machine learning & knowledge extraction

MDPI

Article

Forecasting the Right Crop Nutrients for Specific Crops Based on Collected Data Using an Artificial Neural Network (ANN)

Sairoel Amertet [1,*] and Girma Gebresenbet [2]

[1] High School of Automation and Robotics, Peter the Great Saint Petersburg Polytechnic University, 195220 Saint Petersburg, Russia
[2] Department of Energy and Technology, Swedish University of Agricultural Sciences, P.O. Box 7032, 750 07 Uppsala, Sweden; girma.gebresenbet@slu.se
* Correspondence: sairoel@mtu.edu.et

Abstract: In farming technologies, it is difficult to properly provide the accurate crop nutrients for respective crops. For this reason, farmers are experiencing enormous problems. Although various types of machine learning (deep learning and convolutional neural networks) have been used to identify crop diseases, as has crop classification-based image processing, they have failed to forecast accurate crop nutrients for various crops, as crop nutrients are numerical instead of visual. Neural networks represent an opportunity for the precision agriculture sector to more accurately forecast crop nutrition. Recent technological advancements in neural networks have begun to provide greater precision, with an array of opportunities in pattern recognition. Neural networks represent an opportunity to effectively solve numerical data problems. The aim of the current study is to estimate the right crop nutrients for the right crops based on the data collected using an artificial neural network. The crop data were collected from the MNIST dataset. To forecast the precise nutrients for the crops, ANN models were developed. The entire system was simulated in a MATLAB environment. The obtained results for forecasting accurate nutrients were 99.997%, 99.996%, and 99.997% for validation, training, and testing, respectively. Therefore, the proposed algorithm is suitable for forecasting accurate crop nutrients for the crops.

Keywords: artificial neural network; precision agriculture; big data; validation; forecasting; accurate

Citation: Amertet, S.; Gebresenbet, G. Forecasting the Right Crop Nutrients for Specific Crops Based on Collected Data Using an Artificial Neural Network (ANN). *Mach. Learn. Knowl. Extr.* **2024**, *6*, 1936–1952. https://doi.org/10.3390/make6030095

Academic Editor: Andreas Holzinger

Received: 9 August 2024
Revised: 16 August 2024
Accepted: 19 August 2024
Published: 26 August 2024

1. Introduction

Artificial neural networks (ANNs) can be trained on the historical data of crop yields, weather patterns, soil conditions, and other relevant factors to accurately predict future crop yields. This helps farmers plan their operations more effectively. Artificial neural networks are one of the most important elements of machine learning [1]. They are inspired by the human brain structure, and function as if they are based on interconnected nodes in which simple processing operations take place. The aim of the current paper is to analyze various data sources, such as soil conditions, weather patterns, and historical yields, to predict crop yields with greater accuracy, enabling farmers to optimize resource allocation and maximize production [2]. By analyzing data on soil moisture, nutrient levels, and crop growth stages, ANNs can help farmers determine the optimal timing and amount of irrigation and fertilization, reducing waste and improving resource efficiency [3]. Although the principles of artificial neural networks are incredibly important for different sectors, they have not been applied in precision agriculture. Most likely, different researchers have used CNN and deep learning algorithms for precision agriculture systems in order to identify crop diseases and crop classification [4]. Although those algorithms are frequently used for crop disease identification and classification, they fail to forecast accurate crop nutrients due to the fact that crop nutrients are collected in numbers rather than images [5]. CNN and deep learning tend to be a more powerful and accurate way of solving classification problems based on images. On the other hand, ANN tends to be the most powerful and

accurate way of solving numerical data problems, with image inputs not being necessary [6]. Even if one or more cells of the ANN become corrupted, the generation of outputs will not be affected when a dataset is applied; this is not the case for CNN and deep learning. For these reasons, the authors propose an artificial neural network for forecasting the effects of nutrients on crops and providing good information to the farming system. The objective of the current study is to forecast crop nutrients, providing the best optimal crop nutrients to the crops for precision farming sectors based on the dataset. The crop nutrient dataset used to train, validate, and test the ANN model included rice, maize, chickpeas, kidney beans, pigeon peas, moth beans, mung beans, black gram, lentils, pomegranates, bananas, mangoes, grapes, watermelons, muskmelons, apples, oranges, papayas, coconuts, cotton, jute, and coffee. The novelty of the current study lies in its forecasting of the most accurate crop nutrients based on a numerical array of various crop nutrients. It uses many crop nutrition datasets at the same time, training them together and forecasting the accuracy of the respective crop nutrients. This concept makes the current work unique in comparison to existing works as these were trained on a single crop nutrient dataset.

This paper is organized as follows: A literature review is carried out in Section 2, and mathematical models for an artificial neural network are discussed in Section 3. Results and discussions are presented in Section 4, and conclusions are drawn in Section 5.

2. Literature Review

Neural networks can analyze images of plants and identify early signs of pests or diseases, allowing for timely intervention and the prevention of major crop losses. Neural networks are a key component of agricultural machinery and autonomous systems, which can perform tasks like weeding, harvesting, and mapping field conditions with greater precision and efficiency than manual labor. Neural networks can analyze complex environmental data to help farmers adapt to the effects of climate change, such as changing weather patterns and pest populations [6,7].

Massive parallelism, distributed representation and computing, learning capacity, generalization ability, and adaptivity, which may all appear straightforward but are actually rather complex, are just a few of the amazing qualities of the human brain. For computer scientists, building a machine capable of solving challenging perceptual puzzles at this speed has always been a goal. ANN models are an attempt to use the same technique that the human brain employs to resolve perceptual issues [8,9].

Layered feed-forward ANNs employ the backpropagation algorithm. This algorithm indicates that the artificial neurons are layered, transmitting signals "forward" before faults spread in the opposite direction. Neurons in the input layer provide inputs to the network, and neurons in the output layer provide the network's output. One or more intermediary hidden levels might also exist. By giving the algorithm examples of the inputs and outputs we want the network to compute, the backpropagation algorithm employs supervised learning. The errors in the discrepancy between the actual and expected results is then computed. Reducing this inaccuracy is the goal of the backpropagation algorithm, allowing the ANN to gain knowledge of the training set. The objective of the training process is to fine-tune the initial random weights to minimize error [10,11].

The input that is multiplied by an artificial neuron will be stronger with greater weight. The signal is suppressed by the negative weight because weights can also be negative [12]. The computation of the neuron will vary according to the weights. We can achieve the desired output for particular inputs by varying the weights of an artificial neuron [13], but it would be difficult to determine all of the required weights by hand when dealing with an artificial neural network (ANN) that has hundreds or thousands of neurons [14]. However, there are algorithms that allow us to modify the weights of an ANN and obtain the required output from the network. This process of changing the weights is referred to as training or learning [15]. There are many different kinds of ANNs and applications for them. Hundreds of distinct models that are regarded as ANNs have been built since McCulloch and Pitts' initial neural model in 1943 [16], and there can be variations in the functions,

accepted values, topology, learning techniques, etc., between them [17]. Furthermore, there are several hybrid models in which every neuron has additional qualities beyond the ones we review here [18]. Owing to space constraints, we will only demonstrate one ANN, which is based on the backpropagation method, one of the most widely used models in ANNs, which learns proper weights. Because ANNs handle information, their primary applications are in related domains [19]. An enormous range of ANNs are available for modeling real neural networks, studying animal and machine behavior and control, and designing applications including pattern recognition, forecasting, and data compression.

RNNs excel at processing sequential data, CNNs excel at automatically extracting features from raw data (eliminating the need for manual feature engineering), and tensor flow's computational graph system allows for efficient and flexible model building (although it fails to learn and adapt to new data without explicit programming), making them ideal for complex and dynamic tasks [20]. Although models can continuously learn and improve over time as they are exposed to new data, making them adaptable to changing circumstances and evolving needs, they fail to process information in parallel, making them much faster than traditional algorithms for certain tasks [21]. This is especially beneficial for tasks that require real-time processing, such as precision agriculture data. ANNs can learn and adapt to new data and situations, making them ideal for tasks that are difficult to program through traditional methods, such as data forecasting [22]. ANNs can learn and adapt to new data without being explicitly programmed, making them ideal for complex tasks where traditional algorithms struggle. Conversely, training deep learning models require significant computational resources, including powerful GPUs and large amounts of data. This can be expensive and time-consuming, especially for complex models. ANNs can process information in parallel, making them significantly faster than traditional algorithms for certain tasks, but building and maintaining the infrastructure needed for deep learning, such as high-performance computing clusters and specialized hardware, can be costly. This can be a barrier for individuals and organizations with limited resources. ANNs are relatively fault-tolerant, meaning they can continue to function even if parts of the network are damaged; CNN and deep learning are unable to do this [23]. ANNs can self-regulate their data processing, eliminating the need for complex manual adjustments. Deep learning models can be difficult to interpret, making it challenging to understand how they arrive at their results. This lack of transparency can raise concerns about bias, fairness, and accountability. The CNN, on the other hand, has interpretability challenges, limited effectiveness for sequential data, and tends to be much slower. The ANN is the most powerful tool for numerical datasets, whereas the CNN and deep learning are powerful for image processing. RNNs can suffer from vanishing or exploding gradients, which hinder learning and make it difficult to train the network effectively. This occurs when the gradients of the loss function become too small or too large during backpropagation, making it challenging to update the network's weights accurately [24]. However, ANNs can self-regulate their data processing, eliminating the need for complex manual adjustments. Training RNNs can be computationally expensive, especially for long sequences or large datasets. This is because the network needs to process each element of the sequence individually, making it slower than other types of neural networks; ANNs do not need to do this [25].

3. Mathematical Models of Artificial Neural Network

The soil sensors detect a deficiency in nutrient content in the fields and give the data to ANN algorithms (Figure 1). Then, the ANN algorithm analyzes the crop nutrient content and provides commands to the robot to give optimal nutrients in cases where the deficiency occurred. Based on instructions from the ANN, the robot provides optimal nutrients to the crops. A complete nutrition service is offered by Precise Crop Nutrition, which uses satellite imagery to provide accurate nutrition application plans and maps, supplemented with key advice on optimal rates and timing [26]. The precise N-Maps take raw satellite data and process them in-house to avoid the challenges associated with cloud inclusion. Precise Crop Nutrition also offers variable-rate P, K, Mg, and lime application maps, based

on precision grid soil sampling methodologies and nutrient analysis carried out by an accredited ANN.

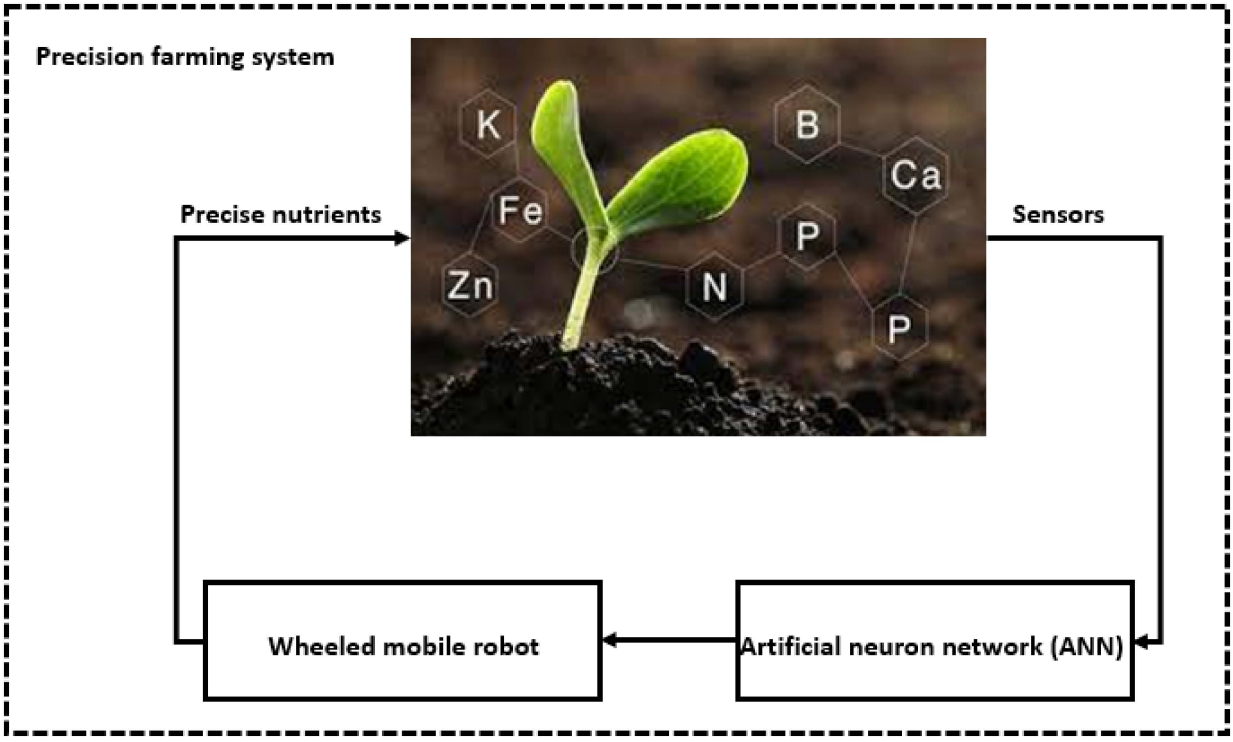

Figure 1. Precision agriculture process [16].

According to the procedures outlined in Figure 2, artificial neural networks (ANNs) are nonlinear statistical data-modeling tools that find complex correlations between input and output. Normalizing the data is the first step in the process. Data normalization is the process of changing data into a specified range. In order to prevent the network from being ill-conditioned, input data in ANNs must be standardized. Achieving the same range of values for every input in the ANN mode is not feasible, guaranteeing a steady convergence of biases and weights. The data partition is the second phase. Random data division (divider) is used in ANN training to use the maximum data when training (generally, splitting the data into training data, validation data, and test data). During the training process, the backpropagation algorithm is used to define the weights on connections and for calculating the outputs. Generally, for some applications, these weights can be used to initialize the neural network and are updated using an online training algorithm. Network weights and biases are updated during training. Validation is used to measure network generalization; when generalization stops improving, it stops training. An independent measure of network performance during and after training is achieved by testing data and has no effect on training [27].

The LMA (Levenberg–Marquardt algorithm) is used as an online training algorithm. The LMA provides a numerical solution to the minimization problem of a nonlinear function. In the field of artificial neural networks, for training small- and medium-sized problems, the LMA is the best option. The LMA is the combination of the steepest descent method and the Gauss–Newton algorithm, combining the speed advantage of the Gauss–Newton algorithm and the stability of the steepest descent method. In many cases, it can converge well even if the error surface is much more complex than in the quadratic situation, making it instantaneous compared to the Gauss–Newton algorithm. In convergent situations, the LMA tends to be a little slower than the Gauss–Newton algorithm, but it converges much faster than the steepest descent method. The basic idea of the LMA is that it executes a combined training process: around areas with a complex curvature,

the LMA switches to the steepest descent algorithm until the local curvature is correct, to make a quadratic approximation, at which point it becomes similar to the Gauss–Newton algorithm, speeding up the convergence significantly [28]. Weights and biases are updated during training, and data are presented according to which network is adjusted based on its error. An independent measure of network performance during and after training is achieved by testing data and has no effect on training. The third step is the architecture of the network, in which a two-layer feed-forward network is applied with a standard function fitting, comprising a sigmoid transfer function in the hidden layer and a linear transfer function in the output layer. The fourth step is the learning algorithm used for training the network to fit the inputs and targets. It helps in achieving accurate results and analysis. The fifth step is the evaluation of the network, which allows for testing of the network using more data, which can be retained if we are not satisfied with the results [29]. The sixth step is to determine the deployable solution; in this way, a trained neural network is generated in the form of a Simulink diagram or code. In this research, this algorithm is implemented because of the ease of model building and it requiring less formal statistical knowledge. Unlike other prediction techniques, ANN does not impose any restrictions (e.g., distribution), and it gives data with a nonconstant difference and high volatility. With the evolving technology of ANNs, motor fault detection problems can easily be solved using an advanced approach based on a useful measurement without the need for expensive equipment and precise mathematical models that are obtained from conventional fault detection techniques. Therefore, it is a more feasible option than any other conventional technique. The nodes in one kind of network are viewed as "artificial neurons", a computational model that draws inspiration from natural neurons. We refer to these as artificial neural networks, or ANNs. Natural neurons have synapses on their membranes, or dendrites, where they receive signals. The neuron is activated and sends out a signal along the axon when the signals it receives are powerful enough to reach a threshold. It is possible that this signal will reach another synapse and cause more neurons to fire.

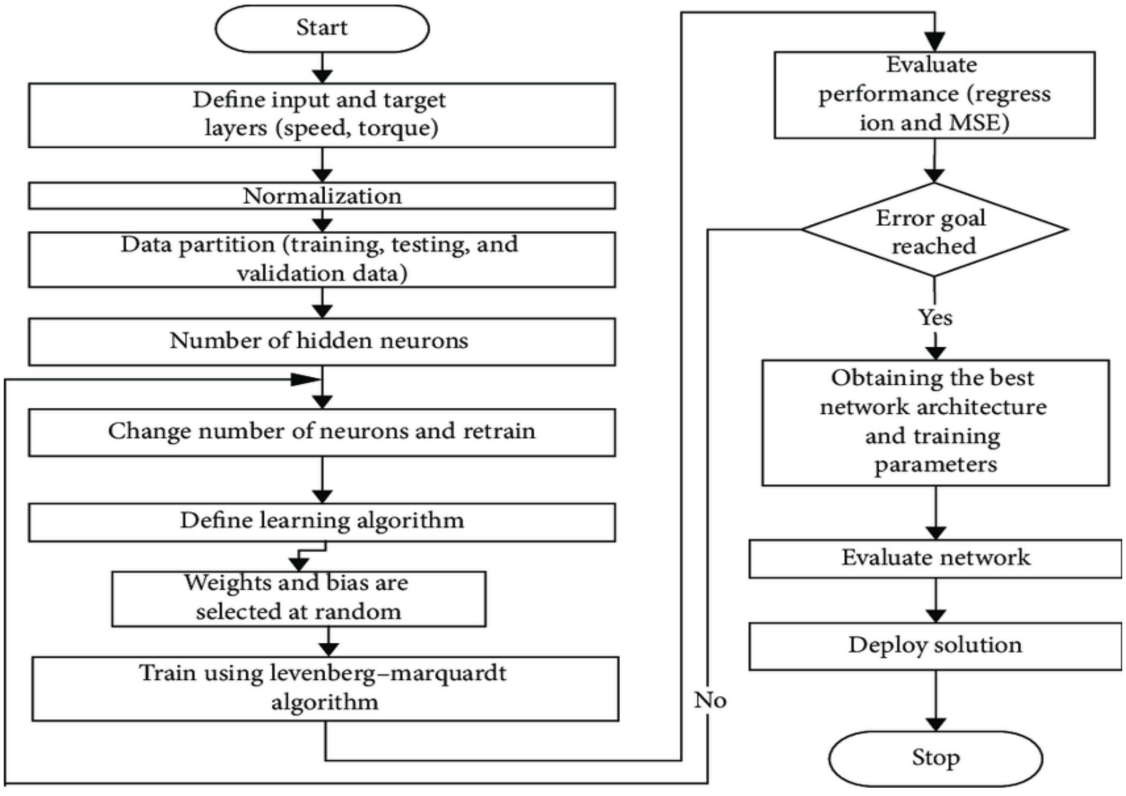

Figure 2. Flow chart of artificial neural network (ANN) [13].

When modeling artificial neurons, the intricacy of genuine neurons is greatly abstracted. These essentially consist of inputs (such as synapses) multiplied by weights (the strength of the corresponding impulses), which are then calculated by a mathematical formula that determines whether the neuron will activate (Figure 3). An additional function, which could be the identity, computes the artificial neuron's output, sometimes based on a threshold. Artificial neurons are combined in ANNs to process information.

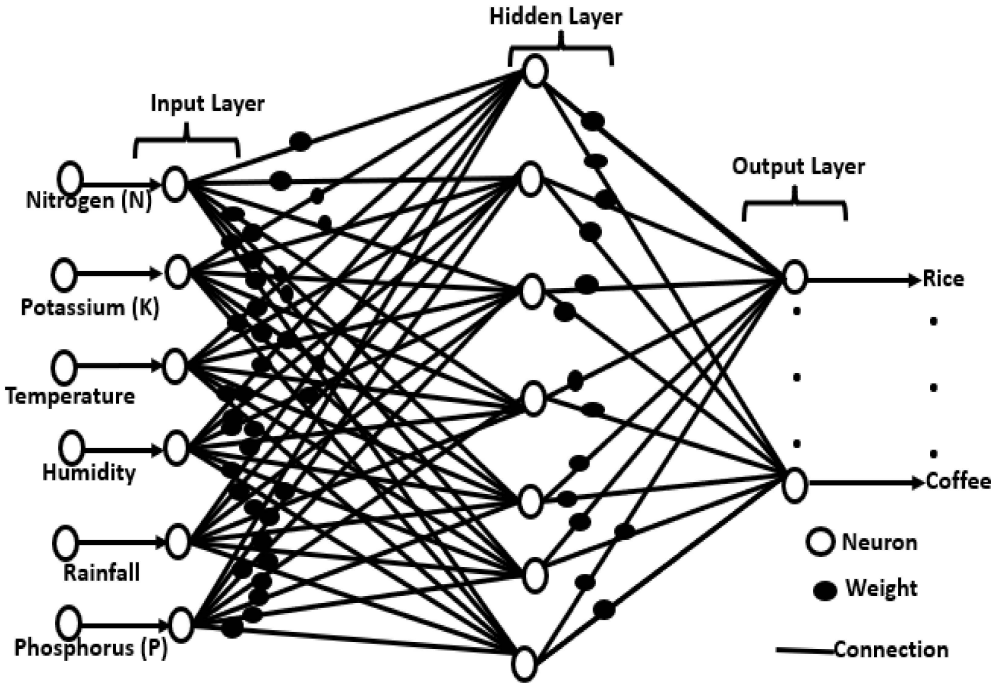

Figure 3. An artificial neuron model [13].

The algorithm is the given instances of the sources of inputs and outputs required for the framework to enlist, and the error generated afterwards can then be found. It is likely that this algorithm will diminish this error until the point that the artificial neural network takes in the training information. The sum of the inputs x_i is multiplied with corresponding weights $w_{j,i}$ when generating activation function.

$$A_f(\overline{x},\ \overline{w}) = \sum_{i=0}^{n} x_i w_{j,\ i} \tag{1}$$

Sigmoidal function is used as the output function for this work.

$$O_j(\overline{x}, \overline{w}) = \frac{1}{1 + e^{A_f(\overline{x},\ \overline{w})}} \tag{2}$$

We must adjust the weights in order to reduce error since the error refines between actual and predicted outcomes based on the weights. The error function for each neuron's output can be represented as follows:

$$E_f(\overline{x},\ \overline{w}\ ,\ d) = \left(O_j(\overline{x}, \overline{w}) - d_j\right)^2 \tag{3}$$

Then, we find how error value depends on the input's outputs and weights.

$$\Delta w_{j,\ i} = -\mu \frac{\partial E}{\partial w_{j.\ i}} \tag{4}$$

$\Delta w_{j,\,i}$ is the adjustment of each weight and "η" is the constant eta. Now, we find out the degree to which error depends on output.

$$\frac{\partial E}{\partial O_j} = 2\left(O_j - d_j\right) \tag{5}$$

To find the degree to which the output depends upon the activation and on weights, we compute the following:

$$\frac{\partial O_j}{\partial w_{j,\,i}} = \frac{\partial O_j}{\partial A_j}\frac{\partial A_j}{\partial w_{j,\,i}} = O_j\left(1 - O_j\right)x_i \tag{6}$$

The difference with respect to each weight will be

$$\Delta w_{j,\,i} = -2\mu\left(O_j - d_j\right)O_j\left(1 - O_j\right)x_i \tag{7}$$

If we must change MSE_{ik}, theweights (MSE_{ik}) of a past layer, we expect first to register how the error depends not on the weight, but on the observation from the past layer, i.e., supplanting w by x, as shown in the equation below.

$$\Delta MSE_{ik} = -\mu\frac{\partial E}{\partial MSE_{ik}} = -\mu\frac{\partial E}{\partial x_{j,i}}\frac{\partial x_j}{\partial MSE_{ik}} \tag{8}$$

$$\frac{\partial E}{\partial w_{j,\,i}} = -2\mu\left(O_j - d_j\right)O_j\left(1 - O_j\right)w_{ji} \tag{9}$$

$$\frac{\partial x_i}{\partial v_{j,\,i}} = x_i(1 - x_i)MSE_{ik} \tag{10}$$

Although there is no certain rule for estimating the number of hidden neurons, an empirical formula proposed by many researchers is used [13–18].

$$N_H = \frac{1}{2}(N_I + N_O) + \sqrt{N_T} \tag{11}$$

where N_I, N_O, N_T, and N_H are the number of input neurons, output neurons, train data, and the number of hidden neurons, respectively. In this study, a 90/10 training/validation split rule was adopted to avoid overfitting. The crop nutrients were split into 1980 and 220 for the training process. Six crop nutrients were taken separately as an unseen test dataset to confirm the model accuracy. Using Equation (11), the number of hidden neurons can be approximated as 10. The training was carried out through trial and error for hidden neurons, starting from 2 to 30, while tracking the performance criteria of mean square error, shown in Equation (12). Precision agriculture is currently popular, assisting farmers in making well-informed decisions about their farming approach. Here, the authors were given access to a dataset that would enable users to create a predictive model that would suggest, in light of different factors, which crops would be best suited for production on a specific farm [26]. This dataset was created by enhancing existing crop-related records on rainfall, climate, and fertilizer. The current work makes use of the following essential elements: temperature in degrees Celsius, relative humidity in percentage, soil pH value, rainfall in millimeters, and the ratios of nitrogen, phosphorus, and potassium in soil (N, P, and K). The authors used these components to obtain the 2200 crop dataset from the AMNIST database, which they then used to conduct the present research.

Multiple R is the Pearson correlation coefficient; it is a value that tells you how strong the linear relationship is. The R square coefficient of determination (Multiple Rx Multiple R) gives the amount of variance the dependent variable can account for by the independent variable. The adjusted R square takes into account the number of independent variables in the analysis and corrects for bias. The standard error of the regression is the average

distance that the observed values fall by from the regression line. A fundamental concern in ANNs is the measure of forecasting error (goodness of fit or how well the forecasting model) for a given dataset and ANN method. Among the best measurement methods are mean square error (MSE), mean absolute deviation (MAD), and mean absolute percent error (MAPE) [27].

$$SE = \sum_{t=1}^{n} \frac{(x_t - \hat{x_t})^2}{n} \tag{12}$$

$$MPE = \sum_{t=1}^{n} \frac{\left(\frac{(|x_t - \hat{x_t}|)^2}{x_t}\right)}{n} \times 100\% \tag{13}$$

$$MAD = \frac{\sum_{t=1}^{n}(|x_t - \hat{x_t}|)}{n} \tag{14}$$

$$\%improvment = \left(\frac{old\ value - new\ value}{old\ value}\right) \times 100 \tag{15}$$

4. Results and Discussion

Figure 4 illustrates how the quantity of hidden nodes affects an ANN's performance. It is evident that the ANN's capacity for prediction is mostly unaffected by the quantity of hidden layer nodes. A network with a single hidden layer node can, nonetheless, map the underlying relationship well. Prediction errors are quite consistent, suggesting that overtraining is not an issue for networks with more hidden layer nodes. Given that cross-validation is the stopping condition, this is to be expected. The network with five hidden layer nodes has the lowest prediction error, as seen in Figure 4. Nonetheless, the network with two hidden layer nodes is regarded as ideal, as it has fewer connection weights and a prediction error that is comparable to the network with five hidden layer nodes (roughly a 0.5% error difference).

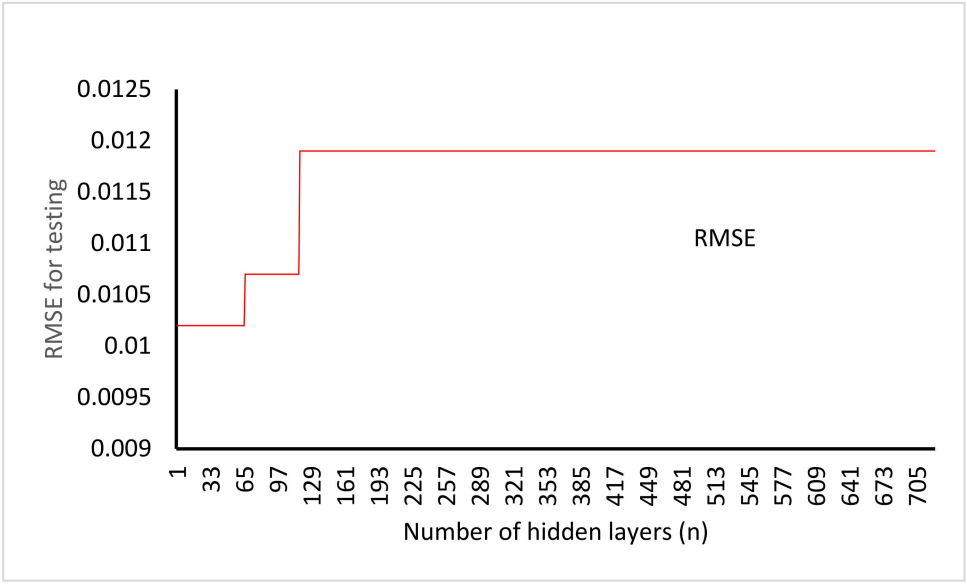

Figure 4. Artificial neural network model's performance with various hidden layer nodes (learning rate $= 0.2$).

Figure 5 shows how the ANN model's performance is mostly unaffected by a variety of parameters, especially in the 0.01–0.6 range. A value of 0.7 for the number of nutrients produced the best forecast. It illustrates the impact of the learning rate and number of factors for the internal parameters that regulate the backpropagation algorithm on model performance.

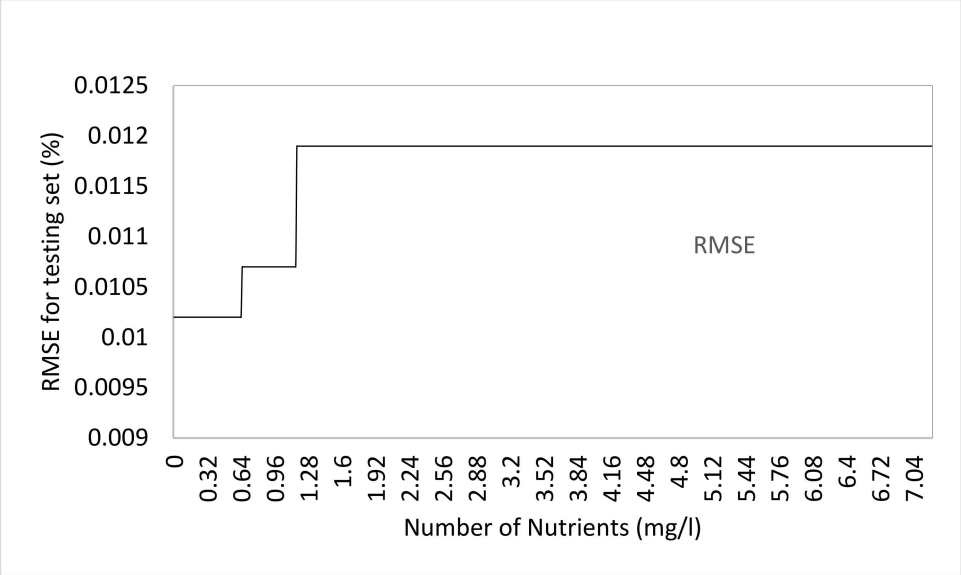

Figure 5. Impact of different element terms on the performance of artificial neural networks (learning rate = 0.2, hidden layer = 2).

It was discovered that the ideal learning rate was 0.2 (as shown in Figure 6), which illustrates the impact of the learning rate and number of factors of the internal parameters that regulate the backpropagation algorithm on model performance.

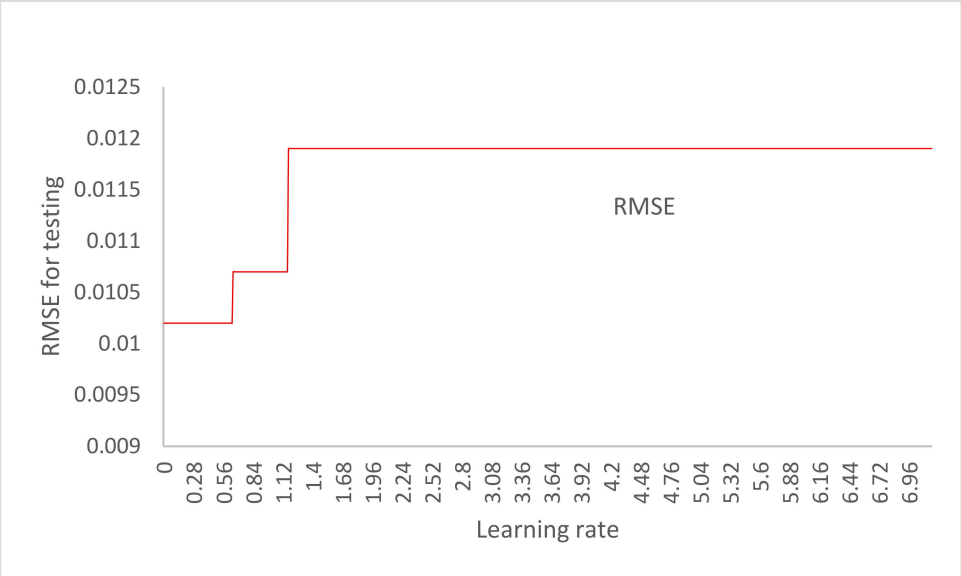

Figure 6. Effect of various learning rates on artificial neural network performance (learning rate = 0.2, hidden layer = 2).

Prediction errors were larger at lower learning rates, most likely because the networks were unable to break out of local maxima on the error surface because of the small step sizes used. Prediction errors somewhat increased with higher learning rates, presumably as a result of the optimization algorithm's pseudorandom behavior close to the local maxima in the error surface brought on by the large step sizes used in the weight space. The ANN trained with two hidden layer nodes, a number of factors value of 0.8, and a learning rate of 0.2 showed minimal effect from varying random starting positions in weight space on prediction inaccuracy. The fact that the error surface in weight space is comparatively simple for the task at hand could be one explanation for this. Furthermore, as has been

previously stated, the model is likely to avoid local maxima in the error surface during training if a learning rate of 0.2 is used.

For all crop nutrients, the trained results demonstrated the same results with no difference throughout the data training. The value of the regression square (R) is presented in Figure 7. Regression squares indicate that the crops' independent (nutrients) variables can explain, for example, how crop training values of regression (R) (99.997%) change all the dependent (crop growth, crop health) variables, validation values of regression (R) (99.996%), and test values of regression (R) (99.997%), and all regression (R) (99.997%) change the crops dependent variables. These results were for rice, maize, chickpeas, kidney beans, pigeon peas, moth beans, mung beans, black gram, lentils, pomegranates, bananas, mangoes, grapes, watermelons, muskmelons, apples, oranges, papayas, coconuts, cotton, jute, and coffee.

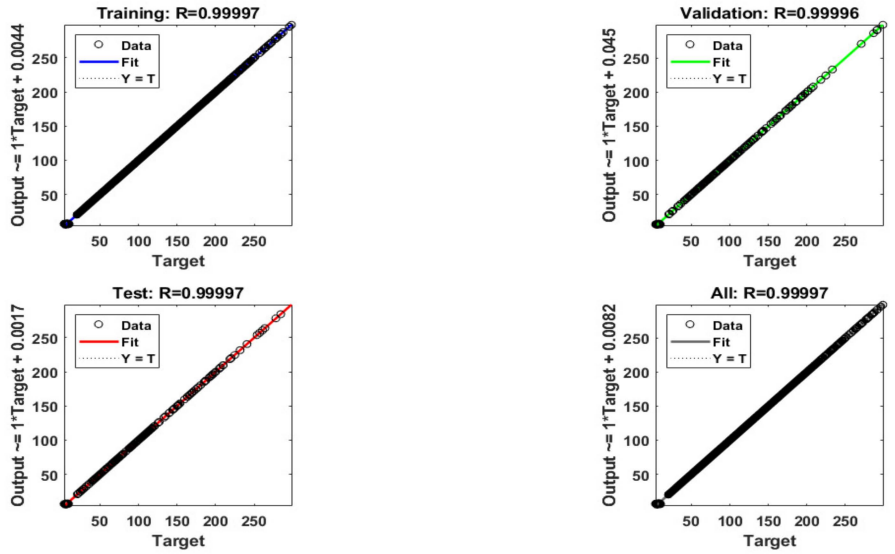

Figure 7. Regression of the system.

The mean squared errors ($\frac{mg}{L}$) of the crops are depicted in Figure 8. The blue line indicates the data train, the green line shows data validation, the red line gives the data test, and the dotted hidden line represents the best fit of the data. Total epochs were 84, and the best validation performance was obtained at epoch 78 with a mean square error (MSE) of 0.999935. The set of sensor values that are predicted by the neural network based on the input data is compared with the expected sensor values, and their difference is calculated to find the mean square error or mean square deviation. Mean square deviation values that are close to zero are good estimators. These results are related to the learning process network. As seen from the presented characteristics, a network of relatively good quality was obtained. Confirmation of this should be reflected in the results obtained in the test sample. Conducted tests showed quality indicators and percentages of correct classification of 95% for beta distribution and 90% for normal distribution. These indicators were lower than those obtained in the learning process, but this is in large part due to the small testing set.

The greatest change in the crop dependent variables (gradient) is shown in Figure 9. The best crop gradient was 2.16, obtained at epoch 84. Crop gradient represents the slope of the tangent of a graph of a function. It gives the direction in which there is a high rate of increase for the considering function (nutrients). "mu" is the control parameter for the backpropagation neural network that is modeled, and the choice of mu directly affects the error convergence. A crop validation check is used to terminate the learning of the neural network. The number of validation checks will depend on the number of successive iterations of the neural network. The values for "mu" and validation checks were 0.001 and 6 at epoch 84, respectively.

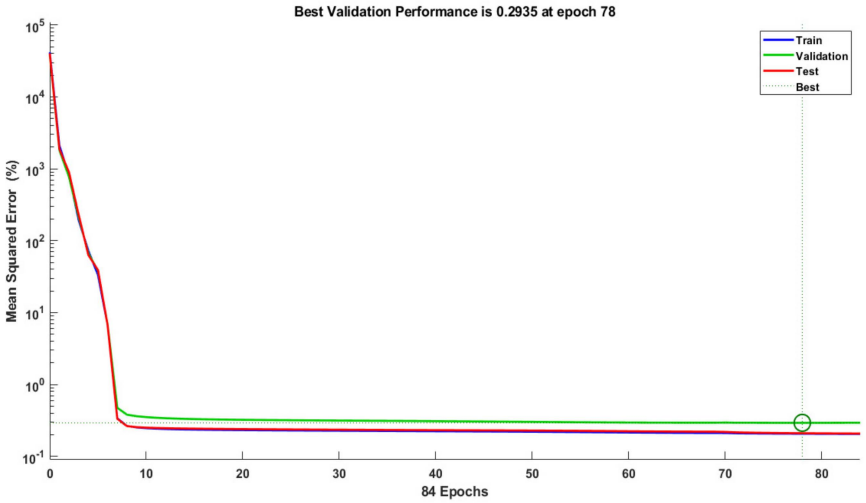

Figure 8. Performance of the system.

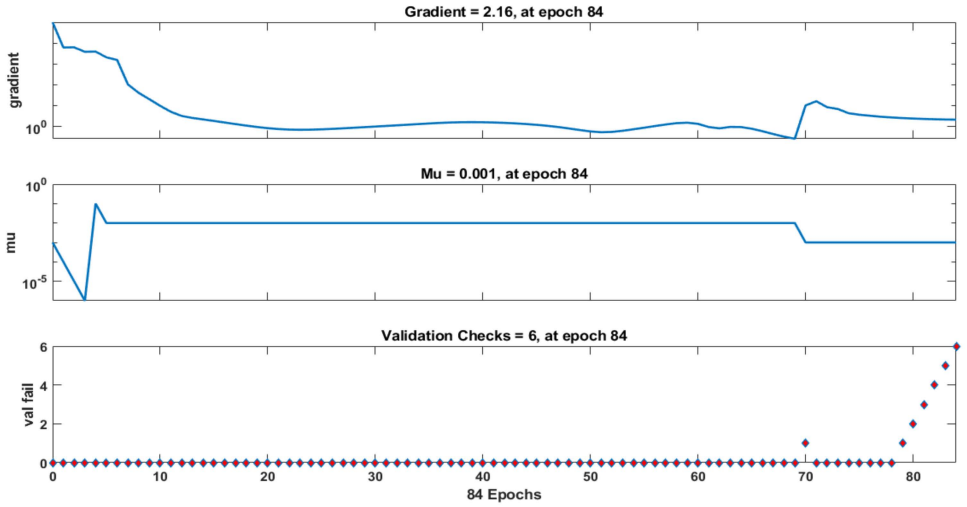

Figure 9. Gradient of the system.

Table 1 summarizes the prediction performance of the ideal neural network model, which consists of two hidden layer nodes, an element value of 0.7, and a learning rate of 0.2. The validation set's r^2 of 0.999 ($\frac{mg}{L}^2$), RMSE of 0.999935 ($\frac{mg}{L}$), and MSE of 0.999935 ($\frac{mg}{L}$) show that the ANN model operates satisfactorily. Additionally, it demonstrates that the model's validation findings are mostly in line with its training and testing outcomes, suggesting that the model can generalize within the training dataset. Obtained results for training, validation, and test for regression (R) were 0.99997 $\frac{mg}{L}$, 0.99996 $\frac{mg}{L}$, and 0.99997 $\frac{mg}{L}$, respectively. This implies that the relationship of crop nutrients is strongly important for crop growth. The magnification of regression (R) for training was 0.99994 $\frac{mg}{L}$; for validation, it was 0.99992 $\frac{mg}{L}$; and for test, it was 0.99994 $\frac{mg}{L}$. This gave the variance of the crops that the dependent variable (crop growth, crop health) can account for with the independent variable (crop nutrients). Absolute errors were obtained, with 0.99997 $\frac{mg}{L}$, 0.99996 $\frac{mg}{L}$, 0.99997 $\frac{mg}{L}$ for training, validation, and test, respectively. Absolute deviation for training, validation, and test were 0.00045453 $\frac{mg}{L}$ each. The smaller the deviation, the better the model prediction, so the ANN model was excellent at forecasting crop nutrients. MSEs for training, validation, and test were 0.999935 $\frac{mg}{L}$, 0.999934 $\frac{mg}{L}$, and 0.999935 $\frac{mg}{L}$, whereas MADs for training, validation, and test were 0.999968 $\frac{mg}{L}$, 0.999967 $\frac{mg}{L}$, and 0.999968 $\frac{mg}{L}$, and MAPEs were 0.045453 $\frac{mg}{L}$ each.

Table 1. Artificial neural network results.

Specifications	R ($\frac{mg}{L}$)	R Square $(\frac{mg}{L})^2$	Absolute Error ($\frac{mg}{L}$)	Absolute Deviation ($\frac{mg}{L}$)	MSE ($\frac{mg}{L}$)	MAD ($\frac{mg}{L}$)	MAPE ($\frac{mg}{L}$)
Training	0.99997	0.99994	0.99997	0.045453	0.999935	0.999968	0.045453
Validation	0.99996	0.99992	0.99996	0.045453	0.999934	0.999967	0.045452
Test	0.99997	0.99994	0.99997	0.045453	0.999935	0.999968	0.045453
All R	0.99997	0.99994	0.99997	0.045453	0.999935	0.999968	0.045453

At epoch 1000, the best performance of crop nutrient prediction was 0.21068, which is well within the R values (Figure 10). At first, it was dynamically dropping, reaching an equilibrium at 300 epochs, and fluctuating between 350 and 550 epochs.

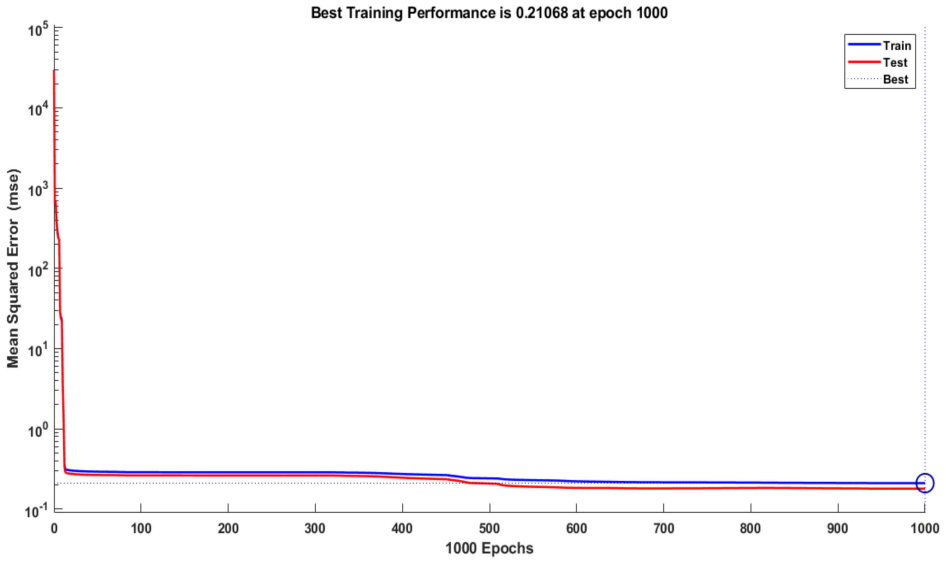

Figure 10. Performance of the system at 1000 epochs.

As shown in Figure 11, at epoch 1000, the gradient, mu, number of parameters, sum squared parameters, and validation check were 0.046278, 50, 80.5641, 31.498, and 0, respectively. When training a neural network using the train function in MATLAB, a validation check = 0 means that the algorithm might be failing to perform validation checks properly, leading to unexpected results at epoch 1000.

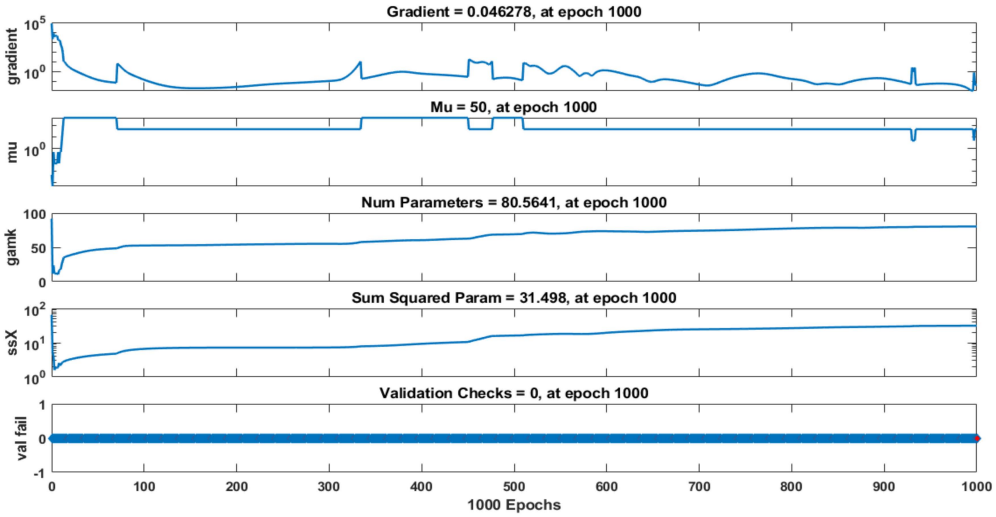

Figure 11. Parameter values at 1000 epochs.

In Figure 12, the values of the regressions (R) were excellent. In all cases, the values of R were the same as 0.99997 without validation. As the number of epochs increased, the degree of prediction decreased, due to the validation check approaching zero.

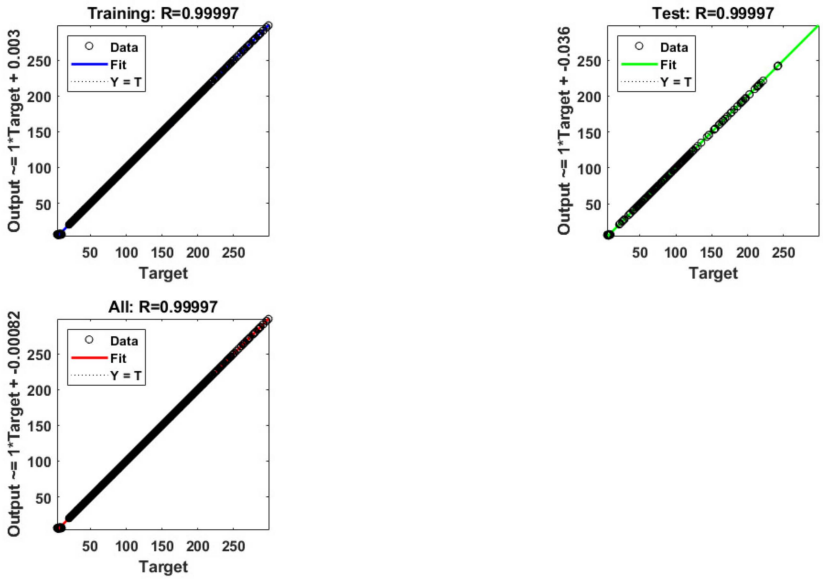

Figure 12. Regression of the system at 1000 epochs.

The ANN predictions were compared against each other using MSE, MAPE, and MAD. Of the three measurements, MAD provides the most accurate and fair comparison of ANN methods (Table 2). MSE, MAPE, and MAD for linear regression were 0.8 $\frac{mg}{L}$, 0.723 $\frac{mg}{L}$, and 0.83 $\frac{mg}{L}$; this gave the linear relationship between a dependent (crop growth, crop health) variable and one or more independent variables (crop nutrients). For exponential regression, these were 0.77 $\frac{mg}{L}$, 0.79 $\frac{mg}{L}$, and 0.87 $\frac{mg}{L}$. From exponential regression, situations where growth starts slowly and then accelerates rapidly could be seen. MSE, MAPE, and MAD for cubic regression were 0.5567 $\frac{mg}{L}$, 0.78 $\frac{mg}{L}$, and 0.86 $\frac{mg}{L}$, with cubic regression in this context implying a relationship between variables, allowing for predictions based on the data. The ANN first-order function links were 0. 999935 $\frac{mg}{L}$, 0.999934 $\frac{mg}{L}$, and 0.999935 $\frac{mg}{L}$. MSE, MAPE, and MAD for single layer multiple input were 0. 999935 $\frac{mg}{L}$, 0.00045453 $\frac{mg}{L}$, and 0.999968 $\frac{mg}{L}$.

Table 2. Error calculation for training set.

Specification	Linear Regression	Exponential Regression	Cubic Regression	ANN First-Order Function Link	Single Layer Multiple Input
MSE $\left(\frac{mg}{L}\right)$	0.8	0.77	0.5567	0.999935	0.999935
MAPE $\left(\frac{mg}{L}\right)$	0.723	0.79	0.78	0.00045453	0.00045453
MAD $\left(\frac{mg}{L}\right)$	0.83	0.87	0.86	0.999968	0.999968

The holdout method is a common technique in model evaluation where the dataset is split into training and test sets (Table 3). During holdout sampling, MSE, MAPE, and MAD for linear regression were 0.6 $\frac{mg}{L}$, 0.43 $\frac{mg}{L}$, and 0.12 $\frac{mg}{L}$; for exponential regression, they were 0.44 $\frac{mg}{L}$, 0.6 $\frac{mg}{L}$, and 0.22 $\frac{mg}{L}$; for cubic, they were 0.34 $\frac{mg}{L}$, 0.14 $\frac{mg}{L}$, and 0.23; for ANN first-order function link, they were 0.999930 $\frac{mg}{L}$, 0.00045453 $\frac{mg}{L}$, and 0.999961 $\frac{mg}{L}$; and for single layer multiple input, they were 0. 999935 $\frac{mg}{L}$, 0.00045453 $\frac{mg}{L}$, and 0.999968 $\frac{mg}{L}$.

Table 3. Error calculation for the holdout sample.

Specification	Linear Regression	Exponential Regression	Cubic Regression	ANN First-Order Function Link	Single Layer Multiple Input
MSE $\left(\frac{mg}{L}\right)$	0.6	0.44	0.34	0.999930	0.999930
MAPE $\left(\frac{mg}{L}\right)$	0.12	0.6	0.14	0.00045451	0.00045451
MAD $\left(\frac{mg}{L}\right)$	0.43	0.22	0.23	0.999961	0.999961

Table 4 shows the comparison of the current results and existing works' results with the same crop nutrients history. The square of regression (R) for the current work over the existing work improved by 15.6%, 15.9%, and 22.1% for the datasets training, testing, and validation, respectively. The root mean squared error (RMSE) improvements by the current works over the existing were training set: 90.01%, testing set: 90.1%, and validation set: 90.9%. Results obtained for mean absolute errors were training set: 85.4%, testing set: 84.4%, and validation set: 88.6%. Even though the current works and the existing works were tested using the same model (ANN), the existing works were not able to demonstrate the best performance. This implies that the existing works mostly lead to underfitting and overfitting, whereas the current work demonstrated generalized forms, since the dataset used win the existing works was relatively small and the attributes were very few.

Table 4. Comparison of the current work with existing work.

Data Set	Current Work			Existing Works [30]			Comparison		
	R Square $\left(\frac{mg}{L}\right)^2$	RMSE $\left(\frac{mg}{L}\right)$	MAE $\left(\frac{mg}{L}\right)$	R Square $\left(\frac{mg}{L}\right)^2$	RMSE $\left(\frac{mg}{L}\right)$	MAE $\left(\frac{mg}{L}\right)$	% R Square of Current Work over Existing Work	% RMSE of Current Work over Existing Work	% MAE of Current Work over Existing Work
Training set	0.99994	0.999935	0.999968	0.865	10.01	6.87	15.6	90	85.4
Testing set	0.99992	0.999934	0.999967	0.863	10.12	6.43	15.9	90.1	84.4
Validation set	0.99994	0.999935	0.999968	0.819	11.01	8.78	22.1	90.9	88.6

Table 5 shows the results of regression and its RMSE to estimate certain crop nutrients based on datasets. From the results, the conclusions were drawn that low bias and low variance were obtained, since individual regression results were similar to the overall trained dataset. The models were properly furcated into the right crop nutrients. The model estimated the amount of the required nutrients for the respective crops. Based on these, the regression values for rice, maize, chickpeas, kidney beans, pigeon peas, moth beans, mung beans, black gram, lentils, pomegranate, bananas, mangoes, grapes, watermelon, muskmelon, apple, orange, papaya, coconut, cotton, jute, and coffee were 0.996, 0.9959, 0.9958, 0.9959, 0.996, 0.9959, 0.9958, 0.9959, 0.9978, 0.9968, 0.996, 0.9959, 0.9958, 0.9959, 0.996, 0.9959, 0.9958, 0.9959, 0.9978, 0.9968, 0.9958, and 0.9966, respectively. These imply that proper nutrients for the crops were forecasted, since during the estimation, the obtained root mean squares were relatively small. These were 0.0056, 0.0067, 0.008, 0.0012, 0.0056, 0.0077, 0.0056, 0.0067, 0.008, 0.0012, 0.0056, 0.0077, 0.0055, 0.0089, 0.0034, 0.0056, 0.0067, 0.008, 0.0012, 0.0056, 0.0077, and 0.0056 for rice, maize, chickpeas, kidney beans, pigeon peas, moth beans, mung beans, black gram, lentils, pomegranate, bananas, mangoes, grapes, watermelon, muskmelon, apple, orange, papaya, coconut, cotton, jute, and coffee, respectively. These imply that the obtained results were relatively excellent at estimating the right crop nutrients for the right crops since they showed high regression value and less error.

Table 5. Estimated outputs and accuracy analyses of the ANN input variables.

Crops	Regression Results ($R^2\left(\frac{mg}{L}\right)^2$)	RMSE ($\frac{mg}{L}$)
Rice	0.996	0.0056
Maize	0.9959	0.0067
Chickpeas	0.9958	0.008
Kidney beans	0.9959	0.0012
Pigeon peas	0.996	0.0056
Moth beans	0.9959	0.0077
Black gram	0.9958	0.0056
Mung beans	0.9959	0.0067
Lentils	0.9978	0.008
Pomegranate	0.9968	0.0012
Bananas	0.996	0.0056
Mangoes	0.9959	0.0077
Grapes	0.9958	0.0055
Watermelon	0.9959	0.0089
Muskmelon	0.996	0.0034
Apple	0.9959	0.0056
Orange	0.9958	0.0067
Papaya	0.9959	0.008
Coconut	0.9978	0.0012
Cotton	0.9968	0.0056
Jute	0.9958	0.0077
Coffee	0.9966	0.0056

5. Conclusions

In this study, we addressed the application of artificial neural networks in precision agriculture. The ability of artificial neural networks (ANNs) to forecast suggested crop nutrients was demonstrated using a backpropagation neural network. A database was compiled from the MNIST data collection, which includes 2200 dataset instances of rainfall, temperature, humidity, phosphorus, potassium, and nitrogen. Based on the ANN used to identify the crop nutrients, the number of epochs was 84, and the best validation performance was obtained at epoch 78. Three conventional techniques were used to compare the outcomes between the measured and predicted crop nutrients that were produced through the use of ANNs. According to the findings, backpropagation neural networks might reasonably accurately forecast crop condition variables and crop nutrients, with an R square = 0.99994 $\frac{\text{mg}}{\text{L}}$, RMSE = 0.999935$\frac{\text{mg}}{\text{L}}$, and MAPE = 0.00045453$\frac{\text{mg}}{\text{L}}$. With a learning rate of 0.2, it was clear that the number of hidden layer nodes had little bearing on the ANN's prediction ability. Even so, a network with just one hidden layer node is still able to accurately map the underlying relationship. The consistency of prediction errors indicates that overtraining does not pose a problem for networks with a higher number of hidden layer nodes. This is expected as cross-validation is the ending condition. A wide range of parameters, particularly in the 0.01–0.6 range, mostly have no effect on the performance of the ANN model. The best forecast was obtained when the number of nutrients was at 0.7. Of the three metrics, MAD offers the most precise and equitable evaluation of ANN techniques. We were informed of a linear relationship between a dependent variable (crop growth, crop health) and one or more independent variables (crop nutrients) by the linear regression's MSE, MAPE, and MAD values of 0.8 mg/L,

0.723 mg/L, and 0.83 mg/L. The first-order function link for the ANN, however, was 0.999935 mg/L, 0.999934 mg/L, and 0.999935 mg/L. For a single layer multiple input, the MSE, MAPE, and MAD values were 0.9999935 mg/L, 0.00045453 mg/L, and 0.999968 mg/L. The results of the forecasting comparison illustrate how the most basic neural network can outperform for crop nutrient identification. The first-order functional link network yielded the best results for the training set. The results showed that one example of the power neural networks can predict the best crop nutrients. The novelty of the current paper is that it uses various crop nutrition type datasets at the same time, training them at the same time and forecasting the accuracy of the respective crop nutrients. Since the existing literature focuses on single crop nutrients when forecasting accurate crop nutrients, current algorithms were used for multipurpose crop nutrition forecasting. Therefore, the proposed algorithms are suitable for forecasting precision crop nutrients.

Author Contributions: S.A.: Visualization, writing, original draft, review and editing, conceptualization, data curation, formula analysis, investigations, methodology, software, validation, project management. G.G.: Supervision, validation, funding, acquisitions, project management, review and editing, conceptualization. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data are contained within the article.

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

References

1. Kujawa, S.; Niedbała, G. Artificial neural networks in agriculture. *Agriculture* **2021**, *11*, 497. [CrossRef]
2. 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]
3. Francik, S.; Ślipek, Z.; Frączek, J.; Knapczyk, A. Present Trends in Research on Application of Artificial Neural Networks in Agricultural Engineering. *Agric. Eng.* **2016**, *20*, 15–25. [CrossRef]
4. Huang, K.H.; Sie, C.Y.; Lin, J.E.; Lee, C.R. LPSD: Low-Rank Plus Sparse Decomposition for Highly Compressed CNN Models. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*; Springer Nature: Singapore, 2024; pp. 353–364.
5. Ghosh, M.; Sing, J.K. Detecting and recognizing faces from video images using multi-deep CNN based rank-level fusion. In Proceedings of the 2024 2nd International Conference on Advancement in Computation & Computer Technologies (InCACCT), Mohali, India, 2–3 May 2024; pp. 283–287.
6. Vaddi, S.R. An Artificial Neural Network Model Supported with Hybrid Multi-Criteria Decision-Making Approaches to Rank Lean Tools for a Foundry Industry. *Trans. FAMENA* **2024**, *48*, 45–68.
7. Saba, H.D.; Sahli, Y. The Role of Artificial Neuron Networks in Intelligent Agriculture (Case Study: Greenhouse). In *Artificial Intelligence for Sustainable Development: Theory, Practice and Future Applications*; Hassanien, A.E., Bhatnagar, R., Darwish, A., Eds.; Studies in Computational Intelligence; Springer International Publishing: Cham, Switzerland, 2021; Volume 912, pp. 45–67. [CrossRef]
8. Kamilaris, A.; Prenafeta-Boldú, F.X. A review of the use of convolutional neural networks in agriculture. *J. Agric. Sci.* **2018**, *156*, 312–322. [CrossRef]
9. Negrete, J.C. Artificial neural networks in mexican agriculture, A overview. *Int. J. Res. Agric. For.* **2018**, *5*, 7.
10. Dahikar, S.S.; Rode, S.V. Agricultural crop yield prediction using artificial neural network approach. *Int. J. Innov. Res. Electr. Electron. Instrum. Control Eng.* **2014**, *2*, 683–686.
11. Huang, Y. Advances in Artificial Neural Networks—Methodological Development and Application. *Algorithms* **2009**, *2*, 973–1007. [CrossRef]
12. Mele, M.M.; Santeramo, F.G. Using an artificial neural networks experiment to assess the links among financial development and growth in agriculture. *Sustainability* **2021**, *13*, 2828. [CrossRef]
13. Monteiro, A.L.; de Freitas Souza, M.; Lins, H.A.; da Silva Teófilo, T.M.; Júnior AP, B.; Silva, D.V.; Mendonça, V. A new alternative to determine weed control in agricultural systems based on artificial neural networks (ANNs). *Field Crops Res.* **2021**, *263*, 108075. [CrossRef]
14. Samborska, I.A.; Alexandrov, V.; Sieczko, L.; Kornatowska, B.; Goltsev, V.; Cetner, M.D.; Kalaji, H.M. Artificial neural networks and their application in biological and agricultural research. *J. NanoPhotoBioSciences* **2014**, *2*, 14–30.
15. Qiu, Y.; Ma, L.; Priyadarshi, R. Deep learning challenges and prospects in wireless sensor network deployment. *Arch. Comput. Methods Eng.* **2024**, 1–24. [CrossRef]

16. Li, K.; Zhu, A.; Zhou, W.; Zhao, P.; Song, J.; Liu, J. Utilizing deep learning to optimize software development processes. *arXiv* **2024**, arXiv:2404.13630.
17. Kaya, M. Feature fusion-based ensemble CNN learning optimization for automated detection of pediatric pneumonia. *Biomed. Signal Process. Control.* **2024**, *87*, 105472. [CrossRef]
18. Shi, H.; Wei, A.; Xu, X.; Zhu, Y.; Hu, H.; Tang, S. A CNN-LSTM based deep learning model with high accuracy and robustness for carbon price forecasting: A case of Shenzhen's carbon market in China. *J. Environ. Manag.* **2024**, *352*, 120131. [CrossRef]
19. Sageengrana, S.; Selvakumar, S.; Srinivasan, S. Optimized RB-RNN: Development of hybrid deep learning for analyzing student's behaviours in online-learning using brain waves and chatbots. *Expert Syst. Appl.* **2024**, *248*, 123267. [CrossRef]
20. Mahto, A.K.; Alam, M.A.; Biswas, R.; Ahmed, J.; Alam, S.I. Short-term forecasting of agriculture commodities in context of indian market for sustainable agriculture by using the artificial neural network. *J. Food Qual.* **2021**, *2021*, 9939906. [CrossRef]
21. Dorokhov, S.; Sibirev, A.V.; Aksenov, A.G. Dynamic Systems Modeling Using Artificial Neural Networks for Agricultural Machines. 2019. Available online: https://inma-ita.ro/inmateh/INMATEH_2_2019/58-07%20Dorokhov.pdf (accessed on 19 May 2024).
22. Parameswari, P.; Rajathi, N.; Kumar, M.V. Artificial Neural Networks in Agriculture: A Survey. *Int. J. Adv. Res. Comput. Commun. Eng.* **2021**, *10*, 7–10.
23. Singh, K.U.; Kumar, A.; Raja, L.; Kumar, V.; Singh kushwaha, A.K.; Vashney, N.; Chhetri, M. An artificial neural network-based pest identification and control in smart agriculture using wireless sensor networks. *J. Food Qual.* **2022**, *2022*, 5801206. [CrossRef]
24. Holzinger, A.; Fister, I.; Kaul, H.-P.; Asseng, S. Human-Centered AI in smart farming: Toward Agriculture 5. *IEEE Access* **2024**, *12*, 62199–62214. [CrossRef]
25. D'emilio, A.; Aiello, R.; Consoli, S.; Vanella, D.; Iovino, M. Artificial neural networks for predicting the water retention curve of sicilian agricultural soils. *Water* **2018**, *10*, 1431. [CrossRef]
26. Koutnik, J.; Greff, K.; Gomez, F.; Schmidhuber, J. A clockwork rnn. In Proceedings of the International Conference on Machine Learning, Beijing, China, 22–24 June 2014; pp. 1863–1871.
27. Li, S.; Li, W.; Cook, C.; Zhu, C.; Gao, Y. Independently recurrent neural network (indrnn): Building a longer and deeper rnn. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018; pp. 5457–5466.
28. Sherstinsky, A. Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *Phys. D Nonlinear Phenom.* **2020**, *404*, 132306. [CrossRef]
29. Sakthipriya, S.; Naresh, R. Precision agriculture based on convolutional neural network in rice production nutrient management using machine learning genetic algorithm. *Eng. Appl. Artif. Intell.* **2024**, *130*, 107682. [CrossRef]
30. Irmak, A.; Jones, J.W.; Batchelor, W.D.; Irmak, S.; Boote, K.J.; Paz, J.O. Artificial neural network model as a data analysis tool in precision farming. *Trans. ASABE* **2006**, *49*, 2027–2037. [CrossRef]