STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **x**, Month 202x, pp 0–21.
Published online in International Academic Press (www.IAPress.org)

Missing Values Imputation for Spatio-Temporal Climate Variable using Graph Convolutional Networks (GCN)

Baraa Wisam Abdulghani, Osamah Basheer Shukur*

Department of Statistics and Informatics, College of Computer science and Mathematics, University of Mosul, Mosul, Iraq

Abstract Climatic time series analysis often suffers from an issue of missing values. Which directly affects the accuracy of analysis and modeling especially with multi-site data which include spatial and temporal dependencies. Although traditional imputation methods do exist, most of them depend mainly on temporal information which limits its efficiency especially with consecutive missing values or when the temporal data are weak. This limitation becomes more critical under the Block Missing Pattern, where consecutive missing values reduce the availability of temporal information and increase the difficulty of the imputation process.This study applies a spatially structured Graph Convolutional Network (GCN) model for missing-value imputation in multivariate climatic time series collected from multiple meteorological stations.covering the period from 1994 to 2024 with a total of 11,329 daily observations. To reduce seasonal heterogeneity, the data were first stratified into hot and cold seasons. Each seasonal dataset was then divided into training (80%), validation (10%), and testing (10%) subsets for model development and performance evaluation. Overall, this study confirms that the use of models that depend on spatial structures is an effective method for handling missing value in time series, and paves the way for developing more advanced models that combines between the spatial and temporal dimensions. The model's performance has been evaluated under the Block Missing Pattern in both of the hot and cold seasons using RMSE, MAE and MAPE metrics, as well as comparing the performance with K-Nearest Neighbors (KNN), Linear Interpolation (LI), and GRU-D. The results show that the Graph Convolutional Network provides a more efficient representation of the climatic data, especially in cases with weak or unavailable temporal data, where a higher accuracy can be achieved by using spatial relationships between stations.The proposed model achieved the lowest error values across most climatic variables in both seasons The results also indicate that the nature of the data plays an essential role in the performance, where the decrease of variability and the stability of values in the hot season contributed to the improvement of the missing data imputation.

Keywords Graph Convolutional Network (GCN), K-Nearest Neighbors (KNN), Climatic Time Series, Spatio-Temporal Forecasting, Missing Values Imputation, Multivariate Time Series

DOI: 10.19139/soic-2310-5070-3928

1. Introduction

Climate time series analysis plays an important role in understanding environmental changes. It is used in a number of fields such as agriculture and water resources management. However, this data usually suffers from missing values as a result of malfunctions with the measuring devices or varying operational conditions. The missing data is considered one of the most persistent challenges in statistical analysis and experimental researches. During the last two decades, systematic and computational improvements led to a notable shift in how missing data is handled, allowing for stronger and more flexible analytical frameworks. The daily climatic data for Nineveh government have been collected from five meteorological stations each representing a different location, which are Mosul, Tal Afar, Ba'aj, Tal Abta and Makhmur. This data includes a group of important climate factors which are solar radiation, maximum and minimum temperatures, relative humidity and wind speed. The time

*Correspondence to: Osamah Basheer Shukur (Email: drosamahannon@uomosul.edu.iq). Department of Statistics and Informations, College of Computer science and Mathematics, University of Mosul, Mosul, Iraq.

ISSN 2310-5070 (online) ISSN 2311-004X (print)

period of the data extends from the 1st January 1994 to the 31st of December 2024, this represents 11,329 daily observations, which provides a long time series that facilitates the analysis of climatic patterns and studying the relationship among these variables across time and space. This data is characterized by a clear temporal correlation as well as spatial relationships among different locations. Many studies have dealt with the issue of missing data from an analytical and systematic approach. Rubin [1] presented a comprehensive framework for the types of missing data, the mechanisms, and patterns of their missingness; this leads to the understanding of the missing data's nature and choosing suitable methods of handling them, where we can classify these patterns to four main categories: Univariate, Multivariate, Monotone, and Arbitrary. According to this classification, the ways of handling missing data can be divided to a number of main categories. One of which are Deletion-Based Methods such as Listwise Deletion or Available-Case Analysis, where records containing missing values have been excluded from the analysis process, in addition to simple imputation methods such as mean imputation or regression models imputation, which depend on replacing missing values with estimates derived from the available data Kang [2].In addition, others studies have presented advanced methods for missing values imputation using machine learning. Stekhoven and Bühlmann [3] introduced the MissForest algorithm which depends on random forests and have shown high efficiency in dealing with complicated and non-linear data. Other models based on neural networks have been developed such as the Gated Recurrent Unit with Decay(GRU-D) which Che, Purushotham [4] developed relying on the representation of the missing data patterns using masking and time interval and combining them through the Decay mechanism. The results show its superiority over other traditional methods. Cao, Wang [5] have also suggested the BRITS model which depends on recurrent neural networks. This model allows for the direct imputation of missing values in temporal series during the learning process within a dynamic framework without any previous assumptions about the nature of the data, showing high efficiency in dealing with temporal non-linear data. Despite these developments, many applied studies still depend on simple imputation strategies.Furthermore, most of these studies focused primarily on temporal relationships within the time series, while not making sufficient use of spatial relationships between different observation sites. This highlights the constant need to more advanced methodologies in dealing with missing data. In this context, effective spatial modeling allows for missing value imputation methods to reconstruct temporal data by using the information available at different locations. However, many traditional methods are unable to capture complicated non-linear dependencies that appear across time and space inside the connected networks of sensors, hence it cannot fully benefit from the relational informational available in spatial data. As a result of this, graph based deep learning models have appeared, such as Graph Neural Networks (GNNs) which is capable of representing spatial relationships among different sites, making it suitable to process missing data in spatial time series and model the relationships between the different sites. Similarly, in recent years, graph-based models have been employed in analyzing spatiotemporal data, as the Graph Convolutional Network model introduced by Kipf and Welling [6] is considered as one of the fundamental models in the field for its ability to represent spatial relationships between nodes. Li, Yu [7] also introduced the DCERNN model which combines spatial and temporal relationships for traffic time series prediction. Yu, Yin [8] also suggested the STGCN model, which combines spatial and temporal convolution within a unified framework. This emphasizes the ability to employ such models in the imputation of missing temporal values. Taguchi, Liu [9] also addressed the problem of missing values in graphs. They proposed integrating missing-value handling with a Graph Convolutional Network (GCN) using a Gaussian Mixture Model (GMM) instead of pre-imputation in order to reduce the impact of imputation errors on model performance. However, this approach requires incorporating an additional probabilistic model into the learning process, making it more complex from a practical perspective. [10] used Median Imputation to fill the gaps in climatic data for air temperature and wind speed in the Rosario Islands in the Colombian Caribbean. The outcomes showed that the method was effective in filling missing data when the data were missing in a sporadic manner. However, it was less efficient when there were long runs of missing data. [11] introduced a model called GCN-ST-MDIR, which is based on a neural network and uses Graph Convolutional Networks (GCNs) to identify patterns of missing values in daily data and choose the most suitable imputation method. The model also includes Transfer Learning (TL) to pre-train the models used in imputation and combines the results of different models. The results indicated that the proposed model significantly improved the performance of missing-value imputation, with an accuracy of 88.48%. [12] compared different methods for imputing missing data in rainfall, temperature, and

relative humidity time series from Mosul station in Iraq between 1980 and 2013. The results showed that Seasonal Decomposition (SD) performed best for univariate data, while the KNN method was among the most competitive methods for multivariate data, especially for rainfall time series. [13] addressed the challenges associated with missing values in spatiotemporal data and presented a self-supervised spatiotemporal representation learning model to learn spatial and temporal patterns implicitly. The results showed that the proposed model was better than the traditional methods. [14] compared a variety of approaches to imputing missing data. The comparison included commonly used statistical approaches such as Median Imputation, Last Observation Carried Forward (LOCF), and Linear Interpolation, in addition to different deep-learning models, including Transformers. The results indicated that Transformer models performed best overall, while Linear Interpolation was also a good baseline method. The results were also highly affected by the missingness scenario used. The Missing Completely at Random (MCAR) pattern resulted in overly optimistic error estimates and reduced the differences among the different methods, while the Structured Gaps pattern showed clearer differences in model performance. Although there are several studies on the problem of missing-value imputation in climatic time series, most of these studies have used classical methods or machine-learning models that primarily focus on the temporal dimension of climatic data, such as Median Imputation, KNN, MissForest, or Recurrent Neural Network (RNN) models. Furthermore, climatic data also have spatial relationships between monitoring stations, in addition to temporal relationships within individual time series. However, in many imputation models, these spatial relationships have not been used sufficiently. While some recent studies used graph-based models to process spatiotemporal data, many of these studies used complex models that need significant computational resources or extra training procedures, which may limit their use in some practical settings. In addition, climatic time series are heterogeneous because of seasonal differences and the related changes in their statistical properties, which may affect the efficiency of missing-value imputation models. However, previous studies have not given enough attention to this issue by dividing the data into more homogeneous groups before the modelling process, especially in applications involving missing-value imputation in climatic time series. Therefore, the current study aims to fill this gap by constructing a Graph Convolutional Network (GCN) model to impute missing values in the climatic time series of monitoring stations in Nineveh Governorate. The proposed model is based on the spatial relationships between stations and the time lags of climatic variables. The study also uses a Temporal Stratification (TS) method to divide the data into hot and cold seasons in order to reduce heterogeneity and improve the model's ability to learn climatic patterns. Its performance is then evaluated and compared with some commonly used imputation methods.

2. Methods and Materials

2.1. K-Nearest Neighbor (KNN)

This is one of the most commonly used methods for imputing missing values, where each missing value is replaced by the average of the k nearest known neighbors, as given by the following equation [9] .

$$y = \frac{1}{k}\sum_{i=1}^{k} y_i \tag{1}$$

Where y_i represents the value of $i-th$neighbor,k denotes the number of neighbors used for imputation.

2.2. Gated Recurrent Unit with Decay - GRU-D

The Gated Recurrent Unit with Decay (GRU-D) is a deep learning model developed for processing time series with missing values. Proposed by [4], it aims to use information about missing values and the time elapsed since the last observation, rather than ignoring them or imputing them using traditional methods. The model is based on the Gated Recurrent Unit (GRU) architecture with the addition of a decay mechanism that reduces the impact of older information as the time since the last observation of a value increases. The model relies on a mask vector that determines whether a value is observed or missing. The model also uses the time interval δ_t^d , which represents the

time elapsed since the last observation of the variable. This variable is used to measure the effect of loss over time. To represent the effect of time on previous information, the decay factor is calculated using the following equation:

$$\gamma_t = exp[-max(0, W_\gamma \delta_t + b_\gamma)] \tag{2}$$

Where W_γ and b_γ represents the coefficients learned during training, and γ_t represents the decay coefficient, which decreases as the time elapsed since the last observation increases. The missing values are then imputed based on the last observed value and the historical mean of the variable, according to the equation:

$$\hat{x}_t^d = m_t^d x_t^d + (1 - m_t^d)(\gamma_{x_t}^d x_{t'}^d + (1 - \gamma_{x_t}^d)\tilde{x}^d) \tag{3}$$

Where $x_{t'}^d$ represents the most recent observed value of the variable d, and $\tilde{x}^d$ represents the historical average of that variable.
To reduce the impact of old information stored in the network's internal memory, a fading mechanism is applied to the previously hidden state as follows:

$$\hat{h}_{t-1} = \gamma_{h_t} \odot h_{t-1} \tag{4}$$

Where $\odot$ indicates the elemental multiplication operation between vectors. After processing the missing values and the hidden state, the new hidden state is updated using the GRU mechanism according to the equation:

$$h_t = (1 - z_t) \odot \hat{h}_{t-1} + z_t \odot \tilde{h}_t \tag{5}$$

Where z_t represents the Update Gate, and $\tilde{h}_t$ represents the hidden candidate state. This mechanism allows the model to utilize historical time information while considering the loss pattern and the time elapsed since the last observation, making it suitable for processing and completing time series with missing values.

2.3. Linear interpolation (LI)

estimates missing values by drawing a straight line between the two known endpoints of a gap. The missing observations are then calculated directly using the equation of that line[16].

$$y = y_1 + \frac{t - t_1}{t_2 - t_1}(y_2 - y_1) \tag{6}$$

where y_1 is the last known observation before the missing value and y_2 is the first known observation after the missing values on times t_1 and t_2 , respectively, and then y will be the missing value on time t where $t_1 < t < t_2$.[17] In the case of several consecutive missing values, they are estimated by dividing the distance between the two known values into equal intervals and calculating the values falling on them.

2.4. Graph Convolutional Network (GCN)

The Graph Convolutional Network is considered a type of neural networks specially designed to handle data represented in the form of graphs. It extends the concept of convolution used in traditional convolutional networks to operate directly on web structured data, allowing the model to learn the nodes representations by combining the information of neighboring nodes and their relationships. The data entered in the model is represented using two main elements:
1. Feature Matrix (X)

$$X \in \mathbb{R}^{N \times C} \tag{7}$$

Where N represents the number of nodes in the graph
C represents the number of features for each node
$\mathbb{R}$represents the real numbers set

If only the time's lagged series which represent the variables of the AR model, are used to define the input structure of the Artificial Neural Network (ANN) in general, or the GCN as in this study, without considering the model parameters or their signs, this approach can be referred to as a hybrid AR-GCN model[18, 22], or

simply as an GCN model [14, 28].Using an AR model instead of an ARIMA model, regardless of certain time series conditions, leads to a simpler input structure for the GCN according to [18, 4, 5]. In the first stage, an AR model is constructed. Specifically, an AR(p) model is employed to determine the input structure of the GCN, where the number of inputs depends on the autoregressive order. The autocorrelation function (ACF) and partial autocorrelation function (PACF) are applied directly to both the original time series and its first-differenced version to estimate the appropriate autoregressive order.

2. Adjacency Matrix (A)

$$X \in \mathbb{R}^{N \times N} \tag{8}$$

It is a matrix that represents the relationships between nodes in the graph, where:

$$a_{ij} = \begin{cases} 1\ , \ if\ there\ is\ an\ edge\ between\ node\ i\ and node\ j \\ 0, \ \ otherwise \end{cases} \tag{9}$$

Where a_{ij} a represents the element of adjacency matrix Athat indicates whether or not there is a relationship between node $i$and node$j$.

$$A = \begin{vmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & a_{2,3} & \cdots & a_{2,n} \\ a_{3,1} & a_{3,2} & a_{3,3} & \cdots & a_{3,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & a_{n,3} & \cdots & a_{n,n} \end{vmatrix} \tag{10}$$

After the adjacency matrix is constructed, it is first modified by adding the values of the identity matrix to the main diagonal elements. This is equal to adding self-loops, which makes every node connected to itself. This aims at enabling each node to pass its own information to itself during the information collection process with neighboring nodes. As is shown in equation (6):

$$\hat{A} = A + I_N \tag{11}$$

Where$\hat{A}$represents the adjacency matrix after adding the self loops.

After that, the degree matrix is defined as a diagonal matrix, where each element represents the number of neighboring nodes for each node, including the node itself after adding the self-loop.

$$\hat{D} = \begin{vmatrix} d_{1,1} & 0 & 0 & \cdots & 0 \\ 0 & d_{2,2} & 0 & \cdots & 0 \\ 0 & 0 & d_{3,3} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & d_{n,n} \end{vmatrix} \tag{12}$$

Where $\hat{D}$ represents the Degree Matrix of $\hat{A}$.

The renormalization process comes next after multiplying the adjacency matrix on both the right and left by the inverse square root of the degree matrix. Thereby achieving a form of balance in representing the relationships among nodes in order to ensure that the high degree nodes with a large number of neighbors do not dominate the information aggregation process [1].

$$\tilde{A} = \hat{D}^{-\frac{1}{2}} \hat{A} \hat{D}^{-\frac{1}{2}} \tag{13}$$

Where $\tilde{A}$ represents the normalized adjacency matrix.

Figure (1) demonstrates the workflow diagram for a Graph Convolutional Network model starting with the data input stage and its processing up until obtaining the predictions.

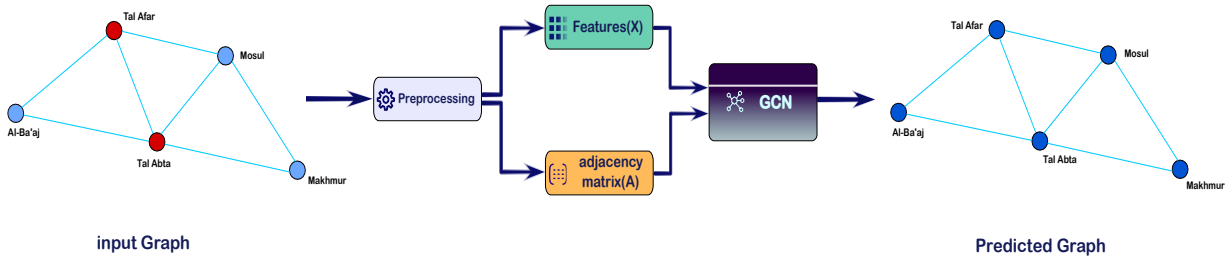

Figure 1. workflow diagram for a Graph Convolutional Network model for missing values imputation.

As is illustrated in figure (1) the data are represented in the form of a graph that contains missing values, where each node represents a meteorological station, while the edges represent the relationships between the nodes, and the missing values are presented in the graph in red.
Relying on this representation, the Graph Convolutional Network process depends on combining the node information with the information from neighboring nodes, where the node representations are updated by applying a grouping process followed by an updating process by using the weight matrix as is shown in the next equation:

$$Z_{l+1} = \sigma\left(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}Z_l W_l\right) \tag{14}$$

σ Represents the activation function (ReLU, Softmax. . . .)

Z_l The node representation function in layer l.

W_l The weight matrix for layer l.

$\hat{A} = A + I_N$ The adjacency matrix after the addition of self-loops.

$\hat{D}$ The degree matrix (a diagonal matrix with each element inside representing the sum of elements in that row)

As is demonstrated in Figure (2) information is collected from neighboring nodes of each node in each layer of the Graph Convolutional Network layers, then the representation of each node is updated by multiplying it with a weight matrix and applying a non-linear activation function, allowing for the model to extract more complex representations. This process is repeated across multiple layers, allowing for the transferring of information between direct neighbors and neighboring neighbors[25]. Thus, enhancing the quality of node representation. Lastly, the model produces the estimated values, where the missing values are imputed according to the spatial relationships among nodes.

The GCN network model relies on the inclusion of residual connections between hidden layers [15]. The residual connections enable the model to carry over information from the previous layer's input [7].Therefore, the output of layer $l + 1$ of the GCN model here is:

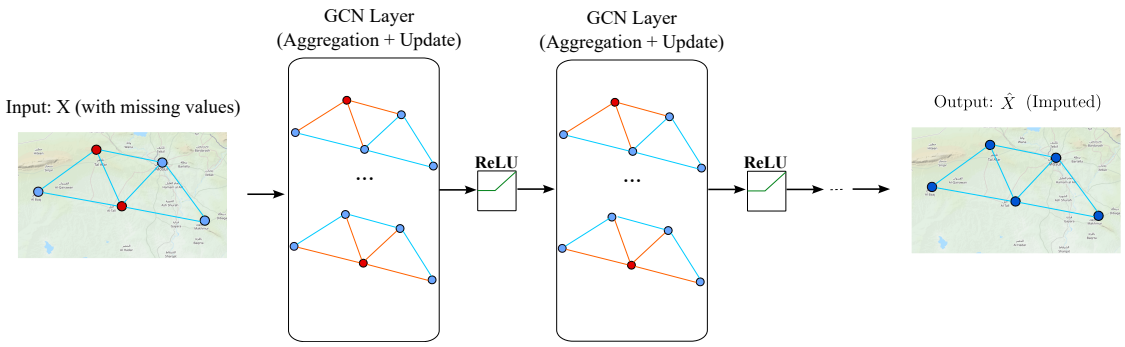

Figure 2. the overall structure for the proposed Graph Convolutional Network for missing value imputation.

$$Z_{l+1} = \sigma\left(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}Z_l W_l\right) + Z_l \tag{15}$$

and the output of layer $l+2$

$$Z_{l+2} = \sigma\left(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}Z_{l+1} W_{l+1}\right) + Z_{l+1} \tag{16}$$

In the output layer, each node computes a weighted sum of its inputs, and since the task of compensating for missing values is formulated as a regression problem, a nonlinear activation function is not used in this layer [17]. The output can be formulated as follows:

$$Z_{l+3} = \hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}Z_{l+2} W_{l+2} \tag{17}$$

The output is represented by a single numerical value per node at each time step, because it represents the predicted value of the missing value at position s at time t. where T represents the number of time steps, S represents the number of spatial locations.

2.4.1. Loss Function .

Choosing the loss function is considered an essential element in missing values imputation models, and predicting the wide range time series considering their direct role in guiding the learning process within the model. As it measures the difference between the true and predicted values in order to optimize the model parameters. It should be mentioned that the loss functions differ according to the task and goal of the model, where there are specialized functions for regression and others for classification and other tasks as well [28, 29]. A number of functions exist in the regression tasks framework, the most famous of which are:
Mean Squared Error (MSE), Mean Absolute Error (MAE), Huber loss, Log-Cosh, Quantile loss, Poisson loss [25]. These functions differ in their features and their sensitivity towards error, making choosing the appropriate function dependent on the nature of the data and the purpose behind the model[31].

2.4.2. Activation function .

The activation functions are a main element in artificial neural networks, as they transform the input signals to output in the next layer. First, the sum of the inputs and their weights is calculated, then the activation function

is applied to the result to obtain the layer outputs, this represents the inputs of the next layer. If the activation function is not used in the network, the output signal will be a simple linear function which works as a simple linear regression model, leading probably to limited performance and weak ability. Thus, the existence of these functions in the network enables them to learn more complex representations such as modeling complex data like pictures, videos, and others. There are various types of activation function, some of which are[21]:
Linear, Sigmoid, Tanh, ReLU, Exponential Linear Unit, SoftMax and others.
These functions differ in their mathematical properties and mechanisms, making selecting an appropriate function dependent on the nature of the model and the type of task. In the Graph Convolutional Networks framework, the Rectified Linear Unit (ReLU) is the most used function in hidden layers, while the SoftMax function is usually used in the output layers of classification tasks. Where the ReLU function is given in the following format:

$$ReLU(x) = max(0, x) \tag{18}$$

Figure (3) demonstrates how a function returns zero when the inputs are negatives, while the positive values pass through unchanged.

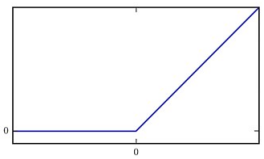

Figure 3. The rectified linear (ReLU) activation function [6].

The SoftMax function is given in the following format:

$$softmax(z)_i = \frac{exp(z_i)}{\sum_{j=1}^{k} exp(z_j)} \tag{19}$$

e^{z_i} serves as the exponential function for the element z_i from the element vector Z after which the exponential function for each element is divided by the sum of the exponential functions for all elements in the vector Z to obtain the possibility of each element, keeping in mind that the sum of the vector's elements probabilities are equal to 1 [6].

2.5. Time Stratification (TS)

The methodology of Time Stratification (TS) is one of the most important effective methods in analyzing time series, as it assists in arranging the data according to seasonal changes, which plays an essential role in determining the characteristics of the time series[34, 35].
This methodology aims at clarifying the effect of seasons or periodic patterns on the data by dividing it to specific seasonal periods. Furthermore, dividing the data to more homogeneous categories assists in reducing random fluctuations, which leads to enhancing the accuracy of statistical and predictive models. Thus, providing more reliable results compared to unclassified data [19, 26].

2.6. Error Measurement

1. Mean Squared Error (MSE) is one of the error measures, defined as the mean squared difference between the actual values and the predicted values. It is one of the most commonly used loss functions in regression problems due to its mathematical properties and stability in the training process [10]. The MSE is computed by the equation:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{20}$$

Where, N is the number of data samples, y_i is the actual value, and $\hat{y}_i$is the predicted value.

2. Root Mean Squared Error (RMSE) is a One of the standard measures used to evaluate model error when dealing with quantitative data defined as square root of the mean squared difference between predicted and actual values[11] . The RMSE is computed as follows:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{21}$$

Where, N is the number of data samples, y_i is the actual value, and $\hat{y}_i$is the predicted value.

3. Mean Absolute Error (MAE) is used to evaluate model performance by calculating the average difference between predicted and actual values, after taking the absolute value of these differences [8]. The MAE is computed as follows:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \tag{22}$$

Where, N is the number of data samples, y_i is the actual value, and $\hat{y}_i$is the predicted value.

4. Mean Absolute Percentage Error (MAPE) is a One of the standard measures used to evaluate model error by calculating the average percentage difference between the pridected and actual values [39]. The MAPE is computed as follows:

$$MAPE = \frac{100}{N}\sum_{i=1}^{N}|\frac{y_i - \hat{y}_i}{y_i}| \tag{23}$$

Where, N is the number of data samples, y_i is the actual value, and $\hat{y}_i$is the predicted value [41].

3. Results and discussion

3.1. Data Description and Study Framework

Data were obtained from agricultural meteorological stations located in Nineveh governate, each pertaining to a different location, namely, Mosul, Tal Afar, Ba'aj, Tal Abta, and Makhmur. This data includes a group of important climatic factors, such as solar radiation, maximum temperature, minimum temperature, relative humidity, and wind speed. The data's time period extends from the 1st of January 1994 to the 31st of December 2024, representing 11,329 daily observations. This provides a long time series that facilitates the analysis of climatic patterns and studying the relationships among these variables across time and space. And considering the large temporal fluctuations and clear seasonal changes in these data in different sites, there is a need for analytical methods capable of capturing each of the temporal and spatial areas of the time series. The time series has been split into two separate seasons according to the time-stratified methodology, to achieve seasonal homogeneity in data. Where each season represents 4 months of the year. The hot seasons includes (June, July, August, and September) that is months 6, 7, 8, and 9. While the cold season includes (December, January, February, and March) which are months 12, 1, 2, and 3. The observations during the hot season reached 3782 observations, while the sightings in the cold season were 3759 observations. The rest of the months are a part of the moderate season and as a result of them being excluded from the analysis in order to focus only on the hot and cold seasons. Since no missing values were available in our data, artificial data missingness has been introduced to the original data so it includes missing values. This has been done by simulating missing data for climatic factors for the five locations. The

missing data were set to a rate of 20% using Block missing pattern, and the missing values were distributed in the form of successive time intervals within the period of the time series for the variable. The Block Missing Pattern was intentionally designed to simulate realistic operational failures in meteorological monitoring systems. In practice, sensor malfunctions, power outages, communication failures, or maintenance activities typically result in consecutive periods of missing observations rather than isolated missing values. Therefore, the missing values were distributed as consecutive time intervals to reproduce these real-world conditions while allowing objective evaluation against known ground-truth values. The reason behind using this method is to evaluate the model's performance under more complex and realistic missing data conditions. To represent the locations of missing values in the X data, a mask vector has been introduced which aims to determine whether the values in the data are observed or missing. Next, a preliminary imputation was performed to the missing values by using one of the traditional methods to ensure the stability of the training process, and K-Nearest Neighbors (KNN) algorithm was chosen and k was set to 8 neighbors for performing the imputation process. The resulting KNN estimates were used only as initial numerical values to replace missing entries before model training. Various models were tested using different configurations of time lags, after which the best model was selected based on various statistical criteria, some of which are the Akaike Information Criterion (AIC) and the Bayesian Information Criterion (BIC) in addition to the Root Mean Square Error (RMSE) as well as testing the significance of lagged variables to be used as inputs in the features matrix. The optimal lag orders were determined separately for each climatic variable using the Akaike Information Criterion (AIC), the Bayesian Information Criterion (BIC), RMSE, and statistical significance tests. For the cold season, the selected lag orders were 1, 3, 4, 5, and 6 for Solar Radiation; 1, 2, 3, 4, and 6 for Maximum Temperature; 1, 2, 4, 6, and 7 for Minimum Temperature; 1, 2, 3, 4, and 6 for Relative Humidity; and 1, 3, 4, 5, and 6 for Wind Speed. For the hot season, the selected lag orders were 1, 2, 3, 5, and 6 for Solar Radiation; 1, 2, 3, 4, and 6 for Maximum Temperature; 1, 2, 4, 5, and 6 for Minimum Temperature; 1, 2, 3, 4, and 6 for Relative Humidity; and 1, 3, 4, 5, and 6 for Wind Speed. The differences in the selected lag orders among climatic variables and between the hot and cold seasons indicate that the temporal persistence of climatic variables is not constant across seasons. Climatic variables are influenced by different atmospheric processes under different seasonal conditions, resulting in distinct temporal dependency structures. Consequently, the statistical selection procedure identified different lag orders for each climatic variable and season based on the temporal dependencies present in the data. After identifying the optimal time lags which gives the best model, these lags were used in generating new time properties for the variable whose missing values are to be imputed. Because using time lags leads to losing some values at the start of the time series, these values have been imputed using one of the traditional methods. Lastly, the data of the time lags were multiplied by the model's coefficients to obtain the final time properties. After the completion of time properties using time lags, these properties are then grouped into a features matrix that represents the temporal state of each location, for them to be later used as initial inputs for the Graph Convolutional Network where is represented in the input layer. Achieving stability for the time series before identifying the time lags has not been taken into consideration, due to the time lags selected for this study not been used to construct AR or ARIMA models in the traditional statistical meaning. They were instead used as a method to extract the time properties that contribute to the estimation of missing values. In addition, the applying of differences to achieve stability could lead to changing the original properties for the time series, while as the imputation of missing values depends mainly on the properties of neighboring observations and their temporally corresponding counterparts [42]. Furthermore, the data were standardized using Z-score normalization, and a Time Stratification approach was adopted to reduce seasonal heterogeneity and improve data homogeneity prior to model training. Therefore, the time lags adopted in this study should not be referred to as AR or ARIMA models, but as time lags selected for their statistical significance, while excluding the insignificant time lags. The A adjacency matrix was constructed based on the spatial relationships among the five meteorological stations, where each station was considered a node in the graph. The relationships between the nodes have been set based on closeness and the geographical distance between the stations. Each season's data has been divided to three groups: Training (80%), validation (10%), and testing (10%) to evaluate the model's performance through different training stages. The adjacency matrix was constructed based on the geographical distances between the meteorological stations, as shown in Table (1). A threshold distance of 100 km was adopted to define the spatial connections between stations. An edge was established between two stations when the geographical

distance separating them was less than 100 km; otherwise, no edge was created. The adjacency relationship can be expressed by the following equation:

$$a_{ij} = \begin{cases} 1\ ,\ if\ Dist_{ij} < 100\ Km \\ 0,\ \ if\ Dist_{ij} \geq 100\ Km \end{cases} \tag{24}$$

Table 1. Geographical distances (km) between the meteorological stations

Stations	**Mosul**	**Tal Afar**	**Ba'aj**	**Tal Abta**	**Makhmur**
Mosul	—	59.970	131.002	68.730	71.443
Tal Afar	59.970	—	76.829	50.594	118.407
Ba'aj	131.002	76.829	—	76.103	168.734
Tal Abta	68.730	50.594	76.103	—	92.634
Makhmur	71.443	118.407	168.734	92.634	—

The 100 km threshold was chosen based on the spatial distribution of stations, as it represents the smallest distance that maintains graph connectivity while avoiding unnecessary links between geographically dispersed stations. Based on the above, a binary adjacency matrix was constructed to represent the spatial connections between the meteorological stations. The resulting adjacency matrix is presented as follows:

$$A = \begin{vmatrix} 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{vmatrix} \tag{25}$$

The adjacency matrix was normalized using symmetric normalization as given in Equation (13). The resulting normalized adjacency matrix is given by the following matrix:

$$A = \begin{vmatrix} 0.2500 & 0.2500 & 0 & 0.2236 & 0.2887 \\ 0.2500 & 0.2500 & 0.2887 & 0.2236 & 0 \\ 0 & 0.2887 & 0.3333 & 0.2582 & 0 \\ 0.2236 & 0.2236 & 0.2582 & 0.2000 & 0.2582 \\ 0.2887 & 0 & 0 & 0.2582 & 0.3333 \end{vmatrix} \tag{26}$$

The Graph Convolutional Networks (GCN) model have been trained using the (Adam) improvement algorithm, where the model's weights were iteratively updated during the training process to minimize prediction errors. The training process has been performed for a period of 2500 training epochs using a learning average of 0.005 to achieve a balance between the stability of the training process and the speed of convergence. These values have been selected based on initial experiments to achieve the best performance for the model. The Glorot (Xavier) initialization has been used to initialize the model's weights, considering its ability to maintain variance balance across layers, helping in enhancing the training's stability and accelerating the convergence process. The proposed GCN architecture consists of three graph convolutional layers. The first and second graph convolutional layers contain 32 hidden units and use the ReLU activation function, while the final output layer produces the final predictions without an activation function. The input data were organized in a three-dimensional structure , where T denotes the number of time steps, S=5 represents the five meteorological stations, and F=5 corresponds to the five temporal lag features used as input variables. A separate model was trained for each climatic variable, with the model output representing the estimated value of the target climatic variable for each station–time observation. The model was optimized using a Masked Mean Squared Error (Masked MSE) loss function, where the loss was computed only on the observed values while excluding missing entries from the optimization process. The initial KNN estimates were used solely to provide numerical values at missing locations and were not treated as

target values during model training. The default Adam hyperparameters provided by MATLAB were adopted, and model validation was performed every 300 training epochs to monitor learning progress and training stability. After adding it to the network and obtaining the predicted values, the available original values are restored, which were not missing, and keeping the predictions of the missing values only using the M mask information. The model's performance is evaluated using error metrics represented with the Root Mean Square Error (RMSE) and the Mean Absolute Error (MAE) and the Mean Absolute Percentage Error (MAPE). Where these metrics are calculated on the missing values using only mask information. This process is repeated for each factor of the five climatic factors to achieve a complete representation of the data across the different sites. The general study framework can be summarized as follows:

1. Dividing the time series for climatic data to two primary seasons (hot and cold) using the time-split method in order to achieve seasonal homogeneity and reducing the contrast within the data, helping to enhance the model's performance. Next, each season's data is divided to three groups: the training group (80%), the validation group (10%), and the testing group (10%) in order to evaluate the model's performance during different training stages as well as testing its ability to generalize.
2. Introducing a 20% artificial missingness in the original data using the block missing pattern for deiffusion. Then, the mask vector is introduced to represent the locations of the missing values in the X data, which is later used to determine whether the values in the data are observed or missing.
3. Performing an initial imputation for the missing values using the K-Nearest Neighbors algorithm to ensure the stability of the training process.
4. The optimal time lags are determined to use them in generating new time properties. In addition to imputing the missing values at the start of the series resulting from using tine lags.
5. Introducing the resulting properties to the Graph Convolutional Network model to extract the spatial and temporal relationships among the sites followed by predicting the missing values.
6. After obtaining the model's outputs, its performance is evaluated using RMSE, MAE, and MAPE metrics where these metrics were calculated only on the missing values according to the mask vector.
7. The previous steps are applied to every factor of the five climatic factors to obtain a comprehensive representation for data across all sites.

The overall workflow of the proposed methodology is illustrated in Figure 4

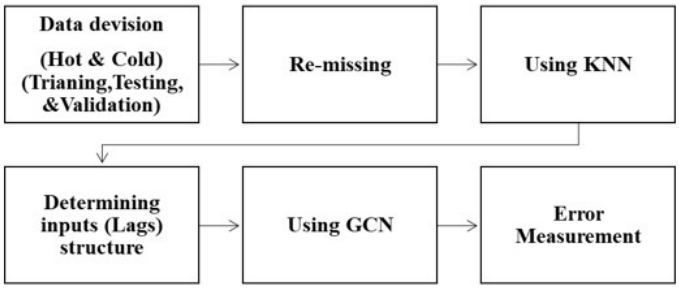

Figure 4. Overall workflow of the proposed methodology for missing value imputation.

In addition to the K-Nearest Neighbors (KNN) method used previously, two additional benchmark methods were considered for comparison. The first was Linear Interpolation (LI), which estimates missing values based on the linear relationship between neighboring observations in a time series. The second was GRU-D, a recurrent neural network designed specifically for handling missing data through the use of masking vectors and time intervals between observations. For consistency, all methods were evaluated using the same missing-value locations and the same evaluation metrics RMSE, MAE, and MAPE. The GRU-D model was implemented using a 15-day time window and five input variables corresponding to the five monitoring stations. A single GRU-D layer with 64 hidden units was used, followed by a dense output layer consisting of five neurons with a linear activation function.

The model was trained using the Adam optimizer with a learning rate of 0.001 for 100 epochs and a batch size of 32. To reduce overfitting, both dropout and recurrent dropout rates were set to 0.3. The output layer was designed to estimate the missing values for the five monitoring stations simultaneously. Model training was performed using a masked MSE loss function, where the loss was computed only from observed values, while model evaluation was carried out exclusively on the missing values.

3.2. K-Nearest Neighbor (KNN)

Table (2) shows the results of the initial imputation of the missing values using K-Nearest Neighbors method during the cold season, where each row represents a climatic factor while every column displays the error metrics being used.

Table 2. the results of the initial imputation of the missing values using K-Nearest Neighbors method during the cold season

	RMSE	MAE	MAPE
Solar	1.8294	1.4435	47.6112
Temp Max	3.2493	2.5383	19.0746
Temp Min	2.9178	2.2991	137.2186
Humidity	12.97	10.3459	18.1046
Wind Speed	0.99154	0.74663	40.4967

Table (2) illustrates the results of the initial imputation of the missing values using K-Nearest Neighbors method during the cold season. The results show a relative increase in the value's error rate in the minimum temperature and humidity variables. This is due to these variables fluctuating nature during the cold season in addition to the effect of consecutive missingness which leads to the temporal information becoming weak or unavailable, this only increases the difficulty of the imputation process. The MAPE values also show a noticeable increase due to the sensitivities of these metrics and their susceptibility to small true values, leading to an inflation in the error rate.
Table (3) shows the results of the initial imputation of the missing values using K-Nearest Neighbors method during the hot season

Table 3. the results of the initial imputation using the K-Nearest Neighbors method during the hot season

	RMSE	MAE	MAPE
Solar	1.0139	0.67521	6.1643
Temp Max	2.5033	1.9592	4.8292
Temp Min	1.9848	1.5614	6.4656
Humidity	5.4941	4.3403	22.3143
Wind Speed	0.90736	0.71265	26.541

Table (3) also illustrates the results of the initial imputation during the hot season, where a relative improvement is noticeable in the performance due to the decrease of data variability in the hot season, which contributed to an improvement in the imputation accuracy and a decrease in value errors. Despite this improvement, some variables still show relatively higher errors as a result of their fluctuating nature.

3.3. Gated Recurrent Unit with Decay - GRU-D

Table (4) shows the results of imputation of the missing values using GRU-D model during the cold season, where each row represents a climatic factor while every column displays the error metrics being used.

Table (5) shows the results of imputation of the missing values using GRU-D model during the hot season, where each row represents a climatic factor while every column displays the error metrics being used.

The results presented in Tables (4) and (5) show that the GRU-D model was able to impute missing values for all climatic variables during both the cold and hot seasons. However, the level of performance varied according to

Table 4. GRU-D performance under block missing pattern in the cold season

	Dataset	**RMSE**	**MAE**	**MAPE**
Solar Radiation	Training	1.6375	1.2541	42.0954
	Validation	1.9270	1.3724	46.9627
	Testing	1.7357	1.2984	39.0843
Temp Max	Training	2.6017	1.9249	16.7231
	Validation	2.8416	2.1619	18.5379
	Testing	2.7154	1.9810	17.8620
Temp Min	Training	2.6619	1.7866	135.1618
	Validation	2.7010	1.8953	97.2294
	Testing	2.6939	1.8751	67.5483
Humidity	Training	9.3851	7.7512	15.3143
	Validation	12.0187	9.6071	16.2761
	Testing	11.6179	8.3912	15.5692
Wind Speed	Training	0.9650	0.7228	39.2073
	Validation	1.1473	0.8179	42.5894
	Testing	0.9708	0.7346	40.1615

Table 5. GRU-D performance under block missing pattern in the hot season

	Dataset	**RMSE**	**MAE**	**MAPE**
Solar Radiation	Training	0.8318	0.6147	5.3647
	validation	0.7993	0.5912	5.1228
	testing	0.8741	0.6450	5.6146
Temp Max	Training	1.9257	1.7527	4.3719
	validation	2.0215	1.8201	4.6670
	testing	1.9404	1.7850	4.1592
Temp Min	Training	1.9274	1.5128	5.2267
	validation	2.1482	1.6249	5.3679
	testing	1.9122	1.5518	4.9370
Humidity	Training	8.2141	6.9510	19.5177
	validation	4.9718	3.8429	20.3757
	testing	7.2500	5.9727	19.2535
Wind Speed	Training	0.8097	0.7071	24.1250
	validation	0.8514	0.7314	24.8151
	testing	0.8869	0.7164	28.9125

the climatic variable and the season. In general, the model achieved lower error values during the hot season than during the cold season for most climatic variables, indicating that the greater stability and lower variability of the data in the hot season contributed to improving imputation accuracy.The results also showed that solar radiation and wind speed were among the variables with the lowest error values, whereas relative humidity recorded the highest error values in both seasons, reflecting the greater difficulty of accurately imputing this variable compared with the other climatic variables.. In addition, the error values were relatively similar across the training, validation, and testing datasets, indicating stable training behavior and a good generalization capability without significant overfitting.

3.4. Linear interpolation (LI)

Table (6) presents the performance results of the Linear Interpolation (LI) method under the block missing pattern in the cold season

Table 6. Linear Interpolation performance under block missing pattern in the cold season

Cold	RMSE	MAE	MAPE
Solar Radiation	1.9121	1.5397	47.8520
Temp Max	3.6009	2.5942	19.5915
Temp Min	2.9410	2.4493	149.2548
Humidity	13.1759	10.7568	18.7076
Wind Speed	1.1387	0.8485	45.8957

Table (7) presents the performance results of the Linear Interpolation (LI) method under the block missing pattern in the hot season.

Table 7. Linear Interpolation performance under block missing pattern in the hot season

Hot	RMSE	MAE	MAPE
Solar Radiation	1.1956	0.7448	6.2444
Temp Max	2.7976	2.2678	4.9564
Temp Min	2.2975	1.6810	6.7419
Humidity	5.8790	4.7103	23.6842
Wind Speed	0.9905	0.7759	28.8985

Tables (6) and (7) present the performance results of the linear interpolation (LI) method under the block missing pattern in the cold and hot seasons. The results indicate that the method generally performed better in the hot season compared to the cold season, with most climatic variables registering lower values for the RMSE, MAE, and MAPE error measures. This can be explained by the lower variability and greater stability of climatic values during the hot season, making the hypothesis of linear variation between adjacent time values more suitable for the nature of the data. In contrast, the method showed a significant increase in error values during the cold season, particularly for the minimum temperature and relative humidity variables, which is attributed to the increased variability in climatic conditions during this season. Overall, the results demonstrate that the performance of the linear interpolation method is affected by the regularity and stability of the time series, achieving higher accuracy when the changes between adjacent values are smoother and less fluctuating

3.5. Graph Convolutional Network (GCN)

This section deals with the results of using the Graph Convolutional Network model on time consecutive data and their performance in compensating for the missing values. Table (8) shows the networks performance under the block missing pattern in the cold season. A relative increase is noticeable in the values' error rate due to the difficulty of imputing the consecutive missing values. Despite this, the network still maintains a consistent behavior across different data sets, indicating that there is no issue with overfitting

Table (8) shows results of the Graph Convolutional Networks (GCN) model in the Block Missing Pattern during the cold season. It is noticeable that the error values are close across training, validation and testing, which indicates that the model is stable and is able to generalize. The results also show differences in the error values among variables. These differences in values reflect the variable nature of these variables, as well as the sensitivity of some metrics such as MAPE to small values.

Table (9) summarizes the performance of Graph Convolutional Network under the block missing pattern in the hot season. The results show a clear convergence in the error values across the training, validation, and testing groups. Indicating the stability of the model and its ability to generalize. Additionally, an improvement in the values' errors is noticeable in the cold season.

Table 8. Graph Convolutional Network performance under block missing pattern in the cold season

	Dataset	RMSE	MAE	MAPE
Solar Radiation	Training	1.4980	1.1679	39.6830
	validation	1.8105	1.4192	44.0295
	testing	1.6005	1.2592	34.2718
Temp Max	Training	2.2595	1.7327	13.5104
	validation	2.2440	1.7705	13.3019
	testing	2.2813	1.6948	11.0177
Temp Min	Training	2.1225	1.6624	105.9934
	validation	2.1546	1.6940	88.6626
	testing	2.1675	1.6411	58.3512
Humidity	Training	8.6632	6.8145	11.9985
	validation	9.8861	8.0768	13.2957
	testing	9.1126	7.0253	11.9474
Wind Speed	Training	0.8356	0.6325	34.6881
	validation	1.0433	0.7944	40.0524
	testing	0.8880	0.6891	39.8157

This is due to the decrease of variability of the climatic data in the hot season, leading to an improvement in prediction accuracy.

Table 9. the performance of Graph Convolutional Network under the block missing pattern in the hot season

	Dataset	RMSE	MAE	MAPE
Solar Radiation	Training	0.7273	0.5027	4.5628
	validation	0.6742	0.4572	4.2596
	testing	0.7593	0.5183	4.7980
Temp Max	Training	1.6111	1.2437	3.0773
	validation	1.5921	1.2528	3.0643
	testing	1.5958	1.1965	2.8564
Temp Min	Training	1.6231	1.2816	5.3896
	validation	1.6659	1.3356	5.4302
	testing	1.3296	1.0732	3.9861
Humidity	Training	4.5157	3.4233	17.4422
	validation	4.2099	3.2638	15.4755
	testing	4.4078	3.4127	15.8542
Wind Speed	Training	0.8193	0.6515	23.3933
	validation	0.7998	0.6074	24.1963
	testing	0.8864	0.7150	28.8901

Table (9) illustrates the results of the Graph Convolutional Network model in the state of block missing pattern in the hot season. The results show a clear stability across the training, validation and testing groups. An improvement in performance is also noticeable compared to the cold season due to the decrease of variability in the data. Solar radiation recorded the lowest error values, while humidity showed the highest RMSE values, with the maximum and minimum temperature remaining within a good performance level.

After separately evaluating the performance of the KNN method, the Linear Interpolation (LI) method, the GRU-D model, and the proposed GCN model, a comprehensive comparison was conducted to assess their effectiveness in imputing missing values under the Block Missing Pattern. The results reported in this study represent the

average performance obtained from more than ten independent repetitions conducted under different missing-value realizations. Statistical significance tests were also performed, specifically using a paired Student's t-test, and all reported comparisons were found to be statistically significant at the 0.05 significance level ($p < 0.05$). In addition, the Diebold–Mariano (DM) test was conducted to further confirm the statistical significance of the differences in imputation accuracy between the proposed GCN model and the competing methods. The obtained p-values ranged from 0.001 to 0.045 across both the hot and cold seasons, and all comparisons were statistically significant ($p < 0.05$). These findings provide additional evidence that the superior performance of the proposed GCN model was not due to random variation. Tables (10) and (11) present the comparison results for the hot and cold seasons, respectively, using the RMSE, MAE, and MAPE evaluation metrics.

Table 10. Comparison of KNN, LI, GRU-D and GCN under block missing pattern in the hot season

Metric	**Dataset**	**KNN**	**LI**	**GRU-D**	**GCN**
RMSE	Solar Radiation	1.0139	1.1956	0.8741	**0.7593**
	Temp Max	2.5033	2.7976	1.9404	**1.5958**
	Temp Min	1.9848	2.2975	1.9122	**1.3296**
	Humidity	5.4941	5.8790	5.2360	**4.4078**
	Wind Speed	0.9074	0.9905	0.8869	**0.8864**
MAE	Solar Radiation	0.6752	0.7448	0.6450	**0.5183**
	Temp Max	1.9592	2.2678	1.7850	**1.1965**
	Temp Min	1.5614	1.6810	1.5518	**1.0732**
	Humidity	4.3403	4.7103	3.9727	**3.4127**
	Wind Speed	**0.7127**	0.7759	0.7154	0.7150
MAPE	Solar Radiation	6.1643	6.2444	5.6146	**4.7980**
	Temp Max	4.8292	4.9564	4.1592	**2.8564**
	Temp Min	6.4656	6.7419	4.9370	**3.9861**
	Humidity	22.3143	23.6842	19.2330	**15.8542**
	Wind Speed	26.5410	28.8985	28.9125	**28.8901**

Table 11. Comparison of KNN, LI, GRU-D and GCN under block missing pattern in the cold season

Metric	**Dataset**	**KNN**	**LI**	**GRU-D**	**GCN**
RMSE	Solar Radiation	1.8294	1.9121	1.7357	**1.6005**
	Temp Max	3.2493	3.6009	2.7154	**2.2813**
	Temp Min	2.9178	2.9410	2.6939	**2.1675**
	Humidity	12.9700	13.1759	11.6179	**9.1126**
	Wind Speed	0.9915	1.1387	0.9708	**0.8880**
MAE	Solar Radiation	1.4435	1.5397	1.2984	**1.2592**
	Temp Max	2.5383	2.5942	1.9810	**1.6948**
	Temp Min	2.2991	2.4493	1.8751	**1.6411**
	Humidity	10.3459	10.7568	8.3912	**7.0253**
	Wind Speed	0.7466	0.8485	0.7346	**0.6891**
MAPE	Solar Radiation	47.6112	47.8520	39.0843	**34.2718**
	Temp Max	19.0746	19.5915	17.8620	**11.0177**
	Temp Min	137.2186	149.2548	67.5483	**58.3512**
	Humidity	18.1046	18.7076	15.5692	**11.9474**
	Wind Speed	40.4967	45.8957	40.1615	**39.8157**

The comparison results show that the Graph Convolutional Network (GCN) model achieved the best overall performance when compared with KNN, Linear Interpolation (LI), and the GRU-D model. In the hot season, GCN

obtained the lowest error values across most climatic variables and evaluation metrics, while GRU-D generally achieved the second-best performance. In the cold season, GCN maintained its superiority across all climatic variables and evaluation metrics, demonstrating its ability to reconstruct missing values even under conditions characterized by higher variability and limited temporal information. This superiority can be attributed to the ability of GCN to exploit spatial relationships between meteorological stations and propagate information through connected nodes in the graph. In contrast, KNN relies primarily on similarity between observations, whereas LI depends only on adjacent temporal values. Although GRU-D is capable of modeling temporal patterns and handling missing values through masking and decay mechanisms, it does not directly exploit the spatial structure among stations. Consequently, GCN was more effective in compensating for the loss of temporal information caused by the Block Missing Pattern, which was reflected in lower RMSE, MAE, and MAPE values across most of the studied climatic variables.

Figure (5) illustrates the scatter plot of the true and imputed minimum temperature values for the cold season using the proposed Graph Convolutional Network (GCN) and K-Nearest Neighbors (KNN) methods. The dashed line represents the ideal agreement line ($y = x$), while the solid lines represent the regression lines for both methods. The imputed values obtained by the proposed GCN are more closely distributed around the ideal line than those produced by KNN, demonstrating the superior imputation performance of the proposed model.

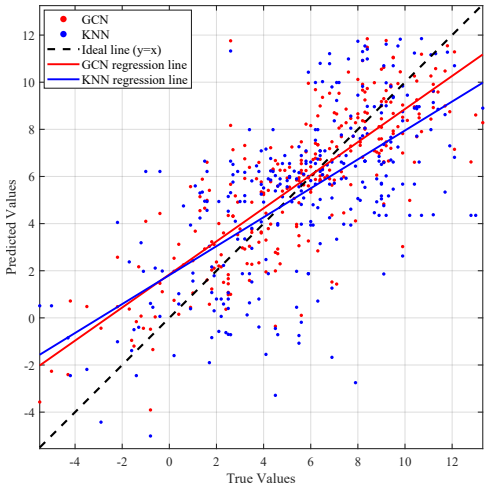

Figure 5. Scatter plot of the true versus imputed minimum temperature values in the cold season using the proposed Graph Convolutional Network (GCN) and K-Nearest Neighbors (KNN) methods. The dashed line indicates the ideal agreement line ($y = x$), whereas the solid lines represent the regression lines for the GCN and KNN methods.

The imputed values obtained by the proposed GCN are more closely distributed around the ideal agreement line than those produced by the KNN method, indicating a stronger agreement with the ground-truth values and demonstrating the superior imputation performance of the proposed model.

Figure (6) illustrates a comparison of the root mean square error (RMSE) values for the KNN, LI, and GRU-D methods and the proposed GCN model across different climatic variables in the cold season. It can be observed that the GCN model achieved the lowest RMSE values for all climatic variables, demonstrating its high effectiveness in imputing missing values under the block missing pattern.

Figure (7) illustrates the Graph Convolutional Networks performance in evaluating missing values for the solar radiation variable in the hot season, where it shows a clear convergence in true and predicted values.

Results show that the Graph Convolutional Network model is achieving a stable performance and is able to be generalized across different climatic variables in both the cold and hot seasons. The results further confirm model's effectiveness in dealing with missing data compared other imputation methods, including K-Nearest Neighbors (KNN), Linear Interpolation (LI), and GRU-D as the model demonstrates a greater ability in dealing with climatic

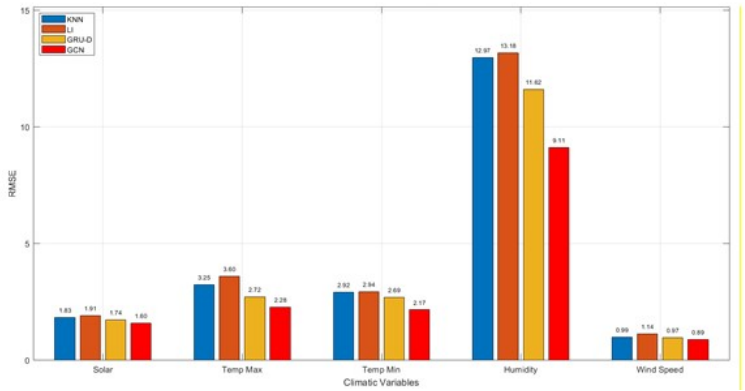

Figure 6. Comparison of KNN, Linear Interpolation (LI), GRU-D, and Graph Convolutional Network (GCN) across different climatic variables in the cold season using the RMSE metric.

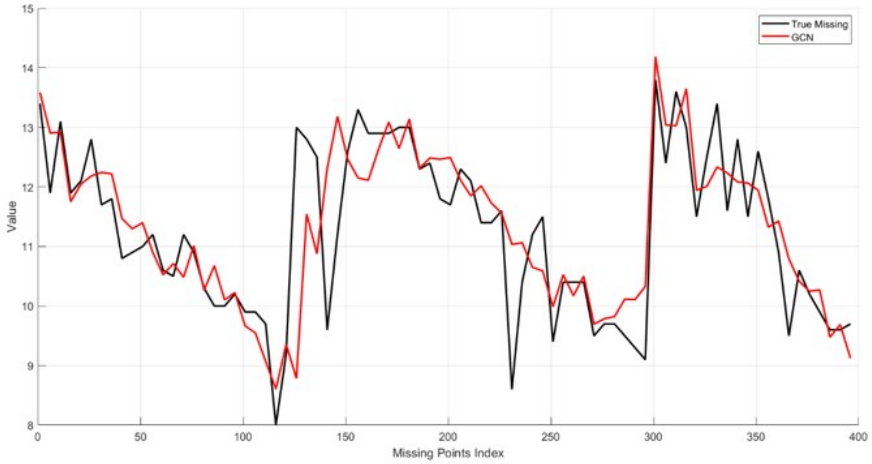

Figure 7. A line plot for a comparison between true and predicted values using the Graph Convolutional Network for the solar radiation variable in the hot season under the Block Missing Pattern.

data. This advantage is due to the Graph Convolutional Network relying on combining data from neighboring nodes through spatial relationships between stations, allowing for imputing the missing values more efficiently.

The results also demonstrated that the model's performance is affected by the data properties, as the data's stability and the reduction in its variability, just like in the hot season, plays an important role in reinforcing the model's performance and reducing errors compared to the cold season. The Block Missing Pattern is distinguished by an interruption in temporal information, increasing the difficulty of the imputation process. Nevertheless, the Graph Convolutional Network is able to reconstruct the missing values by propagating information across connected nodes in the graph, while KNN relies primarily on similarity between observations, LI depends on adjacent temporal observations, and GRU-D focuses mainly on temporal dependencies without directly exploiting spatial relationships between stations. This demonstrates that the models which rely on the spatial structure provides a more comprehensive representation compared to traditional methods.

Despite the promising results obtained in this study, several limitations should be acknowledged. Although the Block Missing Pattern adopted in this study provides a realistic simulation of consecutive missing intervals caused by operational failures in meteorological monitoring systems, it cannot fully represent all missing-data mechanisms encountered in real-world applications, which may be considerably more diverse and complex than the artificial

missing pattern considered in this study. Consequently, the findings of this study should be interpreted within the limitations associated with the use of artificially generated missing data. Furthermore, validating the proposed model using datasets containing naturally occurring missing observations represents an important direction for future research, providing further evidence of its generalizability under real-world operating conditions.

3.6. Comparison with Related Work

[9] addressed the problem of missing node features in graph data by integrating a Gaussian Mixture Model (GMM) with a Graph Convolutional Network (GCN) to handle missing values during the learning process without relying on pre-imputation. In contrast, [13] proposed a self-supervised framework for learning spatiotemporal representations from incomplete data to improve forecasting performance. Although these approaches have demonstrated promising results, they are primarily designed for general graph data or forecasting tasks and rely on additional components that increase training complexity. The present study differs in both objective and application, as it focuses directly on missing-value imputation in multi-site climatic time series under the Block Missing Pattern. This is achieved by exploiting spatial relationships among meteorological stations and temporal lag features within a GCN framework, in addition to employing Temporal Stratification (TS) to address seasonal heterogeneity. Consequently, the proposed approach provides a more application-oriented framework specifically tailored for missing-value imputation in climatic data while still benefiting from the underlying spatial structure of the data.

4. Conclusions

The Graph Convolutional Networks (GCN) model achieves a superior performance in missing data imputation across different variables (climatic factors) and in both the hot and cold seasons. The performance metrics (MAPE, MAE, and RMSE) showed that the suggested model achieved the lowest error in values compared to K-Nearest Neighbors (KNN), Linear Interpolation (LI), and GRU-D, indicating that the model is not limited to only improving the numerical performance, but also succeeds in reconstructing the missing values with a higher accuracy. It can be inferred that using spatial relationships between the monitoring stations is a critical factor in improving the representation's quality. As it allows for passing and combining information between connected nodes in the graph, enabling it to impute for missing values even in cases with weak or interrupted temporal information, unlike KNN, LI, and GRU-D, which are more affected by the loss of temporal information or do not directly exploit spatial relationships between stations. In addition, it has been demonstrated that the data properties themselves play an essential role in the models' performance. As the decrease in variation and the values stability contributes in the model's prediction accuracy, indicating that the quality of data and its stability have an important effect on the efficiency of the compensation process. Overall, and from what have been previously mentioned, we are able conclude that combining spatial data with the field of deep learning, as in the Graph Convolutional Network model, is an effective methodology in processing missing values in time series forecasting, especially in complex scenarios where traditional methods are unable to present an accurate representation of the relationships between data.

REFERENCES

1. D. B. Rubin, *Inference and Missing Data*, Biometrika, vol. 63, no. 3, pp. 581–592, 1976.
2. H. Kang, *The Prevention and Handling of the Missing Data*, Korean Journal of Anesthesiology, vol. 64, no. 5, pp. 402–406, 2013.
3. D. J. Stekhoven and P. Bühlmann, *MissForest—Non-parametric Missing Value Imputation for Mixed-Type Data*, Bioinformatics, vol. 28, no. 1, pp. 112–118, 2012.
4. Z. Che, S. Purushotham, K. Cho, D. Sontag, and Y. Liu, *Recurrent Neural Networks for Multivariate Time Series with Missing Values*, Scientific Reports, vol. 8, no. 1, p. 6085, 2018.
5. W. Cao, D. Wang, J. Li, H. Zhou, L. Li, and Y. Li, *BRITS: Bidirectional Recurrent Imputation for Time Series*, Advances in Neural Information Processing Systems, vol. 31, 2018.
6. T. N. Kipf and M. Welling, *Semi-Supervised Classification with Graph Convolutional Networks*, arXiv preprint arXiv:1609.02907, 2016.

7. Y. Li, R. Yu, C. Shahabi, and Y. Liu, *Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting*, arXiv preprint arXiv:1707.01926, 2017.
8. B. Yu, H. Yin, and Z. Zhu, *Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting*, arXiv preprint arXiv:1709.04875, 2017.
9. H. Taguchi, X. Liu, and T. Murata, *Graph Convolutional Networks for Graphs Containing Missing Features*, Future Generation Computer Systems, vol. 117, pp. 155–168, 2021.
10. C. Contreras Vargas, J. Quintero Ibáñez, and Á. Solanilla, *Aplicación de Ciencia de Datos para la Reconstrucción de Series de Tiempo de Variables Meteorológicas en Islas del Rosario (Caribe Colombiano) entre los Años 2013–2021*, 2022.
11. Y. Yu, et al., *GCN-ST-MDIR: Graph Convolutional Network-Based Spatial-Temporal Missing Air Pollution Data Pattern Identification and Recovery*, IEEE Transactions on Big Data, vol. 9, no. 5, pp. 1347–1364, 2023.
12. K. Qaraghuli, et al., *Univariate and Multivariate Imputation Methods Evaluation for Reconstructing Climate Time Series Data: A Case Study of Mosul Station-Iraq*, Journal of Agrometeorology, vol. 26, no. 3, pp. 318–323, 2024.
13. S. Ke, et al., *GeoMAE: Masking Representation Learning for Spatio-Temporal Graph Forecasting with Missing Values*, arXiv preprint arXiv:2508.14083, 2025.
14. M. Poette, et al., *Benchmarking Imputation Strategies for Missing Time-Series Data in Critical Care Using Real-World-Inspired Scenarios*, Scientific Reports, 2026.
15. S. B. Imandoust and M. Bolandraftar, *Application of K-Nearest Neighbor (KNN) Approach for Predicting Economic Events: Theoretical Background*, International Journal of Engineering Research and Applications, vol. 3, no. 5, pp. 605–610, 2013.
16. H. Junninen, et al., *Methods for Imputation of Missing Values in Air Quality Data Sets*, Atmospheric Environment, vol. 38, no. 18, pp. 2895–2907, 2004.
17. J. Shao, W. Meng, and G. Sun, *Evaluation of Missing Value Imputation Methods for Wireless Soil Datasets*, Personal and Ubiquitous Computing, vol. 21, no. 1, pp. 113–123, 2017.
18. H. Liu, H.-q. Tian, and Y.-f. Li, *Comparison of Two New ARIMA-ANN and ARIMA-Kalman Hybrid Methods for Wind Speed Prediction*, Applied Energy, vol. 98, pp. 415–424, 2012.
19. O. B. Shukur, *Daily Wind Speed Forecasting Through Hybrid AR-ANN and AR-KF Models*, Jurnal Teknologi, 2015.
20. G. P. Zhang, *Time Series Forecasting Using a Hybrid ARIMA and Neural Network Model*, Neurocomputing, vol. 50, pp. 159–175, 2003.
21. M. Khashei and M. Bijari, *An Artificial Neural Network (p, d, q) Model for Time Series Forecasting*, Expert Systems with Applications, vol. 37, no. 1, pp. 479–489, 2010.
22. K. Chen and J. Yu, *Short-Term Wind Speed Prediction Using an Unscented Kalman Filter Based State-Space Support Vector Regression Approach*, Applied Energy, vol. 113, pp. 690–705, 2014.
23. G. Concu, B. De Nicolo, and M. Valdes, *Prediction of Building Limestone Physical and Mechanical Properties by Means of Ultrasonic P-Wave Velocity*, The Scientific World Journal, vol. 2014, no. 1, p. 508073, 2014.
24. A. Begga, F. Escolano, and M. Á. Lozano, *Eigenvector Distance-Modulated Graph Neural Network: Spectral Weighting for Enhanced Node Classification*, Mathematics, vol. 13, no. 17, p. 2895, 2025.
25. M. Ahmed, V. Maume-Deschamps, and P. Ribereau, *Recognizing a Spatial Extreme Dependence Structure: A Deep Learning Approach*, Environmetrics, vol. 33, no. 4, p. e2714, 2022.
26. K. He, X. Zhang, S. Ren, and J. Sun, *Deep Residual Learning for Image Recognition*, In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778, 2016.
27. G. I. Liapis, S. Tsoka, and L. G. Papageorgiou, *Optimisation-Based Feature Selection for Regression Neural Networks Towards Explainability*, Machine Learning and Knowledge Extraction, vol. 7, no. 2, p. 33, 2025.
28. M. Ahmed, V. Maume-Deschamps, and P. Ribereau, *Spatial Risk Measures for Max-Stable and Max-Mixture Processes*, Stochastics, vol. 92, no. 7, pp. 1005–1020, 2020.
29. W. Hashm and M. H. Ahmed, *A Quantitative Spatial Risk Measure for Extreme Events*, In Proceedings of the 2021 7th International Conference on Contemporary Information Technology and Mathematics (ICCITM), IEEE, 2021.
30. J. Terven, et al., *A Comprehensive Survey of Loss Functions and Metrics in Deep Learning*, Artificial Intelligence Review, vol. 58, no. 7, p. 195, 2025.
31. S. Mohsin, et al., *Novel Logistic Extreme Value Distribution: Properties, Applications, and Parameter Estimation Using Classical and Machine Learning Methods*, Mathematical Modelling of Engineering Problems, vol. 12, no. 6, 2025.
32. S. Sharma, S. Sharma, and A. Athaiya, *Activation Functions in Neural Networks*, Towards Data Science, vol. 6, no. 12, pp. 310–316, 2017.
33. I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, Cambridge, MA, 2016.
34. O. Shukura, *Using the MLR and Neuro-Fuzzy Methods to Forecast Air Pollution Datasets*, International Journal on Advanced Science, Engineering and Information Technology, vol. 10, no. 4, pp. 1457–1464, 2020.
35. R. A. S. Snaan and O. B. Shukur, *Using Bayesian Ridge Regression Model and ESN for Climatic Time Series Forecasting*, Statistics, Optimization & Information Computing, vol. 14, no. 3, pp. 1226–1243, 2025.
36. B. J. Malig, et al., *A Time-Stratified Case-Crossover Study of Ambient Ozone Exposure and Emergency Department Visits for Specific Respiratory Diagnoses in California (2005–2008)*, Environmental Health Perspectives, vol. 124, no. 6, p. 745, 2015.
37. A. Tobias, B. Armstrong, and A. Gasparrini, *Analysis of Time-Stratified Case-Crossover Studies in Environmental Epidemiology Using Stata*, In United Kingdom Stata Users' Group Meetings, Stata Users Group, 2014.
38. A. Jadon, A. Patil, and S. Jadon, *A Comprehensive Survey of Regression-Based Loss Functions for Time Series Forecasting*, In International Conference on Data Management, Analytics & Innovation, Springer, 2024.
39. A. Jadon and A. Patil, *A Comprehensive Survey of Evaluation Techniques for Recommendation Systems*, In International Conference on Computation of Artificial Intelligence & Machine Learning, Springer, 2024.
40. J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl, *Evaluating Collaborative Filtering Recommender Systems*, ACM Transactions on Information Systems (TOIS), vol. 22, no. 1, pp. 5–53, 2004.

41. O. B. Shukur and M. H. Lee, *Daily Wind Speed Forecasting Through Hybrid AR-ANN and AR-KF Models*, Jurnal Teknologi (Sciences & Engineering), vol. 72, no. 5, 2015.
42. M. A. H. Alsaeegh and O. B. Shukur, *Using Multiple Regression Model and RNN for Imputing the Missing Values of PM10 Datasets*, International Journal on Advanced Science, Engineering and Information Technology, vol. 10, no. 6, pp. 2582–2592, 2020.