G-GLformer: Transformer with GRU Embedding and Global-Local Attention for Multivariate Time Series Forecasting

Wenjun Yu[1], Jiyanglin Li[2⋆] (✉), Wentao Gao[3], Niangxi Zhuang[4], Wen Li[1⋆] (✉), and Shouguo Du[5]

[1] Shanghai University of International Business and Economics, Shanghai, China
18227071719@163.com, liwen@suibe.edu.cn
[2] Guizhou University of Finance and Economics, Guizhou, China
jylli@mail.gufe.edu.cn
[3] University of South Australia, Adelaide, Australia
wentao.gao@mymail.unisa.edu.au
[4] Guangzhou Nanfang College, Guangzhou, China niangxizhuang01@gmail.com
[5] Shanghai Municipal Big Data Center, Shanghai, China
shouguo.du.sh@outlook.com

Abstract. Time series forecasting plays a vital role in various fields. Due to the special ability of its self-attention mechanism in capturing long-term dependencies, Transformer has been widely used in time series modeling. However, the majority of contemporary Transformer-based models adopt variate tokenization, where the self-attention mechanism is used to extract variable correlations, which weakens the extraction of temporal correlations. Furthermore, the self-attention mechanism extracts correlations within the look-back window. Owing to the absence of a global perspective, the correlations it captures may be influenced by local noise. To tackle these issues, we propose an advanced Transformer architecture entitled G-GLformer, which designs two novel modules, Bidirectional-Patch-GRU-Embedding (BPGE) and Global-Local-Attention (GLA), and integrates them into the Transformer to achieve more accurate forecast. Specifically, the BPGE module is mainly used to model temporal relationships and enhance local semantics. The GLA module integrates the correlation coefficients of the training set data with the data from the local look-back window. This endows the data in the look-back window with a global perspective, making it less susceptible to the influence of noise. Moreover, they can also be used as plug-ins in other models. Extensive experiments on public datasets demonstrate its superior performance over other state-of-the-art models.

Keywords: Time series forecasting · Multivariate time series · Transformer.

⋆ First Corresponding Author: Wen Li, Second Corresponding Author: Jiyanglin Li

1 Introduction

Multivariate time series forecasting holds significant importance in real-world domains such as weather [19], energy [25], transportation [1] and finance [16]. In recent years, various deep learning models have been proposed, significantly pushing the performance boundaries [13, 14, 4, 21]. Transformers have demonstrated impressive performance in time series forecasting, primarily due to their attention mechanisms [18, 26, 11, 9]. Traditionally, the majority of methods based on Transformer have employed temporal tokenization, which considers all variates at a specific timestamp as one single token. However, recent research findings indicate that variate tokenization, in which every variate is embedded as an individual token, surpasses temporal tokenization in terms of capturing the dependencies among variates. Due to its broad applicability to Transformers, variate tokenization has been adopted in recent advancements in multivariate time series forecasting [10, 15, 3]. However, these models overlook two important issues.

First, these models use the self-attention mechanism to model variable correlations instead of temporal correlations. The absence of self-attention mechanism for modeling temporal correlations undoubtedly weakens the model's ability to capture sequential relationships. Although the model adds positional encoding during embedding, this is insufficient for the model to recognize temporal relationships [23]. We contend that Recurrent Neural Networks (RNNs) may offer solutions to these issues with Transformers. This is mainly because the internal structure of RNNs is highly suitable for sequential data and hidden units can serve as an excellent representation of time series. However, when dealing with long time series, RNNs tend to encounter problems such as gradient vanishing or explosion, as well as the inability to capture long-term dependencies [12]. Therefore, we divide the time series into patches of the same size. This method can reduce the length of the sequence, mitigate the problems mentioned above, and enhance the local features of the sequence.

Secondly, these models always extract correlations within a limited look-back window. Due to the lack of a global perspective, the extraction of correlations is vulnerable to the influence of local noise. Based on the above situation, we have provided an example as shown in Figure 1. When there is a high positive global correlation, locally, due to the presence of noise, there may be situations where the variables are uncorrelated or even negatively correlated. Therefore, we attempt to integrate global and local correlations to enhance the robustness of variable correlations extraction. Specifically, we calculate the Pearson correlation coefficients among variates in the training set. Subsequently, we take these Pearson correlation coefficients as a static graph and apply graph convolution to the query and key in the self-attention mechanism, aiming to integrate the global correlations with the local correlations.

To this end, We propose a novel Transformer architecture based on variate tokenization. The model mainly improves the performance of time series prediction through the method of GRU embedding and by integrating the global

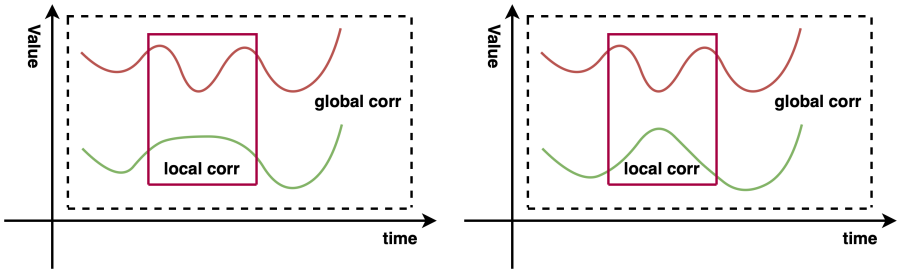

Fig. 1. The figure on the left indicates that there is no local correlation between the two time series. However, there is a high degree of positive correlation except for the locally noisy parts. The right-hand figure shows a negative correlation between the two time series locally, yet there is a high positive correlation except for the locally noisy part.

correlation with the local correlation. The main contributions are summarized as follows:

- We propose the BPGE module to enhance the temporal features of the variate-tokenized Transformer. Furthermore, the BPGE module is only used during the embedding process. Therefore, it will not significantly increase the computational burden.
- The GLA module we proposed innovatively integrates global and local correlations to improve the model's robustness to local noise. In addition, this module can also serve as a plug-in and be applied to other models that use the self-attention mechanism to extract variable correlations.
- Our model is rigorously validated on multiple standard benchmark datasets. Compared to the iTransformer (SOTA), our model achieves an average reduction in Mean Squared Error(MSE) of up to 6.12 % and Mean Absolute Error(MAE) of up to 5.08 %, demonstrating its superior ability in time series forecasting.

2 Related Work

2.1 Transformer-Based Forecasters

Inspired by the success of Transformers in natural language processing, numerous Transformers have been proposed for multivariate time series forecasting. Classic works such as Autoformer [18], Informer [25], Pyraformer [8], and FEDformer [26] represent early Transformer-based time series forecasters. These models adopt the method of temporal tokenization, encoding the values of all variates at a specific time step into an independent token. Then, the self-attention mechanism is used to extract the correlations between different tokens. However, these models fail to capture the correlations among different variates. iTransformer

[9] introduces the inverted Transformer to capture multivariate dependencies and achieves accurate forecasts. Due to the good predictive performance and interpretability, an increasing number of models now adopt variate tokenization. Timer [10] combines several variates originating from diverse domains into a unified time series and regards time series as a single token. MCformer [3] performs tokenization on each individual variate and then blends these variates together, aiming to capture the correlations existing between different variates. Similarly, TimeXer [15] also makes use of variate tokenization when it comes to introducing exogenous variates.

2.2 Dependency Modeling

Time series correlations mainly includes variable correlations and temporal correlations. Some models [11, 6, 23] adopt a Channel Independency(CI) strategy, that is, they only consider temporal correlations. These models solely rely on the historical information of each individual sequence for prediction, overlooking the correlations among different variates. Although this method is robust, it squanders the potential information among different variates. Channel Dependency(CD) strategy [9, 24, 22] model the correlations among all variates, and as a result, they reduce robustness. The primary cause of the lack of robustness is the incorrect extraction of variable correlations due to local noise. Therefore, designing a robust CD method is challenging.

3 Methodology

In this section, we will describe the Bidirectional-Patch-GRU-Embedding module, the Global-Local-Attention module, and the overall architecture of the model. In multivariate time series forecasting, given historical observations $\mathbf{X} \in \mathbb{R}^{T \times N}$ with T time steps and N variates, we predict the future S time steps $\mathbf{Y} \in \mathbb{R}^{S \times N}$. The time series forecasting problem aims to learn a function $f(\cdot)$ that maps the time step T of history to the next time step S:

$$[X_{t-T+1}, X_{t-T+2}, \cdots, X_t] \xrightarrow{f(\cdot)} [\hat{X}_{t+1}, \hat{X}_{t+2}, \cdots, \hat{X}_{t+S}]. \tag{1}$$

The architecture of our proposed model, referred to as G-GLformer, is depicted in Figure 2.

3.1 Bidirectional-Patch-GRU-Embedding

To enhance the model's ability to capture temporal relationships, we adopt the GRU model [2] with a recurrent structure. However, due to the inherent flaws of the recurrent structure, it is unable to capture long-term dependencies. Moreover, it is prone to problems such as vanishing or exploding gradients. Therefore, time series is split into non-overlapping patches, and each patch is projected to a temporal token. The patch technique can effectively reduce the sequence length,

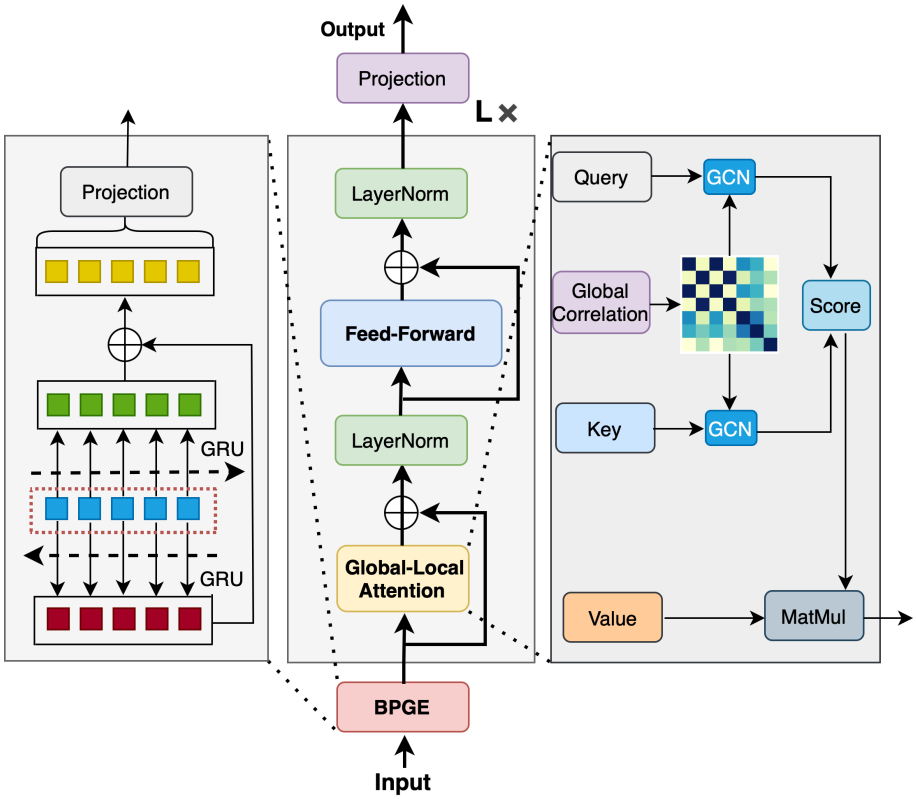

Fig. 2. G-GLformer is mainly composed of three modules: the Bidirectional-Patch-GRU-Embedding module, the Global-Local-Attention module, and the Feed-Forward module.

thus avoiding the above-mentioned problems. The reason for adopting bidirectional GRU is that it can extract more abundant and comprehensive features from the input sequence. During the learning process, the forward and backward GRU capture feature information from different aspects. By combining these features, a more representative feature representation can be provided for subsequent tasks. The overall Bidirectional-Patch-GRU-Embedding is formally stated as:

$$Patchify(\mathbf{X}) = \{P_1, P_2, \cdots, P_S\}, \tag{2}$$

$$\mathbf{P}_{forward} = GRU_{forward}(P_1, P_2, \cdots, P_S), \tag{3}$$

$$\mathbf{P}_{backward} = GRU_{backward}(P_1, P_2, \cdots, P_S), \tag{4}$$

$$\mathbf{P} = PatchEmbed(\mathbf{P}_{forward} + \mathbf{P}_{backward}). \tag{5}$$

Denote by L the length of the patch, by $S = \lfloor \frac{T}{L} \rfloor$ the number of patches split from the time series, and by P_i the i-th patch. Specifically, $Patchify(\cdot)$:

$\mathbb{R}^{N\times T} \rightarrow \mathbb{R}^{N\times S\times L}$ is a process of splitting a time series into non-overlapping patches. $GRU(\cdot) : \mathbb{R}^{N\times S\times L} \rightarrow \mathbb{R}^{N\times S\times D}$ maps a patch with a length of L into a hidden state of dimension D. $PatchEmbed(\cdot) : \mathbb{R}^{N\times S\times D} \rightarrow \mathbb{R}^{N\times D}$ maps all patches into a D-dimensional vector via a trainable linear projector. Through the above operations, we complete the embedding work of the sequence.

3.2 Global-Local-Attention

To capture more reliable multivariate correlations, we introduce a new Global-Local-Attention(GLA) mechanism to calculate attention weights for more effective attention distribution. Taking the first layer of the Encoder as an example, $\mathbf{P} \in \mathbb{R}^{N\times D}$ is the output after passing through the BPGE layer. We form the query, key, and value matrices for each attention head h as:

$$Q^h = \mathbf{P}W_Q^h, \quad K^h = \mathbf{P}W_K^h, \quad V^h = \mathbf{P}W_V^h, \tag{6}$$

where $W_Q^h, W_K^h, W_V^h \in \mathbb{R}^{D\times D_H}$, D_H is the heads' hidden dimension. Then, we apply the Mixhop graph convolution [20] to $Q^h \in \mathbb{R}^{N\times D_H}$ and $K^h \in \mathbb{R}^{N\times D_H}$, where N is the number of variates. The GLA is defined as follows:

$$\tilde{Q}^h = GCN(Q^h, \tilde{A}), \tilde{K}^h = GCN(K^h, \tilde{A}), \tag{7}$$

$$S^h = Softmax(\frac{\tilde{Q}^h(\tilde{K^h})^T}{\sqrt{D_H}})V^h, \tag{8}$$

$$GLA(\mathbf{P}) = \sum_h S^h V^h W_O^h, \tag{9}$$

where $W_O^h \in \mathbb{R}^{D_H\times D}$ is output matrix, $GCN(\cdot)$ denotes Mixhop graph convolution layer and $\tilde{A} \in \mathbb{R}^{N\times N}$ represents the adjacency matrix. The adjacency matrix learning process in our approach involves generating a learnable matrix. Then, the learnable matrix is added to the Pearson correlation coefficients of the training set. The specific process is as follows:

$$\tilde{A} = M + Pr, \tag{10}$$

where $M \in \mathbb{R}^{N\times N}$ is a learnable matrix, and $Pr \in \mathbb{R}^{N\times N}$ is the Pearson correlation coefficient matrix of the training set.

After obtaining the adjacency matrix, we use the Mixhop graph convolution layer to embed the global correlation into the local correlation. The Mixhop graph convolution layer consists of two steps: the information propagation step and the information selection step. The information propagation step is defined as follows:

$$H^{(k)} = \beta H_{in} + (1-\beta)\tilde{A}H^{(k-1)}, \tag{11}$$

where β is a hyper parameter, which controls the ratio of retaining the root node's original states. The information selection step is defined as follows:

$$H_{out} = Linear((Concatenate(H^{(0)}, H^{(1)}, \cdots, H^{(k)}))), \tag{12}$$

where k is the depth of propagation. $H^{(0)} = H_{in}$ represents the input of the graph convolution layer, and H_{out} represents the output of the graph convolution layer. Finally, the output dimension is unified with the input dimension through a linear layer.

3.3 Feed-Forward Layer

Transformer adopts the feed-forward network (FFN) as the basic building block for encoding token representation and it is identically applied to each token. In this paper, we employ FFN to learn variable representations for time series forecasting. In order to reduce overfitting and improve the generalization ability of the model, Dropout is adopted in the FFN layer. FFN employs a Gaussian Error Linear Unit (GELU) activation function to facilitate non-linear transformations. The FFN is defined as follows:

$$X_h = Dropout(GELU(Linear(X_{in}))), \tag{13}$$

$$X_{out} = Dropout(Linear(X_h)) + X_{in}, \tag{14}$$

where X_{in} and $X_{out} \in \mathbb{R}^{N \times D}$ are the input and output of the FFN, $X_h \in \mathbb{R}^{N \times F}$,where F represents the size of the hidden layer.

3.4 Projection Prediction

A linear layer is used to obtain the final prediction $\hat{Y} = [\hat{X}_{t+1}, \hat{X}_{t+2}, \cdots, \hat{X}_{t+S}] \in \mathbb{R}^{N \times S}$. The formula is as follows:

$$\hat{Y} = Projection(X_{out}), \tag{15}$$

where $X_{out} \in \mathbb{R}^{N \times D}$ is the output of the last layer of the Encoder.

We use Mean Squared Error (MSE) to measure the difference between the predicted values $\hat{Y}$ and the ground truth Y. MSE is calculated within S time steps. The formula is as follows:

$$\mathcal{L} = \frac{1}{S} \sum_{i=1}^{S} \left\| \hat{X}_{t+i} - X_{t+i} \right\|_2^2. \tag{16}$$

4 Experiments

4.1 Datasets

We evaluate the performance of G-GLformer on 13 real-world datasets, including Weather, Exchange, ECL, Traffic, ETT (ETTh1, ETTh2, ETTm1, ETTm2) used by Autoformer [18], Solar-Energy used by LST-Net [5] and PEMS datasets (PEMS03, PEMS04, PEMS07, PEMS08) adopted by SCINet [7]. These are widely used multivariate time series datasets, and we handle the datasets the same way as iTransformer [9]. The details of datasets are as follows:

- **ETT (Electricity Transformer Temperature)** [25] includes four subsets. ETTh1 and ETTh2 collect hourly data on 7 different factors from two distinct electricity transformers from July 2016 to July 2018. ETTm1 and ETTm2 record the same factors at a higher resolution of every 15 minutes.
- **Traffic** [18] collects hourly data from the California Department of Transportation, describing road occupancy rates measured by 862 sensors on San Francisco Bay Area freeways.
- **ECL(Electricity)** [18] captures the hourly electricity consumption of 321 clients from 2012 to 2014.
- **Weather** [18] records 21 meteorological factors such as air temperature and humidity every 10 minutes throughout the year 2020.
- **Exchange** [18] records the daily exchange rates of eight different countries ranging from 1990 to 2016.
- **Solar-Energy** [5] includes data from 137 PV plants in Alabama State, with solar power production sampled every 10 minutes during 2006.
- **PEMS** [7] includes traffic network data from California, sampled every 5 minutes, focusing on four public subsets: PEMS03, PEMS04, PEMS07, and PEMS08.

The summary of the datasets is shown in Table 1.

Table 1. Detailed dataset descriptions. Dim denotes the variate number of each dataset. Dataset Size denotes the total number of time points in (Train, Validation, Test) split respectively. Prediction Length denotes the future time points to be predicted and four prediction settings are included in each dataset. Frequency denotes the sampling interval of time points.

Dataset	Dim	Prediction Length	Dataset Size	Frequency
ETTh1,ETTh2	7	$\{96, 192, 336, 720\}$	(8545, 2881, 2881)	Hourly
ETTm1,ETTm2	7	$\{96, 192, 336, 720\}$	(34465, 11521, 11521)	15min
Exchange	8	$\{96, 192, 336, 720\}$	(5120, 665, 1422)	Daily
Weather	21	$\{96, 192, 336, 720\}$	(36792, 5271, 10540)	10min
ECL	321	$\{96, 192, 336, 720\}$	(18317, 2633, 5261)	Hourly
Traffic	862	$\{96, 192, 336, 720\}$	(12185, 1757, 3509)	Hourly
Solar-Energy	137	$\{96, 192, 336, 720\}$	(36601, 5161, 10417)	10min
PEMS03	358	$\{12, 24, 48, 96\}$	(15617, 5135, 5135)	5min
PEMS04	307	$\{12, 24, 48, 96\}$	(10172, 3375, 3375)	5min
PEMS07	883	$\{12, 24, 48, 96\}$	(16911, 5622, 5622)	5min
PEMS08	170	$\{12, 24, 48, 96\}$	(10690, 3548, 3548)	5min

4.2 Baselines

We compare G-GLformer with six state-of-the art models from three categories, including (1) Transformer based models: iTransformer [9] , PatchTST [11], FED-

former [26], Autoformer [18] (2) CNN-based model: TimesNet [17] and (3) Linear-based models: DLinear [23].

4.3 Experimental Results

Table 2 shows the average errors for the four prediction lengths of 96, 192, 336, and 720. Due to the limitation of the length of this paper, we have not presented all the results. As can be seen from the Table 2, G-GLformer achieves a total of 13 first places and 5 second places in long-term time series prediction. However, it can also be observed that although the Mean Squared Error (MSE) of the Traffic dataset achieved the second place for G-GLformer, there is still a significant gap compared with the first-place result of iTransformer. We believe this is due to the presence of a large number of outliers in the Traffic dataset. We will discuss this issue in detail in the "Discussion" section.

Table 2. Full results for the long-term forecasting task. The input sequence length is set to 96 for all baselines. The results in the table are the averages of the prediction lengths of 96, 192, 336, and 720. The best results are in **bold** and the second best are underlined.

Models	**G-GLformer (Ours)**		iTransformer (2024)		PatchTST (2023)		DLinear (2023)		TimesNet (2023)		FEDformer (2022)		Autoformer (2021)	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	**0.433**	**0.433**	0.457	0.449	0.469	0.454	0.456	0.452	0.458	0.450	0.440	0.460	0.496	0.487
ETTh2	**0.377**	**0.402**	0.384	0.407	0.387	0.407	0.559	0.515	0.414	0.427	0.437	0.449	0.450	0.459
ETTm1	0.394	0.401	0.408	0.412	**0.387**	**0.400**	0.403	0.407	0.400	0.406	0.448	0.452	0.588	0.517
ETTm2	0.283	0.328	0.293	0.336	**0.281**	**0.326**	0.350	0.401	0.291	0.333	0.305	0.349	0.327	0.371
ECL	**0.166**	**0.261**	0.176	0.267	0.205	0.290	0.212	0.300	0.192	0.295	0.214	0.327	0.227	0.338
Exchange	**0.353**	**0.400**	0.365	0.407	0.367	0.404	0.354	0.414	0.416	0.443	0.519	0.429	0.613	0.539
Traffic	0.448	**0.281**	**0.422**	0.283	0.481	0.304	0.625	0.383	0.620	0.336	0.610	0.376	0.628	0.379
Weather	**0.252**	**0.277**	0.260	0.280	0.259	0.281	0.265	0.317	0.259	0.287	0.309	0.360	0.338	0.382
Solar-Energy	**0.231**	**0.257**	0.236	0.262	0.270	0.307	0.330	0.401	0.301	0.319	0.291	0.381	0.885	0.711
1^{st} Count	**6**	**7**	1	0	2	2	0	0	0	0	0	0	0	0

Table 3 shows the results of G-GLformer in short-term forecasting. The results show that G-GLformer achieves the best performance in all aspects of short-term prediction. Specifically, G-GLformer has an average decrease of 34.21% and 15.96% in MSE and MAE over the previous SOTA iTransformer [9] in short-term forecasting.

4.4 Ablation Study

We conducted ablation experiments to prove the effectiveness of Bidirectional-Patch-GRU-Embedding and Global-Local-Attention module. We considered 3

Table 3. Full results for the short-term forecasting task. The input sequence length is set to 96 for all baselines. Avg means the average results from all four prediction lengths. The best results are in **bold**.

Models		**G-GLformer (Ours)**		iTransformer (2024)		PatchTST (2023)		DLinear (2023)		TimesNet (2023)		FEDformer (2022)		Autoformer (2021)	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
PEMS03	12	**0.065**	**0.171**	0.069	0.174	0.099	0.216	0.122	0.243	0.085	0.192	0.126	0.251	0.272	0.385
	24	**0.089**	**0.201**	0.098	0.209	0.142	0.259	0.201	0.317	0.118	0.223	0.149	0.275	0.334	0.440
	48	**0.131**	**0.244**	0.164	0.276	0.211	0.319	0.333	0.425	0.155	0.260	0.227	0.348	1.032	0.782
	96	**0.214**	**0.315**	0.240	0.338	0.269	0.370	0.457	0.515	0.228	0.317	0.348	0.434	1.031	0.796
	Avg	**0.125**	**0.233**	0.143	0.249	0.180	0.291	0.278	0.375	0.147	0.248	0.213	0.327	0.667	0.601
PEMS04	12	**0.077**	**0.184**	0.081	0.188	0.105	0.224	0.148	0.272	0.087	0.195	0.138	0.262	0.424	0.491
	24	**0.091**	**0.204**	0.100	0.212	0.153	0.275	0.224	0.340	0.103	0.215	0.177	0.293	0.459	0.509
	48	**0.111**	**0.228**	0.131	0.245	0.229	0.339	0.355	0.437	0.136	0.250	0.270	0.368	0.646	0.610
	96	**0.133**	**0.249**	0.165	0.277	0.291	0.389	0.452	0.504	0.190	0.303	0.341	0.427	0.912	0.748
	Avg	**0.103**	**0.216**	0.119	0.231	0.195	0.307	0.295	0.388	0.129	0.241	0.231	0.337	0.610	0.590
PEMS07	12	**0.066**	**0.163**	0.066	0.164	0.095	0.207	0.115	0.242	0.082	0.181	0.109	0.225	0.199	0.336
	24	**0.076**	**0.175**	0.087	0.190	0.150	0.262	0.210	0.329	0.101	0.204	0.125	0.244	0.323	0.420
	48	**0.083**	**0.181**	0.113	0.218	0.253	0.340	0.398	0.458	0.134	0.238	0.165	0.288	0.390	0.470
	96	**0.102**	**0.202**	0.140	0.246	0.346	0.404	0.594	0.553	0.181	0.279	0.262	0.376	0.554	0.578
	Avg	**0.082**	**0.180**	0.102	0.205	0.211	0.303	0.329	0.395	0.124	0.225	0.165	0.283	0.367	0.451
PEMS08	12	**0.082**	**0.185**	0.088	0.193	0.168	0.232	0.154	0.276	0.112	0.212	0.173	0.273	0.436	0.485
	24	**0.117**	**0.221**	0.138	0.243	0.224	0.281	0.248	0.353	0.141	0.238	0.140	0.236	0.467	0.502
	48	**0.183**	**0.235**	0.339	0.354	0.321	0.354	0.440	0.470	0.198	0.283	0.320	0.394	0.966	0.733
	96	**0.200**	**0.244**	0.418	0.416	0.408	0.417	0.674	0.565	0.320	0.351	0.442	0.465	1.385	0.915
	Avg	**0.146**	**0.221**	0.246	0.302	0.280	0.321	0.379	0.416	0.193	0.271	0.286	0.358	0.814	0.659
1^{st} Count		**20**	**20**	0	0	0	0	0	0	0	0	0	0	0	0

ablation methods and evaluated them on 4 datasets. The following will explain the variants of its implementation:

- **w/o-BPGE:** We removed the Bidirectional-Patch-GRU-Embedding module and used a linear layer as the embedding layer.
- **w/o-GLA:** We replaced the Global-Local-Attention mechanism with the traditional self-attention mechanism.
- **w/o-BPGE&GLA:** We replaced the Bidirectional-Patch-GRU-Embedding with a linear layer and replaced the Global-Local-Attention with the traditional attention mechanism.

As can be seen from Table 4, G-GLformer achieves the best results. When the Bidirectional-Patch-GRU-Embedding module is absent, the model fails to capture the temporal relationships adequately, resulting in a decline in its performance. When the model lacks the Global-Local-Attention module, it leads to the model's lack of a global correlation perspective. As a result, the extraction of correlations is vulnerable to local noise, thus causing a decline in prediction performance. When both modules are absent, the prediction performance achieved by the model is the worst.

Table 4. Ablation analysis of ETTh2, ETTm2, Weather and Solar-Energy datasets. Results represent the average error of prediction length $\{96, 192, 336, 720\}$, with the best performance highlighted in bold black.

Datasets	ETTh2		ETTm2		Weather		Solar-Energy	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
G-GLformer	**0.377**	**0.402**	**0.283**	**0.328**	**0.252**	**0.277**	**0.231**	**0.257**
w/o-BPGE	0.381	0.405	0.290	0.334	0.254	0.278	0.234	0.261
w/o-GLA	0.381	0.404	0.287	0.330	0.257	0.279	0.234	0.260
w/o-BPGE&GLA	0.383	0.407	0.291	0.335	0.260	0.281	0.238	0.263

4.5 Modules Generality

To verify the generality of the modules, we use the Bidirectional-Patch-GRU-Embedding module and the Global-Local-Attention module as plugins to enhance other models.

Global-Local-Attention. To prove the generality of the Global-Local-Attention module on other models, we applied this module to iTransformer [9] and used three different correlation coefficients. As shown in Figure 3, we conducted tests on four datasets, namely PEMS03, PEMS04, PEMS07, and PEMS08, and then used the Pearson correlation coefficient, the Spearman correlation coefficient, and the Kendall correlation coefficient respectively. As can be found from Figure 3, the error of iTransformer with the Global-Local-Attention module is lower than those of the original iTransformer on all the four datasets. There are no significant differences among the three correlation coefficients on PEMS03, PEMS04, and PEMS07. However, there are relatively large fluctuations on PEMS08. The reason for this situation is that the calculation methods of the three correlation coefficients are different and the situations to which they are applicable are also different. We will discuss this issue in the "Discussion" section.

Table 5. Results represent the average error of prediction length $\{96, 192, 336, 720\}$, with the best performance highlighted in bold black.

Models	iInformer		+**BPGE**		iReformer		+**BPGE**		iFlashformer		+**BPGE**		Impr.
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	
ETTh1	0.487	0.465	**0.441**	**0.435**	0.472	0.457	**0.440**	**0.436**	0.454	0.448	**0.441**	**0.438**	5.39%
ETTh2	0.390	0.410	**0.379**	**0.405**	0.384	0.409	**0.381**	**0.406**	0.389	0.410	**0.379**	**0.403**	1.75%
ETTm1	0.402	0.406	**0.389**	**0.399**	0.405	0.409	**0.394**	**0.401**	0.411	0.414	**0.406**	**0.408**	1.96%
ETTm2	0.288	0.332	**0.281**	**0.327**	0.288	0.332	**0.283**	**0.328**	0.292	0.335	**0.287**	**0.330**	1.61%
Impr.	4.02%				3.14%				1.78%				2.78%

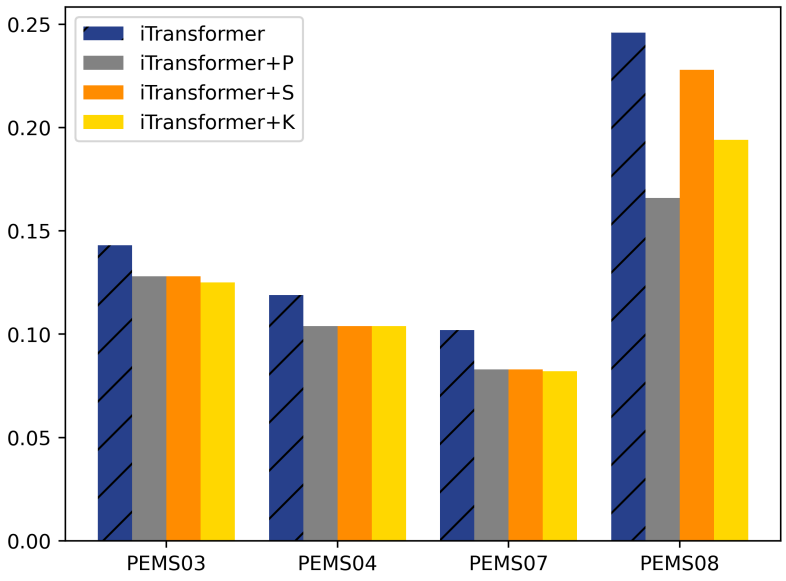

Fig. 3. The results shown in the above figure represent the average error of prediction length $\{96, 192, 336, 720\}$. P, S, and K represent the Pearson correlation coefficient, the Spearman correlation coefficient, and the Kendall correlation coefficient respectively.

Bidirectional-Patch-GRU-Embedding. To verify the generality of the BPGE module, we applied this module to iInformer [9], iReformer [9], and iFlashformer [9]. As can be found from Table 5, the BPGE module enhances the respective backbones by an average of 2.78%. Among them, the BPGE module has the most obvious effect on the ETTh1 dataset, with the error being reduced by an average of 5.39%. This fully demonstrates the effectiveness of the BPGE module.

4.6 Instance Visualization

As can be seen from Figure 4, in the ETTh1 dataset, there is a highly positive correlation between the "MUFL" variable and the "HUFL" variable, and the Pearson correlation coefficient between them is as high as 0.99. When there are no outliers, the time-series diagrams of the "MUFL" variable and the "HUFL" variable are highly consistent.

When outliers appear in the look-back window, as shown in Figure 5, the "HUFL" variable and the "MUFL" variable show a clear divergence in the time-series diagram. The iTransformer [9] model will be affected by local outliers when extracting correlations, because this model extracts correlations within a

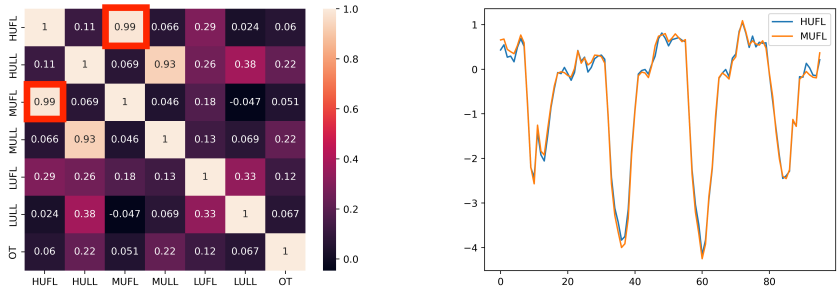

Fig. 4. The left figure shows the overall Pearson correlation coefficient on the ETTh1 training set and the right figure shows a normal look-back window without outliers.

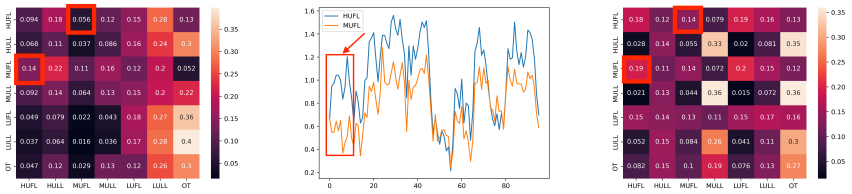

Fig. 5. The figure in the middle is a time-series diagram of a look-back window with outliers. The figure on the left is the score map of iTransformer on this look-back window, and the figure on the right is the score map of G-GLformer on this look-back window.

limited look-back window. As shown in Figure 5, when the iTransformer model uses the self-attention mechanism to extract correlations, the presence of outliers causes it to assign lower weights to two highly correlated time series. Since G-GLformer embeds global correlations, it is more robust to the situation of local outliers. It assigns higher weights to the two variables on the score map. This fully demonstrates that G-GLformer is more robust to local outliers.

5 Discussion

In this section, we will discuss a special case. When there are a large number of outliers in the data, the calculation of the global correlation coefficient will be affected by these outliers, which may lead to the obtained global correlation deviating from the true global correlation. We calculated the average number of outlier points for each channel in seven datasets respectively. As can be seen from Table 6, the average number of outliers per channel in the Traffic dataset is 279.29. The number of outliers in the Traffic dataset is much larger than that in other datasets. A large number of outliers in the Traffic dataset affect the

calculation of global correlations, thus influencing the prediction performance of G-GLformer. This is the reason why the prediction performance of G-GLformer on the Traffic dataset in Table 2 is significantly worse than that of iTransformer [9]. Compared with the Pearson correlation coefficient, we can use the Spearman correlation coefficient, which is more robust to outliers.

Table 6. Average number of extreme points per channel (Z-Score>3).

Datasets	ECL	Solar-Energy	Traffic	PEMS03	PEMS04	PEMS07	PEMS08
Extreme Points	109.29	3.48	**279.29**	18.37	6.07	16.98	17.26

As shown in Figure 6, we replaced the Pearson correlation coefficient with the Spearman correlation coefficient on the Traffic dataset, and MSE decreased for all four prediction lengths. Therefore, in general, G-GLformer is not suitable for datasets with a large number of outliers. When there are a large number of outliers, we can use the Spearman correlation coefficient to alleviate this problem.

6 Conclusion

This paper proposes G-GLformer, a novel Transformer-based architecture that concentrates on tackling the issue of insufficient capture of time lag relationships and the impact of outliers on the extraction of local correlations. Specifically, we introduce the Bidirectional-Patch-GRU-Embedding module to enhance the ability to extract time lag relationships. Besides, we propose the Global-Local-Attention module to enhance the robustness of the model in extracting variable correlations. These innovations jointly boost the ability of G-GLformer to utilize multivariate correlations. Experiments on various datasets have demonstrated its effectiveness and accuracy. In the future, we will explore the performance of G-GLformer on large-scale real-world datasets.

Ethical Statement. The work of our paper focuses on more accurate time series prediction, so there are no ethical issues involved.

Acknowledgement. This work was supported by the Degree Program Development Initiative of Shanghai University of International Business and Economics, Shanghai Action Plan for Science, Technology and Innovation (No. 24BC3200404), National Natural Science Foundation of China (12361056), and Guizhou University of Finance and Economics Introduced Talents for Scientific Research (2022YJ029).

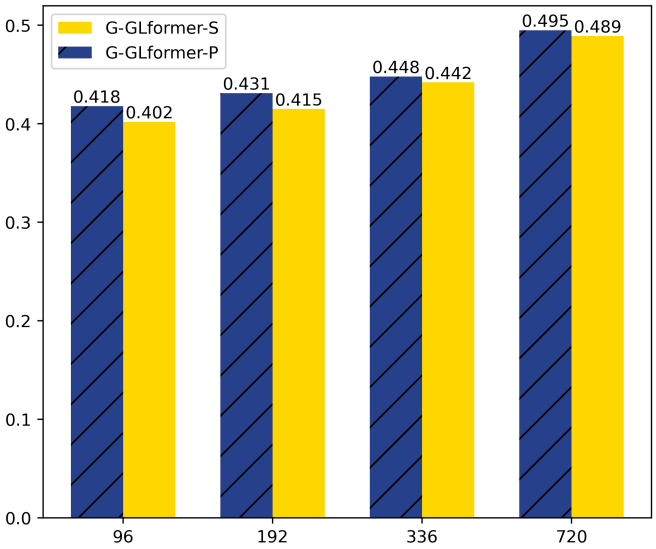

Fig. 6. The results shown in the above figure represent the Mean Squared Error of prediction length $\{96, 192, 336, 720\}$ in Traffic dataset. G-GLformer-S and G-GLformer-P represent G-GLformer using the Spearman correlation coefficient and the Pearson correlation coefficient respectively.

References

1. Chen, Y., Segovia-Dominguez, I., Coskunuzer, B., Gel, Y.: Tamp-s2gcnets: coupling time-aware multipersistence knowledge representation with spatio-supra graph convolutional networks for time-series forecasting. In: International Conference on Learning Representations (2022)
2. Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555 (2014)
3. Han, W., Zhu, T., Chen, L., Ning, H., Luo, Y., Wan, Y.: Mcformer: Multivariate time series forecasting with mixed-channels transformer. IEEE Internet of Things Journal (2024)
4. Hu, Y., Zhang, G., Liu, P., Lan, D., Li, N., Cheng, D., Dai, T., Xia, S.T., Pan, S.: Timefilter: Patch-specific spatial-temporal graph filtration for time series forecasting. arXiv preprint arXiv:2501.13041 (2025)
5. Lai, G., Chang, W.C., Yang, Y., Liu, H.: Modeling long-and short-term temporal patterns with deep neural networks. In: The 41st international ACM SIGIR conference on research & development in information retrieval. pp. 95–104 (2018)
6. Lin, S., Lin, W., Wu, W., Chen, H., Yang, J.: Sparsetsf: modeling long-term time series forecasting with 1k parameters. In: Proceedings of the 41st International Conference on Machine Learning. pp. 30211–30226 (2024)

7. Liu, M., Zeng, A., Chen, M., Xu, Z., Lai, Q., Ma, L., Xu, Q.: Scinet: Time series modeling and forecasting with sample convolution and interaction. Advances in Neural Information Processing Systems **35**, 5816–5828 (2022)
8. Liu, S., Yu, H., Liao, C., Li, J., Lin, W., Liu, A.X., Dustdar, S.: Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. In: International Conference on Learning Representations (2022)
9. Liu, Y., Hu, T., Zhang, H., Wu, H., Wang, S., Ma, L., Long, M.: itransformer: Inverted transformers are effective for time series forecasting. In: The Twelfth International Conference on Learning Representations (2023)
10. Liu, Y., Zhang, H., Li, C., Huang, X., Wang, J., Long, M.: Timer: generative pre-trained transformers are large time series models. In: Proceedings of the 41st International Conference on Machine Learning. pp. 32369–32399 (2024)
11. Nie, Y., Nguyen, N.H., Sinthong, P., Kalagnanam, J.: A time series is worth 64 words: Long-term forecasting with transformers. In: The Eleventh International Conference on Learning Representations (2022)
12. Pascanu, R., Mikolov, T., Bengio, Y.: On the difficulty of training recurrent neural networks. In: Proceedings of the 30th International Conference on International Conference on Machine Learning-Volume 28. pp. III–1310 (2013)
13. Qiu, X., Wu, X., Lin, Y., Guo, C., Hu, J., Yang, B.: Duet: Dual clustering enhanced multivariate time series forecasting. arXiv preprint arXiv:2412.10859 (2024)
14. Wang, S., Li, J., Shi, X., Ye, Z., Mo, B., Lin, W., Ju, S., Chu, Z., Jin, M.: Timemixer++: A general time series pattern machine for universal predictive analysis. arXiv preprint arXiv:2410.16032 (2024)
15. Wang, Y., Wu, H., Dong, J., Qin, G., Zhang, H., Liu, Y., Qiu, Y., Wang, J., Long, M.: Timexer: Empowering transformers for time series forecasting with exogenous variables. In: The Thirty-eighth Annual Conference on Neural Information Processing Systems (2024)
16. Wen, M., Li, P., Zhang, L., Chen, Y.: Stock market trend prediction using high-order information of time series. Ieee Access **7**, 28299–28308 (2019)
17. Wu, H., Hu, T., Liu, Y., Zhou, H., Wang, J., Long, M.: Timesnet: Temporal 2d-variation modeling for general time series analysis. In: The Eleventh International Conference on Learning Representations (2022)
18. Wu, H., Xu, J., Wang, J., Long, M.: Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. Advances in neural information processing systems **34**, 22419–22430 (2021)
19. Wu, H., Zhou, H., Long, M., Wang, J.: Interpretable weather forecasting for worldwide stations with a unified deep model. Nature Machine Intelligence **5**(6), 602–611 (2023)
20. Wu, Z., Pan, S., Long, G., Jiang, J., Chang, X., Zhang, C.: 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. pp. 753–763 (2020)
21. Xu, Z., Zeng, A., Xu, Q.: Fits: Modeling time series with $10k$ parameters. In: The Twelfth International Conference on Learning Representations (2023)
22. Yu, G., Zou, J., Hu, X., Aviles-Rivero, A.I., Qin, J., Wang, S.: Revitalizing multivariate time series forecasting: Learnable decomposition with inter-series dependencies and intra-series variations modeling. In: International Conference on Machine Learning. pp. 57818–57841. PMLR (2024)
23. Zeng, A., Chen, M., Zhang, L., Xu, Q.: Are transformers effective for time series forecasting? In: Proceedings of the AAAI conference on artificial intelligence. vol. 37, pp. 11121–11128 (2023)

24. Zhang, Y., Yan, J.: Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In: The eleventh international conference on learning representations (2023)
25. Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., Zhang, W.: Informer: Beyond efficient transformer for long sequence time-series forecasting. In: Proceedings of the AAAI conference on artificial intelligence. vol. 35, pp. 11106–11115 (2021)
26. Zhou, T., Ma, Z., Wen, Q., Wang, X., Sun, L., Jin, R.: Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In: International conference on machine learning. pp. 27268–27286. PMLR (2022)