energies

MDPI

Article

Multi-Site Wind Speed Prediction Based on Graph Embedding and Cyclic Graph Isomorphism Network (GIN-GRU)

Hongshun Wu and Hui Chen *

School of Electrical and Electronic Engineering, Hubei University of Technology, Wuhan 430068, China; w2379439736@gmail.com

* Correspondence: chenhui@hbut.edu.cn

Abstract: Accurate and reliable wind speed prediction is conducive to improving the power generation efficiency of electrical systems. Due to the lack of adequate consideration of spatial feature extraction, the existing wind speed prediction models have certain limitations in capturing the rich neighborhood information of multiple sites. To address the previously mentioned constraints, our study introduces a graph isomorphism-based gated recurrent unit (GIN-GRU). Initially, the model utilizes a hybrid mechanism of random forest and principal component analysis (PCA-RF) to discuss the feature data from different sites. This process not only preserves the primary features but also extracts critical information by performing dimensionality reduction on the residual features. Subsequently, the model constructs graph networks by integrating graph embedding techniques with the Mahalanobis distance metric to synthesize the correlation information among features from multiple sites. This approach effectively consolidates the interrelated feature data and captures the complex interactions across multiple sites. Ultimately, the graph isomorphism network (GIN) delves into the intrinsic relationships within the graph networks and the gated recurrent unit (GRU) integrates these relationships with temporal correlations to address the challenges of wind speed prediction effectively. The experiments conducted on wind farm datasets for offshore California in 2019 have demonstrated that the proposed model has higher prediction accuracy compared to the comparative model such as CNN-LSTM and GAT-LSTM. Specifically, by modifying the network layers, we achieved higher precision, with the mean square error (MSE) and root mean square error (RMSE) of wind speed at a height of 10 m being 0.8457 m/s and 0.9196 m/s, respectively.

Keywords: multisite wind prediction; PCA-RF; graph embedding; graph neural network; convolutional neural network

check for updates

Citation: Wu, H.; Chen, H. Multi-Site Wind Speed Prediction Based on Graph Embedding and Cyclic Graph Isomorphism Network (GIN-GRU). *Energies* **2024**, *17*, 3516. https://doi.org/10.3390/en17143516

Academic Editor: Frede Blaabjerg

Received: 31 May 2024
Revised: 27 June 2024
Accepted: 10 July 2024
Published: 17 July 2024

1. Introduction

The growing global energy crisis and the critical issue of environmental pollution have highlighted the need for clean and renewable energy sources. Wind energy has garnered significant attention due to its relatively short construction cycle, minimal environmental prerequisites, and vast reserves [1]. This has led to its widespread adoption and rapid development globally. Many nations have recognized the potential of wind power and are actively promoting its generation to capture wind energy. Consequently, the wind power sector has undergone rapid growth and expansion, firmly establishing wind power generation as a field with promising current prospects. Presently, wind speed prediction plays a pivotal role in formulating control strategies for wind farms, which stands as a cornerstone technology that enhances the operational efficiency of wind turbines [2].

Accurate wind speed prediction is crucial to optimizing wind power generation. It enhances wind energy utilization, mitigates wind power's grid impact, and ensures efficient operation of wind farms. To refine wind speed forecasting, researchers worldwide have pioneered various innovative approaches. For instance, data decomposition techniques like wavelet transform [3], empirical modal decomposition [4–6], and variational modal decomposition [7] are employed to analyze wind speed's multiscale attributes. Additionally, data fusion strategies such as Kalman filtering, data assimilation, and multi-source data fusion are utilized [8–10]. These methods provide a comprehensive analysis of various meteorological elements and data sources, which improves the accuracy of weather forecasting and monitoring.

Traditional statistical methods often focus on specific learning facets. With the accumulation of a large amount of historical operational data in the wind power industry, deep learning methods based on large-scale and multi-dimensional data have been increasingly applied to the field of wind speed prediction due to their powerful nonlinear mapping ability. H Xu et al. developed a hybrid deep learning model for wind speed prediction that integrates grid search optimization with a recurrent neural network [11]. This approach effectively corrected the errors of numerical weather prediction to enhance both the precision and dependability of wind speed forecasts. Ai X et al. proposed a wind speed prediction model based on data augmentation and deep learning. The results showed that empirical mode decomposition and data augmentation techniques generated more training data which improved the generalization ability as well as robustness of the wind speed prediction model [12]. B Shao et al. utilized a configuration of long short-term memory (LSTM) networks with diverse architectural complexities to uncover the underlying patterns in wind speed time series data [13]. The researchers integrated the predictive outputs using a nonlinear regression layer, which consisted of support vector regression machines. This integration resulted in a combined prediction model that exhibited high accuracy. However, the abovementioned wind speed prediction methods mostly use the historical and real-time data of wind speed at a single point. Thus, the prediction accuracy of the models still requires further improvement and augmentation.

The integration of comprehensive spatio-temporal data from multiple locations can provide a novel approach to predicting wind speeds. Q Zhu et al. leveraged the spatio-temporal correlation characteristics across various spatial scales to improve wind speed prediction [14]. By employing convolutional neural networks (CNN) for spatial modeling, the method captures the intricate spatial correlations between the macroscopic and microscopic levels, which offers a more comprehensive approach than traditional one-to-one modeling methods. Trebing K et al. employ a multiscale convolutional neural network filter to extract the multi-features influencing long-term wind speed distribution [15]. It introduces a deep convolutional recurrent neural network model for wind speed prediction at various points within a wind farm. This model excels at feature extraction and time-series analysis, which enables simultaneous ultra-short-term wind speed forecasts for each turbine. By integrating spatial and temporal flow field data, the model enhances overall prediction accuracy. W Tuerxun et al. proposed a novel spatio-temporal neural network that employs a deep convolutional neural network coupled with a bidirectional gated recurrent unit [16]. This model comprehensively captures the spatio-temporal dynamics between wind speed and direction across various altitudes within a wind farm, as well as pertinent NWP data. MM Yuan et al. combine CNN with LSTM networks to propose a multifactor spatio-temporal correlation model [17]. It introduces a data representation technique that uses three-dimensional matrices. This innovative model demonstrates enhanced predictive performance in wind speed forecasting. Therefore, by considering not only the direct spatio-temporal distribution of wind speed within the wind farm but also the potential spatio-temporal relationships of wind speed at multiple sites using deep learning algorithms, it will be probable to further improve the prediction accuracy.

This paper introduces a deep learning-based integrated multi-site wind speed prediction model for wind farms, designated as RPCA-GIN-GRU. This model enhances the predictive accuracy for each site by leveraging the feature correlations across multiple sites. Firstly, we apply PCA-RF to the feature set, which captures the essential features and extracts key insights by reducing the dimensions of the remaining features. Secondly, we use a method that combines graph embedding techniques with the Mahalanobis distance metric to construct the network graph's edges, which facilitates a thorough analysis of the relationships between each site's features. Ultimately, the GIN model is leveraged for its proficient learning of graph structure similarities. By conducting deep learning on the spatial attributes of these graph structures, we can then feed the deeply learned feature data into a GRU network for subsequent prediction. This approach generates wind speed predictions that considers the characteristics of each site and their interconnectedness to generate wind speed predictions. Additionally, it summarizes the interactive effects between sites on the predicted wind speeds.

2. Materials and Methods

2.1. Construction of Graph Networks

2.1.1. PCA-RF Fusion Model

Raw meteorological data are complex and multifaceted. However, not all variables are pertinent to changes in wind speed. An overabundance of predictive variables can introduce redundancy, thereby diminishing the model's generalization capabilities. Some residual features have some key information that needs to be extracted. Therefore, we focus on primary feature extraction and dimensionality reduction of residual features from the original meteorological elements. This approach streamlines the dataset and enhances model interpretability and efficiency. Consequently, this research leverages the intrinsic feature extraction capabilities of the random forest algorithm in conjunction with PCA to further diminish the dimensionality of residual features. This approach enables the separate processing of primary and residual features to achieve maximal efficiency. Subsequently, the primary features and the dimensionality-reduced residual features are concatenated to form the dataset used for constructing graph networks.

2.1.2. Random Forests

Random forest is a kind of machine learning algorithm that makes predictions from multiple decision trees and integrates their results. Random forests construct each decision tree using random samples and features, which imparts the model with robustness against overfitting [18]. MDI is a measure of feature importance based on the reduction in Gini impurity of each feature at the split point in the decision tree [19]. Given a dataset containing nodes from C categories, where the probability that node j belongs to category c is denoted as p_j, the Gini impurity of node j is as follows:

$$Gini(j) = 1 - \sum_{c=1}^{C} p_j^2 \tag{1}$$

Gini impurity is a statistical metric that measures the probability of a random classification error for an element chosen from a dataset, given that the classification is random and reflects the distribution of classes in the dataset. Owing to its simplicity and direct computation, MDI is selected for evaluating the significance of feature variables within this study. The MDI value signifies a feature's greater relevance in strengthening the predictive accuracy of the model. The steps for calculating MDI are given as follows:

Step 1. Each decision tree generated through bootstrap sampling on the training set constitutes a random forest.

Step 2. For every tree, the Gini impurity of each node is computed.

Step 3. MDI for each feature is ascertained by averaging the reduction in Gini impurity across all trees.

2.1.3. Principal Component Analysis

PCA is a prevalent technique for reducing data dimensionality. It employs orthogonal transformations to convert correlated variables into a new set of uncorrelated variables, known as principal components [20]. PCA is designed to preserve critical information from the original dataset and reduce dimensionality by decreasing the number of variables. PCA streamlines complex datasets by isolating pivotal features to reduce noise and safeguard essential information.

The fundamental method involves computing the eigenvalues and eigenvectors from the covariance matrix. The eigenvalues represent the core information of the original variables, which are crucial to the remaining features. Subsequently, the dimensionality reduction of the feature matrix facilitates the isolation of distinctive meteorological factors that exert a significant influence on wind speed variations.

2.1.4. PCA-RF Feature Fusion

In this study, the primary and residual feature matrices are based on the ordering of features according to their MDI scores. The residual feature matrix is then subjected to PCA for dimensional reduction. For a wind field with N sites, the feature matrix of the n-th site is expressed as $\boldsymbol{X_n} = \left[\boldsymbol{V}_1^{\boldsymbol{T}}, \boldsymbol{V}_2^{\boldsymbol{T}} \cdots \boldsymbol{V}_{\boldsymbol{M}}^{\boldsymbol{T}}\right]$, where $n = 1, \ldots, N$, M is the number of features per site, the total number of features is $M \times D$ and $V_i = [v_i(1), v_i(2) \cdots v_i(t)]$ $(i = 1, \ldots, M\ t = 1, \ldots, D)$ is the feature vector of the site. This dataset primarily includes measurements taken at 15-m intervals for wind speed, wind direction, temperature, pressure, and other relevant characteristics. The steps of wind speed feature processing for the n-th site are given as follows:

(1) *Build a random forest.* Bootstrap sampling and random feature selection techniques are utilized to generate a training dataset and a corresponding subset of features. These elements form the basis for constructing an individual decision tree. The process of building a decision tree involves randomly selecting a subset of the dataset and recursively splitting the nodes based on optimal segmentation criteria until the stopping criteria of the random forest are met. Through the repetition of this process, a random forest model is assembled with I different decision trees, where the i-th decision tree has T nodes and C categories.

(2) *Calculate the Gini impurity and the amount of its variation.* In tree i, at node j, it is essential to calculate the proportion of category c, represented as $p(c|(i,j),k)$, against the total categories. The variation in Gini impurity is then determined by evaluating the Gini impurity before and after the branching of node j. The calculation formula is as follows:

$$Gini(i,j,k) = 1 - \sum_{c=1}^{C} p(c|(i,j),k)^2 \tag{2}$$

$$\Delta Gini(i,j,k) = Gini(i,j,k) - Gini(i,j_b,k) - Gini\left(i,j_f,k\right) \tag{3}$$

where $j = 1, 2 \ldots T$, $i = 1, 2 \ldots I$, $k = 1, 2 \ldots M$, $c = 1, 2 \ldots C$. j_b and j_f represent the two new nodes after node j is branched. $\Delta Gini(i,j,k)$ are feature vector $\boldsymbol{V_k}$ reduction of Gini impurity after node splitting in decision tree i.

(3) *Calculate the MDI.* MDI for feature $\boldsymbol{V_k}$ is as follows:

$$M(\boldsymbol{V_k}) = \frac{1}{I} \sum\nolimits_{t=1}^{T} \Delta Gini(i,j,k) \tag{4}$$

(4) *Feature selection.* In the feature selection phase, features are ranked by their MDI values. Subsequently, a threshold denoted by γ is determined. Features with an MDI value of $M(V_k)$ are selected to comprise the primary feature matrix X_f, which includes M_f features. Conversely, features with an MDI value of $M(V_k) < \gamma$ are used to construct the auxiliary matrix X_b, which incorporates the remaining $M_b = M - M_f$ features.

(5) *Feature decentralization.* The auxiliary matrix X_b is decentralized to yield the matrix X_b'.

$$X_b' = X_b - \overline{X_b} \tag{5}$$

where $\overline{X_b}$ is the auxiliary matrix X_b.

(6) *Compute eigenvalues and eigenvectors.* The eigenvalue decomposition is performed on the covariance matrix. The feature values are ranked in descending order and

sequentially aggregated until the cumulative contribution rate, denoted by $\eta \geq 0.85$. Consequently, in adherence to this criterion, the primary r features are chosen to constitute the matrix R of $r \times D$.

$$\lambda l = \frac{1}{D-1} X_b{}'^T X_b{}' l \tag{6}$$

$$\eta = \frac{\sum_{i=1}^{r} \lambda_i}{\sum_{k=1}^{M} \lambda_k} \tag{7}$$

where $\lambda = \{\lambda_1, \lambda_2 \ldots \lambda_{M_b}\}$ is a set of eigenvalues, $l = [l_1, l_2 \ldots l_{\mathrm{M_b}}]$ is the matrix of eigenvectors.

(7) *Data dimensionality reduction.* We calculate the dimensionality reduction matrix Y utilizing the subsequent formula:

$$Y = RX_b \tag{8}$$

(8) *Matrix splicing.* By horizontally concatenating the matrices X_f and Y, we obtain a matrix W_n of dimension $\left(M_f + r\right) \times D$.

$$W_n = \left[X_f Y\right] \tag{9}$$

PCA-RF processes all sites to yield a sequence of feature matrices $W_1, W_2 \ldots W_N$, which captures the pivotal features of the dataset. Finally, all the feature matrices are horizontally spliced to obtain a feature matrix W of $g \times D$.

$$W = [W_1 W_2 \ldots W_N] \tag{10}$$

where $g = N * \left(M_f + r\right)$. Figure 1 illustrates the steps of PCA-RF. Initially, the raw datasets from different sites are processed through RF algorithm to extract the primary feature matrix and the auxiliary feature matrix. Subsequently, the auxiliary feature matrix is subjected to PCA for dimensionality reduction to yield the dimensionality reduction matrix. Finally, the final feature matrix is constructed by concatenating the primary feature matrix with the dimensionality reduction matrix.

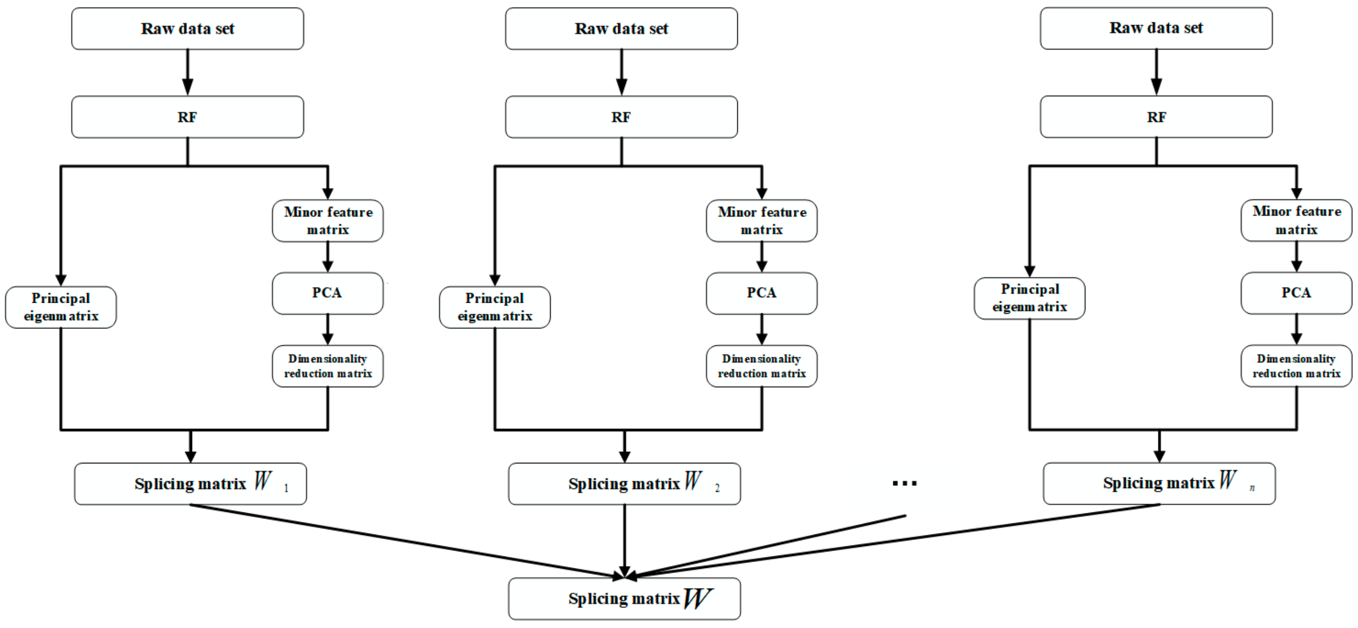


Figure 1. The flow chart of PCA-RF.

2.1.5. Integrative Modeling of Graph Embeddings and Mahalanobis Distances

Graph neural networks combined with graph networks on some feature correlations have achieved excellent results in wind speed prediction [21,22]. However, the performance of graph neural networks is highly dependent on predefined graphs to characterize the relationships between site features, such as those based on the Pearson correlation coefficients. Predefined graphs often do not reflect the complex dynamics of wind speed features, with their quality significantly depending on expert judgment and accurate wind speed measurements. Considering these factors, this paper introduces a novel compositional algorithm to analyze the spatio-temporal attributes of wind speed. The algorithm acquires node embedding vectors by employing graph embedding techniques on a predefined graph, which elucidate the graph's topology and the inter-node relationships [23]. Subsequently, it quantifies the distance between node embedding vectors by employing the Mahalanobis distance [24,25]. The final graph network results from the adjustment of edge relationships in the predefined graph through a filter based on a threshold Mahalanobis distance.

2.1.6. Construction of Predefined Graphs

In this paper, we use the matrix W obtained above to construct the predefined graph $G = (\boldsymbol{V}, E)$, where $\boldsymbol{V} = \left\{\boldsymbol{V}_1^T, \boldsymbol{V}_2^T, \boldsymbol{V}_3^T, \ldots \boldsymbol{V}_g^T\right\}$ represents the set of nodes. We consider each column of data of matrix W as the feature vector of a node and the dimension is $g \times D$ where g is the number of nodes and D is the dimension of each node feature. The set of edges $E = \left(\boldsymbol{e_{ij}}\right)_{i,j=1}^{v}$ embodies the connectivity relationships among nodes within the network, such that there exists an edge connecting any two nodes $\boldsymbol{V_i}$ and $\boldsymbol{V_j}$. In the predefined graphs, the existence of an edge is denoted by $\boldsymbol{e_{ij}} = 1$, which is presented as follows:

$$e_{ij} = \begin{cases} 1, & \frac{\sum_{t=1}^{D} (v_i(t)-\overline{v}_i)\left(v_j(t)-\overline{v}_j\right)}{\sqrt{\sum_{t=1}^{D} (v_i(t)-\overline{v}_i)^2}\sqrt{\sum_{t=1}^{D} \left(v_j(t)-\overline{v}_j\right)^2}} < \delta \\ 0, & \frac{\sum_{t=1}^{D} (v_i(t)-\overline{v}_i)\left(v_j(t)-\overline{v}_j\right)}{\sqrt{\sum_{t=1}^{D} (v_i(t)-\overline{v}_i)^2}\sqrt{\sum_{t=1}^{D} \left(v_j(t)-\overline{v}_j\right)^2}} > \delta \end{cases} \tag{11}$$

where δ is the correlation threshold. $\overline{v}_i$ represents the mean value of the i-th node.

Graph embedding is a process that maps graph data into low-dimensional, dense vectors, preserving the graph's structure and properties. This process facilitates node classification, clustering, link prediction, graph reconstruction, and visualization [26]. In this study, graph embedding techniques are employed to transform a predefined graph into a continuous vector space. This conversion enables the effective learning of features and facilitates subsequent data analysis tasks. Certain shallow graph embedding techniques are initiated by randomly selecting neighboring nodes within the network to create a fixed-length random walk sequence. This sequence is then utilized by the skip-gram model to project the sequence of nodes into a low-dimensional space, which results in an embedding vector that captures the essential structural features of the graph [27]. This method effectively reduces the dimensionality and maintains node contextuality. The specific process is shown in Figure 2. However, the shallow model is unable to capture the highly nonlinear structure, which in turn leads to the generation of non-optimal solutions. Recent breakthroughs in deep learning have profoundly impacted graph analysis, as deep neural network techniques are increasingly employed to advance graph embedding methodologies [28]. Then we use deep neural networks to perform a nonlinear transformation of node features and neighborhood information, which leads to the generation of node embedding vectors that capture the graph's higher-order dependencies. This approach allows for a more profound understanding of node characteristics, which optimizes the graph's structural representation.

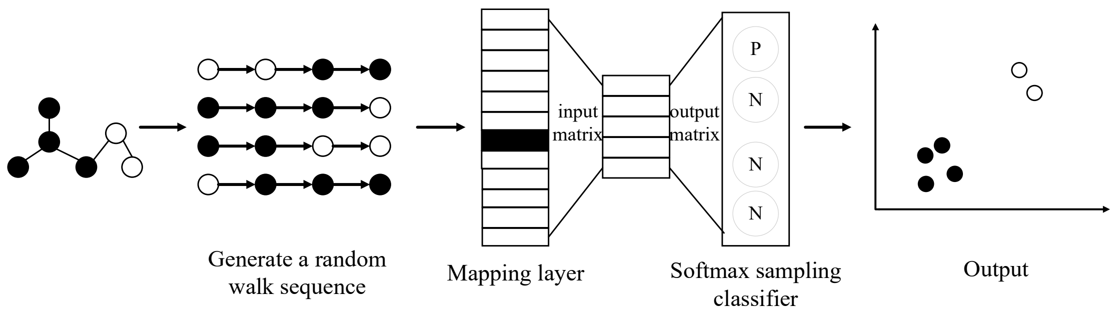


Figure 2. The flow chart of graph embedding.

2.1.7. GraphSAGE-Based Graph Embedding

We employ a deep learning approach grounded in GraphSAGE for graph embedding. This method leverages node feature information to generate embedding vectors for new nodes or subgraphs via an enhanced graph convolutional network. It facilitates incremental updates to node embeddings and preserves the graph's features and structural information. Based on this groundwork, the algorithm establishes 'edges' that precisely delineate the relationships between nodes. The specific process is given as follows:

A. *Sampling*: for node $\boldsymbol{V_i}$, we randomly sample a subset of its neighboring nodes $N(V_i)$ from the set of its neighbor nodes S_k to create a subgraph. This method decreases computational demands yet preserves the heterogeneity among adjacent nodes.

B. *Aggregation*: for node $\boldsymbol{V_i}$, we employ an aggregation function $AGGREGATE^k$ that compresses and transforms the feature vectors of its neighboring nodes $\left\{\boldsymbol{h_u^{k-1}}, \forall \boldsymbol{u} \in N(\boldsymbol{V_i})\right\}$ to generate a new feature vector $\boldsymbol{h_{V_i}^k}$ through aggregation. The formula for aggregation is expressed as

$$\boldsymbol{h_{V_i}^k} = \sigma\Big(Q^k \cdot CONCAT\Big(\boldsymbol{h_{V_i}^{k-1}}, AGGREGATE^k\Big(\Big\{\boldsymbol{h_u^{k-1}}, \forall u \in N(V_i)\Big\}\Big)\Big)\Big) \tag{12}$$

where $\boldsymbol{h_{V_i}^k}$ denotes the node $\boldsymbol{V_i}$ in the first k layer of the embedding vector, and $\boldsymbol{h_{V_i}^{k-1}}$ is the feature vector of node $\boldsymbol{V_i}$. σ denotes the sigmoid activation function. The matrix Q^k represents the learnable weights. $CONCAT$ signifies the splicing operation. $AGGREGATE^k$ is the aggregation function at the first k layer and $N(\boldsymbol{V_i})$ refers to the node $\boldsymbol{V_i}$ neighboring set.

C. *Update*: for node $\boldsymbol{V_i}$, the feature vector $\boldsymbol{h_{V_i}^{k-1}}$ is concatenated with the aggregated feature vector $\boldsymbol{h_{V_i}^k}$. This combined vector then passes through a fully connected layer followed by an activation function, which results in the embedding vector $\boldsymbol{h_{V_i}^{k+1}}$ for node $\boldsymbol{V_i}$. This process facilitates the integration and nonlinear transformation of the features represented by node $\boldsymbol{V_i}$ with those of its neighboring nodes. The updated formula is expressed as follows:

$$\boldsymbol{h_{V_i}^{k+1}} = \sigma\Big(Q^{k+1} \cdot COMBINE\Big(\boldsymbol{h_{V_i}^{k-1}}, \boldsymbol{h_{V_i}^{k}}\Big)\Big) \tag{13}$$

where $COMBINE$ denotes the splicing or summing operation and Q^{k+1} denotes the learnable weight matrix.

2.1.8. Modification of Graph Network Edges

The node embedding vectors obtained through GraphSAGE graph embedding capture the features of the nodes and the relationships between the nodes. However, they are not directly used as components of the graph network. In this study, the edge connections are determined by setting a similarity threshold based on the Mahalanobis distance, which dictates the connectivity between nodes.

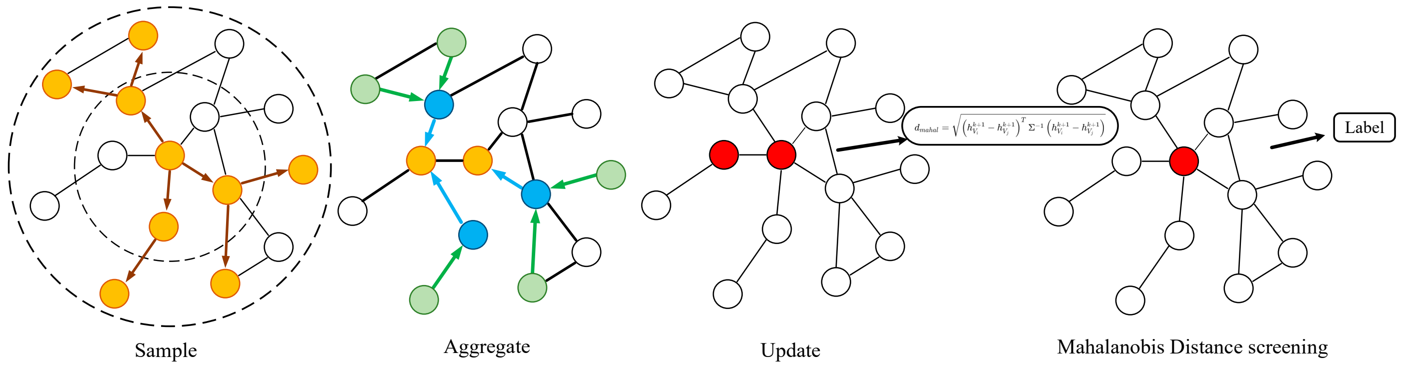
The Mahalanobis distance quantifies the similarity or divergence between two data samples by incorporating their covariance matrix. This metric excels at handling features with varying scales and interdependencies, which effectively neutralizes the impact of scale disparities and feature correlations. Unlike traditional distance measures (such as Euclidean distance), which often assume that individual features are independent, real-world data often exhibit correlations between features. The Mahalanobis distance accounts for these correlations by utilizing the covariance matrix. Consequently, we employ the Mahalanobis distance method for optimizing node embedding vectors. This method is particularly suitable for high-dimensional data influenced by numerous meteorological factors and can accommodate the conditions of non-independent as well as identically distributed dimensions. Its calculation formula is expressed as follows:

$$d_{mahal} = \sqrt{\left(\boldsymbol{h}_{V_i}^{k+1} - \boldsymbol{h}_{V_j}^{k+1}\right)^T \Sigma^{-1}\left(\boldsymbol{h}_{V_i}^{k+1} - \boldsymbol{h}_{V_j}^{k+1}\right)} \tag{14}$$

where $\boldsymbol{h}_{V_i}^{k+1}$ denotes the node embedding vector. The covariance matrix is represented by Σ, and its inverse is denoted as Σ^{-1}. Finally, we can compute the dimension d_{mahal} and the correlation threshold α to determine the connectivity of the optimized edges $E' = (e_{ij})_{i,j=1}^{v}$ within the predefined graph network $G = (V, E)$.

$$e_{ij} = \begin{cases} 1, & d_{mahal} < \alpha \\ 0, & d_{mahal} > \alpha \end{cases} \tag{15}$$

A new graph network $G' = (\boldsymbol{V}, E')$ is thus created. The process of embedding the graph network is illustrated in Figure 3.



Sample
Aggregate
Update
$d_{mahal} = \sqrt{\left(h_{V_i}^{k+1} - h_{V_j}^{k+1}\right)^T \Sigma^{-1}\left(h_{V_i}^{k+1} - h_{V_j}^{k+1}\right)}$
Label
Mahalanobis Distance screening

Figure 3. The flow chart of graph creation.

2.2. Spatio-Temporally Integrated Forecasting Model Based on GIN and GRU

2.2.1. Graph Isomorphism Network

GIN is a robust method for learning graph representations, which closely approximates the graph isomorphism test (WL-test) performance [29,30]. It effectively distinguishes between non-isomorphic graphs. GIN excels at learning node embedding vectors, discerning patterns within graphs to enhance efficiency and capturing structural dependencies to improve performance. The framework of GIN proceeds as follows:

(1) *Aggregation*. The GIN model employs a summation aggregation function to compile the feature vectors $\boldsymbol{h}_{\boldsymbol{u}}^{k-1}$ from the neighboring nodes $\boldsymbol{u} \in \boldsymbol{N}(\boldsymbol{V_i})$ of node $\boldsymbol{V_i}$, which aims to integrate the information from all adjacent nodes.

$$\boldsymbol{a}_{V_i}^{k} = \sum\nolimits_{\boldsymbol{u} \in \boldsymbol{N}(\boldsymbol{V_i})} \boldsymbol{h}_{\boldsymbol{u}}^{k-1} \tag{16}$$

where $\boldsymbol{a}_{V_i}^{k}$ is the temporary aggregation result at layer k, which contains only the information of neighboring nodes.

(2) *Combination*. The aggregated features of neighboring nodes are combined with the target node $\boldsymbol{V_i}$ and the features from the previous layer $\boldsymbol{h_{V_i}^{k-1}}$ to form a new node feature $\boldsymbol{h_{V_i}^{k}}$. This process, facilitated by learnable parameters and the nonlinear transformations of a multilayer perceptron (MLP), enhances the model's ability to learn and represent complex patterns.

$$\boldsymbol{h_{V_i}^{k}} = \boldsymbol{MLP^k}\left(\left(1+\epsilon^k\right)\cdot \boldsymbol{h_{V_i}^{k-1}} + \boldsymbol{a_{V_i}^{k}}\right) \tag{17}$$

where $\epsilon^{\mathbf{k}}$ is a trainable parameter that allows the model to adjust the self-loop when updating the node features with the contribution. $\boldsymbol{h_{V_i}^{k}}$ is the final node feature representation after combination.

2.2.2. Gated Recurrent Unit

GRU is a variant of the recurrent neural network (RNN) designed for processing sequential data, including natural language, speech, and video. Characterized by its dual gating mechanisms—the reset and update gates—GRU regulates the information flow and memory within the network [31,32]. This architecture effectively addresses the vanishing gradient problem, preserves long-term dependencies, and enhances model performance. GRU is particularly adept at extracting temporal correlations, which makes it a prime choice for the second layer in deep learning neural networks. The procedural steps are as follows:

(1) *Determine the values of the update and reset gates*. The update gate assesses the degree to which the hidden state from the preceding timestep is retained in the current timestep. Conversely, the reset gate regulates the proportion of the previous timestep's hidden state that is incorporated into the computation of the current state. The formula for this step is given by

$$\begin{aligned} z_t &= \sigma(U_z[h_{t-1}, x_t] + b_z) \\ r_t &= \sigma(U_r[h_{t-1}, x_t] + b_r) \end{aligned} \tag{18}$$

where z_t and r_t denote the values of the update gate and the reset gate, respectively. The sigmoid activation function, represented by σ, is utilized to facilitate the computation of gradients and enable effective backpropagation. The weight matrices U_z and U_r, along with the bias vectors b_z and b_r, play a pivotal role in this mechanism. Additionally, b_r represents the hidden state from the previous timestep, while x_t corresponds to the input at the current timestep.

(2) *Compute the candidate hidden state*. This represents the candidate hidden state, obtained by applying the hyperbolic tangent (*tanh*) activation function to the current input and the previously reset hidden state. The formulas are expressed as follows:

$$\tilde{h}_t = tanh(U_h[r_t \odot h_{t-1}, x_t] + b_h) \tag{19}$$

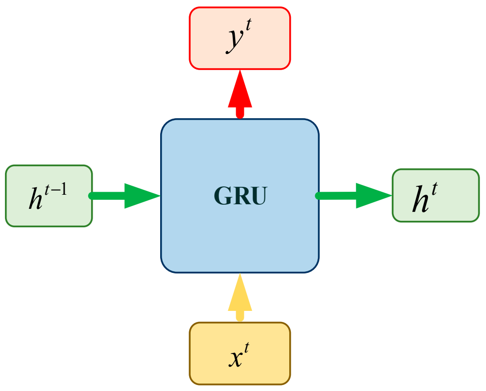
where h_{t-1} is the candidate hidden state, U_h is the weight matrix, b_h is the bias vector, and $\odot$ represents the element-wise multiplication operation.

(3) *Compute the current hidden state*. The current hidden state is computed as the weighted average of the previous hidden state and the candidate hidden state, where the weights are governed by the update gate. The formulas are expressed as follows:

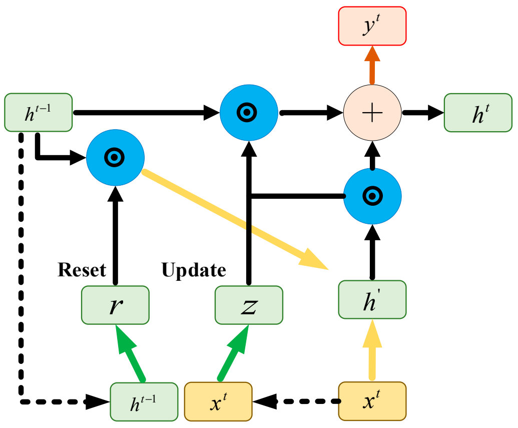
$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \overset{\sim\prime}{h}_t \tag{20}$$

where h_t represents the hidden state at the current moment, z_t denotes the value of the update gate, and $\overset{\sim\prime}{h}_t$ is the candidate hidden state.

These equations collectively illuminate the dynamic update mechanism of the hidden state at each timestep, as orchestrated by gated recurrent units (GRUs). This update methodology is intricately designed to capture long-term dependencies and complex patterns inherent in sequential data. Illustrated in Figure 4, the GRU's input–output structure includes the current input x_t, as well as the transmitted hidden state h_{t-1} from the previous timestep, which retains essential historical information. By amalgamating x_t and h_{t-1}, the GRU calculates the hidden node output y^t for the current timestep and the hidden state h_t, which will be forwarded to the next timestep.



(**a**) Structure of GRU



(**b**) Flow of GRU

Figure 4. GRU visual flow chart.

2.3. Hybrid Model of GE-GIN-GRU Network

In our study, we explore wind speed prediction models based on GIN and GRU. These models capture both multi-site wind speed spatio-temporal correlations and wind direction spatio-temporal correlations, which facilitate accurate local wind speed predictions across various time domains. The data such as wind direction data, wind speed information, temperature information, and barometric pressure information are independent time series from each other. To incorporate feature information affecting loads and prepare it as input for the GIN, we need to process this information to construct a network graph. In our study, we use the PCA-RF method to optimize the nodes. Furthermore, more accurate edge representations are obtained by employing graph embedding techniques in conjunction with Mahalanobis distance. Graph embedding techniques are intended to augment the accuracy of graph network within the GIN framework.

Firstly, at the node level, primary features are extracted using PCA-RF and residual features are dimensionally reduced to retain essential information. This process yields the $g \times D$ fusion matrix W, which serves as the nodes in the predefined graph $G = (\boldsymbol{V},E)$.

Secondly, at the edge level, the predefined graph $G = (\boldsymbol{V},E)$ is subject to aggregation and subsequent updating to obtain the node embedding vector. Subsequently, the Mahalanobis distance between node embedding vectors is employed as a discriminative criterion to establish the edges within the newly formulated graph network $G' = (\boldsymbol{V}, E')$.

Finally, we use the new graph network $G' = (\boldsymbol{V}, E')$ as the input for the GIN-GRU network. The structure of the proposed GIN-GRU network model is illustrated in Figure 5. Within this model, the GIN component primarily handles feature extraction; furthermore, the GRU network focuses on load prediction. The GIN is structured with two convolutional layers, which sequentially perform the aggregation and combination operations. In the GRU network component, we observed that a greater number of GRU network units, which adds to the model's depth, enhances its prediction capability. Consequently, the proposed model includes two layers of GRU networks, with 128 neurons in each layer. In each layer of the GRU network, random deactivation is employed to prevent overfitting. Ultimately, the wind speed prediction vector is generated through the fully connected layer (Dense).

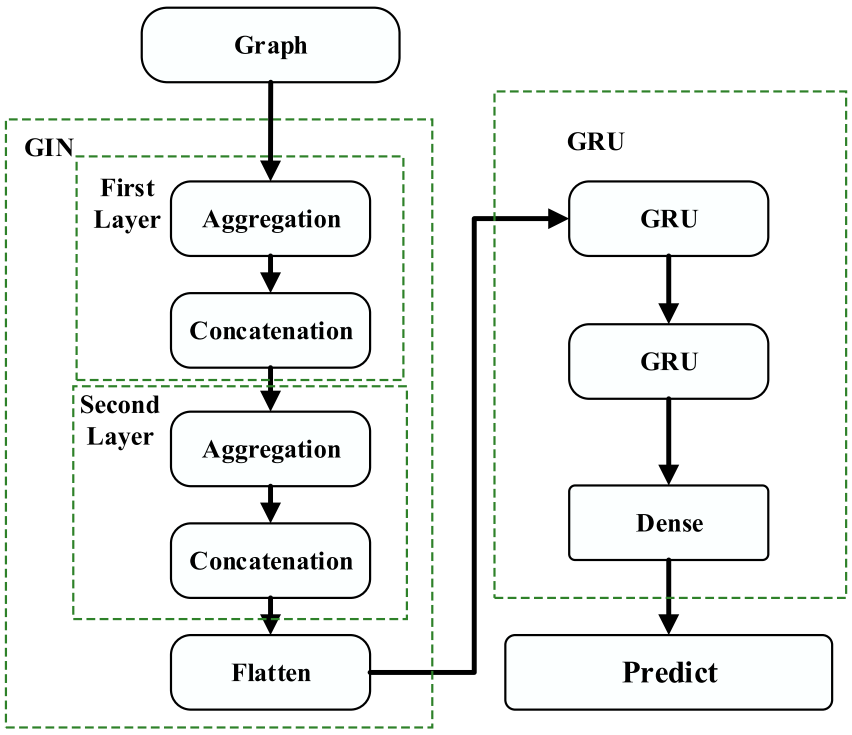


Figure 5. The flow chart of GIN-GRU.

3. Results

This subsection is structured into four different parts: the first part introduces the dataset, the second part analyzes the dimension reduction results of PCA-RF on the dataset, the third part discusses the results of graph networks for graph embedding, and the fourth part examines the prediction results of different models on the same test data.

3.1. Datasets and Settings

3.1.1. Dataset

The dataset is historical data for 20 sites along the California coast. Site IDs are 0, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 69, 70, 71, 72, 73, 74, 75, and 76 for 200 days at 15 min intervals. The size of the preprocessed dataset is set to 5000 and the ratio of the training, validation, and test sets is set to 6:2:2. Table 1 describes the characteristics of the dataset for each site.

3.1.2. Experimental Equipment

This experiment implements the TensorFlow 2.10.0 framework in Python 3.10.9 and accelerates computation through Compute Unified Device Architecture (CUDA). The simulation hardware platform features an Intel Core i7-10875H CPU (manufactured by Intel Corporation, Santa Clara, CA, USA) running at 2.30 GHz with 32GB of RAM, complemented by an Nvidia GeForce RTX 2070 GPU (manufactured by NVIDIA Corporation, Santa Clara, CA, USA).

3.1.3. Error Assessment Criteria

For the multi-site local wind speed prediction with graph network input in the wind farm, we evaluate its effectiveness using several key metrics. These metrics serve as indicators of prediction accuracy and performance. The following evaluation metrics are employed in this simulation: mean squared error (MSE), mean absolute error (MAE), normalized (RMSE), and normalized mean absolute error (MAPE) [33]. The formulas are expressed as follows:

$$E_{MSE} = \frac{1}{n}\sum\nolimits_{i=1}^{n}(y_i - \hat{y}_i)^2, \tag{21}$$

$$E_{RMSE} = \sqrt{\frac{1}{n}\sum\nolimits_{i=1}^{n}(y_i - \hat{y}_i)^2}, \tag{22}$$

$$E_{MAE} = \frac{1}{n}\sum\nolimits_{i=1}^{n}|y_i - \hat{y}_i|, \tag{23}$$

$$E_{MAPE} = \frac{1}{n}\sum\nolimits_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100\%, \tag{24}$$

where n represents the number of samples, y_i denotes the i-th observed value, and $\hat{y}_i$ represents the i-th predicted value. MSE and RMSE quantify the goodness of fit of the model. Smaller values indicate better performance and reduced prediction error. However, it's important to note that MSE and RMSE are sensitive to outliers and can reflect the distribution of prediction errors. The MAE is the mean of the absolute differences between the predicted values and the true values. Unlike MSE, MAE is not influenced by outliers and provides a robust measure of prediction accuracy. MAPE remains unaffected by outliers and provides insight into the relative error across different data points. MSE and RMSE are commonly utilized for evaluating model performance, while MAE and MAPE offer alternative perspectives that take outliers into account, providing a more comprehensive assessment.

Table 1. Site dataset classification.

Category	Variables
Velocity and friction	friction_velocity_2 m
Monin–Obukhov length	inversemoninobukhovlength_2 m
Roughness	roughness_length
Sea temperature	surface_sea_temperature
Pressure	pressure_0 m, pressure_100 m, pressure_200 m
Humidity	relativehumidity_2 m
Precipitation rate	precipitationrate_0 m
Wind speed (at various heights)	windspeed_10 m, windspeed_40 m, windspeed_60 m, windspeed_80 m, windspeed_100 m, windspeed_120 m, windspeed_140 m, windspeed_160 m, windspeed_180 m, windspeed_200 m
Wind direction (at various heights)	winddirection_10 m, winddirection_20 m, winddirection_40 m, winddirection_60 m, winddirection_80 m, winddirection_100 m, winddirection_120 m, winddirection_140 m, winddirection_160 m, winddirection_180 m, winddirection_200 m
Temperature (at various heights)	temperature_2 m, temperature_10 m, temperature_20 m, temperature_40 m, temperature_60 m, temperature_80 m, temperature_100 m, temperature_120 m, temperature_140 m, temperature_160 m, temperature_180 m, temperature_200 m
General weather parameters	wind_speed, wind_direction, pressure, temperature

3.2. *Experimental Results and Analysis*

3.2.1. Analysis of PCA-RF

In our study, we address the challenge of extracting meaningful meteorological features from a dataset containing 46 variables. Previous attempts using individual methods yielded inaccurate results and exhibited bias. Therefore, we employ random forest to identify weather elements that capture essential information from the original variables.

By doing so, we filter out noise and focus on the most relevant features. Subsequently, we employ PCA to downsize the remaining minor weather elements. Finally, we fuse the main features with the dimensionality-reduced data to extract the characteristic weather elements that significantly impact wind changes. Figure 6 shows the MDI of the first 13 features of windspeed_10 m, which is the feature with the largest contribution to the other features of the site numbered 0. This paper observed a significant decline in MDI for all sites after the 10th feature. Consequently, the first 10 features are selected as the main features. Subsequently, the remaining 36 residual features undergo downsizing via PCA, which results in a new downscaled matrix with five principal components. Finally, the main feature matrix and the downscaling matrix are spliced as our feature set to build the graph network.

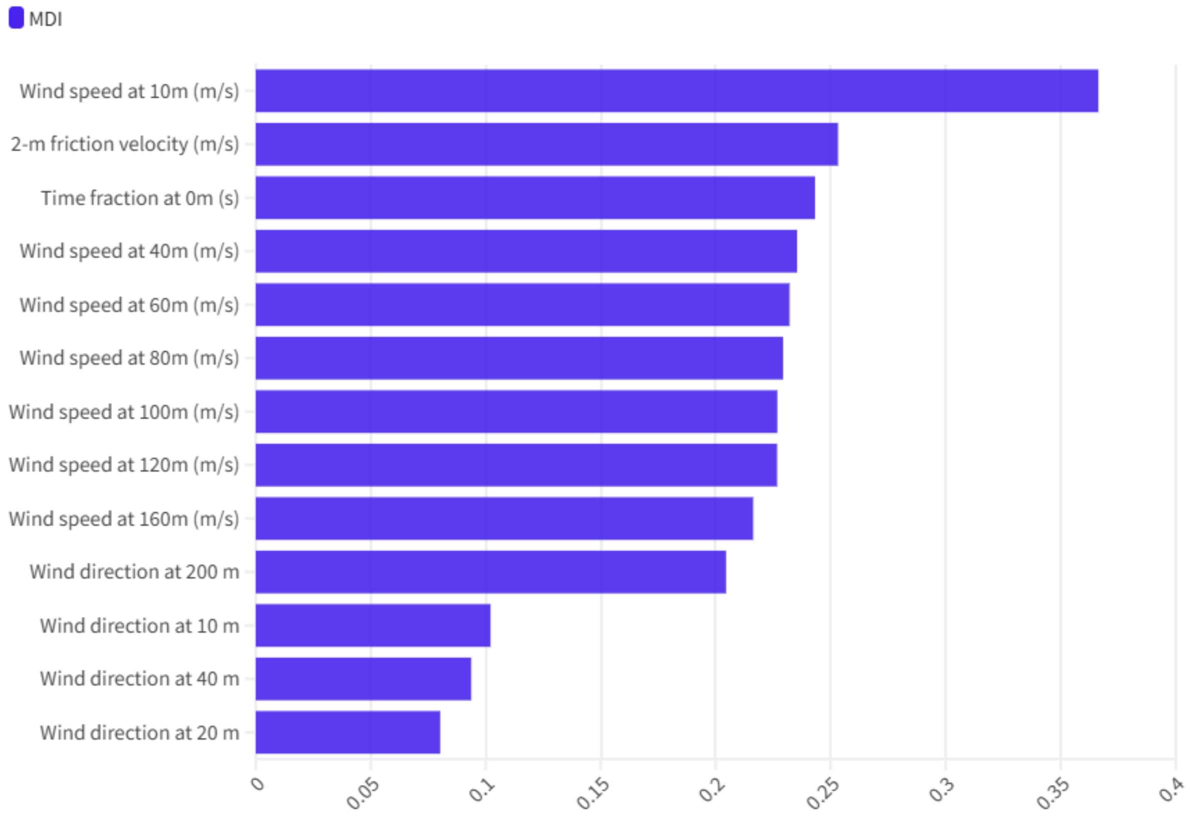


Figure 6. Features of the top thirteen MDI scores.

3.2.2. Evaluation of Graph Embedding for Graph Networks

To verify the effectiveness of graph embedding to build graph networks, we will compare the method of building graph networks solely using the Pearson correlation coefficient, i.e., the PS-GIN-GRU model. Due to the variability in the number of edges resulting from distinct threshold settings, we employ the optimally constructed network graph by both models for training purposes. The constructed graph networks were put into the same GIN-GRU model training and wind speed prediction at 10 m height for 40 days was performed. The model's performance was assessed using four key metrics: MSE, RMSE, MAE, and MAPE. We compared the predicted wind speeds from several stations with the actual measurements to validate the model's effectiveness. The error metrics for wind speed prediction, as obtained from graph networks constructed by different methods, are presented in Table 2.

Our focus was on evaluating the stability of predictions across multiple sites. The GE-GIN-GRU model integrates various neural network algorithms. Notably, this model demonstrates robustness and strong generalization ability. Models constructed solely using Pearson correlation coefficients rely exclusively on the correlation threshold method for building graph networks. While this approach is straightforward, it lacks the adaptability and complexity inherent in the GE-GIN-GRU model. We evaluated both models to assess the stability of wind speed forecasts across various sites. Especially, the GE-GIN-GRU model consistently outperformed the single correlation threshold model in terms of stability. Additionally, the GE-GIN-GRU model demonstrated robust prediction quality across diverse wind speed datasets. Our study underscores the critical role of constructing a graph that captures intricate relationships among wind speed features. By embedding vectors within this graph, we enhance accuracy in wind speed prediction. Particularly, the fusion of graph embedding techniques with neural networks significantly contributes to the model's performance and reliability.

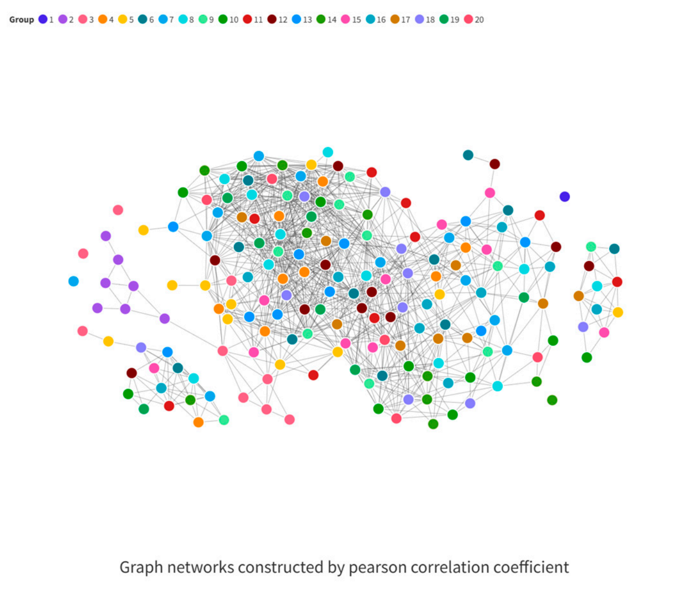
Table 2. Comparison results of different methods of constructing graph networks.

Graph Foundation Models	MSE	RMSE	MAE	MAPE
①PS-GIN-GRU	2.5422	1.5944	1.4224	0.1151
①GE-GIN-GRU	0.8457	0.9196	0.7612	0.0636
②PS-GIN-GRU	2.7184	1.6488	1.4988	0.1231
②GE-GIN-GRU	1.0712	1.0350	0.8645	0.0726
③PS-GIN-GRU	2.7143	1.6475	1.4856	0.1209
③GE-GIN-GRU	0.9258	0.9622	0.8027	0.0685
④PS-GIN-GRU	2.7724	1.6651	1.4912	0.1208
④GE-GIN-GRU	0.9222	0.9603	0.7897	0.0666
⑤PS-GIN-GRU	2.5713	1.6035	1.4319	0.1160
⑤GE-GIN-GRU	0.8507	0.9223	0.7455	0.0624
⑥PS-GIN-GRU	2.1951	1.4816	1.3085	0.1062
⑥GE-GIN-GRU	1.0044	1.0022	0.8317	0.0705
⑦PS-GIN-GRU	2.2201	1.4900	1.3143	0.1066
⑦GE-GIN-GRU	0.9340	0.9665	0.8060	0.0684
⑧PS-GIN-GRU	2.2993	1.5163	1.3373	0.1079
⑧GE-GIN-GRU	1.1580	1.0761	0.9188	0.0761
⑨PS-GIN-GRU	2.2724	1.5074	1.3321	0.1074
⑨GE-GIN-GRU	0.7793	0.8828	0.7233	0.0613
⑩PS-GIN-GRU	2.1376	1.4620	1.2833	0.1032
⑩GE-GIN-GRU	1.0573	1.0283	0.8545	0.0719

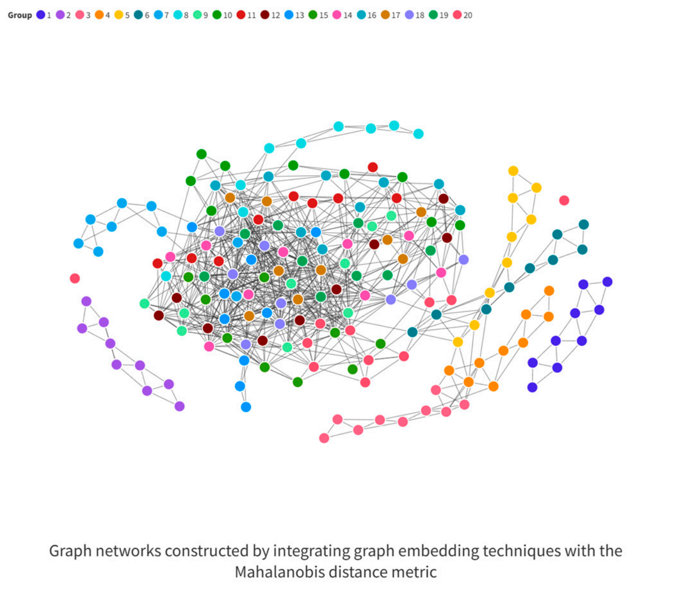
Figure 7 presents the graph networks constructed by various models. With 20 sites in total, the diagram is divided into 20 corresponding groups. Within this diagram, each feature is depicted as a node, and features belonging to the same site are denoted by a uniform color. Figure 7 reveals that the graph networks generated through the Pearson correlation coefficient possess fewer edges compared to the one constructed via graph embeddings. Additionally, the node density within the correlation-based graph networks is comparatively lower, which leads to a more uniform distribution of edge connections. In contrast, the graph networks constructed by graph embeddings demonstrate a concentrated focus on specific nodes to capture their profound connections.

3.2.3. Evaluation and Analysis of GIN-GRU Neural Networks

To further validate the effectiveness of the GIN-GRU neural network, we compared it with recent popular wind speed prediction models: CNN-LSTM proposed by W. Tuerxun [16], GAT-GRU proposed by D. Aykas [34], and GAT-LSTM proposed by A. Flores [35]. For these models, we also constructed a network using the TensorFlow 2.10.0 framework in Python for training and fine-tuning. We evaluated these models using a test set of 20 stations over 40 days for each height wind speed prediction. The input graph network, which is identical across various models, is constructed utilizing consistent graph embedding techniques. We employed four key metrics (MSE, RMSE, MAE, and MAPE) to summarize the results. The results were summarized using four metrics: MSE, RMSE, MAE, and MAPE. The error metrics for the wind speed predictions made by different neural network models are displayed in Tables 3 and 4.


Group 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
Graph networks constructed by pearson correlation coefficient

(**a**) Graph networks constructed by pearson correlation coefficient


Group 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
Graph networks constructed by integrating graph embedding techniques with the Mahalanobis distance metric

(**b**) Graph networks constructed by integrating graph embedding techniques with the Mahalanobis distance metric

Figure 7. Graph networks constructed by different models.

Table 3. Wind speed error metrics predicted by different neural networks at 10 m height.

Error Metrics	CNN-LSTM	GAT-GRU	GAT-LSTM	GE-GIN-GRU
MSE	7.7851	2.1808	2.3968	0.8457
RMSE	2.7902	1.4768	1.5482	0.9196
MAE	2.6851	1.3567	1.4009	0.7612
MAPE	0.2283	0.1152	0.1213	0.0636

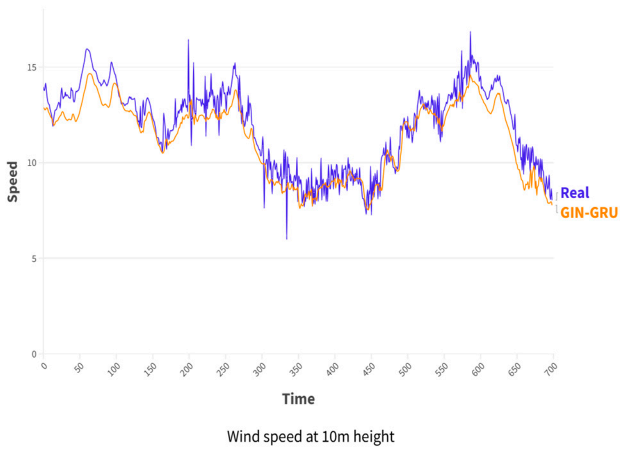
Table 4. Wind speed error metrics predicted by different neural networks at 30 m height.

Error Metrics	CNN-LSTM	GAT-GRU	GAT-LSTM	GE-GIN-GRU
MSE	4.9849	1.3163	1.4214	0.6400
RMSE	2.2327	1.1473	1.1922	0.8000
MAE	2.0489	0.9414	0.9726	0.6076
MAPE	0.1629	0.0759	0.0787	0.0503

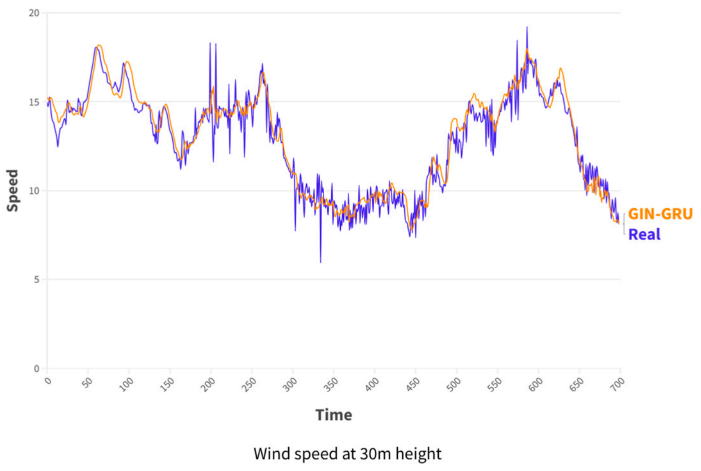
As shown in Figure 8, we compare the wind speed predictions for heights of 10 m and 30 m using the GIN-GRU model.

Despite variations in altitude, the GIN-GRU model consistently predicts wind speed with higher accuracy compared to other models. This result validates the effectiveness of the proposed method.

An examination of wind speed forecasts using various models at the designated site for elevations of 10 m and 30 m uncovers notable trends, as illustrated in Figure 9. While other networks generally align with measured wind speed signals, they often fail to grasp the nuanced graphical structural relationships. In contrast, GIN excels in this regard by leveraging additional spatio-temporal nodes as a predictive foundation.

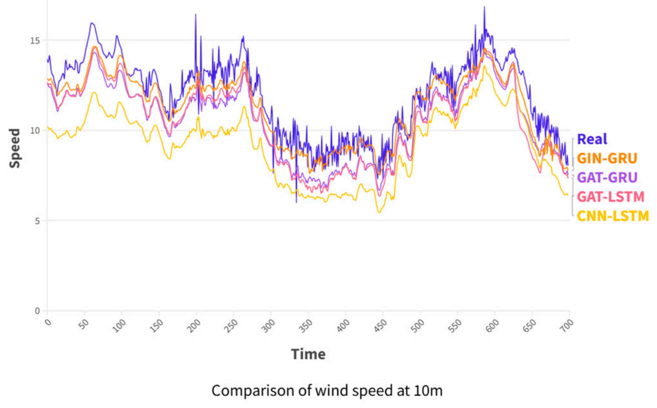


(**a**) wind speed prediction at 10 m height

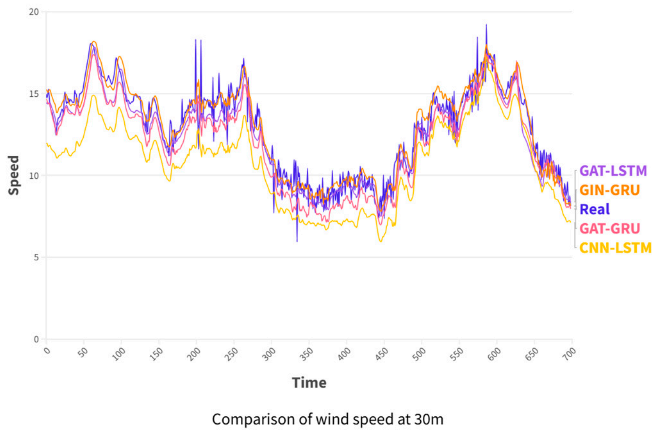


(**b**) wind speed prediction at 30 m height

Figure 8. Result of comparison between GE-GIN-GRU wind speed prediction and real value.



(**a**) wind speed prediction at 10 m height



(**b**) wind speed prediction at 30 m height

Figure 9. GE-GIN-GRU wind speed prediction compared with other models.

Overall, both GE-GIN-GRU and other models effectively track wind speed trends. However, due to the limitations of other models in data processing and their sensitivity to critical information, the prediction error tends to be larger during wind speed fluctuations. GE-GIN-GRU stands out by leveraging information from neighboring stations. This approach better simulates unique wind speed patterns across wind of different heights. Notably, the predicted trend of GE-GIN-GRU closely aligns with the measured wind speed data, particularly during smooth wind speed changes and gradual increases. Additionally, during wind speed fluctuation periods, GE-GIN-GRU leverages its nonlinear fitting ability to approximate the wind speed more accurately than the continuous method.

4. Discussion

To delve deeper into the impact of network graph alterations on wind speed prediction outcomes within graph neural networks, this study will explore two primary dimensions: the variations in nodes within the input graph networks and the modifications in edges within the input graph networks.

4.1. Effect of the Different Nodes on Graph Networks

In this study, we construct the graph nodes using the primary site features obtained through PCA-RF processing. We focus exclusively on the principal features identified after

RF processing as the graph's nodes and deliberately omit the residual feature matrix that is obtained following PCA dimensionality reduction for our comparative analysis. Tables 5 and 6 present the error metrics for wind speed predictions, following the application of different feature selection methods to the nodes.

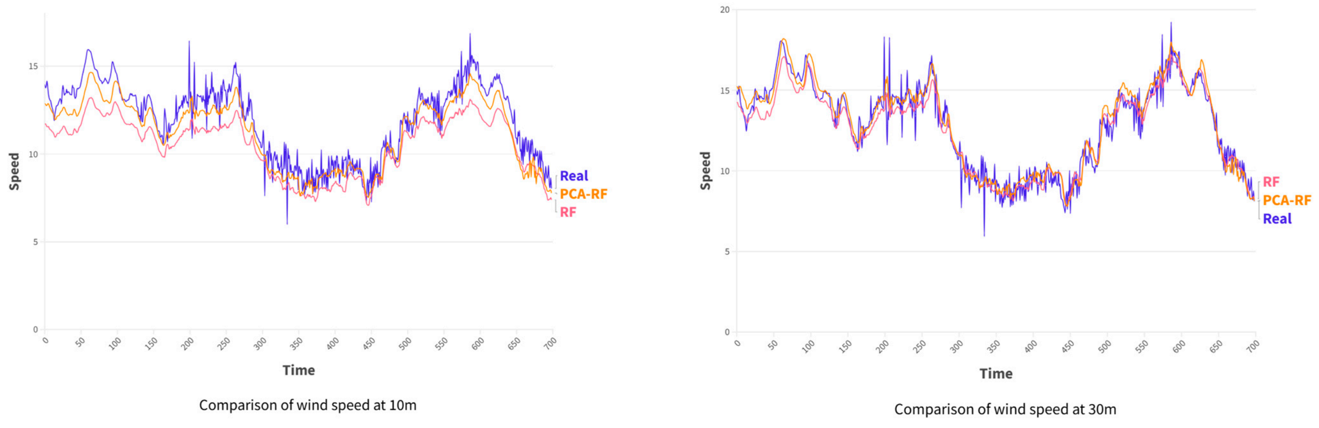
Table 5. Different feature selection methods are utilized to predict the wind speed error metrics at 10 m height.

Error Metrics	RF	PCA-RF
MSE	2.5422	0.8457
RMSE	1.5944	0.9196
MAE	1.4224	0.7612
MAPE	0.1151	0.0636

Table 6. Different feature selection methods are utilized to predict the wind speed error metrics at 30 m height.

Error Metrics	RF	PCA-RF
MSE	0.6945	0.6400
RMSE	0.8334	0.8000
MAE	0.6354	0.6076
MAPE	0.0510	0.0503

The distinction between the RF and PCA-RF models is evident in the diminished quantity of nodes and edges within the graph networks, as presented in Figure 10. To ascertain the influence of varying nodes, we modulate the threshold to align the number of edges as closely as possible. Our empirical observations reveal that both models exhibit parallel accuracy levels in predicting wind speeds at an altitude of 30 m. However, at a 10-m elevation, the performance of the RF model is significantly inferior to that of the PCA-RF model. This discrepancy may stem from the presence of nodes in the residual feature matrix, obtained through the PCA dimensionality reduction, which significantly enhances the precision of wind speed predictions at the 10-m mark. Conversely, the RF model which relies solely on the primary feature matrix yields suboptimal predictions due to an inadequate feature set.



(**a**) wind speed prediction at 10 m height (**b**) wind speed prediction at 30 m height

Figure 10. Comparison between PCA-RF feature screening and RF feature screening.

4.2. *Effect of the Different Edges on Graph Networks*

The construction of additional edges within the input graph networks do not necessarily correlate with the improvement of prediction accuracy. By varying the threshold value, we manipulate the number of edges via graph embedding and the predictive outcomes are expressed as Table 7.

Table 7. The impact of varying the number of network edges on the error metrics of wind speed prediction.

Error Metrics	GE-GIN-GRU (362)	GE-GIN-GRU (1724)	GE-GIN-GRU (3124)
MSE	1.6184	0.8457	1.4014
RMSE	1.2722	0.9196	1.1838
MAE	1.1254	0.7612	1.0025
MAPE	0.0924	0.0636	0.0857

The experimental data suggest that the network achieves peak predictive accuracy when it maintains a balanced number of edges, as illustrated in Figure 11. A paucity of edges compromises accuracy due to the neural network's deficiency in requisite information for effective learning. On the other hand, an overabundance of edges leads to computational inefficiency by introducing unnecessary data, which does not translate to improved accuracy.

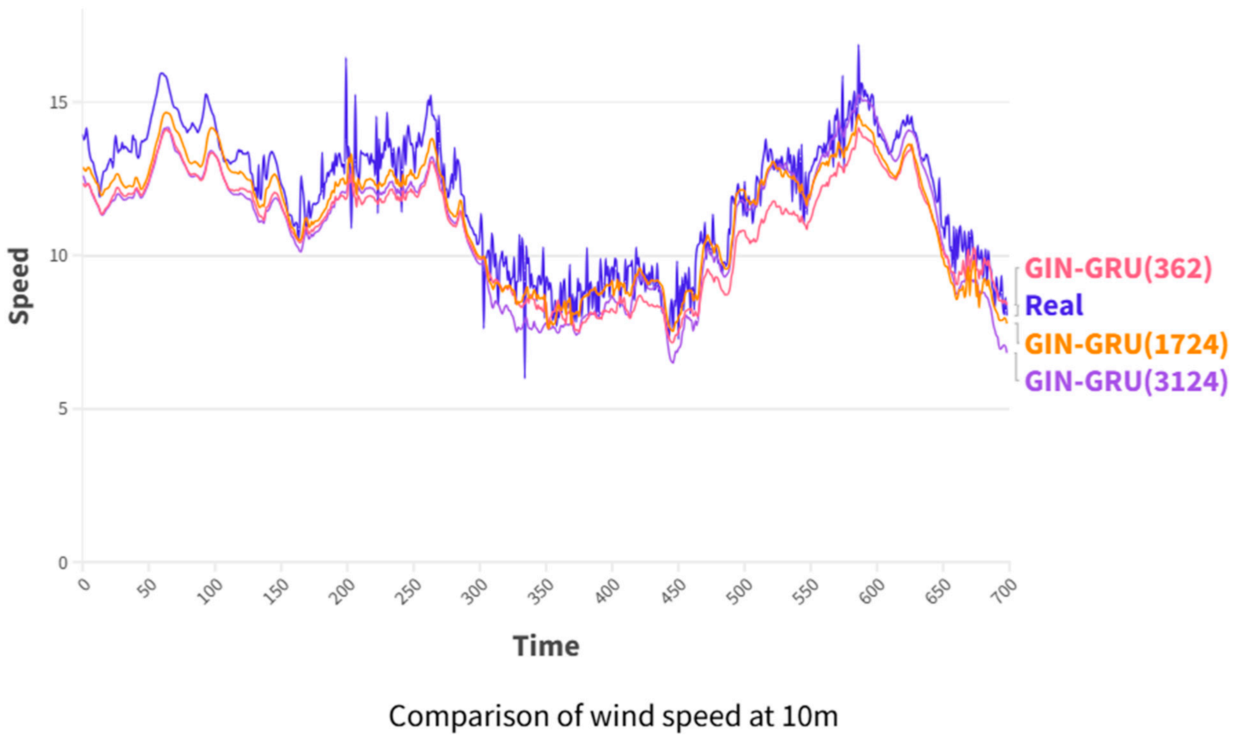


Figure 11. Comparison of varying the number of network edges on the error metrics of wind speed prediction.

Furthermore, the capacity for information processing and filtration varies across graph networks constructed via disparate methodologies. Notably, when a graph network constructed via correlation amasses 1000 edges, there is a significant decline in the accuracy of neural network processing. Comparatively, when the correlation and graph embedding models are used to construct graph networks with an equivalent number of edges, the latter outperforms the former in terms of quality. This superiority is attributed to the graph embedding model's advanced proficiency in feature extraction, understanding of the dataset's topology, and advanced analysis of node relationships, which facilitates the construction of a network graph optimally suited for graph neural network learning. Moreover, the methods by which various neural networks process graphical networks differ significantly. GIN exhibits superior performance over other graph neural networks in managing intricate network edges, which translates to enhanced prediction accuracy.

5. Conclusions

This paper presents the GE-GIN-GRU model, which synthesizes graph embeddings and neural network techniques to streamline wind speed prediction for stations within the study area. The proposed PCA-RF algorithm effectively reduces the number of features involved in the computation and improves the computational efficiency of the model. Subsequently, we employed deep learning-based graph embedding techniques to construct graph networks that capture the interrelationships among the sites. Our graph embedding methods capitalize on the strengths of both GraphSAGE and the Mahalanobis distance. The former excels at extracting intricate connections within wind speed features, forming feature vectors. Meanwhile, the latter demonstrates advantages in processing high-dimensional feature vectors. By fully leveraging the strengths of these two methods, we construct optimized graph networks. The GIN-GRU model seamlessly integrates diverse neural network algorithms to enhance generalization capabilities and improve prediction accuracy. Consequently, it consistently maintains excellent prediction quality and stability across wind speed datasets at varying heights. By fully leveraging the strengths of both models, we achieve deep extraction of spatio-temporal relationship features. In subsequent research, our primary goal is to construct new graph networks by combining the location coordinates of the sites with their corresponding nodes.

Author Contributions: Conceptualization, H.W. and H.C.; methodology, H.W.; resources, H.W.; writing—original draft preparation, H.W.; writing—review and editing, H.W. and H.C.; funding acquisition, H.W. and H.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The wind field data were generated using the Wind Toolkit API (version 2) provided by the National Renewable Energy Laboratory (NREL). Available at https://developer.nrel.gov/docs/wind/wind-toolkit/ (accessed on 15 January 2024).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Valdivia-Bautista, S.M.; Domínguez-Navarro, J.A.; Pérez-Cisneros, M.; Vega-Gómez, C.J.; Castillo-Téllez, B. Artificial Intelligence in Wind Speed Forecasting: A Review. *Energies* **2023**, *16*, 2457. [CrossRef]
2. Chandra, D.R.; Kumari, M.S.; Sydulu, M. A detailed literature review on wind forecasting. In Proceedings of the 2013 International Conference on Power, Energy and Control (ICPEC), Dindigul, India, 6–8 February 2013; pp. 630–634. [CrossRef]
3. Zhu, C.; Zhu, L. Wind Speed Short-Term Prediction Based on Empirical Wavelet Transform, Recurrent Neural Network and Error Correction. In *Journal of Shanghai Jiaotong University (Science)*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 1–12.
4. Bokde, N.; Feijóo, A.; Villanueva, D.; Kulat, K. A review on hybrid empirical mode decomposition models for wind speed and wind power prediction. *Energies* **2019**, *12*, 254. [CrossRef]
5. Zhang, Y.; Zhang, C.; SUN, J.B.; Guo, J.J. Improved Wind Speed Prediction Using Empirical Mode Decomposition. *Adv. Electr. Comput. Eng.* **2018**, *18*, 3–10. [CrossRef]
6. Ren, Y.; Suganthan, P.N. Empirical mode decomposition-k nearest neighbor models for wind speed forecasting. *J. Power Energy Eng.* **2014**, *2*, 176–185. [CrossRef]
7. Wang, L.; Liao, Y. A short-term hybrid wind speed prediction model based on decomposition and improved optimization algorithm. *Front. Energy Res.* **2023**, *11*, 1298088. [CrossRef]
8. Qu, Z.; Hou, X.; Hu, W.; Yang, R.; Ju, C. Wind power forecasting based on improved variational mode decomposition and permutation entropy. *Clean Energy* **2023**, *7*, 1032–1045. [CrossRef]
9. Mohapatra, M.R.; Radhakrishnan, R.; Shukla, R.M. A Hybrid Approach using ARIMA, Kalman Filter and LSTM for Accurate Wind Speed Forecasting. *arXiv* **2023**, arXiv:2311.08550.
10. Che, Y.; Xiao, F. An integrated wind-forecast system based on the weather research and forecasting model, Kalman filter, and data assimilation with nacelle-wind observation. *J. Renew. Sustain. Energy* **2016**, *8*, 053308. [CrossRef]
11. Xu, H.; Chang, Y.; Zhao, Y.; Wang, F. A novel hybrid wind speed interval prediction model based on mode decomposition and gated recursive neural network. *Environ. Sci. Pollut. Res.* **2022**, *29*, 87097–87113. [CrossRef]

12. Ai, X.; Li, S.; Xu, H. Wind speed prediction model using ensemble empirical mode decomposition, least squares support vector machine and long short-term memory. *Front. Energy Res.* **2023**, *10*, 1043867. [CrossRef]
13. Shao, B.; Song, D.; Bian, G.; Zhao, Y. Wind speed forecast based on the LSTM neural network optimized by the firework algorithm. *Adv. Mater. Sci. Eng.* **2021**, *2021*, 4874757. [CrossRef]
14. Zhu, Q.; Chen, J.; Zhu, L.; Duan, X.; Liu, Y. Wind speed prediction with spatio–temporal correlation: A deep learning approach. *Energies* **2018**, *11*, 705. [CrossRef]
15. Trebing, K.; Mehrkanoon, S. Wind speed prediction using multidimensional convolutional neural networks. In Proceedings of the 2020 IEEE symposium series on computational intelligence (SSCI), Canberra, ACT, Australia, 1–4 December 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 713–720.
16. Tuerxun, W.; Xu, C.; Guo, H.; Guo, L.; Zeng, N.; Cheng, Z. An ultra-short-term wind speed prediction model using LSTM based on modified tuna swarm optimization and successive variational mode decomposition. *Energy Sci. Eng.* **2022**, *10*, 3001–3022. [CrossRef]
17. Yuan, M.M.; Gong, F.M.; Li, X. Multifactor Spatio-Temporal Wind Speed Prediction Based on CNN-LSTM. *Comput. Syst. Appl.* **2021**, *30*, 133–141. (In Chinese)
18. Louppe, G. Understanding random forests: From theory to practice. *arXiv* **2014**, arXiv:1407.7502.
19. Li, X.; Wang, Y.; Basu, S.; Kumbier, K.; Yu, B. A debiased MDI feature importance measure for random forests. *Adv. Neural Inf. Process. Syst.* **2019**, *32*, 713–720.
20. Abdi, H.; Williams, L.J. Principal component analysis. *Wiley Interdiscip. Rev. Comput. Stat.* **2010**, *2*, 433–459. [CrossRef]
21. Zhou, J.; Cui, G.; Hu, S.; Zhang, Z.; Yang, C.; Liu, Z.; Sun, M. Graph neural networks: A review of methods and applications. *AI open* **2020**, *1*, 57–81. [CrossRef]
22. Waikhom, L.; Patgiri, R. Graph neural networks: Methods, applications, and opportunities. *arXiv* **2021**, arXiv:2108.10733.
23. Cai, H.; Zheng, V.W.; Chang, K.C.C. A comprehensive survey of graph embedding: Problems, techniques, and applications. *IEEE Trans. Knowl. Data Eng.* **2018**, *30*, 1616–1637. [CrossRef]
24. De Maesschalck, R.; Jouan-Rimbaud, D.; Massart, D.L. The mahalanobis distance. *Chemom. Intell. Lab. Syst.* **2000**, *50*, 1–18. [CrossRef]
25. Leys, C.; Klein, O.; Dominicy, Y.; Ley, C. Detecting multivariate outliers: Use a robust variant of the Mahalanobis distance. *J. Exp. Soc. Psychol.* **2018**, *74*, 150–156. [CrossRef]
26. Goyal, P.; Ferrara, E. Graph embedding techniques, applications, and performance: A survey. *Knowl. Based Syst.* **2018**, *151*, 78–94. [CrossRef]
27. Ju, W.; Fang, Z.; Gu, Y.; Liu, Z.; Long, Q.; Qiao, Z.; Zhang, M. A comprehensive survey on deep graph representation learning. *Neural Netw.* **2024**, *173*, 106207. [CrossRef] [PubMed]
28. Jiang, W. Graph-based deep learning for communication networks: A survey. *Comput. Commun.* **2022**, *185*, 40–54. [CrossRef]
29. Kim, B.H.; Ye, J.C. Understanding graph isomorphism network for rs-fMRI functional connectivity analysis. *Front. Neurosci.* **2020**, *14*, 545464. [CrossRef]
30. Chen, Z.; Villar, S.; Chen, L.; Bruna, J. On the equivalence between graph isomorphism testing and function approximation with gnns. *Adv. Neural Inf. Process. Syst.* **2019**, *32*.
31. Chung, J.; Gulcehre, C.; Cho, K.; Bengio, Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv* **2014**, arXiv:1412.3555.
32. Dey, R.; Salem, F.M. Gate-variants of gated recurrent unit (GRU) neural networks. In Proceedings of the 2017 IEEE 60th international midwest symposium on circuits and systems (MWSCAS), Boston, MA, USA, 6–9 August 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 1597–1600.
33. Chicco, D.; Warrens, M.J.; Jurman, G. The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation. *Peerj Comput. Sci.* **2021**, *7*, e623. [CrossRef]
34. Aykas, D.; Mehrkanoon, S. Multistream graph attention networks for wind speed forecasting. In Proceedings of the 2021 IEEE Symposium Series on Computational Intelligence (SSCI), Orlando, FL, USA, 5–7 December 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 1–8.
35. Flores, A.; Tito-Chura, H.; Yana-Mamani, V. An ensemble GRU approach for wind speed forecasting with data augmentation. *Int. J. Adv. Comput. Sci. Appl.* **2021**, *12*, 569–574. [CrossRef]