


# A multiple-input neural network model for predicting cotton production quantity: A case study

Ioannis E. Livieris <sup>1,2\*</sup> , Spiros D. Dafnis <sup>3</sup>, George Papadopoulos <sup>3</sup>, Dionissios P. Kalivas <sup>4</sup>

<sup>1</sup> Department of Mathematics, University of Patras, GR 265-00 Patras, Greece;

<sup>2</sup> Core Innovation and Technology O.E, GR 11745, Athens, Greece.

<sup>3</sup> Department of Crop Science, Agricultural University of Athens, GR 118-55, Athens; dafnisspyros@gmail.com, gpapadop@aia.gr

<sup>4</sup> Department of Natural Resources Management & Agricultural Engineering, Agricultural University of Athens, GR 118-55, Athens; kalivas@aia.gr

\* Correspondence: livieris@upatras.gr

Version October 29, 2020 submitted to Algorithms

**Abstract:** Cotton constitutes a significant commercial crop and a widely traded commodity around the world. The accurate prediction of its yield quantity could lead to high economic benefits for farmers as well as for the rural national economy. In this research, we propose a multiple-input neural network model for the prediction of cotton's production. The proposed model utilizes as inputs three different kind of data (soil data, cultivation management data and yield management data) which are treated and handled independently. The significant advantages of the selected architecture is that it is able to efficiently exploit mixed data, which usually require to be processed separately, reduces overfitting, and provides more flexibility and adaptivity for low computational cost compared to a classical fully-connected neural network. An empirical study was performed utilizing data from three consecutively years from cotton farms in Central Greece (Thessaly) in which the prediction performance of the proposed model was evaluated against that of traditional neural network-based and state-of-the-art models. The numerical experiments revealed the superiority of the proposed approach.

**Keywords:** Multiple-input neural network; machine learning; expert knowledge; cotton production.

## 1. Introduction

Cotton is a significant commercial crop and a widely traded commodity around the world, which constitutes a critical link in the chain of agricultural activities. It is also commonly known as “white gold” due to its high influence in rural national economy. Cotton crop is a perennial plant and it is grown primarily for seed and fiber, while it is grown commercially as an annual with biological cycle between 140-210 days [1].

In general, the key objective in precision farming and agriculture is the improvement of crop yield quality and production as well as the reduction in environmental pollution and operating costs [2]. Nevertheless, farm mechanization and the application of new technologies lead to the transformation of crop management from a rather qualitative science, which was mainly based on observations to a more quantitative science, which is now based on measurements. In this new data-driven area, a variety of different production features including soil and climate conditions, irrigation management and nutrient availability significantly influence the potential cotton yield and growth [1–5]. Therefore, the prediction of cotton's yield as well as the factors which mostly affect it, could lead to production optimization through the early modification of harvest settings and adjustments.

The traditional way to forecast cotton production is mainly based on the empirical knowledge of the farmer or mostly by the agricultural expert [6–8]. Cultivation and weather information are processed by agriculturists who attempt to perform accurate predictions about future yield production. Nevertheless, the quantity of cotton production possess, by nature, nonlinear behavior since it is affected by several soil and climate factors and it is characterized by by spatio-temporal variability [9,10]. As a result, the problem of conducting accurate

predictions is considered a considerably hard problem. Moreover, the fact that yield monitors revealed that the cotton's production was different even in different parts of the same field [11] makes this prediction problem even more challenging. Therefore, the use and the development of sophisticated decision support tools is considered essential for potentially assisting agricultural investors and farmers gaining significant profits.

During the last decade, the application of artificial intelligence techniques and methodologies emerged the interest of the scientific community; since they constitute the appropriate tools to deal with the noisy and sometimes chaotic nature of cotton's yield production and lead to more accurate predictions. Along this line, research focused on the development of expert systems for the prediction of agriculture production for assisting growth operations [2]. These intelligent systems exploit the high predictive ability of machine learning algorithms, focusing on increasing crop's efficiency and economic benefits, while simultaneously reducing risks and losses [11–15].

In this work, we propose a new neural network model for predicting cotton production, which is based on a multiple-input architecture and constitutes the main contribution. The proposed model utilizes as input three different kind of data, namely soil data, cultivation management data and yield management data. To the best of our knowledge, this is the first approach which utilizes three different types of features. The selected neural network architecture provides that each kind of input data is processed and handled in a different and independent way. The motivation behind our approach is to develop a learning system which is capable to efficiently exploit information from different kind of data, since this kind of data usually require separate treatment. A series of experiments was conducted for evaluating the prediction performance of the proposed model by comparing it against traditional neural network-based and other state-of-the-art models. For our experiments, we utilized data from three consecutively years from cotton farms in Central Greece (Thessaly). The presented results demonstrate the prediction accuracy of the proposed model, providing empirical evidence that the proposed approach is able to develop an accurate and reliable model.

The remainder of this paper is organized as follows: Section 2 presents a brief survey of rewarding studies, regarding the application of machine learning methodologies for cotton prediction. Section 3 presents a detailed description of the utilized data as well as the data preparation process. Section 4 presents the proposed prediction multiple-input neural network model focusing on its advantages and benefits. Section 5 present state-of-the-art models prediction models. Section 6 presents our numerical experiments. Finally, Section 7 discusses our methodology and the findings of this research, and presents our conclusions.

## 2. Related work

During the last decade, the significant advances in digital technology and machine learning renew the interest of scientific community for the development of efficient expert systems for assisting agriculture precision and production. Chlingaryana et al. [2] conducted an excellent review, presented in detail the recent developments performed within the last two decades, focusing on the application of machine learning methods for accurate crop prediction as well as the estimation of nitrogen status. The main findings of this research was that machine learning techniques provide complete and cost-effective solutions for efficiently estimating crop and environment state as well as significant assistance in decision making. Additionally, the authors attempted to gain significant insights on crop prediction by identifying the factors which affect it, and possible directions to support and improve precision agriculture through artificial intelligence. To the best of our knowledge, the application of machine learning methods for predicting cotton crop has been limited compared to other types of crops. In the sequel, we briefly discuss some rewarding studies regarding cotton yield prediction using machine learning models.

Papageorgiou et al. [12] proposed an intelligent knowledge-based model for modelling the behavior of crop cotton yield in precision farming. The proposed model was based on using a soft computing methodology based on Fuzzy Cognitive Maps (FCMs) and on the unsupervised learning algorithm for FCMs for assessing measurement data and updating initial knowledge. The performance of the proposed model was extensively evaluated for 360 cases in a 5 hectares experimental farm for predicting the cotton yield using a two-level classification ("Low" and "High"). Moreover, their used data were collected from central Greece, during the years 2001, 2003 and 2006. Based on the brief experimental analysis, the authors stated that the proposed FCMs

model is able to assist agriculture managers to better understand cotton yield requirements. Along this line, in [11] the authors extended their previous work, including more elegant conditions to increase the efficiency of their intelligent model. Their numerical experiments reported that the updated model outperformed traditional prediction models such as artificial neural networks (ANNs), decision trees and Naive-Bayes. Finally, the authors highlighted that the proposed updated model consists a convenient decision support tool for cotton production due to its sufficient simplicity and interpretability.

Jamuna et al. [13] studied the problem of classifying the quality of cotton's seed utilizing several growth stages of the crop. Their dataset consists of 900 records and 24 features from a set of different cotton categories. The authors conducted a performance evaluation of state-of-art prediction models, including decision trees, Naive-Bayes and ANNs for identifying the quality of cotton's seed ("Good", "Average" and "Bad"). Their experiments showed that ANNs and decision trees provided almost identical performance, reporting 98.58% classification accuracy. However, the decision trees reported significantly lower training time and computational cost.

Haghverdi et al. [14] attempted to determine cotton lint yield in irrigated field using remote sensing technology. More specifically, they utilized ANNs for extracting information from remotely-sensed crop indices in order to predict and map the cotton lint yield of a field in two successive cropping seasons. The data in their research were obtained after conducting an on-farm irrigation experiment on a property of 73 hectares in west Tennessee during the years 2013 and 2014. Their numerical experiments presented some interesting results, revealing that neural networks can efficiently exploit crop indices phenology for predicting crop yield. Additionally, based on their detailed experiments, the authors stated that the use of remote sensing-based ANN models have a great potential to provide reliable and accurate predictions of cotton yield maps.

Nguyen et al. [15] proposed a spatial-temporal multi-task learning model for predicting within-field crop yield. The proposed model was based on a deep dense neural network enforced with dropout layers and a new weighted regularization technique to improve the prediction performance. It exploits different spatial-temporal features by integrating multiple heterogeneous data from difference sources. The data used in their study were collected from a cotton field in west Texas from 2001 to 2003 and include soil properties, weather data, normalized difference vegetation index data and spectral data. Their proposed model was evaluated against state-of-the-art models such as linear regression, decision tree regression, support vector regression as well as ensemble models, such as random forest and XGBoost. Their experimental analysis provided empirical evidence about the superiority of the proposed model against traditional models; hence, the authors stated that it could effectively assist the field of crop prediction.

In this research, we propose a different approach for the development of an accurate model for the prediction of cotton's yield production. More specifically, we propose a new multiple-input neural network model, which exploits mixed features as inputs from three different kinds of data: soil data, cultivation management data and yield management data. The contribution of our approach is that the proposed neural network architecture processes and handles each type of inputs in an independent way, which benefits more the prediction performance, since mixed features of data usually require separate treatment. To the best of our knowledge, none of the mentioned approaches considered exploiting information from three different kinds of cotton data and developing a prediction model by handling them, separately. An advantage provided by the proposed architecture is the considerable flexibility and adaptivity for low computational effort, compared to that of a fully connected neural network with two or more hidden layers.

### 3. Data

In our research, the data concern the cotton yield in kilograms per 0.1 hectare from 350 sampling sites of the Thessaly plain, during the years 2008-2010. For each sampling site, a number of features were obtained from three main categories: *soil* features, *cultivation management* features and *yield management* features. The data were divided into training set (273 instances) which consists the yield production during 2008 and 2009 which ensures a substantial amount of data for training and testing set (119 instances) containing yield production during 2010, which ensures that will be perform with a considerable amount of unseen data. It is also worth mentioning that the maximum cotton production in Greece indicated no significant year-to-year fluctuations

| Feature              | Description                      | Type                | Values  |
|----------------------|----------------------------------|---------------------|---|
| Order                | Pedogenic soil order             | Nominal             | { Alfisol, Inceptisol, Vertisol }                                     |
| Drainage             | Soil drainage                    | Ordinal             | { Very poorly, Poorly, Somewhat poorly, Moderately, Well, Very well } |
| CaCO <sub>3</sub>    | Calcium carbonates               | Ordinal             | { None, Slight, Some, Strong }  |
| Texture <sub>1</sub> | Soil texture in depth: 0-25 cm   | Numerical<br>(Real) | [5, 60]   |
| Texture <sub>2</sub> | Soil texture in depth: 25-75 cm  | Numerical<br>(Real) | [9, 60]   |
| Texture <sub>3</sub> | Soil texture in depth: 75-150 cm | Numerical<br>(Real) | [9, 18.5]   |
| Slope                | Percentage of slope              | Ordinal             | { Little, Moderate, High }  |
| Erosion              | Soil erosion                     | Ordinal             | { No erosion, Slightly, Moderately, Very, Severely }                  |

**Table 1.** List of soil features

| Feature            | Description  | Type                      | Values                        |
|--------------------|--|---------------------------|-------------------------------|
| Re-sowing          | Re-sowing was or not performed                                   | Nominal<br>(Binary)       | { True, False }               |
| Variety            | Variety of seed  | Nominal                   | { Acala (Zeta-0/Zeta-5), 4S } |
| Seed               | Kilograms of seed per 0.1 hectare                                | Numerical<br>(Real)       | [2, 7]                        |
| Germination        | Percentage of germination  | Numerical<br>(Percentage) | [0, 100]                      |
| spacingOut         | Spacing out  | Categorical<br>(Binary)   | { True, False }               |
| XN                 | Kilograms of nitrogen $N - NO_3$ per 0.1 hectare in oxide form   | Numerical<br>(Real)       | [0, 24]                       |
| XP                 | Kilograms of phosphorus $P - PO_4$ per 0.1 hectare in oxide form | Numerical<br>(Real)       | [0, 24]                       |
| Weeding            | Weed control before sowing                                       | Nominal<br>(Binary)       | { True, False }               |
| SPRAY <sub>1</sub> | Weed control after emergence                                     | Nominal<br>(Binary)       | { True, False }               |
| SPRAY <sub>2</sub> | Control for <i>Helicoverpa armigera</i>                          | Numerical<br>(Integer)    | [0, 6]                        |
| SPRAY <sub>3</sub> | Insect control for aphids sp.                                    | Numerical<br>(Integer)    | [1, 7]                        |
| Irrger             | Irrigation before germination                                    | Numerical<br>(Real)       | [0, 4]                        |
| Irrigat            | Irrigation after emergence                                       | Numerical<br>(Real)       | [2, 14]                       |

**Table 2.** List of cultivation management features

| Feature     | Description   | Type                      | Values   |
|-------------|---|---------------------------|--|
| Defoliation | If defoliation was performed.                         | Nominal<br>(Binary)       | { True, False }  |
| prodLoss    | Loss of production due to extreme climatic conditions | Numerical<br>(Percentage) | [0, 100]   |
| prevUse     | Previous use of the farm                              | Nominal                   | { One year cereals, Two years cereals, One year cotton, Two years cotton, Three or four years cotton } |
| typeHarv    | Type of harvesting                                    | Nominal                   | { By hand, Mechanized }.   |
| Days        | Number of days between sowing and harvesting          | Numerical<br>(Integer)    | [141, 257]   |

**Table 3.** List of yield management features

and has remained rather stable during the last twenty years. This implies that the current productions are not considerably different from the potential. Finally, the used data contained no missing values, while the outlier prices were not removed for not destroying the dynamics of data.

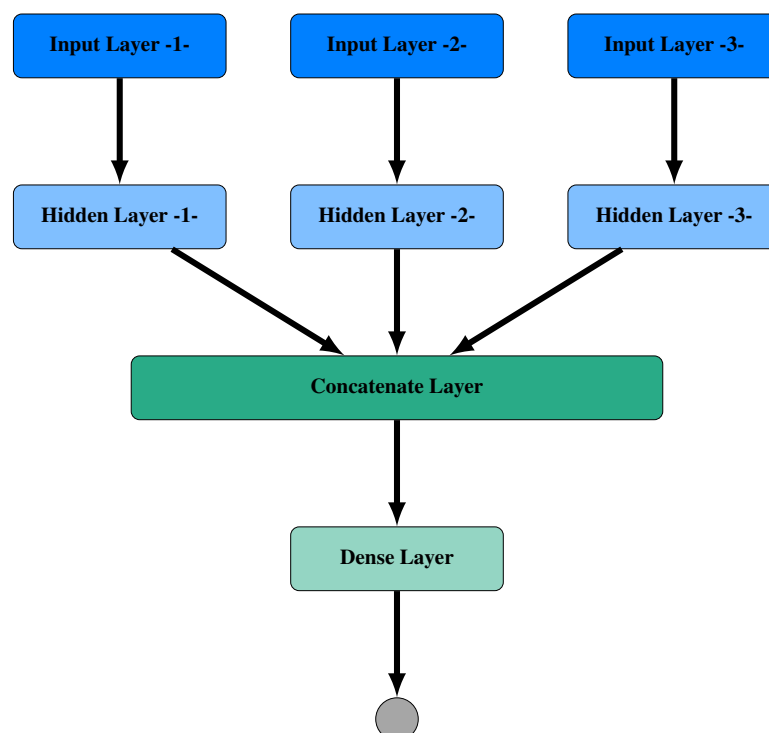
Tables 1, 2 and 3 present the set of features concerning soil, cultivation management and yield management features, respectively, as well as a brief description for each feature.

Finally, it is worth noticing that for maximizing the performance of all prediction models, we applied a variety of feature selection techniques such as univariate feature selection, recursive feature elimination and selection based on feature importance [16–18] as well as attempted to reduce the number of features by analyzing the correlation [19] between them. Nevertheless, any attempt of selecting a subset of the presented features resulted in slightly decreasing the overall performance of all prediction models; thus, all features presented in Tables 1, 2 and 3 were utilized, even the least significant.

#### 4. Proposed multiple-input neural network prediction model

The main contribution of this research is the development of a prediction model for efficiently predicting cotton yield production, which is based on a multiple-input single-output structure. The motivation behind our approach is to develop a learning system which is capable to efficiently exploit useful information from mixed features of data, since this kind of data usually requires separate treatment. To this end, each input kind of data is processed and handled in a different and independent way.

The architecture of the proposed Multiple-input Neural Network (MNN) model is depicted in Figure 1. It consists of three separate input layers (Input Layer -1-, Input Layer -2- and Input Layer -3-), each one has as inputs the soil (Table 1), the cultivation management (Table 2) and the yield management features (Table 3), respectively. Each input layer is followed by a hidden layer (Hidden Layer -1-, Hidden Layer -2-, Hidden Layer -3-), which independently process the input data. It is worth noticing that each hidden layer could be constituted by a classical dense layer or by a more sophisticated recurrent layer such as Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU). Next, the outputs of the three hidden layers are imported and merged by a concatenate layer. This layer is followed by a dense layer and a final output layer of one neuron.



**Figure 1.** Proposed architecture of Multiple-input Neural Network (MNN)

An advantage of the proposed architecture is that, although stacking many hidden layers allows a traditional neural network model to analyze and encode a very complex function from the input to the output, it is usually difficult to train such a model due to the vanishing gradient problem, which implies that the convergence of the training process may be degraded. Additionally, each category of mixed features of data is handled independently and subsequently the processed data are merged and further processed. As a result, the proposed architecture offers more flexibility and adaptivity for low development cost compared to that of a fully connected neural network with two or more hidden layers, which implies that computational effort of the training process is reduced.

In the sequel, we present a brief description of the Dense, LSTM and GRU layers which constitute the main elements of the proposed MNN model.

- **Dense layer:** Dense layers [20] constitute the traditional and the most popular choice for composing a hidden layer in a multilayer neural network. A dense layer is composed of neurons which are connected with all neurons in the previous layer. The operations performed by each neuron can be summarized by

$$o_i = f(W_i x + b_i),$$

where  $o_i$  is the output of the  $i$ -neuron of the layer,  $W_i$  is the vector of weights,  $x$  is the input vector,  $b_i$  is the bias vector and  $f$  is the activation function.

- **LSTM layer:** Long Short-Term Memory (LSTM) layers [21] constitute a special type of Recurrent Neural Networks layers which are characterized by their ability to learn long-term dependencies.

Each LSTM unit in the layer is composed of a memory cell and three gates: input, output and forget. At at time  $t$ , the input gate  $i_t$  and a second gate  $c_t^*$  modulates the amount of information which are stored into the memory state  $c_t$ . The forget gate  $f_t$  modulates the past information which must be vanished or must be kept on the memory cell at the previous time  $t - 1$ . Finally, the hidden state  $h_t$  constitutes the output of the memory cell and it calculated using memory state  $c_t$  and the output gate  $o_t$  which modulates the information used for the output of the memory cell. Summarizing, the following equations describe the operations performed by an LSTM unit.

$$\begin{aligned} i_t &= \sigma(U_i x_t + W_i h_{t-1} + b_i), \\ f_t &= \sigma(U_g x_t + W_g h_{t-1} + b_g), \\ c_t^* &= \tanh(U_c x_t + W_c h_{t-1} + b_c), \\ c_t &= g_t \odot c_{t-1} + i_t \odot c_t^*, \\ o_t &= \sigma(U_o x_t + W_o h_{t-1} + b_o), \\ h_t &= o_t \odot \tanh(c_t), \end{aligned}$$

where  $x_t$  denotes the input of each unit,  $W_*$  and  $U_*$  are matrices of weights and  $b_*$  are the bias vectors with  $i \in \{i, g, c\}$ , the operator  $\odot$  denotes the Hadamard product (component-wise multiplication),  $\sigma$  is the sigmoid function and  $h_t$  is the output of the memory cell which denotes the hidden state.

- **GRU layer:** Gated Recurrent Units (GRU) were originally proposed by Cho et al. [22] and were inspired by the LSTM units, but with simpler implementation and calculations. Its main difference is that a GRU unit has only two gates (update and reset) which modulate the flow of information, without the utilization of memory gates, since it exposes the full hidden content without any control.

The update gate  $z_t$  controls the level the unit updates its content, by taking into consideration a linear sum between the previous state  $h_{t-1}$  and the input  $x_t$ . The reset gate  $r_t$  is computed in a similar manner with the update gate  $z_t$ . Finally, the activation  $h_t$  of a GRU unit constitutes the linear combination between the

previous  $h_{t-1}$  and the candidate activation  $\hat{h}_z$ , which is computed similarly to the traditional recurrent unit. The operations performed by an GRU unit are briefly described by

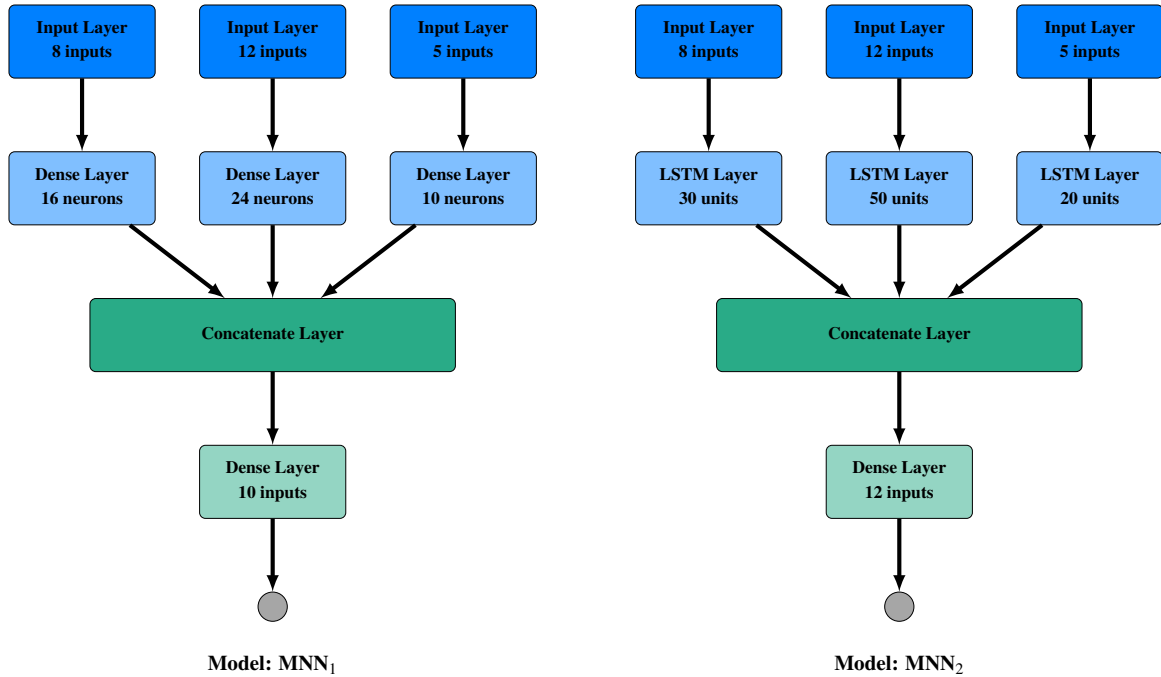
$$\begin{aligned} z_t &= \sigma(W_z x_t + U_z h_{t-1} + b_z), \\ r_t &= \sigma(W_r x_t + U_r h_{t-1} + b_r), \\ \hat{h}_t &= \tanh(W_h x_t + U_h (r_t \odot h_{t-1} + b_h)), \\ h_t &= (1 - z_t) \odot h_{t-1} + z_t \odot \hat{h}_t, \end{aligned}$$

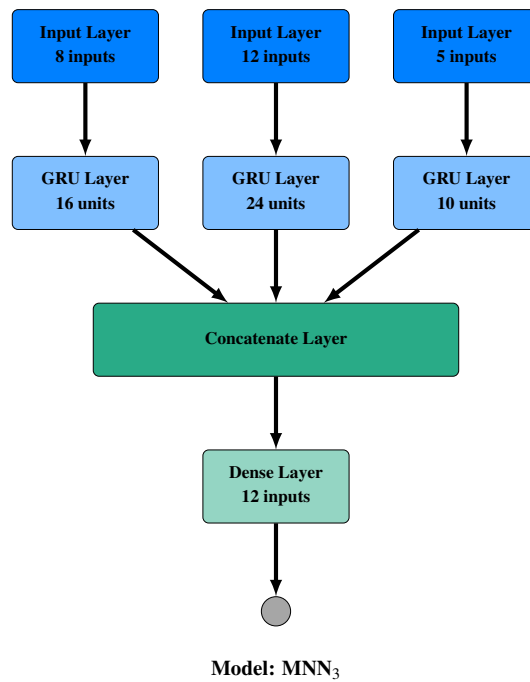
where  $x_t$  denotes the input vector,  $W_*$  and  $U_*$  are matrices of weights with  $i \in \{z, r, h\}$  and  $b_*$  are the bias vectors.

In our numerical experiments, the performance of the proposed MNN model was evaluated using three different architectures (Figure 2), namely:

- MNN<sub>1</sub> utilizes 3 dense layers of 16, 24 and 10 neurons in Hidden Layer -1-, Hidden Layer -2- and Hidden Layer -3-, respectively and a dense layer of 10 neurons after the concatenate layer.
- MNN<sub>2</sub> utilizes 3 LSTM layers of 30, 50 and 20 units in Hidden Layer -1-, Hidden Layer -2- and Hidden Layer -3-, respectively and a dense layer of 12 neurons after the concatenate layer.
- MNN<sub>3</sub> utilizes 3 GRU layers of 16, 24 and 10 units in Hidden Layer -1-, Hidden Layer -2- and Hidden Layer -3-, respectively and a dense layer of 12 neurons after the concatenate layer.

All neurons in the hidden layers used Rectifier Linear Unit (ReLU) activation function while the neuron in the output layer used sigmoid activation. The kernel and bias initializer in all layers were set as default as well as the recurrent initializer in the recurrent layers.





**Figure 2.** Architecture of the proposed MNN<sub>1</sub>, MNN<sub>2</sub> and MNN<sub>3</sub> models.

## 5. State of the art machine learning models

In this section, we briefly present the state-of-the-art machine learning models which have been established in the literature to address prediction benchmarks. These models will be utilized to base models in order to explore and highlight the performance of the proposed model.

More specifically, the models are: Decision Tree Regressor (DTR) [23], Gaussian Processes (GP) [24],  $k$ -Nearest Neighbor Regression ( $k$ NN) [25], Least Absolute Shrinkage and Selection Operator (LASSO) [26], Linear Regression (LR) [27], and Support Vector Regression (SVR) [28], which are described below:

- DTR is a decision tree dedicated for regression problems, which constructs a model tree based on splitting criterions. More analytically, this algorithm develops a tree with decision nodes and leaf nodes, in which the leafs predict the output continuous value utilizing the linear regression algorithm.
- GP constitutes a collection of random variables depending on time or space, in which every collection of those variables has a multivariate normal distribution. The prediction value of this machine learning algorithm is a one-dimensional Gaussian distribution and it is calculated by the similarity between the training instances and the unseen instances.
- $k$ NN is a popular machine learning algorithm, which utilizes various distance mathematic formulas to compute feature similarity between each new instance and a predefined number  $k$  of instances in the training data. For regression problems, the output value is defined by the average value of its  $k$  nearest neighbors.
- LASSO is a linear model trained with  $L_1$  prior as regularizer. This algorithm performs both regularization and variable selection in order to enhance the prediction accuracy. Due to its simplicity and efficiency, it has been successfully extended and applied to a wide variety of statistical models.
- LR probably constitutes the most commonly used algorithm for developing an efficient regression model. This prediction algorithm aims to determine the relationship between one or more explanatory (independent) variables and the dependent variable based on the linear mathematical model.

- SVR is a classical machine learning algorithm which is utilized for predicting continues values. Its main objective is to fit the error within a specified threshold, in contrast to traditional regression algorithms like LR, which focus to minimize the error.

Additionally, the performance of the proposed MNN model was also compare against that of three widely utilized neural network-based models i.e. a fully connected Feed-Forward Neural Network (FFNN), a GRU-based network (GRU) and a Long Short-Term Memory (LSTM) network. Notice in our numerical experiments the hyper-parameters of all regression models were optimized under exhaustive experimentation and are briefly presented in Table 4.

| Model | Description   |
|-------|---|
| DTR   | Splitting criterion: MSE,<br>Max depth = 4,<br>Min. number of samples = 10.   |
| GP    | Level of Gaussian noise = 1,<br>Kernel type = RBF.  |
| FFNN  | Architecture: 2 hidden layers with 50 and 10 neurons and an output layer of 1 neuron,<br>Activation functions = ReLu,<br>Optimizer = Adam.                    |
| GRU   | Architecture: 1 GRU layer with 40 units, 1 dense layer with 10 neurons and an output layer of 1 neuron,<br>Activation functions = ReLu,<br>Optimizer = Adam.  |
| LSTM  | Architecture: 1 LSTM layer with 70 units, 1 dense layer with 20 neurons and an output layer of 1 neuron,<br>Activation functions = ReLu,<br>Optimizer = Adam. |
| kNN   | Number of neighbors = 3,<br>Euclidean distance.   |
| LASSO | $\alpha = 1.0$ ,<br>Tolerance = $10^{-4}$ .   |
| LR    | No parameters specified.  |
| SVR   | Kernel = RBF,<br>$C = 1.0$ ,<br>Tolerance = $10^{-3}$ ,<br>$\epsilon = 0.1$ ,<br>$\gamma = \text{'scale'}$ .  |

**Table 4.** Hyper-parameter specification of all prediction models

## 6. Numerical Experiments

In this section, we evaluate the performance of the proposed MNN model and compare it against that of three widely utilized neural network-based models i.e. Feed-Forward Neural Network (FFNN), a GRU-based network (GRU) and a Long Short-Term Memory (LSTM) network. Additionally, it was also compare against the state-of-the art models: Decision Tree Regressor (DTR), Gaussian Processes (GP),  $k$ -Nearest Neighbor Regression ( $k$ NN), Least Absolute Shrinkage and Selection Operator (LASSO), Linear Regression (LR) and Support Vector Regression (SVR).

In order to avoid overfitting and maximize the efficiency of the proposed models as well as the neural network-based models, 15% of training data were used for validation and early stopping technique based on 'validation loss' was used. Furthermore, any attempt to use regularizers or dropout decreased the overall performance of all these models.

The performance of all prediction models was measured utilizing the metrics: Mean Absolute Error (MAE), Root-Mean Square Error (RMSE), Mean Absolute Percentage Error (MAPE) and symmetric Mean Absolute Percentage Error (sMAPE), which are respectively defined by

$$\text{MAE} = \frac{1}{N} \sum_{t=1}^N |y_t - \hat{y}_t|, \quad \text{RMSE} = \sqrt{\frac{1}{N} \sum_{t=1}^N (y_t - \hat{y}_t)^2}, \quad \text{MAPE} = \frac{1}{N} \sum_{t=1}^N \frac{|y_t - \hat{y}_t|}{|y_t|}, \quad \text{sMAPE} = \frac{100\%}{N} \sum_{t=1}^N \frac{2(|y_t - \hat{y}_t|)}{|y_t| + |\hat{y}_t|}$$

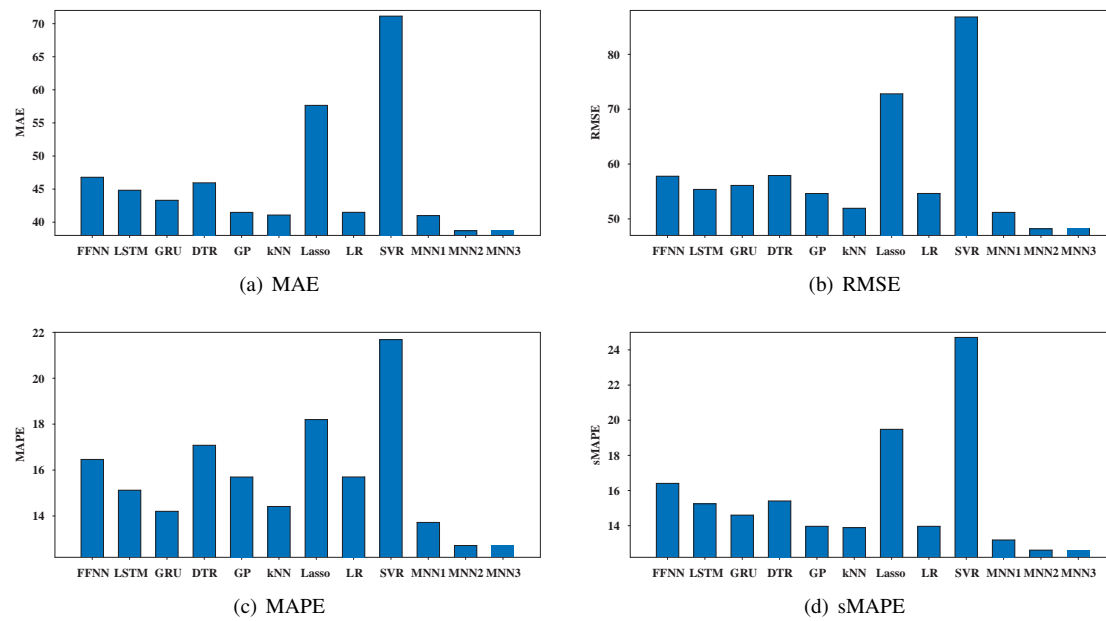
where  $N$  is the number of forecasts,  $y_t$  is the actual value and  $\hat{y}_t$  is the predicted value. In addition, the implementation code was written in Python 3.7 on a laptop (Intel(R) Core(TM) i7-6700HQ CPU 2.6GHz, 16GB RAM) using libraries Tensorflow [29] and Scikit-learn [30].

Table 5 presents the performance of all compared models, regarding the performance metrics RMSE, MAE, MAPE and sMAPE. Additionally, a more representative visualization of all performance metrics is presented in Figure 3. Clearly, the proposed models MNN<sub>1</sub>, MNN<sub>2</sub> and MNN<sub>3</sub> exhibited the best performance, considerably outperforming all neural network-based and state-of-the-art prediction models. More specifically, all versions of the proposed MNN model presented the lowest MAE and RMSE scores. MNN<sub>3</sub> reported the best performance, closely followed by MNN<sub>2</sub>, relative to MAE and RMSE scores. Regarding MAPE and sMAPE performance metrics, MNN<sub>3</sub> reported the lowest (best) scores, followed by MNN<sub>2</sub> which exhibited slightly worst performance. MNN<sub>2</sub> reported 11.8%-41.36% and 8.24%-48.98% lower MAPE and sMAPE scores, respectively, compared to the performance of the state-of-the-art models. Additionally, MNN<sub>2</sub> exhibited 10.5%-22.6% and 13.6%-23.4% lower MAPE and sMAPE scores, respectively compared to the performance of the neural network-based models. LSTM, GP,  $k$ NN and LR presented competitive performance to MNN<sub>1</sub>, however, they were considerably outperformed by the versions of the proposed model which utilized recurrent hidden layers, i.e. MNN<sub>2</sub> and MNN<sub>3</sub>. This implies that the utilization of hidden layers with recurrent units favored the performance of the proposed MNN model. Additionally, LASSO and SVR reported the worst performance, regarding all metrics.

| Model            | Type                            | MAE           | RMSE          | MAPE          | sMAPE         |
|------------------|---------------------------------|---------------|---------------|---------------|---------------|
| FFNN             | Neural network-based regressors | 46.784        | 57.776        | 16.464        | 16.030        |
| LSTM             |                                 | 43.303        | 55.378        | 14.203        | 14.598        |
| GRU              |                                 | 44.826        | 56.103        | 15.124        | 15.243        |
| DTR              | State of the art regressors     | 45.942        | 57.920        | 17.081        | 15.407        |
| GP               |                                 | 41.475        | 54.634        | 15.697        | 13.964        |
| $k$ NN           |                                 | 45.441        | 54.963        | 16.030        | 15.236        |
| LASSO            |                                 | 57.647        | 72.792        | 18.199        | 19.480        |
| LR               |                                 | 41.480        | 54.641        | 15.699        | 13.965        |
| SVR              |                                 | 65.517        | 79.045        | 20.024        | 22.881        |
| MNN <sub>1</sub> | Proposed model                  | 40.479        | 52.741        | 14.272        | 13.632        |
| MNN <sub>2</sub> |                                 | 39.612        | 49.439        | 13.164        | 12.840        |
| MNN <sub>3</sub> |                                 | <b>38.707</b> | <b>48.213</b> | <b>12.713</b> | <b>12.608</b> |

**Table 5.** Performance of the proposed MNN model and the state-of-the-art prediction models

Summarizing, the interpretation of Table 5 reveals that the three different architectures of the proposed MNN model exhibited the best overall performance, regarding all metrics. Furthermore, the utilization of the recurrent layers (LSTM and GRU), instead of the traditional dense layers, benefited its performance considerably. It is also worth mentioning that the proposed model exhibited slightly better performance utilizing GRU layers, instead of LSTM layers. This may be due to the fact that each LSTM unit has more gates for the gradients to flow through, causing steady progress to be more difficult to maintain [22,31]. Nevertheless, this is surprising since GRU frequently suffer from the vanishing gradient problem [32]. By taking these into consideration, we are able to conclude that the vanishing gradient problem rarely or did not occurred in our experiments. A possible explanation for this could be the utilization of ReLU activation function, the less complex architecture of the proposed model compared to that of a fully connected network, as well as the complexity and size of the used dataset. This is worth certainly investigated in the near future.



**Figure 3.** Box-plot for the performance of the proposed MNN model and the state-of-the-art prediction models based on (a) MAE (b) RMSE (c) MAPE (d) sMAPE

## 7. Discussion & conclusions

In this work, we proposed a multiple-input neural network model, called MNN, for the prediction of cotton's yield. The proposed model uses as inputs three different kind of data (soil, cultivation management and yield management) which are treated and handled independently. A significant advantage of the selected architecture is that it is able to efficiently exploit information in mixed data, since this kind of data usually require to be processed separately. Additionally, the proposed architecture is superior to the traditionally fully connected neural network architecture in terms of flexibility and adaptivity for low computational cost.

An empirical study was performed utilizing data from three consecutive years from cotton farms in Central Greece, in which the prediction performance of the proposed model was evaluated against that of traditional neural network-based and other state-of-the-art models. Our numerical experiments revealed the superiority of the proposed approach, providing empirical evidence that agricultural datasets, which consists of different types of data, should be treated utilizing the proposed approach.

Additionally, the performance of the proposed model was evaluated utilizing three different types of hidden layers, i.e. dense, LSTM and GRU. It is worth noticing that the utilization of the recurrent layers, benefitted the performance of MNN considerably, compared with the utilization of traditionally dense layers. Furthermore, the proposed MNN model exhibited slightly better performance utilizing GRU layers instead of LSTM layers. A possible explanation for this is that each LSTM unit has more gates for the gradients to flow through, causing steady progress to be more difficult to maintain [22,31]. In addition, by comparing the performance of MNN<sub>2</sub> and MNN<sub>3</sub>, we can conclude that the vanishing gradient problem was rarely or not occurred in our experiments. This is probably due to the utilization of the ReLU activation function and to the “sparse” architecture of the proposed multi-input neural network model. More analytically, the utilization of ReLU activation function is able to frequently prevent the vanishing gradient problem from occurring, since it only saturates in one direction; while the “sparse” architecture of the proposed model makes it considerably less complicated compared to a fully connected neural network. Another possible reason could be the complexity of the utilized dataset as well its relative small number of training instances. To this end, more experiments utilizing more cotton and other crop datasets are need it, which is definitely included in our future research.

Furthermore, it is worth mentioning that the features used in this research do not constitute a conclusive list. An extension could introduce new features and other criteria, which may potentially influence the prediction performance. Clearly, it is still under consideration which feature have greater impact for predicting cotton

production or which of them should be used by a intelligent model. These questions constitute an interesting aspect for future research. Nevertheless, it is likely that the research to answer these questions could reveal additional information about the cotton yield behavior.

A limitation of this work is that the utilized dataset contained only 349 samples. Based on the preliminary experimental results, it seems that the proposed approach is able to develop a reliable and accurate model. However, we intent to enlarge our database with data from more sample sites and more years in order to perform a exhaustive performance evaluation of the compared models on various dataset as well as a comprehensive statistical analysis (use of nonparametric test and/or a post-hoc tests).

Since the presented numerical experiments are quite encouraging, a interesting next step could be to evaluate the proposed model for the prediction of yield of other crop species such as wheat, trees, maize and vineyards. In our future research, we intend to incorporate ensemble learning strategies (see [33–37] and the references therein) and also incorporate sophisticated preprocessing methodologies [11,12] in our framework for improving the prediction performance. Finally, it is worth noticing that our main expectation is that the proposed approach could be utilized as a reference for decision-making in agricultural production.

**Funding:** This research received no external funding.

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

1. Kalivas, D.; Kollias, V. Effects of soil, climate and cultivation techniques on cotton yield in Central Greece, using different statistical methods. *Agronomie* **2001**, *21*, 73–90.
2. Chlingaryan, A.; Sukkarieh, S.; Whelan, B. Machine learning approaches for crop yield prediction and nitrogen status estimation in precision agriculture: A review. *Computers and electronics in agriculture* **2018**, *151*, 61–69.
3. Hayat, A.; Amin, M.; Afzal, S. Statistical investigation to explore the impact of soil and other characteristics on cotton yield. *Communications in Soil Science and Plant Analysis* **2020**, pp. 1–9.
4. Sawan, Z.M. Climatic variables: Evaporation, sunshine, relative humidity, soil and air temperature and its adverse effects on cotton production. *Information processing in agriculture* **2018**, *5*, 134–148.
5. Dai, J.; Li, W.; Zhang, D.; Tang, W.; Li, Z.; Lu, H.; Kong, X.; Luo, Z.; Xu, S.; Xin, C. Competitive yield and economic benefits of cotton achieved through a combination of extensive pruning and a reduced nitrogen rate at high plant density. *Field Crops Research* **2017**, *209*, 65–72.
6. Ahmad, S.; Hasanuzzaman, M. *Cotton Production and Uses: Agronomy, Crop Protection, and Postharvest Technologies*; Springer Nature, 2020.
7. Jabran, K.; Chauhan, B.S. *Cotton production*; John Wiley & Sons, 2019.
8. Khan, M.A.; Wahid, A.; Ahmad, M.; Tahir, M.T.; Ahmed, M.; Ahmad, S.; Hasanuzzaman, M. World Cotton Production and Consumption: An Overview. In *Cotton Production and Uses*; Springer, 2020; pp. 1–7.
9. Ullah, K.; Khan, N.; Usman, Z.; Ullah, R.; Saleem, F.Y.; Shah, S.A.I.; Salman, M. Impact of temperature on yield and related traits in cotton genotypes. *Journal of integrative agriculture* **2016**, *15*, 678–683.
10. Zonta, J.H.; Brandão, Z.N.; Sofiatti, V.; Bezerra, J.R.C.; Medeiros, J.d.C. Irrigation and nitrogen effects on seed cotton yield, water productivity and yield response factor in semi-arid environment. *Embrapa Algodão-Artigo em periódico indexado (ALICE)* **2016**.
11. Papageorgiou, E.I.; Markinos, A.T.; Gemtos, T.A. Fuzzy cognitive map based approach for predicting yield in cotton crop production as a basis for decision support system in precision agriculture application. *Applied Soft Computing* **2011**, *11*, 3643–3657.
12. Papageorgiou, E.I.; Markinos, A.; Gemptos, T. Application of fuzzy cognitive maps for cotton yield management in precision farming. *Expert systems with Applications* **2009**, *36*, 12399–12413.
13. Jamuna, K.; Karpagavalli, S.; Vijaya, M.; Revathi, P.; Gokilavani, S.; Madhiya, E. Classification of seed cotton yield based on the growth stages of cotton crop using machine learning techniques. 2010 International Conference on Advances in Computer Engineering. IEEE, 2010, pp. 312–315.
14. Haghverdi, A.; Washington-Allen, R.A.; Leib, B.G. Prediction of cotton lint yield from phenology of crop indices using artificial neural networks. *Computers and Electronics in Agriculture* **2018**, *152*, 186–197.

15. Nguyen, L.H.; Zhu, J.; Lin, Z.; Du, H.; Yang, Z.; Guo, W.; Jin, F. Spatial-temporal Multi-Task Learning for Within-field Cotton Yield Prediction. *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2019, pp. 343–354.
16. Brownlee, J. *Machine learning mastery with Python: understand your data, create accurate models, and work projects end-to-end*; Machine Learning Mastery, 2016.
17. Chandrashekar, G.; Sahin, F. A survey on feature selection methods. *Computers & Electrical Engineering* **2014**, *40*, 16–28.
18. Khalid, S.; Khalil, T.; Nasreen, S. A survey of feature selection and feature extraction techniques in machine learning. 2014 Science and Information Conference. IEEE, 2014, pp. 372–378.
19. Benesty, J.; Chen, J.; Huang, Y.; Cohen, I. Pearson correlation coefficient. In *Noise reduction in speech processing*; Springer, 2009; pp. 1–4.
20. Demuth, H.B.; Beale, M.H.; De Jess, O.; Hagan, M.T. *Neural network design*; Martin Hagan, 2014.
21. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural computation* **1997**, *9*, 1735–1780.
22. Cho, K.; Van Merriënboer, B.; Bahdanau, D.; Bengio, Y. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259* **2014**.
23. Loh, W.Y. Classification and regression tree methods. *Wiley StatsRef: Statistics Reference Online* **2014**.
24. Paul, W.; Baschnagel, J. *Stochastic processes*; Vol. 1, Springer, 2013.
25. Aha, D. *Lazy learning*; Springer Science & Business Media, 2013.
26. Hastie, T.; Tibshirani, R.; Wainwright, M. *Statistical learning with sparsity: the Lasso and generalizations*; CRC press, 2015.
27. Seber, G.A.; Lee, A.J. *Linear regression analysis*; Vol. 329, John Wiley & Sons, 2012.
28. Deng, N.; Tian, Y.; Zhang, C. *Support vector machines: optimization based theory, algorithms, and extensions*; Chapman and Hall/CRC, 2012.
29. Abadi, M.; Barham, P.; Chen, J.; Chen, Z.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Irving, G.; Isard, M. Tensorflow: A system for large-scale machine learning. 12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16), 2016, pp. 265–283.
30. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V. Scikit-learn: Machine learning in Python. *the Journal of machine Learning research* **2011**, *12*, 2825–2830.
31. Dey, R.; Salemt, F.M. Gate-variants of gated recurrent unit (GRU) neural networks. 2017 IEEE 60th international midwest symposium on circuits and systems (MWSCAS). IEEE, 2017, pp. 1597–1600.
32. Hu, Y.; Huber, A.; Anumula, J.; Liu, S.C. Overcoming the vanishing gradient problem in plain recurrent networks. *arXiv preprint arXiv:1801.06105* **2018**.
33. Polikar, R. Ensemble learning. In *Ensemble machine learning*; Springer, 2012; pp. 1–34.
34. Dietterich, T.G. Ensemble learning. *The handbook of brain theory and neural networks* **2002**, *2*, 110–125.
35. Zhang, C.; Ma, Y. *Ensemble machine learning: methods and applications*; Springer, 2012.
36. Brown, G. Ensemble Learning. *Encyclopedia of Machine Learning* **2010**, 312.
37. Pintelas, P.; Livieris, I.E. Special Issue on Ensemble Learning and Applications, 2020.