agronomy

MDPI

Article

Edible Mushroom Greenhouse Environment Prediction Model Based on Attention CNN-LSTM

Shuanggen Huang [1], Quanyao Liu [1], Yan Wu [1], Minmin Chen [2], Hua Yin [3,*] and Jinhui Zhao [1,*]

[1] Key Laboratory of Modern Agricultural Equipment in Jiangxi Province, Jiangxi Agricultural University, Nanchang 330045, China; shuanggenhuang1979@jxau.edu.cn (S.H.); 630620474@stu.jxau.edu.cn (Q.L.); wuyan1981@jxau.edu.cn (Y.W.)
[2] Department of Marine Engineering, Shanwei Institute of Technology, Shanwei 516600, China; 13560552877@163.com
[3] School of Software, Jiangxi Agricultural University, Nanchang 330045, China
* Correspondence: yinhua@jxau.edu.cn (H.Y.); zhaojh@jxau.edu.cn (J.Z.)

Abstract: The large-scale production of edible mushrooms typically requires the use of greenhouses, as the greenhouse environment significantly affects the growth of edible mushrooms. It is crucial to effectively predict the temperature, humidity, and carbon dioxide fluctuations within the mushroom greenhouse for determining the environmental stress and pre-regulation of edible mushrooms. To address the nonlinearity, temporal dynamics, and strong coupling of the edible mushroom greenhouse environment, a temperature, humidity, and carbon dioxide prediction model based on the combination of the attention mechanism, the convolutional neural network, and the long short-term memory neural network (A-CNN-LSTM) is proposed. Experimental data were collected from both the inside and outside of the greenhouse, including environmental data and the on–off data of environmental control devices. After completing missing data using linear interpolation, denoising with Kalman filtering, and normalization, the recurrent neural network (RNN) model, long short-term memory (LSTM) model, and A-CNN-LSTM model were trained and tested on the time series data. These models were used to predict the environmental changes in temperature, humidity, and carbon dioxide inside the greenhouse. The results indicate that the A-CNN-LSTM model outperforms the other two models in terms of denoising, non-denoising, and different prediction time steps. The proposed method accurately predicts temperature, humidity, and carbon dioxide levels with errors of 0.17 °C (R^2 = 0.974), 2.06% (R^2 = 0.804), and 8.367 ppm (R^2 = 0.993), respectively. These results indicate improved prediction accuracy for temperature, humidity, and carbon dioxide values inside the edible mushroom greenhouse. The findings provide a decision basis for the precise control of the greenhouse environment.

Keywords: deep learning; edible mushroom greenhouse; greenhouse environment prediction; attention mechanism; A-CNN LSTM model

Citation: Huang, S.; Liu, Q.; Wu, Y.; Chen, M.; Yin, H.; Zhao, J. Edible Mushroom Greenhouse Environment Prediction Model Based on Attention CNN-LSTM. *Agronomy* **2024**, *14*, 473. https://doi.org/10.3390/agronomy14030473

Academic Editor: Gniewko Niedbała

Received: 7 February 2024
Revised: 24 February 2024
Accepted: 25 February 2024
Published: 27 February 2024

1. Introduction

The main environmental factors that influence the growth of edible mushrooms include temperature, humidity, and carbon dioxide levels [1–4]. Stable greenhouse conditions are crucial for the large-scale cultivation of edible mushrooms. However, traditional greenhouses can only monitor the current greenhouse environment, and there is a lag issue in environmental control devices in terms of environmental regulation [5,6]. Currently, the mainstream focus of greenhouse environmental control is primarily on temperature and humidity. Therefore, it is necessary to establish an accurate greenhouse microclimate model to determine the operating status of environmental control devices in advance.

Traditional physical modeling prediction methods require the collection of multiple complex data, which often involve calculations and are difficult to estimate. The environmental factors inside the greenhouse exhibit nonlinearity and temporal characteristics, and

there is coupling between these factors [7–11]. Therefore, traditional physical modeling encounters certain difficulties. Deep learning methods can address this issue as these models can accurately predict the internal greenhouse environment for the next few minutes based on historical data [12–14]. Better prediction of greenhouse temperature and humidity was achieved by combining CNN with GRU compared with that achieved with the BP neural network, LSTM, and GRU. CNN is prone to damaging the integrity of time series [15]. Jung et al. have solved the issue of gradient explosion in RNN by combining RNN with LSTM. The RNN-LSTM model performed the best in predicting greenhouse environments, as determined by comparing the RNN-LSTM model with the ANN and NARX models. However, the RNN-LSTM model was complex and had poor prediction performance for humidity [16]. Qi Liu et al. utilized the Elman neural network to predict the temperature and humidity of different canopies in crops. The predicted RMSE values of temperature and humidity were less than 0.8 °C and 1.5%, respectively. This model employed the gradient descent method to solve the loss function, which was difficult to achieve global optimization under [17]. The present study proposes an A-CNN-LSTM prediction model suitable for greenhouses to address the issues mentioned above. To the best of our knowledge, there is no study on the establishment of a prediction model that combines edible mushroom greenhouse environmental data both inside and outside with on–off data of environmental control equipment.

In this study, a large amount of environmental data from sensors inside and outside the edible mushroom greenhouse, as well as on–off data from control devices, were collected. The collected data were processed using linear interpolation for missing data, then denoising with Kalman filtering, and normalization. Three prediction models, RNN, LSTM, and A-CNN-LSTM, were established using time series inputs. The models were evaluated by comparing the prediction results before and after denoising, the different time steps for prediction, the different sample sizes for prediction, and the different input features for prediction.

2. Design of Data Collection Platform and Data Processing

2.1. Design of Data Collection Platform

The edible mushroom greenhouse used in this experiment was a closed space constructed with fully enclosed metal insulation panels, measuring 130 × 60 × 90 cm in length, width, and height. The top of the greenhouse was equipped with holes connected to a humidifier, while the side had an air conditioning vent. Additionally, a metal mushroom placement rack was installed inside the greenhouse. The environmental information inside and outside the greenhouse was measured using sensors from Shandong Renke Control Technology Co., Ltd. Jinan, China. These sensors included a temperature and humidity carbon dioxide multi-sensor (model RS-CO2WS-N01) and an illuminance sensor (model RS-GZ-N01). These sensors were capable of accurately measuring temperature, humidity, carbon dioxide levels, and light intensity to provide comprehensive environmental data for the experiment. All sensors and environmental control actuators were connected to a smart gateway (STM32F104). The sensors transmitted the environmental information from inside and outside the greenhouse to the gateway using the Modbus protocol. The gateway then utilized a 4G signal to transmit the data to the server. The server analyzed the environmental data and transmitted control signals back to the gateway. This allowed the gateway to control the humidifier and air conditioning system to keep the greenhouse environment within the desired threshold. Additionally, the server received status updates from the environmental control devices through the gateway. This integrated system enabled the real-time monitoring and control of the greenhouse environment, ensuring that it remained within the specified parameters for optimal mushroom cultivation.

2.2. Data Acquisition

The experiment took place in Nanchang, Jiangxi Province. The designed greenhouse was used for data collection from August 2022 to October 2022. Inside the greenhouse, there were two sets of environmental data sensors. They collected data on air temperature

(TEMP1; TEMP2), relative humidity (HUM1; HUM2), and carbon dioxide levels ($CO_2$1; $CO_2$2) inside the greenhouse. Outside the greenhouse, there was another set of environmental data sensors that collected data on outside air temperature (TEMP(out)) and outside air relative humidity (HUM(out)). Additionally, data on the on–off status of the humidifier (HUMBtn) and air conditioning (TEMPBtn) were also collected. The data collection interval for all these sensors was 1 min, and the data were collected continuously for 24 h. The collected data were sent from the gateway to the server. The interior and exterior images of the data collection platform are shown in Figure 1. The structural diagram of the data collection system is shown in Figure 2.

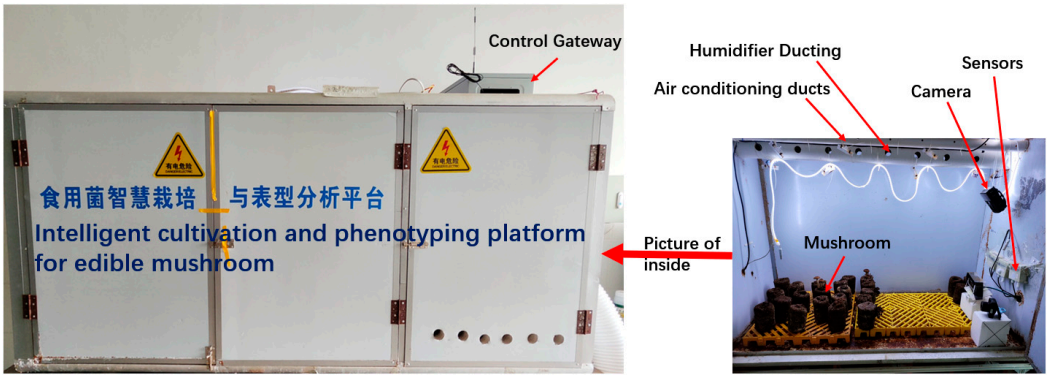


Figure 1. Intelligent cultivation and phenotyping platform for edible mushroom interior and exterior images. The blue Chinese in the figure means Intelligent cultivation and phenotyping platform for edible mushroom.

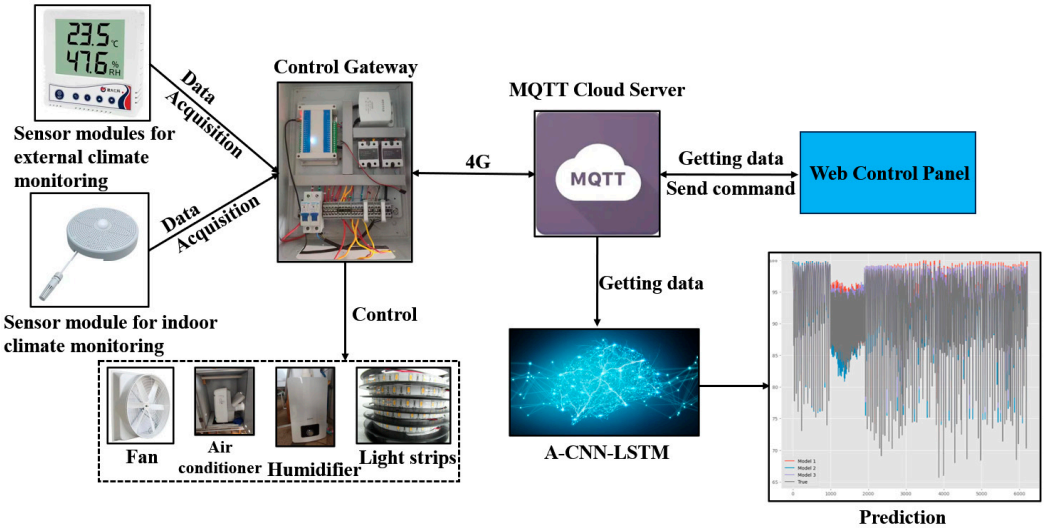


Figure 2. Structure diagram of edible mushroom environmental data collection platform.

2.3. Data Processing

2.3.1. Processing of Missing Data

Due to the high-humidity environment in the greenhouse, there were issues with data loss during the data collection process [18]. In addition, the system utilized a 4G signal to transmit data to the server. However, data may be affected by issues such as network transmission quality or device malfunctions. To address this problem, linear interpolation was employed in this study for data imputation [19]. When short-term data gaps occur, linear interpolation is used to calculate the missing data values. Specifically, for a given

time point, t, if the temperature data are missing, the following formula can be used to estimate the temperature data at that time $T(t)$:

$$T(t) = \frac{T(k) + (T(k+1) - T(k))}{(t(k+1) - t(k))(t - t(k))} \quad (1)$$

where (k) represents the position of the timestamp in the temperature data before the (k) segment; $(T(k))$ and $(T(k + 1))$ represent the temperature values in the (t) and $(t + 1)$ segments, respectively.

Similarly, for a given time point (t), if humidity and carbon dioxide data are missing, linear interpolation can also be used to estimate the data at that time. The principle of linear interpolation is based on using the linear relationship between known data points to estimate the missing values. It is advantageous due to its simplicity and fast computation speed, but it may introduce errors in cases of signal discontinuities or noise. In the case of long-term data gaps, another set of sensor data is used as a substitute.

2.3.2. Data Denoising

Noise handling was considered during the design of the system. This involved the use of appropriate sensors and the incorporation of suitable protective and shielding devices. Additionally, long-distance transmission from the gateway to the server utilized a 4G signal and the increase in the system's data acquisition frequency. However, due to the high-humidity environment in mushroom greenhouses and the influence of environmental control equipment, the indoor environment is prone to fluctuations, resulting in some degree of noise interference. Therefore, in the data processing stage, the Kalman filter can be employed to reduce the impact of noise on the data [20].

The Kalman filter is based on a linear dynamic system model and utilizes state estimation methods. At each time point, it combines the current observation with the previously estimated state history to make predictions and obtain future state estimates. This approach, which consists of two main stages, the prediction stage and the update stage, allows for the estimation of the future state based on the available information. In the prediction stage, a physical model is used to predict the possible value and variance in the next state. In the update stage, the Bayesian theorem is utilized to combine the prior estimate with the observed information, resulting in a merged posterior estimate and its corresponding variance.

Specifically, the state vector prediction formula is as follows:

$$X(k \mid k-1) = AX(k-1 \mid k-1) + BU(k) \quad (2)$$

The covariance matrix prediction formula is as follows:

$$P(k \mid k-1) = AP(k-1 \mid k-1)A^T + Q \quad (3)$$

The state vector update formula is as follows:

$$X(k \mid k) = X(k \mid k-1) + K(k)[Z(k) - HX(k \mid k-1)] \quad (4)$$

The covariance matrix update formula is as follows:

$$P(k \mid k) = (I - K(k)H)P(k \mid k-1) \quad (5)$$

The Kalman gain formula is as follows:

$$K(k) = P(k \mid k-1)H^T \left(HP(k \mid k-1)H^T + R \right)^{-1} \quad (6)$$

In the equations above, $X(k \mid k-1)$ is the state estimate at time k, A is the state transition matrix, B is the control matrix, $U(k)$ is the control input at time k, $P(k \mid k-1)$ is the correlation between the state vector and the covariance matrix, Q is the process noise

covariance matrix, $X(k \mid k)$ is the updated state estimate at time k, H is the measurement matrix, $Z(k)$ is the measurement value at time k, $P(k \mid k)$ is the updated covariance matrix, and R is the measurement noise covariance matrix.

The Kalman filter method can handle time series temperature and humidity data, reducing the interference of random noise on data variations, thus improving data quality and accuracy. When the characteristics of the signal and noise are known, and the system is linear and stationary, the Kalman filter often performs very well [21]. Figure 3 shows the comparison graphs of temperature, humidity and CO_2 data before and after denoising.

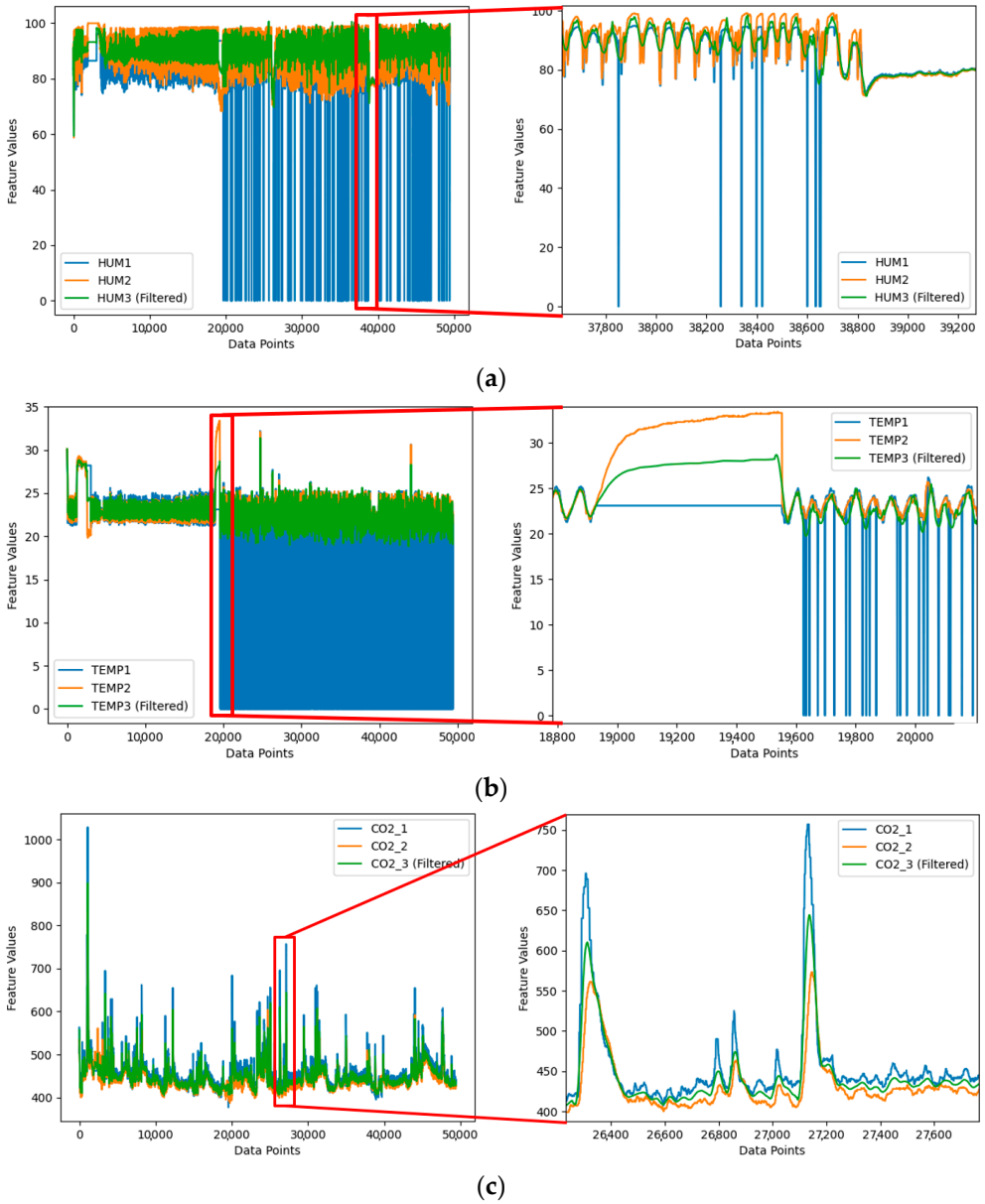


Figure 3. Comparison of humidity and temperature in greenhouse before and after denoising. (**a**) HUM1, HUM2, and filtered HUM3 features. It can be seen that denoising removes the effect of missing sensor values. (**b**) TEMP1, TEMP2, and filtered TEMP3 features. It can be seen that denoising removes the effects of sensor outliers and missing values. (**c**) CO_2_1, CO_2_2, and filtered CO_2_3 features. Visibly denoised data combine information from multiple sensors for more accurate data.

2.3.3. Data Normalization

Normalization is an important step in data pre-processing that scales data of different sizes to a range, making the distribution of the data more reasonable and playing an

important role in data analysis and mining [22]. Min–Max normalization is a method of scaling data to the interval [0, 1] and is also a common method of data normalization. For data sets with stable features and no obvious outliers, this method allows the range of values of different features to be unified on the same scale, while preserving the original data distribution information [23].

The formula for the min-max normalization is as follows:

$$x_{new} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{7}$$

where x is the original value and x_{new} is the normalized result. x_{new} and x_{max} are the minimum and maximum values of the range of values taken by the feature, respectively.

3. Neural Network Prediction Models for Greenhouse

3.1. Recurrent Neural Network Model

Recurrent Neural Network is a powerful neural network that is suitable for handling sequence data. Compared to conventional neural networks and Fully Connected Neural Networks. RNN can handle sequences of arbitrary lengths and introduces the concept of recurrence. RNN is primarily used in fields such as time series analysis, speech recognition, and natural language processing [24]. A recurrent neural network consists of a trainable feedback loop that passes the current input and the hidden state from the previous time step to the next time step. This allows the model to selectively retain temporal information and facilitates learning on large amounts of data. The hidden states store past data that is necessary for generating sequential signals and can be used to make predictions about the future based on this information.

Figure 4 illustrates the recurrent principle and unfolded model of an RNN during computation. In this diagram, U represents the weights from the input layer to the hidden layer, while V represents the weights from the hidden layer to the output layer. Unlike conventional neural networks, the values in the hidden layer of an RNN are not only determined by the current input but also influenced by the hidden state obtained from the previous computation. In practice, the set weights W use the hidden state obtained from the previous computation as the input for the current time step, enabling the network to retain information from the past.

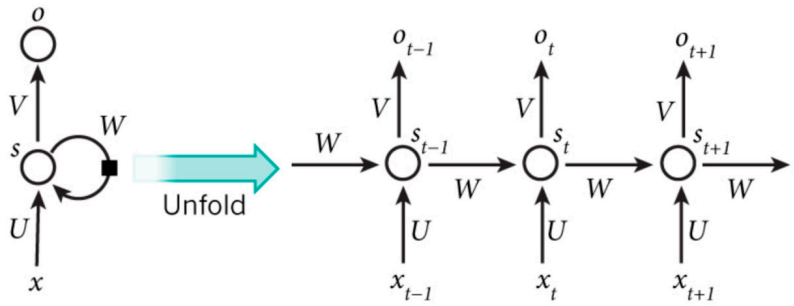


Figure 4. RNN structure diagram.

3.2. LSTM Neural Network Model

Standard recurrent neural networks often encounter issues such as vanishing or exploding gradients when processing long time sequences, particularly in handling dependencies between sequences. To address this problem, Hochreiter and Schmidhuber proposed the LSTM (Long Short-Term Memory) algorithm in 1997 [25]. LSTM, as a specialized type of RNN, is designed for handling sequential data and has shown superior performance in modeling long-term dependencies compared to other RNN architectures. Due to its excellent performance, LSTM has been widely applied in various fields, including natural language processing, speech recognition, video analysis and more.

Unlike standard RNNs, LSTM introduces gate mechanisms that allow cells to selectively receive information from neurons at different time steps, enhancing their ability to understand and predict long-term sequences. The core idea of LSTM is to weigh the input using three gate mechanisms: the input gate, forget gate, and output gate. These gate mechanisms allow useful information to pass through while blocking irrelevant information. Therefore, LSTM can retain important information and discard irrelevant information. The structure of an LSTM memory cell is illustrated in Figure 5.

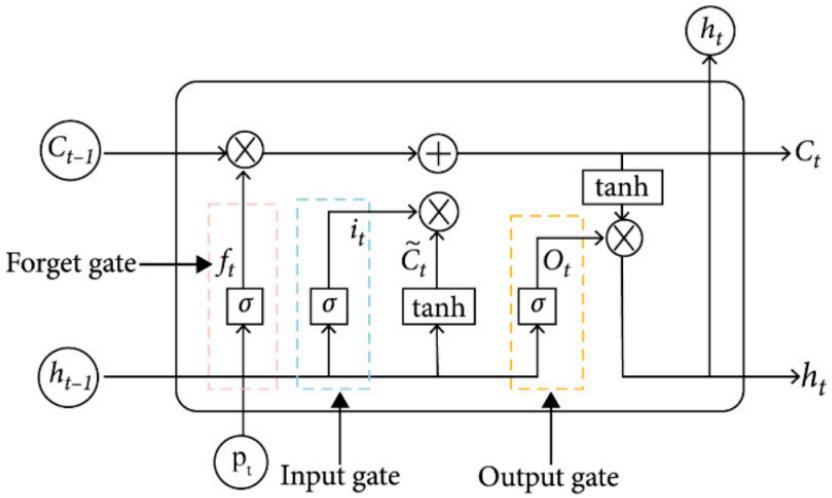


Figure 5. LSTM neural network structure diagram.

The forget gate is responsible for selectively discarding information from the previous long-term memory cell. It determines how to modify and delete records based on the network's input node X_t and the previous hidden state H_{t-1}. The relevant calculation formulas are as follows:

$$f_t = \sigma\left(W_f \cdot [H_{t-1}, X_t] + b_f\right) \tag{8}$$

where, f_t denotes the forget gate, σ denotes the sigmoid activation function, X_t denotes the input at time step t, H_{t-1} denotes the output of the short-term memory at time step $t-1$, W_f denotes the weight matrix for the forget gate and b_f denotes the bias term for the forget gate.

The input gate determines the new content that needs to be added to the memory based on the network's input node X_t and H_{t-1}. The relevant calculation formulas are as follows:

$$i_t = \sigma(W_i \cdot [H_{t-1}, X_t] + b_i) \tag{9}$$

$$\widetilde{C}_t = \tanh(W_c \cdot [H_{t-1}, X_t] + b_c) \tag{10}$$

$$C_t = f_t \times C_{t-1} + i_t \times \widetilde{C}t \tag{11}$$

where, i_t denotes the input gate, $\widetilde{C}_t$ denotes the candidate vector, tanh denotes the hyperbolic tangent activation function, C_t denotes the long-term memory cell at time step t, W_i and W_c denote the weight parameters for the input gate and the candidate vector, respectively, and b_i and b_c denote the corresponding bias terms.

The output gate determines the output value H_t based on the long-term memory cell C_t and the input nodes X_t and H_{t-1}. The calculation formula is as follows:

$$ot = \sigma(Wo \cdot [Ht-1, Xt] + bo) \tag{12}$$

$$Ht = ot \times \tanh(Ct) \tag{13}$$

where, *ot* denotes the output gate, *Wo* denotes the weight matrix for the output gate, *bo* denotes the bias term for the output gate, and *Ht* denotes the output of the model at time step *t*.

Overall, LSTM neural networks excel at handling long sequential data compared to traditional recurrent neural networks [26]. The flexible architecture of LSTM allows them to effectively address a variety of problems related to processing time series data [27].

3.3. Attention Mechanism-Based CNN-LSTM Model

The attention mechanism-based CNN-LSTM model is essentially an addressing mechanism that can extract specific vectors from a set of vectors based on a query vector and combine them with weights. The scoring function can take various forms, such as additive, dot-product, and bilinear, while the importance allocation function indicates the attention level for each piece of information. This reflects the model's ability to selectively focus on key information.

In this paper, environmental data such as indoor temperature, indoor humidity, carbon dioxide level, outdoor temperature, outdoor humidity, humidifier switch state, and air conditioner switch state are inputted into the model. Firstly, local feature extraction is performed in the CNN network, and then the local features from the CNN network are provided to the LSTM network for time series modeling. To better capture long-term dependencies in the sequence, an attention mechanism is introduced to enhance the predictive power of the network. Specifically, after calculating the attention weights, they are applied to the hidden state vectors obtained from the LSTM encoder to focus on the parts of the hidden state that have stronger connectivity and higher importance. Finally, the time series features extracted from the CNN network and the LSTM outputs weighted by attention are concatenated to form the A-CNN-LSTM model, as shown in Figure 6. This model is able to dynamically adjust the importance of different input signals within the model, leading to more accurate predictions.

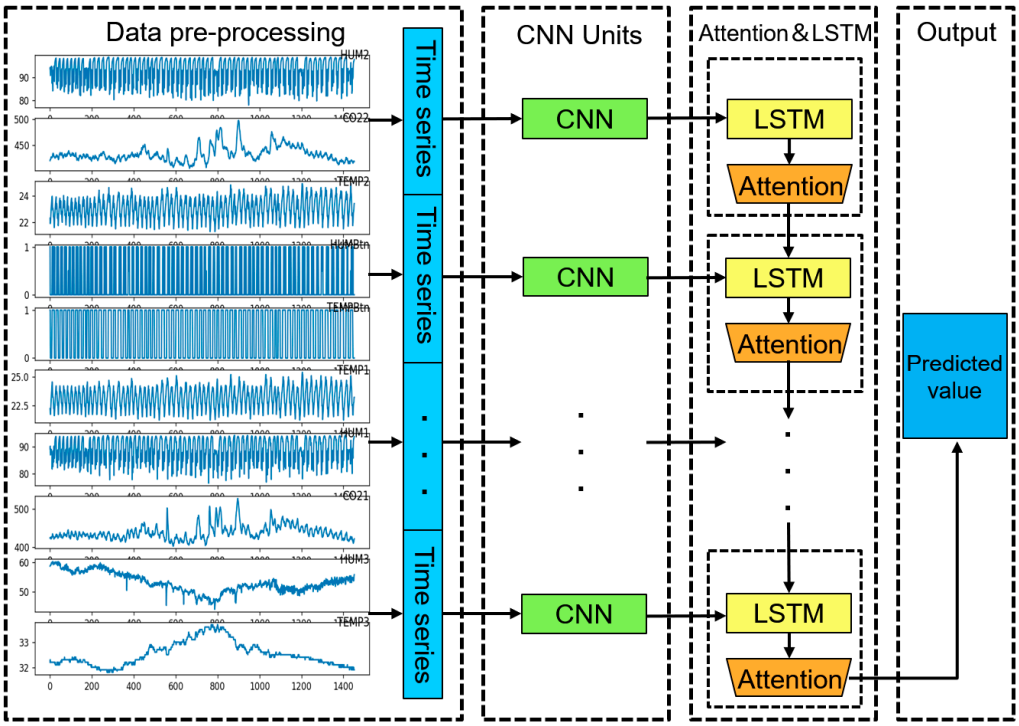


Figure 6. Attention Mechanism-Based CNN-LSTM Model.

3.4. Performance Evaluation

In this paper, two evaluation metrics, *RMSE* (Root Mean Squared Error) and R^2 (R-squared), are selected. These two metrics are commonly used to evaluate time series prediction models. *RMSE* is an important statistical measure that quantifies the difference between actual values and predicted values. It is calculated by taking the square root of the sum of squared errors for each sample point, divided by the number of samples. A smaller *RMSE* value indicates higher prediction accuracy. The formula for *RMSE* is as follows:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{14}$$

where, n denotes the number of samples, y_i denotes the actual value and $\hat{y}_i$ denotes the predicted value.

R^2 is a measure that represents the proportion of the variance in the dependent variable that can be explained by the independent variables. It is an important metric to assess the goodness-of-fit of a model. R^2 ranges from 0 to 1, where 1 indicates the best fit and 0 indicates no predictability or a horizontal line. A value closer to 1 indicates a better fit of the model. The formula for R^2 is as follows:

$$R^2 = 1 - \frac{SS_{residual}}{SS_{total}} \tag{15}$$

where, $SS_{residual}$ represents the sum of squared residuals, which is the total sum of squares of the differences between the actual values and predicted values for all samples. SS_{total} represents the total sum of squares, which is the total sum of squares of the differences between the actual values and the mean value for all samples. R^2 ranges from 0 to 1, where a value closer to 1 indicates a better fit of the model to unknown data. When R^2 equals 0, it means the model cannot explain any variation in the data, and when R^2 is less than 0, it indicates a very poor fit of the model.

4. Experimentation and Analysis

The computer configuration for this experiment was an Intel® Core™ i7-10700 CPU @ 2.9 GHz processor, 16 GB of RAM, and the Windows 10 (64-bit) operating system. The GPU used was the NVIDIA GeForce RTX 3060 Laptop. For deep learning tasks, TensorFlow 2.8.0 was employed as the deep learning framework. The integrated development environment (IDE) used was PyCharm Community Edition 2022.1.3. The programming language utilized for building the temperature and humidity prediction models was Python 3.10.

The experiment employed indoor temperature, indoor humidity, outdoor temperature, outdoor humidity, and the switch status of the air conditioner and humidifier as the dataset. The input feature vectors were normalized using Min-Max Normalization and the dataset was split into training and testing sets in a 9:1 ratio for training and testing purposes. The experiment was trained and tested with 100 epochs and a batch size of 64. Separate output layers were used to predict temperature, humidity, and CO_2, respectively.

4.1. The Effect of Denoising on Prediction Results

To compare the impact of denoising techniques on the prediction results, the data processed with the Kalman filter and the original data were separately input into three neural network models: RNN, LSTM, and A-CNN-LSTM. For each model, the same training and testing dataset were used, and a unified evaluation metric, namely RMSE (root mean squared error) and R^2 (R-squared), was employed.

The experimental results are presented in Table 1. Across all model types and data preprocessing methods, the denoised data preprocessing method outperformed the non-denoised method. This indicates that removing noise from the data can improve the prediction performance of greenhouse environmental data. Before denoising, the RNN model had an RMSE of 10.251 (R^2 = 0.901), which improved to 6.719 (R^2 = 0.903) after

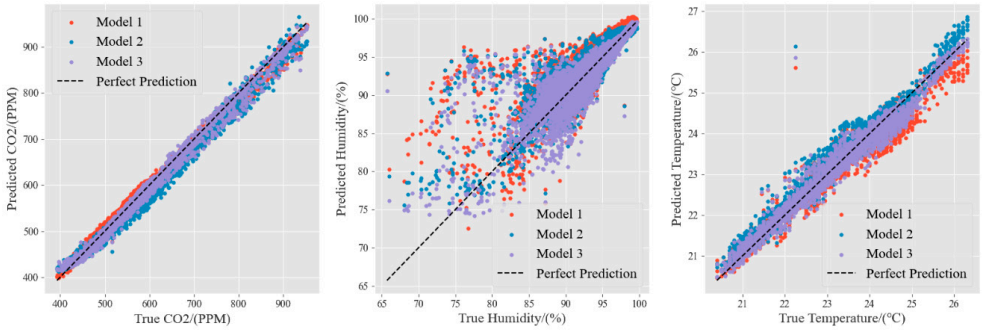
denoising. The LSTM model outperformed the RNN model before denoising, with an RMSE of 6.568 (R^2 = 0.874), and this improved to 5.596 (R^2 = 0.883) after denoising. Similarly, the CNN-LSTM model exhibited an RMSE of 7.868 (R^2 = 0.837) before denoising, which dropped to 5.05 (R^2 = 0.907) after denoising. Compared to the other two models, the A-CNN-LSTM model showed more significant improvement after denoising.

Table 1. Performance comparison of RNN, LSTM, and A-CNN-LSTM models before and after denoising.

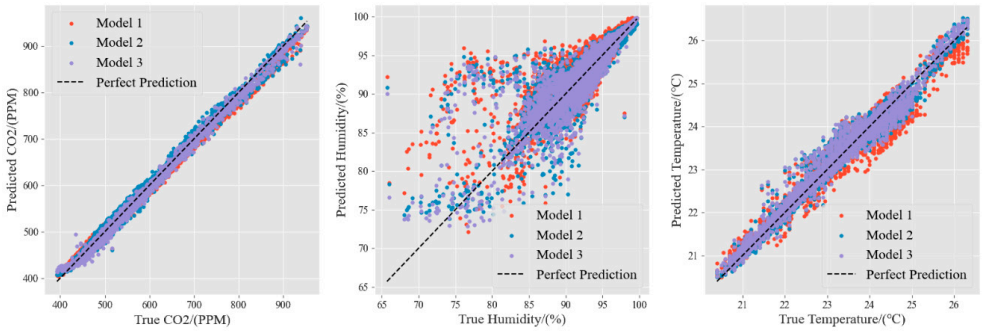
RNN	Network		Temp.		Humidity		CO_2	
	RMSE	**R^2**	**RMSE**	**R^2**	**RMSE**	**R^2**	**RMSE**	**R^2**
non-denoised	10.251	0.901	0.196	0.964	2.308	0.754	17.607	0.968
denoised	6.719	0.903	0.174	0.973	2.278	0.761	11.406	0.987
LSTM	Network		Temp.		Humidity		CO_2	
	RMSE	R^2	RMSE	R^2	RMSE	R^2	RMSE	R^2
non-denoised	6.568	0.874	0.232	0.952	2.661	0.673	14.715	0.978
denoised	5.596	0.883	0.22	0.957	2.535	0.704	9.416	0.991
A-CNN-LSTM	Network		Temp.		Humidity		CO_2	
	RMSE	R^2	RMSE	R^2	RMSE	R^2	RMSE	R^2
non-denoised	7.868	0.837	0.192	0.967	2.293	0.757	9.219	0.991
denoised	5.05	0.907	0.17	0.974	2.06	0.804	8.367	0.993

Compared to the case without denoising, it is evident that models after denoising can significantly reduce the RMSE and increase the R^2 value. Particularly, the A-CNN-LSTM model exhibits remarkably improved prediction performance after undergoing denoising.

Figure 7 provides a comparison of the prediction results for greenhouse meteorological data with denoising and without. (a) represents the results before denoising, while (b) represents the results after denoising. The results clearly show that the prediction results after denoising have better prediction accuracy compared to those without denoising.



(**a**) Prediction results for raw data input.



(**b**) Predicted results for denoised data inputs.

Figure 7. A comparison of temperature, humidity, and carbon dioxide prediction results before and after denoising. (**a**) The prediction results before denoising; (**b**) the prediction results after denoising. It can be seen that using the denoised data for the prediction brings the prediction closer to the true value.

The results demonstrate that in the processing of greenhouse meteorological data, various neural network models, when subjected to denoising, can significantly enhance prediction performance. Particularly, the A-CNN-LSTM model exhibited a remarkable improvement in performance.

4.2. Prediction Results at Different Time Steps

During the experiment comparing different time steps for prediction, the three models mentioned above were denoised using the same dataset. The results for predicting the time steps of 1 min, 5 min, 15 min, and 30 min were compared, and the performance of the models in terms of RMSE and R^2 was evaluated.

The experimental results are presented in Table 2. As the prediction cycle increases, the prediction errors of the models also increase. For a time step of 1 min, the A-CNN-LSTM model performs the best with an RMSE of 5.050 (R^2 = 0.907). For the time steps of 5 min, 15 min, and 30 min, the A-CNN-LSTM model still performs the best with the following RMSE and R2 values: 9.639 (R^2 = 0.887), 10.33 (R^2 = 0.883), and 10.76 (R^2 = 0.886), respectively. Smaller time steps can capture more instantaneous changes, but they may also introduce more noise and unnecessary details. On the other hand, larger time steps can reduce noise and details but may result in the loss of important instantaneous information. This experiment was conducted in a relatively small greenhouse, where instantaneous information was crucial and the data had been denoised. Therefore, it is necessary to select smaller time steps to ensure accuracy. The performance of neural network predictions tends to decline as the prediction time horizon increases. If long-term predictions are required, it is important to strike a balance between prediction accuracy and the desired prediction interval by selecting an appropriate time step. In this regard, the A-CNN-LSTM neural network generally exhibits better prediction performance.

Table 2. The performance comparison of RNN, LSTM, and A-CNN-LSTM models for predicting different time steps.

Predicted Time Step	RNN		LSTM		A-CNN-LSTM	
	RMSE	R^2	RMSE	R^2	RMSE	R^2
1 min	6.719	0.903	5.596	0.883	5.050	0.907
5 min	12.565	0.863	11.364	0.857	9.639	0.887
15 min	13.879	0.847	12.890	0.849	10.33	0.883
30 min	17.146	0.794	15.981	0.817	10.76	0.886

4.3. Prediction Results with Different Sample Sizes

During the comparative experiment with different training sample sizes, the three models were applied to the greenhouse dataset. The sample size gradually increased from 25,000 to 100,000, and predictions were made using a time step of 1 min. The experimental results are presented in Table 3, showing that as the training sample size increased, the prediction performance of all models significantly improved.

Table 3. Prediction results of A-CNN-LSTM with different training sample sizes.

Group Number	Sample Size	Training Time (s)	A-CNN-LSTM		HUM		TEMP		CO_2	
			RMSE	R^2	RMSE	R^2	RMSE	R^2	RMSE	R^2
Group 1	25,000	732	7.038	0.881	2.647	0.683	0.195	0.956	10.75	0.976
Group 2	50,000	1426	5.376	0.897	2.231	0.776	0.184	0.969	8.639	0.982
Group 3	75,000	2079	5.162	0.904	2.113	0.797	0.175	0.972	8.437	0.988
Group 4	100,000	2798	5.050	0.907	2.060	0.804	0.170	0.974	8.367	0.993

Taking the A-CNN-LSTM model as an example, when the training sample size is 25,000, the RMSE is 7.038 and the R^2 is 0.881. However, when the training sample size

is increased to 100,000, the RMSE decreases to 5.050 and the R^2 increases to 0.907. This clearly demonstrates that the sample size is crucial for the accuracy of the prediction model during the training phase. Therefore, in order to achieve the desired accuracy, it is important to collect as large an amount of high-quality training data as possible. Figure 8 shows the relationship between the R^2 values of various predicted values and time for the A-CNN-LSTM neural network under different training sample sizes. It can be observed that as the training sample size increases, the required training time significantly increases. However, except for humidity data, the R^2 values for the remaining data show slow growth. When the sample size increased from 50,000 to 75,000, the model's RMSE only decreased by 3.98%, while the R^2 increased by 0.7%. However, the training time increased by 353 s. Compared with the improvement in RMSE and R^2, the decrease in training efficiency is more significant. Therefore, in this study, the B dataset was selected as a representative for model training to enhance operational efficiency.

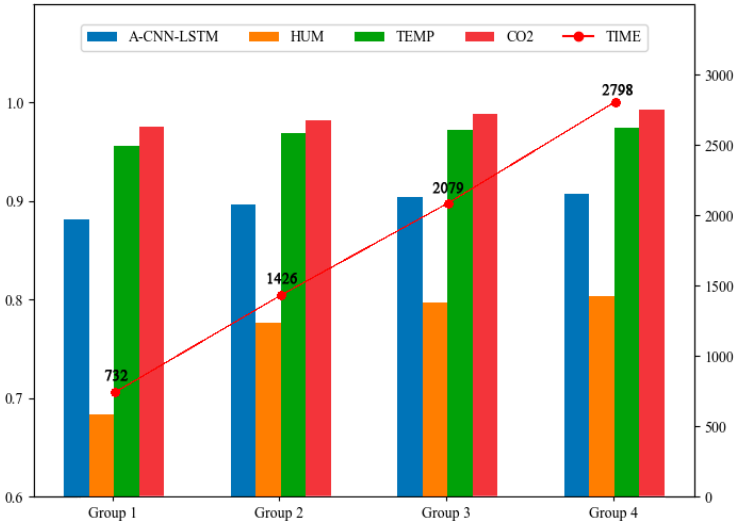


Figure 8. R^2 values and time for various predicted values of the A-CNN-LSTM neural network under different training sample sizes.

4.4. Prediction Results with Different Input Features

The small climatic environment changes in edible mushroom greenhouses are easily influenced by various factors. In order to optimize the greenhouse climate model, the Pearson correlation analysis method ($p < 0.05$) was used to determine the relevant influencing factors of temperature and humidity changes in edible mushroom greenhouses. Based on this, further optimization was conducted [28]. The analysis heatmap is shown in Figure 9. Based on the Pearson correlation analysis, the experiments were divided into three groups. The group 1 includes three features: indoor temperature, indoor humidity, and indoor CO_2 concentration. The group 2 includes five features: indoor temperature, indoor humidity, indoor CO_2 concentration, humidifier switch status, and air conditioner switch status. The group 3 includes seven features: indoor temperature, indoor humidity, indoor CO_2 concentration, outdoor temperature, outdoor humidity, humidifier switch status, and air conditioner switch status. To compare the impact of these three groups of feature combinations on predictive performance, we conducted experiments using the same dataset and inputted the respective feature combinations into the A-CNN-LSTM model. We then calculated the RMSE and R^2 values for each experiment, as well as the training time required for the models.

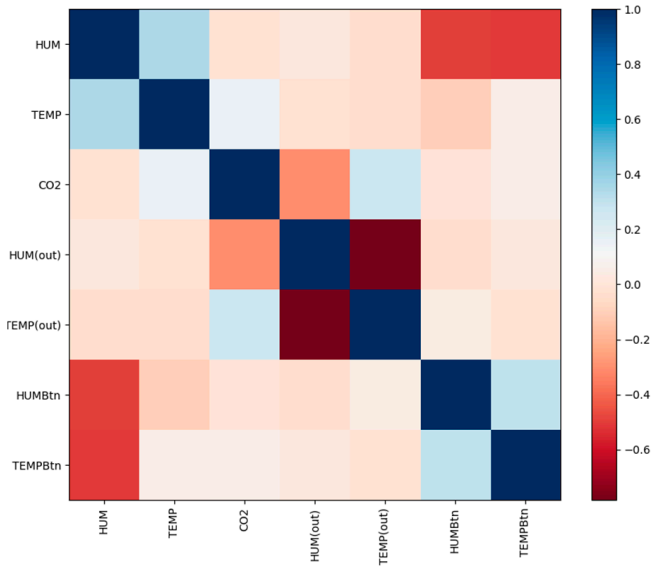


Figure 9. Pearson correlation analysis.

The experimental results are shown in Table 4, and indicate that the three different feature combinations have varying degrees of impact on predictive performance. For temperature prediction, the first group achieved an RMSE of 0.51 (R^2 = 0.87), the second group achieved an RMSE of 0.55 (R^2 = 0.85), and the third group achieved an RMSE of 0.68 (R^2 = 0.79). This indicates that considering more input features improves the accuracy of the prediction model when considering temperature variations.

Table 4. Effect of three different combinations of features on prediction performance.

Input Features	Training Time	Temp.		Humidity		CO_2	
		RMSE	R^2	RMSE	R^2	RMSE	R^2
Group 1	2749.741 s	0.510	0.870	2.450	0.810	8.375	0.993
Group 2	2431.537 s	0.550	0.850	2.500	0.800	8.465	0.992
Group 3	2107.421 s	0.680	0.790	3.080	0.740	8.752	0.989

In terms of humidity prediction, the first group achieved an RMSE of 2.450 (R^2 = 0.810), the second group achieved an RMSE of 2.500 (R^2 = 0.800), and the third group achieved an RMSE of 3.080 (R^2 = 0.740). It can be observed that incorporating outdoor temperature and humidity, as well as variables such as humidifier and air conditioner switch status, significantly improves the performance of the humidity prediction model compared with that of a model that only considers indoor temperature and humidity.

In terms of CO_2 prediction, the impact of different input features on the model's predictive performance is limited. The maximum difference in R^2 among the three groups is only 0.004, which is consistent with the results obtained from Pearson correlation analysis.

This experiment explored the impact of different feature combinations on temperature and humidity prediction in a greenhouse. It was found that incorporating more input features can improve predictive performance but may also increase the training time and instability of the model. Additionally, different feature combinations may have different effects on the performance of different types of models. Therefore, it is necessary to design the optimal feature combination based on the specific problem at hand.

5. Conclusions

A temperature, humidity, and carbon dioxide prediction model based on the combination of A-CNN-LSTM is proposed to address the nonlinearity, temporal dynamics, and strong coupling of the greenhouse environment for edible mushrooms. The experimental

results indicate that the neural network model's prediction accuracy and robustness for greenhouse temperature, humidity, and carbon dioxide can be improved by using a denoising process on the dataset. Among the different models, the dataset processed with denoising performs better and can improve the RMSE and R^2 metrics. As the prediction horizon increases, the prediction error of the models also increases, but the A–CNN–LSTM neural network still outperforms others. The training sample size is crucial for the accuracy of the model, requiring a large amount of high-quality training data. Different input features have varying impacts on the prediction performance, with more input features that are highly correlated leading to improved prediction accuracy.

Although this study achieved certain results in the prediction of microclimate in greenhouses, there are still several directions for further exploration and improvement. These include, for instance, obtaining greenhouse data from all four seasons and optimizing the model accordingly; conducting multi-objective optimization research by integrating the prediction of temperature and humidity in the greenhouse with energy optimization and crop growth objectives; addressing the real-time requirements of greenhouse environment control by studying how to combine the prediction model with actual control systems to achieve real-time greenhouse environment control.

Author Contributions: Conceptualization, S.H., H.Y. and Q.L.; methodology, S.H., J.Z. and Q.L.; software, S.H. and Q.L.; validation, S.H., Q.L. and H.Y.; formal analysis, J.Z.; investigation, Y.W.; resources, S.H.; data curation, Q.L.; writing—original draft preparation, Q.L.; writing—review and editing, S.H. and M.C.; visualization, Q.L.; supervision, H.Y.; project administration, J.Z. and M.C.; funding acquisition, S.H., J.Z. and M.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was sponsored by the National Natural Science Foundation of China (No. 31660485, 62362039), the Science and Technology Planning Project of Jiangxi Education Department (No. GJJ160350 and GJJ200447), and the Universities Characteristic Innovation Project in Natural Science of Guangdong Province (No. 2020KTSCX317).

Data Availability Statement: In this study, we used the datasets obtained from the Smart Cultivation and Phenotyping Platform for Edible Mushrooms of Jiangxi Agricultural University, and the codes for this paper and some of the datasets are stored at the following URL: https://github.com/yyao1016/a-cnn-lstm (6 February 2024).

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

References

1. Shamshiri, R.R.; Jones, J.W.; Thorp, K.R.; Ahmad, D.; Man, H.C.; Taheri, S. Review of optimum temperature, humidity, and vapour pressure deficit for microclimate evaluation and control in greenhouse cultivation of tomato: A review. *Int. Agrophys.* **2018**, *32*, 287–302. [CrossRef]
2. Li, Y.; Ding, Y.; Li, D.; Miao, Z. Automatic carbon dioxide enrichment strategies in the greenhouse: A review. *Biosyst. Eng.* **2018**, *171*, 101–119. [CrossRef]
3. Bellettini, M.B.; Fiorda, F.A.; Maieves, H.A.; Teixeira, G.L.; Ávila, S.; Hornung, P.S.; Júnior, A.M.; Ribani, R.H. Factors affecting mushroom *Pleurotus* spp. *Saudi J. Biol. Sci.* **2019**, *26*, 633–646. [CrossRef]
4. Sánchez, C. Cultivation of *Pleurotus ostreatus* and other edible mushrooms. *Appl. Microbiol. Biotechnol.* **2010**, *85*, 1321–1337. [CrossRef]
5. Tan, X.; Iyer, R.V. Modeling and control of hysteresis. *IEEE Control. Syst.* **2009**, *29*, 26–28.
6. Katsoulas, N.; Baille, A.; Kittas, C. Effect of misting on transpiration and conductances of a greenhouse rose canopy. *Agric. For. Meteorol.* **2001**, *106*, 233–247. [CrossRef]
7. Jarecke, K.M.; Loecke, T.D.; Burgin, A.J. Coupled soil oxygen and greenhouse gas dynamics under variable hydrology. *Soil Biol. Biochem.* **2016**, *95*, 164–172. [CrossRef]
8. Yue, W.; Liu, L.; Chen, S.; Bai, Y.; Li, N. Effects of Water and Nitrogen Coupling on Growth, Yield and Quality of Greenhouse Tomato. *Water* **2022**, *14*, 3665. [CrossRef]
9. Körner, O.; Challa, H. Process-based humidity control regime for greenhouse crops. *Comput. Electron. Agric.* **2003**, *39*, 173–192. [CrossRef]
10. 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. [CrossRef]

11. Ding, Y.; Wang, L.; Li, Y.; Li, D. Model predictive control and its application in agriculture: A review. *Comput. Electron. Agric.* **2018**, *151*, 104–117. [CrossRef]
12. Escamilla-García, A.; Soto-Zarazúa, G.M.; Toledano-Ayala, M.; Rivas-Araiza, E.; Gastélum-Barrios, A. Applications of Artificial Neural Networks in Greenhouse Technology and Overview for Smart Agriculture Development. *Appl. Sci.* **2020**, *10*, 3835. [CrossRef]
13. Fourati, F.; Chtourou, M. A greenhouse control with feed-forward and recurrent neural networks. *Simul. Model. Pract. Theory* **2007**, *15*, 1016–1028. [CrossRef]
14. 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. [CrossRef]
15. Zhao, Q.; Song, Z.; Li, Q.; Zheng, W.; Liu, Y.; Zhang, Z. Multi-point Prediction of Temperature and Humidity of Mushroom Based on CNN-GRU. *Trans. Chin. Soc. Agric. Mach.* **2020**, *51*, 294–303.
16. 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. [CrossRef]
17. Liu, Q.; Ta, N.; Jiao, W.; Tang, H.; Zhao, Z. Spatial Temporal Distribution and Prediction Model of Canopy Temperature and Humidity in Greenhouse. *North. Hortic.* **2019**, *17*, 56–65.
18. Sentelhas, P.C.; Gillespie, T.J.; Santos, E.A. Evaluation of FAO Penman–Monteith and alternative methods for estimating reference evapotranspiration with missing data in Southern Ontario, Canada. *Agric. Water Manag.* **2010**, *97*, 635–644. [CrossRef]
19. Noor, N.M.; Abdullah, M.M.A.B.; Yahaya, A.S.; Ramli, N.A. Comparison of Linear Interpolation Method and Mean Method to Replace the Missing Values in Environmental Data Set. *Mater. Sci. Forum* **2014**, *803*, 278–281. [CrossRef]
20. Ullah, I.; Fayaz, M.; Naveed, N.; Kim, D. ANN Based Learning to Kalman Filter Algorithm for Indoor Environment Prediction in Smart Greenhouse. *IEEE Access* **2020**, *8*, 159371–159388. [CrossRef]
21. Meinhold, R.J.; Singpurwalla, N.D. Understanding the Kalman Filter. *Am. Stat.* **1983**, *37*, 123.
22. Cui, Y.; Xu, Y.; Peng, R.; Wu, D. Layer Normalization for TSK Fuzzy System Optimization in Regression Problems. *IEEE Trans. Fuzzy Syst.* **2023**, *31*, 254–264. [CrossRef]
23. Saranya, C.; Manikandan, G. A study on normalization techniques for privacy preserving data mining. *Int. J. Eng. Technol.* **2013**, *5*, 2701–2714.
24. Grossberg, S. Recurrent neural networks. *Scholarpedia* **2013**, *8*, 1888. [CrossRef]
25. Hua, Y.; Zhao, Z.; Li, R.; Chen, X.; Liu, Z.; Zhang, H. Deep Learning with Long Short-Term Memory for Time Series Prediction. *IEEE Commun. Mag.* **2019**, *57*, 114–119. [CrossRef]
26. Smagulova, K.; James, A.P. A survey on LSTM memristive neural network architectures and applications. *Eur. Phys. J. Spec. Top.* **2019**, *228*, 2313–2324. [CrossRef]
27. Fang, Z.; Crimier, N.; Scanu, L.; Midelet, A.; Alyafi, A.; Delinchant, B. Multi-zone indoor temperature prediction with LSTM-based sequence to sequence model. *Energy Build.* **2021**, *245*, 111053. [CrossRef]
28. Zhang, Y.; Henke, M.; Buck-Sorlin, G.H.; Li, Y.; Xu, H.; Liu, X.; Li, T. Estimating canopy leaf physiology of tomato plants grown in a solar greenhouse: Evidence from simulations of light and thermal microclimate using a Functional-Structural Plant Model. *Agric. For. Meteorol.* **2021**, *307*, 108494. [CrossRef]