agriculture

Article

Hybrid Deep Neural Networks with Multi-Tasking for Rice Yield Prediction Using Remote Sensing Data

Che-Hao Chang [1], Jason Lin [1,*], Jia-Wei Chang [1], Yu-Shun Huang [2], Ming-Hsin Lai [2] and Yen-Jen Chang [1]

[1] Department of Computer Science and Engineering, National Chung Hsing University, No. 145, Xingda Rd., South District, Taichung 40227, Taiwan; g110056155@mail.nchu.edu.tw (C.-H.C.); g111056079@mail.nchu.edu.tw (J.-W.C.); ychang@cs.nchu.edu.tw (Y.-J.C.)

[2] Crop Science Division, Agricultural Research Institute, Council of Agriculture, Executive Yuan, Taichung 41362, Taiwan; hhhuang@tari.gov.tw (Y.-S.H.); mhlai@tari.gov.tw (M.-H.L.)

* Correspondence: jasonlin@nchu.edu.tw

Abstract: Recently, data-driven approaches have become the dominant solution for prediction problems in agricultural industries. Several deep learning models have been applied to crop yield prediction in smart farming. In this paper, we proposed an efficient hybrid deep learning model that coordinates the outcomes of a classification model and a regression model in deep learning via the shared layers to predict the rice crop yield. Three statistical analyses on the features, including Pearson correlation coefficients (PCC), Shapley additive explanations (SHAP), and recursive feature elimination with cross-validation (RFECV), are proposed to select the most relevant ones for the predictive goal to reduce the model training time. The data preprocessing normalizes the features of the collected data into specific ranges of values and then reformats them into a three-dimensional matrix. As a result, the root-mean-square error (RMSE) of the proposed model in rice yield prediction has achieved 344.56 and an R-squared of 0.64. The overall performance of the proposed model is better than the other deep learning models, such as the multi-parametric deep neural networks (MDNNs) (i.e., RMSE = 370.80, R-squared = 0.59) and the artificial neural networks (ANNs) (i.e., RMSE = 550.03, R-squared = 0.09). The proposed model has demonstrated significant improvement in the predictive results of distinguishing high yield from low yield with 90% accuracy and 94% F1 score.

Keywords: crop yield prediction; remote sensing; convolutional neural network; deep learning; multi-task learning

check for updates

Citation: Chang, C.-H.; Lin, J.; Chang, J.-W.; Huang, Y.-S.; Lai, M.-H.; Chang, Y.-J. Hybrid Deep Neural Networks with Multi-Tasking for Rice Yield Prediction Using Remote Sensing Data. *Agriculture* **2024**, *14*, 513. https://doi.org/10.3390/agriculture14040513

Academic Editor: Xanthoula Eirini Pantazi

Received: 18 January 2024
Revised: 17 March 2024
Accepted: 20 March 2024
Published: 22 March 2024

1. Introduction

There are important issues in agriculture, such as crop pest detection, crop symptom detection, and crop yield prediction. However, it is difficult for farmers to manage these problems manually. As data-producing devices have improved in the past decade, these issues have gradually become solvable using artificial intelligence (AI) techniques. Conventional research on crop yield prediction is typically based on farmland yield in the previous year. However, this rule-of-thumb method is not stable and precise because the indirect features such as temperature, rainfall, soil attributes, and sunlight are not in the same conditions yearly. When the environment changes, different crop species have different amplitudes of impact on their yields.

In recent decades, smart farming has become increasingly popular. Remote sensing extracts information from satellites to enable AI models to establish data-driven decisions [1–4]. For example, the USGS Global Visualization Viewer (GloVis) is a fast and easy-to-use online satellite and aerial data search tool. Satellite data usually contain predefined wavelength bands in visible and near-infrared (NIR) spectral regions. Unfortunately, satellites face numerous challenges when acquiring remote sensing data. One problem is that it cannot continuously collect data from specific farmlands because of the maintenance of certain orbital positions, periods, and heights. In addition, the spatial resolution is significantly affected by the satellite altitude, which makes it challenging to customize the spatial resolution. Recently, UAVs and drone technologies have become better solutions for obtaining remote sensing data [5–8]. With such technologies, users have the flexibility to choose when and where to collect remote sensing data and collect the specific spectral wavelength with an appropriate sensor. However, the problems associated with using UAVs include the management of the collected datasets and extracting relevant information, both of which require domain knowledge. The spectral bands collected from UAVs are ideal for calculating vegetation indices, such as the normalized difference vegetation index (NDVI) [9].

Predicting the crop yield is a major issue in smart farming. With the advancement of hardware equipment, such as drones and sensors, users can now obtain rich crop features. Data-driven algorithms have gradually acquired the attention of researchers in agriculture [10]. They have applied data-driven models to improve the performance using available datasets containing crop information, weather, soil conditions, and other environmental features. Data-driven solutions can be classified into machine learning and deep learning. Traditional machine learning has demonstrated good performance in crop yield prediction with classic techniques, such as multiple linear regression, decision tree, random forest, k-nearest neighbor, and support vector machines [10–20]. Despite significant progress in using machine learning techniques for crop yield prediction, the inherent existence of nonlinear reliance between the input and target variables in the datasets is difficult to express using simple linear equations. Hence, these classic machine learning techniques face challenges in improving the next level of performance. Fortunately, the appearance of artificial neural networks (ANNs) has brought the potential to overcome the performance bottlenecks.

ANNs are groundbreaking machine learning algorithms that have garnered remarkable success in enhancing the performance of data-driven models. They can approximate any nonlinear relationship between the input and target variables [21]. Numerous studies have implemented ANN models in agricultural fields [22–25]. In 2007, Ji et al. [22] developed an ANN model to predict Fujian rice yield for a typical mountainous climate and compared its performance with that of a linear regression model. The experimental results indicated that the ANN model demonstrated better performance (R-squared = 0.67) than the traditional models (R-squared = 0.52). Subsequently, Baral et al. (2011) [23] developed an ANN model and particle swarm optimization to predict the rice yield in three different areas using nearly ten years of historical data, including daily mean and maximum temperatures and rainfall. Significant research has been conducted on crop growth in various regions. Çakır et al. (2014) [24] applied an ANN model to predict wheat yield in the southeast region of Turkey using crop and weather conditions. They evaluated the model's performance by varying the number of neurons and inputs to determine the optimal combination to improve results. Bhojani et al. (2020) [25] implemented an ANN to predict wheat yield. The study utilized different meteorological parameter datasets as training data and improved the neural network by incorporating three simple activation functions. In addition, they proposed three new activation functions and tested various configurations of hidden layers and neurons. The results demonstrated that the newly created activation functions outperformed the sigmoid function. Although ANN has delivered better performance in crop yield prediction than other traditional methods, such as regression models, it has to spend much more time training the model without a GPU. In addition, the situation worsens when the predetermined numbers of neurons and hidden layers are large because the model becomes more susceptible to overfitting.

As more crop features can be collected, researchers are attempting to enhance the model complexity by increasing the model depth or width. The primary objective is to equip the model with stronger learning capabilities. Such a complex model is called deep learning. It is considered a subset of machine learning and AI, involving the use of ANNs with multiple layers (deep neural networks, DNNs) to learn and make decisions from data. This approach is powerful within the broader field of AI. Various types of deep learning models have emerged across different domains. One of the earliest deep learning algorithms was the convolutional neural network (CNN). It has demonstrated high efficiency among numerous deep learning algorithms for computer vision tasks, such as image classification and object detection [26]. The key characteristic of CNN for improving training efficiency is weight sharing. The model can automatically learn image patterns by taking advantage of the kernel. With spatial location information, adjacent pixels have a certain correlation degree. Several studies have used CNN models to address crop-prediction issues in smart farming. Villanueva et al. (2018) [27] defined six bitter melon yield ranges and implemented a CNN model with three convolution layers to predict bitter melon crop yields using the leaf veins of the bitter melon. Monga et al. (2018) [28] conducted experiments on grape images using image processing techniques such as scale normalization and contrast enhancement. They then implemented a CNN model with five convolutions and dropout layers to forecast the Pinot Noir grape yield. Recent studies on crop yield prediction have applied distinctive deep learning approaches rather than conditional machine learning approaches. Khaki and Wang (2019) [29] proposed two DNNs, one for yield prediction and the other for validating yield prediction. The utilization of the validation DNN model indirectly fine-tuned the prediction DNN model, enhancing overall performance. Chu and Yu (2020) [30] subsequently proposed another deep learning approach that fused two back-propagation neural networks (BPNNs) and an independent recurrent neural network (IndRNN) to forecast summer and winter rice yields. These studies demonstrated that deep learning models outperformed machine learning models. In a recent study, Kalaiarasi and Anbarasi (2021) [31] introduced the growing-degree day (GDD) as a measure of the effect of weather conditions on crop yield and built a multiparametric deep neural network (MDNN) containing a residual block to predict crop yield. The research findings substantiate that MDNN surpasses DNN in achieving superior performance for crop yield prediction. However, as the learning process involves building representations through a hierarchical structure with increasing complexity, there is no assurance regarding the quality of the final hidden representation.

Regarding crop characteristics, crop growth can be considered a time series containing several time points, each representing the crop's status at different times. The entire time-series record contains all information about the crop. Researchers have applied time-series models to address yield-prediction issues. One of the most famous and earliest time-series models is the recurrent neural network (RNN). One advantage of an RNN is its hidden state, which allows it to store historical information. This information is shared between the neurons in the same layer for more flexible calculations. You et al. (2017) [32] implemented a CNN and LSTM to predict soybean yield using sequential remotely sensed images. They also included a Gaussian process unit to improve the model accuracy, which potentially serve as inspiration for applications in remote sensing and computational sustainability. Khaki et al. (2020) [33] proposed a hybrid model that combines a CNN and an RNN to forecast corn yields. The data used were from the entire Corn Belt in the U.S. three years ago. Although the RNN is a powerful and useful sequential model, it still suffers from issues such as vanishing and exploding gradients [34]. When the time series becomes too long, RNNs may struggle to optimize and adjust effectively. To address this problem, one type of RNN that has been developed is the long short-term memory (LSTM) network. Rußwurm and Korner (2017) [35] applied LSTM to extract dynamic temporal features for classifying crop types using a long-sequence image dataset. The results showed that the LSTM-based model outperformed the single-temporal model. The overall accuracy of the multitemporal LSTM model was reported to be 90.6%, which is higher than that of the single-temporal CNN model (89.2%) and the baseline SVM model (40.9%). Zhong et al. (2019) [36] developed a hybrid deep neural network that combined two different networks–one based on LSTM and the other based on a one-dimensional convolutional neural network (1D-CNN)–to classify summer crops. An improved model based on an LSTM model called the GRU was developed recently. The experiment showed that the performance of the GRU was similar to that of the LSTM but with a faster training time. Yu et al. (2021) [37] proposed a hybrid CNN-GRU model to predict soil water content. The hybrid model combined a CNN with significant feature extraction and a GRU with strong memory capacity, and the experiment showed that the hybrid model outperformed the independent CNN or GRU. Hence, RNN, LSTM, and GRU are widely used to implement different solutions in smart farming. From the above related studies, hybrid models have become a major trend in solving agricultural issues by combining the advantages of different deep neural networks to improve performance.

In this study, we propose an architecture consisting of two deep neural network models: a multi-kernel convolutional neural network (MKCNN) and bidirectional long short-term memory (Bi-LSTM) [38]. We employed multitask learning to train both models interactively, and the proposed hybrid model was utilized to predict rice yield. Van Klompenburg et al. (2020) [39] discovered that some researchers conducted experiments using NDVI and other crop-relevant features. In addition to NDVI, several other useful features have been applied to crop yield prediction. However, researchers must possess domain knowledge to identify the necessary features. Therefore, before training the rice yield prediction model, we proposed a combination of several feature analysis approaches to deal with the Hughes phenomenon, including Pearson correlation coefficients (PCC), SHapley additive extensions (SHAP), and recursive feature elimination with cross-validation (RFECV), to select highly relevant features for training an optimal predictive model.

The remainder of this paper is organized as follows: Section 2 presents the materials used and explains the variety of crop features. It also describes the approaches used for feature analysis and selection. Section 3 describes the preprocessing of features and then describes the proposed model, which consists of MKCNNs and Bi-LSTM. Section 4 illustrates the experimental results with a confusion matrix and uses various evaluation indicators to estimate the model's performance. Finally, Section 5 presents the conclusions of the study.

2. Materials and Feature Analysis

In this section, we introduce the materials used and illustrate how the crop dataset was collected. We then present the application of three different feature analysis algorithms to find useful features.

2.1. Materials

The dataset used in this study contained remote-sensing features and a time series. It was collected from farmland near a laboratory in Wufeng District, Taichung City, Taiwan, at latitude 24.0313498° N and longitude 120.6912149° E. The drone flowed over the field in an S shape at a height of 25 m. The other mission parameters were as follows: forward overlap of 80%, side overlap of 80%, and a ground sample distance of 1.78 cm per pixel. Additionally, calibration photos were taken before and after the flight using an RP-04-1913023-SC calibration panel (MicaSense). Furthermore, the Rededge-M system was equipped with a downwelling light sensor (DLS) to measure and record the ambient light during flight. The specific details of the multispectral camera are presented in Table 1. During image processing using the Pix4D Mapper, the calibration value from the calibration panel and the ambient light information from the DLS module were employed for image calibration. A total of 333 remote sensing images were collected in 2020 and 2021. Due to the inconsistent collection of data points between the two years, we have chosen to select the points that have a time overlap between March 5 and June 2. The remote sensing image is shown in Figure 1, with each block representing an independent farmland. The yield maps for 2020 and 2021 are depicted in Figure 2, comprising fields labeled 42-1 and 42-2. The data are sourced from the Taiwan Agricultural Research Institute. Each field was further partitioned into a grid measuring 10 × 16, with each region encompassing an area of 0.1 hectare. The settings of the flight mission are presented in Table 2. First, we collected a five-band remote sensing image containing RGB, red edges, and NIR. We then cropped each block line to create an independent image. From each independent image, we extracted five bands and calculated vegetation indices, such as NDVI and MSR. In total, 33 different vegetation indices were calculated. In the next subsection, we explain how we filtered and selected the most relevant features for rice crop yield.

Table 1. The specifications of the multi-spectral camera.

Item	Detail
Model	MicaSense Rededge-M
Ground sample distance	8.2 cm px^{-1} per band at 120 m above ground level
Blue band	475 (20) nm *
Green band	560 (20) nm *
Red Band	668 (10) nm *
Near IR band	840 (40) nm *
Red edge band	717 (10) nm *

* Center wavelength (Bandwidth FWHM) nm.

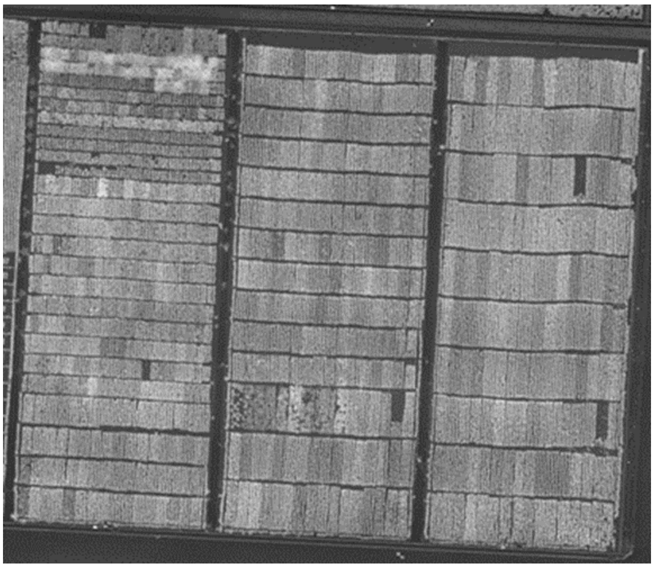


Figure 1. The remote sensing image captured by a drone, with each block representing an independent farmland.

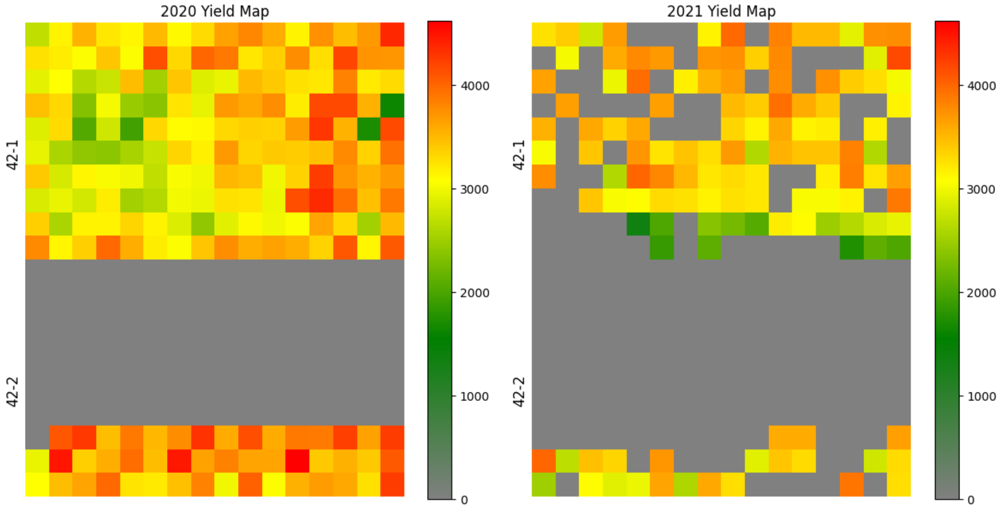


Figure 2. The yield map for years 2020 and 2021 collected from the Taiwan Agricultural Research Institute.

Table 2. The flight mission settings.

Item	Detail
Flight Height	25 m above ground level
Forward overlap	80%
Side overlap	80%
Ground sample distance	1.78 cm px^{-1}

The data supporting the findings of this study are available from the Taiwan Agricultural Research Institute, Council of Agriculture, and Executive Yuan. However, restrictions apply to the availability of these data, as they were used under the license for the current study and are not publicly accessible. Nevertheless, authors can obtain the data upon reasonable request and with permission from the Taiwan Agricultural Research Institute, Council of Agriculture, and Executive Yuan.

2.2. Feature Analysis

Generally, the performance of a classifier progressively improves when the number of features increases below a certain optimal number. After such an optimal number of features is achieved, the feature increase begins to downgrade the classifier's performance [40]. This is commonly known as the curse of dimensionality; the Hughes phenomenon. Therefore, this section proposes a feature analysis process using three different algorithms:

We first applied the PCC algorithm [41] to the collected data, in which the correlation coefficient value falls within the range $[-1, 1]$. The PCC equation is defined as follows:

$$r(x, y) = \frac{\sum_{i=1}^{n} (x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n} (x_i - \overline{x})^2} \sqrt{\sum_{i=1}^{n} (y_i - \overline{y})^2}} \tag{1}$$

where $\overline{x}$ and $\overline{y}$ are the average values of x variables and y variables, respectively. Note that there is a strong positive correlation between features x and y when the correlation coefficient approaches 1. Otherwise, a negative correlation exists. Although the PCC algorithm helps determine the linear correlation between any two variables, it is not a perfect method for determining other relations, such as nonlinear correlation and independence. Therefore, a preset value of 0.4 is defined as a threshold to classify each pairwise feature into one of two groups according to their correlation coefficient value r. That is, the linear correlation group and the other group with $r > 0.4$ and $r \leq 0.4$, respectively. The results of the PCC algorithm are presented in Table 3, where fourteen out of thirty-three features were split into a linear correlation group using PCC. The most relevant features are MSR, NDVI, and OSAVI. The equations for these index features are summarized in Table A1. For the feature pairs classified into the other group, we applied two other algorithms, SHAP and RFECV, to analyze their contribution and importance concerning the given predictive model.

The SHAP value measures the contribution of each feature to a given predictive model. It is a solution concept proposed by the game theory master Lloyd Stowell and originates from cooperative game theory [42]. The SHAP equation is defined as follows:

$$\varphi_j(v) = \sum_{S \subseteq \{x_1, x_2, \ldots, x_p\} \setminus \{x_j\}} \frac{|S|!(p - |S| - 1)!}{p!} \left(v\left(S \cup \{x_j\}\right) - v(S)\right) \tag{2}$$

where $x_1, x_2, \ldots, x_p$ are features for building the predictive model, S is a subset that excludes x_j, and the function v indicates that if S is a participant set, then $v(S)$ is defined as the value of S, describing the total expected sum that the members of S can obtain by cooperation. The advantage of using SHAP is that it reflects each feature's positive and negative impacts on the predictive outcomes. However, since the SHAP value's goal is to determine each feature's contribution, it calculates the contributions of all permutations of the feature combinations. Therefore, the computational complexity of calculating the SHAP value increases exponentially with the number of features. Fortunately, a previous study proved that a predictive model using tree ensembles could reduce computational complexity [43]. Hence, this study applied one of the best tree ensemble implementations, XGBoost, to evaluate the contribution of each feature [44,45]. Two major techniques are applied in XGBoost: gradient boosting and random sampling. In gradient boosting, each tree is related to other trees, and the main objective is to ensure that the trees generated later correct the mistakes of the previous tree. Random sampling helps to analyze the collected data with a lower margin of error, enabling the data to provide more accurate insights into a specific subject matter. The primary concept of XGBoost is to integrate many weak decision trees to form a strong predictive model using the boosting technique. In this study, only XGBoost was used to calculate SHAP values. It was not applied to the rice yield prediction. The resulting SHAP values for each feature are shown in Figure 3. The results showed that the top-contributing features were the PCMSPLOT, reHTNDVI, GBNDVI, and RED.

The two analyses, PCC and SHAP, mentioned above, can identify the features that contribute most significantly to rice yield prediction. However, they did not indicate the optimal number of features the model should be trained to achieve maximum training efficiency. Training a model with too few features may result in poor performance, whereas using too many features can slow down the training process and may even lead to the Hughes phenomenon. Hence, this study employed a third algorithm, the recursive feature elimination with cross-validation (RFECV) algorithm, to determine the most beneficial number of features for model utilization. The RFECV algorithm estimates the maximum number of features required to establish an optimal model [46]. It iteratively estimates the model error for each feature and removes the highest error. Decision trees were implemented to obtain model errors for each feature. The results of the RFECV estimation are shown in Figure 4. The model accuracy was the average of the k-fold cross-validation, where *k* was set to ten. As shown in Figure 4, the prediction model achieved the highest accuracy when employing seven features, which remained consistently stable even after averaging. In addition, this algorithm documents the specific features employed to train the prediction model, including reHTNDVI, PCMSPLOT, GBNDVI, reHT, BLU, reVARI, and RED. It is noteworthy that the first five features aligned precisely with the top six independent SHAP values calculated. Consequently, from the SHAP value and RFECV algorithms, it can be inferred that the utilization of these features is expected to result in greater efficiency of the prediction model.

Table 3. Results of using the PCC for each feature.

<table>
<tr><th></th><th><i>NDVI</i></th><th><i>SAVI</i></th><th><i>OSAVI</i></th><th><i>MSR</i></th><th><i>NIR</i></th><th><i>reHT</i></th><th><i>BLU</i></th><th><i>GRN</i></th><th><i>GRE</i></th></tr>
<tr><td>PCC</td><td><b>0.49</b></td><td>0.41</td><td><b>0.48</b></td><td><b>0.49</b></td><td>0.40</td><td>0.18</td><td>−0.14</td><td>−0.19</td><td>−0.13</td></tr>
<tr><td></td><td><i>RED</i></td><td><i>SAVIl</i></td><td colspan="2"><i>SAVIRE</i>1</td><td><i>SAVIRE</i>2</td><td><i>MCARI</i></td><td colspan="2"><i>NDGR</i></td><td><i>GBNDVI</i></td></tr>
<tr><td>PCC</td><td>−0.24</td><td>0.41</td><td colspan="2">0.46</td><td>0.47</td><td>0.07</td><td colspan="2">0.46</td><td>0.09</td></tr>
<tr><td></td><td><i>NDBL</i></td><td colspan="2"><i>NDRERED</i></td><td><i>reVARI</i></td><td><i>CC</i>1</td><td><i>ARI</i></td><td><i>TGI</i>2</td><td><i>GLI</i>2</td><td><i>NDRE</i></td></tr>
<tr><td>PCC</td><td>0.40</td><td colspan="2">0.22</td><td>0.23</td><td>0.47</td><td>0.41</td><td>−0.16</td><td>0.06</td><td>0.46</td></tr>
<tr><td></td><td><i>SRGRN</i></td><td><i>SRBLU</i></td><td colspan="2"><i>SRGRNRED</i></td><td colspan="2"><i>reHTNDVI</i></td><td><i>PCMSPLOT</i></td><td colspan="2"><i>PCMS</i></td></tr>
<tr><td>PCC</td><td>0.42</td><td>0.40</td><td colspan="2">0.16</td><td colspan="2">0.21</td><td>0.26</td><td colspan="2">0.01</td></tr>
<tr><td></td><td colspan="3"><i>GRNDVI</i></td><td colspan="3"><i>SRRED</i></td><td colspan="3"><i>SRRGE</i></td></tr>
<tr><td>PCC</td><td colspan="3">0.16</td><td colspan="3">0.46</td><td colspan="3">0.45</td></tr>
</table>

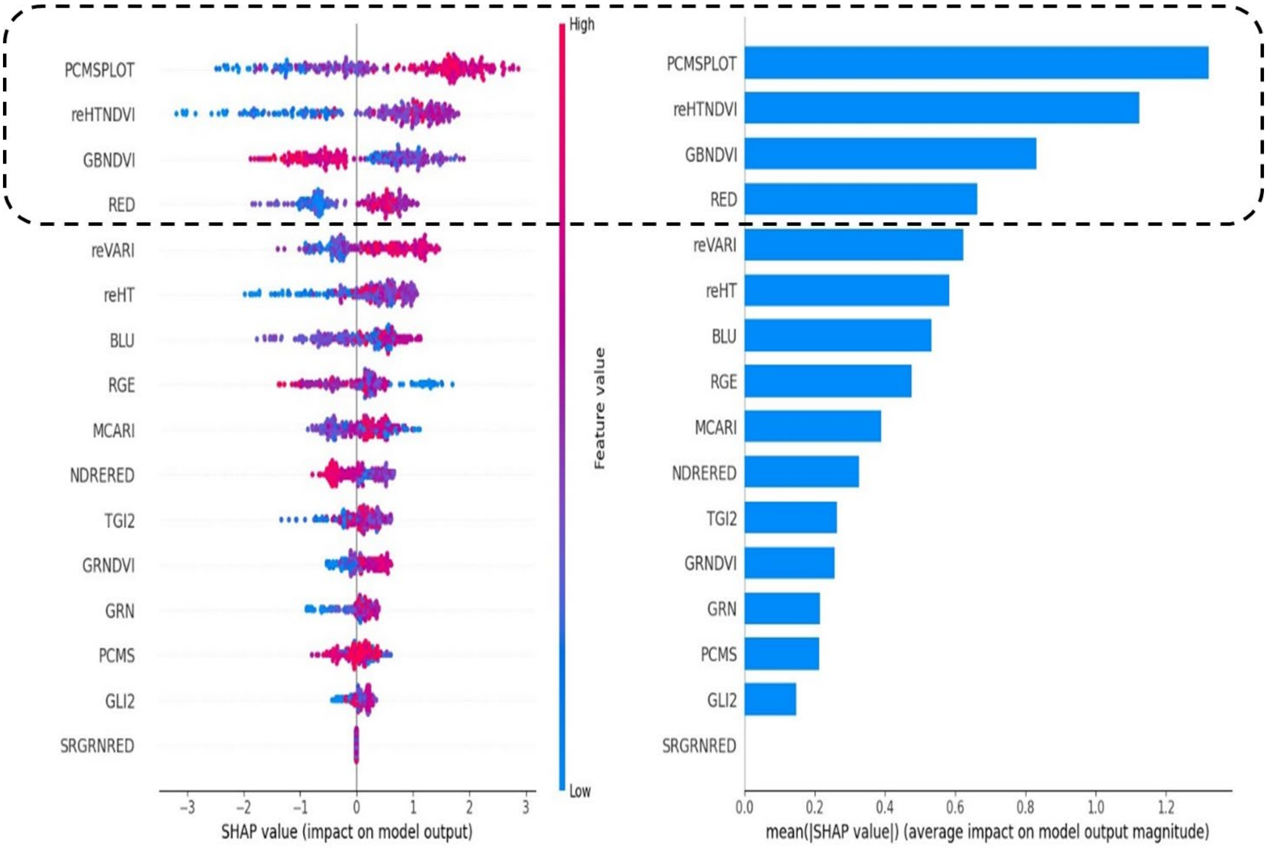


Figure 3. Feature contributions for rice yield using SHAP values, where individual SHAP values and average absolute SHAP values are at the left and right, respectively, with the top-contributing features circled in the dashed box.

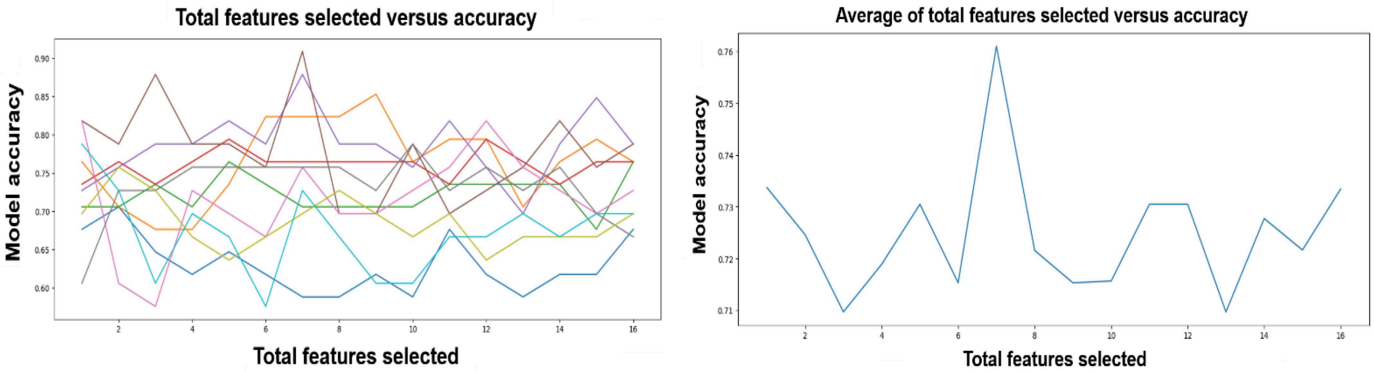


Figure 4. The results of model accuracy for RFECV. The left plot with different colored lines represents the use of different folds as the test set; the right plot shows the average model accuracy for the k-fold cross validation.

The Hughes phenomenon indicates that using more features in model training is not necessarily better. Careful analysis is required to extract useful features to achieve the maximum benefit in prediction. Consequently, we selected three linearly correlated features and four other correlated features for the training process. The rationale for this selection was to strike a balance. Opting for all the linearly correlated features would oversimplify the overall relationship and hinder the model's generalization ability. However, choosing all features from the other correlated sets would make it challenging for the model to train and potentially hinder convergence to an optimal state. Therefore, the following seven features were selected to train the prediction model: MSR, NDVI, OSAVI, PCMSPLOT, reHTNDVI, GBNDVI, and RED.

3. Methodology

In this section, we divide the proposed model into two main processes: (1) preprocessing the data of the selected features; and (2) the architecture of the hybrid model, which consists of MKCNN and Bi-LSTM, to predict crop yield.

3.1. Preprocessing

Since the collected dataset was noisy and unstructured, preprocessing and reconstruction were inevitable steps before implementing the proposed hybrid model. Hence, this study applies the max normalization to each feature, as follows:

$$\hat{x}_i = \frac{x_i - x_i^{min}}{x_i^{max} - x_i^{min}} \tag{3}$$

where $\hat{x}_i$ is the normalized value, x_i is the original value, and x_i^{min} and x_i^{max} are the minimum and maximum values of feature i, respectively. All the feature values were projected into the range $[0, 1]$. The reason for normalization is that the target was predicted based on different feature scales. They do not contribute equally to the model fitting and learned functions, which indirectly decreases model performance.

However, since the MKCNN cannot utilize digital data as an input source, we reconstructed the collected data into a three-dimensional matrix. Each remote sensing feature is defined as $f_1, f_2, \ldots, f_n$, the time stamps are defined as $t_1, t_2, \ldots, t_m$, and each farmland is defined as $l_1, l_2, \ldots, l_p$. The remote-sensing features and time series were set as columns and rows, respectively. Hence, the size of the three-dimensional matrix was $l_p \times t_m \times f_n$. Figure 5 shows a general view of the reconstructed data collected from the farmland, where each block line is reconstructed as a two-dimensional matrix, and the entire farmland is integrated into a three-dimensional matrix.

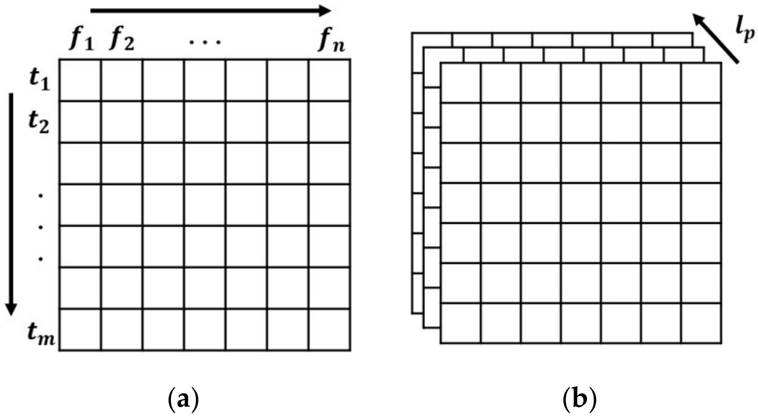


Figure 5. General view of reconstruction collected data for (**a**) a unit of farmland and (**b**) the entire farmland.

For the Bi-LSTM, the time stamps $t_1, t_2, \ldots, t_m$ are split into separate segments, with each segment containing all the remote sensing features $f_1, f_2, \ldots, f_n$. Each timestamp is sequentially fed into a hybrid model.

3.2. Proposed Model

The proposed hybrid model integrates two different classes of deep learning models to analyze the data in block and stream modes, which enriches feature learning. Figure 6 shows a general view of the proposed model architecture. The implementation of MKC-

NNs was inspired by MS-Blocks [47]. In the following subsections, we comprehensively introduce each part of the proposed hybrid model.

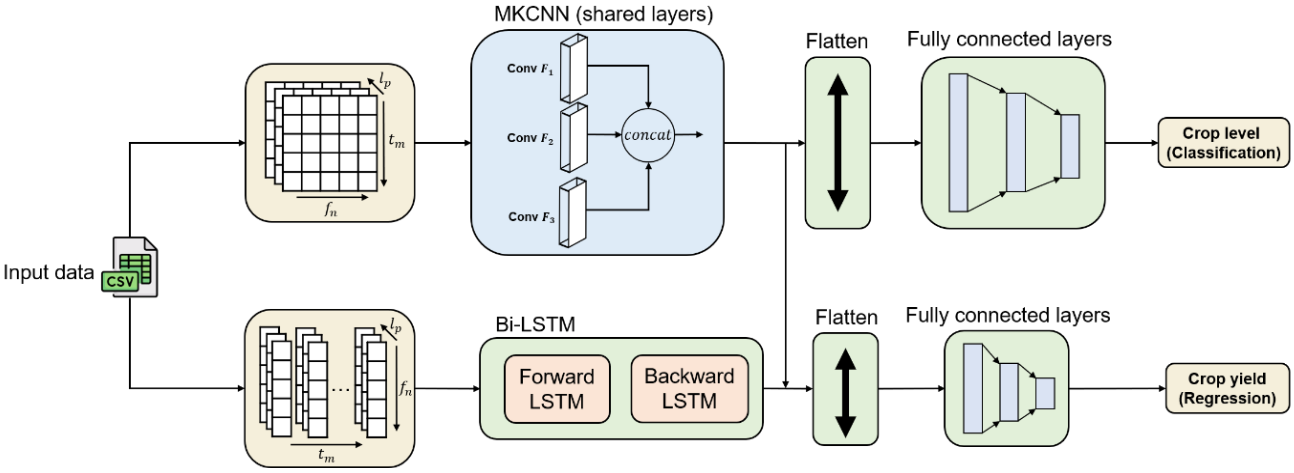


Figure 6. The architecture of the proposed hybrid model consists of MKCNN and Bi-LSTM.

The proposed model can be divided into two main modules: MKCNNs and Bi-LSTM. First, the input data are converted into the corresponding input structure based on the module. For example, the input structure required for an MKCNN is a three-dimensional matrix obtained through a preprocessing operation. Next, the hybrid model received the input data, processed through various operations and flattened into a one-dimensional vector. Subsequently, it passes through the fully connected layers, resulting in the calculation of two values: the predicted crop yield (regression task) and the classification of high or low yield (classification task). The hybrid model also incorporates shared layers from multi-task learning techniques. In this study, the shared layers were integrated into the MKCNN, enabling the model to indirectly improve the prediction results of the regression task while optimizing the classification task. This enhances the overall accuracy of the model. The internal architecture of the proposed model is described in detail in the following sections.

CNN is a major component of MKCNNs, a deep learning model that handles grid-like data, such as images or rows of multi-column data. It consists of four major operations: a convolution layer, padding, an activation function, and a fully connected layer. The convolutional operation is the first operation generally used in CNN models. This can be regarded as calculating the sum of the products of a block of input values and the values of a convolutional kernel, also called a filter. Kernel K was applied to input image I using a sliding window. For each pixel in I, the sum of the element-wise products was calculated using K and stored in the corresponding pixel in F. Once K convolves the entire image, the resulting feature map F is produced. To prevent loss of information at the image's borders, additional pixels are added to the periphery of the image during convolution. The padding operation has two benefits: it allows border patterns to be captured and prevents the image from being continuously compressed, resulting in the loss of block patterns. All the pixels of the image are subjected to an activation function that transforms the output and input into a nonlinear relationship, thereby enabling the deep learning model to have a more expressive meaning. Finally, the high-dimensional data are flattened into a one-dimensional array and imported into the fully connected layer, where the previously extracted features are classified or regressed after weight calculation in the final stage.

The core of MKCNNs is a multi-kernel block, which applies several kernels of different sizes to perform convolution. We believe implementing filters with the same kernel but different sizes can result in different meanings for the collected feature maps. For instance, the Sobel operation for edge detection in an image with filters of varying sizes, such as 3 × 3, 5 × 5, and 7 × 7, produces feature maps with different meanings. These distinct feature maps enable the model to learn from a wider range of features. A schematic of the multi-kernel block is shown in Figure 7.

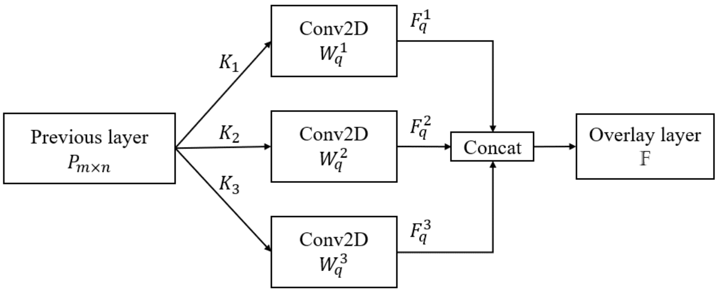


Figure 7. Architecture for the multi-kernel block.

As shown in Figure 7, the inputs are sent to three different convolutional kernels, denoted as K_1, K_2, and K_3 for feature extraction. The previous layer is denoted as P_{m*n}. The convolutional operation is defined as the product of W_q^c and the input layer P as in (4), where W_q^c represents the convolution using c^{th} kernel with q convolutional operations, and the bias of the c^{th} kernel is denoted as $bias_{q,m\times n}^c$. Each feature map is defined as $F_q^c = \left[(f_1)_{m\times n}^c \quad (f_2)_{m\times n}^c \quad \cdots \quad (f_q)_{m\times n}^c \ \right]^T$, where f_i indicates one of the convolved feature maps using c^{th} kernel. Each entire convolved feature map F_q^c with c^{th} kernel is concatenated to form $\mathbb{F}$, as shown in Equation (5).

$$F_q^c = activate\left(W_q^c P + bias_{q,m*n}^c\right) \quad c = 1,\ 2,\ 3 \tag{4}$$

$$\mathbb{F} = \begin{bmatrix} F_q^1 & F_q^2 & F_q^3 \end{bmatrix}^T \tag{5}$$

On the other hand, we aim to retain more detailed feature maps, but using more convolutional operations can result in a lack of fidelity in the feature maps. Thus, we applied a forward mechanism in which each multi-kernel block result was passed to the fully connected layers. For instance, if we implement three multi-kernel blocks, we obtain three status forms: $\mathbb{F}_1$, $\mathbb{F}_2$, and $\mathbb{F}_3$. Next, we concatenate these forms into a one-dimensional matrix and feed them into the fully connected layers. This mechanism can also save more features for training predictive models without sacrificing the fidelity of feature maps with more multi-kernel blocks.

MKCNNs determine high-yield versus low-yield patterns based on a data block area of multiple times and features. However, the original data were in a time series. The feature value at a specific time can relate to either the previous or the latter values. Therefore, we applied Bi-LSTM as another model to analyze the data sequentially. The LSTM is a variant of the RNN that overcomes the problem of gradient vanishing or explosion by integrating a gradient superhighway in the form of a cell state C, in addition to the hidden state h [48]. Given a sequence of data $x_1, x_2, \ldots, x_P$ for P times with output y_{t-1} and hidden state h_{t-1}, the forget gate that decides to discard information can be defined as follows:

$$f_t = \sigma\left(W_f[h_{t-1} \cdot x_t] + bias_f\right) \tag{6}$$

where σ is sigmoid function, and W_f is the weight matrix for the forget gate. The input gate can control whether the input value i_t at time step t is calculated using the previous hidden state h_{t-1}, which is defined in (7). Similarly, the hidden state h_t and output state o_t of the LSTM are defined in Equations (8) and (9), respectively, where the cell C_t at time step t can be taken as an intermediate variable.

$$i_t = \sigma(W_i[h_{t-1} \cdot x_t] + bias_i) \tag{7}$$

$$h_t = o_t \cdot tanh(C_t) \tag{8}$$

$$o_t = \sigma(W_o[h_{t-1} \cdot x_t] + bias_o) \tag{9}$$

LSTM is more efficient and performs better than a simple RNN in building long-term time sequences. However, LSTM applies only previously learned information without considering subsequent information in the time series. In practical scenarios, predictions may require information from the entire input sequence. The model training of Bi-LSTM requires the use of all information from the input sequence. This combines forward and backward information from the input sequence. The output vector contains information from both directions using concatenation operations.

The proposed hybrid model introduced a multi-tasking technique [49] to enhance model performance. Many studies have used the root-mean-square error (RMSE) as a loss function for crop yield prediction. RMSE calculates the distance between the ground truth and the predicted value. In this study, we used the predicted value with the RMSE to determine whether the crop yield was high or low. Our main goal was to achieve greater accuracy in the regression task for predicting the rice yield. We found that simply applying the RMSE did not accurately determine the correct predicted value. Therefore, we attempted to incorporate classification to adjust the predicted value. We introduced a multitasking technique to increase the regression accuracy and prevent overfitting of the proposed model. The shared layers facilitate the coordination of relationships between multiple tasks. By sharing layers, adjustments made to one task can affect the performance of other tasks. Given several tasks, $t_1, t_2, \ldots, t_N$ for N tasks. The loss function is defined as follows:

$$\mathbb{L} = \sum_{i=1}^{N} w_i \mathcal{L}(\theta_i, t_i) \tag{10}$$

where θ_i is the learning weights of t_i, $\mathcal{L}$ is the loss function in t_i, and the w_i, is the proportion of loss in $\mathcal{L}$ [50].

4. Experimental Results

In this section, five indicators were employed to assess the performance of the proposed model. To illustrate the performance of the model, we conducted several experiments, including an ablation experiment. In addition, various hybrid model parameters were evaluated to determine the optimal configuration.

The origin dataset comprises a total of 405 entries. Due to the presence of some data errors, such as missing values, the dataset was cleaned, resulting in a final usable count of 333 entries. The dataset was then divided into training, validation, and testing sets in proportions of 80%, 10%, and 10%, respectively. The validation set was used with an early stopping technique, with the parameter patient set to 100 to avoid overfitting during model training. We collected rice yield data from the past five years and calculated the average value to set the decision boundary. In this study, the decision boundary was set at 3032. Based on this boundary, the positive class was assigned a high yield, and the negative class was assigned a low yield. Five metrics were applied to evaluate the model's performance: accuracy, recall, precision, R-squared $\left(R^2\right)$, and F1 score. These metrics are defined by Equations (11)–(15):

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{\sum_{i=1}^{n}(y_i - \overline{y})^2} \tag{11}$$

$$accuracy = \frac{(TP + TN)}{Total.N} \tag{12}$$

$$precision = \frac{TP}{(TP + FP)} \tag{13}$$

$$recall = \frac{TP}{(TP + FN)} \tag{14}$$

$$F1\ score = 2(\frac{precision * recall}{precision + recall}) \tag{15}$$

where y_i is the true value, $\hat{y}_i$ is the predict value, and $\overline{y}$ is the average value of y variable.

It is necessary to demonstrate that the seven features extracted using the three analytical algorithms benefit model training. The experimental results, depicted in Table 4, with the best results highlighted in bold, indicate that training the model using 33 features does not yield superior results compared to using only seven features. The experimental results suggest that employing all the features for training inevitably leads to the Hughes phenomenon. Hence, the extraction of the most relevant features of rice yield is an indispensable step.

Table 4. Comparison of feature quantity in model training.

Number of Features	*Loss*	R^2
33	387.45	0.54
7	344.56	0.64

The proposed hybrid model sets up shared layers to coordinate the prediction results of MKCNN and Bi-LSTM, in which the task of crop yield prediction is split into two subtasks: regression and classification. The regression subtask provided a scalar output for prediction, whereas the classification subtask provided a discrete result (i.e., high versus low yield). Two loss functions, RMSE and cross-entropy, were used in the regression and classification tasks, respectively. It is important to demonstrate that the hybrid model performs better than a single model. Therefore, an ablation experiment was conducted to test this hypothesis. The results of the ablation experiments are presented in Table 5, with the best results highlighted in bold. The experiment involved implementing single models for crop yield prediction, showcasing the superior performance of the hybrid Bi-LSTM and MKCNN models. The models were constructed both with and without specific components to determine their necessity. As a result, the proposed model, combining MKCNN and Bi-LSTM, achieved the best performance in all aspects except recall. One reason is that the collected data have an imbalanced amount of high-yield versus low-yield data, with more high-yield data than low-yield data. Since the ANN model accurately predicts a high yield for almost all data, it results in the highest recall among the prediction models.

Table 5. Results of ablation experiment.

Model	*Loss*	R^2
CNN	550.13	0.09
LSTM	518.42	0.22
Bi-LSTM	489.07	0.28
MKCNN	395.15	0.53
MKCNN + LSTM (w/o MTL)	404.27	0.51
MKCNN + Bi-LSTM (w/o MTL)	399.77	0.52
MKCNN + LSTM	367.98	0.59
MKCNN + Bi-LSTM	344.56	0.64

First, we implemented a single model to predict crop yield. The results in the first four rows show that the Bi-LSTM model performed better than the LSTM model and that the MKCNN model performed the best. This is because the Bi-LSTM model incorporates both forward and backward information from the input sequence, resulting in better accuracy than the LSTM model. The single MKCNN model outperformed the other models because it first applied multi-kernel block operations, extracting more useful feature maps that were fed to the fully connected layers. However, LSTM-based models are not inherently poor for crop prediction. Their poor performance is owing to the short input sequence length of only seven units, which does not fully leverage the advantages of the LSTM-based model. The hybrid models without the MTL technique perform similarly to a single MKCNN, as shown in Table 5. The model without the MTL used two models that made separate crop predictions and averaged the results for the final prediction. The model proposed in the final row fuses the MKCNN and Bi-LSTM models by exploiting their advantages. The proposed model performed best in the ablation experiment. The MKCNN perfectly extracts feature maps, and Bi-LSTM extracts forward and backward information from the input sequence. Finally, applying shared layers to connect the two models yielded the best performance.

Moreover, we attempt to determine the optimal distribution of the two models. The results are listed in Table 6. We tested various distribution ratios between the regression and classification subtasks, respectively. The model achieved the best performance with a distribution ratio of 3 to 7 compared to other distributions. Therefore, a distribution ratio of 3:7 was identified as optimal for the two models.

Table 6. The performance in different settings of distribution ratios between the regression and classification subtasks.

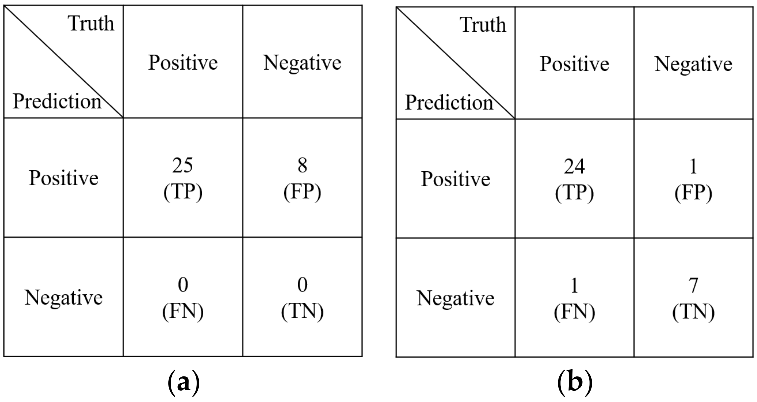
Ratio of Regression and Classification	***Loss***	***R^2***
5:5	354.26	0.62
6:4	346.21	0.64
4:6	350.17	0.63
7:3	349.03	0.63
3:7	344.56	0.64
8:2	356.86	0.62
2:8	355.80	0.62
9:1	347.40	0.64
1:9	346.40	0.64

Table 7 shows the overall performance of the proposed model compared to other deep learning models [29,31,47]. The best results are highlighted in bold. The proposed model outperformed the other deep learning approaches in all metrics except recall.

Table 7. Comparison with other models used for crop yield prediction.

Model	***Loss***	***R^2***	***Accuracy***	***Precision***	***Recall***	***F1***
ANN	550.03	0.09	0.76	0.76	1.0	0.86
DNN [29]	389.86	0.54	0.82	0.92	0.88	0.90
MDNN [31]	370.80	0.59	0.82	0.85	0.92	0.88
MS-Block [47]	372.69	0.58	0.88	0.89	0.96	0.92
Proposed	344.56	0.64	0.90	0.92	0.96	0.94

Detailed information is shown in Figure 8, where we use a confusion matrix to illustrate the performance of the proposed models. The left plot (a) shows the results of the ANN model, and the right plot (b) shows the results of the proposed model. Four parameters were calculated: the true positives (TP), false positives (FP), false negatives (FN), and true negatives (TN). As shown in Figure 8, although the ANN model achieved a perfect recall result of 100%, it failed to predict low-yield instances. This indicates that the ANN model was biased towards predicting high yields even when the ground truth had a low yield. However, the proposed model could forecast both high and low yields accurately. This is because the dataset was imbalanced, and the proposed model was able to handle such imbalanced datasets effectively. However, the accuracy of the ANN model is 14% lower than that of the proposed model. This means that a high false-positive rate for the ANN in predicting high yield led to poor prediction of low yield.



(a)

Truth / Prediction	Positive	Negative
Positive	25 (TP)	8 (FP)
Negative	0 (FN)	0 (TN)

(b)

Truth / Prediction	Positive	Negative
Positive	24 (TP)	1 (FP)
Negative	1 (FN)	7 (TN)

Figure 8. Comparison of the confusion matrices between (**a**) ANN and (**b**) the proposed model.

Next, we discuss the impact of the different parameters used in the proposed model, with a focus on rice yield prediction. To evaluate their effects, we compared loss values, R-squared values, and training times across different settings. The discussion is organized into several subsections, each addressing a specific parameter.

Table 8 lists the performance of the proposed model with different numbers of layers in a fully connected layer. There was a strong relationship between the number of layers and model accuracy. As the number of layers increased, the loss decreased. Although the optimal number of layers is four, as shown in Table 8, the loss almost reaches a convergent state when the number of layers is three. The reason why the loss of four layers is better than three layers is due to slight fluctuations in the hybrid model, resulting in the losses for three layers to five layers being close to each other ($\leq$1.2%). Therefore, we considered the optimal number of layers to be three for a fully connected layer.

Table 8. Comparison of performance with different numbers of fully connected layers.

Layers	*Loss*	R^2	Training Time
1	370.01	0.59	**87.02**
2	364.89	0.60	92.17
3	355.31	0.62	99.18
4	**351.40**	**0.63**	101.79
5	354.52	0.62	105.12

Table 9 lists the performance of the proposed model for different batch sizes. Batch size refers to the number of training samples used in each iteration during training. The batch size has a significant impact on the accuracy of the model as well as its optimization degree and speed. The advantage of a small batch size is that it introduces more randomness during training, which can improve the model's generalization ability. However, using a small batch size may make converging difficult for the models. Similarly, increasing the batch size is not recommended because it may cause the model to fall into local optima. Implementing a larger batch size can decrease the training time and improve convergence but may affect the model's accuracy. Therefore, selecting an optimal batch size is an important issue that must be carefully considered.

Table 9. Performance with different batch sizes.

Batch Size	Loss	R^2	Training Time
16	344.89	0.64	107.29
32	351.50	0.63	105.51
64	355.91	0.62	102.97
100	354.02	0.62	101.81
128	373.48	0.58	99.52

We also tested five network optimizers: Adam, Adagrad, Adadelta, Adamax, and RMSProp. Table 10 displays the best performance achieved by the different network optimizers, whereas Figure 9 shows the detailed loss values observed during the training process using various network optimizers.

Table 10. Comparison of performance with different optimizers for the proposed hybrid model.

Optimizer	Loss	R^2
Adam	347.42	0.64
Adagrad	450.26	0.39
Adadelta	499.34	0.25
Adamax	362.84	0.60
RMSProp	491.15	0.28

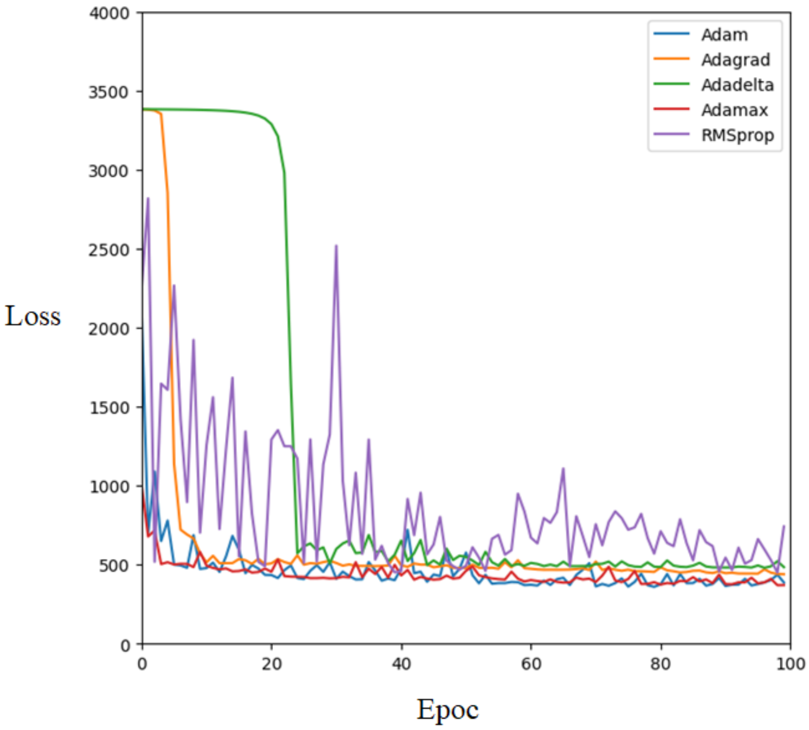


Figure 9. Loss trend during model training with different network optimizers.

The number of epochs was not consistent due to the application of an early stopping technique. Therefore, the first 100 training epochs are selected to represent the training steps of the model. Figure 9 illustrates that Adam was the fastest converging optimizer during training. Interestingly, the Adam optimizer yields the lowest loss in the hybrid model. The reason behind this is Adam combines the advantages of both AdaGrad and RMSProp. It utilizes the same learning rate for each parameter and adapts them independently as the learning progresses. According to the R^2 values in Table 10, the two worst-performing optimizers were Adadelta and RMSprop. The RMSprop alone may not effectively converge to a stable state, leading to significant fluctuations during training. Although Adadelta appeared to converge after 60 epochs, it became trapped in the local optima. However, the choice of a suitable optimizer depends on the model and problem definition.

The experiments show that the depth of the multi-kernel blocks affects the predictive performance. This is illustrated in Figure 10, which shows the effect of the depth of the multi-kernel blocks. The experiment used three different sizes of kernels to extract the feature maps.

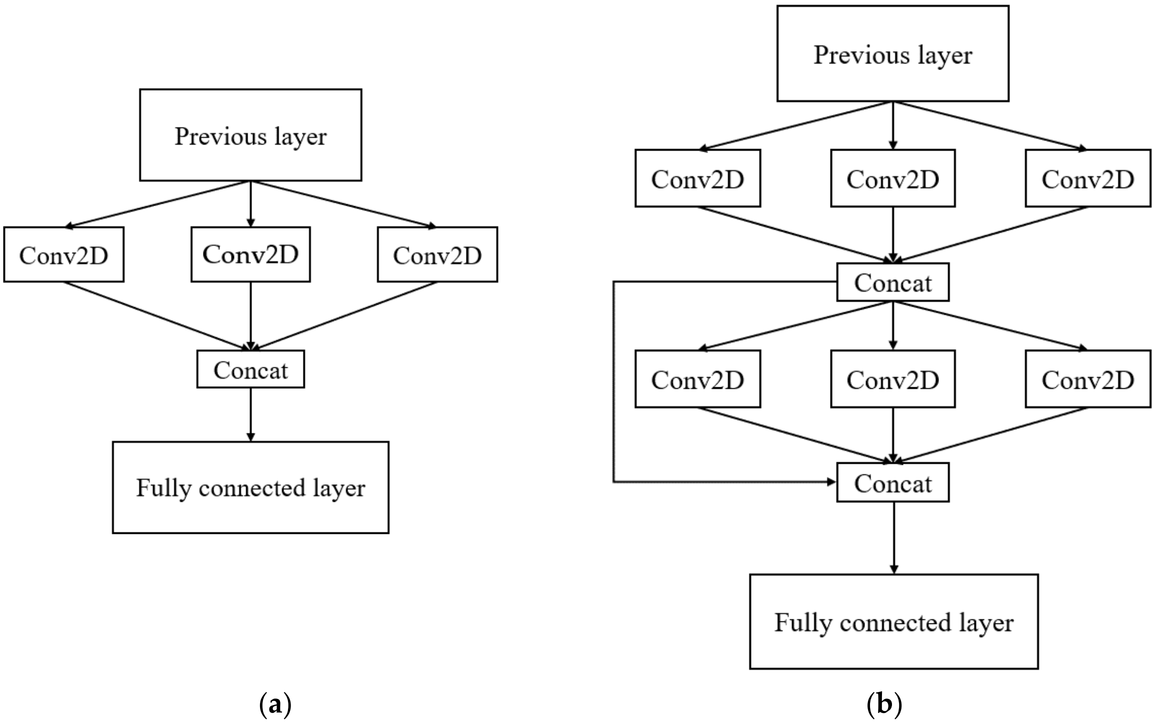


Figure 10. An explanation of different depths of multi-kernel blocks. (**a**) Depth = 1; (**b**) Depth = 2.

Table 11 describes the results for different depths of the multi-kernel blocks. The results indicated that the optimal depth was 3. We can also see that as the depth of the multi-kernel blocks increases, the model performance also improves. However, if the depth of a multi-kernel block exceeds a certain threshold, the overall performance decreases. As shown in Table 11, an infinite increase in the depth of the multi-kernel block leads to overfitting problems rather than the extraction of more useful feature maps. This can result in high accuracy in the training set but poor performance in the testing and validation sets. In addition, as the depth increased, the model training time increased.

As a part of the hybrid model, it is necessary to determine the optimal number of Bi-LSTM neurons. We experimented with five different numbers of neurons in the Bi-LSTM and set up only one Bi-LSTM layer. The results are listed in Table 12. As a result, the loss value gradually decreases with neurons in the Bi-LSTM. However, increasing the number of neurons beyond 64 did not effectively improve the model performance (i.e., the improvement range was less than 1%), and the training time increased. Therefore, this study set the optimal number of neurons to 64.

Table 11. Performance with different depths of multi-kernel blocks.

Depth	Loss	R^2	Training Time
1	369.34	0.59	98.54
2	357.89	0.62	103.81
3	344.56	0.64	107.21
4	365.92	0.60	123.17
5	368.53	0.59	127.82

Table 12. Performance with different number of Bi-LSTM neurons.

Neurons	Loss	R^2	Training Time
16	365.58	0.60	97.49
32	355.94	0.62	98.21
64	345.88	0.64	101.12
128	347.55	0.64	112.28
256	346.97	0.64	127.71

5. Conclusions

This study proposed a hybrid deep learning model for rice yield prediction based on drone environmental data. Three different feature analysis methods were applied to the collected data to select the features that contributed the most to model training. The proposed hybrid model combines the regression and classification loss functions of the shared layers using a multi-tasking technique, which improves the model's overall accuracy. The experimental results show that the proposed hybrid model achieved an F1 score of 94% with only a small amount of data. However, most deep learning models show full performance potential when there is a sufficiently large amount of data. Unfortunately, the cost of collecting such data from agricultural fields is high. Therefore, in future work, we will explore the use of generative adversarial networks to augment real data and boost the performance of the proposed hybrid model to the next level.

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

Funding: This research was partially supported by the National Science and Technology Council, Taiwan, R.O.C. (Grant Nos. NSTC 112-2221-E-005-048, NSTC 111-2634-F-005-001, and NSTC 112-2634-F-005-001-MBK).

Institutional Review Board Statement: Not applicable.

Data Availability Statement: The data that support the findings of this study are available from Taiwan Agricultural Research Institute Council of Agriculture, Executive Yuan, but restrictions apply to the availability of these data, which were used under license for the current study and so are not publicly available. Data are, however, available from the authors upon reasonable request and with permission from Taiwan Agricultural Research Institute Council of Agriculture, Executive Yuan.

Acknowledgments: The toolchain prototype for this paper was made possible with the help of several open-source projects. We sincerely appreciate their contributions to the community.

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

Appendix A

Table A1. Performance with Different Numbers of Bi-LSTM Neurons.

Index Feature	Description
NIR	NIR
$reHT$	Canopy height (from DSM)
RGE	Red edge (735 nm)
RED	Red (660 nm)
GRN	Green (550 nm)
BLU	Blue
$NDVI$	$\frac{NIR - RED}{NIR + RED}$
$SAVI$	$1.5 \times \frac{NIR - RED}{NIR + RED + 0.5}$
$OSAVI$	$1.16 \times \frac{NIR - RED}{NIR + RED + 0.16}$
MSR	$\frac{\frac{NIR}{R} - 1}{\sqrt{\frac{NIR}{R} + 1}}$
$SAVIl$	$2 \times \frac{NIR - RED}{NIR + RED + 1}$
$SAVIRE1$	$1.355 \times \frac{NIR - RGE}{NIR + RGE + 0.355}$
$SAVIRE2$	$1.261 \times \frac{NIR - RGE}{NIR + RGE + 0.261}$
$MCARI$	$RE - RED - 0.2 \times (RE - GRN)(\frac{RGE}{RED})$
$NDGR$	$\frac{NIR - GRN}{NIR + GRN}$
$GBNDVI$	$\frac{GRN - BLU}{GRN + BLU}$
$NDBL$	$\frac{NIR - BLU}{NIR + BLU}$
$NDRERED$	NDRE Red
$reVARI$	$\frac{GRN - RED}{GRN + R - BLU} \times 100 + 100$
$CC1$	$\frac{NIR - RGE}{NIR+RED}$
ARI	$NIR \times \left(\frac{1}{GRN} - \frac{1}{RGE}\right)$
$TGI2$	$-0.5 \times (193 \times (RED - GRN) - 108 \times (RED - BLU))$
$GLI2$	$\frac{G - R + G - B}{2 \times G + R + B}$
$NDRE$	$\frac{NIR - RGE}{NIR + RGE}$
$SRGRN$	$\frac{NIR}{GRN}$
$SRBLU$	$\frac{NIR}{BLU}$
$SRGRNRED$	$\frac{GRN}{RED}$
$reHTNDVI$	$reHT \times NDVI$
$PCMSPLOT$	$1 - \left(\frac{0.98 - NDVI}{0.98 - 0.1}\right)^{0.9}$
$PCMS$	$\frac{thresholded\ plot\ pixel\ \#}{whole\ plot\ pixel\ \#}$
$GRNDVI$	$\frac{GRN - RED}{GRN + RED}$
$SRRED$	$\frac{NIR}{RED}$
$SRRGE$	$\frac{NIR}{RGE}$

References

1. Tripicchio, P.; Satler, M.; Dabisias, G.; Ruffaldi, E.; Avizzano, C.A. Towards smart farming and sustainable agriculture with drones. In Proceedings of the 2015 International Conference on Intelligent Environments, Prague, Czech Republic, 15–17 July 2015; pp. 140–143.
2. Boursianis, A.D.; Papadopoulou, M.S.; Diamantoulakis, P.; Liopa-Tsakalidi, A.; Barouchas, P.; Salahas, G.; Kara-giannidis, G.; Wan, S.; Goudos, S.K. Internet of things (IoT) and agricultural unmanned aerial vehicles (UAVs) in smart farming: A comprehensive review. *Internet Things* **2022**, *18*, 100187. [CrossRef]
3. Kamir, E.; Waldner, F.; Hochman, Z. Estimating wheat yields in Australia using climate records, satellite image time series and machine learning methods. *ISPRS J. Photogramm. Remote Sens.* **2020**, *160*, 124–135. [CrossRef]
4. Schwalbert, R.A.; Amado, T.; Corassa, G.; Pott, L.P.; Prasad, P.V.; Ciampitti, I.A. Satellite-based soybean yield forecast: Integrating machine learning and weather data for improving crop yield prediction in southern Brazil. *Agric. For. Meteorol.* **2020**, *284*, 107886. [CrossRef]
5. Nevavuori, P.; Narra, N.; Lipping, T. Crop yield prediction with deep convolutional neural networks. *Comput. Electron. Agric.* **2019**, *163*, 104859. [CrossRef]
6. Maimaitijiang, M.; Sagan, V.; Sidike, P.; Hartling, S.; Esposito, F.; Fritschi, F.B. Soybean yield prediction from UAV using multimodal data fusion and deep learning. *Remote Sens. Environ.* **2020**, *237*, 111599. [CrossRef]
7. Feng, L.; Zhang, Z.; Ma, Y.; Du, Q.; Williams, P.; Drewry, J.; Luck, B. Alfalfa yield prediction using UAV-based hyperspectral imagery and ensemble learning. *Remote Sens.* **2020**, *12*, 2028. [CrossRef]
8. Fei, S.; Hassan, M.A.; Xiao, Y.; Su, X.; Chen, Z.; Cheng, Q.; Duan, F.; Chen, R.; Ma, Y. UAV-based multi-sensor data fusion and machine learning algorithm for yield prediction in wheat. *Precis. Agric.* **2023**, *24*, 187–212. [CrossRef]
9. Quarmby, N.A.; Milnes, M.; Hindle, T.L.; Silleos, N. The use of multi-temporal NDVI measurements from AVHRR data for crop yield estimation and prediction. *Int. J. Remote Sens.* **1993**, *14*, 199–210. [CrossRef]
10. Rashid, M.; Bari, B.S.; Yusup, Y.; Kamaruddin, M.A.; Khan, N. A comprehensive review of crop yield prediction using machine learning approaches with special emphasis on palm oil yield prediction. *IEEE Access* **2021**, *9*, 63406–63439. [CrossRef]
11. Gopal, P.M.; Bhargavi, R. A novel approach for efficient crop yield prediction. *Comput. Electron. Agric.* **2019**, *165*, 104968. [CrossRef]
12. Shekoofa, A.; Emam, Y.; Shekoufa, N.; Ebrahimi, M.; Ebrahimie, E. Determining the most important physio-logical and agronomic traits contributing to maize grain yield through machine learning algorithms: A new avenue in intelligent agriculture. *PLoS ONE* **2014**, *9*, e97288. [CrossRef] [PubMed]
13. Paul, M.; Vishwakarma, S.K.; Verma, A. Analysis of soil behavior and prediction of crop yield using data mining approach. In Proceedings of the 2015 International Conference on Computational Intelligence and Communication Networks (CICN), Jabalpur, India, 12–14 December 2015; pp. 766–771.
14. Jeong, J.H.; Resop, J.P.; Mueller, N.D.; Fleisher, D.H.; Yun, K.; Butler, E.E.; Timlin, D.J.; Shim, K.-M.; Gerber, J.S.; Reddy, V.R.; et al. Random forests for global and regional crop yield predictions. *PLoS ONE* **2016**, *11*, e0156571. [CrossRef] [PubMed]
15. Pantazi, X.E.; Moshou, D.; Alexandridis, T.; Whetton, R.L.; Mouazen, A.M. Wheat yield prediction using machine learning and advanced sensing techniques. *Comput. Electron. Agric.* **2016**, *121*, 57–65. [CrossRef]
16. Cai, Y.; Guan, K.; Lobell, D.; Potgieter, A.B.; Wang, S.; Peng, J.; Xu, T.; Asseng, S.; Zhang, Y.; You, L.; et al. Integrating satellite and climate data to predict wheat yield in Australia using machine learning approaches. *Agric. For. Meteorol.* **2019**, *274*, 144–159. [CrossRef]
17. Elavarasan, D.; Vincent P M, D.R.; Srinivasan, K.; Chang, C.Y. A hybrid CFS filter and RF-RFE wrapper-based feature extraction for enhanced agricultural crop yield prediction modeling. *Agriculture* **2020**, *10*, 400. [CrossRef]
18. Khosla, E.; Dharavath, R.; Priya, R. Crop yield prediction using aggregated rainfall-based modular artificial neural networks and support vector regression. *Environ. Dev. Sustain.* **2020**, *22*, 5687–5708. [CrossRef]
19. Nishant, P.S.; Venkat, P.S.; Avinash, B.L.; Jabber, B. Crop yield prediction based on Indian agriculture using machine learning. In Proceedings of the 2020 International Conference for Emerging Technology (INCET), Belgaum, India, 5–7 June 2020; pp. 1–4.
20. Sharifi, A. Yield prediction with machine learning algorithms and satellite images. *J. Sci. Food Agric.* **2021**, *101*, 891–896. [CrossRef]
21. Liakos, K.; Busato, P.; Moshou, D.; Pearson, S.; Bochtis, D. Machine learning in agriculture: A review. *Sensors* **2018**, *18*, 2674. [CrossRef]
22. Ji, B.; Sun, Y.; Yang, S.; Wan, J. Artificial neural networks for rice yield prediction in mountainous regions. *J. Agric. Sci.* **2007**, *145*, 249–261. [CrossRef]
23. Baral, S.; Tripathy, A.K.; Bijayasingh, P. Yield prediction using artificial neural networks. In *Computer Networks and Information Technologies, Proceedings of the Second International Conference on Advances in Communication, Network, and Computing, CNC 2011, Bangalore, India, 10–11 March 2011*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 315–317.
24. Çakır, Y.; Kirci, M.; Gunes, E.O. Yield prediction of wheat in south-east region of Turkey by using artificial neural networks. In Proceedings of the 2014 The Third International Conference on Agro-Geoinformatics, Beijing, China, 11–14 August 2014; pp. 1–4.
25. Bhojani, S.H.; Bhatt, N. Wheat crop yield prediction using new activation functions in neural network. *Neural Comput. Appl.* **2020**, *32*, 13941–13951. [CrossRef]
26. LeCun, Y.; Boser, B.; Denker, J.S.; Henderson, D.; Howard, R.E.; Hubbard, W.; Jackel, L.D. Backpropagation applied to handwritten zip code recognition. *Neural Comput.* **1989**, *1*, 541–551. [CrossRef]

27. Villanueva, M.B.; Salenga, M.L.M. Bitter melon crop yield prediction using machine learning algorithm. *Int. J. Adv. Comput. Sci. Appl.* **2018**, *9*, 1–6.
28. Monga, T. Estimating vineyard grape yield from images. In *Advances in Artificial Intelligence, Proceedings of the 31st Canadian Conference on Artificial Intelligence, Canadian AI 2018, Toronto, ON, Canada, 8–11 May 2018*; Springer International Publishing: Cham, Switzerland, 2018; pp. 339–343.
29. Khaki, S.; Wang, L. Crop yield prediction using deep neural networks. *Front. Plant Sci.* **2019**, *10*, 621. [CrossRef] [PubMed]
30. Chu, Z.; Yu, J. An end-to-end model for rice yield prediction using deep learning fusion. *Comput. Electron. Agric.* **2020**, *174*, 105471. [CrossRef]
31. Kalaiarasi, E.; Anbarasi, A. Crop yield prediction using multi-parametric deep neural networks. *Indian J. Sci. Technol.* **2021**, *14*, 131–140. [CrossRef]
32. You, J.; Li, X.; Low, M.; Lobell, D.; Ermon, S. Deep gaussian process for crop yield prediction based on remote sensing data. In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, San Francisco, CA, USA, 4–9 February 2017; Volume 31.
33. Khaki, S.; Wang, L.; Archontoulis, S.V. A CNN-RNN framework for crop yield prediction. *Front. Plant Sci.* **2020**, *10*, 1750. [CrossRef]
34. Bengio, Y.; Simard, P.; Frasconi, P. Learning long-term dependencies with gradient descent is difficult. In Proceedings of the IEEE Transactions on Neural Networks, 5 March 1994; pp. 157–166.
35. Rußwurm, M.; Korner, M. Temporal vegetation modelling using long short-term memory networks for crop identification from medium-resolution multi-spectral satellite images. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, Honolulu, HI, USA, 21–26 July 2017; pp. 11–19.
36. Zhong, L.; Hu, L.; Zhou, H. Deep learning based multi-temporal crop classification. *Remote Sens. Environ.* **2019**, *221*, 430–443. [CrossRef]
37. Yu, J.; Zhang, X.; Xu, L.; Dong, J.; Zhangzhong, L. A hybrid CNN-GRU model for predicting soil moisture in maize root zone. *Agric. Water Manag.* **2021**, *245*, 106649. [CrossRef]
38. Graves, A.; Schmidhuber, J. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Netw.* **2005**, *18*, 602–610. [CrossRef]
39. Van Klompenburg, T.; Kassahun, A.; Catal, C. Crop yield prediction using machine learning: A systematic literature review. *Comput. Electron. Agric.* **2020**, *177*, 105709. [CrossRef]
40. Verleysen, M.; François, D. The curse of dimensionality in data mining and time series prediction. In *Computational Intelligence and Bioinspired Systems, Proceedings of the 8th International Work-Conference on Artificial Neural Networks (IWANN), Vilanova i la Geltrú, Barcelona, Spain, 8–10 June 2005*; Springer: Berlin/Heidelberg, Germany, 2005; pp. 758–770.
41. Benesty, J.; Chen, J.; Huang, Y.; Cohen, I. Pearson correlation coefficient. In *Noise Reduction in Speech Processing*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2009; pp. 1–4.
42. Winter, E. The shapley value. In *Handbook of Game Theory with Economic Applications*; Elsevier: Amsterdam, The Netherlands, 2002; Volume 3, pp. 2025–2054.
43. Papadopoulos, S.; Kontokosta, C.E. Grading buildings on energy performance using city benchmarking data. *Appl. Energy* **2019**, *233–234*, 244–253. [CrossRef]
44. 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, San Francisco, CA, USA, 13–17 August 2016; pp. 785–794.
45. Yu, X.; Ergan, S.; Dedemen, G. A data-driven approach to extract operational signatures of HVAC systems and analyze impact on electricity consumption. *Appl. Energy* **2019**, *253*, 113497. [CrossRef]
46. Guyon, I.; Weston, J.; Barnhill, S.; Vapnik, V. Gene selection for cancer classification using support vector machines. *Mach. Learn.* **2002**, *46*, 389–422. [CrossRef]
47. Li, H.; Zhao, W.; Zhang, Y.; Zio, E. Remaining useful life prediction using multi-scale deep convolutional neural network. *Appl. Soft Comput.* **2020**, *89*, 106113. [CrossRef]
48. Sharma, S.; Rai, S.; Krishnan, N.C. Wheat crop yield prediction using deep LSTM model. *arXiv* **2020**, arXiv:2011.01498.
49. Caruana, R. Multitask learning. *Mach. Learn.* **1997**, *28*, 41–75. [CrossRef]
50. Kendall, A.; Gal, Y.; Cipolla, R. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 7482–7491.

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.