










Article

Long Short-Term Memory Recurrent Neural Network and Extreme Gradient Boosting Algorithms Applied in a Greenhouse's Internal Temperature Prediction

Juan M. Esparza-Gómez ¹, Luis F. Luque-Vega ², Héctor A. Guerrero-Osuna ¹, Rocío Carrasco-Navarro ³, Fabián García-Vázquez ¹, Marcela E. Mata-Romero ⁴, Carlos Alberto Olvera-Olvera ¹, Miriam A. Carlos-Mancilla ^{5,6,*} and Luis Octavio Solís-Sánchez ¹

- ¹ Posgrado en Ingeniería y Tecnología Aplicada, Unidad Académica de Ingeniería Eléctrica, Universidad Autónoma de Zacatecas, Zacatecas 98000, Zacatecas, Mexico; juan.esparza@upsz.edu.mx (J.M.E.-G.); hectorguerrero@uaz.edu.mx (H.A.G.-O.); 31126593@uaz.edu.mx (F.G.-V.); colvera@uaz.edu.mx (C.A.O.-O.); lsolis@uaz.edu.mx (L.O.S.-S.)
- ² Department of Technological and Industrial Processes ITESO AC, Tlaquepaque 45604, Jalisco, Mexico; luisluque@iteso.mx
- ³ Research Laboratory on Optimal Design, Devices and Advanced Materials—OPTIMA, Department of Mathematics and Physics, ITESO, Tlaquepaque 45604, Jalisco, Mexico; rociocarrasco@iteso.mx
- ⁴ Subdirección de Investigación, Centro de Enseñanza Técnica Industrial, C. Nueva Escocia 1885, Guadalajara 44638, Jalisco, Mexico; mmata@ceti.mx
- ⁵ Centro de Investigación, Innovación y Desarrollo Tecnológico CIIDETEC-UVM, Universidad del Valle de México, Tlaquepaque 45601, Jalisco, Mexico
- ⁶ Centro Universitario de los Valles, Universidad de Guadalajara, Ameca 46600, Jalisco, Mexico
- * Correspondence: miriam_carlos@my.uvm.edu.mx; Tel.: +52-33-3669-8400 (ext. 23220)



Citation: Esparza-Gómez, J.M.; Luque-Vega, L.F.; Guerrero-Osuna, H.A.; Carrasco-Navarro, R.; García-Vázquez, F.; Mata-Romero, M.E.; Olvera-Olvera, C.A.; Carlos-Mancilla, M.A.; Solís-Sánchez, L.O. Long Short-Term Memory Recurrent Neural Network and Extreme Gradient Boosting Algorithms Applied in a Greenhouse's Internal Temperature Prediction. *Appl. Sci.* **2023**, *13*, 12341. <https://doi.org/10.3390/app132212341>

Academic Editor: José Miguel Molina Martínez

Received: 23 September 2023
Revised: 28 October 2023
Accepted: 6 November 2023
Published: 15 November 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Abstract: One of the main challenges agricultural greenhouses face is accurately predicting environmental conditions to ensure optimal crop growth. However, the current prediction methods have limitations in handling large volumes of dynamic and nonlinear temporal data, which makes it difficult to make accurate early predictions. This paper aims to forecast a greenhouse's internal temperature up to one hour in advance using supervised learning tools like Extreme Gradient Boosting (XGBoost) and Recurrent Neural Networks combined with Long-Short Term Memory (LSTM-RNN). The study uses the many-to-one configuration, with a sequence of three input elements and one output element. Significant improvements in the R^2 , RMSE, MAE, and MAPE metrics are observed by considering various combinations. In addition, Bayesian optimization is employed to find the best hyperparameters for each algorithm. The research uses a database of internal data such as temperature, humidity, and dew point and external data such as temperature, humidity, and solar radiation, splitting the data into the year's four seasons and performing eight experiments according to the two algorithms and each season. The LSTM-RNN model produces the best results for the metrics in summer, achieving an $R^2 = 0.9994$, RMSE = 0.2698, MAE = 0.1449, and MAPE = 0.0041, meeting the acceptability criterion of ± 2 °C hysteresis.

Keywords: smart farming; greenhouse forecasting; machine learning; microclimate prediction

1. Introduction

Agriculture has become increasingly important in recent years due to its various challenges in producing food from environmental, social, and economic standpoints. The United Nations has projected that the global population will grow to 9.7 billion by 2050 [1], making it essential to produce enough food. However, the available agricultural land is limited, requiring significant efficiency gains to maximize production with scarce resources. Furthermore, special attention should be paid to climate change, which will

require reducing greenhouse gas emissions as its atmospheric concentration is projected to double by 2030 and cause significant global temperature increases [2,3].

Climate-Smart Agriculture (CSA) is a system that utilizes the latest technological advancements to tackle the challenges faced in agriculture. The Food and Agriculture Organization of the United Nations (FAO) defines CSA as a tool that enhances national food security and development goals. It involves various new technologies that work together to achieve sustainable productivity, resilience, and reduction in/removal of greenhouse gas emissions [4,5].

The CSA concept encompasses the Smart Farming (SF) initiative, which promotes eco-friendly agricultural practices that rely on science and technology. SF is comparable to Industry 4.0's smart factories, utilizing Information and Communication Technologies (ICT) like the Internet of Things (IoT), Global Positioning System (GPS), Cloud Computing (CC), Fog Computing (FC), and Big Data (BD) analysis to monitor and manage farms and farming activities [6–8]. These technologies are driving the transformation of the agricultural sector towards a smarter and more sustainable one, contributing to the significant problems facing agriculture [9].

Smart Farming includes the greenhouse concept essential for profitable and sustainable agricultural practices. Greenhouses control important factors affecting crop growth, such as solar radiation, temperature, humidity, light intensity, and carbon dioxide levels, helping to increase yields throughout the year and protect crops from harsh weather conditions and pests [10,11]. However, the unstable conditions within a greenhouse can negatively impact plant growth and ultimately reduce crop production. This issue can be mitigated using Artificial Intelligence (AI) tools that regulate the greenhouse variables [12–14].

Machine Learning (ML) is the subtopic of AI that allows a system to learn from previous experiences and adjust accordingly. By analyzing large amounts of data, ML can make accurate predictions. This technology helps reduce pre-harvest crop loss, often caused by inadequate irrigation and climate conditions. By utilizing data collected from sensors and an automated watering system, farmers can optimize their crop yields and minimize losses [15,16].

Various agricultural tasks can benefit significantly from ML algorithms, which have produced state-of-the-art results. The most prominent models include Linear Regressions (LRs), Support Vector Machines (SVMs), K-Nearest Neighbor (k-NN), Neural Networks (NNs), Random Forest (RF) classifiers, and Decision Trees (DTs) [17]. Additionally, emerging forecasting methods such as Recurrent Neural Networks (RNNs) and Extreme Gradient Boosting (XGBoost) help analyze time series data. The Long Short-Term Memory (LSTM)-RNN model and XGBoost are particularly effective at avoiding the issue of vanishing and exploding gradients during training [18,19].

Predicting microclimate conditions in greenhouses through climate forecasting has become crucially important. This is possible due to the advanced sensors and systems that enable exact measurements and evaluations of the microclimate within seconds. Although different techniques have been developed to model temperature behavior inside greenhouses, they often only use variable monitoring instead of forecasting, limiting the implementation of automatic control to corrective and non-preventative actions, which may only partially satisfy the needs of greenhouse growers. However, AI-based algorithms have been developed to act preventively by adjusting heating/cooling systems, ventilation, and carbonic fertilization supply through actuators installed to ensure optimal growth, maintenance, and pest control of crops in greenhouses [20].

Integrating a preventive model into the automatic control system of greenhouses is vital to ensuring optimal crop growth. Extreme temperatures can negatively affect crop morphology and physiological processes, resulting in floral formation, leaf burn, poor fruit quality, excess transpiration, and a shortened crop lifespan. By effectively controlling the microclimate within the greenhouse, the risk of developing pathogens and damaging crops can be minimized [21].

Previous research by García-Vázquez et al. [22] focused on creating an accurate temperature prediction model for a greenhouse using linear and support vector regression techniques. However, this paper proposes using the same greenhouse data set to predict the internal temperature up to one hour in advance by applying supervised learning algorithms such as LSTM-RNN and XGBoost. These algorithms are used because LSTM excels at processing sequential data. They can remember long-term patterns and learn temporal dependencies through specialized memory units [23]. On the other hand, XGBoost combines multiple weak models to form a more robust model, allowing the model to learn complex relationships between features and obtain more accurate results [24]. Both techniques provide effective time series forecasting; therefore, they are suitable for enabling preventive control of the greenhouse.

The contributions of this paper can be summarized as follows: In order to prevent issues like crop diseases and poor fruit quality caused by uncontrolled microclimate changes inside greenhouses, meteorological data can be analyzed using supervised learning techniques in a controlled environment. By utilizing predictive models and intelligent control systems in agriculture, crop efficiency, and productivity can be significantly enhanced while minimizing the risks associated with sudden changes in greenhouse conditions. The comparison between different ML models lets the researchers/producers analyze which one fits their needs better, for example, adding more computational power if higher prediction accuracy is needed or less computational power with less prediction accuracy, taking into consideration that this LSTM-RNN will need a lot of more power to achieve the level of accuracy that this paper shows later in the results section.

A study on data segmentation based on the annual climate seasons shows that an algorithm per season would benefit temperature prediction due to the particularities of each one, and prediction errors tend to increase when projecting towards extended periods. Also, this technology has the potential to benefit the agricultural industry by improving crop quality and yield while also reducing the consumption of resources, such as energy.

The paper is organized as follows: Section 2 shows the existing works related to SF in greenhouses. Section 3 details the proposed workflow using the Team Data Science Process (TDSP) methodology and explains how forecasting in the greenhouse is implemented. The results are presented in Section 4, while Section 5 provides a discussion. The paper closes with Section 6, which summarizes the findings and presents the conclusions.

2. Related Work

This section explores the current agricultural systems and discusses ways to improve them by leveraging data from different sources. The text presents the latest technological advancements in agriculture, focusing on machine-learning applications and the prediction of climatological variables within greenhouses. This information is valuable for individuals seeking to enhance agricultural practices and increase productivity.

Authors in [25] implemented the LSTM-RNN to work with time series to forecast the temperature and relative humidity inside a greenhouse, considering microclimatic parameters as input components. Solar radiation, external temperature, external humidity, and wind direction were considered impact variables on the forecast. Based on this, the proposal focuses on observing the behavior of the LSTM-RNN through the variation in the number of hidden layers and analyzing the Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and Coefficient of Determination (R^2) to obtain the level of efficiency of the network in the forecasting of temperature and relative humidity.

In [26], the authors presented an RNN-Back-Propagation (BP) model to predict the temperature and humidity of a solar greenhouse in northern China. Climate data, such as air and substrate temperature, humidity, lighting, and carbon dioxide (CO_2) concentration recorded over eight days, were used to build and validate climate prediction models. The results show that the model provides reasonably good predictions, with RSMEs of 0.751 for temperature and 0.781 for humidity, and both R^2 values being above 0.9, outperforming the models in this study.

The study in [27] introduces a solar greenhouse evapotranspiration (ET) estimation model called PSO (Particle Swarm Optimization)-XGBoost. This model uses meteorological and soil moisture data from a greenhouse in China to grow two tomato crops. The PSO-XGBoost model accurately estimates evapotranspiration under different data configurations. The study found that the model's accuracy decreases as input variables are reduced. Moreover, the variables solar radiation and vapor pressure difference were identified as the most critical variables.

The work presented in [28] aims to manage and predict the air temperature in greenhouses using a wireless sensor network for data collection. The prediction model is based on RNN techniques. The prediction algorithm utilizes weather data to predict the missing values in case of missing sensor data. The experimental results show that the RNN-based prediction algorithm can efficiently forecast the air temperature in the greenhouse with an RMSE of 1.50 °C, a Mean Absolute Percentage Error (MAPE) of 4.91%, and an R^2 of 0.965.

The study in [29] discusses using an RNN model as a deep learning architecture that can retain the recent memories of input patterns. This is achieved through the feedback mechanism of its hidden layer outputs to itself. The RNN model is commonly used for time-series-related work, particularly in predicting climate variables. It can analyze input and output data from other identification models to make accurate predictions.

Authors in [30] evaluated Artificial Neural Networks (ANNs), Nonlinear Autoregressive Exogenous (NARX) models, and LSTM-RNN models to predict environmental changes in temperature, humidity, and CO₂ in greenhouses. The best model for all data sets was the LSTM-RNN, with continuous prediction accuracy even after 30 min, obtaining an R^2 of 0.96 for temperature, 0.80 for humidity, and 0.81 for CO₂.

The work in [31] utilized five models to collect and analyze a data set from 27 greenhouses in South Korea. The variables analyzed included internal temperature, relative humidity, radiation, CO₂ concentration, and external temperature. The most efficient model was Bidirectional LSTM (BiLSTM), which resulted in an average R^2 of 0.78 and 0.81 for the pepper and tomato data sets, respectively. The model effectively adapted pre-trained deep learning models and improved their prediction ability in data-limited greenhouse microclimates.

A research study described in [32] aims to improve the accuracy of predicting crop yields in greenhouses. Such precision is essential for making informed decisions about the planning and management of greenhouse agriculture. The researchers developed a new algorithm to achieve this goal by combining a Temporal Convolution Network (TCN) and an RNN. They evaluated the algorithm's performance using data from several greenhouses cultivating tomatoes. The results demonstrate that this new technique outperforms traditional machine learning methods and other deep neural networks in accuracy. The study also highlights that historical performance information is crucial for accurately predicting future crop yields.

In [33], researchers have proposed a Multivariate Long Short-Term Memory (MV-LSTM) neural network model for predicting wind speed. The model is based on the feature selection of the Pearson correlation coefficient and utilizes temperature, humidity, and air pressure data to predict the wind speed at two observation stations in Beijing for the next hour. The study compared the MV-LSTM model with the Auto-Regressive Integrated Moving Average (ARIMA) and LSTM methods. It demonstrated that the MV-LSTM model outperforms the other models regarding prediction accuracy.

In [34], a study was conducted to predict wind energy in ultra-short periods. The researchers developed a hybrid Spatio-Temporal Correlation Model (STCM) using Convolutional Neural Networks (CNN) and LSTM. The model takes multiple meteorological factors as input and reconstructs the data into a matrix for CNN to extract spatial correlation and LSTM for temporal correlation. The model was tested on a wind farm in China and showed a significant improvement in accuracy compared to using CNN or LSTM alone. The error rate was reduced by around 30%. However, the study recognizes the need to investigate the impact of different meteorological factors and improve the model's optimization algorithm.

Research in [35] was conducted to estimate crop ET. Since ET is not easy to measure directly, a simulation model based on XGBoost Regression (XGBR) was developed to obtain it. The researchers used three years of data and eight meteorological factors (net solar radiation, mean temperature, minimum temperature, maximum temperature, relative humidity, minimum relative humidity, maximum relative humidity, and wind speed) to train the XGBR-ET model. They compared it with seven other standard regression models. The results showed that net solar radiation had the highest positive correlation with ET, while wind speed had the lowest correlation.

In [36], the authors proposed a model that utilizes the Light Gradient Boosting Machine (LightGBM) algorithm to predict the internal temperature of a greenhouse. The model uses control and temporal data collected over five years. The study reports that the LGBM model has better tuning capacity and is significantly faster in training than other models like neural networks, BP, RNN, XGBoost, and Stochastic Gradient Boosting (SGB). These results indicate that the LGBM model has great potential for real-time prediction and control of the greenhouse environment.

The work in [37] uses time series data to predict ET and humidity in tomato greenhouses. The researchers applied an LSTM model to predict ET and compared it with the Stanghellini model. During the training phase, the developed ET model had an RMSE of 0.00317 and 0.00356 during the testing phase, with percentage errors of 5.76% and 6.45%, respectively. In addition, a humidity prediction model performed better than traditional LSTM-RNN models.

In [38], the researchers aimed to predict the peak energy consumption of an intelligent farm using various algorithms. They analyzed energy data collected from a unique pepper farm in South Korea between 2019 and 2021. The study compared several machine learning algorithms, including ANN, SVM, RF, XGBoost, k-NN, Gradient Boosting Machine (GBM), and ARIMA. The most successful model was based on RF, achieving an accuracy of 92%. Additionally, the research analyzed the variables' importance, identifying that internal humidity, dew point, and external temperature are critical factors in predicting energy consumption in the innovative farm.

A study in [39] presented a model to predict the temperature inside a greenhouse using time series analysis and the LightGBM. The model considered environmental factors such as humidity, air pressure inside and outside the greenhouse, external temperature, and time series data to make the temperature predictions. The study found that incorporating time series features improved the R^2 and reduced the Mean Square Error (MSE) and the MAE across several prediction models. Other models, such as RNN, SVR, and LR, were also compared, but LightGBM outperformed them all regarding model fit.

In [40], a Dense Neural Network (DNN) framework was proposed to predict temperature and relative humidity measurements. According to the results, the framework shows a high correlation of 0.91 and 0.85 for temperature and humidity, respectively. The framework has significantly reduced prediction errors, with a 68.67% reduction for temperature and 46.21% for relative humidity compared to an approach without the DNN model.

Several investigations, such as [41–44], have also contributed relevant studies to agricultural areas using artificial intelligence focused on the geotechnical properties of soil. Researchers have used machine learning techniques to predict crucial soil properties like thermal conductivity and mechanical behavior. Methods like LR, Gaussian process regression, SVR, DT, RF, and adaptive boosting have been employed to predict soil thermal conductivity. Furthermore, researchers have utilized neural network techniques like LSTM to model the mechanical behavior of frozen soils. This comprehensive approach has provided a deeper understanding of soil properties and has become an essential tool in geotechnical engineering.

3. Materials and Methods

The methodology used to develop the temperature forecast is based on TDSP. This is an agile and iterative approach that helps to develop predictive analytics solutions and

intelligent applications efficiently. By providing a structured lifecycle, TDSP guides the development of data science projects [45].

TDSP is a leading approach in the technical field of machine learning and data science. This methodology is used in this study because it is unique in its structured and project-oriented approach that covers all critical stages of the project lifecycle, from data preparation to model implementation and deployment. Additionally, TDSP incorporates continuous iteration techniques that allow agile adaptation of the model based on feedback and changes in data. With an emphasis on interdisciplinary collaboration and effective communication, TDSP ensures an accurate understanding of real-world problems [45].

TDSP follows a series of steps to develop a data science project. The initial steps, which concentrate on business knowledge and data acquisition, were outlined in [22]. The focus was obtaining information from a curved-roof-type greenhouse between July 2020 and June 2021 for data acquisition. The data collected included various variables obtained from sensors of a weather system; these are listed in Table 1.

Table 1. Greenhouse data set variables.

Variable Nomenclature	Description	Unit of Measurement
Ti	Internal temperature	°C
To	External temperature	°C
Ho	External humidity	%
Hi	Internal humidity	%
Di	Internal dew point	%
Rs	Solar radiation	W/m ²

°C = Degrees Celsius, % = Percentage, W/m² = Watts per square meter.

The third step of the TDSP methodology focuses on modeling, which is based on feature engineering, training, and model evaluation.

3.1. Feature Engineering

Feature engineering plays a vital role in developing ML models and data analysis. Its main objective is to enhance the models' performance and efficiency by selecting, transforming, and creating significant features from the initial data. This crucial process can significantly impact the accuracy and efficiency of the ML algorithms [46].

For data pre-processing, the first step was to detect and delete the null values within the data sequence, where data of the climatic variables of interest were found and did not let the proposed algorithms apply due to not having a valid or consistent value. Secondly, the null data were replaced by using linear interpolation techniques, and the database was grouped into seasons of the year (3 months per interval) because it benefits in preventing underfitting during model training. Finally, they were divided into training and validation data of the LSTM-RNN model.

The database analysis determined that each season has distinct patterns and varying quantities of samples, as shown in Figure 1. Research by García-Vázquez et al. [22] has indicated that dividing the database into annual seasons is helpful to prevent underfitting. This is due to the varying temperature trends observed throughout the year. After considering this information, it was decided to proceed with prediction models that use the seasonal division of data.

Striking a balance between incorporating informative variables and avoiding unrelated ones is important. Incorporating informative variables can enhance the output, while unrelated ones can introduce unnecessary noise to the model. Therefore, the database variables were analyzed to identify significant correlations and determine which variables should be used in the models to ensure accurate predictions.

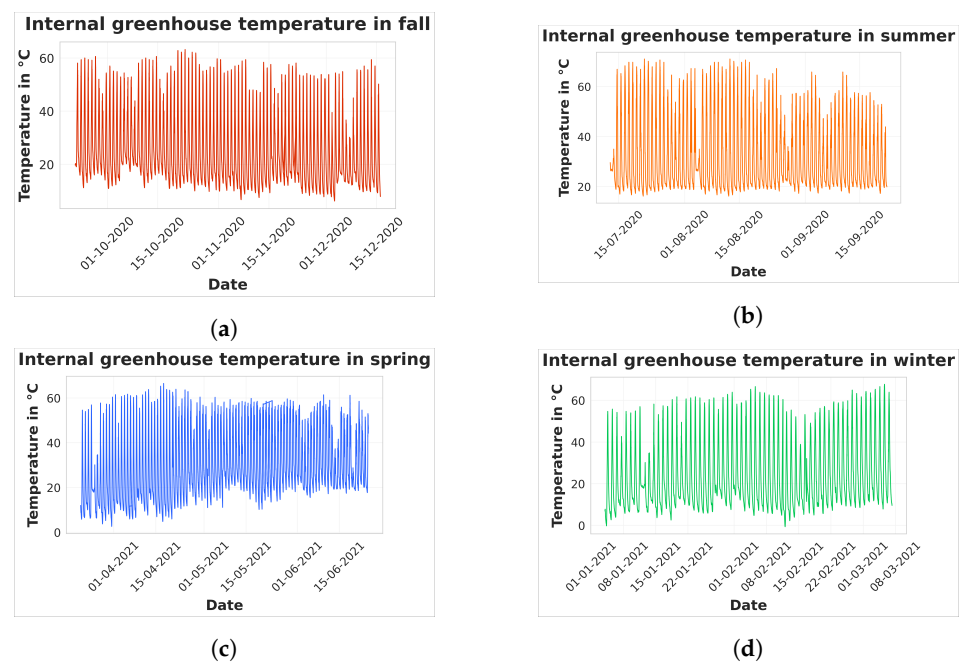


Figure 1. Dataset corresponding to greenhouse internal temperature. Data for fall (a), summer (b), spring (c), and winter (d).

The correlational analysis can be seen in Figure 2, where each variable is shown concerning each year's season. This enabled us to identify the following key points:

- Relative humidity decreases as the internal temperature increases;
- Internal temperature increases as the solar radiation increases;
- The dew point increases as the relative humidity increases;
- The internal temperature is affected by the external temperature.

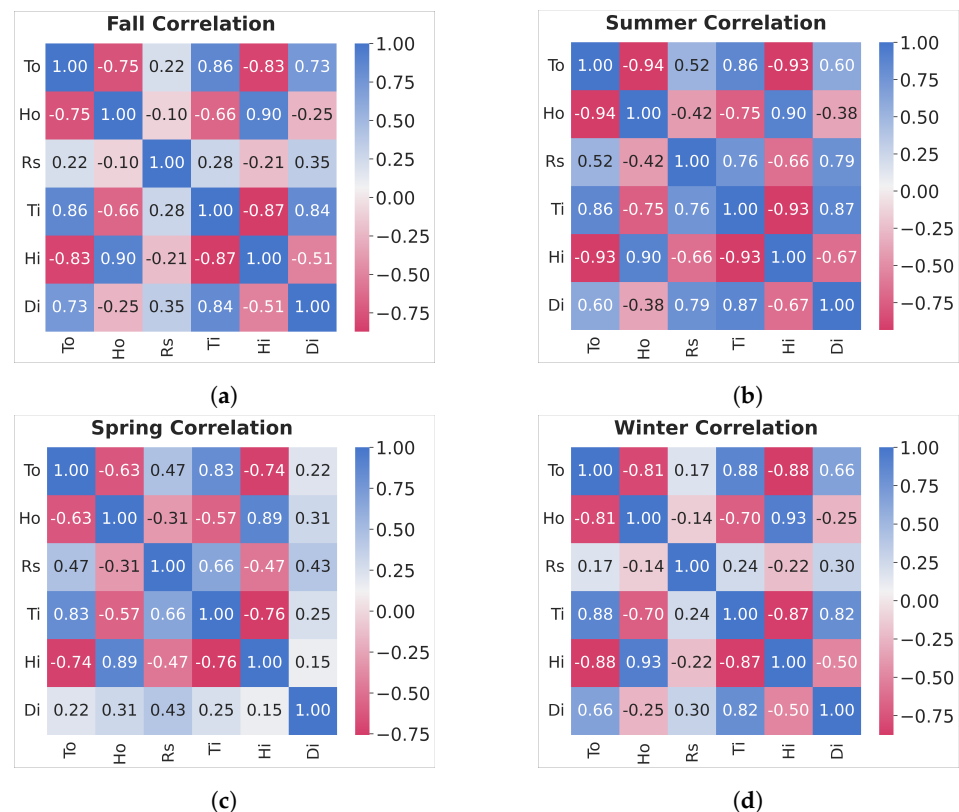


Figure 2. Correlation maps for each year's season. Fall (a), summer (b), spring (c), and winter (d).

In order to create a prediction model for the internal temperature of a greenhouse, this research focuses on the following independent variables: To, Ho, Hi, Di, and Rs, whereas Ti is the dependent variable. The correlation maps show positive correlations between two or three variables; thus, for a sequence of three input elements and one output element, the many-to-one configuration can be used to perform an analysis. By analyzing these variable combinations, irrelevant or redundant features can be eliminated and reduce the risk of overfitting, leading to a better-performing model.

Equation (1) provides the possible combinations for arranging three input elements based on independent variables:

$${}^nC_r = n! \frac{n-r}{r!} r! \quad (1)$$

where n represents the independent variables, and r represents the input variables considered for the model. Table 2 displays the ten combinations used for testing the models in many-to-one analysis.

Table 2. Sequence of input–output variables (many-to-one).

Combination Number	Input Sequence	Output
1	Hi-Di-To	Ti
2	Hi-Ho-To	
3	Hi-To-Rs	
4	Di-Rs-To	
5	Ho-To-Rs	
6	Hi-Di-Rs	
7	Hi-Di-Ho	
8	Hi-Rs-Ho	
9	Di-Rs-Ho	
10	Di-Ho-To	

3.2. Model Training

This paper proposes using the LSTM-RNN and XGBoost algorithms to forecast the internal temperature of a greenhouse due to its adaptability in applications with time series. The subsequent points outline the composition of each algorithm and the method for choosing hyperparameters in these techniques.

3.2.1. LSTM-RNN

An ANN is a computing system inspired by biological neural processing research. The model consists of interconnected processing nodes, or neurons, that work together to solve complex problems. During training, the connections between nodes have their weights adjusted. A typical ANN, called a multiple-layer perceptron network, includes three layers: input, hidden, and output [47]. This is illustrated in Figure 3, and the formulas for these layers are expressed in Equations (2) and (3).

$$y_t = G_{so}(W_{so}s_t + b_0) \quad (2)$$

$$s_t = G_{is}(W_{is}x_t + b_1) \quad (3)$$

where G_{so} and G_{is} represent the activation functions for the hidden-to-output and input-to-hidden layers, respectively. The weight parameters are represented by matrices W_{so} and W_{is} , and biases for each layer are indicated by b_0 and b_1 . The hidden state vector at step t is represented by the variable s_t .

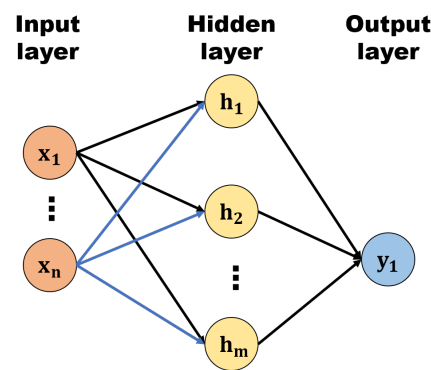


Figure 3. Structural overview of ANN.

RNNs differ from traditional neural networks in that they use feedback loops. This means that the output from each layer is sent back to the RNN to influence the current stage's results. RNNs are effective in classification analysis and working with time series data, allowing for nonlinear trajectory prediction and dynamic system modeling. These networks are frequently selected for processing sequential data. Figure 4 depicts an RNN network temporarily deployed to represent an element of the modeling string.

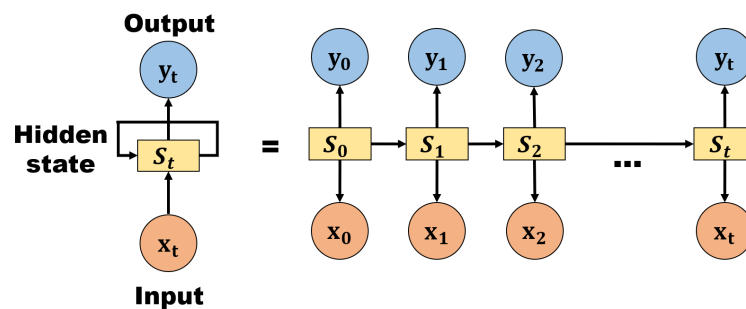


Figure 4. Structural overview of RNN.

The hidden state vector for RNNs in Equation (3) can be adjusted according to Equation (4):

$$s_t = G_{is}(W_{is}x_t + W_{s}s_{t-1} + b_1) \quad (4)$$

where W_s is the matrix of weight parameter for the prior hidden state.

RNNs have different aspects, with LSTM being the most commonly used for time series forecasting. LSTM includes memory cells with three types of gates: input, output, and forget gates. The input gate regulates the flow into the memory cell of input activations, while the output gate regulates the yield flow to the rest of the network of cell activations. The forget gate scales the cell's inner state before adding it to the cell as input through the cell's self-recurring association, thus forgetting or resetting the cell's memory adaptively. The memory cell is the key to the LSTM framework, as it goes directly down the entire chain with the capacity to add or extract cell state information, tightly controlled by structures called gates. These gates are optional data inlet ways consisting of a neural net layer sigmoid and point-sensitive multiplication [47], as shown in Figure 5.

The first step in LSTM is determining which data will be removed from the cell state. This decision is made by the forget gate (f_t), as shown in Equation (5):

$$f_t = \sigma(W_{fx}x_t + W_{fs}s_{t-1} + b_f) \quad (5)$$

where σ is the sigmoid function. The next step is determining what new information should be saved in the cell's state by two actions. The first is the input gate (i_t) selecting which

data to modify, as expressed in Equation (6). The second is a hyperbolic tangent (tanh) layer that generates new values for the applicant values based on Equation (7):

$$i_t = \sigma(W_{ix}x_t + W_{is}s_{t-1} + b_i) \quad (6)$$

$$\tilde{c}_t = \tanh(W_{\tilde{c}x}x_t + W_{\tilde{c}s}s_{t-1} + b_{\tilde{c}}) \quad (7)$$

Then, Equation (8) is used to update the previous state of the cell (c_{t-1}) to the new state (c_t):

$$c_t = f_t \oplus c_{t-1} + i_t \oplus \tilde{c}_t \quad (8)$$

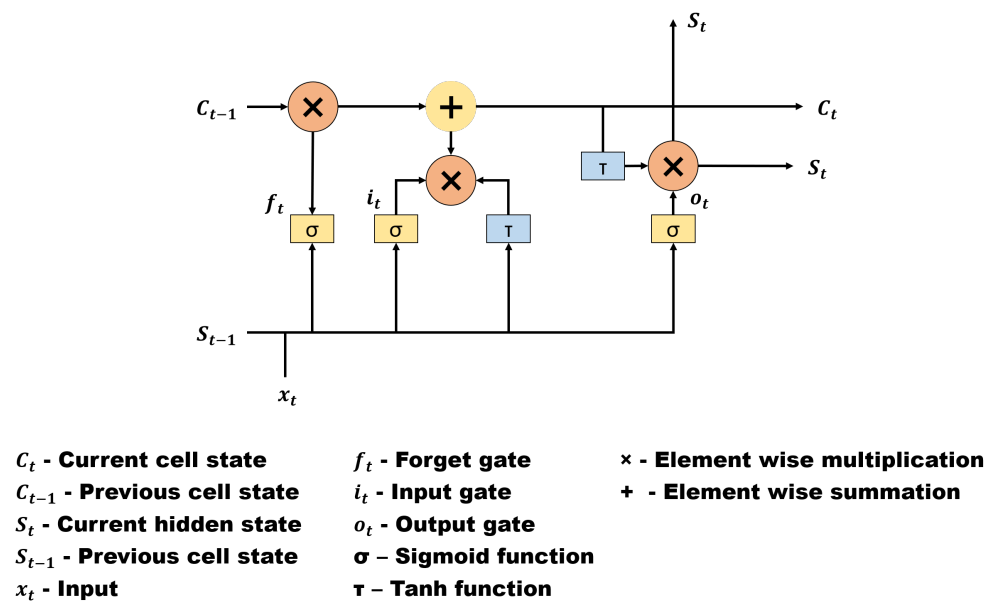


Figure 5. Structural overview of LSTM.

The output gate (o_t) selects the components of the cell state that will be generated as output, as shown in Equation (9). The cell state goes through a tanh layer and is multiplied by the output gate using Equation (10):

$$o_t = \sigma(W_{ox}x_t + W_{os}s_{t-1} + b_o) \quad (9)$$

$$s_t = o_t \oplus \tanh(c_t) \quad (10)$$

where W and b are the matrix and bias of the weight parameter and \oplus is the pointwise multiplication.

3.2.2. XGBoost

The XGBoost algorithm was created to find better ways to enhance decision trees. Initially, it was designed as a self-contained program that generated prediction models from input data. However, XGBoost's integration with standard interface systems enabled it to evolve into a more robust package that utilizes computational resources to produce accurate predictions in less time [48].

Boosting is an assembly method that combines different weak models into a robust model to minimize training errors. Usually, weak models tend to overfit, so having several additive models allows for a better model. In boosting, a random sample of data is selected, and a model is applied and trained sequentially, where each model tries to compensate for the weak points of the previous one [49], as demonstrated in Figure 6.

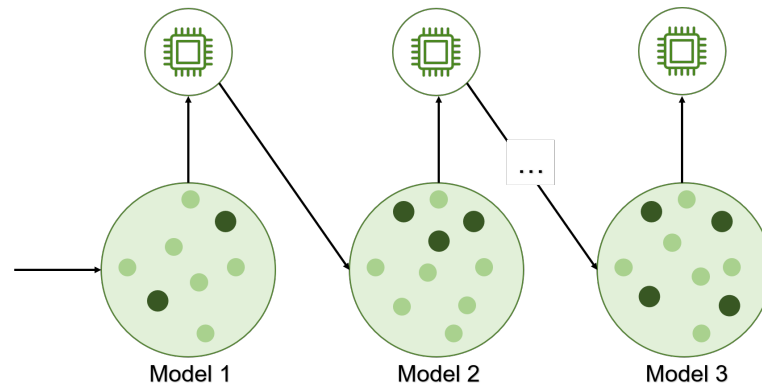


Figure 6. Sequential boosting algorithm.

XGBoost is a variation of the gradient boosting algorithm that adds predictors to an array in sequence to correct errors from previous predictors. This method combines gradient descent and boosting. XGBoost uses this method with minor modifications that improve the regularization target function. Considering a database D with m features and n number of examples $D = \{(x_i, y_i)\} (D = n_i \in \mathbb{R}^m, y_i \in \mathbb{R})$, XGBoost can be used to create an ensemble tree model that uses K additive functions to predict the output \hat{y}_i , as expressed in Equation (11) [50]:

$$\hat{y}_i = \phi(x_i) = \sum_{k=1}^K f_k(x_i), \quad f_k \in \mathcal{F}, \quad (11)$$

where $\mathcal{F} = \{f(x) = w_{q(x)}\} (q : \mathbb{R}^m \rightarrow T, w \in \mathbb{R}^T)$ represents the Classification and Regression Trees (CARTs). In this context, q refers to the structure of a tree that maps an instance to a specific leaf index. T represents the total number of leaves on each tree. Meanwhile, f_k pertains to a distinct tree structure with leaf weights w . K denotes the number of trees utilized in the model.

Equation (11) can be solved by finding the best set of functions by minimizing the regularization function and cost, as expressed in Equation (12):

$$\mathcal{L}(\phi) = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k) \quad (12)$$

where $\Omega(f) = \gamma T + \frac{1}{2} \lambda \|w\|^2$

In Equation (12), l stands for the cost function, which calculates the difference between the predicted value \hat{y}_i and the target value y_i . Ω penalizes the model's complexity, which helps avoid overfitting by smoothing out the final learned weights. The objective function will return to traditional gradient tree boosting if the regularization term is zero.

The tree ensemble model described in Equation (11) contains functions treated as parameters that cannot be optimized using traditional methods in a Euclidean space. To continue training after adding a new function f to the model, another function (tree) is added at the t -th iteration, as shown in Equation (13):

$$\mathcal{L}^{(t)} = \sum_{i=1}^n l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_t) \quad (13)$$

Then, a second-order approximation is used to optimize the objective in a general way, where $g_i = \delta_{\hat{y}^{(t-1)}} l(y_i, \hat{y}^{(t-1)})$ and $h_i = \delta_{\hat{y}^{(t-1)}}^2 l(y_i, \hat{y}^{(t-1)})$. This will result in a simple objective function for step t , as expressed in Equation (14):

$$\tilde{\mathcal{L}}^{(t)} = \sum_{i=1}^n [g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \Omega(f_t) \quad (14)$$

Defining $I_j = \{i | q(x_i) = j\}$ as an leaf instance of j , Equation (14) can be rewritten as Equation (15):

$$\tilde{\mathcal{L}}^{(t)} = \sum_{j=1}^T [(\sum_{i \in I_j} g_i) w_j + \frac{1}{2} (\sum_{i \in I_j} h_i + \lambda) w_j^2] + \gamma T \quad (15)$$

For a fixed structure of $q(x)$, the optimal value of the weight w_j^* of leaf j is obtained with Equation (16):

$$\tilde{\mathcal{L}}^{(t)}(q) = -\frac{1}{2} \sum_{j=1}^T \frac{(\sum_{i \in I_j} g_i)^2}{\sum_{i \in I_j} h_i + \lambda} + \gamma T \quad (16)$$

Enumerating all possible tree structures q is often complicated. Therefore, a greedy algorithm begins with a single leaf and gradually adds branches to the tree. When there is a split, the sets I_L and I_R represent the instances of the left and right nodes, respectively. $I = I_L \cup I_R$ represents the instances of the entire tree. The cost function after a split can be expressed using Equation (17):

$$\mathcal{L}_{split} = \frac{1}{2} \left[\frac{(\sum_{i \in I_L} g_i)^2}{\sum_{i \in I_L} h_i + \lambda} + \frac{(\sum_{i \in I_R} g_i)^2}{\sum_{i \in I_R} h_i + \lambda} - \frac{(\sum_{i \in I} g_i)^2}{\sum_{i \in I} h_i + \lambda} \right] - \gamma \quad (17)$$

When learning trees, a significant issue is determining the best split; thus, the XGBoost algorithm examines all possible splits for continuous features. To efficiently accomplish this, the algorithm first sorts the data based on the feature values and then analyzes the sorted data to gather gradient statistics for the score in Equation (17).

It is common for input x to be sparse in various situations. This means that x may have missing values, null values, frequent null entries in the statistics, and artifacts from one-hot encoding. Therefore, the algorithm must be able to detect sparse data. Although the block structure helps optimize the computational complexity of the node split search, the new algorithm requires an indirect search of gradients and Hessians for each row since these values are accessed by features in order. XGBoost addresses these challenges by utilizing the processor's cache memory. The algorithm caches the gradients and Hessians to compute the similarity scores and output values, improving model training computational time.

3.2.3. Hyperparameters

It is essential to adjust hyperparameters correctly when creating a mathematical prediction model to ensure accurate results. There are two ways to select hyperparameters: using default values provided by the software or manually configuring them. Additionally, data-dependent hyperparameter optimization strategies, such as grid or random search, can be employed. These strategies use second-degree optimization procedures to minimize the expected error of the model by searching for candidate configurations of hyperparameters.

On the other hand, Bayesian optimization is a more complex iterative strategy that can be used to identify the best hyperparameters due to its efficiency and effective uncertainty management. Using probability models, Bayesian optimization guides the search in a focused manner, reducing the number of evaluations necessary. Furthermore, its ability to handle uncertainty and balance exploitation and exploration ensures adaptability to various objective functions, even non-linear or expensive to evaluate. This systematic and versatile methodology maximizes model performance, making it an essential tool in hyperparameter optimization for complex machine learning problems [51].

Various hyperparameters can be modified when utilizing the LSTM-RNN technique, as listed in Table 3 [52,53]. Likewise, Table 4 details the hyperparameters that must be set for the XGBoost algorithm [54,55].

Table 3. Hyperparameters for an LSTM model.

Parameter	Description
Number of units	Number of LSTM memory cells used to store temporal information.
Number of hidden layers	Number of layers of LSTM units in the network.
Number of epochs	Determines how often the machine learning algorithm cycles through the training data set.
Batch size	Number of samples used per gradient update: a larger batch size can speed up the training process but requires more memory.
Learning rate	Represents the size of the steps taken by the algorithm to fit the model weights during training.
Optimizer	Method used to update weights during training.
Activation function	Type of nonlinear activation function.
Regularization	Technique used to prevent overfitting by eliminating neurons in the LSTM layers.

Table 4. Hyperparameters for an XGBoost model.

Parameter	Description
Learning rate	Determines the shrinkage step size.
Max leaves	Sets the limit on the number of nodes to be added.
Max depth	Determines the maximum levels each tree can have.
Min child weight	Sets the minimum weight required for instances in a leaf.
Gamma	Sets the minimum loss reduction needed for further partitioning.
Subsample	Represents the ratio of features/columns used for fitting each tree.
Column sample by tree	Determines the ratio of features/columns used for fitting each tree.

3.3. Model Evaluation

Evaluating a model is essential in developing any ML algorithm; therefore, this paper proposes using four metrics to measure and analyze the model's performance on the LSTM-RNN and XGBoost algorithms to determine its suitability for predicting internal temperature in the greenhouse.

The proposed metrics are as follows: R^2 (Equation (18)), RMSE (Equation (19)), MAE (Equation (20)), and MAPE (Equation (21)).

$$R^2 = 1 - \frac{RSS}{TSS} = 1 - \frac{\sum (y_i - \hat{y}_1)^2}{\sum (y_i - \bar{y}_1)^2} \quad (18)$$

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_1)^2} \quad (19)$$

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_1| \quad (20)$$

$$MAPE = \frac{1}{n} \sum_{i=1}^n \frac{|y_i - \hat{y}_1|}{y_i} \quad (21)$$

where y_i represents the actual value observed, \hat{y}_1 is the value predicted by the model, and n is the total number of observations in the data set.

R^2 measures how much the independent variables affect the dependent variable in terms of variance proportion. It uses the Residual Sum of Squares (RSS) and Total Sum of Squares (TSS). The RMSE is derived from the Mean Square Error (MSE) to standardize their units of measurement. The MSE measures variance by how well a model fits the training data. The RMSE is helpful because it gives more importance to specific data points, significantly impacting overall error if a prediction is incorrect. MAE evaluates the distance between the regressor and real points. The MAE does not heavily penalize outliers due to its norm that smooths out all errors, which provides a generic and bounded performance measure for the model. MAPE is used when variations impact the estimate more than the absolute values. However, this metric is biased toward low forecasts and is therefore unsuitable for evaluating tasks where errors of significant magnitude are expected [56,57].

3.4. Development

As part of the TDSP methodology, the development stage involves constructing and training the machine learning model. This stage includes the following:

- **Analyze models:** The LSTM-RNN and XGBoost supervised learning techniques were used to predict the internal temperature of the greenhouse. Both models were designed to forecast temperatures hourly to maintain a $\pm 2^\circ\text{C}$ hysteresis.
- **Data split:** The initial data set was divided into four seasons of the year. Moreover, the data were split into 80% for training and 20% for testing.
- **Model construction:** An analysis was conducted to determine the optimal combination of input variables and obtain the best response in predicting internal temperature. This was based on three input and one output variable (many-to-one). In addition, a study was conducted to identify the hyperparameters that can enhance the models' performance.
- **Model experimentation:** An experimental setup has been designed to determine the number of experiments required and the prediction time window. For this purpose, eight experiments were performed, corresponding to the four seasons of the year, using two proposed algorithms: LSTM-RNN and XGBoost. The prediction window chosen for these experiments is one hour, which allows for capturing internal temperature changes and making decisions based on the predictions in the greenhouse.
- **Model validation:** The R^2 , RMSE, MAE, and MAPE metrics were used to validate the performance of the predictions. In addition, the prediction graphs were displayed to analyze and determine whether there is an overfit or underfit.

4. Results

4.1. Model Construction

4.1.1. Select Input Variables

Section 3.1 presented a correlation analysis that identified the combination of variables for predicting the internal temperature of a greenhouse. In order to find the best combination, a comprehensive analysis is conducted using the entire year's data, encompassing all four seasons. Tables 5 and 6 display these results for the XGBoost and LSTM-RNN algorithms, respectively.

The best combination analysis resulted in internal humidity, internal dew point, and solar radiation (Hi-Di-Rs) as the best combination for forecasting results for both models, XGBoost and LSTM-RNN, respectively. This combination in XGBoost resulted in an $R^2 = 0.9189$, RMSE = 3.9018, and MAE = 2.6149, whereas LSTM-RNN reached $R^2 = 0.9994$, RMSE = 0.3003, and MAE = 0.0095.

Table 5. Comparison of forecasts using different input variable combinations for XGBoost.

Combination	R ²	RMSE	MAE
Hi-Di-To	0.8411	5.2970	3.9513
Hi-Ho-To	0.8483	5.1754	3.8196
Hi-To-Rs	0.9037	4.1225	2.7973
Di-Rs-To	0.9103	3.9797	2.6819
Ho-To-Rs	0.9047	4.1017	2.7585
Hi-Di-Rs	0.9189	3.9018	2.6149
Hi-Di-Ho	0.8596	4.9783	3.8024
Hi-Rs-Ho	0.9064	4.0640	2.7786
Di-Rs-Ho	0.9124	3.9318	2.7274
Di-Ho-To	0.8451	5.2304	3.8550

Table 6. Comparison of forecasts using different input variable combinations for LSTM-RNN.

Combination	R ²	RMSE	MAE
Hi-Di-To	0.9993	0.3128	0.0123
Hi-Ho-To	0.9762	1.8531	0.0451
Hi-To-Rs	0.9182	3.4352	0.0783
Di-Rs-To	0.8734	4.2746	0.0730
Ho-To-Rs	0.8043	5.3143	0.0853
Hi-Di-Rs	0.9994	0.3003	0.0095
Hi-Di-Ho	0.9989	0.3966	0.0099
Hi-Rs-Ho	0.7856	5.5629	0.3228
Di-Rs-Ho	0.9857	1.4368	0.0471
Di-Ho-To	0.9223	3.3477	0.0730

4.1.2. Hyperparameters

In the context of XGBoost, Bayesian optimization is used to identify the best hyperparameters for the models, due to its ability to identify them. It efficiently and effectively explores the search space using prior knowledge about the objective function. Unlike traditional methods, Bayesian optimization adapts and adjusts as more data are obtained. This helps to reduce the number of model evaluations and allows for a more focused search, resulting in optimal model performance in less time.

XGBoost hyperparameters like learning rate and max depth have consistent behavior across all seasons, while others vary seasonally. This is shown in Table 7.

Table 7. Best hyperparameters for XGBoost.

Season	Learning Rate	Max Depth	Reg Alpha	Reg Lambda	Gamma
Fall	0.3	2	0.2848	1	0.6529
Winter	0.3	2	1×10^{-9}	1	0
Spring	0.3	2	1	0.5071	1
Summer	0.3	2	0.8582	1×10^{-9}	1

The LSTM-RNN model also employed Bayesian optimization to identify the optimal hyperparameters. However, it was observed that the values of each parameter were similar across all stations. Therefore, it was decided to use the same parameters in all experiments conducted on the LSTM-RNN model. Table 8 presents these hyperparameters.

Table 8. Best hyperparameters for LSTM-RNN.

Parameter	Value
Input size	3
Hidden units	250
Epochs	300
Batch size	620
Gradient threshold	0.8
Learning rate	0.005
Drop factor	125
Optimizer	Adam

4.2. Model Experimentation

The LSTM-RNN and XGBoost algorithms provide better approximations to forecast the internal temperature in greenhouses over a long period. These algorithms are designed to handle non-linear data and adapt to changing patterns, such as climate variables, over time. Experiments were conducted with both algorithms to analyze their response to predictions one hour in advance, in which the high temporal resolution is achieved, enabling quick decisions and adjustments to be made in the greenhouse and proposing preventive control for better temperature management in greenhouses.

The data set was divided by seasons, leading to eight experiments for one-hour predictions. These experiments considered the four seasons of the year and the two algorithms proposed. The independent variables included internal humidity, internal dew, and solar radiation (Hi-Di-Rs), whereas the dependent variable was the internal temperature.

4.3. Model Evaluation

4.3.1. XGBoost

Figure 7 and Table 9 present the results of using the XGBoost algorithm to predict the internal temperature of a greenhouse one hour in advance.

This algorithm achieved similar results compared to R^2 ; however, a single metric cannot evaluate a model's performance. Hence, combining metrics for this analysis demonstrated that XGBoost delivered stable and favorable results for temperature forecasting. It is worth noting that the summer season (Figure 7d) exhibited the most unfavorable metrics because of constant temperature fluctuations, such as temperature spikes above 50 °C. In contrast, winter (Figure 7b) yielded the best results, while fall (Figure 7a) and spring (Figure 7c) produced similar results.

Table 9. Evaluation metrics for XGBoost model.

Season	R^2	RMSE	MAE	MAPE
Fall	0.9078	3.8297	2.3289	0.0906
Winter	0.9345	4.4443	2.7693	0.0809
Spring	0.9083	3.6153	2.3391	0.0688
Summer	0.8605	4.0876	2.4037	0.0697

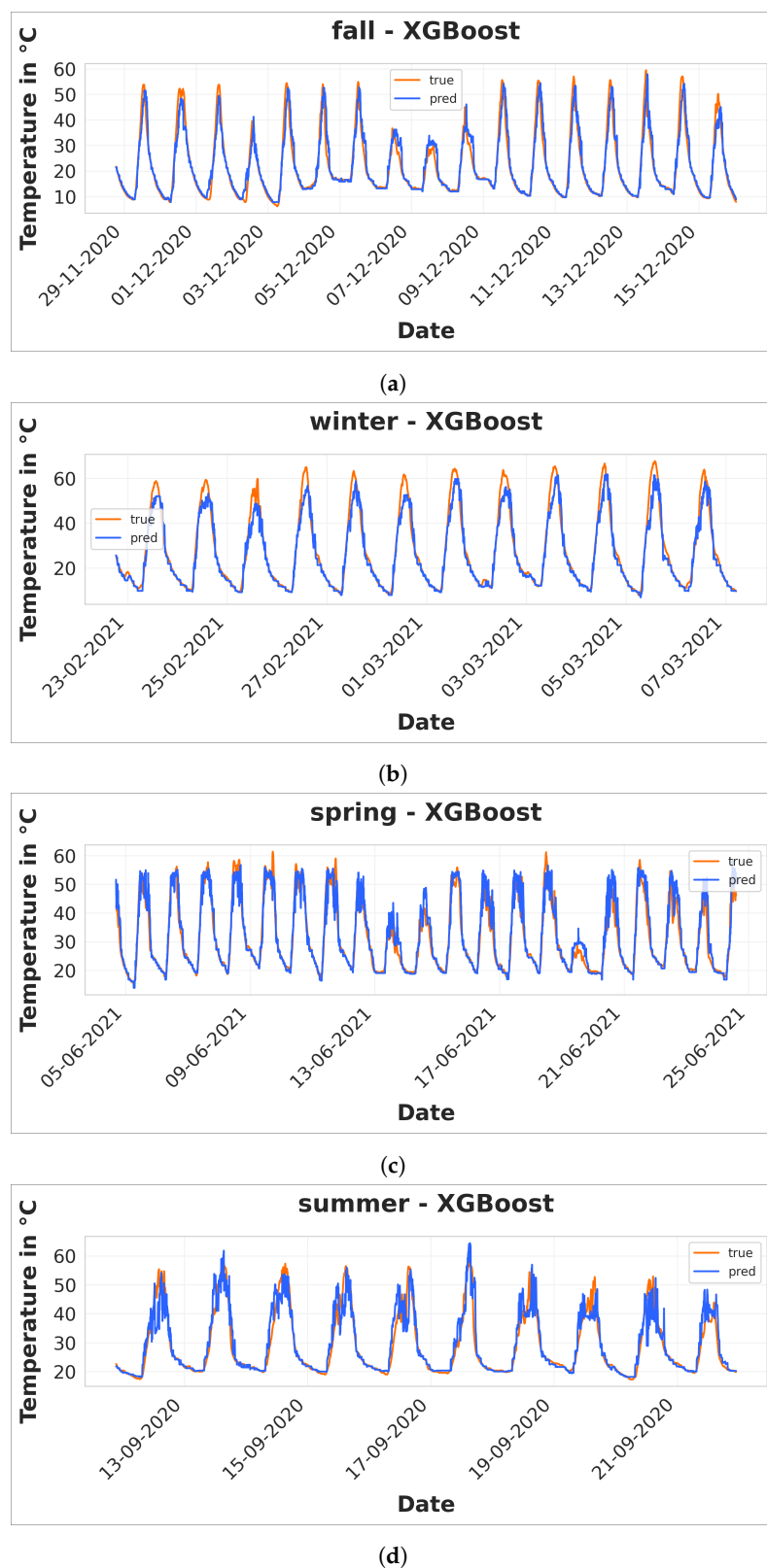


Figure 7. Results for one hour internal temperature prediction using XGBoost. Prediction for fall (a), winter (b), spring (c), and summer (d).

4.3.2. LSTM-RNN

Figure 8 and Table 10 show the outcomes obtained when employing the LSTM-RNN algorithm for forecasting the greenhouse's internal temperature with a one-hour lead time.

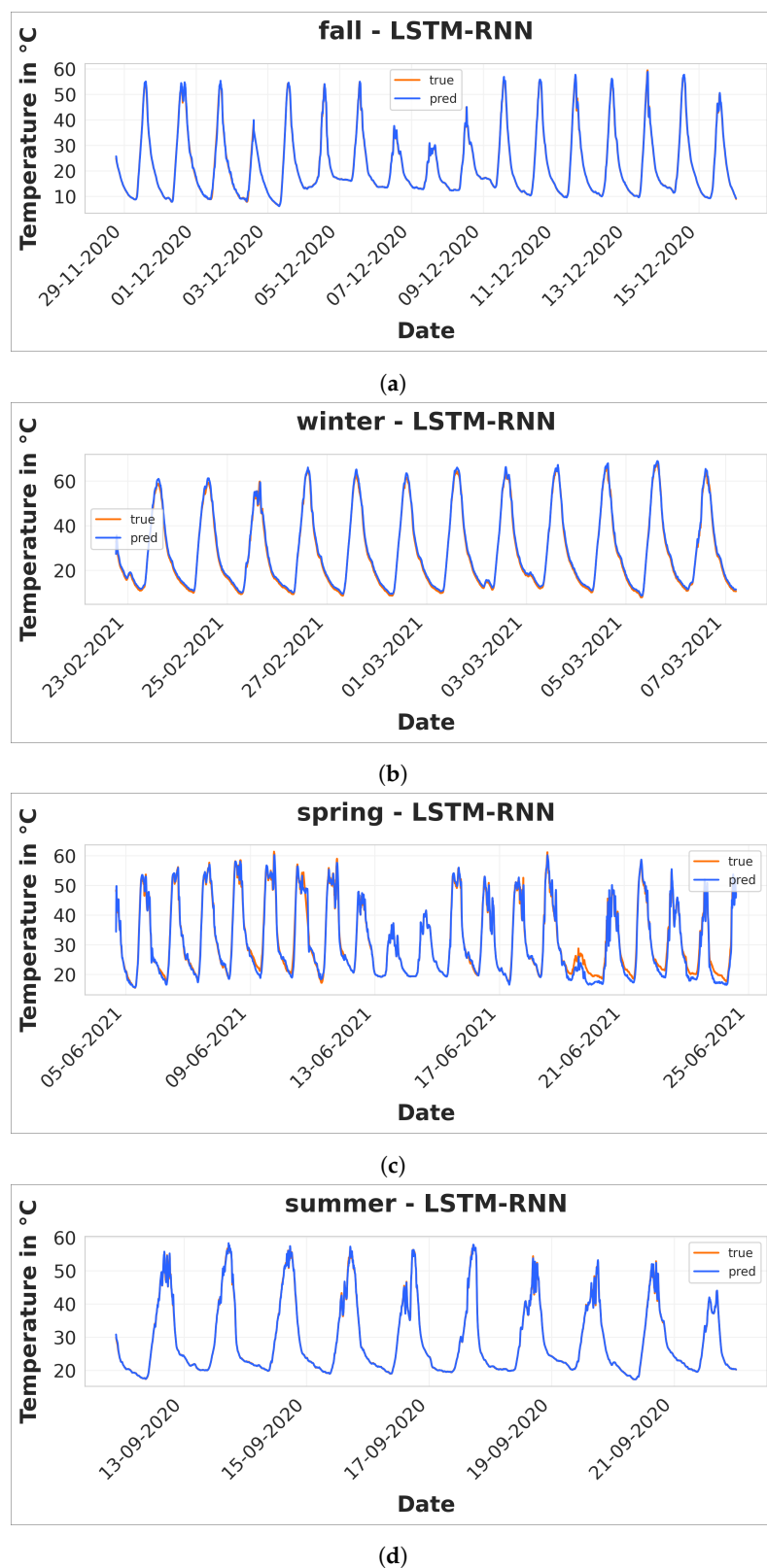


Figure 8. Results in internal temperature prediction using LSTM-RNN. Prediction for fall (a), winter (b), spring (c), and summer (d).

The LSTM-RNN model outperformed XGBoost in all seasons, with the most significant improvement observed during summer (Figure 8d), where XGBoost had the lowest R^2 among all experiments. Additionally, the MAE metric showed a hysteresis below 2 °C in

each season. These experiments demonstrate that the LSTM-RNN model is more effective than XGBoost for this particular application.

Table 10. Evaluation results for LSTM-RNN model.

Season	R ²	RMSE	MAE	MAPE
Fall	0.9969	0.6957	0.4031	0.0161
Winter	0.9947	1.2580	1.0624	0.0483
Spring	0.9784	1.7533	1.2004	0.0415
Summer	0.9994	0.2698	0.1449	0.0041

Figure 9 compares the metrics of the proposed models with those of the year's seasons, highlighting the differences.

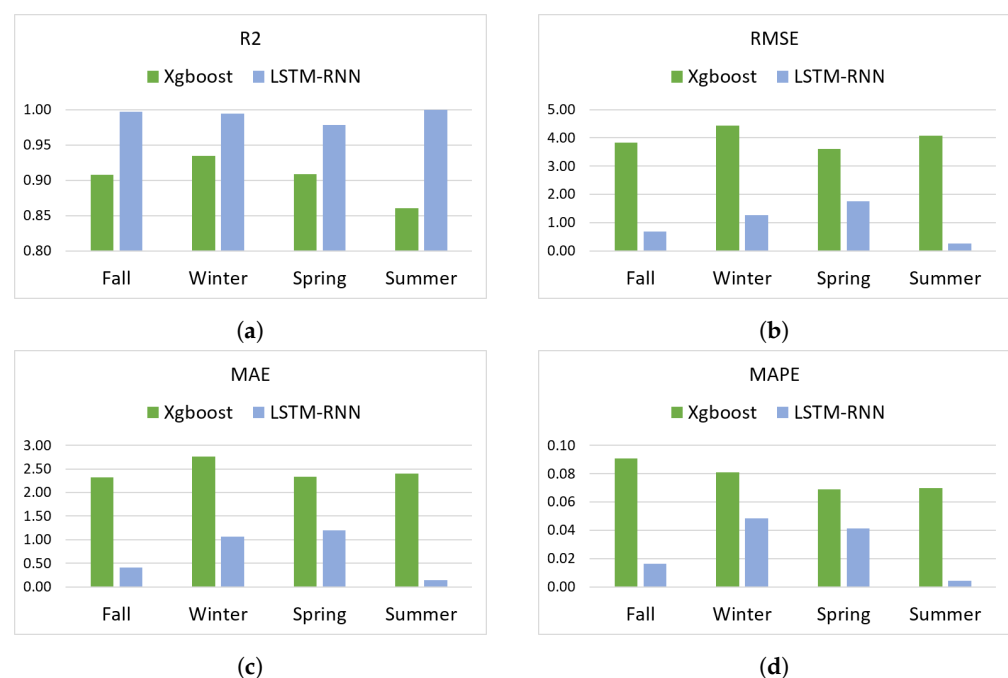


Figure 9. Analysis of metric results of the year's seasons and proposed models. R² (a), RMSE (b), MAE (c), and MAPE (d).

5. Discussion

In this study, two supervised learning models, namely XGBoost and LSTM-RNN, were utilized to predict the internal temperature of a greenhouse up to an hour in advance. The objective was maintaining an acceptable temperature range with a hysteresis of ± 2 °C. The MAE metric calculated the distance between the predicted and actual values. The results indicated that the XGBoost algorithm did not meet the acceptability criterion, as the range in all stations varied between 2.3 °C and 2.4 °C, surpassing the established ± 2 °C hysteresis. On the other hand, LSTM-RNN showed an MAE metric ranging from 0.4 °C to 1.2 °C, indicating that the acceptability criterion was met and the fall season yielded the best results.

In Section 2, a series of studies focused on predicting greenhouse variables. Each investigation had specific objectives, not all centered around making predictions in advance; some studies make predictions to detect patterns and find the best adjustments. This comparison is shown in Table 11.

Table 11. Comparison of studies presented in related works.

Work	Model	R ²	MAE	MAPE	RMSE	Advanced Prediction
Singh et al. [25]	ANN	0.98	0.558	-	0.7	24 h
Hongkangal et al. [26]	BP-RNN	0.953	0.421	-	0.751	N/A
Codeluppi et al. [28]	ANN	0.96	-	0.491	1.5	N/A
Wu et al. [34]	CNN-LSTM	-	2.573	5.316	3.864	5–60 min
Jung et al. [37]	LSTM-RNN	0.84	-	-	0.764	30 min
García-Vázquez et al. [22]	SVM	0.9998	0.0422	0.0015	0.0549	N/A
	Polynomial					
Esparza-Gomez et al.	LSTM-RNN	0.9994	0.1449	0.0041	0.2698	1 h

The study by García-Vázquez et al. uses the same database as the study being discussed. However, it takes a different approach to predicting the internal temperature of a greenhouse using regression methods such as the polynomial SVR algorithm. The results obtained show an R² of 0.9998 and an MAE of 0.0422. On the other hand, Codeluppi et al. used ANN to predict air temperature with an R² of 0.96 and MAPE of 0.49. Meanwhile, Hongkangal et al. combined RNN and BP to predict internal temperature, achieving a model with an R² of 0.95 and an MAE of 0.42.

The studies mentioned use a forward prediction approach instead of relying on future or anticipated information in the modeling process. This approach involves predicting the future value of a variable of interest at the next time step based solely on historical data available up to the time of the prediction. The absence of advanced information in the prediction process ensures that the obtained results represent the model's ability to make accurate and reliable predictions in real time without prior knowledge of future events or data. Therefore, the high R² values and low errors, such as MAE and MAPE, in these studies indicate the effectiveness of the models in making accurate predictions without relying on advanced information. However, depending on the system type, these predictions may not be as valuable as preventive automatic control models for greenhouses.

The work by Wu et al. used the CNN-LSTM algorithm to predict wind energy, a starting point to analyze how the prediction behaves in variables other than internal temperature. They used different steps to make predictions ranging from 5 to 60 min, with an average MAE of 2.573 across all steps. The research conducted by Jung et al. focuses on predicting internal humidity using LSTM-RNN. Their metrics showed an R² value greater than 0.8, making it the most consistent study with our research. However, it should be noted that their predictions were made using a different variable type, with a time frame of 30 min in advance. On the other hand, Singh et al. used ANN to predict the air temperature inside a greenhouse. They obtained an R² of 0.98 and an MAE of 0.558 in their study, which was conducted with a 24-hour in advance prediction. Compared to these studies, our LSTM-RNN approach has demonstrated outstanding performance in predicting the internal greenhouse temperature with a lead time of one hour. The evaluation metrics, R² = 0.9994, MAE = 0.1449, MAPE = 0.0041, and RMSE = 0.2698, indicate a good fit of the model to the greenhouse data. These results suggest that our model has accurate and reliable predictive ability, validating the claim that the approach exhibits excellent fit in predicting the temperature in the greenhouse with a lead time of one hour.

6. Conclusions

The focus of this research was to predict the internal temperature of a greenhouse up to an hour in advance. Supervised learning algorithms, such as LSTM-RNN and XGBoost, were utilized to generate models capable of establishing preventive control of greenhouse conditions. The database used in this research included internal variables such as temperature, humidity, and dew point, as well as external variables such as temperature, humidity, and solar radiation. This paper generated a methodology for constructing models to which machine learning was applied. Based on the LSTM-RNN and XGBoost algorithms, the best combination of input variables (internal humidity, internal

dew point, and solar radiation—Hi-Di-Rs) was found concerning the output variable (internal temperature—Ti). Bayesian optimization was used in each of the algorithms to analyze the best hyperparameters and apply them to the models. This model's construction led to eight experiments focused on the two algorithms and each year's season. The results were evaluated using the R^2 , RMSE, MAE, and MAPE metrics, which showed that LSTM-RNN presented better performance than XGBoost in all seasons. LSTM-RNN had the best result in the summer season with an R^2 value of 0.9994, while XGBoost had the lowest result in the summer season with an R^2 value of 0.8605.

Based on the prediction results of the LSTM-RNN and XGBoost algorithms, it is possible to develop a system that can accurately anticipate the internal conditions of a greenhouse up to an hour in advance. These models enable the control system to make proactive decisions and provide instructions to the various actuators in the greenhouse. However, several challenges impact the models' accuracy and the required computational resources when implementing predictive models based on these algorithms. Two fundamental limitations can be mentioned in particular. Firstly, as the forecasting capacity of the model increases, there is a corresponding increase in computational cost. Secondly, prediction errors such as RMSE and MAE tend to increase when projecting towards extended periods. Therefore, it is essential to balance the model's predictive capacity and the available computational resources to ensure optimal performance.

Author Contributions: Conceptualization, L.F.L.-V., M.A.C.-M. and R.C.-N.; methodology, R.C.-N. and M.A.C.-M.; software, J.M.E.-G., M.E.M.-R. and C.A.O.-O.; validation, H.A.G.-O., L.F.L.-V., M.A.C.-M. and F.G.-V.; formal analysis, J.M.E.-G., F.G.-V. and M.E.M.-R.; investigation, J.M.E.-G., H.A.G.-O. and L.O.S.-S.; resources, M.A.C.-M. and R.C.-N.; data curation, J.M.E.-G.; writing—original draft preparation, J.M.E.-G. and L.F.L.-V.; writing—review and editing, L.F.L.-V., F.G.-V. and H.A.G.-O.; visualization, J.M.E.-G., C.A.O.-O. and H.A.G.-O.; supervision, L.F.L.-V., M.A.C.-M. and R.C.-N.; project administration, H.A.G.-O. and F.G.-V.; funding acquisition, M.A.C.-M., L.O.S.-S. and R.C.-N. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Publicly available datasets were analyzed in this study. This data can be found here: <https://github.com/fabiangarciauaz/GreenhouseData>.

Acknowledgments: The authors want to thank the Mexican National Council of Science and Technology CONACYT for its support to the National Laboratory of Embedded Systems, Advanced Electronics Design and Micro systems (LN-SEDEAM by its initials in Spanish), project numbers 282357, 293384, 299061, 314841, 315947, and 321128, and also for the scholarship number 1012274.

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

Abbreviations

The following abbreviations are used in this manuscript:

AI	Artificial Intelligence
ANN	Artificial Neural Networks
ARIMA	AutoRegressive Integrated Moving Average
BP	Back Propagation
BD	Big Data
CO ₂	Carbon Dioxide
CSA	Climate Smart Agriculture
CC	Cloud Computing
R^2	Coefficient of Determination
CNN	Convolutional Neural Networks
DT	Decision Trees
DNN	Dense Neural Network

ET	Evapotranspiration
To	External temperature
Ho	External humidity
XGBoost	Extreme Gradient Boosting
FC	Fog Computing
FAO	Food and Agriculture Organization of the United Nations
GPS	Global Positioning System
GBM	Gradient Boosting Machine
ICT	Information and Communication Technologies
Di	Internal dew Point
Hi	Internal Humidity
Ti	Internal temperature
IoT	Internet of Things
KNN	K-Nearest Neighbor
LightGBM	Light Gradient Boosting Machine
LR	Linear Regression
LSTM-RNN	Long Short-Term Memory Recurrent Neural Network
ML	Machine Learning
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
MSE	Mean Square Error
MV-LSTM	Multivariate Long Short-Term Memory
NN	Neural Networks
NARX	Nonlinear Autoregressive Exogenous Model
PSO	Particle Swarm Optimization
RF	Random Forest
RNN	Recurrent Neural Networks
RSS	Residual Sum of Squares
RMSE	Root Mean Squared Error
Rs	Solar radiation
SF	Smart Farming
STCM	Spatio-Temporal Correlation Model
SGB	Stochastic Gradient Boosting
SVM	Support Vector Machines
TDSP	Team Data Science Process
TCN	Temporal Convolutional Network
TSS	Total Sum of Squares
XGBR	XGBoost Regression

References

1. United Nations, Department of Economic and Social Affairs, Population Division. *World Population Prospects 2019: Highlights*; United Nations: New York, NY, USA, 2019.
2. Ayaz, M.; Ammad-Uddin, M.; Sharif, Z.; Mansour, A.; Aggoune, E.H.M. Internet-of-Things (IoT)-based smart agriculture: Toward making the fields talk. *IEEE Access* **2019**, *7*, 129551–129583.
3. Mizik, T. Climate-smart agriculture on small-scale farms: A systematic literature review. *Agronomy* **2021**, *11*, 1096.
4. Quarshie, P.T.; Abdulai, S.; Fraser, E.D.G. (Re)assessing Climate-Smart Agriculture practices for sustainable food systems outcomes in sub-Saharan Africa: The case of Bono East Region, Ghana. *Geogr. Sustain.* **2023**, *4*, 112–126.
5. Mutengwa, C.S.; Mnkeni, P.; Kondwakwenda, A. Climate-Smart Agriculture and Food Security in Southern Africa: A Review of the Vulnerability of Smallholder Agriculture and Food Security to Climate Change. *Sustainability* **2023**, *15*, 2882.
6. Janc, K.; Czapiewski, K.; Wójcik, M. In the starting blocks for smart agriculture: The internet as a source of knowledge in transitional agriculture. *NJAS-Wagening. J. Life Sci.* **2019**, *90*, 100309.
7. Nie, J.; Yang, B. A detailed study on GPS and GIS enabled agricultural equipment field position monitoring system for smart farming. *Scalable Comput. Pract. Exp.* **2021**, *22*, 171–181.
8. Kalyani, Y.; Collier, R. A systematic survey on the role of cloud, fog, and edge computing combination in smart agriculture. *Sensors* **2021**, *21*, 5922.
9. Ciruela-Lorenzo, A.M.; Del-Aguila-Obra, A.R.; Padilla-Meléndez, A.; Plaza-Angulo, J.J. Digitalization of agri-cooperatives in the smart agriculture context. proposal of a digital diagnosis tool. *Sustainability* **2020**, *12*, 1325.

10. Ouammi, A.; Achour, Y.; Dagdougui, H.; Zejli, D. Optimal operation scheduling for a smart greenhouse integrated microgrid. *Energy Sustain. Dev.* **2020**, *58*, 129–137.
11. Achour, Y.; Ouammi, A.; Zejli, D. Technological progresses in modern sustainable greenhouses cultivation as the path towards precision agriculture. *Renew. Sustain. Energy Rev.* **2021**, *147*, 111251.
12. Zhou, Y.; Xia, Q.; Zhang, Z.; Quan, M.; Li, H. Artificial intelligence and machine learning for the green development of agriculture in the emerging manufacturing industry in the IoT platform. *Acta Agric. Scand. Sect. B—Soil Plant Sci.* **2022**, *72*, 284–299.
13. Bourechak, A.; Zedadra, O.; Kouahla, M.N.; Guerrieri, A.; Seridi, H.; Fortino, G. At the Confluence of Artificial Intelligence and Edge Computing in IoT-Based Applications: A Review and New Perspectives. *Sensors* **2023**, *23*, 1639. [\[PubMed\]](#)
14. Ramirez-Asis, E.; Bhanot, A.; Jagota, V.; Chandra, B.; Hossain, M.S.; Pant, K.; Almashaqbeh, H.A. Smart logistic system for enhancing the farmer-customer corridor in smart agriculture sector using artificial intelligence. *J. Food Qual.* **2022**, *2022*, 7486974.
15. Murlidharan, S.; Shukla, V.K.; Chaubey, A. Application of Machine Learning in Precision Agriculture using IoT. In Proceedings of the 2021 2nd International Conference on Intelligent Engineering and Management (ICIEM), London, UK, 28–30 April 2021; pp. 34–39.
16. Akkem, Y.; Biswas, S.K.; Varanasi, A. Smart farming using artificial intelligence: A review. *Eng. Appl. Artif. Intell.* **2023**, *120*, 105899.
17. Saleem, M.H.; Potgieter, J.; Arif, K.M. Automation in agriculture by machine and deep learning techniques: A review of recent developments. *Precis. Agric.* **2021**, *22*, 2053–2091.
18. Shehadeh, A.; Alshboul, O.; Al Mamlook, R.E.; Hamedat, O. Machine learning models for predicting the residual value of heavy construction equipment: An evaluation of modified decision tree, LightGBM, and XGBoost regression. *Autom. Constr.* **2021**, *129*, 103827.
19. Gowthaman, T.; Adarsh, V.S.; Kumar, K.S.; Manobharathi, K. A Review of Deep Learning Models for Price Prediction in Agricultural Commodities. *Econ. Aff.* **2023**, *68*, 561–568.
20. Maraveas, C. Incorporating artificial intelligence technology in smart greenhouses: Current State of the Art. *Appl. Sci.* **2022**, *13*, 14.
21. Iddio, E.; Wang, L.; Thomas, Y.; McMorro, G.; Denzer, A. Energy efficient operation and modeling for greenhouses: A literature review. *Renew. Sustain. Energy Rev.* **2020**, *117*, 109480.
22. García-Vázquez, F.; Ponce-González, J.R.; Guerrero-Osuna, H.A.; Carrasco-Navarro, R.; Luque-Vega, L.F.; Mata-Romero, M.E.; Martínez-Blanco, M.d.R.; Castañeda-Miranda, C.L.; Díaz-Flórez, G. Prediction of Internal Temperature in Greenhouses Using the Supervised Learning Techniques: Linear and Support Vector Regressions. *Appl. Sci.* **2023**, *13*, 8531.
23. Ahmed, D.M.; Hassan, M.M.; Mstafa, R.J. A Review on Deep Sequential Models for Forecasting Time Series Data. *Appl. Comput. Intell. Soft Comput.* **2022**, *2022*, 6596397.
24. Zhong, Z.; Lv, S.; Shi, K. A New Method of Time-Series Event Prediction Based on Sequence Labeling. *Appl. Sci.* **2023**, *13*, 5329.
25. Singh, V.K.; Tiwari, K.N. Prediction of greenhouse micro-climate using artificial neural network. *Appl. Ecol. Environ. Res* **2017**, *15*, 767–778.
26. Hongkang, W.; Li, L.; Yong, W.; Fanjia, M.; Haihua, W.; Sigrimis, N.A. Recurrent neural network model for prediction of microclimate in solar greenhouse. *IFAC-PapersOnLine* **2018**, *51*, 790–795.
27. Yu, J.; Zheng, W.; Xu, L.; Zhangzhong, L.; Zhang, G.; Shan, F. A PSO-XGBoost Model for Estimating Daily Reference Evapotranspiration in the Solar Greenhouse. *Intell. Autom. Soft Comput.* **2020**, *26*, 989–1003.
28. Codeluppi, G.; Cilfone, A.; Davoli, L.; Ferrari, G. Ai at the edge: A smart gateway for greenhouse air temperature forecasting. In Proceedings of the 2020 IEEE International Workshop on Metrology for Agriculture and Forestry (MetroAgriFor), Trento, Italy, 4–6 November 2020; pp. 348–353.
29. Gharghory, S.M. Deep network based on long short-term memory for time series prediction of microclimate data inside the greenhouse. *Int. J. Comput. Intell. Appl.* **2020**, *19*, 2050013.
30. Jung, D.H.; Kim, H.S.; Jhin, C.; Kim, H.J.; Park, S.H. Time-serial analysis of deep neural network models for prediction of climatic conditions inside a greenhouse. *Comput. Electron. Agric.* **2020**, *173*, 105402.
31. Moon, T.; Son, J.E. Knowledge transfer for adapting pre-trained deep neural models to predict different greenhouse environments based on a low quantity of data. *Comput. Electron. Agric.* **2021**, *185*, 106136.
32. Gong, L.; Yu, M.; Jiang, S.; Cutsuridis, V.; Pearson, S. Deep learning based prediction on greenhouse crop yield combined TCN and RNN. *Sensors* **2021**, *21*, 4537.
33. Xie, A.; Yang, H.; Chen, J.; Sheng, L.; Zhang, Q. A short-term wind speed forecasting model based on a multi-variable long short-term memory network. *Atmosphere* **2021**, *12*, 651.
34. Wu, Q.; Guan, F.; Lv, C.; Huang, Y. Ultra-short-term multi-step wind power forecasting based on CNN-LSTM. *IET Renew. Power Gener.* **2021**, *15*, 1019–1029.
35. Ge, J.; Zhao, L.; Yu, Z.; Liu, H.; Zhang, L.; Gong, X.; Sun, H. Prediction of greenhouse tomato crop evapotranspiration using XGBoost machine learning model. *Plants* **2022**, *11*, 1923.
36. Cai, W.; Wei, R.; Xu, L.; Ding, X. A method for modelling greenhouse temperature using gradient boost decision tree. *Inf. Process. Agric.* **2022**, *9*, 343–354.
37. Jung, D.H.; Lee, T.S.; Kim, K.; Park, S.H. A deep learning model to predict evapotranspiration and relative humidity for moisture control in tomato greenhouses. *Agronomy* **2022**, *12*, 2169.

38. Venkatesan, S.; Lim, J.; Ko, H.; Cho, Y. A machine learning based model for energy usage peak prediction in smart farms. *Electronics* **2022**, *11*, 218.
39. Cao, Q.; Wu, Y.; Yang, J.; Yin, J. Greenhouse Temperature Prediction Based on Time-Series Features and LightGBM. *Appl. Sci.* **2023**, *13*, 1610.
40. Ajani, O.S.; Usigbe, M.J.; Aboyeji, E.; Uyeh, D.D.; Ha, Y.; Park, T.; Mallipeddi, R. Greenhouse micro-climate prediction based on fixed sensor placements: A machine learning approach. *Mathematics* **2023**, *11*, 3052.
41. Li, K.Q.; Kang, Q.; Nie, J.Y.; Huang, X.W. Artificial neural network for predicting the thermal conductivity of soils based on a systematic database. *Geothermics* **2022**, *103*, 102416.
42. Li, K.Q.; Liu, Y.; Kang, Q. Estimating the thermal conductivity of soils using six machine learning algorithms. *Int. Commun. Heat Mass Transf.* **2023**, *136*, 106139.
43. Li, K.Q.; Yin, Z.Y.; Liu, Y. A hybrid SVR-BO model for predicting the soil thermal conductivity with uncertainty. *Can. Geotech. J.* **2023**. [[CrossRef](#)]
44. Li, K.Q.; Yin, Z.Y.; Zhang, N.; Liu, Y. A data-driven method to model stress-strain behaviour of frozen soil considering uncertainty. *Cold Reg. Sci. Technol.* **2023**, *213*, 103906.
45. What Is the Team Data Science Process (TDSP)? Available online: <https://learn.microsoft.com/en-us/azure/architecture/data-science-process/overview> (accessed on 21 September 2023).
46. Koc, K.; Ekmekcioglu, O.; Gurgun, A.P. Integrating feature engineering, genetic algorithm and tree-based machine learning methods to predict the post-accident disability status of construction workers. *Autom. Constr.* **2021**, *131*, 103896.
47. Thi Kieu Tran, T.; Lee, T.; Shin, J.Y.; Kim, J.S.; Kamruzzama, M. Deep Learning-Based Maximum Temperature Forecasting Assisted with Meta-Learning for Hyperparameter Optimization. *Atmosphere* **2020**, *11*, 487.
48. Meng, Y.; Yang, N.; Qian, Z.; Zhang, G. What makes an online review more helpful: An interpretation framework using XGBoost and SHAP values. *J. Theor. Appl. Electron. Commer. Res.* **2020**, *16*, 466–490.
49. Liew, X.Y.; Hameed, N.; Clos, J. An investigation of XGBoost-based algorithm for breast cancer classification. *Mach. Learn. Appl.* **2021**, *6*, 100154.
50. Chen, T.; Guestrin, C. Xgboost: A scalable tree boosting system. In Proceedings of the 22nd ACM Sigkdd International Conference on Knowledge Discovery and Data Mining, Association for Computing Machinery, San Francisco, CA, USA, 13–17 August 2016; pp. 785–794.
51. Putatunda, S.; Rama, K. A Modified Bayesian Optimization based Hyper-Parameter Tuning Approach for Extreme Gradient Boosting. In Proceedings of the 2019 Fifteenth International Conference on Information Processing (ICINPRO), Bengaluru, India, 20–22 December 2019; pp. 1–6.
52. Sewak, M.; Sahay, S.K.; Rathore, H. Assessment of the Relative Importance of different hyper-parameters of LSTM for an IDS. In Proceedings of the 2020 IEEE Region 10 Conference (Tencon), Osaka, Japan, 16–19 November 2020; pp. 414–419.
53. Bouktif, S.; Fiaz, A.; Ouni, A.; Serhani, M.A. Multi-sequence LSTM-RNN deep learning and metaheuristics for electric load forecasting. *Energies* **2020**, *13*, 391.
54. Putatunda, S.; Rama, K. A comparative analysis of hyperopt as against other approaches for hyper-parameter optimization of XGBoost. In Proceedings of the 2018 International Conference on Signal Processing and Machine Learning, Aalborg, Denmark, 17–20 November 2018; pp. 6–10.
55. Wang, Y.; Ni, X.S. A XGBoost risk model via feature selection and Bayesian hyper-parameter optimization. *arXiv* **2019**, arXiv:1901.08433.
56. Chicco, D.; Warrens, M.J.; Jurman, G. The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation. *PeerJ Comput. Sci.* **2021**, *7*, e623.
57. Uddin, M.G.; Nash, S.; Diganta, M.T.M.; Rahman, A.; Olbert, A.I. Robust machine learning algorithms for predicting coastal water quality index. *J. Environ. Manag.* **2022**, *321*, 115923.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.