialization, data preprocessing, feature extraction, feature fusion, prediction, and result denormalization. The specific steps and key formulas are as follows:

#### 2.6.1. Input and parameter definition

Input: $X\in \mathrm{R}^{B\times L\times R}$, where $B$=batch size, $L$=input sequence length, $R$=input feature dimension

(traffic speed).

Parameters: $P$=predicted sequence length, $C$=weekly cycle length, $D$=model hidden layer dimension, $M$=model type ("Linear" or "MLP").

Output: $y \in \mathrm{R}^{B \times P \times R}$, the traffic speed prediction result.

2.6.2. Detailed calculation steps

Step 1. Parameter initialization. Initialize all model parameters, including convolution kernel weights $W_1$,$W_2$, bias terms $b_1$,$b_2$, and hidden layer parameters of the prediction module. The initialization method adopts the Xavier uniform initialization:

$$W \sim \text{Uniform}\left(-\frac{\sqrt{6}}{\sqrt{n_{in}+n_{out}}}, \frac{\sqrt{6}}{\sqrt{n_{in}+n_{out}}}\right), \tag{24}$$

where $n_{in}$ is the input dimension of the parameter, and $n_{out}$ is the output dimension.

Step 2. Data preprocessing. Calculate the mean value of the input sequence to realize data normalization:

$$\mu = \frac{1}{B \times L} \sum_{b=1}^{B} \sum_{l=1}^{L} X_{b,l,:} \tag{25}$$

$$X_{norm} = \frac{X-\mu}{\sigma+\epsilon} \tag{26}$$

where $\mu$ is the mean value of the input sequence, $\sigma$ is the standard deviation, and $\epsilon = 10^{-8}$ is the smoothing term to avoid division by zero. $B$ is the batch size, and $L$ is the extent of the input historical data. Adjust the dimension of the normalized data by swapping the second and third dimensions:

$$X_{norm\_trans} = \text{transpose}(X_{norm}, [0,2,1]) \tag{27}$$

where $X_{norm\_trans} \in \mathrm{R}^{B \times R \times L}$.

Step 3. Weekly feature extraction. Extract weekly periodic features according to the method described in Section 2.3, and obtain the fused weekly feature $F_{week}$.

Step 4. Spatial feature extraction. Extract spatial correlation features according to the method described in Section 2.4, and obtain the fused spatial feature $F_{spatial}$.

Step 5. 1D convolution aggregation. Perform one-dimensional convolution aggregation on the fused spatiotemporal features to enhance feature representation:

$$F_{agg} = \text{Conv1D}(\text{concat}(F_{week}, F_{spatial}, \text{axis}=2), \mathrm{W}_3, \mathrm{b}_3), \tag{28}$$

where $W_3 \in \mathrm{R}^{3 \times (3+R) \times D}$ is the aggregation convolution kernel weight, $b_3 \in \mathrm{R}^{\mathrm{D}}$ is the bias term, and $F_{agg} \in \mathrm{R}^{B \times L \times D}$ is the aggregated feature.

Step 6. Feature residual connection. Add the aggregated feature to the original normalized data to realize residual connection and avoid gradient disappearance:

$$F_{res} = F_{agg} + X_{norm}, \tag{29}$$

where $F_{res} \in \mathrm{R}^{B \times L \times D}$ is the feature after residual connection.

Step 7. Downsampling. Reshape the residual feature to reduce the sequence length and computational complexity:

$$F_{down} = \text{reshape}(F_{res}, [B, D, L/S]), \tag{30}$$

where $S$ is the downsampling factor. $D$ represents the dimension or number of channels of the intermediate variable. Adjust the dimension by swapping the second and third dimensions:

$$F_{down\_trans}=\text{transpose}(F_{down},[0,2,1]), \tag{31}$$

where $F_{down\_trans}\in\mathrm{R}^{B\times L/S\times D}$.

Step 8. Sparse prediction: According to the model type $M$, perform sparse prediction:

If $M$="Linear", use a linear layer for prediction:

$$y_{down}=F_{down\_trans}\cdot W_{linear}+b_{linear}, \tag{32}$$

where $W_{linear}\in\mathrm{R}^{D\times P}$ is the linear layer weight, $b_{linear}\in\mathrm{R}^{P}$ is the bias term, and $y_{down}\in\mathrm{R}^{B\times L/S\times P}$.

If M="mlp", use a multi-layer perceptron (MLP) for prediction:

$$y_{down}=\text{MLP}(F_{down\_trans}, W_{mlp1}, b_{mlp1}, W_{mlp2}, b_{mlp2}), \tag{33}$$

where

$$W_{mlp1}\in\mathrm{R}^{D\times 2D},\ b_{mlp1}\in\mathrm{R}^{2D},\ W_{mlp2}\in\mathrm{R}^{2D\times P},\ b_{mlp2}\in\mathrm{R}^{P}$$

are the weights and bias terms of the MLP, and $y_{down}\in\mathrm{R}^{B\times L/S\times P}$.

Step 9. Upsampling. Reshape the downsampled prediction result to restore the target sequence length:

$$y_{up}=\text{reshape}(y_{down},[B,P,R]), \tag{34}$$

where $y_{up}\in\mathrm{R}^{B\times P\times R}$ is the upsampled prediction result, $P$ is the number of prediction steps, and $R$ is the number of roads.

Step 10. Denormalization. Restore the prediction result to the original speed scale by restoring the mean value:

$$y=y_{up}\times(\sigma+\epsilon)+\mu, \tag{35}$$

where $y\in\mathrm{R}^{B\times P\times R}$ is the final prediction result.

Step 11. Result output: Return the traffic speed prediction result y.

### *2.7. Loss function and training algorithm*

In line with the current mainstream practice in this field, this paper adopts the classic mean square error (MSE) as the loss function of SparseTSF. This function measures the difference between the predicted and true value, and its formula is as follows:

$$\mathrm{L}(\theta)=\frac{1}{B\times P\times R}\sum_{b=1}^{B}\sum_{p=1}^{P}\sum_{r=1}^{R}(y_{b,p,r}-\hat{y}_{b,p,r})^2 \tag{36}$$

where B denotes the batch size, P denotes the number of prediction steps, and R denotes the number of roads. For model training, this paper uses the Adam optimization algorithm. Due to space limitations, this paper will not go into details one by one and readers are refered to the literature [27].

### *2.8. Generalization ability analysis*

The MSTSF model inherits the lightweight advantage of the original SparseTSF model, and the number of parameters is effectively controlled, which reduces the risk of overfitting and improves generalization ability. The parameter complexity analysis of the core modules is as follows:

(1) Weekly feature extraction module. The core is the one-dimensional convolution layer, and its parameter complexity is $O(R \times D)$, where $R$ is the input feature dimension, and $D$ is the hidden layer dimension.

(2) Spatial feature extraction module. The parameter complexity is mainly concentrated in the spatial convolution layer, which is $O(R^2)$.

According to Occam's razor principle, among all models that can explain the data, the model with fewer parameters has better generalization ability. In addition, the MSTSF model enhances the ability to capture traffic speed change patterns by integrating weekly and spatial features, and its generalization ability is further improved compared with the original SparseTSF model. The specific quantitative verification is shown in Section 3.7.

### *2.9. Advantages and disadvantages of this method*

Compared to the sparseTSF model, this method demonstrates significant advantages in both model architecture and feature extraction. It breaks through the limitations of traditional models, which typically rely on single-period modeling, by innovatively introducing a dual-period feature fusion module. By combining multi-scale convolutions with residual connections, the model is able to precisely capture fine-grained intraday temporal patterns while effectively identifying macro-level weekly trends, thereby substantially enhancing its feature representation capabilities.

However, given that this model integrates various modules, such as multi-scale convolution and feature weighting, its hyperparameter configuration and tuning process are relatively complex. This places high demands on the experience level of engineering personnel.

## 3. Numerical experiments

This section will show in detail the experimental results of the MSTSF model in the multi-step traffic speed test. Through multiple sets of comparative experiments and data analysis, the performance of the model in complex traffic scenarios is intuitively presented. At the same time, the efficiency advantage of the lightweight architecture in improving traffic speed is deeply explored, and theoretical analysis and empirical research are conducted from the dimensions of algorithm complexity and resource occupation to reveal its efficiency and feasibility in practical applications. The contribution of the multi-period feature extraction module and the spatial feature extraction module to the improvement of model performance is quantified to further verify the scientificity and effectiveness of the model design, providing an important reference for subsequent research in the field of traffic prediction.

### *3.1. Dataset*

The traffic speed dataset used in this paper is derived from a field collection of 214 anonymous

road sections in Guangzhou, China [28]. These sections mainly cover urban expressways and main roads. The data collection period is from August 1 to September 30, 2016, and the collection frequency is set to once every 10 minutes, ensuring that the data records the high-frequency dynamic conditions of road traffic. In addition, the study also introduces California expressway data (PeMS) as a supplementary data source [29]. This dataset monitors key traffic indicators such as flow, speed, and occupancy rate of 170 roads at a time interval of 5 minutes, and conducts a 1-day-ahead traffic speed forecast based on this. The collection time is from January 1, 2019, to February 28, 2019, and its rich time series data provides solid data support for traffic flow analysis. The basic characteristics of both datasets are shown in Table 1.

**Table 1.** Characteristics of Guangzhou traffic dataset and PeMS dataset.

| Dataset | Guangzhou traffic dataset | PeMS dataset |
|---|---|---|
| Number of road sections | 214 | 170 |
| Sampling frequency | Every 10 minutes | Every 5 minutes |
| Collection time | August 1, to September 30, 2016 | January 1, to February 28, 2019 |

### *3.2. Prediction effect*

Figures 7 and 8 present the traffic speed prediction results of the MSTSF model for Roads 1 and 2 on the Guangzhou traffic dataset. In the figure, the blue curve accurately depicts the dynamic changes of the actual traffic speed, and the red curve shows the model prediction value. From the visualization results, in the time series dimension, the red prediction curve is highly consistent with the blue actual curve and remains closely aligned even in complex traffic situations. This result fully demonstrates that the model proposed in this paper can effectively capture the changing laws of traffic speed in long-term prediction scenarios, with excellent prediction performance and reliability.

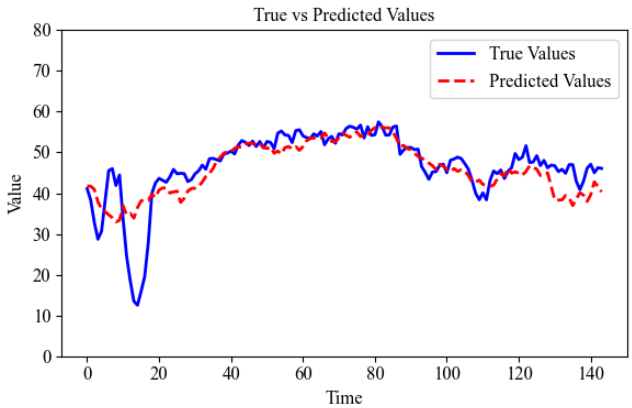


**Figure 7.** Prediction effect of MSTSF model on Road 1 (the blue curve represents actual values, red curve represents predicted values).

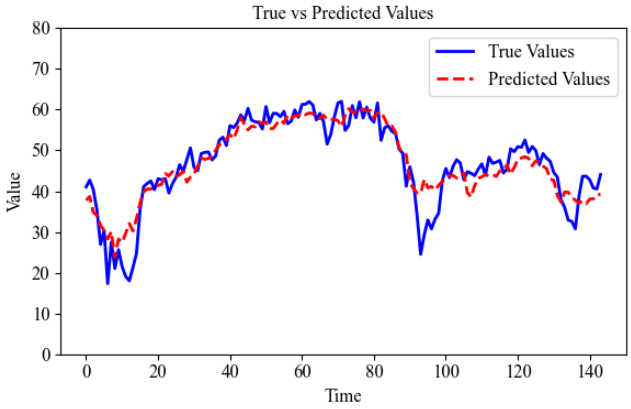


**Figure 8.** Prediction effect of MSTSF model on Road 2 (the blue curve represents the actual values and the red curve represents the predicted values).

### *3.3. Prediction accuracy*

To systematically verify the effectiveness and reliability of the MSTSF model, this study conducted empirical analysis using the Guangzhou traffic dataset as the first evaluation target. This dataset contains multi-dimensional traffic flow data, which can fully test the model's prediction performance in complex spatiotemporal scenarios.

When constructing the training and test sets based on Guangzhou traffic speed data, we use the first eight days of data as the input and the last day of data as output. We then selected data before September 18, 2016, as the training set, and data after September 18, 2016, as the test set. The length of the historical input data for the experiment is 144×7, the length of the predicted output is 144×1, the daily cycle is set to 144, the weekly cycle is set to 144×7, the hidden layer dimension of the MLP layer is set to 512, the convolution kernel sizes of the weekly cycle feature extraction module are set to [3, 5, 7], the Gaussian denoising kernel size is set to 3, and the weight fusion coefficients are set to $\alpha$=0.5, $\lambda$=0.1. The convolution kernel size of the spatial feature extraction module is set to 3, the expansion coefficient is set to [2, 4], and the padding coefficient is set to [2, 4]. For the PeMS dataset, we also use the first eight days of data as input and data after the last day as output. We then select data before February 18, 2019, as the training set, and data after February 18, 2019, as the test set. For the PeMS experiment, the length of the input historical data is 288×7, and the length of the predicted output is 288×1; the daily cycle is set to 288, the weekly cycle is set to 288×7, and the remaining parameters are set similarly to the previous experiment with Guangzhou traffic data. We also randomly select 20% of the data from the training set as the validation set.

After rigorous experimental design and multiple rounds of validation, Table 2 and Figure 9 detail the prediction accuracy evaluation results of the MSTSF model and other models on the Guangzhou traffic dataset and the PeMS dataset. The main evaluation metrics include RMSE, MAE, and mean absolute percentage error (MAPE). The specific values are given Table 2. As can be seen from the table, compared to the large-scale time series prediction model TimesNet, the prediction error is reduced by 33.25%. Furthermore, the MSTSF model achieves optimal results on both datasets.

**Table 2.** Comparison of models on two datasets.

| Prediction step length<br>Dataset | 144 (1 day)<br>Guangzhou traffic dataset | | | 288 (1 day)<br>PeMS dataset | | |
|---|---|---|---|---|---|---|
| | MAE | RMSE | MAPE | MAE | RMSE | MAPE |
| MSTSF (ours) | 2.75 | 4.18 | 9.64% | 2.61 | 6.06 | 8.62% |
| KAN [30] | 3.66 | 5.10 | 13.57% | 3.41 | 6.43 | 7.22% |
| DeepAR [31] | 6.64 | 8.59 | 24.83% | - | - | - |
| DLinear | 3.51 | 4.94 | 12.73% | 3.49 | 6.87 | 7.49% |
| FEDformer [32] | 6.33 | 7.93 | 22.12% | 4.33 | 7.85 | 9.08% |
| Informer [33] | 5.99 | 7.65 | 20.85% | 4.02 | 7.56 | 8.49% |
| MLP | 3.47 | 4.84 | 12.74% | 3.48 | 6.57 | 7.38% |
| NBEATS [34] | 4.79 | 6.32 | 17.23% | 6.63 | 9.81 | 13.17% |
| NHITS [35] | 4.10 | 5.71 | 15.60% | 4.00 | 7.42 | 8.44% |
| TCN [8] | 6.14 | 7.81 | 20.87% | 4.04 | 7.43 | 8.45% |
| TiDE [36] | 3.37 | 4.80 | 12.41% | 3.38 | 6.48 | 7.27% |
| TimesNet [15] | 4.12 | 5.49 | 14.45% | 3.54 | 6.89 | 7.56% |
| PatchTST [37] | 3.00 | 4.59 | 12.11% | 3.65 | 7.16 | 7.83% |
| Autoformer [38] | 5.64 | 7.80 | 20.93% | 4.46 | 8.06 | 9.36% |
| NLinear | 3.15 | 4.61 | 12.18% | 3.49 | 6.62 | 7.41% |
| BiTCN | 3.29 | 4.97 | 13.20% | 3.66 | 7.21 | 7.86% |
| DeepNPTS [39] | 3.27 | 5.26 | 13.48% | 3.91 | 7.53 | 8.34% |
| DilatedRNN [40] | 4.11 | 6.02 | 16.38% | - | - | - |

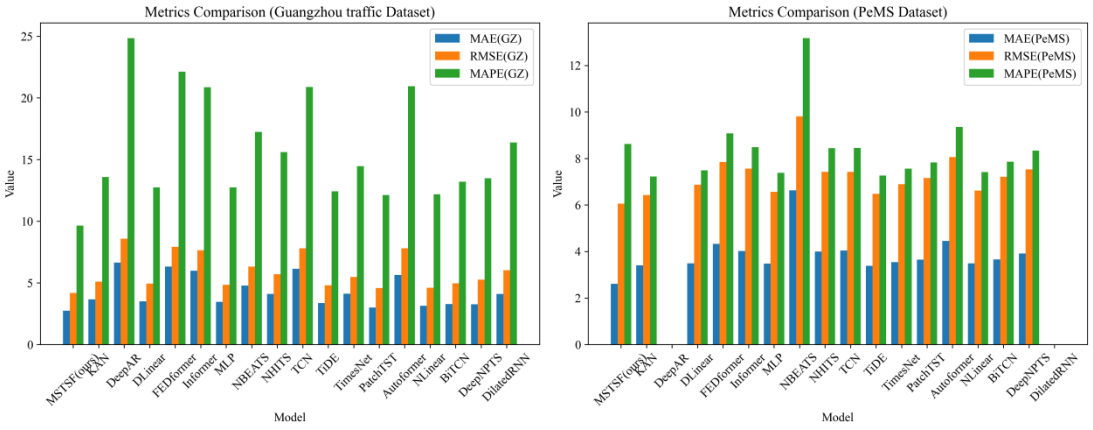


**Figure 9.** Model comparison on two datasets.

Based on the Guangzhou traffic speed dataset, this paper systematically compares the mean absolute error (MAE) performance of the MSTSF model and the DLinear model under different input lengths (144, 432, 1008, 1152) and prediction lengths (144, 288, 432) for multi-source spatiotemporal sequence prediction tasks. This dataset has typical spatiotemporal coupling characteristics.

Table 3 shows that in all 12 groups of comparative experiments, the MSTSF model, with its unique spatiotemporal feature fusion mechanism, achieves lower MAE values. In all 12 groups of scenarios in Table 3, there is an average error reduction of 10% compared with the DLinear model (Results are calculated based on the data in Table 3), which effectively verifies the superiority and generalization ability of the model in complex traffic flow prediction scenarios.

**Table 3.** Comparison of the MSTSF model and the DLinear model on the Guangzhou traffic dataset under different input lengths and prediction lengths.

| | MSTSF prediction length | | | DLinear prediction length | | |
|---|---|---|---|---|---|---|
| Input length | 144 | 288 | 432 | 144 | 288 | 432 |
| 144 | 3.66 | 4.15 | 4.34 | 3.92 | 4.72 | 4.83 |
| 432 | 4.14 | 4.28 | 4.33 | 4.24 | 4.52 | 4.69 |
| 1008 | 2.77 | 3.00 | 3.14 | 3.25 | 3.56 | 3.69 |
| 1152 | 2.83 | 3.18 | 3.49 | 3.24 | 3.58 | 3.79 |

### *3.4. Model efficiency*

In addition to its excellent prediction accuracy, the lightweight advantage of the MSTSF model is particularly prominent. In order to further explore its lightweight characteristics, this paper selects basic models such as DLinear as a reference and conducts systematic comparative analysis. First, model size, that is, the total space occupied by trainable parameters in the model, directly reflects the complexity and storage requirements of the model; second, iteration efficiency, through the time required for one training iteration, quantifies the training resource consumption of the model; third, inference performance uses the single inference time as a metric to evaluate the response speed of the model in actual deployment. The above comparison dimensions comprehensively cover the resource occupancy and operation efficiency of the entire life cycle of the model and provide a scientific basis for objectively evaluating the lightweight advantages of the MSTSF model. The static and runtime indicators of MSTSF and other mainstream models on the Guangzhou traffic dataset are compared with the prediction period of 144 (1 day). The results are shown in Table 4.

Figure 10 shows the performance comparison of different models. The X-axis shows the time required for each model to iterate once in a logarithmic coordinate system, and the Y-axis shows the mean absolute error (MAE). Through in-depth analysis of the chart data, the MSTSF model shows significant advantages: in terms of iteration efficiency, the time required for one iteration is significantly lower than that of other comparison models; and in terms of prediction accuracy, its MAE value performs best among all models, which means that the average deviation between its prediction results and the true value is the smallest. This high efficiency and low error feature fully demonstrates that the model proposed in this paper has stronger practicality and reliability in actual application scenarios.

**Table 4.** Space complexity and time complexity of the model.

| Model | Number of parameters | Iteration time | Inference time |
|---|---|---|---|
| MSTSF (ours) | 13.9 K | 3.94 s | 0.51 s |
| KAN | 3.7 M | 10 s | 5 s |
| DeepAR | 67.5 K | 280 s | 11 s |
| DLinear | 124 K | 1 s | 0.05 s |
| FEDformer | 214 K | 20 s | 20 s |
| Informer | 8.8 K | 68 s | 1 s |
| MLP | 1.8 M | 2 s | 4 s |
| NBEATS | 3.8 M | 4 s | 23 s |
| NHITS | 4 M | 4 s | 11 s |
| TCN | 211 K | 12 s | 6 s |
| TiDE | 7 M | 9 s | 16 s |
| TimesNet | 835 K | 425 s | 34 s |
| PatchTST | 40.6 K | 16 s | 1.06 s |
| Autoformer | 1.1 K | 136 s | 3.62 s |
| NLinear | 60.24 K | 3.8 s | 0.42 s |
| BiTCN | 186 K | 494 s | 5.64 s |
| DeepNPTS | 4.1 M | 34 s | 1.4 s |
| DilatedRNN | 17.4 K | 228 s | 123 s |

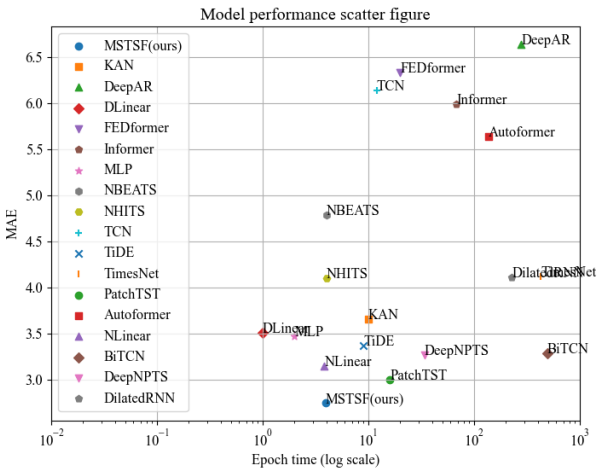


**Figure 10.** Comparison of model performance (mean absolute error and time per iteration).

### *3.5 Model performance indicators*

As shown in Figure 11, the radar chart visually presents the comprehensive performance and ranking of different models on multiple core performance indicators. The figure covers important evaluation dimensions, such as mean absolute percentage error (MAPE), mean absolute error (MAE), root mean square error (RMSE), model space occupancy, single round iteration time, and inference

time. The light blue outline in the radar chart clearly marks the performance of the method proposed in this paper in each indicator dimension, which is convenient for quickly capturing its advantages in the model performance comparison. It can be seen that the model proposed in this paper achieves relatively good results.

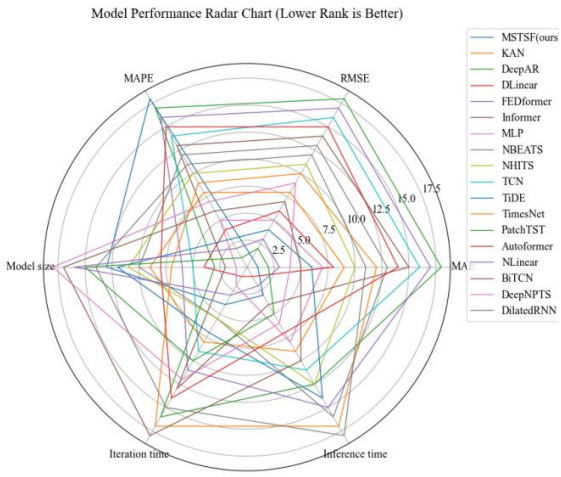


**Figure 11.** Comparison of model performance (radar chart of rankings of six indicators, based on the Guangzhou road dataset).

### *3.6. Result visualization*

In order to further verify the prediction performance of the MSTSF model, this paper intuitively presents some prediction results through visualization. As shown in Figure 12, the research team selected 10 mainstream models and conducted a visual comparative analysis on the Guangzhou traffic speed data set. The chart uses the time step as the X-axis and the corresponding value as the Y-axis. The blue line represents the actual data, and the red line is the model prediction result. Due to space constraints, this paper only presents the prediction results of the top 10 models. By observing Figure 12, it can be found that in both short-term and long-term prediction scenarios, the MSTSF model can accurately capture the periodicity and trend characteristics in the time series data. Compared with other models, the error between its prediction results and the real data is smaller, showing better prediction accuracy.

### *3.7. Ablation experiments*

To verify the performance improvements brought about by this model structure, we conducted ablation experiments. We compared the MAE, RMSE, and MAPE metrics of the SparseTSF model, the model incorporating spatial features, the model incorporating periodic features, and the MSTSF model on the Guangzhou traffic speed dataset and the PeMS dataset. Qualitative schematics of the comparative and ablation experiments presented in this paper are shown in Figure 13.


**Figure 12.** Visualization of prediction results of 10 models.

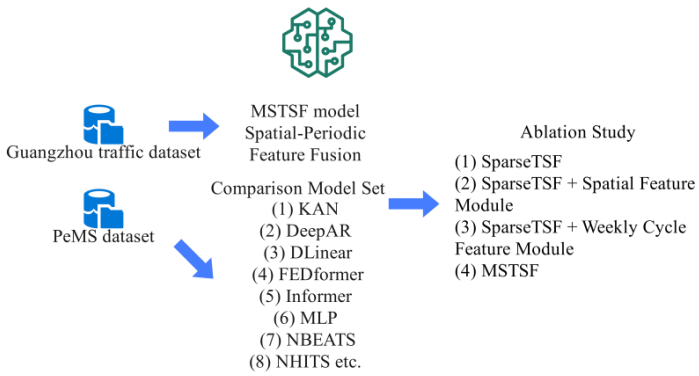


**Figure 13.** Qualitative illustrations of the comparative and ablation experiments presented in this paper.

Table 5 shows the results of the ablation experiments. As can be seen, the integration of spatial and periodic features significantly improves the model's prediction accuracy.

**Table 5.** Ablation experiment results.

| | Guangzhou traffic speed dataset | | | PeMS dataset | | |
|---|---|---|---|---|---|---|
| Model | MAE | RMSE | MAPE | MAE | RMSE | MAPE |
| SparseTSF | 3.93 | 5.35 | 11.03% | 2.91 | 5.69 | 8.38% |
| Spatial model | 3.12 | 4.61 | 10.44% | 2.55 | 5.75 | 8.20% |
| Weekly model | 3.15 | 4.61 | 10.96% | 2.88 | 5.77 | 8.28% |
| MSTSF | 2.75 | 4.18 | 9.64% | 2.61 | 6.06 | 8.62% |

The MSTSF model proposed in this paper is a lightweight model. This study compares the

predictive performance of the MSTSF model against two comparable lightweight models, STLinear and PastNet, across two datasets. The results are shown in Table 6.

**Table 6**. Comparison of the proposed model with similar lightweight models.

| | Guangzhou traffic speed dataset | | | PeMS dataset | | |
|---|---|---|---|---|---|---|
| Model | MAE | RMSE | MAPE | MAE | RMSE | MAPE |
| STLinear [19] | 4.63 | 6.33 | 14.90% | 3.83 | 6.71 | 9.42% |
| PastNet [22] | 6.18 | 8.21 | 20.68% | 4.09 | 8.19 | 11.99% |
| MSTSF | 2.75 | 4.18 | 9.64% | 2.61 | 6.06 | 8.62% |

## 4. Conclusions

This paper proposes a SparseTSF-based multi-period spatial long-term time series prediction model (MSTSF) for traffic speed forecasting. This model, based on the SparseTSF model, adds weekly and spatial feature extraction modules to better capture the weekly periodicity and spatial correlation of traffic speeds. This improved model achieves superior performance in traffic speed forecasting.

This paper conducts experiments on the Guangzhou traffic dataset and the PeMS dataset. The experimental results on these two datasets demonstrate that the MSTSF model performs well in both prediction accuracy and model efficiency. This paper also provides a detailed comparison of MSTSF with various deep learning models, including KAN, DeepAR, DLinear, FEDformer, Informer, MLP, NBEATS, NHITS, TCN, TiDE, and TimesNet. Compared to the large-scale time series prediction model TimesNet, the prediction error is reduced by 33.25%. Compared with deep learning models, the MSTSF model offers advantages such as lightweight, small parameters, high iteration efficiency, and fast inference speed, making it suitable for scenarios with limited computing resources. The MSTSF model has only 13.9K parameters, and one iteration takes 3.94 seconds on the entire dataset. This paper also compares the prediction accuracy of the MSTSF model with the DLinear model under different input and prediction lengths. Experimental results show that the average error is reduced by 10% compared to the classic DLinear model in deep learning. Its innovative multi-period feature fusion and spatial correlation mining mechanism provide new ideas and methods for long-term time series prediction.

SparseTSF and its improved MSTSF model show great research potential. This model further promotes the lightweight process of long-term time series prediction models and can significantly reduce the time and space complexity of the model. In the future, we will focus on exploring the in-depth application of the SparseTSF model in the fields of weather forecasting and power forecasting, among others. and strive to provide more efficient methods for dynamic data prediction in these fields.

## Author contributions

Shan Jiang: Methodology; Yuming Feng: Supervision; Jiang Xiong: Supervision. All authors have read and agreed to the published version of the manuscript.

## Use of Generative-AI tools declaration

The authors declare that they have not used Artificial Intelligence (AI) tools in the creation of this article.

## Acknowledgments

Supported by youth project of science and technology research program of Chongqing Education Commission of China (KJQN202501213), Natural Science Foundation of Chongqing (CSTB2024NSCQ-LZX0083), Science and Technology Plan Project of Wanzhou District (wzstc-20240014).

## Conflict of interest

The authors declare no conflicts of interest.

## References

1. T. Alghamdi, K. Elgazzar, M. Bayoumi, T. Sharaf, S. Shah, Forecasting traffic congestion using ARIMA modeling, In: *15th International Wireless Communications & Mobile Computing Conference (IWCMC)*, 2019, 24–28. https://doi.org/10.1109/IWCMC.2019.8766698
2. B. Dissanayake, O. Hemachandra, N. Lakshitha, D. Haputhanthri, A. Wijayasiri, A comparison of ARIMAX, VAR and LSTM on multivariate short-term traffic volume forecasting, In: *Conference of Open Innovations Association, FRUCT*, 2021, 564–570.
3. M. Castro-Neto, Y. S. Jeong, M. K. Jeong, L. D. Han, Online-SVR for short-term traffic flow prediction under typical and atypical traffic conditions, *Expert Syst. Appl.*, **36** (2009), 6164–6173. https://doi.org/10.1016/j.eswa.2008.07.069
4. R. G. Purnima, S. Kiran, P. A. RG, M. S. Vinotheni, P. Ramesh, Traffic light control system and traffic prediction using machine learning with SVR and RFR, In: *2024 4th International Conference on Mobile Networks and Wireless Communications (ICMNWC)*, 2024. https://doi.org/10.1109/ICMNWC63764.2024.10872340
5. X. Luo, D. Li, Y. Yang, S. Zhang, Spatiotemporal traffic flow prediction with KNN and LSTM, *J. Adv. Transport.*, **2019** (2019), 4145353. https://doi.org/10.1155/2019/4145353
6. Y. Kong, Z. Wang, Y. Nie, T. Zhou, S. Zohren, Y. Liang, et al., Unleashing the power of LSTM in long-term time series forecasting, In: *Proceedings of the AAAI Conference on Artificial Intelligence*, **39** (2025), 11968–11976. https://doi.org/10.1609/aaai.v39i11.33303
7. S. Yan, F. Wen, J. Wu, Capacity matching method for mountain scenic areas during holidays based on highway traffic data, in Chinese, *J. Jilin Uni.*, **55** (2025), 1576–1587.
8. J. Fan, K. Zhang, Y. Huang, Y. Zhu, B. Chen, Parallel spatio-temporal attention-based TCN for multivariate time series prediction, *Neural Comput. Applic.*, **35** (2023), 13109–13118. https://doi.org/10.1007/s00521-021-05958-z
9. Z. Dou, D. Guo, DPSTCN: Dynamic pattern-aware spatio-temporal convolutional networks for traffic flow forecasting, *ISPRS Int. J. Geo-Inform.*, **14** (2024), 10. https://doi.org/10.3390/ijgi14010010

10. A. Zeng, M. Chen, L. Zhang, Q. Xu, Are transformers effective for time series forecasting? In: *Proceedings of the AAAI Conference on Artificial Intelligence*, **37** (2023), 11121–11128. https://doi.org/10.1609/aaai.v37i9.26317
11. Y. Song, R. Luo, T. Zhou, C. Zhou, R. Su, Graph attention informer for long-term traffic flow prediction under the impact of sports events, *Sensors*, **24** (2024), 4796. https://doi.org/10.3390/s24154796
12. S. Liu, X. Wang, An improved transformer based traffic flow prediction model, *Sci. Rep.*, **15** (2025), 8284. https://doi.org/10.1038/s41598-025-92425-7
13. X. Zhang, Z. Zhao, J. Li, ARDE-N-BEATS: An evolutionary deep learning framework for urban traffic flow prediction, *IEEE Int. Things J.*, **10** (2022), 2391–2403. https://doi.org/10.1109/JIOT.2022.3212056
14. D. Cheng, Y. Zhang, H. Li, EEMD-TiDE-based passenger flow prediction for urban rail transit, *Electronics*, **15** (2026), 529. https://doi.org/10.3390/electronics15030529
15. T. Zuo, S. Tang, L. Zhang, H. Kang, H. Song, P. Li, An enhanced TimesNet-SARIMA model for predicting outbound subway passenger flow with decomposition techniques, *Appl. Sci.*, **15** (2025), 2874. https://doi.org/10.3390/app15062874
16. I. E. Livieris, C-kan: A new approach for integrating convolutional layers with kolmogorov–arnold networks for time-series forecasting, *Mathematics*, **12** (2024), 3022. https://doi.org/10.3390/math12193022
17. L. A. I. Sicong, H. U. Yuehong. STKAN: Kolmogorov-arnold networks for spatio-temporal forecasting, In: *ICLR 2026 Conference*, 2016.
18. G. Jin, S. Lai, X. Hao, J. Zhang, M. Zhang, M3-net: A cost-effective graph-free mlp-based model for traffic prediction, In: *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, 2025, 4847–4851. https://doi.org/10.1145/3746252.3760815
19. W. Duan, H. Rao, W. Huang, X. He, Minimalist traffic prediction: Linear layer is all you need, preprint paper, 2023. https://doi.org/10.48550/arXiv.2308.10276
20. D. Wang, G. Guo, T. Ouyang, D. Yu, H. Zhang, B. Li, et al., A lightweight spatio-temporal neural network with sampling-based time series decomposition for traffic forecasting, *IEEE Trans. Intell. Transport. Syst.*, **26** (2025), 8682–8693. https://doi.org/10.1109/TITS.2025.3552010
21. Q. Zheng, M. Shao, Y. Zhang, TLAST: A time-lag aware spatial-temporal transformer for traffic flow forecasting, *IEEE Trans. Intell. Transport. Syst.*, **26** (2025), 13144–13158. https://doi.org/10.1109/TITS.2025.3583391
22. Z. Zhong, H.Wu, PASTNet: A prototype-pattern-driven lightweight model for traffic flow forecasting, *J. Yunnan Uni.*, 2025. https://doi.org/10.7540/j.ynu.20250193
23. J. Zhao, F. Zhuo, Q. Sun, Q. Li, Y. Hua, J. Zhao, DSFormer-LRTC: Dynamic spatial transformer for traffic forecasting with low-rank tensor compression, *IEEE Trans. Intell. Transport. Syst.*, **25** (2024), 16323–16335. https://doi.org/10.1109/TITS.2024.3436523
24. J. Xu, J. Yang, Y. Huang, L. Y. Por, X. Chen, C. Zhao, DFGNet: A dual-pathway graph neural network via frequency decomposition for spatiotemporal forecasting, *Expert Syst. Appl.*, **297** (2026), 129518. https://doi.org/10.1016/j.eswa.2025.129518
25. J. Chen, Q. Shao, D. Chen, W. Yu, Decoupling spatio-temporal prediction: When lightweight large models meet adaptive Hhypergraphs, In: *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2025, 167–178. https://doi.org/10.1145/3711896.3736904

26. S. Lin, W. Lin, W. Wu, H. Chen, J. Yang, Sparsetsf: Modeling long-term time series forecasting with 1k parameters, preprint paper, 2024. https://doi.org/10.48550/arXiv.2405.00946
27. M. Reyad, A. M. Sarhan, M. Arafa, A modified Adam algorithm for deep neural network optimization, *Neural Comput. Applic.*, **35** (2023), 17095–17112. https://doi.org/10.1007/s00521-023-08568-z
28. *OpenData V12.0-Large-scale Traffic Speed Data Set*, OpenITS Org, 2021. Available from: http://www.openits.cn/openData4/824.jhtml. Accessed: 2025-05-09.
29. *California department of transportation*, PeMS, 2025. Available from: http://pems.dot.ca.gov.
30. S. Somvanshi, S. A. Javed, M. M. Islam, D. Pandit, S. Das, A survey on kolmogorov-arnold network, *ACM Comput. Surveys*, **58** (2025), 1–35. https://doi.org/10.1145/3743128
31. D. Salinas, V. Flunkert, J. Gasthaus, T. Januschowski, DeepAR: Probabilistic forecasting with autoregressive recurrent networks, *Int. J. Forec.*, **36** (2020), 1181–1191. https://doi.org/10.1016/j.ijforecast.2019.07.001
32. T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, R. Jin, Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting, In: *Proceedings of the 39th International Conference on Machine Learning, PMLR*, **162** (2022), 27268–27286.
33. H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, et al., Informer: Beyond efficient transformer for long sequence time-series forecasting, In: *Proceedings of the AAAI Conference on Artificial Intelligence*, **35** (2021), 11106–11115. https://doi.org/10.1609/aaai.v35i12.17325
34. M. Kasprzyk, P. Pełka, B. N. Oreshkin, G. Dudek, Enhanced N-BEATS for Mid-Term electricity demand forecasting, *Appl. Soft Comput.*, **182** (2025), 113575. https://doi.org/10.1016/j.asoc.2025.113575
35. C. Challu, K. G. Olivares, B. N. Oreshkin, F. G. Ramirez, M. M. Canseco, A Dubrawski Nhits: Neural hierarchical interpolation for time series forecasting, In: *Proceedings of the AAAI Conference on Artificial Intelligence*, **37** (2023), 6989–6997. https://doi.org/10.1609/aaai.v37i6.25854
36. H. Zheng, Y. Lu, Z. Sun, J. Panneerselvam, X. Sun, L. Liu, Energy optimisation in cloud datacentres with MC-TIDE: Mixed channel time-series dense encoder for workload forecasting, *Appl. Energy*, **374** (2024), 123903. https://doi.org/10.1016/j.apenergy.2024.123903129518
37. X. Huang, J. Tang, Y. Shen, Long time series of ocean wave prediction based on PatchTST model, *Ocean Eng.*, **301** (2024), 117572. https://doi.org/10.1016/j.oceaneng.2024.117572
38. H. Wu, J. Xu, J. Wang, M. Long, Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting, *Adv. Neur. Inform. Proc. Syst.*, **34** (2021), 22419–22430.
39. A. Bhatnagar, A. Paule, T. Schuermann, S. Reiter, O. Bringmann, On-device adaptive battery power prediction for electric vehicles, In: *2025 IEEE 11th International Conference on Edge Computing and Scalable Cloud, IEEE*, 2025. https://doi.org/10.1109/EdgeCom66327.2025.00026
40. H. Li, H. Lv, Q. Liu, X. Liu, Y. Zhang, X. Zhou, A contextually enhanced self-attention dilated RNN for load forecasting, In: *2025 IEEE Smart World Congress (SWC). IEEE*, 2025. https://doi.org/10.1109/SWC65939.2025.00110


AIMS Press