NEURAL NETWORKS in Finance and Investing

Applying Neural Networks to the Extraction of Knowledge from Accounting Reports: A Classification Study

R.H. Berry
Duarte Trigueiros

PROBUS PUBLISHING COMPANY
Chicago, Illinois
Cambridge, England

6

APPLYING NEURAL NETWORKS TO THE EXTRACTION OF KNOWLEDGE FROM ACCOUNTING REPORTS: A CLASSIFICATION STUDY

R. H. Berry and Duarte Trigueiros

INTRODUCTION

This study develops a new approach to the problem of extracting meaningful information from samples of accounting reports. Neural networks are shown to be capable of building structures similar to financial ratios, which are optimal in the context of the particular problem being

Printed with permission of the authors.

dealt with. This approach removes the need for an analyst to search for appropriate ratios before model building can begin.

The internal organization of a neural network model helps identify key features of accounting data and provides new insights into the relative importance of variables for particular modeling tasks. The lack of interpretability of neural network parameters so often reported in other applications of the approach is removed in the accounting context. Much of the internal operation of the networks involves the construction of generalizations of the ratio concepts with which accountants are familiar. Thus, traditional modes of understanding can be brought to bear.

ACCOUNTING DECISION MODELS

Accounting reports are an important source of information for managers, investors, and financial analysts. Statistical techniques have often been used to extract information from them. The aim of such exercises is to construct models suitable for predictive or classification purposes, or for isolating key features of the data. Well-known examples include, in the U.S. context, Altman et al[1] and, in the U.K. context, Taffler.[2]

The procedures used in this vast body of literature are generally similar. The first stage consists of forming a set of ratios from selected items in a set of accounting reports. This selection typically is made in accordance with the prior beliefs of researchers. Next, the normality of these ratio variables is examined and transformations applied, where necessary, to bring it about. Finally, some linear modeling technique is used to find optimal parameters in the least-square sense. Linear regression and Fisher's multiple discriminant analysis are the most popular algorithms. However, logistic regression can also be found in some studies. Foster[3] provides a review of the general area of statistical modeling applied to accounting variables.

The widespread use of ratios as input variables is particularly significant in the present context. This seems to be an extension of their normal use in financial statement analysis. However, there is a problem; there are many possible ratios. Consequently, some researchers utilize a large number of ratios as explanatory variables, others use representative ratios, and still others use factor analysis to cope with the mass of ratio variables and their linear dependence.

THE STATISTICAL CHARACTERIZATION OF ACCOUNTING VARIABLES

The statistical distribution of accounting ratios has been the object of considerable study. The common finding is that ratio distributions are skewed. Horrigan[4] in an early work on this subject, reports positive skewness of ratios and explains it as the result of effective lower limits of zero on many ratios. Barnes[5] in a discussion of the link between firm size and ratio values, suggests that skewness of ratios could be the result of deviations from strict proportionality between the numerator and the denominator variables in the ratio. The underlying idea here, that interest should center on the behavior of the component accounting variables, and not on the ratios that they have traditionally been used to form, is basic to the present research.

Mcleay,[6] in one of the few studies of distributions of accounting variables as opposed to ratios of such variables, reports that accounting variables which are sums of similar transactions with the same sign, such as Sales, Stocks, Creditors, or Current Assets, exhibit cross-section lognormality. Empirical work carried out during the current research project confirms this finding and suggests that the phenomenon of lognormality is much more widespread. Many other positive-valued accounting variables have cross-section distributions that are approximately log normal. Furthermore, where variables can take on positive and negative values, then lognormality can be observed in the subset of positive values and also in the absolute values of the negative subset. Size-related nonfinancial variables such as number of employees also seem to exhibit lognormality. Distributional evidence for 18 accounting and other items, for 14 industry groups over a five-year period, can be found in Trigueiros.[7] In this chapter, lognormality is viewed as a universal distributional form for the cross-section behavior of those variables used as inputs to the neural networks that have been built.

The lognormal distribution is characterized by a lower bound of zero and a tail consisting of a few relatively large values. In any statistical analysis of such variables, based on the least-squares criterion, these few large values will dominate coefficient estimates. Consequently, an analyst is well advised to apply the logarithmic transformation to accounting variables that are to be inputs to least-squares-based techniques to counteract this effect. In what follows, logs

of accounting variables will appear in various linear combinations, having the general form:

$$z = a_1\log(x_1) + a_2\log(x_2) - b_1\log(y_1) - b_2\log(y_2) \quad (1)$$

If the logarithmic transformation is reversed, this linear combination is seen to be equivalent to:

$$k = \frac{x_1^{a_1} x_2^{a_2}}{y_1^{b_1} y_2^{b_2}} \quad (2)$$

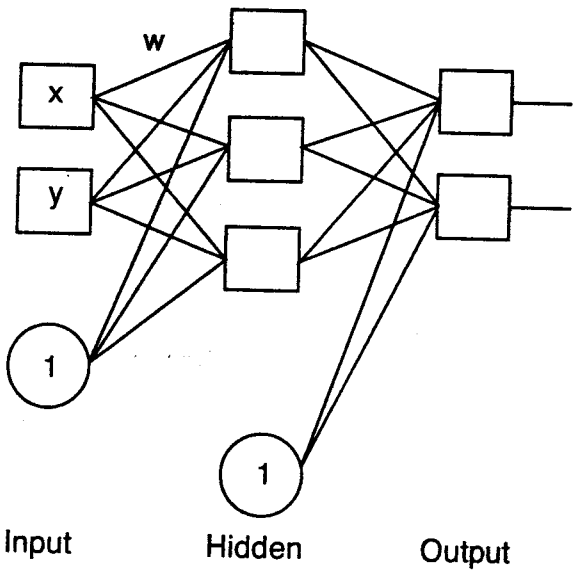
This is a complex ratio form. Had the linear combination been restricted to the difference between two variables, and the coefficients to the value one, then a simple ratio of two variables would have been produced by reversing the logarithmic transformation. The observation that a linear combination, including both positive and negative coefficients, in log space, is equivalent to a ratio form in ordinary space, is fundamental to the interpretation of the neural network coefficients presented in this chapter.

NEURAL NETWORKS

A neural network is a collection of simple computational elements, neurons, that are interconnected. The connections between neurons have weights attached to them. A neuron can receive inputs from other neurons or from sources outside the network, form a weighted combination of these inputs (often called NET), the weights being those assigned to the connections along with the inputs travel, and produce an output (often called OUT) that is sent to other neurons. The output may be simply the weighted combination of inputs, NET, or a nonlinear transformation of NET. This nonlinear transformation is known as a transfer, or squashing, function. The number and pattern of interconnection of the neurons in a network determine the task a network is capable of performing.

The particular network form used in this chapter is known as a multilayer perceptron (MLP). A simple example is shown in Figure 6.1. There are three layers of neurons (each neuron being represented by a rectangle): an input layer, a hidden layer, and an output layer. The

Figure 6.1
A Multilayer Perceptron



neurons in the input layer do not perform weighting or non-linear transformation. They simply send inputs from the world outside the network to the hidden-layer neurons. In Figure 6.1, each input neuron sends its signal to each of the neurons in the hidden-layer. Each of these hidden layer neurons forms a weighted linear combination of the input values and then applies a nonlinear transformation to generate its own output. A common transfer function is the sigmoid, which generates a signal $0 \leq OUT \leq 1$:

$$OUT = \frac{1}{1 + e^{-NET}} \tag{3}$$

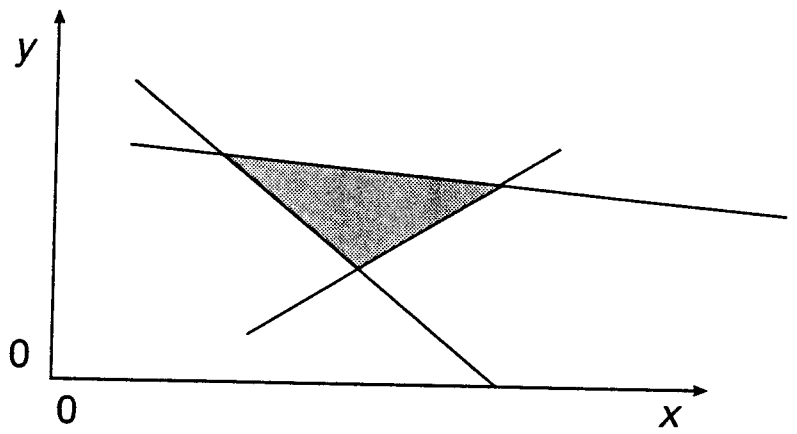
The signals from the hidden-layer neurons are sent to the output layer neurons. In Figure 6.1, each output-layer neuron receives input from each hidden-layer neuron. The neurons in the output layer each form linear combinations of their inputs and apply a nonlinear transforma-

tion before sending their own signals onwards, in their case to the outside world. The sigmoid function again serves as the nonlinear transformation. The circles in Figure 6.1 do not represent neurons. They each send a signal that has a constant value of 1 along weighted connections. This weighted signal becomes part of NET for each receiving neuron. This has the effect of generating a threshold value of NET in each neuron's OUT calculation, above which OUT rises rapidly.

The particular MLP shown in Figure 6.1 is capable of performing a relatively complex classification task, given that appropriate weights have been attached to the interconnections between neurons. Figure 6.2 shows a convex set of (x,y) values. The convex region is formed by the intersection of three half-spaces, each defined by a linear inequality. Each of the three linear relations that define the convex set can be represented by a linear combination of (x,y) values. Thus, each can be represented by one of the three neurons in the hidden layer of the MLP shown in Figure 6.1. Each output-layer neuron then receives over/under signals from the hidden-layer neurons, and, again by weighting and transforming the signals, carries out AND/OR operations to produce an output.

In the network shown in Figure 6.1, one output neuron will produce a value close to 1 if the (x,y) pair being input lies within the convex

Figure 6.2
Classification Problem



region. The other will produce a value close to 0 in these circumstances. The input of an (x,y) pair outside the convex region will cause a reversal of this output pattern. (Given the binary nature of the output signal required, one output neuron could theoretically do the job. However, computational experience shows that economizing on output neurons is a mistake.) In order to model more difficult nonlinear boundaries, additional hidden layers of neurons might have to be added to the network.

The problem left unresolved in the preceding description of the operation of the MLP is, where do the values of the interconnection weights come from? To carry out the classification task appropriately, each interconnection must have an appropriate weight.

The network learns these weights during a training process. To build a network capable of performing a particular classification task, the following actions must be undertaken:

1. The network topology (number of layers and number of neurons in each layer) must be specified.
2. A data set must be collected to allow network training. In the example under discussion, this training set would consist of (x,y) pairs and for each pair, a target value vector (1,0) if the pair lies in the convex region of interest, (0,1) if it does not.
3. Random, small weights are assigned to each interconnection.
4. An (x,y) pair is input to the network.
5. The vector of OUT values from the output neurons is compared with the appropriate target value vector.
6. Any errors are used to revise the interconnection weights.

The training set is processed repeatedly until a measure of network performance based on prediction errors for the whole training set reaches an acceptably low level. Once training has been completed, the network can be used for predictive purposes. There are two styles of training. In the first, weight updating occurs after each individual element of the training set is processed through the MLP. In the second, the entire training set is processed before updating occurs.

The algorithm used to adjust interconnection weights, known as the generalized delta rule, or as the back-propagation method, is usually associated with Rumelhart et al.[8] This algorithm is an enhanced version

of the stochastic gradient-descent optimization procedure. Its virtue is that it is able to propagate deviations backwards through more than one layer of nodes. Thus, it can train networks with one or more hidden layers. For the algorithm to work, the transfer function used in the MLP must be differentiable. Good descriptions of the algorithm exist in several sources, including Pao[9] and Wasserman.[10] Wasserman's approach plays down the mathematics and emphasizes the computational steps.

Minimum least-squares deviation is one possible success criterion that could be used to decide when to curtail the training process. However, there are others such as likelihood maximization. In this case the weights are adjusted to maximize the probability of obtaining the input/output data that constitute the training set.

In general, if the number of nodes in hidden layers is large compared with the number of important features in the data, the MLP behaves just like a storage device. It learns the noise present in the training set, as well as the key structures. No generalization ability can be expected in these circumstances. Restricting the number of hidden-layer neurons, however, makes the MLP extract only the main features of the training set. Thus, a generalization ability appears.

It is its hidden layers of neurons that make the multilayer perceptron attractive as a statistical modeling tool. The outputs of hidden neurons can be considered as new variables, which can contain interesting information about the relationship being modeled. Such new variables, known as internal representations, along with the net topology, can make the modeling process self-explanatory, and so the neural network approach becomes attractive as a form of machine learning.

As stated earlier, if variables are subjected to a logarithmic transformation, then a linear combination of such variables is equivalent to a complex ratio form. If the values input to an MLP are the logs of variable values, then the neurons in the (first, if there are more than one) hidden layer produce NETs that represent complex ratios. The nonlinear transformation effectively reverses the logarithmic transformation, so these complex ratios are inputs to the next layer of neurons where they are linearly combined to model the relation being investigated.

The hidden layer of neurons in the MLP discussed in this chapter is, then, dedicated to building appropriate ratios. The problem of choosing the best ratios for a particular task, which has taxed so many researchers, is thus avoided. The best ratios are discovered by the modeling algorithm, not imposed by the analyst. It will be shown later

that by using an appropriate training scheme these extended ratios can be encouraged to assume a simple and therefore potentially more interpretable form.

AN APPLICATION: MODELING INDUSTRY HOMOGENEITY

The approach described above is now applied to the problem of classifying firms to industries on the basis of financial statement data. The neural network's performance is compared to that of a more traditional discriminant analysis-based approach. To ensure that the discriminant analysis exercise is more than a "straw man," an existing, reputable study based on discriminant analysis is replicated. The neural network approach is then applied using the same raw data.

All companies quoted on the London Stock Exchange are classified into different industry groups according to the Stock Exchange Industrial Classification (SEIC), which groups together companies whose results are likely to be affected by the same economic, political, and trade influences.[11] Although the declared criteria are ambitious, the practice seems to be more trivial, consisting of classifying firms mainly on a end-product basis. The aim here is to attempt to mimic the classification process using accounting variables.

The data for exercises were drawn from the Micro-EXSTAT database of company financial information provided by EXTEL Statistical Services Ltd. This covers the top 70 percent of U.K. industrial companies. Fourteen manufacturing groups were selected according to the SEIC criteria. The list of member firms was then pruned to exclude firms known to be distressed, nonmanufacturing representatives of foreign companies, recently merged, or highly diversified. After pruning, data on 297 firms remained for a six-year period (1982–1987) and a bigger sample (502 cases) for the year 1984. The distribution of firms by industry in this sample is shown in Table 6.1.

The initial analysis of this data followed the traditional statistical modeling approach. This consisted of, first, "forming 18 financial ratios chosen as to reflect a broad range of important characteristics relating to the economic, financial and trade structure of industries."[12] Eight principal components were then extracted to form new variables. Next, these new variables were used as inputs to a multiple discriminant analysis. Only a randomly selected half of the data set was used during this estimation phase of the discriminant analysis.

Table 6.1

Industry Groups and Number of Cases in the One-Year (1984) Data Set

Group	Name	Cases	Percent (%)
1	Building Mat.	31	6.2
2	Metallurgy	19	3.8
3	Paper, Pack	46	9.2
4	Chemicals	45	9.0
5	Electrical	34	6.8
6	Industrial Pl.	17	3.4
7	Machine Tools	21	4.2
8	Electronics	79	15.7
9	Motor Comp.	23	4.6
10	Clothing	42	8.4
11	Wool	19	3.8
12	Misc. Text.	30	6.0
13	Leather	16	3.2
14	Food	80	15.9

The other half was used as a holdout sample to measure the classification accuracy of the resulting model. The exercise was repeated reversing the role of the two half data sets. Lack of consistency of results here would have raised doubts about the appropriateness of the sampling activity undertaken. A detailed description of the ratios used and the modeling procedure adopted can be found in Sundarsanam and Taffler.[12] The results of this exercise were found to be similar to those achieved by Sundarsanam and Taffler.[12] Thus, it was decided that they were an acceptable base case against which to compare the results achieved by an MLP constructed with the same data.

The input data for the neural network approach consisted of eight of the accounting variables that had been building blocks for the 18 ratios previously calculated. The number eight was selected simply to mimic the number of explanatory variables in the discriminant analysis. It must be emphasized that basic accounting variables, not ratios, were used. The selected items were Fixed Assets (FA), Inventory (I), Debtors (D), Creditors (C), Long-Term Debt (DB), Net Worth (NW), Wages (W), and Operating Expenses Less Wages (EX). The variables were chosen

to represent the key balance sheet elements and a rudimentary picture of cost structure.

A logarithmic transformation was applied to these variables. Many of these accounting variables were well suited for a logarithmic transformation. However, some caused problems because of the presence of zero or negative values. In order to transform the negative values of such variables, the following rule was applied:

$$\begin{aligned} x \rightarrow \quad & \log(x), && \text{for } x > 0 \\ x \rightarrow \quad & -\log(|x|), && \text{for } x < 0 \end{aligned}$$

This corresponds to the assumption that negative cases are lognormally distributed in a negative direction.

To avoid the problem of zero values, instead of log 0, a very small number, log 1 = 0, was used. Such an approach is acceptable if the unit of measurement is not far away from the typical value in the data set. An alternative approach to (some) such problem variables would be to ensure that their pattern of variation is reflected in the model by using as input variables some, that in combination define the problem variable, but which are themselves amenable to the logarithmic transformation. The variability of Profit, say, could be brought to the model by the introduction of both Sales and Expenses.

The base of logarithms to be used can be selected in a way that avoids the need for further scaling. The aim of the transformation is to avoid extreme values. With natural logs, the transformed values of the variables being examined ranged from 2 to 18 approximately. Base 10 logs generated a range between 3 and 7. Given the transfer function in use, 2 to 18 is too great a range. The training process would break down. Thus, base 10 logs were used, and the resulting variable values centered on zero for submission to the network.

The eight variables were then input to a succession of differently structured MLPs. The basic format consisted of an input layer of eight neurons, one or two hidden layers with relatively few neurons in each, and an output layer with 14 neurons. Once again the networks were trained using only half the data set, the other half being used to test classification performance. As with the discriminant analysis, the roles of training and testing set were reversed and the consistency of the resulting models examined. The most successful network topology involved one hidden layer with six nodes. The method of determining

this optimal topology, its performance, and the interpretation of its weights is discussed below.

INTERPRETING AND POSTPROCESSING THE OUTPUTS OF AN MLP

There is a problem when using an MLP with multiple output neurons. The implied industry classification, given a set of inputs, may not be easy to identify. This has implications for both network training and use. Each output neuron produces an output value between 0 and 1. It would be most unusual to find 13 zeros and a single 1 in the vector of outputs. Therefore, identifying the predicted classification when overlapping distributions are present requires a probabilistic interpretation of outputs. In accounting applications, population proportions generally bear no relation to the proportions observed in the sample. Therefore, the approaches adopted by other neural network researchers in other application areas, where population and sample probabilities coincide, may not be appropriate. In particular it is most unlikely that sample proportions can be viewed as good estimates of prior probabilities.

Following Baum and Wilczek,[13] several authors advocate a direct interpretation of outputs as probabilities, and show how the usual squared-error criterion can be corrected to achieve likelihood maximization.[14,15] In such cases, the connection weights in the network are adjusted in the gradient direction of the log-likelihood rather than the squared error.

An alternative approach is to interpret the outputs of the MLP as a multidimensional measure of distance to targets. If departures from normality are not severe, this interpretation can be carried out using conventional statistics such as chi-square, Penrose, or Mahalanobis distances. Such measures can be regarded as scores, and conditional probabilities can be deduced from them allowing further Bayesian corrections if required, independent of the proportions observed in the sample. A Bayesian correction independent of the sample proportions could of course also be applied directly to the MLP's outputs if they were interpreted as probabilities.

In this application it was found that interpreting neuron outputs directly as probabilities produced a clear reduction in classification accuracy. There was a severe loss of ability to identify firms belonging

to the smaller industry groups. A Bayesian correction independent of sample proportions was not pursued.

Results are reported in Figure 6.3. Direct interpretation, shown in Figure 6.3(a), ignores nine of the 14 industry groups, but finally achieves a good global performance by classifying the remaining five groups, which are the bigger ones, very well. Figure 6.3(b) shows classification performance when neuron outputs are postprocessed to produce a multidimensional distance measure. As can be seen, this allows the smaller industry groupings to appear. Therefore, although for the sake of efficiency of convergence the likelihood cost function was adopted during training, node outputs were postprocessed as distances.

A relatively simple approach was taken to the definition of a multidimensional distance. For a training set with N cases, consider o_{im}, the output signal produced in output layer neuron m, $1 \le m \le M$ by case i, $1 \le i \le N$. Compute K square deviations, d_{kim}, between neuron m's output for that input vector and each possible target value, t_{km} for that neuron: $d_{kim} = (t_{km} - o_{im})^2$, with k, $1 \le k \le K$. The mean sum of squared deviations from the kth target at neuron m over the whole training set will be:

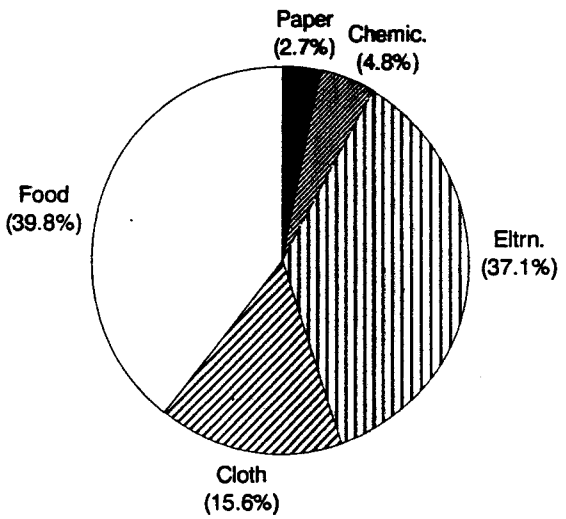
$$\sigma_{km}^2 = \sum_{i=1}^{N} \frac{d_{kim}}{(N-1)} \tag{4}$$

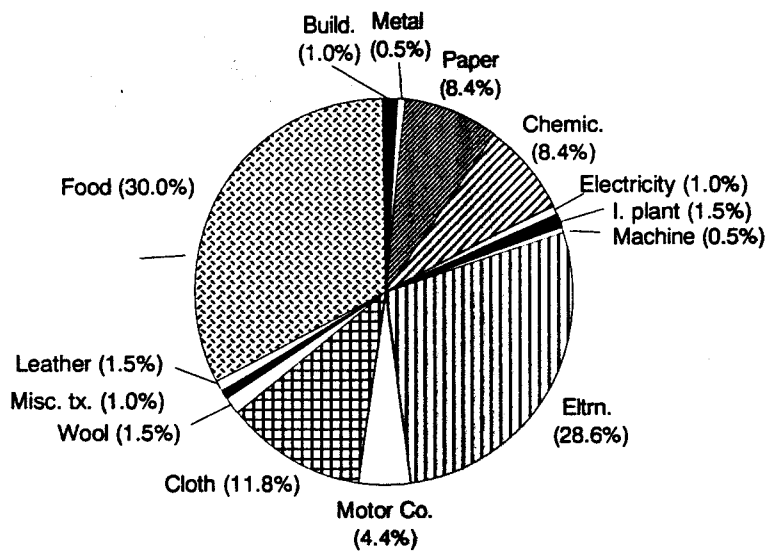
The standardized distances between a neuron's output and the kth target can be added over all nodes to give:

$$D_{ki} = \sum_{m=1}^{M} \frac{d_{kim}}{\sigma_{km}^2} \tag{5}$$

D_{ki} is then the distance between the output vector generated by the ith case in the training set and the kth target. The minimum of these distances identifies the appropriate classification if no Bayesian corrections are needed, that is, if the assumption of equal prior probabilities is acceptable. As part of this research effort, this distance measure's performance has been compared with that of a more elaborate measure, the Mahalanobis distance. The use of the Mahalanobis distance did not produce improved performance.

Figure 6.3
The Impact of Postprocessing Outputs on Classifications





DETERMINING NETWORK TOPOLOGY

The literature gives little guidance on selecting the number of hidden layers, or the number of nodes per hidden layer. Nor is there much advice on the number of times the training set should pass through the network before training is complete. The most common approach to the latter problem is to choose a target value for the training set error and repeat submission of the training set until this target value is achieved.

For the former problem a reasonable approach would seem to be to subdivide the training set into two parts, A and B. Training set A is used in the connection weight updating procedure as usual. Different topologies can be trained using this training set. The classification performance of each of these topologies on training set B can then be examined. The topology that gives best performance is then selected for further work. It is this topology that can then be retrained to generate a simplified structure as described in the next section. The true generalization ability of the network topology can then be checked on the as yet unused testing set.

THE PROCESS OF NETWORK TRAINING

One of the major goals of this research was the evaluation and improvement of the interpretability of multilayer perceptron models. MLPs are often considered unsuitable in applications where self-explanatory power is required. However, in the case of accounting variables, it seems possible to interpret the way the relation has been modeled by looking at the weights connecting input variables with the hidden layer's neurons. These weights are the exponents of the extended ratios involved in the optimal solution.

In order to enhance interpretability, the normal process of training interconnection weights was amended in two ways. First, it was decided to assign to one hidden-layer neuron the task of dealing with the scale effect in financial variables. The failure of ratios to cope with scale effects has been widely discussed in the literature. The weights on connections from input neurons to this hidden-layer neuron were fixed at either 0 or 1 from the outset. Connections with unit weights linked the hidden-layer neuron to only those inputs that were seen as size related. Dedicating one neuron of the hidden layer to representing this scale effect

generated as a bonus an improvement in speed of convergence of the training process.

The weight generation process was also amended in a second way. During training, whenever a new presentation of the entire training set was to begin, one of the neurons of the hidden layer was randomly selected and the connection weights linking it to the input neurons were examined. Any inhibitory weights (close to 0) were penalized by a small factor, typically 0.98. As has been said, the aim was to reduce the number of variables featuring in each of the complex ratios being formed. In a neural network each neuron acts as a modeling unit with a certain number of free parameters. The same output can be obtained with very different combinations of these parameters. Inhibitory weights connecting inputs with the hidden layer appear when the network tries to weaken the contribution of a variable. Therefore, by randomly introducing small penalizations of inhibitory weights during the training the inhibitory weights were encouraged to remain inhibitory. As a by-product, the noninhibitory weights were encouraged to become even less inhibitory. Before the end of training, all the weights connecting inputs to the hidden layer and exhibiting strong inhibitory values were set to 0 and fixed. While this procedure served its purpose, it should only be applied when the basic network topology is known with some confidence.

The results produced in the neural network can be seen in Table 6.2. This shows the extended ratios formed in an MLP with eight inputs, six nodes in one hidden layer, and 14 output nodes, trained with 1984 data. Only two hidden-layer neurons produce ratio forms of substantial complexity. The relative simplicity of the ratio structures achieved bodes well for other applications in the accounting and finance area.

Interpretation of the resulting ratios unfortunately is unclear. One possible explanation for this is that the data set being used does not include an economic basis for the classification decision. Financial statement data is hardly an ideal data set for the application in question; variables such as product type are obviously more relevant. However, the fact that traditional ratios have not been formed does not indicate a failure of the approach. It indicates the unsuitability of these traditional ratio constructs and the need for alternatives.

Apart from these nonstandard training features that stem from the particular application area, two further enhancements to the training process described in the literature were also applied. The first was the utilization of a learning rate particular to each weight.[16,17] The second

Table 6.2

Values of Weights Connecting Input Variables Hidden Nodes after Training with Penalties

	Node number				
Variable	2	3	4	5	6
DB			−6		
NW	8				
W	1			−6	
I	8				
D	2				−2
C				3	
FA	−9	−4		6	−4
EX	−10	4	8	−2	3

was, as has already been mentioned, likelihood maximization instead of squared-deviations minimization.

MLP CLASSIFICATION PERFORMANCE

In order to obtain an estimate of the generalization capacity associated with the MLPs examined here, the original samples were divided randomly into two subsamples of approximately equal size. All models were constructed twice, first with one half of the sample and a check carried out with the other half, and again reversing the roles of the two half-data sets. Results were considered acceptable if both models, when validated with the half sample not used to build them, produced consistent results.

All classification results reported here concern the test set, not the training set. That is, they were obtained by measuring the rate of correct classification the model produced when evaluated by the half set not used to train it. The classification performance on the set used for training depends solely on the number of free parameters and can be increased simply by introducing more neurons into the hidden layers. Such results are therefore uninteresting and are not presented here.

The normal approach to testing a model, by deleting a single observation and predicting its value with the model estimated on the rest

of the data set, and repeating this procedure, is infeasible here. This is because the training of a neural network is time consuming. However, the procedure adopted here will work acceptably with a large enough data set.

It was found that the generalization capacity of the neural MLP was very much dependent on the topology of the net. The number of nodes in the hidden layer seemed to determine the ability of the net to properly generalize. Persistently, good generalizations were obtained whenever the hidden layer had six nodes. Both the 1984 and the six-year data sets exhibited such a feature. Figure 6.4 shows some classification results for different numbers of nodes in the hidden layer when using the six-year data set. Similar patterns, though not showing so great a contrast, were observed when using the 1984 set.

Table 6.3 shows the best generalization results achieved with the traditional methodology (discriminant analysis and ratios) and also with the neural network. As can be seen, the neural network achieved a better performance, with half the number of input variables and within a much simpler framework.

Figure 6.4

Proportion of Correct Classifications versus Hidden-Layer Structure: 6-Year Data Set

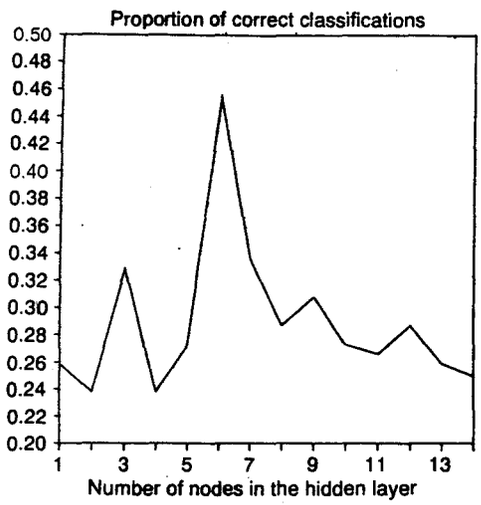


Table 6.3

Classification Results of MLP (Multilayer Perceptron) Compared with MDA (Multiple Discriminant Analysis)

	1984 Data (%)		Six-Year Data (%)	
Input	MDA	MLP	MDA	MLP
18 ratios	29		30	
8 variables		38		45

The need for forming appropriate ratios was avoided as well as the blind pruning of outliers and the extraction of an arbitrary number of factors.

CONCLUSIONS

So far, most applications of neural networks have related to the modeling of difficult relations (pattern recognition) or the mimicking of brain functions. There has been little emphasis on their potential explanatory power. Here, however, it has been argued that in accounting-based applications networks could generate meaningful internal representations. Numerical, continuous-valued observations such as those found in stock returns, or data organized in accounting reports, cannot be efficiently used by traditional expert systems knowledge acquisition tools. Neural networks can now be seen as an alternative self-explanatory tool. In this application hidden units formed ratios very different to those commonly used. If repeated in other application areas, this could shed light on many important issues.

The emphasis on interpretation should not obscure the other finding of the study. The MLP proved able to outperform the classification performance of a traditional discriminant analysis approach. Neither method came close to adequately classifying the testing sets, but there was a substantial improvement when the MLP was used. The fact that there was a potential for improvement was a key fact in determining the particular application area to be studied. It is perhaps worth pointing out that redoing the discriminant analysis, using representations of the ratios produced by the MLP, captured some but not all of the

MLP-based improvement. The remainder may well have related to the ability of the MLP to cope with nonlinear boundaries.

The importance of the MLP's topology cannot be overemphasized. The number of hidden layers, and hidden-layer neurons, can be selected by splitting the training set and adopting a two-phase training process. However, the principle of parsimony should always be borne in mind. If there are too many hidden neurons, the MLP will fail to identify key features and will model the noise in the data set as well. Generalization ability will then be lost.

ENDNOTES

1. A. Altman, R. Haldeman, P. Narayanan, "Zeta Analysis: A New Model for Bankruptcy Risk of Corporations," *Journal of Banking and Finance*, 1977.
2. R. Taffler, "Forecasting Company Failure in the U.K. Using Discriminant Analysis and Financial Ratios Data," *Journal of the Royal Statistical Society*, 1982.
3. G. Foster, *Financial Statement Analysis*, Prentice-Hall, (1986).
4. J. Horrigan, "The Determination of Long-term Credit Standing with Financial Ratios," *Journal of Accounting Research, Supplement. Empirical Research in Accounting: Selected Studies*, 1966.
5. P. Barnes, "Methodological Implications of Non-Normally Distributed Financial Ratios," *Journal of Business, Finance and Accounting*, 1982.
6. S. Mcleay, "The Ratio of Means, the Mean of Ratios and Other Benchmarks," *Finance, Journal of the French Finance Society*, 1986.
7. D. Trigueiros, "The Cross-Section Distribution of Accounting Variables" University of East Anglia: unpublished working paper, 1991.
8. D. Rumelhart, G. Hinton, and R. Williams, "Learning Internal Representations by Error Propagation," in *Parallel Distributed Processing*, (MIT Press, 1986).
9. Y. H. Pao, *Adaptive Pattern Recognition and Neural Networks*, Addison-Wesley, 1989.

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

11. J. Plymen, "Classification of Stock Exchange Securities by Industry," *Journal of the Institute of Actuaries,* 1971.

12. P. Sudarsanam, and R. Taffler, "Industrial Classification in U.K. Capital Markets: A Test of Economic Homogeneity," *Applied Economics* 1985.

13. E. Baum, and F. Wilkzek, "Supervised Learning of Probability Distributions by Neural Networks," *IEEE Conference on Neural Information Processing Systems—Natural and Synthetic,* Denver, 1987.

14. J. Hopfield, "Learning Algorithms and Probability Distributions in Feed-forward and Feed-back Networks," *Proceedings of the National Academy of Science USA,* 1987.

15. S. Solla, E. Levin, and M. Fleisher, "Accelerated Learning in Layered Neural Networks," *Complex Systems,* 1988.

16. R. Jacobs, "Increased Rates of Convergence Through Learning Rate Adaptation," *Neural Networks,* 1988.

17. F. Silva, and L. Almeida, "Speeding Up Backpropagation," INESC, (Lisbon: R. Alves Redol, 1990).