Knowledge-Based Systems xxx (xxxx) xxx

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Long-term multivariate time series forecasting in data centers based on multi-factor separation evolutionary spatial–temporal graph neural networks

Fang Shen [a,*], Jialong Wang [a,*], Ziwei Zhang [b], Xin Wang [b,*], Yue Li [a], Zhaowei Geng [a], Bing Pan [a], Zengyi Lu [a], Wendy Zhao [c], Wenwu Zhu [b,*]

[a] Alibaba Group, China
[b] Department of Computer Science and Technology, BNRist, Tsinghua University, China
[c] Alibaba Group, Bellevue WA, 98004, USA

ARTICLE INFO

Keywords:
Multivariate time series forecasting
Spatial–temporal graph neural network
Relational inference
Graph structure learning

ABSTRACT

Data center infrastructures require constant monitoring to ensure stable and reliable operation and time-series forecasting plays an indispensable role in intelligent operations and maintenance in data centers. However, the potential for accurate time-series predictions is often limited due to the overlooked relationships between data records from independent sensors. Inferring relationships for a potential graph representation of a data center is challenging due to complex relationships between nodes and multiple factors that may cause connections between them. Moreover, graphs change dynamically in long-term predictions, but current methods do not account for future graph changes. To address these challenges, we propose a long-term time-series forecasting framework called Multi-factor Separation Evolutionary Spatial–Temporal Graph Neural Networks (MSE-STGNN). Our framework considers edge diversity, graph changes and spatial–temporal architecture in long-term prediction processes and proposes three modules. Specifically, we propose a Multi-factor Separation (MS) module to separate the factors influencing node connectivity, enabling the acquisition of a graph more closely aligned with actual circumstances; then we propose a Graph Prediction (GP) module to incorporate future graphs to correct errors in the graph on which multi-step predictions depend. Moreover, we propose an Attention-enhanced Spatial–temporal dilated causal convolution module (AS-Conv) to more effectively leverage information pertaining to spatial and historical events. Our experimental results on datasets comprising of temperature and IT power data collected from real-world data centers show that the proposed method outperforms other advanced prediction methods in terms of prediction accuracy, and the learned latent graphs are explainable.

1. Introduction

A data center is a physical facility for centralized management and processing of data, including servers, storage devices, and network devices. Intelligent operation and maintenance of data centers [1–7] refer to the utilization of machine learning and automation technologies to achieve intelligent monitoring and maintenance of data center equipment, thereby enhancing the efficiency and reliability of the data center. This approach to operations and maintenance can detect and solve potential faults in real-time by way of continuous monitoring and automated prediction, thereby reducing maintenance costs while increasing data center productivity. Time series forecasting [8–11] has been widely used in intelligent monitoring of data center equipment [12,13]. For example, time series forecasting can analyze the performance indicators of data center equipment, predict potential faults, and perform timely maintenance to avoid the adverse impact of faults on business operations. It can also analyze business load and resource utilization rates to predict future resource requirements, optimize resource planning, and enhance resource utilization efficiency. Moreover, it can analyze energy consumption data, predict energy consumption trends, plan energy consumption optimization, and reduce energy consumption costs.

The data center infrastructure is outfitted with an array of sensors, each responsible for monitoring the health status of a specific component [14]. Regrettably, the information captured by these sensors is often dissociated from its pertinent context, residing in isolation within the database [2]. However, the detection and preservation of sensor

* Corresponding authors.
E-mail addresses: f.shen@qq.com (F. Shen), quming.wjl@alibaba-inc.com (J. Wang), xin_wang@tsinghua.edu.cn (X. Wang), wwzhu@tsinghua.edu.cn (W. Zhu).

https://doi.org/10.1016/j.knosys.2023.110997
Received 19 June 2023; Received in revised form 21 August 2023; Accepted 12 September 2023

relations in the form of a latent graph would be greatly beneficial to the intelligent operation and maintenance of the data center, facilitating predictions and causal discoveries. Consider, for example, a scenario where the interconnections between cabinets in a single room are unknown, but the prediction of cabinet temperature can be enhanced through consideration of spatial dependencies. The latent relationships between data center nodes can be ascertained from historical data through the application of data-driven techniques. The utilization of graph-based approaches for time series forecasting facilitates the adaptive learning of temporal patterns [15–17]. By analyzing and learning from graphs, associations and patterns in time series data are automatically discovered, which helps in handling complex multivariate time series data. Graph-based time series forecasting has been applied in various domains, including traffic flow prediction [18–20], stock price forecasting [21–24], air quality prediction [25–27], disease occurrence prediction [28], and others.

Accurate time series prediction [29–32] is indispensable in intelligent operation and maintenance of data centers. For instance, the temperature prediction of cabinets can be utilized to forecast the temperature rise in the upcoming period or estimate the time required for the temperature to reach the shutdown temperature, thus assisting businesses in advance escape and migration under high-risk situations. Furthermore, it can also be applied in emergency response to hypothesize and deduce some intervention measures, thereby improving the estimation of the impact surface of the entire temperature rise event. Since graph neural networks (GNN) [33–35] can be utilized to learn complex relationships between variables, they have achieved significant success in spatial–temporal prediction. The pre-requisite for the utilization of GNN entails the establishment of a graph that delineates the inter-relationships between variables, which has consequently led to notable efforts towards constructing graphs for time series forecasting. For instance, the neural relational inference (NRI) [36] method employs a variational autoencoder (VAE) [37] to discern interactions within the encoder, subsequently utilizing the learned interactions in the decoder for predicting physical system trajectories. Shang et al. [38] proposed the graph for time series (GTS) as an improvement of NRI to learn a fixed graph structure, as the output graph in the encoder of NRI tends to change based on the input data. Additionally, MTGNN [15] learns connections through a graph learning layer with learnable embeddings, and subsequently applies spatial–temporal prediction in various domains such as traffic, electricity, and solar energy. StemGNN [16] learns implicit relations employing the attention mechanism, with spatial–temporal features obtained in the frequency domain. The aforementioned methods are all aimed at time series forecasting and learn connections by providing feedback on forecasting accuracy.

However, extant graph-based time series forecasting methods have failed to consider the multifarious factors that contribute to the connection between nodes in intelligent operation and maintenance of data centers. Specifically, these methods assume that edges between two nodes belong to only one category, and do not permit edges to be simultaneously classified according to multiple categories. To illustrate, consider the prediction of cabinet temperature, a scenario in which the reasons for the connection between two cabinets may include sharing a cold aisle, exhibiting similar business loads, and being proximal in space, as shown in Fig. 1. However, previous methods fail to account for the various factors that may underlie connectivity and instead conflate these factors, which may mislead the model. For instance, these models may erroneously conclude that the cabinet power consumption characteristics of one node (a cabinet) are correlated with the cold aisle characteristics of another node, which is patently nonsensical. By contrast, if the features associated with distinct factors that influence edge connectivity can be separated when learning graph structure, the resultant graph will be more readily interpretable.

Moreover, previous graph-based time series forecasting methods have been faulted for their lack of consideration of the ways in which graphs may dynamically change over time in long-term prediction. For instance, to predict cabinet temperature for the next day, temperature is recorded every 2.5 min, necessitating the prediction of 576 time points. However, given that different business processes may run during the day and night, the temperature relationship graph between cabinets may shift dynamically after 288 time points such that some nodes that were previously disconnected may become connected, while some nodes that previously enjoyed connectivity may no longer be connected.

In light of these deficits, we propose a novel graph-based time series prediction model that accounts for the complex factors that underlie edge connectivity and considers the dynamic nature of graph evolution in long-term prediction. The model's framework is illustrated in Fig. 2 and is rooted in the VAE architecture, wherein the encoder infers the relationship between nodes while the decoder is used for time-series prediction and to utilize prediction results as feedback to rectify the graph structure. This approach simulates the operational process of real systems, thereby enabling the learned graph to provide feedback on actual connections. To separate different factors influencing the connections between the nodes, we introduce a multifactor separation layer to infer the underlying factors behind each edge. In data centers, there may be multiple factors causing the nodes to connect, and thus separating the features of different factors helps in learning a graph that is closer to reality. To tackle the problem of dynamic changes in the graph during long-term prediction, we propose a graph prediction module that uses auxiliary variable prediction to guide changes in the graph structure. Additionally, we propose attention-enhanced spatial–temporal dilated causal convolution to better utilize the separated graph structure information and historical temporal information for time-series forecasting. We conduct a comprehensive performance evaluation of our proposed model vis-à-vis contemporary approaches on authentic datasets encompassing cabinet temperature and IT workload in data centers. Our study establishes the efficacy of our model in enhancing prediction accuracy as well as enabling interpretability.

Our main contributions can be summarized as follows:

- To enhance the graph representation of data center infrastructure operation, we propose a Multi-factor Separation (MS) module that captures connection relationships generated by various factors.
- We introduce a Graph Prediction (GP) module to address the problem of changing graph structures in long-term prediction, providing more accurate spatial information for time series predictions.
- We propose an Attention-enhanced Spatial–temporal causal convolution (AS-Conv) module, which better utilizes the spatial features obtained from multifactor separation and addresses the problem of periodic deviations.

2. Related work

In this section, we first review recent latest advancements in time series prediction algorithms without predefined graphs. We subsequently delve into the discussion of time series prediction algorithms for graphs with predefined structures. Lastly, we provide an overview of the current state-of-the-art in the field of time series prediction and further explore the limitations of existing graph-based time series prediction methods in the context of intelligent data center operations.

The graph-based time series prediction methods can be roughly divided into two categories: those without predefined graphs and those with graph structures. In the former, the structure of the graph needs to be learned before prediction, or the structure of the graph needs to be learned while predicting, and in most time series predictions, there is no predefined graph structure. On the other hand, the latter utilizes existing graph structures and historical time series information to predict future values. The incorporation of graph structures in time series prediction has been shown to improve the accuracy and provide a better understanding of the underlying dynamics of the system.

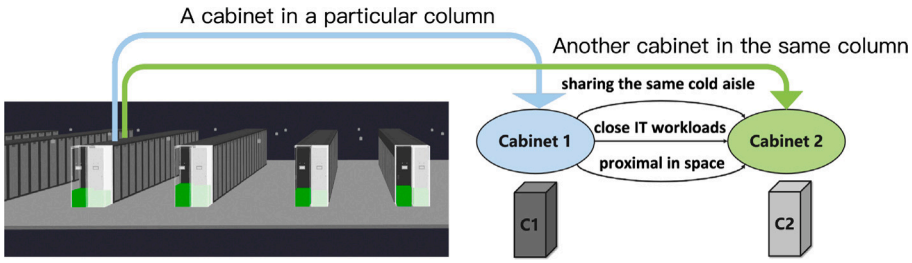


Fig. 1. In data centers, an edge between two nodes may belong to multiple categories simultaneously. In the scenario of predicting the temperature of cabinets, there may be three reasons why two cabinets are connected: they share the same cold aisle, have similar business loads, and are close in space.

2.1. Prediction with predefined graphs

The various variations of GNN have contributed to the expansion of deep models to non-Euclidean spaces, ultimately leading to the achievement of state-of-the-art performance in various applications such as recommender systems, drug discovery and social networks. Despite their remarkable success, GNN are impeded by static graph structures, which restricts their performance when confronted with dynamic data. To address this limitation, Spatial–temporal GNN have been developed as an extension of GNN to incorporate the temporal dimension. Recently, diverse Spatial–temporal Graph Neural Network algorithms have been proposed and have exhibited superior performance compared to other traditional and deep learning algorithms in time-dependent applications.

Spatial–temporal graph convolutional networks (STGCN) [39] introduces graph convolutional networks (GCN) [40–42] into spatial–temporal prediction for the first time. STGCN eliminates the use of recurrent neural network (RNN) [43,44] structures entirely, instead utilizing only convolutional structures. The model constructs a temporal gated convolution module (TGC) using a gated linear unit and 1d convolution structure, and then inserts a spatial graph convolution module between two temporal gated convolution modules to form a fundamental spatial–temporal convolutional unit. By stacking these spatial–temporal convolutional units and introducing residual modules between them, STGCN effectively extracts the medium- and long-term spatial–temporal features of the spatial–temporal graph data while reducing the number of model parameters and increasing training speed. However, the STGCN model exhibits a relatively coarse extraction of temporal features despite its fast training. In light of this, Guo et al. [45] proposed the attention based spatial–temporal graph convolutional networks (ASTGCN) model, which incorporates attention mechanisms within the basic spatial–temporal feature extraction module. By integrating attention mechanisms during the extraction of both temporal and spatial features, the model is capable of capturing the dynamic and complex spatial–temporal correlations of nodes.

Li et al. [46] propose the diffusion convolutional recurrent neural network (DCRNN) framework, which is designed to capture both the spatial and temporal dependencies in traffic flow. Specifically, DCRNN leverages bidirectional random walks on the graph to capture spatial dependency, and an encoder–decoder architecture with scheduled sampling to capture temporal dependency. Wu et al. [47] proposes Graph WaveNet for spatial–temporal graph modeling by developing a novel adaptive dependency matrix and learning it through node embedding. The model is equipped with a stacked dilated 1D convolution component whose receptive field grows exponentially as the number of layers increases, enabling it to handle very long sequences. Zheng et al. [48] introduce the graph multi-attention network (GMAN) as a means of predicting traffic conditions at different locations on a road network graph. GMAN utilizes an encoder–decoder structure where multiple spatial–temporal attention blocks are incorporated into both the encoder and decoder to model the influence of spatial–temporal factors on traffic conditions. The encoder is responsible for encoding the input traffic features while the decoder generates the output sequence. An attention layer is employed between the encoder and decoder to convert the encoded traffic features.

Hadou et al. [49] presents the space–time GNN (ST-GNN) which specifically designed to process the latent space–time topology of time-varying network data. The proposed architecture employs a composition of time and graph convolutional filters followed by pointwise nonlinear activation functions, which can mimic the diffusion process of signals. Chen et al. [50] propose the time-aware multipersistence spatio-supra GCN (TAMP-S2GCNets), which combines the emerging field of topological data analysis with time-aware deep learning. They leverage the tools of multipersistence to capture hidden time-conditioned properties and summarize them as a time-aware multipersistence Euler-Poincar'e surface, which they prove to be stable. A supragraph convolution module that concurrently accounts for intra- and inter-spatio-temporal dependencies.

2.2. Prediction without predefined graphs

The algorithm for learning graph structures in the context of time series prediction has gradually emerged since the inception of GNN. NRI, proposed by Kipf et al. [36] is a GNN-based approach designed for modeling and predicting the behavior of complex systems with interacting components, such as physical systems. This approach combines relational inference and message passing to learn the relationships between the system's components and their dynamics. The model has been shown to achieve high precision in modeling the dynamics of physical systems, real motion tracking, and sports analytics data, which has established its potential to advance the understanding and prediction of complex systems. GTS [38] introduces a novel method for forecasting multiple time series using a GNN, which integrates structure learning inspired by NRI and a recurrent graph convolution forecaster based on the inferred graph as in DCRNN. The approach is centered on the optimization of the expectation over the graph distribution, which is parameterized by a neural network and formulated as a probabilistic graphical model. This process culminates in a single differentiable objective that encapsulates the graph distribution. Furthermore, GTS demonstrates enhanced computational efficiency in comparison to LDS [51], a recently developed meta-learning graph-based method.

MTGNN [15] presents a GNN framework for multivariate time series (MTS) forecasting that automatically extracts directed relations among variables using a graph learning module. A mix-hop propagation layer and a dilated inception layer are proposed to capture spatial and temporal dependencies within the time series. The model achieves state-of-the-art performance on benchmark datasets and on-par performance with other approaches on traffic datasets that provide extra structural information. StemGNN [16] proposes the spectral temporal GNN for multivariate time-series forecasting, which captures both intra-series temporal correlations and inter-series correlations jointly in the spectral domain using Graph Fourier Transform (GFT) and Discrete Fourier Transform (DFT). The proposed model learns inter-series correlations automatically from data without using pre-defined priors and achieves improved accuracy on ten real-world datasets compared to existing methods.

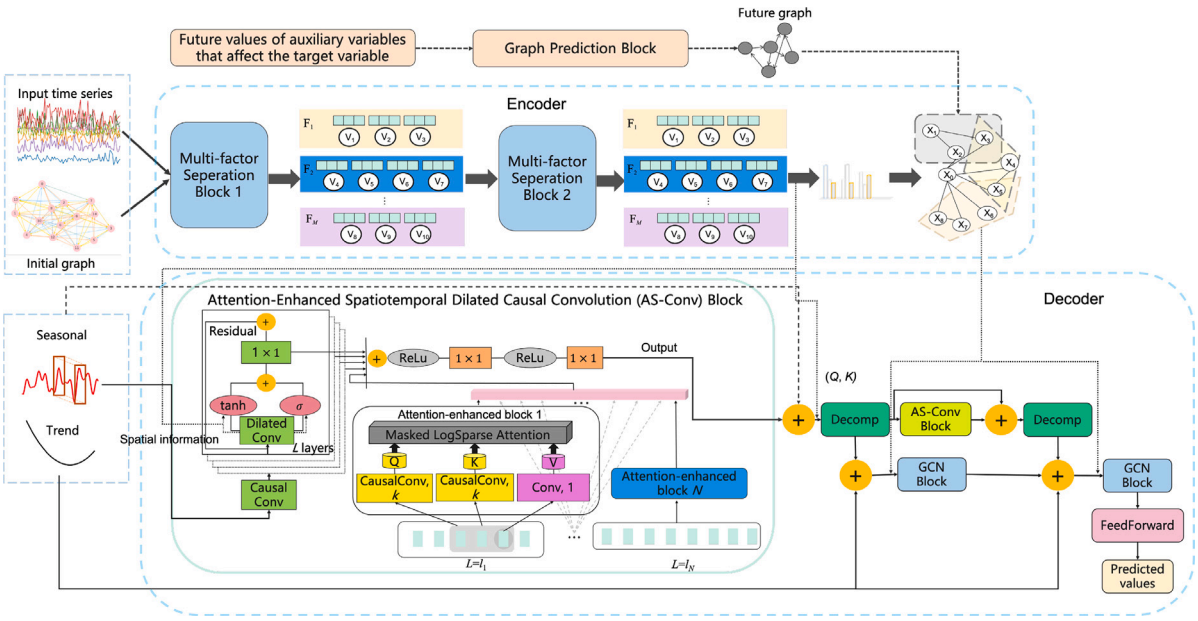


Fig. 2. MSE-STGNN architecture. The encoder inputs the historical data and uses the multi-factor separation module to learn the latent graph. The decoder inputs the learned graph and the separated time series to predict future values.

Shao et al. [52] proposes a framework that enhances Spatial-Temporal Graph Neural Networks (STGNNs) for MTS forecasting by incorporating a scalable time-series pre-training model (STEP). The pre-training model is designed to learn temporal patterns from long-term historical MTS data and generate segment-level representations that provide contextual information for short-term time series input to STGNNs, improving modeling of dependencies between time series. Experiments on three real-world datasets demonstrate that the proposed framework significantly enhances the performance of STGNNs and captures temporal patterns effectively. Ye et al. [53] proposes a method for MTS forecasting using GNN that can model dynamic and evolving interactions of variables. The method includes a hierarchical graph structure with dilated convolution to capture scale-specific correlations among time series, and a recurrent manner to construct a series of adjacency matrices representing evolving correlations at each layer. The proposed method outperforms existing approaches in single-step and multi-step forecasting tasks.

The connectivity relationships between nodes in data centers are often unknown and require learning through data-driven approaches. However, existing methods for predicting graphs without predefined topologies have neglected the diversity of connections in data center scenarios and the dynamic changes in graph structures in long-term predictions. Consequently, there exists a need for novel approaches that can effectively model the complexities of connectivity patterns in data centers while accounting for temporal variations in graph structures.

3. Model structure

In this section, we present a long-term time series prediction algorithm based on the multifactorial dissociation evolutionary graph, which enhances the prediction performance by considering the diversity of edges and the variation of the graph in the long-term prediction process, and improving the network prediction structure. This section aims to provide a comprehensive exposition on the following issues: (1) the global architecture of the MSE-STGNN framework, as depicted in the accompanying schematic diagram; (2) the algorithmic workflow of the multi-factor separation module; (3) the intricate structure of the causal convolution module enhanced by the attention mechanism; and (4) the specific implementation details of the dynamic graph augmentation module. By elucidating these key components and their interconnections, we seek to establish a thorough understanding of the MSE-STGNN methodology and its potential applications in the field of data center intelligent operation and maintenance.

3.1. MSE-STGNN framework

Preliminary. The present study proposes a GNN model that operates on a graph, where the input entails the historical data associated with each node. Specifically, the historical value of the ith node is represented as $[x_i^1, x_i^2, \ldots, x_i^T]$, where T connotes the total number of time steps recorded in history. Meanwhile, the value of all nodes at a single time step is denoted as $[x_1^t, x_2^t, \ldots, x_N^t]$, where N represents the total number of nodes. We can represent all time steps of all nodes as $\mathbf{x} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N]$. Given the relationship between the sensors i and j, denoted by a_{ij}, we posit that the GNN model has the capacity to simulate the interconnections between different nodes and their operational states. Notably, the types of connections that we learn in this context are discrete.

Model structure. The MSE-STGNN model is comprised of two interdependent modules that are concomitantly trained: an encoder that is equipped with the capacity to infer connections through historical data and a decoder that leverages the learned graph to predict future values. The fundamental objective of the model is to conduct synchronized learning of the interrelationships between the nodes, as well as the forecast of the future states of each node. The graphical structure of the model is exhibited in Fig. 2.

Specifically, the encoder produces a factorized distribution $q_\theta(\mathbf{A}|\mathbf{x})$ of a_{ij}, where a_{ij} is a discrete variable representing the type of relationship between nodes i and j. On the other hand, the decoder models $p_\phi(\mathbf{x}|\mathbf{A})$ by leveraging the acquired graph structure and historical values, thereby enabling the exploitation of spatial–temporal properties to predict future states.

3.2. Encoder: Multi-factor separation

In this section, we describe how the encoder learns the graph structure with multi-factor separation. The encoder deduces the relationship by initializing a fully connected graph at the outset. Prior knowledge is then harnessed to eliminate edges that would not feasibly exist between

Algorithm 1 Multi-factor Separation

Input: $\mathbf{x}_o \in \mathbb{R}^{d_{in}}$: the feature vector of node o; $\{\mathbf{x}_{neigh} : (o, neigh) \in E\}$: the feature vectors of the neighbors of node o;
Output: The factor-separation representation of node o: $\mathbf{y}_o = [\mathbf{r}_1, \mathbf{r}_2, \cdots, \mathbf{r}_M]$, $\mathbf{r}_m \in \mathbb{R}^{\frac{d_{out}}{M}}$ $(1 \le m \le M)$;
while $i \in o \cup \{neigh : (o, neigh) \in E\}$ **do**
  for $m = 1, 2, \ldots, M$ **do**
    $\mathbf{u}_{i,m} \leftarrow \frac{\delta(\mathbf{W}_m^\top \mathbf{x}_i + \mathbf{b}_m)}{\|\delta(\mathbf{W}_m^\top \mathbf{x}_i + \mathbf{b}_m)\|_2}$;
  end for
end while
Initialize M feature representations for node o: $o_m \leftarrow \mathbf{u}_{o,m}, \forall m = 1, 2, \ldots, M$;
while iteration $k \le K$ **do**
  for $neigh \in (o, neigh) \in E$ **do**
    $p_{neigh,m} \leftarrow \mathbf{u}_{neigh,m}^T \mathbf{o}_m / \epsilon, \forall m = 1, 2, \cdots, M$
    $p_{neigh,m} \leftarrow sigmoid(p_{neigh,m}), \forall m = 1, 2, \cdots, M$
  end for
  for $m = 1, 2, \ldots, M$ **do**
    $\mathbf{r}_m^{k+1} \leftarrow \frac{\mathbf{u}_{o,m} + \sum_{neigh:(o,neigh)\in E} p^t_{neigh,m} \mathbf{u}_{neigh,m}}{\|\mathbf{u}_{o,m} + \sum_{neigh:(o,neigh)\in E} p^t_{neigh,m} \mathbf{u}_{neigh,m}\|_2}$
  end for
end while
$\mathbf{y}_o \leftarrow [\mathbf{r}_1, \mathbf{r}_2, \cdots, \mathbf{r}_M]$

two nodes, thus resulting in a sparse graph. The encoder can be modeled as $q_\theta(a_{ij}|\mathbf{x}) = softmax(f_{en,\theta}(\mathbf{x}))$, given the input $\mathbf{x}_1, \ldots, \mathbf{x}_N$. In a data center, there are various factors that contribute to the connections between nodes. Therefore, the encoder utilizes the proposed multi-factor separation module to extract spatial features. Assuming that there are M factors responsible for the connection between two nodes, an edge is denoted as $(o, neigh) \in E$ to indicate the connection between node o and its neighbors $neigh$. Additionally, each node $o \in V$ in the graph G is represented by a feature vector $\mathbf{x}_o \in \mathbb{R}^{d_{in}}$.

The neighbors of a node usually contain abundant information, the majority of graph convolutional networks employ the information from neighbors in order to enhance the representation of the node's features. The fundamental component of the majority of graph convolutional networks is a functional layer denoted by $f(\cdot)$, which yields a representation for a given node by taking into account both the intrinsic attributes of the node as well as those of its surrounding neighbors:

$$\mathbf{y}_o = f(\mathbf{x}_o, \{\mathbf{x}_{neigh} : (o, neigh) \in E\}). \tag{1}$$

The output $\mathbf{y}_o \in \mathbb{R}^{d_{out}}$ represents the feature of node o. To differentiate the characteristics of various latent factors, we aim for $\mathbf{y}_o = [\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_M]$ to be a factor-separation representation consisting of M distinct independent components, where $\mathbf{r}_m \in \mathbb{R}^{\frac{d_{out}}{M}}$ $(1 \le m \le M)$. Each captures the specific characteristics of node o with regards to the corresponding latent factor. For a single node o and its neighbors $\{neigh : (o, neigh) \in E\}$, the feature vector of node o is projected into M different subspaces:

$$\mathbf{u}_{i,m} = \frac{\delta(\mathbf{W}_m^\top \mathbf{x}_i + \mathbf{b}_m)}{\|\delta(\mathbf{W}_m^\top \mathbf{x}_i + \mathbf{b}_m)\|_2}, \tag{2}$$

where $\mathbf{W}_m \in \mathbb{R}^{d_{in} \times \frac{d_{out}}{M}}$ and $\mathbf{b}_m \in \mathbb{R}^{\frac{d_{out}}{M}}$ represent the parameters for factor m to be learned, and δ is the activation function. It is assumed that $\mathbf{u}_{i,m}$ describes the feature of the mth factor of node i. We then use $\mathbf{u}_{i,m}, \forall m = 1, 2, \ldots, M$, to initialize the features of M independent components.

After initializing the features of m components for all nodes, the similarity between each component of their features is calculated for each neighbor node of o. This similarity is then used to perform multi-label classification tasks for all neighbor nodes, where the classes that neighbor nodes belong to indicate which factor will lead to the connection between the two nodes:

$$p_{neigh,m} = \mathbf{u}_{neigh,m}^T \mathbf{o}_m / \epsilon, \forall m = 1, 2, \ldots, M \tag{3}$$

$$p_{neigh,m} = sigmoid(p_{neigh,m}), \forall m = 1, 2, \ldots, M, \tag{4}$$

where ϵ is the parameter that determines the level of probability hardness following similarity calculation. In the data center scenario, there may be multiple categories of connections between nodes. Thus, we utilize the sigmoid function for multi-label classification. For instance, the probability of connection between cabinets could be 0.9 due to spatial proximity or 0.9 due to sharing the same cold aisle, which cannot be achieved by a single classification using the softmax function.

After obtaining the connection probability for each factor, we proceed to identify the largest cluster for each factor. In our approach, we allow each neighbor to belong to multiple clusters across multiple subspaces simultaneously. This means that the neighbors between subspaces will have overlapping parts:

$$\mathbf{r}_m^{k+1} = \frac{\mathbf{u}_{o,m} + \sum_{neigh:(o,neigh)\in E} p^t_{neigh,m} \mathbf{u}_{neigh,m}}{\|\mathbf{u}_{o,m} + \sum_{neigh:(o,neigh)\in E} p^t_{neigh,m} \mathbf{u}_{neigh,m}\|_2}. \tag{5}$$

Eqs. (3)–(5) are iterated for K times to search for the largest cluster. The ultimate output feature is the result of concatenating each of the representations produced by factor-separation: $[\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_M]$.

Next, a graph is sampled from $q_\theta(a_{ij}|\mathbf{x})$. Since the edge variables a_{ij} are discrete, we adopt the Gumbel Softmax strategy [54,55] that samples from a continuous approximation of the discrete distribution, followed by the reparameterization trick. Finally, the factor-separation features form a distribution that can be sampled using the Gumbel Softmax strategy:

$$a_{ij} = softmax(([\mathbf{r}_1, \ldots, \mathbf{r}_M] + [\mathbf{g}_1, \ldots, \mathbf{g}_M])/\lambda), \tag{6}$$

where $\mathbf{g}_m \in \mathbb{R}^{\frac{d_{out}}{M}}$ is the vector samples from the $Gumbel(0, 1)$ distribution and λ is the parameter controlling the smoothness of each sample.

3.3. Decoder: Spatial–temporal prediction

The decoder leverages the historical time series and learned graph structure to forecast future values, wherein the predicted outcomes serve as feedback to assist the relational inference stage in achieving a more accurate graph that captures the operational state of the system. To achieve this, the decoder formulates a conditional distribution as follows:

$$p_\phi(\mathbf{x}^{t+1}, \ldots, \mathbf{x}^{t+T} | \mathbf{x}^1, \ldots, \mathbf{x}^t, s^1, \ldots, s^{t+T}, \mathbf{A}), \tag{7}$$

where $[s^1, \ldots, s^{t+T}]$ is a time-dependent covariate vector that is assumed to be known for all time steps, e.g., hour-of-the-day. The variable a serves as the representation of the acquired graph structure, which the prediction is reliant on. To accomplish this goal, a spatial–temporal architecture has been implemented in the decoder module, as evidenced by the schematic diagram depicted in Fig. 2.

Time series decomposition. In the current study, each time series is subjected to a decomposition process that divides it into trend and seasonal components, which are then processed individually for prediction. The trend component is derived using a moving average kernel, which is also employed in other state-of-the-art models such as Autoformer [56] and FEDformer [57]. Additionally, the seasonal component is obtained by subtracting the trend component from the original time series, resulting in a more nuanced understanding of the underlying patterns at play:

$$\begin{aligned} \mathbf{x}_{tr} &= AvgPool(Padding(\mathbf{x})) \\ \mathbf{x}_s &= \mathbf{x} - \mathbf{x}_{tr}. \end{aligned} \tag{8}$$

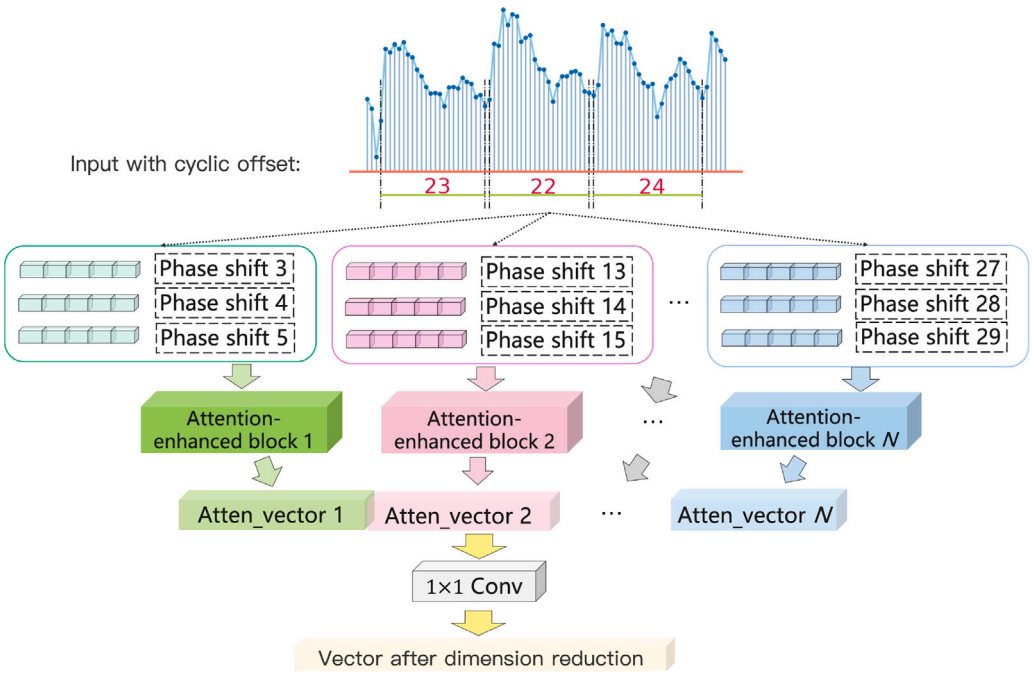


Fig. 3. Periodicity deviation attention.

3.3.1. Attention-enhanced graph-based dilated causal convolution

Following the process of temporal decomposition, the seasonal component is subsequently channeled into the temporal feature extraction module. We utilize the encoder–decoder architecture and compare it with other similar approaches based on their decoder designs. For example, FEDformer employs a frequency-domain attention mechanism in the decoder, while Autoformer introduces an Auto-Correlation module to capture dependencies and aggregate similar patterns between sub-sequences. In contrast to the decoder modules in Transformer-based methods [58] such as Autoformer and FEDformer, we introduce a novel attention-enhanced conditional dilated causal convolution module. The dilated convolution module enhances the receptive field to capture longer sequence features comprehensively, and the attention mechanism can differentiate contributions at different time steps and handle periodic offsets. Furthermore, we incorporate spatial information from graphs into this module.

Graph-based dilated causal convolution. Initially, we introduce the first part of AS-Conv, the graph-based dilated causal convolution. Dilated causal convolution for time series data is first used for speech synthesis, and its distinctive feature is learning long-term dependencies with high computational efficiency. We employ two activation functions, namely sigmoid and tanh, to facilitate the learning process of amplitude, phase, and frequency components of the time series data. This approach demonstrated superior fitting performance on waveforms with certain periodicity. Specifically, sigmoid function contributes to the learning of amplitude, while tanh function is responsible for the learning of phase and frequency. In order to effectively leverage the information contained in learned graph structures, we propose utilizing the multi-factor separated spatial feature representation within the encoder as a global condition:

$$\mathbf{z} = tanh(\mathbf{W}_{f,l} * \mathbf{x} + \mathbf{E}_{f,l}^{\top}\mathbf{g}) \odot \sigma(\mathbf{W}_{h,l} * \mathbf{x} + \mathbf{E}_{h,l}^{\top}\mathbf{g}), \tag{9}$$

where $E_{*,l}$ refers to a linear projection that can be learned, with the vector $E_{*,l}g$ being broadcast over the time dimension. Here we use the residual connection which effectively mitigates the model overfitting. To speed up convergence and enable the training of much deeper models, both residual connections and parameterized skip connections are utilized throughout the network. Fig. 2 displays a residual block of our model, which is stacked multiple times in the network.

Periodicity deviation attention. Although the dilated causal convolution can significantly increase the receptive field, it is unable to distinguish the different impact of the preceding temporal values on predicting the current value. The contribution of the preceding temporal values to predicting the next temporal value may vary, for instance, in predicting the temperature of a cabinet tonight, the temperature of the cabinet last night may be more informative than that of today. Furthermore, to address the issue of periodic offset, feeding the attention module with truncated data of various periods could better capture the features of different periods.

In the context of data centers, both data from environmental monitoring system (EMS) and IT data exhibit their own periodicity. However, certain nodes may not show clear periodicity, and periodicity may sometimes exhibit a deviation of 1–2 time points. To address this issue, we propose a periodic offset attention mechanism, which incorporates input data with artificially shifted time points, such as offset by 3–5 time points, 13–15 time points, 26–28 time points, etc. The time points for offset can be determined based on the observed actual periodic shifts in the time series. These features are concatenated together and reshaped using a 1×1 convolutional layer before being aggregated with the results of the graph-based dilated causal convolution. The framework of this module is shown in Fig. 3.

Causal convolution attention. The self-attention mechanism in the Transformer model [58] has been widely adopted in various natural language processing (NLP) tasks due to its ability to capture long-term dependencies. However, when dealing with time series data, the conventional point-wise attention generated by queries and keys may not be sufficient to accommodate the local patterns exhibited by values at successive time points. For instance, in a data center, the temperature of a single point may appear normal, but when considering continuous points over time, abnormal fluctuations may occur compared to the previous pattern. To address this issue, this study proposes a novel approach that utilizes a $1 \times k$ causal convolution with stride 1 to generate queries and keys, instead of the conventional matrix multiplication approach. This approach enables the generated queries and keys to capture more pattern changes, which is crucial for detecting anomalies in data centers. Specifically, the $1 \times k$ causal convolution is applied only

to the seasonal component, and the attention vector of seasonal features is generated as follows:

$$\begin{aligned}\mathbf{Q}_s &= Conv1d\_Q(\mathbf{x}_s)\\ \mathbf{K}_s &= Conv1d\_K(\mathbf{x}_s)\\ \mathbf{V}_s &= Conv1d\_V(\mathbf{x}_s)\\ Attention(\mathbf{Q}_s,\mathbf{K}_s,\mathbf{V}_s) &= softmax(\frac{\mathbf{Q}_s\mathbf{K}_s^{\top}\cdot\mathbf{MASK}}{\sqrt{d_k}})\mathbf{V}_s,\end{aligned} \tag{10}$$

where $Conv1d\_Q$ and $Conv1d\_K$ have the filter size of $1\times k$ and stride 1 while $Conv1d\_V$ has the filter size of 1×1 and stride 1. The mask matrix **MASK** is used to prevent the use of future information by setting elements of the upper triangular to $-\infty$.

3.3.2. Graph prediction block

In the context of long-term temporal prediction, the generated graph utilized is an unchanging graph learned by the encoder. However, in many prediction scenarios within a data center, the connectivity of the graph may vary. For instance, at the IT power consumption level, two cabinets may have a considerable degree of correlation in the front half of an hour, but no business relationship in the latter half. When using a static graph for long-term temporal prediction, inaccurate predictions may occur towards the end of the period. Therefore, we propose a graph prediction block that assists long-term forecasting and achieves more precise results by predicting future graph structure. In the graph prediction block, we predict the future values of auxiliary variables that affect the target variable to construct the future graph. For instance, in predicting cabinet temperature, we construct the future graphs by utilizing the predicted values of the IT power consumption and the cold aisle temperature respectively, both of which have an impact on the cabinet temperature. The graph learning layer is tailored to extract one-way relationships, as demonstrated below:

$$\mathbf{P}_1 = tanh(\alpha\mathbf{E}_1\Theta_1) \tag{11}$$

$$\mathbf{P}_2 = tanh(\alpha\mathbf{E}_2\Theta_2) \tag{12}$$

$$\mathbf{A}^{pred} = ReLU(tanh(\alpha(\mathbf{P}_1\mathbf{P}_2^{\top} - \mathbf{P}_2\mathbf{P}_1^{\top}))). \tag{13}$$

In our approach, the node embeddings, $\mathbf{E}_1$ and $\mathbf{E}_2$, are initialized randomly and are learnable during training. The model parameters, Θ_1 and Θ_2, are used to train the embeddings. The saturation rate of the activation function is controlled by α. The graph adjacency matrix achieves asymmetry through Eq. (13). The subtraction term and ReLU activation function help regularize the adjacency matrix such that if a_{ij} is positive. To make the adjacency matrix sparse while reducing computation cost for graph convolution, we select the top-k closest nodes as neighbors for each node, retaining the weights for connected nodes, and setting the weights of non-connected nodes to zero.

To handle scenarios where multiple auxiliary variables influence the predictive variable, a separate prediction of each auxiliary variable is required, followed by the construction of an internal graph for each variable. The combination of these graphs is obtained as follows:

$$\mathbf{A}^{future} = r_m\sum_{m=1}^{M}\mathbf{A}_m^{pred} \tag{14}$$

where r_m assigns a weight to each variable based on its importance, $\mathbf{A}_m^{pred}$ is the graph generated from each predicted auxiliary variable, and $\mathbf{A}^{future}$ is the final graph structure generated by the GP module. Notably, the GP module can be executed concurrently with the encoder, thereby incurring no additional time overhead.

The integration of the learned graphs from the graph prediction block and multi-factor separation module is accomplished as follows:

$$a_{ij}^{future} = [1-(\frac{growth\_step}{pred\_len})^{\mu}]a_{ij} + (\frac{growth\_step}{pred\_len})^{\mu}a_{ij}^{pred}, \tag{15}$$

where a_{ij}^{future} is the probability of the relation type in the graph learned by the multi-factor separation module and a_{ij}^{pred} is the probability of the relation type in the graph learned by the graph predicting block. μ represents the rate of increase in the weight of a_{ij}^{pred}. The parameter $growth\_step$ represents the speed at which the coefficient to increase, indicating the number of steps after which growth will occur. $pred\_len$ stands for the total number of predicted steps. As the prediction time steps increase, the GP module gradually becomes active, as shown in Fig. 4. With the increase of prediction steps, the weight of the GP module automatically increases, and the weight is multiplied by a growth coefficient every $growth\_step$ time points.

3.3.3. Spatial–temporal feature fusion

The effective extraction and utilization of spatial features are critical to the training and performance of the model. In order to preserve the spatial information features, we directly pass the feature factors separated by the encoder to the decoder, and further fuse them with the spatial–temporal feature-extracted features before passing them to the next layer. Spatial features are crucial for comprehensive understanding and correct learning of graph structures. This strategy is adopted to encourage the encoder to learn the connections that simulate the system's operational state through feedback from the decoder.

3.3.4. The manner of generating predictions

The generation of predictions can generally be divided into two methods: iterative approach and generative approach. The iterative method obtains predictions through a sequential process, while the generative method directly generates multiple values. The iterative method boasts the advantage of obtaining a lower variance due to its autoregressive nature. However, it may accumulate errors gradually and significantly decrease the prediction speed. On the other hand, when unbiased single-step prediction cannot accurately acquire predictions, the generative approach can generate more accurate predictions and significantly expedite the prediction process. Therefore, it is commonly used for long-term sequence prediction. For generative approach, the decoder inputs the following vector:

$$\mathbf{x}_{de} = [[\mathbf{x}^1,\ldots,\mathbf{x}^t],[\mathbf{x}^{t+1},\ldots,\mathbf{x}^{t+T}]], \tag{16}$$

where $[\mathbf{x}^1,\ldots,\mathbf{x}^t]$ is input historical time series while $[\mathbf{x}^{t+1},\ldots,\mathbf{x}^{t+T}]$ is the placeholder for the target to be predicted. When the prediction time step is less than a certain threshold value γ, an iterative method is used, and when the prediction time step is greater than γ, a generative method is used to generate the predicted values. Since this paper focuses more on long-term prediction, a generative method is employed.

3.3.5. Loss function

Given the input samples $\mathbf{x}$, the encoder outputs the distribution $q_\theta(\mathbf{a}_{ij}|\mathbf{x})$ for each edge. Then a graph $\mathbf{A}$ is sampled from the continuous estimation of the distribution. The decoder inputs the learned graph $\mathbf{A}$ and historical time series, then outputs the predicted value $p_\phi(\hat{\mathbf{x}}|\mathbf{A})$. The loss function consists of two parts:

$$\mathcal{L} = \sum_i\sum_{t=1}^{T}\|\mathbf{x}_i^t - \hat{\mathbf{x}}_i^t\|^2 - \sum_{i\neq j}H(q_\theta(\mathbf{a}_{ij}|\mathbf{x})). \tag{17}$$

The first term is related to the prediction accuracy of the decoder, calculating the gap between the ground-truth values and the predicted values. The second term represents the sum of entropies and the design of the second term is to avoid the nodes between each separated part overlapping too much. Thus the negative value of the sum of entropies of the edge type is utilized to reduce confusion.

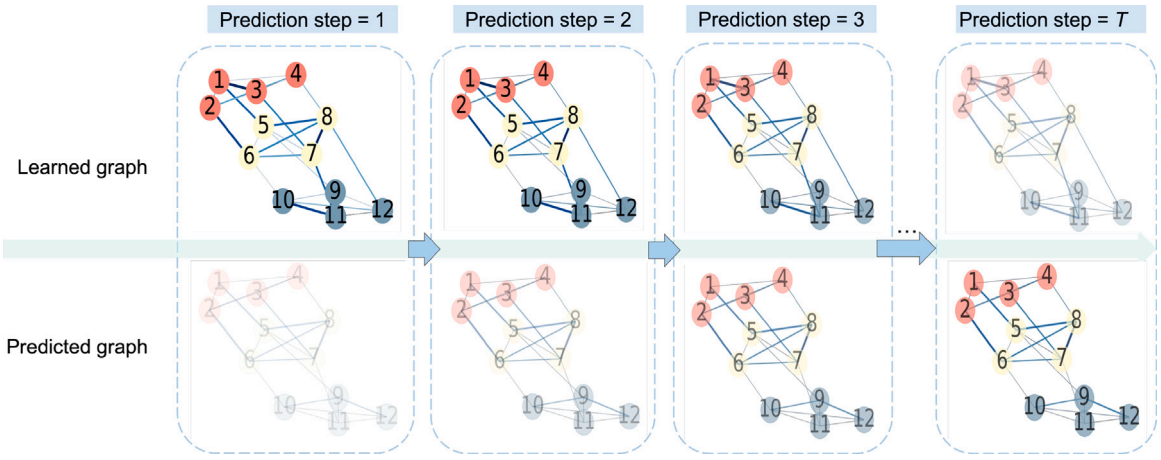


Fig. 4. The schematic diagram of the GP module.

3.4. Requirements and limitations

The proposed method requires multivariate time series prediction with various interdependent relationships between variables. If the target variable is influenced by upstream information, upstream node features must be added to the GP module in order to aid in the learning of more accurate graph structures and features. For time series prediction where there is no obvious relationship between variables, non-graph-based prediction methods may be more suitable. Additionally, in real-world scenarios with a large number of variables, the graph can be partitioned into multiple subgraphs for learning.

3.5. Time and space complexity analysis

For the MSE-STGNN model, the multi-factor separation module requires the most analysis of time and space complexity, while the time and space complexity of other parts of the model can be easily obtained by referring to [59]. In this section, we provide a detailed analysis of the time and space complexity of the proposed multi-factor separation module, which is described in Algorithm 1.

The multi-factor separation module can be divided into two main parts. The first part is feature mapping, consisting of steps 1–5 in Algorithm 1. This part maps the features of M factor types from the original space to M feature spaces. For each node, the time complexity of this part is $O\left(Nd_{out}\ d_{in}\right)$ and the space complexity is $O\left(Nd_{out}\right)$, where N is the number of nodes, d_{in} is the dimensionality of the input feature vector, and d_{out} is the dimensionality of the output feature vector. The second part is obtaining the features of each factor through the Expectation-Maximization (EM) algorithm in steps 7–15 in Algorithm 1, which has a time complexity of $O\left(KNd_{out}\right)$ and a space complexity of $O\left(d_{in}d_{out}\right)$, where K is the number of iterations, N is the number of nodes, and d_{in} and d_{out} are the input and output dimensions of the feature vector, respectively. In particular, this part has been proven in [5] to converge to a point estimate of $\{o\}_{m=1}^{M}$ that maximizes the marginal likelihood $p\left(\mathbf{u}_{i,m} : i = o \cup (o, neigh) \in G, 1 \le m \le M; \{o\}_{m=1}^{M}\right)$.

Taking into account the above analysis, the overall time complexity of the multi-factor separation module is $O\left(Nd_{out}(d_{in}+K)\right)$, and the space complexity is $O\left(d_{out}\left(d_{in}+N\right)\right)$. It can be seen that both the time and space complexity are independent of the number of factors M. Therefore, even in complex systems with a large number of factors, the time and space complexity will not increase.

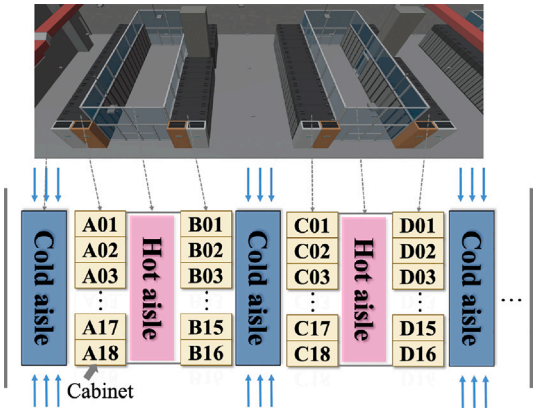


Fig. 5. Cabinets arrangement and locations of cold aisles.

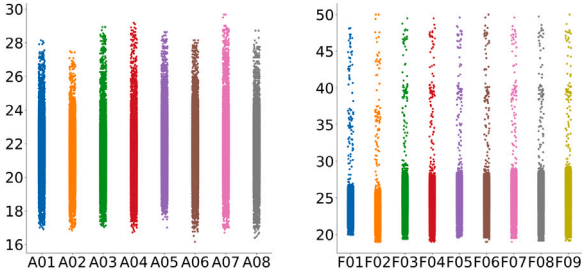


Fig. 6. The distributions of the normal cabinet temperature and the cabinet temperature with alarm records.

4. Experiments

4.1. Experimental settings

4.1.1. Datasets

Here we provide a description of three real-world datasets for the following experiments.

- **DC-Temp-Normal**. This dataset consists of temperature and related metrics from a data center, encompassing 369 cabinets without alarms between January 2022 and January 2023. The dataset includes three key variables: cabinet temperature, cold

aisle temperature, and IT power consumption. Data is sampled at a rate of one data point every 2.5 min. The cabinet temperature reflects the impact of several factors, including the heat generated by the IT workload, the cold air from the cold aisle, and the temperature effects of other cabinets, which are accounted for using the proposed model. Fig. 5 illustrates the relationship between the cabinets and the cold aisle within the data center.

- **DC-Temp-Abnormal**: This dataset includes temperature data from 145 cabinets with temperature alarms that occurred in three rooms of a data center between January 2022 and January 2023. Alarms are triggered when the cabinet temperature exceeded a certain threshold or when a temperature rise is detected. Each cabinet includes three data points: the cabinet temperature, the cold aisle temperature, and the IT power consumption of the cabinet. Data is sampled every 2.5 min per point. Fig. 6 illustrates the distribution of data with and without temperature alarms.
- **DC-Power**: This dataset includes IT power consumption data for 77 cabinets over time in three rooms of a data center between January 2022 and January 2023. Data is sampled every 2.5 min per point.
- **DC-Air-Normal**: This dataset comprises data collected from air conditioning sensors, IT power consumption, and cold aisle temperature from three separate rooms between June 2022 and June 2023. The air conditioning sensor data includes water inlet temperature, water valve opening, return air temperature, fan speed, and supply air temperature. Each room contains 20 air conditioners and 18 cold aisle temperature measurement points. The data in the dataset is sampled every 2.5 min. In this dataset, the simultaneous prediction of the supply air temperature and cold aisle temperature results in the existence of two types of nodes in the graph, resulting in a total of 38 nodes. Other air conditioning sensor data and IT power consumption serve as influencing factors for predicting the supply air temperature and cold aisle temperature.
- **DC-Air-Abnormal**: DC-Air-Abnormal is another dataset that includes data collected from air conditioning sensors, IT power consumption, and cold aisle temperature from three different rooms between June 2022 and June 2023. Any cold aisle temperature measurement point that exceeds a certain threshold or displays a temperature rise is referred to as abnormal. The air conditioning sensor data still includes water inlet temperature, valve opening, return air temperature, fan speed, and supply air temperature. Each room contains 20 air conditioners and 18 cold aisle temperature measurement points. The data in the dataset is sampled every 2.5 min. In this dataset, we predict the future values of cold aisle temperature, with other air conditioning sensor data and IT power consumption used as influencing factors.

  It is worth noting that these datasets are all based on real physical operation systems, with very complex influencing factors. Taking the DC-Air-Abnormal dataset as an example, there are six types of factors that affect the cold aisle temperature, including five types of air conditioning measurement points (supply air temperature, water inlet temperature, valve opening, return air temperature, fan speed) and the IT power consumption. However, within a single room in the dataset, there are 20 measurement points for each air conditioning sensor data type (20×5), and each cold aisle temperature measurement point corresponds to one IT power consumption data point (18 cold aisle temperature measurement points), resulting in a total of 118 influencing factors. Therefore, the influencing factors are numerous and complex.

4.1.2. Data preprocessing, parameter tuning, and overfitting prevention

The data preprocessing procedure involves several steps aimed at ensuring the quality and reliability of the data:

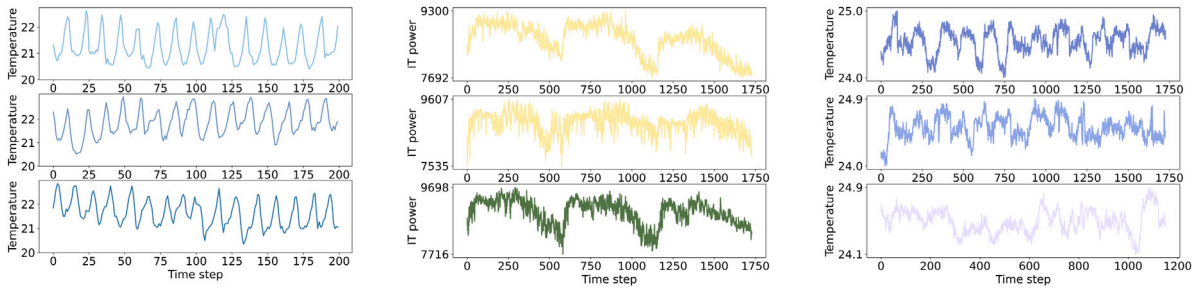
- Eliminating outliers. The first step is identifying and filtering outliers based on industry knowledge and data distribution. Outliers are values that are inconsistent with the expected behavior of the system, such as a rise in cold aisle temperature without a corresponding increase in cabinet temperature or a cold aisle temperature that exceeds a certain threshold.
- Smoothing the data. Exponential smoothing is used to further refine the data and filter out noise. The smoothing constant is set to 0.9, which strikes a balance between filtering out noise and preserving important features of the data.
- Normalizing the data. Finally, the data is normalized using the Gaussian normalization to ensure all features in the dataset have the same scale. This prevents any single feature from dominating the model.

In the following, we will describe the setting of model parameters, including the number of kernels in the dilated causal convolution, the length of the periodic offset window in the attention vector, the number of factors in the multi-factor separation module, and other training parameter settings.

Causal convolution structures. In the realm of time series forecasting, the periodicity of a sequence often remains uncertain. To address the challenge of expanding the receptive field swiftly, many experiments have employed an exponential causal convolution structure featuring lengths of 1, 2, 4, 8, and 16, which constitutes a generalized structured design. Nonetheless, in the context of predicting the cabinet temperature, IT power consumption and cold aisle temperature, a conclusion can be reached by observing the historical cabinet data, as illustrated in Fig. 7. Specifically, the cabinet temperature exhibits a cyclic pattern in the range of 10–14, and a periodicity of length 14 stands out as a robust feature. Correspondingly, the IT power consumption manifests a periodicity of length 48. There is no obvious periodic feature observed in the dataset of cold aisle temperature. For predicting the cold aisle temperature, capturing the influence of upstream features on its variation is crucial to enhance the accuracy of the prediction. Notably, the exponential causal convolution structure solely captures the periodicity of 14 cycles for the cabinet temperature cycle in an implicit manner. In response, we optimize the structure of the model by replacing the exponential causal convolution structure with the causal convolution structures of 1, 2, 3, 4, 10, 12, and 14. By doing so, our model can directly learn the periodicity of both 14 and 576 cycles.

Periodic deviation. The periodic instability of the temperature in the server cabinet sometimes causes a deviation of 1–2 points in a period. Therefore, in addition to the aforementioned improvements, based on the cyclic properties of the data we introduce several additional feature window sets, such as 5–7, 13–15, and 27–29 as the input group of the attention layer. The weight of the attention vector for each group is calculated through a 1d convolution layer. All of the groups of window features are flattened and concatenated, followed by another 1d convolution layer to adjust the feature vector dimension. The adjusted feature vector is then aggregated with the causal convolution to predict the final result.

Overfitting prevention. To avoid overfitting, we implement the following measures: Firstly, we ensure an ample amount of data, with each dataset containing one year's worth of data, totaling 210, 240 time points, enabling the model to absorb the diversity of the data. Secondly, during the training phase, we add dropout layers with a certain probability after each sublayer in each module of the model, randomly setting some neuron outputs to zero to prevent overfitting. Additionally, we apply batch normalization to normalize the inputs of each output layer, reducing the interdependence between neurons in each layer. Furthermore, during training, we employ early stopping technique by monitoring the loss function value on the validation set. If the loss function on the validation set does not decrease within a certain number of epochs, we stop the training process. Lastly, we employ cross-validation by dividing the training set into multiple subsets and



(a) Cabinet temperature (b) IT power consumption (c) Cold aisle temperature

Fig. 7. Periodic changes of the cabinet temperature, IT power consumption and cold aisle temperature.

Table 1
The model parameters and training parameters of the proposed method.

	Parameter	Value
MS module	MS block numbers	2
	Number of hidden units per factor *n_hidden*	48
	Mapping matrix $\mathbf{W}_m^{\top}$	$(input\_dim, M \times n\_hidden)$
	Dropout rate	0.35
	Number of iterations	6
	Size of the sampled neighborhood	10
	Number of graph convolution layers	5
GP module	Rate of increase in the weight μ	0.75
	Growth step	4
AS-Conv	As-Conv block numbers	2
	Number of the graph-based dilated causal convolution layer	2
	Periodicity deviation	(3∼5, 13∼15, 26∼28)
	Causal convolution filter size	$(1 \times 2, 1 \times 3, 1 \times 4, 1 \times 10, 1 \times 12, 1 \times 14)$
Others	Dimension of fully connected layer	2048
	Epochs	25
	Learning rate	0.01
	Extra iterations before early-stopping	8
	Optimizer	Adam

using one subset as the validation set in each iteration. We train the model multiple times, obtaining multiple models, and average the results or conduct voting to obtain the final result.

Due to the fact that the temperature of cabinets is influenced by three distinct factor types, the default number of factor types in the factor separation module for the cabinet temperature datasets has been established as 3. Conversely, the IT power consumption of server cabinets is unaffected by external environmental factors and is solely reliant upon the operating patterns of the business and the model of the server itself. Therefore, within the factor separation module, the default number of factor types has been established as 1 for this specific variable. For the DC-Air-Normal dataset, there are five factors that affect the supply air temperature and cold aisle temperature, and the default number of factor types is set to 5. Similarly, for the DC-Air-Abnormal dataset, there are six factors that affect the cold aisle temperature, and the default number of factor types is set to 6. However, these are just default values, and the optimal number of factor types needs to be obtained through hyperparameter tuning. The model parameters and training parameters are summarized in Table 1. The format of the training samples is predetermined as a four-dimensional array with dimensions of $[32, *, 3, 48]$ and $[32, *, 3, 96]$, where the initial dimension indicates the batch size, followed by the number of nodes, channel, and time steps, respectively. To ensure an unbiased evaluation, the dataset is partitioned into a training subset (70%), a validation subset (15%), and a test subset (15%) utilizing the time series data partitioning approach. In order to determine the efficacy of the model, both Mean Squared Error (MSE) and Mean Absolute Error (MAE) have been employed as evaluation metrics. The experiments are conducted using PyTorch [60], repeated three times, on four NVIDIA Tesla V100 32 GB GPUs.

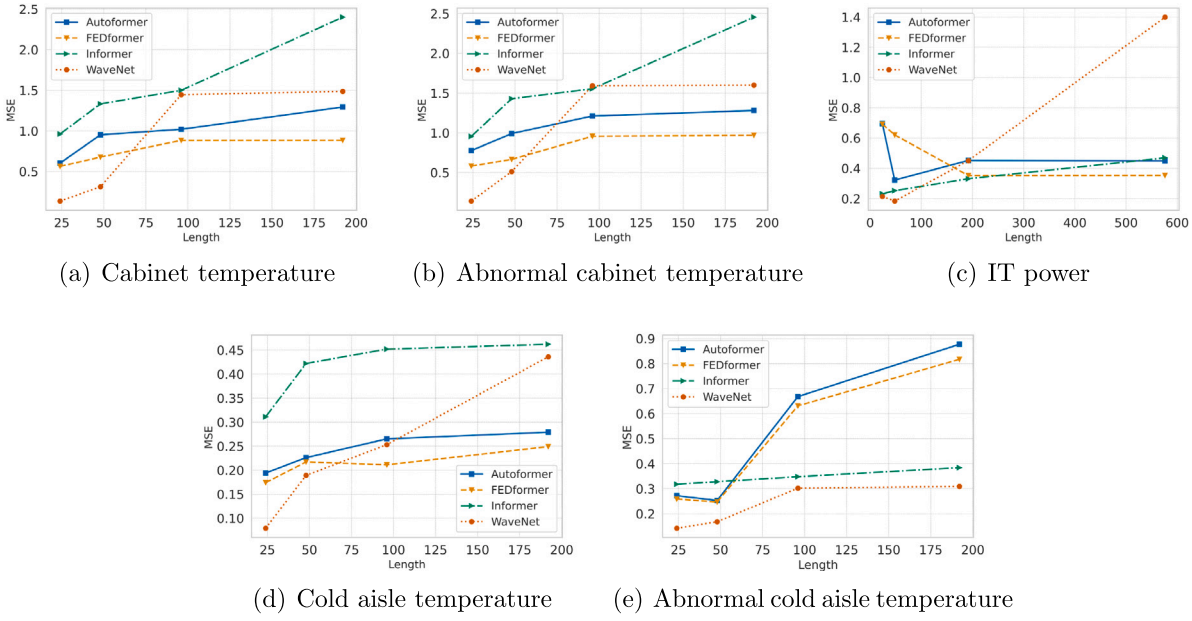
4.2. Inspired experiments

In this study, we conduct an empirical comparison between advanced Transformer-based methods, namely Informer, Autoformer, and FEDformer, and WaveNet [61] in three datasets. WaveNet is chosen due to its demonstrated superior performance in time series prediction tasks. In addition to comparing with the original methods, we also evaluate the performance of these methods when combined with graphs. Specifically, we adopt an encoder–decoder architecture, where we learn the graph structure in the encoder and embed the original method for spatial–temporal prediction in the decoder. The comparisons are focused on the MSE and MAE at different prediction horizons. The results, presented in Table 2, lead to several conclusions.

Firstly, the incorporation of graph structures in short-term prediction can significantly reduce prediction errors compared to algorithms without graph structures. This finding suggests that graph structures contain rich information in the data center scenario and can facilitate better multivariate time series prediction. Secondly, the introduction of graph structures increases prediction errors as the prediction horizon increases. This indicates that graph structures only act on short-term prediction and are unable to support long-term prediction, likely due to the dynamic changes that may occur in the graph structures over time. Thirdly, WaveNet achieves the lowest prediction errors in short-term prediction, while FEDformer performs the best in long-term prediction. This suggests that the structure of WaveNet helps capture short-term time series features, while FEDformer's frequency-enhanced attention mechanism can better capture longer-term features. Lastly, as the prediction horizon increases, we observe that the prediction error of the cabinet and the cold aisle temperature dataset shows an upward trend while the prediction error of the cabinet IT power consumption dataset decreases or remains relatively stable, except for WaveNet which shows

Table 2
Empirical study on the inspiration sources: a comparative analysis of predictive performance between Transformer-based methods and WaveNet, as well as their combination with graphs.

Methods	Metric	DC-Temp-Normal				DC-Temp-Abnormal				DC-power				DC-Air-Normal				DC-Air-Abnormal			
		24	48	96	192	24	48	96	192	24	48	192	576	24	48	96	192	24	48	96	192
Autoformer with Graph	MSE	0.602	0.951	1.019	1.293	0.776	0.990	1.211	1.281	0.695	0.322	0.452	0.449	0.194	0.226	0.265	0.279	0.272	0.253	0.668	0.877
	MAE	0.621	0.771	0.892	0.966	0.658	0.885	0.902	0.989	0.605	0.401	0.498	0.552	0.342	0.398	0.439	0.499	0.338	0.422	0.678	0.761
Autoformer	MSE	0.654	0.954	1.018	1.263	0.782	0.992	1.209	1.278	0.712	0.341	0.378	0.423	0.203	0.248	0.277	0.315	0.279	0.256	0.675	0.885
	MAE	0.625	0.794	0.855	0.957	0.705	0.896	0.890	0.988	0.668	0.465	0.485	0.527	0.360	0.414	0.456	0.511	0.343	0.431	0.683	0.774
FEDformer with Graph	MSE	0.562	0.676	**0.883**	**0.884**	0.581	0.665	0.957	0.969	0.692	0.621	0.352	0.353	0.174	0.217	**0.211**	**0.249**	0.182	0.192	**0.246**	**0.311**
	MAE	0.575	0.643	**0.762**	**0.798**	0.591	0.627	0.804	0.826	0.609	0.580	0.458	0.452	0.328	0.369	**0.354**	**0.428**	0.246	0.254	**0.336**	**0.458**
FEDformer	MSE	0.567	0.681	0.887	0.892	0.589	0.689	**0.902**	**0.862**	0.704	0.625	**0.327**	**0.323**	0.190	0.223	0.226	0.272	0.219	0.226	0.263	0.322
	MAE	0.588	0.654	0.776	0.761	0.601	0.659	**0.792**	**0.776**	0.613	0.582	**0.444**	**0.449**	0.347	0.395	0.392	0.467	0.330	0.359	0.398	0.487
Informer with Graph	MSE	0.961	1.332	1.499	2.400	0.954	1.429	1.554	2.455	0.231	0.252	0.331	0.469	0.311	0.422	0.452	0.462	0.318	0.328	0.348	0.384
	MAE	0.822	0.974	1.114	1.517	0.826	0.956	1.254	1.368	0.352	0.350	0.455	0.554	0.455	0.554	0.527	0.539	0.458	0.519	0.576	0.451
Informer	MSE	0.970	1.334	1.496	2.384	0.979	1.431	1.500	2.355	0.233	0.259	0.329	0.460	0.320	0.431	0.500	0.432	0.329	0.334	0.353	0.392
	MAE	0.838	0.988	1.051	1.307	0.845	0.995	1.157	1.329	0.371	0.393	0.423	0.528	0.465	0.567	0.602	0.564	0.474	0.531	0.591	0.488
WaveNet with Graph	MSE	**0.134**	**0.311**	1.445	1.486	**0.140**	**0.511**	1.592	1.599	**0.214**	**0.183**	0.451	1.399	**0.079**	**0.189**	0.253	0.336	**0.142**	**0.168**	0.302	0.309
	MAE	**0.305**	**0.629**	0.981	0.987	**0.314**	**0.608**	0.988	0.923	**0.450**	**0.366**	0.624	1.658	**0.292**	**0.348**	0.322	0.491	**0.243**	**0.232**	0.464	0.471
WaveNet	MSE	0.138	0.500	1.423	1.478	0.142	0.523	1.522	1.183	0.234	0.188	0.447	1.369	0.180	0.234	0.288	0.335	0.151	0.179	0.455	0.676
	MAE	0.324	0.617	0.911	0.967	0.329	0.627	0.932	0.899	0.489	0.385	0.595	1.644	0.346	0.429	0.373	0.453	0.299	0.293	0.574	0.760



(a) Cabinet temperature (b) Abnormal cabinet temperature (c) IT power

(d) Cold aisle temperature (e) Abnormal cold aisle temperature

Fig. 8. The trend of the MSE as the prediction horizon increases.

an increase in MSE, as shown in Fig. 8. This suggests that cabinet temperature exhibits short-period fluctuations while cabinet power consumption displays long-period fluctuations.

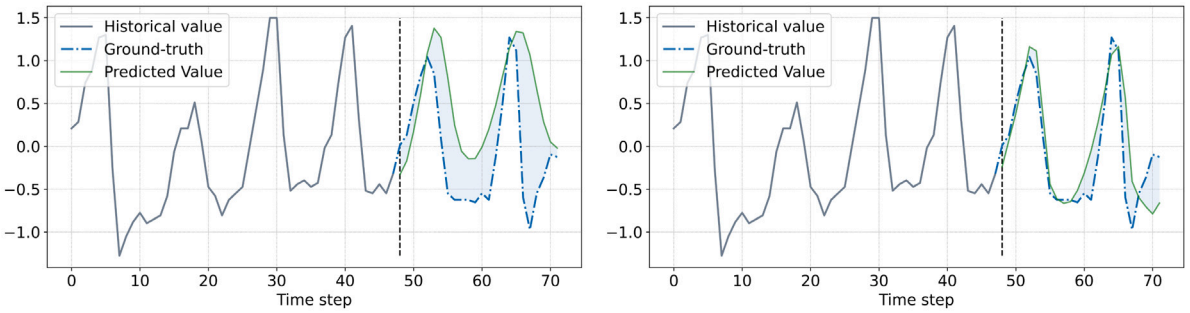
Fig. 9 showcases a comparative analysis between the sophisticated time-series prediction algorithm, FEDformer, and the causal convolution model WaveNet for short-term forecasting. The presented result unequivocally demonstrates that the causal convolution model closely approximates the actual curve, thus attesting to its efficacy in short-term prediction. In contrast, WaveNet, which relies on two activation functions, sigmoid and tanh, respectively, is better suited for fitting periodic signals. This is because the sigmoid function enables the framework to learn the oscillation amplitude of the signal, while the tanh function facilitates the learning of phase and frequency. The skip connections module in WaveNet endows the original data with the ability to influence the prediction results, thus enabling the learning of additional information when the periodicity is not apparent and cannot be learned by the convolution layers. The residual connections module enhances the algorithm's ability to train effectively on large datasets, thereby circumventing potential issues of gradient vanishing or exploding.

Based on these findings, we identify several shortcomings of existing models. Graph-based prediction cannot support long-term prediction, while WaveNet struggles with long-term prediction despite performing well in short-term prediction. To address these issues, we propose a GP module and an AS-Conv module.

4.3. Main results

To authenticate the effectiveness of the proposed model, a comprehensive evaluation is conducted by comparing its prediction performance against that of other state-of-the-art algorithms, encompassing both graph-based and non-graph-based methodologies. The comparative algorithms employed in this paper are as follows:

- ARIMA: The auto-regressive integrated moving average [62]
- TPA-LSTM: The temporal pattern attention long short-term memory network [32]
- Autoformer: A decomposition architecture with an auto-correlation mechanism based on Transformer [56]
- Informer: An efficient transformer-based model for long sequence time-series forecasting [63]
- FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting [57]
- WaveNet: A generative model based on the PixelCNN [61,64]
- MTGNN: Multivariate time series forecasting with graph neural networks [15]



(a) FEDformer (b) WaveNet

Fig. 9. Comparative analysis of predicted curves by FEDformer and the causal convolution model WaveNet for short-term forecasting.

Table 3
The comparative evaluation of MSE-STGNN's predictive capability vis-à-vis baseline models with respect to MSE and MAE metrics.

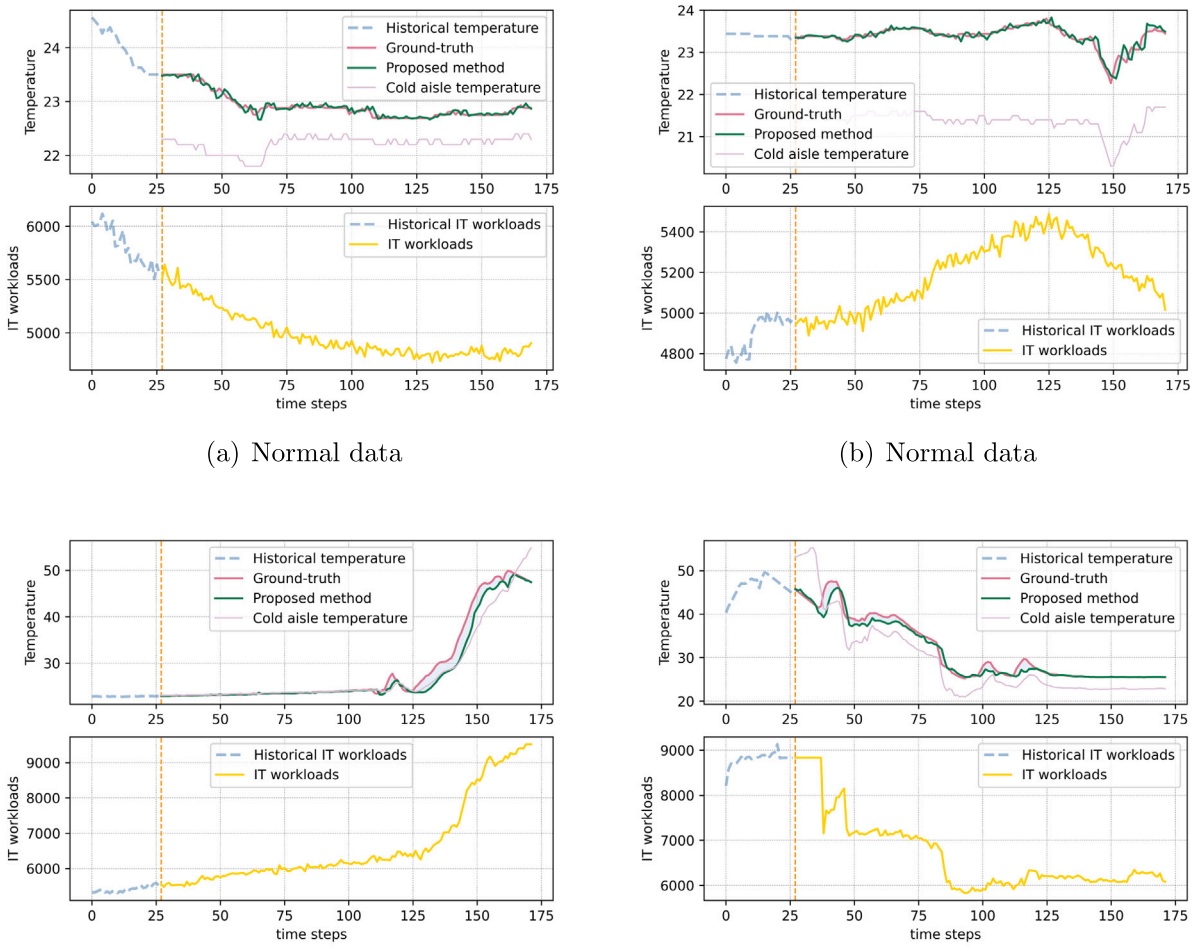
Methods	Metric	DC-Temp-Normal				DC-Temp-Abnormal				DC-power				DC-Air-Normal				DC-Air-Abnormal			
		24	48	96	192	24	48	96	192	24	48	192	576	24	48	96	192	24	48	96	192
ARIMA	MSE	0.854	1.125	1.499	1.277	0.810	1.446	1.296	2.481	0.688	0.359	0.486	1.371	0.387	0.385	0.467	0.554	0.392	0.495	0.526	0.802
	MAE	0.772	1.022	1.067	1.057	0.804	1.028	0.953	1.423	0.547	0.424	0.558	1.651	0.438	0.433	0.517	0.603	0.449	0.559	0.697	0.962
TPA-LSTM	MSE	0.368	0.648	1.446	1.479	0.993	1.025	1.216	2.357	0.761	0.458	0.451	1.296	0.264	0.370	0.450	0.395	0.281	0.292	0.625	0.798
	MAE	0.559	0.704	0.958	0.972	0.869	0.890	0.919	1.331	0.699	0.497	0.605	1.503	0.420	0.524	0.568	0.538	0.352	0.442	0.646	0.812
Autoformer	MSE	0.654	0.954	1.018	1.263	0.782	0.992	1.209	1.278	0.712	0.341	0.378	0.423	0.203	0.248	0.277	0.315	0.279	0.256	0.675	0.885
	MAE	0.625	0.794	0.855	0.957	0.705	0.896	0.890	0.988	0.668	0.465	0.485	0.527	0.360	0.414	0.456	0.511	0.343	0.431	0.683	0.774
Informer	MSE	0.970	1.334	1.496	2.384	0.979	1.431	1.500	2.355	0.239	0.259	0.297	0.460	0.320	0.431	0.500	0.432	0.329	0.334	0.353	0.392
	MAE	0.838	0.988	1.051	1.307	0.845	0.995	1.157	1.329	0.371	0.393	0.423	0.528	0.465	0.567	0.602	0.564	0.474	0.531	0.591	0.488
FEDformer	MSE	0.567	0.681	0.887	**0.852**	0.589	0.689	0.902	0.862	0.704	0.625	0.327	0.323	0.190	0.223	0.226	0.272	0.151	0.179	0.263	0.322
	MAE	0.588	0.654	0.776	**0.761**	0.601	0.659	0.792	0.776	0.613	0.582	0.444	0.449	0.347	0.395	0.392	0.467	0.299	0.293	0.398	0.487
WaveNet	MSE	0.138	0.511	1.423	1.478	0.152	0.523	1.522	1.183	0.234	0.188	0.447	1.369	0.180	0.234	0.288	0.335	0.219	0.226	0.455	0.676
	MAE	0.324	0.622	0.911	0.967	0.339	0.627	0.932	0.899	0.450	0.385	0.595	1.644	0.346	0.429	0.373	0.453	0.330	0.359	0.574	0.760
MTGNN	MSE	0.643	0.952	1.020	1.281	0.775	0.994	1.213	1.277	0.693	0.319	0.453	0.448	0.256	0.283	0.239	0.276	0.243	0.278	0.387	0.417
	MAE	0.630	0.774	0.901	0.971	0.652	0.889	0.912	0.974	0.598	0.398	0.503	0.549	0.360	0.425	0.395	0.417	0.375	0.429	0.523	0.546
StemGNN	MSE	0.959	1.329	1.495	2.398	0.955	1.430	1.556	2.459	0.233	0.258	0.334	0.356	0.260	0.292	0.250	0.285	0.310	0.344	0.316	0.394
	MAE	0.820	0.971	1.112	1.512	0.831	0.959	1.258	1.369	0.359	0.360	0.458	0.458	0.362	0.430	0.407	0.425	0.511	0.580	0.494	0.485
GTS	MSE	0.564	0.671	0.901	0.878	0.579	0.664	0.956	0.902	0.691	0.628	0.355	0.363	0.147	0.203	0.192	0.169	0.141	0.170	0.260	0.317
	MAE	0.577	0.635	0.889	0.805	0.591	0.631	0.809	0.833	0.609	0.581	0.471	0.475	0.309	0.397	0.357	0.325	0.251	0.281	0.391	0.481
NRI	MSE	0.561	0.673	0.899	0.871	0.578	0.662	0.954	0.894	0.689	0.619	0.349	0.352	0.104	0.156	0.168	0.153	0.149	0.160	0.237	0.278
	MAE	0.573	0.639	0.885	0.795	0.588	0.624	0.801	0.821	0.604	0.572	0.452	0.451	0.255	0.344	0.331	0.308	0.296	0.311	0.389	0.429
I2A-RI	MSE	0.562	0.676	0.882	0.869	0.566	0.635	0.951	0.895	0.692	0.628	0.345	0.353	0.081	0.124	0.151	0.141	0.091	0.096	0.108	0.113
	MAE	0.301	0.617	0.761	0.765	0.551	0.623	0.793	0.826	0.625	0.583	0.449	0.454	0.222	0.305	0.312	0.294	0.253	0.261	0.271	0.278
Our method	MSE	**0.131**	**0.500**	**0.878**	0.853	**0.137**	**0.510**	**0.889**	**0.849**	**0.227**	**0.179**	**0.283**	**0.301**	**0.051**	**0.086**	**0.149**	**0.117**	**0.073**	**0.088**	**0.095**	**0.106**
	MAE	**0.301**	**0.617**	**0.761**	0.765	**0.319**	**0.604**	**0.772**	**0.754**	**0.326**	**0.359**	**0.394**	**0.418**	**0.182**	**0.242**	**0.327**	**0.287**	**0.232**	**0.246**	**0.260**	**0.267**

- StemGNN: Spectral temporal graph neural network [16]
- GTS: Graph for time series [38]
- NRI: Neural relational inference [36]
- I2A-RI: Inter-and-intra domain attention relational inference [65]

To facilitate a more rigorous comparison of predictive performance, we have established fixed input sequence lengths of 48 for the temperature dataset and 96 for the IT power consumption dataset. These fixed sequence lengths are utilized both during the training and evaluation stages of our model. Furthermore, the predicted sequence lengths are also fixed at 24, 48, 96, and 192 for temperature datasets, and at 24, 48, 192, and 576 for the IT power consumption dataset. Here, we set a longer prediction horizon for the IT power consumption dataset due to the observation of longer periods exhibited by the dataset. In the design of the prediction horizon, we take both short-term and long-term predictions into account, demonstrating the versatility of our proposed method. To ensure a fair comparison, the input format of other methods is kept consistent with the proposed method, and hyperparameters are set based on the best parameters reported in the paper or selected using a hyperparameter search for optimal performance. The parameter settings of the comparison methods are summarized in the Section A of the supplementary material.

The comparative results between our proposed approach and state-of-the-art methods are presented in Table 3. Upon observation of the table, it is evident that MSE-STGNN consistently achieves the best predictive performance across various prediction horizons on each dataset. Overall, on the DC-Temp-Normal dataset, MSE-STGNN reduces the MSE by 21% compared to the second-best graph-based time series prediction method (I2A-RI). On the DC-Temp-Abnormal dataset, MSE-STGNN reduces the MSE by 23%. On the DC-Power dataset, MSE-STGNN reduces the MSE by 51%. On the DC-Air-Normal dataset, MSE-STGNN reduces the MSE by 19% and on the DC-Air-Abnormal dataset, MSE-STGNN reduces the MSE by 11%. Notably, on the challenging DC-Temp-Abnormal dataset with unstable cycles, MSE-STGNN outperforms the second-best graph prediction method (I2A-RI) by 76% (0.566→ 0.137) in the input-48-predict-24 setting, by 19% (0.635→ 0.510) in the input-48-predict-48 setting, by 6% (0.951→ 0.889) in the input-48-predict-96 setting, and by 5% (0.895→ 0.849) in the input-48-predict-192 setting. The results of this study demonstrate that the MSE-STGNN algorithm consistently attains maximum predictive proficiency for both short-term and long-term temperature forecasts in data center operations. As temperature monitoring is a pivotal task for intelligent data center management, MSE-STGNN can effectively identify early indications of anomalous temperature patterns and provide more accurate predictions of temperature trends once such patterns have already emerged. Our findings thus offer valuable insights into the potential deployment of the MSE-STGNN algorithm as a resilient and dependable predictive instrument within the realm of data center intelligent maintenance and operation.

Furthermore, we present a comparative analysis in Figs. 10 and 11 between the predicted curve and the ground-truth curve. In cabinet temperature prediction, we highlight three distinct prediction scenarios, namely the normal case, data prior to an alarm, and data post an alarm. It can be observed that the predicted curve of MSE-STGNN manifests an exceptional proximity to the ground-truth curve in both normal



(a) Normal data

(b) Normal data

(c) Before the alarm record

(d) After the alarm record

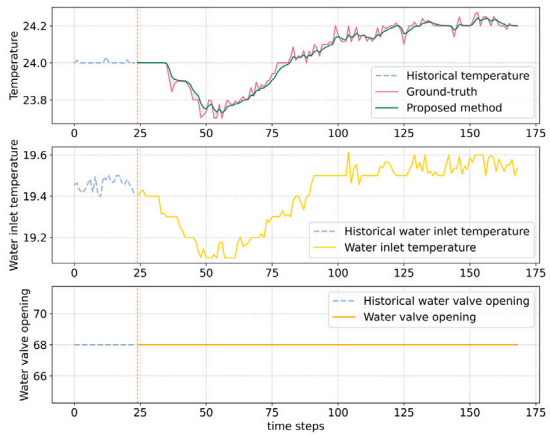
Fig. 10. Comparison of the predicted values and the ground-truth values for the proposed method in the cabinet temperature prediction.

and abnormal conditions, thereby validating that MSE-STGNN effectively leverages the temperature-affecting characteristics. Moreover, it adeptly captures and assimilates the influences of the IT power consumption and cold aisle temperature, ultimately utilizing the acquired relationships to efficiently predict future trends. In the prediction of cold aisle temperature, we present prediction curves for two scenarios: normal and abnormal cold aisle temperatures. From Fig. 11, it can be observed that the predicted curves closely match the true curves. The cold aisle temperature is mainly influenced by the upstream air conditioning water inlet temperature and water valve opening. To provide a clearer explanation, we only show the effects of these two key factors. Under normal conditions, the water valve opening typically remains constant, and it can be seen that the predicted trend of the cold aisle temperature is consistent with that of the air conditioning water inlet temperature. This indicates that the model effectively utilizes the water inlet temperature information during the prediction process. Under anomalous conditions, the water valve opening changes, and the figure shows that the predicted sudden change point trend is consistent with that of the water inlet temperature and water valve opening sudden change points. This indicates that the model is capable of capturing the influence of multiple key factors on the prediction.

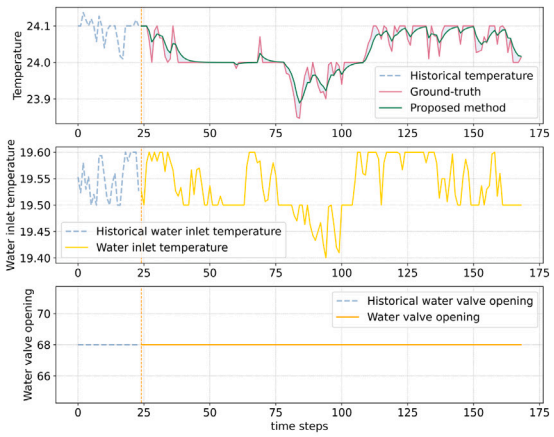
4.4. Ablation study

In this section, we present our ablation experiments to evaluate the effectiveness of the proposed modules in improving the predictive performance of the proposed method. We add modifications to the most primitive NRI in three areas: the graph learning module (GL) in the encoder, the prediction module in the decoder, and the proposed GP module. The GL module is divided into the Multi-Factor Separation (MS) module and the Non-Multi-Factor Separation (MS) module. The prediction module includes three types: Gated Recurrent Unit (GRU), FEB+FEA (the same as the decoder of FEDformer), and AS-Conv. The GP is divided into two cases, with and without this module. We report the results using datasets for the cabinet temperature prediction and the cold aisle temperature prediction, which have more factors to consider. MSE-STGNN has four variants: (1) MSE-STGNN V1: Only using the MS module; (2) MSE-STGNN V2: Only using the AS-Conv module; (3) MSE-STGNN V3: Using both the MS module and the AS-Conv module, and (4) MSE-STGNN V4: Simultaneously using the proposed MS, AS-Conv, and GP modules.

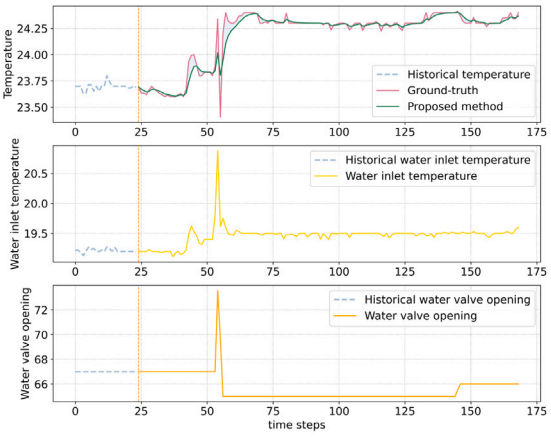
Table 4 summarizes the results of our ablation experiments. As a baseline, we use the original NRI method and compare it with the other methods, with it being placed first in the comparison. If the results of the other methods are better than the baseline, the results are highlighted. From the table, we see that both NRI with GP and NRI+FEDformer with GP show improvement in 16/16 cases. This indicates that both NRI and NRI+FEDformer methods improve prediction accuracy after adding GP modules, suggesting that the model can better predict future changes after adding GP modules. MSE-STGNN V1 shows predictive performance improvement in 13/16 cases,



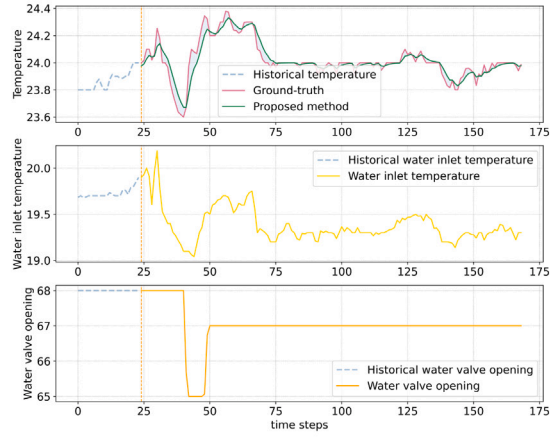
(a) Normal data



(b) Normal data



(c) Abnormal data



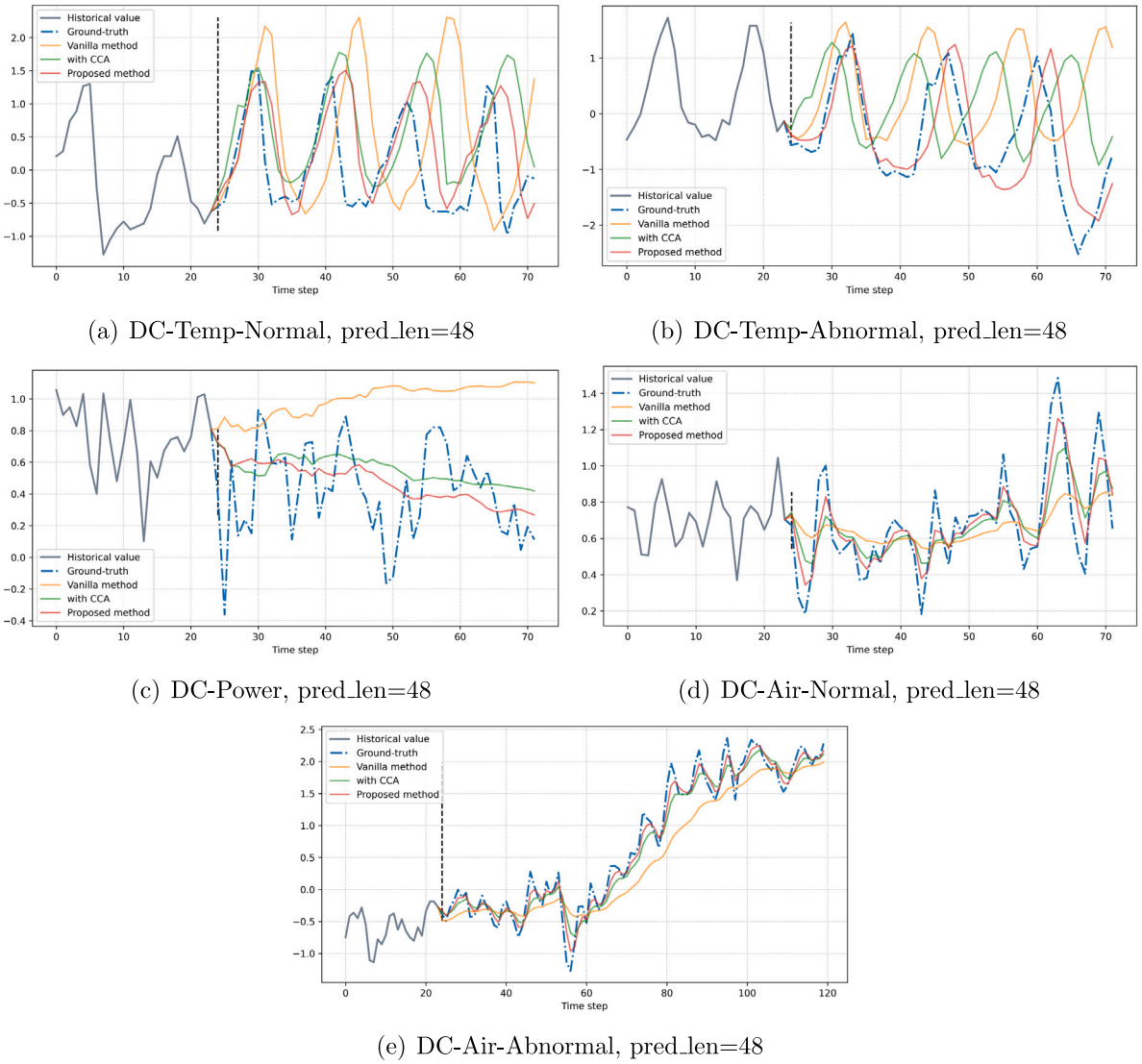
(d) Abnormal data

Fig. 11. Comparison of the predicted values and the ground-truth values for the proposed method in the cold aisle temperature prediction.

Table 4
Ablation studies: Seven variants of NRI are compared with baselines. The results are highlighted if they outperform the baseline.

Methods	GL	Pred	GP		DC-Temp-Normal				DC-Temp-Abnormal				DC-Air-Normal				DC-Air-Abnormal			
					24	48	96	192	24	48	96	192	24	48	96	192	24	48	96	192
NRI	~~MS~~	GRU	~~GP~~	MSE	0.561	0.673	0.899	0.871	0.578	0.662	0.954	0.894	0.104	0.156	0.168	0.153	0.149	0.160	0.237	0.278
				MAE	0.573	0.639	0.885	0.795	0.588	0.624	0.801	0.821	0.255	0.344	0.331	0.308	0.296	0.311	0.389	0.429
NRI	~~MS~~	GRU	GP	MSE	**0.560**	**0.672**	**0.883**	**0.861**	**0.577**	**0.639**	**0.937**	**0.876**	**0.099**	**0.152**	**0.163**	**0.149**	**0.145**	**0.153**	**0.231**	**0.269**
				MAE	**0.572**	**0.633**	**0.879**	**0.782**	**0.574**	**0.604**	**0.772**	**0.803**	**0.252**	**0.339**	**0.329**	**0.301**	**0.291**	**0.302**	**0.381**	**0.411**
NRI+FEDformer	~~MS~~	FEB+FEA	~~GP~~	MSE	0.562	0.676	**0.883**	0.874	0.581	0.665	0.957	0.899	**0.103**	**0.155**	0.169	0.156	0.151	0.163	**0.227**	**0.258**
				MAE	0.575	0.643	**0.762**	0.798	0.591	0.627	0.804	0.826	**0.250**	**0.341**	0.334	0.309	0.298	0.316	**0.378**	**0.395**
NRI+FEDformer	~~MS~~	FEB+FEA	GP	MSE	**0.559**	**0.672**	**0.879**	**0.866**	**0.576**	**0.639**	**0.917**	**0.871**	**0.091**	**0.148**	**0.158**	**0.144**	**0.123**	**0.150**	**0.198**	**0.213**
				MAE	**0.570**	**0.633**	**0.759**	**0.772**	**0.583**	**0.602**	**0.785**	**0.803**	**0.249**	**0.342**	**0.321**	**0.299**	**0.273**	**0.296**	**0.311**	**0.322**
MSE-STGNN V1	MS	FEB+FEA	~~GP~~	MSE	**0.560**	0.673	**0.880**	0.872	0.579	**0.659**	**0.944**	**0.884**	**0.089**	**0.145**	**0.153**	**0.141**	**0.118**	**0.148**	**0.195**	**0.201**
				MAE	**0.572**	0.639	**0.761**	0.789	0.588	**0.619**	**0.792**	**0.813**	**0.246**	**0.339**	**0.319**	**0.295**	**0.269**	**0.292**	**0.302**	**0.313**
MSE-STGNN V2	~~MS~~	AS-Conv	GP	MSE	**0.134**	**0.511**	**0.879**	0.872	**0.140**	**0.511**	0.955	0.897	0.102	**0.122**	**0.150**	**0.140**	**0.089**	**0.094**	**0.106**	**0.109**
				MAE	**0.305**	**0.629**	**0.759**	0.794	**0.314**	**0.608**	0.806	0.823	0.252	**0.298**	**0.314**	**0.291**	**0.251**	**0.259**	**0.268**	**0.275**
MSE-STGNN V3	MS	AS-Conv	~~GP~~	MSE	**0.134**	**0.511**	**0.879**	**0.869**	**0.141**	**0.511**	**0.913**	**0.870**	**0.054**	**0.089**	**0.158**	**0.121**	**0.079**	**0.092**	**0.102**	**0.108**
				MAE	**0.305**	**0.629**	**0.759**	**0.791**	**0.316**	**0.608**	**0.783**	**0.803**	**0.188**	**0.246**	**0.338**	**0.293**	**0.239**	**0.251**	**0.269**	**0.271**
MSE-STGNN V4	MS	AS-Conv	GP	MSE	**0.131**	**0.500**	**0.878**	**0.853**	**0.130**	**0.510**	**0.889**	**0.849**	**0.051**	**0.086**	**0.149**	**0.117**	**0.073**	**0.088**	**0.095**	**0.106**
				MAE	**0.301**	**0.617**	**0.745**	**0.765**	**0.319**	**0.617**	**0.761**	**0.754**	**0.182**	**0.242**	**0.327**	**0.287**	**0.232**	**0.246**	**0.260**	**0.267**

indicating that without the MS module, it is more difficult to extract better information advantageous for prediction from the graph. MSE-STGNN V3, with the addition of the MS module, shows prediction performance improvement in 16/16 cases, although the improvement is not significant. However, the MS module not only improves prediction performance but also improves model interpretability. MSE-STGNN V4 also shows prediction performance improvement in 16/16 cases, and the improvement is more significant, indicating that the GP module can



(a) DC-Temp-Normal, pred_len=48

(b) DC-Temp-Abnormal, pred_len=48

(c) DC-Power, pred_len=48

(d) DC-Air-Normal, pred_len=48

(e) DC-Air-Abnormal, pred_len=48

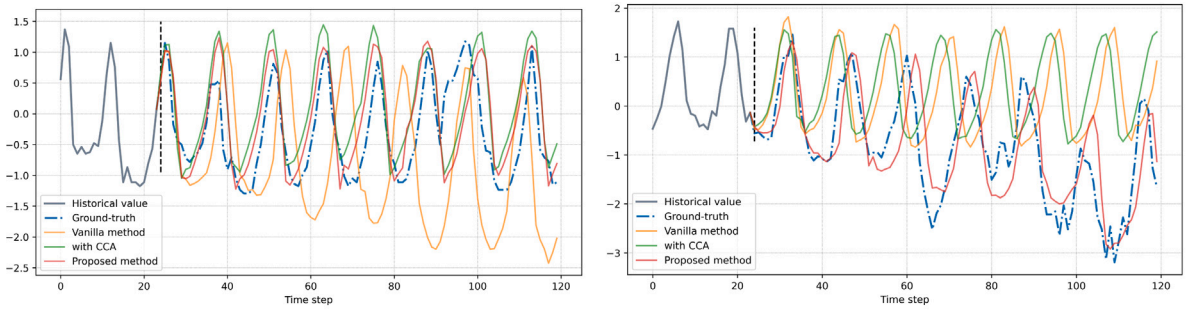
Fig. 12. Ablation study: Prediction curves of different modules with a prediction length of 48.

effectively improve prediction performance. The results also show that the GP module is more helpful for long-term prediction than short-term prediction, which further reflects the dynamic nature of the graph in the scenario of the data center intelligent operation and maintenance. If long-term prediction is required, it is indeed necessary to estimate the future changes of the graph first. Overall, the three proposed modules have practical value in data center scenarios for dynamic graph-based time-series prediction and model interpretability.

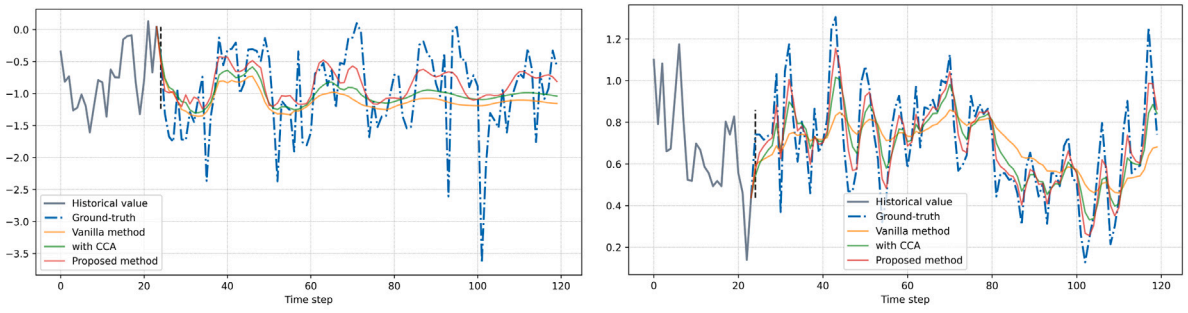
We also compare the prediction performances of the original NRI method, the NRI method with the causal convolution with attention (CCA) module, and the proposed method with three modules. We present the comparison using various prediction curves, as depicted in Figs. 12 and 13. The results for different prediction horizons of three datasets are shown in the figures, where only the predictions with a horizon of 48 and 96 could be clearly displayed. It is observed that the proposed method demonstrates the closest fit to the true curves across all three datasets. It is worth noting that while the predictions for the DC-Power dataset appeared relatively smoother compared to the true curve, our proposed method outperforms the other methods in capturing the trends and fluctuations in the data.

4.5. The selection of the number of factor types

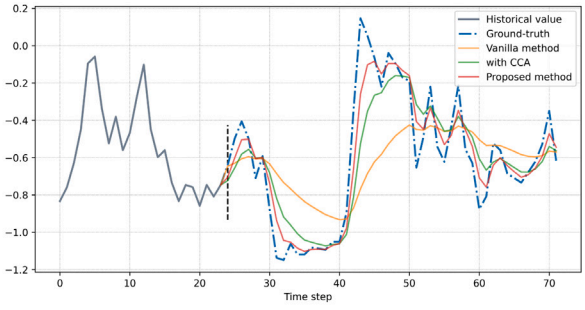
In order to validate the efficacy and interpretability of the MS module in capturing real-world scenarios, we conduct a thorough analysis of the MSE for each factorization configuration ranging from 1 to 6 factor types. The results are presented in Fig. 14. Our findings reveal that the optimal number of factor types for the DC-Temp-Normal and DC-Temp-Abnormal datasets is $M = 3$, as the cabinet temperature is influenced by the heat generated by IT power consumption, the cold air generated by the cold aisle temperature, and the ambient temperature of the surrounding cabinets, which aligns with the three-factor influence model established by the MS module. Similarly, for the DC-Power dataset, $M = 2$ emerges as the optimal configuration, indicating that IT power consumption is not only affected by its own factor types, but also by other variables such as server models, providing us with valuable insights for more accurate predictions of future IT power consumption. Regarding the DC-Air-Normal and DC-Air-Abnormal datasets, the optimal number of factor types is $M = 3$. However, we input more than three factor types into the model, specifically five and six factor types, which indicates that the model is capable of automatically selecting the most useful factor types for prediction. Importantly, our results demonstrated that when $M = 1$, the proposed method degrades to a



(a) DC-Temp-Normal, pred_len=96 (b) DC-Temp-Abnormal, pred_len=96

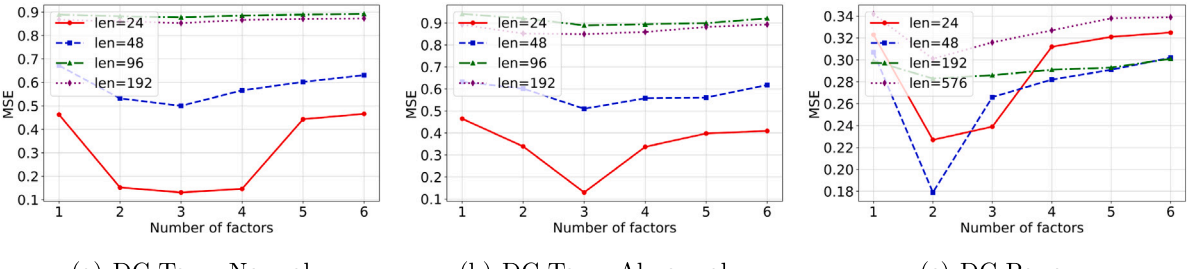


(c) DC-Power, pred_len=96 (d) DC-Air-Normal, pred_len=96

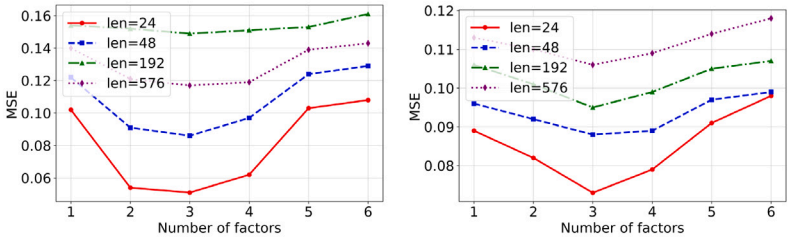


(e) DC-Air-Abnormal, pred_len=96

Fig. 13. Ablation study: Prediction curves of different modules with a prediction length of 96.



(a) DC-Temp-Normal (b) DC-Temp-Abnormal (c) DC-Power



(d) DC-Air-Normal (e) DC-Air-Abnormal

Fig. 14. The impact of the number of factors in the factor separation module on prediction results.

Table 5
The Kolmogrov–Smirnov test P-values generated by various graph-based temporal prediction models on temperature datasets with different prediction horizons are presented. A higher P-value indicates a lower likelihood of rejecting the hypothesis that the input sequence and predicted output originate from the same distribution. The optimal results are highlighted for further analysis.

Methods		MTGNN	StemGNN	GTS	NRI	I2A-RI	MSE-STGNN	True
DC-Temp-Normal	24	0.0221	0.0192	0.0463	0.0431	0.0616	**0.0812**	0.0461
	48	0.0122	0.0104	0.0423	0.0407	0.0482	**0.0562**	0.0393
	96	0.0181	0.0125	0.0227	0.0223	0.0258	**0.0383**	0.0291
	192	0.0096	0.0044	0.0279	0.0259	0.0291	**0.0351**	0.0224
DC-Temp-Abnormal	24	0.0212	0.0134	0.0338	0.0363	0.0474	**0.0521**	0.0163
	48	0.0091	0.0043	0.0112	0.0134	0.0192	**0.0211**	0.0128
	96	0.0011	0.0013	0.0024	0.0023	**0.0051**	0.0032	0.0079
	192	0.0004	0.0005	0.0007	0.0008	**0.0022**	0.0014	0.0013
DC-Power	24	0.0001	0.0001	0.0007	0.0008	0.0008	**0.0009**	0.0010
	48	0.0001	0.0001	0.0007	0.0011	0.0017	**0.0018**	0.0020
	192	0.0102	0.0093	0.0136	0.0196	0.0236	**0.0411**	0.0390
	576	0.0141	0.0002	0.0129	0.0168	0.0087	**0.0496**	0.0480
DC-Air-Normal	24	0.0031	0.0023	0.0037	0.0149	0.0088	**0.0446**	0.0580
	48	0.0013	0.0009	0.0018	0.0109	0.0328	**0.0369**	0.0360
	96	0.0132	0.0098	0.0201	0.0106	0.0547	**0.0598**	0.0550
	192	0.0103	0.0106	0.0121	0.0122	**0.0167**	0.0165	0.0170
DC-Air-Abnormal	24	0.0016	0.0001	0.0001	0.0001	0.0178	**0.0188**	0.0190
	48	0.0046	0.0098	0.0032	0.0048	0.0258	**0.0296**	0.0310
	96	0.0001	0.0001	0.0009	0.0009	0.0008	**0.0100**	0.0101
	192	0.0001	0.0002	0.0008	0.0008	0.0288	**0.0457**	0.0488
Count		0	0	0	0	3	17	

prediction level similar to that of I2A-RI, underscoring the generality of the MS module.

4.6. Analysis of distribution in predictive results

In this section, we first analyze the similarity between the input sequence and the predicted sequence distributions of various graph-based time series prediction models. Furthermore, we conduct an investigation to determine whether there exists statistical significance among the prediction outcomes of different models. Kolmogrov–Smirnov test is employed to examine whether the prediction results generated by different models are consistent with the input sequences. The results are presented in Table 5. In the experiment, the input sequence length is fixed at 48, and the null hypothesis is that the two sequences, input and prediction, are derived from the same distribution. On all the datasets, a common P-value of 0.01 is set. As can be observed from the results presented in the table, MTGNN and StemGNN exhibite smaller P-values compared to other methods. In contrast, the P-values of I2A-RI and MSE-STGNN are much larger, while MSE-STGNN has an average increase of 40.1% compared to I2A-RI, indicating that the output sequences generated by MSE-STGNN are more similar to the distribution of the input sequence compared to other models. These findings provide further evidence for the efficacy of the three designed modules.

We employ the Kruskal–Wallis test to compare the statistical significance between the prediction curves obtained from other graph-based prediction methods and those obtained from the proposed method. We set a significance level of 0.01, and if the P-value is less than the significance level, we consider there to be a significant difference between the two prediction methods. The results are presented in Table 6, where we highlight the results with P-values greater than 0.01 for each method. From the table, we can observe that the prediction results of NRI and I2A-RI are not significantly different from those of the proposed method in 13/20 cases, which are significantly better than the other three methods. Both NRI and I2A-RI share a similar architecture with the proposed method, as they also adopt the VAE framework. The comparison table of the methods also indicates that these two methods outperform other graph-based prediction methods, further demonstrating the superiority of this architecture.

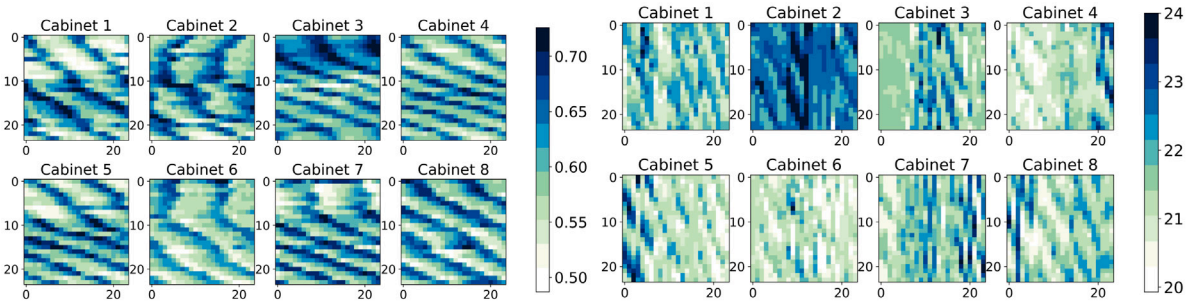
Table 6
The P-values obtained from the Kruskal–Wallis test between the predictive curves obtained from other graph-based prediction methods and those obtained from the proposed method. A higher P-value indicates a smaller likelihood of rejecting the null hypothesis that the predictive results of the two methods are from the same distribution. Results with a P-value greater than 0.01 for each method are highlighted.

Methods		MTGNN	StemGNN	GTS	NRI	I2A-RI
DC-Temp-Normal	24	0.0001	0.0001	0.0001	0.0001	0.0072
	48	0.0001	0.0001	0.0000	0.0001	0.0026
	96	0.0056	0.0025	0.0002	**0.1539**	**0.2556**
	192	**0.2119**	**0.2003**	0.0002	**0.6407**	**0.7804**
DC-Temp-Abnormal	24	**0.0468**	**0.0314**	**0.0688**	**0.0696**	**0.0726**
	48	0.0002	0.0002	0.0002	**0.0357**	**0.2600**
	96	0.0001	0.0001	0.0001	**0.2482**	**0.4544**
	192	**0.3012**	**0.2905**	**0.3074**	**0.2094**	**0.2261**
DC-Power	24	0.0001	0.0001	0.0001	0.0001	0.0001
	48	0.0000	0.0000	0.0000	0.0001	0.0001
	192	**0.0101**	0.0056	**0.0371**	**0.0107**	0.0049
	576	**0.5059**	**0.7247**	**0.8952**	**0.9741**	**0.6710**
DC-Air-Normal	24	0.0096	**0.0466**	**0.1588**	**0.2357**	**0.0239**
	48	**0.0143**	**0.0405**	**0.0214**	**0.0238**	**0.5406**
	96	0.0002	0.0003	0.0001	0.0009	0.0061
	192	0.0001	0.0000	0.0014	0.0006	**0.1968**
DC-Air-Abnormal	24	**0.0764**	0.0065	**0.0266**	**0.0889**	**0.0926**
	48	**0.1933**	0.0044	**0.1417**	**0.0233**	**0.1859**
	96	0.0001	0.0001	**0.0103**	0.0018	0.0002
	192	0.0001	0.0001	0.0088	**0.6915**	**0.7099**
Count		8	6	9	13	13

4.7. Visual analysis

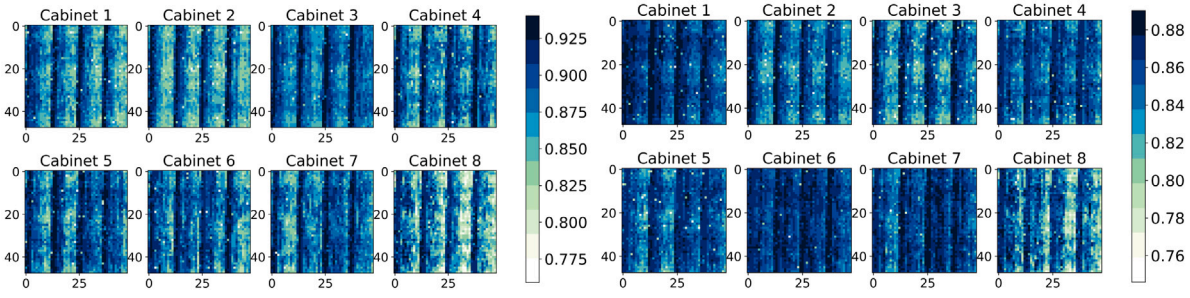
4.7.1. Visualization of attention maps

We analyze the impact of the periodicity deviation attention on the results. The addition of the attention module is intended to compensate for the inherent inability of the causal convolution to effectively capture periodic shifts. To this end, we visualize the parameters of the attention vector, as illustrated in Figs. 15, 16 and 17. The horizontal axis represents the 24 historical time points preceding the prediction,



(a) DC-Temp-Normal (b) DC-Temp-Abnormal

Fig. 15. Attention map for the MSE-STGNN training on the cabinet temperature dataset. The illustration exhibits the attention vector's capability to capture the deviation of periodicity and the abrupt points of anomalies.



(a) Time segment 1 in DC-Power dataset (b) Time segment 2 in DC-Power dataset

Fig. 16. Attention map for the MSE-STGNN training on the IT power dataset. The illustration delineates the remarkable ability of the attention vector to discern the subtle deviations in the periodicity of the IT power consumption of cabinet units.

whereas the vertical axis represents the 24 time points to be predicted in the future. We randomly select 8 cabinets and cold aisles for demonstration purposes, with darker colors indicating higher attention vector weights.

In Fig. 15, our findings reveal that the attention vector is highly effective in capturing periodic shifts. In particular, within the DC-Temp-Normal dataset, Cabinet 1 displays a significant level of attention towards the nearest two time points and the future values between the 10th and 13th time points. This is indicative of the attention vector's ability to effectively capture periodic shifts, thereby serving as an excellent complement to the causal convolution. Moreover, the results depicted in the figure demonstrate that the attention mechanism is capable of identifying exceptional instances. For instance, in the DC-Temp-Abnormal dataset, Cabinet 2 displays a considerable level of attention towards the value around the 10th historical time point, which corresponds to an unusual temperature change. Consequently, the attention vector can effectively capture sudden changes, serving as a potent auxiliary variable for predicting future values.

Within the DC-Power dataset, as shown in Fig. 16, our analysis reveals that the attention vector is highly adept at capturing various types of periodic shifts. As the IT power consumption cycle is relatively unstable, the attention mechanism is capable of effectively capturing their different cycles. As such, the attention vector proves to be highly effective in facilitating the capture of periodic shifts and exceptional instances, thereby enhancing the accuracy of the predictions.

The attention maps for the DC-Air-Normal and DC-Air-Abnormal datasets are presented in Fig. 17. The figure illustrates that for the cold aisle temperature in the normal state, the model uniformly relies on the values at different previous time steps during prediction. In contrast, for the abnormal state, the model tends to focus more on the values at specific previous time steps. This observation implies that the model can effectively capture changes in historical time steps during the abnormal state, resulting in more accurate predictions of future time steps. Additionally, the figure reveals that despite the relative instability of the cold channel temperature dataset compared to the cabinet temperature dataset, the attention mechanism can still capture temporal features of different periods.

4.7.2. *Visualization of the dynamic change process of the graph*

To illustrate the evolution of the graph structure during the training process, we use the prediction of cold aisle temperature as an example. The results are presented in Fig. 18, which shows the changes in the graph as the number of epochs increases, with a graph plotted every five epochs. The first 20 nodes in the graph represent the air conditioning supply temperature, while the last 18 nodes represent the cold aisle temperature. The graph contains multiple types of nodes. To better visualize the impact of each node on the others, the nodes are colored according to their influence, with darker colors indicating a greater influence on the other nodes.

Initially, we input a fully-connected graph and removed self-connected edges. Fig. 18(a) shows the graph after training for five epochs, with a relatively uniform distribution of edges, indicating significant interactions between the air conditioning supply temperature, the cold aisle temperature, and nodes of the same type. Fig. 18(b) shows the graph after training for 10 epochs, with a significant reduction in the number of edges, indicating that the model automatically eliminates redundant edges during the iteration process to reduce computational costs. Fig. 18(c) shows the graph after training for 15 epochs, with a sparser structure, but still with a relatively uniform distribution of edges. Fig. 18(d) shows the graph after training for 20 epochs, with significant changes in the structure, and the high-weight edges mainly concentrated on the right half of the graph, which represents the impact of air conditioning supply temperature on the cold aisle temperature. This is consistent with the physical system. Fig. 18(e) shows the graph after training for 25 epochs, with a converged model and a sparse graph structure, where the high-weight edges are mainly concentrated on the air conditioning supply temperature and the interactions between the cold aisle temperature nodes. Thus, we can conclude from the

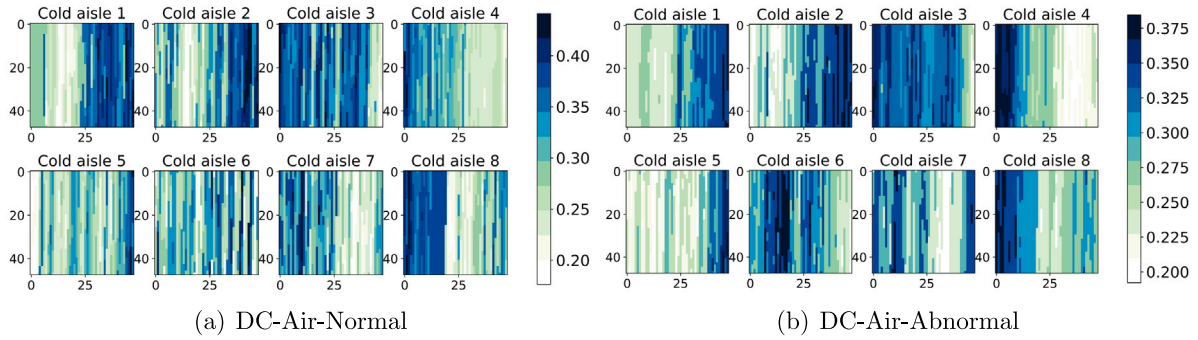


Fig. 17. Attention map for the MSE-STGNN training on the cold aisle temperature dataset.

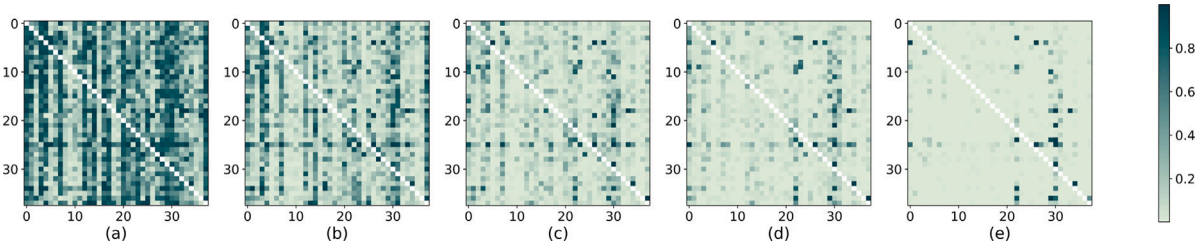


Fig. 18. The visualization of the dynamic change process of the graph.

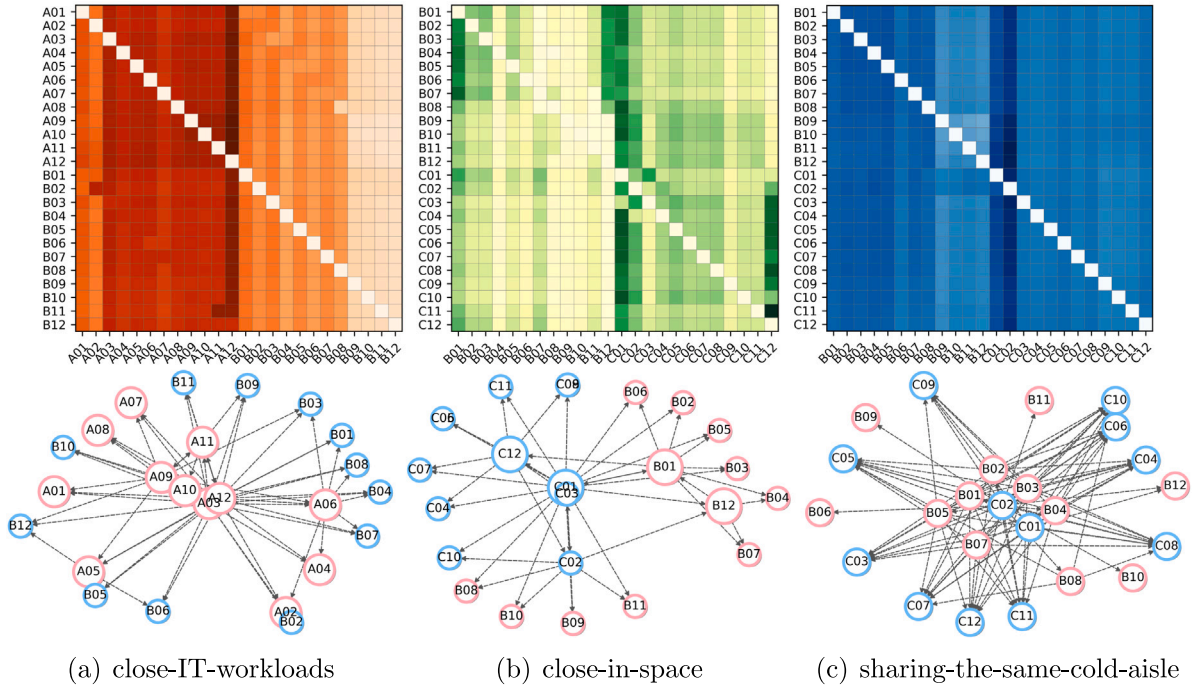


Fig. 19. The relations between the cabinets after disentanglement.

evolving graphs that the model can learn interpretable graphs step by step, which can be used not only for prediction but also for root cause analysis and other scenarios.

4.7.3. Visualization of the MS graph

We visualize the graph after the MS module using an example from the cabinet temperature dataset. In the cabinet temperature dataset, it is hypothesized that the connectivity between cabinets is influenced by three factors: close IT workloads, close proximity in physical space, and sharing the same cold aisle. This paper presents the learned matrices and graphs obtained by separating these three factors, as shown in Fig. 19. The degree of correlation between two nodes is indicated by the grid color of the matrix, with darker colors indicating stronger correlations. The graphs display only the edges with the highest connection probabilities for clarity.

Fig. 19(a) illustrates the relationships inferred through the close-IT-workloads factor. The matrix indicates that cabinets in the A-column have a greater influence on the other cabinets, which is further supported by the graph where cabinets A01-A12 have more out edges than B01-B12. This can be attributed to the fact that A01-A12 are network cabinets, and thus exert a greater impact on other cabinets in the system.

Fig. 19(b) depicts the relationships inferred through the close-in-space factor. The graph reveals that cabinets B01, B12, C01, and C12 have a higher number of edges than the other cabinets. This can be explained by the fact that these four cabinets are closest to the air conditioner, and thus the air conditioner has a stronger influence on these cabinets than on others.

Finally, Fig. 19(c) presents the relationships inferred through the sharing-the-same-cold-aisle factor. The graph shows that each cabinet is connected to other cabinets, which is attributed to the fact that B-column and C-column cabinets share the same cold aisle. Therefore, all the cabinets are interconnected through this shared space.

Overall, the MS module successfully segregates the three factors and produces interpretable relationships that demonstrate the feasibility of building a latent graph in this manner.

5. Conclusions

This paper proposes a method for constructing the graph among variables in data centers and utilizing it for multivariate time series forecasting. The framework first considers the diversity of edges in the process of constructing the graph, and proposes a multi-factor separation module to separate the factors that affect node connectivity, thus obtaining a graph that is more consistent with the actual situation. Then, considering the changes in graph structure in long-term forecasting, the graph prediction module is proposed to gradually include the future graph structure during the prediction process, in order to correct the errors in graph structure that multi-step prediction depends on. In addition, this paper also proposes an attention-enhanced spatial–temporal dilated causal convolution module to more effectively utilize information related to space and historical events. We conducted extensive comparative experiments and validation experiments on real data center datasets, and the experimental results show that the framework proposed in this paper outperforms other advanced prediction methods in terms of prediction accuracy, and the learned graph structure is interpretable.

Certainly, this work has some limitations. Firstly, the number of factor types required for the model to achieve optimal performance needs to be determined through hyperparameter search. Furthermore, although this graph-based prediction method outperforms non-graph-based methods in time series prediction with interrelationships between variables, it may be less efficient. Therefore, two areas are worth investigating based on this work in the future: firstly, enabling the model to automatically learn the optimal number of factor types, which is crucial as it significantly impacts the prediction results. Secondly, the learned graph is sparse, and exploring more efficient spatial–temporal prediction on sparse graphs is of great importance in improving the prediction efficiency of the model.

CRediT authorship contribution statement

Fang Shen: Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Data curation. **Jialong Wang:** Writing – review & editing. **Ziwei Zhang:** Writing – review & editing, Methodology. **Xin Wang:** Writing – review & editing, Methodology. **Yue Li:** Data curation. **Zhaowei Geng:** Formal analysis. **Bing Pan:** Visualization. **Zengyi Lu:** Writing – review & editing. **Wendy Zhao:** Writing – review & editing. **Wenwu Zhu:** Writing – review & editing.

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.

Data availability

Data will be made available on request.

Acknowledgments

This work was supported by the National Key Research and Development Program of China (No. 2020AAA0106300), National Natural Science Foundation of China (No. 62222209, 62250008, 62102222), Beijing National Research Center for Information Science and Technology under Grant No. BNR2023RC01003, BNR2023TD03006, and Beijing Key Lab of Networked Multimedia.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.knosys.2023.110997.

References

[1] X. Xiao, J. Sun, J. Yang, Operation and maintenance (o&m) for data center: An intelligent anomaly detection approach, Comput. Commun. 178 (2021) 141–152.
[2] S. Jadon, J.K. Milczek, A. Patankar, Challenges and approaches to time-series forecasting in data center telemetry: A survey, 2021, arXiv preprint arXiv:2101.04224.
[3] J. Xue, F. Yan, R. Birke, L.Y. Chen, T. Scherer, E. Smirni, Practise: Robust prediction of data center time series, in: 2015 11th International Conference on Network and Service Management (CNSM), IEEE, 2015, pp. 126–134.
[4] S. Ouhame, Y. Hadi, A. Ullah, An efficient forecasting approach for resource utilization in cloud data center using CNN-LSTM model, Neural Comput. Appl. 33 (2021) 10043–10055.
[5] J. Ma, P. Cui, K. Kuang, X. Wang, W. Zhu, Disentangled graph convolutional networks, in: International Conference on Machine Learning, PMLR, 2019, pp. 4212–4221.
[6] M. Omori, Y. Nakajo, M. Yoda, Y. Joshi, H. Nishi, Energy-efficient task distribution using neural network temperature prediction in a data center, in: 2019 IEEE 17th International Conference on Industrial Informatics (INDIN), 2019.
[7] S. Asgari, H. Moazamigoodarzi, P.J. Tsai, S. Pal, I.K. Puri, Hybrid surrogate model for online temperature and pressure predictions in data centers, Future Gener. Comput. Syst. 114 (1) (2020).
[8] G.S. Kohli, P.S. Kaur, A. Singh, J. Bedi, TransLearn: A clustering based knowledge transfer strategy for improved time series forecasting, Knowledge-Based Syst. (Aug.5) (2022) 249.
[9] E. Otovi, M. Njirjak, D. Jozinovi, G. Maua, A. Michelini, I. Stajduhar, Intra-domain and cross-domain transfer learning for time series data—How transferable are the features? Knowl.-Based Syst. 239 (2022) 107976.
[10] R. Salles, K. Belloze, F. Porto, P.H. Gonzalez, E. Ogasawara, Nonstationary time series transformation methods: An experimental review, Knowledge Based Syst. 164 (JAN.15) (2019) 274–291.
[11] Y.S. Lee, L.I. Tong, Forecasting time series using a methodology based on autoregressive integrated moving average and genetic programming, Knowl.-Based Syst. 24 (1) (2011) 66–72.
[12] J. Huang, Z. Chai, H. Zhu, Detecting anomalies in data center physical infrastructures using statistical approaches, J. Phys.: Conf. Ser. 1176 (2) (2019) 022056.
[13] M. Marwah, R. Sharma, C. Bash, Thermal anomaly prediction in data centers, in: 2010 12th IEEE Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems, IEEE, 2010, pp. 1–7.
[14] D. Alves, K. Obraczka, R. Lindberg, Identifying relevant data center telemetry using change point detection, in: 2020 IEEE 9th International Conference on Cloud Networking (CloudNet), IEEE, 2020, pp. 1–4.
[15] 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.
[16] D. Cao, Y. Wang, J. Duan, C. Zhang, X. Zhu, C. Huang, Y. Tong, B. Xu, J. Bai, J. Tong, et al., Spectral temporal graph neural network for multivariate time-series forecasting, Adv. Neural Inf. Process. Syst. 33 (2020) 17766–17778.
[17] A. Deng, B. Hooi, Graph neural network-based anomaly detection in multivariate time series, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, 2021, pp. 4027–4035, (5).
[18] L. Cai, K. Janowicz, G. Mai, B. Yan, R. Zhu, Traffic transformer: Capturing the continuity and periodicity of time series for traffic forecasting, Trans. GIS 24 (3) (2020) 736–755.
[19] L. Bai, L. Yao, C. Li, X. Wang, C. Wang, Adaptive graph convolutional recurrent network for traffic forecasting, Adv. Neural Inf. Process. Syst. 33 (2020) 17804–17815.
[20] A. Khaled, A. Elsir, Y. Shen, TFGAN: Traffic forecasting using generative adversarial network with multi-graph convolutional network, Knowledge-Based Syst. (Aug.5) (2022) 249.

[21] Y. Sun, A hybrid approach by integrating brain storm optimization algorithm with grey neural network for stock index forecasting, in: Abstract and Applied Analysis, vol. 2014, Hindawi, 2014.
[22] B. Son, Y. Lee, S. Park, J. Lee, Forecasting global stock market volatility: The impact of volatility spillover index in spatial-temporal graph-based model, J. Forecast. (2023).
[23] D. Cheng, F. Yang, S. Xiang, J. Liu, Financial time series forecasting with multi-modality graph neural network, Pattern Recognit. 121 (2022) 108218.
[24] A. Yh, B. Xm, D.A. Yong, Natural visibility encoding for time series and its application in stock trend prediction, Knowl.-Based Syst. (2021).
[25] J. Han, H. Liu, H. Xiong, J. Yang, Semi-supervised air quality forecasting via self-supervised hierarchical graph neural network, IEEE Trans. Knowl. Data Eng. (2022).
[26] L. Zhang, D. Li, Q. Guo, J. Pan, Deep spatio-temporal learning model for air quality forecasting, Int. J. Comput. Commun. Control 16 (2) (2021).
[27] X.-B. Jin, Z.-Y. Wang, J.-L. Kong, Y.-T. Bai, T.-L. Su, H.-J. Ma, P. Chakrabarti, Deep spatio-temporal graph network with self-optimization for air quality prediction, Entropy 25 (2) (2023) 247.
[28] H. Lu, S. Uddin, Disease prediction using graph machine learning based on electronic health data: A review of approaches and trends, Healthcare 11 (7) (2023) 1031.
[29] A.S. Weigend, Time Series Prediction: Forecasting the Future and Understanding the Past, Routledge, 2018.
[30] Z. Han, J. Zhao, H. Leung, K.F. Ma, W. Wang, A review of deep learning models for time series prediction, IEEE Sens. J. 21 (6) (2019) 7833–7848.
[31] S. Du, T. Li, Y. Yang, S.-J. Horng, Multivariate time series forecasting via attention-based encoder–decoder framework, Neurocomputing 388 (2020) 269–279.
[32] S.-Y. Shih, F.-K. Sun, H.-y. Lee, Temporal pattern attention for multivariate time series forecasting, Mach. Learn. 108 (8) (2019) 1421–1441.
[33] F. Scarselli, M. Gori, A.C. Tsoi, M. Hagenbuchner, G. Monfardini, The graph neural network model, IEEE Trans. Neural Netw. 20 (1) (2008) 61–80.
[34] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, M. Sun, Graph neural networks: A review of methods and applications, AI open 1 (2020) 57–81.
[35] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, S.Y. Philip, A comprehensive survey on graph neural networks, IEEE Trans. Neural Netw. Learn. Syst. 32 (1) (2020) 4–24.
[36] T. Kipf, E. Fetaya, K.-C. Wang, M. Welling, R. Zemel, Neural relational inference for interacting systems, in: International Conference on Machine Learning, PMLR, 2018, pp. 2688–2697.
[37] D.P. Kingma, M. Welling, Auto-encoding variational bayes, 2013, arXiv preprint arXiv:1312.6114.
[38] C. Shang, J. Chen, J. Bi, Discrete graph structure learning for forecasting multiple time series, 2021, arXiv preprint arXiv:2101.06861.
[39] B. Yu, H. Yin, Z. Zhu, Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting, 2017, arXiv preprint arXiv:1709.04875.
[40] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, 2016, arXiv preprint arXiv:1609.02907.
[41] M. Schlichtkrull, T.N. Kipf, P. Bloem, R. Van Den Berg, I. Titov, M. Welling, Modeling relational data with graph convolutional networks, in: The Semantic Web: 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, Proceedings 15, Springer, 2018, pp. 593–607.
[42] S. Zhang, H. Tong, J. Xu, R. Maciejewski, Graph convolutional networks: a comprehensive review, Comput. Soc. Netw. 6 (1) (2019) 1–23.
[43] M. Schuster, K.K. Paliwal, Bidirectional recurrent neural networks, IEEE Trans. Signal Process. 45 (11) (1997) 2673–2681.
[44] W. Zaremba, I. Sutskever, O. Vinyals, Recurrent neural network regularization, 2014, arXiv preprint arXiv:1409.2329.
[45] S. Guo, Y. Lin, N. Feng, C. Song, H. Wan, Attention based spatial-temporal graph convolutional networks for traffic flow forecasting, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, 2019, pp. 922–929, (01).
[46] Y. Li, R. Yu, C. Shahabi, Y. Liu, Diffusion convolutional recurrent neural network: Data-driven traffic forecasting, 2017, arXiv preprint arXiv:1707.01926.
[47] Z. Wu, S. Pan, G. Long, J. Jiang, C. Zhang, Graph wavenet for deep spatial-temporal graph modeling, 2019, arXiv preprint arXiv:1906.00121.
[48] C. Zheng, X. Fan, C. Wang, J. Qi, Gman: A graph multi-attention network for traffic prediction, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, 2020, pp. 1234–1241, (01).
[49] S. Hadou, C.I. Kanatsoulis, A. Ribeiro, Space-time graph neural networks, 2021, arXiv preprint arXiv:2110.02880.
[50] Y. Chen, I. Segovia-Dominguez, B. Coskunuzer, Y. Gel, TAMP-s2gcnets: coupling time-aware multipersistence knowledge representation with spatio-supra graph convolutional networks for time-series forecasting, in: International Conference on Learning Representations, 2022.
[51] L. Franceschi, M. Niepert, M. Pontil, X. He, Learning discrete structures for graph neural networks, in: International Conference on Machine Learning, PMLR, 2019, pp. 1972–1982.
[52] Z. Shao, Z. Zhang, F. Wang, Y. Xu, Pre-training enhanced spatial-temporal graph neural network for multivariate time series forecasting, in: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2022, pp. 1567–1577.
[53] J. Ye, Z. Liu, B. Du, L. Sun, W. Li, Y. Fu, H. Xiong, Learning the evolutionary and multi-scale graph structure for multivariate time series forecasting, in: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2022, pp. 2296–2306.
[54] E. Jang, S. Gu, B. Poole, Categorical reparameterization with gumbel-softmax, 2016, arXiv preprint arXiv:1611.01144.
[55] C.J. Maddison, A. Mnih, Y.W. Teh, The concrete distribution: A continuous relaxation of discrete random variables, 2017.
[56] 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.
[57] T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, R. Jin, Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting, 2022, arXiv preprint arXiv:2201.12740.
[58] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, 2017, arXiv.
[59] D. Blakely, J. Lanchantin, Y. Qi, Time and space complexity of graph convolutional networks, Accessed: Dec 31 (2021).
[60] A. Paszke, S. Gross, F. Massa, A. Lerer, S. Chintala, Pytorch: An imperative style, high-performance deep learning library, 2019.
[61] A.v.d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, K. Kavukcuoglu, Wavenet: A generative model for raw audio, 2016, arXiv preprint arXiv:1609.03499.
[62] R. Adhikari, R.K. Agrawal, An introductory study on time series modeling and forecasting, LAP LAMBERT Acad. Publ. (2013).
[63] H. Zhou, 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 Intelligence, vol. 35, 2021, pp. 11106–11115, (12).
[64] A. Borovykh, S. Bohte, C.W. Oosterlee, Conditional time series forecasting with convolutional neural networks, 2017, arXiv preprint arXiv:1703.04691.
[65] F. Shen, Z. Li, B. Pan, Z. Zhang, J. Wang, W. Zhao, X. Wang, W. Zhu, Inter-and-Intra Domain Attention Relational Inference for Rack Temperature Prediction in Data Center, Springer, Cham, 2022.