Contents lists available at ScienceDirect

Applied Energy

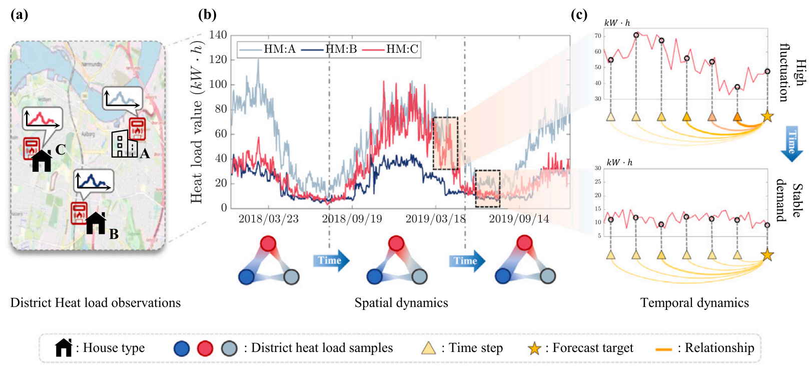
journal homepage: www.elsevier.com/locate/apenergy

Sparse dynamic graph learning for district heat load forecasting

Yaohui Huang [a,d,1], Yuan Zhao [b,1], Zhijin Wang [a,*], Xiufeng Liu [c,*], Yonggang Fu [a]

[a] College of Computer Engineering, Jimei University, Yinjiang Road 185, 361021 Xiamen, China
[b] School of Economics and Management, Lanzhou University of Technology, Langongping Road 287, 730050, Lanzhou, China
[c] Department of Technology, Management and Economics, Technical University of Denmark, 2800 Kgs. Lyngby, Denmark
[d] School of Automation, Central South University, South Lushan Road 932, 410083 Changsha, China

GRAPHICAL ABSTRACT



Graphical Abstract: An illustration of the challenges in district heat load forecasting. (a) Spatial representation of different house types and their heat load demands. (b) Time series of heat load demand for a typical week. (c) Temporal dynamics of heat load demand for different house types.

ARTICLE INFO

Keywords:
District heating
Dynamic graph neural network
Spatio-temporal forecasting
Heat load prediction
Sparse graph learning

ABSTRACT

Accurate heat load forecasting is crucial for the efficient operation and management of district heating systems. This study introduces a novel Sparse Dynamic Graph Neural Network (SDGNN) framework designed to address the complexities of forecasting heat load in district heating networks. The proposed model represents the district heating network as a dynamic graph, with nodes corresponding to consumers or heat sources and edges denoting temporal dependencies. The SDGNN framework comprises three key components: (1) a sparse graph learning module that identifies the most relevant nodes and edges, (2) a spatio-temporal memory enhancement module that captures both short-term and long-term dependencies, and (3) a temporal fusion module that integrates node representations into a comprehensive global forecast. Evaluated on a real-world district heating dataset from Denmark, the SDGNN model demonstrates superior accuracy and efficiency compared to existing methods. The results indicate that the SDGNN framework effectively captures intricate spatio-temporal patterns in historical heat load data, achieving up to 5.7% improvement in RMSE, 7.4% in MAE, and 5.7% in CVRMSE over baseline models. Additionally, incorporating meteorological factors into the model further enhances its predictive performance. These findings suggest that the SDGNN framework is a robust and scalable solution for district heat load forecasting, with potential applications in other domains involving spatio-temporal graph data.

* Corresponding authors.
E-mail addresses: yhhuang5212@gmail.com (Y. Huang), bmyzhao@outlook.com (Y. Zhao), zhijin@jmu.edu.cn (Z. Wang), xiuli@dtu.dk (X. Liu), yonggangfu@jmu.edu.cn (Y. Fu).
[1] Equal contributor.

https://doi.org/10.1016/j.apenergy.2024.123685
Received 11 March 2024; Received in revised form 18 May 2024; Accepted 9 June 2024
Available online 24 June 2024

1. Introduction

District heating is a centralized system of energy distribution that provides heat to residential and commercial buildings from a common source [1]. It is widely adopted in urban areas around the world as a sustainable and efficient solution for meeting the heating demand [2]. However, the optimal operation and management of district heating systems depend on accurate forecasting of the heat load, which is the amount of heat required by the consumers at a given time [3]. Forecasting the heat load is a challenging task due to the complex and dynamic nature of the district heating network, which involves multiple factors and uncertainties [4,5].

The very efficacy of these systems relies on balancing the supply with dynamic demand patterns. Factors like external temperature, insulation properties of buildings, human behaviors, and other unpredictable events interweave to create a fluctuating demand landscape [6,7]. Fig. 1 offers a tangible representation of these challenges. The spatial representation (part a) underscores the varied heat load demands across different house types, emphasizing the critical need to understand and predict individualized consumption patterns. The time series (part b) and the temporal dynamics (part c) further demonstrate the intricate interplay of spatial and temporal factors that make forecasting a formidable task [4].

Various statistical and machine learning methods have been proposed to address the forecasting problem of district heat load. These methods can be divided into two main categories: univariate and multivariate. Univariate methods, such as ARIMA, exponential smoothing, and seasonal decomposition, use only historical heat load data as input and ignore other relevant factors [8,9]. These methods are simple and easy to implement, but they often fail to capture the non-stationarity, non-linearity, and long-term dependencies in the data. Multivariate methods, such as artificial neural networks (ANNs), support vector machines, and random forests, use multiple input variables, such as meteorological data, calendar information, building data, etc., to forecast the heat load [10]. These methods are more powerful and flexible than univariate methods, but they also have some limitations. For example, they require large amounts of data and computational resources to train and optimize their parameters. They also tend to ignore or oversimplify the spatial dependencies and interactions among different consumers or heat sources in the network. However, both univariate and multivariate methods are based on traditional forecasting methodologies, such as linear regression or time-series analyses, which often falter in the face of the complex and dynamic nature of district heat demand [11]. Furthermore, as data influxes grow exponentially with the integration of IoT devices and smart meters, the need for more sophisticated predictive tools becomes palpable. Many state-of-the-art models, while adept at handling vast datasets, falter when confronted with abrupt demand spikes or novel consumption patterns [6,12]. Others suffer from issues of scalability or require vast computational resources, rendering them unsuitable for real-time forecasting in dynamic urban landscapes [13]. These limitations underscore an exigent need for innovative solutions.

In this paper, we propose a novel graph-based learning framework for district heat load forecasting. Our framework leverages the advantages of graph neural networks (GNNs), which are a class of deep learning models that can effectively capture both the structural and temporal information of graph-structured data [14]. We model the district heating network as a dynamic graph, where each node represents a time step, and each edge represents a temporal relationship or dependency between distinct consumer or heat source. Specifically, we focus on forecasting the thermal demand of individual buildings connected to a district heating network. Each building's heat load is represented as a node in the graph, and the edges capture temporal dependencies between the heat load patterns of different buildings. We design a sparse dynamic graph neural network (SDGNN) model (The complete list of acronym used in this paper can be found in Table 1) that can learn from both the node features (such as temporal dependencies between time steps) and the edge features (such as relationships between heat load meters) of the graph. Our model consists of three main components: a sparse graph learning component that constructs a dynamic graph structure from raw data by selecting the most relevant nodes and edges; a spatio-temporal memory enhancement component that enriches the node representations by capturing both short-term and long-term dependencies; and a temporal fusion component that aggregates the node representations into a global representation for forecasting. This component employs a reweighting mechanism for node representations, which is based on their relevance and importance to the forecasting task. This way, it can capture the global trends and patterns of the heat load demand across the network and reduce the noise and redundancy in the data. The primary applications of the SDGNN model include optimization of district heating operations, where accurate building-level heat load forecasts enable better management of heat production and distribution, leading to increased efficiency and reduced costs. Additionally, the model supports demand response and control strategies by leveraging building-level heat load predictions to optimize energy consumption patterns. It also facilitates anomaly detection, as deviations between predicted and actual heat loads can help identify potential issues with building equipment or occupant behavior. The main contributions of our paper are:

- We present a novel graph-based learning framework that can capture both temporal sequences and spatial relationships in a unified manner for forecasting the heat demand and supply of district heating systems.
- We propose a neural network structure that integrates a graph sparse module, memory enhancement unit, and temporal fusion component to enhance the performance and efficiency of graph learning.
- We evaluate our framework on a real-world district heating dataset from Denmark, and demonstrate that it achieves state-of-the-art results in terms of forecasting accuracy and computational complexity, and provides interpretable insights into the heat dynamics of district heating systems.

The rest of this paper is organized as follows. In Section 2, we review some related work on district heat load forecasting methods. In Section 3, we present our proposed SDGNN model in detail. In Section 4, we conduct extensive experiments to evaluate our model and compare it with several baselines. We conclude in Section 5 and discuss some future directions for research.

Table 1
Abbreviation and description.

Abbreviation	Description
ABPA	Adaptive Backpropagation Algorithm
ANNs	Artificial Neural Networks
BN	Batch Normalization
CVRMSE	Coefficient of Variation of Root Mean Square Error
DHS	District Heating System
FCN	Fully Connected Network
FLOPS	Floating-point Operations Per Second
GNNs	Graph Neural Networks
GPU	Graphics Processing Unit
LSTM	Long Short-Term Memory
MACs	Multiply-Accumulate Operations
MAE	Mean Absolute Error
MSE	Mean Square Error
PCC	Pearson Correlation Coefficient
ReLU	Rectified Linear Unit
RMSE	Root Mean Square Error
RNNs	Recurrent Neural Networks
SCC	Spearman's rank Correlation Coefficient
SDGNN	Sparse Dynamic Graph Neural Network
SVMs	Support Vector Machines

2. Related work

The task of district heat load forecasting is a complex and multifaceted problem that has garnered significant attention in both academia and industry. Various methodologies have been explored to address this issue, ranging from traditional statistical models to advanced machine learning techniques. Additionally, the emergence of graph-based models and the incorporation of spatio-temporal memory have opened new avenues for improving the accuracy and robustness of forecasts. This section aims to provide a comprehensive review of these methodologies, discussing their strengths, limitations, and applicability in the context of district heat load forecasting.

2.1. Heat load forecasting methods

The landscape of load forecasting has experienced a paradigm shift, signifying a substantial transformation in both methodology and application. This evolution is marked by a transition from traditional time-series econometric models to advanced machine learning and deep learning techniques. Sarajcev et al. [15] utilize ensemble learning and aggregate load clustering for short-term forecasting, while Mishra et al. [16] emphasize the enduring relevance of traditional methods across various contexts. This divergence underscores a pressing issue: the integration of emerging machine learning algorithms with well-established statistical models.

Data-driven models, especially regression-based supervised learning techniques, have gained prominence in heat demand prediction within District Heating Systems (DHS) [17]. Idowu et al. [18] investigate machine learning approaches like support vector regression and decision trees in both residential and commercial settings. Kurek et al. [19] build upon this by examining forecasting methods for heat demand in the European Union’s largest District Heating Network, considering seasonal variations and a 72-hour time horizon. These studies collectively underscore the expanding versatility of machine learning techniques. Dixon [20] introduces exponential smoothed Recurrent Neural Networks (RNNs), tailored to model non-stationary dynamical systems commonly found in industrial settings. Dixon’s work acts as a linchpin in harmonizing the strengths of machine learning with the robustness and interpretability of traditional statistical models. However, it is essential to recognize that traditional algorithms have limitations, including susceptibility to low accuracy and high sensitivity to data noise. To address these challenges, Song et al. [21] introduce a Convolutional Neural Network–Long Short-Term Memory (CNN–LSTM) algorithm, adept at capturing the complex dynamics of heating load. Their work highlights the promise of hybrid models that amalgamate features of both neural networks and traditional statistical methods to enhance forecasting accuracy and reliability.

A notable advancement is Huang et al.’s [4] Active Graph Recurrent Network (Ac-GRN), which not only improves forecast accuracy but also introduces a layer of explainability. This focus on explainability aligns with a broader trend in machine learning aimed at making advanced algorithms more transparent and accountable. This trend is supported by an exhaustive study [22] that evaluates both the strengths and weaknesses of machine learning techniques in thermal load forecasting. The study serves as a critical appraisal of the current state-of-the-art, emphasizing the need for models that are not only accurate but also interpretable. Machine learning’s applicability extends to the building sector, as demonstrated by Bassi et al. [23], who validate gradient boosting algorithms like LightGBM, CatBoost, and XGBoost. Their work shows that these algorithms can outperform traditional methods, delivering more accurate and reliable forecasts. However, it is crucial to acknowledge that machine learning models have inherent limitations, such as the challenge of extrapolating beyond known data ranges. Faber et al. [24] address this by introducing “Deep DHC”, a hybrid method that synergizes the strengths of decision trees and neural networks. This hybrid approach reflects a growing trend to integrate different machine learning techniques to offset individual weaknesses and enhance overall performance. Additionally, the complexities of electric load forecasting have been amplified by the emergence of decentralized energy generation and fluctuating work patterns. Borghini et al. [25] address these challenges using tabnet, a machine learning model that has shown promise in navigating these complexities. Their work attests to the adaptability and versatility of machine learning models in tackling evolving challenges in load forecasting.

While ANNs yield promising results, their limitations in long-term forecasting cannot be overlooked. Mohammed et al. [26] tackle this by developing an ANN model enhanced with an Adaptive Backpropagation Algorithm (ABPA). Complementing this, Giamarelos et al. [27] introduce a mixed power-load forecasting scheme that exhibits competitive accuracy across multiple prediction horizons. These advancements pave the way for our further research, especially in addressing the limitations of existing models and in incorporating additional data sources for more accurate and robust forecasting.

2.2. Graph-based models and spatio-temporal memory in forecasting

This subsection delves into the confluence of graph-based models, notably Graph Neural Networks (GNNs), and spatio-temporal memory mechanisms like Long Short-Term Memory (LSTM) in the realm of forecasting. The discussion is anchored in existing literature and emphasizes the synergistic impact of these technologies, especially in the context of district heat load forecasting.

Graph Neural Networks (GNNs) have revolutionized forecasting by offering a sophisticated means to model intricate spatial relationships [28]. Unlike conventional machine learning models, which falter when dealing with irregular data structures, GNNs excel in capturing the spatial dependencies inherent in diverse fields such as traffic management and mobile networks [29–31]. Utilizing advanced techniques like spatial transformers and graph learning modules, GNNs dynamically model directed spatial dependencies, thereby capturing real-time conditions and flow directionality [29,31]. Their adaptability to various spatial dependency patterns, facilitated by multi-head attention mechanisms, further elevates their forecasting accuracy and reliability [5,30].

Incorporating spatio-temporal memory mechanisms, such as LSTM and Transformer architectures, significantly augments the predictive prowess of graph-based models. These mechanisms excel in capturing both long-term and short-term temporal dependencies, thereby enriching the understanding of data dynamics. For example, in wind speed forecasting, hybrid models combining convolutional neural networks and LSTM have demonstrated superior accuracy by capturing complex spatio-temporal correlations [32]. Likewise, in solar power forecasting, graph-convolutional LSTM and Transformer models have been employed to yield fine-grained predictions by capturing intricate spatio-temporal dependencies [33]. These mechanisms not only enhance temporal sequence understanding but also enable the integration of additional contextual factors like weather conditions and regional distributions, thereby rendering the forecasts more reliable and actionable [34,35].

The fusion of graph-based models with spatio-temporal memory mechanisms forms a robust framework for forecasting, enabling a nuanced understanding of both spatial and temporal dependencies. This integrated approach has proven particularly effective in diverse applications, including solar power and traffic flow forecasting. For instance, graph-convolutional long short-term memory (GCLSTM) and graph-convolutional transformer (GCTrafo) models have been developed to achieve forecasts with higher spatial and temporal resolution in multi-site photovoltaic power forecasting [33]. Similarly, in traffic flow forecasting, a graph-based temporal attention framework has outperformed several state-of-the-art models by leveraging both spatial and temporal correlations [36]. In epidemic forecasting, the introduction of CausalGNN has incorporated underlying causal mechanisms into

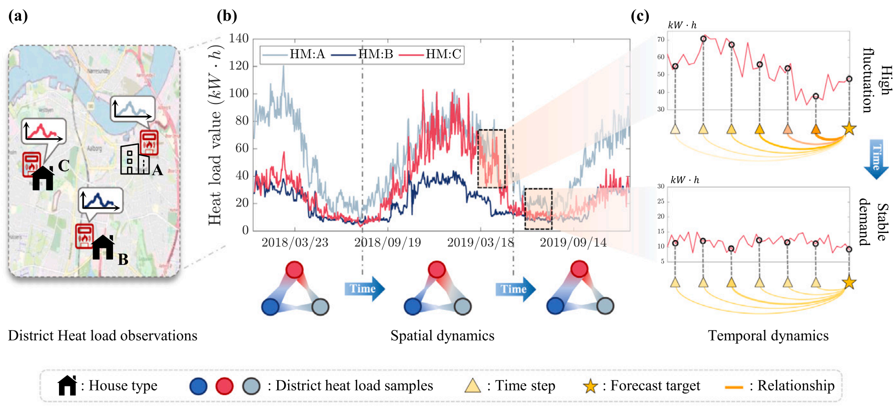


Fig. 1. An illustration of the challenges in district heat load forecasting. (a) Spatial representation of different house types and their heat load demands. (b) Time series of heat load demand for a typical week. (c) Temporal dynamics of heat load demand for different house types.

spatio-temporal embeddings, thereby enhancing forecast robustness and accuracy [37]. These advancements highlight the indispensable nature of this integrated approach for tackling complex forecasting challenges [38].

In the niche area of district heat load forecasting, the synergy between graph-based models and spatio-temporal memory mechanisms presents a promising avenue for innovation. However, existing literature on this domain is scarce and mostly focused on smart grid load forecasting [39], which is not directly applicable to district heating systems. Therefore, there is a need for a novel graph-based learning framework that can capture both spatial and temporal information in district heat load forecasting, and address the challenges of data sparsity, network complexity, and dynamic changes. In this paper, we propose a sparse dynamic graph neural network that considers the sparsity of the graph structure and integrates graph convolutional and attention mechanisms to enhance the performance and efficiency of graph learning.

3. Methodology

This section explains the methodology of our study on district heat load forecasting using the SDGNN model. We begin by defining the problem and the input and output variables for forecasting. Then, we give an overview of the SDGNN model and its modules, and describe how we implemented it.

3.1. Problem formulation

The problem of district heat load forecasting can be formulated as follows: given a set of historical heat load records $\mathcal{L} = \{\boldsymbol{X}_1, \boldsymbol{X}_2, \dots, \boldsymbol{X}_Q\}$ for Q buildings in a district, where $\boldsymbol{X}_q = \{x_{q,1}, x_{q,2}, \dots, x_{q,T}\}$ is a time series of heat load values for building q at T time steps, and a set of exogenous factors $\mathcal{Z} = \{\boldsymbol{Z}_1, \boldsymbol{Z}_2, \dots, \boldsymbol{Z}_Q\}$ for each building, where $\boldsymbol{Z}_q = \{z_{q,1}, z_{q,2}, \dots, z_{q,T}\}$ is a vector of external variables that may affect the heat load demand for building q. Our primary goal in this paper is to predict the future heat load values $\hat{\mathcal{L}} = \{\hat{\boldsymbol{X}}_1, \hat{\boldsymbol{X}}_2, \dots, \hat{\boldsymbol{X}}_Q\}$ for each individual building q at a horizon of τ time steps ahead. This involves forecasting heat load values for all heat meters for the upcoming τ time steps based on historical observations from the preceding T time steps.

$$\{\hat{\boldsymbol{X}}_{t+1}, \hat{\boldsymbol{X}}_{t+2}, \dots, \hat{\boldsymbol{X}}_{t+\tau}\} \leftarrow \mathcal{F}_{\Phi}(\boldsymbol{X}_{t-T+1}, \boldsymbol{X}_{t-T}, \dots, \boldsymbol{X}_t; \Phi), \tag{1}$$

where $\hat{\boldsymbol{X}}_{t+i}$ is the prediction values of the next i-steps. Φ denotes the learnable parameters in the forecasting model $\mathcal{F}_{\Phi}$.

3.2. Model overview

To solve this problem, we propose an SDGNN model for district heat load forecasting, illustrated in Fig. 2. The history heat load observations are used to collect the load records of different districts and train the SDGNN model. The SDGNN model is composed of three modules: graph learning module, spatio-temporal memory enhancement, and temporal fusion. The graph learning module constructs the sparse graph structure and employs the multi-layer graph convolutional neural network to capture the graph dynamics. The spatio-temporal memory enhancement captures the spatial and temporal dependencies among the districts by aggregating the features from the dynamic graph. The temporal fusion module generates the future heat load predictions by combining the outputs of the previous modules and skipping some intermediate layers. The framework aims to achieve accurate and robust district heat load forecasting by leveraging the spatio-temporal information and global context of the data. In the following subsections, we will describe each module in detail.

3.3. Sparse dynamic graph neural network

Fig. 3 illustrates the proposed SDGNN that consists of three main components: graph learning module, spatio-temporal memory enhancement, and temporal fusion. The heat load observations are the input data for the model, which are time series of heat load values for different districts. The patching component is a preprocessing step that divides the time series into patches of equal length and assigns each patch a label based on its pattern. The graph construction component is the core of the model, which builds a dynamic graph for each patch based on the spatio-temporal dependencies and global context of the data. The sparse graph learning component learns the temporal dependencies among the districts by constructing a sparse dynamic graph based on the historical data. The spatio-temporal memory enhancement component enriches the node features by incorporating the historical and future information using a global convolutional neural network and attention mechanism. The temporal fusion component integrates linear trend features with nonlinear representations to generate the final output, which is the heat load prediction for each district. The temporal fusion component also avoids over-nonlinearity and enhances the stability of the prediction model. In the following, we will describe the three modules in the graph component in detail.

3.3.1. Sparse graph learning

In district heating systems, the heat demand for a specific time period often exhibits correlations with past periods characterized by similar weather conditions, days of the week, and seasonality. Capturing these correlations and patterns is vital for efficient forecasting.

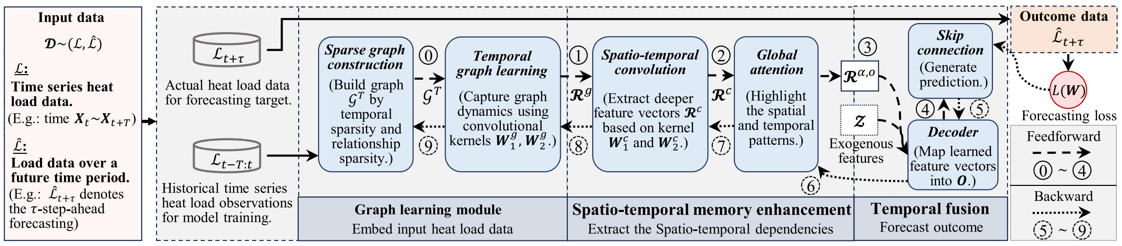

Input data
Time series heat load data.
Load data over a future time period.
Actual heat load data for forecasting target.
Historical time series heat load observations for model training.
Sparse graph construction
Temporal graph learning
Spatio-temporal convolution
Global attention
Skip connection
Decoder
Exogenous features
Outcome data
Forecasting loss
Feedforward
Backward
Graph learning module
Spatio-temporal memory enhancement
Temporal fusion

Fig. 2. The framework of district heat load forecasting using proposed SDGNN model.

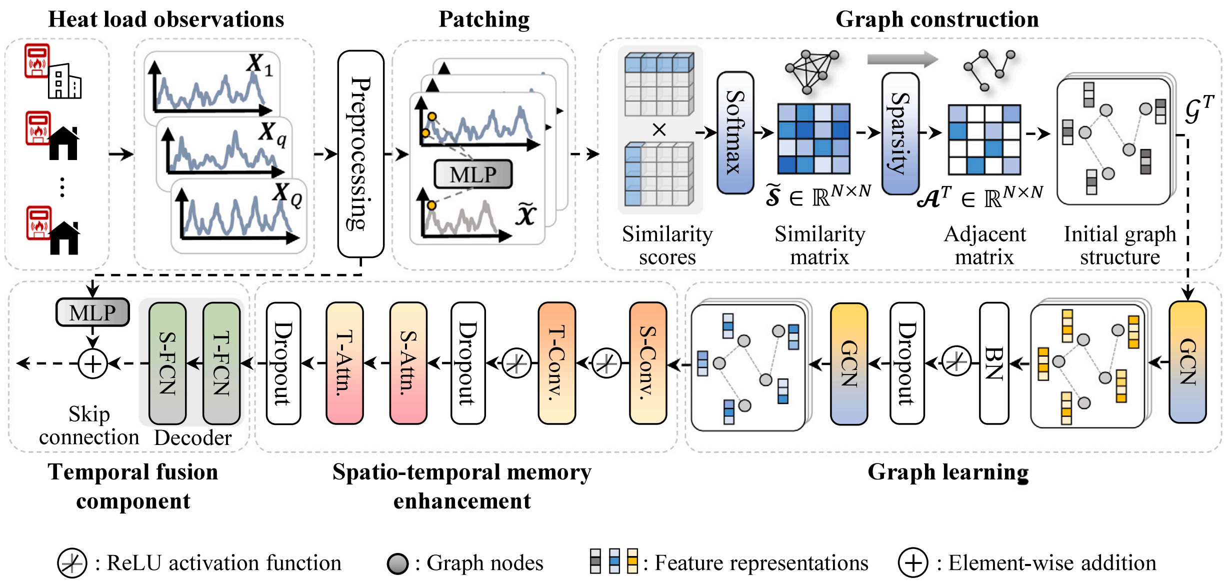


Fig. 3. The overview of proposed sparse dynamic graph neural network model.

Graph-based learning offers a promising approach to capture the inherent temporal patterns and dependencies within heat load data. Let $\mathcal{G} = \{\mathcal{V}; \mathcal{E}; \mathcal{A}\}$ represent a graph structure. $\mathcal{V}$ denotes the set of nodes, with each node $\boldsymbol{v}_t \in \mathcal{V}$ corresponding to a distinct time point $\boldsymbol{X}_t$. The set of edges, denoted as $\mathcal{E}$, consists of individual edges represented by $\boldsymbol{e}_{i,j} \in \mathcal{E}$. Each edge serves to establish a temporal relationship or dependency between the time steps $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$. These intricate relationships can also be represented by the adjacency matrix $\mathcal{A}$. It is important to note that this graph does not represent the physical layout of the district heating network. Instead, it captures the temporal relationships between the heat load demands of individual buildings. For example, if two buildings tend to have similar heat load patterns over time, perhaps due to similar occupancy schedules or responses to weather changes, their corresponding nodes in the graph will have a strong connection.

The sparse graph learning process consists of two phases: the sparse graph construction and the temporal graph learning. In the sparse graph construction phase, sequence information is condensed to establish meaningful node relationships by distilling raw heat meter observations. Connections between nodes are dynamically and adaptively pruned to reflect temporal variations, retaining only the most correlated links. In the temporal graph learning phase, the obtained dynamic graph structure is fed into multi-layer graph convolution networks to capture inter-dependencies between nodes.

Sparse graph construction. As shown in Fig. 4, the sparse graph construction process operates along two primary dimensions: temporal sparsity and relationship sparsity. Temporal sparsity is intended to compress the sequence information obtained from raw data, thereby reducing the number of graph nodes. Conversely, relationship sparsity is centered on distilling and refining the edges between nodes, representing the associations between each time step. To achieve temporal

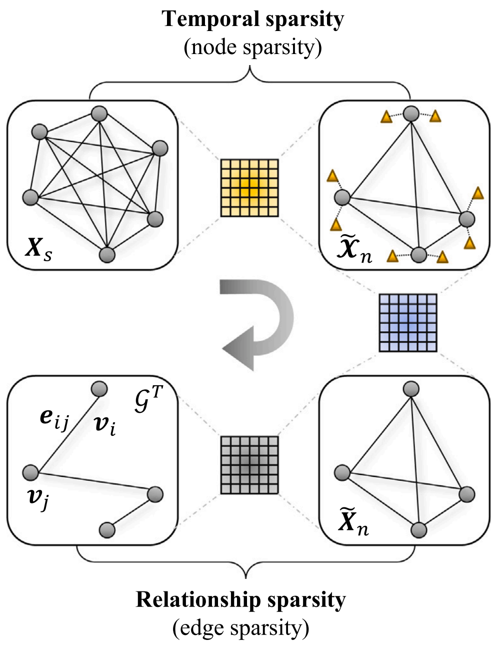


Fig. 4. The visualization of the proposed sparse graph construction.

sparsity, we employ a patching strategy to preprocess heat meter observations before incorporating them into the graph learning component [40]. Let P be a constant that denotes the patch length. Using this strategy, every continuous set of P time steps of observations is grouped into a single time slot represented as:

$$\{\underbrace{\cdots}_{\cdots},\underbrace{\boldsymbol{X}_s,\boldsymbol{X}_{s+1},\ldots,\boldsymbol{X}_{s+P-1}}_{\tilde{\boldsymbol{X}}_n},\underbrace{\boldsymbol{X}_{s+P},\boldsymbol{X}_{s+P+1},\ldots,\boldsymbol{X}_{s+2P-1}}_{\tilde{\boldsymbol{X}}_{n+1}},\underbrace{\cdots}_{\cdots}\}, \quad (2)$$

where $\tilde{\boldsymbol{X}} \in \mathbb{R}^{P\times Q}$ denotes heat load values for a time slot, while $\boldsymbol{X} \in \mathbb{R}^{1\times Q}$ represents the heat load for a single time step. $s \in [1, L]$ is a time step and $n \in [1, N]$ denotes a time slot. $N = \lceil T/P \rceil$ is defined as the ceiling of the division of T by P, denoting the total number of patches obtained after processing. The set of patched observations can be represented as $\mathcal{X} = \{\tilde{\boldsymbol{X}}_1, \tilde{\boldsymbol{X}}_2, \ldots, \tilde{\boldsymbol{X}}_N\} \in \mathbb{R}^{N\times P\times Q}$. If T is not an exact multiple of P, we apply padding using the final value $\boldsymbol{X}_t$ to ensure an even division. To extract representations from each time slot, a linear transformation is utilized to encode the temporal information within the patched tensor. This process can be expressed as follows:

$$\tilde{\mathcal{X}} = \text{Encoder}(\boldsymbol{W}^{enc}; \mathcal{X}), \quad (3)$$

where $\tilde{\mathcal{X}} \in \mathbb{R}^{N\times Q}$ denotes the final output resulting from the temporal sparsity process. $\boldsymbol{W}^{enc}$ represents the learnable weight parameters. Temporal sparsity effectively reduces the count of input tokens from T to approximately $\frac{T}{P}$, leading to a decrease in memory consumption and computational complexity within the graph learning components. This reduction process optimizes both training time and memory usage.

The objective of relationship sparsity is to selectively prune relationships between nodes within the graph. This is accomplished using the k-Nearest Neighbor approach. Specifically, during the graph structure construction, node similarities are calculated through the dot product of the input $\tilde{\mathcal{X}}$, as expressed in:

$$\mathcal{S} = \frac{\tilde{\mathcal{X}}^\top \cdot \tilde{\mathcal{X}}}{\sqrt{N}}, \quad (4)$$

where $\mathcal{S} \in \mathbb{R}^{N\times N}$ is the similarity matrix. N denotes the number of time slots. $\top$ represents the transpose operation. The resulting similarity matrix is subsequently normalized using the softmax function:

$$\tilde{\mathcal{S}}_{i,j} = \frac{\exp(\mathcal{S}_{i,j})}{\sum_{n=1}^{N}\exp(\mathcal{S}_{i,n})}, \quad (5)$$

where $\exp(\cdot)$ represents the exponential function. $\tilde{\mathcal{S}}_{i,j}$ is an element of the normalized similarity matrix $\tilde{\mathcal{S}} \in \mathbb{R}^{N\times N}$. For each node, the top-k maximum values in the weight matrix $\tilde{\mathcal{S}}$ are selected. k represents the ϵ sparsity of the edges in the graph, and it is calculated as $k = \epsilon \cdot N^2$. Corresponding entries in the adjacency matrix $\mathcal{A}$ set to 1, with all others set to 0. This approach simplifies the graph structure without reliance on predefined relationships. As the input time window $\boldsymbol{X}_{t-T+1:t}$ varies, the relationships between nodes also evolve, resulting in a dynamic adjacency matrix $\mathcal{A}$ that changes with time variations.

The generated graph can be expressed as $\mathcal{G}^T = \{\mathcal{V}^T; \mathcal{E}^T; \mathcal{A}^T\}$. Within this graph, $\mathcal{V}^T \in \mathbb{R}^N$ represents the set of nodes. Each node $\boldsymbol{v}_t \in \mathcal{V}^T$ corresponds to a distinct time slot. $\mathcal{E}^T \in \mathbb{R}^k$ is the set of edges, where each edge $\boldsymbol{e}_{i,j} \in \mathcal{E}^T$ establishes the temporal relationship or dependency between the time slots v_i and v_j. This temporal connectivity also can be represented by the adjacency matrix $\mathcal{A}^T$.

Temporal graph learning. The temporal graph learning component applies graph convolution operations to the patched inputs and the dynamic adjacency matrix. The graph convolution operations use learnable weights, biases, and activation functions to update the node features. The graph convolution operations can be expressed as follows:

$$\boldsymbol{W}_1^g *_G \tilde{\mathcal{X}} = \boldsymbol{D}^{-\frac{1}{2}} \cdot \hat{\mathcal{A}}^T \cdot \boldsymbol{D}^{-\frac{1}{2}} \cdot \tilde{\mathcal{X}} + \boldsymbol{b}_1^g, \quad (6)$$

where $*_G$ signifies the graph convolution operation. $\tilde{\mathcal{X}} \in \mathbb{R}^{N\times Q}$ denotes the input data after temporal sparsity. $\hat{\mathcal{A}}^T = \mathcal{A}^T + \boldsymbol{I}_u$ refers to the adjacency matrix with self-loops. $\boldsymbol{I}_u$ is a unit matrix. $\boldsymbol{D}$ denotes the diagonal matrix, $\boldsymbol{D}_{i,i} = \sum_j \mathcal{A}_{i,j}$ and $\boldsymbol{D}_{i,j} = 0$ for $i \neq j$. $\boldsymbol{W}_1^g$ and $\boldsymbol{b}_1^g$ are the learnable weights and bias, respectively. To aggregate information from a broader set of neighbors and enhance the capability of the model to capture intricate relationships within the graph, two graph convolution components are stacked to produce the final representations:

$$\boldsymbol{R}^g = \boldsymbol{W}_2^g *_G (\ \boldsymbol{M}^g(\gamma) \odot \sigma(\text{BN}(\boldsymbol{W}_1^g *_G \tilde{\mathcal{X}}))\ ), \quad (7)$$

where $\boldsymbol{R}^g \in \mathbb{R}^{N\times Q}$ is the generated representation in graph learning component. $\sigma(\cdot)$ represents the Rectified Linear Unit (ReLU) activation. $\boldsymbol{M}^g(\cdot)$ denotes dropout mask, γ is dropout rate. BN$(\cdot)$ represents batch normalization layer employed to stabilize and expedite the training process. $\boldsymbol{W}^g$ is the learnable weighting matrix. $\odot$ denotes hadamard product.

The temporal graph learning component learns the temporal dependencies among the districts by updating their node features based on their historical data and their dynamic graph structure. The temporal graph learning component outputs a set of node representations that can capture the spatio-temporal information and global context of the data.

3.3.2. *Spatio-temporal memory enhancement*

The spatio-temporal memory enhancement module integrates both a spatio-temporal convolution component and a global attention mechanism. While graph neural network captures intrinsic topological relationships within graph structures, it may not comprehensively accounts for intricate temporal dynamics and nuanced spatial variations. In contrast, spatio-temporal convolution excels in modeling both spatial and temporal dependencies, which is crucial when dealing with dynamically evolving graph structures and temporally varying node or edge attributes. The integration of spatio-temporal convolution partially alleviates the inherent limitations of conventional graph learning. Furthermore, this combined approach not only enhances the model's expressiveness, enabling the extraction of more intricate features, but also provides a form of regularization, potentially preventing overfitting [41]. The generated representation is formulated as follows:

$$\boldsymbol{R}^c = \boldsymbol{M}^c(\gamma) \odot \sigma(\boldsymbol{W}_2^c * \sigma(\boldsymbol{W}_1^c * \boldsymbol{R}^g)^\top), \quad (8)$$

where $\boldsymbol{R}^c$ represents the output from the spatio-temporal convolution. $\boldsymbol{W}^c$ denotes learnable parameters of the convolution kernel. $*$ is the conventional convolution operation. $\boldsymbol{M}^c(\gamma)$ denotes the dropout mask generated based on a dropout rate of γ.

The spatio-temporal convolution can capture both spatial and temporal dependencies inherent in the data, but its ability to focus on specific temporal patterns or crucial events may be limited. To address this shortcoming and emphasis critical temporal sequences and features, we integrate a global attention mechanism, enhancing the model's sensitivity to key spatio-temporal variations:

$$\boldsymbol{S}^{\alpha_1} = \text{softmax}(\boldsymbol{W}_2^{\alpha_1} \cdot \sigma(\boldsymbol{W}_1^{\alpha_1} \cdot \boldsymbol{R}^c) + \boldsymbol{b}^{\alpha_1}), \quad (9)$$

$$\boldsymbol{S}^{\alpha_2} = \text{softmax}(\boldsymbol{W}_2^{\alpha_2} \cdot \sigma(\boldsymbol{W}_1^{\alpha_2}(\boldsymbol{R}^c \odot \boldsymbol{S}^{\alpha})^\top) + \boldsymbol{b}^{\alpha_2}), \quad (10)$$

$$\boldsymbol{R}^{\alpha,o} = \boldsymbol{M}^{\alpha}(\gamma) \odot ((\boldsymbol{R}^c \odot \boldsymbol{S}^{\alpha_1})^\top \odot \boldsymbol{S}^{\alpha_2}), \quad (11)$$

where $\boldsymbol{R}^{\alpha,o}$ represents the generation of the global attention mechanism. $\boldsymbol{S}^{\alpha}$ denotes the attention score, while softmax$(\cdot)$ is the softmax function used to normalize these scores. $\boldsymbol{W}^{\alpha}$ is the weight matrix, and $\boldsymbol{b}^{\alpha}$ corresponds to the bias term. $\boldsymbol{M}^{\alpha}(\gamma)$ indicates the dropout mask with the dropout rate γ.

3.3.3. *Temporal fusion component*

Excessive nonlinearity can lead to unstable training dynamics, with oscillating loss and challenging convergence [42]. To address this challenge, the temporal fusion component utilizes a skip connection architecture. This architecture effectively integrates the raw input information with the enhanced representation derived from the spatio-temporal memory enhancement phase, resulting in the final prediction. Prior to this stage, we initially construct a decoder comprising dual embedding layers, each designed with a fully connected network (FCN). The embedding structures aims to map the condensed feature representation back to the desired target output shape:

$$\boldsymbol{O} = \mathrm{FCN}_{spat}(\mathrm{FCN}_{temp}(\boldsymbol{R}^{\alpha,o})), \tag{12}$$

where $\boldsymbol{O} \in \mathbb{R}^{\tau\times Q}$ denotes the generation of decoder. $\mathrm{FCN}_{spat}(\cdot)$ and $\mathrm{FCN}_{temp}(\cdot)$ are embedding layers to decode the feature matrix in variable and temporal dimension, respectively. To incorporate exogenous meteorological factors, the decoder's output can be expressed as:

$$\boldsymbol{O} = \mathrm{FCN}_{spat}(\ \mathrm{FCN}_{temp}([\boldsymbol{R}^{\alpha,o} \parallel \mathrm{Emb}(\mathcal{Z})])\ ), \tag{13}$$

where $\mathcal{Z}$ represents the input meteorological factors, and $[\cdot \parallel \cdot]$ denotes the concatenation operation. $\mathrm{Emb}(\cdot)$ signifies a linear transformation used for embedding the exogenous inputs to align with the shape of $\boldsymbol{R}^{\alpha,o}$.

Subsequently, a skip connection architecture is employed to integrate information derived from both the linear representation of the original heat load observations and the condensed feature representation arising from the spatio-temporal memory enhancement module:

$$\hat{\boldsymbol{X}}_{t+1:t+\tau} = \boldsymbol{O} \oplus (\boldsymbol{W}^o \cdot \boldsymbol{X}_{\mu:t} + \boldsymbol{b}^o), \tag{14}$$

where $\hat{\boldsymbol{X}}_{t+1:t+\tau} \in \mathbb{R}^{\tau\times Q}$ is the prediction values for the next τ steps, $\oplus$ denotes the element-wise addition operation. $\boldsymbol{X}_{\mu:t}$ represents the past μ time steps observations. $\boldsymbol{W}^o$ and $\boldsymbol{b}^o$ are trainable parameters responsible for linearly transforming the input data to align with the target output shape.

The chosen loss function for optimizing the model parameters $\boldsymbol{W}$ is the Mean Square Error (MSE), which quantifies the disparity between predicted and actual data. To circumvent overfitting, an L_2 regularization term is included. The combined loss function $L(\boldsymbol{W})$ is formally described as follows:

$$L(\boldsymbol{W}) = \mathbb{E}_X\left[\frac{1}{Q}\sum_{i=1}^{Q}\|\hat{\boldsymbol{X}}^{(i)}_{t+1:t+\tau} - \boldsymbol{X}^{(i)}_{t+1:t+\tau}\|_2^2\right] + \lambda\|\boldsymbol{W}\|_2^2, \tag{15}$$

where Q denotes the total number of heat meters, and λ is the regularization parameter. The loss for each meter is calculated separately, aggregated, and then averaged, constituting a comprehensive loss measure across all heat meter observations. The pseudo-code that describes the process of training SDGNN with meteorological factors (i.e. SDGNN*) is presented in Algorithm 1.

Algorithm 1: Pseudo-code for training SDGNN* in a batch instance

```
Input: The historical observations of district heat meters
       ($\boldsymbol{X}_{1:T}, \boldsymbol{X}_{T+1:\tau}$), initialize model $\mathcal{F}_\Theta$
Output: The trained model $\mathcal{F}_\Theta$
// Feed forward and backward updating
1  Trainer($\boldsymbol{X}_{1:T}, \mathcal{Z}_{1:T}, \boldsymbol{X}_{T+1:\tau}, \mathcal{F}_\Theta$):
2      foreach patch n in N do
3          $\tilde{\boldsymbol{X}}_n \in \mathbb{R}^{P\times Q}$ ← group the heat load observations by Eq. (2)
4          $\tilde{\mathcal{X}}_n \in \mathbb{R}^{1\times Q}$ ← encoder each segment of data by Eq. (3)
5      $\tilde{\mathcal{X}} = [\tilde{\mathcal{X}}_1 \parallel \cdots \parallel \tilde{\mathcal{X}}_N] \in \mathbb{R}^{N\times Q}$ ← aggregate data patches
6      $\tilde{\mathcal{S}} \in \mathbb{R}^{N\times N}$ ← calculate and normalize the similarity matrix by Eq. (4) and Eq. (5)
7      $\mathcal{G}_T$ ← graph construction based on $\tilde{\mathcal{X}}$ and $\tilde{\mathcal{S}}$
8      $\boldsymbol{R}^g$ ← capture the graph dynamics by Eq. (6) and Eq. (7)
9      $\boldsymbol{R}^c$ ← extract intricate features by Eq. (8)
10     $\boldsymbol{R}^{\alpha,o}$ ← enhance spatio-temporal representation using global attention mechanism by Eq. (9)–(11)
11     $[\boldsymbol{R}^{\alpha,o} \parallel \mathrm{Emb}(\mathcal{Z}_{1:T})]$ ← combine the linearly embedded meteorological factor features
12     $\boldsymbol{O}$ ← output alignment using fully connected networks by Eq. (13)
13     $\hat{\boldsymbol{X}}_{T+1:\tau}$ ← make prediction using $\boldsymbol{O}$ and $\boldsymbol{X}_{u:T}$ by Eq. (14)
14     Loss $\mathcal{L}$ ← $\hat{\boldsymbol{X}}_{T+1:\tau}$ and $\boldsymbol{X}_{T+1:\tau}$ using MSE by Eq. (15)
15     Backward using Adam optimizer
16     return $\mathcal{F}_\Theta$
```

Table 2
An example of heat load data for a month.

Day	Building A	Building B	Building C
1	10	15	8
2	12	17	9
3	11	16	10
⋮	⋮	⋮	⋮
30	9	14	7

Table 3
An example of patch data for the first three days.

Patch	Building A	Building B	Building C
1	{10, 12, 11}	{15, 17, 16}	{8, 9, 10}
2	{11, 13, 12}	{16, 18, 17}	{9, 10, 11}
⋮	⋮	⋮	⋮
10	{12, 14, 9}	{17, 19, 14}	{10, 11, 7}

Table 4
An example of the heat load predictions for the next three hours.

Day	Building A	Building B	Building C
Day	Building A	Building B	Building C
31	11	16	9
32	13	18	10
⋮	⋮	⋮	⋮
30+τ	12	17	11

3.4. Illustrative example

To further clarify the operation of the SDGNN model, we present a step-by-step walkthrough using a simplified example. Consider a scenario where we have daily heat load data from three buildings (A, B, and C) over a month (30 days). Our goal is to forecast the heat load for each building for the next τ day. Table 2 shows an example of the heat load data for the first 7 days.

Step 1. Patching. We divide the 30 days into patches of length $P = 3$ days, resulting in 10 patches. Each patch contains a 3-day sequence of heat load values for each building, as shown in Table 3.

Step 2. Sparse graph learning.

- **Temporal Sparsity**: Each patch from Table 3 becomes a node in our dynamic graph by linear embedding. This reduces the number of nodes from 30 to 10.
- **Relationship Sparsity**: We calculate the similarity between these patches using attention scores and the k-nearest neighbor algorithm. Let us assume that if Building A and C exhibit more similar heat load demands in Patch 1, we keep the connection between them and remove the connection with Building B. The algorithm would establish strong connections between their corresponding nodes in the graph.
- **Temporal Graph Learning**: The model uses graph convolution to learn how heat load patterns propagate through the network. For instance, if a cold front hits Building A in Patch 1, the model

might learn to predict increased heat load demands for Buildings A and C due to their assumed connection in the graph.

Step 3. Spatio-temporal memory enhancement. The spatio-temporal memory enhancement component further analyzes the graph's spatial and temporal relationships. Imagine building B consistently experiences higher heat load than buildings A and C. The convolution would highlight this spatial difference and focus more on the high demand period.

Step 4. Temporal fusion. The decoder transforms the learned features back into the original hourly heat load format. The skip connection combines this output with a linear transformation of the most recent heat load values. This ensures the final prediction considers both long-term patterns captured by the SDGNN and the immediate trends in the data.

By processing the data through these steps, the SDGNN model generates a τ-day ahead heat load forecast for each of the three buildings, as shown in Table 4.

4. Experiment

4.1. Experimental setup

To ensure consistency across all models, a uniform dataloader was employed, segmenting data into 70% (766 data points) for training, 10% (109 data points) for validation, and the remaining 20% (219 data points) for testing. The dataloader employed window lengths as $T \in \{15, 30, 45\}$ days, with a batch size of 16. Forecasting was carried out over continuous spans as $\tau \in \{3, 5, 7\}$ time steps, corresponding to half a week, a workweek, and a full week, respectively. Each model was trained using the Adam optimizer with the MSE as the loss function. The MSE served as the loss function. To enhance robustness, we performed each experiment five times using different random seeds, and the reported results represent the average values. The grid search method was applied to identify optimal hyperparameters. Specific settings for each model can be found in Table 10.

All models were implemented in the PyTorch v1.12.1 framework. Experiments were conducted on a server equipped with an Intel(R) Xeon(R) Gold 6226R CPU (2.90 GHz), 128 GB of RAM to handle large datasets and computational requirements, and a NVIDIA RTX A6000 48 GB GPU.

4.2. Data and processing

We collected two types of data for this study: weather data and district heating consumption data. The weather data, obtained from https://opendatadocs.dmi.govcloud.dk, includes four variables that may influence the heat consumption patterns of buildings. These variables encompass outdoor temperature, solar radiation intensity, wind speed, and relative humidity. The district heating consumption data were sourced from Zenodo at https://doi.org/10.5281/zenodo.6563114, containing hourly readings of smart heat meters from 3127 residential buildings in Aalborg, Denmark, over a three-year period (2018–2020). This data also includes contextual information such as dwelling type, construction year, and energy efficiency level of each building. We excluded 105 buildings that were unoccupied or lacked dwelling type information. Our focus was on single-family houses, terraced houses, and apartments, which make up 3021 buildings in the dataset. This dataset was previously utilized by us in another study [4], where we applied different methods for heat load prediction. In this study, we aim to improve the prediction accuracy and efficiency by employing a novel deep learning approach.

We divided the data into three sets: training, validation, and test. The training set comprised 70% of the data, the validation set 10%, and the test set the remaining 20%. We used the training set to train our model and update its parameters, the validation set to tune the hyperparameters and select the best model, and the test set to evaluate the final performance of our model and compare it with other models. While the dataset does not explicitly provide the physical layout of the district heating network, the location of each building within the district is implicitly considered as a spatial attribute. This spatial information, although not directly represented in the graph structure, influences the individual building heat load patterns and consequently, the learned temporal dependencies between buildings in our model.

The descriptive statistics and correlation coefficients of the heat load data and meteorological variables are summarized in Table 5. This table provides key insights into the input data used for our model. The statistics cover three types of buildings: single-family houses, terraced houses, and apartments, along with a general category for all buildings. The table also includes detailed statistics for four meteorological variables: outdoor temperature, solar radiation intensity, relative humidity, and wind speed. These variables are used as auxiliary inputs for the model. Table 5 shows that single-family houses constitute the majority of the dataset (81.4%), followed by terraced houses (15.7%) and apartments (2.9%). The descriptive statistics indicate significant variability in the heat load across different building types, with single-family houses exhibiting the highest mean heat load. Among the meteorological variables, outdoor temperature and solar radiation intensity show strong negative correlations with the heat load, as indicated by the Pearson and Spearman correlation coefficients. Relative humidity and wind speed also exhibit significant correlations, albeit to a lesser extent.

For the data processing, we employ a sliding window approach to generate input–output pairs for each smart meter. As illustrated in Fig. 5, for daily heating load prediction, each time point x is aggregated from 24 h of heating load, this approach uses the past T time steps of data to predict the heat load for the next τ time steps. This window size is chosen based on our empirical observation that heat load patterns tend to repeat daily. The sliding window approach effectively captures the temporal dependencies and seasonal variations in the heat load data. By incrementally shifting the window for each input–output pair, we can generate data pairs for each smart meter. Fig. 5 provides a detailed visualization of this process, showing how heat load observations and meteorological factors are used as inputs across T time steps to predict the heat load for the next τ time steps. This method is applied consistently across all three data sets: training, validation, and test.

4.3. Baselines

To evaluate the efficacy of the proposed SDGNN, we selected eight end-to-end prediction models for comparison. These include traditional statistical methods, conventional deep learning methods, static graph-based learning methods, and dynamic graph-based learning methods. The descriptions of these baselines are as follows:

(1) **HI** [43]: This model emphasizes the significance of recent data points in a time series and uses them directly for predictions.
(2) **LSTM** [44]: A recurrent neural network variant with memory cells and three distinct gates, adept at capturing long-term dependencies in sequential data.
(3) **TSFM** [45]: A vanilla transformer model designed for time series, proficiently capturing and highlighting key temporal patterns with its attention mechanism.
(4) **TGAT** [46]: This model combines a graph attention component with static graph relationships, enabling adaptive capture of temporal dependencies in time series prediction.
(5) **GDGCN** [47]: It employs a dynamic graph structure, complemented by a parameter-sharing mechanism and a distinctive temporal graph block for spatio-temporal time series prediction.

Table 5
The descriptive statistics and correlation coefficients of heat meters and meteorological factors. STD denotes the standard deviation, while PCC and SCC represent the Pearson correlation coefficient and Spearman's rank correlation coefficient, respectively.

Type	Symbol	Percent	Counts	Min	Max	Medium	Mean	STD	PCC	SCC
Target data	Single-family house	81.4	2459	0	462.328	42.975	49.685	37.341	–	–
	Terraced house	15.7	474	0	317.153	21.227	27.458	24.475	–	–
	Apartment	2.9	88	0	420.911	20.953	29.533	33.387	–	–
	All buildings	100	3021	0	462.328	37.11	45.61	36.526	–	–
Meteorological	Outdoor temperature	–	1	−7.883	23.712	8.45	9.159	6.138	−0.953***	−0.966***
	Solar radiation intensity	–	1	1.804	391.483	94.046	126.14	106.669	−0.709***	−0.738***
	Relative humidity	–	1	45.758	99.892	84.252	82.422	10.93	0.370***	0.424***
	Wind speed	–	1	1.15	12.829	4.685	4.996	2.059	0.102***	0.101***

*** indicates the p-value less than 0.001, signifying a highly significant correlation.

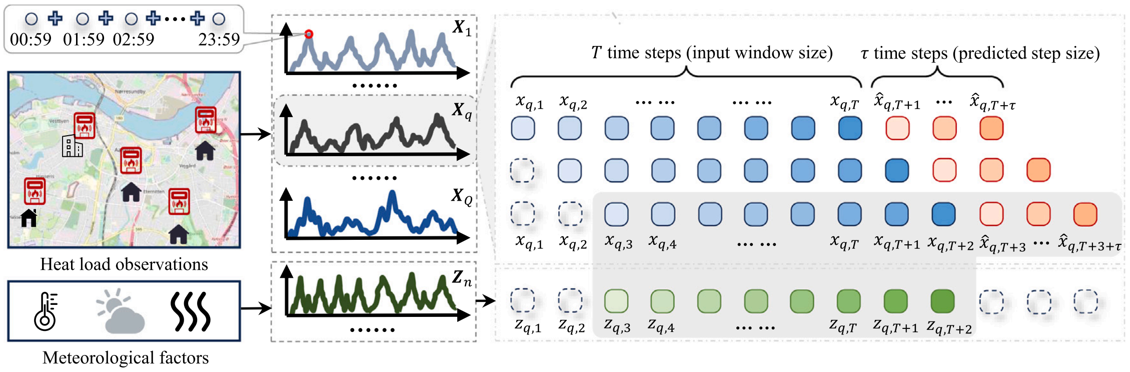

00:59 01:59 02:59 23:59
X_1
X_q
X_Q
Z_n
Heat load observations
Meteorological factors
T time steps (input window size)
τ time steps (predicted step size)
$x_{q,1}$ $x_{q,2}$ $x_{q,T}$ $\hat{x}_{q,T+1}$ $\hat{x}_{q,T+\tau}$
$x_{q,3}$ $x_{q,4}$ $x_{q,T+1}$ $x_{q,T+2}$ $\hat{x}_{q,T+3}$ $\hat{x}_{q,T+3+\tau}$
$z_{q,1}$ $z_{q,2}$ $z_{q,3}$ $z_{q,4}$ $z_{q,T}$ $z_{q,T+1}$ $z_{q,T+2}$

Fig. 5. The visualization of data organization.

(6) **AGCRN** [48]: This dynamic graph model takes into account both node-specific patterns and inter-dependencies within time series data, enabling the simultaneous learning of spatial and temporal correlations.
(7) **StemGNN** [49]: A spectral–temporal dynamic graph neural network captures inter-series links and temporal patterns in the spectral domain, merging graph and discrete Fourier transforms to process multivariate time series.
(8) **GWNet** [50]: It stands for GraphWaveNet, a model that combines wavelet transformations with graph convolution for effective spatial–temporal forecasting.

4.4. Evaluation metrics

Following prior studies [4,5], we employ three metrics to evaluate the performance of all methods for district heat load prediction, including Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and Coefficient of Variation of Root Mean Square Error (CVRMSE). Specifically, MAE measures the average absolute discrepancies, RMSE gives prominence to larger errors, and CVRMSE offers a relative assessment of model errors, facilitating comparisons across diverse datasets or scales. Such metrics are widely adopted in district heat load forecasting for their capacity to capture the error distribution and account for varying heat demand magnitudes [4]. The formulas for these metrics are provided below:

$$\mathrm{RMSE}(\hat{\boldsymbol{X}}, \boldsymbol{X}) = \sqrt{\frac{1}{Q \times L_{\text{test}}} \sum_{i=1}^{Q} \sum_{j=1}^{L_{\text{test}}} (\hat{X}_{i,j} - X_{i,j})^2}, \tag{16}$$

$$\mathrm{MAE}(\hat{\boldsymbol{X}}, \boldsymbol{X}) = \frac{1}{Q \times L_{\text{test}}} \sum_{i=1}^{Q} \sum_{j=1}^{L_{\text{test}}} |\hat{X}_{i,j} - X_{i,j}|, \tag{17}$$

$$\mathrm{CVRMSE}(\hat{\boldsymbol{X}}, \boldsymbol{X}) = \frac{\mathrm{RMSE}(\hat{\boldsymbol{X}}, \boldsymbol{X})}{\bar{\boldsymbol{X}}}, \tag{18}$$

where $\hat{X}$ represents the predicted values, X represents the actual values, and $\bar{X}$ signifies the mean of the actual values. Q denotes the count of heat meters, and L_{test} is the number of test samples. A lower metric value indicates superior model accuracy.

4.5. Comparison on multi-step prediction

Table 6 presents the multi-step forecasting performance of the proposed SDGNN in comparison to 8 baseline models on district heat load observations. Evaluations are conducted based on three distinct metrics, encompassing three input window sizes denoted as T, and three prediction steps. The results demonstrate that the SDGNN exhibits superior performance across the majority of metrics and horizons. Specifically, it surpasses other models with maximum improvements of up to 5.7%, 7.4%, and 5.7% over comparable models in terms of RMSE, MAE, and CVRMSE, respectively. This performance highlights the capability of SDGNN to efficiently capture intricate spatio-temporal patterns in historical heat load data.

As the window size T increases from 15 to 45, the prediction accuracy of most models tends to remain stable or slightly improve. This suggests that having a larger historical data window can help capture more relevant spatio-temporal patterns, enhancing forecasting accuracy. However, it is important to note that the improvement is not consistent across all models. Simply increasing the window size is not guaranteed for better performance, especially when the model's inherent architecture may not efficiently handle large-scale sequence data. Additionally, for short-term steps (e.g., 3 and 5), the SDGNN model consistently outperforms most of its counterparts, achieving optimal accuracy in all configurations. However, in scenarios with a limited input window size for long-term multi-step forecasting, SDGNN's performance lags behind models like GDGCN. This phenomenon underscores the inherent challenges of multi-step forecasting, especially as the forecast step extends. While the proposed SDGNN has strengths, there is room for improvement to address the complexities of long-term multi-step forecasting.

Among the baseline models, GDGCN and AGCRN perform better than the others, but they still lag behind the SDGNN models in most scenarios. This implies that using a dynamic graph structure to capture

Table 6
The multi-step performance comparison across three metrics and step lengths for district heat load observations with various window sizes T. The colors gray, green, and yellow denote the first, second, and third-best results, respectively. SDGNN* indicates that SDGNN considers meteorological factors.

T	Steps	Metrics	HI	LSTM	TSFM	TGAT	GDGCN	AGCRN	StemGNN	GWNet	SDGNN (ours)	Imp. (%)	SDGNN* (ours)	Imp. (%)
15	3	RMSE	9.468	20.324	9.762	11.254	8.152	9.051	9.072	11.060	7.984	+2.1	7.717	+5.3
		MAE	5.952	12.982	6.706	8.114	5.298	5.535	6.340	7.301	5.107	+3.6	5.047	+4.7
		CVRMSE	0.289	0.620	0.298	0.343	0.249	0.276	0.277	0.337	0.243	+2.1	0.235	+5.3
	5	RMSE	10.382	20.729	10.038	12.151	8.751	9.124	9.288	12.448	8.597	+1.8	8.249	+5.7
		MAE	6.647	13.395	6.989	9.003	5.808	5.754	6.511	8.749	5.594	+2.8	5.404	+6.1
		CVRMSE	0.319	0.636	0.308	0.373	0.269	0.280	0.285	0.382	0.264	+1.8	0.253	+5.7
	7	RMSE	11.210	20.460	10.216	12.340	9.192	9.402	9.576	13.011	9.209	-0.2	8.574	+6.7
		MAE	7.241	13.165	7.095	9.181	6.072	6.158	6.824	9.207	6.031	+0.7	5.762	+5.1
		CVRMSE	0.347	0.633	0.316	0.382	0.284	0.291	0.296	0.402	0.285	-0.2	0.265	+6.7
30	3	RMSE	9.468	21.989	9.927	12.829	8.424	8.509	9.396	10.751	8.002	+5.0	7.845	+6.9
		MAE	5.952	14.105	6.867	9.109	5.526	5.415	6.298	7.231	5.102	+5.8	5.024	+7.2
		CVRMSE	0.289	0.671	0.303	0.391	0.257	0.259	0.287	0.328	0.244	+5.0	0.239	+6.9
	5	RMSE	10.382	22.974	10.144	12.975	8.785	9.052	9.488	12.577	8.608	+2.0	8.312	+5.4
		MAE	6.647	14.862	7.035	9.233	5.798	5.700	6.452	8.739	5.585	+2.0	5.392	+5.4
		CVRMSE	0.319	0.705	0.311	0.398	0.270	0.278	0.289	0.386	0.264	+2.0	0.255	+5.4
	7	RMSE	11.210	21.824	10.063	13.895	9.484	9.425	9.602	12.366	9.117	+3.3	8.557	+9.2
		MAE	7.241	14.382	6.990	9.728	6.389	6.192	6.734	8.964	5.978	+3.5	5.738	+7.3
		CVRMSE	0.347	0.675	0.311	0.430	0.293	0.292	0.297	0.383	0.282	+3.3	0.265	+9.2
45	3	RMSE	9.468	21.862	10.054	18.844	8.463	8.560	9.053	11.488	7.983	+5.7	7.823	+7.6
		MAE	5.952	13.899	7.026	12.883	5.678	5.515	6.326	7.731	5.107	+7.4	5.015	+9.1
		CVRMSE	0.289	0.667	0.307	0.575	0.258	0.261	0.276	0.350	0.243	+5.7	0.239	+7.6
	5	RMSE	10.382	21.821	10.051	17.593	8.708	8.931	9.422	11.832	8.597	+1.3	8.294	+4.8
		MAE	6.647	14.056	6.942	12.886	5.832	5.758	6.591	8.213	5.597	+2.8	5.391	+6.4
		CVRMSE	0.319	0.670	0.309	0.540	0.267	0.274	0.289	0.363	0.264	+1.3	0.255	+4.8
	7	RMSE	11.210	21.710	9.990	18.737	9.419	9.356	9.648	12.409	9.108	+2.7	8.642	+7.6
		MAE	7.241	14.155	6.939	14.889	6.389	6.237	6.761	8.699	5.991	+4.0	5.719	+8.3
		CVRMSE	0.347	0.672	0.309	0.580	0.291	0.289	0.298	0.384	0.282	+2.7	0.267	+7.6

inter-dependencies seems to be effective in multi-step forecasting. The HI model, which emphasizes recent data points, typically performs in the middle or lower tier of baseline models. This suggests that while giving weight to recent data points can be beneficial, it might not be sufficient to capture the complex spatio-temporal patterns inherent in district heat load observations. Across various configurations, LSTM's performance is consistently worse. This indicates that while LSTM can capture temporal sequences effectively, it struggles with the spatial aspect inherent in the district heat load data. While TSFM outperforms LSTM in most cases, it does not consistently rank among the top-tier models. The strength of TSFM lies in their attention mechanisms, emphasizing significant temporal patterns. Nevertheless, the district heat load observations demand an intricate balance of spatial and temporal considerations, while it captures temporal dependencies to a degree, it may struggle to integrate spatial correlations.

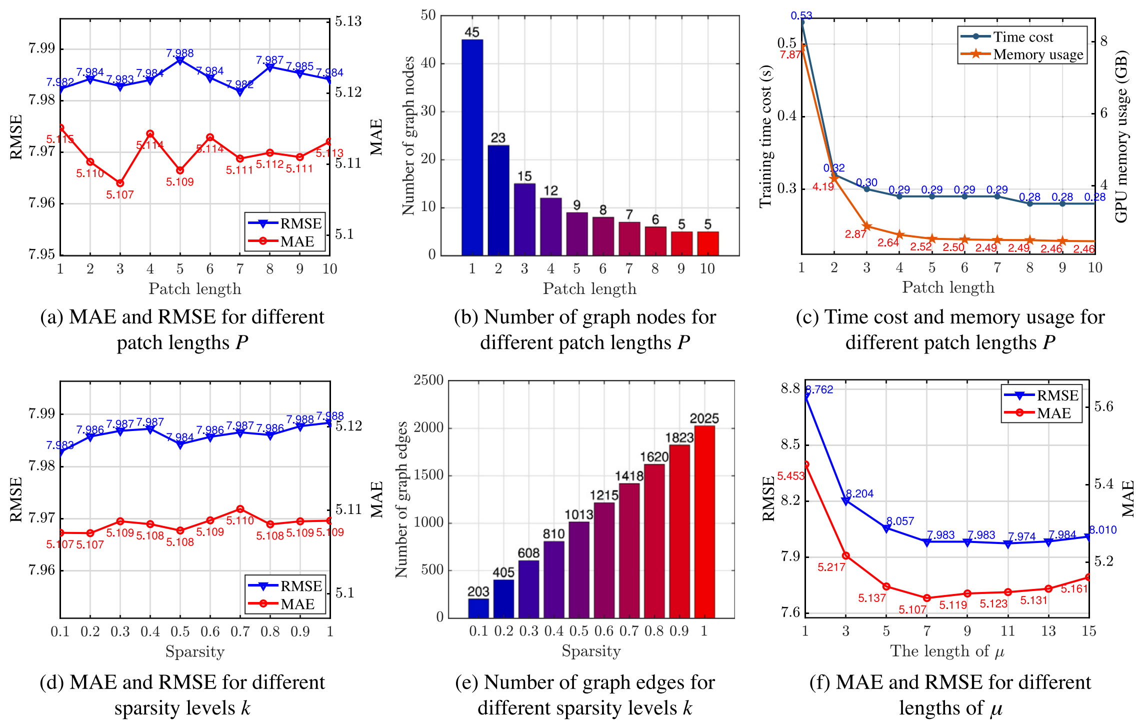
By incorporating meteorological factors into SDGNN results in the modified model SDGNN*, which consistently demonstrates marked improvements in performance over the original SDGNN. The enhancements in metrics such as CVRMSE and RMSE reach up to 9.2% compared to the optimal baseline methods. This underscores the importance of integrating meteorological information in district heat load forecasting and highlights the robustness and adaptability of the SDGNN framework. The significant performance gains with SDGNN* validate the model's ability to effectively fuse exogenous information, enhancing its predictive capabilities.

4.6. Sensitivity analysis

To analyze the impact of different parameter or variable values on our model's performance, we conduct a sensitivity analysis by adjusting one parameter or variable at a time and observing the resultant performance changes. The input window size T is fixed at 45, and the prediction step τ is set to 3. Fig. 6 shows the sensitivity analysis results for three key parameters: patch length P, sparsity k, and the length of μ.

Fig. 6(a)-(c) display the sensitivity analysis results for the patch length P, which denotes the length of time slots in the temporal sparsity process. In Fig. 6(a), both MAE and RMSE are used as metrics for model accuracy. The results indicate slight fluctuations in accuracy across different patch lengths. The patch length of 3 results in the lowest MAE, while patch lengths of 1 and 7 achieve the lowest RMSE, with patch length 3 having the second-best RMSE. Due to the minimal fluctuation in MAE and RMSE values across different patch lengths, sparsifying the graph based on patch perspective is deemed feasible. Fig. 6(b) illustrates the number of graph nodes, representing the count of time slots following temporal sparsity. This inverse relationship aligns with the patching strategy, where larger patches result in fewer, more aggregated nodes within the graph. Fig. 6(c) shows the training time cost and GPU memory usage across different patch lengths. The training time cost sharply decreases as the patch length increases from 1 to 3 and then stabilizes at around 0.29 seconds for patch lengths greater than 4. GPU memory usage also decreases with increasing patch length, settling at approximately 2.5G for patch lengths 5 and beyond. This reduction in computational resources with increased patch length can be attributed to the decreased complexity of the graph, with fewer nodes requiring less memory and computational costs. We chose a patch length of 3 for the experiment because it optimally balances granularity, accuracy, and computational resource utilization in processing.

Fig. 6(d)-(e) illustrate the relationship between the degree of sparsity in the graph and the prediction accuracy metrics MAE and RMSE. In Fig. 6(d), the errors show minor fluctuations as the sparsity increases, indicating the model's stability in accuracy across varying degrees of graph sparsity. Fig. 6(e) shows the number of edges in the graph significantly reduces as the sparsity increases from 0.1 to 1. This reduction implies a more concise and efficient graph representation, particularly beneficial for computational efficiency and resource conservation. Therefore, the sparsity is set at 0.1 for the experiment. Utilizing fewer relationships in the graph ensures computational efficiency while maintaining commendable accuracy.



(a) MAE and RMSE for different patch lengths P

(b) Number of graph nodes for different patch lengths P

(c) Time cost and memory usage for different patch lengths P

(d) MAE and RMSE for different sparsity levels k

(e) Number of graph edges for different sparsity levels k

(f) MAE and RMSE for different lengths of μ

Fig. 6. Sensitivity analysis for different model parameters. (a)-(c) Sensitivity analysis of patch length P. (d)-(e) Sensitivity analysis of sparsity k. (f) Sensitivity analysis of the length of μ.

Fig. 6(f) explores the effect of different values of μ (context window) on prediction accuracy. Both RMSE and MAE initially decrease sharply as μ increases from 1 to 5, indicating improved prediction accuracy with longer context windows. The metrics achieve their lowest values at $\mu = 7$. Beyond this point, the sensitivity diminishes, and the accuracy stabilizes, suggesting that additional context beyond this point does not significantly enhance accuracy. This could be because the model has already detected the primary heat load patterns, and further context either becomes superfluous or introduces redundancy. These insights highlight the importance of tuning μ to an optimal length to extract the maximum predictive efficacy from the model. All corresponding experiments are conducted using the optimal window size settings for the proposed SDGNN.

4.7. Ablation analysis

Table 7 presents an ablation analysis that demonstrates the relative contributions of each component to the overall performance of the proposed SDGNN model. We evaluated five variants of our SDGNN model. *w/o* TS removes the temporal sparsity. *w/o* Graph removes the total sparse graph learning unit. *w/o* Conv removes the spatio-temporal convolution component. *w/o* Attn removes the global attention mechanism, while *w/o* Skip removes the skip connection architecture before output.

Across different step values, the SDGNN model consistently exhibits robust performance, consistently ranking either first or within the top three across all metrics. The temporal sparsity intends to compress the graph's nodes to reduce complexity without sacrificing accuracy. The similar performance of SDGNN with and without this process indicates its effective node sparsification, maintaining accuracy with fewer computations. The graph structure in SDGNN is designed to capture spatial dependencies inherent in the data. When this graph component is removed, there is a slight degradation in the model's performance in certain scenarios. This behavior underscores the importance of the graph structure. However, in some cases, the removal of the graph leads to better accuracy. This could be attributed to the graph embedding process emphasizing not only useful patterns but also inadvertently accentuating noise, potentially impacting the prediction accuracy. In the absence of the convolution component, performance slightly declines, emphasizing the crucial role of the convolutional layer in refining spatio-temporal features. In long-term forecasting, attention mechanisms can introduce unnecessary complexity and computational overhead, potentially overemphasizing noise. Models without attention often yield more stable predictions, prioritizing broad trends over short-term fluctuations. Notably, bypassing the skip connections significantly decreases performance across all metrics. This decline highlights the essential role of skip connections in maintaining efficient gradient flow and retaining key features.

4.8. Study of dynamic graph structure

To investigate the dynamics of the proposed graph learning component, we randomly select a heat meter to visualize the evolving dynamic graph. As shown in Fig. 7, the internal structure of the graph undergoes transformations across different epochs for the same data segment. During the period spanning Epoch 10 to 50, the model tends to establish expansive connections, reflecting its exploration of potential relationships within the data. This dynamic evolution signifies the model's adaptation to heat load patterns and the establishment of temporal dependencies. As the model evolves, it likely identifies and prunes less pertinent connections. After Epoch 50, the model attains a stable representation of the relationships, resulting in minimal alterations to the graph structure.

Fig. 8 illustrates the model's sensitivity to different data instances, with each instance representing a single time series segment from the same heat meter. The variations in the graph structure highlight the model's capacity to adapt its representation to distinct temporal characteristics. This adaptability ensures the model can accurately capture the sparse and dynamic relationships inherent in diverse time series segments.

Table 7
The ablation analysis of the proposed SDGNN. Gray and green represent the best and second-best results, respectively.

Steps	Metrics	SDGNN	*w/o* Patch	*w/o* Graph	*w/o* Conv	*w/o* Attn	*w/o* Skip
3	RMSE	7.9828	7.9782	7.9874	8.0078	7.9846	16.2531
	MAE	5.1073	5.1161	5.1174	5.1262	5.1196	11.7689
	CVRMSE	0.2434	0.2433	0.2436	0.2442	0.2435	0.4956
5	RMSE	8.5970	8.5971	8.6023	8.6141	8.6061	17.3060
	MAE	5.5968	5.6073	5.6094	5.6077	5.6055	12.3263
	CVRMSE	0.2640	0.2640	0.2641	0.2645	0.2642	0.5314
7	RMSE	9.1082	9.1122	9.1156	9.1153	9.1033	18.9640
	MAE	5.9909	5.9961	6.0014	5.9986	5.9877	13.5059
	CVRMSE	0.2818	0.2819	0.2820	0.2820	0.2816	0.5866

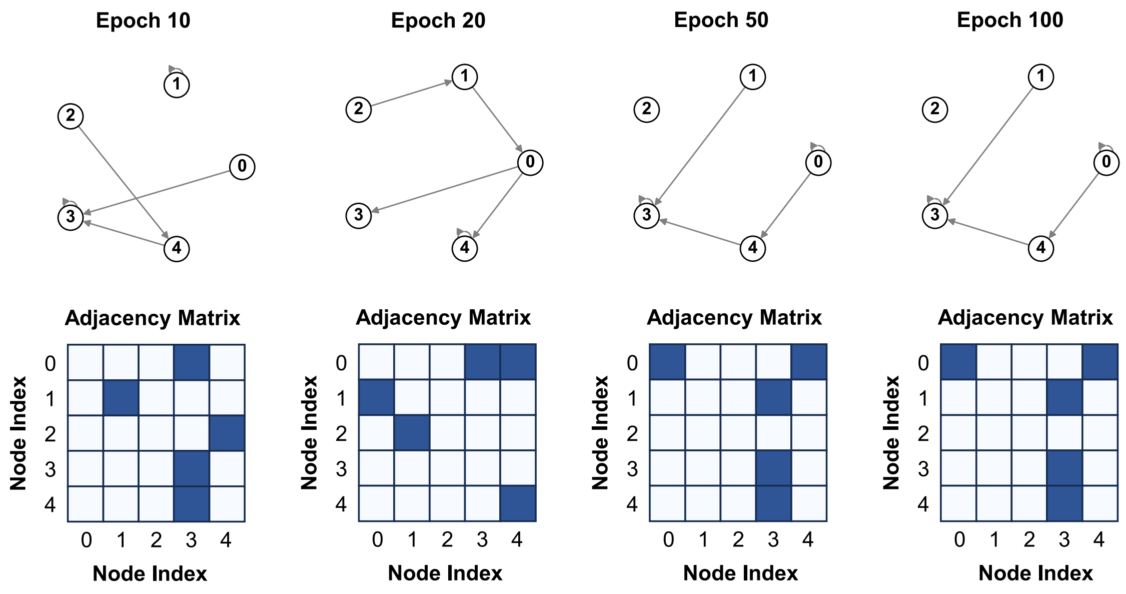


Fig. 7. The dynamic graph changes across epochs.

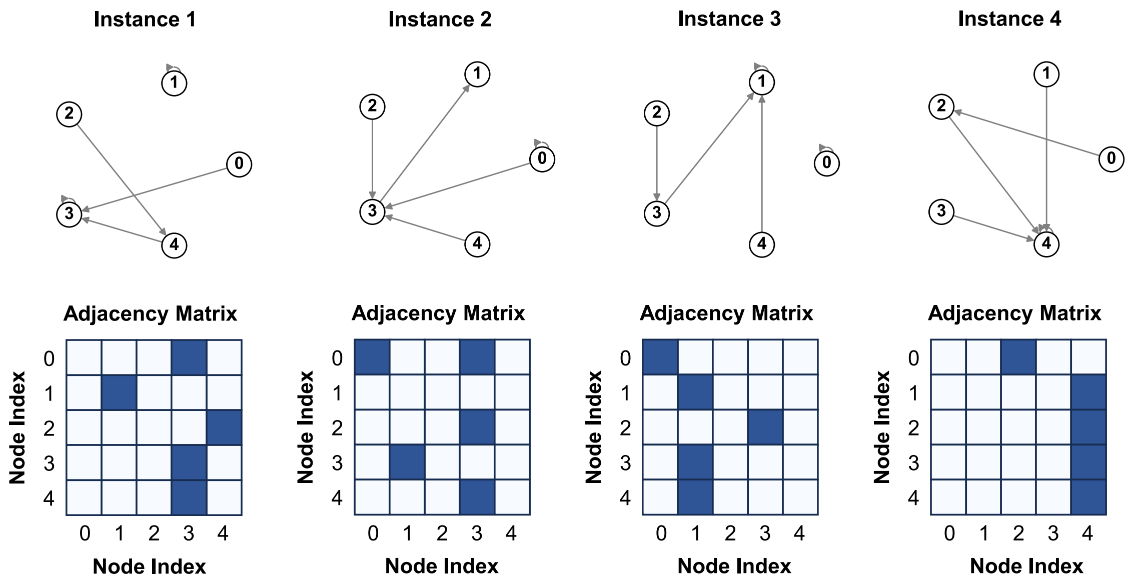
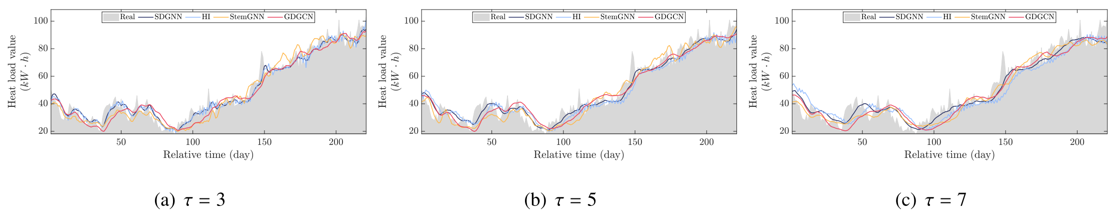
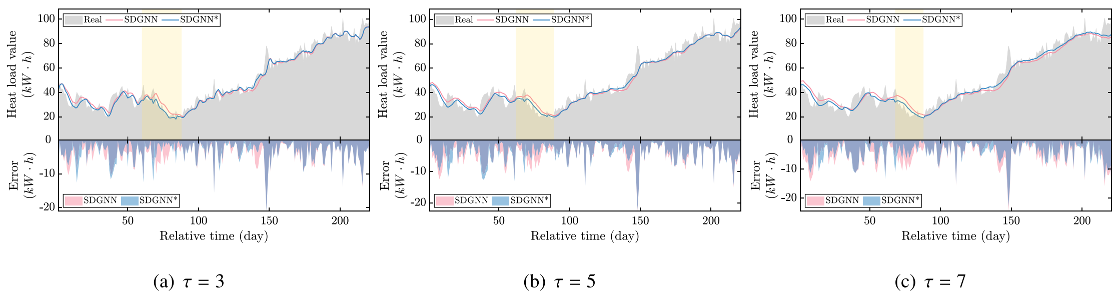


Fig. 8. The dynamic graph changes across batch instances.



(a) τ = 3 (b) τ = 5 (c) τ = 7

Fig. 9. The visualizations comparing the real values with the SDGNN predictions and the three other benchmark predictions; Unit of τ: day.



(a) τ = 3 (b) τ = 5 (c) τ = 7

Fig. 10. The visualizations among real values, SDGNN predictions, and SDGNN* predictions and prediction error; Unit of τ: day.

4.9. Study of prediction performance

Fig. 9 presents the actual heat load values in comparison with the predictions made by SDGNN, HI, StemGNN, and GDGCN. SDGNN maintains promising performance in tracking the progress of heat load across different prediction steps τ, especially during low-demand periods when the prediction step *tau* is set to 3 or 5. As the prediction step τ increases, distancing further from known data points, uncertainties and complexities in data trends intensify. The capability of the prediction model to capture peak values gradually decreases. When compared to the HI model, which emphasizes recent data points, SDGNN demonstrates a more balanced approach, ensuring accuracy across various time spans without over-relying on recent data. GDGCN and StemGNN also closely match the actual data. This observation emphasizes the effectiveness of dynamic graph structures in heat load forecasting.

To further compare SDGNN and SDGNN*, we visualize heat load predictions for various prediction step lengths (i.e., 3, 5, and 7) in Fig. 10. The input window size remains fixed at 45. These results demonstrate that SDGNN*, which incorporates meteorological factors, closely aligns with real heat load observations compared to SDGNN, particularly during the highlight period. This performance disparity is particularly noticeable when the prediction step τ is set to 3 or 5. By incorporating meteorological factors, SDGNN* demonstrates improved accuracy in capturing stable load variations, particularly during medium and low-demand periods. However, during periods of high heat load demand, the difference in accuracy between SDGNN* and SDGNN appears minimal. A possible reason for this observation is that heat load might be influenced by a combination of factors during high-demand periods, such as user behavior, equipment efficiency, building insulation, etc. Consequently, the additional meteorological information incorporated into SDGNN* may not provide a significant advantage over SDGNN in these specific steps. This phenomenon highlights the intricate nature of district heat load in long-term multi-step forecasting and high-demand periods.

4.10. Evaluation of results

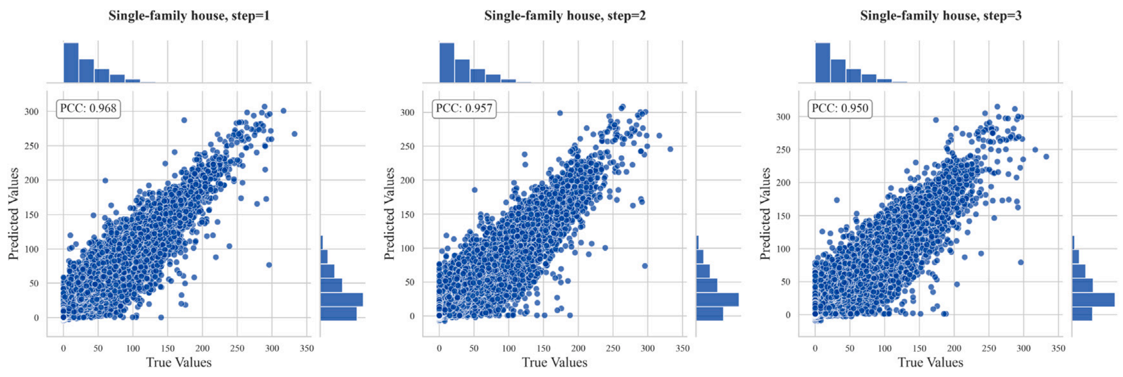
To ensure the physical plausibility of our energy consumption predictions, we compare the distributions of the predicted and actual heat load values. This comparison helps validate that our model accurately captures the variability and patterns in the energy demand. Fig. 11 presents scatter plots of the actual vs. predicted heat load values for the three building types, including single-family houses, terraced houses, and apartments, across three prediction steps. Each subplot includes scatter plots showing the correlation between predicted and actual values, as well as the PCC to quantify the accuracy of the predictions. The window size T is fixed at 45, and the prediction step τ is set to 3.

- *Single-family houses:* For single-family houses, the PCC values for steps 1, 2, and 3 are 0.984, 0.977, and 0.950 respectively, indicating a high degree of correlation between the predicted and actual heat loads. The scatter plots show a strong linear relationship, confirming that the predicted values closely match the actual distribution. This validates the model's ability to capture the heat load patterns for single-family houses effectively.
- *Terraced houses:* For terraced houses, the PCC values for steps 1, 2, and 3 are 0.983, 0.974, and 0.939 respectively. The scatter plots and corresponding PCC values demonstrate that the predicted heat loads are closely aligned with the actual values, with slight deviations becoming more noticeable at step 3. This indicates that while the model performs well, the accuracy slightly decreases with longer prediction horizons.
- *Apartments:* For apartments, the PCC values for steps 1, 2, and 3 are 0.984, 0.975, and 0.939 respectively. The scatter plots illustrate a strong linear correlation, indicating that the model's predictions are well-aligned with the actual heat load distributions. However, similar to terraced houses, the predictions for apartments show a slight decrease in accuracy at step 3.

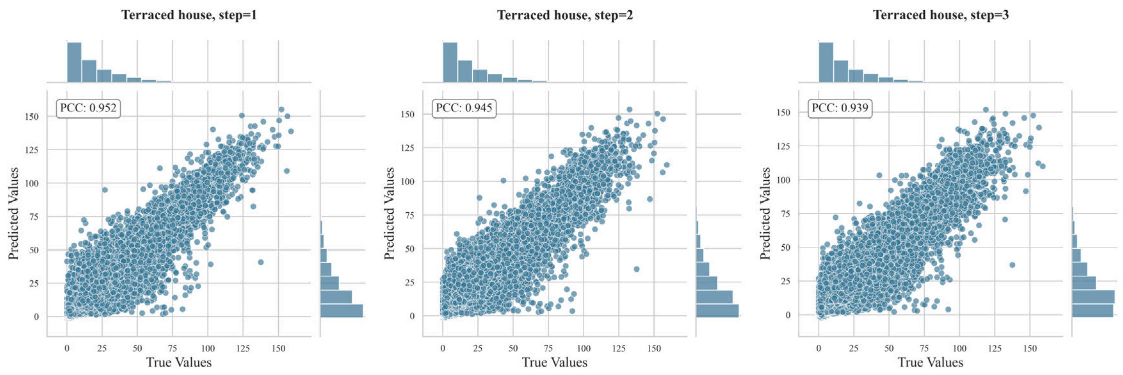
Therefore, Fig. 11 demonstrates that the model effectively captures the overall distribution and variability of the energy demand across different building types and prediction steps. The high PCC values indicate that the model's predictions are highly correlated with the actual heat load values, validating its accuracy and reliability in forecasting energy consumption.

5. Conclusions and future work

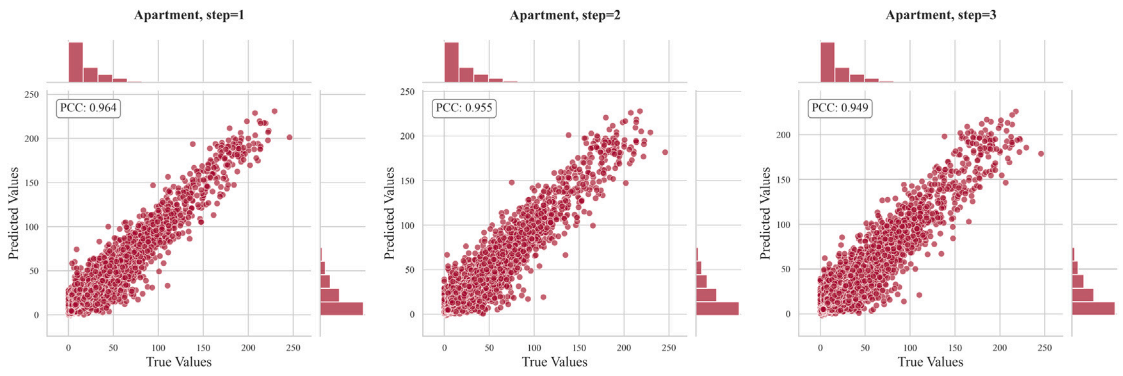
This paper proposed a novel sparse dynamic graph neural network (SDGNN) for district heat load forecasting, a pivotal concern in the realm of urban energy management. SDGNN can capture the complex and dynamic spatio-temporal patterns in heat load data by learning a sparse and adaptive graph structure. SDGNN also incorporates meteorological factors as exogenous inputs to enhance its predictive



(a) Single-family house



(b) Terraced house



(c) Apartment

Fig. 11. Scatter plots comparing the actual and predicted heat load values for different building types.

performance. We conducted extensive experiments on a real-world dataset and compared SDGNN with several state-of-the-art baselines. The results showed that SDGNN outperformed the baselines in terms of accuracy, efficiency, and robustness. We also performed sensitivity, ablation, and visualization analyses to demonstrate the effectiveness and interpretability of SDGNN.

The experiments revealed several key findings that support the superiority of SDGNN over other models. First, SDGNN achieved superior performance across the majority of metrics and horizons, surpassing other models with maximum improvements of up to 5.7% in RMSE, 7.4% in MAE, and 5.7% in CVRMSE. Second, SDGNN demonstrated a balanced blend of computational efficiency and predictive performance, ranking competitively in efficiency metrics and outperforming other models in accuracy metrics. Third, SDGNN exhibited robustness and stability across different parameter or variable values, indicating its adaptability to diverse data characteristics. Fourth, SDGNN showed its capability to efficiently capture intricate spatio-temporal patterns by learning a sparse and dynamic graph structure that evolves across epochs and instances. Fifth, SDGNN enhanced its predictive performance by integrating meteorological factors as exogenous inputs, achieving marked improvements over the original SDGNN.

The theoretical contributions of this work lie in the innovative integration of dynamic graph structures to capture inter-dependencies, emphasizing the importance of balancing spatial and temporal considerations. From a practical standpoint, these advancements hold significant implications for urban planners and energy managers, offering them a robust tool for more accurate and efficient heat load forecasting.

For future work, we plan to extend SDGNN to handle multivariate time series forecasting, where multiple types of energy loads are considered simultaneously. We also intend to explore other ways of integrating exogenous information, such as user behavior, building characteristics, and equipment efficiency, into SDGNN. Additionally, we will explore incorporating multiple meteorological data sources to better capture local variations in weather conditions, such as wind speed and solar radiation, which will improve the model's generalizability to different climate zones and larger district heating systems. Moreover, we aim to apply SDGNN to other domains that involve spatio-temporal data, such as traffic flow forecasting, air quality prediction, and epidemic modeling.

CRediT authorship contribution statement

Yaohui Huang: Writing – review & editing, Writing – original draft, Software, Methodology, Data curation, Conceptualization. **Yuan Zhao:** Writing – review & editing, Writing – original draft, Validation, Methodology, Conceptualization. **Zhijin Wang:** Writing – review & editing, Writing – original draft, Project administration, Methodology, Funding acquisition, Conceptualization. **Xiufeng Liu:** Writing – review & editing, Writing – original draft, Project administration, Methodology, Formal analysis, Conceptualization. **Yonggang Fu:** Writing – review & editing, Writing – original draft, Validation, Methodology, Investigation.

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

The authors do not have permission to share data.

Acknowledgments

This research was supported in part by the Fujian Province Natural Science Foundation of Fujian Province (CN) (nos. 2021J01857, 2022J01820, and 2022J01335), Education Department of Fujian Province (CN) (nos. JAT210224 and JAT231193), and REST4CITY (no. 101075582) funded by the European Union.

Appendix A. Additional experiments

The effectiveness of our proposed method, SDGNN, was validated using the JERICHO-E-usage dataset [51], available at https://doi.org/10.6084/m9.figshare.c.5245457.v1. This dataset includes hourly energy consumption patterns from 38 regions in Germany throughout 2019, with district space heating consumption data categorized by different energy types (e.g., residential, industrial, commerce). The data resolution was aggregated from hourly to daily for consistency with the district heating data in Aalborg, Denmark. The data was divided into 70% for training, 10% for validation, and 20% for testing. The input window size T is set as 45. The experimental results are presented in Table 8, which evaluates the methods across three metrics (RMSE, MAE, CVRMSE) and three different time horizons (3, 5, 7 steps) for district heat load observations in these three categories.

The experiment results demonstrate that SDGNN consistently outperformed other methods, achieving the best performance in most cases. Specifically, SDGNN had the lowest RMSE, MAE, and CVRMSE across multiple time horizons and categories. The superior performance of SDGNN demonstrates its robustness and effectiveness in capturing temporal dynamics and generalizing across different heat load prediction tasks. In comparison, due to the limited training data, the performance of almost graph-based models is generally modest. The GDGCN perform relatively well due to their ability to capture both linear and nonlinear heat load patterns. The lightweight nature of HI model allowed it to adapt quickly to the available data without overfitting, making it effective for short-term predictions where recent data trends are more indicative of future patterns. Overall, the results highlight the robustness and superiority of SDGNN, especially in handling long-term dependencies and generalizing across different scenarios despite data limitations.

Appendix B. Comparison on computational efficiency

Table 9 compares the efficiency and performance of various models on district heat load observations. To maintain consistency in the evaluation, each model was subjected to a fixed step, denoted as $\tau = 3$, and a window size of $T = 15$. These settings ensure a standardized comparison across all models. The assessment utilizes metrics that cover both computational efficiency (e.g., MACs, FLOPS, latency, training time, and GPU memory usage) and predictive performance (e.g., RMSE, MAE, and CVRMSE).

The metric of Multiply-Accumulate Operations (MACs) is used to measure the computational cost of each model. A lower MACs value implies enhanced efficiency. TSFM achieves the lowest computational cost at 0.001G, followed closely by LSTM and our SDGNN model. In terms of Forward FLOPS, TSFM demonstrates high computational efficiency, followed by LSTM and SDGNN. Forward latency evaluates the speed at which a model completes a forward pass, with a lower value being preferable. LSTM exhibits the fastest latency, closely followed by TSFM and SDGNN. In contrast, both AGCRN and StemGNN demonstrate significantly higher latency durations. Moreover, LSTM demonstrates the fastest training speed, and TSFM is the most memory-efficient model. In contrast, GDGCN and AGCRN have considerably higher memory usage and require more time cost.

A balance exists between computational efficiency and predictive capability. While LSTM is computationally efficient, its prediction accuracy lags behind that of other models. The proposed SDGNN model offers a balanced blend of computational efficiency and predictive performance, outperforming other models in accuracy metrics and ranking competitively in efficiency metrics. Notably, models such as AGCRN and GDGCN, despite their robust predictive strengths, are associated with considerable computational costs, diminishing their suitability for real-time applications.

Appendix C. Convexity and boundedness of the loss function

We consider the loss function $L(\boldsymbol{W})$ defined in Eq. (15), where $\boldsymbol{W}$ is the weight matrix of SDGNN. We prove that $L(\boldsymbol{W})$ is convex and bounded by showing that it satisfies the following properties:

(1) **First-Order Condition**: The loss function $L(\boldsymbol{W})$ satisfies the first-order condition for convex functions, which states:

$$L(\theta \boldsymbol{W}_1 + (1-\theta)\boldsymbol{W}_2) \leq \theta L(\boldsymbol{W}_1) + (1-\theta) L(\boldsymbol{W}_2), \tag{19}$$

for all $\theta \in [0,1]$ and any two weight matrices $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$.

To prove this, we use the fact that $L(\boldsymbol{W})$ is a linear combination of convex functions, namely MSE. Since a linear combination of convex functions is also convex, we can apply the first-order condition to each term separately and then add them up to obtain the desired result.

(2) **Second-Order Condition**: The Hessian matrix $\nabla^2 L(\boldsymbol{W})$ is positive semi-definite, which also indicates convexity. To prove this, we use the fact that $L(\boldsymbol{W})$ is twice differentiable with respect to $\boldsymbol{W}$, and we calculate the Hessian matrix as follows:

$$\nabla^2 L(\boldsymbol{W}) = 2(\nabla^2 \mathrm{MSE}(\hat{\boldsymbol{X}}, \boldsymbol{X})), \tag{20}$$

where $\hat{\boldsymbol{X}} = f(\boldsymbol{X}; \boldsymbol{W})$ is the output of SDGNN given the input data matrix $\boldsymbol{X}$. Using the chain rule and the definitions of MSE, we can simplify the Hessian matrix as:

$$\nabla^2 L(\boldsymbol{W}) = 4(\frac{\partial f}{\partial \boldsymbol{W}})^T(\frac{\partial f}{\partial \boldsymbol{W}}), \tag{21}$$

where $\bar{\boldsymbol{X}} = mean(\boldsymbol{X})$ is the mean value of the input data matrix. Since both terms are positive semi-definite matrices, their sum is also positive semi-definite, which proves the second-order condition.

(3) **Bounding Norms**: We impose an upper bound $M > 0$ on the Frobenius norm of the weight matrix, $\|\boldsymbol{W}\|_F \leq M$, to show that the loss function is bounded.

To prove this, we use the fact that MSE is bounded functions, since they are non-negative and have finite upper bounds. For example,

Table 8
The multi-step performance comparison across three metrics and step lengths for district heat load observations in three category collections on the JERICHO-E-usage dataset. The colors gray and green denote the first, second-best results, respectively. The MAE and RMSE values have been scaled down by $1e6$ for easier comparison.

Type	Steps	Metrics	HI	LSTM	TSFM	TGAT	GDGCN	AGCRN	StemGNN	GWN	SDGNN
Residence	3	RMSE	17.104	49.367	32.362	45.462	18.607	24.823	34.353	31.932	13.613
		MAE	10.489	42.441	24.247	37.919	13.109	18.917	24.954	23.913	8.842
		CVRMSE	0.403	1.163	0.763	1.071	0.438	0.585	0.809	0.752	0.321
	5	RMSE	18.020	48.999	41.634	40.717	22.015	22.763	33.960	27.371	14.241
		MAE	11.185	42.062	33.530	32.573	16.329	16.933	24.641	19.647	9.324
		CVRMSE	0.428	1.165	0.990	0.968	0.523	0.541	0.807	0.651	0.339
	7	RMSE	18.345	48.645	40.445	41.609	27.166	24.361	33.577	28.606	14.606
		MAE	11.458	41.688	32.238	33.586	21.405	18.270	24.215	20.322	9.510
		CVRMSE	0.440	1.167	0.970	0.998	0.652	0.584	0.805	0.686	0.350
Industry	3	RMSE	1.330	4.061	1.793	1.668	1.077	1.484	2.659	1.148	0.614
		MAE	0.811	3.199	1.073	1.070	0.726	0.952	1.537	0.734	0.366
		CVRMSE	0.416	1.269	0.560	0.521	0.337	0.464	0.831	0.359	0.192
	5	RMSE	1.256	4.052	1.765	1.661	1.106	1.507	2.305	1.350	0.625
		MAE	0.759	3.193	1.065	1.072	0.748	1.001	1.336	0.790	0.376
		CVRMSE	0.393	1.269	0.553	0.520	0.346	0.472	0.722	0.423	0.196
	7	RMSE	0.738	4.047	1.784	1.630	0.999	1.563	2.390	1.331	0.635
		MAE	0.455	3.190	1.072	1.065	0.691	1.033	1.371	0.829	0.385
		CVRMSE	0.231	1.269	0.559	0.511	0.313	0.490	0.749	0.417	0.199
Commerce	3	RMSE	4.999	15.113	8.025	7.925	4.072	5.260	9.440	4.679	2.535
		MAE	3.386	12.925	5.526	5.552	3.153	3.891	6.525	3.462	1.655
		CVRMSE	0.387	1.169	0.621	0.613	0.315	0.407	0.730	0.362	0.196
	5	RMSE	4.663	15.064	8.137	8.156	4.002	5.623	10.306	4.886	2.584
		MAE	3.114	12.891	5.602	5.709	3.115	4.133	7.309	3.722	1.680
		CVRMSE	0.362	1.169	0.631	0.633	0.310	0.436	0.799	0.379	0.200
	7	RMSE	2.910	15.040	13.875	6.510	4.211	5.064	9.345	4.956	2.591
		MAE	1.908	12.879	11.506	4.842	3.295	3.659	6.497	3.507	1.710
		CVRMSE	0.226	1.168	1.077	0.505	0.327	0.393	0.726	0.385	0.201

Table 9
The comparison of efficiency and performance for proposed and baseline methods. The step τ and window size T are fixed at 3 and 15, respectively. Gray, green, and yellow denote the first, second, and third-best results.

Model	Forward MACs (GMACs)	Forward FLOPS (GFLOPs)	Forward latency (ms)	Training time cost (s)	GPU memory usage (GB)	RMSE	MAE	CVRMSE
HI	–	–	–	–	–	9.468	5.952	0.289
LSTM	0.005	0.27	1.48	0.29	1.99	20.324	12.982	0.620
TSFM	0.001	0.002	2.68	0.30	1.72	9.762	6.706	0.298
TGAT	0.17	0.35	13.82	1.18	2.01	11.254	8.114	0.343
GDGCN	0.15	19.55	26.65	26.29	27.66	8.152	5.298	0.249
AGCRN	1.67	117.42	858.52	45.61	27.28	9.051	5.535	0.276
StemGNN	111.67	389.66	513.81	62.29	24.32	9.072	6.340	0.277
GWNet	1.68	82.84	26.4	18.94	13.40	11.060	7.301	0.337
SDGNN (ours)	0.01	0.02	3.99	0.32	2.16	7.984	5.107	0.243

MSE has an upper bound of $\|\hat{\boldsymbol{X}} - \boldsymbol{X}\|_F^2 / n_m n_n$, where n_m and n_n are the dimensions of the data matrix. Therefore, by applying the triangle inequality and using the fact that $\|f(\cdot;\cdot)\|_F \le \|\boldsymbol{W}\|_F$, we can obtain an upper bound for the loss function as:

$$L(\boldsymbol{W}) \le \frac{1}{n_m n_n} \|\hat{\boldsymbol{X}} - \boldsymbol{X}\|_F^2 \le C\|\boldsymbol{W}\|_F^2, \tag{22}$$

where $C > 0$ is a constant that depends on the data matrix $\boldsymbol{X}$. By imposing the upper bound $M > 0$ on the Frobenius norm of the weight matrix, we can ensure that the loss function is bounded by CM^2.

By satisfying these properties, we can conclude that the loss function is convex and bounded, which ensures the existence and uniqueness of the optimal solution.

Appendix D. Convergence of gradient descent

We consider the gradient descent algorithm for minimizing the loss function $L(\boldsymbol{W})$, which updates the weight matrix as follows:

$$\boldsymbol{W}_{t+1} = \boldsymbol{W}_t - \eta_t \nabla L(\boldsymbol{W}_t), \tag{23}$$

where t is the iteration index, $\eta_t > 0$ is the learning rate, and $\nabla L(\boldsymbol{W}_t)$ is the gradient of the loss function at iteration t. We prove that gradient descent converges to the optimal solution in a finite number of steps by showing that it satisfies the following conditions:

(1) **Lipschitz Continuity**: The gradient of the loss function $\nabla L(\boldsymbol{W})$ is Lipschitz continuous with Lipschitz constant $L > 0$, which means that:

$$\|\nabla L(\boldsymbol{W}_1) - \nabla L(\boldsymbol{W}_2)\|_F \le L\|\boldsymbol{W}_1 - \boldsymbol{W}_2\|_F, \tag{24}$$

for any two weight matrices $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$.

To prove this, we use the fact that the gradient of the MSE loss function is Lipschitz continuous. For example, using the chain rule and the definition of MSE, we can write:

$$\nabla MSE(\hat{\boldsymbol{X}}, \boldsymbol{X}) = 2(\frac{\partial f}{\partial \boldsymbol{W}})^T(\hat{\boldsymbol{X}} - \boldsymbol{X}), \tag{25}$$

where $f(\cdot;\cdot)$ is the SDGNN function. Since $f(\cdot;\cdot)$ is a linear function of $\boldsymbol{W}$, we can easily see that its partial derivative with respect to $\boldsymbol{W}$ is also a linear function of $\boldsymbol{W}$, and hence Lipschitz continuous. By assuming Lipschitz continuity with Lipschitz constant $L > 0$, we can ensure that

Table 10
Hyper-parameter settings.

Model	Parameter	Option range
LSTM	Hidden dimension	$\{2^4, 2^5, 2^6\}$
TSFM	Attention heads	$\{2^2, 2^3, 2^4\}$
	Embedding dimension	$\{2^3, 2^4, 2^5\}$
TGAT	Attention heads	$\{2^2, 2^3, 2^4\}$
	Embedding dimension	$\{2^5, 2^6, 2^7\}$
	Hidden dimension	$\{2^3, 2^4, 2^5\}$
	Layers	1–5 (1 per step)
GDGCN GWNet	Residual channels	$\{2^3, 2^4, 2^5, 2^6\}$
	Dilation convolution channels	$\{2^3, 2^4, 2^5\}$
	Skip connection channels	$\{2^4, 2^5, 2^6\}$
	Number of layers	1–5 (1 per step)
GDGCN	Spatial embedding dimension	5–20 (5 per step)
	Temporal embedding dimension	5–20 (5 per step)
GWNet	Hidden dimension	$\{2^3, 2^4, 2^5\}$
	The numbers of blocks	1–5 (1 per step)
AGCRN	Embedding dimension	1–5 (1 per step)
	Hidden dimension	$\{2^4, 2^5, 2^6\}$
	Number of layers	1–3 (1 per step)
	Order of Chebyshev polynomials	1–3 (1 per step)
StemGNN	Encoder layers	1–3 (1 per step)
	The numbers of blocks	1–3 (1 per step)
SDGNN	Hidden dimension of GCN	$\{2^4, 2^5, 2^6\}$
	GCN out channels	$\{2^4, 2^5, 2^6\}$
	Kernel size	3–9 (2 per step)
	CNN out channels	$\{2^3, 2^4, 2^5, 2^6\}$

gradient descent is stable under a suitable choice of the learning rate $0 < \eta_t < 2/L$.

(2) **Strong Convexity**: The loss function $L(\boldsymbol{W})$ is $\mu > 0$-strongly convex, which means that:

$$L(\theta \boldsymbol{W}_1 + (1-\theta)\boldsymbol{W}_2), \tag{26}$$

is less than the convex combination of $L(\boldsymbol{W}_1)$ and $L(\boldsymbol{W}_2)$ by a positive amount that depends on the distance between $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$, which means that:

$$L(\theta \boldsymbol{W}_1 + (1-\theta)\boldsymbol{W}_2) < \theta L(\boldsymbol{W}_1) + (1-\theta)L(\boldsymbol{W}_2) - \frac{\mu}{2}\|\boldsymbol{W}_1 - \boldsymbol{W}_2\|_F^2, \tag{27}$$

for all $\theta \in [0,1]$ and any two weight matrices $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$.

Since a linear combination of strongly convex functions is also strongly convex, we can apply the strong convexity condition to each term separately and then add them up to obtain the desired result.

(3) **Nesterov's Acceleration**: We apply Nesterov's accelerated gradient descent technique, which improves the convergence rate by using a momentum term that incorporates the previous weight update, as follows:

$$\boldsymbol{V}_{t+1} = \gamma_t \boldsymbol{V}_t + \eta_t \nabla L(\boldsymbol{W}_t), \tag{28}$$

$$\boldsymbol{W}_{t+1} = \boldsymbol{W}_t - \boldsymbol{V}_{t+1}, \tag{29}$$

where t is the iteration index, $\eta_t > 0$ is the learning rate, $\gamma_t > 0$ is the momentum coefficient, $\nabla L(\boldsymbol{W}_t)$ is the gradient of the loss function at iteration t, $\boldsymbol{V}_t$ is the velocity vector at iteration t, and $\boldsymbol{V}_0 = 0$.

To prove that Nesterov's acceleration improves the convergence rate, we use the fact that it reduces the Lipschitz constant of the gradient by a factor of $(1-\gamma_t)^2$, which leads to a faster decrease of the loss function value. We also use the fact that it preserves the strong convexity of the loss function, which ensures that the optimal solution is unique and stable.

(4) **Complexity Analysis**: Under these conditions, we can use the big-O notation to analyze the computational complexity of each iteration and the total number of iterations required to reach an $\epsilon > 0$-optimal solution, which means that:

$$L(\boldsymbol{W}_t) - L(\boldsymbol{W}^*) < \epsilon, \tag{30}$$

where t is the iteration index, $\epsilon > 0$ is the desired accuracy, and $\boldsymbol{W}^*$ is the optimal solution.

To perform the complexity analysis, we use the fact that each gradient computation takes $O(n_m n_n n_h)$ time, where n_m, n_n, and n_h are the dimensions of the data matrix, the hidden layer, and the output layer, respectively. We also use the fact that each weight update takes $O(n_h^2)$ time, where n_h is the dimension of the hidden layer. Therefore, each iteration takes $O(n_m n_n n_h + n_h^2)$ time. To calculate the total number of iterations required to reach an ϵ-optimal solution, we use the fact that Nesterov's acceleration achieves a convergence rate of $O(1/t^2)$, where t is the iteration index. Therefore, by solving for t in terms of epsilon, we obtain:

$$t = O(1/sqrt(\epsilon)). \tag{31}$$

Hence, by multiplying the time complexity per iteration by the number of iterations, we obtain the total time complexity as $O((n_m n_n n_h + n_h^2)/sqrt(\epsilon))$.

Appendix E. Hyper-parameter settings

See Table 10.

References

[1] Lund H, Möller B, Mathiesen BV, Dyrelund A. The role of district heating in future renewable energy systems. Energy 2010;35(3):1381–90.
[2] Østergaard DS, Smith KM, Tunzi M, Svendsen S. Low-temperature operation of heating systems to enable 4th generation district heating: A review. Energy 2022;248:123529.
[3] Wei Z, Ren F, Yue B, Ding Y, Zheng C, Li B, et al. Data-driven application on the optimization of a heat pump system for district heating load supply: A validation based on onsite test. Energy Convers Manage 2022;266:115851.
[4] Huang Y, Zhao Y, Wang Z, Liu X, Liu H, Fu Y. Explainable district heat load forecasting with active deep learning. Appl Energy 2023;350:121753.
[5] Wang Z, Liu X, Huang Y, Zhang P, Fu Y. A multivariate time series graph neural network for district heat load forecasting. Energy 2023;278:127911.
[6] Lumbreras M, Garay-Martinez R, Arregi B, Martin-Escudero K, Diarce G, Raud M, et al. Data driven model for heat load prediction in buildings connected to district heating by using smart heat meters. Energy 2022;239:122318.
[7] Sun C, Liu Y, Cao S, Gao X, Xia G, Qi C, et al. Integrated control strategy of district heating system based on load forecasting and indoor temperature measurement. Energy Rep 2022;8:8124–39.

[8] Fang T, Lahdelma R. Evaluation of a multiple linear regression model and SARIMA model in forecasting heat demand for district heating system. Appl Energy 2016;179:544–52.
[9] Zhou Y, Wang L, Qian J. Application of combined models based on empirical mode decomposition, deep learning, and autoregressive integrated moving average model for short-term heating load predictions. Sustainability 2022;14(12):7349.
[10] Akhtar S, Shahzad S, Zaheer A, Ullah HS, Kilic H, Gono R, et al. Short-term load forecasting models: A review of challenges, progress, and the road ahead. Energies 2023;16(10):4060.
[11] Ntakolia C, Anagnostis A, Moustakidis S, Karcanias N. Machine learning applied on the district heating and cooling sector: A review. Energy Syst 2021;1–30.
[12] Ding Y, Timoudas TO, Wang Q, Chen S, Brattebø H, Nord N. A study on data-driven hybrid heating load prediction methods in low-temperature district heating: An example for nursing homes in Nordic countries. Energy Convers Manage 2022;269:116163.
[13] Acquaviva A, Apiletti D, Attanasio A, Baralis E, Bottaccioli L, Cerquitelli T, et al. Forecasting heating consumption in buildings: A scalable full-stack distributed engine. Electronics 2019;8(5):491.
[14] Wu Z, Pan S, Chen F, Long G, Zhang C, Philip SY. A comprehensive survey on graph neural networks. IEEE Trans Neural Netw Learn Syst 2020;32(1):4–24.
[15] Sarajcev P, Jakus D, Vasilj J. Ensemble learning with time-series clustering for aggregated short-term load forecasting. In: 2020 IEEE 20th mediterranean electrotechnical conference. MELECON, IEEE; 2020, p. 553–8.
[16] Mishra S, Glaws A, Palanisamy P. Predictive analytics in future power systems: A panorama and state-of-the-art of deep learning applications. In: Optimization, learning, and control for interdependent complex networks. Springer; 2020, p. 147–82.
[17] Calikus E, Nowaczyk S, Sant'Anna A, Gadd H, Werner S. A data-driven approach for discovering heat load patterns in district heating. Appl Energy 2019;252:113409.
[18] Idowu S, Saguna S, Åhlund C, Schelén O. Applied machine learning: Forecasting heat load in district heating system. Energy Build 2016;133:478–88.
[19] Kurek T, Bielecki A, Świrski K, Wojdan K, Guzek M, Białek J, et al. Heat demand forecasting algorithm for a Warsaw district heating network. Energy 2021;217:119347.
[20] Dixon M. Industrial forecasting with exponentially smoothed recurrent neural networks. Technometrics 2022;64(1):114–24.
[21] Song J, Zhang L, Xue G, Ma Y, Gao S, Jiang Q. Predicting hourly heating load in a district heating system based on a hybrid CNN-LSTM model. Energy Build 2021;243:110998.
[22] Leiprecht S, Behrens F, Faber T, Finkenrath M. A comprehensive thermal load forecasting analysis based on machine learning algorithms. Energy Rep 2021;7:319–26.
[23] Bassi A, Shenoy A, Sharma A, Sigurdson H, Glossop C, Chan JH. Building energy consumption forecasting: A comparison of gradient boosting models. In: The 12th international conference on advances in information technology. 2021, p. 1–9.
[24] Faber T, Finkenrath M. Load forecasting in district heating systems using stacked ensembles of machine learning algorithms. In: 14th international renewable energy storage conference 2020. Atlantis Press; 2021, p. 1–4.
[25] Borghini E, Giannetti C. Short term load forecasting using TabNet: A comparative study with traditional state-of-the-art regression models. Eng Proc 2021;5(1):6.
[26] Mohammed NA, Al-Bazi A. An adaptive backpropagation algorithm for long-term electricity load forecasting. Neural Comput Appl 2022;34(1):477–91.
[27] Giamarelos N, Papadimitrakis M, Stogiannos M, Zois EN, Livanos N-AI, Alexandridis A. A machine learning model ensemble for mixed power load forecasting across multiple time horizons. Sensors 2023;23(12):5436.
[28] Lin X, Chen R, Huang L, Liu Z, Niu X, Guo G, et al. ChirpTracker: A precise-location-aware system for acoustic tag using single smartphone. IEEE Internet Things J 2024;11(1):848–62.
[29] Xu M, Dai W, Liu C, Gao X, Lin W, Qi G-J, et al. Spatial-temporal transformer networks for traffic flow forecasting. 2020, arXiv preprint arXiv:2001.02908.
[30] Zhang Q, Chang J, Meng G, Xiang S, Pan C. Spatio-temporal graph structure learning for traffic forecasting. In: Proceedings of the AAAI conference on artificial intelligence, vol. 34, (no. 01):2020, p. 1177–85.
[31] Hu N, Zhang D, Xie K, Liang W, Hsieh M-Y. Graph learning-based spatial-temporal graph convolutional neural networks for traffic forecasting. Connect Sci 2022;34(1):429–48.
[32] Chen Y, Zhang S, Zhang W, Peng J, Cai Y. Multifactor spatio-temporal correlation model based on a combination of convolutional neural network and long short-term memory neural network for wind speed forecasting. Energy Convers Manage 2019;185:783–99.
[33] Simeunović J, Schubnel B, Alet P-J, Carrillo RE. Spatio-temporal graph neural networks for multi-site PV power forecasting. IEEE Trans Sustain Energy 2021;13(2):1210–20.
[34] Sarabu A, Santra AK. Human action recognition in videos using convolution long short-term memory network with spatio-temporal networks. Emerg Sci J 2021;5(1):25–33.
[35] Xu L, Li Q, Yu J, Wang L, Xie J, Shi S. Spatio-temporal predictions of SST time series in China's offshore waters using a regional convolution long short-term memory (RC-LSTM) network. Int J Remote Sens 2020;41(9):3368–89.
[36] Zhang S, Guo Y, Zhao P, Zheng C, Chen X. A graph-based temporal attention framework for multi-sensor traffic flow forecasting. IEEE Trans Intell Transp Syst 2021;23(7):7743–58.
[37] Wang L, Adiga A, Chen J, Sadilek A, Venkatramanan S, Marathe M. Causalgnn: Causal-based graph neural networks for spatio-temporal epidemic forecasting. In: Proceedings of the AAAI conference on artificial intelligence, vol. 36, (no. 11):2022, p. 12191–9.
[38] Jiao X, Li X, Lin D, Xiao W. A graph neural network based deep learning predictor for spatio-temporal group solar irradiance forecasting. IEEE Trans Ind Inf 2021;18(9):6142–9.
[39] Yanmei J, Mingsheng L, Yangyang L, Yaping L, Jingyun Z, Yifeng L, et al. Enhanced neighborhood node graph neural networks for load forecasting in smart grid. Int J Mach Learn Cybern 2023;1–20.
[40] Nie Y, Nguyen NH, Sinthong P, Kalagnanam J. A time series is worth 64 words: Long-term forecasting with transformers. In: Proceedings of the 11th international conference on learning representations. Kigali, Rwanda: OpenReview.net; 2023.
[41] Wan L, Zeiler M, Zhang S, Le Cun Y, Fergus R. Regularization of neural networks using DropConnect. In: Proceedings of the 30th international conference on machine learning, vol. 28, Atlanta, Georgia, USA: PMLR; 2013, p. 1058–66.
[42] Krieg SJ, Burgis WC, Soga PM, Chawla NV. Deep ensembles for graphs with higher-order dependencies. In: The eleventh international conference on learning representations. Kigali, Rwanda: OpenReview.net; 2023.
[43] Cui Y, Xie J, Zheng K. Historical inertia: A neglected but powerful baseline for long sequence time-series forecasting. In: Proceedings of the 30th ACM international conference on information & knowledge management. Queensland, Australia: Association for Computing Machinery; 2021, p. 2965–9.
[44] Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput 1997;9(8):1735–80.
[45] Wang C, Wang Y, Ding Z, Zheng T, Hu J, Zhang K. A transformer-based method of multienergy load forecasting in integrated energy system. IEEE Trans Smart Grid 2022;13(4):2703–14.
[46] Zhao H, Wang Y, Duan J, Huang C, Cao D, Tong Y, et al. Multivariate time-series anomaly detection via graph attention network. In: Proceedings of the 20th IEEE international conference on data mining. IEEE; 2020, p. 841–50.
[47] Xu Y, Han L, Zhu T, Sun L, Du B, Lv W. Generic dynamic graph convolutional network for traffic flow forecasting. Inf Fusion 2023;100:101946.
[48] Bai L, Yao L, Li C, Wang X, Wang C. Adaptive graph convolutional recurrent network for traffic forecasting. In: Proceedings of the 2020 advances in neural information processing systems, vol. 33, Virtual: Curran Associates, Inc.; 2020, p. 17804–15.
[49] Cao D, Wang Y, Duan J, Zhang C, Zhu X, Huang C, et al. Spectral temporal graph neural network for multivariate time-series forecasting. In: Proceedings of the 2020 advances in neural information processing systems, vol. 33, Curran Associates, Inc.; 2020, p. 17766–78.
[50] Wu Z, Pan S, Long G, Jiang J, Zhang C. Graph WaveNet for deep spatial-temporal graph modeling. In: Proceedings of the 28th international joint conference on artificial intelligence. Macao, China: ijcai.org; 2019, p. 1907–13.
[51] Priesmann J, Nolting L, Kockel C, Praktiknjo A. Time series of useful energy consumption patterns for energy system modeling. Sci Data 2021;8(1):148.