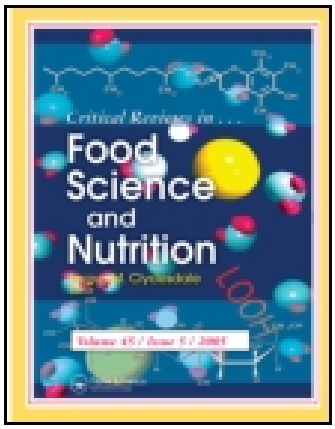

Critical Reviews in Food Science and Nutrition
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/bfsn20

Application of Artificial Neural Networks (ANNs) in Wine Technology

Halil Baykal [a] & Hatice Kalkan Yildirim [a]
[a] Ege University, Food Engineering Department , Bornova , İzmir , Turkey
Accepted author version posted online: 17 Oct 2011.Published online: 07 Feb 2013.

To cite this article: Halil Baykal & Hatice Kalkan Yildirim (2013) Application of Artificial Neural Networks (ANNs) in Wine Technology, Critical Reviews in Food Science and Nutrition, 53:5, 415-421, DOI: 10.1080/10408398.2010.540359

To link to this article: http://dx.doi.org/10.1080/10408398.2010.540359

Critical Reviews in Food Science and Nutrition, 53:415–421 (2013)

ISSN: 1040-8398 / 1549-7852 online
DOI: 10.1080/10408398.2010.540359

Taylor & Francis
Taylor & Francis Group

Application of Artificial Neural Networks (ANNs) in Wine Technology

HALIL BAYKAL and HATICE KALKAN YILDIRIM
Ege University, Food Engineering Department, Bornova, İzmir, Turkey

In recent years, neural networks have turned out as a powerful method for numerous practical applications in a wide variety of disciplines. In more practical terms neural networks are one of nonlinear statistical data modeling tools. They can be used to model complex relationships between inputs and outputs or to find patterns in data. In food technology artificial neural networks (ANNs) are useful for food safety and quality analyses, predicting chemical, functional and sensory properties of various food products during processing and distribution. In wine technology, ANNs have been used for classification and for predicting wine process conditions. This review discusses the basic ANNs technology and its possible applications in wine technology.

Keywords Wine process, optimization, artificial neural network

THE BASIC PRINCIPLES OF ARTIFICIAL NEURAL NETWORK

An artificial neural network (ANN) is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information. Neural networks have been an explosion of interest over the last few years and are being successfully applied across an extraordinary range of problem domains in areas as diverse as finance, medicine, engineering, geology and physics (Irudayaraj, 2007). Indeed, anywhere that there are problems of prediction, classification or control, neural networks are being introduced. Neural networks are increasingly used as effective for general purpose, nonlinear regression tools, and for developing models governed by complex relationships. Neural networks are computer systems that learn the events by using examples of real brain function's products and determine how to react to events from environment. ANNs are used successfully at learning, relating, classification, generalization, determination specialty and optimization like human brain functions. They make own experiences with information that they get from examples and then make similar decisions for similar subjects (Cheung and Cannons, 2002; Smith, 2006).

Neural network simulations appear to be a recent development (Stergiou and Siganos, 2007). However, this field was established before the advent of computers and has survived at least one major setback and several eras. In 1943, the first step toward ANNs: Warren McCulloch and Walter Pitts modeled a simple neural network with electrical circuits. In the years following the Dartmouth Project, John von Neumann suggested imitating simple neuron functions by using telegraph relays or vacuum tubes. Also, Frank Rosenblatt, a neurobiologist of Cornell, began work on the perceptron. A single-layer perceptron was found to be useful in classifying a continuous-valued set of inputs into one of two classes. In 1959 Bernard Widrow and Marcian Hoff of Stanford developed models they called Adaline and Madaline. These models were named for their use of Multiple ADAptive LINear Elements. MADALINE was the first neural network to be applied to a real world problem. It is an adaptive filter, which eliminates echoes on phone lines. This neural network is still in commercial use. Minsky and Pappert realized insufficiency of perceptron and recommended a multilayer model. But, they did not recommend a method on how to change weights of hidden layers. Rummelhart and his friends solved this problem with a back propagation algorithm. Several events caused a renewed interest. John Hopfield of Caltech showed how such networks could work and what they could do with clarity and mathematical analysis.

The brain is principally composed of a very large number (circa 10,000,000,000) of *neurons*, massively interconnected. Each neuron is a specialized cell, which can propagate an electrochemical signal. A typical biological neuron is made up of soma or cell body, dendrites and axon (Fig. 1). In the human brain a typical neuron collects signals from others

Address correspondence to Hatice Kalkan Yildirim, Ege University, Food Engineering Department, Bornova, İzmir, Turkey. E-mail: hatice.kalkan@ege.edu.tr

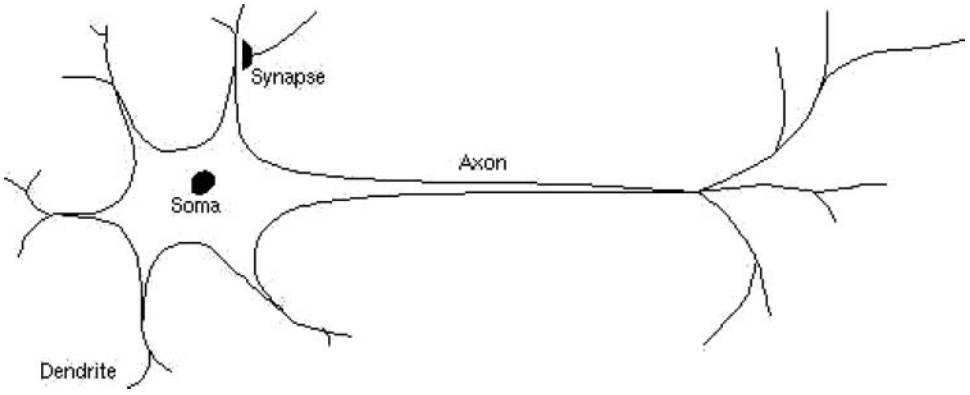

Figure 1 Biological neural network.

through a host of fine structures called *dendrites*. Each neuron may be connected to hundreds of surrounding neurons through a network of dendrites. The *axon*, which is a long, thin tube starting at the base of soma, splits into multiple endbulbs called boutons that almost touch the dendrites of other neurons. The neuron sends out spikes of electrical activity through axon. The axons of one cell connect to the dendrites of another via a synapse. At the end of each branch, a structure called a *synapse* converts the activity from the axon into electrical effects that inhibit or excite activity from the axon into electrical effects that inhibit or excite activity in the connected neurons. When a neuron is activated, it *fires* an electrochemical signal along the axon. This signal crosses the synapses to other neurons, which may in turn fire. A neuron fires only if the total signal received at the cell body from the dendrites exceeds a certain level. ANNs represent extremely simplified formal models of biological neurons and their interconnections without making any attempt to model the biological system itself (Huang et al. 2007).

The first mathematical model of an artificial neuron was originally proposed by McCullock and Pitts. An artificial neuron is an elementary processing unit with several inputs and a single output (Fig. 2). Each input comes via a connection that has a strength (or *weight*); these weights correspond to synaptic efficacy in a biological neuron. For example, a number of inputs $X_1, X_2, \ldots, X_n$ associated with respective weights $W_{j.1}, W_{j.2}, \ldots, W_{j.n}$ form a combined input Net_j to the jth neuron, which is expressed as the weighted sum of the inputs.

$$\text{Net}_j = \sum W_{j.i} X_i$$

Sometimes a bias is added to the net input:

$$\text{Net}_j = \sum W_{j.i} X_i + \text{bias}_j$$

The weighted sum of the inputs is formed and the threshold subtracted to compose the *activation* of the neuron. The activation signal is passed through an activation function (also known as a transfer function) to produce the output of the neuron.

$$Y_j = f(\text{Net}_j) = f\left(\sum W_{j.i} X_i \right). \in$$

The output of a neuron is determined by the nature of its activation function. The most frequently used activation functions are the identity, the linear threshold, the sigmoid, and the hyperbolic tangent. The step function is more suitable as a class identifier because its fixed outputs are independent of the magnitude of net input. The sigmoid represents a convenient form and is most frequently used:

$$Y_j = 1/(1 + \exp[(-\text{Net}_j + \beta)/\theta]),$$

where β is the bias or weight of an imaginary neuron that is always firing. A positive value of β shifts the activation function to the left along the horizontal axis and θ modifies the shape of the activation function. The sigmoid and hyperbolic tangent function approach the limiting minimum and maximum values asymptotically. They are frequently used in forms that transform the net inputs ranging between $[-\infty, +\infty]$ to real numbers between [0,1] and [–1,+1], respectively. Also, their continuously differentiable nature meets the requirement of learning algorithms that follow the gradient descent method.

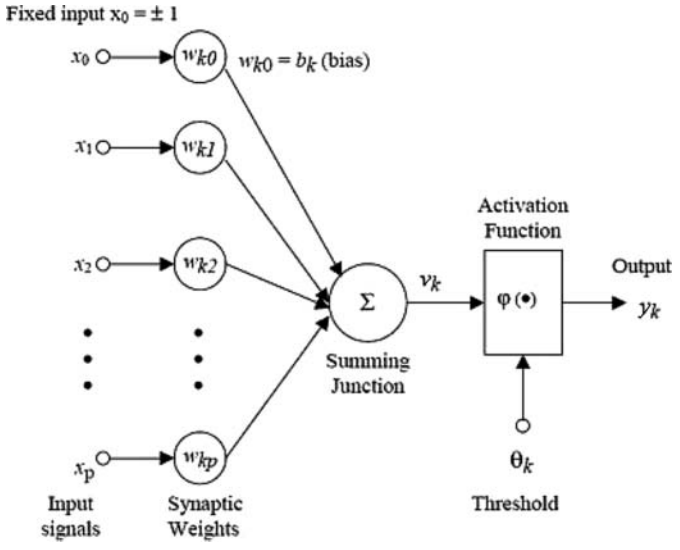

Figure 2 Schematic presentation of an artificial neuron.

The various architectures of ANNs are patterned after our limited understanding of biological neural networks. The nature of nodes and the structure of the interconnections define the ANN architecture. They may have a feed-forward or feedback architecture, or a combination of both. Feed-forward ANNs allow signals to travel one way only: from input to output. There is no feedback (loops); the output of any layer does not affect that same layer. Feed-forward ANNs tend to be straightforward networks that associate inputs with outputs. They are extensively used in pattern recognition. A typical feed-forward network has neurons arranged in a distinct layered topology. The input layer is not really neural at all. These units simply serve to introduce the values of the input variables. The hidden and output layer neurons are each connected to all of the units in the preceding layer. Again, it is possible to define networks that are partially connected to only some units in the preceding layer; however, for most applications fully connected networks are better. Networks with feedback architecture have feedback loops in their interunit connection path. Feedback networks are very powerful and can get extremely complicated. Feedback networks are dynamic; their "state" is changing continuously until they reach an equilibrium point. They remain at the equilibrium point until the input changes and a new equilibrium needs to be found. Feedback architectures are also referred to as interactive or recurrent, although the latter term is often used to denote feedback connections in single-layer organizations.

The commonest type of ANN consists of three groups, or layers, of units: a layer of "input" units is connected to a layer of "hidden" units, which is connected to a layer of "output" units (Fig. 3) (Öztemel, 2003; Uğur, 2006).

- The activity of the input units represents the raw information that is fed into the network.
- The activity of each hidden unit is determined by the activities of the input units and the weights on the connections between the input and the hidden units.

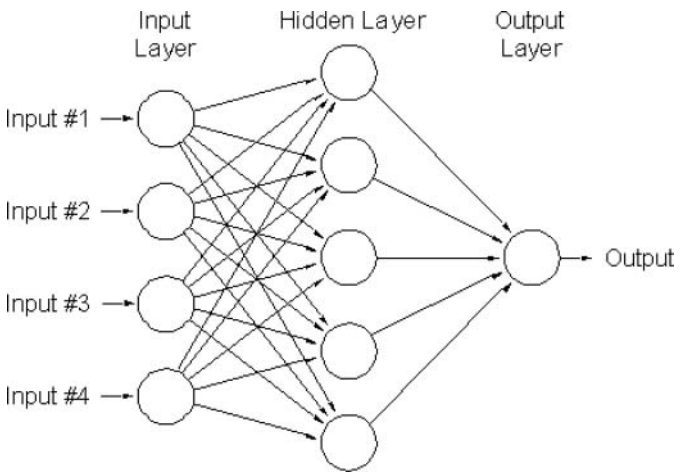

Figure 3 Schemsatic presentation of artificial neural network.

- The behavior of the output units depends on the activity of the hidden units and the weights between the hidden and output units.

The network receives the training data at the nodes in the input layer. The number of the nodes in the input layer depends on the number of variables or categories in the input data. The hidden layers are between the input and output layers. The number of hidden layers and the number of neurons in a layer can only be determined through experimentation.

The best-known example of a neural network training algorithm is back propagation. Modern second-order algorithms such as conjugate gradient descent and *Levenberg-Marquardt* are substantially faster (e.g., an order of magnitude faster) for many problems, but *back propagation* still has advantages in some circumstances and is the easiest algorithm to understand.

The learning by the brain is associated with the synaptic efficiency or modification of the strength, or weight of the interconnections. In ANNs, the processing element or neurons computes the outputs by applying the transfer function to the weighted sum of inputs. Learning implies the adjustment of weights or strength of the connections to match the network output with the actual result for a given training set of data. The neural network learns by changing the weighted input through modification of the weights according to a learning law without being programmed.

All learning methods used for adaptive neural networks can be classified into two major categories:

- Supervised learning which incorporates an external teacher, so that each output unit is told what its desired response to input signals ought to be. During the learning process global information may be required. Paradigms of supervised learning include error-correction learning, reinforcement learning, and stochastic learning. An important issue concerning supervised learning is the problem of error convergence, that is, the minimization of error between the desired and computed unit values. The aim is to determine a set of weights, which minimizes the error. One well-known method, which is common to many learning paradigms is the least mean square (LMS) convergence.
- Unsupervised learning uses no external teacher and is based upon only local information. It is also referred to as self-organization, in the sense that it self-organizes data presented to the network and detects their emergent collective properties. Paradigms of unsupervised learning are Hebbian learning and competitive learning.

The Delta rule was developed by Widrow and Hoff for the classification of linearly independent training patterns. If nonlinear relationships exist between the inputs and outputs, a multilayer perceptron model is used for mapping the nonlinear functions. A multilayer feed-forward neural network consists of nodes in the input layer, hidden layer and output layer. The

input layer simply directs the inputs to the hidden layer. The processing only takes place in the hidden and output layers. The position of a node in input, hidden, output layers is denoted by the subscripts i, j, and k, respectively. Thus, x_i is the value of the ith node in the input layer. ($1 < i < N$), y_j is the value of the jth node in the hidden layer ($1 < j < M$), and y_k is the value of the node in the output layer ($1 < k < K$). Similarly, $w_{j,i}$ represents the weight associated with the connection from the ith input node the jth node in the next layer. The length of input vector equals the number of nodes in the input layer. Any input vector is modified by the weight vector to produce a weighted sum, which is transformed by an activation function into an output vector.

The net input to the jth node in the hidden layer including the threshold, Net_j, is transformed by the sigmoid function to produce the output.

$$y_j(1 < j < M) = f(\text{Net}_j) = f\left(\sum_{i=1}^{N} W_{j.i} X_i\right)$$

The net input to the kth node of the output layer including the threshold, Net_k, after transformation becomes:

$$y_k(1 < k < M) = f(\text{Net}_k) = f\left(\sum_{i=1}^{M} W_{k.j} Y_j\right)$$

The mean squared error between the computed and target outputs is minimized by the generalization of the Delta rule.

$$E(p) = 1/2 \sum_{k=1}^{K} (Y_k - 0_k)2$$

The weight change at the outer layer is given by

$$\Delta w_{k.j} = -\eta = \frac{\alpha E(p)}{\alpha w_{k.j}} = \eta \delta_{kyj}$$

where

$$\delta_k = (y_k - 0_k) f'(\text{Net}_j) = (y_k - 0_k) f(\text{Net}_k)[1 - f(\text{Net}_k)].$$

The weight change for the inner layer is given by

$$\Delta w_{j.i} = -\eta \frac{\alpha \text{E}(p)}{\alpha w_{j.i}} = \eta \mu_j y_i,$$

where

$$\mu_j = \left(\sum \delta_k w_{kj}\right) f'(\text{Net}_j) = \left(\sum \delta_k w_{kj}\right) f(\text{Net}_j)[1 - f(\text{Net}_j)].$$

The new weights can then be calculated by

$$w_{kj}(\text{new}) = w_{kj}(\text{old}) + \Delta w_{kj} \quad (\text{for outer layer}),$$

$$w_{j,i}(\text{new}) = w_{j,i}(\text{old}) + \Delta w_{j,i} \quad (\text{for inner layer}).$$

A neural network is characterized by its architecture, activation function for the node output and learning algorithm for determining the weights of the connections. Radial basis functions (RBFs) are powerful techniques for interpolation in multidimensional space. An RBF is a function, which has built into a distance criterion with respect to a center. RBFs have been applied in the area of neural networks where they may be used as a replacement for the sigmoidal hidden layer transfer characteristic in multilayer perceptrons. The primary difference between back-propagation networks and RBF networks is in the nature of hidden layer. In RBF networks, the nodes in the hidden layer have an RBF or a statistical transformation based on the Gaussian activation function with a center corresponding to a maximal output for any given input value. Learning Vector Quantization (LVQ) networks provide discrete outputs, and are used mainly for classification problems based on competitive unsupervised learning. Their performance is generally better than the multilayer feedforward or RBF networks for certain classification problems. The self-organizing map (SOM) invented by Teuvo Kohonen performs a form of unsupervised learning. A set of artificial neurons learn to map points in an input space to coordinates in an output space. The input space can have different dimensions and topology from the output space, and the SOM will attempt to preserve these. Hopfield networks have been used as associated memory tools for solving optimization problems. They have recurrent architecture in which the output of a processing node is fed back through the connecting weights to all the other nodes except itself. This network guarantees that its dynamics will converge. If the connections are trained using Hebbian learning then the Hopfield network can perform as robust content-addressable memory, resistant to connection alteration.

APPLICATION OF NEURAL NETWORK IN WINE TECHNOLOGY

Wine processing is a complex process. A lot of factors effect on process. So, predicting and optimization of processing conditions are very difficult. In wine technology ANNs can be used for classification of wines according to origin, producers, type, and for optimization of wine blending and electronic nose for sensory analyses.

Kruzlicova et al. (2009) classified Slovak white wines using ANNs and discriminant techniques. They used a multilayer perceptron technique using quick propagation and quasi-Newton propagation algorithms. The developed methodology was applied to classify Slovak white wines of different variety, year of

production and from different producers. They analyzed wine samples by the gas chromatography-mass spectrometry (GC-MS) technique taking into consideration mainly volatile species, which highly influence the wine aroma (terpenes, esters, alcohols) . The analytical data were evaluated by means of the ANN and the classification results were compared with the analysis of variance (ANOVA). They used 36 samples of three varieties, produced by four producers in the course of 3 years characterized by 19 finally accepted variables in the training process of ANN. For the feature selection they determined best seven variables by the backward selection process for all three kinds of wine classification and they used the best seven variables for training the new ANN with the seven input and three output neurons. They obtained percent or near percent classification success for all three classification criteria.

Cichelli et al. (2000) classified Montepulciano d'Abruzzo wines by linear discriminant analysis (LDA) and ANNs. They used data relative to the chemical composition of Montepulciano d'Abruzzo wines for classification of the wines by using ANN and LDA. The data came from the chemical analysis of 116 wine samples produced during 2 years in three zones of Montepulciano d'Abruzzo hypothesized to be from different pedoclimatic zones. Classification of the samples was according to the year of production only, as well as according to the pedoclimatic zone only, as attempted. The results showed that the analysis allows information to be obtained, which is particularly useful for characterizing the wines according to the year of production.

Aires-De Sousa (1996) used artificial back-propagation neural networks (BPNNs) for two different problems of wine classification. In one case, each of five $16 \times 2 \times 2$ BPNNs was trained with a different set of 21 samples to distinguish between young red wines of two Spanish Certified Brands of Origin (Ribera del Duero and Toro) on the basis of 15 anthocyanin contents. The networks were tested with 26 examples not involved in the training process and gave average correct predictions of 88% for Toro wines and 91% for Ribera wines. In another case, five $23 \times 8 \times 8$ BPNNs were applied to classify eight Portuguese varietal wines of Vitis vinifera according to grape variety. Percent composition of 22 free amino acids was available for 42 samples (of 7 different vintages) and was used as input data. Each network was trained with a different set of 26 examples and tested with the other 16, obtaining an average success rate of 73%. The result of this work was: two layer BPNNs could be used to deduce which amino acids are characteristic of each variety of wine . The ANN gave better predictions than LDA.

Gaeta et al. (2008) used neural network for wine identification by using spectrum of wine. They used two different types of spectrum analysis: visible-ultraviolet (VIS-UV) and neighbor infrared (NIR). For NIR spectrum analysis seven frequencies, and for VIS-UV five frequencies were used to classification. The readings of the absorption at these frequencies have to be considered to classify the input pattern . They employed two neural models: the multilayered perceptron (MLP) with algorithm of back propagation and the neural fuzzy model (NF). The net is trained by presenting a set of samples, which are the real values of the frequencies of the spectrum of absorption and the targets corresponding to the correct classification. For both NIR and VIS-UV 100% or near 100% success was obtained at different configuration of ANN models.

Ferrier and Block (2001) developed a blending optimization method that uses ANNs to model the potentially nonlinear response of the blending based on sensory data from the base wines and a limited number of blends. This method has been developed and verified by constructing a series of 24 wines from these base wines. Each wine was profiled by descriptive analysis with a trained panel and the sensory data were modeled with an ANN. After choosing specific target attributes for the final blend, they employed an optimization algorithm to predict the optimal blend for this set of goals. Optimal blends chosen with this methodology had sensory characteristics close to the goal characteristics and to experimental blends with similar composition. All ANNs used in the algorithm were back-propagation networks trained with the Broyden-Fletcher-Goldfarb-Shano (BFGS) learning rule. Each network had an input layer with three inputs corresponding to the fractional composition of the blend, a hidden layer of three nodes using tangent-sigmoid transfer functions and a single linear output layer that gave a prediction of the aroma being examined. For each of the six aroma attributes measured by descriptive analysis, the neural network fit was evaluated by comparing the predictions with the actual mean sensory data. Finally the composition of the blend was calculated for goal wines.

Magarino et al. (2003) classified Spanish denomination of origin (DO) rose wines according to their geographical origin by using classical multivariate analysis techniques such as factor analysis and stepwise LDA and ANNs. Seventy commercial rose wines from four different Spanish DO (Ribera del Duero, Rioja, Valdepenas, and La Mancha) and two successive vintages were studied. They measured 19 different variables in these wines. The stepwise linear discriminant analyses (SLDA) model selected 10 variables obtaining a global percentage of correct classification of 98% and global prediction of 97.3%. The ANN model selected seven variables, five of which were also selected by the SLDA model and it gave a 100% correct classification for training and prediction. So both models can be considered satisfactory and acceptable, the selected variables being useful to classify and differentiate these wines by their origin. Furthermore, the casual index (CI) analysis gave information that can be easily explained from an enological point of view. The effect of each input variable on the wine classification was evaluated in the form of CI, calculated from the trained full and reduced ANN coefficients. Large positive CI means important influence of this variable on the corresponding classification group.

Vlassides et al. (2001) developed an optimization method in which historical process data are used to train an ANN for correlation of processing inputs and outputs. An optimization routine is used in conjunction with the trained neural network to find optimal processing conditions given the desired product characteristics and any constraints on inputs. They have demonstrated

that trained neural networks can be used successfully to predict the yeast fermentation kinetics, as well as chemical and sensory properties of the finished wine, based solely on the properties of the grapes and the intended processing. A hybrid neural network training method, Stop Training with Validation (STV), has been developed to find the most desirable neural network architecture and training level. All neural networks used for these studies were feed-forward networks with one hidden layer trained using a back-propagation algorithm. A sigmoidal transfer function was used for the hidden nodes and linear transfer functions used for the input and output nodes. First, three input parameters are grape maturity level, skin contact time prior to inoculation, and primary fermentation temperature. They predicted relationship between malolactic duration and these parameters. The average root mean square error (RMSE) was found to be 2%. They predicted successfully brix curves for yeast fermentation kinetics; and then, neural networks were trained to predict several chemical characteristics of the final wine as a function of grape maturity level, skin contact time, and fermentation temperature. For these chemical characteristics (final ethanol, final glycerol level, total phenolics, color, total pH), prediction errors ranged from less than 1% for pH to 5.4% for color. Finally, a simple example of optimization based on a neural network trained with historical data.

Ren and Li (2006) built a physics and chemistry information database of wine bodies and a management information system and achieved wine auto-blending through some training to physics chemistry information of wine bodies by exercising back-propagation network. There were three steps in the analyzing process, first with the liquor samples offered by users, they designed the database management system and provided some functions for the information, in order to insure validity of the data and the robustness of the program. After building the winy sample information database, dactylogram atlas of the different liquor samples should be established to figure physics and chemistry information. Finally, in the MATLAB they designed proper neural network algorithm to train the liquor samples data and dredge up the connotative information among them.

Yamazaki and Ludermir (2002) classified vintages of wine by an artificial nose with neural networks. A pattern recognition system of an artificial nose with neural networks, which implement temporal processing, is presented. They compared the results to those obtained by a MLP neural network. The signals generated by an array of six conducting polymer sensors exposed to the odors of three different vintages of the same wine were analyzed. One hidden layer MLP neural network was implemented. The input layer has 6 units, because each pattern is composed of six values. The output layer has 3 units and each one corresponds to one vintage of wine. Hidden and output units implement sigmoidal activation function. They trained five different topologies. To train network, the back-propagation algorithm was used. For the 4-hidden-node topology, standard deviations of errors were high. Using 8 hidden units, all the runs stopped training by achieving the maximum number of epochs. The mean values of the errors are smaller than those obtained by the 4-hidden-node topology. The 12-hidden-node topology presented a small increase in mean and standard deviation values, but the results are still better than those obtained by the topology with 4 intermediate units. The topology with 16 hidden nodes obtained the smallest mean and standard deviation values for validation and test set classification errors. For the topology with 20 hidden units, mean and standard deviation values of validation and test set classification errors were close to those obtained by the 16-hidden-node topology.

Zhu et al. (2004) studied optimization of operating conditions in rice heat blast process for Chinese rice wine production by combinational utilization of neural network and genetic algorithms. They focused on three major performance indexes (starch gelatinization ratio, total fat content, amino nitrogen content in the roasted rice), which largely affect the performance of the rice heat blast process and rice wine quality. The relationship between these performance indexes and the corresponding operation variables were modeled by an ANN via learning sets of experimental data. Based on the ANN models obtained, genetic algorithms were used to optimize the operating conditions of the rice heat blast process. The results showed the power in determination of optimal operating conditions by the combinational utilization of ANN and genetic algorithms. Three standard three-layer feed-forward ANNs were used for modeling of the rice heat blast process. The input layer consists of three neurons and a set of operating variables, the blast temperature, the blast time and the rice soaking time. These were input into each of the corresponding neurons as the input variables. The output layer had only one neuron, where one of the three major performance indexes corresponds to the set of the operating conditions, the starch gelatinization ratio or the total fat content or amino nitrogen content in the roasted rice. This was used as the learning signal for the output unit. The estimating discrepancy in the starch gelatinization ratio model may account for or reflect that a more complicated relationship exists between the starch gelatinization ratio and the operating variables in comparison with those for fat and amino nitrogen content.

Chandra et al. (2007) combined and compared neural networks with decision trees for wine classification. They first trained neural networks and then combined with decision trees in order to extract knowledge learnt in the training process. ANNs are used for the classification of Italian wines obtained from a region, which has three different wine cultivars. Wines are classified according to their respective cultivar using the chemical analysis of the 13 major chemical constituents. They ran 10 trial experiments with randomly selected 80% of the available data for training and the remaining for testing the networks generalization performance. The network topology used was: 13 neurons in the input layer for each chemical constituent, 4 neurons in the hidden layer and 3 neurons in the output layers representing each wine cultivar. The mean and 90% confidence interval for training and generalization classification performance are 99.7% and 98.7%, respectively. Then

they extracted decision trees from trained neural networks using the J48 decision tree algorithm. The extracted rules showed that 3 chemical constituents (flavanoids, proline, and color intensity) play a vital role in the classification of wines according to the relevant cultivator.

CONCLUSIONS

ANNs are capable of learning complex relationships and generalizing solutions from known patterns of input/output data. Thus, ANNs are good for the modeling of complex systems for which exact models or expected performances are not known. Success of neural networks application depends on the size, quality and preprocessing of the training data, type, and structure of the neural network and the learning algorithm for a particular problem. ANN can be a very useful tool for the classification of wines using the chemical, physical and sensory properties. The other application of neural network in wine technology is optimization of blending. Prediction of wine process parameters is difficult yet, because fermentation is a very dynamic process requiring interpreting of many variables at the same time. So, the field of using of ANN in wine production for process optimization needs further researches.

REFERENCES

Aires-De Sousa, J. (1996). Verifiying wine origin: A neural network approach. *Am. J. Enol. Vitic.* **47**(4):410–414.

Chandra, R., Chaudhary, K., and Kumar, A. (2007). The Combination and Comparison of Neural Networks with Decision Trees for Wine Classification. School of Science and Technology, The University of Fiji, Lautoka, Fiji.

Cheung, V. and Cannons, K. (2002). An Introduction to Neural Networks. Electrical & Computer Engineering, University of Manitoba, Manitoba, Canada.

Cichelli, A., Damiani, F., Murmura, F., Simonetti, M. S., Odoardi, M., and Damiani, P. (2000). Classification of montepulciano d'Abruzzo wines by linear discriminant analysis and artificial neural networks. *Am. J. Enol. Vitic*. **51**(2):108–114.

Ferrier, J. and Block, D. E. (2001). Neural network-assisted optimization of wine blending based on sensory analysis. *Am. J. Enol. Vitic.* **52**(4):386–395.

Gaeta, M., Marsella, M., Miranda, S., and Salerno, S. (2008). Using Neural Networks for Wine Identification. IEEE Xplore.: http://xplorebcpaz.ieee.org/Xplore/home.jsp?tag=1

Huang, Y., Kangas, L. J., and Rasco, B. A. (2007). Applications of artificial neural networks (ANNs) in food science. *Crit. Rev. Food Sci.* **47**:113–126.

Irudayaraj, J. (2007). Food Processing Operations Modeling-Design and Analysis. Marcel Dekker, Inc., New York .

Kruzlicova, D., Mocak, J., Balla, B., Petka, J., Farkova, M., and Havel, J. (2009). Classification of Slovak white wines using artificial neural networks and discriminant techniques. *Food Chem.* **112**:1046–1052.

Magarino, S. P., Heras, M. O., Gonzalez-San Jose, M. L., and Boger, Z. (2003). Comparative study of artificial neural network and multivariate methods to classify DO rose wines. *Talanta* **62**:983–990.

Öztemel, E. (2003). Yapay sinir ağları Papatya Yayıncılık, İstanbul, Turkey.

Ren, J. and Li, Z. (2006). A system of wine blending based on neural network. *IFIP* **207**:604–609.

Smith, L. (2006). An introduction to neural network. NSYN Meeting. Edingburgh, Scotland.

Stergiou, C. and Siganos, D. (2007). Neural networks. Available at: http://www.docstoc.com.

Uğur, A. (2006). Yapay sinir ağlari. Lecture notes. Computer Engineering, Ege University, İzmir, Turkey.

Uğur, A. and Yapay Sinir Ağları. (2006). Lecture Notes. Computer Engineering, Ege University, İzmir, Turkey.

Vlassides, S., Ferrier, J. G., and Block, D. E. (2001). Using historical data for bioprocess optimization: Modeling wine characteristics using artificial neural networks and archived process information. *Biotechnol. Bioeng.* **73**: 55–68.

Yamazaki, A. and Ludermir, T. B. (2002). Classification of vintages of wine by an artificial nose with neural networks. *Electronic Lett.* **37**:1466–1467.

Zhu, Y., Zhang, J., Shi, Z., and Mao, Z. (2004). Optimization of operating conditions in rice heat blast process for Chinese rice wine production by combinational utilization of neural network and genetic algorithms. *J. Ens. Brew.* **110**(2):117–123.