



# GeoSAT: long-term time series forecasting with GeoSparse attention and tempo-spatial feature extraction

Lidong Wang<sup>1</sup> · Tao Chen<sup>1</sup> · Xun Li<sup>1</sup> · Tao Huang<sup>2</sup> · Bin Zhou<sup>1</sup> · Yin Zhang<sup>3</sup> · Shuya Ke<sup>1</sup>

Received: 7 March 2026 / Accepted: 11 May 2026  
© The Author(s) 2026

## Abstract

Long-term time series forecasting (LTSF) holds a foundational role in numerous real-world scenarios such as energy monitoring and industrial equipment scheduling, where forecasting systems frequently operate under limited computational resources and require real-time responsiveness. However, existing methods often struggle to precisely extract fine-grained temporal patterns, including long-range trends and cross-variable dependencies, while maintaining a balance between prediction accuracy and computational efficiency in practical deployments. In tackling this challenge, this paper proposes a lightweight LTSF model named GeoSAT that utilizes geometric sparse (GeoSparse) attention and a tempo-spatial feature extraction module. The model first decouples the input time-series data into coarse-grained trend and residual components. For the residual component, a stacked stationary wavelet transform (SSTW) is employed to transform the embedded time domain into a multi-scale frequency domain that are then stacked into a unified frequency-domain representation, enabling the model to capture fine-grained temporal patterns at different scales. Then, compared to traditional sparse attention that relies solely on dot-product, we introduce the GeoSparse attention mechanism applied to the unified frequency-domain representation. It preserves the dot product and introduces the wedge product to effectively capture scalar similarities across different scales as well as geometric correlations between variables. Through sparsification, it highlights critical frequency-domain information and reduces time complexity. For the trend component, a tempo-spatial feature extraction module is employed to model the spatiotemporal domain through adaptive temporal-frequency filtering within variables and one-dimensional convolution between variables. Residual connections are used to extract both the temporal and spatial patterns, thereby capturing the global trend and cross-variable relationships. Evaluations on eight public datasets demonstrate that the proposed model effectively capture the fine-grained features and outperform traditional LTSF methods while ensuring a lightweight architecture. The code is available at: <https://github.com/IntelligentComp-Lab/GeoSAT>.

**Keywords** Long-term time series forecasting · Sparse attention · Stationary wavelet transform · Tempo-spatial feature extraction

✉ Lidong Wang  
wld@hznu.edu.cn

Tao Chen  
2024112032004@stu.hznu.edu.cn

Xun Li  
lixun666@stu.hznu.edu.cn

Tao Huang  
tao.huang1@jcu.edu.au

Bin Zhou  
zhoubin@hznu.edu.cn

Yin Zhang  
yinzhang@zju.edu.cn

Shuya Ke  
kesy@hznu.edu.cn

<sup>1</sup> School of Engineering, Hangzhou Normal University, Hangzhou, China

<sup>2</sup> College of Science and Engineering, James Cook University, Townsville, Australia

<sup>3</sup> School of Computer Science and Technology, Zhejiang University, Hangzhou, China

## 1 Introduction

Multivariate time series (MTS) refers to a collection of multiple variables recorded synchronously at the same timestamp, where each variable can be viewed either as an independent univariate sequence or as a separate channel within the same signal (Hamilton 2020). Multivariate long-term time series forecasting (LTSF) has been widely adopted in critical domains such as meteorological analysis (Hittawe et al. 2024), power dispatching (Zhang et al. 2023), urban traffic management (Lv et al. 2024), and epidemic spread modeling (Bertozzi et al. 2020). The core objective of MTS forecasting is to model dynamic temporal patterns from historical observations and subsequently predict future trends. In recent years, the increasing availability and strategic importance of time series data have driven rapid advancements in forecasting research. Deep learning, in particular, has significantly pushed the boundaries of forecasting accuracy and robustness. Nonetheless, LTSF remains a long-standing challenge due to the inherent complexity and diversity of temporal dynamics.

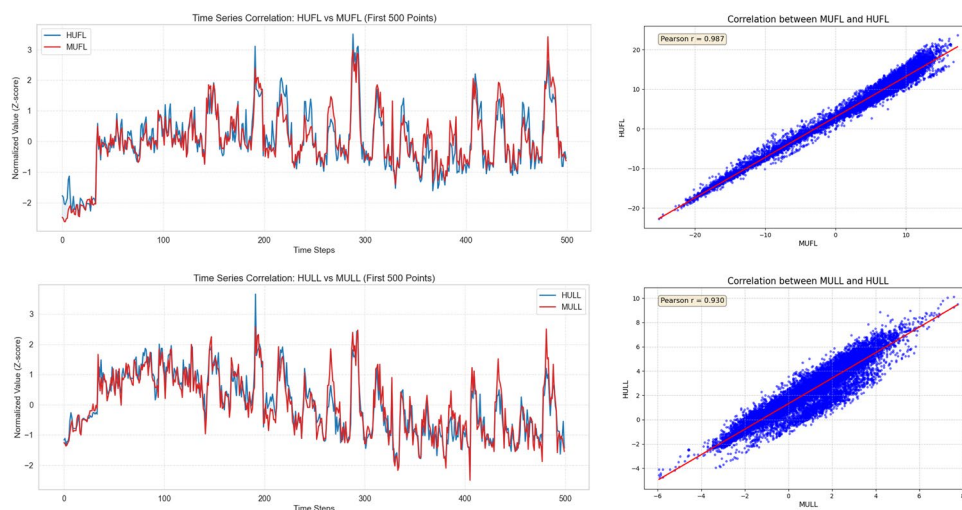
Early forecasting approaches, including neural networks, decision trees, SVM, and random forests (Qiu et al. 2017), generated predictions by identifying underlying patterns. Subsequently, several methods based on RNNs and CNNs have been proposed, including SCINet (Liu et al. 2022a), LSTNet (Lai et al. 2018), and TCN (Bai et al. 2018). However, these methods have limited ability to model long-term dependencies within sequences, focusing primarily on features at a single time scale, and lacking specialized mechanisms to model the complex multi-scale periodic variations in time series. Recent mainstream approaches have concentrated on MLP-based methods (Shiblee et al. 2008) and Transformer-based methods (Wen et al. 2022). Transformer-based approaches have made

significant advancements, with models such as Contiformer (Chen et al. 2023), Tarnet (Chowdhury et al. 2022), Ts-fastformer (Lee et al. 2024), YAO (Zhang et al. 2025) and TVC Former (Liu et al. 2025) emerging. These methods are capable of capturing long-term dependencies in sequences and improving forecasting performance through continuous enhancements to their attention mechanisms. However, recent studies indicate that MLP-based models have outperformed most Transformer models. Furthermore, the simplicity and low computational overhead of MLP architectures have made them increasingly popular among researchers. Models such as MARINA (Xie et al. 2022), MSD-Mixer (Zhong et al. 2023), and TCM (Jiang et al. 2025) have achieved outstanding performance, thus breaking the dominance of Transformers in time series forecasting.

Despite the progress of existing techniques, several challenges persist: (1) Most methods decompose the series into coarse-grained trend and seasonal features; however, real-world time series contain redundant and complex information that current methods fail to effectively extract as fine-grained temporal patterns; (2) Existing self-attention mechanisms treat all time points equally, hindering the capture of complex correlations between variables (e.g., clear positive correlations exist between variables in the ETTh1 dataset, as shown in Fig. 1). Furthermore, the computational cost of traditional attention grows quadratically with the sequence length,  $O(L^2)$ ; (3) Time series often exhibit intricate characteristics across temporal, spatial, and frequency dimensions, requiring comprehensive multi-domain feature extraction while maintaining computational efficiency and scalability.

To address these challenges, we propose the GeoSAT model, a novel lightweight time series forecasting framework that utilizes Top-k geometric sparse (GeoSparse)

**Fig. 1** Correlations between two variable groups in the ETTh1 dataset (MUFL: Medium Useful Load, HUFL: High Useful Load, HULL: High Useless Load, MULL: Medium Useless Load); left: time series trends, right: correlation scatter plots



attention and tempo-spatial feature extraction, built upon the decomposition of time series embedding vectors. Unlike existing models, GeoSAT decomposes the time series into trend and residual components, and designs distinct processing modules for each, targeting the long-term stability of the trend component and the data complexity of the residual component. For the trend component, we introduce a tempo-spatial feature extraction module to learn distinct information from both the temporal and spatial dimensions, effectively capturing within-variable and cross-variable features, and capturing global trend and cross-variable relationships through residual connections. For the residual component, we first apply stacked stationary wavelet transform (SSWT) to convert it into the frequency domain, obtaining fine-grained multi-frequency band coefficients that effectively capture complex time series patterns such as transient variations, periodic modes, and long-term trends. Subsequently, by combining dot-product and wedge-product operations, we design the GeoSparse attention mechanism to capture the relationships between variables. This sparsification highlights important features while significantly reduces the computational complexity.

The contributions of this paper are as follows:

- (1) A novel lightweight LTSF framework, GeoSAT, is proposed based on trend-residual decomposition and multi-domain feature extraction, balancing model lightweighting design and accuracy.
- (2) A GeoSparse attention mechanism is introduced, combining SSWT-based residual information representation with dot-product and wedge-product operations to capture fine-grained correlations and enhance computational efficiency.
- (3) A tempo-spatial feature extraction module is designed to learn features within (temporal domain) and between (spatial domain) variables for the trend component using a transposed structure, with residual fusion to construct global trend representations and inter-variables relationships.
- (4) Experimental results on 8 real-world benchmark datasets demonstrate that GeoSAT outperforms existing mainstream methods in terms of both model lightweighting and accuracy.

In the following sections, Sect. 2 provides an overview of the current research. Section 3 presents the detailed framework of GeoSAT. Section 4 introduces the relevant baselines, experimental details, and a comparison of experimental results. Section 5 concludes the paper with a summary and outlooks for future research.

## 2 Related works

Multivariate LTSF has attracted significant attention recently, leading to substantial progress across multiple research directions. This section reviews prior work from three perspectives: time-series decomposition, Transformer-based models, and MLP-based approaches.

### 2.1 Time decomposition

Time series decomposition (Wen et al. 2019) aims to decompose sequential data into several independent components in order to uncover its intrinsic patterns. This method provides a foundation for trend analysis and forecasting tasks by extracting specific patterns from historical data, facilitating a clearer understanding of the structural characteristics of time series.

Traditional statistical models rely on linearity and stationarity assumptions, making them insufficient for complex real-world scenarios such as industrial sensing (Nizam et al. 2022) and financial markets (Masini et al. 2023). Early machine learning models required extensive manual feature engineering and lacked robust generalization capabilities. Although RNN-based approaches (Waqas and Humphries 2024) enhanced temporal modeling, they suffered from gradient instability and computational limitations.

With the development of deep learning, decomposition-enhanced forecasting models have advanced significantly. Bandara et al. (2025) proposed a multiple-seasonal trend decomposition framework with improved performance and reduced complexity. Hu et al. (2025) introduced a hybrid multi-scale decomposition module that captures temporal patterns at different scales. Su et al. (2024a) decomposed sequences into trend and stationary components across dominant frequencies, boosting forecasting performance via hybrid enhancement. Huang et al. (2024) combined moving-average-based decomposition with deep cross decomposition to effectively extract and integrate seasonal and trend structures.

### 2.2 Transformer-based methods

The Transformer architecture (Vaswani et al. 2017) has become a cornerstone in sequence modeling. Zhou et al. (2021) introduced Informer, which leverages sparse attention and generative decoding to improve efficiency. Wu et al. (2021) incorporated decomposition and periodic autocorrelation for improved temporal modeling. Liu et al. (2022b) applied pyramid multi-scale attention to capture hierarchical

temporal features. Zhou et al. (2022) integrated seasonal-trend decomposition with frequency-enhanced attention for improved global and local pattern understanding. Yu et al. (2024) optimized trend decomposition and autocorrelation learning to capture periodic behaviors more effectively.

Although these models improve efficiency and modeling capability, many still struggle to capture fine-grained correlations across variables and face high computational and memory demands. To address cross-variable relationships, Wang et al. (2024a) proposed Graphformer, combining sparse graph attention with GCNs. Wang et al. (2024b) further emphasized exogenous-endogenous variable interplay and introduced TimeXer for efficient exogenous variable processing. Despite these advances, Transformer-based methods remain computationally heavy and less scalable for long sequences in resource-constrained environments.

### 2.3 MLP-based methods

MLP-based models have recently emerged as efficient alternatives to Transformer-based methods, offering strong performance with lower computational overhead. Zeng et al. (2023) questioned the effectiveness of the Transformer attention mechanism for time series forecasting and demonstrated that simple decomposition and linear layers yield better performance than Transformer-based models. Yi et al. (2023) introduced an efficient time series forecasting method using frequency-domain MLP, opening new directions for the application of MLP in time series modeling and proving the potential of frequency-domain learning. Ekambaram et al. (2023) utilized a lightweight MLP architecture with time series-specific enhancement components. Wang et al. (2024c) proposed an MLP-based FTMLP model to address the issues of neglecting complex relationships between features and the lack of physical interpretability in current methods. Das et al. (2023) presented the TiDE model, an MLP-based encoder-decoder structure, which uses MLP for encoding and decoding, retaining the computational efficiency of linear models.

From the aforementioned studies, it can be concluded that Transformer-based methods have gradually improved through continuous architectural innovations. In contrast, MLP-based methods are efficient in time series modeling. Despite this, self-attention models still struggle to capture fine-grained temporal patterns between variables, while MLP models have weak cross-variable correlation modeling capabilities, and their accuracy requires further improvement. Additionally, balancing the efficiency and accuracy of time series modeling remains a significant challenge. To address these issues, this paper proposes GeoSAT model, which integrates information extraction from both spatiotemporal domain and frequency domain, ensuring the model's lightweight and accuracy.

## 3 GeoSAT model

This section presents the proposed GeoSAT framework for long-term time series forecasting, which consists of two branches: a geometric sparse attention mechanism (Geo-Sparse Attention) and a tempo-spatial feature extraction module. The overall architecture is illustrated in Fig. 2. GeoSAT consists of three core components: a feature decomposition module, an SSWT-based GeoSparse attention module, and a tempo-spatial feature extraction module.

The feature decomposition module embeds the time series into high-dimensional vectors and decomposes them into trend and residual components. For the trend component, the tempo-spatial feature extraction module applies adaptive frequency filtering in the time domain and one-dimensional convolution in the spatial domain to extract feature information within and between variables. For the residual component, the GeoSparse attention module decomposes the sequence into multiple-scale frequency bands using stacked stationary wavelet transform (SSWT). Geometric attention weights are calculated within each frequency band, and multi-scale information is aggregated. Finally, the outputs of the SSWT-based GeoSparse attention and the tempo-spatial feature extraction module are linearly mapped and concatenated, followed by a denormalization process to return to the original time domain, forming the predicted sequence.

### 3.1 Problem definition

In this work, the problem of multivariate LTSF is formulated as follows. Given a historical observation sequence  $X = \{x_1, \dots, x_L | x_i \in \mathbb{R}^M\}$ , where  $X \in \mathbb{R}^{M \times L}$ , the task is to predict the future sequence  $X' = \{x_{L+1}, \dots, x_{L+T} | x_i \in \mathbb{R}^M\}$ , where  $X' \in \mathbb{R}^{M \times L'}$ . Here,  $L$  and  $L'$  represent the lengths of the input and output sequences, respectively, and  $M$  denotes the number of variables or channels. The key notations used throughout this paper are summarized in Table 1.

### 3.2 Feature decomposition

Existing time series often exhibit complex temporal behaviors, including seasonality, trends, and various irregular influencing factors. First, we perform a normalization operation on the input sequence, defined as:

$$\tilde{x}_i^j = \frac{x_i^j - \mu^j}{\sigma^j} \quad (1)$$

where  $x_i^j$  represents the value of the  $j$ -th channel at the  $i$ -th time step in the original data, and  $\mu^j$  and  $\sigma^j$  represent the mean and standard deviation of the  $j$ -th channel over all time steps, respectively.

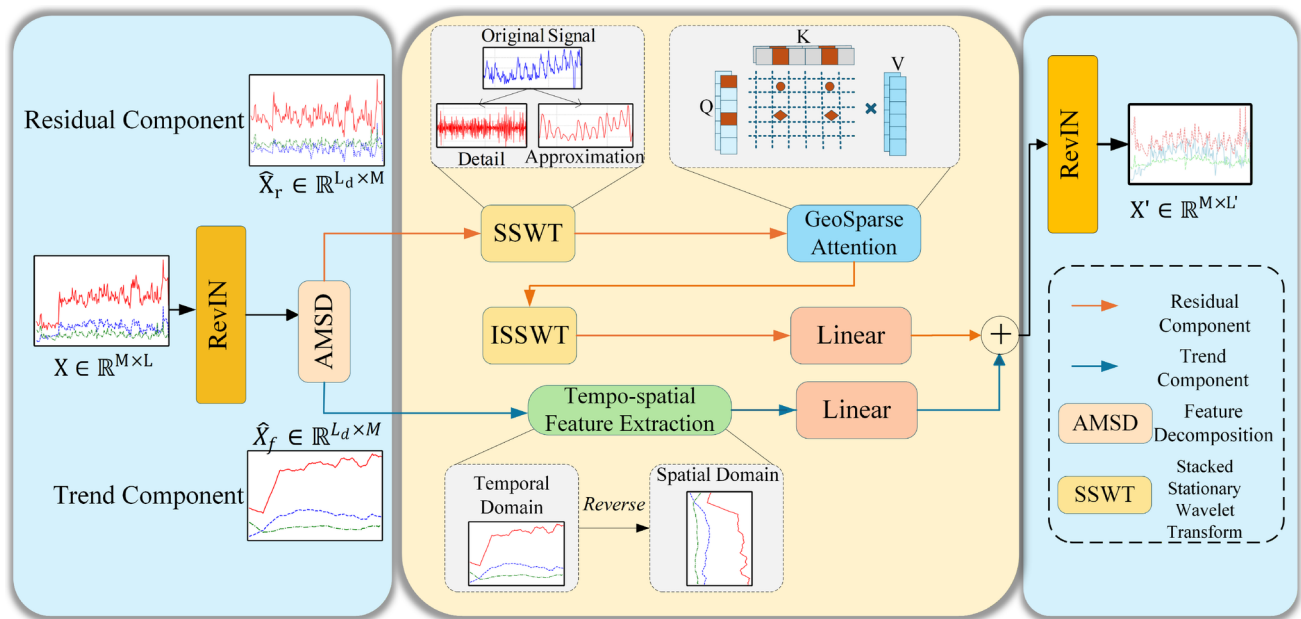


Fig. 2 The framework of GeoSAT

Table 1 Key notations

| Terms                  | Notations                                                              |
|------------------------|------------------------------------------------------------------------|
| $X$                    | Input MTS                                                              |
| $X'$                   | Forecasted $_{MTS}$                                                    |
| $L$                    | Length of the input MTS                                                |
| $L'$                   | Length of the predicted series                                         |
| $M$                    | Number of channels/variables                                           |
| $L_d$                  | Size of the embedding dimension                                        |
| $\tilde{x}_i$          | Normalized input time series                                           |
| $\hat{X}$              | Output of inverse embedding                                            |
| $\hat{X}_f$            | Trend component after feature decomposition                            |
| $\hat{X}_r$            | Residual component after feature decomposition                         |
| $h_0, h_1$             | Low-pass and high-pass filters of SSWT                                 |
| $s$                    | Decomposition scale of SSWT                                            |
| $\hat{X}_{RES}$        | Output of stacked approximation and detail coefficients                |
| $A_t^{(s)}, D_t^{(s)}$ | Approximation and detail coefficients obtained from SSWT decomposition |
| $Z_{dot}$              | $Q - K$ dot product in GeoSparse Attention                             |
| $Z_{wedge}$            | Geometric distance between $Q$ and $K$ in GeoSparse Attention          |
| $Z$                    | Weight matrix after combining dot product and geometric distance       |
| $X'_{RES}$             | Result of aggregating the attention matrix with the Value matrix       |
| $\hat{X}'_r$           | Output of ISSWT                                                        |
| $X_{freq}$             | Output of temporal domain                                              |
| $X_{SD}$               | Output of spatial domain                                               |
| $\hat{X}'_f$           | Output of tempo-spatial feature extraction                             |

Each feature of the time series is then mapped individually to a high-dimensional space through a linear layer. This “inverted” embedding approach is particularly suitable for handling MTS, as it allows the model to process the temporal patterns of each variable separately. By using different processing paths for trend and residual features, the model can more effectively capture the complex dependencies within multivariate time series. For the normalized sequence  $\tilde{X} \in \mathbb{R}^{M \times L}$ , the inverted embedding can be represented as:

$$\hat{X} = \text{Linear}(\tilde{X}) \in \mathbb{R}^{L_d \times M} \quad (2)$$

where  $L_d$  denotes the size of the mapped space.

### 3.2.1 Adaptive Multi-Scale Sequence Decomposition (AMSD)

Considering the potential independent distribution characteristics among variables in MTS, the module utilizes multiple groups of depthwise separable convolution kernels to parallelly extract trend features at different receptive fields, and introduces a learnable attention mechanism to dynamically integrate multi-scale information. We adopt a channel-independent approach using convolution kernels of different sizes for smoothing. The channel-independent processing prevents information aliasing across different feature channels in MTS data, thereby accurately capturing the multi-scale trends of each individual feature. Through a channel-independent convolution design, the module can precisely handle the variable-specific nature of MTS, enhancing the model's ability to capture trend changes. AMSD module allows the network to automatically focus on the most relevant temporal scales based on the data characteristics.

For the  $i$ -th variable in the input sequence  $\hat{X}$ , we apply independent convolution kernels for smoothing, and the operation can be represented as:

$$\text{Trend}_i^j = \text{Cov1D}^j(\hat{X}_i) \quad (3)$$

where  $i = 1, 2, \dots, M$ . The module is designed to process  $M$  variables independently to avoid cross-variable information interference. For a given set of convolution kernel sizes  $T = \{t_1, t_2, \dots, t_n\}$ ,  $t_j$  represents the kernel size, corresponding to different time scales (such as short-term, medium-term, and long-term).

To integrate trend information from different scales, a set of trainable weight parameters  $\alpha = \{\alpha_1, \alpha_2, \dots, \alpha_n\}$  is introduced corresponding to each scale, enabling automatic weight learning for each scale and assigning high weights to the critical trend scales adaptively. The trend features extracted from each scale are weighted and aggregated by calculating the Softmax distribution of the scale weights. The AMSD outperforms standard moving average by simultaneously extracting multi-scale trends and adaptively

learning their importance, leading to more accurate and robust time series decomposition. The weighted fusion of multi-scale trend components is as follows:

$$\hat{X}_{f_i} = \sum_{j=1}^n \text{Softmax}(\alpha_j) \cdot \text{Trend}_i^j \quad (4)$$

Equation (4) aggregates the trend terms of the  $i$ -th variable across all scales. Upon completion of the aggregation for all  $M$  variables, the final trend component  $\hat{X}_f$  is obtained.

After obtaining the final trend estimate, the residual part  $\hat{X}_r$  is computed as follows:

$$\hat{X}_r = \hat{X} - \sum_{i=1}^M \hat{X}_{f_i} = \hat{X} - \hat{X}_f \quad (5)$$

where  $\hat{X}_f$  and  $\hat{X}_r$  represent the extracted smooth component and residual component, respectively. Considering the complexity of the residual component, it is processed by the subsequent GeoSparse attention module, while the trend component is passed to the tempo-spatial feature extraction module for further processing.

## 3.3 SSWT-based GeoSparse attention

GeoSparse Attention enhances traditional self-attention through geometric transformations. As shown in Fig. 3, to map the input sequence to a frequency domain more suitable for capturing temporal geometric structures, the sequence is decomposed into multi-scale frequency bands using Stacked Stationary Wavelet Transform (SSWT). In the frequency domain, geometric products are used to effectively capture variable correlations. Attention weights are computed for each frequency band using different attention heads, and the multi-scale information is then aggregated. To accommodate long-sequence scenarios, Top-k sparsification is applied to retain key dependencies, thereby reducing computational complexity.

### 3.3.1 SSWT

The Stationary Wavelet Transform (SWT) was originally proposed to address the shift sensitivity issue of the Discrete Wavelet Transform (DWT) (Nason and Silverman 1995). In SWT, the high-pass and low-pass filters are upsampled at each decomposition level, ensuring that the approximate and detail coefficients generated after each decomposition layer have the same length as the original signal. This makes SWT invariant to both time and translation, making it suitable for processing heterogeneous time series data (e.g., time series of different variables), and extracting both local and global patterns across scales.

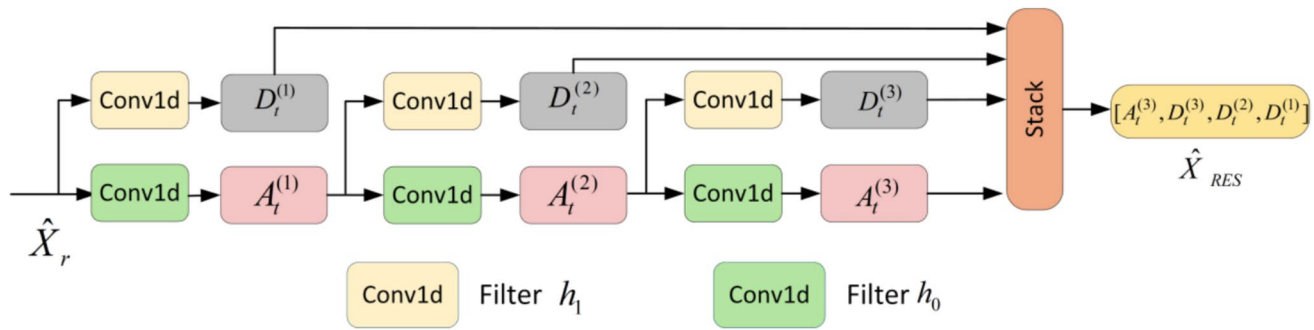


Fig. 3 The structure of SSWT

SSWT further extends SWT by performing multi-scale sparse stacking on the outputs. The specific definition is as follows:

$$\hat{X}_{RES} = SSWT(\hat{X}_r; h_0, h_1) : \mathbb{R}^{L_d \times M} \rightarrow \mathbb{R}^{L_d \times (s+1) \times M} \quad (6)$$

where  $h_0, h_1 \in \mathbb{R}^{M \times z}$  are learnable low-pass and high-pass filters with a kernel size of  $z$ ,  $s$  is the decomposition scale, and  $\hat{X}_{RES}$  represents the output in the frequency domain space.

Based on the mother wavelet  $\varphi(t)$  and the scaling function  $\phi(t)$ , the SWT (subband decomposition) decomposes the signal into  $s$  scales of approximation coefficient  $A_t^{(s)}$  (low-frequency components) and detail coefficient  $D_t^{(s)}$  (high-frequency components). The decomposition process of SSWT can be recursively defined as follows:

Initialization, we set  $A_t^{(0)} = \hat{x}_t$ , where  $\hat{x}_t$  represents the signal  $\hat{X}_r$  at a particular time point. For each scale  $s \in [1, 2, \dots, S]$ , we compute:

$$A_t^{(s)} = \sum_z h_0^{(s-1)}(z) \cdot A_{(t+z)}^{(s-1)} \quad (7)$$

$$D_t^{(s)} = \sum_z h_1^{(s-1)}(z) \cdot A_{(t+z)}^{(s-1)} \quad (8)$$

Here,  $h_0^{(s)}$  and  $h_1^{(s)}$  are upsampled versions of the original filters  $h_0$  and  $h_1$ , where  $2^{s-1} - 1$  zeros are inserted between every two coefficients of the original filters. This upsampling ensures that the filter length and scale are compatible, and these parameters are learnable.

For the input sequence in Fig. 3, the first layer of  $A_t^{(1)}$  and  $D_t^{(1)}$  are obtained by convolution with the low-pass and high-pass filters, respectively. Then,  $A_t^{(1)}$  is further decomposed through low-pass and high-pass filters to obtain the second layer's approximation coefficients  $A_t^{(2)}$  and detail coefficients  $D_t^{(2)}$ . The above process continues until the specified decomposition scale  $s$  is reached (with  $s = 3$  shown in the Fig. 3 as an example). Finally, all the detail coefficients  $D_t^{(s-1)}, D_t^{(s)}$ ,

along with the last approximation coefficient  $A_t^{(s)}$  obtained from the final decomposition, are stacked together to produce the final output, denoted as:

$$\hat{X}_{RES} = Stack(A_t^{(s)}, D_t^{(s)}, D_t^{(s-1)}, \dots, D_t^{(1)}) \quad (9)$$

The stacking operation adds a new dimension, where the approximation coefficient  $A_t^{(s)} \in \mathbb{R}^{M \times L_d}$  and the detail coefficient  $D_t^{(s)} \in \mathbb{R}^{M \times L_d}$  from the SWT are stored hierarchically. After stacking, the output  $\hat{X}_{RES} \in \mathbb{R}^{L_d \times (s+1) \times M}$  is obtained, enabling the multi-scale features to be processed simultaneously after the wavelet decomposition transformation.

It has to be noted that the wavelet filters in SSWT are initialized with canonical wavelet bases and then treated as learnable parameters. Unlike fixed wavelets, learnable wavelets adapt the decomposition to the LTSF task. The model automatically learns task-optimal time–frequency representations without manual wavelet selection, while preserving interpretability. During training, the coefficients are adaptively fine-tuned via back propagation to better fit the data distribution. Our SSWT decomposition and reconstruction strictly follow the standard stationary wavelet transform: we use dilated convolution without down-sampling to maintain shift-invariance and temporal resolution. The inverse SSWT can still perfectly reconstruct the original signal, which ensures that the wavelet mathematical properties are maintained.

### 3.3.2 GeoSparse attention

Existing research has witnessed continuous advancements in attention mechanisms. For instance, the cosine attention mechanism offers the advantage of computational simplicity and effectively captures the similarity between vectors (Luo et al. 2018). However, its normalization of vector lengths results in the loss of magnitude information. Starting from the positional information of tokens in the sequence, RoPE (Su et al. 2024b) uses a rotation transformation to ensure that the attention score naturally depends only on the relative

position difference, independent of their absolute positions, focusing on modeling the relative positional relationship between tokens.

Despite these advances, existing attention mechanisms still face limitations in capturing the geometric structure between features (such as linear independence and spatial complementarity) and in handling long sequences. The dot-product only measures scalar-level similarity, making it difficult to capture the complex positive and negative correlations among multiple variables. Furthermore, the computational complexity of fully connected attention grows quadratically with the input series length, making it inefficient for long sequences forecasting tasks. In contrast to the above methods, GeoSparse Attention is derived from the perspective of geometric algebra. It is a lightweight design that integrates dot and wedge products to preserve length information and model richer geometric relationships between vectors.

As shown in Fig. 4, GeoSparse Attention replaces the traditional dot product with wedge products, decomposing the attention computation into two parts: scalar similarity (dot product) and bi-vector geometric relations (wedge product).

For the multi-scale signal  $\hat{X}_{RES}$  obtained from wavelet transform decomposition, we adopt a scale-head correspondence design, where each scale is used as input to GeoSparse attention. Specifically, suppose that  $s$  is the wavelet

decomposition scale, we apply each coefficient matrix ( $Q, K$ , and  $V$ ) corresponding to the  $s + 1$  scales to different attention heads in the attention computation. In total, there are  $s + 1$  attention heads, with each attention head corresponding to a distinct  $Q, K$ , and  $V$  matrix.

**Wedge product review** For Query-Key pair  $(q_i, k_j)$  in matrix  $Q, K$ , the geometric product between them can be expressed as:

$$\text{wedge\_product}(q_i, k_j) = q_i \cdot k_j + q_i \wedge k_j \quad (10)$$

where  $q_i \cdot k_j$  means dot product,  $q_i \wedge k_j$  means wedge product. Unlike the traditional dot product, the geometric product is composed of a dot product (scalar) and a wedge product (bivector). While the dot product component represents the similarity and correlation between vectors, the wedge product explicitly encodes the plane spanned by them. In MTS, this plane can be interpreted as the independence and complementarity between two channels. While the wedge product does not directly quantify "likeness" as the dot product does, it characterizes the "dissimilarity" between vectors. By integrating both the dot and wedge products, the geometric product enables a more comprehensive modeling of cross-channel dependencies. Since the output of  $q_i \wedge k_j$  is in the form of a bivector, we adopt the magnitude of the wedge product, denoted as  $\|q_i \wedge k_j\|$ , as the final output. The specific calculation is formulated as follows:

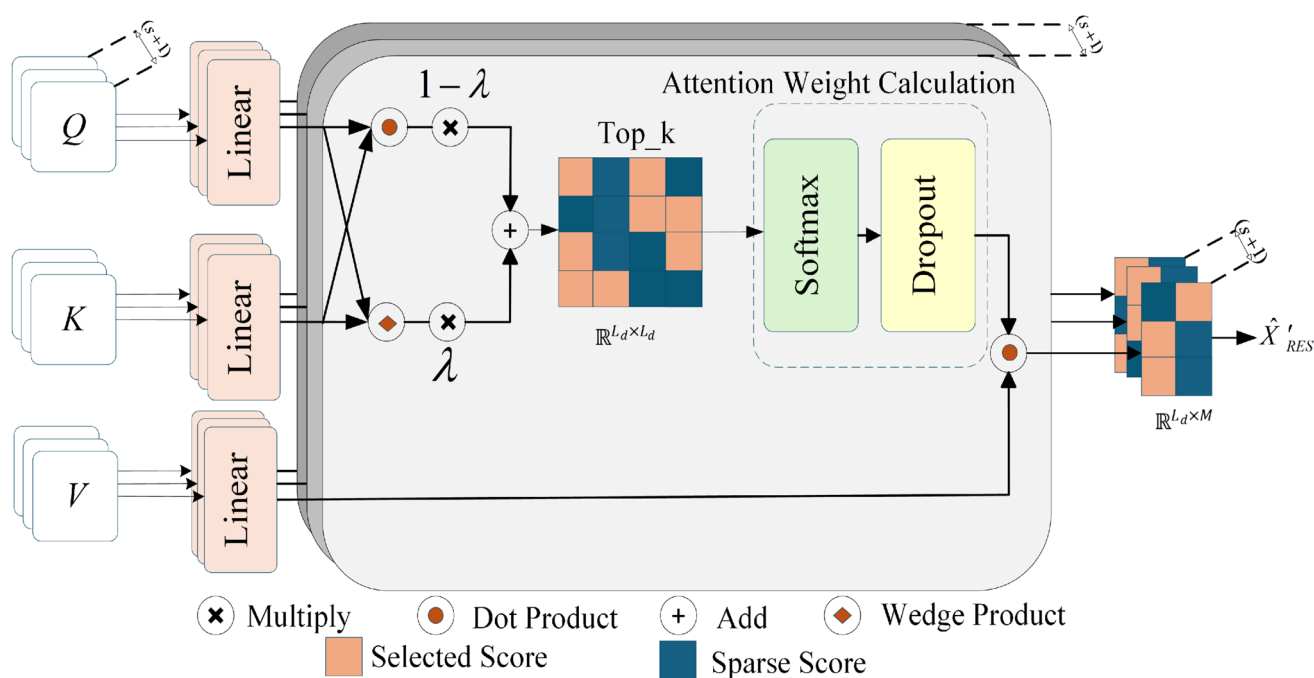


Fig. 4 GeoSparse attention module

$$||q_i \wedge k_j|| = \sqrt{(q_i \cdot q_i)(k_j \cdot k_j) - (q_i \cdot k_j)^2} \quad (11)$$

**GeoSparse attention calculation** First, the similarity matrix  $Z_{dot}$  between the query  $Q$  and the key  $K$  is computed for each attention head using the standard dot-product formulation:

$$Z_{dot} = QK^T \in \mathbb{R}^{L_d \times L_d} \quad (12)$$

where  $Q, K \in \mathbb{R}^{L_d \times M}$ . In geometric attention, the core is to measure the perpendicular distance between two vectors using the wedge norm. For the given  $Q$  and  $K$ , we first compute their squared  $\ell_2$ -norms to obtain the norm-squared tensors  $Q_{norm}$  and  $K_{norm}$ . The norm-squared tensor for  $Q_{norm}$  is as follows:

$$Q_{norm} = [q_1^2, q_2^2, \dots, q_M^2] \quad (13)$$

where  $q_m \in \mathbb{R}^{L_d}$  represents the  $m$ -th column vector in the matrix  $Q_{norm}$ . The corresponding norm-squared tensor  $K_{norm}$  is computed analogously. Finally, the wedge norm  $Z_{wedge}$ , which characterizes the geometric (bi-vector) relationship between  $Q$  and  $K$ , is computed as:

$$Z_{wedge} = ||Q \wedge K|| = \sqrt{Q_{norm} \times K_{norm}^T - Z_{dot}^2} \quad (14)$$

Secondly, we integrate the two through parameter  $\lambda \in [0, 1]$  to obtain a blended score:

$$Z = (1 - \lambda) \cdot Z_{dot} + \lambda \cdot Z_{wedge} \quad (15)$$

where  $Z \in \mathbb{R}^{L_d \times L_d}$ , and  $Z_{i,j}$  represents the combined attention of the  $i$ -th Query to the  $j$ -th Key. This mixed score captures both the synergistic patterns between features and the complementary dependencies, making it especially suitable for modeling cross-dimensional dynamic relationships in MTS.

To adapt to long sequence scenarios, Top- $k$  sparsification is applied to retain key dependencies. Specifically, for each  $Z_i$ , we retain the top  $k$  values with the highest scores, setting the remaining values to  $-\infty$ . Traditional attention computes the scores for all Query-Key pairs with a complexity of  $O(L_d \cdot L_d)$ , while GeoSparse Attention reduces the complexity to  $O(L_d \cdot k)$ , aiming to preserve the most relevant information while significantly improving computational efficiency. The sparsification process is described as follows:

$$\hat{Z}_{i,j} = \begin{cases} Z_{i,j} & \text{if } Z_{i,j} \in \text{Top}_k(Z_i, k) \\ -\infty & \text{else} \end{cases} \quad (16)$$

where  $i, j \in \{1, 2, \dots, L_d\}$ ,  $\text{Top}_k(\cdot, k)$  indicates the top  $k$  scores.

Finally, the attention weight matrix  $Z_{i,j}$  is multiplied with the matrix  $V$ :

$$X'_{RES} = Z_{i,j} \cdot V \quad (17)$$

where  $X'_{RES} \in \mathbb{R}^{L_d \times M}$  represents the final feature matrix. This matrix contains the Value information corresponding to the Top- $k$  Keys that the Query attends to. The output feature of an attention head is denoted as  $X'_{RES}$ , and we stack the output features of  $s + 1$  attention heads to obtain  $\hat{X}'_{RES} \in \mathbb{R}^{L_d \times (s+1) \times M}$ . These steps illustrate that important feature information for each Query-Key pair can be extracted through GeoSparse attention.

### 3.3.3 Inverse SSWT

After feature extraction through GeoSparse attention, the wavelet signal needs to be inverse-transformed into time-series data via the Inverse SSWT (ISSWT). Taking  $s = 2$  as an example, the number of iterations is set as the decomposition scale, and the approximation coefficients (denoted as  $A'_t{}^{(2)}$ ) and detail coefficients (denoted as  $D'_t{}^{(1)}$  and  $D'_t{}^{(2)}$ ) are separated from the input coefficients  $\hat{X}'_{RES}$ . The detail coefficients at each scale are processed through cyclic padding (shown on the right side of Fig. 5) and convolution operations (using reconstruction filters ( $g_0$ ) and ( $g_1$ )). In the circular padding phase, given a time series  $X$ , padding of two left and right boundary values is performed. For left boundary padding, the last part of the original signal is reversed and truncated; for right boundary padding, the original signal is truncated from the beginning in the forward direction. The next step involves scaling by dividing by 2, ensuring signal energy conservation to maintain the accuracy of the reconstruction.

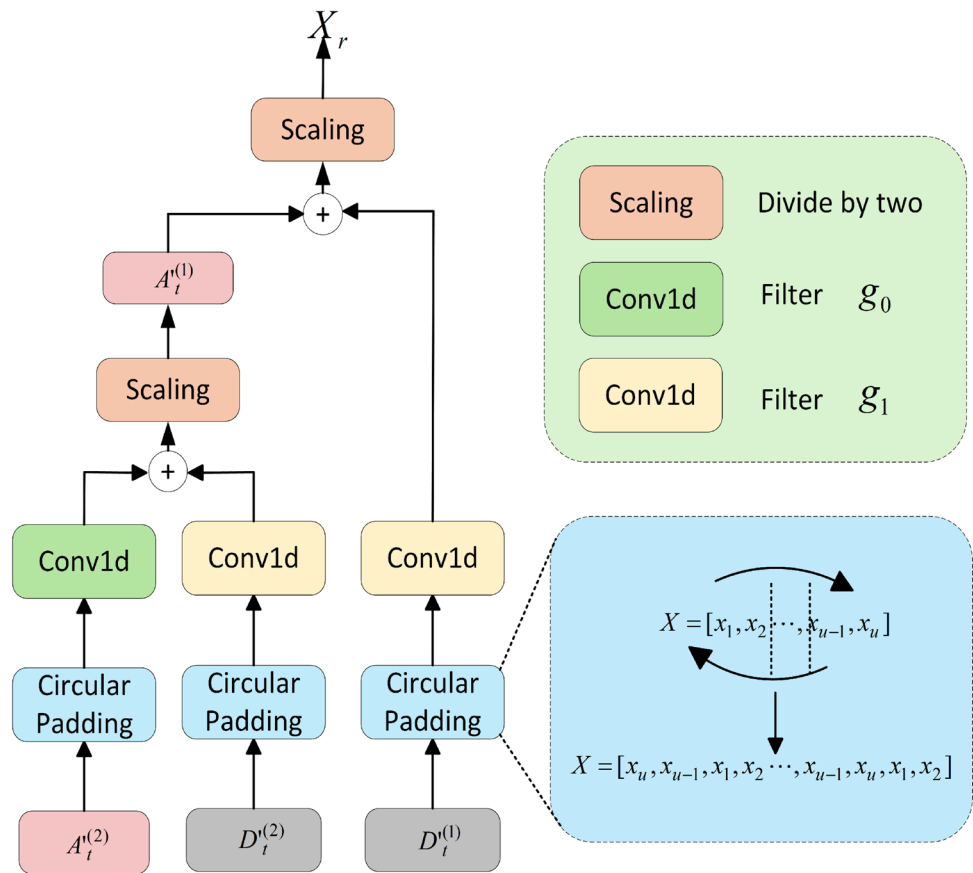
Then, the current layer's detail coefficients  $D'_t{}^{(2)}$  are fused with the approximation coefficients  $A'_t{}^{(2)}$  of the same layer, updating the approximation coefficients  $A'_t{}^{(1)}$ , until the  $s$ -layer process is complete and the final reconstructed signal is output. ISSWT preserves the temporal continuity of the signal through dilated convolutions and cyclic padding.

$$X_r = \text{ISSWT}(\hat{X}'_{RES}; g_0, g_1) : \mathbb{R}^{L_d \times (s+1) \times M} \rightarrow \mathbb{R}^{L_d \times M} \quad (18)$$

$$\hat{X}'_r = \text{Linear}(X_r) \quad (19)$$

The parameters  $g_0$  and  $g_1$  represent the wavelet reconstruction filters, corresponding to the low-frequency reconstruction kernel and high-frequency reconstruction kernel, respectively. These filters are used to recombine the wavelet low-frequency and high-frequency coefficients to reconstruct the original signal. Finally,  $X_r$  is linearly projected to produce an output sequence of predicted length  $L'$ .

Based on the above analysis, the pseudo of SSWT-based GeoSparse attention can be summarized as follows:

**Fig. 5** The structure of ISSWT**Algorithm 1** Pseudo of the SSWT-based GeoSparse attention.

---

|                                                                                                                                  |                                                               |
|----------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------|
| Input: $\hat{X}_r \in \mathbb{R}^{L_d \times M}$ :The residual component, $k$ : sparse coefficient                               |                                                               |
| Output: $\hat{X}_r' \in \mathbb{R}^{L' \times M}$ :The reconstructed residual component                                          |                                                               |
| 1: $\hat{X}_{RES} = SSWT(\hat{X}_r; h_0, h_1): \mathbb{R}^{L_d \times M} \rightarrow \mathbb{R}^{L_d \times (s+1) \times M}$     | ▷ Conduct Stacked Stationary Wavelet Transform on $\hat{X}_r$ |
| 2: $(Q, K, V)_u \leftarrow \hat{X}_{RES}, u \in [1, s+1]$                                                                        | ▷ Assign $\hat{X}_{RES}$ to $Q, K, V$ for different heads     |
| 3: $Z \leftarrow (1-\lambda) \cdot Z_{dot} + \lambda \cdot Z_{wedge}$                                                            | ▷ Calculate Geometric product scores                          |
| 4: $\hat{Z}_{i,j} \leftarrow \begin{cases} Z_{i,j} & \text{if } Z_{i,j} \in Top\_k(Z_i, k) \\ -\infty & \text{else} \end{cases}$ | ▷ Process of sparsification                                   |
| 5: $X'_{RES} \leftarrow Z_{i,j} \cdot V$                                                                                         | ▷ Value aggregation                                           |
| 6: $X_r = ISSWT(\hat{X}'_{RES}; g_0, g_1): \mathbb{R}^{L_d \times (s+1) \times M} \rightarrow \mathbb{R}^{L_d \times M}$         | ▷ Inverse SSTW                                                |
| 7: $\hat{X}_r' \leftarrow Linear(X_r)$                                                                                           | ▷ Project back to $\hat{X}_r'$                                |
| 8: Return $\hat{X}_r'$                                                                                                           |                                                               |

---

### 3.4 Tempo-spatial feature extraction module

For the trend component, in order to effectively capture the complex spatio-temporal dependencies in MTS data, we propose a hybrid module by combining adaptive frequency filtering within variables (time dimension) and spatial local convolution between variables (spatial dimension). This module models the global periodic features along the time dimension and the inter-variable correlations along the spatial dimension.  $\hat{X}_f$  is input into this module, and after processing by adaptive frequency filtering and spatial local convolution, the two parts of the data are connected through a residual connection. As shown in Fig. 6, this fusion of information from two distinct perspectives facilitates feature extraction both within and between variables.

#### 3.4.1 Adaptive frequency filtering

Traditional time-domain methods are limited by local receptive fields, making it difficult to efficiently capture long-period global patterns. To address this, we use the Discrete Fourier Transform (DFT) to convert the input signal into the frequency domain. Given the input tensor  $\hat{X}_f$ , we first apply the Real Fast Fourier Transform (RFFT) along the time dimension:

$$X_{TD} = RFFT(\hat{X}_f) \in \mathbb{R}^{(L_d/2+1) \times M} \quad (20)$$

To enable the model to selectively attend to different frequency components, we introduce a set of learnable complex weight parameters  $W_1 \in \mathbb{R}^{L_d/2+1}$ . These weights are shared across all channels and applied to perform element-wise modulation of the spectrum:

$$X'_{TD} = X_{TD} \odot W_1 \quad (21)$$

Subsequently, the signal is restored to the time domain through the Inverse Fast Fourier Transform (IFFT), and a residual connection is introduced to retain the original time-domain information:

$$X_{freq} = LayerNorm(RFFT^{-1}(X'_{TD}) + \hat{X}_f) \quad (22)$$

The mechanism allows the model to capture global patterns, addressing the limitations of local window methods in modeling long sequences.

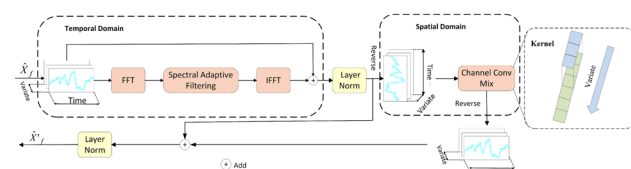


Fig. 6 The architecture of tempo-spatial feature extraction module

#### 3.4.2 Spatial-domain channel mixing

Following the extraction of temporal features, we employ a 1D convolutional network for feature fusion to model the spatial interdependencies across channels. Unlike the global perspective of frequency-domain operations, spatial mixing focuses on the interaction of locally adjacent variables. We swap the variable dimension and time dimension of the tensor  $X_{freq}$  processed in the frequency domain and apply a 1D convolution operation:

$$X_{SD} = Conv1d(X_{freq}^T) \quad (23)$$

This convolution operation aggregates the feature information of adjacent variables through a sliding window mechanism, thereby capturing the spatial dependencies or topological structure between variables. The final output also uses residual connections and layer normalization:

$$\hat{X}'_f = LayerNorm(X_{SD}^T + X_{freq}) \quad (24)$$

Frequency-domain operations leverage the global characteristics of the Fourier transform, effectively overcoming the receptive field limitations of traditional time-domain methods in long-range dependency modeling, enabling the precise capture of inherent temporal patterns within signals at low computational cost. Meanwhile, spatial-domain convolutions capture the relationships between multiple variables. This cross-domain collaborative mechanism enables modeling of both spatial and temporal dimensions and deep feature fusion, providing a more expressive and robust representation for handling complex MTSF tasks.

### 3.5 Loss function

To jointly optimize predictive accuracy and structural sparsity, we design a composite loss function, which includes a basic loss term  $L_{base}$  for supervised prediction accuracy and a regularization term  $L_{reg}$  aimed at enhancing the interpretability and sparsity of the attention mechanism. The total loss function is defined as follows:

$$L_{total} = L_{base}(Y, \hat{Y}) + \gamma \cdot L_{reg}(W) \quad (25)$$

where  $Y$  and  $\hat{Y}$  denote the true values and predicted values, respectively;  $W$  denotes the attention weight matrix learned by the model; and  $\gamma$  is a tunable parameter used to balance the contributions of the two loss terms.

The basic loss  $L_{base}$  is constructed using the MSE to better penalize larger prediction errors:

$$L_{base}(Y, \hat{Y}) = \frac{1}{N} \sum_{i=1}^N (Y_i - \hat{Y}_i)^2 \quad (26)$$

where  $N$  denotes the number of training samples. For the sparse attention regularization term, it operates on the attention weight matrix  $W$ . Its design is motivated by the goal of encouraging the model to learn concise and interpretable spatiotemporal dependencies. It is defined as  $L_{reg} = \text{mean}(|W|)$ . By applying this to the attention weights, we can utilize the attention mechanism to focus on a few key input elements during training, rather than distributing attention across all inputs. During training, the model simultaneously optimizes predictive accuracy and structural sparsity by minimizing the total loss. During validation and testing, we evaluate the model's pure prediction performance using only the basic loss term. A detailed parameter analysis is presented in Sect. 4.7.

### 3.6 Complexity analysis

In this section, we analyze the computational complexity of the GeoSAT model. For the inverted embedding module, the input is projected into a high-dimensional space with a complexity of  $O(L \cdot M \cdot L_d)$ . For the AMSD module, a moving-average operation is applied to decompose the input time series, resulting in a complexity of  $O(M \cdot L_d)$ . The residual component is further processed by the SSWT based GeoSparse attention mechanism. For the SSWT

module, the complexity is  $O(s \cdot L_d \cdot M)$ . For the GeoSparse attention module, the complexity becomes  $O(k \cdot L_d \cdot M)$ , compared with  $O(L_d^2 \cdot M)$  before sparsification. Under our model configuration  $k \ll L_d$ , this sparsification significantly reduces computational cost, effectively improving model efficiency while achieving superior performance. For the trend component, we apply the tempo-spatial feature extraction module for processing, whose complexity is  $(\log L_d \cdot L_d \cdot M)$ . In summary, the overall complexity of GeoSAT is  $O(L \cdot M \cdot L_d + M \cdot L_d + s \cdot L_d \cdot M + k \cdot L_d \cdot M + \log L_d \cdot L_d \cdot M)$ .

In GeoSparse Attention module, conventional attention requires computing correlation weights for all Query-Key pairs, whereas Top-k sparsification allows us to focus only on the top-k most important keys, making the attention score computation nearly linear in complexity. In addition, although the SSWT involves  $s$  layers of convolution due to multi-scale decomposition, the value of  $s$  is usually a small constant, so its impact on overall efficiency is limited. Overall, GeoSparse attention achieves substantial reductions in computational cost while preserving modeling capability by combining wavelet-based multi-scale feature extraction. When  $L_d$  is set to a small value, the overall model complexity approaches linear time. This design is particularly suitable for efficient training and inference on long sequences or multivariate scenarios.

The pseudocode of GeoSAT is shown in Algorithm 2.  
Algorithm 2 Pseudocode of GeoSAT.

---

|                                                                                                                                                   |                                           |
|---------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------|
| Input : $X \in \mathbb{R}^{M \times L}$ : the input sliding window, $\lambda$ : score weight, $s$ : decomposition scale, $k$ : sparse coefficient |                                           |
| Output: $X' \in \mathbb{R}^{M \times L'}$ : predicted time series                                                                                 |                                           |
| 1: $\tilde{X} \leftarrow \text{Revin}(X)$                                                                                                         | ▷ Input normalization                     |
| 2: $\hat{X} \leftarrow \text{Linear}(\tilde{X}^T)$                                                                                                | ▷ Embed into a high-dimensional vector    |
| 3: $\hat{X}_r, \hat{X}_f \leftarrow \text{Decompose}(\hat{X})$                                                                                    | ▷ Decompose the embedding vector          |
| 4: $\hat{X}'_r \leftarrow \text{GeoSparseAttention}(\hat{X}_r)$                                                                                   | ▷ Geometric sparse attention              |
| 5: $\hat{X}'_f \leftarrow \text{TSFE}(\hat{X}_f)$                                                                                                 | ▷ Tempo-spatial feature extraction module |
| 6: $X' \leftarrow \text{Revin}(\hat{X}'_r + \hat{X}'_f)^T$                                                                                        | ▷ Inverse normalization                   |
| 7: Return $X'$                                                                                                                                    |                                           |

---

## 4 Experiments

In this section, we evaluate the effectiveness of the GeoSAT framework through comparisons with recent state-of-the-art models, ablation studies, parameter analysis, sensitivity analysis, and visualization-based analysis.

### 4.1 Dataset

In this paper, the performance of GeoSAT is evaluated on eight LTSF benchmark datasets, including the Solar Energy (Lai et al. 2018), Weather, ECL, Traffic, and four ETT datasets (ETTh1, ETTh2, ETTm1, ETTm2) (Hu et al. 2025). Detailed information about the datasets is presented in Table 2.

### 4.2 Baselines

We compare GeoSAT model with 11 well-known MTS forecasting models, including:

- (1) TimeKAN (Huang et al. 2025) (ICLR 2025): A KAN-based model that learns the specific features of different frequency bands using multi-order KAN, enabling effective modeling of patterns across various frequencies.
- (2) TCM (Jiang et al. 2025) (Neurocomputing 2025): A highly efficient and lightweight MLP-based network model that captures sequence and channel dependencies by utilizing Token MLP and Channel MLP.
- (3) TimeMixer (Wang et al. 2024d) (ICLR 2024): An MLP-based model that approaches the complex temporal variations in time series forecasting from a novel multi-scale mixing perspective.
- (4) MSGNet (Cai et al. 2024) (AAAI 2024): A GNN-based model that designs a multi-scale graph convolution method, dynamically learning a dedicated adjacency matrix for each scale.

- (5) TiDE (Das et al. 2023) (arXiv 2023): A MLP-based model that introduces a dense time series encoder, using MLPs for both encoding and decoding.
- (6) DLinear (Zeng et al. 2023) (AAAI 2023): A MLP-based model that demonstrates that a less complex approach can outperform Transformer-based models.
- (7) PatchTST (Nie 2022) (ICLR 2023): A Transformer-based model that splits the time series into sub-sequence-level patches, with each channel representing a univariate time series, operating independently across channels.
- (8) Crossformer (Zhang and Yan 2023) (ICLR 2023): A Transformer-based model that segments the time series along each dimension and captures cross-dimensional dependencies using a hierarchical encoder-decoder architecture.
- (9) FEDformer (Zhou et al. 2022) (ICML 2022): A Transformer-based model that incorporates seasonal-trend decomposition, combining Transformer with frequency analysis to reduce computational costs.
- (10) TimesNet (Wu et al. 2022) (ICLR 2023): A CNN-based model that transforms the data into two-dimensional form via Fourier transform and extracts features through two-dimensional convolutions.
- (11) SCINet (Liu et al. 2022a) (NeurIPS 2022): A CNN-based model that introduces a novel neural network architecture called Sample Convolution and Interaction Network.

### 4.3 Experimental details

The model is trained using the Adam optimizer for up to 10 epochs, with early stopping patience configured to monitor validation loss. For the four ETT datasets, the training, validation, and test sets are split in a 6:2:2 ratio, while the remaining four datasets follow a 7:1:2 split. All datasets use a sequence length of 96, with a batch size of 24 for the Traffic dataset and 256 for the others. The wavelet decomposition scale  $s$  is set to 3 by default, corresponding to high-, medium-, and low-frequency signals. The  $k$  value in GeoSparse attention is set to  $L_d/4$ .

Owing to the divergent time–frequency representations in the data, a differentiated wavelet initialization strategy is employed: for hourly high-variance datasets (ETTh1/ETTh2/ECL/Traffic), the db1 (Haar) wavelet is applied to enhance transient feature extraction; for minute-level high-frequency datasets (ETTm1/ETTm2/Weather/Solar-Energy), longer filter groups such as bior3.1, db4, and db8 are used to improve the representation of stationary time series patterns. All experiments are performed in the PyTorch framework and conducted on an NVIDIA 5060 GPU with 8 GB of memory.

**Table 2** Eight LTSF benchmark datasets

| Dataset      | Variables | Timesteps | Frequency |
|--------------|-----------|-----------|-----------|
| ETTh1        | 7         | 69680     | 15 min    |
| ETTh2        | 7         | 69680     | 15 min    |
| ETTh1        | 7         | 17420     | 1 h       |
| ETTh2        | 7         | 17420     | 1 h       |
| Electricity  | 321       | 26304     | 1 h       |
| Traffic      | 862       | 17544     | 1 h       |
| Weather      | 21        | 52696     | 10 min    |
| Solar-Energy | 137       | 52179     | 10 min    |

#### 4.4 Comparison of different methods

In this section, we compare GeoSAT with all baseline models. For each datasets, we conduct predictions at four forecasting horizons {96, 192, 336, 720}, with the look-back window size fixed at 96. As shown in Table 3, GeoSAT demonstrates robust performance across different benchmarks. On the ETT datasets, our model achieves the best overall performance, obtaining optimal or sub-optimal MSE/MAE results on 99% of the metrics across the eight datasets. Moreover, GeoSAT consistently achieves the best results under all four forecasting horizons on the Weather and Solar datasets. Notably, on the Solar dataset, the MSE improves by 16.9%, indicating that our model can effectively forecast even in scenarios with a large number of channels. According to the primary metric, GeoSAT significantly outperforms all other models.

We compare the performance of GeoSAT with multiple baseline models across eight datasets by averaging the evaluation metrics MSE and MAE over four forecasting horizons, as shown in Fig. 7. The left panel illustrates the MSE comparison, while the right panel shows the MAE comparison, with a larger enclosed area indicating the better the performance. Obviously, GeoSAT has the largest enclosed area. Overall, GeoSAT achieves the best comprehensive performance among all compared models.

#### 4.5 Comparative analysis

In this section, we compare GeoSAT with three of the most competitive baselines (TimeMixer, TCM and TimeKAN) from several perspectives, including forecasting results, visualization analysis, different look-back window settings, and model performance versus parameter scale.

##### 4.5.1 Forecasting performance comparison

Compared with TimeKAN, our model achieves superior performance across all other datasets and horizons. The only exception occurs on the ETTm1 dataset at the 720-step forecasting horizon, where GeoSAT exhibits a marginal MSE gap of approximately 1%. TimeKAN adopts a decomposition–learning–fusion framework that combines frequency-domain decomposition with the nonlinear fitting capability of KAN. Although it achieves many second-best results, it performs unsatisfactorily on high-dimensional datasets such as Solar, ECL, and Traffic.

Compared with TimeMixer, GeoSAT reduces the MSE by 8.5% on the ETTh2 dataset. On the complex Solar-Energy dataset, GeoSAT achieves the best MSE and MAE, outperforming TimeMixer by 16.9% and 10.7%, respectively. Although TimeMixer adopts a multi-scale approach, it underperforms on high-dimensional datasets because its

mixing mechanism is limited to linear or low-order interactions. The reliance on average pooling and ensemble prediction may cause information loss during scale transitions. In contrast, GeoSAT introduces the GeoAttention mechanism in the frequency domain and combines intra-variable and cross-variable MLPs to process the data, effectively capturing complex high-order relationships both within and between variables.

Compared with TCM, GeoSAT reduces the MSE by 5.3% and the MAE by 1.8% on the ETTh1 dataset. On the ETTh2 dataset, GeoSAT outperforms TCM by 8.3% in MSE and 4.2% in MAE, demonstrating its strong ability to capture complex temporal patterns in hourly electricity data. Although TCM adopts a dual-MLP architecture with affine transformation to reduce computational complexity, it still underperforms on datasets with intricate spatial–temporal dependencies like ETTh1 and ETTh2. The reliance on simple MLP-based mixing may limit its ability to model high-order interactions within and across variables.

##### 4.5.2 Visualization of forecasting comparison

To provide the visualization of forecast cases, we select the ETTm2 and Solar datasets and compare our model with TimeKAN, TCM, and TimeMixer. Figures 8, 9 illustrate the performance comparison of the four methods under different forecasting horizons. It can be observed that our method achieves superior forecasting performance. It provides more accurate predictions for various extreme values, exhibits the smallest deviation from the ground truth, and captures periodic trends more precisely. Empirical evaluations validate that our model can effectively capture long-term trends and achieve more stable forecasting than others.

##### 4.5.3 Comparison of different look-back windows

Forecasting performance is analyzed under varying look-back window sizes. Using the Weather dataset as an example, we keep all other parameters identical. The selected look-back window sizes are {96, 192, 336, 720}. As shown in Fig. 10, GeoSAT consistently achieves “low MSE” across all forecasting horizons. Moreover, our model attains the best performance under all window sizes. When the forecasting horizons are 336 and 720, the MSE of TCM increases sharply as the look-back window grows. TimeMixer shows larger fluctuations than GeoSAT but smaller than TCM, indicating that its design is more stable than TCM’s but still inferior to GeoSAT in terms of robustness. While delivering inferior performance to GeoSAT, TimeKAN exhibits a similar performance pattern with relatively small fluctuations and minimal sensitivity to the increase in look-back window size, showing a steady downward trend in MSE.

**Table 3** The LTSF results on eight real-world public datasets: red for optimal performance and blue for secondary ranks

| Model Metric          |     | GeoSAT(Ours) |       | TCM(2025) |       | TimeKAN(2025) |       | TimeMixer(2024) |       | MSGNet(2024) |       | PatchTST(2023) |       |
|-----------------------|-----|--------------|-------|-----------|-------|---------------|-------|-----------------|-------|--------------|-------|----------------|-------|
|                       |     | MSE          | MAE   | MSE       | MAE   | MSE           | MAE   | MSE             | MAE   | MSE          | MAE   | MSE            | MAE   |
| ETTh1                 | 96  | 0.322        | 0.360 | 0.317     | 0.361 | 0.324         | 0.363 | 0.328           | 0.363 | 0.326        | 0.366 | 0.352          | 0.374 |
|                       | 192 | 0.361        | 0.381 | 0.366     | 0.385 | 0.363         | 0.387 | 0.364           | 0.384 | 0.376        | 0.397 | 0.390          | 0.393 |
|                       | 336 | 0.391        | 0.404 | 0.396     | 0.404 | 0.391         | 0.405 | 0.391           | 0.406 | 0.417        | 0.422 | 0.421          | 0.414 |
|                       | 720 | 0.462        | 0.443 | 0.471     | 0.446 | 0.457         | 0.444 | 0.458           | 0.445 | 0.481        | 0.458 | 0.462          | 0.449 |
|                       | Avg | 0.384        | 0.397 | 0.388     | 0.399 | 0.384         | 0.400 | 0.385           | 0.400 | 0.400        | 0.411 | 0.406          | 0.408 |
| ETTh2                 | 96  | 0.172        | 0.257 | 0.179     | 0.263 | 0.179         | 0.260 | 0.176           | 0.259 | 0.177        | 0.262 | 0.183          | 0.270 |
|                       | 192 | 0.240        | 0.300 | 0.248     | 0.306 | 0.241         | 0.302 | 0.242           | 0.303 | 0.247        | 0.307 | 0.255          | 0.314 |
|                       | 336 | 0.299        | 0.340 | 0.311     | 0.346 | 0.306         | 0.344 | 0.304           | 0.342 | 0.312        | 0.346 | 0.305          | 0.343 |
|                       | 720 | 0.397        | 0.397 | 0.414     | 0.405 | 0.405         | 0.408 | 0.395           | 0.398 | 0.414        | 0.403 | 0.412          | 0.404 |
|                       | Avg | 0.277        | 0.324 | 0.288     | 0.330 | 0.283         | 0.329 | 0.279           | 0.326 | 0.288        | 0.330 | 0.289          | 0.333 |
| ETTh1                 | 96  | 0.372        | 0.396 | 0.380     | 0.398 | 0.375         | 0.398 | 0.381           | 0.401 | 0.390        | 0.411 | 0.460          | 0.447 |
|                       | 192 | 0.426        | 0.425 | 0.446     | 0.434 | 0.422         | 0.427 | 0.440           | 0.433 | 0.442        | 0.442 | 0.512          | 0.477 |
|                       | 336 | 0.438        | 0.439 | 0.493     | 0.456 | 0.447         | 0.439 | 0.501           | 0.462 | 0.480        | 0.468 | 0.546          | 0.496 |
|                       | 720 | 0.483        | 0.471 | 0.498     | 0.477 | 0.483         | 0.482 | 0.501           | 0.482 | 0.494        | 0.488 | 0.544          | 0.517 |
|                       | Avg | 0.430        | 0.433 | 0.454     | 0.441 | 0.432         | 0.437 | 0.456           | 0.445 | 0.452        | 0.452 | 0.516          | 0.484 |
| ETTh2                 | 96  | 0.273        | 0.339 | 0.312     | 0.355 | 0.296         | 0.348 | 0.292           | 0.343 | 0.328        | 0.371 | 0.308          | 0.355 |
|                       | 192 | 0.365        | 0.390 | 0.390     | 0.404 | 0.374         | 0.391 | 0.374           | 0.395 | 0.402        | 0.414 | 0.393          | 0.405 |
|                       | 336 | 0.363        | 0.401 | 0.414     | 0.426 | 0.423         | 0.435 | 0.428           | 0.443 | 0.435        | 0.443 | 0.427          | 0.436 |
|                       | 720 | 0.413        | 0.432 | 0.427     | 0.445 | 0.443         | 0.449 | 0.454           | 0.458 | 0.417        | 0.441 | 0.436          | 0.450 |
|                       | Avg | 0.354        | 0.391 | 0.386     | 0.408 | 0.384         | 0.406 | 0.387           | 0.407 | 0.396        | 0.417 | 0.391          | 0.412 |
| ECL                   | 96  | 0.149        | 0.242 | 0.155     | 0.259 | 0.175         | 0.266 | 0.153           | 0.244 | 0.165        | 0.274 | 0.181          | 0.270 |
|                       | 192 | 0.158        | 0.251 | 0.173     | 0.274 | 0.184         | 0.274 | 0.166           | 0.256 | 0.184        | 0.292 | 0.188          | 0.274 |
|                       | 336 | 0.178        | 0.270 | 0.189     | 0.292 | 0.197         | 0.287 | 0.184           | 0.275 | 0.195        | 0.302 | 0.204          | 0.293 |
|                       | 720 | 0.215        | 0.304 | 0.216     | 0.311 | 0.241         | 0.322 | 0.226           | 0.313 | 0.231        | 0.332 | 0.246          | 0.324 |
|                       | Avg | 0.175        | 0.267 | 0.183     | 0.284 | 0.199         | 0.287 | 0.182           | 0.272 | 0.194        | 0.300 | 0.205          | 0.290 |
| Weather               | 96  | 0.156        | 0.203 | 0.156     | 0.204 | 0.164         | 0.208 | 0.165           | 0.212 | 0.163        | 0.212 | 0.177          | 0.218 |
|                       | 192 | 0.204        | 0.247 | 0.208     | 0.254 | 0.209         | 0.252 | 0.209           | 0.253 | 0.212        | 0.254 | 0.225          | 0.259 |
|                       | 336 | 0.262        | 0.288 | 0.263     | 0.292 | 0.264         | 0.293 | 0.264           | 0.293 | 0.272        | 0.299 | 0.278          | 0.297 |
|                       | 720 | 0.340        | 0.341 | 0.346     | 0.348 | 0.347         | 0.344 | 0.342           | 0.345 | 0.350        | 0.348 | 0.354          | 0.348 |
|                       | Avg | 0.241        | 0.270 | 0.243     | 0.275 | 0.246         | 0.274 | 0.245           | 0.276 | 0.249        | 0.278 | 0.259          | 0.281 |
| Solar Energy          | 96  | 0.179        | 0.251 | 0.237     | 0.295 | 0.242         | 0.300 | 0.215           | 0.294 | 0.227        | 0.282 | 0.234          | 0.286 |
|                       | 192 | 0.191        | 0.252 | 0.256     | 0.318 | 0.284         | 0.324 | 0.237           | 0.275 | 0.283        | 0.307 | 0.267          | 0.310 |
|                       | 336 | 0.214        | 0.276 | 0.288     | 0.323 | 0.306         | 0.333 | 0.252           | 0.298 | 0.296        | 0.330 | 0.290          | 0.315 |
|                       | 720 | 0.203        | 0.256 | 0.294     | 0.329 | 0.304         | 0.327 | 0.244           | 0.293 | 0.300        | 0.333 | 0.289          | 0.317 |
|                       | Avg | 0.197        | 0.259 | 0.269     | 0.316 | 0.284         | 0.321 | 0.237           | 0.290 | 0.277        | 0.313 | 0.270          | 0.307 |
| Traffic               | 96  | 0.463        | 0.297 | 0.508     | 0.342 | 0.489         | 0.342 | 0.468           | 0.293 | 0.561        | 0.322 | 0.526          | 0.347 |
|                       | 192 | 0.471        | 0.300 | 0.609     | 0.387 | 0.502         | 0.366 | 0.477           | 0.302 | 0.573        | 0.334 | 0.522          | 0.332 |
|                       | 336 | 0.483        | 0.304 | 0.640     | 0.402 | 0.516         | 0.374 | 0.500           | 0.305 | 0.588        | 0.346 | 0.517          | 0.334 |
|                       | 720 | 0.506        | 0.317 | 0.715     | 0.442 | 0.539         | 0.382 | 0.509           | 0.321 | 0.601        | 0.358 | 0.552          | 0.352 |
|                       | Avg | 0.481        | 0.305 | 0.618     | 0.393 | 0.512         | 0.366 | 0.489           | 0.305 | 0.581        | 0.340 | 0.529          | 0.341 |
| 1 <sup>st</sup> Count |     | 36           | 39    | 1         | 0     | 2             | 0     | 1               | 1     | 0            | 0     | 0              | 0     |

**Table 3** (continued)

| Model Metric          |     | Cross-former(2023) |       | TiDE(2023) |       | TimesNet(2023) |       | DLinear(2023) |       | SCINet(2022) |       | FED-former(2022) |       |
|-----------------------|-----|--------------------|-------|------------|-------|----------------|-------|---------------|-------|--------------|-------|------------------|-------|
|                       |     | MSE                | MAE   | MSE        | MAE   | MSE            | MAE   | MSE           | MAE   | MSE          | MAE   | MSE              | MAE   |
| ETTm1                 | 96  | 0.404              | 0.426 | 0.364      | 0.387 | 0.338          | 0.375 | 0.345         | 0.372 | 0.418        | 0.438 | 0.379            | 0.419 |
|                       | 192 | 0.450              | 0.451 | 0.398      | 0.404 | 0.374          | 0.387 | 0.380         | 0.389 | 0.439        | 0.450 | 0.426            | 0.441 |
|                       | 336 | 0.532              | 0.515 | 0.428      | 0.425 | 0.410          | 0.411 | 0.413         | 0.413 | 0.490        | 0.485 | 0.445            | 0.459 |
|                       | 720 | 0.666              | 0.589 | 0.487      | 0.461 | 0.478          | 0.450 | 0.474         | 0.453 | 0.595        | 0.550 | 0.543            | 0.493 |
|                       | Avg | 0.513              | 0.495 | 0.419      | 0.419 | 0.400          | 0.406 | 0.403         | 0.407 | 0.486        | 0.481 | 0.448            | 0.453 |
| ETTm2                 | 96  | 0.287              | 0.366 | 0.207      | 0.305 | 0.187          | 0.267 | 0.193         | 0.292 | 0.286        | 0.377 | 0.203            | 0.287 |
|                       | 192 | 0.414              | 0.492 | 0.290      | 0.364 | 0.249          | 0.309 | 0.284         | 0.362 | 0.399        | 0.445 | 0.269            | 0.328 |
|                       | 336 | 0.597              | 0.542 | 0.377      | 0.422 | 0.321          | 0.351 | 0.369         | 0.427 | 0.637        | 0.591 | 0.325            | 0.366 |
|                       | 720 | 1.104              | 0.763 | 0.558      | 0.524 | 0.408          | 0.403 | 0.554         | 0.522 | 0.960        | 0.735 | 0.421            | 0.415 |
|                       | Avg | 0.757              | 0.611 | 0.358      | 0.404 | 0.291          | 0.333 | 0.350         | 0.401 | 0.571        | 0.537 | 0.305            | 0.349 |
| ETTh1                 | 96  | 0.423              | 0.448 | 0.479      | 0.464 | 0.384          | 0.402 | 0.386         | 0.400 | 0.654        | 0.599 | 0.376            | 0.419 |
|                       | 192 | 0.471              | 0.474 | 0.525      | 0.492 | 0.436          | 0.429 | 0.437         | 0.432 | 0.719        | 0.631 | 0.469            | 0.470 |
|                       | 336 | 0.570              | 0.546 | 0.565      | 0.515 | 0.491          | 0.469 | 0.481         | 0.459 | 0.778        | 0.659 | 0.459            | 0.465 |
|                       | 720 | 0.653              | 0.621 | 0.594      | 0.558 | 0.521          | 0.500 | 0.519         | 0.516 | 0.836        | 0.699 | 0.506            | 0.507 |
|                       | Avg | 0.529              | 0.522 | 0.541      | 0.507 | 0.458          | 0.450 | 0.456         | 0.452 | 0.747        | 0.647 | 0.453            | 0.465 |
| ETTh2                 | 96  | 0.745              | 0.584 | 0.400      | 0.400 | 0.340          | 0.374 | 0.333         | 0.387 | 0.707        | 0.620 | 0.358            | 0.397 |
|                       | 192 | 0.877              | 0.656 | 0.528      | 0.509 | 0.402          | 0.414 | 0.477         | 0.476 | 0.860        | 0.689 | 0.429            | 0.439 |
|                       | 336 | 1.043              | 0.731 | 0.643      | 0.571 | 0.452          | 0.452 | 0.594         | 0.541 | 1.000        | 0.744 | 0.496            | 0.487 |
|                       | 720 | 1.104              | 0.763 | 0.874      | 0.679 | 0.462          | 0.468 | 0.831         | 0.657 | 1.249        | 0.838 | 0.463            | 0.474 |
|                       | Avg | 0.942              | 0.684 | 0.611      | 0.540 | 0.414          | 0.427 | 0.559         | 0.515 | 0.954        | 0.723 | 0.437            | 0.449 |
| ECL                   | 96  | 0.219              | 0.314 | 0.237      | 0.329 | 0.168          | 0.272 | 0.197         | 0.282 | 0.247        | 0.345 | 0.193            | 0.308 |
|                       | 192 | 0.231              | 0.322 | 0.236      | 0.330 | 0.184          | 0.289 | 0.196         | 0.285 | 0.257        | 0.355 | 0.201            | 0.315 |
|                       | 336 | 0.246              | 0.337 | 0.249      | 0.344 | 0.198          | 0.300 | 0.209         | 0.301 | 0.269        | 0.369 | 0.214            | 0.329 |
|                       | 720 | 0.280              | 0.363 | 0.284      | 0.373 | 0.220          | 0.320 | 0.245         | 0.333 | 0.299        | 0.390 | 0.246            | 0.355 |
|                       | Avg | 0.244              | 0.334 | 0.252      | 0.344 | 0.193          | 0.295 | 0.212         | 0.300 | 0.268        | 0.365 | 0.214            | 0.327 |
| Weather               | 96  | 0.195              | 0.271 | 0.202      | 0.261 | 0.172          | 0.220 | 0.196         | 0.255 | 0.221        | 0.306 | 0.217            | 0.296 |
|                       | 192 | 0.209              | 0.277 | 0.242      | 0.298 | 0.219          | 0.261 | 0.237         | 0.296 | 0.261        | 0.340 | 0.276            | 0.336 |
|                       | 336 | 0.272              | 0.335 | 0.287      | 0.335 | 0.280          | 0.306 | 0.283         | 0.335 | 0.309        | 0.378 | 0.339            | 0.380 |
|                       | 720 | 0.379              | 0.401 | 0.351      | 0.386 | 0.365          | 0.359 | 0.345         | 0.381 | 0.377        | 0.427 | 0.403            | 0.428 |
|                       | Avg | 0.264              | 0.321 | 0.271      | 0.320 | 0.259          | 0.287 | 0.265         | 0.317 | 0.292        | 0.363 | 0.309            | 0.360 |
| Solar Energy          | 96  | 0.310              | 0.331 | 0.312      | 0.399 | 0.250          | 0.292 | 0.290         | 0.378 | 0.237        | 0.344 | 0.242            | 0.342 |
|                       | 192 | 0.734              | 0.725 | 0.339      | 0.416 | 0.296          | 0.318 | 0.320         | 0.398 | 0.280        | 0.380 | 0.285            | 0.380 |
|                       | 336 | 0.750              | 0.735 | 0.368      | 0.430 | 0.319          | 0.330 | 0.353         | 0.415 | 0.304        | 0.389 | 0.282            | 0.376 |
|                       | 720 | 0.769              | 0.765 | 0.370      | 0.425 | 0.338          | 0.337 | 0.356         | 0.413 | 0.308        | 0.388 | 0.357            | 0.427 |
|                       | Avg | 0.641              | 0.639 | 0.347      | 0.418 | 0.301          | 0.319 | 0.330         | 0.401 | 0.282        | 0.375 | 0.292            | 0.381 |
| Traffic               | 96  | 0.644              | 0.429 | 0.805      | 0.493 | 0.593          | 0.321 | 0.650         | 0.396 | 0.788        | 0.499 | 0.587            | 0.366 |
|                       | 192 | 0.665              | 0.431 | 0.756      | 0.474 | 0.617          | 0.336 | 0.598         | 0.370 | 0.789        | 0.505 | 0.604            | 0.373 |
|                       | 336 | 0.674              | 0.420 | 0.762      | 0.477 | 0.629          | 0.336 | 0.605         | 0.373 | 0.797        | 0.508 | 0.621            | 0.383 |
|                       | 720 | 0.683              | 0.424 | 0.719      | 0.449 | 0.640          | 0.350 | 0.645         | 0.394 | 0.841        | 0.523 | 0.626            | 0.382 |
|                       | Avg | 0.667              | 0.426 | 0.761      | 0.473 | 0.620          | 0.336 | 0.625         | 0.383 | 0.804        | 0.509 | 0.610            | 0.376 |
| 1 <sup>st</sup> Count |     | 0                  | 0     | 0          | 0     | 0              | 0     | 0             | 0     | 0            | 0     | 0                | 0     |

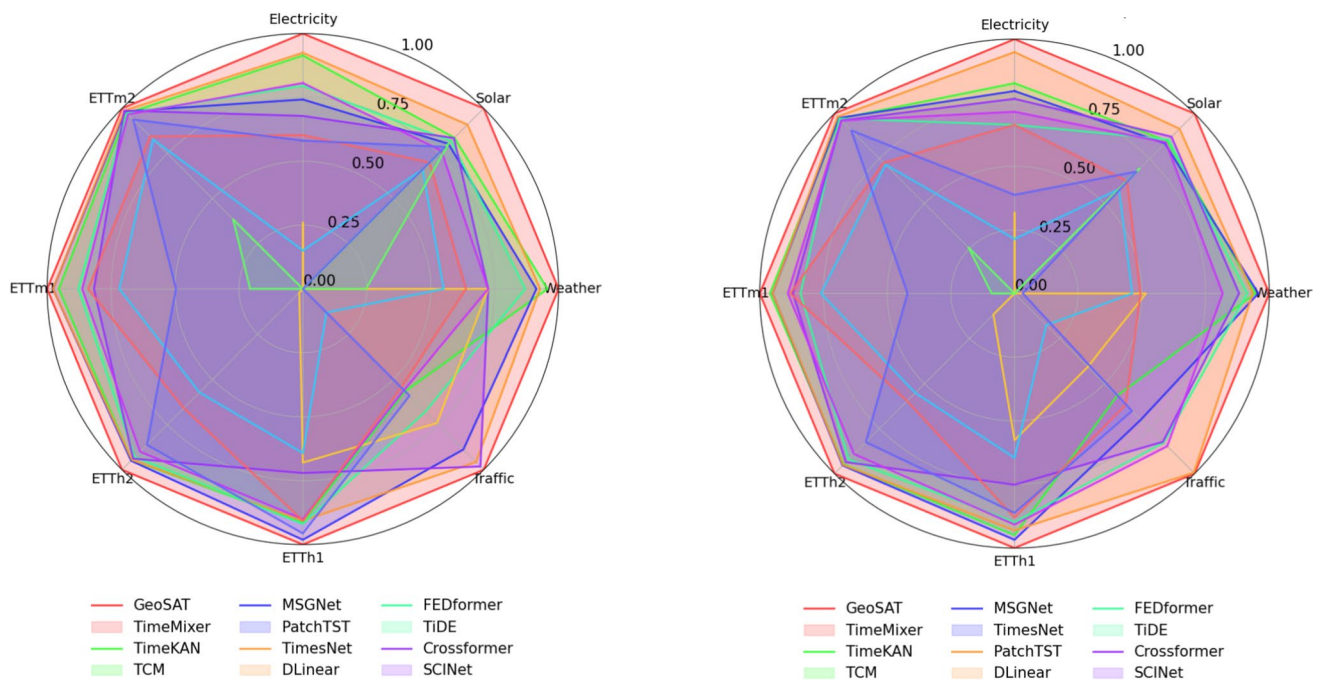


Fig. 7 Model performance comparison chart

#### 4.5.4 Efficiency comparison

To verify the lightweight design of our model, we compare GeoSAT with TimeKAN, TimeMixer, and TCM in terms of total parameters and GPU memory. For fairness, the prediction length and input sequence length are set to 96. As shown in Table 4, GeoSAT consistently achieves the lowest MSE across all datasets, outperforming other models and demonstrating its superior prediction accuracy.

The number of parameters in GeoSAT is much lower than that of TimeMixer, TCM and TimeKAN. For example, on the Weather dataset, GeoSAT requires only 54,000 parameters, while TimeMixer exceeds 100,000, and TCM even reaches over 300,000. On the Solar dataset, GeoSAT has about 245,000 parameters, TimeMixer has around 600,000, and TCM has as many as 800,000. Although the number of parameters in TimeKAN is similar to that of GeoSAT, it is still slightly higher, indicating that GeoSAT's network structure is more efficient and significantly reduces redundant parameters. In terms of GPU memory usage, GeoSAT consistently uses less memory than TimeKAN, TimeMixer, and TCM. For the Weather dataset, GeoSAT occupies only 1576 MB, whereas TimeKAN requires 2 GB, TimeMixer nearly 6 GB, and TCM around 1.8 GB. On the Solar dataset, GeoSAT also shows a clear advantage. These results suggest that GeoSAT has higher hardware efficiency in practical operation and lower deployment costs.

In terms of GPU memory usage, TCM consistently outperforms GeoSAT. For example, in the Weather dataset, TCM uses 1,832 MB of memory, while GeoSAT only uses 1,576 MB. In the Solar dataset, the gap is even more significant, with TCM occupying 3,869 MB and GeoSAT only 1,472 MB.

In terms of training time, GeoSAT is generally faster than TCM. For example, in the ETTh1 dataset, GeoSAT requires only 3.6 s per training epoch, while TCM requires 30.3 s; in the Solar dataset, GeoSAT takes 17.3 s, while TCM takes 52.3 s. This demonstrates that while TCM's dual-branch parallel design benefits feature extraction, it also brings higher computational overhead. Overall, its parameter count and resource consumption are significantly higher than GeoSAT's.

In terms of inference times, GeoSAT also achieves the best performance, obtaining the smallest inference time across all datasets. For example, On the ETTh1 dataset, GeoSAT requires only 0.0912 s, while TCM takes 0.8596 s. On the ETTm2 dataset, GeoSAT achieves an inference time of 0.0948 s, while TimeMixer requires 1.0370 s. The above results confirm the lightweight and efficient nature of the GeoSAT architecture.

#### 4.6 Ablation study

We conduct an ablation study to evaluate the contributions of four key components: the Adaptive Multi-Scale Sequence

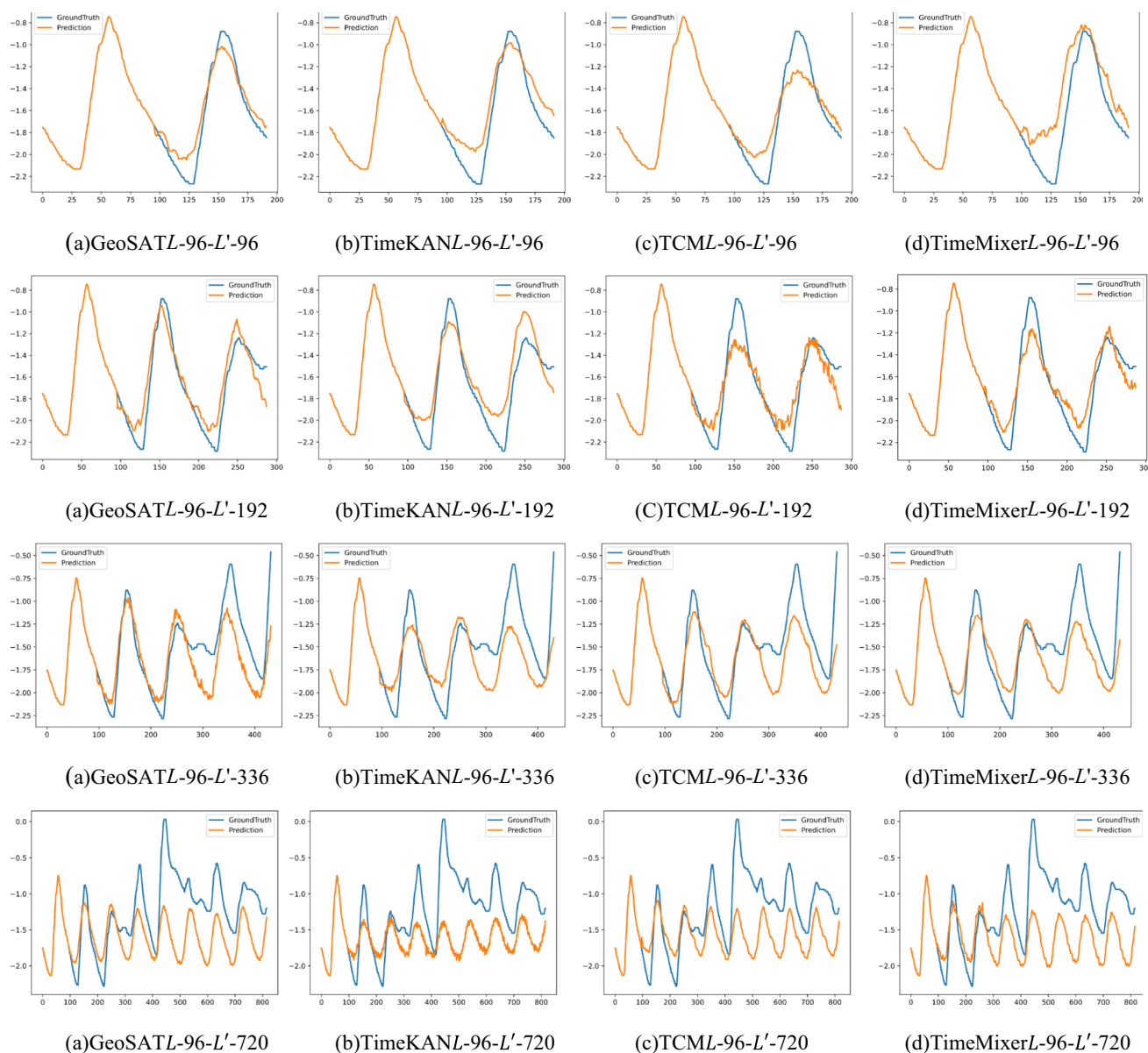


**Fig. 8** Prediction examples of different models on the ETTm2

Decomposition, the GeoSparse Attention module, the time-domain adaptive frequency filtering module and the spatial-domain one-dimensional convolution module. In the decomposition module ablation experiment, we use the standard seasonal trend decomposition method as a comparative baseline for the experiment. Table 5 summarizes the results across eight datasets, with the metrics averaged over four forecasting horizons. The findings consistently demonstrate that all three modules contribute positively to improving forecasting performance.

After removing the AMSD module (w/o AMSD), the performance on all datasets declines, with the MSE on the

Solar Energy dataset increasing by 8.6%. This indicates that proper feature decomposition is crucial for modeling long-term trends. Disabling the GeoSparse Attention module (w/o GeoSparse Attention) leads to a 10.8% worsening of the MSE on the Solar dataset, confirming the module's advantage in capturing local changes. Removing the Adaptive Frequency Filtering module (w/o Adaptive Frequency Filtering) results in a significant 6.2% increase in the MSE on the Traffic dataset, while removing the Spatial One-Dimensional Convolution module causes the MSE on the Traffic dataset to worsen by 8.7%. This suggests that both of these modules play a critical role in multivariate dataset prediction by

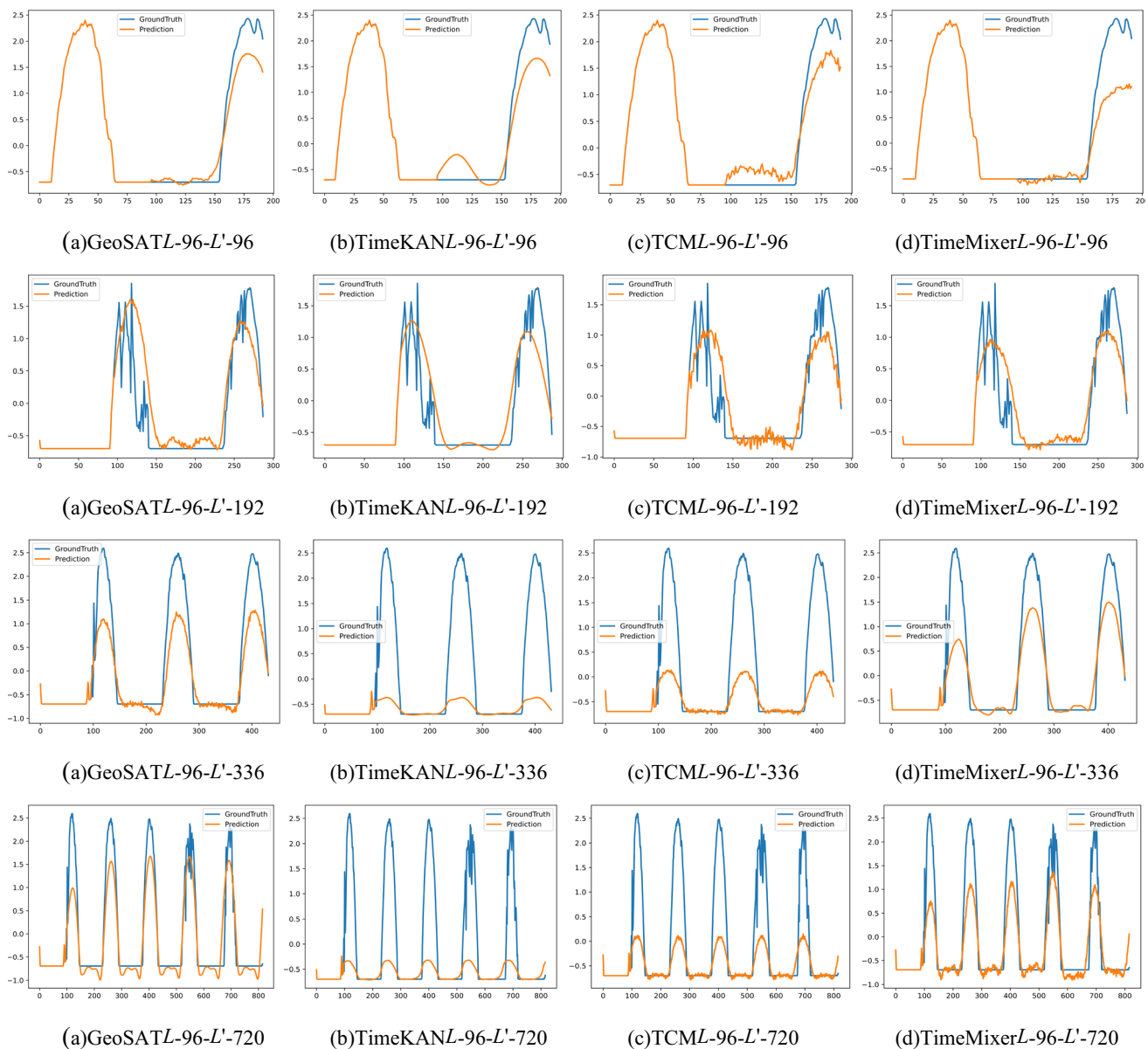


**Fig. 9** Prediction examples of different models on solar-energy

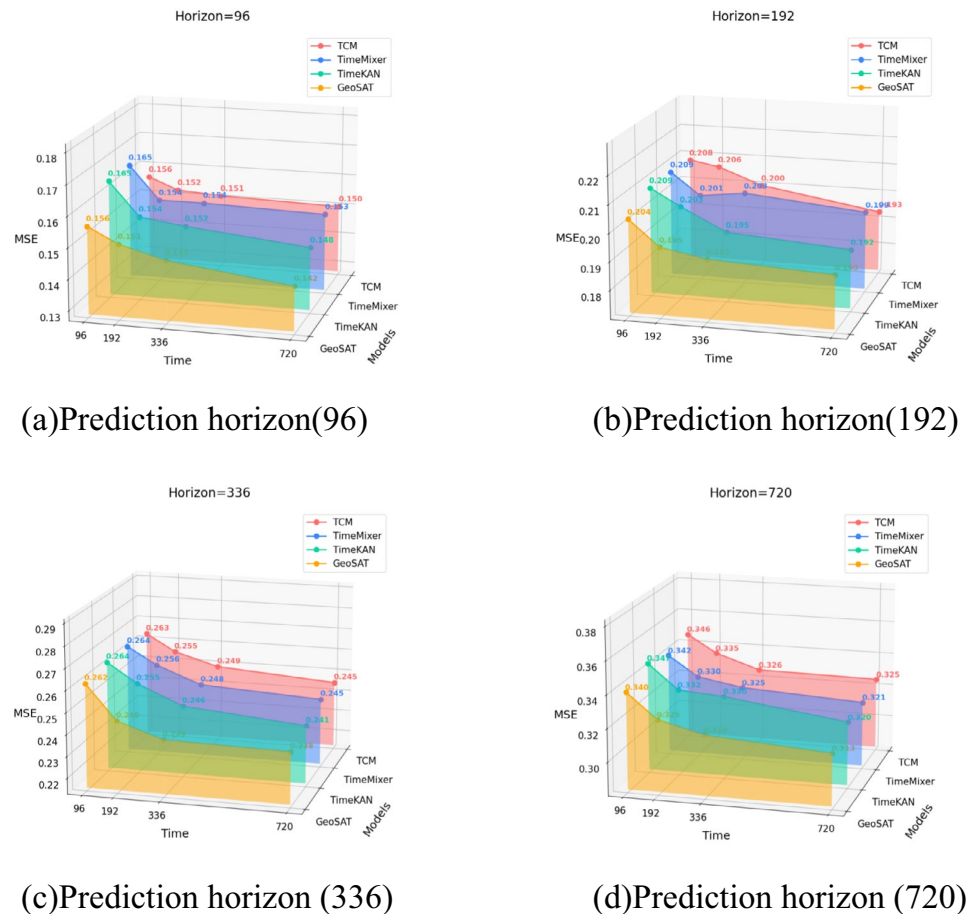
effectively capturing the correlations and long-term trends between variables. Overall, these experiments demonstrate that all four modules in GeoSAT make a positive contribution to improving time series forecasting performance.

#### 4.7 Parameter sensitivity analysis

We conduct a parameter sensitivity analysis focusing on three key hyperparameters: the wavelet decomposition scale  $s$ , learning rate, and GeoSparse Attention weight  $\lambda$ . We use ETTm1, ETTh2, Weather, and Solar-Energy with a look-back window size of 96 and forecasting horizons of 96 and 192. The detailed results are shown in Fig. 11. For

the decomposition scale  $s$ , we set values in the range  $[1, 5]$ . As the decomposition scale increases, the MSE curves of all four datasets show minor fluctuations but maintain overall stability, and the relative positions between curves are consistent. Except for the Solar dataset, the other datasets achieve the best performance when the decomposition scale is 3. Different datasets exhibit varying performance under different learning rates. For example, ETTh2 achieves its best forecasting performance at a rate of 0.006. Accordingly, we select a learning rate within the interval  $[0.002, 0.01]$  for each dataset in our experiments. When adjusting the GeoSparse Attention weight  $\lambda$ , the MSE curves show minor fluctuations on ETTm1, ETTh2, and Weather. From the above

**Fig. 10** Comparison of different methods under various look-back window sizes



analysis, we assign  $s = 3$  and configure distinct GeoSparse Attention weights for different datasets.

## 4.8 Visualization analysis and interpretability

### 4.8.1 Feature map visualization

To further validate the effectiveness and interpretability of GeoSparse attention in capturing inter-variable correlations, we conduct a feature map visualization experiment. We choose ETTh1 and Weather datasets, comparing the correlations of features before and after the application of the attention mechanism. As shown in Figs. 12 and 13, the correlation weights between most variables change, and the correlations after attention align more closely with the actual relationships. For instance, Fig. 12a shows that in the ETTh1 dataset, the correlation between variable 4 and variable 6 is negative before attention. After incorporating GeoSparse attention, Fig. 12b shows a significant improvement in the correlation. Variable 4 and variable 6 represent MULL and OT (oil temperature), respectively. From a physical perspective, medium-load conditions are the most common in the real world, and unnecessary loads lead to

additional energy losses, which are eventually dissipated as heat, thereby increasing the oil temperature. This implies that an increase in MULL influences the change in OT. Our GeoSparse Attention mechanism captures and amplifies this correlation, thereby increasing the correlation weight between these variables.

In Fig. 14, we focus on the points with the most significant changes in variable correlation for further analysis. The first group of variables consists of temperature (T) and maximum wind speed (max.wv). In a real-world scenario, these two variables are often correlated, as rising temperatures during warm fronts or strong convective events are frequently accompanied by increased wind speeds. The second group consists of wind speed (wv) and potential temperature (Tpot). The Earth's surface absorbs intense solar radiation, leading to the formation of high-Tpot air masses. This induces atmospheric instability, with warm air rising rapidly and being replaced by surrounding air, generating strong wind speeds both vertically and horizontally. The third group of variables is shortwave radiation (SWDR) and air density ( $\rho$ ). Our attention mechanism captures the positive correlation between these variables. In reality, an increase in shortwave radiation raises the atmospheric temperature, which in turn

**Table 4** Comparison of model performance and resource utilization across different datasets

| Dataset | Model     | MSE   | Total Parameters | GPU Memory Footprint(MB) | Training times(s) | Inference Times(s) |
|---------|-----------|-------|------------------|--------------------------|-------------------|--------------------|
| Weather | GeoSAT    | 0.156 | 54,952           | 1,576                    | 30.0              | 0.1826             |
|         | TCM       | 0.156 | 381,834          | 1,832                    | 39.2              | 0.2837             |
|         | TimeKAN   | 0.164 | 56,897           | 2,142                    | 65.6              | 0.2432             |
|         | TimeMixer | 0.165 | 107,777          | 5,911                    | 73.6              | 0.2322             |
| Solar   | GeoSAT    | 0.179 | 24,5978          | 1,472                    | 17.3              | 0.1040             |
|         | TCM       | 0.237 | 842,758          | 3,869                    | 52.3              | 0.2940             |
|         | TimeKAN   | 0.242 | 533,785          | 3,529                    | 136.2             | 0.8342             |
|         | TimeMixer | 0.215 | 592,055          | 5,113                    | 119.2             | 0.3024             |
| ETTh1   | GeoSAT    | 0.372 | 22,598           | 904                      | 3.6               | 0.0912             |
|         | TCM       | 0.382 | 158,708          | 1,284                    | 30.3              | 0.8596             |
|         | TimeKAN   | 0.375 | 23,475           | 1,269                    | 57.3              | 0.0992             |
|         | TimeMixer | 0.381 | 77,769           | 1,433                    | 43.6              | 0.1019             |
| ETTh2   | GeoSAT    | 0.273 | 22,654           | 756                      | 3.3               | 0.0879             |
|         | TCM       | 0.312 | 158,694          | 1,845                    | 30.1              | 0.1337             |
|         | TimeKAN   | 0.296 | 30,873           | 1,225                    | 32.3              | 0.0921             |
|         | TimeMixer | 0.295 | 75,584           | 1,326                    | 41.8              | 0.6578             |
| ETTm1   | GeoSAT    | 0.322 | 22,598           | 936                      | 12.2              | 0.0807             |
|         | TCM       | 0.317 | 158,708          | 1,289                    | 33.5              | 0.2766             |
|         | TimeKAN   | 0.324 | 28,619           | 1,216                    | 36.9              | 0.0901             |
|         | TimeMixer | 0.328 | 75,497           | 1,228                    | 42.3              | 0.1819             |
| ETTm2   | GeoSAT    | 0.172 | 22,654           | 958                      | 13.7              | 0.0948             |
|         | TCM       | 0.179 | 158,708          | 1,864                    | 33.0              | 0.8594             |
|         | TimeKAN   | 0.179 | 38,123           | 1,263                    | 82.3              | 0.0984             |
|         | TimeMixer | 0.176 | 95,433           | 1,264                    | 76.8              | 1.0370             |
| ECL     | GeoSAT    | 0.149 | 1,101,682        | 4,862                    | 55.1              | 0.3732             |
|         | TCM       | 0.155 | 1,818,372        | 5,102                    | 59.9              | 0.9034             |
|         | TimeKAN   | 0.175 | 1,775,657        | 5,964                    | 201.5             | 1.1340             |
|         | TimeMixer | 0.153 | 1,683,681        | 5,885                    | 362.3             | 0.4259             |
| Traffic | GeoSAT    | 0.463 | 1,126,568        | 3,368                    | 46.4              | 0.4127             |
|         | TCM       | 0.508 | 6,042,526        | 3,458                    | 64.8              | 1.0146             |
|         | TimeKAN   | 0.489 | 2,779,985        | 7,728                    | 432.6             | 1.3233             |
|         | TimeMixer | 0.464 | 3,265,225        | 5,133                    | 203.2             | 1.4575             |

The evaluation metrics include MSE, the total number of parameters, GPU memory usage (MB), training times(s) and inference times(s).

**Table 5** Ablation experimental results

| Model                   | ETTh1        |              | ETTm1        |              | ETTh2        |              | ETTm2        |              |
|-------------------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
|                         | MSE          | MAE          | MSE          | MAE          | MSE          | MAE          | MSE          | MAE          |
| GeoSAT                  | <b>0.430</b> | <b>0.433</b> | <b>0.384</b> | <b>0.397</b> | <b>0.354</b> | <b>0.391</b> | <b>0.277</b> | <b>0.324</b> |
| w/o AMSD                | 0.432        | 0.434        | 0.395        | 0.411        | 0.356        | 0.394        | 0.280        | 0.329        |
| w/o GeoSparse Attention | 0.436        | 0.434        | 0.391        | 0.407        | 0.361        | 0.398        | 0.288        | 0.333        |
| w/o Frequency Filtering | 0.434        | 0.438        | 0.388        | 0.408        | 0.366        | 0.402        | 0.298        | 0.330        |
| w/o Spatial Convolution | 0.436        | 0.435        | 0.385        | 0.401        | 0.364        | 0.399        | 0.297        | 0.335        |
| Model                   | Solar Energy |              | Weather      |              | ECL          |              | Traffic      |              |
|                         | MSE          | MAE          | MSE          | MAE          | MSE          | MAE          | MSE          | MAE          |
| GeoSAT                  | <b>0.197</b> | <b>0.259</b> | <b>0.241</b> | <b>0.270</b> | <b>0.175</b> | <b>0.267</b> | <b>0.481</b> | <b>0.305</b> |
| w/o AMSD                | 0.214        | 0.278        | 0.245        | 0.273        | 0.183        | 0.275        | 0.497        | 0.339        |
| w/o GeoSparse Attention | 0.218        | 0.281        | 0.244        | 0.274        | 0.188        | 0.282        | 0.515        | 0.352        |
| w/o Frequency Filtering | 0.208        | 0.275        | 0.244        | 0.273        | 0.185        | 0.276        | 0.511        | 0.347        |
| w/o Spatial Convolution | 0.222        | 0.292        | 0.245        | 0.275        | 0.187        | 0.279        | 0.523        | 0.355        |

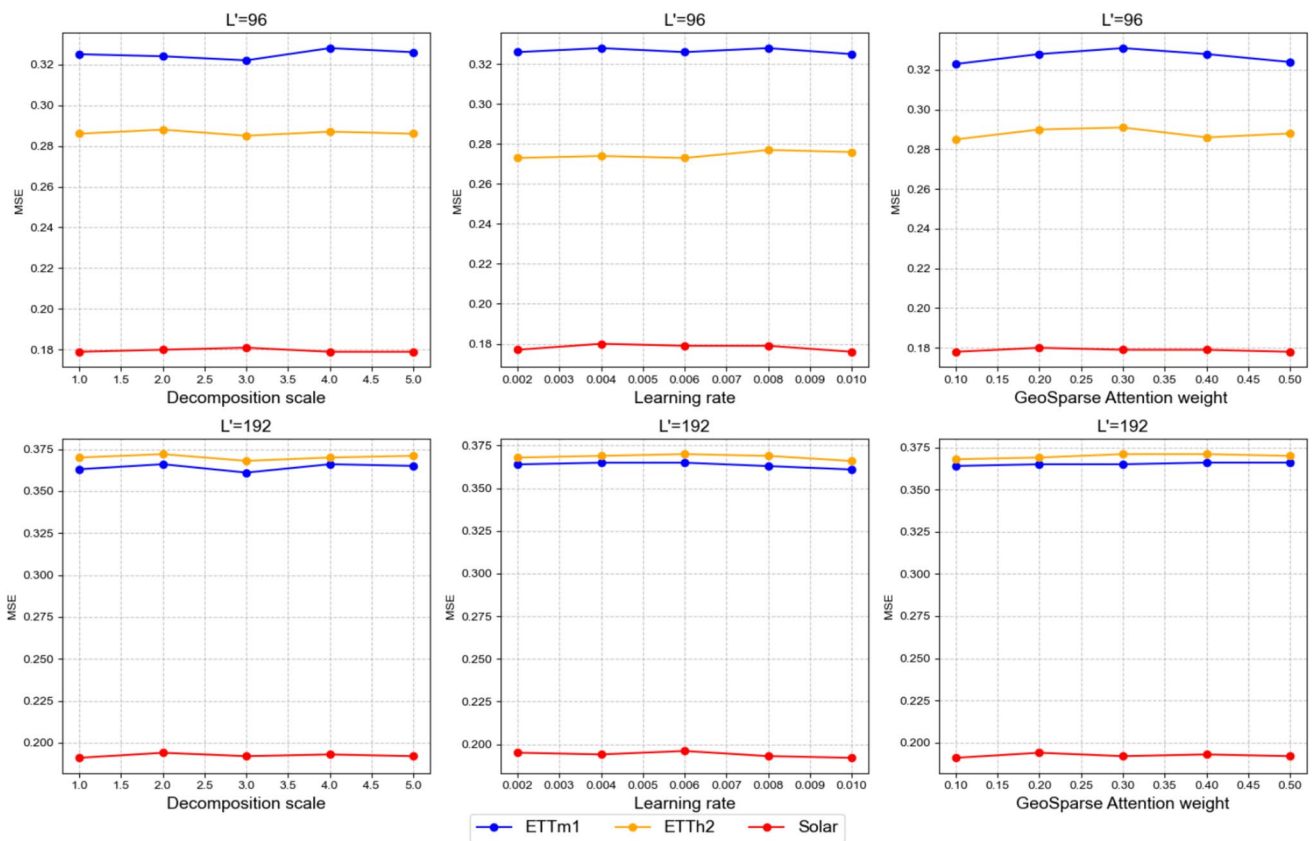


**Fig. 11** Sensitivity analysis of the hyperparameters

**Fig. 12** Feature correlation heatmap on ETTh1

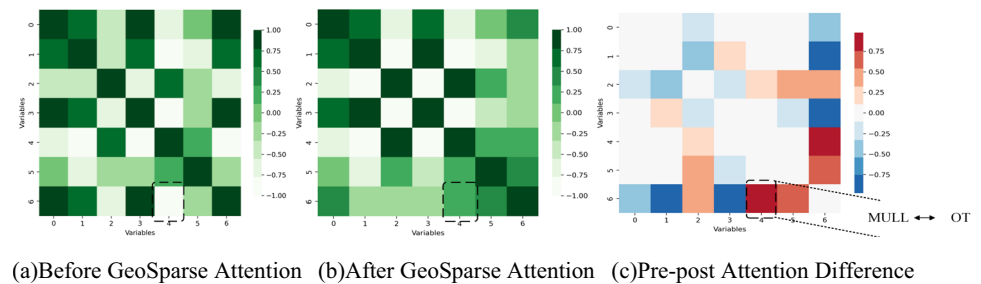
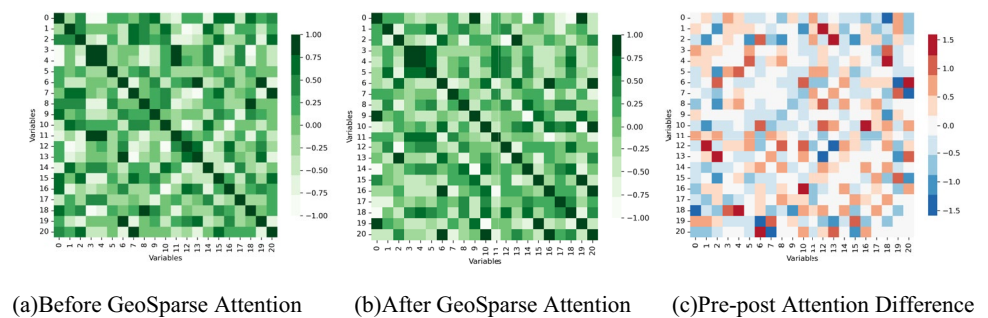


**Fig. 13** Feature correlation heatmap on weather



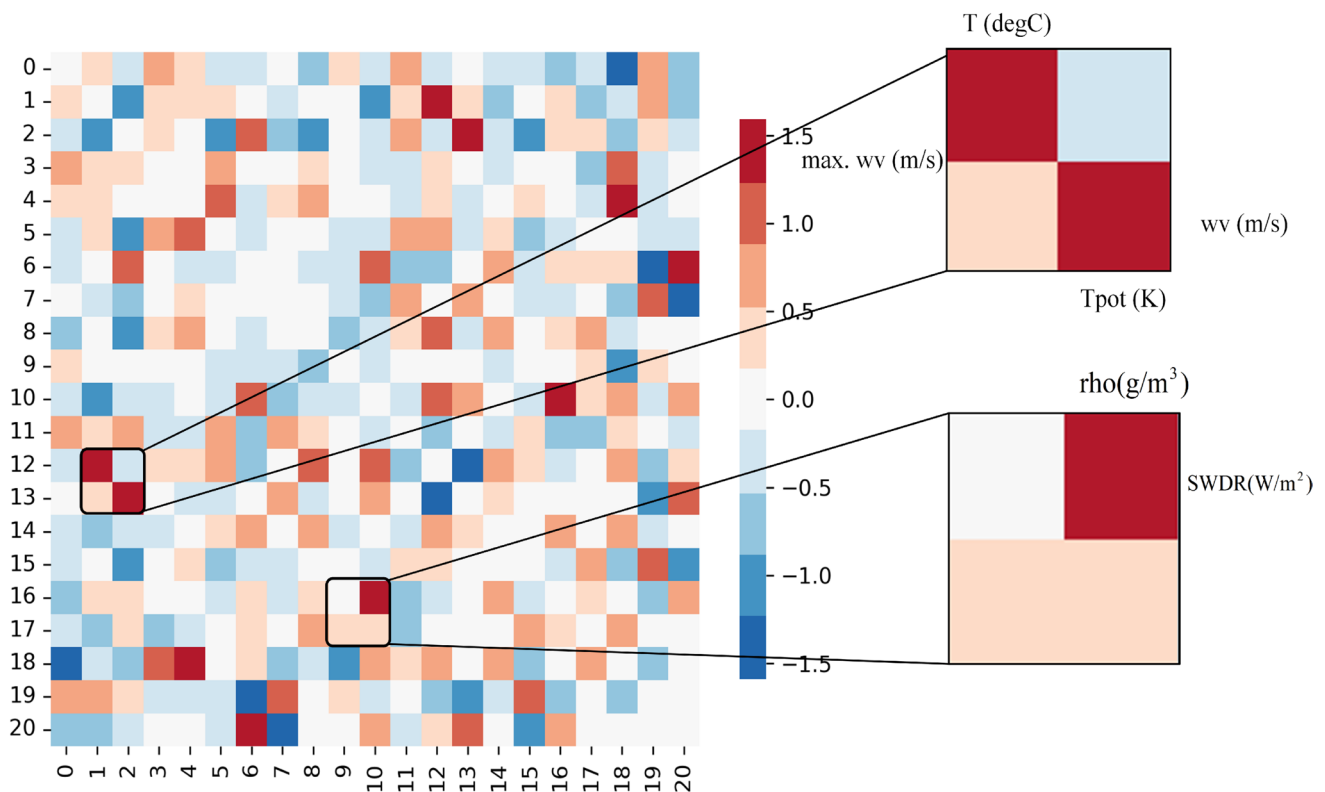


Fig. 14 Examples of feature correlation analysis on weather

increases air density, and this correlation is also learned by the attention mechanism. Overall, we can conclude that GeoSAT is capable of capturing and amplifying the correlations between variables, thereby enhancing the correlation weights, which can then be applied to time-series forecasting.

#### 4.8.2 Sparse visualization

This section presents a sparse visualization analysis of GeoSparse Attention. Specifically, we conduct a visual comparison of the weight matrix before and after sparsification. The experiments are performed on the ETTh1 and Weather, as shown in Fig. 15. Panels (a) and (b) illustrate the comparison of the ETTh1 dataset before and after sparsification, while panels (c) and (d) show the corresponding results for the Weather dataset. As shown in Fig. 15, the weight distribution before sparsification is relatively scattered. After sparsification, lower weights are assigned to less important parts of the weight matrix, effectively suppressing their influence, while higher weights are preserved for more significant components.

This approach allows us to focus on key features while ignoring less influential points. On one hand, this reduces the computational load in subsequent steps, and highlights the critical information, making the computation more efficient and streamlined.

To investigate the impact of Top-k sparsity on prediction accuracy, we fix all other hyperparameters and vary only the Top-k ratio in the attention mechanism. The prediction errors (MSE/MAE) are evaluated on the ETTh1 and Weather datasets, and the experimental results are shown in Table 6. The model achieves the lowest error when the sparsity level is 0.25. When no sparsification is applied, the MSE on the ETTh1 dataset deteriorates by 1.6% and the MAE by 1.0%; on the Weather dataset, the MSE and MAE deteriorate by 1.9% and 2.0%, respectively. The prediction accuracy at Top-k ratios of 0.5 and 0.75 is lower than that at 0.25. This indicates that sparsification not only reduces the computational cost of the model but also enhances the key attention weights while suppressing the less important ones, thereby mitigating the influence of irrelevant weights on the prediction target.

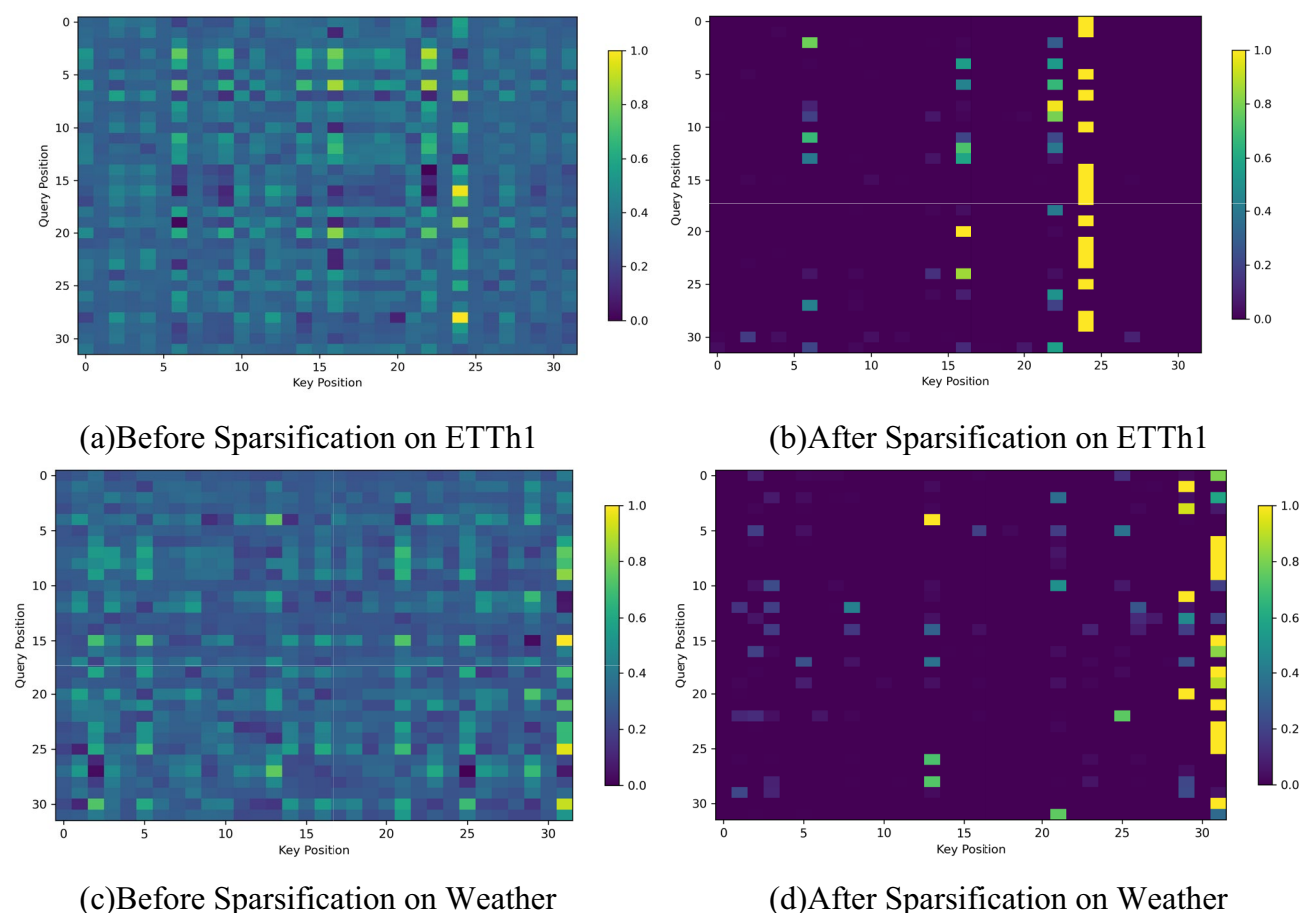


**Fig. 15** Examples of sparse visualization for GeoSparse attention

**Table 6** The impact of Top-k ratio on prediction results

| Dataset | Top-k ratio | MSE   | MAE   |
|---------|-------------|-------|-------|
| ETTh1   | 0.25        | 0.372 | 0.396 |
|         | 0.50        | 0.375 | 0.399 |
|         | 0.75        | 0.377 | 0.400 |
|         | 1.00        | 0.378 | 0.400 |
| Weather | 0.25        | 0.156 | 0.203 |
|         | 0.50        | 0.158 | 0.206 |
|         | 0.75        | 0.158 | 0.207 |
|         | 1.00        | 0.159 | 0.207 |

## 5 Conclusion

This paper explores a novel direction for time series forecasting by integrating SSWT-based GeoSparse Attention with tempo-spatial feature extraction module. Specifically, we decompose the original data into residual and trend components. The residual component is then decomposed into multi-scale features using SSWT, and important

frequency domain information at each scale is extracted via the GeoSparse Attention mechanism. This approach addresses the limitations of traditional Transformer-based methods, which overlook frequency domain information and inter-variable correlations, and are unable to capture local patterns. Moreover, sparsification ensures the computational efficiency. For the trend component, we employ the tempo-spatial feature extraction module to effectively learn the global patterns and cross-variable relationship. Extensive experiments on multiple real-world datasets demonstrate that GeoSAT outperforms existing models in LTSF by effectively modeling complex correlations between variables and global patterns of signals, while simultaneously maintaining a lightweight design and prediction accuracy.

Although GeoSAT shows strong overall performance, different time series datasets may exhibit diverse structural characteristics, and the model may not always achieve optimal results across all scenarios. In future work, we aim to design domain-specific modules, such as multi-scale decomposition, graph structure modeling, and external variable fusion, to enhance the model's generalization

ability in complex real-world scenarios. For example, in applications with strong spatiotemporal dependencies (e.g., traffic prediction, weather analysis), we can introduce dynamic graph neural networks and spatiotemporal attention mechanisms to explicitly model the evolutionary patterns of the system, thereby improving prediction performance.

**Acknowledgements** The work of Lidong Wang was supported by the Joint Fund of Zhejiang Provincial Natural Science Foundation of China under Grant No.LGEY26E090014; The work of Shuya Ke was supported by the National Natural Science Foundation of China (No.12501705).

**Author contribution** Lidong Wang: Writing-review & editing, Writing-original draft, Supervision, Project administration, Methodology, Conceptualization. Tao Chen: Writing-original draft, Visualization, Validation, Software. Xun Li: Investigation, Data curation. Tao Huang: Writing-review & editing. Bin Zhou: Supervision, Project administration, Funding acquisition. Yin Zhang: Visualization, Validation. Shuya Ke: Data curation, Funding acquisition.

**Data availability** We have made our code publicly available on GitHub, <https://github.com/IntelligentComp-Lab/GeoSAT>.

## Declarations

**Competing interests** The authors declare no competing interests.

**Open Access** This article is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License, which permits any non-commercial use, sharing, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if you modified the licensed material. You do not have permission under this licence to share adapted material derived from this article or parts of it. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by-nc-nd/4.0/>.

## References

- Bai S, Kolter JZ, Koltun V (2018) An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. arXiv preprint arXiv:1803.01271
- Bandara K, Hyndman RJ, Bergmeir C (2025) MSTL: a seasonal-trend decomposition algorithm for time series with multiple seasonal patterns. *Int J Oper Res* 52(1):79–98
- Bertozzi AL, Franco E, Mohler G, Short MB, Sledge D (2020) The challenges of modeling and forecasting the spread of COVID-19. *Proc Natl Acad Sci U S A* 117(29):16732–16738
- Cai W, Liang Y, Liu X, Feng J, Wu Y (2024) Msgnet: Learning multi-scale inter-series correlations for multivariate time series forecasting. In: *Proceedings of the AAAI conference on artificial intelligence* 38(10):11141–11149
- Chen Y, Ren K, Wang Y, Fang Y, Sun W, Li D (2023) Contiformer: continuous-time transformer for irregular time series modeling. *Adv Neural Inf Process Syst* 36:47143–47175
- Chowdhury RR, Zhang X, Shang J, Gupta RK, Hong D (2022) Tarnet: Task-aware reconstruction for time-series transformer. In: *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data Mining* pp. 212–220
- Das A, Kong W, Leach A, Mathur S, Sen R, Yu R (2023) Long-term forecasting with tide: Time-series dense encoder. arXiv preprint arXiv:2304.08424
- Ekambaram V, Jati A, Nguyen N, Sinthong P, Kalagnanam J (2023) Tsmixer: Lightweight mlp-mixer model for multivariate time series forecasting. In: *Proceedings of the 29th ACM SIGKDD conference on knowledge discovery and data mining* pp. 459–469
- Hamilton JD (2020) *Time series analysis*. Princeton University Press
- Hittawe MM, Harrou F, Togou MA, Sun Y, Knio O (2024) Time-series weather prediction in the Red Sea using ensemble transformers. *Appl Soft Comput* 164:111926
- Hu Y, Liu P, Zhu P, Cheng D, Dai T (2025) Adaptive multi-scale decomposition framework for time series forecasting. In: *Proceedings of the AAAI conference on artificial intelligence* 39(16):17359–17367
- Huang S, Liu Y, Zhang F, Li Y, Li J, Zhang C (2024) Crosswavenet: a dual-channel network with deep cross-decomposition for long-term time series forecasting. *Expert Syst Appl* 238:121642
- Huang S, Zhao Z, Li C, Bai L (2025) Timekan: Kan-based frequency decomposition learning architecture for long-term time series forecasting. arXiv preprint arXiv:2502.06910
- Jiang H, Liu D, Ding X, Chen Y, Li H (2025) TCM: an efficient lightweight MLP-based network with affine transformation for long-term time series forecasting. *Neurocomputing* 617:128960
- Lai G, Chang WC, Yang Y, Liu H (2018) Modeling long-and short-term temporal patterns with deep neural networks. In: *The 41st international ACM SIGIR conference on research & development in information retrieval* pp. 95–104
- Lee S, Hong J, Liu L, Choi W (2024) Ts-fastformer: fast transformer for time-series forecasting. *ACM Trans Intell Syst Technol* 15(2):1–20
- Liu M, Zeng A, Chen M, Xu Z, Lai Q, Ma L, Xu Q (2022a) Scinet: time series modeling and forecasting with sample convolution and interaction. *Adv Neural Inf Process Syst* 35:5816–5828
- Liu Z, Feng Y, Liu H, Tang R, Yang B, Zhang D, Jia W, Tan J (2025) TVC Former: a transformer-based long-term multivariate time series forecasting method using time-variable coupling correlation graph. *Knowl-Based Syst* 314:113147
- Liu S, Yu H, Liao C, Li J, Lin W, Liu AX, Dustdar S (2022b) Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. In: *# Placeholder\_Parent\_Metadata\_Value#*
- Luo C, Zhan J, Xue X, Wang L, Ren R, Yang Q (2018) Cosine normalization: Using cosine similarity instead of dot product in neural networks. In: *International conference on artificial neural networks*, pp. 382–391. Springer
- Lv Z, Ma Z, Xia F, Li J (2024) A transportation revitalization index prediction model based on spatial-temporal attention mechanism. *Adv Eng Inform* 61:102519
- Masini RP, Medeiros MC, Mendes EF (2023) Machine learning advances for time series forecasting. *J Econ Surv* 37(1):76–111
- Nason GP, Silverman BW (1995) The stationary wavelet transform and some statistical applications. In: *Wavelets and statistics* pp. 281–299
- Nie Y (2022) A time series is worth 64 words: long-term forecasting with transformers. In: *The eleventh international conference on learning representations*

- Nizam H, Zafar S, Lv Z, Wang F, Hu X (2022) Real-time deep anomaly detection framework for multivariate time-series data in industrial IoT. *IEEE Sens J* 22(23):22836–22849
- Qiu X, Zhang L, Suganthan PN, Amaratunga GA (2017) Oblique random forest ensemble via least square estimation for time series forecasting. *Inf Sci* 420:249–262
- Shiblee M, Kalra PK, Chandra B (2008) Time series prediction with multilayer perceptron (MLP): A new generalized error based approach. In: *International conference on neural information processing* (pp. 37–44). Berlin, Heidelberg: Springer Berlin Heidelberg
- Su J, Xie D, Duan Y, Zhou Y, Hu X, Duan S (2024a) MDCNet: long-term time series forecasting with mode decomposition and 2D convolution. *Knowl Based Syst* 299:111986
- Su J, Ahmed M, Lu Y, Pan S, Bo W, Liu Y (2024b) Roformer: enhanced transformer with rotary position embedding. *Neurocomputing* 568:127063
- Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, ... Polosukhin I (2017) Attention is all you need. *Adv Neural Inf Process Syst* 30
- Wang Y, Long H, Zheng L, Shang J (2024a) Graphformer: adaptive graph correlation transformer for multivariate long sequence time series forecasting. *Knowl Based Syst* 285:111321
- Wang Y, Wu H, Dong J, Qin G, Zhang H, Liu Y, Qiu Y, Wang J, Long M (2024b) Timexer: empowering transformers for time series forecasting with exogenous variables. *Adv Neural Inf Process Syst* 37:469–498
- Wang H, Mo Y, Dai H, Yin N, Fan S, Li B (2024c) FTMLP: MLP with feature-temporal block for multivariate time series forecasting. *Neurocomputing* 607:128365
- Wang S, Wu H, Shi X, Hu T, Luo H, Ma L, ... Zhou J (2024) Timemixer: Decomposable multiscale mixing for time series forecasting. In: *The Twelfth International Conference on Learning Representations*
- Waqas M, Humphries UW (2024) A critical review of RNN and LSTM variants in hydrological time series predictions. *MethodsX* 13:102946
- Wen Q, Gao J, Song X, Sun L, Xu H, Zhu S (2019) RobustSTL: A robust seasonal-trend decomposition algorithm for long time series. In: *Proceedings of the AAAI conference on artificial intelligence* 33(01):5409–5416
- Wen Q, Zhou T, Zhang C, Chen W, Ma Z, Yan J, Sun L (2022) Transformers in time series: A survey. *arXiv preprint arXiv:2202.07125*
- Wu H, Xu J, Wang J, Long M (2021) Autoformer: decomposition transformers with auto-correlation for long-term series forecasting. *Adv Neural Inf Process Syst* 34:22419–22430
- Wu H, Hu T, Liu Y, Zhou H, Wang J, Long M (2022) Timesnet: Temporal 2d-variation modeling for general time series analysis. In: *The eleventh international conference on learning representations*
- Xie J, Cui Y, Huang F, Liu C, Zheng K (2022) MARINA: an mlp-attention model for multivariate time-series analysis. In: *Proceedings of the 31st ACM international conference on information & knowledge management* pp. 2230–2239
- Yi K, Zhang Q, Fan W, Wang S, Wang P, He H, ... Niu Z (2023) Frequency-domain MLPs are more effective learners in time series forecasting. *Adv Neural Inf Process Syst* 36:76656–76679
- Yu Y, Ma R, Ma Z (2024) Robformer: a robust decomposition transformer for long-term time series forecasting. *Pattern Recognit* 153:110552
- Zeng A, Chen M, Zhang L, Xu Q (2023) Are transformers effective for time series forecasting?. In: *Proceedings of the AAAI conference on artificial intelligence* 37(9):11121–11128
- Zhang Y, Long M, Chen K, Xing L, Jin R, Jordan MI, Wang J (2023) Skilful nowcasting of extreme precipitation with NowcastNet. *Nature* 619(7970):526–532
- Zhang S, Xu J, Zhang W, Zhang Y, Ni C (2025) You only adapt once: an adaptive transformer for dynamic multivariate time series forecasting across time-varying topologies and multi-patterns. *Inf Sci*. <https://doi.org/10.1016/j.ins.2025.122350>
- Zhang Y, Yan J (2023) Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In: *The eleventh international conference on learning representations*
- Zhong S, Song S, Zhuo W, Li G, Liu Y, Chan SHG (2023) A multi-scale decomposition mlp-mixer for time series analysis. *arXiv preprint arXiv:2310.11959*
- Zhou H, Zhang S, Peng J, Zhang S, Li J, Xiong H, Zhang W (2021) Informer: Beyond efficient transformer for long sequence time-series forecasting. In: *Proceedings of the AAAI conference on artificial intelligence* 35(12):11106–11115
- Zhou T, Ma Z, Wen Q, Wang X, Sun L, Jin R (2022) Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In: *International conference on machine learning* pp. 27268–27286

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.