Conference on Prognostic Factors and Staging in Cancer Management: Contributions of Artificial Neural Networks and Other Statistical Methods

Supplement to ***Cancer***

Artificial Neural Networks

Opening the Black Box

Judith E. Dayhoff, Ph.D.[1]
James M. DeLeo[2]

[1] Complexity Research Solutions, Inc., Silver Spring, Maryland.

[2] Clinical Center, National Institutes of Health, Bethesda, Maryland.

Artificial neural networks now are used in many fields. They have become well established as viable, multipurpose, robust computational methodologies with solid theoretic support and with strong potential to be effective in any discipline, especially medicine. For example, neural networks can extract new medical information from raw data, build computer models that are useful for medical decision-making, and aid in the distribution of medical expertise. Because many important neural network applications currently are emerging, the authors have prepared this article to bring a clearer understanding of these biologically inspired computing paradigms to anyone interested in exploring their use in medicine. They discuss the historical development of neural networks and provide the basic operational mathematics for the popular multilayered perceptron. The authors also describe good training, validation, and testing techniques, and discuss measurements of performance and reliability, including the use of bootstrap methods to obtain confidence intervals. Because it is possible to predict outcomes for individual patients with a neural network, the authors discuss the paradigm shift that is taking place from previous "bin-model" approaches, in which patient outcome and management is assumed from the statistical groups in which the patient fits. The authors explain that with neural networks it is possible to mediate predictions for individual patients with prevalence and misclassification cost considerations using receiver operating characteristic methodology. The authors illustrate their findings with examples that include prostate carcinoma detection, coronary heart disease risk prediction, and medication dosing. The authors identify and discuss obstacles to success, including the need for expanded databases and the need to establish multidisciplinary teams. The authors believe that these obstacles can be overcome and that neural networks have a very important role in future medical decision support and the patient management systems employed in routine medical practice. ***Cancer*** **2001;91:1615–35.** *© 2001 American Cancer Society.*

KEYWORDS: artificial neural networks, computational methodology, medicine, paradigm.

Artificial neural networks are computational methodologies that perform multifactorial analyses. Inspired by networks of biological neurons, artificial neural network models contain layers of simple computing nodes that operate as nonlinear summing devices. These nodes are richly interconnected by weighted connection lines, and the weights are adjusted when data are presented to the network during a "training" process. Successful training can result in artificial neural networks that perform tasks such as predicting an output value, classifying an object, approximating a function, recognizing a

Presented at the Conference on Prognostic Factors and Staging in Cancer Management: Contributions of Artificial Neural Networks and Other Statistical Methods, Arlington, Virginia, September 27–28, 1999.

The authors would like to thank Dr. Richard Levine for his enlightening support and guidance in the development of this article.

Address for reprints: Judith E. Dayhoff, Ph.D., Complexity Research Solutions, Inc., 12657 English Orchard Court, Silver Spring, MD 20906; E-mail: DayhoffJ@pressroom.com

Received December 18, 2000; accepted January 26, 2001.

pattern in multifactorial data, and completing a known pattern. Many applications of artificial neural networks have been reported in the literature, and applications in medicine are growing.

Artificial neural networks have been the subject of an active field of research that has matured greatly over the past 40 years. The first computational, trainable neural networks were developed in 1959 by Rosenblatt as well as by Widrow and Hoff and Widrow and Stearns.[1–3] Rosenblatt perceptron was a neural network with two layers of computational nodes and a single layer of interconnections. It was limited to the solution of linear problems. For example, in a two-dimensional grid on which two different types of points are plotted, a perceptron could divide those points only with straight line; a curve was not possible. Whereas using a line is a linear discrimination, using a curve is a nonlinear task. Many problems in discrimination and analysis cannot be solved by a linear capability alone.

The capabilities of neural networks were expanded from linear to nonlinear domains in 1974 by Werbos,[4] with the intention of developing a method as general as linear regression, but with nonlinear capabilities. These multilayered perceptrons (MLPs) were trained via gradient descent methods, and the original algorithm became known as "back-error propagation." Artificial neural networks were popularized by Rumelhart and McClelland.[5] A variety of neural network paradigms have been developed, analyzed, studied, and applied over the last 40 years of highly active work in this area.[6–9]

After the invention of the MLP and its demonstration on applied problems, mathematicians established a firm theoretical basis for its success. The computational capabilities of three-layered neural networks were proven by Hornik et al.,[10] who showed in their general function approximation theorem that, with appropriate internal parameters (weights), a neural network could approximate an arbitrary nonlinear function. Because classification tasks, prediction, and decision support problems can be restated as function approximation problems, this finding showed that neural networks have the potential for solving major problems in a wide range of application domains. A tremendous amount of research has been devoted to methods of adjusting internal parameters (weights) to obtain the best-fitting functional approximations and training results.

Neural networks have been used in many different fields. Military applications include automatic target recognition, control of flying aircraft, engine combustion optimization, and fault detection in complex engineering systems. Medical image analysis has resulted in systems that detect tumors in medical images, and systems that classify malignant versus normal cells in cytologic tests. Time series predictions have been conducted with neural networks, including the prediction of irregular and chaotic sequences.[11–14] More recently, decision support roles for neural networks have emerged in the financial industry[15-17] and in medical decision support.[18]

Applications of neural networks to such a broad spectrum of fields serves to substantiate the validity of researching the use of neural networks in medical applications, including medical decision support. Because of the critical nature of medical decision-making, analytic techniques such as neural networks must be understood thoroughly before being deployed. A thorough understanding of the technology must include the architecture of the network, its paradigm for adjusting internal parameters to reflect patterns or predictions from the data, and the testing and validation of the resulting network. Different formulations for benchmarking or measuring performance must be understood, as well as the meaning of each measured result. It is equally important to understand the capabilities of the neural network, especially for multivariate analysis, and any limitation that arises from the network or from the data on which the network is trained. This article is intended to address the need for bringing an increased understanding of artificial neural networks to the medical community so that these powerful computing paradigms may be used even more successfully in future medical applications.

In the next section, we discuss multifactorial analysis, the building of multifactorial databases, and the limitations of one-factor-at-a-time analysis. In the third section we consider the internal architecture of a neural network, its paradigm for adjustment (training) of internal parameters (weights), and its proven capabilities. In the fourth section we discuss measurements of the performance of neural networks, including the use of verification data sets for benchmarking a trained network, the receiver operating characteristic (ROC) plot, and confidence and prediction intervals. In the fifth section we discuss the uses of neural networks in medical decision support systems. The sixth section summarizes our major findings concerning the current role of neural networks in medicine, and Section 7 concludes with a vision of the emerging uses of neural networks in medicine.

MULTIFACTORIAL ANALYSIS

Neural networks can play a key role in medical decision support because they are effective at multifactorial analysis. More specifically, neural networks can employ multiple factors in resolving medical predic-

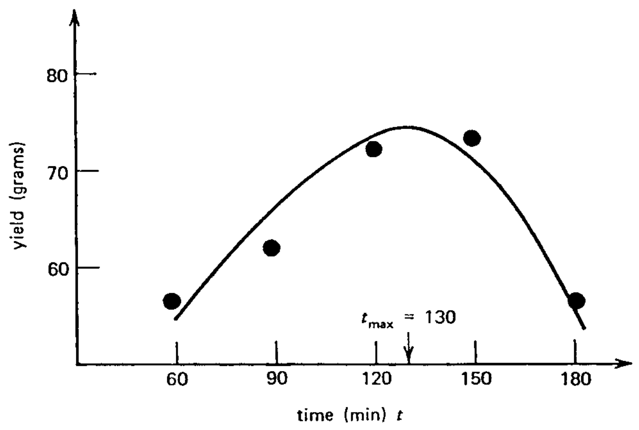

FIGURE 1. First set of experiments showing yield versus reaction time, with the temperature held fixed at 225 °C. Reproduced with permission from Box GEP, Hunter WG, Hunter JS. Statistics for experimenters. New York: John Wiley & Sons, 1978.

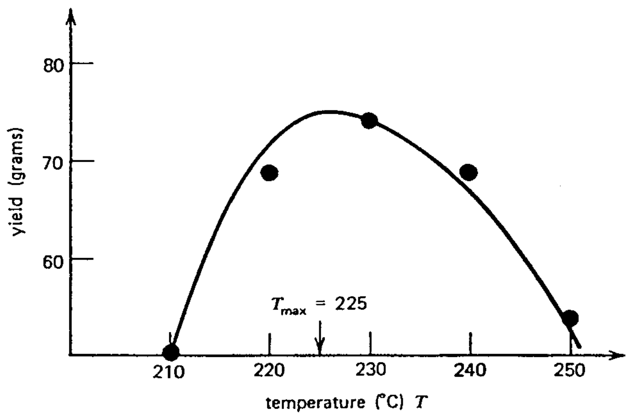

FIGURE 2. Second set of experiments showing yield versus temperature, with the reaction time held fixed at 130 minutes. Reproduced with permission from Box GEP, Hunter WG, Hunter JS. Statistics for experimenters. New York: John Wiley & Sons, 1978.

tion, classification, pattern recognition, and pattern completion problems. Many medical decisions are made in situations in which multiple factors must be weighed. Although it may appear preferable in medical decision-making to use a single factor that is highly predictive, this approach rarely is adequate because there usually is no single definitive factor. For example, there usually is no single laboratory test that can provide decisive information on which to base a medical action decision. Yet a single-factor approach is so easy to use that it is tempting to simplify a multifactorial situation by considering only one factor at a time. Yet it should be clear that, when multiple factors affect an outcome, an adequate decision cannot be based on a single factor alone.

When a single-factor approach is applied to the analysis of multifactorial data, only one factor is varied at a time, with the other factors held constant. To illustrate the limitations of this approach, consider a situation in which a chemist employs a one-factor-at-a-time analysis to maximize the yields from a chemical reaction.[19] Two variables are used: reaction time and temperature. First, the chemist fixes the temperature at T = 225 °C, and varies the reaction time, *t*, from 60 minutes to 180 minutes. Yields are measured and results are plotted (Fig. 1). A curve suggested by the points is shown, leading to the conclusion that for T = 225 °C, the best reaction time is 130 minutes, in which the yield is approximately 75 g.

After a classic one-variable-at-a-time strategy, the chemist now fixes the reaction time at the "best" value of 130 minutes, and varies the temperature T from 210 °C to 250 °C. Yield data are plotted in Figure 2, which shows an approximate peak at 225 °C. This temperature is the same as that used for the first series of runs; again, a maximum yield of roughly 75 g is obtained. The chemist now believes he is justified to conclude that the *overall* maximum is a yield of approximately 75 g, achieved with a reaction time of 130 minutes and a temperature of 225 °C.

Figure 3 shows the underlying response surface for this illustrative example. The horizontal and vertical axes show time and temperature. Yield is represented along a third axis, directed upward from the figure, perpendicular to the axes shown. Contour lines (curves along which yields are the same) are plotted at 60, 70, 80, 90, and 91 g of yield, and the contours form a hill rising up from the two-dimensional plane of the graph. The actual peak occurs at 91 g of yield, at a temperature of 255 °C and a time of 67 minutes. The one-variable-at-a-time approach led to the conclusion of a peak yield of 75 g at 225 °C and 130 minutes, which is only part of the way up the hill depicted in Figure 3; the point selected by the chemist with the one-variable-at-a-time approach was *not* at the top of the hill (the true optimum point). Limitations of the one-variable-at-a-time analysis become starkly obvious with this specific example. If you only change one variable at a time and attempt to maximize results, *you may never reach the top of the hill.*

Although this illustration is of a single algorithm, the illustration serves to demonstrate the point that utilizing the one-variable-at-a-time approach sometimes can be devastating to the analysis results. Because predictive analysis, via artificial neural networks and other statistical methods, is treated as a global minimization problem (e.g., minimizing the difference between the predictor's output and the real data result), minimization, like maximization in the chemical

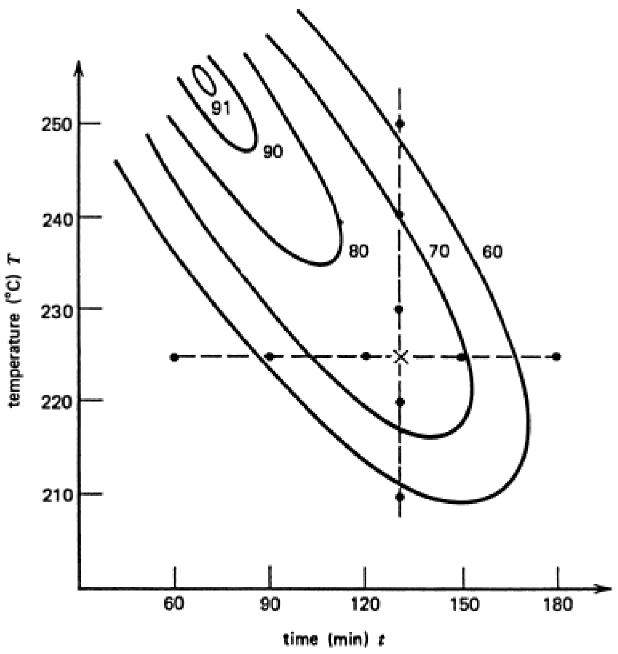

FIGURE 3. True model representing yield versus reaction time and temperature, with points shown for the one-variable-at-a-time approach. Reproduced with permission from Box GEP, Hunter WG, Hunter JS. Statistics for experimenters. New York: John Wiley & Sons, 1978.

reaction yield example, can be missed by the single factor approach.

We submit the following conclusions from this illustrative example:

1. Searching for a single key predictive factor is not the only way to search
2. Knowing a single key predictive factor is not the only way to be knowledgeable
3. Varying a single factor at a time, and keeping other factors constant, can limit the search results.

Neural network technology is intended to address these three key points. The inputs of a neural network are comprised of one or many factors, putatively related to the outputs. When multiple factors are used, their values are input simultaneously. The values of any number of these factors then can be changed, and the next set of values are input to the neural network. Any predictions or other results produced by the network are represented by the value(s) of the output node(s), and many factors are weighted and combined, followed by a weighting and recombining of the results. Thus, the result or prediction does not have to be due to a single key predictive factor but rather is due to a weighting of many factors, combined and recombined nonlinearly.

In a medical application, the problem to be solved could be a diagnostic classification based on multiple clinical factors, whereby the error in the diagnosis is minimized by the neural network for a population of patients. Alternatively, the problem could be the classification of images as containing either normal or malignant cells, whereby the classification error is minimized. Both the diagnostic problem and the image analysis problem are examples of classification problems, in which one of several outcomes, or classes, is chosen or predicted. A third example would be predicting a drug dosage level for individual patients, which illustrates a function fitting application.

In medical decision-making, in which tasks such as classification, prediction, and assessment are important, it is tempting to consider only a single definitive variable. However, we seldom have that luxury because usually no single factor is sufficiently definitive, and many decisions must be made based on weighing the presence of many factors. In this case, neural networks can provide appropriate decision support tools because neural networks take into account many factors at the same time by combining and recombining the factors in many different ways (including nonlinear relations) for classification, prediction, diagnostic tasks, and function fitting.

Building Multifactorial Databases

The building of modern medical databases has suggested an emergent role for neural networks in medical decision support. For the first time in history, because of these growing databases, we have the ability to track large amounts of data regarding substantial and significant patient populations. First, there is a need to analyze data to further our understanding of the patterns and trends inherent in that data, and to ascertain its predictive power in supporting clinical decision-making. Equally important is the need for feedback between the analysis of results and the data collection process. Sometimes the analysis of results indicates that the predictive ability of the data is limited, thus suggesting the need for new and different data elements during the collection of the next set of data. For example, results from a new clinical test may be needed. As each database is analyzed, neural networks and statistical analysis can demonstrate the extent to which disease states and outcomes can be predicted from factors in the current database. The accuracy and performance of these predictions can be measured and, if limited, can stimulate the expansion of data collection to include new factors and expanded patient populations.

Databases have been established in the majority of major medical institutions. These databases originally were intended to provide data storage and re-

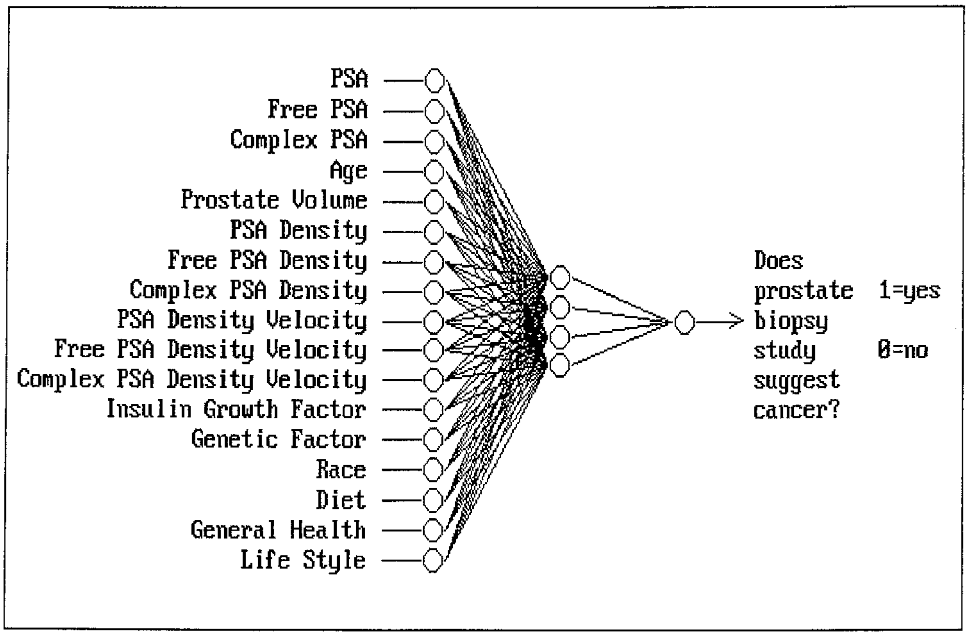

FIGURE 4. Neural network for predicting the outcome of a prostate biopsy study. PSA: prostate specific antigen.

trieval for clinical personnel. However, there now is an additional goal: to provide information suitable for analysis and medical decision support by neural networks and multifactorial statistical analysis. The result of an initial analysis will provide new information with which to establish second-generation databases with an increased collection of data relevant for diagnostic purposes, diagnostic studies, and medical decision support.

Comparisons of computerized multivariate analysis with human expert opinions have been performed in some studies, and some published comparisons identify areas in which neural network diagnostic capabilities appear to exceed that of the experts.[20,21] Because of their success, neural networks and certain other multifactorial analyses can be viewed as new and enhanced tools for extracting medical information from existing databases, and for providing new information to human experts for medical decision support. Traditionally, expert opinions have been developed from the expert's practical clinical experience and mastery of the published literature. Currently we can, in addition, employ neural networks and multivariate analysis to analyze the multitude of relevant factors simultaneously and to learn the trends in the data that occur over a population of patients. The neural network results then can be used by the clinician.

Today, each physician treats a particular selection of patients. Because a particular type of patient may (or may not) visit a particular physician, the physician's clinical experience becomes limited to a particular subset of patients. This "localized knowledge" possibly could be captured and distributed using neural network and related multifactorial techniques. In addition, this "localized knowledge" could be expanded to more of a "global knowledge" by applying the computational methods to expanded patient databases. A physician then could have access to neural networks trained on a population of patients that is much larger than the subset of patients the physician sees in his or her practice.

When a neural network is trained on a compendium of data, it builds a predictive model based on that data. The model reflects a minimization in error when the network's prediction (its output) is compared with a known or expected outcome. For example, a neural network could be established to predict prostate biopsy study outcomes based on factors such as prostate specific antigen (PSA), free PSA, complex PSA, age, etc. (Fig. 4). The network then would be trained, validated, and verified with existing data for which the biopsy outcomes are known. Performance measurements would be taken to report the neural network's level of success. These measurements could include the mean squared error (MSE), the full range of sensitivity and specificity values (i.e., the ROC plot associated with the continuous variable output [0–1] of the neural network), and confidence and prediction intervals. The trained neural network then can be used to classify *each new individual patient*. The predicted classification could be used to support the clinical decision to perform biopsy or support the decision to not conduct a biopsy. This is a qualitatively different approach (a paradigm shift) compared with previous methods, whereby statistics concerning given patient populations and subpopulations are computed and published and a new individual patient then is referenced to the closest matching patient population for clinical decision support. With previous methods, the average outcome for each patient population or subpopulation is used during decision-making. With this new multivariate approach, we are ushering in a new era in medical decision support, whereby neural networks and multifactorial analysis have the potential to produce a meaningful prediction that is unique to each patient. Furthermore, applying ROC methodology to model outputs can tailor the decision for the individual patient further by examining the "cost" tradeoffs between false-positive and false-negative classifications.

WHAT IS INSIDE THE BLACK BOX?

Artificial neural networks are inspired by models of living neurons and networks of living neurons. Artificial neurons are nodes in an artificial neural network, and these nodes are processing units that perform a nonlinear summing function, as illustrated in Figure 5. Synaptic strengths translate into weighting factors along the interconnections. In artificial neural networks, these internal weights are adjusted during a "training" process, whereby input data along with corresponding desired or known output values are submitted to the network repetitively and, on each repetition, the weights are adjusted incrementally to bring the network's output closer to the desired values. Particular artificial neurons are dedicated to input or output functions, and others ("hidden units") are internal to the network.

Neural networks were pioneered by Rosenblatt as well as Widrow and Hoff and Widrow and Stearns.[1–3] Rosenblatt was interested in developing a class of models to describe cognitive processes, including pattern recognition, whereas the latter two groups focused on applications, such as speech recognition and time series predictions (e.g., weather models). Other early contributors included Anderson,[22] Amari,[23] Grossberg,[24] Kohonen,[25] Fukushima,[26] and Cooper,[59] to name a few of the outstanding researchers in this field. These contributions were chronicled by Anderson and Rosenfeld.[27] The majority of the early models, such as the preceptron invented by Rosenblatt,[1] had two layers of neurons (an input layer and an output layer), with a single layer of interconnections with weights that were adjusted during training. However, some models increased their computational capabilities by adding additional structures in sequence before the two-layer network. These additional structures included optical filters, additional neural layers with fixed random weights, or other layers with unchanging weights. Nevertheless, the single layer of trainable weights was limited to solving linear problems, such as linear discrimination, –drawing a line, or a hyperplane in n dimensions, to separate two areas (no curves allowed).[8]

In 1974, Werbos[4] extended the network models beyond the perceptron, a single trainable layer of weights, to models with two layers of weights that were trainable in a general fashion, and that accomplished nonlinear discrimination and nonlinear function approximation. This computational method was named "back-error propagation." Neural networks later were popularized in 1986 by Rumelhart and McClelland.[5]

By far the most popular architecture currently is the multilayered perception (MLP), which can be trained by back-error propagation or by other training methods. The MLP typically is organized as a set of interconnected layers of artificial neurons. Each artificial neuron has an associated output activation level, which changes during the many computations that are performed during training. Each neuron receives inputs from multiple sources, and performs a weighted sum and a "squashing function" (also known as an "activation" function) (Fig. 6). The most popular squashing function is the sigmoid function, as follows:

$$f(x) = \frac{1}{1 + e(-x)} \tag{1}$$

in which x is the input to the squashing function and, in the neural network, is equal to S_j (node j), the sum of the products of the incoming activation levels with their associated weights. This incoming sum (for node j) is computed as follows:

$$S_j = \sum_{i=0}^{n} w_{ji} a_i \tag{2}$$

in which w_{ji} is the incoming weight from unit i, a_i is the activation value of unit i, and n the number of units that send connections to unit j. A "bias" unit (n = 0) is included (Fig. 5).[8] The computation of the weighted sum (Equation 2) is followed by application of the sigmoid function, illustrated in Figure 6, or another nonlinear squashing function.

Artificial neurons (nodes) typically are organized into layers, and each layer is depicted as a row, or collection, of nodes. Beginning with a layer of input neurons, there is a succession of layers, each interconnected to the next layer. The majority of studies utilize three-layer networks in which the layers are fully in-

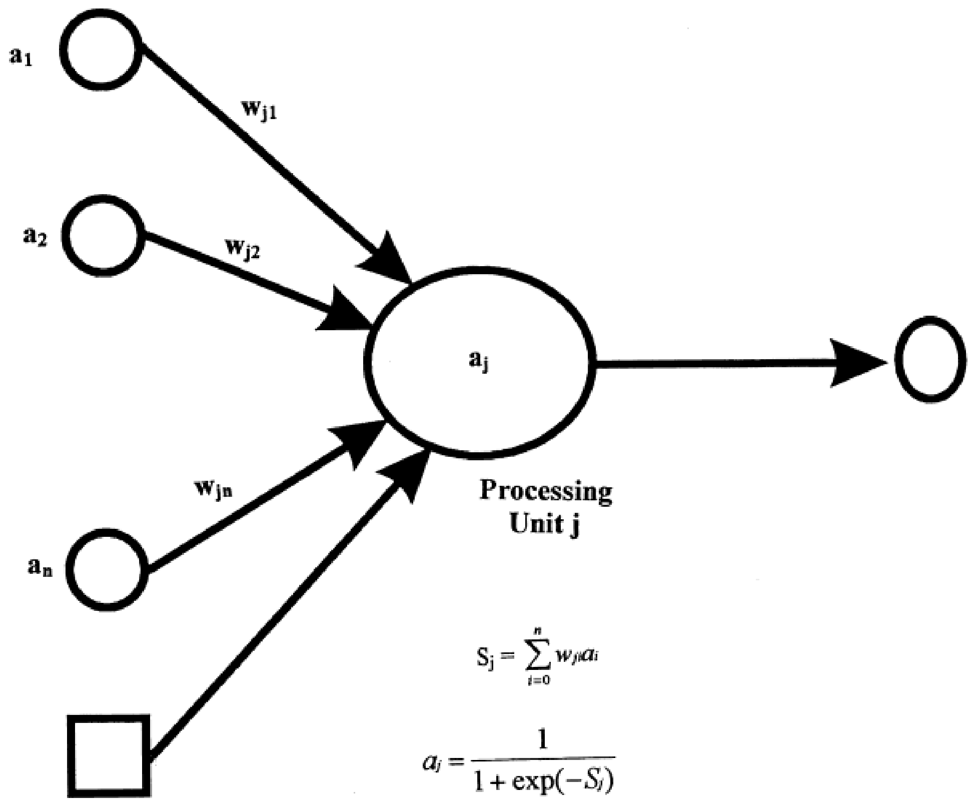
a_1
a_2
a_n
w_{j1}
w_{j2}
w_{jn}
a_j
Processing Unit j
$S_j = \sum_{i=0}^{n} w_{ji} a_i$
$a_j = \frac{1}{1+\exp(-S_j)}$

FIGURE 5. Illustration of an artificial neural network processing unit. Each unit is a nonlinear summing node. The square unit at the bottom left is a bias unit, with the activation value set at 1.0. S_j = incoming sum for unit j, a_j = activation value for unit j, and w_{ji} = weight from unit i to unit j.

terconnected. This means that each neuron is connected to all neurons in the next layer, as depicted in Figure 7. Each interconnection has an associated "weight," denoted by w, with subscripts that uniquely identify the interconnection. The last layer is the output layer, and activation levels of the neurons in this layer are considered to be the output of the neural network. As a result, the general form of Equations 1 and 2 becomes

$$a_{j,k+1} = \frac{1}{1 + \exp(-\sum w_{ji,k} a_{i,k})} \tag{3}$$

in which $a_{i,k}$ represents the activation values of node i in layer k, and $w_{ji,k}$ represents the weight associated with the connection from the ith node of the kth layer to the jth node of layer k+1. Because there typically are three layers of nodes, there then are two layers of weights, and k = 1 or 2.

Initially, the weights on all the interconnections are set to be small random numbers, and the network is said to be "untrained." The network then is presented with a training data set, which provides inputs and desired outputs to the network. Weights are adjusted in such a way that each weight adjustment increases the likelihood that the network will compute the desired output at its output layer.

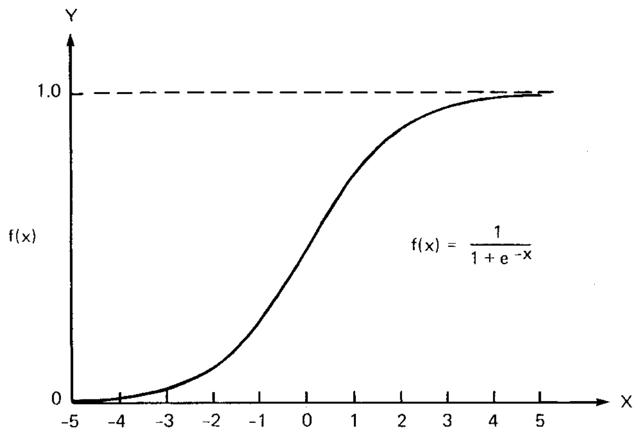

FIGURE 6. The sigmoid function.

Attaining the appropriate parameter values (weights) in the neural network requires "training." Training is comprised of many presentations of data to the neural network, and the adjustment of its internal parameters (weights) until appropriate results are output from the network. Because training can be difficult, a tremendous number of computational options and enhancements have been developed to improve the training process and its results. Adjustment of weights in training often is performed by a gradient

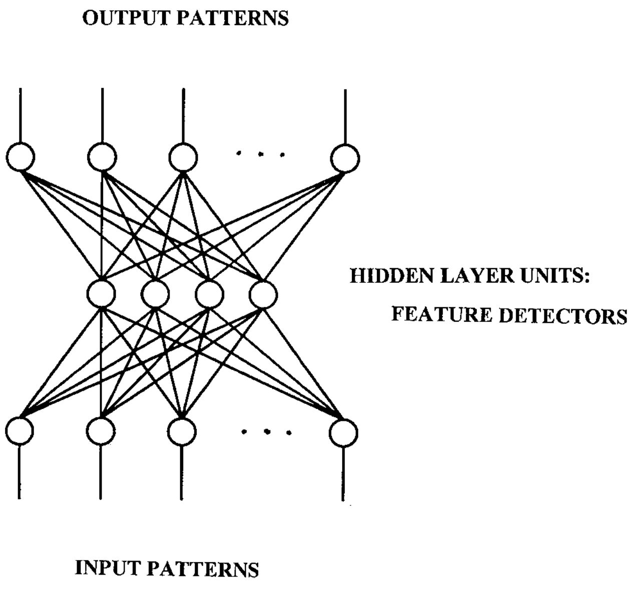

FIGURE 7. A three-layer, fully interconnected feedforward neural network.

descent computation, although many improvements to the basic gradient descent algorithm have been designed.

The amount to which the neural network is in error can be expressed by the MSE calculation is determined as follows:

$$\mathrm{MSE} = \frac{1}{P}\sum_{p=1}^{P}\sum_{i=1}^{n}(d_{i,p} - a_{i,3})^2 \tag{4}$$

in which $d_{i,p}$ is the desired output of output unit i for input pattern p, P is the total number of patterns in the data set, n is the number of output units, and the sums are taken over all data patterns and all output units. Sometimes the root-mean-square error (RMS) is calculated, which simply is the square root of the MSE.

Consider a three-dimensional plot that shows the RMS error produced by the neural network on the z-axis (e.g., vertical axis), and two of the weights of the neural network on the x and y axis (horizontal plane). This graphing procedure then represents an "error surface" that appears like a mountainous terrain. In this mountainous terrain, we are seeking the minimum (i.e., the bottom of the lowest valley), which reflects the minimum error that can be attained. The weights at which this minimum is attained correspond to the x and y values associated with the bottom of the valley. An analogy would be that the x and y values (corresponding to the weights) would be the longitude and latitude of the bottom of a valley in the mountainous terrain. A local minimum is the bottom of any valley in the mountainous terrain and a global minimum is the bottom of the lowest valley of the entire mountainous region. The gradient descent computation is intuitively analogous to a scenario in which a skier is dropped from an airplane to a random point on the mountainous terrain. The skier's goal is to find the lowest possible elevation. At each point, the skier checks all 360 degrees of rotational direction, and takes a step in the direction of steepest descent. This will result in finding the bottom of a valley nearby to the original starting point. This bottom certainly will be a local minimum; it may be a global minimum as well.

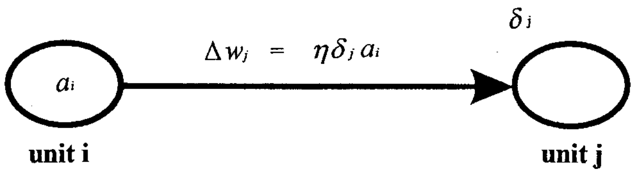

FIGURE 8. The adjustment of the value of an individual weight.

Gradient descent weight-training starts with inputting a data pattern to the neural network. This determines the activation values of the input nodes. Next, forward propagation ensues, in which first the hidden layer updates its activation values followed by updates to the output layer, according to Equation 3. Next, the desired (known) outputs are submitted to the network. A calculation then is performed to assign a value to the amount of error associated with each output node. The formula for this error value is as follows:

$$\delta_{j,3} = (d_j - a_{j,3})\, f'(S_{j,3}) \tag{5}$$

in which d_j is the desired output for output unit j, $a_{j,3}$ is the actual output for output unit j (layer 3), f(x) is the squashing function, and $S_{j,3}$ is the incoming sum for output unit j, as in Equation 2.

After these error values are known, weights on the incoming connections to each output neuron then can be updated. The update equation is as follows:

$$\Delta w_{ji,k} = \eta \delta_{j,k+1} a_{i,k} \tag{6}$$

in which k = 2 while updating the layer of weights on connections that terminate at the output layer.

Figure 8 illustrates the updating of the weight along a single connection; the error delta value of the target neuron is multiplied by the activation value of the source neuron. The variable η is the "learning rate parameter."

As the "backwards propagation" ensues, an error value next is calculated for each hidden node, as follows:

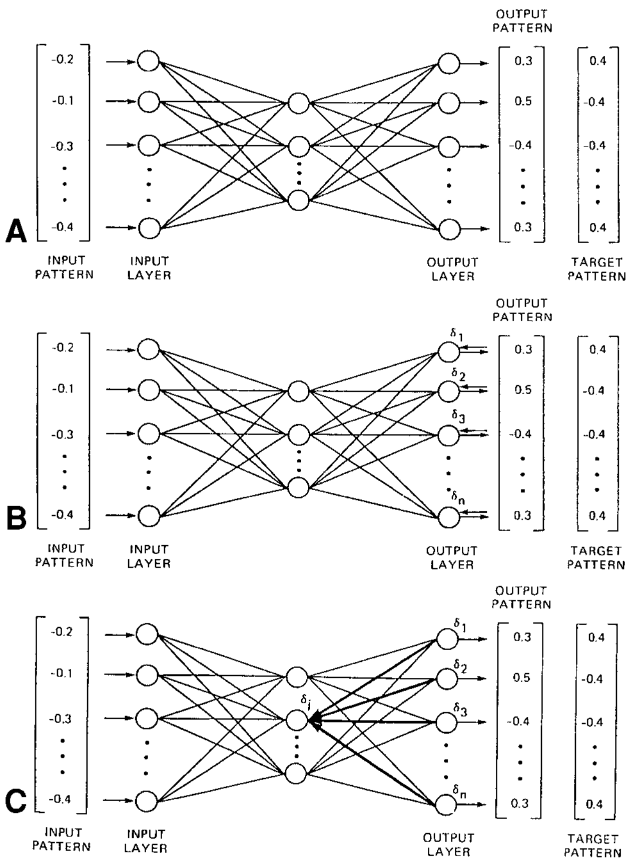

FIGURE 9. Back-error propagation's flow of calculations. (A) In the forward propagation step, the activation levels of the hidden layer are calculated and the activation levels of the output layer then are calculated. The activation levels of the output layer become the output pattern, which is aligned with the target pattern (the desired output). (B) An error difference is taken for each entry in the output and target patterns, and from this the error delta value δ is calculated for each output unit i. Then, for each output unit, weights on its incoming connections are adjusted. (C) An error delta value is calculated for each unit in the hidden layer. Next, for each hidden unit, weights on its incoming connections are adjusted. Reproduced with permission from Dayhoff JE. Neural network architectures: an introduction. New York: Van Nostrand Reinhold, 1990.

$$\delta_{i,2} = (\sum_j \delta_{j,3} w_{ji,2}) f'(S_{i,2}) \tag{7}$$

After these error values are known, weights on the incoming connections to each hidden neuron then can be updated. The update equation (Equation 6) is used again, substituting $k = 1$ for weights on connections that start at the first layer. Figure 9 illustrates the backward flow of computations during this training paradigm. The derivation for these equations is based directly on the gradient descent approach, and uses the chain rule and the interconnected structure of the neural network. Mathematical details were given by Rumelhart and McClelland[5] and Mehrotra et al.[8]

There are other training algorithms that seek to find a minimum in the error surface (e.g., the mountainous terrain described earlier). Some of these algorithms are alternatives to gradient descent, and others are strategies that are added to the gradient descent algorithm to obtain improvement. These alternative strategies include jogging the weights, reinitializing the weights, the conjugate gradient technique, the Levenberg–Marquant method, and the use of momentum.[8,28] Many other techniques have been developed for the training of neural networks. Some of these techniques are tailored toward finding a global rather than a local minimum in the error surface, such as using genetic algorithms for training. Other techniques are good for speeding the training computations,[29] or may provide other advantages in the search for a minimal point. In addition, there is a set of techniques for improving the activation function, which usually has been a sigmoid function, but has been expanded to include other nonlinear functions[30,31] and can be optimized during training.[30–33] Some algorithms address finding the optimal size of the hidden layer and allow the number of hidden layer nodes to grow and/or shrink during training.[8]

It should always be recognized that results with neural networks always depend on the data with which they are trained. Neural networks are excellent at identifying and learning patterns that are in data. However, if a neural network is trained to predict a medical outcome, then there must be predictive factors among the inputs to the network before the network can learn to perform this prediction successfully. In more general terms, there have to be patterns present in the data before the neural network can learn (the patterns) successfully. If the data contain no patterns or no predictive factors, then the neural network performance cannot be high. Thus, neural networks are not only dependent on the data, but are limited by the information that is contained in those data.

It is important to consider that a neural network is simply a set of equations. If one considers all different types of neural networks, with varied node updating schemes and different training paradigms, then one has a collection or class of systems of equations. As such, neural networks can be considered a kind of language of description for certain sets of systems of equations. These equations are linked together, through shared variables, in a formation diagrammed as a set of interconnected nodes in a network. The equations form a system with powerful and far-reaching learning capabilities, whereby complex relations

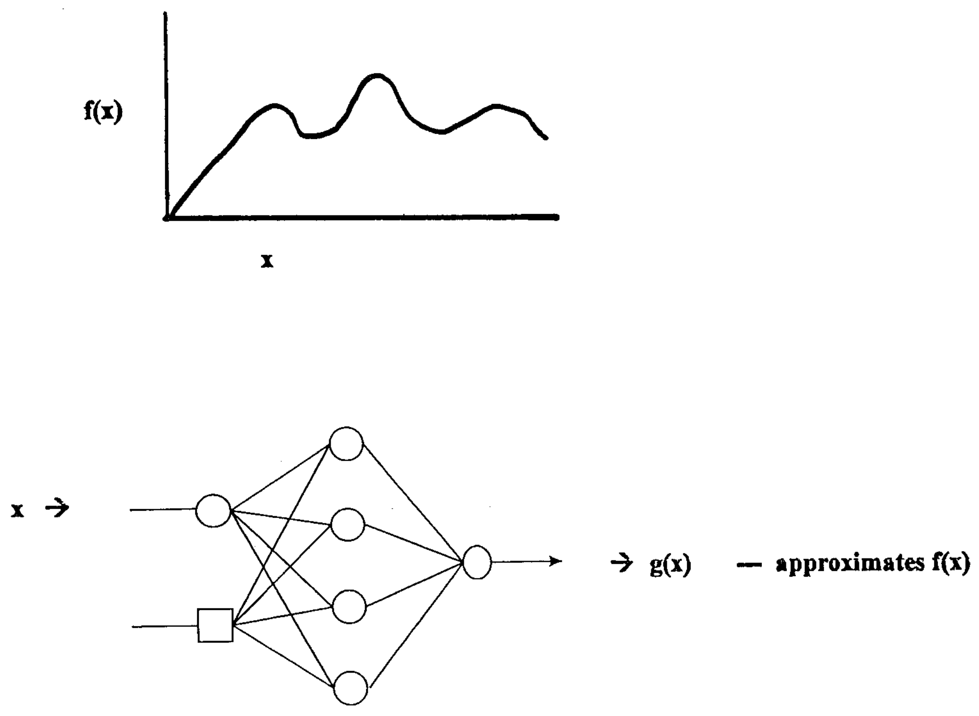

FIGURE 10. Function approximation. Top: a function f(x). Bottom: a neural network configured to determine an approximation to f(x), given the input x. Neural network weights exist that approximate any arbitrary nonlinear function.

can be learned during training and recalled later with different data.

Many of the same or similar equations were in existence before they were labeled "neural networks." But the relabeling and further development of these equations, under the auspices of the field of neural networks, has provided many benefits. For example, when a neural network is used to describe a set of equations, the network diagram immediately shows how those equations are related, showing the inputs, outputs, and desired outputs and intuitively is easier to conceptualize compared with methods that involve equations alone.

The conceptual paradigm of an array of simple computing elements highly interconnected with weighted connections has inspired and will continue to inspire new and innovative systems of equations for multivariate analyses. These systems serve as robust solvers of function approximation problems and classification problems. Substantial theory has been published to establish that multilayered networks are capable of general function approximation.[10,34–36] These results are far-reaching because classification problems, as well as many other problems, can be restated as function approximation problems.

Function Approximation

A general function approximation theorem has been proven for three-layer neural networks (MLPs).[10] This result shows that artificial neural networks with two layers of trainable weights are capable of approximating any nonlinear function. This is a powerful computational property that is robust and has ramifications for many different applications of neural networks. Neural networks can approximate a multifactorial function (e.g., the "underlying function") in such a way that creating the functional form and fitting the function are performed at the same time, unlike nonlinear regression in which a fit is forced to a prechosen function. This capability gives neural networks a decided advantage over traditional statistical multivariate regression techniques.

Figure 10 illustrates a function approximation problem addressed by a neural network. The plot at the top is a function for which a neural network approximation is desired. The function computes a value $y = f(x)$ for every value of x. The neural network is trained to input the value of x, and to produce, as output, an approximation to the value f(x). The neural network is trained on a section of the function. Theoretic results show us that regardless of the shape of the function, a neural network with appropriate weights and configuration can approximate the function's values (e.g., appropriate weight values exist).

Sometimes function approximation problems occur directly in medicine, such as in drug dosing applications. Time series prediction tasks, such as predicting the next value on a monitor or blood test, also are useful in medicine and can be restated in terms of

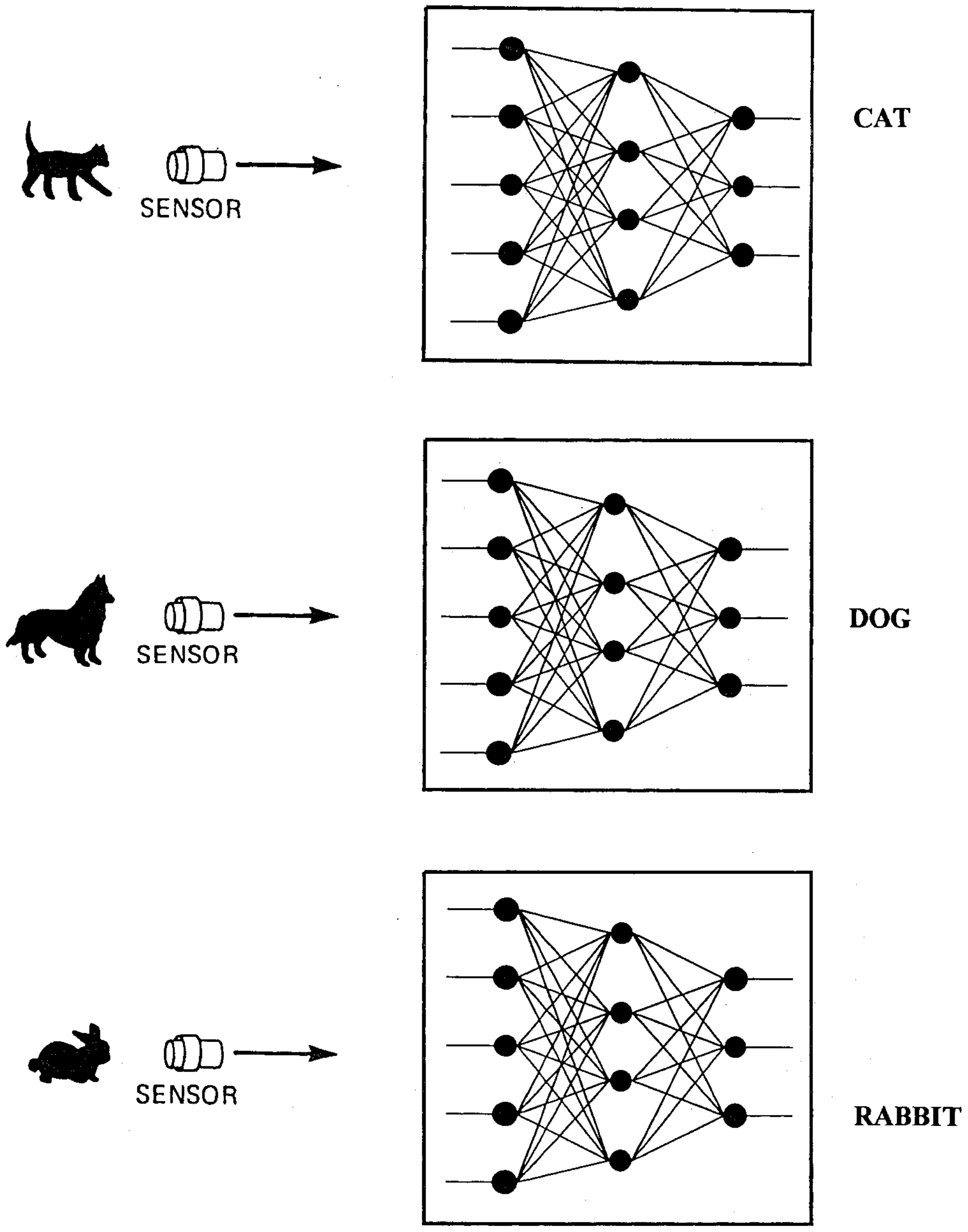

FIGURE 11. A neural network is configured with three output units that represent "cat," "dog," and "rabbit." The training set is comprised of vectors that represent images of a cat, dog, or rabbit, paired with the corresponding output unit having the value "1" whereas the other output units have the value "0." After training, the same neural network can recognize any of the three animals.

function approximation. Classification tasks, such as classifying tissue into malignant stages, occur frequently in medicine and can be restated as function approximation problems in which a particular function value represents a class, or a degree of membership in a class. Diagnosis is an instance of a classification task in which the neural network determines a diagnosis (e.g., a class or category) for each set of indicators or symptoms. Thus, the neural network's function approximation capabilities are directly useful for many areas of medicine, including diagnosis, prediction, classification, and drug dosing.

Pattern Recognition

Figure 11 illustrates the approach for applying neural networks to pattern recognition and classification. On the left are silhouettes of three different animals: a cat, a dog, and a rabbit. Each image is represented by a vector, which could be the pixel values of the image or a preprocessed version of the image. This vector has different characteristics for the three different animals. The neural network is to be trained to activate a different output unit for each animal. During training, the neural network is presented with paired elements: an input pattern (the image of one animal) and the corresponding desired output pattern (an "on" value for the output unit that represents the presented animal and an "off" value for the output units that do not represent the presented animal). After each presentation, the internal weights of the neural network are adjusted. After many presentations of the set of animals to be learned, the weights are adjusted until the neural network's output matches the desired outputs on which it was trained. The result is a set of internal weight values that has learned *all three patterns at the same time.*

How does such a network recognize three different objects with the same internal parameters? An explanation lies in its representation of the middle (hidden) layer of units as feature detectors. If each of these units becomes activated exclusively in the presence of a certain feature, such as "long ears" or "pointed tail," then the second layer of weights can combine the features that are present to activate the appropriate output unit. The first layer of weights combines specific parts of the input pattern to activate the appropriate feature detectors. For example, certain feature detectors (hidden units) could activate when presented with the features of "long ears" or "round tail." Connections originating at these feature detectors then could develop strong weights if they terminate at the output unit that represents "rabbit." In this fashion, the neural network can build multilevel computations that perform complex tasks such as pattern classification with simple, layered units.

A classification task can be represented as an n-output node neural network (one node for each class) with an output value of 1 designating "class member" and 0 designating "class nonmember." Thus, classification problems may be subsumed as function approximation problems, in which an output node produces a number from 0–1, and values above a threshold (for example, 0.5) represent Class 1, whereas the others represent Class 0. Alternatively, the output value in the 0–1 interval, in response to a pattern presented to the network, may be interpreted as an index (reflecting a probability or degree of membership) that associates the pattern with the class corresponding to the output node. The general function approximation theorem then is extensible, and can be used to demonstrate the general and powerful capabilities of neural networks in classification problems.

MEASUREMENTS OF PERFORMANCE AND RELIABILITY

There exist many different performance measurements for neural networks. Simple performance measures can be employed to express how well the neural network output matches data with known outcomes. Performance metrics include the MSE and RMS. Occasionally percent-correct is used as a performance measurement. The area under the ROC plot is a more extensive performance measure to use with classification neural networks. Fuller use of ROC plots is a more elaborate way of demonstrating the performance of classification neural networks and will be discussed in more detail later.

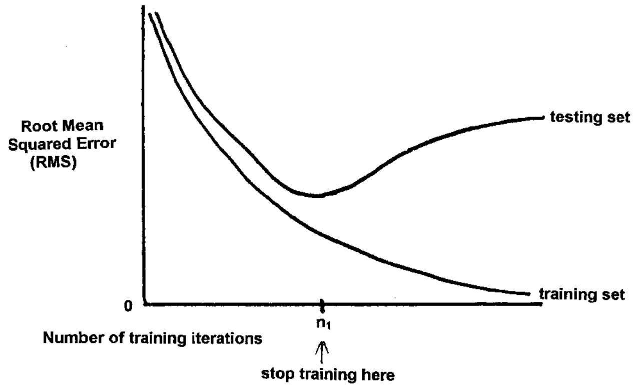

FIGURE 12. A classic training curve in which a training set gives decreasing error as more iterations of training are performed, but the testing (validation) set has a minimum at n_1. Training iterations beyond n_1 are considered to be "overtraining."

A comparison with experts can be conducted to measure performance. Does the neural network predict an outcome or diagnosis as often as a trained expert does? Does the neural network agree with the experts as often as they agree with each other?

Training Sets, Testing/Validation Sets, and Verification Sets

It is important to note that when training a neural network, three nonoverlapping sets of data must be used. Typically, a researcher would start with a compendium of data from a single population of patients. This data then is divided, at random, into three subsets: the training set, the validation (or testing) set, and the verification set. The training set is used for the adjustment of weights during training. The testing or validation set is used to decide when to stop training. The need for the testing set is motivated by the graph in Figure 12, which illustrates an RMS of the training and testing sets plotted as a function of the number of training iterations. The RMS on the training set decreases with successive training iterations, as does the RMS on the test set, up to a point. Then, at n_1 iterations, the RMS on the test set begins to increase whereas the RMS on the training set continues to decrease. This latter region of training is considered to be "overtraining."

An intuitive description of "overtraining" is as follows. Suppose that one is instructing a child on how to recognize the letters "A" and "B." An example A and an example B are presented to the child. The child soon can identify the A as an "A," and the B as a "B." However, when the child then is presented with a new picture of an A and a B, the child fails to recognize them correctly because the child has memorized the particular (exact) angles and contours of the original two drawings rather than learning the overall structure of the "A" and "B. " This type of error arises from a failure to generalize, and can be considered memorization of the data.

To avoid overtraining, the training is stopped when the error on the test set is at a minimum, which occurs at n_1 iterations in Figure 12. To report the performance of the network, a separate dataset, called the verification set, then is used. The performance on this set is reported and used in determining the validity of the neural network results. The verification set is not used at all during training or testing (validation). Thus the verification set is independent of the neural network training, and the neural network results on the verification set can be considered a true (unbiased) prediction of the neural network performance on new data. The performance on the verification set provides a proper benchmark evaluation metric for the performance of the neural network as a predictor or classifier when deployed for actual use.

ROC Methodology and Neural Networks

ROC methodology is a computational methodology that emerged during World War II in the field of signal processing with applications to radar detection. It has a very important connection to neural network methodology applied to classification applications. A "receiver," such as a radar system, has certain "operating characteristics," such as the rate at which it identifies an enemy aircraft when indeed there is one (its sensitivity, or true-positive rate) and the rate at which it identifies a nonenemy aircraft when indeed a nonenemy aircraft is present (its specificity, or true-negative rate). The use of ROC methodology later was expanded to psychophysics applications (vision and hearing testing),[37] and then to other disciplines before it eventually was applied to medicine in the field of radiology.[38,39] ROC methodology now is fairly well known in many medical subspecialties, particularly in the clinical laboratory for comparing the accuracy of competing tests,[40] and in diagnosis for trading off sensitivity and specificity measures.[41,42] One important feature of ROC methodology is that it readily incorporates prevalence and misclassification cost factors in decision-making. In general, ROC methodology can be applied to any observer or system (i.e., "the receiver,"), human, mechanical, or computerized, that is charged with the task of choosing between dichotomous alternatives such as disease-free or diseased. Ideally, such systems will exhibit high levels of sensitivity and specificity. Although in many systems parameters that affect sensitivity and specificity are adjustable, these two measures unfortunately offset each other so that a gain in one usually is at the expense of the other.

Because there are many good references to ROC methodology in the literature, we will only introduce the fundamentals in the current study.[43] The central feature of ROC methodology is the ROC plot, which is derived from a dataset representing observations of values of a single independent variable, x (e.g., a laboratory test value), paired with associated values for a dependent variable, d, which corresponds to the degree of membership an entity with the value x has in some set such as "the set of all patients with a particular disease," in which 0 indicates no membership, 1 indicates full membership, and continuous values inside the 0–1 interval indicate partial (fuzzy) degrees of membership in the set. For ROC methodology to be useful, it is important that there be a general underlying monotonic relation between the independent variable, x, and the dependent classification variable, d. This means that as the x values increase, d has a general overall tendency to either increase or decrease, but not to, for example, increase and then decrease or exhibit other erratic behavior. It also is important that the classification values, d, are derived from a trusted gold standard, in which the actual classes for the training data are known.

ROC methodology originally was developed for dichotomous outcomes (i.e., outcomes in which an event either was or was not a member of the positive class [$d = 1$]). Later, DeLeo and Campbell[44] extended ROC methodology to fuzzy ROC methodology in which an event could have a continuous degree of membership in the 0–1 interval in the class as well as 0 and 1 values.[43,45–48]

The ROC plot typically is graphed as a plot of sensitivity (true-positive fraction) on the ordinate (y-axis) and specificity (false-positive fraction) on the abscissa (x-axis). An empiric ROC plot can be derived directly from the raw data by sorting the x values in ascending rank order with their corresponding d values and then computing sensitivity and specificity values for all points between all unique x values. Sensitivity simply is the sum of all positive cases ($d = 1$) above the current x value divided by the total number of known positive cases. Specificity is the sum of all negative cases ($d = 0$) below the current x-value di-

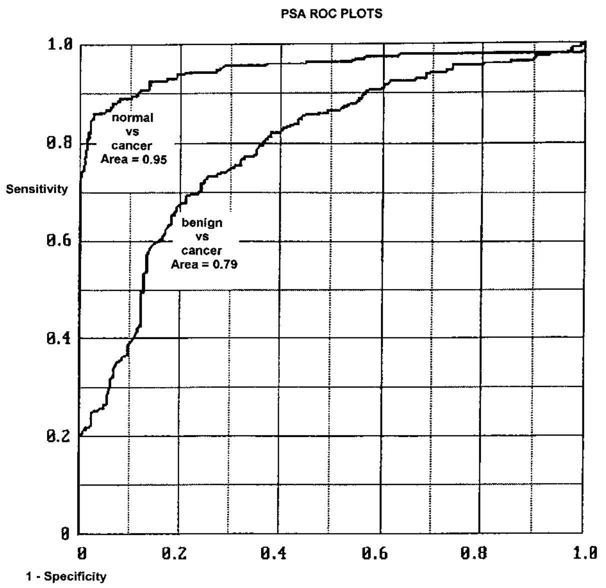

FIGURE 13. Prostate specific antigen (PSA) receiver operating characteristic (ROC) plots. Sensitivity is plotted as a function of (1-specificity). The PSA value at which a cutoff between the "benign" versus "cancer" diagnosis is made is varied across the lower curve, whereas the cutoff between the "normal" versus "cancer" diagnosis is plotted in the upper curve.

vided by the total number of known negative cases. Theoretic ROC plots are obtained by either smoothing empiric ROC plots or by smoothing sensitivity versus x and specificity versus x functions prior to constructing the ROC plot. Because such smoothing introduces biases, it generally is not recommended.

Figure 13 shows two empiric ROC plots related to PSA cancer detection. These ROC plots were derived from data provided by Dr. Axel Semjonow at the University of Munster. One ROC plot reflects how well PSA differentiates healthy patients from those with prostate carcinoma. The other plot reflects how well PSA differentiates between patients with benign prostate hyperplasia (BPH) and those with prostate carcinoma. Note that the areas under the ROC plots (AURP) are given. Because the ROC plot resides in a 1×1 grid, its maximum area is 1. The AURP is a statistic: the probability that any randomly drawn patient with prostate carcinoma will have a PSA value higher than any randomly drawn healthy patient (or BPH patient in the second example). In short, the AURP measures how well the independent variable (PSA value) separates two dichotomous patient groups. In addition to having a sensitivity value and a specificity value, each point on the ROC plot has two other important attributes associated with it: a PSA value and a likelihood (the slope at that point) of being in the prostate carcinoma (or BPH) class.

Typically, ROC plots have been used to classify prospective events by means of a preset threshold value for the independent variable, which is selected after careful consideration is given to sensitivity versus specificity tradeoffs, and prevalence and misclassification cost concerns. An alternative strategy is to determine where the PSA value for the particular prospective patient falls on the ROC plot and use the likelihood ratio value to compute a probability or "degree-of-membership-value-in-the-prostate carcinoma-class," and use the latter value with other information (including prevalence and misclassification costs or correct classification benefits) in making decisions regarding patient management.

There are two important ways that well established ROC methodology can be employed when neural network methodology is applied to classification problems.[43,49–51] The first is to use the AURP of the ROC plot formed with the classification node (continuous) output values as the independent variable as a performance measure rather than the MSE measure because we really need a measure of dichotomous class separation and not a measure of goodness of function fit such as the MSE provides. Thus the output value of a classification neural network node is treated as a continuously valued "composite index" bearing the classification influence of all of the input variables. The second way to use ROC methodology is related to sharpening final classifications with sensitivity versus specificity tradeoffs, and prevalence and misclassification costs. This approach allows us to choose an operating point on the ROC plot based on the factors just mentioned, and to shift to another operating point more appropriate to the particular patient.

Confidence in Using Neural Networks: Confidence and Prediction Intervals

When a neural network has passed successfully through the training, validation, and verification phases, a humanized front-end software module to collect case-specific data can be designed and coupled to the neural network for use in a "production" environment. Because the neural network has undergone generalized learning (versus memorization) it is, in a literal mathematical sense, interpolating or extrapolating results for new incoming cases. Because success cannot be assured if the data values for new cases extend beyond the ranges of the values used in training, it is very important that the front-end module screens new input data for acceptability and either refuses to process cases with data that are outside acceptable limits or posts a cautionary notice when it does process "extreme" cases. In cases within the range in which the neural network is trained, a measure of confidence can be made by calculating a confidence or prediction interval.

When a medical practitioner performs a patient care action influenced by the output of a neural network, it obviously is very important that the practitioner has confidence in the neural network. In general, a trained, tested, and verified neural network does not provide a unique solution because its final trained resting state is determined by several factors, including the original randomized starting weights, the number of cases presented in a training cycle, the order of presentation of cases during a training cycle, and the number of training cycles. Other mathematical alterations during training such as the use of momentum, adjusting the learning constant, "jogging" the weights, etc., also will affect the final state of the trained neural network. We might draw interesting anthropomorphic analogies here to human learning and expertise. If we believe that a single well trained neural network is good, then we might believe that 1000 are better, just as we might believe that if one well trained human expert is good, then 1000 might be better.

With this idea in mind, for a particular application, we suggest producing a large cadre of neural networks with each one trained using bootstrap sampling (random sampling with replacement) for rich training variability.[52–54,58] The resulting "production" systems would input values for individual case predictive variables, screen them for acceptability, and then process them with each of the 1000-member cadre of neural networks, just as it could with a 1000-member human expert cadre to predict, for example, a diagnostic category, a survival probability, or a drug dosage level. One thousand answers would be generated. A fictional illustration of a frequency distribution of the output values of a 1000-cadre of neural networks is shown in Figure 14. From unsmoothed, skewed, nonparametric distributions such as the one shown, measures of central tendency such as the mean, mode, and median and measures of variance such as the 90%, 95%, and 99% Gaussian and nonparametric predictive intervals can be extracted. We recommend using the median or mode and nonparametric predictive intervals because they are unbiased, natural estimators of central tendency and variance. One potential problem with using parametric prediction intervals is that impossible ranges may result such as the 99% parametric predictive interval based on the data in Figure 14, which has an impossibly high end probability value of 1.03.

In the example in Figure 14, the median value was 0.72, the mode was 0.76, and the 90% nonparametric predictive interval was 0.47–0.84. If this distribution

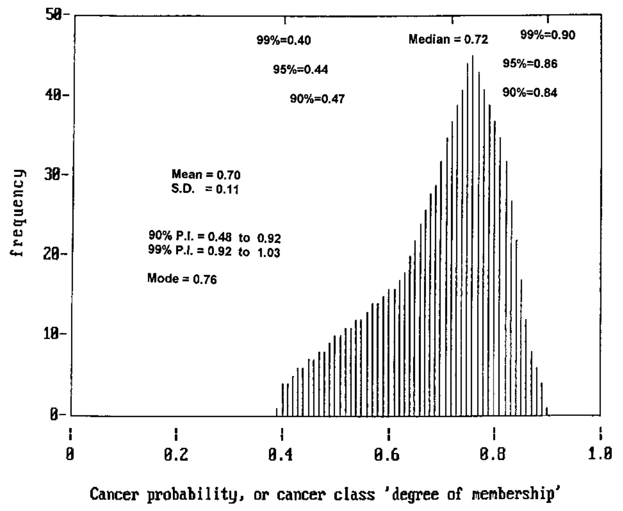

FIGURE 14. Distribution of output of 1000 different neural networks with parametric and nonparametric predictive intervals (PI). SD: standard deviation.

was produced for a particular patient from 1000 neural networks designed to predict an event such as cancer remission or coronary heart disease, we could state with 90% confidence that the probability the event will occur ranges from 0.47–0.84 with a median value of 0.72. If the collection of neural networks was trained to predict the optimal daily mean dose of a particular medication, such as levothyroxine, for an individual patient, and that when we trained the 1000 neural networks we scaled the output dose on a 0–200-μg linear scale, then we might choose the mode value, 0.76, and scale it back up (0.76 $\times$ 200 μg) to arrive at a mean daily dose of 152 μg of levothyroxine for that patient and be 90% certain that the optimum mean daily dose lies between 94–168 μg. For function-fitting problems such as drug dosing, a narrow predictive interval raises confidence in using the median values whereas wider intervals lower confidence. With classification problems such as survival prediction, the same thinking applies; however, when prevalence and misclassification considerations arise, these distributions must undergo nonlinear transformations before measures of central tendency and variance are determined, a subject that is beyond the scope of the current study.[55]

MEDICAL DECISION SUPPORT

Medical decision support with neural networks is a field that we believe is ready for serious development, as evidenced by the fact that the number of studies and results are growing at an accelerating rate. Neural networks and other multifactorial computational methods employed alone or in combination to per-

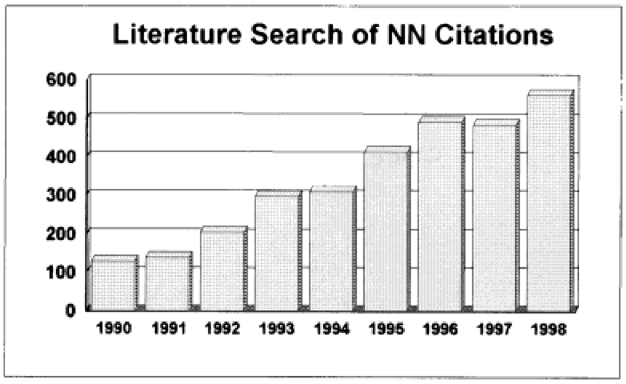

FIGURE 15. Literature search of neural network (NN) citations showing the number of citations in the medical literature for the years 1990–1998.

form "intelligently" are being subsumed under the new rubric "smart computing." Fully developed "smart computing" systems are said to be the products of "smart engineering."[56] We believe that a coincidence of three factors has set the stage for the serious, rapid development of these systems. These factors are 1) ever-increasing medical databases, 2) the proliferation of new medical markers, and 3) the accumulation of in-depth experiential knowledge gained over the last decade by responsible neural network experts.

A recent search of the medical literature demonstrates a growing active interest in the use of neural networks. As Figure 15 indicates, the number of references has grown from 130 in 1990 to 570 in 1998, a near-linear rate of 55 new citations per year. The references demonstrate a wide range of medical applications, as summarized in Table 1. There are many additional references in the computer science, engineering, mathematics, and statistical literature that describe other medical applications and/or neural network methodology that is applicable to medical applications. The first author of the current study (J.E.D.) is a biophysicist who has worked on neural network architectures, with applications to general prediction and classification problems, including research in medical imaging. The second author (J.M.D.) is a computational methodologist who has been actively working at the National Institutes of Health on several biomedical applications of neural networks including individual patient survival prediction,[49–51] diagnosis,[57] drug target dosing, laboratory instrument quality control, image processing and analysis, and text searching.

The literature and the experience of both authors suggest the following five factors as favorable indicators for applying neural network methods to medical applications: 1) outcomes influenced by multiple factors, 2) the desirability of constructing composite indices from multiple tests, 3) the fact that prior mathematic models do not exist or have serious limitations, 4) a need for results that apply to each individual rather than to populations, and 5) a need for robust nonlinear modeling. The authors' combined experience further suggests that important issues to consider very carefully for successful applications of neural networks include: 1) methods of data collection; 2) data types and data scaling techniques; 3) prevalence and misclassification cost issues; 4) training, validation, and verification schema; 5) performance metrics; 6) treatment of missing and censored data; and 7) providing predictive and confidence intervals.

To illustrate briefly the practical applications of neural networks in medical decision support systems, we have presented examples pertaining to breast carcinoma survival estimation, prostate carcinoma detection, and coronary heart disease risk prediction. The first example is illustrated in Figure 16 which shows a single hidden layer MLP and uses patient age, the number of active lymph nodes, estrogen receptor status, and progesterone receptor status to predict "living versus deceased" status for breast carcinoma patients

TABLE 1
Applications of Neural Networks in Medicine Published Between 1990–1998, Including Topics Under Diagnosis, Histology, Imaging, Outcome Predictions, and Waveforms

Medical applications of neural networks (1990–1998)
Diagnosis
Appendicitis
Dementia
Myocardial infarction
Prostate carcinoma
STDs
Histology
Pap smear
WBD differential
Imaging
NMR scans
Perfusion scans
PET scans
X-rays
Outcome predictions
Cancer
CPR
Surgery
Waveforms
Arrhythmias
Chromatography
EEGs
EKGs

STDs: sexually transmitted diseases, Pap: Papanicolaou, WBD: whole blood differential; NMR: nuclear magnetic resonance; PET: positron emission tomography; CPR: cardiopulmonary resuscitation; EEG: electroencephalography; EKG: electrocardiography.

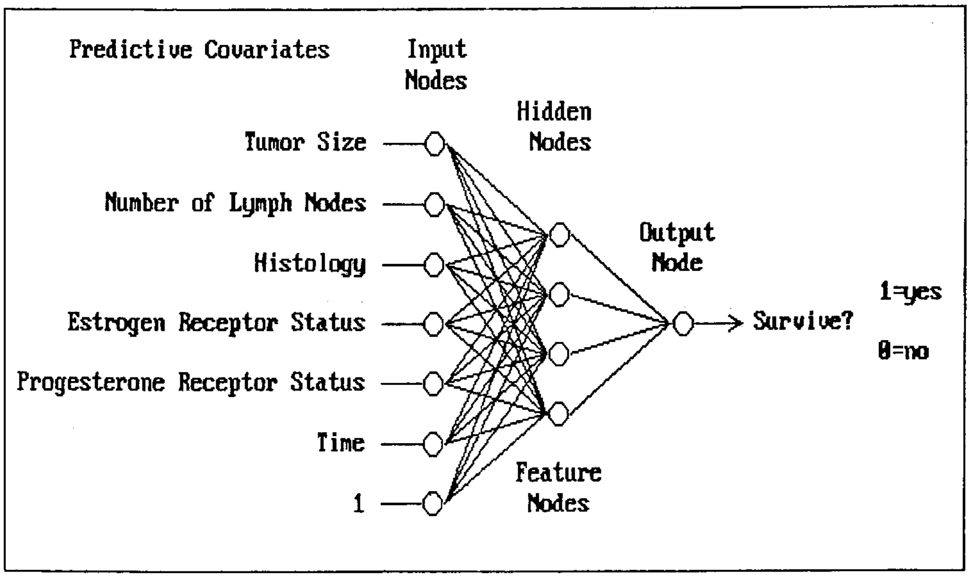

FIGURE 16. Survival neural network for breast carcinoma. This network was trained to predict a 5-year survival outcome.

at a particular point in time. The second example illustrates the use of several putative predictive covariates in a single hidden layer MLP to diagnose the presence of prostate carcinoma (Fig. 5). The third example illustrates the kind of plot possible from sets of neural networks trained to predict coronary heart disease risk from established risk factors with prediction intervals at various points in time (Fig. 17).

SUMMARY: THE ROLE OF NEURAL NETWORKS

Biologically inspired artificial neural networks are highly robust multifactorial mathematic models that have been applied successfully in the prediction, classification, function estimation, and pattern recognition and completion problems in many disciplines, including medicine. Because all these problems can be formulated as function estimation problems, and because there now is theoretic evidence that neural networks can approximate any function with any level of accuracy, these computing paradigms, when used properly and with understanding, have the potential to be highly effective in practically any discipline. The objective of this article was to address the need to bring to the medical community an increased understanding of artificial neural networks so that these robust computing paradigms may be used even more successfully in future medical applications.

Many medical decisions are made in which multiple factors must be weighed. We explored the problems that can arise when multiple factors are used in tandem (one-at-a-time) rather then simultaneously, as they are with neural networks. Neural network outputs could be thought of as "composite variables" formed by nonlinear combinations of the originally input independent variables. We discussed how existing large medical databases constructed originally for recordkeeping purposes now can be explored with neural network and related methodologies for many "data discovery" purposes. If limited success is encountered in such attempts, this could motivate a search for supplementary parameters to improve models. Multivariate methods such as neural networks reportedly have produced better results than human experts in certain studies. Neural networks also have been shown to extract new medical information from raw data.

Computer modeling of medical expertise with these methods allows for accurate analyses and the rapid distribution of local knowledge specific to a patient population in a particular medical environment, as well as global knowledge when the techniques can be applied successfully to larger patient populations. We have shown an example of a multifactorial neural network for the detection of prostate carcinoma and for use in managing the individual patient. This suggests a qualitatively different approach (a paradigm shift) compared with previous "bin-model" methods in which individual patients are treated based on the statistical groups in which they fit (e.g., The American Joint Committee on Cancer staging system). With neural networks and similar computational paradigms we can make predictions for individual patients and mediate (adjust) these predictions by applying ROC methodology to incorporate prevalence and misclassification costs.

We looked inside the black box and observed that artificial neural networks are inspired by models of living neurons, that the artificial neuron is a simple

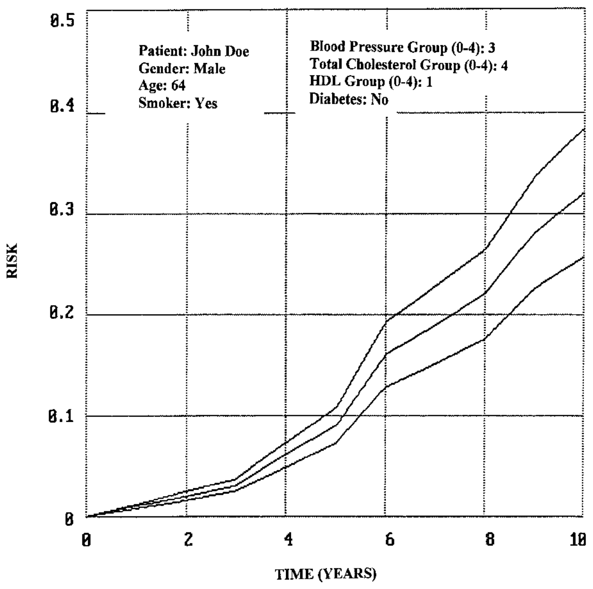

FIGURE 17. Coronary heart disease risk-time plot with 90% confidence intervals. This is a simulation illustrating the type of results that can be obtained with a compendium of trained neural networks. HDL: high density lipoprotein.

nonlinear summing device, and that the neural network is a richly interconnected arrangement of these simple computing nodes. We briefly reviewed the history of the development of a particular class of neural networks based on the back error propagation process leading up to the popular MLP, for which we presented all relevant equations. We cited the important reference to Hornik et al.,[10] which offers the assurance that a two-layered neural network can approximate any nonlinear multifactorial function. Unlike more conventional techniques, with neural networks the function being fit changes as the fit is taking place; thus the neural network technique is very robust. We also explained how a neural network draws out features in the hidden layer and weights these features in the output layer. We showed that any classification problem can be subsumed as a function approximation problem, thereby assuring that the general function approximation theorem by Hornik et al.[10] applies to classification problems as well.

We also discussed measurements of performance and reliability, specifically MSE, percent-correct, area under the ROC plot, and comparison with human experts. We then described how neural networks usually are developed with a training and testing (validation) set, respectively, to conduct and stop training, and a verification dataset to provide an unbiased benchmark of performance.

Because ROC methodology has a long association with single-factor classification problems in medicine, it should be obvious that it could be applied to multifactorial classification problems. When these problems are approached with neural networks or any other multifactorial methodology, they can be thought of as producing a "composite index" that can be subjected to ROC methodology. We briefly traced the history of ROC methodology, including its early use in medical applications by Lusted[38,39] and its "fuzzification" by DeLeo and Campbell.[43,44] We presented the fundamentals of ROC methodology and showed some examples relating to prostate carcinoma patients.

The ROC plot area is a measure of "reparability" and can be used to monitor neural network training. A point on the ROC plot corresponds to an output value of the machine-learning paradigm as well as the sensitivity, specificity, and likelihood that the event (patient) associated with that point is a member of a particular one of the two classes associated with an output node. The output for a given event can be viewed as a probability or fuzzy degree of membership that can be mediated (adjusted) by prevalence and misclassification cost values, as stated earlier.

We mentioned that production systems employed after proper training, testing, and verification must monitor new input cases to make sure they are contained within a reasonable range of the original data. We then suggested that bootstrapping techniques can be used to produce unbiased nonparametric measures of both central tendency and confidence intervals, and we provided classification problem examples related to cancer remission and coronary heart disease risk, as well as a function-fitting problem related to dosing with levothyroxine. In all cases, we recommend using the median value and associating reliability with the confidence interval widths.

Finally we discussed "smart computing" and "smart engineering," new terms describing systems built with components such as neural networks, either alone or as hybrids. We suggested that the time is right for the serious development of "smart engineered" medical systems based on "smart computing" methodologies. We listed features of problems that are suitable for these methods and we emphasized the need to be very careful concerning the following issues: data collection; data types and scaling; prevalence and misclassification costs; training, testing, and validation schema; performance metrics; treatment of missing and censored data; and prediction and confidence intervals. We then ended with illustrations of neural networks for breast carcinoma survival prediction, prostate carcinoma detection, and coronary heart disease risk prediction.

DISCUSSION AND CONCLUSIONS

Neural networks currently are being used actively in many fields. Successful applications of neural networks include military target recognition, financial decision support, detection of manufacturing defects, machine monitoring, machine diagnosis, robotics, and control systems. Success in the medical field is being demonstrated in the areas of diagnostic applications, image processing and analysis, instrument monitoring, drug discovery, and bioinformatics. Some applications have been the subject of published research only, whereas others have moved into commercialization. Biomedical scientists and other health care workers who now are interested in applying neural networks to medical decision support systems have the advantage of tapping into the wealth of experience gained from numerous diverse and previously successful applications.

Neural networks now are well established as viable, multipurpose, computational methodologies. Over the past 40 years, extensive research has been performed by a multidisciplinary community of researchers, including scientists, engineers, mathematicians, and psychologists. This research has addressed the design and development of neural networks; covered the invention of new computational paradigms, languages, and architectures; and encompassed the development of theory and statistics that can be used to understand and benchmark the methodology for specific applications. The perceptron architecture was invented first, without the benefit of any theory to support it. Its pattern recognition capability also was demonstrated in applied problems before any theory that explained why it worked was developed, and without any consideration given to its computational limitations. After decades of further work, the limitations of the original perceptron have been addressed, and we now have multilayer perceptrons with proven general computational abilities along with analysis and theoretic results that demonstrate why these computations work.

In addition, we can employ established statistical methods including ROC methodology and confidence intervals to measure, report, and benchmark neural network performance. The application of neural networks to the field of medicine is a newly emerging phenomena. For efforts in this area to be truly successful, we believe that three major obstacles must be overcome. The first has to do with the reluctance to change to the new, qualitatively different way of thinking that arises when applying neural networks and other multifactorial models to decision formulation. Predicting outcomes for individual patients is very different from predicting outcomes for groups of patients. The effects of this paradigm shift must be considered carefully in terms of immediate personal impact on the patients as well as with respect to the new liability issues likely to arise. The second obstacle is that building appropriate databases is a substantial task that has only just begun. Databases must be expanded and data collection must be held to appropriate standards for the most beneficial results. A third obstacle is the difficulty in establishing good multidisciplinary teams that include physicians and other health care delivery personnel, database experts, and experts on neural networks, statistics, and computational methodology. We believe that all these obstacles can be overcome, and that the successful use of neural networks and multivariate analysis in the field of medicine now is highly possible.

We believe there is a powerful coincidence in the fact that neural network methodology now is highly developed, at the same time that medical data are readily available, with databases rapidly expanding. New ways of thinking are emerging through the work of multidisciplinary teams, providing results from neural network and other multivariate methods. Neural networks clearly are beginning to play an important and hopeful emerging role in medical decision support systems.

REFERENCES

1. Rosenblatt F. Principles in neurodynamics. Washington, DC: Spartan Books, 1962.
2. Widrow B, Hoff M. Adaptive switching circuits. August IRE WESCON Convention Record, Part 4: 1960:96–104.
3. Widrow B, Stearns SD. Adaptive signal processing. Englewood Cliffs, NJ: Prentice-Hall, 1985.
4. Werbos PJ. Beyond regression: new tools for prediction and analysis in the behavioral sciences. [Ph.D. thesis]. Cambridge, (MA): Harvard Univ., 1974.
5. Rumelhart DE, McClelland JL. Parallel distributed processing. Vols. 1 and 2. Cambridge, MA: MIT Press, 1986.
6. Dayhoff JE. Neural network architectures: an introduction. New York: Van Nostrand Reinhold, 1990.
7. Haykim S. Neural networks: a comprehensive foundation. New York: Macmillan College Publishing Company and Maxwell Macmillan International, 1994.
8. Mehrotra K, Mohan CK, Ranka S. Elements of artificial neural networks. Cambridge, MA: A Bradford Book, MIT Press, 1997.
9. Levin DS. Introduction to neural and cognitive modeling. Hillsdale NJ: Lawrence Erlbaum Associates, 1990.
10. Hornik K, Stinchcomb X, White X. Multilayer feedforward networks are universal approximators. *Neural Net* 1989;2: 359–66.
11. Lin DT, Dayhoff JE, Ligomenides PA. Adaptive time-delay neural network for temporal correlation and prediction. Presented in: SPIE Intelligent Robots and Computer Vision XI: biological, neural net, and 3-D methods, 1992;1826:170–81.

12. Lin DT, Dayhoff JE, Ligomenides PA. Learning spatiotemporal topology using an adaptive time-delay neural network. Presented at the World Congress on Neural Networks (WCNN-93), Portland, Oregon. 1993;1:291–4.
13. Lin DT, Dayhoff JE, Ligomenides PA. Spatiotemporal topology and temporal sequence identification with an adaptive time-delay neural network. Presented in: SPIE Intelligent Robots and Computer Vision XII: algorithms and techniques. 1993;2055:536–45.
14. Lin DT, Dayhoff JE, Ligomenides PA. Learning spatiotemporal topology using an adaptive time-delay neural network. Presented at: World Congress on Neural Networks (WCNN-94) II, 1994:231–6.
15. Dagli C, Schierholt K. Stock market prediction using different neural network classification architectures. In: Proceedings of the Conference on Computational Intelligence in Financial Engineering (CIFEr). New York: IEEE Press, 1996.
16. Bentz Y, Boone L, Connor J. Modeling stock return sensitivities to economic factors with the Kalman filter and neural networks. In: Proceedings of the Conference on Computational Intelligence in Financial Engineering (CIFEr). New York: IEEE Press, 1996.
17. Brauner E, Dayhoff JE. Neural network training techniques for a gold trading model. In: Proceedings of the Conference on Computational Intelligence for Financial Engineering (CIFEr). New York: IEEE Press, 1997:000–000.
18. Loch T, Leuschner I, Genberg C, Weichert-Jacobsen K, Kuppers F, Yfantis E, et al. Artificial neural network analysis (ANNA) of prostatic transrectal ultrasound. *Prostate* 1999;39: 198–204.
19. Box GEP, Hunter WG, Hunter JS. Statistics for experimenters. New York: John Wiley & Sons, 1978.
20. Burke HB, Goodman PH, Rosen DB, Henson DE, Weinstein JN, Harrell FE, et al. Artificial neural networks improve the accuracy of cancer survival prediction. *Cancer* 1997;79(4): 857–62.
21. Snow PB, Smith DS, Catalona WJ. Artificial neural networks in the diagnosis and prognosis of prostate cancer: a pilot study. *J Urol* 1994;152:1923–6.
22. Anderson JA. A simple neural network generating an interactive memory. *Math Biosci* 1972;14:197–220.
23. Amari SI. Neural theory of association and concept-formation. *Biol Cybern* 1977;26:175–85.
24. Grossberg S. Adaptive pattern classification and universal recording. I. Parallel development and coding of neural feature detectors. *Biol Cybern* 1976;23:121–34.
25. Kohonen T. Self-organization and associative memory. 2nd ed. New York: Springer Verlag, 1988.
26. Fukushima K, Miyake S, Ito T. Neocognitron: a neural network model for a mechanism of visual pattern recognition. IEEE Trans. Systems, man, and cybernetics SMC-13, 1983: 826–34.
27. Anderson JA, Rosenfeld E. Neurocomputing: foundations of research. Cambridge, MA: MIT Press, 1988.
28. Press WH, Flannery BP, Teukolsky SA, Vetterling WT. Numerical recipes in C. New York: Cambridge University Press, 1988.
29. Fahlman SE. An empirical study of learning speed in backpropagation networks. Technical Rep. CMU-CS-88-162. Pittsburgh: Carnegie-Mellon University, 1988.
30. Park J, Sandberg IW. Universal approximation using radial-basis function networks. *Neural Comput* 1993;3:246–57.
31. Park J, Sandberg IW. Approximation and radial-basis-function networks. *Neural Comput* 1993;5:305–16.
32. Chen CT, Chang WD. A feedforward neural network with function shape autotuning. Neural Net 1996;9(4):627–41.
33. Xu S, Zhang M. Adaptive higher-order feedforward neural networks. In: Proceedings of International Joint Conference on Neural Networks (IJCNN99). New York: IEEE Press, 1999:69.
34. Hornik K. Some new results on neural network approximation. *Neural Net* 1993;6:1069–72.
35. Chen T, Chen H. Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems. *Neural Net* 1995;6(4):911–7.
36. Cybenko G. Approximation by superposition of a signoidal function. *Math Control Signals Syst* 1989;2:303–14.
37. Green DM, Swets JA. Signal detection theory and psychophysics. New York: John Wiley & Sons, Inc., 1966.
38. Lusted LB. Decision making studies in patient management. *N Engl J Med* 1971;284:416–24.
39. Lusted LB. ROC recollected [Editorial]. *Med Decis Making* 1984;4:131–4.
40. Zweig M, Campbell G. Receiver operating characteristic (ROC) curves. A fundamental tool in clinical medicine. *Clin Chem* 1993;39:561–77.
41. Heffner JE, Brown LK, Barbieri C, DeLeo JM. Pleural fluid chemical analysis in parapneumonic effusion. A meta-analysis. *Am J Respir Crit Care Med* 1995;151(6):1700–8.
42. Heffner JE, Brown LK, Barbieri CA, Harpel KS, DeLeo J. Prospective validation of an acute respiratory distress syndrome predictive score. *Am J Respir Crit Care Med* 1995; 152(5 Pt 1):1518–26.
43. DeLeo JM, Campbell G. Receiver operating characteristic (ROC) methodology in artificial neural networks with biomedical applications. In: Proceedings of the World Congress on Neural Networks. Sponsored by the International Neural Network Society, Washington, DC, July 17–21, 1995. Volume 2. Hillsdale, NJ: Lawrence Erlbaum Associates, 1995:735–9.
44. DeLeo JM, Campbell G. The fuzzy receiver operating characteristic function and medical decisions with uncertainty. In: Ayyub BM, editor. Proceedings of the First International Symposium on Uncertainty Modeling and Analysis. New York: IEEE Computer Society Press, 1990:694–9.
45. Zadeh L. Fuzzy sets. *Information Control* 1965:8338–53.
46. Adlassnig KP. A fuzzy logical model of computer-assisted medical diagnosis. *Methods Inf Med* 1980;19:141–8.
47. Campbell G, DeLeo JM. Fundamentals of fuzzy receiver operating characteristic (ROC) functions. Computing science and statistics. In: Malone L, Berk L, editors. Proceedings of the Twenty-First Symposium on the Interface. Alexandria, VA: American Statistical Association, 1989:543–8.
48. DeLeo JM. The fuzzy receiver operating characteristic function and medical decision making with uncertainty. In: Ayyub BM, editor. Proceedings of the First International Symposium on Uncertainty Modeling and Analysis. New York: IEEE Computer Society Press, 1990:694–9.
49. DeLeo JM. A neural network approach to cancer patient survival estimation. Computer Applications for Early Detection and Staging of Cancer Workshop, Bethesda, MD: National Cancer Institute/American Joint Committee on Cancer, National Institutes of Health 1993.
50. DeLeo JM. Receiver operating characteristic function as a tool for uncertainty management in artificial neural network decision-making. In: Ayyub BM, editor. Proceedings of the Second International Symposium on Uncertainty Modeling and Analysis. New York: IEEE Computer Society Press, 1993: 141–5.

51. DeLeo JM. Receiver operating characteristic laboratory (ROCLAB); software for developing decision strategies that account for uncertainty. In: Ayyub BM, editor. Proceedings of the Second International Symposium on Uncertainty Modeling and Analysis. New York: IEEE Computer Society Press, 1993:318–25.
52. Efron B. The jackknife, the bootstrap, and other resampling plans. Philadelphia: Society for Industrial and Applied Mathematics, 1982.
53. Efron B, Tibshirani RJ. An introduction to the bootstrap. New York: Chapman & Hall, International Thomson Publishing, 1993.
54. Campbell G, DeLeo JM. Bootstrapping ROC plots. Computing science and statistics. In: Meyer XX, Rosenberger XX, editors. Proceedings of the Twenty-Seventh Symposium on the Interface. Alexandria, VA: American Statistical Association, 1996:420–4.
55. Remaley AT, Sampson ML, DeLeo JM, Remaley NA, Farsi BD, Zweig MH. Prevalence-value-accuracy plots: a new method for comparing diagnostic tests based on misclassification costs. *Clin Chem* 1999;45(7):934–41.
56. DeLeo J. Smart computing. ADVANCE, March 2000; 97–8.
57. Litvan I, DeLeo JM, Hauw JJ, Daniel SE, Jellinger K, McKee A, et al. What can artificial neural networks teach us about neurodegenerative disorders with extrapyramidal features? *Brain* 1996;119:831–9.
58. Carney JG, Cunningham P, Bhagwan U. Confidence and prediction intervals for neural network ensembles. Proceedings of the International Joint Conference on Neural Networks. New York: IEEE Press, 1999.
59. Cooper LN. A possible organization of animal memory and learning. In: Lundquist B, Lundquist S, editors. Proceedings of the Nobel Symposium on Collective Properties of Physical Systems. New York: Academic Press, 1973:252–64.
60. Reggia JA. Neural computation in medicine [review]. *Artif Intell Med* 1993;5:143–57.