

Using Graph Neural Networks (GNN) and Gated Recurrent Unit (GRU) based on Regression Modelling for Climatic Forecasting

Osamah Basheer Shukur*

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

Abstract Most climate time series forecasts suffer from inaccuracy, which can negatively and directly impact the effects of forecasting on human life and living organisms, especially when the data has both temporal and spatial dimensions. The use of traditional forecasting methods when dealing with complex non-linear patterns, as well as the problem of data heterogeneity resulting from variations in the temporal (seasonal) and spatial dimensions, as is the case with multi-site climate data, can lead to reduced forecast accuracy. This study used multivariable climate time series data collected from meteorological stations in Nineveh Governorate, Iraq, for daily observations from the beginning of 1994 to the end of 2024. Graph neural networks (GNN), a deep learning technique, were used to process the spatio-temporal data by establishing node-to-node connections based on neighboring information to improve forecast quality by combining the spatio-temporal dimensions. To address the long-term and short-term memory challenges faced by climate data, a gated recurrent unit (GRU) was employed as an alternative to long short-term memory (LSTM) Networks. GRUs offer fewer variables and parameters, simpler computational processes, similar performance, high predictive accuracy, and faster computation speed. The GNN and GRU methods were hybridized using Multiple Linear Regression (MLR), with normal temperature as the response variable. This hybridization provided a statistical basis for determining the input variable structure of both methods, thereby improving the forecasting accuracy of the proposed hybrid methods, MLR-GNN and MLR-GRU, which are structured according to sound scientific principles. The results of this study demonstrate that MLR-GNN achieved more accurate forecasting results compared to MLR-GRU, and both models outperformed the traditional MLR model. The results also indicate that the nature of the data, in terms of low variability and stable values, suggests that the improvement in predictive results when using MLR-GRU hybridization was slight. From this, it can be concluded that the geographical proximity of cities, which ensures rapid air mass movement, reinforces the importance of the spatial dimension in climate relationships, in addition to the temporal dimension, unless there are problems with variability and stability. This is because all cities are affected by a uniform climate pattern due to their small size and proximity, thus allowing for an absolute advantage in using MLR-GNN hybridization.

Keywords Graph Neural Networks (GNN), Gated Recurrent Unit (GRU), Climatic Time Series, Spatio-Temporal Forecasting, Hybrid MLR-GNN, Hybrid MLR-GRU

DOI:10.19139/soic-2310-5070-4550

1. Introduction

Climatic time series forecasting is depended often to understand the environmental changes and resources management. However, the climate time series forecasts by using traditional forecasting approach may suffer from inaccuracy results, which can negatively impact the effects of forecasting on human life especially when the data has both temporal and spatial dimensions. Through dealing with climate data in many studies, it has been proven that it contains many complex non-linear patterns resulting from the problem of data heterogeneity due to differences within the temporal (seasonal) and spatial dimensions, especially multi-site climate data, while several traditional methods are unable to capture these problems which may certainly lead to a decrease in the

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

accuracy of the forecast. GNN will be used in this study as a type of neural network that specializes in processing spatio-temporal data by relying on a graphical structure consisting of nodes and their connected edges. It also relies on the property of passing messages continuously and repeatedly between the connected nodes. Therefore, each node will collect much more information than usual from its neighbors to produce output variables that represent predictions for the spatio-temporal model. This type of network is scalable by applying the same models with the same or different weights without needing to worry about the order of the nodes. The network will update the representation of each node after collecting information from the neighboring nodes in order to represent the spatial relationship structure of the data. Information is exchanged between the neighboring nodes with repetition until the network reaches stability and achieves what was assumed in the normalization process by extracting the representations and complex relationships through several layers to ensure the transfer of information from the neighbors directly and more smoothly, and to ensure that the representation of each node is improved more accurately so that the final network model can be evaluated [1]. The Gate Recurrent Unit (GRU), which specializes in predicting time series and studying time interval variation, was also used. It is often suggested as an alternative to Long Short Term Memory Networks (LSTMs) by reducing the number of gates, thus ensuring the fewest possible variables and parameters that actually affect the results. It also ensures greater simplicity in computational operations, with similar performance and high predictive accuracy, along with higher computational speed. The GRU architecture was designed with only two gates: a reset gate and an update gate. This allows for learning long-term relationships with high efficiency and without going through a memory cell, thereby improving the speed and efficiency of training neural networks [2]. To improve the accuracy of predictions, the Multiple Linear Regression (MLR) model was used as a statistical model to represent the mathematical relationship between the influence of several explanatory variables on a single response variable. MLR was used for prediction by hybridizing it with the GNN and GRU models to provide a statistical basis for structuring the input variables of both networks. In this study, normal temperature was considered the response variable, dependent on the influence of several climatic variables such as solar radiation, maximum temperature, minimum temperature, relative humidity, and wind speed. Liu et al. [3] proposed the GR-GNN hybrid model, combining GNN and GRU networks, to improve the prediction accuracy of three datasets in the field of health and early warning of epidemics across different geographic regions. The model's multi-step prediction capabilities were observed to be significantly superior to other traditional prediction methods. Kazadi et al. [4] also proposed a spatio temporal model capable of predicting floods and simulating water spreads. This model, GNN-GRU, hybridizes GNN for the spatial component with GRU for different time sequences and intervals. The proposed model demonstrated remarkable accuracy, surpassing other traditional time series prediction models, particularly in simulating physical systems and processing water flow velocity vectors, especially in flood scenarios. Yang et al. [5] studied the prediction of baseball throwing speeds based on the biomechanical and kinematic principles of joints. They designed a hybrid model by combining GNNs for spatial relationships between joints and GRUs for modelling the temporal sequence of the throwing motion. The proposed model achieved high accuracy in predicting throwing speed. In general, the studies mentioned above, along with many previous studies, represent a modern and important trend in utilizing GNN and GRU hybrid networks to build spatio-temporal models that outperform traditional models, particularly with time series data, across the various application aspects covered by these studies. The results of this study can be summarized that MLR-GNN achieved more accurate forecasting results and outperformed the results of MLR-GRU, while both models outperformed the traditional MLR model. The results also indicate that the nature of the data, in terms of low variability and stable values, suggests that the improvement in predictive results when using MLR-GRU hybridization was slight. Wang [6] This research presents a modern approach to multivariate time series mining by hybridizing Graph Neural Networks (GNNs) with the self-attention mechanisms of the Transformer model. The researcher addresses the challenge of simultaneously modeling spatial and temporal dependencies by introducing an adaptive adjacency matrix that enables the automatic discovery of dynamic, local relationships between variables, combined with the Transformer's superior ability to predict long-range temporal dependencies. The model demonstrates superior efficiency, strong generalization capabilities, and robustness compared to traditional statistical models. Temür [7] makes a scientific contribution to the field of forecasting carbon emissions resulting from fossil-fuel-based electricity generation in Turkey. It employs a hybrid model that integrates Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRU) with linear regression

analysis to link energy generation to emissions. The methodology incorporates feature engineering—specifically through the use of lagged variables—to enhance the model’s predictive capability. Results demonstrate that the integrated hybrid architecture outperforms individual models across various evaluation metrics, confirming its efficiency in accurately capturing temporal patterns as well as linear and non-linear relationships. The originality of this study lies not in proposing a novel neural architecture, but in introducing a two-stage hybrid mechanism that employs Multiple Linear Regression (MLR) both as a statistical prior to structure the inputs for GNN and GRU models and as a linear component in the final prediction process. Essentially, the study bridges interpretable statistical regression with deep learning for spatiotemporal data, while validating the model in a climatically transitional region (Nineveh Governorate) that serves as a boundary between Mediterranean and arid climates.

2. Methods and Materials

2.1. Graph Neural Networks (GNN)

Graph Neural Networks (GNNs) are an important type of neural network specializing in processing spatiotemporal data. They rely on a graph structure composed of nodes and their connected edges. This specialization in dealing with graphs distinguishes them from other types of networks used for images or text, as they learn through spatiotemporal relationships by continuously and repeatedly passing messages between connected nodes, known as edges. Therefore, each node gathers significantly more information than usual from its surroundings, building information-rich models of individual variables and their structural characteristics in spatiotemporal patterns. The general structure of a graph neural network typically consists of inputs or variables, which are represented by graphs containing nodes (entities) or spatial labels, and edges that represent relationships. The network then produces outputs, which are predictions generated from a spatiotemporal model using the artificial neural network. Therefore, the main components of any graph can be summarized as follows: nodes or vertices, which represent individual entities (places) within the neural network; edges, which represent the connections and relationships between nodes; node attributes or node variables, which express the characteristics of the nodes; and finally, edge attributes or variables, which often express the characteristics of the edges, connections, and relationships that link the nodes together. Graphical neural networks differ from other traditional artificial neural networks in that those networks assume each input variable is independent of the others. Graphical neural networks address this problem and assume relationships between variables within a single entity and between similar variables in different entities through message passing and information flow across the connections between entities and nodes. This leads to improved prediction accuracy during both training and testing periods. Furthermore, these networks are scalable by applying the same models, with the same or different weights, to all nodes and edges, regardless of their configuration, without needing to worry about the order of the nodes or swapping them [8, 9]. Gori et al. [10] developed the concept of graph neural networks in deep learning to process large datasets of a graphical nature by finding connections between nodes based on neighbor information, i.e., spatial connections. Initially, graph neural networks suffered from poor prediction accuracy due to a lack of structural integrity. Improvements were made by developing concepts regarding the graphs used as inputs and the models that the network uses to achieve optimal deep learning. In 2016, Kipf and Welling [11] introduced a semi-supervised classification method called the convolutional neural network. This method involves wrapping image filters around all graph points, enabling the network to read all node points and their various edge connections. This improves predictions and classifications, enhancing the deep learning capabilities of the graph neural network. In 2017, Hamilton et al. [12] at Stanford University proposed an alternative approach to address the scalability problem faced by graph neural networks. Instead of storing the entire graph, the new method, called GraphSAGE, relies on neighbor-based learning by sampling neighbors of similar size. This improves the readability of large graphs and enhances the accuracy of prediction outputs. In 2017, Veličković et al. [13] proposed Graph Attention Networks (GATs), which learn only from the closest or most important neighbors. This improves predictive performance by focusing on significant relationships rather than assigning equal importance to all connections between nodes. The message passing feature, which is often relied upon by graph neural networks as a fundamental principle, involves

several sequential steps and stages to provide a stable framework for the network, as follows: 1. Initialization of models and relationships, where each node in the graph begins with an initial vector representing the spatial and temporal characteristics of the data. 2. Generation of messages for each edge connecting two nodes in the graph. Information for these messages is obtained from the model of the current source node, the current model of the next node (the access node), and the characteristics of the edges and connections between the two nodes. 3. Each node collects messages from its neighbors using a specific collection function based on several criteria, such as the average of all incoming messages, the sum of all messages, the maximum value of each variable dimension, or by assigning weight or importance to neighbors and prioritizing the most important ones. 4. The node is updated after integrating the information obtained from neighbors, which is compiled with the node's existing model, to produce an updated model. This ensures the neural network is trained, that the activation function is efficient and non-linear, and that connections and edge links from previous layers are skipped or ignored. 5. The previous steps are repeated k times, with multiple message-passing rounds to obtain more and more diverse information from neighbors until the k -th iteration is reached, as illustrated in Figure 1 below [9].

In this study, the use of the GNN model is based on the idea of collecting information from neighboring nodes

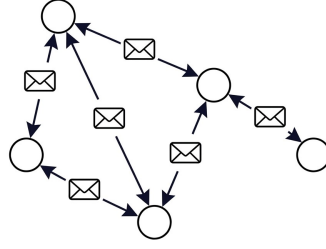


Figure 1. Message passing concept for GNN.

using the Normalized Adjacency Matrix. In it, the representation of each node in the network is updated through the weighted average of the representations of its neighbors. That is, the formula used is what can be called the Symmetric Normalization formula, which ensures renormalization after multiplying the adjacency matrix from both sides using the inverse square root of the degree matrix. This ensures the achievement of balance in representing relationships between nodes to confirm that high-degree nodes and a large number of neighbors do not dominate when collecting information, as in the equation 1 below when $\tilde{A}$ can represent the normalized adjacency matrix.

$$\tilde{A} = \hat{D}^{-\frac{1}{2}} \hat{A} \hat{D}^{-\frac{1}{2}} \quad (1)$$

When the adjacency matrix A is based on learning the model from the data during training and a_{ij} represents the element of adjacency matrix, i.e., by building on the proximity and distance of the spatial relationships between the adjacency nodes, such as follows.

$$A = \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & a_{2,3} & \cdots & a_{2,n} \\ a_{3,1} & a_{3,2} & a_{3,3} & \cdots & a_{3,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & a_{n,3} & \cdots & \end{bmatrix} \quad (2)$$

and

$$a_{ij} = \begin{cases} 1, & \text{if there is an edge between node } i \text{ and node } j \\ 0, & \text{otherwise} \end{cases} \quad (3)$$

while $\hat{A} = A + I_N$ is the matrix which represents the adjacent nodes after adding the autoloops, and the Degree Matrix can be represented as a diagonal matrix as follows.

$$\hat{D} = \begin{pmatrix} d_{1,1} & 0 & 0 & \cdots & 0 \\ 0 & d_{2,2} & 0 & \cdots & 0 \\ 0 & 0 & d_{3,3} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \end{pmatrix} \quad (4)$$

As mentioned previously, the GNN will update the representation of each node after collecting information from neighboring nodes to represent the spatial relationship structure of the data. The GNN will also use message passing to exchange information between neighboring nodes, with iterations occurring until the network stabilizes and meets the normalization assumptions. The figure 2 below illustrates the normalization mechanism, which is a graph representing the data in the target variable. In this graph, the nodes represent the geospatial locations of the edges, thus representing the relationships between the nodes [14]. From Figure 2, the vector of node A must

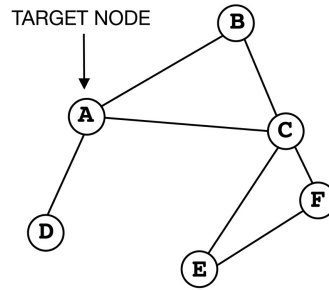


Figure 2. The target node and its neighbors in GNN graph.

include components of its neighbors B, C, and D, regardless of the order of the neighbor nodes, by calculating the average of the neighbor nodes' characteristics. The stages of multi-layer message passing of GNN in detail, and their mechanism of operation after defining the target variable in the figure above, can be represented in Figure 3 below.

The GNN's mechanism of operation, after identifying the target variable, involves collecting information from

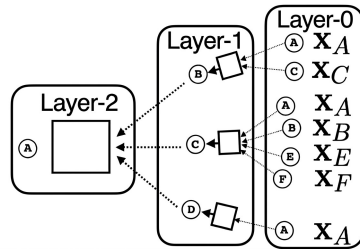


Figure 3. Multi-layer message passing in GNN.

neighboring nodes and then updating the representation for each node by multiplying its current or initial representation by weights. After that, a transformation or activation function σ will be applied, which will be determined using ReLU and Softmax. This allows the model to extract complex representations and relationships.

These operations continue to be repeated across several layers to ensure that information is transferred from neighbors directly and more smoothly, which ensures that the representation for each node is improved more accurately. The final network model is then evaluated through estimated and predictive values after relying on accurate representations of the spatial relationships between the nodes. Where Z_l represents a matrix to represent the nodes in the layer l , W_l represent the weight matrix for layer l , σ is the activation function, and $\hat{D}$ is the degree matrix then the final model of GNN, from which layer $l + 1$ outputs are resulted, can be represented by the following equation.

$$Z_{l+1} = \sigma \left(\hat{D}^{-\frac{1}{2}} \hat{A} \hat{D}^{-\frac{1}{2}} Z_l W_l \right) \quad (5)$$

From this, continuous observations will be obtained after applying the regression and prediction tasks by using a linear activation function or the corrected linear unit ReLU in the hidden layers, which gives the function zero for input variables x when they are negative, while positive values are passed. In general, the SoftMax function may be used for some cases in the output layer when e^{z_i} represents the exponential function of the target variable, as shown in the equations below, respectively [15].

$$ReLU(x) = \max(0, x) \quad (6)$$

$$\text{softmax}(z)_i = \frac{\exp(z_i)}{\sum_{j=1}^k \exp(z_j)} \quad (7)$$

2.2. Gated Recurrent Unit (GRU)

Iterative neural networks are a popular type of neural network algorithm used in many studies for predicting complex, nonlinear variables with a multivariable structure consisting of one target variable and several input variables. As reported in numerous studies, it is also possible to predict a single variable using autoregression. In a traditional iterative neural network, the weights and neurons of the hidden layers are often updated at each time point and in each iteration by combining the current weighted inputs with the states in the hidden layers of the previous iteration. This allows the network to transfer information over time or across successive observations. In practice, this results in a gradual fading of values towards zero or a gradual inflation of values. This prevents the network from linking observations or connecting widely separated time sequences, negatively impacting the prediction and its accuracy. The primary reason for the inability to link widely separated events may be the exponential growth of the gradient components of long-term gradients, which grow much faster than the corresponding norm for short-term gradient components, particularly during model training. The equation for the hidden state in the recurrent neural network (RNN) will be as follows [16].

$$h_t = f(Ux_t + Wh_{t-1} + b) \quad (8)$$

when

$U \in \mathbb{R}^{n \times m}$, $W \in \mathbb{R}^{n \times n}$, $h_t \in \mathbb{R}^{n \times 1}$, $x_t \in \mathbb{R}^{m \times 1}$, x_t represents the input vector at time index t , $\tanh$ represents the hyperbolic tangent activation function, h_t represents the hidden state at the current time t , and b is a bias vector ($n * 1$).

Additionally, the model parameters resulting from learning and training consist of two: first is the weight matrix W , which relates the previous hidden state to the current state, and second is U matrix, which relates the input variable to the hidden state. Figure 4 describes RNN such as explained in its equation.

where V in the figure 4 represents the weights matrix for linking the hidden state to the output variable, and the output variable equation with bias can be written as follows.

$$y_t = Vh_t + b_y \quad (9)$$

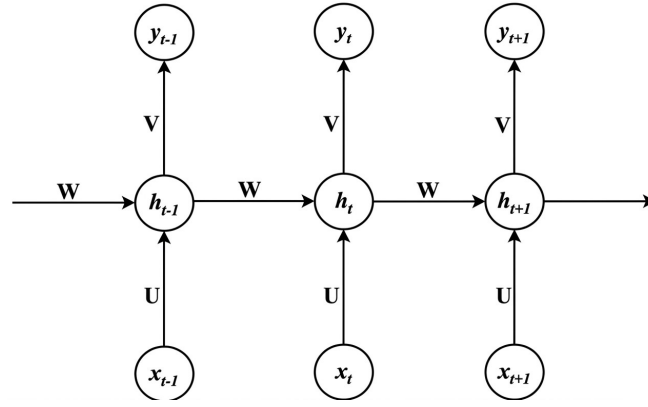


Figure 4. RNN mechanism.

To address the problem of long-term and short-term memory, the use of Long-Term Memory Networks (LSTMs) has been proposed in numerous studies. These networks possess the ability to retain important information over extended periods and are specifically designed to handle data with interconnected observations that vary over time. This is achieved through the use of a memory cell consisting of three gates: an input gate, a forget gate, and an output gate. This allows for reasonable prediction accuracy, but at the cost of greater computational complexity and a relatively large number of parameters.

In this model, c_t represents the memory cell state vector at the current time t , $\tilde{c}_t$ represents the candidate cell state. f_t and i_t represent, respectively, the forget gate, which determines how much of the previous information is retained, and the input gate, which determines how much of the new information entering the memory state is retained. Additionally, o_t represents the output gate, which determines how much information is passed from the memory state to the hidden state. Finally, $\odot$ represents the dot product, element by element. Therefore the memory state will be updated based on the following equation [16, 17].

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \quad (10)$$

The state of the candidate memory is calculated based on the following equation.

$$\tilde{c}_t = \tanh(U_c h_{t-1} + W_c x_t + b_c) \quad (11)$$

However, the final hidden state, when o_t represents the output gate signal, is calculated based on the following equation.

$$h_t = o_t \odot \tanh(c_t) \quad (12)$$

The three network gates can be calculated respectively as follows.

$$i_t = \sigma(U_i h_{t-1} + W_i x_t + b_i) \quad (13)$$

$$f_t = \sigma(U_f h_{t-1} + W_f x_t + b_f) \quad (14)$$

$$o_t = \sigma(U_o h_{t-1} + W_o x_t + b_o) \quad (15)$$

Such as described by figure 5.

To address the complexity of calculations and the large number of parameters and variables involved in the model, networks similar to Long Short Term Memory Networks (LSTMs) were proposed, but with a simplified structure consisting of only two gate layers: a reset gate (Y) and an update gate (Z). These networks, called Gate

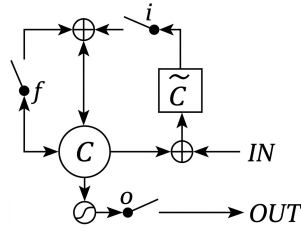


Figure 5. LSTM mechanism.

Recurrent Unit Networks (GRUs), are derived from the scientific basis of Recurrent Neural Networks (RNNs) and specialize in data with related observations, particularly time series data. This type of network was proposed as an alternative to LSTMs, ensuring fewer variables and parameters that actually influence the learning process, greater simplification of calculations, similar performance, high predictive accuracy, and higher computational speed. The GRU structure is designed with two gates—a reset gate and an update gate—to learn long-term relationships efficiently and without relying on a separate memory cell, as is the case in LSTMs. This allows for the ignoring of minor and irrelevant variables and details during the training and learning process, thus improving the speed and efficiency of neural network training.

GRU mechanism is described by figure 6.

The following equations illustrate the mathematical description of the GRU gated recurrent unit network. Note

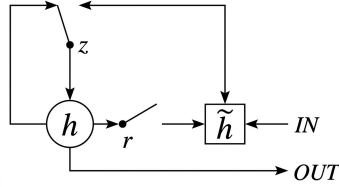


Figure 6. GRU mechanism.

that the input gate and forget gate in the LSTM network are abbreviated to the reset gate in the GRU network, and in the mathematical description, $1 - z_t$ and z_t in the equations represent the input gate and forget gate, respectively [17].

$$r_t = \sigma(U_r h_{t-1} + W_r x_t + b_r) \quad (16)$$

$$z_t = \sigma(U_z h_{t-1} + W_z x_t + b_z) \quad (17)$$

$$\tilde{h}_t = \tanh(U_h(r_t \odot h_{t-1}) + W_h x_t + b_h) \quad (18)$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t \quad (19)$$

The first equation represents the Reset Gate, the second represents the Update Gate, the third represents the Candidate Hidden State, and the fourth represents the Final Hidden State when r_t represents the Reset Gate, z_t represents the Update Gate, $\tilde{h}_t$ represents the Candidate Hidden State, and (U_r, U_z, U_h) represent the weight matrices of the hidden state [3, 15].

2.3. Multiple linear regression (MLR)

Multiple linear regression (MLR) can be defined as a statistical model that represents the mathematical relationship between the influence of several explanatory variables on a single response (objective) variable. In other words, it is used to predict the response variable influenced by the explanatory variables [18]. This research will use MLR to determine the structure of parameter-weighted input variables that influence the objective variable and are incorporated into both the GNN and GRU methods. The general model of MLR can be formulated as follows.

$$y_i = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p + e_i \quad (20)$$

where y_i is the response (target) variable, $x_1, x_2, \dots, x_p$ are the explanatory variables, β_0 is the intercept with y axis, $\beta_1, \beta_2, \dots, \beta_p$ are the regression parameters, and e_i is the random error [19, 20, 21, 22, 23].

2.4. Hybridization Mechanism

The hybridization process proposed in this study can be summarized in two sequential stages. The first stage involves hybridization at the representation level, entailing the estimation of coefficients for the explanatory variables of a Multiple Linear Regression (MLR) model for each city individually; these estimated coefficients are then used to define the input representation structure for the GNN and GRU networks [24]. In this stage, each explanatory variable is multiplied by its corresponding estimated coefficient to form a weighted input vector that preserves the statistical sign and the relative importance of each variable's contribution.

The second stage involves training the GNN and GRU networks using these structured inputs to generate the final forecasting.

Consequently, the proposed framework does not represent a conventional ensemble learning method; rather, it is a two-stage hybrid model in which the MLR model plays a pivotal role by providing statistical prior information for input structuring. It should be noted that Equations (25) through (30) pertain solely to the second stage.

The proposed hybrid model frameworks (MLR-GNN) and (MLR-GRU) share some conceptual overlap with ensemble learning or feature selection techniques. Methodologically, the Multiple Linear Regression (MLR) component in this study serves as a bridge between interpretable statistical modeling and spatiotemporal deep learning. The key differences lie in several fundamental aspects such as followed.

1. The proposed MLR framework is not employed as a parallel prediction tool—as is the case with traditional ensemble learning methods—but rather as a preliminary statistical input structure that defines the representation and weights of explanatory variables before they are fed into the GNN and GRU models.
2. While feature selection methods are typically used to identify and rank a subset of variables based on importance, this study utilizes MLR to represent the complete set of variables by incorporating the weight and sign (positive/negative) of each coefficient into the input structure. This enables the neural networks to learn nonlinear and spatiotemporal corrections using the MLR model as a baseline reference.
3. The GNN model relies on a spatial message-passing mechanism between weather stations via a normalized adjacency matrix—a feature that cannot be captured by traditional ensemble learning methods. Furthermore, the GRU model introduces a "temporal gating" mechanism across sequential windows; this differs from the structural architecture of traditional ensemble learning methods and constitutes the core methodological contribution of this work.

2.5. Root Mean Squared Error (RMSE)

RMSE is a One of error measurements which used to evaluate adequacy of forecasting when dealing with quantitative data. RMSE can be computed such as follows [13].

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^N (e_i)^2} \quad (21)$$

Where, n is the number of observations, the forecasting error $e_i = (y_i - \hat{y}_i)$, y_i is the actual value, and $\hat{y}_i$ is the predicted value.

2.6. Improvement Measurement

Arithmetically, measuring the improvement percentage can be calculated for two compared RMSE values such as follows.

$$Improvement = \frac{RMSE_{method1} - RMSE_{method2}}{RMSE_{method1}} \times 100\% \quad (22)$$

3. Results and discussion

3.1. Data Description and The Framework of Study

In this study, multivariate climate time series data were collected from meteorological stations in Nineveh Governorate in Iraq for daily observations from the beginning of 1994 to the end of 2024, totalling 11,329 daily observations. These were divided into training data from the beginning until the end of 2023, amounting to 10,963 observations, and test data for 2024, amounting to 366 observations. The data were obtained from agricultural meteorological stations in six administrative areas of Nineveh Governorate, namely Mosul, Talafar, Sinjar, Al-Baaj, Tal-Abtah, and Makhmore. The data included several climatic factors, five of which were assumed to be influential explanatory variables: solar radiation, maximum temperature, minimum temperature, relative humidity, and wind speed. The response variable was assumed to be normal temperature. The dataset was obtained from a doctoral thesis in geography at the University of Mosul and underwent preprocessing and quality control using remote sensing tools. No additional transformations to achieve stationarity were applied, as the proposed GNN and GRU models rely on spatiotemporal variation as a fundamental learning signal; removing seasonality or applying differencing operations would eliminate the spatiotemporal diversity that these networks were designed to capture and extract. The dataset used in this study was obtained from a specialized source (a doctoral dissertation in physical geography at the University of Mosul), where it had already undergone geographic preprocessing. No transformations to achieve stationarity were applied, as the proposed GNN and GRU models are designed to learn from spatiotemporal variability rather than from components stripped of trends or seasonal effects. Applying differencing or seasonal adjustment would result in the loss of the very diversity these networks aim to capture; consequently, such measures would prove counterproductive for this type of model architecture [25]. The framework of this study includes each of the following.

1. Identifying the response (target) and input variables

where: The target variable is normal temperatures (the arithmetic mean of maximum and minimum temperatures), and the input variables are the climatic variables: solar radiation, maximum temperature, minimum temperature, relative humidity, and wind speed.

2. Data modelling using MLR to determine the most suitable model which uses as a baseline for comparisons.

3. Determining the inputs structure of the hybrid models.

4. Hybrid model forecasting, which branches into:

a. The MLR-GNN hybrid model, by combining GNN and MLR for spatio-temporal relationships.

b. The MLR-GRU hybrid model, by combining GRU and MLR to address range variations and complex patterns.

5. Performance comparison for the performance of the traditional MLR and both hybrid models MLR-GNN and MLR-GRU. Figure 7 illustrates the general framework of this study.

This study is subject to several limitations that should be noted prior to the analysis: the evaluation relies on a single temporal split of the data into training and testing sets. Furthermore, the adjacency matrix is based on a fixed distance threshold (100 km), whereas constructing the graph based on correlation or machine learning could allow for more flexible spatial relationships. The study also focuses on predicting specific values (point forecasting) and does not provide probabilistic intervals to quantify uncertainty; additionally, its evaluation is limited to temperature

as the target variable, leaving scope for future research to examine other climatic variables.

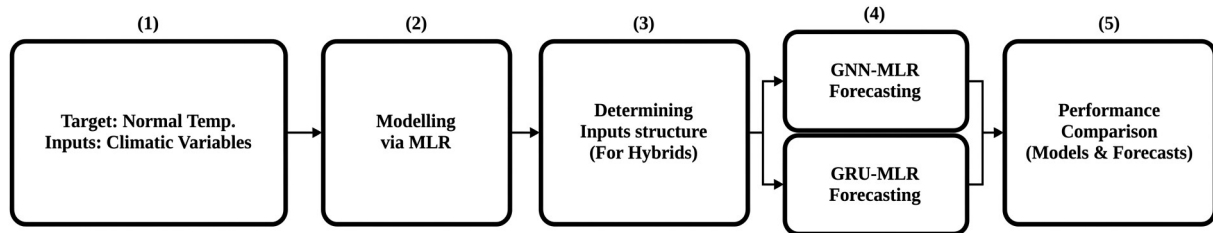


Figure 7. The general framework of the study.

3.2. MLR-GNN Hybrid Model

In general, after modelling using Multiple Linear Regression (MLR) for all the six cities, it was found that all of them contained significant parameters for all their variables, as determined by their p-values. Table 1 shows the results of the MLR modelling (parameter values, significance, R^2 , RMSE) for all cities.

Table 1. MLR modelling results (parameter values, significance, R^2 , RMSE)

	β_0	β_1	β_2	β_3	β_4	β_5	R^2	RMSE
Mosul	-0.02032 **	0.03615 **	0.52740 **	0.45624 **	0.00717 **	-0.00037	99.62%	0.0458
Talafar	0.01985 **	0.01775 **	0.63304 **	0.35878 **	0.00833 **	-0.00034	99.49%	0.0692
Sinjar	0.02150 **	0.01644 **	0.67070 **	0.32467 **	0.01768 **	-0.00029	99.26%	0.0654
Al_Baaj	-0.01559 **	0.03770 **	0.52636 **	0.45775 **	0.00317 **	-0.00345 **	99.51%	0.0433
Tal_Abtah	0.02604 **	0.02656 **	0.65415 **	0.33876 **	0.01352 **	-0.00537 **	99.33%	0.0581
Makhmore	-0.02227 **	0.04106 **	0.51814 **	0.46378 **	0.00211 **	-0.00171 **	99.55%	0.0441

where ** means that this parameter is strongly significant.

where x_1 is the Solar Radiation, x_2 is the Maximum Temperature, x_3 is the Minimum Temperature, x_4 is the Relative Humidity, and x_5 is the Wind Speed.

From Table 1, it was observed that all p-values were significant (p-value ≤ 0.05) except for the parameters of wind speed variable for the cities (Mosul, Talafar, and Sinjar), which were insignificant, while the parameters for the same variable for the cities (Al-Baaj, Tal Abtah, and Makhmore) were significant. Therefore, for the most, all the parameters were significant except for the parameters of the wind speed variable, which were significant at most. Therefore, they were adopted and included in the model, even though the parameters for three cities are not significant. This can be justified because the values of their parameters are essentially very small (less than 0.004).

The values of those parameters will be multiplying by the observations of wind speed variable for those cities, they will have values very close to zero. Therefore, they will not have an effect or importance on the response variable (normal temperature) in the models for this three cities (Mosul, Talafar, Sinjar).

The high coefficients of determination (R^2) approximately 99.8 are attributed to the strong linear relationship between the normal temperature and its explanatory variables (particularly maximum and minimum temperatures), which collectively account for the majority of the data variance. This aligns with the results of the multiple linear regression (MLR) baseline model, which achieves an R^2 of approximately 99.5; hybrid neural network-based models improve upon this baseline by capturing residual non-linear and spatio-temporal patterns.

For applying GNN, the binary adjacency matrix should be structured to represent the spatial connections (The distance in km) between the climatic stations such as follows.

$$A = \begin{array}{c|cccccc} & \text{Mosul} & \text{Talafar} & \text{Sinjar} & \text{Al-Baaj} & \text{Tal-Abtah} & \text{Makhmore} \\ \hline \text{Mosul} & 1 & 1 & 0 & 0 & 0 & 1 \\ \text{Talafar} & 1 & 1 & 1 & 0 & 0 & 0 \\ \text{Sinjar} & 0 & 1 & 1 & 1 & 0 & 0 \\ \text{Al-Baaj} & 0 & 0 & 1 & 1 & 1 & 0 \\ \text{Tal-Abtah} & 0 & 0 & 0 & 1 & 1 & 0 \\ \text{Makhmore} & 1 & 0 & 0 & 0 & 0 & 1 \end{array} \quad (23)$$

where 1 represents a connection or proximity, while a 0 means no connection. Additionally, each city is self-connected to maintain its own information. The adjacency matrix was constructed based on the spatial distances between the stations. 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.

$$a_{ij} = \begin{cases} 1, & \text{if } Dist_{ij} < 100 \text{ Km} \\ 0, & \text{if } Dist_{ij} \geq 100 \text{ Km} \end{cases} \quad (24)$$

Each node also connects to itself (a self-loop) to preserve its own information during message passing. Subsequently, the adjacency matrix is symmetrically normalized; this process ensures balanced information flow and numerical stability. After the normalization of the adjacency matrix, the normalized adjacency matrix will be as follows.

$$A_{\text{norm}} = \begin{array}{c|cccccc} & \text{Mosul} & \text{Talafar} & \text{Sinjar} & \text{Al-Baaj} & \text{Tal-Abtah} & \text{Makhmore} \\ \hline \text{Mosul} & 0.333 & 0.333 & 0.000 & 0.000 & 0.000 & 0.408 \\ \text{Talafar} & 0.333 & 0.333 & 0.333 & 0.000 & 0.000 & 0.000 \\ \text{Sinjar} & 0.000 & 0.333 & 0.333 & 0.333 & 0.000 & 0.000 \\ \text{Al-Baaj} & 0.000 & 0.000 & 0.333 & 0.333 & 0.408 & 0.000 \\ \text{Tal-Abtah} & 0.000 & 0.000 & 0.000 & 0.408 & 0.500 & 0.000 \\ \text{Makhmore} & 0.408 & 0.000 & 0.000 & 0.000 & 0.000 & 0.500 \end{array} \quad (25)$$

Thus, the central cities are Mosul, Tal Afar, Sinjar, and Al-Baaj, each with three connections (one self-connection in addition to two neighbor connections), while the peripheral cities are Tal Abtah and Makhmore, each with two connections (one self-connection in addition one neighbor connection). It is evident that the unnormalized view focuses on Mosul receiving information from three neighbors, while Tal Abtah receives connections from only two. With normalization, all cities receive balanced information. It should be noted that the more connected central cities do not necessarily dominate the less connected peripheral cities. Numerical stability is also observed, ensuring no gradient rises or falls.

After applying GNN by assuming that the inputs structure of GNN is as MLR results by multiplying

(Parameters $\times$ variables) point to point. Table 2 shows the results of the hybrid MLR-GNN method for all cities divided into training and testing stages by using RMSE and improvement measurements.

Table 2. The results of the hybrid MLR-GNN method for all cities for training and testing RMSE.

	Mosul	Talafar	Sinjar	Al.Baaj	Tal_Abtah	Makhmore
Train	0.02123	0.02909	0.03470	0.02603	0.03160	0.02668
Test	0.02826	0.04773	0.04882	0.03371	0.04714	0.02938
Improvement	53.65%	57.96%	46.94%	39.88%	45.61%	39.50%
R²	99.82%	99.73%	99.74%	99.85%	99.72%	99.83%

Table 2 shows that the forecasting performance results of the MLR-GNN hybrid model for the training and testing periods are outperformed the results of the traditional MLR model such as in Table 1. R² values in both tables are calculated from the training data and it can be proofed that both MLR and MLR-GNN are perfect in climatic data modelling. and Improvement values in Table 2 are also calculated from the training data and can be showed that MLR-GNN has great performance compared to MLR. It is possible to express the MLR-GNN hybrid models in mathematical equations for each city individually as mentioned in the equations below.

$$\text{Mosul: } \hat{Y}_{\text{hybrid}} = 0.8003 \hat{Y}_{\text{MLR}} + 0.1992 \hat{Y}_{\text{GNN}} \quad (26)$$

$$\text{Talafar: } \hat{Y}_{\text{hybrid}} = 0.8202 \hat{Y}_{\text{MLR}} + 0.1840 \hat{Y}_{\text{GNN}} \quad (27)$$

$$\text{Sinjar: } \hat{Y}_{\text{hybrid}} = 0.7498 \hat{Y}_{\text{MLR}} + 0.2511 \hat{Y}_{\text{GNN}} \quad (28)$$

$$\text{Al-Baaj: } \hat{Y}_{\text{hybrid}} = 0.9671 \hat{Y}_{\text{MLR}} + 0.0326 \hat{Y}_{\text{GNN}} \quad (29)$$

$$\text{Tal Abtah: } \hat{Y}_{\text{hybrid}} = 0.9824 \hat{Y}_{\text{MLR}} + 0.0189 \hat{Y}_{\text{GNN}} \quad (30)$$

$$\text{Makhmore: } \hat{Y}_{\text{hybrid}} = 0.9991 \hat{Y}_{\text{MLR}} + 0.0009 \hat{Y}_{\text{GNN}} \quad (31)$$

Figure 8 shows the comparison between the expected values of the MLR-GNN model and the actual values of normal temperatures in six cities of Nineveh Governorate during the period from January to May 2024. The figure of comparing the expected values of the MLR-GNN model with the actual values of normal temperatures confirms that the forecasted performance results of the MLR-GNN hybrid model for the first 5 months of the test periods have the perfect performance, These forecasting results reflect the maximum forecasting quality of the MLR-GNN hybrid model.

3.3. MLR-GRU Hybrid Model

After modelling using MLR such was shown in Table 1 above, GRU will be applied based on the inputs structure which has been obtained from MLR. After applying MLR-GRU by depending on the same inputs structure of MLR which was used previously with MLR-GNN in addition to determine some of requirements such as:

The size of hidden layer and batch size= 32, the size of window= 7 (7 days), the maximum epochs= 100, the initial learning rate= 0.001, and the reduce factor of learning rate (per 20 times)= 0.5.

MLR-GRU hybrid model can be expressed in mathematical equations for each city individually as mentioned in the equations below.

$$\text{Mosul: } \hat{Y}_{\text{hybrid}} = 0.5032 \cdot \hat{Y}_{\text{MLR}} + 0.4968 \cdot \hat{Y}_{\text{GRU}} \quad (32)$$

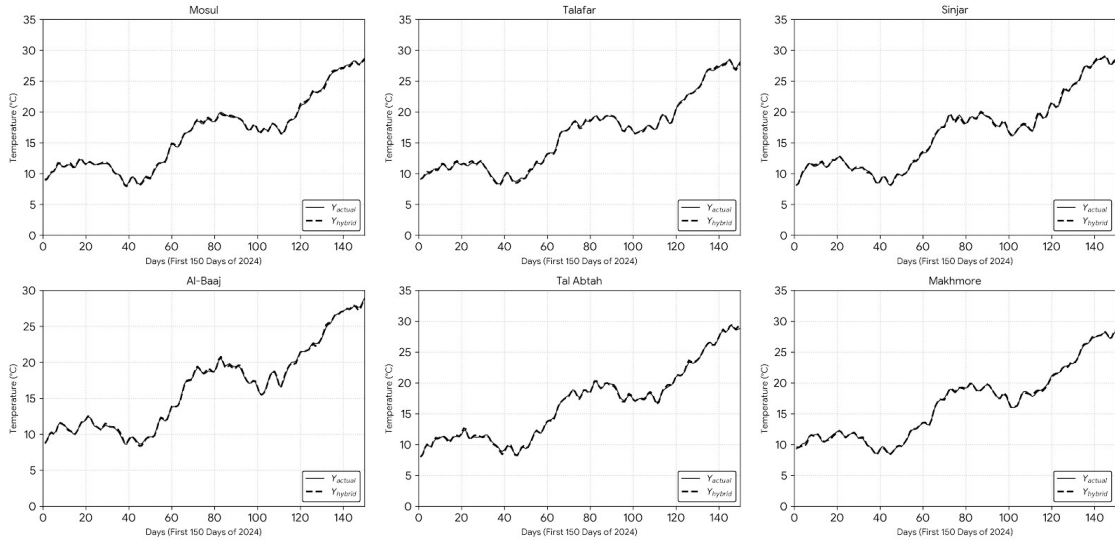


Figure 8. The actual and the MLR-GNN testing forecasting of normal temperature (from January to May 2024) for six cities.

$$(\alpha_{\text{MLR}} = 0.5032, \quad \alpha_{\text{GRU}} = 0.4968)$$

$$\textbf{Talafar: } \hat{Y}_{\text{hybrid}} = 0.5036 \cdot \hat{Y}_{\text{MLR}} + 0.4964 \cdot \hat{Y}_{\text{GRU}} \quad (33)$$

$$(\alpha_{\text{MLR}} = 0.5036, \quad \alpha_{\text{GRU}} = 0.4964)$$

$$\textbf{Sinjar: } \hat{Y}_{\text{hybrid}} = 0.5040 \cdot \hat{Y}_{\text{MLR}} + 0.4960 \cdot \hat{Y}_{\text{GRU}} \quad (34)$$

$$(\alpha_{\text{MLR}} = 0.5040, \quad \alpha_{\text{GRU}} = 0.4960)$$

$$\textbf{Al-Baaj: } \hat{Y}_{\text{hybrid}} = 0.5038 \cdot \hat{Y}_{\text{MLR}} + 0.4962 \cdot \hat{Y}_{\text{GRU}} \quad (35)$$

$$(\alpha_{\text{MLR}} = 0.5038, \quad \alpha_{\text{GRU}} = 0.4962)$$

$$\textbf{Tal Abtah: } \hat{Y}_{\text{hybrid}} = 0.5040 \cdot \hat{Y}_{\text{MLR}} + 0.4960 \cdot \hat{Y}_{\text{GRU}} \quad (36)$$

$$(\alpha_{\text{MLR}} = 0.5040, \quad \alpha_{\text{GRU}} = 0.4960)$$

$$\textbf{Makhmore: } \hat{Y}_{\text{hybrid}} = 0.5033 \cdot \hat{Y}_{\text{MLR}} + 0.4967 \cdot \hat{Y}_{\text{GRU}} \quad (37)$$

$$(\alpha_{\text{MLR}} = 0.5033, \quad \alpha_{\text{GRU}} = 0.4967)$$

Table 3 shows the results of the hybrid MLR-GRU method for all cities for training and testing forecasting by using RMSE and improvement measurements.

Table 3 shows that the forecasting performance results of the MLR-GRU hybrid model for the training and testing

Table 3. The results of the hybrid MLR-GRU method for all cities for training and testing RMSE.

	Mosul	Talafar	Sinjar	Al_Baaj	Tal_Abtah	Makhmore
Train	0.0428	0.0539	0.0512	0.0387	0.0540	0.0422
Test	0.0436	0.0677	0.0633	0.0430	0.0579	0.0441
Improvement	6.55%	22.11%	21.71%	10.62%	7.06%	4.31%
R²	99.80%	99.50%	99.60%	99.70%	99.80%	99.80%

periods are outperformed the results of the traditional MLR model in Table 1, while the forecasting results of

MLR-GNN are the best compared to bot MLR and MLR-GRU for the training and testing periods. R^2 values for the training data of MLR-GRU can also proof that this method has perfect fitting with climatic data modelling, while Improvement measurement in Table 3 can reflect that the improvement performance is acceptable but it can't be compared to MLR-GNN performance.

Figure 9 can show the comparison between the expected values of the MLR-GRU model and the actual values of normal temperatures in six cities of Nineveh Governorate during the period from January to May 2024.

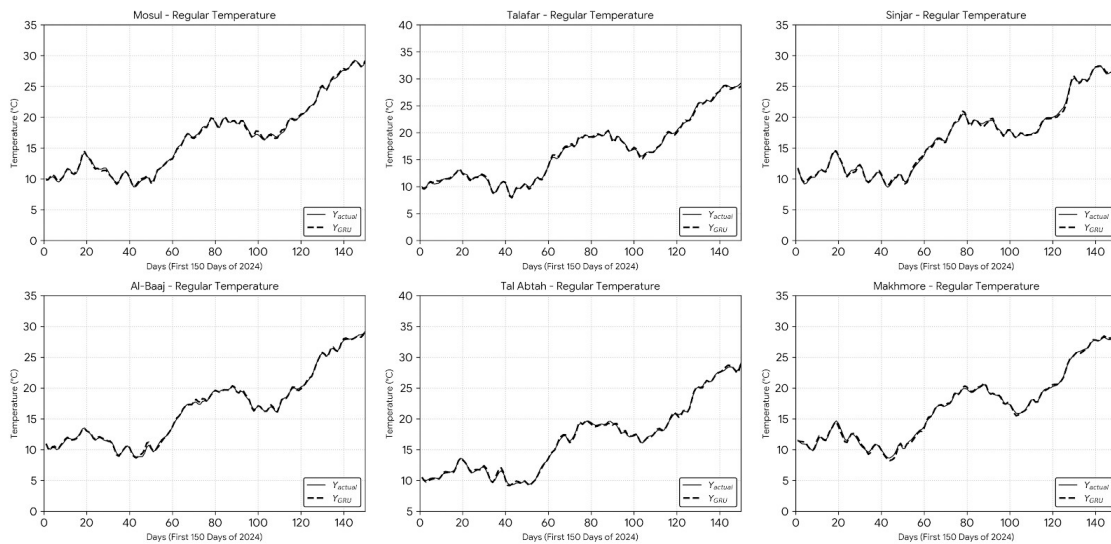


Figure 9. The actual and the MLR-GRU testing forecasting of normal temperature (from January to May 2024) for six cities.

Figure 9 of comparing the expected values of the MLR-GRU model with the actual values of normal temperatures confirms that the forecasted performance results of the MLR-GRU hybrid model for the first 5 months of the test periods have the perfect performance, These forecasteing results reflect the maximum forecasting quality of the MLR-GRU hybrid model.

This study does not aim to compare neural architectures; rather, it seeks to demonstrate that structuring neural inputs using a robust statistical model—specifically, Multiple Linear Regression (MLR)—provides an effective hybrid strategy grounded in sound methodological principles. Relying solely on Graph Neural Networks (GNNs) or Gated Recurrent Units (GRUs) would have necessitated arbitrary decisions regarding input structure and weights, thereby conflicting with the statistical foundation of this work. The two-stage hybrid mechanism ensures that the neural components are based on a statistically justified representation, and the results confirm that this approach enhances prediction accuracy compared to the conventional MLR model.

4. Conclusions

The results obtained by using a graphical neural network (GNN) model, specifically via the MLR-GNN hybrid model, demonstrated superior forecasted performance for normal temperatures, taking into account the influence of various climatic factors. Furthermore, the MLR-GRU hybrid model also showed better forecasted performance than the traditional MLR model for typical temperatures, though it certainly did not reach the level of performance achieved by the MLR-GNN hybrid model. Therefore, it can be concluded that the MLR-GNN hybrid model can be used for forecasting when the spatio-temporal relationships are considered. GNN is a specialized spatio-temporal

forecasting method, and when combined with a statistical modelling technique like MLR, it leads to optimal forecasting performance and significantly improved forecasting results compared to other methods. The MLR-GRU hybrid model is equally important for forecasting performance as a model specializing in time series modelling, but it is not specialized in spatio-temporal relationships. However, when it comes to seasonal and cyclical influences and the varying time scales of the complex components included in climate time series data, this MLR-GRU hybrid model will play a crucial role in improving modelling and forecasting performance, regardless of the spatial relationships between different climate stations. In general, hybridizing the neural networks GNN and GRU by integrating them practically with MLR results is an effective methodology for multivariable climate time series forecasting.

REFERENCES

1. F. Corradini, F. Gerosa, M. Gori, C. Lucheroni, M. Piangerelli, and M. Zannotti, *A systematic literature review of spatio-temporal graph neural network models for time series forecasting and classification*, Neural Networks, p. 108269, 2025.
2. Y. Gao, H. Wu, H. Liao, X. Chen, S. Yang, and H. Song, *A fault diagnosis method for rolling bearings based on graph neural network with one-shot learning*, EURASIP Journal on Advances in Signal Processing, vol. 2023, no. 1, p. 101, 2023.
3. X.-d. Liu, B.-h. Hou, Z.-j. Xie, N. Feng, and X.-p. Dong, *Integrating gated recurrent unit in graph neural network to improve infectious disease prediction: an attempt*, Frontiers in Public Health, vol. 12, p. 1397260, 2024.
4. A. Kazadi, J. Doss-Gollin, A. Sebastian, and A. Silva, *FloodGNN-GRU: a spatio-temporal graph neural network for flood prediction*, Environmental Data Science, vol. 3, p. e21, 2024.
5. C. Yang, P. Jin, and Y. Chen, *Leveraging graph neural networks and gate recurrent units for accurate and transparent prediction of baseball pitching speed*, Scientific Reports, vol. 15, no. 1, p. 7745, 2025.
6. J. Wang, *Multivariate time series forecasting and classification via GNN and transformer models*, Journal of Computer Technology and Software, vol. 3, no. 9, 2024.
7. G. TEMÜR, *Forecasting Carbon Emissions in Turkey's Energy Production Using a Regression-Based Hybrid LSTM-GRU Fusion Model*, Journal of Engineering Research, vol. 14, no. 2, 2026.
8. S. T. Lynch, P. Derakhshan, and S. Lynch, *A novel hybrid temporal fusion transformer graph neural network model for stock market prediction*, AppliedMath, vol. 5, no. 4, p. 176, 2025.
9. A. Kumar, A. Sinha, and Y. P. Singh, *Enhanced Traffic Prediction Using Spatial–Temporal Graph Neural Networks*, IEEE Access, vol. 14, pp. 107120–107129, 2026.
10. M. Gori, G. Monfardini, and F. Scarselli, *A new model for learning in graph domains*, In Proceedings. 2005 IEEE International Joint Conference on Neural Networks, vol. 2, pp. 729–734, 2005.
11. T. N. Kipf and M. Welling, *Semi-supervised classification with graph convolutional networks*, arXiv preprint arXiv:1609.02907, 2016.
12. W. L. Hamilton, R. Ying, and J. Leskovec, *Representation learning on graphs: Methods and applications*, arXiv preprint arXiv:1709.05584, 2017.
13. P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, *Graph attention networks*, International Conference on Learning Representations (ICLR), vol. 6, no. 2, 2018.
14. W. Zhao and Y. Jiang, *The Spatiotemporal Prediction Model of Urban Carbon Emissions Based on Graph Neural Networks*, International Journal of Information Technologies and Systems Approach (IJITSA), vol. 18, no. 1, pp. 1–17, 2025.
15. P. Vourlioti, T. Mamouka, M. Banti, C. Paraskevas, S. Kotsopoulos, V. Alexandridis, and G. Kalantzi, *HPC-Driven oceanographic predictions with Graph Neural Networks (GNNs) and Gated Recurrent Units (GRUs)*, Procedia Computer Science, vol. 255, pp. 103–111, 2025.
16. J. C. Heck and F. M. Salem, *Simplified minimal gated unit variations for recurrent neural networks*, In 2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS), pp. 1593–1596, 2017.
17. H. Zeng, C. Dong, R. Fu, K. Su, X. Leng, and C. Guo, *A gated Recurrent unit considering spatial correlation for short-term traffic volume forecasting in highway tunnels*, Engineering Applications of Artificial Intelligence, vol. 159, p. 111796, 2025.
18. S. A. Jumaa, and O. S. Ibrahim, *Addressing missing values in structural equation modeling with application to digital transformation survey data and its role in improving the quality of healthcare in Mosul*, Statistics, Optimization Information Computing, vol. 16, no. 3, 2026.
19. O. B. Shukur, *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.
20. O. S. Ibrahim, and Z. Y. Algarni, *Improving nonlinear regression model estimation based on coati optimization algorithm*, Statistics, Optimization Information Computing, vol. 14, no. 2, pp. 923–936, 2025.
21. O. S. Ibrahim, and Z. Y. Algarni, *Hybrid V-Support Vector Regression as Statistical Methodology for Forecasting Daily Natural Gas Prices*, Industrial Engineering Management Systems, vol. 24, no. 4, pp. 694–705, 2025.
22. M. A. H. Alsaeeh, 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.
23. R. A. Snaan, and O. B. Shukur, *Using Bayesian Ridge Regression Model and ESN for Climatic Time Series Forecasting*, Statistics, Optimization and Information Computing, vol. 14, no. 3, pp. 1226–1243, 2025.
24. S. T. Hasan, and O. B. Shukur, *Using Bayesian AR-ESN for climatic time series forecasting*, Statistics, Optimization and Information Computing, vol. 14, no. 6, pp. 3816–3832, 2025.

25. K. R. N. Fathi, *Atlas of Forecasting for Climate Nineveh Governorate*, Ph.D. dissertation, Dept. Geogr., College of Arts, Al-Iraqia University, Baghdad, Iraq, 2026.