Predictive modeling in food mycology using adaptive neuro-fuzzy systems.

Mahdi Amina
Vassilis Kodogiannis
Andrzej Tarczynski

School of Electronics and Computer Science

Predictive Modeling in Food Mycology Using Adaptive Neuro-Fuzzy Systems

Mahdi Amina
Centre for System Analysis
School of Informatics
University of Westminster
London W1W 6UW, UK
mahdiamina@gmail.com

Vassilis Kodogiannis
Centre for System Analysis
School of Informatics
University of Westminster
London W1W 6UW, UK
kodogiv@wmin.ac.uk

Andrzej Tarczynski
Centre for System Analysis
School of Informatics
University of Westminster
London W1W 6UW, UK
tarczya@wmin.ac.uk

Abstract - **Fungal growth leads to spoilage of food and animal feeds and to formation of mycotoxins and potentially allergenic spores. There is a growing interest in predictive modeling microbial growth as an alternative to time consuming traditional, microbiological enumeration techniques. Several statistical models have been accounted to describe the growth of different micro-organisms. However neural networks, as highly nonlinear approximator scheme, have the potential of modeling some complex, phenomena better than the others. The application of adaptive neuro-fuzzy systems in predictive microbiology is presented in this paper. This technique is used to build up a model of the joint effect of water-activity, pH level and temperature to predict the maximum specific growth rate of the Ascomycetous Fungus Monascus Ruber. The proposed scheme is compared against standard neural network approaches. Neuro-fuzzy systems offer an alternative and powerful technique to model microbial kinetic parameters and could thus become an efficient tool in predictive mycology.**

Index Terms: *Fuzzy-Neural networks, Modeling, Rule optimization, Parameter/ Structure learning.*

I. INTRODUCTION

Growth-predictive models are currently accepted as informative tools that assist rapid and cost-effective assessment of microbial growth for product development, risk assessment, and education purposes [1]. More recently, predictive microbiology has been used to forecast the growth of spoilage micro-organisms in order to study the shelf life of a food product. Fungal spoilage of food commodities causes significant economic losses. Although industrial standards have been greatly improved in the last years, food spoilage by fungi is still a major concern for both food producers and regulatory agencies. Today, there is a need for understanding fungal growth in foods, particularly those factors associated with new manufacturing processing and packaging techniques [2]. Fungal presence in food may adversely affect not only the organoleptic value of the commodity but most importantly its nutritional value by producing toxic metabolites, thus a public health risk is inevitable [3]. It is mainly due to the appearance of visible mycelium and off-flavor development, leading to consumer rejection. Improvement of food quality and safety, demands the development of appropriate tools allowing prediction of fungal growth. In the last years, predictive microbiology has been focused on food-borne pathogens. Predictive modeling of filamentous fungi.
had not received the same level of attention [4]. Recently, the situation has changed and a growing number of studies are available in the literature dealing with the predictive modeling approach of fungi [5].
Polynomial models have been widely used in predictive microbiology for the quantitative assessment of the effects of various environmental factors on fungal growth [6]. However, a major disadvantage of these models is that they are developed from linear and quadratic combinations of variables; use of such simplified models may not be justified. Neural networks (NNs) have been deployed in recent years as an alternative to conventional statistical models, due to their ability to describe highly complex and non-linear problems in many fields of science. The NN-based methodologies have been applied in predictive food microbiology [7]. The main characteristics of NNs, such as (i) non-linearity, allowing better fit to the data, (ii) noise-insensitivity, providing more accurate predictions in the presence of uncertain data and measurement errors, enabling application of the model to unknown data make them an interesting tool in an area which is dominated by statistical analysis tools [8]. Several published works indicate that neural network-based models produce better estimation of kinetic parameters of micro-organisms than response surface models. In a recent study, NNs have been compared with response surface models in modeling the growth rate of *L. plantarum* and *E. coli.* It was reported that the NN approach outperformed the statistical models based on its lower standard error of prediction (SEP) term, despite the fact that NN models had higher degree of complexity [9].

Monascus is an ascomycetous fungus traditionally used for the production of food coloring, fermented foods and beverages in southern China, Taiwan, Japan, Thailand, Indonesia and the Philippines [15]. Members of the genus can commonly survive heat treatments and grow under reduced oxygen levels, resulting in food spoilage. *Monascus ruber* is a widespread ascomycetous fungus in Europe, as it is common in silage and deteriorating grain. One characteristic of the genus is the production of ascospores capable of surviving heat treatment. Subsequently they can grow under reduced-oxygen environment and cause food spoilage. In Greece, the fungus has been isolated from the brine of thermally-processed green olives of the *Conservolea* variety and may result in economically significant spoilage losses [10]. Spoilage may result from the development of a mycelia mat on the surface of the olives, and from a softening of the fruits and changes in the pH of the final product. Temperature, pH and water

978-1-4244-3806-8/09/$25.00 © 2009 IEEE

activity (a_w) are generally regarded as the principal controlling factors during fermentation and subsequent storage of table olives. A combination of these factors could effectively control the growth of the fungus during storage. Predictive modeling has been extensively used mainly to predict bacterial growth as a function of environmental factors such as temperature, pH and Water Activity [15]. However, model development of filamentous fungal growth has not received the same level of attention as that of bacterial growth. A few studies concerning fungal growth have dealt with the predictive modeling approach [15].

In recent years, the fuzzy neural network approach has gained considerable interest as a tool capable of solving real-world problems, such as modeling and control of highly complex systems, signal processing and pattern recognition [14]. Extensive experimentation has demonstrated that the class of feed-forward fuzzy neural networks exhibits a number of significant advantages compared to the neural network models. First, the neural networks are global models where training is performed on the entire pattern range. On the contrary, owing to the partition of the input space, the fuzzy models perform a fuzzy blending of local models in space. As a result, faster convergence, comparing to other traditional methods, is achieved during learning for a specific task. Secondly, fuzzy neural networks are capable of incorporating both numerical data (quantitative information) and expert's knowledge (qualitative information) and describe them in the form of linguistic IF-THEN rules. In that respect, they provide a unified framework for integrating the computational parallelism and low-level learning of neural networks with the high-level reasoning of fuzzy systems. The above feature assists determining the initial structure, and leads to models with fewer parameters compared to neural networks.

Based on the Takagi–Sugeno-Kang (TSK) fuzzy system structure, Jang [12] developed an adaptive network -ANFIS (which can be regarded as a neural network) in which hybrid learning rules are used. An error backpropagation algorithm is applied to tune premise parameters and the least-squares estimation technique is issued to estimate consequence parameters. The membership functions are chosen to be bell shape functions (highly nonlinear functions, e.g., the Gaussian type). The performance of the network is indeed very good. Nevertheless, the network suffers from the limitation that if the dimension of the system is large, the number of input fuzzy partitions is large and hence the required number of rules and consequence parameters will be very large. The least-squares estimation algorithm cannot be implemented easily because the calculation of very large matrices is required. Thus, the application of the network is limited to some low-dimensional systems.

In the present work, the predictive capability of an adaptive neuro-fuzzy system was tested on this fungus in comparison to conventional neural networks approaches. More specifically, the purpose of the present work is (i) to develop an intelligent methodology based on neuro-fuzzy networks to predict the combined effect of temperature, water activity and pH on the maximum specific growth rate of *Monascus ruber van Tieghem*, and (ii) to compare the prediction accuracy of the proposed intelligent scheme and classic neural networks approaches such as multilayer perceptrons and radial basis functions.

II. ADAPTIVE FUZZY NEURAL NETWORK

Over the past two decades, research has been carried out to combine the computational power of the neural networks with the inference attributes of the fuzzy logic systems. These attempts have led to the development of the Fuzzy Neural Network (FNN), a layer-based connectionist structure tool where expert knowledge is submerged in the form of IF-THEN rules [24]. A typical Sugeno-type adaptive neuro fuzzy inference network (ANFIS) is illustrated in Fig.1

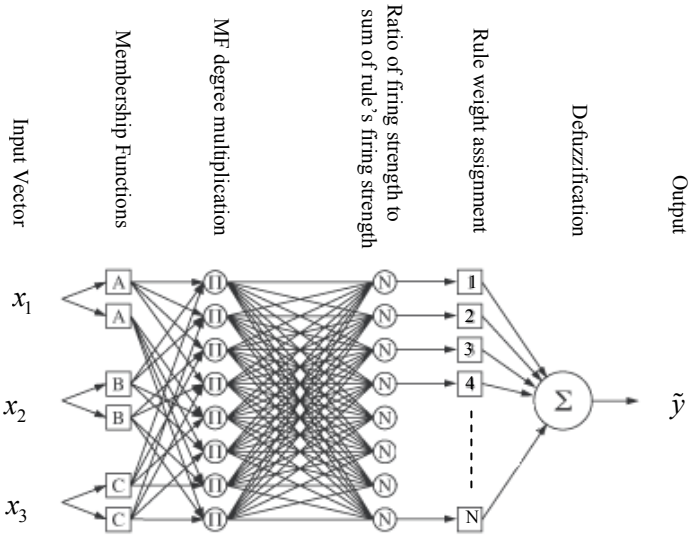

Fig.1- Three Input and One output ANFIS structure

In conventional ANFIS structure the number of membership functions and therefore the number of rules is fixed, and increases significantly by increasing the number of membership functions (MFs) and input dimensionality.

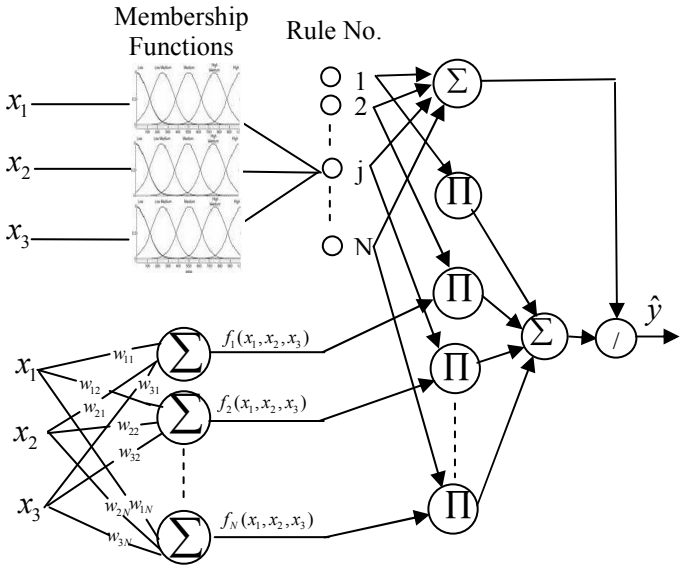

Fig.2- Three input and one output Adaptive Neuro-fuzzy scheme

An alternative adaptive fuzzy neural network (AFNN) proposed by J.Theocharis[18] is applied in this paper. Its main characteristics are the self-construction ability, parameter learning ability and rule extraction ability. An outline is shown in Fig. 2. In contrary to ordinary ANFIS, the adaptive FNN has a structure-learning mechanism which creates/adjusts the structure of its premise part as training proceeds. The fuzzy inference system considered in this network follows Takagi Sugeno's IF-THEN rules [19], in the form of:

$$R^{(j)}: \text{IF } x_1 \text{is} A_1^j \text{ AND } x_2 \text{is} A_2^j \text{ AND} \ldots \text{AND } x_m \text{is} A_m^j$$
$$\text{THEN}$$
$$f_j = w_{oj} + w_{1j}x_1 + w_{2j}x_2 + \ldots + w_{mj}x_m \quad \text{for} \quad j = 1,\ldots,n \qquad (1)$$

Equation (1) is an *"AND"* rule which means all the conditions of the IF part must be met simultaneously in order for the result of the THEN part to occur, where *m* is the dimension of input components, *n* depicts number of rules and w_{ij} are the polynomial coefficients, linearly connecting the input variables to the rules' outputs f_j. Finally, A_i^j denote the labels of fuzzy sets outputs. Each linguistic label A_i^j , is associated with a membership function, which specifies the degree to which a given μ_i satisfies the quantifier A_i^j [18][24].

The membership functions considered in this paper are of Gaussian type

$$\mu_{A_{ij}}(x_i) = \exp\left[-\frac{(x_i - m_{ij})^2}{\sigma_{ij}^2}\right] \qquad (2)$$

The degree of fulfilment represents the degree to which each rule participates in the output defined by a t-norm(*) operator is defined as follows:

$$\mu_j = \mu_{A_1^j}(x_1) * \ldots * \mu_{A_m^j}(x_m) \qquad (3)$$

The algebraic product of the membership functions of each premise axis is chosen as the t-norm operator. For each input membership term the below equation is considered

$$\mu_i^{max} = \max\{\mu_{A_i,p_i}(x_i), \quad p_i = 1,\ldots,P_i\}$$
$$i = 1,\ldots,m. \qquad (4)$$

The term μ_i^{max} represents the max value of the membership belonging to the term set p for x_i.

The structure of proposed FNN comprises of three major modules[18]:

- Premise Part: It calculates not only the rule coordinates through the MFs, but also firing strengths for each rule. The structure of the fuzzy inference system is determined by adjusting this part. Both structure learning and parameter influence premise part.
- Consequent Part: It deals with consequent functions f_j as in equation (1) which is linear polynomials of input vector components.
- Defuzzification Part: It involves the defuzzification process. This is performed by combining the outputs of the premise and consequent part and provides the final output of the fuzzy system. The Weighted-Average scheme has been used to produce a fuzzy output $\hat{y}$, for each input vector $\vec{x}$, by

$$\hat{y} = \frac{\sum_{j=1}^{N} \mu_j f_j^k}{\sum_{j=1}^{n} \mu_j} \qquad (5)$$

The training of AFNN is performed by the following three phases:

- Initial membership functions and corresponding rules creation by using a subset of the training data set. This step is conducted off-line, which is referred to as respective premise/consequent parameter setting [18].
- Selection of input patterns by computing node outputs in all network layers and the max-membership terms, μ_i^{max} , for each premise axis. By observing the maximum stimulating level of the term nodes μ_i^{max} it can be verified whether they are greater or smaller than the prescribed lower membership threshold δ. For those where the degree of fulfilment by current MFs is less than a predefined threshold, or in other words the input vector is not adequately spanned by current MFs ($\mu_i^{max} < \delta$), network inserts a new membership function in the respective term set and calculates its parameters (mean and deviation) according to equations (6)(7).

Let $A_{i,new}(m_{i,new}, \sigma_{i,new})$ denote the new MF

$$m_{i,new} = x_i(k+1)$$
$$\sigma_{i,new} = \sigma_{i,nearest} \cdot \frac{\left|m_{i,new} - x^+\right|}{\left|x^+ - m_{i,nearest}\right|} \qquad (6)$$

where

$$x^+ = m_{i,nearest} + h^+.\sigma_{i,nearest} \quad \text{if} \quad m_{i,new} > m_{i,nearest}$$
$$x^+ = m_{i,nearest} - h^+.\sigma_{i,nearest} \quad \text{if} \quad m_{i,new} < m_{i,nearest} \qquad (7)$$
$$h^+ = \sqrt{[2\text{Ln}(\frac{1}{\beta})]}$$

And β exhibits the degree of overlapping between membership functions. In this case when at least one new membership is created, then a new fuzzy rule is created by combining the new memberships and an appropriate set of already existing memberships. The consequent parameters (polynomial weights) of the new rule are initially defined by:

$$w_{0i} = y_i^d(k+1) \qquad i = 1,\ldots,m \qquad (8)$$

Where $y_i^d(k)$ is the desired output for the k'th input training instance. The remaining weight parameter are set to zero($w_{1i} = w_{2i} = \cdots = w_{mi} = 0$).

In the case when $\mu_i^{max} > \delta$ meaning that the input vector is sufficiently covered by the existing fuzzy membership functions, the term set remains unchanged.

In an alternative scenario where the degree of fulfilment by the membership functions is large enough but the respective fuzzy rule is missing, a fuzzy rule is created by proper permutation of term node coordinates.

- Parameter fine tuning using the classic back-propagation algorithm. The back-propagation (BP) method is a gradient based algorithm which is usually used to perform parameter learning of both neural networks and fuzzy neural systems. BP is a simple, well established and easily applicable optimization method [24]; the learning task is

accomplished by minimizing a single objective function, as shown in equation (9).

$$e_k = \frac{1}{2}(\hat{y}(k) - y_d(k))^T(\hat{y}(k) - y_d(k)) \quad \textbf{(9)}$$

As training proceeds, parameter learning is simultaneously conducted to adjust the network parameters. The final updated equations are:

$$\begin{cases} m_{ij}(t+1) = m_{ij}(t) - \alpha_m \dfrac{\partial e_k}{\partial m_{ij}}(t) + \eta_m \dfrac{\partial e_k}{\partial m_{ij}}(t)(t-1) \\ \dfrac{\partial e_k}{\partial m_{ij}} = \sum_p [\hat{y}(k) - y^d(k)] \times (f_p - \hat{y}(k)) \times \dfrac{\mu_p}{\sum_{\lambda=1}^{N} \mu_\lambda} \times \dfrac{(x_i(k) - m_{ij})}{\sigma_{ij}^2} \end{cases} \quad \textbf{(10)}$$

$$\begin{cases} \sigma_{ij}(t+1) = \sigma_{ij}(t) - \alpha_\sigma \dfrac{\partial e_k}{\partial \sigma_{ij}}(t) \\ \dfrac{\partial e_k}{\partial \sigma_{ij}} = \sum_p [\hat{y}(k) - y^d(k)] \times (f_p - \hat{y}(k)) \times \dfrac{\mu_p}{\sum_{\lambda=1}^{N} \mu_\lambda} \times \dfrac{(x_i(k) - m_{ij})}{\sigma_{ij}^3} \end{cases} \quad \textbf{(11)}$$

$$\begin{cases} w_{ij}(t+1) = w_{ij}(t) - \alpha_w \dfrac{\partial e_k}{\partial w_{ij}} \\ \dfrac{\partial e_k}{\partial w_{ij}} = [\hat{y}(k) - y^d(k)] \times \dfrac{\mu_j}{\sum_{\lambda=1}^{N} \mu_\lambda} . x_i(k) \\ \dfrac{\partial e_k}{\partial w_{0j}} = [\hat{y}(k) - y^d(k)] \times \dfrac{\mu_j}{\sum_{\lambda=1}^{N} \mu_\lambda} \end{cases} \quad \textbf{(12)}$$

α_m, α_σ and α_w representing the mean, deviation and polynomial-weights learning rates respectively and η_m is momentum which is used for updating means. Training is carried out on-line on the basis of real-time data.

III. FUNGUS GROWTH MODELING

This section illustrates the ability of the AFNN to perform combined structure and parameter learning of a non-linear three input and one output system. The three dimension input data is normalised in such a way that the maximum of each input column is equal to 0.9 and minimum equal to 0.1. However the error evaluation is based on de-normalised data. The fixed parameters of the system are defined in the form of a vector $\psi = [\alpha_m, \alpha_\sigma, \alpha_w, \eta_m, \beta, \delta] = [0.001, 0.0001, 0.01, 0.05, 0.5, 0.6]$. The rule base is automatically generated along a model formed by the FNN input-output components.

As can be seen from Fig. 3, the trained FN approximates the desired function quite accurately such that the observed output almost completely overlaps the desired output.

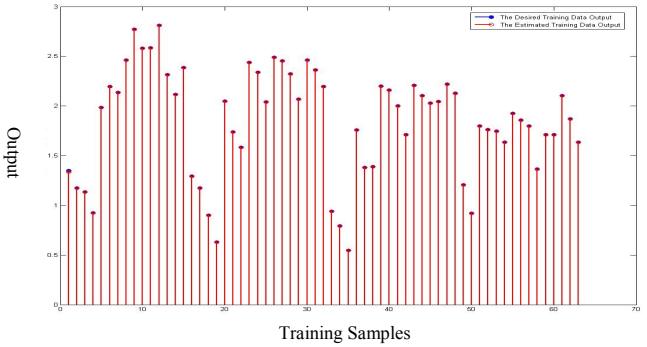

Fig.3– Training samples simulation results

The training procedure starts with just one single rule and only one membership for each premise axis as depicted in Fig.4 (a). The total MFs inserted as a result of structure learning is shown in Fig.4 (b) and the final position of the MFs after imposing parameter learning and fine tunings illustrated in Fig.4(c), five MFs for temperature axis, four MFs for water – activity and four dedicated to pH axis. The exact centre positions and also deviation with and without parameter learning are mentioned in Table 2.

Fig.5a, 5b and 5c depict each input pair and output of AFNN, for this particular training case we attempted 63 pairs, the error measure is decreasing until a MSE of 3.1623e-005 is finally achieved. At the end, the structure finalised with 16 epochs and 50 fuzzy rules. Observing the curves (Fig.5) reveals that the maximum growth occurs in case of lower Water-Activity, mid-high temperatures and in almost any pH depends on temperature and water activity, which of course shows that pH does not play a significant role in growth comparing to other two.

The error criteria are as in table 1, the detailed definition of each error criterion can be found in appendix.

TABLE 1 – Training and Testing error coefficients for AFNN

	Training	**Testing**
Mean Square Error	3.1623e-005	0.0818
Root Mean Square Error	0.0056	0.2860
Mean Absolute Error	0.0041	0.2215
Mean Absolute Relative Error	0.3095	3.5379
Coefficient of Determination	0.9999	0.8984

TABLE 2-Normalised means and deviations of all Membership Functions with and without parameter learning

		Without Parameter Learning					With Parameter Learning				
Temp. Axis	Mean	0.1	0.3	0.5	0.7	0.9	0.0531	0.2530	0.4530	0.6529	0.8529
	Dev.	0.08	0.08	0.08	0.08	0.08	0.0397	0.0396	0.0395	0.0395	0.0394
W.A. Axis	Mean	0.1	0.45	0.77	0.9	-----	0.0585	0.4089	0.7307	0.8558	---------
	Dev.	0.07	0.07	0.07	0.07	-----	0.0323	0.0296	0.0269	0.0243	---------
pH Axis	Mean	0.1	0.36	0.63	0.9	-----	0.1152	0.3818	0.6484	0.9150	---------
	Dev.	0.06	0.06	0.06	0.06	-----	0.0808	0.0807	0.0806	0.0805	---------

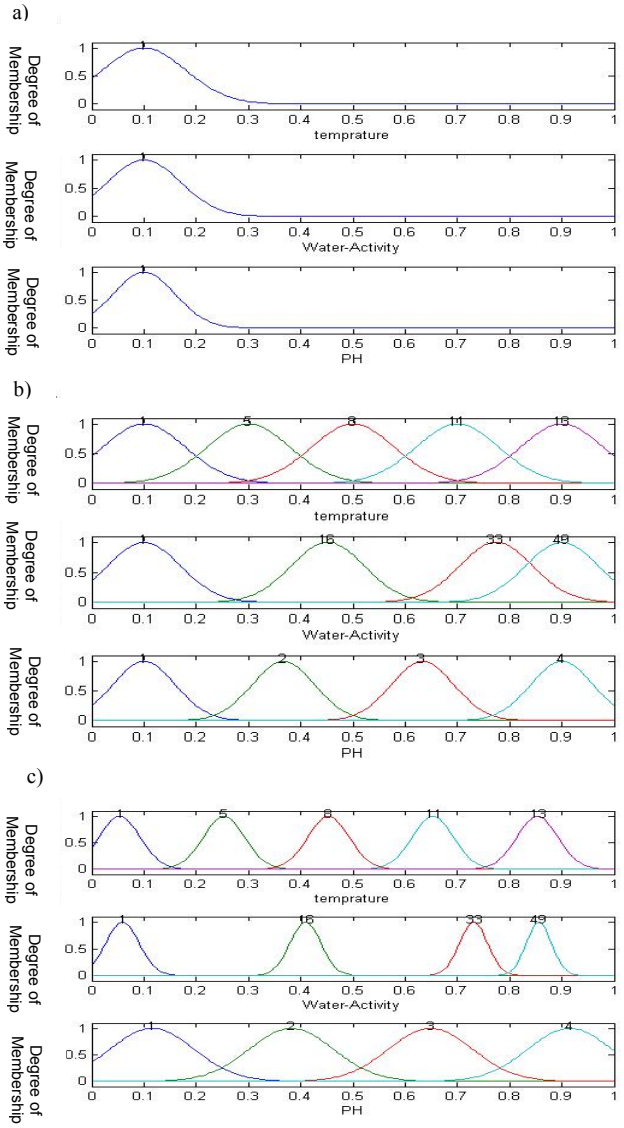

Fig.4– (a) Initial membership functions for each normalized input.(b)-Memberships after structure learning process for each normalized input (c)- Final membership function together with parameter learning which caused some changes in their means and deviations**.**

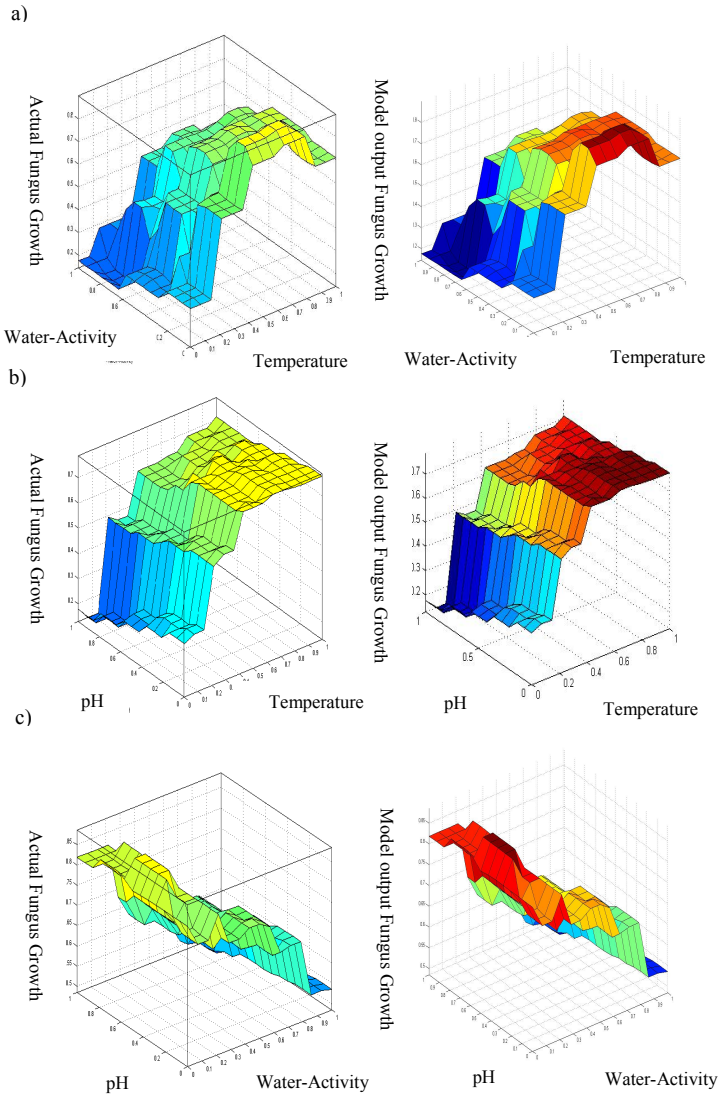

Fig.5– a) Temperature and Water Activity vs. estimated output curve for normalized training data and the actual output curve b) Predicted Temperature and PH output surface and actual output surface c) PH and Water Activity predicted and desired output surface.

IV. FUNGUS GROWTH MODELING BY MLP

A three-input one-output Multi Layer Perceptron (MLP) has been designed to estimate the output. MLP has a simple structure but as it shown in Fig.6 it requires more than 17000 epochs to achieve the desired output.

The suggested network contains a single hidden layer (3 layer MLP) comprises of 30 neurons with sigmoid as an activation function and Back propagation (BP) training algorithm [17][26], The learning rate used here is $\alpha = 0.15$ and momentum is $\eta = 0.45$. The number of training epochs and the squared error is depicted in Fig.6

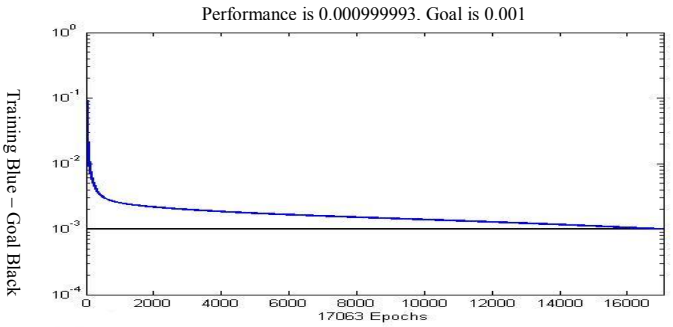

Fig.6– Number of epochs and the related sum square error

The training results demonstrated in Fig.7 which shows the difference between the desired output and observed output.

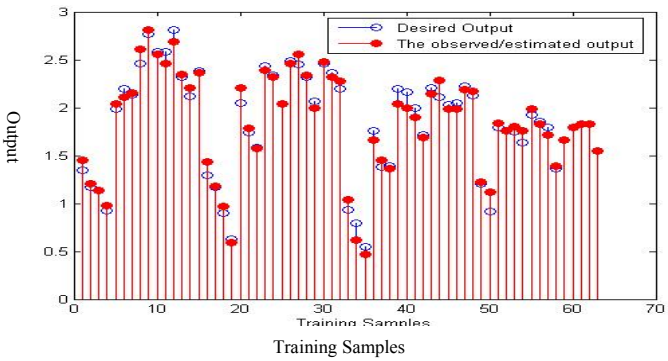

Fig.7 -MLP training results for each input pattern

The final statistical error coefficients are logged in Table 3

TABLE 3 – Training and testing data statistical error coefficients for MLP

	Training	**Testing**
Mean Square Error	0.0080	0.1071
Root Mean Square Error	0.0895	0.3273
Mean Absolute Error	0.0753	0.3133
Mean Absolute Relative Error	4.4037	17.9191
Coefficient of Determination R^2	0.982	0.4283

V. FUNGUS GROWTH MODELING BY OLS-RBF

In this section, we use OLS-RBF to model the fangus growth in accordance to three inputs (temperature, water activity and pH). OLS-RBF networks approximate an unknown function by locally constructing receptive fields around a set of centres, while these centres chosen by linear regression or rank revealing techniques called Orthogonal Least Square (OLS) algorithm [16]. The RBF network and the OLS algorithm have the following fixed constants: The RBF is a Gaussian with width $\sigma = 0.8$ and desired Error Reduction Ratio is $\rho = 0.001$.

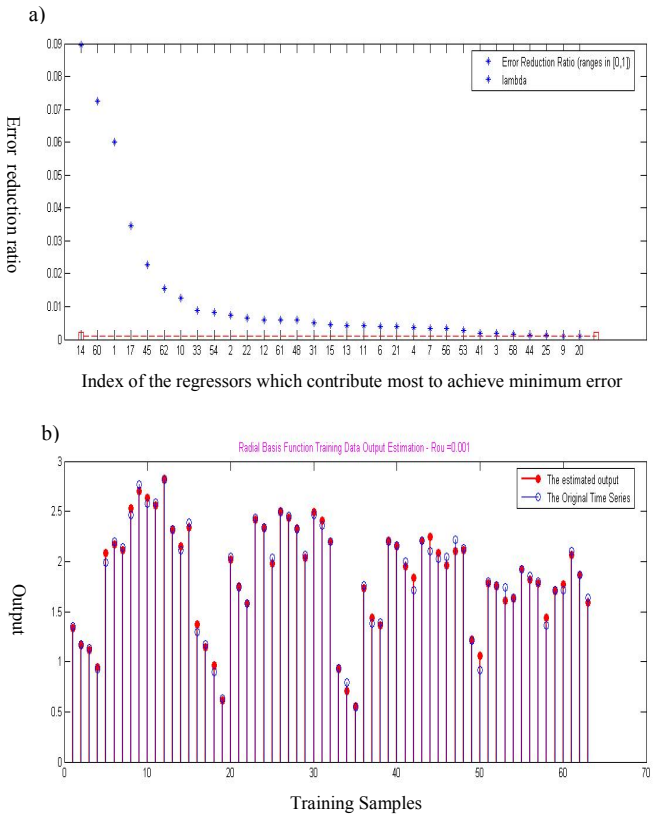

Fig. 8- a) OLS-RBF regressors index and the and their contribution to error reduction
b) OLS-RBF training results for each input pattern

The results are plotted in Fig.8(a), the asterisks represent the regressor indices versus the error ratio reduction, in this network 31 regressors has been chosen in the sequence of {14,60,1,17,45,62,10,33,54,2,22,12,61,48,31,15,13,11,6,21,4,7,56 53,41,3,58,44,25,9,20}. This sequence shows the data indices which are chosen to act as regressors, obviously selecting more than 31 regressors cannot further improve the model quality significantly. The accuracy of the function approximation shown in Fig.8 (b). It is visible that the greater deviation σ is chosen for the RBFs, the smaller number of regressors is used, however this reduction comes of the expense of increasing MSE. Table 4 summarizes the estimated error values.

TABLE 4 - Training and testing data statistical error coefficients for OLS-RBF

	Training	**Testing**
Mean Square Error	0.0027	0.0844
Root Mean Square Error	0.0516	0.2906
Mean Absolute Error	0.0442	0.2547
Mean Absolute Relative Error	2.7199	18.8620
Coefficient of Determination R^2	0.9911	0.8999

VI. CONCLUSION

This paper outlines an adaptive scheme for implementing neuro-fuzzy algorithms and the comparison of identification of the effect of temperature, water activity, and pH on the maximum specific growth rate of fungus by three techniques: Adaptive FNN, MLP and OLS-RBF.

According to Table 1,3 and 4, the simulation results have shown that AFNN method generates superior results and outperforms the other two. The FNN structure is effectively divided into three major parts, namely the premise part, the consequent part and the defuzzification part. The training algorithm used has a two-stage procedure; for each training data, structure learning is first imposed to construct the correct network skeleton. Then, parameter learning is performed to adjust the network parameters by means of the back-propagation algorithm in order to optimize a predefined error tolerance. This method identifies quickly and easily the significant input variables. Because the initial structure and weights of the neural network are set properly, FNN exhibits faster convergence, it can be seen from the number of epochs the AFNN is a much faster algorithm than Back Propagated MLP and RBF. RMSE values of the AFNN performed well for the training and reasonably good for the testing data set based on both graphical plots and statistical indices. In summary, the applied FNN training algorithm, as expected, well fitted to any microbiological system. It serves as a better alternative to microbiological processes predictive modelling scheme based on some of its interesting properties such as: containing only necessary number of rules, fast convergence, simple structure, less training time and of course adjustable performance.

VII. REFERENCES

[1] Ross, T., McMeekin, T.A., Review Paper: "*Predictive microbiology"*. Int. J. Food Microbiology.**,** 23**,** 241–264. (1994).

[2]Gibson, A.M., Hocking, A.D., "*Advances in the predictive modeling of fungal growth in food"*. Trends Food Sci. Techno l., 8, 353-358 (1997).

[3] Murphy, P.A., Hendrich, S., Landgren, C., Bryant, C.M., "*Food mycotoxins":* an update. J. Food Sci., 71, 51-65 (2006).

[4] Dantigny, P., Guilmart, A., Bensoussan, M., "*Basis of predictive mycology*". Int. J. Food Microbiol., 100, 187-196. (2005).

[5] Parra, R., Magan, N., "*Modeling the effect of temperature and water activity on growth of Aspergillus Niger strains and applications for food spoilage moulds"*. J. Appl. Microbiol., 97, 429-438. (2004).

[6] Cuppers, H.G.A.M., Oomes, S., Brul, S., "*A model for the combined effects of temperature and salt concentration on growth rate of food spoilage molds"*. Appl. Environ. Microbiol., 63, 3764-3769 (1997).

[7] Hajmeer, M.N., Basheer, I.A., Najjar, Y.M.," *Computational neural networks for predictive microbiology II. Application to microbial growth"*. Int. J. Food Microbiol*.,* 34**,** 51-66 (1997).

[8] Basheer, I.A. and Hajmeer, M., (2000). *Artificial neural networks: fundamentals, computing, design, and application.* J. Microbiol. Meth., 43, 3-31.

[9] Garcia-Gimeno, R.M., Hervás-Martínez, C., Barco-Alcalà, E., Zurera-Cosano, G., Sanz-Tapia, E., An Artificial Neural Network approach to Escherichia coli *O157:H7* growth estimation. J. Food Sci., 68, 639-645. (2003).

[10] Rosso, L., Lobry, J.R., Bajard, S. and Flandrois, J.P. Convenient model to describe the combined effects of temperature and pH on microbial growth. Applied and Environmental Microbiology, 61, 610–616 (1995).

[11] V. Kodogiannis, M. Boulougoura, E. Wadge, J.N. Lygouras, The usage of soft-computing methodologies in interpreting capsule endoscopy, *Engineering Applications in Artificial Intelligence*, 20, , 539-553 (2007)

[12] J.-S. R. Jang, "*ANFIS: Adaptive-Network-based Fuzzy Inference Systems"*, IEEE Trans. on Systems, Man, and Cybernetics, 23, 665-685 (1993)

[13]C. H. Chen, "*Fuzzy logic and neural network handbook"* (McGraw-Hill, 1996).

[14] C.T. Lin and C. S. G. Lee, "*Neural fuzzy systems: a neuro-fuzzy synergism to intelligent systems"* - Prentice Hall (1996).

[15] E.Z. Panagou, V.Kodogiannis, G.J-E. Nychas, "*Modeling fungal growth using radial basis function neural networks: The case of the ascomycetous fungus Monascus rubber van Tieghem"*, Int. Journal of Food Microbiology, Elsevier, (2007), Vol. 117, pp. 276-286

[16] S Chen et all. "*Orthogonal Least Square algorithm for training multi-output radial basis function Networks".* IEEE Transaction (1992)

[17] S.Seung "*Multilayer perceptron and backpropagation"* (2002) 9.641 Lecture 4

[18] J.Theocharis G. Vachtsevanos "*Adaptive Fuzzy Neural network identifier of Discrete- Time nonlinear Dynamic Systems*" Journal of Intelligent and Robotic Systems (1996)

[19] A Khotanzad, C Chung " *Application of MLP to vision problem*" Naural Computing and application (1998)

[20] Heng H.Fun, M.T. Hagan "*Recursive Orthogonal Least squares learning with automatic weight selection for Gaussian Neural Network"*

[21] M.L.Kothari, S. Madnani, R Segal " *Orthogonal Least squares learning algorithm based radial basis function network adaptive power stability*" IEEE(1997)

[22] O. Nelles, "*Non-linear System Identification*" Springer, published (2001)

[23] H. Iyatomi, M. Hagiwara " *Adaptive fuzzy inference neural network"* The Journal of Pattern recognition (2003)

[24] P. Mastorocostas, J Theocharis "*FUNCOM: A constrained learning algorithm for fuzzy neural network*" Fuzzy Sets & Systems (1997)

[25] Y. Lin, A. Cunningham "*A New Approach to Fuzzy-Neural System Modelling"* IEEE Transaction on fuzzy systems, vol. 3, No. (1995)

[26] L. Faussett "*Fundamental of Neural Networks* Prentice Hall Int'l (1994)

Appendix

$$\text{Mean Square Error(MSE)} = \frac{\sum_{i=1}^{N}(\text{Output}_{\text{estimated}} - \text{Output}_{\text{desired}})^2}{N}$$

$$\text{Root Mean Square Error(RMSE)} = \sqrt{\frac{\sum_{i=1}^{N}(\text{Output}_{\text{estimated}} - \text{Output}_{\text{desired}})^2}{N}}$$

$$\text{Mean Absolute Error(MAE)} = \frac{\sum_{i=1}^{N}|\text{Output}_{\text{estimated}} - \text{Output}_{\text{desired}}|}{N}$$

$$\text{Mean Absolute Relative Error(MARE)} = \frac{100}{N} \times \sum_{i=1}^{N} \frac{|\text{Output}_{\text{estimated}} - \text{Output}_{\text{desired}}|}{\text{Output}_{\text{desired}}}$$

$$\text{Coefficient of Determination}(R^2) = 1 - \frac{\sum(\text{Output}_{\text{estimated}} - \text{Output}_{\text{desired}})^2}{\sum(\text{Output}_{\text{desired}} - \bar{\text{O}}\text{utput}_{\text{desired}})^2}$$

Where $\bar{\text{O}}\text{utput}_{\text{desired}}$ is the average of desired outputs.