A Memory Guided Transformer for Time Series Forecasting

Yunyao Cheng
Aalborg University
yunyaoc@cs.aau.dk

Chenjuan Guo*
East China Normal University
cjguo@dase.ecnu.edu.cn

Bin Yang
East China Normal University
byang@dase.ecnu.edu.cn

Haomin Yu
Aalborg University
haominyu@cs.aau.dk

Kai Zhao
Aalborg University
kaiz@cs.aau.dk

Christian S. Jensen
Aalborg University
csj@cs.aau.dk

ABSTRACT

Accurate long-term forecasting from multivariate time series has important real-world applications. However, achieving this so is challenging. Thus, analyses reveal that time series that span long durations often exhibit dynamic and disrupted correlations. State-of-the-art methods employ attention mechanisms to capture dynamic correlations, but they often do not contend well with disrupted correlations, which reduces prediction accuracy. We introduce local and global information concepts and then leverage these in a Memory Guided Transformer, called the Memformer. By integrating patch-wise recurrent graph learning and global attention, the Memformer aims to capture dynamic correlations and take disrupted correlations into account. We also integrate a so-called Alternating Memory Enhancer into the Memformer to capture correlations between local and global information. We report on experiments that offer insight into the effectiveness of the Memformer at capturing dynamic correlations and its robustness to disrupted correlations. The experiments offer evidence that the new method is capable of advancing the state-of-the-art in forecasting accuracy on real-world datasets.

PVLDB Reference Format:
Yunyao Cheng, Chenjuan Guo, Bin Yang, Haomin Yu, Kai Zhao, and Christian S. Jensen. A Memory Guided Transformer for Time Series Forecasting. PVLDB, 18(2): 239 - 252, 2024.
doi:10.14778/3705829.3705842

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at Memformer (last accessed on October 18, 2024).

1 INTRODUCTION

With the spread of the Internet of Things and cyber-physical systems, we are witnessing a proliferation of multivariate time series data. Among the many applications of such data, forecasting is an important one [29]. A multivariate time series is a sequence of temporally aligned values. In forecasting, previous, or historical, values are used to predict future values. When the historical and forecasting horizons each exceeds 96 time steps, a time series is often called long-term [27].

The variables in a time series are often correlated. For instance, in Figure 1, the variables HUFL and MUFL exhibit similar temporal patterns, as do the variables HULL and MULL. Using the Pearson correlation coefficient to analyze the correlations between these variables, we observe that the correlations often oscillate stably around an average Pearson correlation coefficient. This stable oscillation over time, as illustrated in Figure 1(a), represents **global information** in the time series. Furthermore, the correlations among channels in each subsequence exhibit unique characteristics, which we refer to as **local information**. Next, we observe two types of correlations that occur widely in long-term multivariate time series: **dynamic correlations** and **disrupted correlations**. In Figure 1(a), the fluctuating correlations of the different subsequences in the gray boxes represent the dynamic correlations. In Figure 1(b), the shifts in the distributions of the observations cause outliers that result in disrupted correlations. Outliers occur when the statistical distributions of time series change due to external environmental changes, sampling variations, system evolution, or other factors. Such outliers are prevalent in time series. They disrupt the stationarity of time series and stable correlations, with models often incorrectly capturing the disrupted correlations.

Long-term multivariate time series, compared to short-term ones, encompass a larger number of observations. This fact implies that, across the entire long-term time series, there are more and greater variations in dynamic correlations, and it faces a higher risk of outliers. Thus, we study the problem of improving the accuracy of long-term multivariate time series forecasting by enabling a model to capture the dynamic correlations among subsequences and mitigate the impact of disrupted correlations.

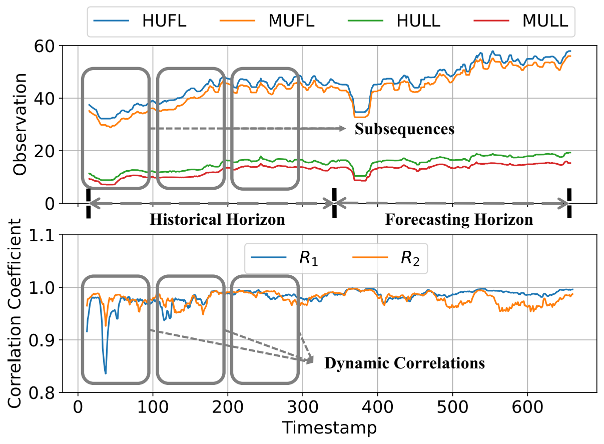
Three main branches of methods for long-term multivariate time exist: methods based on channel-independent mechanisms [6, 25, 27], linear models [11, 21, 46], and transformer-based methods that consider correlations [24, 36, 47]. The channel-independence and linear models do not capture correlations among variables; therefore, they are unaffected by dynamically changing and disrupted correlations. These methods result in the loss of important correlation information. In contrast, assuming that capturing correlations among variables is crucial for improving prediction accuracy, transformer-based methods use attention mechanisms to capture correlations among variables.

In spite of the notable progress achieved by current methods at long-term forecasting, two important challenges persist.

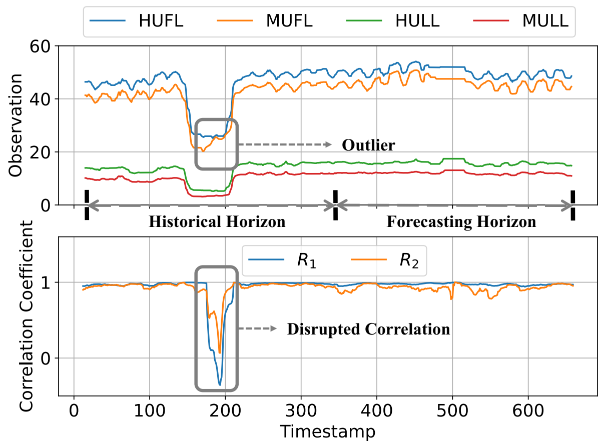
Challenge 1: It is difficult for existing methods to capture dynamic correlations while mitigating the impact of disrupted correlations in

*: Corresponding author.

Proceedings of the VLDB Endowment, Vol. 18, No. 2 ISSN 2150-8097.
doi:10.14778/3705829.3705842



(a) Dynamic correlations. The Average $R_1 = 0.995$ and $R_2 = 0.990$.



(b) Disrupted correlation. The Average $R_1 = 0.908$ and $R_2 = 0.963$.

Figure 1: Two samples from ETTh2 dataset. R_1 is the Pearson correlation coefficient of variables HUFL and MUFL over time, and R_2 is that of HULL and MULL. These are computed using a sliding window of length 24 with step length 1.

long-term time series. Existing methods [36, 47] utilize the attention mechanism to model correlations among variables. However, when capturing dynamic correlations, methods also capture disrupted correlations because the attention mechanism tends to assign higher weights to outliers [2]. This reduces model robustness [7].

Challenge 2: It is difficult for existing methods to understand and establish the association between local and global information. Global information is the collective result of all local information, while local information is simultaneously influenced by global information. Understanding and modeling this association enables improved capture of spatiotemporal features, thereby enhancing prediction accuracy. Existing methods only consider global information [24, 42] or focus solely on local information [36, 47].

To address these challenges, we propose a novel transformer model named Memory Guided Transformer (Memformer).

Addressing challenge 1: To enable the model to capture dynamic correlations while mitigating the impact of disrupted correlations, the Memformer employs a two-tiered framework that integrates patch-wise recurrent graph learning and global attention. Patch-wise recurrent graph learning divides a time series into patches and constructs an independent relational graph for each patch. The aggregation mechanism of graph representation learning learns correlations by aggregating the features of nodes and their neighbors, thereby avoiding the assignment of higher weights to outliers. Then, dynamic correlations can be captured by multiple graphs. In particular, constructing correlations at the patch level rather than at the timestamp level leverages multi-scale information and reduces both time and space complexity. Next, compared to traditional attention mechanisms, global attention imposes a global information constraint that differs from the traditional explicit regularization terms included in loss functions; instead, it implicitly acts as a regularization term on the features. When minimizing the loss, the model must consider the loss of global information, thus mitigating the impact of disrupted correlations and enhancing robustness.

Addressing challenge 2: To enable a model to understand and establish the association between local and global information, we propose a novel memory network named Alternating Memory Enhancer (AME). Memory networks emulate the workings of human memory by constructing a memory and enabling models to read it. Such memories can learn local and global information from the data, and their interactions establish associations. AME includes local and global enhancers. The local enhancer provides local information to the Memformer to capture dynamic correlations, while the global enhancer offers global information to enhance model robustness. The enhancers share a global memory, and the information they output is influenced by this memory, thereby establishing an association between local and global information. Further, to address the uneven convergence speeds of the two enhancers, we propose an alternating training mechanism that balances the different training difficulties of the two.

In summary, we make three main contributions:

- We propose a transformer model named Memformer, which employs patch-wise recurrent graph learning to capture dynamic correlations and global attention to mitigate the impact of disrupted correlations, thereby enhancing robustness.
- We propose a memory network named AME, which establishes the association between local and global information and provides this to the Memformer, thereby further enhancing prediction accuracy.
- We report on extensive experiments, finding that the Memformer can effectively capture dynamic correlations, is robust to outliers, and is capable of state-of-the-art prediction accuracy.

The remainder of the paper is organized as follows. Section 2 covers preliminaries. Section 3 details the proposed methodology. Section 4 reports on the experimental study. Section 5 reviews related work, and Section 6 concludes.

2 PRELIMINARIES

2.1 Multivariate Time Series Forecasting

A multivariate time series is a sequence of observations $\mathbf{x}_t \in \mathbb{R}^N$, where t is a timestamp and N indicates the number of variables in an observation. A model $\mathcal{F}$ uses a historical horizon $\mathbf{H} = \langle \mathbf{x}_{t-H+1}, \mathbf{x}_{t-H+2}, \ldots, \mathbf{x}_t \rangle$ to forecast a future horizon $\mathbf{F} = \langle \hat{\mathbf{x}}_{t+1}, \hat{\mathbf{x}}_{t+2}, \ldots, \hat{\mathbf{x}}_{t+F} \rangle$, where H and F are the lengths of historical and future horizons, respectively. The forecasting procedure is formulated as follows.

$$\mathcal{F}_{\Phi}(\mathbf{x}_{t-H+1}, \mathbf{x}_{t-H+2}, \ldots, \mathbf{x}_t) = (\hat{\mathbf{x}}_{t+1}, \hat{\mathbf{x}}_{t+2}, \ldots, \hat{\mathbf{x}}_{t+F}), \tag{1}$$

where $\mathbf{\Phi}$ represents the learnable parameters.

2.2 Graph Convolutional Network

A graph $G = (\mathcal{V}, \mathcal{E})$ comprises of a set of nodes $\mathcal{V}$, and a set of edges $\mathcal{E}$. In time series forecasting, graph nodes represent variables, and adjacent matrix $\mathbf{G}$ captures the correlations among the variables. Thus, if the nodes of two variables are not connected by an edge, the variables are uncorrelated. We employ a self-adaptive graph learning component [37]. This is because correlations among variables in multivariate time series are often hidden. A self-adaptive adjacency matrix $\mathbf{G}$ is learned during training. The learning procedure is defined as follows.

$$\mathbf{G} = \mathrm{softmax}(\mathrm{ReLU}(\mathbf{E}\mathbf{E}^{\mathrm{T}})), \tag{2}$$

where softmax(ReLU($\cdot$)) is a random walk that normalizes the non-negative part of the matrix product of a learnable embedding matrix $\mathbf{E}$ and its transposed matrix $\mathbf{E}^{\mathrm{T}}$. Graph Convolutional Networks utilize the diffusion convolution operation to allow features to diffuse through the graph structure to the variables represented by the nodes. It is defined as follows.

$$\mathbf{C} = \sum_{k=1}^{K} \mathbf{G}^k \mathbf{H} \mathbf{W}, \tag{3}$$

where K denotes the number of aggregations used in the diffusion convolution operation, determining the extent to which features of variables are propagated through the graph structure. Next, $\mathbf{H}$ denotes input temporal features, $\mathbf{C}$ denotes the correlated features output by the graph convolution, and $\mathbf{W}$ denotes the learnable parameters.

2.3 Self-attention

Self-attention is a core component of transformer-based models, allowing a feature at a certain position in a sequence to capture the relevances of features at other positions in the sequence. This mechanism enhances a model's ability to understand and process sequences by considering the interdependencies among the elements in the sequence. Given an input sequence $\mathbf{H} = \langle \mathbf{x}_{t-H+1}, \mathbf{x}_{t-H+2}, \ldots, \mathbf{x}_t \rangle$, the self-attention computation is defined as follows.

$$\mathrm{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathrm{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^{\mathrm{T}}}{\sqrt{D_k}}\right)\mathbf{V}, \tag{4}$$

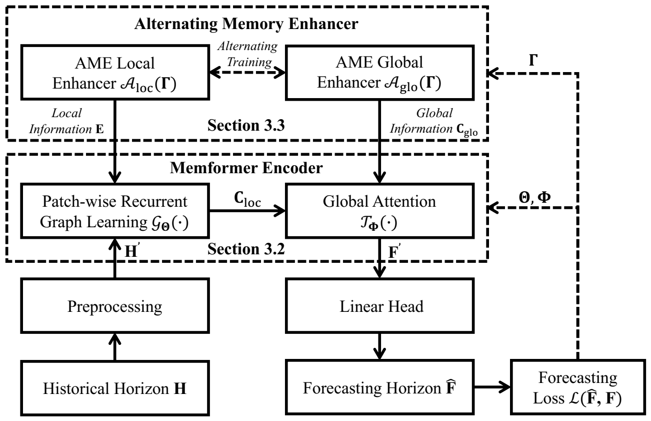
where $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$ denote the query, key, and value matrices obtained by linear transformations of the input sequence $\mathbf{H}$. Further, $\sqrt{D_k}$ is a scaling factor used to mitigate potential vanishing and exploding gradient issues caused by large dot products, and softmax($\cdot$) is the softmax function, which is used to convert similarities into attention scores. The attention score serves as weights applied to the value vectors in $\mathbf{V}$ to produce the final output.

3 METHODOLOGY

This section details the proposed Memformer. Section 3.1 introduces the backbone of the Memformer, focusing on how the Memformer utilizes the historical horizon to predict the forecasting horizon. Section 3.2 provides details on the Memformer encoder, including how patch-wise recurrent learning captures dynamic correlations using local information and how the global attention utilizes global information to enhance the robustness. Section 3.3 covers the AME, including how the local and global enhancers provide local and global information, and the alternating training mechanism of AME.

3.1 Memformer Backbone

As illustrated in Figure 2, the input to the Memformer consists of the historical horizon $\mathbf{H}$, and the learnable parameter set of the memory network $\mathbf{\Gamma}$. Its outputs are the forecasting horizon $\hat{\mathbf{F}}$ and the learned parameter sets $\mathbf{\Theta}$, $\mathbf{\Phi}$, and $\mathbf{\Gamma}$, where $\mathbf{\Theta}$, $\mathbf{\Phi}$, and $\mathbf{\Gamma}$ denote the parameter sets for patch-wise recurrent graph learning, global attention, and AME, respectively.



Alternating Memory Enhancer
AME Local Enhancer $\mathcal{A}_{\mathrm{loc}}(\mathbf{\Gamma})$
Alternating Training
AME Global Enhancer $\mathcal{A}_{\mathrm{glo}}(\mathbf{\Gamma})$
$\mathbf{\Gamma}$
Local Information $\mathbf{E}$
Section 3.3
Global Information $\mathbf{C}_{\mathrm{glo}}$
Memformer Encoder
Patch-wise Recurrent Graph Learning $\mathcal{G}_{\mathbf{\Theta}}(\cdot)$
$\mathbf{C}_{\mathrm{loc}}$
Global Attention $\mathcal{T}_{\mathbf{\Phi}}(\cdot)$
$\mathbf{\Theta}, \mathbf{\Phi}$
$\mathbf{H}'$
Section 3.2
$\mathbf{F}'$
Preprocessing
Linear Head
Historical Horizon $\mathbf{H}$
Forecasting Horizon $\hat{\mathbf{F}}$
Forecasting Loss $\mathcal{L}(\hat{\mathbf{F}}, \mathbf{F})$

Figure 2: Memformer backbone. The Memformer Encoder takes the preprocessed feature H′ as input, and the output representation F′ is used for prediction. The AME provides both local and global information to the Memformer Encoder to enhance the quality of features. To ensure stable training, the local and global enhancers are trained alternately.

The preprocessing for the input historical horizon $\mathbf{H}$ involves instance normalization. Applying instance normalization individually to each time series mitigates the issue of internal covariate shift, enabling models to more effectively grasp the intricate temporal dynamics inherent in time series. Instance normalization is defined as $\mathbf{H}' = (\mathbf{H} - \mu)/\sqrt{(\sigma^2 + \text{constant})}$, where $\mathbf{H}'$ denotes the preprocessed feature, μ and σ denote the mean and variance of the sample, respectively, and "constant" is a small positive real number included to ensure numerical stability.

The preprocessed feature $\mathbf{H}'$ enters the Memformer encoder, which performs patch-wise recurrent graph learning $\mathcal{G}_{\Theta}(\cdot)$ and global attention $\mathcal{T}_{\Phi}(\cdot)$. The patch-wise learning captures dynamic correlations among the input features $\mathbf{H}'$ using the local information $\mathbf{E}$ provided by the AME local enhancer $\mathcal{A}_{\text{loc}}(\cdot)$, outputting local correlated features $\mathbf{C}_{\text{loc}}$. Global attention utilizes the global information $\mathbf{C}_{\text{glo}}$ provided by the AME global enhancer $\mathcal{A}_{\text{glo}}(\cdot)$ to impose constraints on the input features $\mathbf{C}_{\text{loc}}$ to enhance model robustness, with the output features denoted as $\mathbf{F}'$. To ensure stable training of AME, we train its local and global encoders alternately. For example, during the first iteration, we freeze the parameters of the global encoder and train the parameters of the Memformer Encoder and the local encoder. In the next iteration, we freeze the parameters of the local encoder and train the parameters of the Memformer Encoder and the global encoder. This process proceeds until all parameters converge. Via the collaboration of the Memformer encoder and AME, the Memformer captures dynamic correlations and also integrates robust constraints into the representation $\mathbf{F}'$. Sections 3.2 and 3.3 cover the Memformer encoder and AME.

The representation $\mathbf{F}'$ undergoes a linear head, which maps the model's extracted features to the target output space, defined as $\hat{\mathbf{F}} = \mathbf{W}\mathbf{F}' + \mathbf{b}$, where $\mathbf{W}$ denotes the weights of the linear layer and $\mathbf{b}$ is a bias term. Then, the forecasting horizon $\hat{\mathbf{F}}$ and the ground truth $\mathbf{F}$ are used to measure the discrepancy between the model's predictions and the actual target values. Specifically, we use the mean absolute error (MAE) loss. The loss for each time series is gathered and averaged over N variables to get the overall objective loss, as follows.

$$\mathcal{L}(\hat{\mathbf{F}}, \mathbf{F}) = \mathbb{E}\left(\frac{1}{N}\sum_{n=1}^{N}|\hat{\mathbf{F}}_{t:t+F}^{(n)} - \mathbf{F}_{t:t+F}^{(n)}|_1\right), \tag{5}$$

where $\mathbb{E}(\cdot)$ denotes the expectation function used to obtain the expected value of the L1 norm $|\cdot|_1$ between the forecasting horizon $\hat{\mathbf{F}}_{t:t+F}^{(n)}$ and the ground truth $\mathbf{F}_{t:t+F}^{(n)}$ for the n-th variable.

3.2 The Memformer Encoder

Delving into the details of the Memformer encoder, we focus on the principles of patch-wise recurrent graph learning and global attention. Specifically, we elaborate on how patch-wise recurrent graph learning effectively captures dynamic correlations. Moreover, we consider how global attention uses global information to enhance robustness.

3.2.1 Patch-wise Recurrent Graph Learning. The capture of correlations in long-term multivariate time series are crucial to achieve accurate forecasting due to the inherent relationships among variables that evolve over time. Patch-wise recurrent graph learning constructs dynamic correlations at the patch level. We incorporate a recurrent structure among patches to capture complex temporal dynamics.

In Figure 3, the left side represents the patch-wise recurrent graph learning of the Memformer, which takes a preprocessed multivariate time series $\mathbf{H}' = \langle \mathbf{x}_1', \mathbf{x}_2', \ldots, \mathbf{x}_H' \rangle \in \mathbb{R}^{H\times N}$ and a series of learnable local information $\mathbf{E} = \langle \mathbf{E}_1, \mathbf{E}_2, \ldots, \mathbf{E}_P \rangle \in \mathbb{R}^{P\times N\times M}$ as inputs, where P is the number of patches into which a time series is segmented, and M denotes the dimensionality of the information. It outputs a local correlated feature $\mathbf{C}_{\text{loc}} = \langle \mathbf{C}_{\text{loc}1}, \mathbf{C}_{\text{loc}2}, \ldots, \mathbf{C}_{\text{loc}P} \rangle \in \mathbb{R}^{P\times T\times N}$, which diffuses the correlations among multiple variables, and where T is the dimensionality of each correlated feature. Before the diffusion convolution layer, we segment the time series into patches and obtain corresponding graphs, as the input to the diffusion convolution layer consists of patch-wise temporal features and graphs.

Using a stride S and a patch size T, we segment $\mathbf{H}'$ into patches denoted as $\mathbf{P} = \langle \mathbf{P}_1, \mathbf{P}_2, \ldots, \mathbf{P}_P \rangle$, where the i-th patch is $\mathbf{P}_i \in \mathbb{R}^{T\times N}$. When $S \geq T$, patches are disjoint. Hence, the number of patches P is $P = \lfloor \frac{H-T}{S} \rfloor + 2$. Then, unlike traditional adaptive graph learning, we acquire graphs corresponding to each patch. For the i-th patch, the corresponding graph $\mathbf{G}_i \in \mathbb{R}^{N\times N}$ is defined as follows.

$$\mathbf{G}_i = \text{softmax}(\text{ReLU}(\mathbf{E}_i\mathbf{E}_i^{\text{T}})), \tag{6}$$

where $\mathbf{E}_i\mathbf{E}_i^{\text{T}}$ is the matrix product of the local information $\mathbf{E}_i$ and the transposed local information $\mathbf{E}_i^{\text{T}}$.

After obtaining a series of temporal features $\mathbf{P} = \langle \mathbf{P}_1, \mathbf{P}_2, \ldots, \mathbf{P}_P \rangle$ and their corresponding graphs $\mathbf{G} = \langle \mathbf{G}_1, \mathbf{G}_2, \ldots, \mathbf{G}_P \rangle$, these features are processed through a recurrent diffusion convolutional layer to capture the dynamic correlations. Apart from the correlations among variables, temporal dynamics exist among the patches. Thus, apart from utilizing the diffusion convolution operation, we use a gated recurrent unit, a recurrent neural network, to capture the temporal dynamics among patches. The whole procedure is defined as follows.

$$\begin{aligned}
z_i &= \text{sigmoid}(\sum_{k=1}^{K}\mathbf{G}_i^k[\mathbf{P}_i, \mathbf{C}_{\text{loc}(i-1)}]\mathbf{\Theta}_z + \mathbf{b}_z) \\
r_i &= \text{sigmoid}(\sum_{k=1}^{K}\mathbf{G}_i^k[\mathbf{P}_i, \mathbf{C}_{\text{loc}(i-1)}]\mathbf{\Theta}_r + \mathbf{b}_r) \\
\widetilde{\mathbf{C}_i} &= \tanh(\sum_{k=1}^{K}\mathbf{G}_i^k[\mathbf{P}_i, (r_i \odot \mathbf{C}_{\text{loc}(i-1)})]\mathbf{\Theta}_{\widetilde{\mathbf{C}}} + \mathbf{b}_{\widetilde{\mathbf{C}}}) \\
\mathbf{C}_{\text{loc}i} &= z_i \odot \mathbf{C}_{\text{loc}(i-1)} + (1 - z_i) \odot \widetilde{\mathbf{C}_i},
\end{aligned} \tag{7}$$

where sigmoid$(\cdot)$ and tanh$(\cdot)$ denote the sigmoid and hyperbolic tangent activation functions; $\mathbf{\Theta} = \{\mathbf{\Theta}_z, \mathbf{\Theta}_r, \mathbf{\Theta}_{\widetilde{\mathbf{C}_i}}, \mathbf{b}_z, \mathbf{b}_r, \mathbf{b}_{\widetilde{\mathbf{C}_i}}\}$ denotes the set of learnable parameters within the recurrent units; z_i and r_i denote the update and reset gates, respectively, of the i-th unit; and $\mathbf{C}_{\text{loc}i}$ denotes the local correlated feature of the i-th patch. At this point, the Memformer encoder captures each dynamic correlation at each patch and utilizes a recurrent structure to capture their temporal dynamics.

3.2.2 Global Attention. Modeling correlations independently for each patch increases the risk of the model being affected by outliers. Applying constraints similar to regularization terms to features using global information is an effective way to mitigate the impact of disturbed correlation. Therefore, we introduce global attention, which utilizes the global information provided by AME to impose constraints on the attention scores. The attention scores with global constraints act on the value matrix, producing more robust temporal features.

The right side in Figure 3 shows the global attention of the Memformer, with its input being the transposed local correlated features $\mathbf{C}_{\text{loc}}^{\text{T}} = \langle \mathbf{C}_{\text{loc}}^{(1)}, \mathbf{C}_{\text{loc}}^{(2)}, \ldots, \mathbf{C}_{\text{loc}}^{(N)} \rangle \in \mathbb{R}^{N\times T\times P}$ and global information

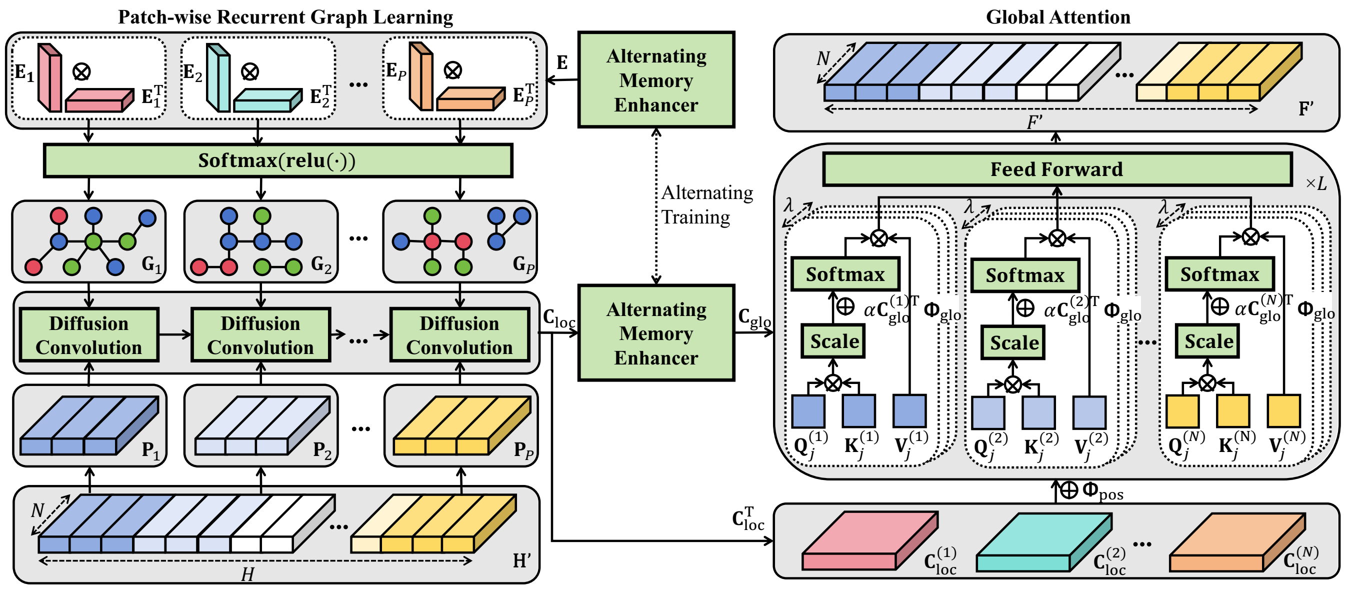


Figure 3: The Memformer encoder.

$C_{glo} = \langle C_{glo}^{(1)}, C_{glo}^{(2)}, \ldots, C_{glo}^{(N)} \rangle \in \mathbb{R}^{N \times D_{glo} \times P}$, along with the position encoding of the patches $\Phi_{pos} \in \mathbb{R}^{D_{att} \times P}$, where D_{glo} and D_{att} denote the hyperparameters for the tensor dimensionalities of C_{glo} and Φ_{pos}, respectively. The output of the global attention is the representation $\mathbf{F}' \in \mathbb{R}^{F' \times N}$, where F' is the temporal dimension of the representation. The representation $\mathbf{F}'$ is then passed through the linear head and flattened to match the forecasting horizon with length F.

The transposed local correlated features C_{loc}^T first pass a linear transformation with an added position embedding Φ_{pos} to map the features into a latent space of dimensionality D_{att}. Then, the features transformed through the linear operation are converted into the query, key, and value matrices for the global attention, denoted as $\mathbf{Q} \in \mathbb{R}^{N \times \lambda \times P \times D_{att}}$, $\mathbf{K} \in \mathbb{R}^{N \times \lambda \times P \times D_{att}}$, and $\mathbf{V} \in \mathbb{R}^{N \times \lambda \times P \times D_{att}}$, respectively, where λ is the number of heads. The transformation to get the j-th head and n-th variable of matrices $\mathbf{Q}_j^{(n)}$, $\mathbf{K}_j^{(n)}$, and $\mathbf{V}_j^{(n)}$ is defined as follows.

$$\begin{aligned} \mathbf{Q}_j^{(n)} &= [\Phi^P C_{loc}^{(n)} + \Phi_{pos}]^T \Phi_j^Q \\ \mathbf{K}_j^{(n)} &= [\Phi^P C_{loc}^{(n)} + \Phi_{pos}]^T \Phi_j^K \\ \mathbf{V}_j^{(n)} &= [\Phi^P C_{loc}^{(n)} + \Phi_{pos}]^T \Phi_j^V, \end{aligned} \tag{8}$$

where $\Phi^P \in \mathbb{R}^{D_{att} \times T}$ denotes the learnable parameter for the linear transformation, and $\Phi_j^Q \in \mathbb{R}^{D_{att} \times D_k}$, $\Phi_j^K \in \mathbb{R}^{D_{att} \times D_k}$, and $\Phi_j^V \in \mathbb{R}^{D_{att} \times D_{att}}$ represent the learnable parameters that generate the query, key, and value matrices, respectively, where D_k is the dimensionality of the query and key matrices.

The global attention consists of L residual-connected attention blocks, as illustrated in Figure 3. Unlike the multi-head attention block in the vanilla transformer, we introduce global information C_{glo} when integrating features from the query, key, and value matrices. The process is defined as follows.

$$\begin{aligned} \mathbf{O}_j^{(n)} &= \text{Attention}(\mathbf{Q}_j^{(n)}, \mathbf{K}_j^{(n)}, \mathbf{V}_j^{(n)}, \mathbf{G}_{glo}^{(n)}) \\ &= \left[\text{softmax}\left(\frac{\mathbf{Q}_j^{(n)} \mathbf{K}_j^{(n)T}}{\sqrt{D_k}} \right) + \alpha C_{glo}^{(n)T} \Phi_{glo} \right] \mathbf{V}_j^{(n)}, \end{aligned} \tag{9}$$

where $\mathbf{O}_j^{(n)} \in \mathbb{R}^{P \times D_{att}}$ is the output of the attention operation, $\Phi_{glo} \in \mathbb{R}^{D_{glo} \times P}$ is a linear transformation that maps the global information to the latent space of the attention, $\frac{1}{\sqrt{D_k}}$ is the scale operation that scales the attention score to stabilize gradients during training, and α is a hyperparameter that controls the importance of global information. The output $\mathbf{O}_j^{(n)}$ of the attention operation needs to pass the two layers of a fully connected feedforward neural network, defined as follows.

$$\mathbf{H}_{FF}^l = \text{ReLU}(\mathbf{O}_j^{(n)} \Phi_{FF1}^l + \mathbf{b}_{FF1}^l) \Phi_{FF2}^l + \mathbf{b}_{FF2}^l, \tag{10}$$

where ReLU(·) denotes the Rectified Linear Unit activation function, $\mathbf{H}_{FF}^l$ is the output hidden states of the l-th attention block, Φ_{FF1}^l and Φ_{FF2}^l are learnable parameter matrices, and $\mathbf{b}_{FF1}^l$ and $\mathbf{b}_{FF2}^l$ are learnable biases in the l-th attention block. Residual connections exist between the blocks in the global attention, and they are defined as follows.

$$\mathbf{H}_{FF}^l = \mathbf{H}_{FF}^{l-1} + \text{LayerNorm}(\text{Attention}(\mathbf{H}_{FF}^{l-1})), \tag{11}$$

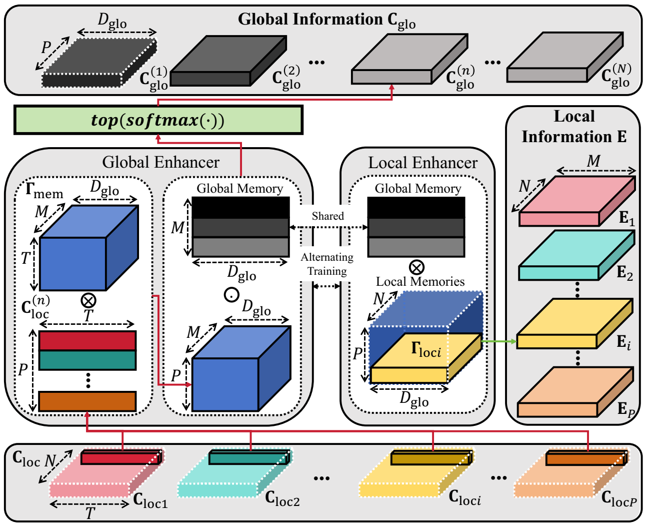
where LayerNorm(·) represents the layer normalization that aims to reduce internal covariate shift and accelerate training. After the L attention blocks, the global attention finally outputs the representation $\mathbf{F}'$. In the global attention, all learnable parameters are grouped into a set $\Phi = \{\Phi_{pos}, \Phi^P, \Phi_j^Q, \Phi_j^K, \Phi_j^V, \Phi_{glo}, \Phi_{FF1}, \Phi_{FF2}, \mathbf{b}_{FF1}, \mathbf{b}_{FF2}\}$ for convenience of description.

3.3 Alternating Memory Enhancer

We proceed to detail the AME. Specifically, we describe how the AME's local and global enhancers construct local and global information. Additionally, we describe the alternating training mechanism.

3.3.1 Structure of the AME. The local and global information are crucial for the Memformer encoder to be able to capture dynamic correlations and to be robust to outliers. We propose a novel component, AME, a memory network designed for understanding and establishing associations between local and global information. It includes local and global enhancers.

The local enhancer takes as input the global memory $\mathbf{\Gamma}_{\text{glo}} \in \mathbb{R}^{M\times D_{\text{glo}}}$ and the local memory $\mathbf{\Gamma}_{\text{loc}} = \{\mathbf{\Gamma}_{\text{loc}1}, \mathbf{\Gamma}_{\text{loc}2}, \dots, \mathbf{\Gamma}_{\text{loc}i}, \dots, \mathbf{\Gamma}_{\text{loc}P}\} \in \mathbb{R}^{P\times N\times D_{\text{glo}}}$, and its output is the local information $\mathbf{E}$. These memories are learnable parameters. The right side of Figure 4 illustrates the generation local information $\mathbf{E}_i$ corresponding to the i-th patch in the patch-wise recurrent graph learning, and the process is defined as $\mathbf{E}_i = \mathbf{\Gamma}_{\text{loc}i}\mathbf{\Gamma}_{\text{glo}}^{\text{T}}$. The local information requires local and global memories because each patch's local correlation possesses local specificity and global universality. The construction method can integrate these two types of memories.



Global Information $\mathbf{C}_{\text{glo}}$
D_{glo}
P
$\mathbf{C}_{\text{glo}}^{(1)}$
$\mathbf{C}_{\text{glo}}^{(2)}$
$\mathbf{C}_{\text{glo}}^{(n)}$
$\mathbf{C}_{\text{glo}}^{(N)}$
$top(softmax(\cdot))$
Global Enhancer
$\mathbf{\Gamma}_{\text{mem}}$
M
T
$\mathbf{C}_{\text{loc}}^{(n)}$
Global Memory
Shared
Alternating Training
Local Enhancer
Global Memory
Local Memories
N
$\mathbf{\Gamma}_{\text{loc}i}$
Local Information E
$\mathbf{E}_1$
$\mathbf{E}_2$
$\mathbf{E}_i$
$\mathbf{E}_P$
$\mathbf{C}_{\text{loc}}$
$\mathbf{C}_{\text{loc}1}$
$\mathbf{C}_{\text{loc}2}$
$\mathbf{C}_{\text{loc}i}$
$\mathbf{C}_{\text{loc}P}$

Figure 4: Alternating Memory Enhancer.

The global enhancer receives the local correlated features $\mathbf{C}_{\text{loc}}$ and the global memory $\mathbf{\Gamma}_{\text{glo}}$ as input and then outputs the global information $\mathbf{C}_{\text{glo}}$. The global memory $\mathbf{\Gamma}_{\text{glo}}$ shared between the local and global enhancers ensures that the local enhancer is constrained by the global information when generating local information. The left side of Figure 4 depicts how the n-th variable of the local correlated features $\mathbf{C}_{\text{loc}}^{(n)}$ is transformed into the corresponding global information $\mathbf{C}_{\text{glo}}^{(n)}$ through the global enhancer. $\mathbf{C}_{\text{loc}}^{(n)} \in \mathbb{R}^{P\times T}$ initially undergoes a linear transformation to project the tensor dimensions into the space of the global memory. It is defined as follows.

$$\mathbf{I}^{(n)} = \mathbf{C}_{\text{loc}}^{(n)}\mathbf{\Gamma}_{\text{mem}} + \mathbf{b}_{\text{mem}}, \tag{12}$$

where $\mathbf{I}^{(n)} \in \mathbb{R}^{P\times M\times D_{\text{glo}}}$ denotes the output of the linear transformation, representing an inquiry tensor used to obtain the weights corresponding to the current inquiry across the M entries in the global memory. $\mathbf{\Gamma}_{\text{mem}} \in \mathbb{R}^{T\times M\times D_{\text{glo}}}$ and $\mathbf{b}_{\text{mem}}$ represent the learnable parameter tensor and bias in the linear transformation, respectively. The process of using the inquiry tensor $\mathbf{I}^{(n)}$ to obtain the weights is defined as follows.

$$\mathbf{a}_m^{(n)} = \frac{\exp(\mathbf{I}_m^{(n)}\mathbf{\Gamma}_{\text{glo}}^{\text{T}}[m])}{\sum_{m=1}^{M}\exp(\mathbf{I}_m^{(n)}\mathbf{\Gamma}_{\text{glo}}^{\text{T}}[m])}, \tag{13}$$

where $\mathbf{a}_m^{(n)} \in \mathbb{R}^{P\times 1}$ denotes the weight of the current inquiry for the m-th memory, $\mathbf{I}_m^{(n)} \in \mathbb{R}^{P\times D_{\text{glo}}}$ indicates the inquiry of the inquiry tensor for the m-th memory, $\mathbf{\Gamma}_{\text{glo}}[m] \in \mathbb{R}^{1\times D_{\text{glo}}}$ represents the m-th entry of the global memory, and $\exp(\cdot)$ is an exponential function. The n-th global information $\mathbf{C}_{\text{glo}}^{(n)}$ is the weighted sum of the top κ entries from the global memory with the highest weights for the current inquiry. The process is defined as follows.

$$\mathbf{C}_{\text{glo}}^{(n)} = \sum_{m=1}^{\kappa}\text{top}(\mathbf{a}^{(n)})\mathbf{\Gamma}_{\text{glo}}[m], \tag{14}$$

where function $\text{top}(\cdot)$ extracts the top κ entries based on their weights, facilitating a focus on the most pertinent information. This function enhances the salience of crucial data while mitigating the impact of less significant or extraneous data.

3.3.2 Alternating Training. Due to the generation of local information needing local and global memories, each iteration updates both the local and global memories simultaneously, potentially leading to unstable training and reduced model robustness. Therefore, AME applies an alternating training mechanism: in each iteration of model training, AME alternates between activating the training of local and global memories to stabilize the training process and enhance robustness.

As shown in Algorithm 1, the input to AME is the historical horizon and ground truth $\mathbf{H}$ and $\mathbf{F}$, the local and global memories $\mathbf{\Gamma}_{\text{loc}}$ and $\mathbf{\Gamma}_{\text{glo}}$, and hyperparameters including the local training step ϵ, and the learning rates of the local and global enhancers η_{loc} and η_{glo}. Its output includes the local and global information $\mathbf{E}$ and $\mathbf{C}_{\text{glo}}$, the learned local and global memories $\mathbf{\Gamma}_{\text{loc}}$ and $\mathbf{\Gamma}_{\text{glo}}$, the tensor $\mathbf{\Gamma}_{\text{mem}}$, and the bias $\mathbf{b}_{\text{mem}}$.

AME, as indicated in the first line in Algorithm 1, randomly initializes its learnable parameters $\mathbf{E}$, $\mathbf{\Gamma}_{\text{glo}}$, $\mathbf{\Gamma}_{\text{loc}}$, $\mathbf{\Gamma}_{\text{mem}}$, and bias $\mathbf{b}_{\text{mem}}$. Then, each epoch of AME training is divided into two phases, as indicated by lines 2 to 11 and 12 to 20: updating the local memory $\mathbf{\Gamma}_{\text{loc}}$ and the global memory $\mathbf{\Gamma}_{\text{glo}}$, along with the tensor $\mathbf{\Gamma}_{\text{mem}}$ and the bias $\mathbf{b}_{\text{mem}}$. These phases utilize different learning rates, η_{loc} and η_{glo}, to control the magnitude of parameter updates for the two types of memories. Specifically, given the higher dimensionality of the local memory compared to the global memory and the varying difficulty of their training, we introduce a hyperparameter, the local learning step ϵ, to allow more update iterations for the local memory, thereby balancing the convergence rates of the two memories.

Algorithm 1 AME alternating training

```
Input: Historical horizon and ground truth $\mathbf{H}$, $\mathbf{F}$; local and global memories $\boldsymbol{\Gamma}_{\mathrm{loc}}$, $\boldsymbol{\Gamma}_{\mathrm{glo}}$;
    local training step $\epsilon$; learning rates $\eta_{\mathrm{loc}}$, $\eta_{\mathrm{glo}}$ for local and global enhancers
Output: Local and global information $\mathbf{E}$, $\mathbf{C}_{\mathrm{glo}}$; learned local and global memories $\boldsymbol{\Gamma}_{\mathrm{loc}}$,
    $\boldsymbol{\Gamma}_{\mathrm{glo}}$, tensor $\boldsymbol{\Gamma}_{\mathrm{mem}}$, and bias $\mathbf{b}_{\mathrm{mem}}$
1: Initialisation: Initializing local and global memories $\boldsymbol{\Gamma}_{\mathrm{loc}}$, $\boldsymbol{\Gamma}_{\mathrm{glo}}$, tensor $\boldsymbol{\Gamma}_{\mathrm{mem}}$, and
   bias $\mathbf{b}_{\mathrm{mem}}$ randomly
2: while $\boldsymbol{\Gamma}_{\mathrm{loc}}$, $\boldsymbol{\Gamma}_{\mathrm{glo}}$, $\boldsymbol{\Gamma}_{\mathrm{mem}}$, and $\mathbf{b}_{\mathrm{mem}}$ are not converged do
3:     for iteration = 0 to $\epsilon$ do
4:         $\mathbf{H}' \leftarrow \mathrm{Preprocessing}(\mathbf{H})$
5:         $\mathbf{E} \leftarrow \mathcal{A}_{\mathrm{loc}}(\boldsymbol{\Gamma}_{\mathrm{loc}}, \boldsymbol{\Gamma}_{\mathrm{glo}})$
6:         $\mathbf{C}_{\mathrm{loc}} \leftarrow \mathcal{G}_{\Theta}(\mathbf{H}', \mathbf{E})$
7:         $\mathbf{C}_{\mathrm{glo}} \leftarrow \mathcal{A}_{\mathrm{glo}}(\mathbf{C}_{\mathrm{loc}}, \boldsymbol{\Gamma}_{\mathrm{glo}})$
8:         $\mathbf{F}' \leftarrow \mathcal{T}_{\Phi}(\mathbf{C}_{\mathrm{loc}}, \mathbf{C}_{\mathrm{glo}})$
9:         $\hat{\mathbf{F}} \leftarrow \mathrm{LinearHead}(\mathbf{F}')$
10:        $\boldsymbol{\Gamma}_{\mathrm{loc}} \leftarrow \boldsymbol{\Gamma}_{\mathrm{loc}} - \eta_{loc} \nabla_{\boldsymbol{\Gamma}_{\mathrm{loc}}} \mathcal{L}(\hat{\mathbf{F}}, \mathbf{F})$
11:    end for
12:    $\mathbf{H}' \leftarrow \mathrm{Preprocessing}(\mathbf{H})$
13:    $\mathbf{E} \leftarrow \mathcal{A}_{\mathrm{loc}}(\boldsymbol{\Gamma}_{\mathrm{loc}}, \boldsymbol{\Gamma}_{\mathrm{glo}})$
14:    $\mathbf{C}_{\mathrm{loc}} \leftarrow \mathcal{G}_{\Theta}(\mathbf{H}', \mathbf{E})$
15:    $\mathbf{C}_{\mathrm{glo}} \leftarrow \mathcal{A}_{\mathrm{glo}}(\mathbf{C}_{\mathrm{loc}}, \boldsymbol{\Gamma}_{\mathrm{glo}})$
16:    $\mathbf{F}' \leftarrow \mathcal{T}_{\Phi}(\mathbf{C}_{\mathrm{loc}}, \mathbf{C}_{\mathrm{glo}})$
17:    $\hat{\mathbf{F}} \leftarrow \mathrm{LinearHead}(\mathbf{F}')$
18:    $\boldsymbol{\Gamma}_{\mathrm{glo}} \leftarrow \boldsymbol{\Gamma}_{\mathrm{glo}} - \eta_{glo} \nabla_{\boldsymbol{\Gamma}_{\mathrm{glo}}} \mathcal{L}(\hat{\mathbf{F}}, \mathbf{F})$
19:    $\boldsymbol{\Gamma}_{\mathrm{mem}} \leftarrow \boldsymbol{\Gamma}_{\mathrm{mem}} - \eta_{glo} \nabla_{\boldsymbol{\Gamma}_{\mathrm{mem}}} \mathcal{L}(\hat{\mathbf{F}}, \mathbf{F})$,
20:    $\mathbf{b}_{\mathrm{mem}} \leftarrow \mathbf{b}_{\mathrm{mem}} - \eta_{glo} \nabla_{\mathbf{b}_{\mathrm{mem}}} \mathcal{L}(\hat{\mathbf{F}}, \mathbf{F})$
21: end while
```

4 EXPERIMENTAL STUDY

4.1 Experimental Setup

4.1.1 Datasets and Evaluation Metrics. We perform experimental studies on seven benchmark datasets.

Weather[1]**:** Weather records meteorological data throughout one year at a sampling frequency of once per 10 minutes.

Traffic[2]**:** Traffic records congestion data from the San Francisco Bay area. This dataset has a sampling frequency of once per hour and covers 48 months.

Electricity[3]**:** Electricity records the electricity consumption with a sampling frequency of once per 15 minutes.

ETT Datasets[4]**:** ETT data comprising four datasets: ETTh1, ETTh2, ETTm1, and ETTm2, which record the oil temperature and other associated variables. The sampling rates of ETTh1 and ETTh2 are once per hour, while ETTm1 and ETTm2 are once every 15 minutes.

Table 1: Dataset related statistics.

Dataset	N	Length	Split Ratio	H	F
Weather	21	52696	7:1:2	336	96~720
Traffic	862	17544	7:1:2	336	96~720
Electricity	321	26304	7:1:2	336	96~720
ETTh1	7	17420	7:1:2	336	96~720
ETTh2	7	17420	7:1:2	336	96~720
ETTm1	7	69680	7:1:2	336	96~720
ETTm2	7	69680	7:1:2	336	96~720

Table 1 presents statistics of the seven datasets. Here, N is the total count of variables, while "Length" is the count of timestamps. The proportion of data allocated for training, validation, and testing is given by "Split Ratio." The lengths of the historical and forecasting horizons are denoted by H and F, respectively.

Metrics: To evaluate the prediction performance of Memformer and other baseline models, we utilize widely recognized time series forecasting measures, namely mean absolute error (MAE) and mean squared error (MSE) [27, 46].

4.1.2 Baselines. We select models proficient at long-term sequence forecasting as baselines. The channel-independence and linear models do not account for correlation among multiple variables. The transformer models based on channel-independent mechanisms are ModernTCN [25] and PatchTST [27], while the linear models include DLinear and NLinear [46]. We also consider long-term time series forecasting models that can capture correlations, including iTransformer [24], CARD [36], Crossformer [47], and MTGNN [42].

4.1.3 Implementation Details. All experiments are conducted on a server with Linux 18.04, an Intel Xeon W-2155 CPU @ 3.30GHz, and two RTX GPUs with 24GB memory.

We use grid search combined with a greedy strategy to identify the optimal hyperparameters for the above methods. This strategy involves adjusting one hyperparameter at each training procedure while keeping the remaining hyperparameters constant. Additionally, we carefully tune the hyperparameters based on the recommendations of the baseline models. We utilize the Adam optimizer with learning rates η, η_{loc}, and η_{glo} chosen from the set $\{0.01, 0.003, 0.001, 0.0003, 0.0001\}$. The entries of global memory M are chosen among $\{2, 4, 8, 16, 32, 64\}$. The dimension of the global memory D_{glo} is chosen among $\{4, 8, 16, 32, 64, 128\}$. The weight of global information α is chosen among $\{0.1, 0.2, 0.5, 1.0, 2.0\}$. Additionally, the local learning step ϵ is chosen among $\{1, 2, 3, 5, 10\}$.

4.2 Comparison of Forecasting Accuracy

We compare the prediction accuracy of all methods on publicly available real-world benchmark datasets. The experimental setup follows that of the baselines, where the length H of the historical horizon is fixed at 336 and the forecasting horizon F is set to 96, 192, 336, or 720. The results are shown in Table 2. The best result is in boldface, while the second-best result is underlined. Memformer achieves the best prediction accuracy across almost all datasets.

Models that capture correlations, such as iTransformer, CARD, Crossformer, and MTGNN, achieve some second-best results when the length of the historical horizon is up to 192, indicating that correlation contributes to prediction accuracy. However, as the length of the historical horizon increases, the models face more complex and dynamic correlations and greater risks of distribution shift outliers, leading to a rapid decay in prediction accuracy.

Baselines that do not capture correlations include transformer models based on channel-independence mechanisms and linear models. Transformer models based on channel-independence mechanisms effectively incorporate multiscale information and achieve many of the second-best results. However, as they do not capture correlations, their prediction accuracy is not as high as that of Memformer. The structures of linear models are simple, making it difficult to achieve good prediction accuracy on large-scale datasets such as Traffic and Electricity.

[1]Weather Dataset (last accessed on October 18, 2024).
[2]Traffic Dataset (last accessed on October 18, 2024).
[3]Electricity Dataset (last accessed on October 18, 2024)
[4]ETT Datasets (last accessed on October 18, 2024)

Table 2: Forecasting comparison of methods. Best results are in boldface, and second-best results are underlined.

Models		Memformer		ModernTCN		PatchTST		NLinear		DLinear		iTransformer		CARD		Crossformer		MTGNN	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Weather	96	0.151	**0.185**	0.155	0.201	0.152	0.199	0.182	0.232	0.176	0.237	0.174	0.214	$\underline{0.150}$	$\underline{0.188}$	**0.145**	0.211	0.342	0.385
	192	$\underline{0.197}$	**0.231**	0.198	0.245	$\underline{0.197}$	0.243	0.225	0.269	0.220	0.282	0.221	0.254	0.202	$\underline{0.238}$	**0.190**	0.259	0.427	0.445
	336	**0.247**	**0.274**	0.251	0.286	$\underline{0.249}$	0.283	0.271	0.301	0.265	0.319	0.278	0.296	0.260	$\underline{0.282}$	0.259	0.326	0.506	0.523
	720	**0.318**	**0.326**	0.321	0.336	$\underline{0.320}$	$\underline{0.335}$	0.338	0.348	0.323	0.362	0.358	0.347	0.343	0.353	0.332	0.382	0.510	0.527
Traffic	96	**0.361**	**0.230**	0.368	0.253	$\underline{0.367}$	$\underline{0.251}$	0.410	0.279	0.410	0.282	0.395	0.268	0.419	0.269	0.511	0.292	0.516	0.308
	192	**0.381**	**0.239**	$\underline{0.384}$	0.261	0.385	$\underline{0.259}$	0.423	0.284	0.423	0.287	0.417	0.276	0.443	0.276	0.523	0.311	0.534	0.324
	336	**0.394**	**0.245**	$\underline{0.397}$	0.270	0.398	$\underline{0.265}$	0.435	0.290	0.436	0.296	0.433	0.283	0.460	0.283	0.530	0.300	0.540	0.335
	720	**0.432**	**0.267**	0.440	0.296	$\underline{0.434}$	$\underline{0.287}$	0.464	0.307	0.466	0.315	0.467	0.302	0.490	0.299	0.573	0.313	0.557	0.343
Electricity	96	**0.130**	**0.217**	$\underline{0.131}$	0.228	**0.130**	$\underline{0.222}$	0.141	0.237	0.140	0.237	0.132	0.228	0.141	0.233	0.186	0.281	0.202	0.314
	192	**0.147**	**0.232**	0.150	0.242	$\underline{0.148}$	$\underline{0.240}$	0.154	0.248	0.153	0.249	0.154	0.249	0.160	0.250	0.208	0.300	0.266	0.349
	336	**0.162**	**0.249**	0.171	0.265	$\underline{0.167}$	$\underline{0.261}$	0.171	0.265	0.169	0.267	0.172	0.267	0.173	0.263	0.323	0.369	0.328	0.373
	720	**0.199**	**0.281**	0.203	0.294	0.202	0.291	0.210	0.297	0.203	0.301	0.204	0.296	$\underline{0.197}$	$\underline{0.284}$	0.404	0.423	0.422	0.410
ETTh1	96	**0.362**	**0.385**	0.382	0.401	0.375	0.399	$\underline{0.374}$	0.394	0.375	0.399	0.386	0.405	0.383	$\underline{0.391}$	0.377	0.419	0.401	0.442
	192	**0.386**	**0.404**	0.420	0.424	0.414	0.421	0.408	$\underline{0.415}$	$\underline{0.405}$	0.416	0.441	0.436	0.435	0.420	0.410	0.439	0.587	0.601
	336	**0.402**	**0.421**	$\underline{0.427}$	0.434	0.431	0.436	0.429	$\underline{0.427}$	0.439	0.443	0.487	0.458	0.479	0.442	0.440	0.461	0.736	0.643
	720	**0.436**	**0.452**	0.450	0.461	0.449	0.466	$\underline{0.440}$	$\underline{0.453}$	0.472	0.490	0.503	0.491	0.471	0.461	0.519	0.524	0.916	0.750
ETTh2	96	**0.264**	**0.321**	0.276	0.342	$\underline{0.274}$	0.336	0.277	0.338	0.289	0.353	0.297	0.349	0.281	$\underline{0.330}$	0.770	0.529	0.735	0.643
	192	**0.314**	**0.358**	0.340	0.381	$\underline{0.339}$	$\underline{0.379}$	0.344	0.381	0.383	0.418	0.380	0.400	0.363	0.381	0.848	0.657	0.859	0.717
	336	**0.312**	**0.364**	$\underline{0.329}$	$\underline{0.378}$	0.331	0.380	0.357	0.400	0.448	0.465	0.428	0.432	0.411	0.418	0.859	0.674	1.050	0.849
	720	**0.374**	**0.410**	0.392	0.433	$\underline{0.379}$	$\underline{0.422}$	0.394	0.436	0.605	0.551	0.427	0.445	0.416	0.431	1.221	0.825	1.336	0.963
ETTm1	96	**0.285**	**0.336**	0.292	0.346	$\underline{0.290}$	$\underline{0.342}$	0.306	0.348	0.299	0.343	0.334	0.368	0.316	0.347	0.320	0.373	0.428	0.446
	192	**0.323**	**0.358**	$\underline{0.332}$	0.368	$\underline{0.332}$	0.369	0.349	0.375	0.335	$\underline{0.365}$	0.377	0.391	0.363	0.370	0.372	0.411	0.551	0.505
	336	**0.365**	**0.381**	0.367	0.393	$\underline{0.366}$	0.392	0.375	0.388	0.369	$\underline{0.386}$	0.426	0.420	0.392	0.390	0.429	0.441	0.706	0.622
	720	**0.419**	**0.409**	0.422	0.429	$\underline{0.420}$	0.424	0.433	0.422	0.425	$\underline{0.421}$	0.491	0.459	0.458	0.425	0.573	0.531	0.982	0.764
ETTm2	96	**0.160**	**0.245**	0.166	0.256	$\underline{0.165}$	0.255	0.167	0.255	0.167	0.260	0.180	0.264	0.169	$\underline{0.248}$	0.254	0.348	0.442	0.483
	192	**0.215**	**0.285**	0.222	0.293	$\underline{0.220}$	$\underline{0.292}$	0.221	0.293	0.224	0.303	0.250	0.309	0.234	$\underline{0.292}$	0.370	0.433	0.642	0.570
	336	**0.263**	**0.317**	0.276	$\underline{0.327}$	0.278	0.329	$\underline{0.274}$	$\underline{0.327}$	0.281	0.342	0.311	0.348	0.294	0.339	0.511	0.527	0.726	0.658
	720	**0.350**	**0.372**	$\underline{0.365}$	$\underline{0.383}$	0.367	0.385	0.368	0.384	0.397	0.421	0.412	0.407	0.390	0.388	0.901	0.689	1.139	0.862

Table 3: Ablation study on Memformer.

Models		Memformer		w/o Graph		w/o Recurrent		w/o Local		w/o Global		w/o Sharing		w/o Alternating	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Weather	96	**0.151**	**0.185**	0.152	0.197	0.155	0.194	0.159	0.204	0.153	0.195	0.151	0.187	0.155	0.199
	192	**0.197**	**0.231**	0.200	0.235	0.204	0.248	0.199	0.255	0.202	0.238	0.199	0.234	0.202	0.235
	336	**0.247**	**0.274**	0.252	0.279	0.254	0.298	0.257	0.307	0.251	0.284	0.252	0.279	0.255	0.286
	720	**0.318**	**0.326**	0.334	0.341	0.332	0.355	0.364	0.380	0.323	0.333	0.324	0.335	0.334	0.360
Electricity	96	**0.130**	**0.217**	0.133	0.226	0.132	0.224	0.132	0.243	0.131	0.223	0.131	0.223	0.138	0.235
	192	**0.147**	**0.232**	0.154	0.245	0.155	0.248	0.153	0.250	0.150	0.238	0.152	0.241	0.158	0.252
	336	**0.162**	**0.249**	0.169	0.260	0.174	0.268	0.179	0.270	0.169	0.258	0.170	0.261	0.181	0.274
	720	**0.199**	**0.281**	0.208	0.299	0.220	0.317	0.231	0.341	0.205	0.293	0.210	0.300	0.229	0.339
ETTh2	96	**0.264**	**0.321**	0.271	0.329	0.269	0.326	0.322	0.369	0.266	0.324	0.266	0.324	0.294	0.347
	192	**0.314**	**0.358**	0.328	0.365	0.325	0.362	0.458	0.478	0.320	0.364	0.318	0.361	0.372	0.401
	336	**0.312**	**0.364**	0.329	0.376	0.334	0.381	0.530	0.517	0.317	0.370	0.319	0.370	0.380	0.419
	720	**0.374**	**0.410**	0.379	0.421	0.401	0.437	0.705	0.627	0.385	0.422	0.388	0.425	0.437	0.463

4.3 Ablation Study

To assess the effectiveness of the components making up Memformer, we use three datasets with varying numbers of variables: ETTh2 (7 variables), Weather (21 variables), and Electricity (321 variables).

The columns "w/o Graph" in Table 3 show the results of patch-wise recurrent graph learning when replacing the graph convolutional structure with a self-attention mechanism, where self-attention operates among variables. The prediction accuracy of "w/o Graph" exhibits less degradation on the dataset with fewer variables, ETTh2, while experiencing higher degradation on the dataset with many variables, Electricity. This indicates the advantage of using graph convolution on large-scale datasets.

The columns "w/o Recurrent" show the results of patch-wise recurrent graph learning after removing the gated recurrent unit structure, meaning that the patches no longer possess short-term and long-term memory across them. The results indicate less degradation in prediction accuracy for forecasting horizons 96 and 192, while a more pronounced decline is observed for forecasting horizons 336 and 720. This suggests that the recurrent structure among patches facilitates the capture of temporal dynamics among patches.

The columns "w/o Local" show the results of patch-wise recurrent graph learning after removal of the influence of the local enhancer, meaning that local information is no longer provided individually to each patch, but that all patches instead utilize the same static graph embedding. As the forecasting horizon increases, the prediction accuracy of "w/o Local" declines more rapidly than that of Memformer, indicating that the static graph embedding struggles to capture the dynamic changes in correlation over time.

The columns "w/o Global" show the results of global attention. Here, we remove the influence of the global information, by setting the hyperparameter α, controlling the weight of global information, to 0. We observe that removing the global information decreases prediction accuracy compared to that of Memformer, suggesting that global information is beneficial for time series forecasting.

The columns "w/o Sharing" cover the scenario where the local and global enhancers of AME no longer share a global memory; each local and global enhancer has its own global memory. Local memories are no longer constrained by global memory trained by the global enhancer, thus removing the regularization effect of global memory. The prediction accuracy of "w/o Sharing" decreases compared to that of Memformer, especially when the forecasting horizon is 720. This suggests that sharing global memory for the local and global enhancers serves as regularization, reducing the risk of overfitting.

The columns "w/o Alternating" show the results of AME when removing the alternating training mechanism, meaning that the learnable parameters in the local and global enhancers are no longer updated alternately for each training iteration. We observe a decrease in prediction accuracy for "w/o Alternating" compared to that of Memformer, indicating that AME struggles to achieve stable optimization of its learnable parameters to improve prediction accuracy.

4.4 Robustness Study

We proceed to assess the extent to which hierarchical capture of information at both local and global levels contributes to enhancing the robustness of the model, particularly its robustness to distribution shift outliers. Distribution shifts are common outliers in time series. In multivariate time series, distribution shift outliers may be both independent and dependent. Independent distribution shift outliers denote instances where multiple variables experience distribution shifts at different time points, while dependent distribution shift outliers denote instances where multiple variables undergo distribution shifts simultaneously.

We select the ETTh2 dataset, which exhibits clear physical relationships among variables, to investigate the impact of the two types of outliers on Memformer's robustness. The historical and forecasting horizons H and F are set to 336 and 96, respectively. We add 1, 2, 4, and 8 outliers to each sample in the dataset. The outliers are introduced by adding or subtracting five times the standard deviation of a randomly selected 32-length time window within the historical horizon, with a 50% probability of either adding or subtracting. Additionally, outliers are independently added to all variables within a sample. Then, the method of adding dependent distribution shift outliers is similar to that of adding independent outliers. We add outliers to all variables within the same time window.

As shown in Table 4, for the independent outliers, the prediction accuracies of all models decrease with the increase in the number of outliers. Specifically, iTransformer and MTGNN, which rely on static correlations, experience significant declines in prediction accuracy. This is because there are fixed physical relationships among variables in ETTh2 in the absence of outliers. The random occurrences of outliers disrupt these static physical relationships, making it challenging for static components to construct correlations effectively. Consequently, iTransformer and MTGNN exhibit the most significant decline in prediction accuracy. The linear models NLinear and DLinear also show sensitivity to outliers. The outliers disrupt the stationarity of the time series, leading to a significant reduction in accuracy. Memformer constructs dynamically changing correlations patch-wised and captures these features through the local enhancer, which helps alleviate the impact of outliers. Although the overall physical relationships among variables are disrupted, local relationships that can be captured by Memformer still exist. The findings indicate that Memformer achieves robustness to independent distribution shift outliers.

Then, for the dependent outliers, the prediction accuracy of models based on channel-independence mechanisms, such as ModernTCN and PatchTST, as well as dynamically correlated transformer models like Crossformer and CARD, decay under dependent distribution shifts similarly to their performance under independent shifts. However, the declines in prediction accuracy for iTransformer and MTGNN are less pronounced. Despite the alteration in time series stationarity caused by the dependent distribution shifts, the original physical relationships among the changed local variables are preserved, and static components can capture these physical relationships. Hence, the static components of iTransformer and MTGNN act as regularizers, reducing the decline in prediction accuracy caused by outliers. Furthermore, the linear models NLinear and DLinear experience a more severe decrease in prediction accuracy. Memformer's global memory retains global information and incorporates this regularization information into training via the global enhancer, thus mitigating such outliers. The findings indicate that Memformer is capable of robustness against dependent distribution shifts.

4.5 Dynamic Correlation Study

Using the ETTh2 dataset, which exhibits linear correlations among its variables, we investigate Memformer's ability to capture dynamic correlations. The length of the historical and forecasting horizons H and F are set to 336 and 96, respectively. We randomly select 1, 2, 4, and 8 time windows of length 32 and perform a base-10 logarithmic transformation on the variables HULL, MULL, and LULL. If a logarithmic transformation is applied to one of two variables that have a linear relationship, the linear relationship between the two variables transforms into an exponential relationship. Then, these variables' correlations with HUFL, MUFL, and LUFL in the selected time windows change from linear to exponential. In this way, we introduce dynamic correlations into the dataset.

As shown in Table 5, Memformer achieves the best prediction accuracy. Then, as the number of logarithmic transformations increases, the prediction accuracy of both Memformer and the baselines decreases, with Memformer exhibiting the smallest decline. Furthermore, the models capable of capturing dynamic correlations, CARD and Crossformer, exhibit small declines in prediction accuracy. Specifically, CARD achieves the second-best prediction accuracy. This highlights the necessity of modeling dynamic correlations.

4.6 Visualization

We proceed to visualize the predictions of Memformer and its dynamically changing correlations.

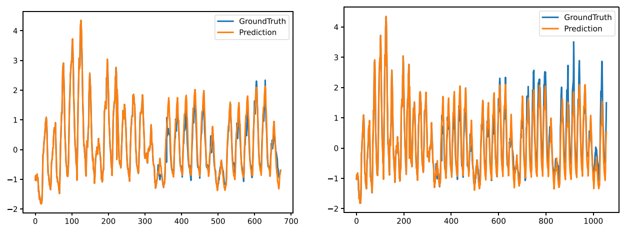
4.6.1 Forecasting. We randomly select a sample from Electricity, which has a large number of variables (321) and is long (26,304), to

Table 4: Accuracy across Distribution Shifts. Best results are in boldface, and second-best results are underlined.

Models		Memformer		ModernTCN		PatchTST		NLinear		DLinear		iTransformer		CARD		Crossformer		MTGNN	
Type	Outliers	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Independent	1	**0.271**	**0.328**	0.284	0.349	0.284	0.346	0.306	0.383	0.378	0.422	0.308	0.360	0.292	0.358	0.794	0.548	0.900	0.685
	2	**0.274**	**0.331**	0.288	0.356	0.288	0.352	0.353	0.411	0.490	0.498	0.356	0.431	0.299	0.368	0.814	0.578	0.993	0.731
	4	**0.278**	**0.340**	0.301	0.360	0.294	0.358	0.402	0.431	0.524	0.515	0.478	0.482	0.303	0.371	0.835	0.580	1.141	0.748
	8	**0.280**	**0.344**	0.311	0.374	0.308	0.369	0.465	0.480	0.572	0.543	0.552	0.543	0.313	0.375	0.886	0.599	1.464	0.902
Dependent	1	**0.265**	**0.325**	0.286	0.355	0.287	0.351	0.324	0.385	0.408	0.441	0.319	0.362	0.290	0.356	0.804	0.555	0.763	0.659
	2	**0.266**	**0.333**	0.296	0.357	0.288	0.352	0.327	0.386	0.414	0.452	0.325	0.371	0.302	0.363	0.810	0.572	0.771	0.680
	4	**0.276**	**0.336**	0.299	0.358	0.297	0.356	0.342	0.404	0.475	0.491	0.334	0.392	0.315	0.390	0.813	0.574	0.785	0.693
	8	**0.280**	**0.347**	0.314	0.381	0.309	0.375	0.484	0.538	0.758	0.673	0.352	0.395	0.324	0.406	0.825	0.588	0.858	0.709

Table 5: Accuracy across Dynamic Correlations. Best results are in boldface, and second-best results are underlined.

Models	Memformer		ModernTCN		PatchTST		NLinear		DLinear		iTransformer		CARD		Crossformer		MTGNN	
Transformations	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
1	**0.266**	**0.322**	0.282	0.364	0.290	0.356	0.335	0.393	0.405	0.436	0.325	0.374	0.285	0.336	0.777	0.534	0.778	0.667
2	**0.269**	**0.327**	0.301	0.369	0.293	0.360	0.339	0.396	0.409	0.442	0.338	0.386	0.287	0.340	0.784	0.540	0.792	0.695
4	**0.272**	**0.332**	0.305	0.372	0.303	0.364	0.343	0.407	0.457	0.478	0.359	0.412	0.291	0.345	0.792	0.547	0.808	0.704
8	**0.276**	**0.338**	0.319	0.385	0.317	0.380	0.502	0.546	0.693	0.574	0.399	0.431	0.296	0.358	0.806	0.560	0.883	0.718



(a) Forecasting Horizon $F = 336$. (b) Forecasting Horizon $F = 720$.

Figure 5: Prediction on the Electricity dataset.

assess the predictive ability of Memformer. As the forecasting horizon increases, the prediction task becomes more challenging. Figure 5 presents the prediction results of Memformer with forecasting horizons of F=336 and F=720. The figure shows that Memformer is capable of effective prediction on long-term time series.

4.6.2 Dynamic Correlations. We visualize the dynamic changes in correlation over time for a randomly selected sample from ETTh2. Figure 6(a)–(d) illustrate that the correlations of this sample from the 1st to the 40th patch are dynamically. We represent the corresponding graphs of these patches as heatmaps, where higher values indicate stronger positive correlations among variables, and lower values indicate stronger negative correlations among variables. Specifically, as shown in Figure 6(b), the correlation at the 14th patch of the time series is more distinct compared to other patches. Patch-wise recurrent graph learning enables the capture of this correlation.

4.7 Parameter Sensitivity Study

We conduct sensitivity analyses of four critical hyperparameters of Memformer, including the number of entries of global memory M, the dimensionality of the global memory D_{glo}, the weight of the global information α, the number of local learning steps ϵ, and the stride of patches S. Figure 7 reports the changes in MAE on ETTh2 when varying the hyperparameters, with the length of the historical horizon set to 336 and the length forecasting horizon set to 96, 192, 336, or 720.

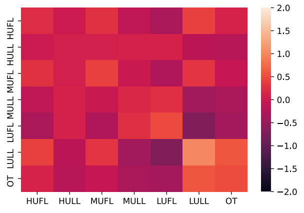
As seen in Figure 7(a), when M is 8 or higher, the MAE tends to stabilize. This indicates that the M of the memory network should not be set to be too small, as the top function automatically selects the most relevant memories to enhance features. Additionally, Figure 7(b) shows that D_{glo} achieves the best MAE when set to 32. This suggests that setting D_{glo} too small or too large may lead to underfitting or overfitting, requiring adjustment based on the distribution of the dataset to find the optimal D_{glo}.

Further, as shown in Figure 7(c), the value of α that yields the best MAE is 0.2. We find that α, as a regularization term, may hinder the capture of local information if its value is set too high, while a smaller α contributes to improving the generalization of Memformer. Then, as seen in Figure 7(d), ϵ achieves the best MAE when set to 5. At this point, Memformer effectively balances the difficulty of training between the local and global enhancers.

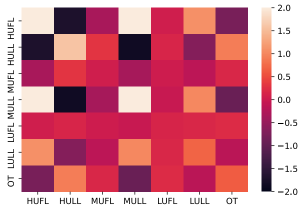
Finally, in Figure 7(e), we set patch size T to 32 and vary S among $T/4$, $T/2$, T, $2T$, and $4T$, i.e., 8, 16, 32, 64, and 128. The MAE of Memformer increases as S increases. This is because, with a higher S, the number of patches decreases, leading to a loss of valuable information. Furthermore, when $S \leq T$, the increase in Memformer's MAE is small, whereas when $S > T$, the increase is more pronounced. This is because, when $S \leq T$, all information in the time series is retained, whereas when $S > T$, some information is lost. We recommend setting the ratio of T to S to 2.

4.8 Scalability Study

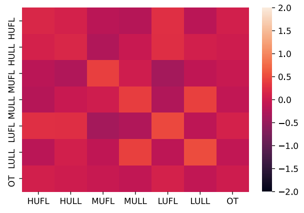
We proceed to investigate the scalability of Memformer with respect to the length H of the historical horizon and the number N of variables. Next, we provide the time and space complexity of Memformer. Additionally, we present a scalability experiment involving Memformer and the baselines. We use the large-scale dataset Electricity to assess scalability. The Electricity dataset consists of 321 variables, has a length of 26,304, and a total of 26,976 samples.



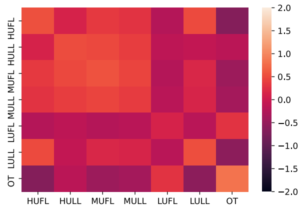
(a) Correlation of the 1-st Patch.



(b) Correlation of the 14-th Patch.

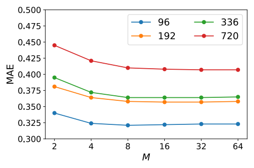


(c) Correlation of the 27-th Patch.

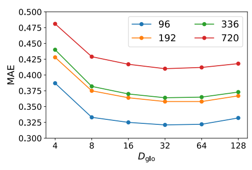


(d) Correlation of the 40-th Patch.

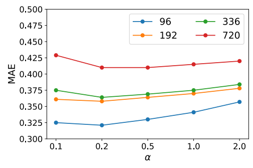
Figure 6: Visualization of dynamic correlations.



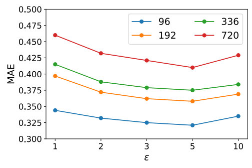
(a) MAE When Varying M.



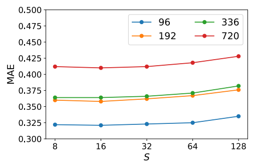
(b) MAE When Varying D_{glo}.



(c) MAE When Varying α.

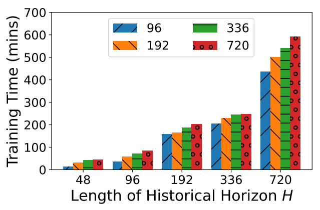


(d) MAE When Varying ϵ.

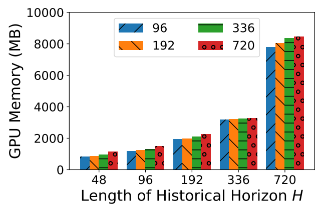


(e) MAE When Varying S.

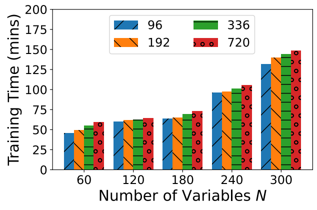
Figure 7: Parameter sensitivity of Memformer.



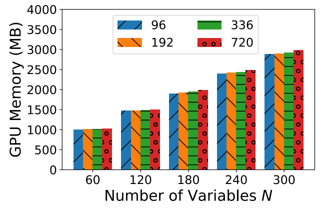
(a) Training Time When Varying H.



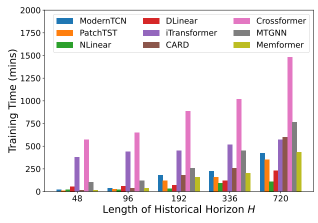
(b) Memory Use When Varying H.



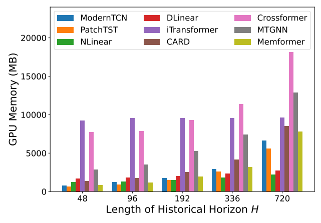
(c) Training Time When Varying N.



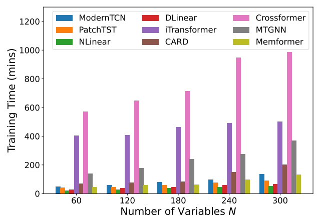
(d) Memory Use When Varying N.



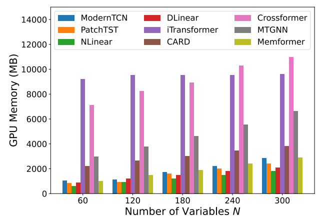
(e) Time Comparison When Varying H.



(f) Memory Comparison When Varying H.



(g) Time Comparison When Varying N.



(h) Memory Comparison When Varying N.

Figure 8: Scalability Study for Memformer and Baselines.

4.8.1 Length of the Historical Horizon. We utilize all variables in this experiment. As shown in Figure 8(a) and Figure 8(b), we vary H from 48 to 720 and measure the training time and GPU memory usage for four different F: 96, 192, 336, and 720. We observe that the time and space costs of Memformer are linearly correlated with H. This is due to the patch-wise processing of temporal features, and it is in contrast to the quadratic complexity associated with self-attention. The quadratic costs of time and space are reduced to linear costs by processing temporal features patch-wise. The increase in F increases the training time slightly but has a negligible impact on GPU memory usage.

4.8.2 Number of Variables. Here, we set H to 336. As shown in Figure 8(c) and Figure 8(d), we vary N from 60 to 300. We observe that the time cost of Memformer is correlated quadratically with N, while the space cost is correlated linearly with N. These findings occur because the time complexity of graph convolutional operations are quadratic in N, while the storage space of local information maintained by Memformer is linear in N. Changes to F do not affect the time and space costs of Memformer markedly.

4.8.3 Complexity of Memformer. The most time and space-intensive components of Memformer are the Memformer Encoder and the

AME. The time complexity of the Memformer Encoder is $O(\sqrt{H}N^2 + HN)$, and its space complexity is $O(HN)$. For the AME, the time and space complexities are $O(\sqrt{H}N)$. Therefore, the overall time complexity of Memformer is $O(\sqrt{H}N^2 + HN)$, and the space complexity is $O(HN)$.

4.8.4 Scalability. As shown in Figure 8(e) and Figure 8(f), we compare the impact of varying H on the training time and GPU memory usage of Memformer and the baselines. We fix F at 96. Memformer exhibits better training time and memory usage than all baselines that capture correlations, such as iTransformer, CARD, Crossformer, and MEGNN. The linear models NLinear and DLinear perform the best, followed by the models based on channel-independent mechanisms, ModernTCN and PatchTST. Next, we vary N and report the results in Figure 8(g) and Figure 8(h). Here, we fix H and F at 336 and 96, respectively. Memformer still exhibits better training time and memory usage than baselines that capture correlations. In summary, Memformer achieves state-of-the-art performance among the methods that capture correlations.

5 RELATED WORK

Long-term Time Series Forecasting. Long-term time series forecasting generally refers to forecasting tasks where the historical or forecasting horizons are at least 96, requiring time and space complexities to be within acceptable ranges. Long-term time series forecasting primarily consists of two branches: transformer-based and linear models.

The Transformer [35] was initially applied in natural language processing (NLP) [16] and computer vision (CV) [12]. LogTrans [19] was the first to apply transformers to time series forecasting, proposing sparse attention to reduce the time complexity of self-attention from $O(H^2)$ to $O(H(\log H)^2)$. Subsequent studies reduce the complexity of attention. Informer [49] and Autoformer [39] reduce the attention time complexity to $O(H \log H)$, while Triformer [9], Pyraformer [22], and FEDformer [50] further lower it to $O(H)$. Following this, transformer-based studies branch in two directions: models based on channel-independence mechanisms and models capturing correlations among multiple channels. PatchTST [27] and Pathformer [6] utilize channel-independence mechanisms, which improve model transferability and robustness. In the other branch, Crossformer [47], CARD [36], and iTransformer [24] utilize attention mechanisms to establish correlations among channels, thus improving prediction accuracy when the historical horizon is short.

The linear models TiDE [11], RLinear [21], NLinear [46], and DLinear [46] reconsider the permutation invariance of self-attention and propose a paradigm for long-term time series forecasting. This paradigm involves processing based on temporal features as the encoder and a purely linear layer decoder. Compared to transformer-based models, linear models have lower time and space complexities. However, their structures are simple and are challenged by large datasets with many samples and variables. Additionally, they struggle to match the performance of transformer-based models at handling outliers.

Memory Networks. Early research [13] on memory networks focuses on combining neural networks with external memory and proposes an end-to-end solution. Subsequently, memory networks make advances in fields such as NLP [18, 34, 44] and CV [3, 14, 30]. In the time series field, memory networks are applied to outlier detection [33] and forecasting [15, 23, 38]. Specifically, DAMA-Net [38] employs memory networks to enhance multi-head attention for predicting non-standard human activity sequences. MAGL [23] and MegaCRN [15] leverage the advantages of graph neural networks and memory networks to improve the prediction accuracy of short-term time series. To the best of our knowledge, the present study is the first to propose the use of memory networks for long-term time series forecasting.

Graph Neural Networks. Graph neural networks are commonly used on traffic data. The early DCRNN method [20] requires priors such as road graphs or Euclidean distances to enable training. Subsequently, Graph WaveNet [43] introduces an adaptive graph construction method that does not rely on priors. MTGNN [42] and GTS [31] propose a parameterized k-degree discrete graph to enhance the performance of graph propagation. AGCRN [1] and CCRNN [45] introduce convolutional filters based on node embeddings and adaptive multi-layer graph convolutions, respectively. Next, STEP [32] utilizes transformer pretraining components to improve the capture of long-term dependencies, while MegaCRN [15] and MTSF-DG [48] construct independent adaptive graphs for each timestamp to adapt to road network relationships that evolve over time. Although graph neural networks achieve good prediction accuracy at short-term traffic flow forecasting, the challenge lies in ensuring that models capture dynamic correlations and maintain low complexity when applied to long-term time series.

6 CONCLUSIONS

Our study highlights the importance of addressing both dynamic and disrupted correlations in long-term time series forecasting. We introduce the Memformer framework, which effectively captures dynamic correlations and also mitigates the impact of disrupted correlation through the integration of patch-wise recurrent graph learning and global attention mechanisms. Further, the framework integrates an Alternating Memory Enhancer that enables effective association between local and global information and enhances prediction accuracy. We report on experiments that offer evidence that Memformer is successful at handling dynamic correlations and disrupted correlations and is capable of state-of-the-art forecasting performance across diverse real-world datasets. This research paves the way for more accurate and reliable long-term time series forecasting in spatiotemporal data analysis. The Memformer has a number of sensitive hyperparameters, which require users to understand the data distribution and make nontrivial hyperparameter setting decisions. In future research, it is of interest to integrate automatic hyperparameter tuning techniques. Further, it is of interest to explore the potential of Memformer in the settings of auto machine learning [40, 41], light-weight models [4, 5, 26] and traffic tasks [8, 10, 17, 28].

ACKNOWLEDGMENTS

This research was supported in part by the Innovation Fund Denmark project DIREC (9142-00001B), Independent Research Fund Denmark (8048-00038B), and the Villum Fonden (40567).

REFERENCES

[1] Lei Bai, Lina Yao, Can Li, Xianzhi Wang, and Can Wang. 2020. Adaptive graph convolutional recurrent network for traffic forecasting. *Advances in Neural Information Processing Systems* 33 (2020), 17804–17815.

[2] Yelysei Bondarenko, Markus Nagel, and Tijmen Blankevoort. 2024. Quantizable transformers: Removing outliers by helping attention heads do nothing. *Advances in Neural Information Processing Systems* 36 (2024).

[3] Qi Cai, Yingwei Pan, Ting Yao, Chenggang Yan, and Tao Mei. 2018. Memory Matching Networks for One-Shot Image Recognition. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*. 4080–4088.

[4] David Campos, Bin Yang, Tung Kieu, Miao Zhang, Chenjuan Guo, and Christian S. Jensen. 2024. QCore: Data-Efficient, On-Device Continual Calibration for Quantized Models. *Proc. VLDB Endow.* 17, 11 (2024), 2708–2721.

[5] David Campos, Miao Zhang, Bin Yang, Tung Kieu, Chenjuan Guo, and Christian S. Jensen. 2023. LightTS: Lightweight Time Series Classification with Adaptive Ensemble Distillation. *Proc. ACM Manag. Data* 1, 2 (2023), 171:1–171:27.

[6] Peng Chen, Yingying Zhang, Yunyao Cheng, Yang Shu, Yihang Wang, Qingsong Wen, Bin Yang, and Chenjuan Guo. 2024. Pathformer: Multi-scale transformers with Adaptive Pathways for Time Series Forecasting. In *The Twelfth International Conference on Learning Representations.*

[7] Yunyao Cheng, Peng Chen, Chenjuan Guo, Kai Zhao, Qingsong Wen, Bin Yang, and Christian S. Jensen. 2023. Weakly Guided Adaptation for Robust Time Series Forecasting. *Proc. VLDB Endow.* 17, 4 (2023), 766–779.

[8] Yunyao Cheng, Bin Wu, Li Song, and Chuan Shi. 2019. Spatial-Temporal Recurrent Neural Network for Anomalous Trajectories Detection. In *Advanced Data Mining and Applications - 15th International Conference, ADMA 2019, Dalian, China, November 21-23, 2019, Proceedings*, Vol. 11888. 565–578.

[9] Razvan-Gabriel Cirstea, Chenjuan Guo, Bin Yang, Tung Kieu, Xuanyi Dong, and Shirui Pan. 2022. Triformer: Triangular, Variable-Specific Attentions for Long Sequence Multivariate Time Series Forecasting. *International Joint Conference on Artificial Intelligence*, 1994–2001.

[10] Razvan-Gabriel Cirstea, Bin Yang, and Chenjuan Guo. 2019. Graph Attention Recurrent Neural Networks for Correlated Time Series Forecasting.. In *MileTS19@KDD*.

[11] Abhimanyu Das, Weihao Kong, Andrew Leach, Shaan K Mathur, Rajat Sen, and Rose Yu. 2023. Long-term Forecasting with TiDE: Time-series Dense Encoder. *Transactions on Machine Learning Research* (2023).

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *International Conference on Learning Representations.*

[13] Alex Graves, Greg Wayne, and Ivo Danihelka. 2014. Neural turing machines. *arXiv preprint arXiv:1410.5401* (2014).

[14] Tengda Han, Weidi Xie, and Andrew Zisserman. 2020. Memory-augmented dense predictive coding for video representation learning. *European Conference on Computer Vision*, 312–329.

[15] Renhe Jiang, Zhaonan Wang, Jiawei Yong, Puneet Jeph, Quanjun Chen, Yasumasa Kobayashi, Xuan Song, Shintaro Fukushima, and Toyotaro Suzumura. 2023. Spatio-temporal meta-graph learning for traffic forecasting. *AAAI Conference on Artificial Intelligence* 37, 7, 8078–8086.

[16] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *NAACL-HLT*, 4171–4186.

[17] Duc Kieu, Tung Kieu, Peng Han, Bin Yang, Christian S. Jensen, and Bac Le. 2024. TEAM: Topological Evolution-aware Framework for Traffic Forecasting. *Proc. VLDB Endow.* 18 (2024).

[18] Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Romain Paulus, and Richard Socher. 2016. Ask me anything: Dynamic memory networks for natural language processing. *International Conference on Machine Learning*, 1378–1387.

[19] Shiyang Li, Xiaoyong Jin, Yao Xuan, Xiyou Zhou, Wenhu Chen, Yu-Xiang Wang, and Xifeng Yan. 2019. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. *Advances in Neural Information Processing Systems* 32 (2019).

[20] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. 2018. Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting. *International Conference on Learning Representations.*

[21] Zhe Li, Shiyi Qi, Yiduo Li, and Zenglin Xu. 2023. Revisiting long-term time series forecasting: An investigation on linear mapping. *arXiv preprint arXiv:2305.10721* (2023).

[22] Shizhan Liu, Hang Yu, Cong Liao, Jianguo Li, Weiyao Lin, Alex X Liu, and Schahram Dustdar. 2021. Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. *International Conference on Learning Representations.*

[23] Xiangyue Liu, Xinqi Lyu, Xiangchi Zhang, Jianliang Gao, and Jiamin Chen. 2022. Memory augmented graph learning networks for multivariate time series forecasting. *ACM International Conference on Information & Knowledge Management*, 4254–4258.

[24] Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. 2024. itransformer: Inverted transformers are effective for time series forecasting. *International Conference on Learning Representations* (2024).

[25] Donghao Luo and Xue Wang. 2024. ModernTCN: A modern pure convolution structure for general time series analysis. *International Conference on Learning Representations.*

[26] Hao Miao, Ziqiao Liu, Yan Zhao, Chenjuan Guo, Bin Yang, Kai Zheng, and Christian S. Jensen. 2024. Less is More: Efficient Time Series Dataset Condensation via Two-fold Modal Matching. *Proc. VLDB Endow.* 18 (2024).

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

[28] Simon Aagaard Pedersen, Bin Yang, and Christian S. Jensen. 2020. Anytime Stochastic Routing with Hybrid Learning. *Proc. VLDB Endow.* 13, 9 (2020), 1555–1567.

[29] Xiangfei Qiu, Jilin Hu, Lekui Zhou, Xingjian Wu, Junyang Du, Buang Zhang, Chenjuan Guo, Aoying Zhou, Christian S. Jensen, Zhenli Sheng, and Bin Yang. 2024. TFB: Towards Comprehensive and Fair Benchmarking of Time Series Forecasting Methods. *Proc. VLDB Endow.* 17, 9 (2024), 2363–2377.

[30] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. 2016. Meta-learning with memory-augmented neural networks. *International Conference on Machine Learning*, 1842–1850.

[31] Chao Shang and Jie Chen. 2021. Discrete Graph Structure Learning for Forecasting Multiple Time Series. *International Conference on Learning Representations.*

[32] Zezhi Shao, Zhao Zhang, Fei Wang, and Yongjun Xu. 2022. Pre-training enhanced spatial-temporal graph neural network for multivariate time series forecasting. *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 1567–1577.

[33] Junho Song, Keonwoo Kim, Jeonglyul Oh, and Sungzoon Cho. 2024. Memto: Memory-guided transformer for multivariate time series anomaly detection. *Advances in Neural Information Processing Systems* 36 (2024).

[34] Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. 2015. End-to-end memory networks. *Advances in Neural Information Processing Systems* 28 (2015).

[35] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in Neural Information Processing Systems* 30 (2017).

[36] Xue Wang, Tian Zhou, Qingsong Wen, Jinyang Gao, Bolin Ding, and Rong Jin. 2024. CARD: Channel Aligned Robust Blend Transformer for Time Series Forecasting. *International Conference on Learning Representations.*

[37] Zhaonan Wang, Renhe Jiang, Hao Xue, Flora D Salim, Xuan Song, and Ryosuke Shibasaki. 2022. Event-aware multimodal mobility nowcasting. *AAAI Conference on Artificial Intelligence* 36, 4, 4228–4236.

[38] Zhen Wang, Yang Zhang, Ai Jiang, Ji Zhang, Zhao Li, Jun Gao, Ke Li, Chenhao Lu, and Zujie Ren. 2021. Improving irregularly sampled time series learning with time-aware dual-attention memory-augmented networks. *ACM International Conference on Information & Knowledge Management*, 3523–3527.

[39] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. 2021. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in Neural Information Processing Systems* 34 (2021), 22419–22430.

[40] Xinle Wu, Xingjian Wu, Bin Yang, Lekui Zhou, Chenjuan Guo, Xiangfei Qiu, Jilin Hu, Zhenli Sheng, and Christian S. Jensen. 2024. AutoCTS++: zero-shot joint neural architecture and hyperparameter search for correlated time series forecasting. *VLDB J.* 33, 5 (2024), 1743–1770.

[41] Xinle Wu, Xingjian Wu, Dalin Zhang, Miao Zhang, Chenjuan Guo, Bin Yang, and Christian S. Jensen. 2024. Fully Automated Correlated Time Series Forecasting in Minutes. *Proc. VLDB Endow.* 18 (2024).

[42] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, Xiaojun Chang, and Chengqi Zhang. 2020. Connecting the dots: Multivariate time series forecasting with graph neural networks. *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 753–763.

[43] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. 2019. Graph wavenet for deep spatial-temporal graph modeling. *International Joint Conference on Artificial Intelligence*, 1907–1913.

[44] Caiming Xiong, Stephen Merity, and Richard Socher. 2016. Dynamic memory networks for visual and textual question answering. *International Conference on Machine Learning*, 2397–2406.

[45] Junchen Ye, Leilei Sun, Bowen Du, Yanjie Fu, and Hui Xiong. 2021. Coupled layer-wise graph convolution for transportation demand prediction. *AAAI Conference on Artificial Intelligence* 35, 5, 4617–4625.

[46] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. 2023. Are transformers effective for time series forecasting? *AAAI Conference on Artificial Intelligence* 37, 9, 11121–11128.

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

[48] Kai Zhao, Chenjuan Guo, Yunyao Cheng, Peng Han, Miao Zhang, and Bin Yang. 2023. Multiple time series forecasting with dynamic graph modeling. *VLDB Endowment* 17, 4 (2023), 753–765.

[49] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. 2021. Informer: Beyond efficient transformer for long sequence time-series forecasting. *AAAI Conference on Artificial Intelligence* 35, 12, 11106–11115.

[50] Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. 2022. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. *International Conference on Machine Learning*, 27268–27286.