INDUSTRIAL MATHEMATICS,
The Journal of the Industrial Mathematics Society, Volume 45, Part 1, 1995

Application of Neural Network Classifiers to Disease Diagnosis

Murali Shanker
Department of ADMS, College of Business
Kent State University
Kent, OH, U.S.A. 44242-0001

ABSTRACT

Classification is an essential decision making tool, especially for the diagnosis of diseases. Unfortunately, while many classification procedures exist, many of the methods suffer in the presence of statistical outliers or overlapping groups. Recently, artificial neural networks have been suggested as tools for classification. In this paper, we determine the suitability of neural network classifiers in the diagnosis of thyrotoxicosis. Experimental and clinical data are used for classification. We found that neural networks provided consistently good classification results.

I. INTRODUCTION

Classification has emerged as an important decision making tool. It has been used for a variety of applications including credit scoring (Capon 1982), prediction of events like credit card usage (Awh and Waters 1974) and tender offer outcomes (Walking 1985), and as a tool in medical diagnosis (Booth and Isenhour 1986; Crooks, Murray, et al. 1959). Unfortunately, while several classification procedures exist (Patuwo, Hu, et al. 1993), many of the current methods do not work well in the presence of statistical outliers or overlapping groups. For example, most least-squares based procedures are affected by outliers (Booth and Isenhour 1986; Hogg 1979).

In recent years, artificial neural networks (ANNs) have been suggested as an alternative tool for classification (Denton, Hung, et al. 1990; Huang and Lippmann 1987). The idea of neural computing grew out of a desire to capture pattern recognition capabilities of a biological brain. McCulloch and Pitts (1943) developed the first model of a physiological brain called "McCulloch-Pitts Neuron," which became the basis for almost all artificial neural networks where nodes are likened to neurons and arcs to dendrites or axons. Now, ANNs have been developed for recognition of such ill-defined objects as handwritten characters (Le Cun, Boser, et al. 1990; Martin and Pittman

1990), finger prints (Leung, Engeler, et al. 1990), and double spirals (Lang and Witbrock 1988). They also have been developed for detection of faults in a chemical process (Hoskins, Kaliyur, et al. 1990), explosives in airline baggage (Shea and Liu 1990), and prediction of bank failures (Tam and Kiang 1992). ANNs, unlike traditional classifiers like linear discriminant analysis (LDA) and quadratic discriminant analysis (QDA), are nonparametric and are able to adjust the form of the discrimination to fit the data. As ANNs can approximate, arbitrarily closely, any mapping function (Lippmann 1987), they might prove to be useful classification tools.

In this paper, we evaluate the effectiveness of ANNs classifiers on datasets involving the diagnosis of thyrotoxicosis. Specifically, we use the datasets considered by Hu et al. (1989) in their comparison of five classifiers, including three linear-programming classifiers. In each case, the data consisted of statistical outliers and group overlap, and hence generally proved difficult to classify.

The rest of the paper is organized as follows. The next section reviews artificial neural networks. Section III describes the data and presents the results, while conclusions are given in Section IV.

II. ARTIFICIAL NEURAL NETWORKS

Let $G = (N,A)$ denote a neural network where N is the node set and A the arc set containing only directed arcs. G is assumed to be acyclic in that it contains no directed circuit. The node set N is partitioned into three subsets: N_I, N_O, and N_H. N_I is the set of input nodes, N_O is that of output nodes, and N_H that of hidden nodes. In a popular form called the multilayer perceptron, all input nodes are in one layer, the output nodes in another layer, and the hidden nodes are distributed into several layers in between. The knowledge learned by a network is stored in the arcs and nodes, in the form of arc weights and node values called biases. We will use the term k-layered network to mean a layered network with k-2 hidden layers. Figure 2 shows a three-layered network.

When a pattern is presented to the network, the variables of the pattern activate some of the neurons (nodes). Let x_i^p and a_i^p be, respectively, the input and the activation value at node i corresponding to pattern p. If $i \in N_I$, then x_i^p is the value of the ith variable in the input pattern p and $a_i^p = x_i^p$. If i is not an input node, then the two variables are defined as follows:

$$x_i^p = \theta_i + \sum_{k \in N_i} w_{ki} a_k^p \tag{1}$$

and

$$a_i^p = F(x_i^p),$$

where N_i denotes the set of nodes sending signals to node i, w_{ki} is the weight of arc (k,i), and θ_i is the bias. F is called the activation function, and it is typically a "squashing" function that normalizes the input signals so that the activation value is between 0 and 1. The most popular choice for F is the logistic function, defined by

$$F(x) = (1 + e^{-x})^{-1}. \tag{2}$$

Since the network G is acyclic, computation can proceed from the input nodes, one layer at a time, toward the output node until the activation values of all the output nodes are determined.

To simplify notation, we shall define for each node i an extra arc $(0,i)$ such that $w_{0i} = \theta_i$. Let $\mathbf{w}$ denote the vector of all weights. This vector is determined by "training" the network with a set of training patterns. For pattern p, let E^p be defined by

$$E^p = \sum_{i \in N_o} \left| a_i^p - t_i^p \right|^l, \tag{3}$$

where t_i^p is the target for node i and l is a non-negative real number. Then training refers to the determination of $\mathbf{w}$ such that $\sum_p E^p$ is minimized.

The efficiency of solving this unconstrained minimization problem arises from the fact that the network is acyclic and thus the partial derivatives can be computed exactly in an iterative fashion. Consider the derivative for one input pattern p.

$$\frac{\partial E^p}{\partial w_{ki}} = \frac{\partial E^p}{\partial x_i^p} \frac{\partial x_i^p}{\partial w_{ki}} = \delta_i^p a_k^p. \tag{4}$$

The last equation comes from equation (1). The term δ_i^p is defined as the rate of change of the pattern error with respect to the input to node i and it can be computed as

$$\delta_i^p = \frac{\partial E^p}{\partial x_i^p} = \frac{\partial E^p}{\partial a_i^p} \frac{\partial a_i^p}{\partial x_i^p} \overset{\text{d}}{=} \frac{\partial E^p}{\partial a_i^p} \bar{F}_i^p, \tag{5}$$

where

$$\bar{F}_i^p = a_i^p (1 - a_i^p),$$

if F is the logistic function.

It can be seen that the only quantity that is dependent on the network structure is the partial derivative of E^p with respect to the activation value a_i^p, and it will be computed recursively. In the following, we assume that l in (3) is an integer and greater than 1.

$$\frac{\partial E^p}{\partial a_i^p} = \begin{cases} l(a_i^p - t_i^p)^{l-1} & \text{if } i \in N_O \text{ and } l \text{ even,} \\ \pm l(a_i^p - t_i^p)^{l-1}, & \text{if } i \in N_O \text{ and } l \text{ odd,} \\ \sum_{j \in T_i} \frac{\partial E^p}{\partial x_j^p} \frac{\partial x_j^p}{\partial a_i^p} = \sum \delta_j^p w_{ij}, & \text{otherwise,} \end{cases} \quad (6)$$

where the sign when l is odd is equal to the sign of $a_i^p - t_i^p$, and T_i denotes the set of nodes receiving signals from i. The derivative of E^p with general l can be established, but, since functions of this type are rarely used, we will not develop it here.

The process for computing the partial derivatives is as follows. For each of the output nodes, use equations (5) and (6) to compute δ_i^p. Then, for each arc going into an output node, use equation (4) to find the partial derivative with respect to its weight. Since the network is acyclic, there must be a node i where T_i consists of output nodes only. Compute δ_i^p using equations (5) and (6). (If the network is layered, then all the nodes in the layer below the output layer can have their δ_i^p computed.) Then, for the arcs coming into node i, use equation (4) to compute the partial derivatives. Repeat until all partial derivatives are computed.

Back-propagation, developed by Werbos (1974) and popularized by Rumelhart (1986), is also known as Generalized Delta Rule and is applicable when the objective function is that of least squares, i.e., $l = 2$ in E^p, and the network is layered. In most implementations, it is also assumed that the activation function is logistic. The weight vector **w** is updated as follows:

$$w \leftarrow w + \eta \Delta w,$$

where η is often referred to as the learning rate and Δw is defined by

$$\Delta w = \{\Delta w_{ki}\},$$

where

$$\Delta w_{ki} = \sum_p \delta_i^p a_k^p.$$

So it can be seen that Δw is the steepest-descent direction, and therefore the back-propagation is a steepest-descent method with fixed step size η. A complete update of all the weights is called an *epoch*.

The computational experience with back-propagation has been mixed. On one hand, it has been used in most feed-forward networks, including the examples cited in the introductory section. On the other hand, stories of no convergence or slow convergence have been reported (Rumelhart, Hinton, et al. 1986; Tesauro and Janssens 1988; Wasserman 1989).

Recognizing that the wealth of theory and algorithms of nonlinear optimization could be useful for neural-network training, Subramanian and

Hung (1993) developed the GRG2-based procedure for training neural networks. GRG2 is a widely distributed nonlinear optimization system (Lasdon and Waren 1986), and it has been shown to be particularly effective for problems where the objective and constraint functions are highly nonlinear. As the objective function in neural-network training is highly nonlinear, a GRG2-based procedure is expected to do well. Another reason for using GRG2 is that it is sometimes useful to put bounds on the variables **w**, for example, when the network contains many local minima. Bounds are handled implicitly in GRG2 and do not degrade the performance of the algorithm.

Reduced gradient refers to the gradient vector for a subset of variables that are free to change without violating the constraints. For an unconstrained optimization problem, the reduced gradient and the gradient are the same. In GRG2, the first search direction is always the steepest-descent search direction. After that, either the BFGS formula (for Boyden-Fletcher-Goldfarb-Shanno (see Gill, Murray, et al. 1981)) or any one of four conjugate gradient methods is used. Once the search direction is determined, a line search is carried out to find the optimal step size. As in most effective codes, GRG2 performs the line search only approximately. Sophisticated rules for restarting the search direction with the steepest descent and for accepting a line search are used to ensure convergence and efficient computation. As explained in Subramanian and Hung (1993), GRG2 includes most techniques that have been suggested for the improvement of the back-propagation algorithm, and therefore a training procedure based on GRG2 should be competitive with back-propagation in terms of quality of solution and computational efficiency. Their results (Subramanian and Hung 1993) support this assertion. For the training of neural networks in this research, we use the GRG2-based procedure.

The next section presents the data and the experimental analysis.

III. EXPERIMENTAL ANALYSIS

The datasets used in this study are those considered by Hu et al. (1989). They consist of the following:

- the classification of clinical patients as having or not having thyrotoxicosis, based on a set of observable signs and symptoms and the results of 4-hr glandular uptake studies with radioiodine (Table I),
- a set of simulated data by Freed and Glover (1986) modified to include overlap between groups (Table II).

The datasets are shown in Tables I and II. Clearly, in both cases, there is an overlap between groups. In addition, observation #6 in Table II is a statis-

tical outlier for class 2. For both datasets, it appears that a nonlinear mapping function would provide the best classification.

Hu et al. (1989) test the performance of 5 classifiers on the above datasets. The five classifiers included linear discriminant analysis (LDA), robust partial discriminant analysis (RPDA), and three linear-programming classifiers: minimize the maximum deviation (MMD), minimize the sum of interior distances (MSID), and minimize the sum of deviations (MSD). In contrast to LDA, which can be influenced by outliers, RPDA was proposed as an outlier-resistant procedure (Booth and Isenhour 1986). Research on linear-programming classifiers indicates that the most promising model is MSD (Hu, Fisher, et al. 1989; Freed and Glover 1986), with performance comparable to other parametric and non-parametric classifiers (Patuwo, Hu, et al. 1993). For the datasets in Tables I and II, Hu et al. (1986) found that MSD produced the best classification.

Table I. Thyrotoxicosis Data

Observation	Sign and Symptom Index	Glandular Uptake	Class
1	11	29.0	1
2	11	62.4	1
3	12	45.0	1
4	12	35.0	1
5	13	40.0	1
6	13	17.3	1
7	14	84.9	2
8	16	65.8	2
9	17	50.5	2
10	18	18.5	1
11	18	62.3	2
12	19	71.0	2
13	19	65.0	2
14	27	41.7	1

Table II. Freed and Glover Data

Observation	X_1	X_2	Class
1	–1	3	2
2	–1	4	2
3	1	3	2
4	4	9	2
5	2	7	2
6	1	1	2
7	0	2.5	1
8	2	3	1
9	–4	–9	1
10	3	4	1
11	–1	–3	1

Before neural networks can be employed for classification, it is necessary to determine the appropriate architecture to use for a dataset. Network architecture refers to the number of layers, the number of nodes in each layer, and the number of arcs and nodes they connect. Other network design decisions include the choice of activation functions, and whether to include biases or not. All neural networks considered for this study have either zero or one hidden layer. Node biases occur only in output nodes, and the activation function used is the logistic function mentioned earlier. (An example of a typical neural network used in our study is shown in Figure 2.) In neural network literature, a popular choice for the number of hidden nodes is $2n+1$, where n is the number of input variables. For the dataset in Table I, we would have two input variables, one to represent sign and symptom index, and the other for glandular uptake. Thus the above rule would indicate a network with 5 hidden nodes. Our research on neural networks has suggested that in many instances a smaller network would suffice to achieve good classification.

The final neural network architecture used and the computation time for classification on an RS6000 workstation is shown in Table III. Classification results, presented as a comparison between neural network (NN) and other classifiers, are shown in Tables IV and V.

The results in Tables IV and V indicate that NN performed as well or better than other classifiers. In addition, overlapping or outlying points (e.g., #6 in Table II), were correctly classified without difficulty. But, for the thyrotoxico-

Table III. Neural Network Architecture and Computation Time

Data	Neural Network Architecture: Input Nodes	Hidden Nodes	Output Nodes	Computation Time (Milliseconds)
Thyrotoxicosis	2	0	1	6
Freed and Glover	2	2	1	27

Table IV. Classification Results: Thyrotoxicosis Data

	LDA	RPDA	MMD	MSID	MSD	NN
Misclassified	8, 14	14	3, 8, 14	3, 8, 14	None	9
Not classified	N/A	3, 8	N/A	N/A	N/A	N/A

Table V: Classification Results: Freed and Glover Data

	LDA	RPDA	MMD	MSID	MSD	NN
Misclassified	6, 8	6, 7	1, 2, 3, 6, 8, 10	1, 2, 3, 6, 8, 10	6	None
Not classified	N/A	3, 8, 10	N/A	N/A	N/A	N/A

sis data in Table I, observation #9 was incorrectly classified. For this dataset, the neural network architecture has zero hidden nodes, therefore the output from this network can be represented by $F(x) = (1 + e^{-(\theta_3 + w_{13}x_1 + w_{23}x_2)})^{-1}$, where θ_3 is the bias at node 3 (output node), x_1 and x_2 the input at nodes 1 and 2 (input nodes), respectively, and w_{ij} the arc weight between nodes i and j. As we use logistic activation function, the final network output will be between 0 and 1. In our classifier, all output values below 0.5 are classified as class 1, with the rest classified as class 2. This separation line is shown in Figure 1, where points above the separation line are classified as class 2 points, while below the separation line the points are classified into class 1. As the neural network used in this particular case is equivalent to logistic regression, higher orders of neural network would be required to correctly classify #9. In any case, the neural network classifiers were not only good classifiers for the chosen datasets, but also computationally efficient (Table III).

IV. CONCLUSIONS

Clearly, neural networks can serve as an important tool for classification. Furthermore, unlike other classifiers, neural networks have several advantages. Some of them are listed below:

- As neural networks are not so much programmed as they are *trained with data*, neural networks are likely to improve with experience (more data).
- No assumptions need be made about the underlying distribution of data for classification using neural networks. This makes it attractive for real-world problems, where the data are rarely free from errors, and the distributional form is usually unknown.
- As neural networks can approximate any arbitrary function, they should serve as excellent tools for function mapping.
- Neural networks, even for large data sets, can be trained in a relatively short time. As such, a wider range of problems and architectures can be considered for classification.

However, neural networks also have several disadvantages. Some of them are listed below.

- Unlike classifiers like LDA, and like linear-programming classifiers, no statistical inference procedures yet exist for neural networks. As such, statistical testing of parameters cannot yet be performed.
- Previous studies in using neural networks classifiers suggest that they are better than traditional methods and LP models in classifying training samples, but do worse when they are used on "test" (population) data (Patuwo, Hu, et al. 1993). This suggests that care should be taken in using neural-network classifiers for prediction.

In conclusion, neural networks classifiers appear attractive in two respects: generality and flexibility. Flexibility in that neural networks can be easily changed by considering different architectures, and generality in that neural network depends only on the data provided.

ACKNOWLEDGMENTS

We would like to thank an anonymous referee for helpful comments on an earlier version of this paper, and to LCI, Kent State University, for providing computational support.

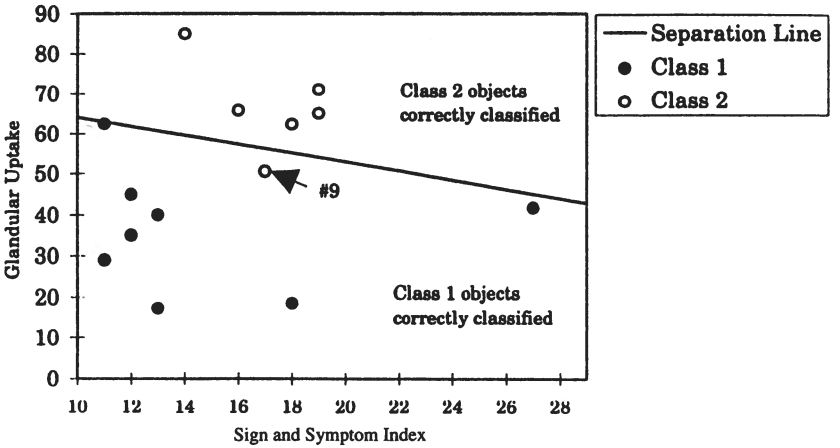
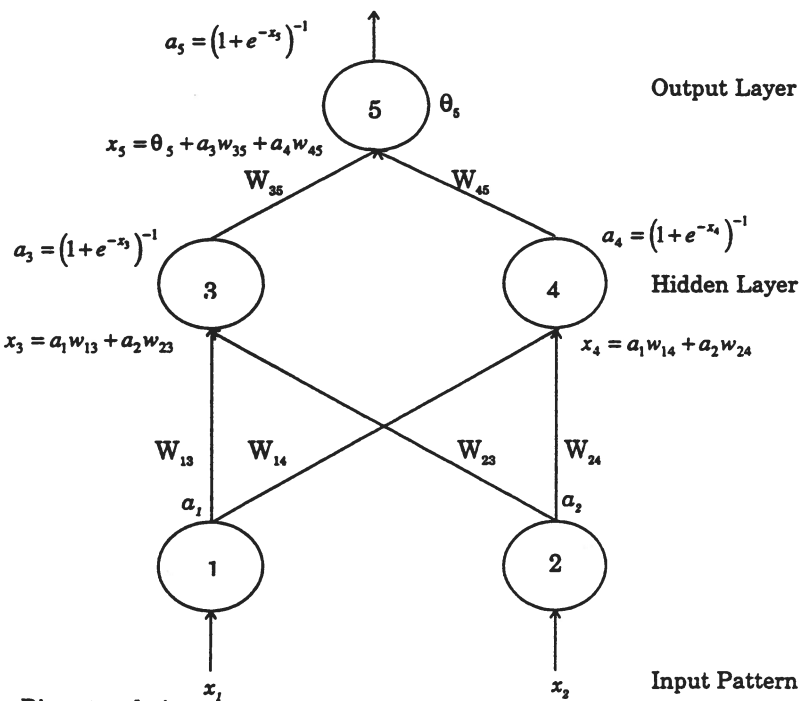


Figure 1. Classification of Thyrotoxicosis Data Using Neural Networks.



θ_i Bias at node i
x_i Input at node i
a_i Output at node i
w_{ij} Arc weight from node i to node j

Figure 2. A 3-Layered Neural Network.

REFERENCES

Awh, R.Y. and D. Waters (1974). "A Discriminant Analysis of Economic, Demographic, and Attitudinal Characteristics of Bank Charge-Card Holders: A Case Study." *Journal of Finance* 29: 973–980.

Booth, D.E. and T.L. Isenhour (1986). "On Robust Partial Discriminant Analysis as a Decision-Making Tool with Clinical and Analytical Chemical Data." *Computers and Biomedical Research* 19: 1–12.

Capon, N. (1982). "Credit Scoring Systems: A Critical Analysis." *Journal of Marketing* 46: 82–91.

Crooks, J., I.P.C. Murray, et al. (1959). "Statistical Methods Applied to the Clinical Diagnosis of Thyrotoxicosis." *Q. J. Med.* 28: 211.

Denton, J.W., M.S. Hung, et al. (1990). "A Neural Network Approach to the Classification Problem." *Expert Systems with Applications* 1: 417–424.

Freed, N. and F. Glover (1986). "Evaluating Alternative Linear Programming Models to Solve the Two-Group Discriminant Problem." *Decision Sciences* 17: 151–162.

Gill, G.E., W. Murray, et al. (1981). *Practical Optimization*. Academic Press, London.

Hogg, R.V. (1979). "Statistical Robustness: One View of Its Use in Applications Today." *American Statistics* 33: 108.

Hoskins, J.C., K.M. Kaliyur, et al. (1990). "Incipient Fault Detection and Diagnosis Using Artificial Neural Networks." *Proceedings of the International Joint Conference on Neural Networks* I: 485–493.

Hu, M., D. Fisher, et al. (1989). "Linear Programming Models as Classification Tools in Disease Diagnosis." *Industrial Mathematics* 39(2): 159–167.

Huang, W.Y. and R.P. Lippmann (1987). "Comparisons Between Neural Net and Conventional Classifiers." *IEEE 1st International Conference on Neural Networks* : 485–493.

Lang, K.J. and M.J. Witbrock (1988). "Learning to Tell Two Spirals Apart." *Proceedings of the 1988 Connectionists Models Summer School* : 52–59.

Lasdon, L.S. and A.D. Waren (1986). *GRG2 User's Guide*, School of Business Administration, University of Texas at Austin, Austin, TX.

Le Cun, Y., B. Boser, et al. (1990). "Handwritten Digit Recognition with a Back-Propagation Network." *Advances in Neural Information Processing Systems* 2: 396–404.

Leung, M.T., W.E. Engeler, et al. (1990). *Fingerprint Processing Using Backpropagation Neural Networks*. Proceedings of the International Joint Conference on Neural Networks I: 15–20.

Lippmann, R.P. (1987). "An Introduction to Computing with Neural Nets." *IEEE ASSP Magazine* 4: 2–22.

Martin, G.L. and J.A. Pittman (1990). "Recognizing Hand-Printed Letters and Digits." *Advances in Neural Information Processing Systems* 2: 405–414.

McCulloch, W.S. and W. Pitts (1943). "A Logic of the Ideas Immanent in Nervous Activity." *Bulletin of Mathematical Biophysics* 5: 115–133.

Patuwo, E., M.Y. Hu, et al. (1993). "Two-Group Classification Using Neural Networks." *Decision Sciences* 24(4): 825–845.

Rumelhart, D.E., G.E. Hinton, et al. (1986). "Learning Internal Representations by Error Propagation." *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, MIT Press, Cambridge, MA.

Shea, P.M. and F. Liu (1990). "Operational Experience with a Neural Network in the Detection of Explosives in Checked Airline Baggage." *Proceedings of the International Joint Conference on Neural Networks* II: 175–178.

Subramanian, V. and M.S. Hung (1993). "A GRG2-Based System for Training Neural Networks: Design and Computational Experience." *ORSA Journal on Computing* 5(4): 386–394.

Tam, K.Y. and M.Y. Kiang (1992). "Managerial Application of Neural Networks: The Case of Bank Failure Predictions." *Management Science* 38(7): 926–947.

Tesauro, G. and B. Janssens (1988). "Scaling Relationships in Back-Propagation Learning." *Complex Systems* 2, 39–44.

Walking R.A. (1985). "Predicting Tender Offer Success: A Logistic Analysis." *Journal of Finance and Quantitative Analysis* 20: 461–478.

Wasserman, P.D. (1989). *Neural Computing: Theory and Practice.* Van Nostrand Reinhold.

Werbos, P. (1974). Beyond Regression, New Tools for Prediction and Analysis in the Behavioral Sciences, Ph.D. thesis, Harvard University, Boston, MA.