Hindawi Publishing Corporation
ISRN Artificial Intelligence
Volume 2013, Article ID 430986, 9 pages
http://dx.doi.org/10.1155/2013/430986

Research Article

Yield Prediction for Tomato Greenhouse Using EFuNN

Kefaya Qaddoum, E. L. Hines, and D. D. Iliescu

School of Engineering, University of Warwick, Coventry CV4 7AL, UK

Correspondence should be addressed to E. L. Hines; e.l.hines@warwick.ac.uk

Received 9 January 2013; Accepted 28 January 2013

Academic Editors: C. Kotropoulos, H. Ling, and L. S. Wang

In the area of greenhouse operation, yield prediction still relies heavily on human expertise. This paper proposes an automatic tomato yield predictor to assist the human operators in anticipating more effectively weekly fluctuations and avoid problems of both overdemand and overproduction if the yield cannot be predicted accurately. The parameters used by the predictor consist of environmental variables inside the greenhouse, namely, temperature, CO_2, vapour pressure deficit (VPD), and radiation, as well as past yield. Greenhouse environment data and crop records from a large scale commercial operation, Wight Salads Group (WSG) in the Isle of Wight, United Kingdom, collected during the period 2004 to 2008, were used to model tomato yield using an Intelligent System called "Evolving Fuzzy Neural Network" (EFuNN). Our results show that the EFuNN model predicted weekly fluctuations of the yield with an average accuracy of 90%. The contribution suggests that the multiple EFUNNs can be mapped to respective task-oriented rule-sets giving rise to adaptive knowledge bases that could assist growers in the control of tomato supplies and more generally could inform the decision making concerning overall crop management practices.

1. Introduction

Greenhouse production systems require implementing computer-based climate control systems, including carbon dioxide (CO_2) supplementation. The sort of systems we are concerned with here are normally in use all year-round so as to maximize product and thus are typically applied in scenarios where the greenhouse crops have a long growing cycle. The technological advances and the sophistication of greenhouse crop production control systems do not mean that greenhouse operation does not rely on human expertise to decide on the optimum values for yield weekly amount. Practiced greenhouse tomato growers and researchers evaluate plant responses and growth mode by observations of the plant morphology. Tomato growers use this information in decision making depending on climate conditions and crop management practices to shift the plant growth toward a "balanced" growth mode, or to be able to accurately predict regular crops amounts each year.

One of the dynamic and complex systems is tomato crop growth, and few models have studied it previously. Two of the dynamic growth models are TOMGRO [1, 2] and TOMSIM [3, 4]. Both models depend on physiological processes, and they model biomass dividing, crop growth, and yield as a function of several climate and physiological parameters. Their use is limited, especially for practical application by growers, by their complexity, and by the difficulty in obtaining the initial condition parameters required for implementation [3]. Moreover, critical measurements are required for calibration and validation for each application.

Tompousse [5] predicts yields in terms of the weight of harvested fruits. Their model was developed in France for heated greenhouses and required that the linear relationships of both flowering rate and fruit growth period were in an appropriately warm environment; when the system was implemented in unheated plastic greenhouses, in Portugal, for example, the model performed poorly and was only tested for short production cycles of less than 15 weeks.

Adams [6] proposed a greenhouse tomato model and implemented it in the form of a graphical simulation tool (HIPPO). A key objective of the model was to explain the weekly fluctuations of greenhouse tomato yields as characterized by fruit size and harvest rates. This model required hourly climate data in order to determine the rates of growth of leaf truss and flower production.

Although the seasonal fluctuations of yield in greenhouse crops is generally understood to be influenced by the periodic variation of solar radiation and air temperature, greenhouse growers are also interested in the short- and long-term fluctuations of yield. There are a number of useful tools that can help growers when they are making short- and long-term decisions. For example, there are crop models that predict yield rates and produce quality in defining climate control strategies, in synchronizing crop production with market demands, in handling the labour force, in emerging marketing strategies, and in maintaining a consistent year-round produce quality.

As we will show, EFuNN offers the advantage that it is able to model nonlinear system relationships and has been shown in other applications to be very robust when applied to data which is relatively imprecise, incomplete, and uncertain. EFuNN has been successfully applied in applications such as forecasting, control, optimization, and pattern recognition [7]. Intelligence is added to the process by computing the degree of uncertainty and computing with linguistic terms (fuzzy variables). More accuracy is obtained compared to mechanistic models.

Numerous studies have applied either neural networks (NNs) or fuzzy logic in greenhouse production systems. However, most of them have focused on modeling the air temperature in greenhouse environments [8–11] or optimal control of CO_2 with NN. Recent techniques have included modeling the greenhouse environment with hierarchical fuzzy modeling [12] or controlling the environment with optimized fuzzy control [13, 14]. Other studies concerning plant modeling have been reported: [15] implemented a hybrid neurofuzzy approach in terms of the system identification and modeling of the total dry weight yield of tomato and lettuce [16] and developed a fuzzy model to predict net photosynthesis of tomato crop canopies, and the results obtained correlated well with the results. TOMGRO [1] was used to model the prediction processes.

The objective of this study is to investigate how an IS technique such as EFuNN performs when applied to current crop and climate records from greenhouse growers, weekly prediction of greenhouse tomato yield from environmental and crop-related variables. Yield was characterized by yield per unit area (*Yield*, kg/m^2).

The rest of this paper is organized as follows. In Section 2, we discuss the methodology introduction and materials and methods. Section 3 is results, and finally Section 4 presents the conclusions.

2. Background to Rule Extraction

A wide variety of methods are now available, recently reviewed in [1, 17, 18]. Reference [17] revisits the Andrews classification of rule extraction methods and emphasizes distinction between decompositional and pedagogical approaches. Rule extraction methods usually start by finding a minimal network, in terms of number of hidden units and overall connectivity. The next simplification, the key feature of the method, is to cluster the hidden unit activations, then extract combinations of inputs, which will activate each hidden unit, singly or together, and thus the output generates rules as the general form of rules shown as follows:

$$\text{If } (x1 \le t1) \text{ AND} \cdots \text{AND } (xp \ge tp) \text{ Then } Ci. \quad (1)$$

Taha and Ghosh [19] suggest binary inputs generating a truth table from the inputs and simplifying the resultant Boolean function. The growth of computational time with number of attributes makes minimizing the size of the neural network essential and some methods evolve minimal topologies. The pedagogical approaches treat the neural network as a black box [20] and use the neural network only to generate test data for the rule generation algorithm.

2.1. Taxonomy of Rule Extraction Algorithms. It is now becoming apparent that algorithms can be designed which extract understandable representations from trained neural networks, enabling them to be used for decision-making. In this section, we use a taxonomy presented in [21] which uses three criteria for classification of rule extraction algorithms: scope of use, type of dependency with the method of solution of the type "black box," and format of the extracted rules. The algorithms can be a regression or classification algorithms. There are some algorithms that can be applied to both cases, such as the G-REX [21]. On the second criterion, an algorithm is considered independent if it is totally independent of the model type black box used (such as ANN and Support Vector Machines). The algorithms that use information of the black-box methods are called dependent methods. Regarding the format of the extracted rules, the methods can be classified into descriptive and predictive. The predictive algorithms perform extraction of rules that allow the expert to make an easy prediction for each possible observation from input space. If this analysis cannot be made directly, the algorithms are known only as descriptive.

3. Materials and Methods

3.1. EFuNN Evolving Fuzzy Neural Networks and the EFuNN Algorithm

3.1.1. Fuzzy Background. Fuzzy inference systems (FISs) are very useful for inference and handling uncertainty. The basic models presented are [14]. Some important issues that must be considered when building an FIS are identification of structure and estimation of parameters. Efficient structure identification optimizes the number of fuzzy rules and yields better convergence [22]. Different membership functions (MFs) can be attached to the neurons (triangular, Gaussian, etc.). The number of rules and the membership functions were estimated by the designers in early implementations of FIS [14]. A more efficient structure is then employed to optimize the number of rules in adaptive techniques which are appropriate for learning parameters that change slowly; handle complex systems with speedily changing characteristics, considering the fact that it takes a long time after every important change in the system to relearn model parameters [23, 24].

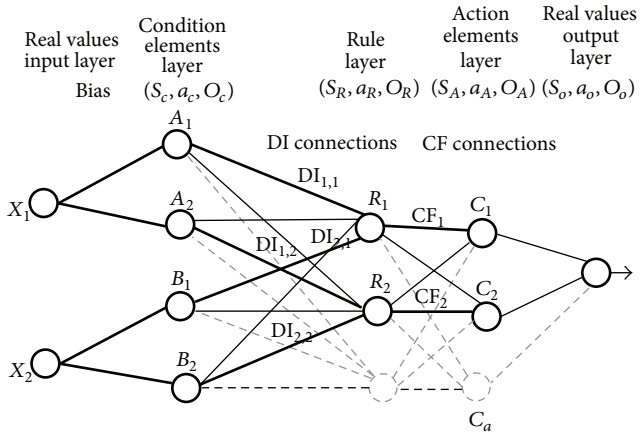

Figure 1: FuNN architecture.

References [14, 25] describe some techniques to learn fuzzy structure and parameters. In recent years, several evolving neurofuzzy systems (ENFSs) have been simulated, and these systems use online learning algorithms that can extract knowledge from data and perform a high-level adaptation of the network structure, as well as learning of parameters.

3.1.2. The General Fuzzy Neural Network (FuNN) Architecture. The fuzzy neural network (FuNN) (Figure 1) is connectionist feed-forward architecture with five layers of neurons and four layers of connections [26]. The first layer of neurons receives input information. The second layer calculates the fuzzy membership degrees of the input values which belong to predefined fuzzy membership functions, for example, small, medium, and large. The third layer of neurons represents associations between the input and the output variables, fuzzy If-Then rules. The fourth layer calculates the degrees to which output membership functions are matched by the input data, and the fifth layer does defuzzification and calculates values for the output variables. An FuNN has both the features of a neural network and a fuzzy inference machine. Several training algorithms have been developed for FuNN [26]: a modified back-propagation algorithm; a genetic algorithm; structural learning with forgetting; training and zeroing; combined modes. Several algorithms for rule extraction from FuNN have also been developed and applied. Each of them represents each rule node of a trained FuNN as an If-Then fuzzy rule.

3.2. EFUNN Architecture. EFuNNs are FuNN structures that evolve according to the Evolving Connectionist Systems (ECOS) principles. That is, all nodes in an EFuNN are generated through learning. The nodes representing membership functions can be modified during learning. As in FuNN, each input variable is represented here by a group of spatially arranged neurons to represent different fuzzy domain areas of this variable. Different membership functions can be attached to these neurons (triangular, Gaussian, etc.). New neurons evolve in this layer if for a given input vector the corresponding variable value does not belong to any of the existing membership functions to a membership degree greater than a membership threshold, and this means that new fuzzy label neuron or an input variable neuron can be created during the adaptation phase of an EFuNN.

3.3. Yield Records and Climate Data. Crop records and climate data from a tomato greenhouse operation in *WSG* (*Wight Salads Group*) *in the Isle of Wight*, *United Kingdom*, were used to design and train evolving fuzzy neural networks for yield prediction. We used fuzzy inference system for implementing input parameter characterization. The data includes six datasets from two production cycles (S1: 2004 to 2007 and S2: 2008) and one greenhouse section (New Site). The total number of records is 1286, and each record included 14 parameters characterizing the weekly greenhouse environment and the crop features.

The environmental parameters which are to be controlled are the vapor pressure deficit (VPD) and the differential temperature between the daytime to nighttime ($T_d - T_n$). The setpoints for each environmental treatment were VPD = 2.0 kPa, 0.6 kPa, and uncontrolled; T_d/T_n = 26°C/18°C, 20°C/20°C, and 22°C/18°C, respectively, for each compartment. The date when they were planted and the period over which the crops were allowed to grow in the case of both datasets are summarized in Table 1.

The tomatoes were grown in greenhouses on a high-wire system with hydroponic, CO_2 supplementation, and computer climate/irrigation control. The greenhouses were equipped with hot-water heating pipes and roof vents for passive cooling.

The crop records consisted of 12 plant samples per greenhouse section that were randomly selected and continuously measured during the production cycle. The crop record data were collected by direct observation and by manually measuring or counting each of the morphological features. This system included an electronic sensor unit which measured the air temperature, humidity, and CO_2 concentration in each of the greenhouse sections. Outside weather conditions were determined via a weather station. Daytime, nighttime, and 24 h averages of daily climate data from outside and inside of the greenhouse section were also obtained by the grower; weekly averages were computed to match the weekly crop records.

3.4. Modeling the Parameters. Yield (Y_a, kg m^{-2} week^{-1}) is of interest to greenhouse growers as a means via which they can develop short-term crop management strategies; it is also useful for labour management strategies. Greenhouse tomato cumulative yield can be described, either as fresh weight (Cockshull, 1988) or as dry mass [27]. Both of these studies were performed in northern latitudes, without CO_2 enrichment, for short production periods (<100 days), and the plants were cultivated in soil, not hydroponically. These are not currently standard cultivation methods because most of the greenhouse growers make use of high-technology production facilities.

Yield development is influenced mainly by fruit temperature. This parameter is inversely related to fruit development

Table 1: Summary of datasets. Crop records include samples per week in New House Sites.

Season (years)	Greenhouse	Transplanted on WOY[a] and date	Crop duration[b] (weeks)	Cultivar
		Commercial DS		
1	New h21	(26) June 24, 2004	42	Campari
(2004-2005)	New h22	(28) July 8, 2004	47	Campari
2	New h21	(31) July 28, 2005	43	Campari
(2004–2007)	New h22	(25) June 16, 2005	44	Campari
	New h22	(33) Aug. 11, 2006	36	Cherry
	New h22	(41) Oct. 6, 2007	39	Cherry

[a, b]WOY: week of the year number.

```
for each evolving layer neuron h do
    Create a new rule r
    for each input neuron i do
        Find the condition neuron c with the largest
        weight W_{c,h}
        Add an antecedent to r of the form "i is c
        W_{c,h}" where W_{c,t} is the confidence factor for
        that antecedent
    end for
    for each output neuron o do
        Find the action neuron a with the largest weight
        W_{h,a}
        Add a consequent to r of the form "o is a
        W_{h,a}" where W_{h,a} is the confidence factor for
        that consequent
    end for
end for
```

Algorithm 1: EFuNN algorithm steps.

rate and shows a linear relationship with air temperature [28, 29]. This relationship is shown in Algorithm 1 for all the datasets, which show the relation between yield and air temperature (Ti_n 24).

3.4.1. Data Processing. The steps of the preprocessing include making average through a certain amount of the certain point of some data records. We preprocessed 5 environmental variables (CO_2, temperature, vapor pressure deficit (VPD), yield, *and radiation*) for different tomato cultivars, from different greenhouses in WSG area, which were not ready for processing but had to be pre-processed. For instance, some values in some tomato records were missing and we had to replace them with 0, being not ready to be fed to an artificial neural network for processing. Thus, three preprocessing steps were taken.

(1) Edit each data file and group same tomato cultivar and type in one file with all environmental variables of that cultivar gathered from different greenhouses.

(2) For each cultivar, store values in a Microsoft Excel spreadsheet. Next, in the spreadsheet, 0 replaced null values. For some VPD and radiation missing values are averaged.

(3) The Excel spreadsheet content was converted to.dat file input to be fed into Matlab and EFUNN application.

3.5. Neural Network Model. Computational NNs have proven to be a powerful tool to solve several types of problems in various real life fields where approximation of nonlinear functions, classification, identification, and pattern recognition are required. NNs are mathematical representations of biological neurons in the way that they process information as parallel computing units. In general, there are two types of neural network architecture: (1) static (feedforward), in which no feedback or time delays exist, and (2) dynamic neural networks, whose outputs depend on the current or previous inputs, outputs, or states of the network (Demuth et al. [30]).

We consider a neural network that consists of an input layer with nþ1 nodes, a hidden layer with h units, and an output layer with l units as follows:

$$y_i = g\left(\sum_{j=1}^{h} wijf\left(\sum_{k=1}^{n+1} vjkXk\right)\right), \quad i = 1, 2, \ldots, l, \qquad (2)$$

where xk indicates the kth input value, yi the ith output value, vjk a weight connecting the kth input node with

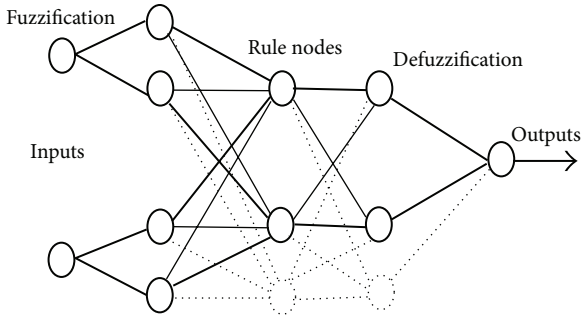

Figure 2: Simple EFuNN structure.

the *j*th hidden unit, and *wij* a weight between the *j*th hidden unit and the *i*th output unit. We start learning with one hidden layer, which would be updating the weights and thresholds to minimize the objective function by any optimization algorithm then terminate the network training with this number of hidden units when a local minimum of the objective function has been reached. If the desired accuracy is not reached, increase the number of hidden units with random weights and thresholds and repeat updating; otherwise, finish training and stop.

3.6. The EFuNN Algorithm. EFuNN consists of a five-layer structure, which begins with the input layer representing input variables, and each input variable is a presentation of group of arranged neurons representing a fuzzy quantization of this variable, which is then presented to the second layer of nodes. Different membership functions (MFs) can be attached to these neurons (triangular, Gaussian, etc.), where continuing modifications of nodes with a membership functions can be applied during learning.

Through node creation and consecutive aggregation, an EFuNN system can adjust over time to changes in the data stream and at the same time preserve its generalization capabilities. If EFuNNs use linear equations to calculate the activation of the rule nodes (instead of using Gaussian functions and exponential functions), the EFuNN learning procedure is faster. EFuNN also produces a better online generalization, which is a result of more accurate node allocation during the learning process. As Algorithm 1 shows, EFuNN allows for continuous training on new data, further testing, and also training of the EFuNN on the test data in an online mode, leading to a significant improvement of the accuracy.

The third layer contains rule nodes that evolve through hybrid-supervised learning. These rule nodes present prototypes of input-output data associations, where each rule node is defined by two vectors of connection weights $W1(r)$ and $W2(r)$; also, *W2* is adjusted through learning depending on output error, and *W1* is modified based on similarity measure within input space in the local area.

The fourth layer of neurons represents fuzzy quantification of the output variables, similar to the input fuzzy neurons representation. The fifth layer represents the real values for the output variables. In the case of a "one-of-*n*" mode, EFuNN transmits the maximum activation of the rule to the next level. In the case of a "many-of-*n*" mode, the activation values of m ($m > 1$) rule nodes that are above an activation threshold are transmitted further in the connectionist structure.

In this paper, EFuNN's: Figure 2 shows the evolving algorithm is based on the principle that rule nodes only exist if they are needed. As each training example is presented, the activation values of the nodes, the rule and action layers, and the error over the action nodes are examined; if the maximum rule node activation is below a set threshold, then a rule node is added. If the action node error is above a threshold value, a rule node is added. Finally, if the radius of the updated node is larger than a radius threshold, then the updating process is undone and a rule node is added.

EFuNN has several parameters to be optimized and they are

(1) number of input and output;

(2) learning rate for *W1* and *W2;*

(3) pruning control;

(4) aggregation control;

(5) number of membership functions;

(6) shape of membership functions;

(7) initial sensitivity threshold;

(8) maximum radius;

(9) *M*-of-*n* value.

3.7. Input/Output Parameters. The architecture and training mode of EFuNN depends on the input and output parameters as determined by the problem being solved. The tomato crop data includes biological entities that show nonlinear dynamic behaviour and whose response depends not only on several environmental factors but also on the current and previous crop conditions. Greenhouse tomatoes have long production cycles, and the total yield is determined by these parameters. The effects of several environmental factors (light, CO_2, air humidity, and air temperature) have both short- and long-term impacts on tomato plants. Air temperature directly affects fruit growth, and according to [28], it is the influential parameter on the growth process Figure 3.

Greenhouse tomatoes have a variable fruit growth period, ranging from 40 to 67 days. This deviation results from the changing 24-hour average air temperature in the greenhouse. Here the objective was to design an EFuNN that is simple enough and accurate enough to predict the variables of interest, since for many practical problems variables have

$$D_n = \frac{(1/2)\left(\sum_{i=1}^{c} \left|I_i - W_{i,n}\right|\right)}{\sum_{i=1}^{c} W_{i,n}} \tag{3}$$

different levels of importance and make different contributions to the output. Also, it is necessary to find an optimal normalization and assign proper importance factors to the variables, reduce the size of input vectors, and keep only the most important variables. This dynamic network architecture was chosen because of its memory association and learning

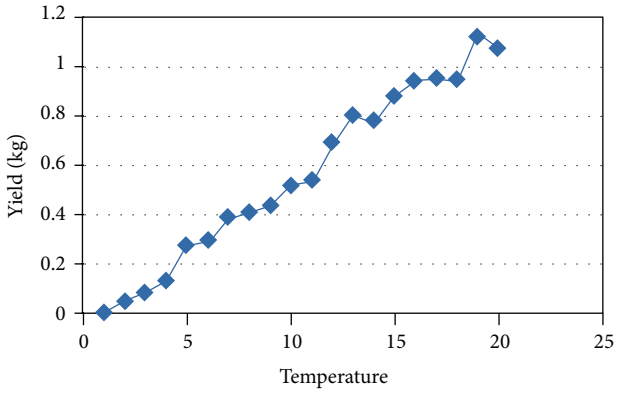

Figure 3: 24 h average air temperature relationship with yield grown in tomatoes greenhouse.

capability with sequential and time-varying patterns, which is most likely the biological situation for tomato plants.

4. Experiments Setup: Training and Performance Evaluation

The training process implies iterative adjustment of the biases of the network by minimizing a performance function when presenting input and target vectors to the network. The mean square error (MSE), selected as the performance function (Table 2), was calculated as the difference between the target output and the network output.

The supervised learning in EFuNN is built on the previously explained principles, so that when a new data example is presented, the EFuNN creates a new rule node to memorize the two input and output fuzzy vectors and/or adjusts the winning rule node. After a certain number of examples are applied, some neurons and connections may be pruned or aggregated and only the best are chosen. Different pruning rules can be applied in order to realize the most successful pruning of unnecessary nodes and connections. Although there are several options for growing the EFuNN, we restricted the learning algorithm to the 1-of-n algorithm.

Using Gaussian functions and exponential functions, the EFuNN learning procedure produces a better online generalization, which is a result of more accurate node allocation during the learning process. EFuNN allowed continuous training on new data; we did further testing and also training of the EFuNN on the test data in an online mode, which led to a significant improvement in accuracy. A significant advantage of EFuNNs is the local learning which allowed a fast adaptive learning where only few connections and nodes are changed or created after the entry of each new data item. This is in contrast to the global learning algorithms where, for each input vector, all connection weights changed.

Tomato plants were included in the training, testing, and validation process. The initial datasets were randomly divided so that 60% (771) of the records were assigned to the training set, 20% (257) to the validation set, and 20% (257) to the test set. The generalization in each of the networks

Table 2: Test results and performance comparison of demand forecasting.

	EFuNN	ANN (Multilayer perceptron)
Learning epochs	1	2500
Training error (RMSE)	0.0013	0.116
Testing error (RMSE)	0.0092	0.118
Computational load (in billion flops)	0.536	87.2

Table 3: Results from 4 runs of the EFuNN.

RMS (training)	No. of rule nodes	Accuracy (%)
0.0045	77	82.95
0.0023	48	86.00
0.0029	75	84.91
0.0028	81	80.79

was improved by implementing the early stopping method through the validation set. After training each of the networks to a satisfactory performance, the independent validation set was used to evaluate the prediction performance and to present the results.

We initialized $x(t)$, $x(t-1)$, $x(t-2)$, $x(t-n)$ in order to predict the $x(t+1)$, and the training was replicated three times using three different samples of training data and different combinations of network parameters. We used four Gaussian MFs for each input variable, as well as the following evolving parameters: number of membership functions = 3; sensitivity threshold $Sthr = 0.99$; learning rate = 0.25; error threshold $Errthr = 0.001$; learning rates for first and second layer = 0.05. EFuNN uses a one-pass training approach. The network parameters were determined using a trial and error approach. The training was repeated three times after reinitializing the network and the worst errors were reported in Figure 5. Online learning in EFuNN resulted in creating 2122 rule nodes as depicted in Figure 6. We illustrate the EFuNN training results, and the training performance is summarized in Table 3.

An investigation of the extracted rule set shown in Figure 4 from run 2 indicates that different compartments from the mentioned environmental variables affected yield prediction. Rules can be obtained from three different kinds of sources: human experts, numerical data, and neural networks. All the obtained rules can be used in rule selection method to obtain a smaller linguistic rule-based system with a higher performance. We explain the fuzzy arithmetic-based [31] approach to linguistic rule extraction from trained EFuNN for modeling problems using some computer simulations. Assume that our training output was five rules; for the nonlinear function realized by the trained neural network, we assume that the five terms (rules) are given for each of the two input variables. We also assume that the same five terms are given for the output variable.

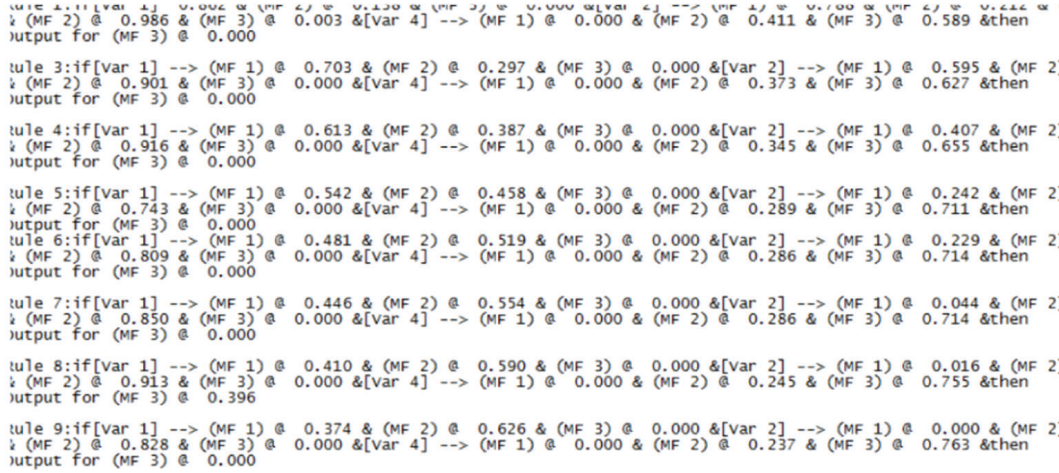

Figure 4: Some rules extracted from EFuNN simulation.

The number of combinations of terms is 25. Each combination is presented to the trained neural network as a linguistic input vector $Aq = \{Aqi, \ldots, Aq2\}$. The corresponding fuzzy output Oq is calculated by fuzzy arithmetic. This calculation is numerically performed for the h-level sets of Aq for $h = 0.1, 0.2, \ldots, 1.0$. The fuzzy output Oq is compared with each of the five linguistic terms. The linguistic term with the minimum difference from the fuzzy output Oq is chosen as the consequent part Bq of the linguistic rule Rq with the antecedent part Aq. For example, let us consider the following linguistic rule.

Rule Rq: If $X1$ is medium and $X2$ is small Then y is Bq.

To determine the consequent part Bq, the antecedent part of the linguistic rule Rq is presented to the trained neural network as the linguistic input vector. The corresponding fuzzy output, Oq is calculated.

As a result, the rules shown in Table 3 will be shrinked into the range of [5–10] rules, which make it more efficient for operators to work with.

The results obtained from this study indicated that the EFuNN had successfully learnt the input variables and was then able to use these variables as a means of identifying the estimated yield amount.

From the results obtained from testing the EFuNN and its relative performance against Multilayer Perceptron, it is evident that the EFuNN models this task far better than any of the other models. In addition, the rules extracted from the EFuNN reflect the nature of the training data set and confirm the original hypothesis that more rules would need to be evolved to get better predictions. Accounting for the poor performance of the Multilayer Perceptron could be explained by examining the nature of this connectionist model. Having a fixed number of hidden nodes limits the number of hyperplanes these models can use to separate the complex feature space. Selection of the optimal number of hidden nodes becomes a case of trial and error. If these are too large, then the ability for the Multilayer Perceptron to generalize for new instances of records is reduced due to the possibility of overlearning the training examples. This work has sought to describe a problematic area within horticulture produce management, that of predicting yields in greenhouses. The EFuNN has clearly stood out as an appropriate mechanism for this problem and has performed comparably well against other methods. But the most beneficial aspect of EFuNN architecture is the rule extraction ability. By extracting the rules from the EFuNN, we can analyse why the EFuNN made it more clear and identified deficiencies in its ability to generalize to new unseen data instances.

Table 4: Accuracy results for different algorithms.

Classifier	C-week	Week + 1	Week + 2
MLP	81.108	81.309	78.531
RBF	77.44	80.371	78.885
EFuNN	79.557	86.257	83.992

In terms of the size of the architecture for the Multilayer Perceptron, this was extremely large. This was because of the length of the input vector. Input vectors of this size require a significant amount of presentations of the training data for the Multilayer Perceptron to successfully learn the mapping between the input vectors and the output vectors. In addition, the small numbers of hidden nodes contained in the Multilayer Perceptron were unable to represent the mapping between the inputs and outputs. Raising the number of hidden nodes increased the ability for the Multilayer Perceptron to learn but created a structure that was an order of magnitude larger and thus took even longer to train. In conclusion it can be seen from the results of this experiment that the advantages of the EFuNN are twofold. One, the time taken to train the EFuNN was far less than Multilayer Perceptron. And two, the EFuNN in Table 4 was able to successfully predict yield for a given period of time in Figure 5. This indicates that the EFuNN has the ability to store a better representation of the temporal nature of the data and, to this end, generalize better than the other methods.

Results shown in Table 4 show the accuracy of our method compared to other classifiers like Bayesian, RBF

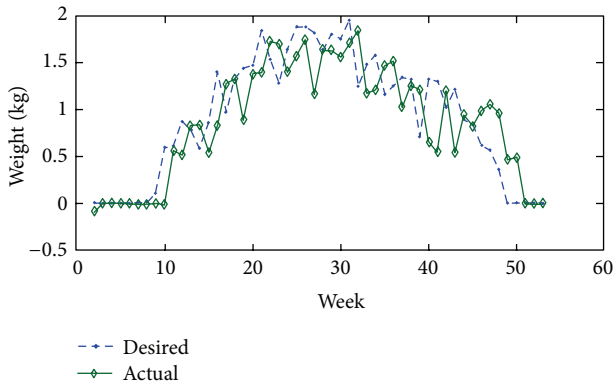

Figure 5: Training result with EFuNN.

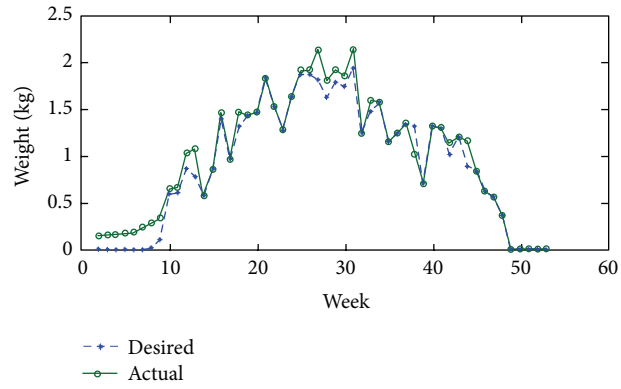

Figure 6: Test results and performance comparison of demand forecasts.

Network, and so forth. Results for those classifiers were constructed using Weka experimenter; Weka is a collection of machine learning algorithms for data mining tasks, and the algorithms can either be applied directly to a data set or called from your own Java code. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization. It is also well suited for developing new machine learning schemes (http://www.cs.waikato.ac.nz/ml/weka/).

Performance of the learning algorithm is evaluated by the accuracy (%) on Cherry and Campari tomato data in different greenhouses, for four years (2004–2007).

In this paper, we have described how EFuNN can be applied in the domain of horticulture, especially in the challenging area of deciding support for yield prediction, which leads to the production of well-determined amounts. These results do not provide a mechanistic explanation of the factors influencing these fluctuations. However, knowing this information in advance could be valuable for growers for making decisions on climate and crop management. Some of the advantages of the neural network model implemented in this study include the following: (1) The input parameters of the model are currently recorded by most growers, which makes the model easy to implement; (2) the model can "learn" from datasets with new scenarios (new cultivars, different control strategies, improved climate control, etc.); (3) less-experienced growers could use the system because the decision-making process of the most experienced growers is captured by the data used in the trained networks, and production could thereby become more consistent.

5. Conclusion

It is feasible to implement Intelligent System (IS) techniques, including NN and fuzzy logic, for modeling and predicting of a greenhouse tomato production system. Data from experimental trials and from a commercial operation for complete production cycles allowed the modeling of tomato yields. IS techniques were robust in dealing with imprecise data, and they have a learning capability when presented with new scenarios and need to be tested on different tomato cultivars.

Experimentation results show that EFuNN performed better than other techniques like ANN in terms of low RMSE error and less computational loads (performance time). As showed in Figure 6, ANN training needs more epochs (longer training time) to achieve a better performance. EFuNN makes use of the knowledge of FIS and the learning done by NN. Hence, the neurofuzzy system is able to precisely model the uncertainty and imprecision within the data as well as to incorporate the learning ability of NN. Even though the performance of neurofuzzy systems is dependent on the problems domain, very often the results are better while compared to a pure neural network approach. Compared to NN, an important advantage of neurofuzzy systems is its reasoning ability (*If-Then* rules) within any particular state. A fully trained EFuNN could be replaced by a set of *If-Then* rules.

As EFuNN adopts a single pass training (1 epoch), it is more adaptable and easy for further online training which might be highly useful for online forecasting. However, an important disadvantage of EFuNN is the determination of the network parameters like number and type of MF for each input variable, sensitivity threshold, error threshold, and the learning rates.

Similar procedures might be used to automatically adapt the optimal combination of network parameters for the EFuNN.

Acknowledgments

The authers would like to thank the Wight Salads Group (WSG) Company and Warwick Horticultural Research Institute (WHRI) for providing the datasets which have been used in this research and for their help and cooperation in providing all the necessary information. They acknowledge the support of the Horticulture Development Company (HDC) who has provided the Ph.D. degree studentship support.

References

[1] J. W. Jones, E. Dayan, L. H. Allen, H. van Keulen, and H. Challa, "Dynamic tomato growth and yield model (TOMGRO)," *Transactions of the ASAE*, vol. 34, no. 2, pp. 663–672, 1991.

[2] J. W. Jones, A. Kening, and C. E. Vallejos, "Reduced state-variable tomato growth model," *Transactions of the ASAE*, vol. 42, no. 1, pp. 255–265, 1999.

[3] H. Heuvelink, "Growth, development and yield of a tomato crop: periodic destructive measurements in a greenhouse," *Scientia Horticulturae*, vol. 61, no. 1-2, pp. 77–99, 1995.

[4] E. Heuvelink, *Tomato growth and yield: quantitative analysis and synthesis [Ph.D. thesis]*, Wageningen Agricultural University, Wageningen, The Netherlands, 1996.

[5] P. Abreu, J. F. Meneses, and C. Gary, "Tompousse, a model of yield prediction for tomato crops: calibration study for unheated plastic greenhouses," *Acta Horticulturae*, vol. 519, pp. 141–150, 2000.

[6] S. R. Adams, "Predicting the weekly fluctuations in glasshouse tomato yields," *Acta Horticulturae*, vol. 593, pp. 19–23, 2002.

[7] R. M. de Moraes and L. dos Santos Machado, "Evaluation system based on EFuNN for on-line training evaluation in virtual reality," in *Proceedings of the 10th Iberoamerican Congress on Pattern Recognition, Image Analysis and Applications (CIARP '05)*, vol. 3773 of *Lecture Notes in Computer Science*, pp. 778–785, 2005.

[8] H. U. Frausto, J. G. Pieters, and J. M. Deltour, "Modelling greenhouse temperature by means of auto regressive models," *Biosystems Engineering*, vol. 84, no. 2, pp. 147–157, 2003.

[9] R. Linker, I. Seginer, and P. O. Gutman, "Optimal CO_2 control in a greenhouse modeled with neural networks," *Computers and Electronics in Agriculture*, vol. 19, no. 3, pp. 289–310, 1998.

[10] I. Seginer, "Some artificial neural network applications to greenhouse environmental control," *Computers and Electronics in Agriculture*, vol. 18, no. 2-3, pp. 167–186, 1997.

[11] I. Seginer, T. Boulard, and B. J. Bailey, "Neural network models of the greenhouse climate," *Journal of Agricultural Engineering Research*, vol. 59, no. 3, pp. 203–216, 1994.

[12] P. Salgado and J. B. Cunha, "Greenhouse climate hierarchical fuzzy modelling," *Control Engineering Practice*, vol. 13, no. 5, pp. 613–628, 2005.

[13] H. Ehrlich, M. Kühne, and J. Jäkel, "Development of a fuzzy control system for greenhouses," *Acta Horticulturae*, vol. 406, pp. 463–470, 1996.

[14] L. A. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, no. 3, pp. 338–353, 1965.

[15] B. T. Tien, *Neural-fuzzy approach for system identification [Ph.D. thesis]*, Wageningen Agricultural University, Wageningen, The Netherlands, 1997.

[16] B. Center and B. P. Verma, "A fuzzy photosynthesis model for tomato," *Transactions of the ASAE*, vol. 40, no. 3, pp. 815–821, 1997.

[17] M. W. Craven and J. W. Shavlik, "Using sampling and queries to extract rules from trained neural networks," in *Proceedings of the 11th International Conference on Machine Learning*, pp. 37–45, 1994.

[18] HortiMax, *Village Farms, USA, Reports Record Tomato Production*, HortiMax Growing Solutions, Rancho Santa Margarita, Calif, USA, 2008.

[19] I. Taha and J. Ghosh, "Three techniques for extracting rules from feedforward networks," *Intelligent Engineering Systems through Artificial Neural Networks*, vol. 6, pp. 5–10, 1996.

[20] R. Linker and I. Seginer, "Greenhouse temperature modeling: a comparison between sigmoid neural networks and hybrid models," *Mathematics and Computers in Simulation*, vol. 65, no. 1-2, pp. 19–29, 2004.

[21] J. Huysmans, B. Baesens, and J. Vanthienen, *Using Rule Extraction to Improve the Comprehensibility of Predictive Models*, Technical Report KBI, 0612, vol. 43, Department of Decision Sciences and Information Management, Katholieke Universiteit Leuven, Leuven, Belgium, 2006.

[22] P. Costa, *A quantified approach to tomato plant growth status for greenhouse production in a semi arid climate [Ph.D. thesis]*, University of Arizona, Tucson, Ariz, USA, 2007.

[23] N. Kasabov, "Adaptable neuro production systems," *Neurocomputing*, vol. 13, no. 2–4, pp. 95–117, 1996.

[24] N. Kasabov, *Foundations of Neural Networks, Fuzzy Systems, and Knowledge Engineering*, MIT Press, Cambridge, Mass, USA, 1996.

[25] M. H. Jensen, "Steering your tomatoes towards profit," in *Greenhouse Crop Production and Engineering Design Short Course*, Controlled Environment Agriculture Center, University of Arizona, Tucson, Ariz, USA, 2004.

[26] J. J. Buckley and E. Eslami, *An Introduction to Fuzzy Logic and Fuzzy Sets*, Physica, Heidelberg, Germany, 2002.

[27] M. T. Hagan, H. B. Demuth, and M. H. Beale, *Neural Network Design*, University of Colorado, Boulder, Colo, USA, 1995.

[28] E. Heuvelink, "Developmental process," in *Tomatoes*, E. Heuvelink, Ed., Crop Production Science in Horticulture Series, pp. 53–83, CABI Publishing, Wallingford, UK, 2005.

[29] E. Heuvelink and M. Dorais, "Crop growth and yield," in *Tomatoes*, E. Heuvelink, Ed., Crop Production Science in Horticulture Series, pp. 85–144, CABI Publishing, Wallingford, UK, 2005.

[30] H. Demuth, M. Beale, and M. Hagan, *Neural Network Toolbox 5: User's Guide*, The MathWorks, Inc., Natick, Mass, USA, 2007.

[31] H. Ishibuchi, T. Nakashima, and M. Nii, *Classification and Modeling with Linguistic Information Granules: Advanced Approaches to Linguistic Data Mining*, Springer, Berlin, Germany, 2004.