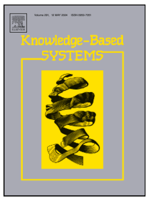
Knowledge-Based Systems 329 (2025) 114389

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys



Toward Transformer-compatible multivariate time series learning via visibility graph-based structural encoding

Ting Chen[a,b], Xinyue Ren[a], Jinzhou Lai[c], Hongming Tan[a,b], Fangming Liu[b,d], Wai Kin Victor Chan[a,*]

[a] *Tsinghua Shenzhen International Graduate School, Tsinghua University, Shenzhen, 518055, China*
[b] *Pengcheng Laboratory, Shenzhen, 518055, China*
[c] *Hong Kong Polytechnic University, Hong Kong, China*
[d] *School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, 430074, China*

A R T I C L E I N F O

Keywords:
Multivariate time series
Visibility graph
Temporal structure
Transformer
Representation learning
Task-general model

A B S T R A C T

Multivariate time series (MTS) modeling plays a crucial role in understanding complex systems. However, existing Transformer-based approaches often struggle to capture essential temporal structures, leading to information loss and even attention dispersion. To address these challenges, we propose **MVGFormer**, a novel Transformer-compatible Multivariate Time Series framework guided by Visibility Graph principles. By explicitly establishing connections between time points based on visibility criteria, we introduce a graph-based sparse Attention (VG-Attention) mechanism, which selectively focuses on crucial temporal dependencies while filtering out irrelevant noise. This sparse Attention significantly mitigates the impact of quadratic complexity, improving scalability for larger time series data. Moreover, considering existing models often overlook the global dependencies within MTS, we extract consensus information across channels and aggregate the multiplex visibility graph into a consensus graph, revealing potential cross-layer patterns. Compared to single-channel models, MSE decreases by 2.82 %, classification accuracy increases by 9.73 %, and training speed improves by 67.48 %. Experimental results across 25 real-world datasets demonstrate that MVGFormer outperforms most existing models in four main tasks, including forecasting, classification, imputation, and anomaly detection. Overall, our approach provides a fresh perspective on adapting Transformers to better understanding temporal dependencies within time series data.

1. Introduction

In complex systems, multivariate time series (MTS) offer a richer and more comprehensive perspective compared to univariate analysis, enabling a better understanding of system behaviors. MTS analysis has been widely applied and extensively studied in numerous practical scenarios, such as physiological signal classification for medical diagnosis, meteorological factor prediction for weather forecasting, and anomaly detection in industrial maintenance monitoring data.

Recently, Transformer-based methods have demonstrated outstanding performance in time series analysis. These methods primarily explore different tokenization paradigms, such as treating each time point as a token [1–5] or grouping multiple consecutive time points into a single token [6–8]. This success can be attributed to the fact that both language dependencies and temporal dependencies fundamentally explore “relationships” between data points (words or time points). However, there is a key distinction between the two: language dependencies primarily focus on the order and grammatical structure between words in a sentence, such as subject-verb agreement or syntactical rules. Whereas, temporal dependencies in time series are quite different, emphasizing sequential patterns such as periodicity, trends, and seasonality over time. For example, a time series might show a regular pattern of peaks and valleys (periodicity), or data points might gradually increase or decrease over time (trend).

The two types of dependencies, though involve connections between data points, differ significantly in how they are structured and how they evolve. As a result, methods like full-Attention or its variants, which rely on pairwise associations, struggle to directly capture meaningful temporal structures from scattered time points [2,9]. They may even introduce noise and potential attention dispersion when applied to time series data. This noise can hinder the accurate understanding of temporal structures and lead to incomplete or inaccurate analysis of temporal

* Corresponding author.
E-mail address: chanw@sz.tsinghua.edu.cn (W.K.V. Chan).

https://doi.org/10.1016/j.knosys.2025.114389
Received 21 May 2025; Received in revised form 21 August 2025; Accepted 30 August 2025
Available online 2 September 2025

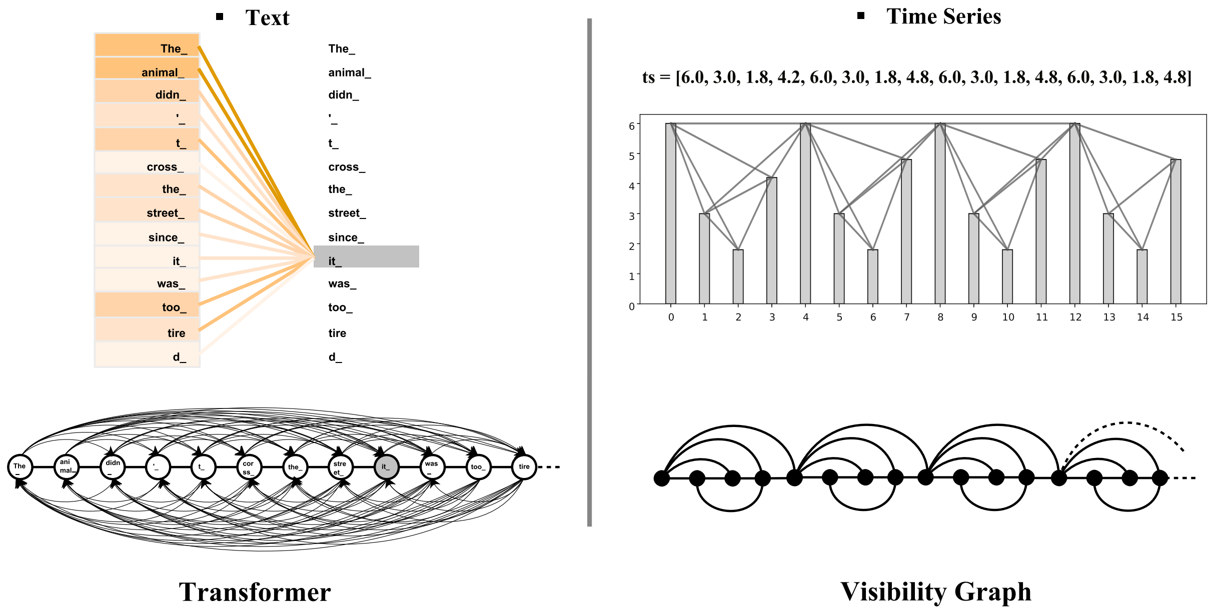

Text
Time Series
ts = [6.0, 3.0, 1.8, 4.2, 6.0, 3.0, 1.8, 4.8, 6.0, 3.0, 1.8, 4.8, 6.0, 3.0, 1.8, 4.8]
The_ animal_ didn_ '_ t_ cross_ the_ street_ since_ it_ was_ too_ tire d_
Transformer
Visibility Graph

Fig. 1. A novel perspective for analyzing time series through the visibility graph criteria.

patterns, especially when temporal dependencies are deeply obscured in intricate patterns. Ultimately, it may result in meaningless attention maps and information loss.

To effectively recognize temporal patterns, this paper adopts a methodology inspired by the Visibility Graph criteria [10], which explicitly constructs temporal connections between time points based on sequential features, as depicted in Fig. 1. This conversion inherits several intrinsic structural properties of the time series [11], and provides a global view to enhance the memory capabilities of the series. In contrast to the current GNN-based methods (GNN4TS) [12], which generate graphs either by heuristics or learning from the series, the visibility graph offers a fresh perspective in constructing time point connections. It possesses universal, theoretical, interpretable, simple, and effective attributes. Therefore, our goal is to address the limited temporal understanding of traditional Transformer, leveraging the strengths of visibility graph to enhance the Transformer's ability to represent time series.

Specifically, our approach consists of two main stages: constructing the visibility graph of temporal relationships and encoding it with the Visibility Graph Transformer. In the first stage, to address the quadratic time complexity $\mathcal{O}(N^2)$ of the visibility graph, we propose a sliding window approach that avoids redundant calculations of time point relationships, achieving linear time complexity. Building upon this, we construct a consensus visibility graph for multivariate time series, which captures the common patterns or interactions across channels and reveals the global dependencies of MTS. In the second stage, we refine a new Attention similarity calculation by adopting the visibility graph-structured approach (VG-Attention), moving away from the fully connected learning typically used in standard Transformer mechanisms. This refinement ensures that our model focuses on the most significant relationships among scattered time points, reducing learning costs and mitigating attention dispersion. Experimental results show that MVGFormer outperforms existing methods, achieving state-of-the-art performance across four major time series analysis tasks. Our contributions can be summarized as follows:

- We introduce MVGFormer, a visibility graph-guided Transformer framework for better capturing both structural properties and temporal variations of MTS.
- We propose a sliding window-based visibility graph (SVG), which reduces the computational complexity from $\mathcal{O}(N^2)$ to $\mathcal{O}(N)$, making it more suitable for large-scale time series data.
- To capture global dependencies in MTS, we propose a consensus visibility graph that integrates both temporal and channel-wise dependencies based on graph-theoretic consensus relationships.
- Based on the consensus graph, we propose a Visibility Graph-based Attention (VG-Attention) mechanism, which focuses on learning crucial structural relationships within temporal patterns, effectively addressing attention dispersion and information loss.
- As a universal model, MVGFormer consistently achieves state-of-the-art performance in four mainstream time series analysis tasks, surpassing most of current Transformer-based methods.

2. Related works

Multivariate time series analysis. In recent years, various deep learning models have been proposed for temporal modeling, such as MLP-based, CNN-based, Graph-based, and Transformer-based models. A number of these methods have been designed for specific downstream tasks. For instance, in forecasting tasks, models like Rlinear [13] and DLinear [9] utilize a single layer of fully connected neural networks to model the relationships between past and future data points for multi-step prediction. In classification tasks, methods like InceptionTime [14], Rocket [15], EEG-Inception [16], and TC-BPPV [17] treat multivariate time series as matrix and leverage Convolutional Neural Network or Attention architectures to generate a rich set of features for time series classification. From a graph perspective, approaches such as STGNN [18], Graph WaveNet [19], MTGNN [20], Copula-based Hybrid-GTS [21], and AutoGRN [22] integrate temporal dynamics with physical or relational structures, aiming to capture complex dependencies among variables to enhance predictive performance. Additionally, some models are designed specifically for anomaly detection, such as the Anomaly Transformer [23], which focuses on capturing contextual deviations in time series data.

To overcome the limitations of task-specific models, TimesNet [11] introduces a task-general backbone called TimesBlock, which adaptively extracts multi-periodicity from time series. TimesNet has demonstrated outstanding performance on five main time series analysis tasks, including short- and long-term forecasting, imputation, classification, and anomaly detection. Therefore, we hold the view that a strong temporal representation ability enables the model to identify anomalies and capture intricate temporal patterns, making it adaptable to various downstream tasks.

Visibility graph. The Visibility Graph [10] was first introduced in PNAS in 2008 and has since become a powerful tool for time series analysis, particularly in fields such as physiology [24,25], economics [26,27], and climate studies [28]. The core idea behind the visibility graph is to transform time series data into a graph, where each data point is represented as a node, and edges are formed based on geometric visibility criteria-two points are connected if they can "see" each other without any other points blocking the line of sight. This technique has proven effective in capturing the inherent structures of time series data and has been applied in various domains to analyze underlying dynamics. Building on this principle, several extensions of the visibility graph have been developed, including the Horizontal Visibility Graph (HVG) [29] and the Limited Penetrable Visibility Graph (LPVG) [30,31]. Furthermore, the visibility graph criterion has been successfully applied in image processing [32], where it has been used to extract two-dimensional spatial features, achieving promising results when compared to traditional models like ResNet [33].

Recent efforts have focused on combining visibility graphs with advanced techniques to address their limitations and enhance their applicability. For example, AVGNet [34] integrates visibility graphs with Graph Neural Networks (GNNs), creating a more adaptive graph structure for signal classification tasks. Additionally, MAGNN [35] combines multi-scale graph learning techniques with visibility graphs for multivariate time series forecasting, effectively preserving temporal dependencies at different scales. These methods aim to improve the flexibility and scalability of visibility graphs while maintaining their interpretability and simplicity. Moreover, recent reviews [36,37] have provided comprehensive analyses of visibility graph criteria and their applications, highlighting that network-based analytical techniques are valuable tools for extracting meaningful insights from time series data across diverse domains. However, visibility graphs still face challenges, particularly in terms of high computational complexity, which limits their applicability to large-scale time series data. Meanwhile, the introduction of Transformer-based models presents a promising approach to enhance the visibility graph's ability in understanding and modeling complex temporal relationships,allowing for a more effective capture of long-term dependencies within time series data.

Transformer-based time series representation. Transformers have demonstrated excellent performance in time series forecasting [1,2, 6–8,38,39]. Through the Attention mechanism, these methods effectively uncover the temporal dependencies between time points. Notably, Autoformer [2] uses an Auto-Correlation mechanism to capture series-wise temporal dependencies based on learned periods. To address complex temporal patterns, Autoformer employs a deep decomposition architecture to extract seasonal and trend components from the input series. Later, FEDformer [5] incorporated a mixture-of-expert design to improve seasonal-trend decomposition and introduced sparse Attention in the frequency domain. Furthermore, to address inter-variable correlations, iTransformer [40], TimeXer [7] and MultiPatchFormer [8] have innovatively use an inverse Transformer and cross Attention to capture multivariate dependencies, yielding excellent results. Previous Transformer-based methods have explored various effective techniques for time series. In this work, we aim to enhance the Transformer's understanding of temporal dependencies by incorporating visibility graph criteria. This approach provides a fresh perspective, enabling the model to capture the complex network connections inherent in time series data and deepen its understanding of temporal relationships.

3. MVGFormer

Our proposed MVGFormer, illustrated in Fig. 2, comprises three stages: (1) Multiplex visibility graph: the projection of multivariate time series to a multi-layer network; (2) Channel-wise consensus information extraction, and (3) the Visibility graph transformer model for encoding time series representations.Temporal Embedding

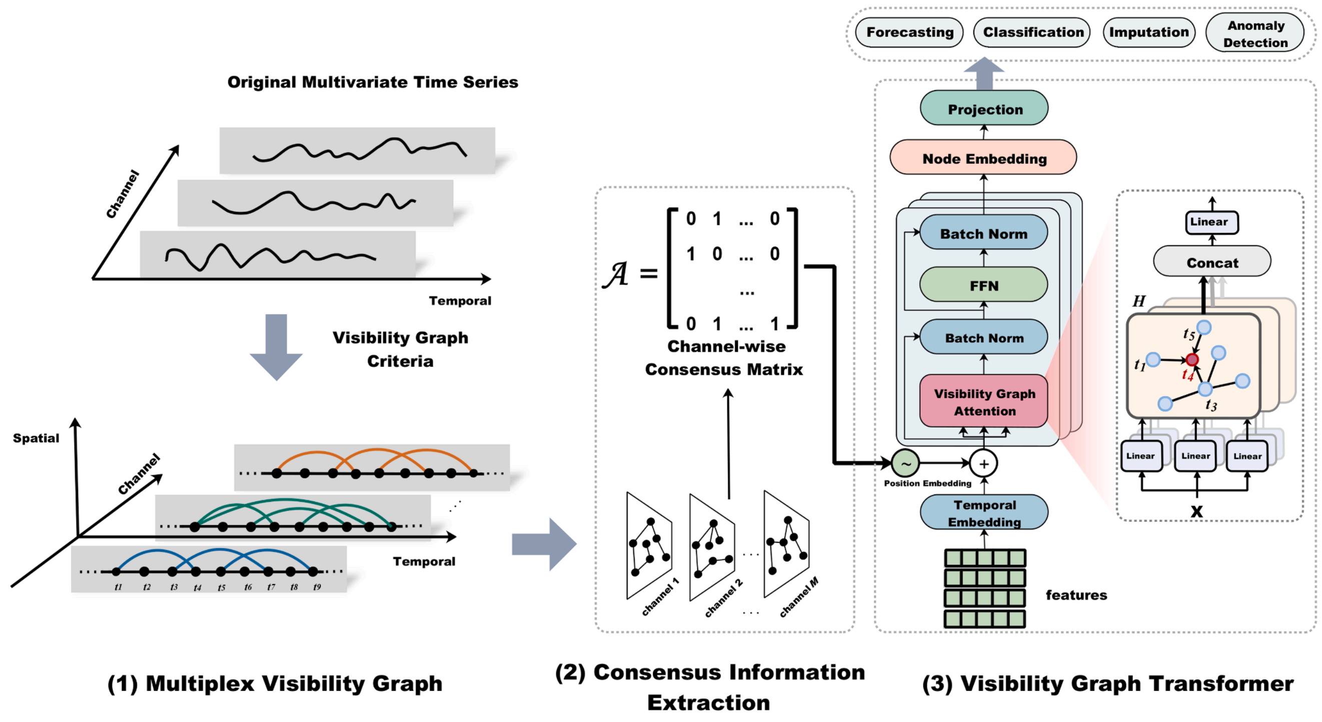

Original Multivariate Time Series
Channel
Temporal
Visibility Graph Criteria
Spatial
Channel
Temporal
t1 t2 t3 t4 t5 t6 t7 t8 t9
$\mathcal{A} = \begin{bmatrix} 0 & 1 & \dots & 0 \\ 1 & 0 & \dots & 0 \\ & & \dots & \\ 0 & 1 & \dots & 1 \end{bmatrix}$
Channel-wise Consensus Matrix
channel 1
channel 2
channel M
Forecasting
Classification
Imputation
Anomaly Detection
Projection
Node Embedding
Batch Norm
FFN
Batch Norm
Visibility Graph Attention
Position Embedding
Temporal Embedding
features
Linear
Concat
H
t1 t3 t4 t5
Linear
Linear
Linear
X
(1) Multiplex Visibility Graph
(2) Consensus Information Extraction
(3) Visibility Graph Transformer

Fig. 2. The framework of MVGFormer. **(1) Projection stage:** Converts multivarite time series into a multiplex visibility graph using visibility graph criteria. **(2) Consensus information extraction:** Aggregate the multiplex graph into a single-layer graph, obtaining a aggregated graph $\mathcal{A}$ with channel-wise consensus relationships as the model input. **(3) Encoding stage:** Leverage the generated temporal topological structure, overlaying multiple Visibility Graph Transformer layers for iterative learning, involving Temporal Embedding, Visibility Graph Attention and Batch Normalization.

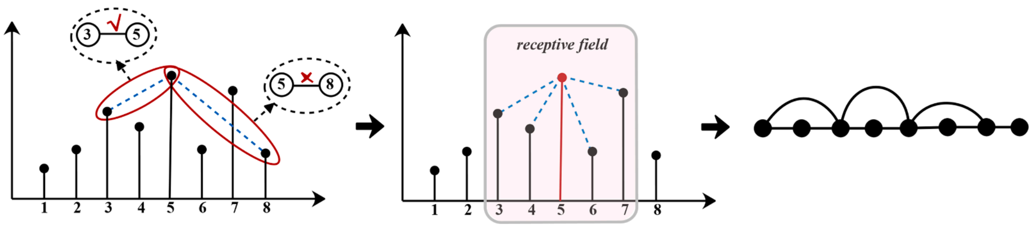


Fig. 3. The process of mapping time series into a graph by visibility graph criteria.

3.1. Preliminary

Visibility graph. For a univariate time series consisting of N real-valued data $\{\mathbf{x}(t)\}_{t=1}^{N}$, following the visibility algorithm, nodes typically represent specific time points, and edges indicate the connection between these time points fulfilling visibility graph criteria:

$$x(t_c) < x(t_b) + \left(x(t_a) - x(t_b)\right)\frac{t_b - t_c}{t_b - t_a}. \quad (1)$$

where $\left(t_a, x(t_a)\right)$ and $\left(t_b, x(t_b)\right)$ are two arbitrary data values, with an additional data point $\left(t_c, x(t_c)\right)$ positioned between them. A straight line, termed the "visibility line", connects points in the series data without intersecting any intermediate data heights, as depicted in Fig. 3.

The advantages of visibility graph. This conversion highlights the inherent ability of the visibility graph to capture both ordered and chaotic structures present in time-series data. The insights are drawn from the research of [10], which identified two core advantages: **(1)** Visibility criteria quantify the "receptive field" of each time point. Similar to the receptive field in Convolutional Neural Networks (CNNs), the "visibility line" offers a global view, allowing the model to assess the influence of each temporal point. As shown in Fig. 4, panel (a) illustrates that the influence range of periodic peaks is confined within each cycle, while in panel (b), the influence of sharp drops extends to all subsequent points. **(2)** The Visibility Graph is applicable to various types of time-series data, including both periodic and non-periodic series [10]. Specifically, ordered series are represented by regular graphs (visualized in different colors for clarity), while random series correspond to exponential random graphs.

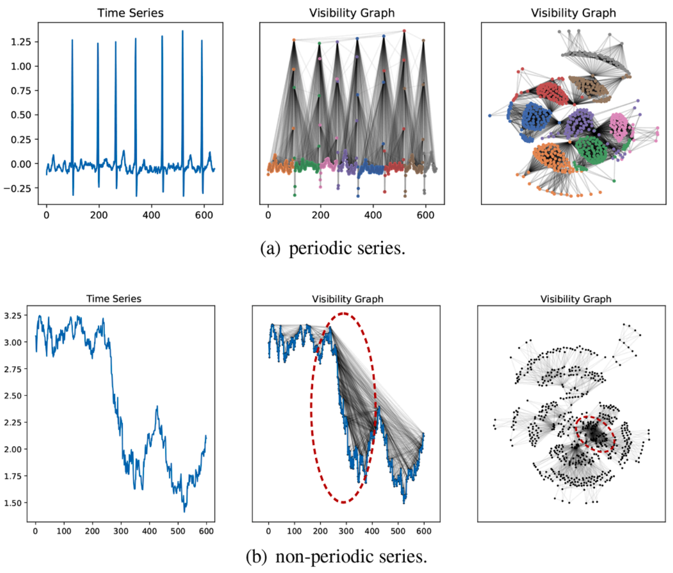


Fig. 4. Visibility graph criteria for periodic and non-periodic time series.

Table 1
Notation table.

Symbol	Definition
N	Length of the time series (number of time points / nodes)
M	Number of channels (variables) in the multivariate time series Number of layers in the multiplex visibility graph
d	Embedding dimension (hidden size of node representations)
$x^{[\alpha]}(t)$	Value of the α-th channel at time point t
t_i, i	The i-th time point (node i)
$A^{[\alpha]}$	Adjacency matrix of the visibility graph for the α-th channel
$\mathcal{M}$	Multiplex visibility graph consisting of all channel-wise graphs
$\mathcal{A}$	Consensus visibility graph aggregated from $\mathcal{M}$
$\mathcal{N}(i)$	Neighbor set of node i in the visibility graph

3.2. Multiplex visibility graph

In this section, we apply the visibility graph to multivariate time series. Firstly, to address the issue of high time complexity, we introduce an improved approach, the Sliding Window Visibility Graph (SVG), which achieves linear complexity. Then, we construct a visibility graph for each variable, leading to the multiplex visibility graph for multivariate time series, as illustrated in Fig. 2(b).

3.2.1. Sliding window visibility graph

Traditional methods of constructing visibility graph for time series data require full traversal of the dataset to compute visibility relationships between all pairs of points, resulting in a time complexity of $\mathcal{O}(N^2)$, where N is the sequence length. This approach becomes computationally expensive and inefficient for large-scale datasets.

To address these limitations, we introduce the Sliding Window Visibility Graph (SVG), which processes time series in a fixed-size and overlapping window, shown in Fig. 5. As the window slides across the time series, only the newly added time points and the removed time points are updated, significantly reducing the amount of recalculation. This results in an overall linear time complexity of $\mathcal{O}(N)$, offering a substantial improvement over traditional methods. By combining the sliding window technique with traditional visibility graph algorithms, SVG ensures efficient, real-time updates and linear time complexity, making it highly suitable for large-scale time series analysis and real-time data streams. Additionally, for longer time series or lightweight scenarios, we can further explore the divide-and-conquer strategy proposed in [41], combined with the sliding window strategy, which can reduce the time complexity to $\mathcal{O}(\log N)$.

3.2.2. Multiplex visibility graph structure

Based on the proposed SVG, we extend the visibility approach to introduce a novel structure, termed the multiplex visibility graph [42], denoted as $\mathcal{M}$. This graph constructed from an M-channel time series $\{\mathbf{x}(t)\}_{t=1}^{N}$. At any given time point t, each $\mathbf{x}(t)$ is a vector $\left(x^{[1]}(t), x^{[2]}(t), \ldots, x^{[M]}(t)\right) \in \mathbb{R}^M$, sourced from M distinct sensors. The

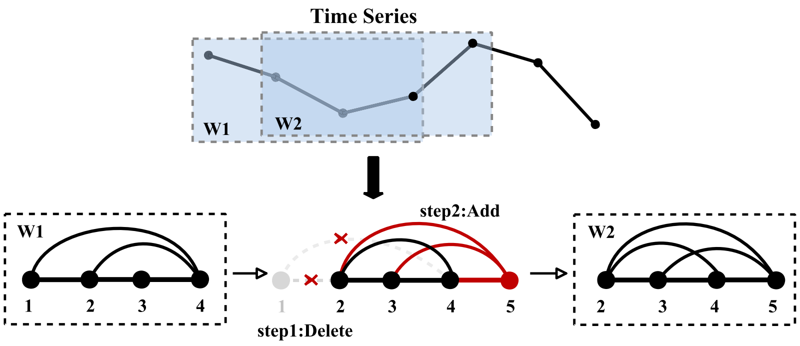


Fig. 5. Sliding window visibility graph : Time complexity from $O(N^2)$ to $O(N)$.

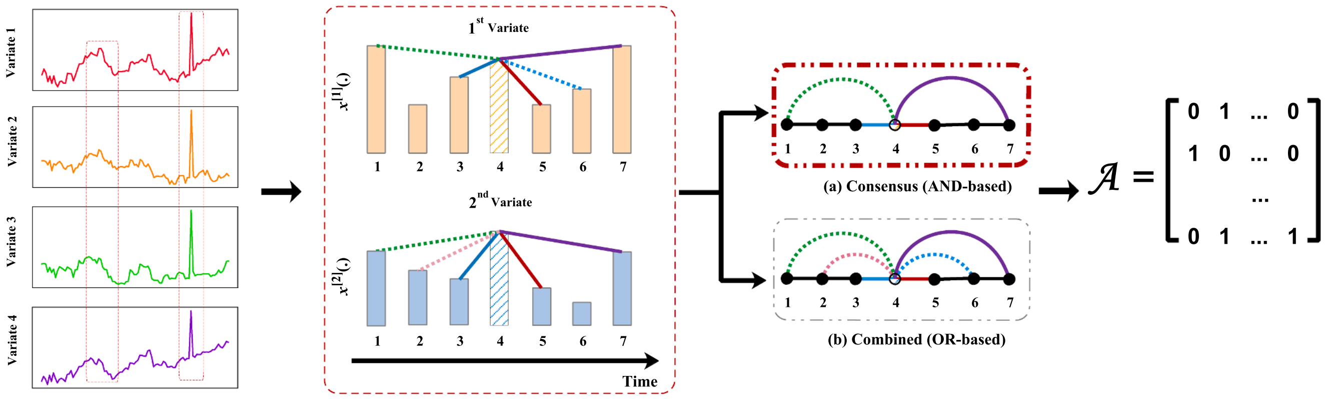

Variate 1
Variate 2
Variate 3
Variate 4
1st Variate
2nd Variate
$x^{[1]}(\cdot)$
$x^{[2]}(\cdot)$
Time
(a) Consensus (AND-based)
(b) Combined (OR-based)
$\mathcal{A} = \begin{bmatrix} 0 & 1 & \dots & 0 \\ 1 & 0 & \dots & 0 \\ & & \dots & \\ 0 & 1 & \dots & 1 \end{bmatrix}$

Fig. 6. Layer aggregation mechanisms: (a) Consensus Visibility Graph, which requires an edge in every layer to synchronize channel interactions across different time points, following the AND-based logic; and (b) Combined Visibility Graph, which preserves all edge relationships to allow diverse channel interactions across layers, following the OR-based logic.

structure of $\mathcal{M}$ is multi-layered, with each layer α containing a visibility graph that corresponds to the time series of a specific variable $\{x^{[\alpha]}(t)\}_{t=1}^{N}$. Specifically, the multiplex visibility graph $\mathcal{M}$ comprises a composite of adjacency matrices, collectively represented as

$$\mathcal{M} = \{A^{[1]}, A^{[2]}, \dots, A^{[M]}\}, \tag{2}$$

where each $A^{[\alpha]}$ is the $N \times N$ adjacency matrix corresponding to layer α. In this multiplex graph, each matrix $A^{[\alpha]} = \{e_{ij}^{[\alpha]}\}$ denotes the connectivity: $e_{ij}^{[\alpha]} = 1$ signifies a link between nodes (time points) t_i and t_j in the α-th layer, while $e_{ij}^{[\alpha]} = 0$ indicates no link, applicable for all node pairs $t_i, t_j = 1, 2, \dots, N$. To better describe the structure of the multiplex visibility graph and simplify the notation, we uniformly use $e_{ij}^{[\alpha]}$ and node i, j in subsequent sections, where node i, j corresponds to the time points t_i, t_j in the original time series. Furthermore, to ensure consistency and clarity of the symbols, the unified symbol definitions are provided in Table 1.

3.3. Consensus information extraction

The multiplex visibility graph, with its layered structure and shared nodes, encapsulates the underlying correlations across different channels in multivariate time series. To better understand these relationships from a global perspective, we explore the common patterns or interactions between layers using consensus relationships [43–47] in the context of multiplex graphs. One intuitive idea is to construct an aggregated graph (we call the consensus visibility graph) by merging adjacency matrices to capture the consensus information across all views, revealing the global dependencies of the multivariate time series.

For the aggregation algorithm, we draw inspiration from the classical concept of subgraph isomorphism in complex network analysis [48,49], which helps identify topological structures consistently present across all layers. These recurring subgraphs reveal structural consensus [50], capturing coordinated behaviors across channels and facilitating global temporal understanding. Based on this theoretical foundation, we define the Consensus Visibility Graph in Definition 1 as a unified representation that integrates multi-layer connectivity patterns derived from the multiplex visibility graph.

Definition 1 (Consensus visibility graph)**.** To capture the joint visibility structure across all layers in the multiplex graph $\mathcal{M}$, we define the Consensus Visibility Graph as a single-layer fused graph $\mathcal{A}$, where each node corresponds to a time point, and the edge set $e_{ij} \in \mathcal{A}$ is constructed based on inter-layer agreement. As shown in Fig. 6(a). an edge e_{ij} exists between nodes i and j if and only if the visibility condition is satisfied across all layers $\alpha = 1, \dots, M$, according to the following rule when for each intermediate time instance t_k, with $t_i < t_k < t_j$:

$$\mathcal{A} = \{e_{ij} \mid \forall\alpha, x^{[\alpha]}(t_k) < \min\left(x^{[\alpha]}(t_i), x^{[\alpha]}(t_j)\right),\ \alpha = 1, \dots, M\} \tag{3}$$

where $\mathcal{A}$ is the set of consensus edges, and $e_{ij} = 1$ signifies a consensus link between time point i and j across all channels. In fact, we can approach the problem from a matrix perspective, drawing an analogy to the logical AND operation on adjacency matrix $A^{[\alpha]}$ to implement the consensus structure search. This allows us to simplify the multiplex graph $\mathcal{M}$ into a single-layer topology $\mathcal{A}$ using an AND-based aggregation mechanism (MVG-AND).

$$\mathcal{A} = \varphi_{\mathrm{AND}}(\mathcal{M}) = \prod_{\alpha=1}^{M} A^{[\alpha]} \tag{4}$$

For each pair of nodes i, j,

$$\mathcal{A}_{ij} = \varphi_{\mathrm{AND}}\left(\mathcal{M}_{ij}\right) = \prod_{\alpha=1}^{M} A_{ij}^{[\alpha]} = \begin{cases} 1, & \text{if } e_{ij} = 1 \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

To validate the effectiveness of this method, we compare it with the Combined Visibility Graph shown in Fig. 6(b), which emphasizes the diversity of connections by retaining edges present in any layer, according to the following rule when for each intermediate time instance t_k, with $t_i < t_k < t_j$:

$$\mathcal{A} = \left\{ e_{ij} \mid \exists\alpha,\ x^{[\alpha]}(t_k) < \min\left(x^{[\alpha]}(t_i), x^{[\alpha]}(t_j)\right),\ \alpha = 1, \dots, M \right\} \tag{6}$$

Similarly, the Combined Visibility Graph graph can be constructed using an OR-based aggregation mechanism (MVG-OR), as

$$\begin{aligned} \mathcal{A} = \varphi_{\mathrm{OR}}(\mathcal{M}) &= \bigvee_{\alpha=1}^{M} A^{[\alpha]} \\ &= \mathbf{1}_{N\times N} - \prod_{\alpha=1}^{M}\left(\mathbf{1}_{N\times N} - A^{[\alpha]}\right), \end{aligned} \tag{7}$$

where $\bigvee$ represents the logical OR operation applied over the layers. $\mathbf{1}_{N\times N} \in \mathbb{R}^{N\times N}$ is the matrix with all elements equal to one, and

$$\mathcal{A}_{ij} = \varphi_{OR}\left(\mathcal{M}_{ij}\right) = 1 - \prod_{\alpha=1}^{M}\left(1 - A_{ij}^{[\alpha]}\right). \tag{8}$$

By comparing these two approaches, we assess the impact of focusing on commonality (AND-based) versus diversity (OR-based) in capturing key relationships in multivariate time series. In fact, the experiments show that MVG-AND outperforms MVG-OR, as illustrated in Fig. 9.

Based on this, we construct an consensus visibility graph $\mathcal{A}$ using Multiplex Visibility Graph (Section 3.2) and Consensus Relationship Extraction (Section 3.3), which integrates temporal dependencies and channel-wise consensus information. The overall process is shown in Algorithm 1.

Algorithm 1: Consensus visibility graph $\mathcal{A}$.

```
Input: $\mathbf{X} \in \mathbb{R}^{M\times N}$ ($M$ variables, $N$ time points), sliding window
       size $W_{\mathrm{SVG}}$, Aggregation_type (“AND” or “OR”)
Output: consensus visibility graph $\mathcal{A}$
1  Step 1: Multiplex visibility graph construction;
2  $w \leftarrow \mathbf{X}[1 : W_{\mathrm{SVG}}]$ // Initialize window
3  $\mathcal{G} \leftarrow$ BuildGraph($w$) // Initial visibility graph
4  $\mathcal{M} \leftarrow [\mathcal{G}]$ // List of layer graphs
5  for $idx = W_{SVG}$ to $N$ do
6      $ts \leftarrow \mathbf{X}[idx - W_{\mathrm{SVG}} + 1 : idx]$ // SVG
7      $new\_node \leftarrow \mathbf{X}[idx]$;
8      $old\_node \leftarrow \mathbf{X}[idx - W_{\mathrm{SVG}}]$;
9      $\mathcal{G} \leftarrow$ RemoveEdges($\mathcal{G}, old\_node$);
10     if CheckVisibility($new\_node, \mathcal{G}$) then
11         $\mathcal{G} \leftarrow$ AddEdge($\mathcal{G}, new\_node$);
12     $\mathcal{M}$.append($\mathcal{G}$);
13 Step 2: Consensus matrix extraction;
14 $\mathcal{A} \leftarrow \emptyset$ // Initialize aggregated graph
15 for $\alpha = 1$ to $M$ do
16     if Aggregation_type = “OR” then
17         $\mathcal{A} \leftarrow \mathcal{A} \times (1 - \mathcal{M}[\alpha])$;
18     else if Aggregation_type = “AND” then
19         $\mathcal{A} \leftarrow \mathcal{A} \times \mathcal{M}[\alpha]$;
20 if Aggregation_type = “OR” then
21     return $1 - \mathcal{A}$;
22 else if Aggregation_type = “AND” then
23     return $\mathcal{A}$;
```

3.4. Visibility graph transformer

The consensus visibility graph $\mathcal{A}$, with both sequential and graph-based features, promises focused attention and significant computational efficiency compared to traditional fully connected sequence methods, which have quadratic complexity. This section explains the learning mechanism of the proposed Visibility Graph Transformer architecture: Temporal Embedding, Visibility Graph Attention Mechanism, Batch Normalization and Feed Forward Network.

Temporal embedding. Initially, we prepare the input node embeddings to be passed to the Visibility Graph Transformer layer. For the consensus graph $\mathcal{A}$, each node i features node embeddings $\beta_i \in \mathbb{R}^{M\times 1}$, where M represents the number of variables. To address inconsistencies in units across variables and reduce distributional discrepancies among individual input time series, the input node β_i undergoes Z-score normalization along the temporal dimension, and then passes via a linear projection to be embedded into d-dimensional hidden features $\hat{h}_i^{\,0}$, as:

$$\hat{h}_i^0 = W_i^0 \bar{\beta}_i + p^0 \tag{9}$$

where

$$\bar{\beta}_i = \frac{\beta_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}}, \tag{10}$$

and $\mu_{\mathcal{B}}^{(j)} \leftarrow \frac{1}{N}\sum_{i=1}^{N} \beta_i^{(j)}; \sigma_{\mathcal{B}}^{2\,(j)} \leftarrow \frac{1}{N}\sum_{i=1}^{N}\left(\beta_i^{(j)} - \mu_{\mathcal{B}}^{(j)}\right)^2$, $j = 1, 2, \dots, M$ is the dimension index of multivariate time series, N is the input series length, $\mu_{\mathcal{B}}, \sigma_{\mathcal{B}} \in \mathbb{R}^{M\times 1}$, correction factor ϵ is set to $1e-5$, and $W_i^0 \in \mathbb{R}^{d\times M}$, $p^0 \in \mathbb{R}^d$ are the parameters of the linear projection layers. To capture temporal dependencies in time series, we embed pre-computed timestamp position encodings using the sinusoidal embedding method often used in NLP.

$$T_i^{(j)} = \begin{cases} \sin\left(i/10000^{2k/d}\right), & \text{if } j = 2k \\ \cos\left(i/10000^{2k/d}\right), & \text{if } j = 2k+1 \end{cases} \tag{11}$$

where i denotes the sequence position of the time points, j denotes the dimension index, and d denotes the dimension of the temporal encoding, $k = 0, 1, 2, \dots, d/2 - 1$. And then add to the node features $\hat{h}_i^0$ to obtain the initial node embedding

$$h_i^0 = \hat{h}_i^0 + T_i. \tag{12}$$

Visibility graph attention mechanism. The standard (Dense) Transformers Attention mechanism, as illustrated in Fig. 7, proves effective for language sequences due to the inherent challenge of pre-determining the inter-token relationships. In this context, it involves matrix multiplication ($\otimes$) for pairwise node similarity calculation, as:

$$\text{Full-Attention}(Q, K, V) = \operatorname{softmax}\left(\frac{Q \otimes K^T}{\sqrt{d}}\right)V, \tag{13}$$

where Q, K, and V are linear projections of input embedding. We aim for Attention to focus on the intrinsic characteristics of time series (such as periodicity, trends, fractality, etc.). Therefore, compared to the fully connected Full-Attention mechanism, our approach incorporates reliable connections between time points, thus adding additional graph structure information into the input, as follows:

$$\text{VG-Attention}(Q, K, V, \mathcal{A}) = \sum_{(i,j)\in\mathcal{A}} \alpha_{ij} V_j, \tag{14}$$

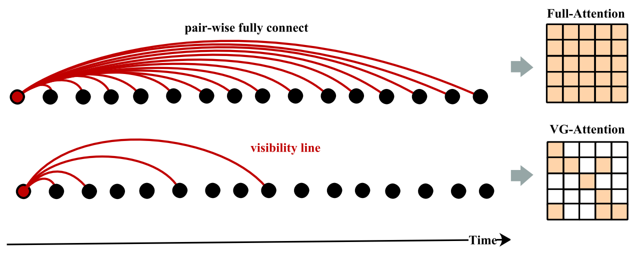


Fig. 7. Visibility graph attention mechanism.

Table 2
Summary of experiment benchmarks.

Tasks	Benchmarks	Metrics	Window Size (W_{SVG})
Long-term Forecasting	ETTh1, ETTh2, Weather, Exchange Rate, ILI	MSE, MAE	96 ~ 720 (ILI: 24 ~ 60)
Short-term Forecasting	M4(6 subsets)	SMAPE, MASE, OWA	6 ~ 48
Classification	UEA (17 subsets)	Accuracy	29 ~ 1751
Anomaly Detection	SMD, MSL, SMAP, SWaT, PSM	Precision, Recall, F1-Socre	100
Imputation	ETTh1, ETTh2, ETTm1, ETTm2, Weather	MSE, MAE	96

where

$$\alpha_{ij} = \frac{\exp\left(\frac{Q_i K_j^T}{\sqrt{d}}\right)}{\sum_{k\in\mathcal{N}(i)} \exp\left(\frac{Q_i K_k^T}{\sqrt{d}}\right)}$$

and $(i,\ j)$ denotes an edge between nodes i and j in consensus graph $\mathcal{A}$, $\mathcal{N}(i)$ denotes the neighbor set of node i, and Q_i, K_j represents the feature vector of node i and j, respectively. This approach will effectively reduce the computational load and addresses the issue of attention dispersion, which often arises from superfluous relationships in the sequence. Specifically, the update of node h_i^ℓ is formulated as:

$$\hat{h}_i^{\ell+1} = O_h^\ell \|_{k=1}^H \left(\sum_{j\in\mathcal{N}(i)} \alpha_{ij}^{k,\ell} V^{k,\ell} h_j^\ell \right), \tag{15}$$

where $Q^{k,\ell}, K^{k,\ell}, V^{k,\ell} \in \mathbb{R}^{d_k\times d}, O_h^\ell \in \mathbb{R}^{d\times d}$, $k = 1$ to H denotes the number of heads, and $\|$ denotes concatenation.

Batch normalization & feed forward network. The attention outputs $\hat{h}_i^{\ell+1}$ are subsequently fed into a Feed Forward Network (FFN), which is surrounded by residual connections and normalization layers. Differing from the Layer Normalization typically used around the Transformer feed forward layers, we apply Batch Normalization for normalization since features from different channels being non-comparable; for example, variables such as temperature, humidity, and atmospheric pressure in a multivariate weather dataset are not directly comparable. Batch Normalization allows us to focus more effectively on the distribution of different samples within the same channel, a decision supported by evidence from the ablation study presented in Table 11. The FFN then performs two linear transformations, incorporating the non-linear activation function ReLU to extract deeper-level features, as:

$$\hat{h}_i^{\ell+1} = \mathrm{BatchNorm}\left(h_i^\ell + \hat{h}_i^{\ell+1}\right), \tag{16}$$

$$\hat{\hat{h}}_i^{\ell+1} = FFN(\hat{h}_i^{\ell+1}) = W_2^\ell \mathrm{ReLU}\left(W_1^\ell \hat{h}_i^{\ell+1}\right), \tag{17}$$

$$h_i^{\ell+1} = \mathrm{BatchNorm}\left(\hat{\hat{h}}_i^{\ell+1} + \hat{h}_i^{\ell+1}\right), \tag{18}$$

where $W_1^\ell \in \mathbb{R}^{2\ d\times d}, W_2^\ell \in \mathbb{R}^{d\times 2d}$ denote weight matrices, $\hat{h}_i^{\ell+1}, \hat{\hat{h}}_i^{\ell+1}$ denote intermediate representations. Finally, for the final layer $\ell+1$ of the Graph Transformer model, De-normalization is performed to revert the outputs to their original statistical properties, as:

$$h_i = (W_i^{\ell+1} h_i^{\ell+1} + \mu_{\mathcal{B}})\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}, \tag{19}$$

where $W_i^{\ell+1} \in \mathbb{R}^{M\times d}$ is employed to project the learned node vector representation back to the original feature dimension M for utilization in downstream tasks.

Time complexity analysis. The time complexity of the Full-Attention and VG-Attention mechanisms differs significantly. Full-Attention has a time complexity of $\mathcal{O}(N^2\ d)$, where N is the number of nodes (or time points) and d is the dimension of the node embedding. This complexity arises from the matrix multiplication $Q \otimes K^T$, the softmax operation, and the final matrix multiplication with V, all of which scale quadratically with N. In contrast, VG-Attention has a time complexity of $\mathcal{O}(Ed)$, where E is the number of edges in the graph. This is linear with respect to the number of edges, and the computation can be more efficient when the graph is sparse (i.e., $E \ll N^2$). As a result, VG-Attention tends to be more efficient, particularly in sparse graphs, compared to the fully connected Full-Attention mechanism.

Table 3
Experiment configuration of MVGFormer.

Tasks / Configurations	Model Hyper-parameter			Training Process		
	Layers	dmin[a]	dmax[a]	LR[b]	Batch Size	Epochs
Long-term Forecasting	3	32	512	10^{-4}	32	10–30
Short-term Forecasting	3	16	64	10^{-3}	16	30
Imputation	3	64	128	10^{-3}	16	10–30
Classification	3	32	64	10^{-3}	16	30
Anomaly Detection	3	32	128	10^{-4}	128	10

[a] $d_{model} = \min\left\{\max\left\{2^{\lceil \log_2 C\rceil}, d_{\min}\right\}, d_{\max}\right\}$, where C is input series dimension.
[b] LR means the initial learning rate.

Space Complexity Analysis. The introduction of the visibility graph requires additional memory to store the adjacency matrix. Specifically, for a multivariate time series input of shape $M \times N$, where M is the number of channels and N is the number of time points, constructing visibility graphs independently for each channel requires storing M adjacency matrices of size $N \times N$, resulting in a total space complexity of $O(M \times N^2)$. However, by introducing the Consensus Visibility Graph mechanism, these multiple graphs are aggregated into a single fused adjacency matrix, requiring only $O(N^2)$ space. This significantly reduces memory usage and enhances scalability, particularly as the number of channels increases.

4. Experiments

To verify the effectiveness of MVGFormer, we conducted extensive comparison experiments following the settings of TimesNet [11]. The benchmark summary is presented in Table 2, and the experiment configurations are detailed in Table 3.The code is available at https://anonymous.4open.science/r/MVGFormer.

Dataset details. To evaluate the performance of the proposed MVGFormer, we conduct experiments on 7 real-world datasets on forecasting task, including ETT (ETTh1, ETTh2, ETTm1, ETTm2), Exchange Rate, Weather and ILI, as shown in Table 2. And 17 UEA datasets[1] on classification task, including the gesture, action and audio recognition, medical diagnosis by heartbeat monitoring and other practical domain datasets. Moreover, we extend our approach on univariate time series like M4 dataset[2] to demonstrate the efficiency on all types of time series.

Baselines. We compare MVGFormer against well-established and advanced models across all five tasks, including CNN-based Model: TimesNet [11], TCN [51]; MLP-based models: LightTS [52], DLinear [9]

[1] https://www.timeseriesclassification.com/
[2] https://github.com/M4Competition/M4-methods/tree/master/Dataset

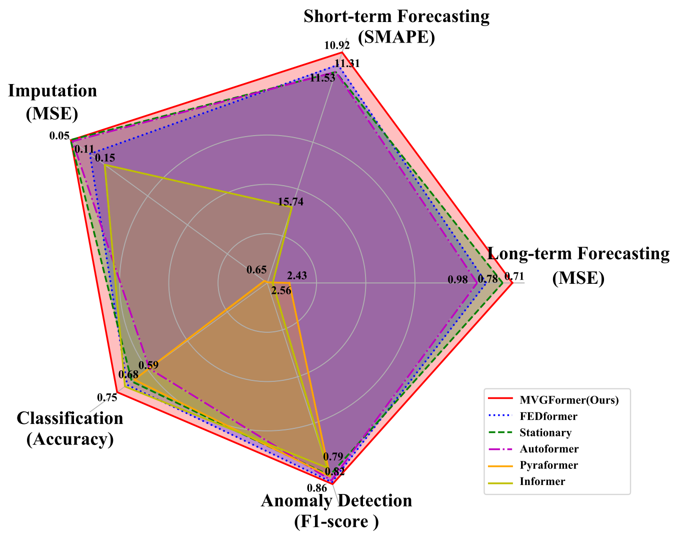


Fig. 8. Model performance comparison with Transformer-based Models.

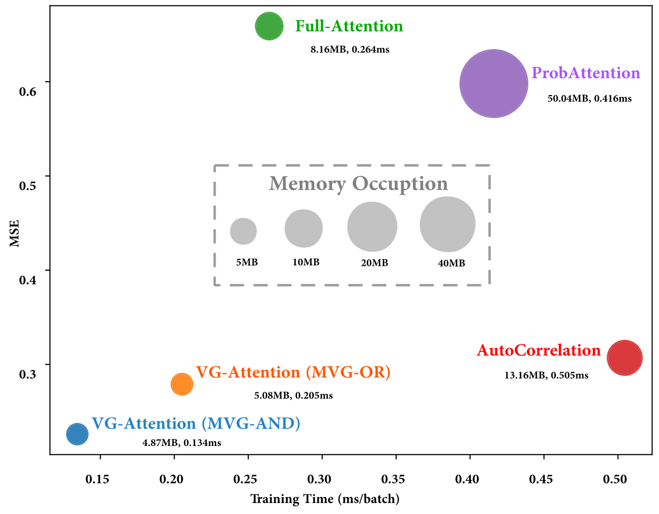


Fig. 9. Comparison of the time and memory consumption between the Full-Attention, the ProbAttention, AutoCorrelation and VG-Attention.

and WPMixer [53]; RNN-based models: LSSL [54]; GNN-based models: STGNN [18], Graph WaveNet [19] and MTGNN [20]; Transformer-based models: Informer [1], Pyraformer [38], Autoformer [2], PatchTST [6], iTransformer [39] and TimeXer [7].

Among them, TimesNet is a task-general model known for handling multiple time series tasks, providing an effective backbone for learning periodicities and trends. It currently shows state-of-the-art (SOTA) performance across a variety of tasks. iTransformer and TimeXer are currently a SOTA model in long-term forecasting tasks, demonstrating impressive results. Additionally, Autoformer, Informer, STGNN and Graph WaveNet are classic Transformer-based or GNN-based variants that have served as important benchmarks in the field for time series forecasting.

Besides, we also compare the state-of-the-art models for each specific task, such as Anomaly Transformer [23] for anomaly detection, Rocket [15] and TodyNet [55] for classification. Overall, more than 25 baselines are included for a comprehensive comparison. All the models are trained/tested on a single Nvidia V100-32G GPU.

4.1. Main results

As a task-general model, MVGFormer achieves consistent state-of-the-art performance on five mainstream analysis tasks. To demonstrate its strengths, we primarily compared it with existing point-wise Transformer-based models, such as FEDformer [5], Autoformer [2], Informer [1], and others, as shown in Fig. 8.

Additionally, the sparse nature of the VG-Attention matrix means fewer parameters need to be stored and computed, improving memory efficiency. Therefore, as shown in Fig. 9, VG-Attention shows significant advantages in both runtime and memory consumption compared to the corresponding Attention variants (such as Full-Attention, ProbAttention [1], and AutoCorrelation [2]). Moreover, we can observe that the MVG-AND layer aggregation strategy outperforms the MVG-OR strategy in both MSE and training time, as it better captures the consensus relationship between channels. Therefore, the AND-based aggregation strategy is used in subsequent experiments.

4.2. Long-term forecasting

Setup. To evaluate the model performance in forecasting task, we conduct experiments on five real-world benchmarks datasets, including Electricity (ETTh1, ETTh2) [1], Weather[3], Exchange [56] and Illness (ILI)[4]. Following the setting of TimesNet, the input sequence length is set as 36 for ILI and 96 for the others.

Results. MVGFormer demonstrates outstanding performance in more than **32.5 %** (13/40) cases in the forecasting task, as evidenced in Table 4, surpassing both general-purpose TimesNet and the advanced Transformer-based models. This superior performance may be attributed to the global perspective provided by the visibility graph criteria, which incorporate both historical and future information. Such criteria enhances the model ability to extract periodic and trend patterns, facilitating more accurate predictions for distant future time points. Moreover, it is evident that no single model excels across all datasets. For example, TimeXer performs particularly well on the ETTh2 dataset, while DLinear shows strength primarily on the Exchange dataset. In contrast, MVGFormer exhibits uniform performance across all datasets, indicating its potential applicability to a broader range of scenarios and superior generalizability.

4.3. Short-term forecasting

Setup. In this work, the proposed method MVGFormer is primarily designed for multivariate time series data. Actually, it can also be applied to univariate time series data without the procedure of layer aggregation. Therefore, we conducted comparative experiments using the univariate M4 dataset, which contains yearly, quarterly, and monthly collected marketing data. For the short-term forecasting metric, we adopt the symmetric mean absolute percentage error (SMAPE), mean absolute scaled error (MASE) and overall weighted average (OWA) as the metrics, where OWA is a special metric used in M4 competition.

Results. The M4 dataset consists of 100,000 time series collected from various sources at different frequencies, leading to diverse temporal variations and making forecasting more challenging. As shown in Table 5, MVGFormer consistently outperforms advanced Transformer-based and MLP-based models across all sampling frequencies. This is because the visibility graph, grounded in temporal features, is independent of the sampling frequency. High-frequency data results in denser subgraphs, while low-frequency data shows sparser connections.

4.4. Anomaly detection

Setup. Anomaly detection in industrial monitoring data is crucial for maintenance, but labeling challenges arise due to anomalies being

[3] https://www.bgc-jena.mpg.de/wetter/

[4] https://gis.cdc.gov/grasp/fluview/fluportaldashboard.html

Table 4
Long-term forecasting task. Results are presented for four different prediction lengths: $\{24, 36, 48, 60\}$ for ILI and $\{96, 192, 336, 720\}$ for other datasets. A lower MSE and MAE indicate superior performance, with the best results highlighted in **bold**. The asterisk (*) denotes "former" models.

Models		**MVGFormer (Ours)**		WPMixer (2025)		MultiPatch* (2025)		TimeXer (2024)		PatchTST (2024)		iTrans* (2024)		TimesNet (2023)		LightTS (2022)		DLinear (2023)		FED* (2022)		Stationary (2022)		Auto* (2021)		Pyra* (2021)		In* (2021)		LSSL (2022)	
datasets	pred len	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.392	0.412	0.380	**0.392**	0.377	0.397	0.388	0.420	0.379	0.399	0.395	0.409	0.384	0.402	0.424	0.432	0.386	0.400	**0.376**	0.419	0.513	0.491	0.449	0.459	0.664	0.612	0.865	0.713	0.548	0.528
	192	0.451	0.445	0.437	0.430	0.427	**0.428**	0.438	0.448	0.427	0.442	0.449	0.441	0.436	0.429	0.475	0.462	0.437	0.432	**0.420**	0.448	0.534	0.504	0.552	0.482	0.790	0.681	1.008	0.792	0.542	0.526
	336	0.491	**0.454**	0.478	0.456	0.462	0.457	0.473	0.460	0.472	0.459	0.492	0.465	0.491	0.469	0.518	0.488	0.481	0.459	**0.459**	0.465	0.588	0.535	0.521	0.496	0.891	0.738	1.107	0.809	1.298	0.942
	720	0.497	**0.483**	0.508	0.486	0.507	0.488	**0.484**	0.490	0.525	0.508	0.524	0.506	0.521	0.500	0.547	0.533	0.519	0.516	0.506	0.507	0.643	0.616	0.514	0.512	0.963	0.782	1.181	0.865	0.721	0.659
ETTh2	96	0.323	**0.369**	0.296	0.379	0.292	0.386	**0.286**	0.388	0.309	0.376	0.302	0.381	0.340	0.372	0.397	0.437	0.333	0.387	0.358	0.397	0.476	0.458	0.346	0.388	0.645	0.597	3.755	1.525	1.616	1.036
	192	0.409	**0.412**	0.368	0.429	0.370	0.434	**0.288**	0.439	0.376	0.418	0.381	0.419	0.402	0.414	0.520	0.504	0.477	0.476	0.429	0.439	0.512	0.493	0.456	0.452	0.788	0.683	5.602	1.931	2.083	1.197
	336	0.479	0.463	0.423	0.438	0.418	0.430	**0.364**	**0.420**	0.419	0.434	0.424	0.434	0.452	0.452	0.626	0.559	0.594	0.541	0.496	0.487	0.552	0.551	0.482	0.486	0.907	0.747	4.721	1.835	2.970	1.439
	720	0.472	0.473	0.467	0.466	**0.423**	0.442	0.432	**0.432**	0.440	0.467	0.428	0.445	0.462	0.468	0.863	0.672	0.831	0.657	0.463	0.474	0.562	0.560	0.515	0.511	0.963	0.783	3.647	1.625	2.576	1.363
Weather	96	0.178	0.229	0.166	0.212	0.171	0.211	**0.162**	**0.210**	0.177	0.219	0.179	0.218	0.172	0.228	0.182	0.242	0.196	0.255	0.217	0.296	0.173	0.223	0.266	0.336	0.622	0.556	0.300	0.384	0.174	0.252
	192	0.226	0.267	**0.208**	**0.249**	0.218	0.254	0.207	0.261	0.224	0.260	0.228	0.260	0.219	0.261	0.227	0.287	0.237	0.296	0.276	0.336	0.245	0.285	0.307	0.367	0.739	0.624	0.598	0.544	0.238	0.313
	336	0.285	0.305	**0.263**	**0.289**	0.276	0.297	0.264	0.292	0.281	0.300	0.282	0.300	0.280	0.306	0.282	0.334	0.283	0.335	0.339	0.380	0.321	0.338	0.359	0.395	1.004	0.753	0.578	0.523	0.287	0.355
	720	**0.343**	**0.348**	0.345	0.352	0.355	0.352	0.347	0.351	0.358	0.351	0.360	0.352	0.365	0.359	0.352	0.386	0.345	0.381	0.403	0.428	0.414	0.410	0.419	0.428	1.420	0.934	1.059	0.741	0.384	0.415
Exchange	96	**0.081**	**0.216**	0.088	0.220	0.092	0.219	0.096	0.221	0.088	0.307	0.097	0.221	0.107	0.234	0.116	0.262	0.088	0.218	0.148	0.278	0.111	0.237	0.197	0.323	1.748	1.105	0.847	0.752	0.395	0.474
	192	0.207	**0.307**	0.177	0.317	0.179	0.313	0.198	0.319	0.189	0.309	0.181	0.310	0.226	0.344	0.215	0.359	**0.176**	0.315	0.271	0.380	0.219	0.335	0.300	0.369	1.874	1.151	1.204	0.895	0.776	0.698
	336	0.449	0.491	0.343	0.422	0.351	0.431	0.335	**0.419**	0.341	0.422	0.339	0.422	0.367	0.448	0.377	0.466	**0.313**	0.427	0.460	0.500	0.421	0.476	0.509	0.524	1.943	1.172	1.672	1.036	1.029	0.797
	720	1.065	0.783	0.890	0.706	0.887	0.713	0.936	0.739	0.895	0.710	0.944	0.723	0.964	0.746	0.831	0.699	**0.839**	**0.695**	1.195	0.841	1.092	0.769	1.447	0.941	2.085	1.206	2.478	1.310	2.283	1.222
ILI	24	**1.654**	**0.839**	2.299	0.955	2.360	1.179	2.408	1.071	2.214	1.072	2.446	1.001	2.317	0.934	8.313	2.144	2.398	1.040	3.228	1.260	2.294	0.945	3.483	1.287	7.394	2.012	5.764	1.677	4.381	1.425
	36	2.216	1.036	2.425	0.980	2.345	1.072	2.291	1.052	2.327	0.910	2.176	1.054	1.972	0.928	6.631	1.902	2.646	1.088	2.679	1.080	**1.825**	**0.848**	3.103	1.148	7.551	2.031	4.755	1.467	4.442	1.416
	48	**1.995**	0.932	2.506	1.020	2.160	0.989	2.253	0.928	2.040	**0.875**	2.226	0.963	2.238	0.945	7.299	1.982	2.614	1.086	2.622	1.078	2.010	0.900	2.669	1.085	7.662	2.057	4.763	1.469	4.559	1.443
	60	**1.964**	0.939	2.536	1.025	2.142	1.083	2.285	0.939	1.986	1.023	2.329	1.098	2.027	**0.928**	7.283	1.985	2.804	1.146	2.857	1.157	2.178	0.963	2.770	1.125	7.931	2.100	5.264	1.564	4.651	1.474
AVG		**0.709**	**0.508**	0.780	0.511	0.741	0.529	0.747	0.515	0.723	0.513	0.759	0.521	0.737	0.511	1.824	0.757	0.850	0.558	0.910	0.583	0.784	0.547	0.981	0.606	2.429	1.066	2.558	1.123	1.798	0.899
1st Count		**13**		5		2		9		1		0		1		0		4		3		2		0		0		0		0	

For fairness, all experimental results use the same input length and batch size;
"AVG" indicates the average performance across all datasets and prediction lengths;
"1st Count" represents the number of best performances achieved.

Table 5
Short-term forecasting task on M4. The prediction lengths are in [6, 48] and results are weighted averaged from several datasets under different sample intervals. The asterisk (*) denotes “former” models.

Model		**MVGFormer (Ours)**	iTransformer (2024)	ETSformer (2022)	LightTS (2022)	DLinear (2023)	FED* (2022)	Stationary (2022a)	Auto* (2021)	Pyra* (2021a)	In* (2021)	LogTrans* (2019)	Re* (2020)	LSTM (1997)	TCN (2019)	LSSL (2022)
Yearly	SMAPE	**13.582**	14.009	18.009	14.247	16.965	13.728	13.717	13.974	15.53	14.727	17.107	16.169	176	14	61.675
	MASE	**3.015**	3.182	4.487	3.109	4.283	3.048	3.078	3.134	3.711	3.418	4.177	3.8	31.033	3.364	19.953
	OWA	**0.795**	0.829	1.115	0.827	1.058	0.803	0.807	0.822	0.942	0.881	1.049	0.973	9.29	0.88	4.397
Quarterly	SMAPE	**10.512**	10.791	13.376	11.364	12.145	10.792	10.958	11.338	15.449	11.360	13.207	13.313	172.808	11.122	65.999
	MASE	**1.233**	1.287	1.906	1.328	1.52	1.283	1.325	1.365	2.35	1.401	1.827	1.775	19.753	1.36	17.662
	OWA	**0.927**	0.959	1.302	1	1.106	0.958	0.981	1.012	1.558	1.027	1.266	1.252	15.049	1.001	9.436
Monthly	SMAPE	**13.246**	13.695	14.588	14.014	13.514	14.26	13.917	13.958	17.642	14.062	16.149	20.128	143.237	15.626	64.664
	MASE	**0.994**	1.069	1.368	1.053	1.037	1.102	1.097	1.103	1.913	1.141	1.66	2.614	16.551	1.274	16.245
	OWA	**0.926**	0.978	1.149	0.981	0.956	1.012	0.998	1.002	1.511	1.024	1.34	1.927	12.747	1.141	9.879
Others	SMAPE	**4.999**	5.811	7.267	15.88	6.709	4.954	6.302	5.485	24.786	24.46	23.236	32.491	186.282	7.186	121.844
	MASE	**3.515**	4.063	5.24	11.434	4.953	3.264	4.064	3.865	18.581	20.96	16.288	33.355	119.294	4.677	91.65
	OWA	**1.08**	1.252	1.591	3.474	1.487	1.036	1.304	1.187	5.538	5.879	5.013	8.679	38.411	1.494	27.273
Weighted (Average)	SMAPE	**12.255**	12.676	14.718	13.525	13.639	12.84	12.78	12.909	16.987	14.086	16.018	18.2	160.031	13.961	67.156
	MASE	**1.642**	1.757	2.408	2.111	2.095	1.701	1.756	1.771	3.265	2.718	3.01	4.223	25.788	1.945	21.208
	OWA	**0.881**	0.927	1.172	1.051	1.051	0.918	0.93	0.939	1.48	1.23	1.378	1.775	12.642	1.023	8.021

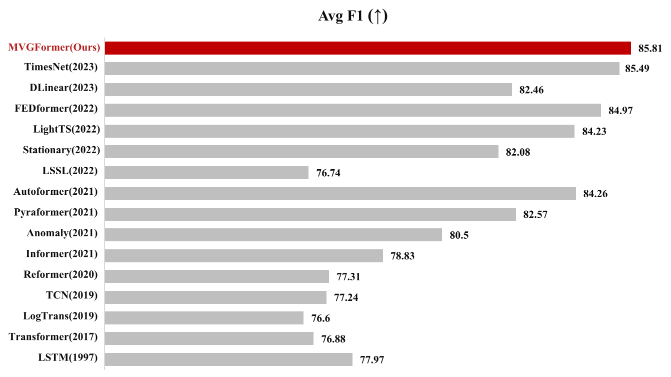


Fig. 10. Anomaly detection task. The F1-score (as %) is computed as evaluation metrics for each dataset, with higher values indicating superior performance.

obscured in large-scale datasets. We compare our model on five real-world datasets, including MSL [57], SMD [58], SWaT [59], SMAP [57] and PSM [60]. Consistent with previous works, such as TimesNet [11] and Anomaly Transformer [23], we adopted the same preprocessing and evaluation methods, including sliding window segmentation, as well as the anomaly criterion and anomaly threshold setting. Specifically, we treat reconstruction as an unsupervised point-wise representation learning task, using reconstruction error as the anomaly detection criterion. To ensure fairness across experiments, we retain the same anomaly criterion (reconstruction error) while varying the base models for reconstruction.
Results. As depicted in Fig. 10, MVGFormer excels in anomaly detection, outperforming TimesNet and advanced Transformer-based models such as FEDformer [5], Pyraformer [38] and Autoformer [2]. By leveraging visibility criteria, MVGFormer effectively identifies anomalies as high-degree nodes within visibility graphs, thereby centralizing attention on them. In contrast, standard Transformer methods often distribute attention across multiple edges due to pairwise correlation calculations, which can lead to weakened or overlooked anomaly detection. Our graph-based sparse attention mechanism ensures more focused and precise anomaly detection, thereby enhancing precision and reducing the likelihood of misinterpretation or oversight.

4.5. *Classification*

Setup. To evaluate the model's capacity in high-level representation learning, 17 multivariate time series datasets from UEA [61] are selected for sequence-level classification.
Results. As shown in Table 6, MVGFormer demonstrates superior performance over existing methods, including the classical method Rocket [15], the deep-learning method InceptionTime [14], and the GNN-based method TodyNet [55]. Specially, it shows remarkable effectiveness in processing physiological signals, as seen with AtrialFibrillation, StandWalkJump, SelfRegulationSCP2.

4.6. *Imputation*

Setup. Real-world system malfunctions often lead to partial data missing from continuously collected time series, posing challenges for downstream analysis. Imputation thus becomes essential in practical scenarios. Following the setting (random mask ratios {12.5 %, 25 %, 37.5 %, 50 %}) used in TimesNet, we conduct experiments on the ETT (4 subsets) [1] and Weather datasets, where missing data is prevalent.

Results. Table 7 shows that MVGFormer, using MSE and MAE metrics, outperforms most Transformer-based methods in **50 %** of the cases. This is attributed to the visibility graph extraction of temporal trends and the reliable sparse VG-Attention mechanism focusing on the crucial temporal structural dependency. These factors empower our algorithm to exhibit robust fitting capabilities, both locally and globally. And this fitting proficiency also proves advantageous in forecasting tasks.

4.7. *Model analysis*

4.7.1. *Efficiency analysis*

To provide a comprehensive comparison, we showcase the results of the forecasting task as depicted in Figs. 11 and 12. We can observe that MVGFormer exhibits stronger local fitting capabilities and trend judgment abilities, thanks to the “receptive field” of the visibility graph. This enables it to better adapt to temporal pattern extraction compared to Transformer-based methods such as Autoformer [2], Stationary [3] and Transformer.

4.7.2. *The benefit of VG-Attention*

From the previous experimental results, we can see that MVGFormer demonstrated excellent performance in both forecasting and classification tasks. To further analyze why VG-Attention works, specifically how the geometric visibility rules of the visibility graph guide Attention learning, we compare it with other Attention mechanisms, such as standard Full-Attention and the probabilistic sparse version of ProbAttention [1]. As shown in Figs. 13–15, we selected three typical time series types: periodic, trending, and random non-stationary, and visualized the learned Attention maps.

A closer look at these visualizations, we can see that, compared to standard Full-Attention and ProbAttention, VG-Attention concentrates its attention on specific key time points or time intervals, such as periodic local modules (Fig. 13(b)), long-term trending structures (Fig. 14(b)), and key fluctuations (Fig. 15(b)). In contrast, **(1)** the attention distribution of **Full-Attention** is relatively uniform, failing to focus on key time points. **(2) ProbAttention**, through its sparsification mechanism, reduces attention to less important time points, which, although decreasing computational load, may miss critical temporal dependencies.Therefore, VG-Attention stands out for the more focused attention distribution, offering a novel paradigm for Transformer models to better capture and understand temporal sequences.

4.7.3. *The longer history length*

Theoretically, due to its global perspective, the visibility graph is expected to be more suitable for long time series compared to other Transformer-based methods. To test this, we compared MVGFormer with FEDformer [5], Autoformer[2], Informer [1], and Transformer models to determine if they can capture more historical information and achieve better forecasting performance as the length of the historical time series increases. As shown in Fig. 16, we found that only MVGFormer and Pyraformer [38] exhibit lower MSE as the input time series lengthens. Interestingly, traditional Transformer models perform worse as the input length increases, likely due to the fact that Full-Attention tends to capture more noise as the number of time series tokens grows, leading to dispersed attention.

4.7.4. *Representation capability*

As shown in Fig. 17, we visualized the embedding representations of multivariate time series. The embeddings learned by MVGFormer align well with the characteristics of the time series. Specifically, the model captures key segments of the series (Fig. 17(a-b)), such as peaks and large fluctuations, with more prominent embeddings. For stable and periodic time series (Fig. 17(c)), the representations are more uniformly distributed. Additionally, we observed that MVGFormer effectively captures other complex temporal features, such as anomalies and multi-periodicity (Fig. 17(d-e)). This further explains its strong performance in anomaly detection tasks. Furthermore, as seen in Fig. 18, MVGFormer demonstrates a strong representation capability in classification tasks, effectively distinguishing different categories of multivariate time series in the vector space.

Table 6
Classification task. We report the classification accuracy (%) as the result, with the best results in **bold**.

Datasets	**MVGFormer (Ours)**	PatchTST (2024)	iTransformer (2024)	TodyNet (2023)	TimesNet (2023)	DLinear (2023)	Rocket (2021)	HIVE-COTE (2020)	TapNet (2020)	InceptionTime (2020)	ResNet (2019)	Autoformer (2021)	Pyraformer (2021)	Informer (2021)
ArticularyWordRecognition	97.92	95.67	46.33	98.70	98.00	97.33	**99.56**	97.99	98.70	98.53	98.26	55.33	89.00	96.33
AtrialFibrillation	**66.67**	46.67	33.33	46.70	40.00	20.00	24.89	29.33	30.22	40.00	35.33	46.66	33.33	40.00
BasicMotions	**100**	70.00	85	100	95.00	87.50	99.00	100	100	100	100	57.50	100	92.50
Epilepsy	98.44	93.48	68.84	97.10	84.05	57.97	99.08	**100**	96.09	95.43	99.18	78.26	65.21	82.60
ERing	82.03	96.67	92.96	91.50	92.96	81.48	**98.05**	94.26	92.34	92.10	87.19	72.22	80.37	93.70
FingerMovements	**64.58**	53.00	56	57.00	54.00	57.00	55.27	53.77	51.33	55.90	51.60	55.00	58.00	64.00
HandMovementDirection	45.31	58.11	52.7	**64.90**	55.40	63.51	44.59	37.79	32.34	40.41	42.30	41.89	43.24	63.51
Handwriting	28.49	25.18	27.18	43.60	32.10	18.58	56.67	50.41	32.95	47.84	**59.78**	15.41	22.35	27.76
Libras	70.62	79.44	90.00	85.00	75.55	52.22	90.61	90.28	83.68	83.61	**94.11**	75.55	63.33	80.00
LSST	59.01	52.64	19.71	61.50	35.52	31.58	**63.15**	53.84	44.33	34.21	42.94	42.86	32.80	38.76
NATOPS	92.50	82.22	87.78	**97.20**	91.66	92.22	88.54	82.85	93.90	94.44	97.11	75.55	81.66	89.44
PenDigits	96.76	96.48	97.06	98.70	98.31	85.13	99.56	97.19	98.00	98.75	**99.64**	98.42	98.42	98.22
RacketSports	90.63	76.97	79.61	80.30	86.18	75.65	**92.79**	90.64	87.40	89.07	91.23	80.26	84.21	85.52
StandWalkJump	**66.67**	40.00	60	46.70	46.66	33.33	45.56	40.67	35.11	30.00	38.67	53.33	47.44	40.00
UWaveGestureLibrary	69.69	84.38	87.5	85.00	85.30	81.25	**94.43**	91.31	88.39	88.28	88.35	49.06	76.56	81.87
SelfRegulationSCP1	**91.32**	82.59	91.15	89.80	90.78	89.07	86.55	86.02	84.21	84.69	76.11	56.65	83.75	89.76
SelfRegulationSCP2	**60.63**	52.78	53.89	55.00	53.88	57.22	51.35	51.67	84.21	52.04	50.24	51.66	56.11	50.55
1st Count	**6**	0	0	2	0	0	5	1	0	0	3	0	0	0

"1st Count" represents the number of best performances achieved.

Table 7
Imputation task. We compare the model performance under different missing degrees $\{12.5\,\%, 25\,\%, 37.5\,\%, 50\,\%\}$. The best results are in **bold** and second best underlined.

Models		**MVGFormer (Ours)**		iTransformer (2024)		ETSformer (2022)		LightTS (2022)		DLinear (2023)		FEDformer (2022)		Stationary (2022)		Autoformer (2021)		Pyraformer (2021)		Informer (2021)		LSSL (2022)		TCN (2019)	
Datasets	mask ratio	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	0.125	0.061	0.176	0.096	0.218	0.126	0.263	0.240	0.345	0.151	0.267	0.070	0.190	**0.060**	**0.165**	0.074	0.182	0.857	0.609	0.114	0.234	0.422	0.461	0.599	0.554
	0.25	**0.071**	0.208	0.122	0.247	0.169	0.304	0.265	0.364	0.180	0.292	0.106	0.236	0.080	**0.189**	0.090	0.203	0.829	0.672	0.140	0.262	0.412	0.456	0.610	0.567
	0.375	0.107	0.221	0.153	0.276	0.220	0.347	0.296	0.382	0.215	0.318	0.124	0.258	**0.102**	**0.212**	0.109	0.222	0.830	0.675	0.174	0.293	0.421	0.461	0.628	0.577
	0.5	**0.128**	**0.239**	0.197	0.314	0.293	0.402	0.334	0.404	0.257	0.347	0.165	0.299	0.133	0.240	0.137	0.248	0.854	0.691	0.215	0.325	0.443	0.473	0.648	0.587
ETTh2	0.125	**0.041**	**0.132**	0.098	0.213	0.187	0.319	0.101	0.231	0.100	0.216	0.095	0.212	0.042	0.133	0.044	0.138	0.976	0.754	0.305	0.431	0.521	0.555	0.410	0.494
	0.25	0.053	0.155	0.126	0.244	0.279	0.39	0.115	0.246	0.127	0.247	0.137	0.258	**0.049**	**0.147**	0.050	0.149	1.037	0.774	0.322	0.444	0.487	0.535	0.419	0.490
	0.375	**0.056**	**0.154**	0.157	0.273	0.4	0.465	0.126	0.257	0.158	0.276	0.187	0.304	0.056	0.158	0.060	0.163	1. 107	0.800	0.353	0.462	0.487	0.529	0.429	0.498
	0.5	0.067	0.175	0.197	0.305	0.602	0.572	0.136	0.268	0.183	0.299	0.232	0.341	**0.065**	**0.170**	0.068	0.173	1.193	0.838	0.369	0.472	0.484	0.523	0.467	0.529
Weather	0.125	0.035	0.055	0.041	0.091	0.057	0.141	0.047	0.101	0.039	0.084	0.041	0.107	0.027	0.051	**0.026**	**0.047**	0.140	0.220	0.037	0.093	0.036	0.095	0.176	0.287
	0.25	0.037	0.060	0.052	0.116	0.065	0.155	0.052	0.111	0.048	0.103	0.064	0.163	**0.029**	0.056	0.030	**0.054**	0.147	0.229	0.042	0.100	0.042	0.104	0.187	0.293
	0.375	0.040	0.065	0.077	0.159	0.081	0.180	0.058	0.121	0.057	0.117	0.107	0.229	0.033	0.062	**0.032**	**0.060**	0.156	0.240	0.049	0.111	0.047	0.112	0.172	0.281
	0.5	0.044	0.070	0.121	0.215	0.102	0.207	0.065	0.133	0.066	0.134	0.183	0.312	0.037	0.068	**0.037**	**0.067**	0.164	0.249	0.053	0.114	0.054	0.123	0.195	0.303
ETTm1	0.125	**0.025**	**0.106**	0.046	0.148	0.067	0.188	0.075	0.180	0.058	0.162	0.035	0.135	0.026	0.107	0.034	0.124	0.670	0.541	0.047	0.155	0.101	0.231	0.510	0.493
	0.25	0.033	**0.118**	0.061	0.173	0.096	0.229	0.093	0.206	0.08	0.193	0.052	0.166	**0.032**	0.119	0.046	0.144	0.689	0.553	0.063	0.180	0.106	0.235	0.518	0.500
	0.375	**0.038**	**0.131**	0.079	0.198	0.133	0.271	0.113	0.231	0.103	0.219	0.069	0.191	0.039	0.131	0.057	0.161	0.737	0.581	0.079	0.200	0.116	0.246	0.516	0.499
	0.5	0.051	0.146	0.108	0.233	0.186	0.323	0.134	0.255	0.132	0.248	0.089	0.218	**0.047**	**0.145**	0.067	0.174	0.770	0.605	0.093	0.218	0.129	0.26	0.519	0.496
ETTm2	0.125	**0.020**	**0.087**	0.052	0.152	0.108	0.239	0.034	0.127	0.062	0.166	0.056	0.159	0.021	0.088	0.023	0.092	0.394	0.470	0.133	0.27	0.15	0.298	0.307	0.441
	0.25	**0.023**	**0.091**	0.071	0.179	0.164	0.294	0.042	0.143	0.085	0.196	0.080	0.195	0.024	0.096	0.026	0.101	0.421	0.482	0.135	0.272	0.159	0.306	0.263	0.402
	0.375	**0.025**	**0.096**	0.091	0.205	0.237	0.356	0.051	0.159	0.106	0.222	0.110	0.231	0.027	0.103	0.03	0.108	0.478	0.521	0.155	0.293	0.18	0.321	0.250	0.396
	0.5	**0.029**	**0.104**	0.118	0.233	0.323	0.421	0.059	0.174	0.131	0.247	0.156	0.276	0.030	0.108	0.035	0.119	0.568	0.560	0.200	0.333	0.210	0.353	0.246	0.389
1st Count		**20**		**0**		0		0		0		0		13		7		0		0		0		0	

"1st Count" represents the number of best performances achieved.

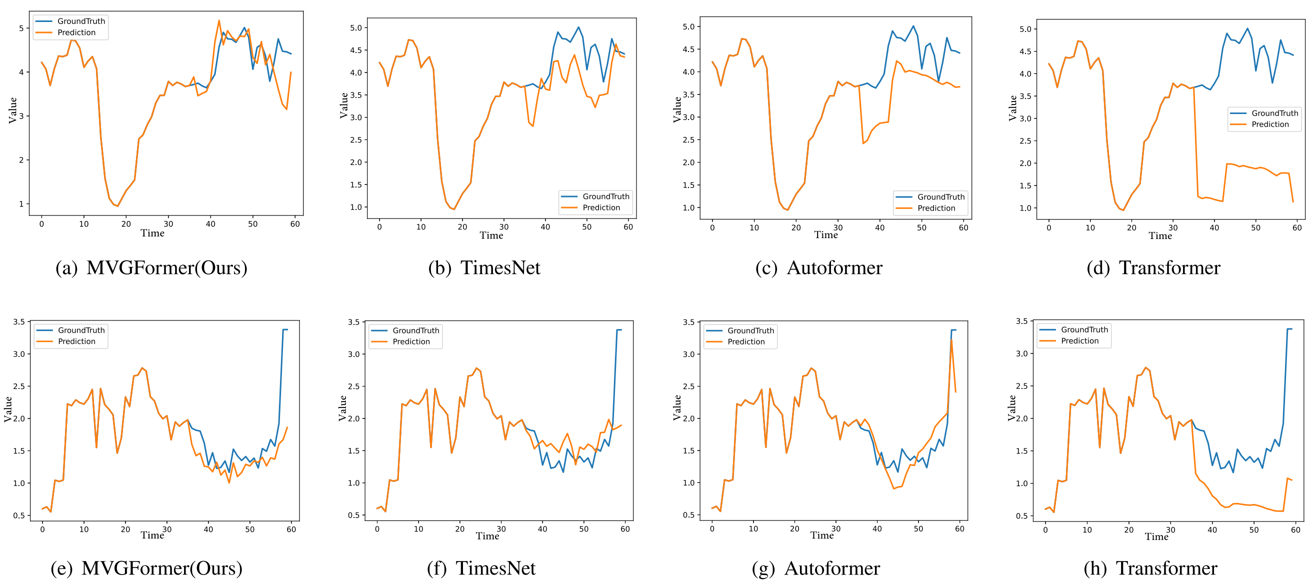


Fig. 11. Forecasting Performance: Visualization of ILI predictions from different models under the input-36-predict-24 setting. The x-axis represents the time dimension, and the y-axis represents the corresponding observed values. The blue lines represent the ground truth, while the orange lines represent the predicted values.

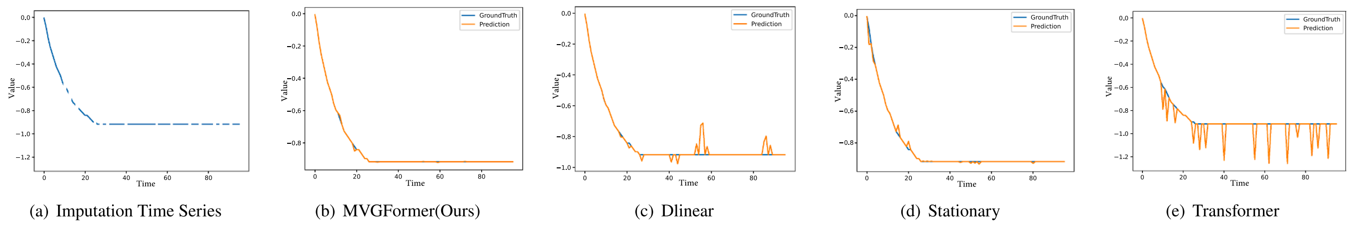


Fig. 12. Imputation Performance: Visualization of ETTh2 imputation results given by different models under the 12.5 % mask ratio setting. The x-axis represents the time dimension, and the y-axis represents the corresponding observed values. The blue lines represent the ground truth and the orange lines represent predicted values.

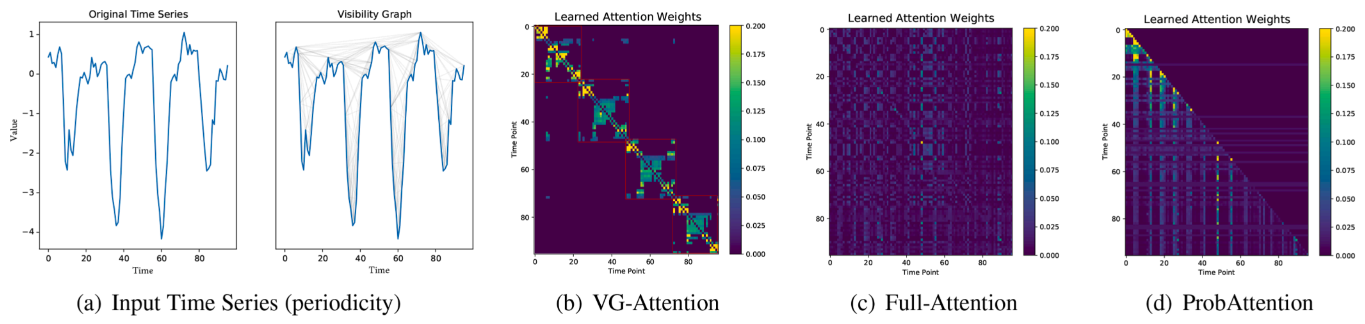


Fig. 13. Visualization of learned temporal Attention for **periodic** time series. (b) is from our VG-Attention; (c) is from standard Full-Attention, fully connected learning; (d) is from Informer [1], a sparse attention algorithm.

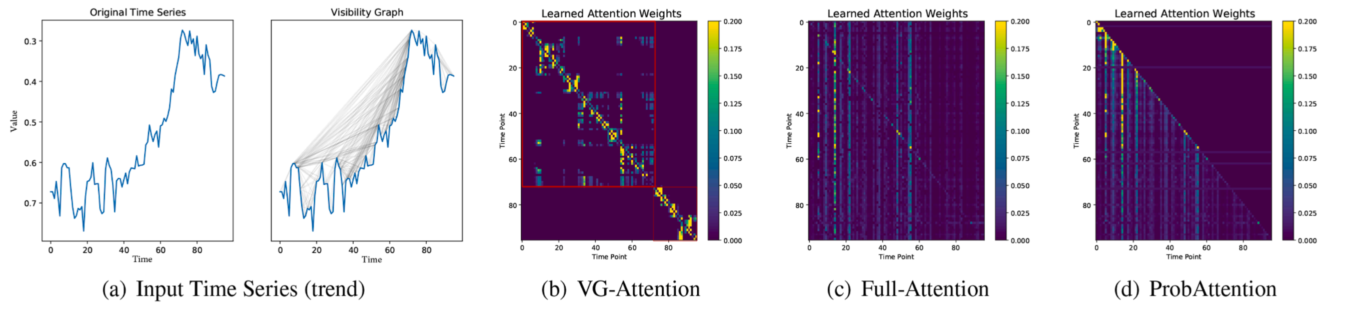
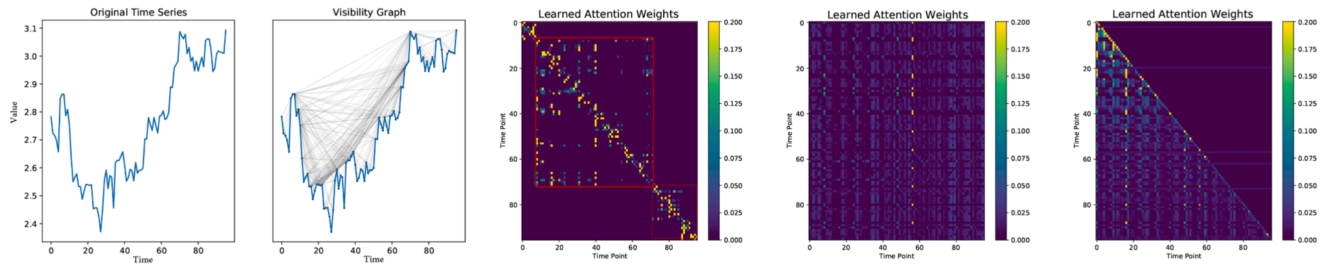
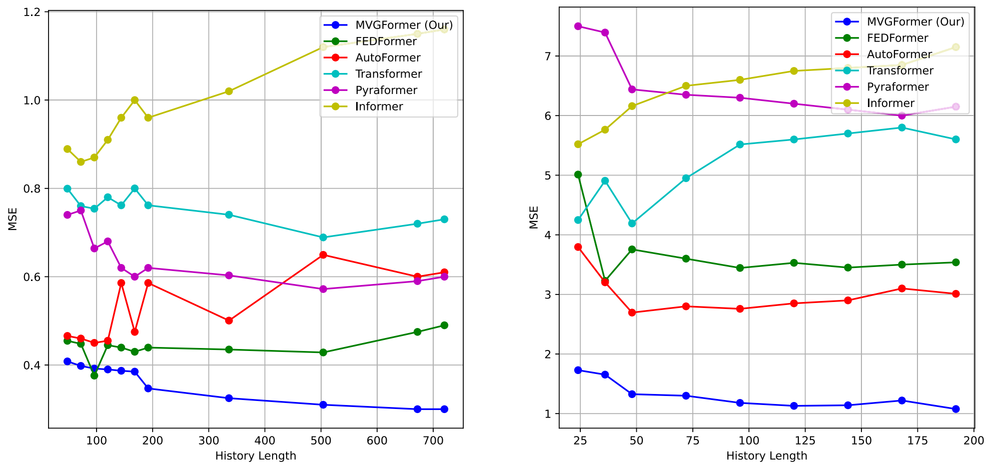


Fig. 14. Visualization of learned temporal Attention for **trend-based** time series. (b) is from our VG-Attention; (c) is from standard Full-Attention, fully connected learning; (d) is from Informer [1], a sparse attention algorithm.



(a) Input Time Series (non-stationary) (b) VG-Attention (c) Full-Attention (d) ProbAttention

Fig. 15. Visualization of learned temporal Attention for **non-stationary** time series. (b) is from our VG-Attention; (c) is from standard Full-Attention, fully connected learning; (d) is from Informer [1], a sparse attention algorithm.



(a) ETTh1 (b) Illness

Fig. 16. Impact of input horizon on forecasting results. The MSE results (Y-axis) of models with different lookback length (X-axis) and fixed prediction length T on the ETTh1 ($T = 96$) and Illness ($T = 24$) datasets.

4.7.5. *Comparison with pure graph-based methods*

To further validate the effectiveness of the visibility graph construction, we compare MVGFormer with other GNN-based models, including the classic STGNN [18], Graph WaveNet [19] and MTGNN [20], as shown in Table 8. Using the Exchange Rate dataset and following the setup from the MTGNN model, we conduct one-step forecasting comparisons based on MAE, RMAE, and MAPE metrics, where Horizon (3,6,9,12) represents the predicted future time steps. The results demonstrate that MVGFormer outperforms these GNN4TS methods, which we attribute to two main reasons: **(1)** STGNN and Graph WaveNet heavily rely on prior graph structure information, which is particularly effective in traffic-related tasks; **(2)** MTGNN constructs inter-channel structural relationships purely through a learning method but neglects temporal dependencies, leading to inferior performance in regression tasks compared to Transformer-based methods. In contrast, MVGFormer compensates for the shortcomings of both, thus achieving superior performance.

Table 8
Comparison with GNN4TS methods.

Horizon	Metric	STGCN	Graph WaveNet	MTGNN	MVGFormer
3	MAE	1.58	1.44	1.07	**0.33**
	RMSE	3.27	2.99	2.12	**0.51**
	MAPE	4.59 %	3.89 %	2.31 %	**1.04 %**
6	MAE	2.12	1.97	1.35	**0.38**
	RMSE	4.36	4.22	2.65	**0.58**
	MAPE	4.60 %	3.55 %	2.27 %	**1.23 %**
9	MAE	2.24	1.97	1.47	**0.38**
	RMSE	4.39	3.97	2.80	**0.59**
	MAPE	4.91 %	3.47 %	2.24 %	**1.47 %**
12	MAE	2.17	1.79	1.48	**0.38**
	RMSE	4.71	3.29	2.86	**0.58**
	MAPE	5.10 %	3.57 %	2.21 %	**1.52 %**

4.8. *Ablation studies*

In this section, we conduct ablation studies to evaluate the effectiveness of each individual component in our proposed method. By systematically removing or modifying different parts of the system, we aim to assess their contribution to the overall performance.

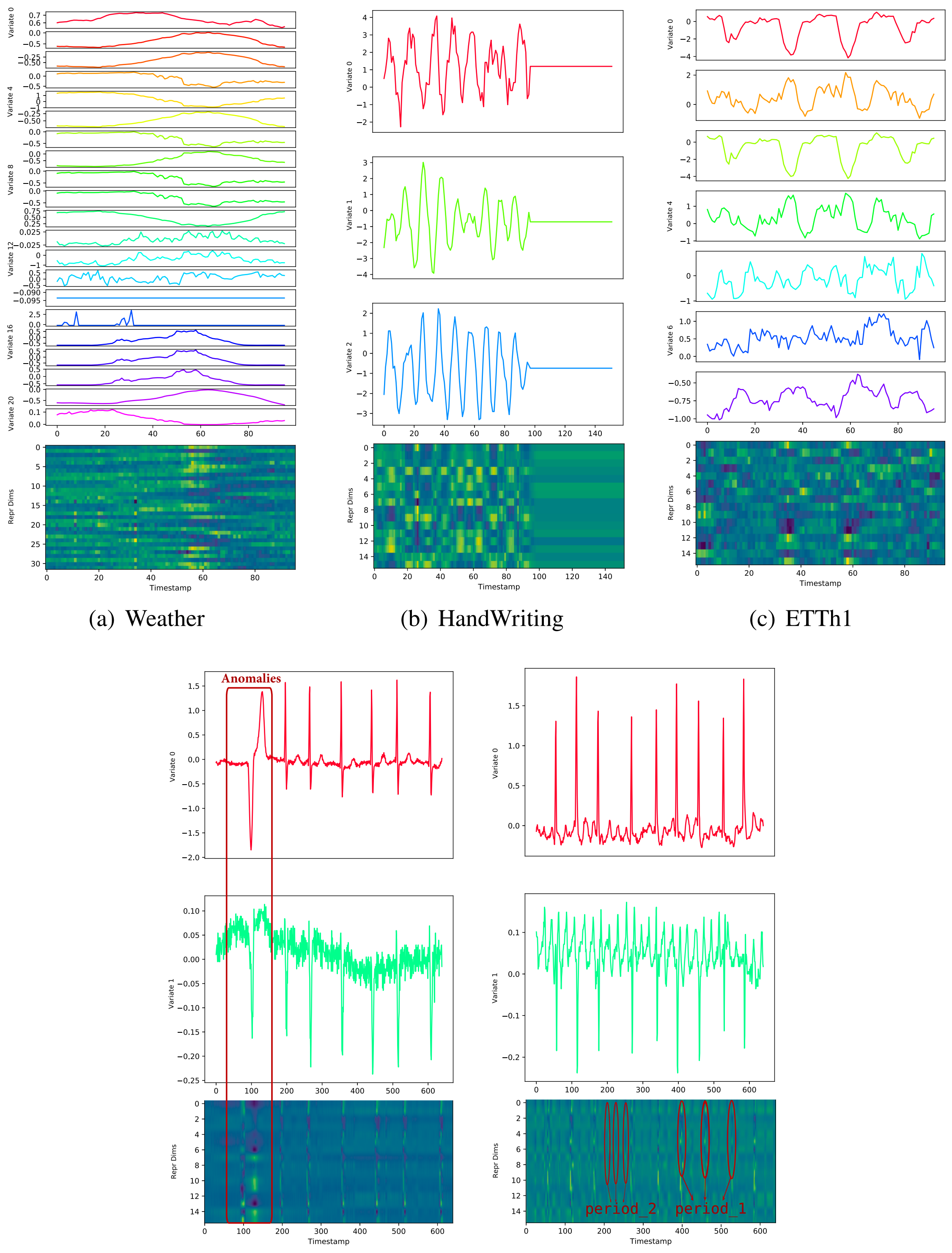
4.8.1. *Main ablation studies*

We conduct the main ablation studies to assess the contribution of each core module in MVGFormer. As shown in Table 9, we focus on three primary components: layer aggregation, consensus visibility graph, and VG-Attention, as follows:

Table 9
Performance comparison of MVGFormer and ablated variants (w/o layer aggregation, consensus graph, VG-Attention) on various datasets. Best results in **bold**.

Model Variant	ETTh1		Weather		Exchange		ILI	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
MVGFormer	**0.458**	**0.448**	**0.257**	**0.287**	**0.450**	**0.4392**	**1.957**	**0.931**
w/o Layer Aggregation	0.482	0.461	0.266	0.298	0.4725	0.451	2.014	0.963
w/o Consensus Graph	0.473	0.459	0.263	0.295	0.466	0.446	2.003	0.955
w/o VG-Attention	0.495	0.475	0.273	0.309	0.489	0.468	2.058	0.978

Results are averaged over four prediction lengths: 96, 192, 336, and 720 for ETTh1, Weather, Exchange; 24, 36, 48, and 60 for ILI.



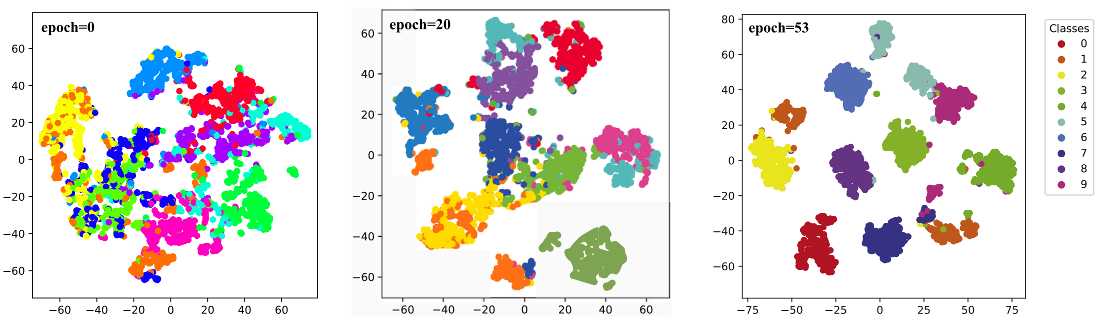
(a) Weather (b) HandWriting (c) ETTh1

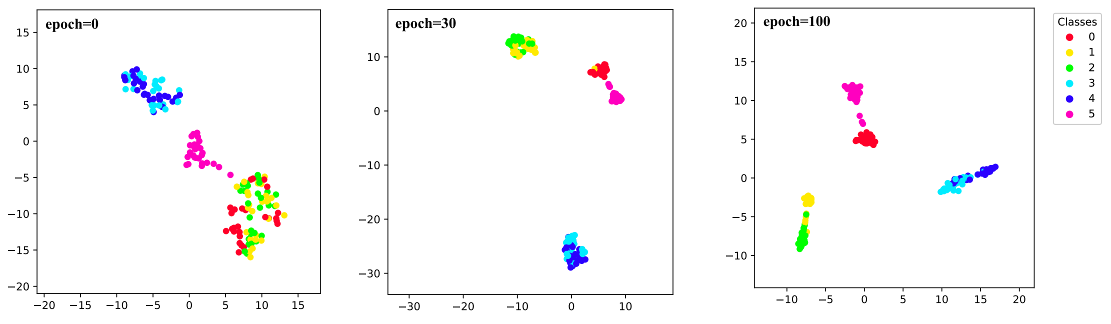
(d) AF:Anomalies (e) AF:Multi-period

Fig. 17. Representation Capability (1):Visualization of multivariate time series embeddings across different datasets (a-c), highlighting anomalies and multi-period patterns in the AtrialFibrillation (AF) dataset (d-e).

- **w/o Layer aggregation:** Each channel is processed independently without multi-layer aggregation before being passed into VG-Attention.
- **w/o Consensus graph:** The consensus visibility graph is replaced by a simple combined graph that merges all edges across layers and time.
- **w/o VG-attention:** The VG-Attention module is replaced with a standard linear projection layer.

These results clearly indicate that all three components are crucial to the effectiveness of MVGFormer. Notably, removing VG-Attention leads to the most significant performance drop, demonstrating its key


(a) PenDigits


(b) NATOPS

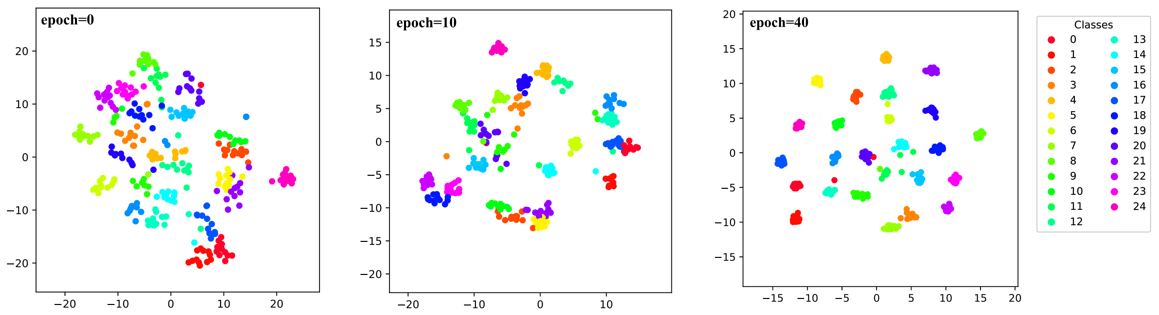

(c) ArticularyWordRecognition

Fig. 18. Representation Capability (2): Embedding visualization using t-SNE on the PenDigits, NATOPS and ArticularyWordRecognition (AWR) from the UEA datasets, depicting the evolution of the learning process during model training.

Table 10
The ablation study of channel-wise consensus information extraction using layer aggregation.

	Dataset	Dims	Aggregation	MSE/Accuracy	Training Time (s/epoch)
Forecasting	Weather	21	w/o_Aggregation	0.234	4200.03s
			w_Aggregation	**0.226**	**2226.09s**
	ETTh1	7	w/o_Aggregation	0.457	325.61s
			w_Aggregation	**0.451**	**135.97s**
	Exchange	7	w/o_Aggregation	0.215	219.85s
			w_Aggregation	**0.207**	**89.83s**
Classification	HandMovementDirection	10	w/o_Aggregation	43.40	32.75s
			w_Aggregation	**45.31**	**3.62s**
	FingerMovements	28	w/o_Aggregation	58.07	3.71s
			w_Aggregation	**64.58**	**0.38s**
	NATOPS	24	w/o_Aggregation	81.45	2.38s
			w_Aggregation	**92.50**	**0.31s**

role in modeling long-range temporal dependencies through the graph structure.

4.8.2. Impact of the channel-wise consensus

To further validate the advantages of channel-wise consensus information extraction in terms of both performance and efficiency, we conducted additional experiments to evaluate MVG-AND layer aggregation mechanism (with and without aggregation) on forecasting and classification tasks, as shown in Table 10. In the w/o_Aggregation scenario, the multivariate time series is treated as univariate, with each channel processed independently (Channel-Independent, CI) and encoded using VG-Attention. Interestingly, after applying MVG-AND for layer

Table 11
The ablation study for Batch Normalization and Layer Normalization on pred_len = 96.

Datasets	*_batch_norm		*_layer_norm	
Metric	MSE	MAE	MSE	MAE
ETTh1	**0.392**	**0.412**	0.396	0.412
ETTh2	**0.323**	**0.369**	0.335	0.371
Weather	**0.178**	**0.229**	0.181	0.233
Exchange	**0.081**	**0.216**	0.101	0.224
ILI	**1.654**	**0.839**	2.297	1.022

aggregation to extracts consensus relationships between channels, both experimental performance and training speed improve significantly. The MSE is reduced by an average of **2.82 %**, classification accuracy increases by an average of **9.73 %**, and training speed improves by an average of **67.48 %**. This demonstrates that strengthening the consensus relationships, leading to sparser connections, does not compromise temporal dependencies but actually enhances inter-channel correlations. Furthermore, as the number of channels increases, performance improves, further emphasizing the importance and necessity of extracting consensus relationships to get a global dependencies for multivariate time series.

4.8.3. Batch normalization vs layer normalization

In MVGFormer, each token (node) corresponding to the values of different channels. Unlike traditional Transformers, which use LayerNorm, this approach requires a different normalization method because the features in multivariate time series data may have different scales (units). BatchNorm normalizes each feature (i.e., each channel) across all time steps in the batch, addressing the issue of inconsistent feature scales and ensuring uniform scaling during training. In contrast, LayerNorm normalizes all features at each time step, which does not effectively resolve the issue of scale inconsistency across time steps. As shown in Table 11, compared to LayerNorm, BatchNorm results in a **10.81 %** improvement in MSE and a **4.74 %** improvement in MAE. Therefore, BatchNorm is more suitable for MVGFormer.

5. Case study: sensitivity to noise

Although the theoretical definition of the Visibility Graph is invariant to affine transformations [10], it is still sensitive to high-frequency noise. For example, consider a periodic sequence $T_1 = \{3, 1, 3, 1, 3, 1, \dots\}$, which, when noise is added, becomes $T_1' = \{3.1, 1.3, 2.7, 1.2, 3.2, 1.1, \dots\}$. This noise may introduce unnecessary connections in the visibility graph, such as a spurious visibility connection between 3.1 (at t_0) and 3.2 (at t_4). This disrupts the original periodic structure, adds edges that do not follow the periodicity, and degrades the graph's interpretability and effectiveness.

To empirically examine this, we gradually added Gaussian noise to the normalized input time series from the ILI dataset[5], which contains low-frequency, noise-free data. As shown in Table 12, the performance of MVGFormer steadily degrades as the noise level increases. This degradation occurs because noise introduces spurious edges into the visibility graph, distorting the temporal structure and misguiding the Attention mechanism, which leads to a focus on non-informative points.

These results reveal a practical limitation of visibility graph-based methods in real-world noisy scenarios. Noise introduces errors that degrade model performance. Future work could address this by incorporating graph denoising strategies, such as denoising autoencoders or refining visibility graph construction to mitigate noise. Several studies have proposed solutions to this issue, such as the Limited Penetrable Visibility Graph (LPVG) [30], and the Circular Limited Penetrable Visibility Graph (CLPVG) [31], both of which demonstrate improved noise resilience. Therefore, a promising direction for future research is the development of more robust visibility graph models to enhance model generalization under noisy conditions, enabling better handling of noisy time series data in real-world settings.

[5] The ILI dataset, reports the weekly number of patients with influenza in the United States, collected by the Centers for Disease Control and Prevention (CDC) from 2002 to 2021. Available at https://gis.cdc.gov/grasp/fluview/fluportaldashboard.html.

Table 12
Discussion on the sensitivity of Visibility Graph to noise.

Noise Level	w/o noise	$\sigma = 0.05$	$\sigma = 0.10$	$\sigma = 0.20$
MSE	**1.654**	1.698	1.812	2.037
MAE	**0.839**	0.861	0.910	0.993

6. Conclusion

MVGFormer, as a general-purpose model, surpasses most current state-of-the-art Transformer-based models, and even TimesNet, across four main tasks. This marks a significant advancement in time-series analysis by incorporating temporal structure into the Transformer architecture, enhanced by the optimized VG-Attention mechanism. This approach provides a novel perspective on guiding Attention in multivariate time series models, utilizing complex network theory to address the challenges of attention dispersion inherent in standard Transformer architectures. Future work will focus on cross-variable visibility principle, further enhancing the representation capability of multivariate time series and uncovering the potential relationships between variables.

CRediT authorship contribution statement

Ting Chen: Writing – original draft, Validation, Methodology, Conceptualization; **Xinyue Ren:** Writing – review & editing, Project administration; **Jinzhou Lai:** Visualization, Resources; **Hongming Tan:** Software; **Fangming Liu:** Funding acquisition, Formal analysis; **Wai Kin Victor Chan:** Methodology, Funding acquisition, Formal analysis.

Data availability

The data is provided with a link in the paper.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This research was funded by the Shenzhen Science and Technology Innovation Commission (WDZC20200818121348001) and the Guangdong Pearl River Plan (2019QN01X890). Additionally, this work was also supported in part by the Major Key Project of PCL under Grant PCL2024A06 and PCL2022A05, and in part by the Shenzhen Science and Technology Program under Grant RCJC20231211085918010.

References

[1] Z. Haoyi, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, W. Zhang, Informer: beyond efficient transformer for long sequence time-series forecasting, in: Proceedings of the AAAI Conference on Artificial Iintelligence, 35, 2021, pp. 11106–11115.
[2] H. Wu, J. Xu, J. Wang, M. Long, Autoformer: decomposition transformers with auto-correlation for long-term series forecasting, Adv. Neural Inf. Process. Syst. 34 (2021) 22419–22430.
[3] Y. Liu, H. Wu, J. Wang, M. Long, Non-stationary transformers: exploring the stationarity in time series forecasting, Adv. Neural Inf. Process. Syst. 35 (2022) 9881–9893.
[4] G. Woo, C. Liu, D. Sahoo, A. Kumar, S. Hoi, Etsformer: exponential smoothing transformers for time-series forecasting, arXiv preprint arXiv:2202.01381 (2022).

[5] Z. Tian, Z. Ma, Q. Wen, X. Wang, L. Sun, R. Jin, Fedformer: frequency enhanced decomposed transformer for long-term series forecasting, in: International Conference on Machine Learning, PMLR, 2022, pp. 27268–27286.
[6] Y. Nie, N.H. Nguyen, P. Sinthong, J. Kalagnanam, A time series is worth 64 words: long-term forecasting with transformers, in: The Eleventh International Conference on Learning Representations, 2023.
[7] Y. Wang, H. Wu, J. Dong, G. Qin, H. Zhang, Y. Liu, Y. Qiu, J. Wang, M. Long, Timexer: empowering transformers for time series forecasting with exogenous variables, Adv. Neural Inf. Process. Syst. 37 (2024) 469–498.
[8] V. Naghashi, M. Boukadoum, A.B. Diallo, A multiscale model for multivariate time series forecasting, Sci. Rep. 15 (1) (2025) 1565.
[9] A. Zeng, M. Chen, L. Zhang, Q. Xu, Are transformers effective for time series forecasting?, in: Proceedings of the AAAI Conference on Artificial Intelligence, 37, 2023, pp. 11121–11128.
[10] L. Lacasa, B. Luque, F. Ballesteros, J. Luque, J.C. Nuno, From time series to complex networks: the visibility graph, Proc. Natl. Acad. Sci. 105 (13) (2008) 4972–4975.
[11] H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, M. Long, TimesNet: temporal 2D-variation modeling for general time series analysis, in: The Eleventh International Conference on Learning Representations, 2023.
[12] M. Jin, H.Y. Koh, Q. Wen, D. Zambon, C. Alippi, G.I. Webb, I. King, S. Pan, A survey on graph neural networks for time series: Forecasting, classification, imputation, and anomaly detection, arXiv preprint arXiv:2307.03759 (2023).
[13] Z. Li, S. Qi, Y. Li, Z. Xu, Revisiting long-term time series forecasting: An investigation on linear mapping, arXiv preprint arXiv:2305.10721 (2023).
[14] H. Ismail Fawaz, B. Lucas, G. Forestier, C. Pelletier, D.F. Schmidt, J. Weber, G.I. Webb, L. Idoumghar, P.-A. Muller, F. Petitjean, Inceptiontime: finding alexnet for time series classification, Data Min. Knowl. Discov. 34 (6) (2020) 1936–1962.
[15] A. Dempster, F. Petitjean, G.I. Webb, ROCKET: exceptionally fast and accurate time series classification using random convolutional kernels, Data Min. Knowl. Discov. 34 (5) (2020) 1454–1495.
[16] J. Sun, S. Takeuchi, I. Yamasaki, Prototypical inception network with cross branch attention for time series classification, in: 2021 International Joint Conference on Neural Networks (IJCNN), IEEE, 2021, pp. 1–7.
[17] X. Qiu, S. Shi, X. Tan, C. Qu, Z. Fang, H. Wang, Y. Gao, P. Wu, H. Li, Gram-based attentive neural ordinary differential equations network for video nystagmography classification, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 21339–21348.
[18] X. Wang, Y. Ma, Y. Wang, W. Jin, X. Wang, J. Tang, C. Jia, J. Yu, Traffic flow prediction via spatial temporal graph neural network, in: Proceedings of the Web Conference 2020, 2020, pp. 1082–1092.
[19] Z. Wu, S. Pan, G. Long, J. Jiang, C. Zhang, Graph wavenet for deep spatial-temporal graph modeling, in: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, International Joint Conferences on Artificial Intelligence Organization, 2019.
[20] Z. Wu, S. Pan, G. Long, J. Jiang, X. Chang, C. Zhang, Connecting the dots: multivariate time series forecasting with graph neural networks, in: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 753–763.
[21] X. Qiu, J. Qian, H. Wang, X. Tan, Y. Jin, An attentive copula-based spatio-temporal graph model for multivariate time-series forecasting, Appl. Soft. Comput. 154 (2024) 111324.
[22] H. Wang, X. Qiu, Y. Xiong, X. Tan, AutoGRN: an adaptive multi-channel graph recurrent joint optimization network with copula-based dependency modeling for spatio-temporal fusion in electrical power systems, Inf. Fus. 117 (2025) 102836.
[23] J. Xu, H. Wu, J. Wang, M. Long, Anomaly transformer: time series anomaly detection with association discrepancy, in: International Conference on Learning Representations, 2021.
[24] S. Sannino, S. Stramaglia, L. Lacasa, D. Marinazzo, Visibility graphs for fMRI data: multiplex temporal graphs and their modulations across resting-state networks, Netw. Neurosci. 1 (3) (2017) 208–221.
[25] Y. Xiu, X. Ren, T. Zhang, Y. Chen, L. Jiang, D. Li, X. Wang, L. Zhao, W.K. Chan, Time labeled visibility graph for privacy-preserved physiological time series classification, in: 2022 7th International Conference on Cloud Computing and Big Data Analytics (ICCCBDA), IEEE, 2022, pp. 280–284.
[26] R. Flanagan, L. Lacasa, Irreversibility of financial time series: a graph-theoretical approach, Phys. Lett. A 380 (20) (2016) 1689–1697.
[27] Y. Huang, X. Mao, Y. Deng, Natural visibility encoding for time series and its application in stock trend prediction, Knowl. Based Syst. 232 (2021) 107478.
[28] D. Zhao, X. Yang, W. Song, W. Zhang, D. Huang, Visibility graph analysis of the sea surface temperature irreversibility during El Niño events, Nonlinear Dyn. 111 (18) (2023) 17393–17409.
[29] B. Luque, L. Lacasa, F. Ballesteros, J. Luque, Horizontal visibility graphs: exact results for random time series, Phys. Rev. E Stat. Nonlinear Soft Matter Phys. 80 (4) (2009) 046103.
[30] T.-T. Zhou, N.-D. Jin, Z.-K. Gao, Y.-B. Luo, Limited penetrable visibility graph for establishing complex network from time series, Acta Phys. Sinica. 61 (2012) 030506.
[31] Q. Xuan, J. Zhou, K. Qiu, D. Xu, S. Zheng, X. Yang, CLPVG: circular limited penetrable visibility graph as a new network model for time series, Chaos Interdiscip. J. Nonlinear Sci. 32 (1) (2022).
[32] J. Iacovacci, L. Lacasa, Visibility graphs for image processing, IEEE Trans. Pattern Anal. Mach. Intell. 42 (4) (2019) 974–987.
[33] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[34] Q. Xuan, J. Zhou, K. Qiu, Z. Chen, D. Xu, S. Zheng, X. Yang, Avgnet: adaptive visibility graph neural network and its application in modulation classification, IEEE Trans. Network Sci. Eng. 9 (3) (2022) 1516–1526.
[35] L. Chen, D. Chen, Z. Shang, B. Wu, C. Zheng, B. Wen, W. Zhang, Multi-scale adaptive graph neural network for multivariate time series forecasting, IEEE Trans. Knowl. Data Eng. 35 (10) (2023) 10748–10761.
[36] T. Wen, H. Chen, K.H. Cheong, Visibility graph for time series prediction and image classification: a review, Nonlinear Dyn. 110 (4) (2022) 2979–2999.
[37] H. Azizi, S. Sulaimany, A review of visibility graph analysis, IEEE Access. 12 (2024) 93517–93530.
[38] S. Liu, H. Yu, C. Liao, J. Li, W. Lin, A.X. Liu, S. Dustdar, Pyraformer: low-complexity pyramidal attention for long-range time series modeling and forecasting, in: International Conference on Learning Representations, 2021.
[39] Y. Liu, T. Hu, H. Zhang, H. Wu, S. Wang, L. Ma, M. Long, iTransformer: inverted transformers are effective for time series forecasting, in: The Twelfth International Conference on Learning Representations, 2024.
[40] Y. Liu, T. Hu, H. Zhang, H. Wu, S. Wang, L. Ma, M. Long, iTransformer: inverted transformers are effective for time series forecasting, in: The Twelfth International Conference on Learning Representations, 2024. https://openreview.net/forum?id=JePfAI8fah.
[41] X. Lan, H. Mo, S. Chen, Q. Liu, Y. Deng, Fast transformation from time series to visibility graphs, Chaos Interdiscip. J. Nonlinear Sci. 25 (8) (2015).
[42] L. Lacasa, V. Nicosia, V. Latora, Network structure of multivariate time series, Sci. Rep. 5 (1) (2015) 15508.
[43] A. Lancichinetti, S. Fortunato, Consensus clustering in complex networks, Sci. Rep. 2 (1) (2012) 336.
[44] K. Zhan, F. Nie, J. Wang, Y. Yang, Multiview consensus graph clustering, IEEE Trans. Image Process. 28 (3) (2018) 1261–1270.
[45] Z. Li, C. Tang, X. Liu, X. Zheng, W. Zhang, E. Zhu, Consensus graph learning for multi-view clustering, IEEE Trans. Multimedia 24 (2021) 2461–2472.
[46] X. Li, S. Yin, X. Liu, C. Gao, Z. Wang, V.I. Nekorkin, Consensus subspace graph regularization based on prior information for multiplex network clustering, Eng. Appl. Artif. Intell. 135 (2024) 108851.
[47] Z. Gu, S. Feng, J. Yuan, X. Li, Consensus representation-driven structured graph learning for multi-view clustering, Appl. Intell. 54 (17) (2024) 8545–8562.
[48] M. Kivelä, M.A. Porter, Isomorphisms in multilayer networks, IEEE Trans. Network Sci. Eng. 5 (3) (2017) 198–211.
[49] M. Wu, J. Chen, S. He, Y. Sun, S. Havlin, J. Gao, Discrimination reveals reconstructability of multiplex networks from partial observations, Commun. Phys. 5 (1) (2022) 163.
[50] S. Du, Z. Cai, Z. Wu, Y. Pi, S. Wang, UMCGL: universal multi-view consensus graph learning with consistency and diversity, IEEE Trans. Image Process. (2024).
[51] J.-Y. Franceschi, A. Dieuleveut, M. Jaggi, Unsupervised scalable representation learning for multivariate time series, Adv. Neural Inf. Process. Syst. 32 (2019).
[52] T. Zhang, Y. Zhang, W. Cao, J. Bian, X. Yi, S. Zheng, J. Li, Less is more: fast multivariate time series forecasting with light sampling-oriented mlp structures, arXiv preprint arXiv:2207.01186 (2022).
[53] M.M.N. Murad, M. Aktukmak, Y. Yilmaz, Wpmixer: efficient multi-resolution mixing for long-term time series forecasting, in: Proceedings of the AAAI Conference on Artificial Intelligence, 39, 2025, pp. 19581–19588.
[54] A. Gu, K. Goel, C. Re, Efficiently modeling long sequences with structured state spaces, in: International Conference on Learning Representations, 2021.
[55] H. Liu, X. Liu, D. Yang, Z. Liang, H. Wang, Y. Cui, J. Gu, TodyNet: temporal dynamic graph neural network for multivariate time series classification, arXiv preprint arXiv:2304.05078 (2023).
[56] G. Lai, W.-C. Chang, Y. Yang, H. Liu, Modeling long-and short-term temporal patterns with deep neural networks, in: The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, 2018, pp. 95–104.
[57] K. Hundman, V. Constantinou, C. Laporte, I. Colwell, T. Soderstrom, Detecting Spacecraft Anomalies Using Lstms and Nonparametric Dynamic Thresholding, in: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining, 2018, pp. 387–395.
[58] Y. Su, Y. Zhao, C. Niu, R. Liu, W. Sun, D. Pei, Robust anomaly detection for multivariate time series through stochastic recurrent neural network, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 2828–2837.
[59] A.P. Mathur, N.O. Tippenhauer, SWaT: a water treatment testbed for research and training on ICS security, in: 2016 International Workshop on Cyber-Physical Systems for Smart Water Networks (CySWater), IEEE, 2016, pp. 31–36.
[60] A. Abdulaal, Z. Liu, T. Lancewicki, Practical approach to asynchronous multivariate time series anomaly detection and localization, in: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, 2021, pp. 2485–2494.
[61] A. Bagnall, H.A. Dau, J. Lines, M. Flynn, J. Large, A. Bostrom, P. Southam, E. Keogh, The UEA multivariate time series classification archive, 2018, arXiv preprint arXiv:1811.00075 (2018).