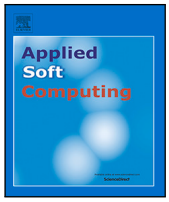
Applied Soft Computing Journal 154 (2024) 111324

Contents lists available at ScienceDirect

Applied Soft Computing

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



An attentive Copula-based spatio-temporal graph model for multivariate time-series forecasting

Xihe Qiu [a,1], Jiahui Qian [a,1], Haoyu Wang [a,1], Xiaoyu Tan [b,*], Yaochu Jin [c,*]

[a] School of Electronic and Electrical Engineering, Shanghai University of Engineering Science, Shanghai, 201620, China
[b] INF Technology (Shanghai) Co., Ltd, Shanghai, 201210, China
[c] School of Engineering, Westlake University, Hangzhou, 310030, Zhejiang, China

ARTICLE INFO

Keywords:
Electricity consumption
Copula method
Attention mechanism
Spatio-temporal graph
Time-series prediction

ABSTRACT

Time-series forecasting is widely applied to electricity consumption. However, accurate prediction for tasks is challenging due to intricate spatial dependencies and non-linear temporal dynamics. Existing models have limited capability for considering correlation factors, leading to reduced accuracy. Incorporating geographical information can enhance predictions in multivariate models. Graph neural networks effectively capture variable interdependencies, and including location information between nodes complements these dependencies. Therefore, we propose an attentive spatio-temporal graph neural network framework for accurate time-series forecasting. Our approach incorporates time-series and geographical factors to enhance prediction accuracy. We create a geometrical graph using node locations and a probabilistic graph structure learned from node embedding to capture non-linear temporal dynamics. The attention mechanism facilitates feature crossover, improving spatial-related features. We model representation and correlation information based on joint distributions in the nodes, separating them into edge densities and Copula densities. We link the graph structure and the covariance matrix in the Copula densities. Extensive evaluations of the public electrical consumption dataset demonstrate that our approach outperforms state-of-the-art models, significantly improving accuracy in multi-factor time-series forecasting tasks such as electricity consumption.

1. Introduction

Time-series forecasting primarily involves predicting future values based on past observations, with the potential to capture temporal dynamics [1]. It is crucial in various areas such as climate research, market analysis, traffic control, and energy network management [2]. These models range from early autoregressive methods to more recent deep-learning methods [3]. Exploring the intricacies of time-series forecasting helps understand the dynamic interplay between historical patterns and future predictions, which is essential for informed decision-making across domains.

Electrical energy consumption is a critical task of time-series forecasting. The objective is to predict future electricity usage based on historical consumption patterns and correlational location data. It is a complex task influenced by historical consumption patterns and correlated location data [4], characterized by intricate spatial and temporal dependencies [5]. The temporal dynamics of electricity consumption series make long-term forecasting challenging, as peak consumption times may lead to non-smoothness. In addition, unique spatial correlations exist due to varying weather conditions across regions. Current, electricity forecasting tasks are usually based on univariate time-series forecasting. Shilpa et al. [6], Bercu et al. [7], and Barak et al. [8] forecast energy consumption based on the traditional ARIMA model. The development of support vector regression machines [9], and deep recurrent neural networks [10] have also been applied to electrical energy forecasting tasks. However, these methods only consider the relationship between the time dimensions.

Recent research of multivariate forecasting models [11] leverages the interdependence among variables to generate robust predictive capabilities [12]. Shang et al. [13] introduce the graph for time-series (GTS) model for multivariate time-series prediction, which employs probabilistic graphical models and demonstrates promising results. Ma et al. [14] distinguish graph encoding forms for role representation and correlation roles, effectively incorporating correlation information into graph neural networks. Recently, graph-based multivariate time-series forecasting methods have gained popularity due to their ability to leverage interrelationships between variables for forecasting purposes. Zhang et al. [15] explore the utilization of spatio-temporal digraph convolutional networks for taxi pickup location recommendations,

* Corresponding authors.
E-mail addresses: yulin.txy@inftech.ai (X. Tan), jinyaochu@westlake.edu.cn (Y. Jin).
[1] These authors have contributed equally to this work.

https://doi.org/10.1016/j.asoc.2024.111324
Received 17 June 2023; Received in revised form 6 December 2023; Accepted 23 January 2024
Available online 30 January 2024

demonstrating the efficacy of graph-based approaches in capturing spatial and temporal dependencies. Qi et al. [16] employed a spatio-temporal graph convolutional network for long-term traffic flow prediction. Deng et al. [17], further illustrate the versatility of graph-based methods for anomaly detection in multivariate time series using graph neural networks. In time-series prediction tasks, the observed graph in the data may exhibit multiple correlations with the outcome. A graph plays a representation role if it enhances the feature representation, whereas it plays a correlation role when it directly encodes the correlation between the outcomes of connected nodes. In spatio-temporal prediction, nearby locations may exhibit correlation, and the graph structure can effectively leverage both roles for predictive purposes in distinct ways.

In this work, we propose an attentive Copula-based spatio-temporal graph model (i.e., Hybrid-GTS) for precise prediction of electricity consumption. We construct a directed graph using regional electric companies as nodes to model the pairwise distance correlations between regions. The edge weights represent the distances between the companies. Additionally, we employ a probabilistic graph structure to capture nonlinear temporal dynamics and an attention mechanism to crossover features to emphasize the importance of spatial features and facilitate feature interactions, thereby improving model performance and generalization ability. Furthermore, we leverage the Copula method to generate more effective node features by more directly utilizing correlation information. Experimental results show that our model consistently outperforms or matches the state-of-the-art baseline on various downstream tasks. We have summarized the main contributions as follows:

- We construct an adjacency matrix using a Gaussian threshold kernel with the electric companies as nodes and the spatial distances between pairs of electric companies as edges to generate a more efficient distance graph structure, which makes better use of spatial information.
- We present a novel end-to-end graph learning framework to jointly and iteratively learn discrete graph structures. Then we calculate attention weights based on the attention mechanism between the distance and discrete graph structures. Learning through feature intersection enables discrete graph structures to learn more spatially informative features.
- We introduce the Copula method into the spatio-temporal graph model. The Copula method offers better representation and correlation information based on the joint distribution of node outcomes, which separates them into edge densities and Copula densities, establishing a link between the graph structure and the covariance matrix in the Copula densities.

2. Related work

2.1. Graph neural networks

Graph neural networks (GNNs) rapidly emerge in deep learning to process graph-structured data. Graph convolutional networks (GCNs) are first introduced by Bruna et al. [18], which connect spectral graph theory. Defferrard et al. [19] propose ChebNet, which improves GCN through fast local convolutional filters. Kipf et al. [20] simplify ChebNet and achieve state-of-the-art performance in semi-supervised classification tasks. Seo et al. [21] combine ChebNet with recurrent neural networks for structured sequence modeling. Yu et al. [22] model sensor networks as undirected graphs and apply ChebNet and convolutional sequence models for prediction. The drawback of spectral-based convolutions mentioned above is that they usually require the graph to be undirected to compute a meaningful spectral decomposition. From the spectral domain to the vertex domain, Atwood et al. [23] propose diffusion convolutional neural network (DCNN), which defines convolution as a diffusion process at each node in the input of the graph structure. Hechtlinger et al. [24] propose GraphCNNs generalize convolution to graphs by convolving each node with its p nearest neighbors. However, both approaches neglect temporal dynamics and predominantly address static graphs.

2.2. Time-series prediction

Li et al. [25] use a diffusion process on a directed graph to model traffic flow, capturing spatial dependencies through bi-directional random wandering. They also model sensor networks as weighted directed graphs and employ an encoder–decoder architecture with predetermined sampling to capture temporal dependencies. Lu et al. [26] explore the application of a differential evolution-based approach to address the dynamic challenges posed by cyber-attacks in cyber–physical power systems. By introducing a three-stage methodology, they provide a novel perspective on mitigating cyber threats, and enhancing the robustness of power systems. The ability to capture temporal dependencies between time series can be applied to various spatio-temporal prediction problems, where the temporal history captures dynamics over time, and spatial correlations reflect relationships between samples from different spatial locations. However, this method applies only to problems with known graph structures, and real-world graphs are typically noisy. Franceschi et al. [27] treat learning the parameters of graph structures and convolutional graph networks as a two-layer programming problem, training the parameters of the graph convolutional networks while learning the discrete and sparse dependency structure between data nodes. However, optimizing this approach internally is computationally demanding, and scalability becomes challenging with an increasing number of time-series data. Shang et al. [13] describe the problem of learning an unknown graph structure as learning a probabilistic graph model by optimizing the average performance of the graph distribution. However, despite its computational efficiency, this approach does not integrate correlation information.

2.3. Correlation information

Correlation graph information enhances the training of graph neural networks. Ma et al. [28] propose a semi-supervised graph-based learning generation framework that utilizes correlation graph information to enhance the training of graph neural networks. This approach instantiates multiple distributions through joint distributions of node features, labels, and graph structure and then employs scalable variational inference techniques to approximate Bayesian posteriors. Qu et al. [29] introduce graph Markov neural networks (GMNNs), which utilize conditional random fields to model the joint distribution of object labels. The approach iterates through a pseudo-likelihood variational EM algorithm for updating. While these methods rely on correlation information, these methods primarily focus on classification models.

Jia et al. [30] apply a multivariate normal distribution to model the correlation of node outcomes, using GNNs as the base regressor to capture the dependence of results on vertex features, and then further model the regression residuals on all vertices. The correlation structure is learned through training by maximizing the edge likelihood of the observed vertex labels. During inference, the predicted outcome of the test vertices is obtained by maximizing the conditional probability given the training labels. Ma et al. [14] decompose the joint distribution of node outcomes using the Copula function, which is commonly used in multivariate statistics to model representational and correlation information. Separating the representation and correlation information into edge density and Copula density allows for better exploitation of the roles of graph representation and correlation. While these methods reasonably leverage graph correlation information, they are limited to traditional regression tasks such as node prediction and are not well-suited for temporal models.

Leveraging ensemble LSTM, NNCT weight integration, and population extremal optimization, zhao et al. [31] present an effective solution for enhancing the accuracy of traffic flow predictions. Bahdanau et al. [32] introduce the attention mechanism in natural language processing (NLP) and demonstrated its significance in learning interrelationships between elements in sequence-to-sequence models. While existing research extensively models temporal graphical structures, few approaches incorporate correlation information, especially in the context of energy forecasting.

2.4. Energy forecasting

Historical data on electricity consumption is usually recorded in time-series [33], and past data can be analyzed to predict future electricity consumption. Machine learning-based methods benefit from their ability to handle non-linear data, making them an effective tool for performing power forecasting tasks. Ceperic et al. [9] propose a generic model for improving support vector regression (SVR) machines to predict short-term electrical loads, reducing operator interaction during model construction by using a feature selection algorithm for automatic model input selection and a particle swarm-based global optimization technique to optimize SVR hyperparameters. Rahman et al. [10] optimize a new deep recurrent neural network (RNN) model for medium- and long-term electricity load forecasting. Xu et al. [34] propose a hybrid model combining a linear regression (LR) model and a deep belief network (DBN) model for the prediction of time-series data. Yang et al. [35] introduce a hybrid load prediction model that integrates an extreme learning machine (ELM) with an improved whale optimization algorithm. They employ Huber loss, which is insensitive to outliers, as the objective function during ELM training. Qiu et al. [36] introduce a k-nearest neighbor attentive deep autoregressive network for electricity consumption prediction, leveraging the time-series information. However, these approaches neglect the integration of the timing model with the diagram and ignore the important correlation information.

To address the aforementioned issues, we propose an attentive Copula-based spatio-temporal graph model called Hybrid-GTS for precise prediction of electricity consumption. Hybrid-GTS enhances the GTS model by integrating location information to generate distance graph structures and learning discrete graph structures while employing attention mechanisms to extract significant features and enhance feature cross for the improved acquisition of spatial correlation information. To enhance multi-temporal prediction, we apply the Copula method to separate the model prediction results into edge density and Copula density, thereby utilizing correlation information to improve algorithm performance.

3. Methods

Fig. 1 provides an overview of our proposed Hybrid-GTS. In this model, the Gumbel reparameterization method learns the discrete graph structure to represent nonlinear temporal dynamics, while the directed graph represents pairwise distance correlation between nodes. The attention mechanism is utilized to enhance the acquisition of important spatial correlation relationships. Moreover, the model incorporates prior knowledge, such as k-nearest neighbor (kNN) graphs, to improve the quality of the discrete graph structure. This helps the model capture the global signal more effectively and apply it to each sequence.

We model the discrete graph structure and perform predictions using a diffusion convolutional graph neural network. The resulting joint distribution of the predictions is separated into the edge and Copula density based on the Copula method. This allows the separate modeling of representational and correlation information to generate more efficient embeddings and improve model performance. The model architecture utilizes a T-step window to predict the next τ steps, which can be divided into distance graph generation, discrete graph learning, and prediction process.

Distance graph generation: as shown in Fig. 1(a), a directed graph is constructed using regional electric companies as nodes and pairwise spatial distances as edge weights to capture spatial correlations.

Discrete graph learning: as shown in Fig. 1(b), a discrete graph structure is learned using a Gumbel reparameterization method to represent nonlinear temporal dynamics. Node features are extracted and used to parameterize the discrete graph.

Prediction process: The distance graph, discrete graph, node features, and attention mechanism are integrated into an encoder–decoder architecture for multi-step electricity consumption forecasting. The decoder jointly models representational and correlational information using a Copula-based approach.

Electricity forecasting aims to predict future electricity consumption based on previous electricity consumption through N electric companies. We represent the electric company network as a weighted directed graph $g = (v, e, w)$, where v is a set of nodes, e is a set of edges, and w denotes the weighted adjacency matrix of node distances.

Firstly, the notation is collated, and the training data is represented by a three-dimensional vector X. The three dimensions represent features, time, and n-sequences, respectively. The superscript table below denotes the sequence, and the superscript denotes the time. X_i^t denotes the tth time step of the ith sequence of all features. The model will use the T' step window to predict the next T steps. The objective of the power prediction problem is to learn the function $f(\cdot)$, using ℓ to denote the loss function between the prediction and the underlying facts, with the following typical training objective.

$$\sum_{t=1}^{T} \ell\left(f\left(A, w, X_{t+1:t+T}\right), X_{t+T+1:t+T+\tau}\right) \tag{1}$$

where t denotes the time step index ranging from 1 to the total number of time steps, T. Key variables include ℓ representing the loss function, f denoting the prediction model, A indicating the adjacency matrix of the graph, and w signifying the weight matrix of the graph. Additionally, $X_{t+1:t+T}$ represents historical input data within a time window, and $X_{t+T+1:t+T+\tau}$ denotes the future time window data to be predicted.

3.1. Graph structure parameterization

The binary matrix $A \in [0,1]_{n\times n}$ is itself hard to parameterize, such that A is a random variable of the matrix Bernoulli distribution parameterized by $\theta \in [0,1]_{n\times n}$ such that A_{ij} is independent for all (i, j) pairs with $A_{ij} \sim Ber(\theta_{ij})$. Here, θ_{ij} is the probability of success of the Bernoulli distribution. The training objective is modified as follows:

$$E_{A\sim Ber(\theta)}\left[\sum_{t=1}^{T} \ell\left(f\left(A, w, X_{t+1:t+T}\right), X_{t+T+1:t+T+\tau}\right)\right] \tag{2}$$

Then we utilize the Gumbel [37] reparameterization technique,

$$A_{ij} = \text{sigmoid}\left(\left(\log\left(\theta_{ij}/\left(1-\theta_{ij}\right)\right) + \left(g_{ij}^1 - g_{ij}^2\right)\right)/s\right) \tag{3}$$

where $g_{ij}^1, g_{ij}^2 \sim Gumbel(0,1)$ for all i, j. When the temperature $s \to 0$, $A_{ij} = 1$ with probability θ_{ij} and a residual probability of θ. In practice, we gradually anneal s to zero in training.

For the parameterization of θ, each time series is extracted by features. Specifically, we convolve along the time dimension and vectorize along this dimension. Finally, we reduce the dimensionality with a fully connected layer to generate feature vectors, which map the matrix X_i to a vector z_i for each i. A pair of vectors (z_i, z_j) is then mapped to the scalar $\theta_{ij} \in [0,1]$, as detailed in Fig. 1(b).

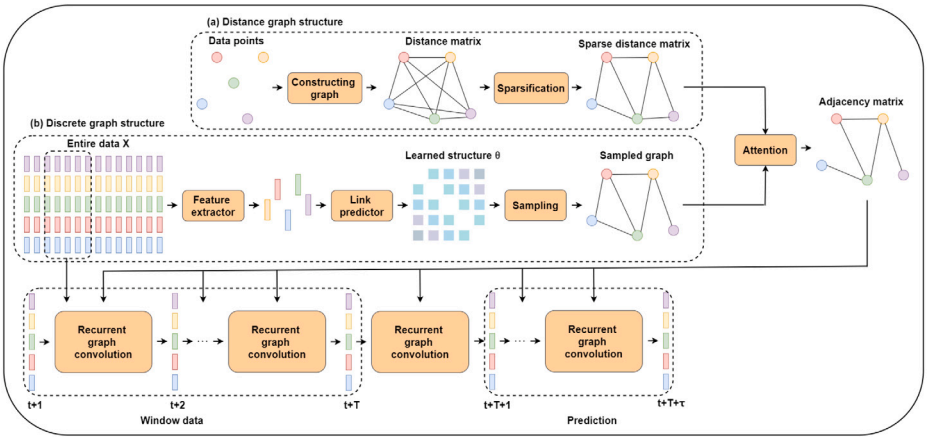


Fig. 1. The overall architecture of the model. (a) indicates the process of generating the distance graph structure; (b) shows the process of parameterizing θ; and the prediction process is presented at the bottom of the figure.

3.2. Distance graph structure

We then generate a distance graph structure based on the spatial information between the nodes with the distance between the electric companies as edges, using the electric companies as nodes. The discrete and distance graph structures are combined into the final spatio-temporal graph through an attention mechanism.

We calculate the pairwise spatial distances between electric companies and construct the adjacency matrix using a Gaussian threshold kernel:

$$\mathcal{G}_{ij} = \exp(-\frac{\text{dist}(v_i, v_j)^2}{\sigma^2}) \tag{4}$$

Where $\mathcal{G}_{ij}$ denotes the edge weight between electric company v_i and v_j, $dist(v_i, v_j)$ represents the spatial distance from electric company v_i to v_j, and σ is the standard deviation of the space. Here only the distance from the nearest node of each electric company is kept, and other distances are set to 0. This paper generates the distance graph structure based on the spatial information between nodes and the distance between electric companies as edges, as detailed in Fig. 1(a). The final spatio-temporal graph is generated by combining the discrete and distance graph structures through the Bahdanau attention mechanism [32].

The attention mechanism distributes weight values reflecting the dependencies between elements, with higher weights assigned to more strongly correlated features. This study proposes a method for learning a graph structure with spatial information by integrating the spatial graph structure with a learnable discrete graph structure. Specifically, attention weights are calculated between these two graph structures and multiplied with the discrete graph structure to obtain the desired one.

The general idea is to obtain the attention score $score\left(g_s, g_a\right)$ by comparing the distance graph embedding g_s and the discrete graph embedding g_a:

$$score\left(g_s, g_a\right) = v^\top \tanh\left(W_1 g_s + W_2 g_a\right), \tag{5}$$

where v, W_1, W_2 are the connection weights.

The attention scores are then normalized using the Softmax function to obtain the attention weights α_{sa}:

$$\alpha_{sa} = \frac{\exp\left(score\left(g_s, g_a\right)\right)}{\sum_{a'=1}^{a} \exp\left(score\left(g_s, g_{a'}\right)\right)} \tag{6}$$

Finally, the final embedding c_a is obtained by weighting the discrete graph embedding g_a according to the attention weights α_{sa}.

$$c_a = \sum_a \alpha_{sa} g_a \tag{7}$$

3.3. Correlation information modeling

We employ the Copula method, commonly utilized in multivariate statistics, to model the conditional joint distribution of predicted outcomes. This technique is applied to separate the labeled joint distribution into representational and correlation components.

Sklar's theorem [38] states that the multivariate joint distribution $F = (Y_1, \ldots, Y_N)$ of an arbitrary random vector Y can be expressed as a one-dimensional marginal distribution $F_I(y) = P(Y_I \leqslant y)$ and a Copula C: $[0,1]^n \rightarrow [0,1]$ describing the dependence structure among variables.

$$F\left(y_1, \ldots, y_n\right) = C\left(F_1\left(y_1\right), \ldots, F_p\left(y_n\right)\right) \tag{8}$$

In other words, the joint distribution can be decomposed into the edge and Copula densities. This decomposition allows modeling the joint distribution in two steps: (1) learning the edge F_i; (2) learning the Copula C.

The goal of this paper is to model the representation and correlation information in the conditional joint distribution of node outcomes, decomposing them into Copula density and edge density:

$$f(\mathbf{y}; \mathbf{X}, \mathcal{G}) = c\left(u_1, \ldots, u_n; \mathbf{X}, \mathcal{G}\right) \prod_{i=1}^{n} f_i\left(y_i; \mathbf{X}, \mathcal{G}\right) \tag{9}$$

Where $u_i = F_i(y_i)$ signifies that u_i is a variable derived by transforming the original variable y_i through the marginal distribution function F_i into the interval [0,1]. f_i is the Probability Density Function (PDF) of Y_i and c is the Copula density. In this formulation, the representative and correlation information is naturally divided into Copula density c and edge density f_i, where $i = 1, \ldots, n$, where both the edge density and Copula density are conditioned on the node feature $\mathbf{X}$ and the graph $\mathcal{G}$. Then a suitable family of distributions is chosen for each of these two densities. The distributions are parameterized as a function of $\mathbf{X}$ and $\mathcal{G}$.

The Copula method decomposes the joint distribution of prediction labels into representation and correlation components. We use negative log-likelihood to constrain the model to use the correlation information better, learn better features, and thus improve the model's prediction performance.

3.4. Prediction

For model prediction, this paper uses a sequence-to-sequence (Seq2Seq) model to map $X^i_{t+1:t+T}$ to $X^i_{t+T+1:t+T+\tau}\tau$ for each sequence i. The sequence-to-sequence model is usually cyclic. However, because of the available graph structure between sequences, this paper uses cyclic graph convolution to process all sequences simultaneously instead of the usual cyclic mechanism processing each sequence separately.

Specifically, for each time step t', the sequence-to-sequence model takes $X_{t'}$ of all sequences as input and updates the hidden internal state from $H_{t'-1}$ to $H_{t'}$. The encoder part of Seq2Seq performs a cyclic update from $t' = t - T'$ to $t' = t$, producing H_t as a summary of the input. The decoder part uses H_t to continue the loop and evolve hidden states for another T-step. For each hidden state $H_{t'}$, $t' = t+1 : t+T$ is used as both the output $\hat{X}_{t'}$ and the input for the next step.

The prediction model is based on an existing framework diffusion convolutional recurrent neural network (DCRNN) [25] designed for directed graphs, and this diffusion is characterized by a random restart probability $\alpha \in [0,1]$ and a state transfer matrix $D_O^{-1}A$ on the graph G wanders. Here $D_O = diag(A1)$ is the out-degree diagonal matrix, and $\mathbf{1} \in \mathbb{R}^N$ denotes the all-ones vector. After many time steps, such a Markov process converges to a smooth distribution $\mathcal{P} \in \mathbb{R}^{N\times N}$, where the ith row $\mathcal{P}_{i,:} \in \mathbb{R}^N$ denotes the possibility of diffusion from node $v_i \in \mathcal{V}$.

The smooth distribution of the diffusion process can be expressed as a weighted combination of infinite random wanderings on the graph and calculated in closed form:

$$\mathcal{P} = \sum_{k=0}^{\infty} \alpha(1-\alpha)^k \left(D_O^{-1}A\right)^k \tag{10}$$

Where k is the diffusion step, in practice, using a finite K-step truncation of the diffusion process and assigning trainable weights to each stage, including the reverse diffusion process, bidirectional diffusion provides greater flexibility to the model.

The time dependence is modeled with a gated recurrent unit (GRU) [39], which replaces the matrix multiplication in the GRU with a diffusion convolution:

$$R_{t'} = \text{sigmoid}\left(W_R \star_A \left[X_{t'} \parallel H_{t'-1}\right] + b_R\right) \tag{11}$$

$$C_{t'} = \tanh\left(W_C \star_A \left[X_{t'} \parallel \left(R_{t'} \odot H_{t'-1}\right) + b_C\right]\right. \tag{12}$$

$$U_{t'} = \text{sigmoid}\left(W_U \star_A \left[X_{t'} \parallel H_{t'-1}\right] + b_U\right) \tag{13}$$

$$H_{t'} = U_{t'} \odot H_{t'-1} + \left(1 - U_{t'}\right) \odot C_{t'} \tag{14}$$

Where $X_{t'}$, $H_{t'-1}$ denote the input and output at time t, $R_{t'}$ and $U_{t'}$ are the reset gate and update gate at time t, respectively, and W_R, W_C, and W_U are the parameters of the corresponding filters. The graph convolution $\star_A$ is defined as:

$$W_Q \star_A Y = \sum_{k=0}^{K} \left(w_{k,1}^Q \left(D_O^{-1}A\right)^k + w_{k,2}^Q \left(D_I^{-1}A^T\right)^k\right) Y \tag{15}$$

Here, D_O and D_I are the out-degree and in-degree matrices, $D_O^{-1}A$ and $D_I^{-1}A^T$ represent the transfer matrices of the diffusion process, and the inverse diffusion process, respectively, $\parallel$ is a series along the characteristic dimension, $w_{k,1}^Q$, $w_{k,2}^Q$, b_Q are the parameters of the model, where $Q = R, U, C$, and the diffusion degree K is the hyperparameter. Like the GRU model, diffusion convolutional gated recurrent unit (DCGRU) can be used to construct recurrent neural network layers and trained by backpropagation in time. In multi-step ahead prediction, a sequence-to-sequence architecture is used, and the encoder and decoder are recurrent neural networks with DCGRU. During training, historical time series are fed to the encoder, and the final state of the encoder is used to initialize the decoder. The decoder generates predictions based on previous ground truth observations, which are replaced by forecasts generated by the model during testing. The encoder–decoder architecture is shown in Fig. 2.

The underlying training loss of the model is the mean absolute error between the prediction and the underlying facts:

$$\ell_{\text{base}}^t \left(\hat{X}_{t+T+1:t+T+\tau}, X_{t+T+1:t+T+\tau}\right) = \frac{1}{\tau} \sum_{t'=t+T+1}^{t+T+\tau} \left|\hat{X}_{t'} - X_{t'}\right| \tag{16}$$

In addition, regularization improves the quality of graph structure by injecting a prior knowledge of pairwise interactions into the model. Sometimes the graphs in the time series are known, and if the explicit structure is unknown, neighborhood graphs (e.g., kNN [40] graphs) can be used as reasonable knowledge. If k is small, using kNN encourages sparsity, which avoids the disadvantage of the ℓ_1 constraint, which cannot be easily imposed because the graph is not the original variable to be optimized.

Then, the cross entropy between θ and the k-neighborhood graph A_{ij}^k is used as a regularization to improve the quality of the graph:

$$\ell_{\text{reg}} = \sum_{ij} -A_{ij}^k \log\theta_{ij} - \left(1 - A_{ij}^k\right) \log\left(1 - \theta_{ij}\right) \tag{17}$$

Finally, the joint distribution of the predicted labels $\hat{X}_{t'}$ is decomposed into representation and correlation components based on the Copula method to separate them into edge densities f_i and Copula densities c:

$$f\left(\hat{X}_{t'}; X_{t'}, \mathcal{G}\right) = c\left(u_1, \ldots, u_n; X_{t'}, \mathcal{G}\right) \prod_{i=1}^{n} f_i\left(\hat{X}_{t'}^i; X_{t'}, \mathcal{G}\right) \tag{18}$$

The Gaussian copula density function is a pivotal concept in understanding the joint distribution of a multivariate dataset when using the Gaussian copula [41]. It is expressed as:

$$c_\Sigma(u_1, \ldots, u_n) = |\Sigma|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}\mathbf{q}^T \Sigma^{-1}\mathbf{q}\right) \tag{19}$$

Here, the term $|\Sigma|^{-\frac{1}{2}}$ represents the inverse of the square root of the determinant of the correlation matrix Σ, and it ensures normalization of the density function [42].

The vector $\mathbf{q}$ is defined as $\mathbf{q} = (q_1, \ldots, q_n)^T$, where each component q_i is given by:

$$q_i = \Phi^{-1}(u_i) \tag{20}$$

Where $\Phi^{-1}(u_i)$ is the inverse of the cumulative distribution function (CDF) of the standard normal distribution, transforming the uniform marginals u_i into the standard normal space [43].

The joint density can be written as the product of the Copula density and the edge density, and the loss function $\mathcal{L}$, i.e., the negative log-likelihood:

$$\mathcal{L} = -\log f\left(\hat{X}_{t'}; X_{t'}, \mathcal{G}\right) = -\log c(u; \mathcal{G}) - \sum_{i=1}^{m} \log f_i\left(\hat{X}_{t'}^i; \eta_i\left(X_{t'}, \mathcal{G}; \gamma\right)\right) \tag{21}$$

Where $\eta_i(\cdot;\gamma)\cdot$ denotes the normal distribution.

Then the total loss function is:

$$L = \sum\nolimits_t \ell_{\text{base}}^t + \lambda\ell_{\text{reg}} + \mathcal{L} \tag{22}$$

Where $\lambda > 0$ is the regularized magnitude.

Our approach employs the Copula functions to characterize dependencies in time series data, without being restricted to a specific copula family. As long as the copula function meets certain conditions, such as closure and differentiability, our method remains applicable. This imparts a high degree of generality and flexibility to our approach. The pseudo-code of our proposed Hybrid-GTS is shown in the Algorithm 1.

4. Experiment and analysis

4.1. Datasets and experimental settings

Datasets: PJM Hourly Energy Consumption Data[2] is a dataset for the U.S. electricity market that records hourly electricity consumption in the PJM Interconnection grid area, including timestamps (date and hour), area codes, and electricity consumption in megawatt hours. In this paper, the experiments are based on the PJM dataset that contains

[2] https://www.pjm.com/markets-and-operations/data-dictionary

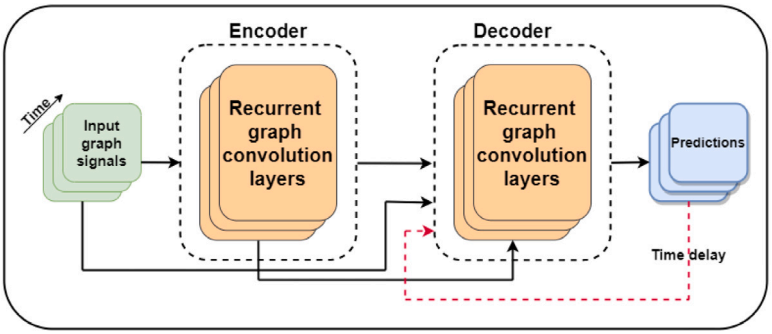


Fig. 2. The encoder-decoder architecture.

Algorithm 1: Hybrid-GTS

input : Entire data X and the distance adjacency matrix $\mathcal{G}$

$A^k \leftarrow k\text{NN}(X, k)$;
while *Stopping condition is not met* **do**
  $A \sim Ber(\theta)$;
  $G \leftarrow \text{Attention}(\text{Encoder}(X, \mathcal{G}), \text{Encoder}(X, A))$;
  $\hat{X} \leftarrow \text{Decoder}(G, A)$;
  while *Inner objective decreases* **do**
    $\ell_{\text{base}} \leftarrow \frac{1}{\tau}\sum_{|\hat{X}_{t'} - X_{t'}|}$;
    $\ell_{\text{reg}} \leftarrow \sum_{ij} -A^k_{ij}\log\theta_{ij} - \left(1 - A^k_{ij}\right)\log\left(1-\theta_{ij}\right)$;
    $\mathcal{L} \leftarrow -\log c(u;\mathcal{G}) - \sum_{i=1}^{m}\log f_i\left(\hat{X}^i_{t'};\eta_i\left(X_{t'},\mathcal{G};\gamma\right)\right)$;
    Minimize($\ell_{\text{base}} + \ell_{\text{reg}} + \mathcal{L}$)
  end
end
return: Prediction data $\hat{X}$

electricity consumption data from five electric utilities, and 55 months of data from January 1, 2014, to August 2, 2018, are collected for the experiments. The electric company data is aggregated into a 12-h window, and 70% is used for training, 20% for testing, and 10% for validation.

Experimental settings: The experiments are executed on an Intel Core i9-12900k processor and an NVIDIA GeForce RTX 3090 GPU, utilizing the Pytorch 1.7.0 deep learning framework and accelerated by CUDA 11.0. The hyperparameters of the model are set as follows, Epochs: 200, batch size: 64, the initial learning rate of 0.001, decay ratio of learning rate of 0.1, k value of 3 in kNN, and a convolutional kernel size of 10 in the feature extractor.

4.2. Baselines

Regarding hyperparameter selection in the proposed Hybrid-GTS model, we performed a grid search over comparable dimensions to existing baselines like latent dimensionality, attention mechanisms, and regularization tradeoff factors. The search determines optimal settings that maximize validation performance.

We evaluate various widely used and state-of-the-art time-series forecasting methods on the public electricity consumption dataset:

HA: The Historical Average model calculates the average of historical data points from the past. It provides a basic benchmark by assuming that future values will, on average, resemble past values.

VAR: Vector autoregression (VAR) [44] is a method that models the interdependencies among multiple time-series variables. VAR captures the linear relationships between each variable and its own lagged values, as well as the lagged values of other variables in the system, making it suitable for multivariate time series analysis.

DCRNN: Diffusion convolutional recurrent neural network(DCRNN) [25] captures spatial dependence using bi-directional random wandering on the graph and temporal dependence using an encoder–decoder architecture and scheduled sampling. It employs diffusion convolution to capture spatial relationships through bi-directional random walks on the graph while using a recurrent neural network with an encoder–decoder architecture for temporal aspects.

GTS: Graph for time series (GTS) [13] is designed to handle unknown graph structures. It simultaneously learns the graph structure and the parameters of GNNs, making it adept at uncovering hidden spatial dependencies in the data.

GTS with Attention: This paper enhances the GTS model by incorporating a distance graph architecture utilizing node location information and introducing an attention mechanism for feature crossover to augment spatial-related features. This modification aims to improve the model's ability to focus on key areas of the graph that are more predictive of future values.

GTS with Copula: This paper uses the Copula method to model the representational and correlation information based on the joint distribution of node results based on the GTS model, which makes better use of the correlation information.

All methods are evaluated with three metrics: mean absolute error (MAE), root mean square error (RMSE), and mean absolute percentage error (MAPE).

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{23}$$

where y_i is the actual value, $\hat{y}_i$ is the predicted value, and n is the number of predictions. A lower MAE indicates better model performance.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{24}$$

A lower RMSE similarly indicates better predictive ability.

$$MAPE = \frac{100}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{25}$$

For all metrics, a smaller value indicates superior model performance in terms of prediction accuracy. We include detailed evaluations using these metrics in the Experiments section to quantify model improvements.

4.3. Main results

We first evaluate the performance of Hybrid-GTS by comparing it to all previously mentioned baselines for the tasks of 3, 6, and 12-h advance forecasting. Using American Electric Power (AEP) as an example, the visualization of the Hybrid-GTS model for electricity consumption prediction is shown in Fig. 3, where it is clear that the longer the lead time, the worse the forecast.

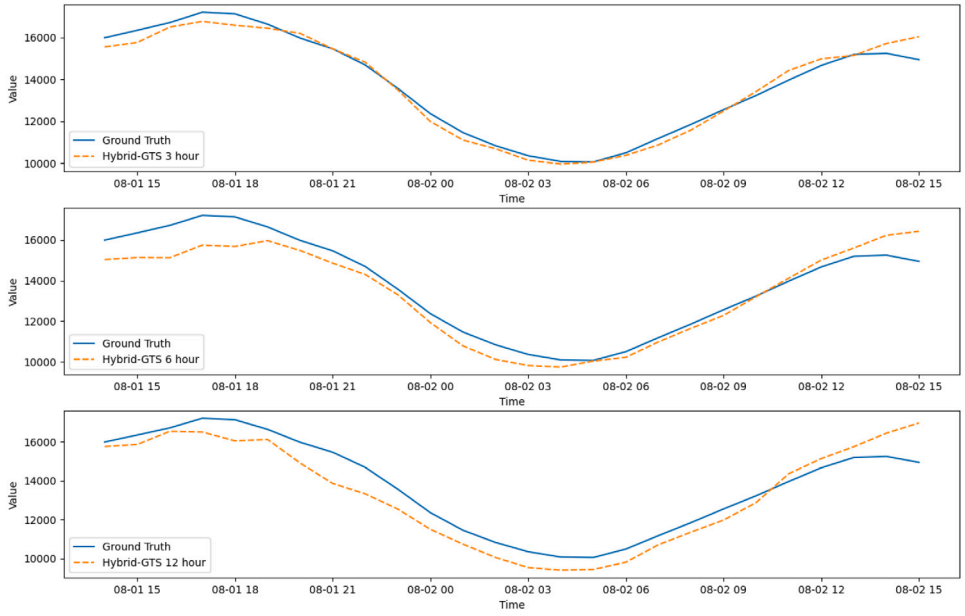


Fig. 3. Power consumption forecast visualization by AEP.

We evaluated the proposed Hybrid-GTS model against an array of baseline methods, including historical averages (HA), vector autoregression (VAR), diffusion convolutional RNN (DCRNN), vanilla graph temporal series (GTS), GTS with attention, and GTS with Copula modeling. Table 1 displays the predictions for the next 3-h, 6-h, and 12-h on publicly available datasets. Lower values indicate better model performance for all metrics.

For the VAR and DCRNN benchmark models, we utilized model specifications and hyperparameter values from their original papers or public code repositories to ensure the methods achieve state-of-the-art performance.

Specifically, the VAR model order was set to 24 based on common practice for hourly electricity data. The number of convolutional filter taps in DCRNN was set to 3, as reported in the paper. We performed a grid search over recurrent layer hidden units $\in \{16, 32, 64\}$ and graph convolutional layers $\in \{1, 2, 3\}$ for DCRNN, selecting the optimal values on the validation data to maximize predictive accuracy.

For the GTS-based benchmarks, we implemented the models according to the algorithm pseudocode and details provided in the original paper. A grid search was conducted over comparable hyperparameters including latent dimensionality, attention mechanisms and regularization tradeoff term.

Our Hybrid-GTS model inherits the validated hyperparameter ranges from these baselines during our grid search to determine optimal settings. Therefore, the experimental comparisons keep model specification options consistent to ensure fair assessment.

Experimental results demonstrate that, in general, deep learning techniques outperform non-deep learning methods, except for vector autoregression, which exhibits comparable performance to deep learning methods on specific metrics. Meanwhile, the inclusion of attention mechanisms significantly enhances the model's performance. Within the graph-based paradigm, GTS notably outperforms DCRNN for predictions at 6 and 12 h, while DCRNN exhibits slightly superior performance for predictions at 3 h. The critical distinction between the two methods is that DCRNN employs a pre-specified graph, while GTS learns the graph parametrically. This suggests that graphs derived from parametric learning cannot completely replace the fundamental graph structure, particularly for short-term forecasting. By merging the attention-based sparse distance graph architecture with parametric GTS, this study significantly improves model performance, thereby demonstrating the efficacy of the generated distance graph structure. Notably, the Hybrid-GTS method, which combines the Copula method and attention mechanism, outperforms GTS and DCRNN, demonstrating that modeling the representational and correlation information of the joint distribution of node outcomes can make more effective use of the correlation information and learning better feature embeddings.

4.4. Ablation studies

4.4.1. Effects of different sparsification degrees of distance graph

To verify the effect of different sparsification degrees of the generated distance graph structure on the model, we find the optimal distance graph structure by controlling the size of the sparsification factor S. The size of S means that the edges of the S nodes closest to the current node are retained. S=2 implies that only the edges between the two nearest nodes to the current node are kept. And the edges between the other nodes and the current node are deleted. The experimental results are shown in Table 2. It can be seen that when the sparsification factor S becomes larger, it does not have a particularly significant impact on the short-term forecasting of 3 and 6 h. But it significantly impacts the forecasts of 12 h because the electricity consumption data for the current area node only correlates with the nearest area. The information from distant regions cannot positively impact the electricity consumption forecasts. The correlation between the nodes has a negligible impact on the 3-h and 6-h forecast horizons due to the short forecast time. However, for medium and long-term predictions spanning 12 h, the influence of neighboring nodes on the data increases proportionally due to the extended lead time. By sparsifying the distance graph structure, the model can focus only on the electricity consumption information of neighboring nodes, and the learned node features are more effective.

4.4.2. Effects of different attention mechanisms

To verify the effect of different attention mechanisms on the performance of the model, we compare the performance of varying attention mechanisms through experiments. The experimental setups are consistent in terms of parameters for a fair comparison, and the experimental results are shown in Table 3. The efficacy of Hard attention [45] is comparatively lower, possibly due to the random nature of the process. Hard attention selects a subset of the encoder's hidden layer output for computation based on the given probability, rather than using the entire input. The information received by this random sampling is not

Table 1
Predictive performance of different methods.

	Metrics	HA	VAR	DCRNN	GTS	GTS with attention	GTS with copula	Hybrid-GTS
3 h	MAE	827.15	140.54	134.6	142.51	131.15	130.31	**126.37**
	RMSE	1370.14	232.63	222.83	236.04	217.21	214.94	**208.84**
	MAPE	13.06%	2.52%	2.43%	2.55%	2.36%	2.38%	**2.33%**
6 h	MAE	827.15	237.48	235.2	225.57	210.86	211.65	**210.7**
	RMSE	1370.14	395.1	391.78	374.6	357.12	354.42	**353.78**
	MAPE	13.06%	4.12%	4.03%	3.95%	3.66%	3.7%	**3.68%**
12 h	MAE	827.15	354.06	346.92	330.03	316.3	309.86	**302.94**
	RMSE	1370.14	591.42	586.6	551.78	529.17	513.9	**507.68**
	MAPE	13.06%	5.95%	5.76%	5.77%	5.49%	5.36%	**5.22%**

Table 2
Effect of different sparse degree of distance graph structure on the model.

	Metrics	S = 4	S = 3	S = 2
3 h	MAE	130.86	130.32	**126.37**
	RMSE	218.49	217.03	**208.84**
	MAPE	2.33%	**2.32%**	2.33%
6 h	MAE	212.29	213.64	**210.7**
	RMSE	358.28	358.3	**353.78**
	MAPE	3.7%	3.7%	**3.68%**
12 h	MAE	321.57	307.37	**302.94**
	RMSE	534.62	514.96	**507.68**
	MAPE	5.54%	5.31%	**5.22%**

Table 3
Impact of different attention mechanisms on the model.

	Metrics	Hard attention	Sparse attention	Bahdanau attention
3 h	MAE	234.2	143.75	**126.37**
	RMSE	384.81	238.46	**208.84**
	MAPE	4.14%	2.64%	**2.33%**
6 h	MAE	286.57	223.92	**210.7**
	RMSE	470.03	376.74	**353.78**
	MAPE	4.96%	3.93%	**3.68%**
12 h	MAE	361.02	327.09	**302.94**
	RMSE	596.88	544.72	**507.68**
	MAPE	6.16%	5.68%	**5.22%**

complete enough. Sparse attention [46] only retains values in a small region and forces most of the attention to be zero to reduce the amount of attention computed, and also reduces the information of acceptable nodes, which can significantly reduce the time complexity for multi-node data. Still, sparse attention can reduce the prediction accuracy for data sets with fewer nodes. In this paper, we employ Bahdanau attention [32], which retains more information while allowing the node embedding to learn both distance correlation and time correlation and has the most noticeable improvement on the model prediction effect. In summary, using a simpler attention mechanism is recommended for data with fewer nodes in order to improve correlation learning.

4.4.3. Effects of using non-gaussian copulas

We conduct an additional ablation study to demonstrate the effectiveness of employing non-Gaussian copula functions, highlighting the flexibility inherent in the Copula approach. We perform experiments employing an alternative copula methodology: the Gumbel copula, which models upper tail asymmetry commonly observed in risk modeling. The experimental setup is identical to the same optimization, training, and evaluation protocol as the original experiments for fairness. We employ grid search to fine-tune hyperparameters, such as degrees of freedom, for each non-Gaussian copula under consideration. The convolution networks, attention mechanisms, loss functions, and all other model components remain constant; only the parametric copula density function is altered.

Table 4
Impact of different copula approaches on the model.

	Metrics	No copula	Gumbel copula	Gaussian copula
3 h	MAE	263.41	152.37	**126.37**
	RMSE	431.69	253.87	**208.84**
	MAPE	4.65%	2.81%	**2.33%**
6 h	MAE	330.24	298.73	**210.7**
	RMSE	543.46	497.23	**353.78**
	MAPE	5.68%	5.19%	**3.68%**
12 h	MAE	447.82	421.53	**302.94**
	RMSE	738.71	700.39	**507.68**
	MAPE	7.66%	7.25%	**5.22%**

The Gaussian copula obtains superior accuracy over all other approaches, proving its ability to effectively capture temporal dependencies and improve predictions in Table 4. While non-Gaussian copulas exhibit inferior performance, they still significantly reduce errors compared to no-copula modeling.

The experimental results demonstrate how the copula modularization enables flexible "plug-and-play" of different multivariate distributions. It will empirically prove the adaptability of our approach to non-Gaussian dependencies, highlighting the methodology's versatility.

4.5. Discussion

Our research presents several advantages, such as the effective use of graph neural networks to capture variable interdependencies and the enhancement of multivariate model predictions through the integration of geographical data. The graph time series model further extends these advancements by enabling simultaneous structure and graph neural network learning, even for graphs with unknown structures, while the incorporation of attention mechanisms has significantly boosted model performance.

5. Conclusions

In this paper, we propose a multivariate spatio-temporal graph model leveraging the attention mechanism and Copula method for electricity consumption forecasting. The model learns the graph structure between multiple time series and predicts them simultaneously with an end-to-end model to maximize the use of pairwise interactions between data streams. A directed graph represents pairwise distance correlations between regions, where the nodes correspond to regional electric companies and the edge weights represent the distances between them. Additionally, a discrete graph structure is employed to depict nonlinear time-dynamic information. The attention mechanism enhances the spatially relevant information of the discrete graph by enabling feature crossover between the two graphs. Our unique approach of modeling using the joint distribution of node results into edge and Copula densities connects the graph structure directly to the Copula covariance matrix. Comparative analysis with contemporary

baseline methods reveals our model's superior predictive accuracy. Future research will focus on optimizing and addressing the challenge of incorporating comprehensive correlation factors to further improve the performance of the proposed model.

CRediT authorship contribution statement

Xihe Qiu: Data curation, Funding acquisition, Investigation, Writing – original draft, Methodology. **Jiahui Qian:** Methodology, Software, Writing – original draft. **Haoyu Wang:** Methodology, Validation, Visualization, Writing – review & editing. **Xiaoyu Tan:** Conceptualization, Formal analysis, Investigation, Project administration, Supervision. **Yaochu Jin:** Project administration, Resources, Supervision, Validation, Writing – review & editing.

Declaration of competing interest

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

Data availability

Data will be made available on request.

Acknowledgments

This work is supported by the National Natural Science Foundation of China, The Research Project of Shanghai Municipal Natural Science Foundation (Grant No. 62102241, No. 23ZR1425400).

References

[1] S. Du, T. Li, Y. Yang, S.-J. Horng, Multivariate time series forecasting via attention-based encoder–decoder framework, Neurocomputing 388 (2020) 269–279.
[2] X. Liu, M. Qin, Y. He, X. Mi, C. Yu, A new multi-data-driven spatiotemporal PM2. 5 forecasting model based on an ensemble graph reinforcement learning convolutional network, Atmos. Pollut. Res. 12 (10) (2021) 101197.
[3] H. Liu, G. Yan, Z. Duan, C. Chen, Intelligent modeling strategies for forecasting air quality time series: A review, Appl. Soft Comput. 102 (2021) 106957.
[4] M. Liu, J. Li, X. Cheng, B. Zhou, Q. Chen, Short term electricity load forecasting using hybrid prophet-LSTM model optimized by BPNN, Energy 224 (2021) 120326.
[5] C. Wang, Y. Wang, Z. Ding, T. Zheng, J. Hu, K. Zhang, A transformer-based method of multienergy load forecasting in integrated energy system, IEEE Trans. Smart Grid 13 (4) (2022) 2703–2714.
[6] G. Shilpa, G. Sheshadri, Short-term load forecasting using ARIMA model for Karnataka state electrical load, Int. J. Eng. Res. Dev. 13 (7) (2017) 75–79.
[7] S. Bercu, F. Proïa, A SARIMAX coupled modelling applied to individual load curves intraday forecasting, J. Appl. Stat. 40 (6) (2013) 1333–1348.
[8] S. Barak, S.S. Sadegh, Forecasting energy consumption using ensemble ARIMA–ANFIS hybrid algorithm, Int. J. Electr. Power Energy Syst. 82 (2016) 92–104.
[9] E. Ceperic, V. Ceperic, A. Baric, A strategy for short-term load forecasting by support vector regression machines, IEEE Trans. Power Syst. 28 (4) (2013) 4356–4364.
[10] A. Rahman, V. Srikumar, A.D. Smith, Predicting electricity consumption for commercial and residential buildings using deep recurrent neural networks, Appl. Energy 212 (2018) 372–385.
[11] K. Wang, K. Li, L. Zhou, Y. Hu, Z. Cheng, J. Liu, C. Chen, Multiple convolutional neural networks for multivariate time series prediction, Neurocomputing 360 (2019) 107–119.
[12] X. Tang, H. Yao, Y. Sun, C. Aggarwal, P. Mitra, S. Wang, Joint modeling of local and global temporal dynamics for multivariate time series forecasting with missing values, in: AAAI, vol. 34, (no. 04) 2020, pp. 5956–5963.
[13] C. Shang, J. Chen, J. Bi, Discrete graph structure learning for forecasting multiple time series, in: International Conference on Learning Representations, 2021.
[14] J. Ma, B. Chang, X. Zhang, Q. Mei, CopulaGNN: Towards integrating representational and correlational roles of graphs in graph neural networks, in: International Conference on Learning Representations, 2020.
[15] Y. Zhang, G. Shen, X. Han, W. Wang, X. Kong, Spatio-temporal digraph convolutional network-based taxi pickup location recommendation, IEEE Trans. Ind. Inform. 19 (1) (2022) 394–403.
[16] X. Qi, G. Mei, J. Tu, N. Xi, F. Piccialli, A deep learning approach for long-term traffic flow prediction with multifactor fusion using spatiotemporal graph convolutional network, IEEE Trans. Intell. Transp. Syst. (2022).
[17] A. Deng, B. Hooi, Graph neural network-based anomaly detection in multivariate time series, in: AAAI, vol. 35, (no. 5) 2021, pp. 4027–4035.
[18] J. Bruna, W. Zaremba, A. Szlam, Y. LeCun, Spectral networks and deep locally connected networks on graphs, in: 2nd International Conference on Learning Representations, ICLR 2014, 2014.
[19] M. Defferrard, X. Bresson, P. Vandergheynst, Convolutional neural networks on graphs with fast localized spectral filtering, in: Proceedings of the 30th International Conference on Neural Information Processing Systems, 2016, pp. 3844–3852.
[20] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, in: ICLR, 2017.
[21] Y. Seo, M. Defferrard, P. Vandergheynst, X. Bresson, Structured sequence modeling with graph convolutional recurrent networks, in: ICONIP 2018, Siem Reap, Cambodia, December 13-16, 2018, Proceedings, Part I 25, Springer, 2018, pp. 362–373.
[22] B. Yu, H. Yin, Z. Zhu, Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting, in: Proceedings of the 27th International Joint Conference on Artificial Intelligence, 2018, pp. 3634–3640.
[23] J. Atwood, D. Towsley, Diffusion-convolutional neural networks, in: Advances in Neural Information Processing Systems, vol. 29, 2016.
[24] Y. Hechtlinger, P. Chakravarti, J. Qin, A generalization of convolutional neural networks to graph-structured data, 2017, arXiv preprint arXiv:1704.08165.
[25] Y. Li, R. Yu, C. Shahabi, Y. Liu, Diffusion convolutional recurrent neural network: Data-driven traffic forecasting, in: ICLR, 2018.
[26] K.-D. Lu, Z.-G. Wu, T. Huang, Differential evolution-based three-stage dynamic cyber-attack of cyber-physical power systems, IEEE/ASME Trans. Mechatronics 28 (2) (2022) 1137–1148.
[27] L. Franceschi, M. Niepert, M. Pontil, X. He, Learning discrete structures for graph neural networks, in: International Conference on Machine Learning, PMLR, 2019, pp. 1972–1982.
[28] J. Ma, W. Tang, J. Zhu, Q. Mei, A flexible generative framework for graph-based semi-supervised learning, Adv. Neural Inf. Process. Syst. 32 (2019).
[29] M. Qu, Y. Bengio, J. Tang, Gmnn: Graph markov neural networks, in: International Conference on Machine Learning, PMLR, 2019, pp. 5241–5250.
[30] J. Jia, A.R. Benson, Residual correlation in graph neural network regression, in: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 588–598.
[31] F. Zhao, G.-Q. Zeng, K.-D. Lu, EnLSTM-WPEO: Short-term traffic flow prediction by ensemble LSTM, NNCT weight integration, and population extremal optimization, IEEE Trans. Veh. Technol. 69 (1) (2019) 101–113.
[32] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, in: International Conference on Learning Representations, 2015.
[33] J. Bedi, D. Toshniwal, Deep learning framework to forecast electricity demand, Appl. Energy 238 (2019) 1312–1326.
[34] W. Xu, H. Peng, X. Zeng, F. Zhou, X. Tian, X. Peng, A hybrid modelling method for time series forecasting based on a linear regression model and deep learning, Appl. Intell. 49 (8) (2019) 3002–3015.
[35] Y. Yang, Z. Tao, C. Qian, Y. Gao, H. Zhou, Z. Ding, J. Wu, A hybrid robust system considering outliers for electric load series forecasting, Appl. Intell. 52 (2) (2022) 1630–1652.
[36] X. Qiu, Y. Ru, X. Tan, J. Chen, B. Chen, Y. Guo, A k-nearest neighbor attentive deep autoregressive network for electricity consumption prediction, Int. J. Mach. Learn. Cybern. (2023) 1–12.
[37] E. Jang, S. Gu, B. Poole, Categorical reparameterization with gumbel-softmax, in: International Conference on Learning Representations, 2017.
[38] M. Sklar, Fonctions de repartition an dimensions et leurs marges, Publ. Inst. Statist. Univ. Paris 8 (1959) 229–231.
[39] J. Chung, C. Gulcehre, K. Cho, Y. Bengio, Empirical evaluation of gated recurrent neural networks on sequence modeling, in: NIPS 2014 Workshop on Deep Learning, December 2014, 2014.
[40] S. Zhang, X. Li, M. Zong, X. Zhu, D. Cheng, Learning k for knn classification, ACM Trans. Intell. Syst. Technol. 8 (3) (2017) 1–19.
[41] P. Jaworski, F. Durante, W.K. Härdle, Copula Theory and Its Applications, vol. 198, Springer, 2010.
[42] G. Geenens, Copula modeling for discrete random vectors, Depend. Model. 8 (1) (2020) 417–440.
[43] H. Joe, Dependence Modeling with Copulas, CRC Press, 2014.
[44] M.R. Abrigo, I. Love, Estimation of panel vector autoregression in Stata, Stata J. 16 (3) (2016) 778–804.
[45] K. Xu, J. Ba, R. Kiros, et al., Show, attend and tell: Neural image caption generation with visual attention, in: Proceedings of the 32nd International Conference on Machine Learning, vol. 37, PMLR, 2015, pp. 2048–2057.
[46] G. Zhao, J. Lin, Z. Zhang, et al., Explicit sparse transformer: Concentrated attention through explicit selection, 2019, arXiv preprint arXiv:1912.11637.