arXiv:2307.01616v1 [cs.LG] 4 Jul 2023

SageFormer: Series-Aware Graph-Enhanced Transformers for Multivariate Time Series Forecasting

Zhenwei Zhang
Department of Electronic Engineering
Tsinghua University
Beijing, China
zzw20@mails.tsinghua.edu.cn

Xin Wang
Department of Electronic Engineering
Tsinghua University
Beijing, China
wangxin20@mails.tsinghua.edu.cn

Yuantao Gu
Department of Electronic Engineering
Tsinghua University
Beijing, China
gyt@tsinghua.edu.cn

Abstract

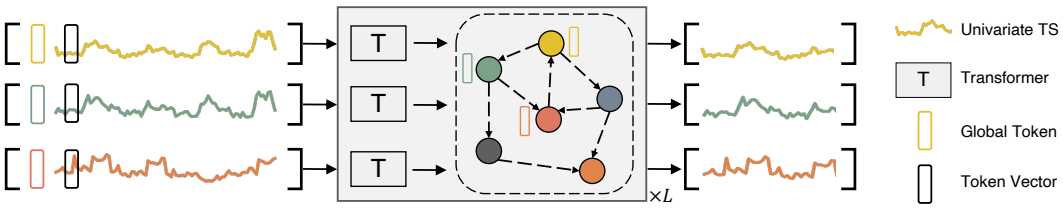
Multivariate time series forecasting plays a critical role in diverse domains. While recent advancements in deep learning methods, especially Transformers, have shown promise, there remains a gap in addressing the significance of inter-series dependencies. This paper introduces SageFormer, a Series-aware Graph-enhanced Transformer model designed to effectively capture and model dependencies between series using graph structures. SageFormer tackles two key challenges: effectively representing diverse temporal patterns across series and mitigating redundant information among series. Importantly, the proposed series-aware framework seamlessly integrates with existing Transformer-based models, augmenting their ability to model inter-series dependencies. Through extensive experiments on real-world and synthetic datasets, we showcase the superior performance of SageFormer compared to previous state-of-the-art approaches.

1 Introduction

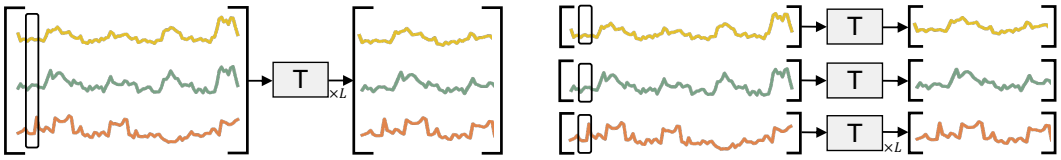
Multivariate Time Series (MTS) data, composed of multiple univariate series, serves as the foundation for MTS forecasting tasks that aim to predict future trends based on a fixed window of historical sequences. The value of MTS forecasting extends beyond academic interest, with essential applications in everyday life and numerous industries. It facilitates resource management and strategic planning across diverse domains such as energy [1], finance [2], and weather [3]. In recent years, deep learning methods [4, 5], particularly Transformer structures [6, 7, 8, 9], have achieved remarkable breakthroughs in MTS forecasting tasks compared to traditional methods (e.g., ARIMA, SSM [10]).

In many Transformer-based studies, the dependencies between different series are often overlooked. These models typically amalgamate different series into hidden temporal embeddings through linear transformations (series-mixing framework, Figure 1b). These models primarily concentrate on temporal dependencies, largely overlooking the relationships between series[11]. Recently, some studies [12, 13] discovered that intentionally neglecting inter-series dependencies modeling (series-independent framework, Figure 1c) could improve prediction results due to its robustness towards distribution drifts [14]. However, the series-independent framework entirely disregards the dependencies between series, resulting in suboptimal results on specific datasets (see section 4.5). These findings highlight the challenges in modeling series dependencies, making accurate capture

Preprint. Under review.



(a) Series-aware framework (Ours)



(b) Series-mixing framework (c) Series-independent framework

Figure 1: Illustration of three different ways of modeling series dependencies: (a) The proposed series-aware framework. Prior to the original input tokens into the Transformer encoder, we incorporate learnable global tokens to capture the intrinsic features of each series. The embedding tokens are processed through multiple SageFormer layers, where temporal encoding and graph aggregation are performed iteratively. (b) The series-mixing framework combines all series at each timestamp into a single token vector. (c) The series-independent framework handles each series separately, improving the learning of unique temporal patterns for different series.

in MTS forecasting tasks a promising research direction. To address this research gap, we propose SageFormer in this paper.

Present work. In this paper, we examine inter-series dependencies in long-term MTS forecasting problems. To accurately model the inter-series dependencies, we propose Series-Aware Graph-Enhanced Transformer (**SageFormer**, Figure 1a), a series-aware Transformer model enhanced with graph neural networks (GNN). By learning relationships between time series using graph structures, we aim to distinguish series using global tokens and improve the modeling ability for diverse temporal patterns across various series through graph aggregation. SageFormer can function as a universal extension for Transformer-based structures, better utilizing the dependencies between series and achieving superior performance without greatly affecting model complexity. We contend that the proposed SageFormer addresses two challenges in inter-series dependencies modeling:

1. How can diverse temporal patterns among series be effectively represented?
   We introduce a series-aware approach that extends the existing series-independent framework by incorporating several global tokens before input tokens. These tokens capture global information for each variable through self-attention and facilitate series interaction via graph aggregation. The addition of global tokens enables SageFormer to learn not only individual series' temporal patterns but also focus on dependencies between series, thereby enhancing diversity and overcoming series-independent limitations (see section 4.5).
2. How can the impact of redundant information across series be avoided?
   We propose using sparsely connected graph structures to reduce the impact of redundant information in unrelated series. In MTS forecasting, not all information is useful due to redundancy in time and series dimensions [15]. To evaluate model effectiveness with sparse data, we designed Low-rank datasets with varying series numbers (see Section 4.5). Our model's performance remains stable as series dimensions increase, utilizing low-rank properties effectively. In contrast, the series-mixing method suffers from prediction deterioration as series dimensions grow.

Our contributions are threefold:

- We introduce a novel series-aware framework that serves as a universal extension for Transformer-based models. It effectively utilizes graph structures to exploit the dependencies between series without noticeably increasing the model's complexity.

- We propose SageFormer, a series-aware Transformer model for long-term MTS forecasting. By integrating GNN, SageFormer efficiently captures inter-series dependencies, transcending the limitations of existing Transformer-based models in modeling these dependencies.
- Experimental results demonstrate that our model attains state-of-the-art performance on both real-world and synthetic datasets.

2 Related Works

Multivariate Time Series Forecasting. MTS forecasting models can generally be categorized into statistical and deep models. Many forecasting methods begin with traditional tools such as the Vector Autoregressive model and the Vector Autoregressive Moving Average [16, 17]. These typical statistical MTS forecasting models assume linear dependencies between series and values. With the advancement of deep learning, various deep models have emerged and often demonstrate superior performance compared to their statistical counterparts. Temporal Convolutional Networks [5, 18] and DeepAR [19] consider MTS data as sequences of vectors, employing CNNs and RNNs to capture temporal dependencies.

Transformers for MTS Forecasting. Recently, Transformer models with self-attention mechanisms have excelled in various fields [20, 21, 22, 23]. Numerous studies aim to enhance Transformers for MTS forecasting by addressing their quadratic complexity. Notable approaches include Informer [7], introducing ProbSparse self-attention and distilling techniques; Autoformer [8], incorporating decomposition and auto-correlation concepts; FEDformer [9], employing a Fourier-enhanced structure; and Pyraformer [24], implementing pyramidal attention modules. PatchTST [13] divides each series into patches and uses a series-independent Transformer to model temporal patterns. While these models primarily focus on reducing temporal dependencies modeling complexity, they often overlook crucial inter-series dependencies.

Inter-series dependencies for MTS Forecasting. Numerous methods have been proposed to explicitly enhance inter-series dependencies in MTS forecasting. LSTnet [4] employs CNN for inter-series dependencies and RNN for temporal dependencies. GNN-based models [25, 26, 27, 28], such as MTGNN [28], utilize temporal and graph convolution layers to address both dependencies. STformer [29] flattens multivariate time series into a 1D sequence for Transformer input, while Crossformer [11] employs dimension-segment-wise embedding and a two-stage attention layer for efficient temporal and inter-series dependencies capture respectively. Most CNN and GNN-based models struggle to capture long-term temporal dependencies. STformer [29] and Crossformer [11] extend 1-D attention to 2-D, but they fail to reveal the relationships between series explicitly. Unlike the methods mentioned above, our proposed SageFormer serves as a general framework that can be applied to various Transformer-based models, utilizing graph structure learning to enhance their ability to capture inter-series dependencies.

3 Methodology

3.1 Problem Definition

In this paper, we concentrate on long-term MTS forecasting tasks. Let $\mathbf{x}_t \in \mathbb{R}^C$ denote the value of C series at time step t. Given a historical MTS instance $\mathbf{X}_t = [\mathbf{x}_t, \mathbf{x}_{t+1}, \cdots, \mathbf{x}_{t+L-1}] \in \mathbb{R}^{C \times L}$ with length L, the objective is to predict the next T steps of MTS values $\mathbf{Y}_t = [\mathbf{x}_{t+L}, \cdots, \mathbf{x}_{t+L+T-1}] \in \mathbb{R}^{C \times T}$. The aim is to learn a mapping $f(\cdot) : \mathbf{X}_t \rightarrow \mathbf{Y}_t$ using the proposed model (we omit the subscript t when it does not cause ambiguity).

We employ graphs to represent inter-series dependencies in MTS and briefly overview relevant graph-related concepts. From a graph perspective, different series in MTS are considered nodes, and relationships among series are described using the graph adjacency matrix. Formally, the MTS data can be viewed as a signal set $\mathcal{G} = \{\mathcal{V}, \mathbf{X}_t, \mathbf{A}\}$. The node set $\mathcal{V}$ contains C series of MTS data and $\mathbf{A} \in \mathbb{R}^{C \times C}$ is a weighted adjacency matrix. The entry a_{ij} indicates the dependencies between series i and j. If they are not dependent, a_{ij} equals zero. The main symbols used in the paper and their meanings are detailed in Table 6 in the Appendix A.

3.2 Overview

SageFormer is designed to augment the capability of Transformer-based models in addressing inter-series dependencies. The overall architecture adheres to a Transformer encoder pipeline, conforming to the series-aware framework. The decoder portion of the Transformer is omitted and replaced with a linear decoder head (FlattenHead in Algorithm 1), proving to be more efficient [12, 13, 30]. SageFormer's encoding workflow, summarized as Algorithm 1, encompasses three key components: (1) series-aware global tokens, (2) graph structure learning, and (3) iterative message passing.

Algorithm 1: SageFormer's Workflow

Input: The input MTS history $\mathbf{X}$
Output: The predicted MTS future $\mathbf{Y}$.

begin
- $\mathcal{X}^{(0)} \leftarrow GlobalEembedding(\mathbf{X})$; `/* series-aware global tokens */`
- $A \leftarrow$ GraphLearning ; `/* graph structure learning */`
- **for** $c = 1, \ldots, C$ **do**
  - $\mathcal{X}^{1}_{:c:} \leftarrow \text{TEB}\left(\mathcal{X}^{0}_{:c:}\right)$;
- **for** $l = 2, \ldots, L-1$ **do** `/* iterative message passing */`
  - **for** $m = 1, ..., M$ **do**
    - $\widehat{\mathcal{X}}^{(l)}_{:m:} \leftarrow \text{GNN}\left(\mathcal{X}^{(l)}_{:m:}, \mathbf{A}\right)$; `/* graph aggregation */`
  - **for** $c = 1, \ldots, C$ **do**
    - $\mathcal{X}^{l+1}_{c::} \leftarrow \text{TEB}\left(\widehat{\mathcal{X}}^{(l)}_{c::}\right)$; `/* temporal encoding */`
- $\hat{\mathcal{X}} \leftarrow \left\{\mathcal{X}^{L}_{m::} | m > M\right\}$
- Return $Y \leftarrow \text{FlattenHead}\left(\hat{\mathcal{X}}\right)$

3.3 Series-aware Global Tokens

Drawing inspiration from the application of the class token in natural language models [21] and Vision Transformer [31], we prepend learnable tokens for each series to encapsulate their corresponding global information. In Section 3.5, we employ these global tokens, rather than all tokens, to capture inter-series dependencies, thereby enhancing the series awareness of each sub-series.

Following PatchTST [13], the input MTS $\mathbf{X} \in \mathbb{R}^{C \times L}$ is reshaped into $\mathcal{X}_p = \{\mathbf{X}_1, \cdots, \mathbf{X}_C\} \in \mathbb{R}^{C \times N \times P}$, where P is the subsequence patch length, C is the number of time series, and $N = \lfloor (L-P)/S \rfloor + 2$ denotes the number of patches, S indicates the non-overlapping length between adjacent patches. $\mathbf{X}_c = \{\mathbf{x}^1_c, \cdots, \mathbf{x}^N_c\} \in \mathbb{R}^{N \times P}$ represents the patched sequence for series c. A consistent latent vector of size D is maintained across Transformer encoding blocks (TEB), with a trainable linear projection ($\mathbf{E} \in \mathbb{R}^{P \times D}$) mapping $\mathcal{X}_p$ to the D-dimensional space (Equation 1).

M learnable embeddings (global tokens) $\mathbf{g}_i \in \mathbb{R}^D$ are added before the patched sequences, representing each series' global information after self-attention, resulting in an effective input sequence length of $M + N$. The prepended global tokens are designed to facilitate interaction across series. Positional information is enhanced via 1D position embeddings $\mathbf{E}_{pos}$. The final embedding of $\mathbf{X}$ is $\mathcal{X}^{(0)} \in \mathbb{R}^{C \times (N+M) \times D}$, where

$$\mathcal{X}^{(0)}_{c::} = \left[\mathbf{g}_1; \cdots; \mathbf{g}_M; \mathbf{x}^1_c\mathbf{E}; \cdots; \mathbf{x}^N_c\mathbf{E}\right] + \mathbf{E}_{pos}. \tag{1}$$

3.4 Graph Structure Learning

The adjacency matrix is learned end-to-end, capturing implicit relationships across series without requiring prior knowledge. In MTS forecasting tasks, we postulate that inter-series dependencies are unidirectional (e.g., power load affects oil temperature, but not vice versa), resulting in a directed relationship represented by the learned graph.

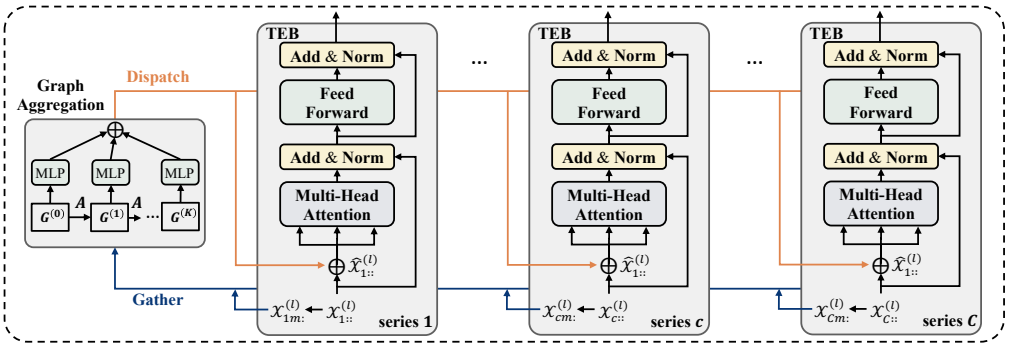


Figure 2: Illustration of the iterative message-passing process in SageFormer. Each layer begins with graph aggregation, where global tokens from all series are gathered and processed by the multi-hop GNN component (leftmost rectangle). Graph-enhanced global tokens are then dispatched to their original series and encoded by TEB. The weights of each TEB are shared among all series.

Specifically, the entire graph structure learning module can be described by the following equations:

$$\mathbf{M}_1 = \mathrm{act}_1(\mathbf{E}\mathbf{\Theta}_1);\ \mathbf{M}_2 = \mathrm{act}_1(\mathbf{E}\mathbf{\Theta}_2) \tag{2}$$

$$\mathbf{A}' = \mathrm{Relu}(\mathbf{M}_1\mathbf{M}_2^T - \mathbf{M}_2\mathbf{M}_1^T) \tag{3}$$

Node embeddings of series are learned through randomly initialized $\mathbf{E} \in \mathbb{R}^{N\times C}$. Subsequently, $\mathbf{E}$ is transformed into $\mathbf{M} \in \mathbb{R}^{N\times C}$ using $\mathbf{\Theta} \in \mathbb{R}^{C\times C}$, with the nonlinear act_1 (Equation 2). Following the approach of MTGNN [28], Equation 3 is employed to learn unidirectional dependencies. For each node, its top k nearest nodes are selected as neighbors, setting the weights of non-connected nodes to zero, yielding the final sparse adjacency matrix $\mathbf{A} \in \mathbb{R}^{C\times C}$.

3.5 Iterative Message Passing

The embedding tokens (outlined in Section 3.3) are processed by SageFormer encoder layers, where temporal encoding and graph aggregation are conducted iteratively (Figure 2). This approach aims to disseminate the global information gathered during the GNN phase among all tokens within each series. As a result, the model captures inter-series dependencies through iterative message passing.

Graph Aggregation The graph aggregation phase aims to fuse each series' information with its neighbors' information, thereby enhancing each series with related patterns. For each series in the l-th layer, we take the first M embeddings as the global tokens of layer l: $\mathbf{G}_i^{(l)} \leftarrow \mathcal{X}_{:i:}^{(l)} \in \mathbb{R}^{C\times D}$, $i \le M$. The global tokens of layer l are gathered from all series and passed into the GNN for graph aggregation. For simplicity, we employ the same model as [25, 28] for graph aggregation:

$$\widehat{\mathbf{G}}_i = \sum_{d=0}^{D} \tilde{\mathbf{A}}^d \mathbf{G}_i \mathbf{W}_d,\ i \le M \tag{4}$$

Equation 4 represents multi-hop information fusion on the graph, where D denotes the depth of graph aggregation and $\tilde{\mathbf{A}}$ is the graph Laplacian matrix. Each of the embeddings $\widehat{\mathbf{G}}_i$ is dispatched to its original series and then concatenated with series tokens, resulting in graph-enhanced embeddings $\widehat{\mathcal{X}}^{(l)}$.

Temporal Encoding The graph-enhanced embeddings can later be processed by any Transformer component (Transformer[20], Informer [7], FEDformer [9], etc.). We choose the vanilla Transformer encoder [20] as our backbone. The output of the TEB functions as input token-level embeddings for the following encoding layer. Previously aggregated information from the GNN is disseminated to other tokens within each series via self-attention, enabling access to related series information. This process enhances the expressiveness of our model compared to series-independent models.

4 Experiments

4.1 Experimental Setup

Datasets. To evaluate our proposed SageFormer, extensive experiments have been conducted on eight mainstream real-world datasets, including Weather, Traffic, Electricity, ILI(Influenza-Like Illness), and four ETT(Electricity Transformer Temperature) datasets. Details of these multivariate datasets are shown in Table 1. Among all datasets, Traffic and Electricity have more series, which can better reflect the effectiveness of the proposed method.

Table 1: The statistics of the eight mainstream datasets.

Datasets	Traffic	Electricity	Weather	ILI	ETTh1/ETTh2	ETTm1/ETTm2
Number of Series	862	321	21	7	7	7
Timesteps	17,544	26,304	52,696	966	17,420	69,680
Temporal Granularity	1 hour	1 hour	10 mins	1 week	1 hour	5 mins

Baselines and Task Settings. We compare our proposed method with nine popular models for long-term MTS forecasting problems as baselines, including three models that explicitly utilize inter-series dependencies: Crossformer[11], MTGNN[28], and LSTnet[4]; two series-independent neural models: DLinear[12] and PatchTST[13]; and four series-mixing transformer-based models: Transformer[20], Informer[7], Autoformer[8], and Non-stationary Transformer[32].

Implementation details. For model training and evaluation, we adopt the same settings as in [30]. The entire dataset is rolled with a stride of 1 to generate different input-output pairs, and Train/Val/Test sets are zero-mean normalized using the mean and standard deviation of the training set. Performance is evaluated over varying future window sizes on each dataset. The past sequence length is set as 36 for ILI and 96 for the others. Mean Square Error (MSE) and Mean Absolute Error (MAE) serve as evaluation metrics. All experiments are conducted five times, and the mean of the metrics is reported. Details regarding datasets, baselines, implementation, and hyper-parameters can be found in Appendix A.

4.2 Main Results

Table 2: Long-term forecasting task. Bold/underline indicates the best/second. Blue background marks the models explicitly utilizing inter-series dependencies; green marks series-independent neural models; yellow marks series-mixing transformer-based models. All the results are averaged from 4 different prediction lengths, that is $\{24, 36, 48, 60\}$ for ILI and $\{96, 192, 336, 720\}$ for the others. See Table 8 in Appendix A for the full results.

Models	**SageFormer (Ours)**		Crossformer [11]		MTGNN [28]		LSTnet [4]		PatchTST [13]		DLinear [12]		Stationary [32]		Autoformer [8]		Informer [7]		Transformer [20]	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Traffic	**0.436**	**0.285**	0.570	0.312	0.592	0.317	0.736	0.450	0.471	0.298	0.625	0.383	0.624	0.340	0.628	0.379	0.854	0.416	0.661	0.363
Electricity	**0.175**	**0.273**	0.314	0.366	0.333	0.378	0.440	0.494	0.200	0.288	0.212	0.300	0.193	0.296	0.227	0.338	0.311	0.397	0.272	0.367
Weather	**0.249**	**0.275**	0.256	0.305	0.290	0.348	0.768	0.672	0.256	0.279	0.265	0.317	0.288	0.314	0.338	0.382	0.634	0.548	0.611	0.557
ETTm1	**0.387**	**0.399**	0.509	0.507	0.566	0.537	1.947	1.206	0.389	0.401	0.403	0.407	0.481	0.456	0.588	0.517	0.961	0.734	0.936	0.728
ETTm2	**0.277**	**0.322**	1.433	0.747	1.287	0.751	2.639	1.280	0.280	0.328	0.350	0.401	0.306	0.347	0.327	0.371	1.410	0.810	1.478	0.873
ETTh1	**0.431**	**0.433**	0.615	0.563	0.679	0.605	2.113	1.237	0.443	0.443	0.456	0.452	0.570	0.537	0.496	0.487	1.040	0.795	0.919	0.759
ETTh2	**0.374**	**0.403**	2.170	1.175	2.618	1.308	4.382	2.008	0.381	0.404	0.559	0.515	0.526	0.516	0.450	0.459	4.431	1.729	4.492	1.691
Exchange	**0.354**	**0.400**	0.756	0.645	0.786	0.674	1.681	0.197	**0.354**	**0.400**	**0.354**	0.414	0.461	0.454	0.613	0.539	1.550	0.998	1.386	0.898
ILI	2.113	**0.877**	3.417	1.214	4.861	1.507	5.300	1.657	2.065	0.882	2.616	1.090	**2.077**	0.914	3.006	1.161	5.137	1.544	4.784	1.471

Long-term forecasting results. Table 2 presents the forecasting results for the proposed SageFormer and other baseline models. The table shows that the proposed model consistently achieves state-of-the-art performance across all benchmarks and prediction lengths. Notably, SageFormer significantly outperforms other deep models on datasets with a large number of series, attaining a **7.4%** average MSE reduction ($0.471 \rightarrow 0.436$) on Traffic and a **9.3%** average MSE reduction ($0.193 \rightarrow 0.175$) on Electricity compared to previous state-of-the-art results. Our model exhibits substantial improvement on every dataset, particularly compared to models that explicitly utilize inter-series dependencies. This indicates that our proposed method effectively enhances the model's ability to capture relationships among multiple series.

Framework generality. Furthermore, our model serves as a versatile extension for Transformer-based architectures. To validate this, we apply the SageFormer framework to three prominent Transformers and report the performance enhancement of each model as Table 3. Our method consistently improves the forecasting ability of different models, demonstrating that SageFormer is an effective, universally applicable framework. By leveraging graph structures, it can better utilize the interdependencies among various sequences, ultimately achieving superior predictive performance.

Table 3: Performance promotion by applying the proposed framework to Transformer and its variants. We report all prediction lengths' averaged MSE/MAE (same as Table 2). Full results (under all prediction lengths) see Table 7 in Appendix A

Dataset	Transformer		**+Ours**		Informer		**+Ours**		FEDformer		**+Ours**	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Traffic	0.661	0.362	**0.549**	**0.293**	0.610	0.376	**0.578**	**0.318**	0.764	0.416	**0.707**	**0.396**
Electricity	0.272	0.367	**0.202**	**0.290**	0.214	0.327	**0.207**	**0.305**	0.311	0.397	**0.223**	**0.319**
Weather	0.611	0.557	**0.290**	**0.348**	0.309	0.360	**0.285**	**0.323**	0.634	0.548	**0.269**	**0.304**
ETTh1	0.919	0.759	**0.459**	**0.456**	0.440	0.460	**0.433**	**0.442**	1.040	0.795	**0.673**	**0.577**

4.3 Ablation Study

The ablation studies were conducted to address two primary concerns: 1) the impact of graph aggregation and 2) the impact of series-aware global tokens. We designate SageFormer variants without specific components as shown in Table 4.

First, the experiments validated the effectiveness of the graph structure in our time series prediction model. Removing the graph aggregation module from each encoder layer resulted in a substantial decline in prediction accuracy. On the Traffic dataset, the average decrease was 7.3%, and on the seven-series ETTh1 dataset, it was 2.8%, showing that graph structures enhance performance more in datasets with numerous series. Second, series-aware global tokens enhanced the model's prediction accuracy while reducing computational overhead. If all tokens (not just global tokens) participated in graph propagation calculations, the model's performance would decline by 6.3% and 1.6% on the Traffic and ETTh1 datasets, respectively. Lastly, we discovered that techniques like sparse constraints and directed graphs in graph construction were more effective for larger datasets (e.g., Traffic). In comparison, they had little impact on smaller datasets' prediction results. This finding suggests that applying sparse constraints can mitigate the impact of variable redundancy on the model while conserving computational resources.

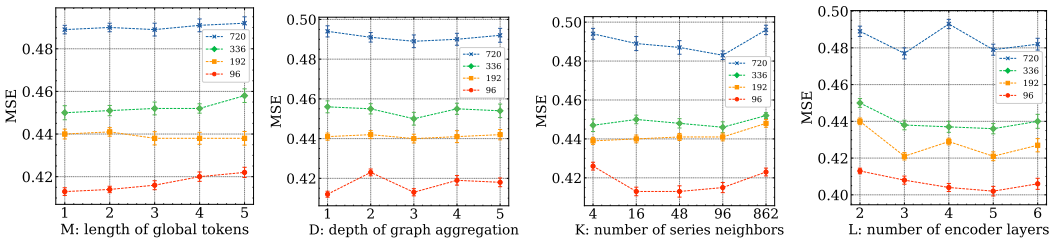
4.4 Effect of Hyper-parameters

In this section, we examine the impact of four hyperparameters on our proposed SageFormer model: global token length, depth of graph aggregation, the number of nearest neighbors, and the depth of encoder layers. We conduct a sensitivity analysis on the Traffic dataset (Figure 3). For each of the four tasks, SageFormer consistently delivers stable performance, regardless of the selected value.

Global token length (Figure 3a): The model's performance remains consistent across all prediction windows, irrespective of the value of M. To optimize computational efficiency, we set M=1. **Depth of graph aggregation** (Figure 3b): The model demonstrates robust performance with varying graph aggregation depths. To balance accuracy and efficiency, we set d=3. **Number of nearest neighbors** (Figure 3c): Larger k values generally yield better results, but performance declines when a fully connected graph is utilized. This suggests sequence redundancy in MTS forecasting tasks, so we

Table 4: Model architecture ablations (MSE metrics are reported). *-Graph Aggregation*: Elimination of graph aggregation in the encoder; *-Global Tokens*: Graph information propagation applied to all tokens without global tokens; *-Sparse Graph*: Removal of the k-nearest neighbor constraint in graph structure learning; *-Directed Graph*: Modification of graph structure learning from directed to undirected graphs.

Datasets	Traffic				AVG	ETTh1				AVG
Prediction Lengths	96	192	336	720		96	192	336	720	
SageFormer	**0.408**	**0.421**	**0.438**	**0.477**	**0.436**	0.377	**0.423**	**0.459**	**0.465**	**0.431**
- Graph Aggregation	0.446	0.452	0.466	0.506	0.468	**0.372**	0.439	0.468	0.491	0.443
- Global Tokens	0.420	0.440	0.457	0.507	0.456	0.381	0.431	0.462	0.478	0.438
- Sparse Graph	0.416	0.441	0.445	0.484	0.447	0.377	0.423	0.459	0.467	0.432
- Directed Graph	0.415	0.442	0.449	0.494	0.450	0.379	0.424	0.461	0.468	0.433



(a) (b) (c) (d)

Figure 3: Evaluation on hyper-parameter impact. (a) MSE against the length of global tokens on the Traffic dataset. (b) MSE against the graph aggregation depth on the Traffic dataset. (c) MSE against the number of nearest neighbors on the Traffic dataset. (d) MSE against SageFormer encoder layers on the Traffic dataset.

select k=16. **SageFormer encoder layers** (Figure 3d): Increasing the number of encoding layers results in a higher parameter count for the model and its computational time. No significant reduction is observed after the model surpasses three layers, leading us to set the model's layers to 3.

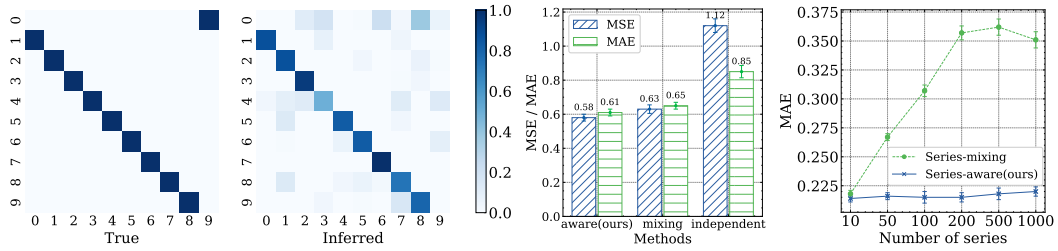
4.5 Synthetic Datasets

Directed Cycle Graph Dataset. In this section, we investigate the adjacency matrices inferred by SageFormer using a synthetic dataset consisting of N=10 nodes. Each series value $x_{i,t}$ is sampled from another series $(i-1 \ mod \ N)$ with temporal lag $\tau = 10$, resulting in a directed cycle graph for the adjacency matrix. The dataset details are provided in Appendix A.

Figure 4a represents the actual inter-series dependencies alongside our inferred results, effectively demonstrating that our method successfully recovers these dependencies. As Figure 4b indicates, our proposed series-aware framework outperforms the previous series-mixing and series-independent frameworks, achieving the lowest MAE and MSE test losses. Importantly, the series-independent framework's performance is notably poor in this context, with an MSE exceeding 1. This deficiency stems from its disregard for the significant inter-series dependencies inherent in this dataset, particularly given that each sub-series is nearly equivalent to white noise in this dataset.

Low-rank Dataset. To assess the effectiveness of different models in handling sparse data, we designed multiple Low-rank MTS datasets with varying numbers of series. Inspired by the Discrete Sine Transformation, we generate arbitrary signals as the sum of distinct sinusoids combined with Gaussian noise. The same sinusoids are shared among different nodes, creating the low-rank property. The dataset details are provided in Appendix A.

Figure 4c presents the prediction MAE results for datasets with varying numbers of series (N) using the series-mixing method and our approach. It can be observed that the prediction performance of the series-mixing method deteriorates rapidly as the number of series increases since it encodes all series information into the same token. In contrast, the MAE of our method does not increase with



(a) (b) (c)

Figure 4: Evaluation on synthetic datasets. (a) The left side displays the heat map of the actual adjacency matrix, while the right side presents the inferred adjacency matrix by SageFormer, illustrating the effectiveness of our proposed method in learning the inherent graph structure.; (b) Prediction results of three different methods on the Directed Cycle Graph dataset; (c) Prediction MAE results for low-rank datasets with varying numbers of series (N). We selected the Nonstationary Transformer for the series-mixing method, and for the series-independent method, we chose PatchTST as a representative.

the growth in the number of series, indicating that our designed approach can effectively exploit the low-rank characteristics of the dataset.

4.6 Computational Efficiency Analysis

We compared the computational efficiency of our model, SageFormer, with other Transformer-based models (Table 5). Although SageFormer's complexity is theoretically quadratic to historical series length T, a large patch length P in practice brings its runtime close to linear complexity models. An additional $O(C^2)$ complexity is due to standard graph convolution operations, but techniques exist to reduce this to linear complexity [33, 34]. In the decoder part, the complexity of SageFormer is simplified to linear, owing to the streamlined design of the linear decoder head.

We also evaluated running time and memory consumption on the Traffic dataset, which has the most variables. SageFormer balances running time and memory usage well, achieving a running time of 0.31 ± 0.03 seconds per batch and consuming 12.42 GB of memory. This result is slightly slower compared to the PatchTST [13] model but is faster than the Crossformer [11] model. These outcomes suggest that our proposed SageFormer model presents a competitive trade-off between efficiency and prediction accuracy.

Table 5: Computational complexity per layer of Transformer-based models. T denotes the length of the historical series, τ represents the length of the prediction window, C is the number of series, and P corresponds to the segment length of each patch.

Methods	Encoder layer	Decoder layer	Time(s/batch)	Memory(GB)
Transformer [20]	$O(T^2)$	$O\left(\tau(\tau+T)\right)$	0.07 ± 0.01	2.39
Informer [7]	$O(T \log T)$	$O\left(\tau(\tau+\log T)\right)$	0.04 ± 0.04	2.31
FEDformer [9]	$O(T)$	$O\left(\tau+T/2\right)$	0.18 ± 0.03	3.13
PatchTST [13]	$O(CT^2/P^2)$	$O(\tau)$	0.22 ± 0.04	11.44
Crossformer [11]	$O(CT^2/P^2)$	$O(C\tau(\tau+T)/P^2)$	0.82 ± 0.05	22.53
SageFormer **(ours)**	$O(CT^2/P^2+C^2)$	$O(\tau)$	0.31 ± 0.03	12.42

5 Conclusion and Future Work

This paper presented SageFormer, a novel approach for modeling inter-series dependencies in long-term Multivariate Time Series (MTS) forecasting tasks. By amalgamating graph neural networks (GNN) with Transformer structures, SageFormer can effectively capture diverse temporal patterns and harness dependencies among various series. Our model has demonstrated impressive versatility through extensive experimentation, delivering state-of-the-art performance on real-world and synthetic datasets. SageFormer thus presents a promising solution to overcome the limitations of series

dependencies modeling in MTS forecasting tasks and exhibits potential for further advancements and applications in other domains involving inter-series dependencies.

We also acknowledged the limitations of our work and briefly delineated potential avenues for future research. While SageFormer achieves exceptional performance in long-term MTS forecasting, the dependencies it captures do not strictly represent causality. As a result, some dependencies may prove unreliable in practical scenarios due to the non-stationary nature of time series. Our primary focus on enhancing long-term forecasting performance has led to some degree of overlooking the interpretability of the graph structure. Moving forward, our work's graph neural network component could be improved to learn causal relationships between variables and reduce its complexity. The framework proposed in this paper could also be applied to non-Transformer models in the future.

References

[1] Zulfiqar Ahmad Khan, Tanveer Hussain, Amin Ullah, Seungmin Rho, Miyoung Lee, and Sung Wook Baik. Towards efficient electricity forecasting in residential and commercial buildings: A novel hybrid cnn with a lstm-ae based framework. *Sensors*, 20(5):1399, 2020.

[2] Yunyue Zhu and Dennis Shasha. Statstream: Statistical monitoring of thousands of data streams in real time. In *VLDB'02: Proceedings of the 28th International Conference on Very Large Databases*, pages 358–369. Elsevier, 2002.

[3] Rafal A Angryk, Petrus C Martens, Berkay Aydin, Dustin Kempton, Sushant S Mahajan, Sunitha Basodi, Azim Ahmadzadeh, Xumin Cai, Soukaina Filali Boubrahimi, Shah Muhammad Hamdi, et al. Multivariate time series dataset for space weather data analytics. *Scientific data*, 7(1):227, 2020.

[4] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long-and short-term temporal patterns with deep neural networks. In *SIGIR*, 2018.

[5] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

[6] Shiyang Li, Xiaoyong Jin, Yao Xuan, Xiyou Zhou, Wenhu Chen, Yu-Xiang Wang, and Xifeng Yan. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. In *NeurIPS*, 2019.

[7] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *AAAI*, 2021.

[8] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with Auto-Correlation for long-term series forecasting. In *NeurIPS*, 2021.

[9] Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *ICML*, 2022.

[10] James Durbin and Siem Jan Koopman. *Time series analysis by state space methods*, volume 38. OUP Oxford, 2012.

[11] Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *International Conference on Learning Representations*, 2023.

[12] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *AAAI*, 2023.

[13] Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *International Conference on Learning Representations*, 2023.

[14] Lu Han, Han-Jia Ye, and De-Chuan Zhan. The capacity and robustness trade-off: Revisiting the channel independent strategy for multivariate time series forecasting, 2023.

[15] Zhe Li, Zhongwen Rao, Lujia Pan, and Zenglin Xu. Mts-mixers: Multivariate time series forecasting via factorized temporal and channel mixing. *arXiv preprint arXiv:2302.04501*, 2023.

[16] George EP Box and Gwilym M Jenkins. Some recent advances in forecasting and control. *J. R. Stat. Soc. (Series-C)*, 1968.

[17] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[18] Rajat Sen, Hsiang-Fu Yu, and Inderjit S Dhillon. Think globally, act locally: A deep neural network approach to high-dimensional time series forecasting. In *NeurIPS*, 2019.

[19] David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *Int. J. Forecast.*, 2020.

[20] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

[21] J. Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2019.

[22] Linhao Dong, Shuang Xu, and Bo Xu. Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5884–5888. IEEE, 2018.

[23] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021.

[24] Shizhan Liu, Hang Yu, Cong Liao, Jianguo Li, Weiyao Lin, Alex X Liu, and Schahram Dustdar. Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. In *ICLR*, 2021.

[25] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *International Conference on Learning Representations*, 2018.

[26] Bing Yu, Haoteng Yin, and Zhanxing Zhu. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI)*, 2018.

[27] Defu Cao, Yujing Wang, Juanyong Duan, Ce Zhang, Xia Zhu, Congrui Huang, Yunhai Tong, Bixiong Xu, Jing Bai, Jie Tong, et al. Spectral temporal graph neural network for multivariate time-series forecasting. *Advances in neural information processing systems*, 33:17766–17778, 2020.

[28] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, Xiaojun Chang, and Chengqi Zhang. Connecting the dots: Multivariate time series forecasting with graph neural networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 753–763, 2020.

[29] Jake Grigsby, Zhe Wang, and Yanjun Qi. Long-range transformers for dynamic spatiotemporal forecasting. *arXiv preprint arXiv:2109.12218*, 2021.

[30] Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis, 2023.

[31] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

[32] Yong Liu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Non-stationary transformers: Rethinking the stationarity in time series forecasting. In *NeurIPS*, 2022.

[33] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. Simplifying graph convolutional networks. In *International conference on machine learning*, pages 6861–6871. PMLR, 2019.

[34] Ron Levie, Federico Monti, Xavier Bresson, and Michael M Bronstein. Cayleynets: Graph convolutional neural networks with complex rational spectral filters. *IEEE Transactions on Signal Processing*, 67(1):97–109, 2018.

A Supplementary Material

A.1 Table of Notations

In the paper, scalars are denoted by lowercase letters, vectors by bold lowercase letters, matrices by bold uppercase letters, and tensors by calligraphic letters. Important notations that appear in this paper are summarized in Table 6

Table 6: Table of Notations

Symbol	Description
C	Number of series in MTS data
$\mathbf{x}_t \in \mathbb{R}^C$	The value of C series at time step t
$\mathbf{X}_t \in \mathbb{R}^{C \times L}$	A historical MTS instance with length L
$\mathbf{Y}_t \in \mathbb{R}^{C \times T}$	The next T steps of MTS values to be predicted
$f(\cdot)$	Mapping function from $\mathbf{X}_t$ to $\mathbf{Y}_t$
$\mathcal{G}$	MTS signal set viewed from a graph perspective
$\mathcal{V}$	Node set containing C series of MTS data
$\mathbf{A} \in \mathbb{R}^{C \times C}$	Weighted adjacency matrix
a_{ij}	Entry indicating the dependencies between series i and j
$\mathcal{X} \in \mathbb{R}^{C \times (N+M) \times D}$	Final embedding tensor with M global tokens of dimension D
TEB	Transformer Encoding Block
GNN	Graph Neural Network
FlattenHead	Linear decoder head replacing the traditional Transformer decoder

A.2 Benchmarking Datasets

Our experiments are carried out on six real-world datasets as described below:

1. The Traffic dataset[1] contains data from 862 sensors installed on highways in the San Francisco Bay Area, measuring road occupancy. This dataset is recorded on an hourly basis over two years, resulting in a total of 17,544 time steps.
2. The Electricity Consumption Load (Electricity) dataset[2] records the power usage of 321 customers. The data is logged hourly over a two-year period, amassing a total of 26,304 time steps.
3. The Weather dataset[3] is a meteorological collection featuring 21 variates, gathered in the United States over a four-year span.
4. The Electricity Transformers Temperature (ETT) datasets[4] are procured from two electricity substations over two years. They provide a summary of load and oil temperature data across seven variates. For ETTm1 and ETTm2, the "m" signifies that data was recorded every 15 minutes, yielding a total of 69,680 time steps. ETTh1 and ETTh2 represent the hourly equivalents of ETTm1 and ETTm2, respectively, each containing 17,420 time steps.
5. The Exchange dataset[5] details the daily foreign exchange rates of eight different countries, including Australia, British, Canada, Switzerland, China, Japan, New Zealand and Singapore ranging from 1990 to 2016.
6. The Influenza-Like Illness (ILI) dataset[6] is maintained by the United States Centers for Disease Control and Prevention. It collates patient information on a weekly basis for a period spanning from 2002 to 2021.

[1] https://pems.dot.ca.gov/
[2] https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014
[3] https://www.bgc-jena.mpg.de/wetter
[4] https://github.com/zhouhaoyi/ETDataset
[5] https://github.com/laiguokun/multivariate-time-series-data
[6] https://gis.cdc.gov/grasp/fluview/fluportaldashboard.html

A.3 Hyper-parameter Selection and Implementation Details

We conduct all experiments five times, each implemented in PyTorch, and carried out on a single NVIDIA TITAN RTX 3090 24GB GPU. The repeated experimentation helps mitigate the effects of random variations in reported values. The look-back window size is set to 96, while the forecasting horizon ranges between 96 and 720. Model optimization is achieved through the Adam optimizer with a learning rate of $1e-4$. The models are trained for 20 epochs or fewer, utilizing an early stop mechanism to prevent overfitting.

By default, SageFormer comprises two encoder layers, a head count (H) of 16, and a model hidden dimension (d) of 512. We designate the number of global tokens as $M = 1$; the node embedding dimension is set to 16, the number of nearest neighbors is $K = 16$, and the graph aggregation depth is $D = 3$. Furthermore, we employ a patch length (P) of 24 and a stride (S) of 8, consistent with the parameters utilized in PatchTST. For the larger datasets (Traffic, Electricity), a model with three encoder layers is adopted to enhance the expressive capacity.

A.4 Baselines

All baselines are reproduced using the original paper's configuration or the official code. However, the only exception is that we standardize the look-back window across all models to 36 for the ILI dataset and 96 for the remaining datasets to ensure a fair comparison. Consequently, some discrepancies may exist between our input-output setting and those reported in the referenced papers. We exclude traditional time series forecasting models (such as ARIMA, LSTM) from our baselines, as Transformer-based models have been demonstrated to outperform these in long-term forecasting tasks [7].

A.5 More on Synthetic Datasets

Directed Cycle Graph Dataset. This synthetic dataset comprises a panel of $N = 10$ time series, each with a length of 10000. For each series i at timestep t, the data is sampled from the past step $t - \tau$ of another time series $i - 1 \bmod N$ from the same panel. The formal generation process can be expressed as follows:

$$x_{i,t} \sim \mathcal{N}\left(\beta x_{(i-1 \bmod N),t-10}; \sigma^2\right), \tag{5}$$

where $x_{i,t}$ represents the i-th time series at time step t. The operator $\sim$ indicates that the i-th series at time t is generated from a normal distribution, denoted by $\mathcal{N}$, with mean and variance parameters. The mean of this distribution is given by $\beta x_{(i-1 \bmod N),t-10}$, which represents the scaled value of the preceding time series $(i - 1 \bmod N)$ at a previous time step $(t - 10)$. β is a scaling factor applied to this value. The variance of the distribution is given by σ^2, a constant that determines the degree of deviation allowed from the mean.

The dataset exhibits a directed cycle graph structure due to its distinct data generation process. Each time series i at time step t is modeled based on the past values of the preceding series in the panel, specifically, the $t - \tau$ step of the series $i - 1 \bmod N$. This creates a cyclical dependency among the series, with each one being influenced by its predecessor in the panel. Such a design results in a directed cycle graph structure, as observed in the multivariate time series adjacency matrix. This arrangement effectively mirrors the pattern of a previous series with a temporal lag, as evidenced in Figure 5, which illustrates the first 120 timesteps of these ten series.

Low-rank Dataset. This synthetic dataset comprises time series of length 10000 with varying numbers of series. Each series is generated as a weighted sum of different sinusoids, each with its own frequency and amplitude. This generation process can be formally expressed as:

$$x_i = \sum_{m=1}^{M} B_{i,m} sin\left(2\pi\omega_{\lfloor i/K \rfloor,m} t\right) + \epsilon_{i,t}, \tag{6}$$

In the above equation, $B_{i,m}$ is sampled from a uniform distribution $\tilde{B}_{i,m} \sim U(0.4, 1)$ and subsequently normalized by $B_{i,m} = \frac{\tilde{B}_{i,m}}{\sum_m \tilde{B}_{i,m}}$. $\omega_{\lfloor i/K \rfloor,m}$ is also sampled from a uniform distribution

$U(0, 0.2)$, and $\epsilon_{i,t}$ is sampled from a Gaussian distribution $\mathcal{N}(0, 0.2^2)$. Lastly, $\omega_{\lfloor i/K \rfloor, m}$ is shared among different series within the same group, resulting in the low-rank characteristic of the dataset. We select $M = 3$ and $K = 2$ for these experiments.

The low-rank property in this dataset arises due to the shared frequencies $\omega_{\lfloor i/K \rfloor, m}$ among series within the same group. This results in a limited number of unique patterns across the series, causing the dataset to effectively exhibit a low-rank structure. Despite the high number of series, the shared characteristics permit a concise, low-dimensional representation, signifying the low-rank property.

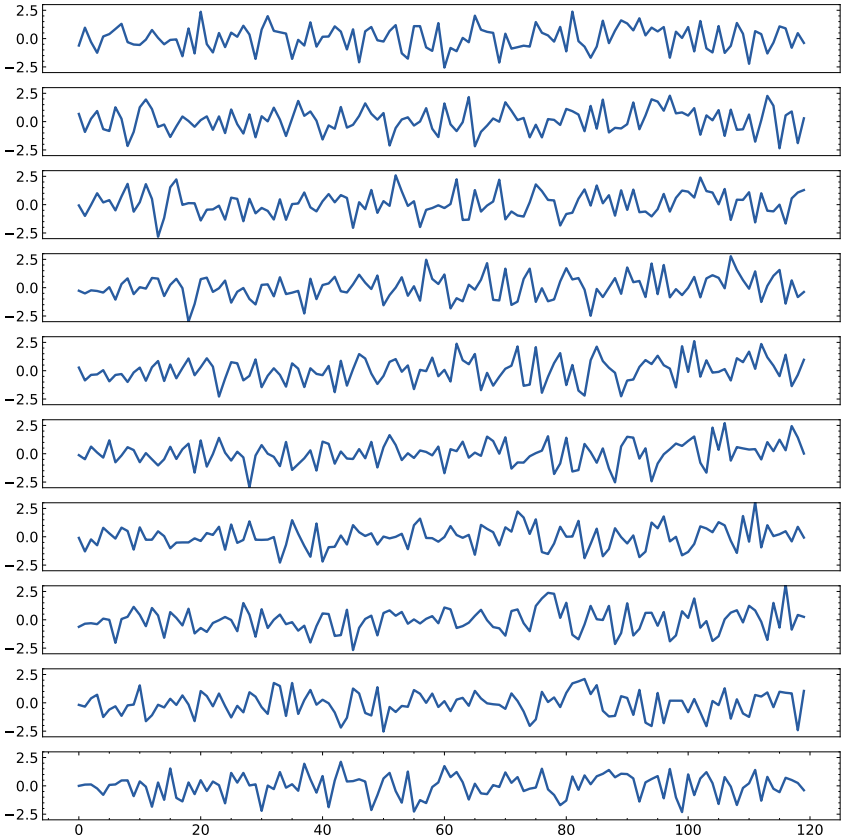


Figure 5: First 120 timesteps of the Directed Cycle Graph Dataset.

A.6 Full Results

Due to the space limitation of the main text, we place the full results of experiments in the following: long-term forecasting reults in Table 8 and framework generality results in Table 7.

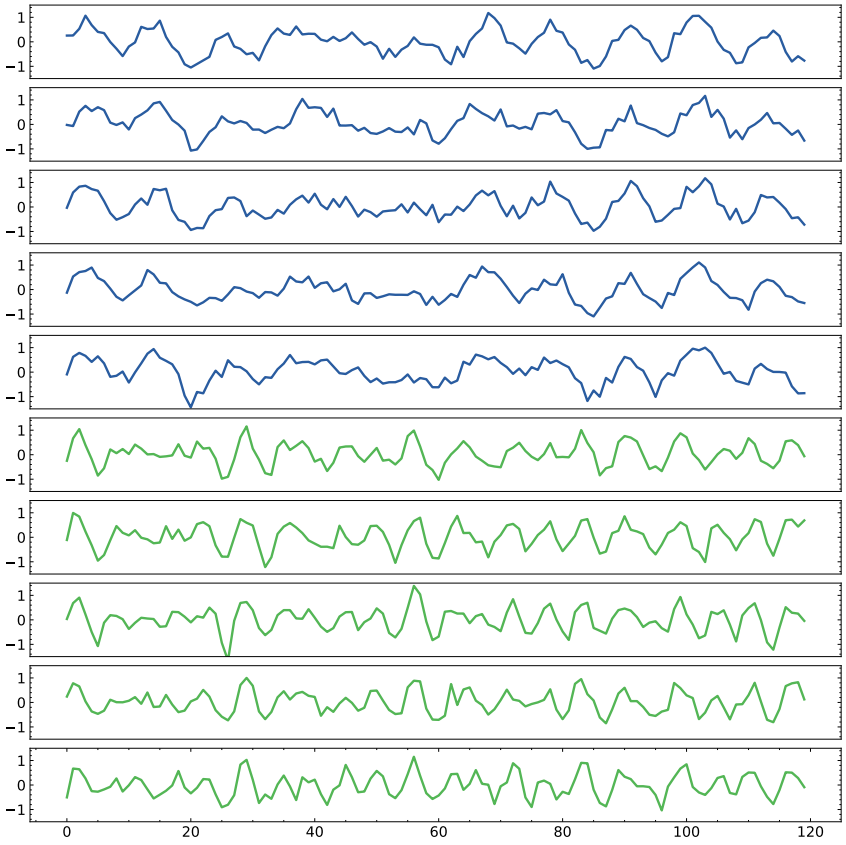


Figure 6: First 120 timesteps of the Low-rank Dataset.

Table 7: Performance promotion by applying the proposed framework to Transformer and its variants. We report the averaged MSE/MAE of all prediction lengths (same as Table 2).

Dataset		Transformer		**+Ours**		Informer		**+Ours**		FEDformer		**+Ours**	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Traffic	96	0.644	0.354	**0.526**	**0.281**	0.587	0.366	**0.561**	**0.318**	0.719	0.391	**0.672**	**0.382**
	192	0.662	0.365	**0.536**	**0.287**	0.604	0.373	**0.572**	**0.320**	0.696	0.379	**0.688**	**0.369**
	336	0.661	0.361	**0.553**	**0.296**	0.621	0.383	**0.582**	**0.321**	0.777	0.420	**0.708**	**0.405**
	720	0.677	0.370	**0.581**	**0.306**	0.626	0.382	**0.598**	**0.312**	0.864	0.472	**0.761**	**0.428**
	Avg	0.661	0.362	**0.549**	**0.293**	0.610	0.376	**0.578**	**0.318**	0.764	0.416	**0.707**	**0.396**
Electricity	96	0.259	0.358	**0.186**	**0.272**	0.193	0.308	**0.182**	**0.278**	0.274	0.368	**0.188**	**0.275**
	192	0.265	0.363	**0.188**	**0.275**	0.201	0.315	**0.191**	**0.285**	0.296	0.386	**0.212**	**0.311**
	336	0.273	0.369	**0.203**	**0.293**	0.214	0.329	**0.206**	**0.313**	0.300	0.394	**0.223**	**0.327**
	720	0.291	0.377	**0.234**	**0.320**	0.246	0.355	**0.248**	**0.345**	0.373	0.439	**0.267**	**0.361**
	Avg	0.272	0.367	**0.202**	**0.290**	0.214	0.327	**0.207**	**0.305**	0.311	0.397	**0.223**	**0.319**
Weather	96	0.399	0.424	**0.189**	**0.252**	0.217	0.296	**0.208**	**0.276**	0.300	0.384	**0.178**	**0.224**
	192	0.566	0.537	**0.235**	**0.299**	0.276	0.336	**0.249**	**0.312**	0.598	0.544	**0.226**	**0.262**
	336	0.631	0.582	**0.295**	**0.359**	0.339	0.380	**0.296**	**0.322**	0.578	0.523	**0.287**	**0.305**
	720	0.849	0.685	**0.440**	**0.481**	0.403	0.428	**0.385**	**0.382**	1.059	0.741	**0.384**	**0.423**
	Avg	0.611	0.557	**0.290**	**0.348**	0.309	0.360	**0.285**	**0.323**	0.634	0.548	**0.269**	**0.304**
ETTh1	96	0.780	0.685	**0.384**	**0.403**	0.376	0.419	**0.377**	**0.408**	0.865	0.713	**0.578**	**0.512**
	192	0.906	0.755	**0.435**	**0.433**	0.420	0.448	**0.431**	**0.440**	1.008	0.792	**0.669**	**0.558**
	336	0.980	0.797	**0.479**	**0.459**	0.459	0.465	**0.455**	**0.456**	1.107	0.809	**0.699**	**0.592**
	720	1.008	0.800	**0.538**	**0.528**	0.506	0.507	**0.467**	**0.462**	1.181	0.865	**0.745**	**0.644**
	Avg	0.919	0.759	**0.459**	**0.456**	0.440	0.460	**0.433**	**0.442**	1.040	0.795	**0.673**	**0.577**

Table 8: Full results for the long-term forecasting task. We compare extensive competitive models under different prediction lengths. The input sequence length is set to 36 for the ILI dataset and 96 for the others. *Avg* is averaged from all four prediction lengths.

Models		**SageFormer (Ours)**		Crossformer* [11]		MTGNN* [28]		LSTnet* [4]		PatchTST* [13]		DLinear* [12]		Stationary [32]		Autoformer [8]		Informer [7]		Transformer [20]	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Traffic	96	**0.408**	**0.271**	0.544	0.307	0.574	0.312	0.711	0.432	0.450	0.287	0.650	0.396	0.612	0.338	0.613	0.388	0.719	0.391	0.644	0.354
	192	**0.421**	**0.279**	0.566	0.315	0.587	0.315	0.722	0.441	0.456	0.292	0.598	0.370	0.613	0.340	0.616	0.382	0.696	0.379	0.662	0.365
	336	**0.438**	**0.283**	0.571	0.314	0.594	0.318	0.741	0.451	0.471	0.297	0.605	0.373	0.618	0.328	0.622	0.337	0.777	0.420	0.661	0.361
	720	**0.477**	**0.308**	0.599	0.313	0.612	0.322	0.768	0.474	0.509	0.317	0.645	0.394	0.653	0.355	0.660	0.408	0.864	0.472	0.677	0.370
	Avg	**0.436**	**0.285**	0.570	0.312	0.592	0.317	0.736	0.450	0.472	0.298	0.625	0.383	0.624	0.340	0.628	0.379	0.764	0.416	0.661	0.363
Electricity	96	**0.147**	**0.246**	0.213	0.300	0.243	0.342	0.382	0.452	0.175	0.266	0.197	0.282	0.169	0.273	0.201	0.317	0.274	0.368	0.259	0.358
	192	**0.161**	**0.259**	0.290	0.351	0.298	0.364	0.401	0.482	0.184	0.274	0.196	0.285	0.182	0.286	0.222	0.334	0.296	0.386	0.265	0.363
	336	**0.180**	**0.279**	0.348	0.389	0.368	0.396	0.419	0.477	0.200	0.290	0.209	0.301	0.200	0.304	0.231	0.338	0.300	0.394	0.273	0.369
	720	**0.213**	**0.309**	0.405	0.425	0.422	0.410	0.556	0.565	0.240	0.322	0.245	0.333	0.222	0.321	0.254	0.361	0.373	0.439	0.291	0.377
	Avg	**0.175**	**0.273**	0.314	0.366	0.333	0.378	0.440	0.494	0.200	0.288	0.212	0.300	0.193	0.296	0.227	0.338	0.311	0.397	0.272	0.367
Weather	96	**0.162**	**0.206**	**0.162**	0.232	0.189	0.252	0.682	0.594	0.175	0.216	0.196	0.255	0.173	0.223	0.266	0.336	0.300	0.384	0.399	0.424
	192	0.211	**0.250**	**0.208**	0.277	0.235	0.299	0.755	0.652	0.219	0.256	0.237	0.296	0.245	0.285	0.307	0.367	0.598	0.544	0.566	0.537
	336	0.271	**0.294**	**0.265**	0.320	0.295	0.359	0.782	0.683	0.277	0.297	0.283	0.335	0.321	0.338	0.359	0.395	0.578	0.523	0.631	0.582
	720	**0.345**	**0.343**	0.388	0.391	0.440	0.481	0.851	0.757	0.353	0.346	**0.345**	0.381	0.414	0.410	0.419	0.428	1.059	0.741	0.849	0.685
	Avg	**0.247**	**0.273**	0.256	0.305	0.290	0.348	0.768	0.672	0.256	0.279	0.265	0.317	0.288	0.314	0.338	0.382	0.634	0.548	0.611	0.557
ETTm1	96	0.324	**0.362**	0.361	0.402	0.428	0.446	1.339	0.913	**0.319**	0.361	0.345	0.372	0.386	0.398	0.505	0.475	0.672	0.571	0.701	0.609
	192	**0.368**	**0.387**	0.403	0.440	0.457	0.469	1.542	1.009	0.370	0.389	0.380	0.389	0.459	0.444	0.553	0.496	0.795	0.669	0.829	0.676
	336	**0.401**	**0.408**	0.551	0.535	0.579	0.562	1.920	1.234	0.406	0.410	0.413	0.413	0.495	0.464	0.621	0.537	1.212	0.871	1.024	0.783
	720	**0.457**	**0.441**	0.720	0.649	0.798	0.671	2.987	1.669	0.461	0.444	0.474	0.453	0.585	0.516	0.671	0.561	1.166	0.823	1.189	0.843
	Avg	**0.388**	**0.400**	0.509	0.507	0.566	0.537	1.947	1.206	0.389	0.401	0.403	0.407	0.481	0.456	0.588	0.517	0.961	0.734	0.936	0.728
ETTm2	96	**0.173**	**0.255**	0.253	0.347	0.289	0.364	0.723	0.655	0.177	0.261	0.193	0.292	0.192	0.274	0.255	0.339	0.365	0.453	0.515	0.534
	192	**0.239**	**0.299**	0.421	0.483	0.456	0.492	1.285	0.932	0.241	0.303	0.284	0.362	0.280	0.339	0.281	0.340	0.533	0.563	1.424	0.892
	336	**0.299**	**0.338**	1.276	0.805	1.432	0.812	3.064	1.556	0.302	0.343	0.369	0.427	0.334	0.361	0.339	0.372	1.363	0.887	1.183	0.829
	720	**0.395**	**0.395**	3.783	1.354	2.972	1.336	5.484	1.978	0.401	0.403	0.554	0.522	0.417	0.413	0.433	0.432	3.379	1.338	2.788	1.237
	Avg	**0.277**	**0.322**	1.433	0.747	1.287	0.751	2.639	1.280	0.280	0.328	0.350	0.401	0.306	0.347	0.327	0.371	1.410	0.810	1.478	0.873
ETTh1	96	**0.377**	**0.397**	0.421	0.448	0.522	0.490	1.654	0.982	0.372	0.396	0.386	0.400	0.513	0.491	0.449	0.459	0.865	0.713	0.780	0.685
	192	**0.423**	**0.425**	0.534	0.515	0.542	0.536	1.999	1.218	0.439	0.433	0.437	0.432	0.534	0.504	0.500	0.482	1.008	0.792	0.906	0.755
	336	**0.459**	**0.445**	0.656	0.581	0.736	0.643	2.655	1.369	0.468	0.456	0.481	0.459	0.588	0.535	0.521	0.496	1.107	0.809	0.980	0.797
	720	**0.465**	**0.466**	0.849	0.709	0.916	0.750	2.143	1.380	0.491	0.486	0.519	0.516	0.643	0.616	0.514	0.512	1.181	0.865	1.008	0.800
	Avg	**0.431**	**0.433**	0.615	0.563	0.679	0.605	2.113	1.237	0.443	0.443	0.456	0.452	0.570	0.537	0.496	0.487	1.040	0.795	0.919	0.759
ETTh2	96	**0.286**	**0.338**	1.142	0.772	1.843	0.911	3.252	1.604	0.302	0.346	0.333	0.387	0.476	0.458	0.346	0.388	3.755	1.525	2.590	1.282
	192	**0.368**	**0.394**	1.784	1.022	2.439	1.325	5.122	2.309	0.379	0.398	0.477	0.476	0.512	0.493	0.456	0.452	5.602	1.931	6.464	2.102
	336	**0.413**	**0.429**	2.644	1.404	2.944	1.405	4.051	1.742	0.418	0.431	0.594	0.541	0.552	0.551	0.482	0.486	4.721	1.835	5.851	1.970
	720	**0.427**	**0.449**	3.111	1.501	3.244	1.592	5.104	2.378	0.423	0.440	0.831	0.657	0.562	0.560	0.515	0.511	3.647	1.625	3.063	1.411
	Avg	**0.374**	**0.403**	2.170	1.175	2.618	1.308	4.382	2.008	0.381	0.404	0.559	0.515	0.526	0.516	0.450	0.459	4.431	1.729	4.492	1.691
Exchange	96	**0.082**	**0.200**	0.256	0.367	0.289	0.388	0.593	0.711	0.087	0.205	0.088	0.218	0.111	0.237	0.197	0.323	0.847	0.752	0.527	0.566
	192	**0.175**	**0.298**	0.469	0.508	0.441	0.498	1.164	1.047	0.183	0.305	0.176	0.315	0.219	0.335	0.300	0.369	1.204	0.895	0.942	0.737
	336	0.331	0.416	0.901	0.741	0.934	0.774	1.544	1.196	0.321	**0.410**	**0.313**	0.427	0.421	0.476	0.509	0.524	1.672	1.036	1.485	0.946
	720	0.826	0.683	1.398	0.965	1.478	1.037	3.425	1.833	**0.825**	**0.680**	0.839	0.695	1.092	0.769	1.447	0.941	2.478	1.310	2.588	1.341
	Avg	**0.354**	**0.399**	0.756	0.645	0.786	0.674	1.681	1.197	0.354	0.400	0.354	0.414	0.461	0.454	0.613	0.539	1.550	0.998	1.386	0.898
ILI	24	2.180	0.858	3.110	1.179	4.265	1.387	4.975	1.660	**2.066**	**0.854**	2.398	1.040	2.294	0.945	3.483	1.287	5.764	1.677	4.644	1.434
	36	2.214	**0.877**	3.429	1.222	4.777	1.496	5.322	1.659	2.173	0.881	2.646	1.088	**1.825**	0.848	3.103	1.148	4.755	1.467	4.533	1.442
	48	2.068	**0.876**	3.451	1.203	5.333	1.592	5.425	1.632	2.032	0.892	2.614	1.086	**2.010**	0.900	2.669	1.085	4.763	1.469	4.758	1.469
	60	1.991	**0.898**	3.678	1.255	5.070	1.552	5.477	1.675	1.987	0.899	2.804	1.146	2.178	0.963	2.770	1.125	5.264	1.564	5.199	1.540
	Avg	2.113	**0.877**	3.417	1.215	4.861	1.507	5.300	1.657	2.065	0.882	2.616	1.090	2.077	0.914	3.006	1.161	5.137	1.544	4.784	1.471
1st Count		60		3		0		0		6		2		2		0		0		0	

$*$ means that there are some mismatches between our input-output setting and their papers. We adopt their official codes and only change the length of input and output sequences for a fair comparison.