REPRESENTING MULTI-VIEW TIME-SERIES GRAPH STRUCUTRES FOR MULTIVARIATE LONG-TERM TIME-SERIES FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Multivariate long-term time-series forecasting task is a very challenging task in real-world application areas, such as electricity consumption and influenza-like illness forecasting. At present, researchers are focusing on designing robust and effective models, and have achieved good results. However, there are several issues with existing models that need to be overcome to ensure they provide optimal performance. First, the lack of a relationship structure between multivariate variables needs to be addressed. Second, most models only have a weak ability to capture local dynamic changes across the entire long-term time-series. And, third, the current models suffer from high computational complexity and unsatisfactory accuracy. To address these issues, we propose a novel method called Multi-view Time-series Graph Structure Representation (MTGSR) for multivariate long-term time-series forecasting tasks. MTGSR uses graph convolutional networks (GCNs) to construct topological relationships in the multivariate long-term time-series from three different perspectives: time, dimension, and crossing segments. Variation trends in the different dimensions of the multivariate long-term time-series are extracted through a difference operation so as to construct a topological map that reflects the correlations between the different dimensions. Then, to capture the dynamically changing characteristics of the fluctuation correlations between adjacent local sequences, MTGSR constructs a cross graph by calculating the correlation coefficients between adjacent local sequences. Extensive experiments on five different datasets show that MTGSR reduces errors by 20.41% over the state-of-the-art while maintaining linear complexity. Additionally, memory use is decreased by 66.52% and running time is reduced by 78.09%.

1 INTRODUCTION

In reality, a large amount of time-series data is produced in various fields, such as weather forecasting (Hewage et al., 2021; Rasp et al., 2020), electricity power planning (Qader et al., 2022; Oreshkin et al., 2021), disease propagation prejudgment (Li et al., 2021; Zimmer & Yaesoubi, 2020), and more. Although challenging to model the long-term relationships and multivariate correlations within these real-world time-series are important elements of most practical forecasting tasks involving these data. Thus, in this paper, we focus on multivariate long-term time-series forecasting task, which has higher requirements for models than ordinary time-series forecasting tasks. In recent years, deep learning models have been thoroughly investigated for their power at multivariate long-series forecasting tasks with many achieving good results (Liu et al., 2021; Torres et al., 2021; Lim & Zohren, 2021). For example, Transformer-based models, the mainstream framework for multivariate long-term time-series forecasting tasks, relies on multi-head self-attention as a core mechanism for extracting powerful characteristics from historical data (Nikita et al., 2020; Zhou et al., 2021; Xu et al., 2021; Zhou et al., 2022). These characteristics are then analyzed to predict long sequences containing data from farther in the future.

However, there are still several extremely challenging issues in multivariate long-term time-series forecasting tasks that need to be addressed. First, existing models do not construct relationships between multivariate variables. Rather, they pay more attention to capturing the temporal features of the series, which means they simply use dimensional mappings to extract blurry relationships

between multivariate variables. Topologies between different variables cannot effectively be constructed using this approach. Second, in addition to the relationship between tokens, the characteristics of dynamically changing fluctuations between local sequences in long-term time-series are also important. Yet most existing models process sequences from a global view such that the features of the local fluctuations are entangled with the overall features. Third, the current models still have room to improve accuracy. And, further, most of the models that perform well have a high computational complexity caused by complex structures.

To construct a relationship graph of multivariate variables, we turned to graph convolutional networks (GCNs) (Welling & Kipf, 2016). GCNs are typical graph neural networks used to extract the features of vertices connected by edges in a graph. One advantage of GCNs is that they can generate a more representative topology and richer node properties by passing information between neighboring nodes. Hence, we attempted to build a model by treating the multivariate variables in a multivariate long-term time-series as the nodes of a graph and using the correlations between different variables fluctuating over time as the weights of the graph's edges. Through experiments, we found that these operations could generate dimensional graphs with rich spatial features. Moreover, the GCNs could be used to extract more appropriate topological features between multivariate variables from the obtained dimensional graphs. Additionally, we subsequently found that the process of generating graphs from the time-series and the GCNs was also good for extracting several other graph characteristics-including the temporal characteristics of the long-term time-series and the characteristics of the local sub-sequences with dynamically changing fluctuations.

Inspired by these preliminary studies, we developed a novel and effective model named Multi-view time-series Graph Structure Representation (MTGSR) for multivariate long-term time-series forecasting tasks. MTGSR extracts disentangled information from the input time-series to dynamically generate graph structures from three perspectives: the time-view, the dimension-view, and a cross-view. In terms of the time-view, MTGSR builds a time graph using Time Graph Generator by calculating the correlations for all the dimensional information between different timestamps. Unlike the normal process of generating a time graph, MTGSR adds a differential operation to process the inputs when generating dimensional graphs so as to extract valid information from the relative fluctuations of the time-series across different dimensions. The cross-view takes into account the correlations of all the fluctuations that dynamically change over time between two adjacent local sequences in the long-term time-series. Because extracting features directly from the entire length of the time-series would result in too many redundant information and probably cause the model to overfit, the objects MTGSR's three graph generators use are local sequences split from the intact multivariate long-term time-series. Benefiting from this design, the scale of the parameters is greatly reduced to the point that the model has a linear complexity. Further, Inspired by Transformer's multi-head attention mechanism, a multi-head mechanism is used at MTGSR's input stage to improve its ability to capture different features from the input sequence. This strategy proves to increase the prediction accuracy of the model. In fact, MTGSR outperforms the state-of-the-art model on five data benchmarks in terms of accuracy, memory use, and running times. The contributions of this paper are summarized as follows:

- We propose a novel model named Multi-view Time-series Graph Structure Representation (MTGSR) for multivariate long-term time-series forecasting tasks. MTGSR uses GCNs to learn the complex disentangled characteristics in multivariate long-term time-series from three perspectives: the time view, the dimension view, and the cross-segment view.
- To construct topologies between multivariate variables, MTGSR uses a GCN-based Dimension Graph Generator to dynamically learn the structural relationships in the multivariate long-term time-series after differencing operations.
- To capture the dynamically changing characteristics of the fluctuation correlations between adjacent local sequences in the whole long-term time-series, MTGSR construct a cross-segment graph by calculating the correlation coefficients between adjacent local sequences through the Cross-segments Graph Generator.
- Extensive experiments with five datasets show that MTGSR reduces errors by 20.41% while maintaining a linear complexity compared to the state-of-the-art framework FEDformer. Additionally, MTGSR reduces memory use by 66.52% and running time by 78.09%.

2 RELATED WORK

2.1 DEEP LEARNING MODELS FOR MULTIVARIATE LONG-TERM TIME-SERIES FORECASTING

Multivariate long-term time-series forecasting is an important research direction in the field of time prediction (Liu et al., 2021; Torres et al., 2021). In the original field of time-series forecasting, recurrent neural networks (RNNs) (Stankeviciute et al., 2021; Qin et al., 2017; Madan & Mangipudi, 2018) are one of the more widely used deep learning models. As the requirements for the forecasting task increase, the length of the time-series that models need to predict is growing longer. For this reason, an enhanced version of the RNN, LSTM, has been used to model long-term time-series prediction tasks. However, LSTM-based models (Smyl, 2020; Sagheer & Kotb, 2019; Shen et al., 2020), which iteratively generate prediction sequences, produce cumulative errors and the errors affect the models' prediction accuracy. As a way to address this problem, temporal convolution networks (TCNs) have emerged (Wan et al., 2019; Shen et al., 2020). These frameworks directly generate all prediction sequences by imitating the principles of a convolution neural network (CNN).

In recent years, Transformer (Vaswani et al., 2017), a model with stronger theoretical advantages, has shown great power over tasks such as audio processing (Gong et al., 2022; Sajid et al., 2021), natural language processing (Wolf et al., 2020; Guo et al., 2019; Zhang & Zhang, 2020), and time-series forecasting (Wu et al., 2020; Li et al., 2019; 2021). However, limited by the high computational complexity $\mathcal{O}(L^2)$ of self-attention mechanisms that sits at their core, Transformer-based models cannot be used directly to handle long-term time-series. Some Transformer variants do focus on reducing the computational complexity of the self-attention mechanism. For instance, LogTrans (Li et al., 2019)incorporate a sparse attention mechanism named LogSparse attention, which reduces the model's computational complexity to $\mathcal{O}(LlogL)$. Reformer (Nikita et al., 2020) presents a novel local-sensitive hashing (LSH) attention based on a hash algorithm, which also has $\mathcal{O}(LlogL)$ complexity. In addition, Informer (Zhou et al., 2021) introduces a query sparsity measurement with a distilling mechanism that yields low computational complexity. However, these variant models do not perform well enough in terms of prediction accuracy. Compared with the above models, Autoformer (Xu et al., 2021) is much more accurate. Autoformer includes a time-series decomposition module and an AutoCorrelation mechanism in place of self-attention. However, with the advent of FEDformer (Zhou et al., 2022), the prediction accuracy of the Transformer variants has risen again. FEDformer combines a time-series decomposition module and frequency enhanced blocks to greatly improve prediction accuracy. Moreover, it reduces computational complexity to $\mathcal{O}(L)$.

However, when extracting temporal features from long-term time-series, none of these models consider the relationships between multiple variables from a dimensional perspective. Yet there are meaningful relationships between different variables in most real-world datasets, and extracting these relationships is an important part of studying multivariate long-term time-series prediction.

2.2 GRAPH CONVOLUTIONAL NETWORKS

Graph convolutional networks (GCNs) (Welling & Kipf, 2016) are representative graph neural networks that have been widely used in link prediction (Yun et al., 2021; Yan et al., 2021), social networks(Tong, 2020; Tian et al., 2021b) and graph anomaly detection (Tian et al., 2021a; Markovitz et al., 2020). GCNs can aggregate the information from adjacent nodes and filter out interference in a graph. Thus, GCNs have an advantage when processing graph structure data. Today, researchers can dynamically generate graphs from time-series data by building graph structure learning models (Zhao et al., 2021b; Fatemi et al., 2021), where GCNs are used to extract the characteristics of the processed data and the generated graph structures. This has expanded the applicability of GCNs to a wider range of tasks, such as traffic forecasting (Guo et al., 2021; Sofianos et al., 2021; Lan et al., 2022), health data processing (Zhao et al., 2021a; Ntemi et al., 2022; Zhang et al., 2022), and so on. Particularly in short-term forecasting tasks, such as traffic forecasting, GCNs have performed very well as mainstream model frameworks. But, to the best of our knowledge, no GCN-based model has currently been designed for multivariate long-term time-series forecasting tasks given multi-domain datasets.

In multivariate long-term time-series forecasting tasks, existing models lack the ability to extract dependencies between multivariate variables from multivariate long-term time-series. Most of these

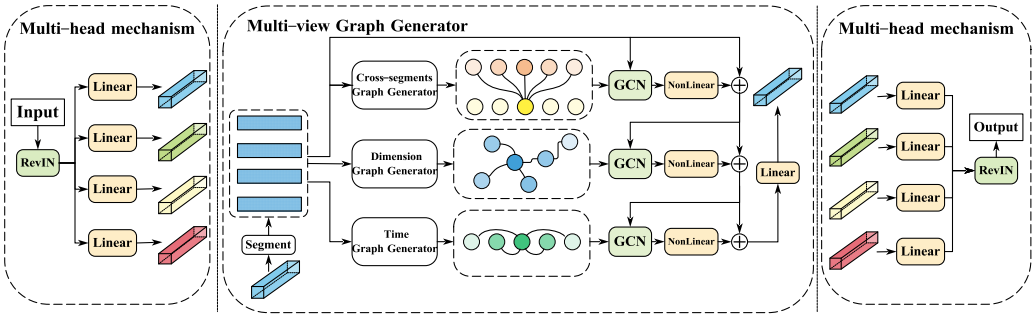


Figure 1: Multi-view Time-series Graph Structure Representation architecture. The multi-head mechanism provides stable and informative hidden features to the Multi-view Graph Generator module. The Multi-view Graph Generator module extracts the disentangled characteristics of the time stamps, the multivariate variables, and the crossing segments.

models focus their attention on extracting features in the time and frequency domains of the time-series. By contrast, our framework is designed to use GCNs from a dimensional perspective to construct the complex relationships between the multivariate variables in multivariate long-time time-series.

3 METHODOLOGY

The task with multivariate long-term time-series forecasting is to predict long-term future sequences $\boldsymbol{P} \in \mathbb{R}^{L_p \times d_{in}}$ with minimal errors based on historical time-series data $\boldsymbol{X} \in \mathbb{R}^{L_x \times d_{in}}$. As mentioned, several problems currently exist in multivariate long-term time-series forecasting-these being: 1) the lack of a relationship structure between multivariate variables; 2) weak ability to capture local dynamic changes across the full long-term time-series; and 3) high computational complexity and unsatisfactory accuracy. To address these problems, our framework incorporates GCNs that process dynamically-constructed graphs with topological information from three different perspectives: the time view, the dimension view, and the cross-segments view. Inspired by multi-head attention, we designed a multi-head mechanism to improve the model's ability to capture different features. Within this mechanism, RevIN (Kim et al., 2021) improves the model's prediction accuracy with datasets of different distributions. Lastly, we propose Multi-view Time-series Graph Structure Representation (MTGSR), a model with high-precision and low complexity. The overall architecture of our framework is shown in Figure 1.

3.1 MULTI-HEAD MECHANISM

The multi-head mechanism consists of two modules: RevIN and a parallel linear layer group. This mechanism is designed to provide stable and informative features for subsequent graph generators, with the detailed structure shown in Figure 1.

In a real world dataset, the distributions of the overall data differ over time. Hence, the RevIN module (Kim et al., 2021) dynamically normalizes the input sequence. This process means that input sequences normalized by RevIN all conform to the same distribution each time, which improves the effectiveness of the features extracted by the model. The formulation is:

$$\boldsymbol{X}_n = RevIN(\boldsymbol{X}) \tag{1}$$

where $\boldsymbol{X} = \{\boldsymbol{x}^1, \ldots, \boldsymbol{x}^{L_x} | \boldsymbol{x}^i \in \mathbb{R}^{d_{in}}\}$ denotes the input sequence, $RevIN(\cdot)$ denotes the function of RevIN module and $\boldsymbol{X}_n$ denotes the hidden layer sequence normalized by the RevIN module.

To extract informative features, a parallel linear layer group replaces a single linear layer, with reference to the principle of the multi-head attention mechanism in Transformer (Vaswani et al., 2017). This mechanism enhances the model's ability to discover the information in $\boldsymbol{X}_n$ from different representation subspaces at different positions. Formally, it is expressed as:

$$\boldsymbol{H}_i = \boldsymbol{X}_n \boldsymbol{W}_i + b_i \tag{2}$$

where $\boldsymbol{H}_i \in \mathbb{R}^{L_x \times d}$ denotes the hidden features, and $\boldsymbol{W}_i \in \mathbb{R}^{d_{in} \times d}$ represents the learnable parameter matrix of the $i\text{-}th$ head in hidden layer. b_i is the corresponding learnable bias.

3.2 MULTI-VIEW GRAPH GENERATOR

The multi-view graph generator module extracts the characteristics of the multivariate long-term time-series from multiple perspectives. It contains three graph generators: including Crosssegments Graph Generator, Dimension Graph Generator and Time Graph Generator. The module relies on GCNs to process the graphs generated by the three graph generators so as to obtain the features that contain different subspace information from the input sequence.

From experiments, we found that learning the relationship between each pair of tokens across the entire multivariate long-term time-series resulted in an oversized and overfit model. So, to overcome this problem, we split the input sequence into several cross segments of the same size l, where adjacent segments partially cover each other. The process is formulated as:

$$\boldsymbol{S}_1, \boldsymbol{S}_2, \ldots, \boldsymbol{S}_n = Split(\boldsymbol{H}_i) \tag{3}$$

where $\boldsymbol{S}_j \in \mathbb{R}^{l \times d}$ represents the $j\text{-}th$ segment and $Split(\cdot)$ represents the splitting function. The overall architecture of Multi-view Graph Generator module is shown in Figure 1. Details of the three generators are provided in the following sections and the architectures of them are shown in Appendix A.

3.2.1 CROSS-SEGMENTS GRAPH GENERATOR

To learn the variations in the fluctuations of the local sub-sequences, we designed the Cross-segments Graph Generator to dynamically generate a relationship graph between two sub-sequences. This generator first projects two adjacent segments onto the feature space through a linear mapping function. Then, the features are normalized. The formulation is:

$$\begin{aligned} \boldsymbol{F}_j &= Norm(\boldsymbol{W}_{c1}\boldsymbol{S}_j + b_{c1}) \\ \boldsymbol{F}_{j+1} &= Norm(\boldsymbol{W}_{c2}\boldsymbol{S}_{j+1} + b_{c2}) \end{aligned} \tag{4}$$

where $\boldsymbol{W}_{c1}, \boldsymbol{W}_{c2} \in \mathbb{R}^{l \times l}$ are two learnable weight matrices, and b_{c1} and b_{c2} are two learnable bias. $Norm(\cdot)$ represents the normalization function. All the learnable parameters are shared in each Cross-segments Graph Generator. The generator then calculates the correlation value of the two feature sequences $\boldsymbol{F}_j$ and $\boldsymbol{F}_{j+1}$ to construct a relationship graph, which is formulated as:

$$\boldsymbol{G}_{cross} = Softmax2d(\boldsymbol{F}_j \boldsymbol{F}_{j+1}^T) \tag{5}$$

where $\boldsymbol{G}_{cross} \in \mathbb{R}^{L \times L}$ is the relationship graph of the feature sequences $\boldsymbol{F}_j$ and $\boldsymbol{F}_{j+1}$. To further extract the information on the graph $\boldsymbol{G}_{cross}$ and update the information of the corresponding elements in the feature sequence $\boldsymbol{F}_j$, a GCN is used to process the graph $\boldsymbol{G}_{cross}$ and feature sequence $\boldsymbol{F}_j$. The formulation is:

$$\boldsymbol{F}'_j = \tilde{\boldsymbol{D}}^{-\frac{1}{2}} \tilde{\boldsymbol{G}}_{cross} \tilde{\boldsymbol{D}}^{-\frac{1}{2}} \boldsymbol{F}_j \boldsymbol{W}_{cross} \tag{6}$$

where $\boldsymbol{F}'_j$ is the updated feature sequence, $\tilde{\boldsymbol{D}}^{-\frac{1}{2}} \tilde{\boldsymbol{G}}_{cross} \tilde{\boldsymbol{D}}^{-\frac{1}{2}}$ is the normalized adjacency matrix of $\boldsymbol{G}_{cross}$ and $\boldsymbol{W}_{cross}$ is a learnable matrix. In our experiment, we simplify Equation 6 to Equation 7 to get more efficient with almost no effect loss of accuracy.

$$\boldsymbol{F}'_j = \boldsymbol{G}_{cross} \boldsymbol{F}_j \boldsymbol{W}_{cross} \tag{7}$$

$\boldsymbol{F}'_j$ is then feature-mapped by a nonlinear layer using a residual connection (He et al., 2016) followed by another nonlinear layer. This formulated as:

$$\boldsymbol{F}_{cross} = \boldsymbol{F}'_j + \sigma(\boldsymbol{F}'_j \boldsymbol{W}_{nlc} + b_{nlc}) \tag{8}$$

where $\boldsymbol{F}_{cross} \in \mathbb{R}^{l \times d}$ is the updated feature sequence, σ is a activation function, $W_{nlc} \in \mathbb{R}^{d \times d}$ and b_{nlc} are the learnable parameters of the nonlinear layer.

3.2.2 DIMENSION GRAPH GENERATOR

The Dimension Graph Generator dynamically generates a dimension graph that represents the correlation coefficients of each pair of multivariate variables. This generator takes the resulting feature sequence $\boldsymbol{F}_{cross}$ (Section 3.2.1) as its input. The next step is to perform a differential operation on $\boldsymbol{F}_{cross}$ to establish the trend of fluctuations in the series. The formulation is:

$$\boldsymbol{F}_d = Diff(\boldsymbol{F}_{cross}) \tag{9}$$

where $\boldsymbol{F}_d \in \mathbb{R}^{(l-1)\times d}$ is the output of the differential operation $Diff(\cdot)$. Then, through a linear mapping and normalization operation, the feature sequence $\boldsymbol{F}_d$ is updated to $\boldsymbol{F}_n$, formulated as:

$$\boldsymbol{F}_n = Norm(\boldsymbol{W}_d\boldsymbol{F}_d + b_d) \tag{10}$$

where $\boldsymbol{W}_d \in \mathbb{R}^{(l-1)\times(l-1)}$ and b_d are two learnable parameters. Next, the correlation coefficients of each dimension are calculated in the feature sequence $\boldsymbol{F}_n$, and a dimension graph is constructed.

$$\boldsymbol{G}_{dim} = Softmax2d(\boldsymbol{F}_n^T\boldsymbol{F}_n) \tag{11}$$

where $\boldsymbol{G}_{dim} \in \mathbb{R}^{d\times d}$ is the dimension graph and $Softmax2d$ is a normalization function. Finally, a GCN is used to extract information from the dimension graph $\boldsymbol{G}_{dim}$ to update the feature sequence $\boldsymbol{F}_{cross}$. The formulation is:

$$\begin{aligned} \boldsymbol{F}'_d &= \boldsymbol{G}_{dim}\boldsymbol{F}_{cross}\boldsymbol{W}_{dim} \\ \boldsymbol{F}_{dim} &= \boldsymbol{F}'_d + \sigma(\boldsymbol{F}'_d\boldsymbol{W}_{nld} + b_{nld}) \end{aligned} \tag{12}$$

where $\boldsymbol{F}_{dim} \in \mathbb{R}^{l\times d}$ is the updated feature sequence, $\boldsymbol{W}_{dim}, \boldsymbol{W}_{nld} \in \mathbb{R}^{d\times d}$ and b_{nld} are the learnable parameters.

3.2.3 TIME GRAPH GENERATOR

The Time Graph Generator is a structure that extracts the relationship between each timestamp from the perspective of time. We contend that the values for each timestamp have an important relationship in this perspective, so, unlike the Dimension Graph Generator, the Time Graph Generator does not calculate the difference. The formulation is:

$$\begin{aligned} \boldsymbol{F}_t &= Norm(\boldsymbol{W}_t\boldsymbol{F}_{dim} + b_t) \\ \boldsymbol{G}_{time} &= Softmax2d(\boldsymbol{F}_t\boldsymbol{F}_t^T) \end{aligned} \tag{13}$$

where $\boldsymbol{G}_{time} \in \mathbb{R}^{l\times l}$ is the obtained time graph, and $\boldsymbol{W}_t \in \mathbb{R}^{l\times l}$ and b_t are the learnable parameters. A GCN is then used to update the feature sequence, which is formulated as:

$$\begin{aligned} \boldsymbol{F}'_t &= \boldsymbol{G}_{time}\boldsymbol{F}_{dim}\boldsymbol{W}_{time} \\ \boldsymbol{F}_{time} &= \boldsymbol{W}_o(\boldsymbol{F}'_t + \sigma(\boldsymbol{F}'_t\boldsymbol{W}_{nlt} + b_{nlt})) + b_o \end{aligned} \tag{14}$$

where $\boldsymbol{F}_{time} \in \mathbb{R}^{l\times d}$ is the updated feature sequence, and $\boldsymbol{W}_{time}, \boldsymbol{W}_{nlt} \in \mathbb{R}^{d\times d}, \boldsymbol{W}_o \in \mathbb{R}^{l\times l}$, b_o and b_{nlt} are the learnable parameters.

Finally, the feature sequences produced in parallel by the multi-head mechanism are combined to give the final prediction sequence. The formulation is:

$$\begin{aligned} \boldsymbol{S}'_j &= \boldsymbol{W}_j\boldsymbol{F}_j^{time} \\ \boldsymbol{P} &= RevIN(Concat(\boldsymbol{S}'_1, \boldsymbol{S}'_2, \ldots, \boldsymbol{S}'_{n'})) \end{aligned} \tag{15}$$

where $\boldsymbol{P} \in \mathbb{R}^{L_p\times din}$ is the prediction sequence, $\boldsymbol{F}_j^{time}$ represents the updated feature sequence of $j\text{-}th$ head, $\boldsymbol{W}_j$ is the learnable matrix of $j\text{-}th$ head, and $Concat(\cdot)$ are the concatenation function.

4 EXPERIMENTS

To evaluate MTGSR's performance, we performed extensive experiments on five publicly available datasets covering different fields. Additionally, we selected five baselines for comparison.

Methods		MTGSR		FEDformer		Autoformer		Informer		LogTrans		Reformer	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm1	96	**0.349**	**0.385**	0.379	0.419	0.505	0.475	0.672	0.571	0.600	0.546	0.538	0.528
	192	**0.380**	**0.402**	0.426	0.441	0.553	0.496	0.795	0.669	0.837	0.700	0.658	0.592
	336	**0.416**	**0.425**	0.445	0.459	0.621	0.537	1.212	0.871	1.124	0.832	0.898	0.721
	720	**0.472**	**0.458**	0.543	0.490	0.671	0.561	1.166	0.823	1.153	0.820	1.102	0.841
Exchange	96	**0.095**	**0.218**	0.139	0.276	0.134	0.270	0.847	0.752	0.968	0.812	1.065	0.829
	192	**0.178**	**0.302**	0.256	0.369	0.272	0.374	1.204	0.895	1.04	0.851	1.188	0.960
	336	**0.331**	**0.416**	0.426	0.464	0.488	0.510	1.672	1.036	1.659	1.081	1.357	0.976
	720	**0.936**	**0.693**	1.090	0.800	1.367	0.901	2.478	1.310	1.941	1.127	1.510	1.016
Weather	96	**0.158**	**0.208**	0.217	0.296	0.231	0.312	0.300	0.384	0.458	0.490	0.689	0.596
	192	**0.205**	**0.254**	0.276	0.336	0.278	0.343	0.598	0.544	0.658	0.589	0.752	0.638
	336	**0.263**	**0.309**	0.339	0.380	0.335	0.378	0.578	0.523	0.797	0.652	0.639	0.596
	720	**0.345**	**0.341**	0.403	0.428	0.429	0.436	1.059	0.741	0.869	0.675	1.130	0.792
Electricity	96	**0.170**	**0.249**	0.193	0.308	0.197	0.312	0.274	0.368	0.258	0.357	0.312	0.402
	192	**0.178**	**0.282**	0.201	0.315	0.208	0.321	0.296	0.386	0.266	0.368	0.348	0.433
	336	**0.187**	**0.293**	0.214	0.329	0.213	0.328	0.300	0.394	0.280	0.380	0.350	0.433
	720	**0.217**	**0.317**	0.246	0.355	0.245	0.352	0.373	0.439	0.283	0.376	0.340	0.420
Illness	24	**1.516**	**0.765**	2.203	0.963	3.825	1.345	4.388	1.360	4.322	1.381	4.400	1.382
	36	**1.402**	**0.757**	2.272	0.976	3.319	1.216	4.651	1.391	4.186	1.332	4.783	1.448
	48	**1.455**	**0.778**	2.209	0.981	2.854	1.122	4.581	1.419	4.476	1.411	4.832	1.465
	60	**1.740**	**0.875**	2.545	1.061	3.227	1.232	4.583	1.432	4.766	1.477	4.882	1.483

Table 1: Multivariate long-term time-series forecasting results on five datasets with an input length of I = 96 and a prediction length of $O \in \{96, 192, 336, 720\}$. Note that with the ILI dataset, we used a prediction length of $O \in \{24, 36, 48, 60\}$). A lower MSE indicates better performance; the best results are highlighted in bold.

4.1 DATASETS

In this section, we show the description of the five datasets: 1) ETT , which contains seven attributes, such as load and oil temperature, collected every 15 minutes from electricity transformers between July 2016 and July 2018. 2) Exchange , with eight attributes, which records the daily exchange rate from eight countries between 1990 to 2016. 3) Weather , containing 21 weather-related attributes collected every 10 minutes for the whole of 2020. 4) Electricity , which consists of 321 customers and records their hourly electricity consumption between 2012 and 2014. 5) ILI , containing seven patient attributes, collected by the Centers for Disease Control and Prevention of the United States between 2002 and 2021. These datasets were split into training, validation, and testing sets for experimentation according to a 7:1:2 ratio.

4.2 IMPLEMENTATION DETAILS

We chose L2 as the loss function to train the model and selected mean square error (MSE) and mean absolute error (MAE) as the evaluation metrics. We used the ADAM optimizer with an initial learning rate of 0.001. The batch size was set to 64. The number of heads in the multi-head mechanism was set to 8, and the hidden dimension in each head of the multi-head mechanism was set to 64. All experiments were repeated three times, and their average values were recorded as the result. The size of segments in MTGSR were set to 12 for the ETT, Electricity, and Illness datasets, and to 24 for the Exchange and Weather datasets. A hyperparameter sensitivity analysis is provided in Section 4.5.1. All experiments were run on a single Nvidia RTX3090 24GB GPU.

4.3 BASELINE

We selected five of the latest state-of-the-art methods as baselines to compare with MTGSR, including FEDformer Zhou et al. (2022), Autoformer Xu et al. (2021), Informer Zhou et al. (2021), LogTrans Li et al. (2019) and Reformer Nikita et al. (2020).

4.4 MAIN RESULTS

We set the input sequence to a fixed length and evaluated the performance of the proposed MTGSR and baselines over four prediction lengths with the ETTm1, Exchange, Weather and Electricity

Size of Segments	Metric	Exchange				Electricity			
		96	192	336	720	96	192	336	720
12	MSE	0.103	0.195	0.360	1.106	**0.170**	**0.178**	**0.187**	**0.217**
	MAE	0.220	0.316	0.429	0.798	**0.246**	**0.282**	**0.293**	**0.317**
24	MSE	**0.095**	**0.178**	**0.331**	**0.936**	0.171	0.184	0.192	0.223
	MAE	**0.218**	**0.302**	**0.416**	**0.693**	0.279	0.290	0.299	0.321
48	MSE	0.096	0.187	0.350	1.051	0.175	0.185	0.193	0.227
	MAE	0.219	0.309	0.430	0.767	0.286	0.294	0.301	0.323

Table 2: The experimental results of MTGSR with segment sizes of 12, 24, and 48 on the Exchange and Electricity datasets.

datasets: 96, 192, 336, and 720 and to 24, 36, 48, and 60 with the Illness dataset. Table 1 shows the results of the experiments. MTGSR yielded better results than the state-of-the-arts in all benchmarks and all prediction length settings. MTGSR's improvements were particularly pronounced on the datasets with strong correlations between multivatiate attributions, such as the Exchange, Weather and ILI datasets, at **24.54%**, **26.41%**, and **31.78%**, respectively. On the other two datasets ETTm1 and Electricity datasets, MTGSR also showed good performance with a respective **9.58%** and **11.94%** reduction in MSE. Overall, MTGSR reduced errors by an average of **20.41%** across all experiments. These results demonstrate that no matter whether the prediction is short-term (at a prediction length of 96) or long-term (a prediction length of 720), MTGSR's performance is relatively stable. At the same time, MTGSR maintains high precision while remaining low in computational complexity and in terms of model scale, making it easier to migrate MTGSR to edge devices for deployment. Section 4.5.3 next contains details of the hyperparameter sensitivity, the complexity analysis, and the model scale.

4.5 Model Analysis

In this section, we show the hyperparameter sensitivity analysis, an ablation study of each graph generator and the efficiency analysis. The prediction sequences and heatmaps of each view of MTGSR are visualized in Appendix B. Results of some expanded experiments on the ETT series datasets are shown in Appendix C.

4.5.1 Hyperparameter Sensitivity

We performed extended experiments to study MTGSR's hyperparameters, including the size of the segments for the Multi-view Graph Generator and the number of heads in the multi-head mechanism. In terms of the segment size, we selected three values - 12, 24 and 48 - and performed experiments with the Exchange and Electricity datasets. The results are shown in Table 2. With the Exchange dataset, the best-performing hyperparameter setting was 24. With the Electricity dataset, the optimal segment sizeb was 12. Hence, from these experiments, we determined that segment size needs to be selected depending of the datasets. With the remaining datasets, the optimal segment size was 12 for the ETTand Illness datasets, and 24 for the Weather dataset. In terms of the number of heads in the multi-head mechanism, we performed experiments with the Exchange dataset, setting the number of heads to 1,4,8, and 16. Table 3 shows the results, indicating that MTGSR performed best with 8 heads.

Head	Metric	Exchange			
		96	192	336	720
1	MSE	0.097	0.188	0.373	1.037
	MAE	0.221	0.312	0.442	0.772
4	MSE	0.098	0.183	0.359	1.005
	MAE	**0.218**	0.310	0.432	0.754
8	MSE	**0.095**	**0.178**	**0.331**	**0.936**
	MAE	**0.218**	**0.302**	**0.416**	**0.693**
16	MSE	0.103	0.198	0.363	0.973
	MAE	0.225	0.316	0.435	0.744

Table 3: The experimental results of MTGSR with head number of 1, 4, 8, 16 on Exchange datasets.

4.5.2 The Effect of Dimension Graph and Cross-segments Graph

To verify the effectiveness of the dimension graph and the cross-segments graph, we conducted ablation experiments. We removed the Dimension Graph Generator and the Cross-segments Graph Generator to get two variants: MTGSR† and MTGSR‡. Compared to MTGSR, MTGSR† and MTGSR‡ decreased prediction accuracy by 8.86% and 8.56%, respectively, and the results are shown in Table 4. After removing the dimension graph, there was a large decline with the Exchange and Weather

Methods	Metric	Exchange				Weather				Electricity				ILI			
		96	192	336	720	96	192	336	720	96	192	336	720	24	36	48	60
MTGSR	MSE	**0.095**	**0.178**	**0.331**	**0.936**	**0.158**	**0.209**	**0.263**	**0.345**	**0.170**	**0.178**	**0.187**	**0.217**	**1.516**	**1.402**	**1.455**	**1.740**
	MAE	**0.218**	**0.302**	**0.416**	**0.693**	**0.208**	**0.254**	**0.309**	**0.341**	**0.246**	**0.282**	**0.293**	**0.317**	**0.765**	**0.757**	**0.778**	**0.875**
MTGSR†	MSE	0.110	0.191	0.407	1.179	0.179	0.232	0.286	0.369	0.175	0.189	0.199	0.235	1.627	1.444	1.610	1.850
	MAE	0.236	0.315	0.461	0.819	0.217	0.264	0.305	0.351	0.284	0.295	0.305	0.334	0.765	0.764	0.800	0.871
MTGSR‡	MSE	0.098	0.182	0.398	1.128	0.178	0.227	0.284	0.366	0.177	0.181	0.202	0.226	1.768	1.615	1.615	1.920
	MAE	0.220	0.304	0.457	0.793	0.213	0.262	0.303	0.354	0.287	0.286	0.309	0.327	0.796	0.800	0.809	0.911

[1] MTGSR†: MTGSR removes Dimension Graph Generator.

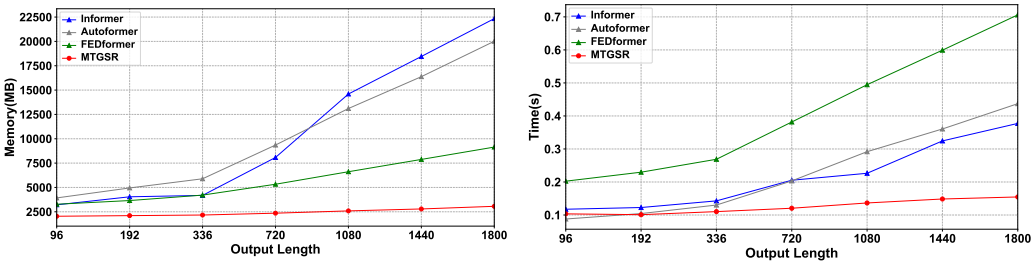
[2] MTGSR‡: MTGSR removes Cross-segments Graph Generator.

Table 4: Ablation experiments with MTGSR, MTGSR† and MTGSR‡ on four dataset: Exchange, Weather, Electricity and ILI. The best results appear in bold; suboptimal results are underlined.

datasets, with decreases in accuracy of 14.47% and 9.05%, respectively. This is because, there is a strong relationship between the multivariate variables on both the Exchange and Weather datasets. With the ILI dataset, there is a strong relationship between the fluctuations of local adjacent segments; hence, MTGSR‡'s prediction accuracy without the cross-segment graph was greatly reduced, decreasing by 11.68%. Through the experimental results, the dimension graph and cross-segments graph have different degrees of improvement depending on the characteristics of the dataset.

4.5.3 Efficiency Analysis

To prove the efficiency of MTGSR, we performed extensive experiments comparing the model size and runtime of MTGSR to the baselines Informer, Autoformer, and FEDformer. Figure 2a shows that MTGSR is a linear model with a greatly reduced model size, compared to FEDformer and the other models. In the case of a prediction length of 1800, MTGSR reduced memory use by **66.52%** over FEDformer. Figure 2b shows the comparison between the runtimes of the four models for one iteration. Compared to FEDformer, MTGSR ran **78.09%** faster at a prediction length of 1800.



(a) Memory Efficieny Analysis

(b) Running Time Efficiency Analysis

Figure 2: Efficiency Analysis. In both analyses, we perform the experiments at seven prediction lengths. The results show that MTGSR used significantly less memory and has a faster running time than the current state-of-the-arts.

5 Conclusion

In this paper, we proposed a novel and efficient model named Multi-view Time-series Graph Structure Representation (MTGSR) with a linear computational complexity for multivariate long-term time-series forecasting problems. To more comprehensively extract disentangled characteristics from multivariate long-term time-series, MTGSR uses GCNs to learn the characteristics from three perspectives. MTGSR generates dimension graphs and cross-segment graphs to learn the structural relationships between multivariate variables as well as the dynamically changing characteristics of the fluctuation correlations between adjacent local sequences. In a comprehensive series of experiments, MTGSR outperforms the current state-of-the-art models and exhibits lower memory usage and faster running speeds than the state-of-the-art models. In the future, we will continue to study applications for graph neural networks that pertain to multi-view graph construction problems and multivariate long-term time-series.

REFERENCES

Bahare Fatemi, Layla El Asri, and Seyed Mehran Kazemi. Slaps: Self-supervision improves structure learning for graph neural networks. *Advances in Neural Information Processing Systems*, 34: 22667–22681, 2021.

Yuan Gong, Cheng-I Lai, Yu-An Chung, and James Glass. Ssast: Self-supervised audio spectrogram transformer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 10699–10709, 2022.

Kan Guo, Yongli Hu, Yanfeng Sun, Sean Qian, Junbin Gao, and Baocai Yin. Hierarchical graph convolution network for traffic forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 151–159, 2021.

Maosheng Guo, Yu Zhang, and Ting Liu. Gaussian transformer: a lightweight approach for natural language inference. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 6489–6496, 2019.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Pradeep Hewage, Marcello Trovati, Ella Pereira, and Ardhendu Behera. Deep learning-based effective fine-grained weather forecasting model. *Pattern Analysis and Applications*, 24(1):343–366, 2021.

Taesung Kim, Jinhee Kim, Yunwon Tae, Cheonbok Park, Jang-Ho Choi, and Jaegul Choo. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International Conference on Learning Representations*, 2021.

Shiyong Lan, Yitong Ma, Weikang Huang, Wenwu Wang, Hongyu Yang, and Pyang Li. Dstagnn: Dynamic spatial-temporal aware graph neural network for traffic flow forecasting. In *International Conference on Machine Learning*, pp. 11906–11917. PMLR, 2022.

Liang Li, Yuewen Jiang, and Biqing Huang. Long-term prediction for temporal propagation of seasonal influenza using transformer-based model. *Journal of biomedical informatics*, 122:103894, 2021.

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. *Advances in Neural Information Processing Systems*, 32:5243–5253, 2019.

Bryan Lim and Stefan Zohren. Time-series forecasting with deep learning: a survey. *Philosophical Transactions of the Royal Society A*, 379(2194):20200209, 2021.

Zhenyu Liu, Zhengtong Zhu, Jing Gao, and Cheng Xu. Forecast methods for time series data: a survey. *IEEE Access*, 9:91896–91912, 2021.

Rishabh Madan and Partha Sarathi Mangipudi. Predicting computer network traffic: a time series forecasting approach using dwt, arima and rnn. In *2018 Eleventh International Conference on Contemporary Computing (IC3)*, pp. 1–5. IEEE, 2018.

Amir Markovitz, Gilad Sharir, Itamar Friedman, Lihi Zelnik-Manor, and Shai Avidan. Graph embedded pose clustering for anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10539–10547, 2020.

Kitaev Nikita, Kaiser Lukasz, Kevskaya Anselm, et al. Reformer: The efficient transformer. *Proceedings of ICLR*, 2020.

Myrsini Ntemi, Ioannis Sarridis, and Constantine Kotropoulos. An autoregressive graph convolutional long short-term memory hybrid neural network for accurate prediction of covid-19 cases. *IEEE Transactions on Computational Social Systems*, 2022.

Boris N Oreshkin, Grzegorz Dudek, Paweł Pełka, and Ekaterina Turkina. N-beats neural network for mid-term electricity load forecasting. *Applied Energy*, 293:116918, 2021.

Mohammed Redha Qader, Shahnawaz Khan, Mustafa Kamal, Muhammad Usman, and Mohammad Haseeb. Forecasting carbon emissions due to electricity power generation in bahrain. *Environmental Science and Pollution Research*, 29(12):17346–17357, 2022.

Yao Qin, Dongjin Song, Haifeng Chen, Wei Cheng, Guofei Jiang, and Garrison W Cottrell. A dual-stage attention-based recurrent neural network for time series prediction. In *IJCAI*, 2017.

Stephan Rasp, Peter D Dueben, Sebastian Scher, Jonathan A Weyn, Soukayna Mouatadid, and Nils Thuerey. Weatherbench: a benchmark data set for data-driven weather forecasting. *Journal of Advances in Modeling Earth Systems*, 12(11):e2020MS002203, 2020.

Alaa Sagheer and Mostafa Kotb. Unsupervised pre-training of a deep lstm-based stacked autoencoder for multivariate time series forecasting problems. *Scientific reports*, 9(1):1–16, 2019.

Usman Sajid, Xiangyu Chen, Hasan Sajid, Taejoon Kim, and Guanghui Wang. Audio-visual transformer based crowd counting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2249–2259, 2021.

Zhipeng Shen, Yuanming Zhang, Jiawei Lu, Jun Xu, and Gang Xiao. A novel time series forecasting model with deep learning. *Neurocomputing*, 396:302–313, 2020. ISSN 0925-2312.

Slawek Smyl. A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting. *International Journal of Forecasting*, 36(1):75–85, 2020. ISSN 0169-2070. M4 Competition.

Theodoros Sofianos, Alessio Sampieri, Luca Franco, and Fabio Galasso. Space-time-separable graph convolutional network for pose forecasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 11209–11218, 2021.

Kamile Stankeviciute, Ahmed M Alaa, and Mihaela van der Schaar. Conformal time-series forecasting. *Advances in Neural Information Processing Systems*, 34:6216–6228, 2021.

Yu Tian, Guansong Pang, Yuanhong Chen, Rajvinder Singh, Johan W Verjans, and Gustavo Carneiro. Weakly-supervised video anomaly detection with robust temporal feature magnitude learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4975–4986, 2021a.

Zhiliang Tian, Wei Bi, Zihan Zhang, Dongkyu Lee, Yiping Song, and Nevin L Zhang. Learning from my friends: few-shot personalized conversation systems via social networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 13907–13915, 2021b.

Guangmo Tong. Stratlearner: learning a strategy for misinformation prevention in social networks. *Advances in Neural Information Processing Systems*, 33:15546–15555, 2020.

José F Torres, Dalil Hadjout, Abderrazak Sebaa, Francisco Martínez-Álvarez, and Alicia Troncoso. Deep learning for time series forecasting: a survey. *Big Data*, 9(1):3–21, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

Renzhuo Wan, Shuping Mei, Jun Wang, Min Liu, and Fan Yang. Multivariate temporal convolutional network: A deep neural networks approach for multivariate time series forecasting. *Electronics*, 8(8):876, 2019.

Max Welling and Thomas N Kipf. Semi-supervised classification with graph convolutional networks. In *J. International Conference on Learning Representations (ICLR 2017)*, 2016.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pp. 38–45, 2020.

Sifan Wu, Xi Xiao, Qianggang Ding, Peilin Zhao, Ying Wei, and Junzhou Huang. Adversarial sparse transformer for time series forecasting. *Advances in Neural Information Processing Systems*, 33, 2020.

Jiehui Xu, Jianmin Wang, Mingsheng Long, et al. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in Neural Information Processing Systems*, 34:22419–22430, 2021.

Zuoyu Yan, Tengfei Ma, Liangcai Gao, Zhi Tang, and Chao Chen. Link prediction with persistent homology: An interactive view. In *International Conference on Machine Learning*, pp. 11659–11669. PMLR, 2021.

Seongjun Yun, Seoyoon Kim, Junhyun Lee, Jaewoo Kang, and Hyunwoo J Kim. Neo-gnns: Neighborhood overlap-aware graph neural networks for link prediction. *Advances in Neural Information Processing Systems*, 34:13683–13694, 2021.

Haopeng Zhang and Jiawei Zhang. Text graph transformer for document classification. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

Xiang Zhang, Marko Zeman, Theodoros Tsiligkaridis, and Marinka Zitnik. Graph-guided network for irregularly sampled multivariate time series. In *International Conference on Learning Representations, ICLR*, 2022.

Bo Zhao, Xianmin Zhang, Zhenhui Zhan, Qiqiang Wu, and Haodong Zhang. Multi-scale graph-guided convolutional network with node attention for intelligent health state diagnosis of a 3-prr planar parallel manipulator. *IEEE Transactions on Industrial Electronics*, 2021a.

Jianan Zhao, Xiao Wang, Chuan Shi, Binbin Hu, Guojie Song, and Yanfang Ye. Heterogeneous graph structure learning for graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 4697–4705, 2021b.

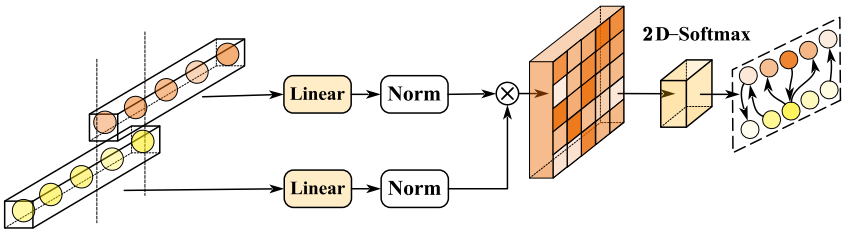
Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of AAAI*, 2021.

Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *Proc. 39th International Conference on Machine Learning (ICML 2022)*, 2022.

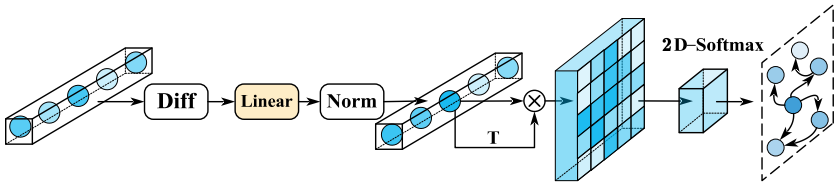
Christoph Zimmer and Reza Yaesoubi. Influenza forecasting framework based on gaussian processes. In *International Conference on Machine Learning*, pp. 11671–11679. PMLR, 2020.

A THE ARCHITECTURES OF THREE PERSPECTIVE GRAPH GENERATORS

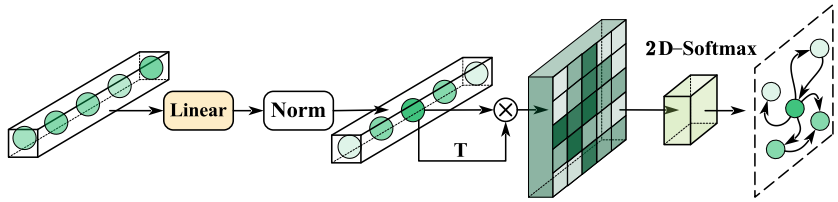
In Figures 3a-3c, we show the detail architectures of three perspective graph generators: Cross-segments Graph Generator, Dimension Graph Generator and Time Graph Generator. All graph generators build topology graphs based on correlation coefficients and obtain the final generated graph through a $2D\text{-}Softmax(\cdot)$ normalization module.



(a) Cross-segments Graph Generator. By calculating the cross-correlation value of two adjacent crossed sub-sequences, the Cross-segments Graph Generator generates asymmetrical graphs to represent the dynamically changing characteristics of the sub-sequences' fluctuations.



(b) Dimension Graph Generator. The Dimension Graph Generator dynamically generates a dimension graph that represents the correlation coefficients of each pair of multivariate variables. Differential operator is used to establish the trend of fluctuations in the series.



(c) Time Graph Generator. The Time Graph Generator extracts the relationship between each timestamp from the perspective of time.

Figure 3: The architectures of three perspective graph generators: Cross-segments Graph Generator.

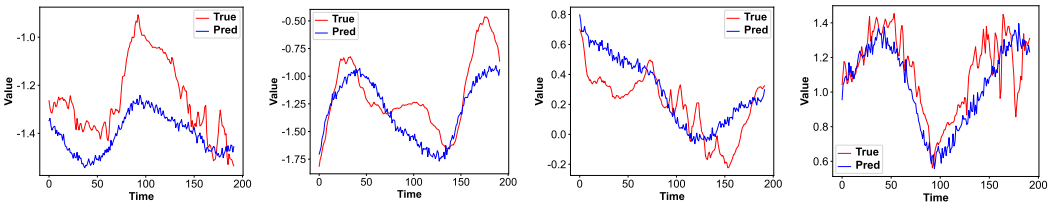
B VISUALIZATION

B.1 WAVEFORM VISUALIZATION

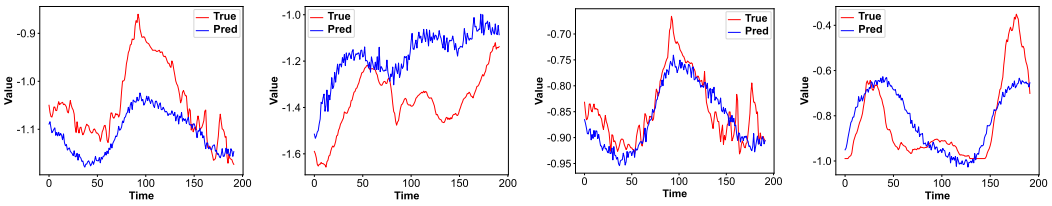
This section illustrates MTGSR's predictions with the Weather and Electricity datasets alongside those of the state-of-the-art FEDformer. Figures 4a-4h show the eight selected dimension sequences for MTGSR, while Figures 5a-5h show those of FEDformer. Comparing the prediction sequences for each corresponding dimension, MTGSR yields better predictions than FEDformer across all dimensions. Further, because MTGSR benefits from the cross-segment graph characteristics, the local trends have a better fit than with FEDformer, as shown in Figures 4c and Figure 5c.

B.2 HEATMAP VISUALIZATION

The heat maps from three perspectives for MTGSR with the Weather, Exchange, and ETTm1 datasets appear below. The darker the color of the pixel blocks, the higher the correlation between

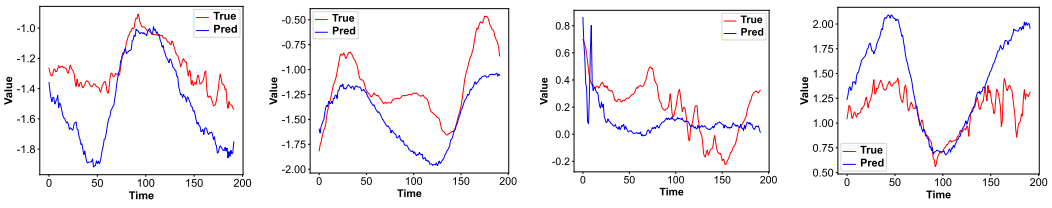


(a) MTGSR 1th (b) MTGSR 2th (c) MTGSR 3th (d) MTGSR 4th

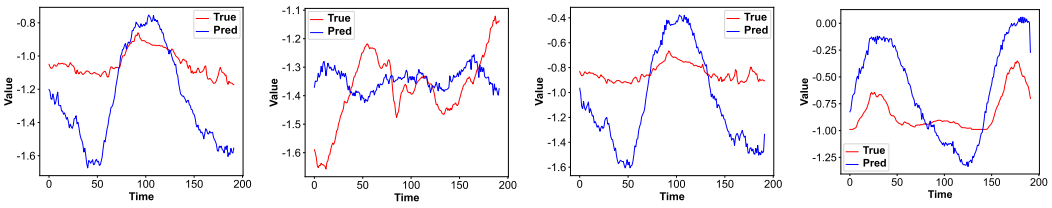


(e) MTGSR 5th (f) MTGSR 6th (g) MTGSR 7th (h) MTGSR 8th

Figure 4: Visualization of MTGSR on eight selected dimensions of Weather dataset.



(a) FEDformer 1th (b) FEDformer 2th (c) FEDformer 3th (d) FEDformer 4th



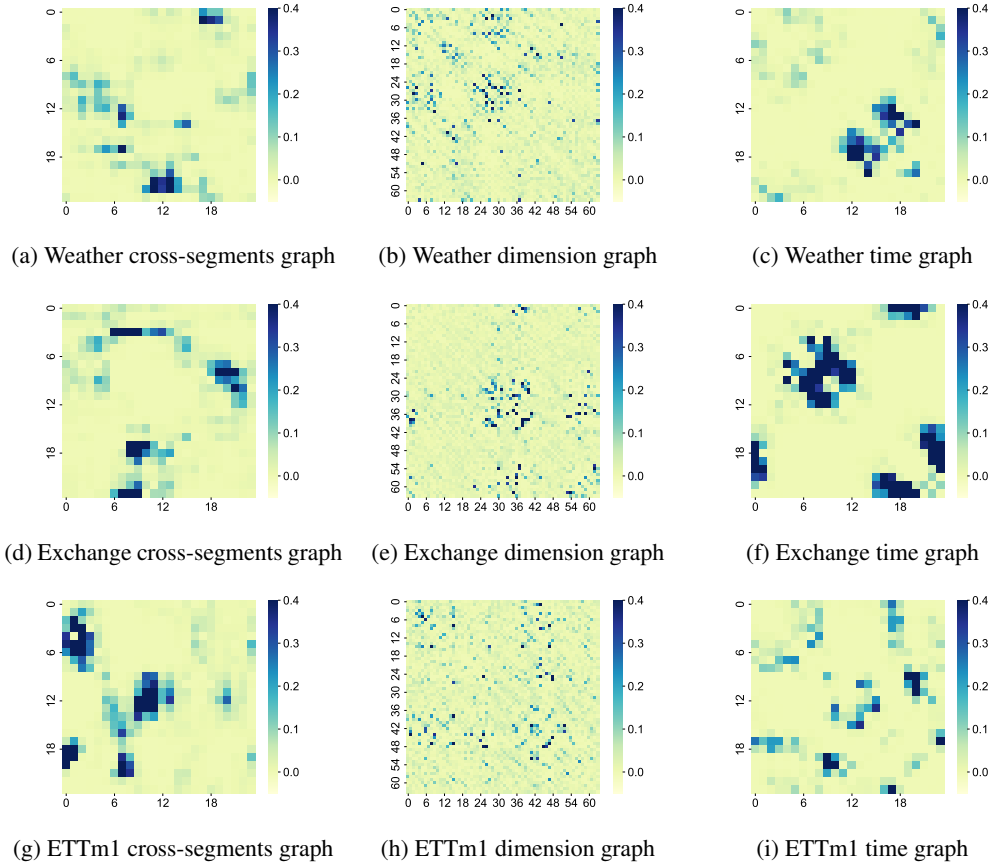
(e) FEDformer 5th (f) FEDformer 6th (g) FEDformer 7th (h) FEDformer 8th

Figure 5: Visualization of FEDformer on eight selected dimensions of Weather dataset.

the two points. Figure 6 shows the different kinds of heat maps of each dataset. Heat maps of different datasets exhibit different characteristics. In Figures 6c 6f and 6i, the ETTm1 dataset has richer periodic variables, so it has richer timing characteristics than the other two trend component-rich datasets. Figures 6b 6e and 6h show the different relationships of each variables in each dataset. For example, the relationships between the first several multivariate variables in the weather dataset are more close. Unlike the other two kinds of graphs, the cross-segments graphs are asymmetrical because they calculate the mutual correlation between two local crossing segments. Figures 6a 6d and 6g show the fluctuation relationship between adjacent local segments of three datasets with different local trends and cyclical variations.

C Extend Experiments on ETT series datasets

We performed extensive experiments on the ETT series datasets, including ETTh1, ETTh2, ETTm1 and ETTm2. The results are shown in Table 5. On the ETTh2, ETTm1 and ETTm2 datasets, MTGSR respectively reduced the error rate by 8.47%, 3.33%, and 9.58% compared to the state-of-the-art FEDformer.



(a) Weather cross-segments graph (b) Weather dimension graph (c) Weather time graph

(d) Exchange cross-segments graph (e) Exchange dimension graph (f) Exchange time graph

(g) ETTm1 cross-segments graph (h) ETTm1 dimension graph (i) ETTm1 time graph

Figure 6: Heatmap Visualization.

Methods		ACDN		FEDformer		Autoformer		Informer		LogTrans		Reformer	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.390	**0.408**	**0.376**	0.419	0.449	0.459	0.865	0.713	0.878	0.740	0.837	0.728
	192	0.465	0.452	**0.420**	**0.448**	0.500	0.482	1.008	0.792	1.037	0.824	0.923	0.766
	336	0.496	0.468	**0.459**	**0.465**	0.521	0.496	1.107	0.809	1.238	0.932	1.097	0.835
	720	0.552	0.514	**0.506**	**0.507**	0.514	0.512	1.181	0.865	1.135	0.852	1.257	0.889
ETTh2	96	**0.314**	**0.358**	0.346	0.388	0.358	0.397	3.755	1.525	2.116	1.197	2.626	1.317
	192	**0.398**	**0.409**	0.429	0.439	0.456	0.452	5.602	1.931	4.315	1.635	11.12	2.979
	336	**0.422**	**0.432**	0.496	0.487	0.482	0.486	4.721	1.835	1.124	1.604	9.323	2.769
	720	**0.440**	**0.447**	0.463	0.474	0.515	0.511	3.647	1.625	3.188	1.540	3.874	1.697
ETTm1	96	**0.349**	**0.385**	0.378	0.418	0.505	0.475	0.672	0.571	0.600	0.546	0.538	0.528
	192	**0.380**	**0.402**	0.426	0.441	0.553	0.496	0.795	0.669	0.837	0.700	0.658	0.592
	336	**0.416**	**0.425**	0.445	0.459	0.621	0.537	1.212	0.871	1.124	0.832	0.898	0.721
	720	**0.472**	**0.458**	0.543	0.490	0.671	0.561	1.166	0.823	1.153	0.820	1.102	0.841
ETTm2	96	**0.185**	**0.265**	0.203	0.287	0.255	0.339	0.365	0.453	0.768	0.642	0.658	0.619
	192	**0.256**	**0.310**	0.269	0.328	0.281	0.340	0.533	0.563	0.989	0.757	1.078	0.827
	336	0.327	**0.356**	**0.325**	0.366	0.339	0.372	1.363	0.887	1.334	0.872	1.549	0.972
	720	**0.420**	**0.408**	0.421	0.415	0.422	0.419	3.379	1.338	3.048	1.328	2.631	1.242

Table 5: Multivariate long sequence time-series forecasting results on the ETT series datasets. The best results are highlighted in bold.