"GENERALIZATION WITH NEURAL NETWORKS:
AN APPLICATION IN THE
FINANCIAL DOMAIN"

by

Soumitra DUTTA*
and
Shashi SHEKHAR**

N° 92/30/TM/FIN

* Assistant Professor of Information Systems, INSEAD, Boulevard de Constance, Fontainebleau 77305 Cedex, France.

** University of Minnesota, Minneapolis, MN 55455, U.S.A..

Printed at INSEAD,
Fontainebleau, France

Generalization with Neural Networks: An Application in the Financial Domain

Soumitra Dutta

INSEAD
Fontainebleau, France 77305
dutta@freiba51.bitnet
Tel: 33-1-60724017

Shashi Shekhar

University of Minnesota
Minneapolis, MN 55455
shekhar@umn-cs.cs.umn.edu
Tel: 612-624-8307

Abstract

Neural networks have traditionally been applied for the problem of recognizing patterns in data. More recently, researchers are exploring the use of neural networks for learning a domain model from training data, and then using the learnt domain model to generalize about the problem domain. The benefits of using neural networks for generalization are high, specially in domains lacking well defined models (which makes it difficult or impossible to formulate mathematical and heuristic models for the domains). This paper explores the use of neural networks for generalization in the problem of predicting corporate bond ratings. The task of assigning ratings to corporate bonds is poorly understood, and lacks a well defined mathematical or heuristic model. A comparison is made between the effectiveness of neural network models and conventional statistical models. Our results indicate that neural networks can be used profitably for generalization problems.

1. Introduction

This section introduces neural network applications and describes the recognition and generalization problem types solved by neural networks. It also outlines the focus and structure of the paper.

1.1 Applications of Neural Networks

Neural networks consist of an interconnected network of simple processing units termed as neurons. A key distinguishing feature of neural networks is that the various units operate independently and in parallel. Instead of sequentially executing a series of program instructions (as is the case for all traditional computers), they

evaluate many competing hypotheses simultaneously. Thus they have the greatest potential in the recognition of patterns from large amounts of noisy data (domains such as speech and image recognition) which require high computation rates, the ability to pursue several hypotheses in parallel, and where current best systems are far from equalling human performance. Despite their individual simple structures, the entire network can as a whole demonstrate some remarkable and subtle properties such as an ability to dynamically adapt to a changing environment. There are also many different networks, each having its own operating procedures and its own specific set of advantages and disadvantages.

Neural network classifiers have been used in a variety of domains. Lippman [23] has identified three different kinds of tasks solved by neural networks. The first is the classical decision theory problem of trying to determine which class best represents an input pattern, where the input pattern has been corrupted by noise (e.g., speech recognition with noisy signals). The second use of neural networks is as a content addressable memory whose function is to generate the complete pattern when only a partial pattern is given as input (e.g., library information retrieval from partial information such as keywords). A third task solved by neural networks is vector quantizing or clustering the inputs into a set of clusters (e.g., in speech and image transmissions).

1.2 Recognition By Neural Networks

Prior to the application of a neural network, it has to be *trained* to recognize patterns from the domain under consideration. The training phase of neural networks can either be *supervised* (i.e., the training phase presents both the input pattern and the correct classification of the input) or *unsupervised* (when the training phase presents the input patterns without the corresponding labels). Though unsupervised training is used for clustering, supervised training is more commonly used for the classification of patterns and associative memories.

Figure 1 depicts supervised training where the (input, output_label) pairs (I_1, O_1), (I_2, O_2),...,(I_n, O_n), are used for training the network and the trained network is tested with the input I_j $(1 \leq i \leq n)$ corrupted by noise. The trained network is expected to reproduce the output O_j corresponding to I_j, in spite of the presence of noise. Such a task has been termed as the *recognition* problem (as in shape recognition and

speech recognition). Alternatively the network can be thought of as classifying the input noisy patterns into the various output categories (labels).

Figure 1 about here

Figure 2 shows one possible (input, output_label) pair that can be used to train a network to recognize numbers. Here the input is a 6X6 pixel image in which each pixel can be either 0 (light) or 1 (dark) and the output is the corresponding label (here the number 0). Figure 3 shows some possible noisy variations on the input pattern of Figure 2. An appropriately trained neural network should produce the correct output_label (here the number "0") for each of the noisy patterns shown in Figure 3. Note that in general, the input I_n and the output_label O_n in the training (input, output_label) pairs can be vectors of a set of values (i.e., a neural network accepts multiple inputs and produces multiple outputs). For example, the network of Figure 3 would typically have 36 inputs (one input corresponding to each pixel) and ten outputs (one output corresponding to each number 0,1,...,9).

Figures 2 & 3 about here

The ability to solve the recognition problem with noisy data is one of the distinguishing advantages of neural networks. This strength forms the basis of the use of neural networks in tasks requiring pattern recognition abilities (such as in speech and image recognition). Experience has shown that conventional rule-based technology is not very useful for pattern recognition problems.

1.3 Generalization By Neural Networks

A variant of the recognition problem is the *generalization* problem depicted in Figure 4. The training phases of the recognition and generalization problems are similar. However, in generalization problems, the trained network is tested with input I_{n+1}, which is distinct from the inputs I_1, I_2,..., I_n used for training the network. The network is expected to correctly predict the output O_{n+1} for the (previously unseen) input I_{n+1} from the model of the domain it has learned from the training input-output pairs. An example of a generalization problem is the prediction of future trends in the economy on the basis of prior data and observed historical trends.

Figure 4 about here

An alternate graphical representation (based on reference [8]) of generalization is given in Figure 5. D represents a certain domain whose model is to be learnt by the neural network. The learning sample shown to the network during the training phase is some subset T of the domain D. For testing, another disjoint set Y (as shown in Figure 5) is chosen. The performance of the neural network for Y measures the accuracy of generalization by the network. It is easy to see from this graphical interpretation that there are many different possible ways for the network to generalize from T (as shown by G_1, G_2, and G_3 in Figure 6). Thus the choice of the learning sample T and the testing sample Y is important (some domain knowledge is helpful for their selection). With an appropriate choice of T and Y, it is observed that neural networks can generalize "sensibly". This ability to generalize is an important advantage of neural networks as compared to rule-based systems or other conventional solution techniques which are usually unable to respond adequately to "new, unseen" situations. Neural networks used for generalization problem solving have also been termed as *expert networks* [6] as they are able to respond to unseen inputs and thus display some "expertise" about the domain under consideration.

Figures 5 & 6 about here

The generalization capabilities of neural networks are particularly useful in domains lacking a well defined domain model. It is difficult to apply conventional mathematical techniques in such domains as most mathematical models require a parametric formulation of the domain. Erroneous assumptions about the underlying distributions in formulating a parametric model distorts the results of the models. Rule-based systems also cannot be built as they require expert (deep) knowledge about the domain. Neural networks are useful in domains with poor models because they are non-parametric and make weaker assumptions about the shapes of the underlying distributions than traditional statistical classifiers [23]. They are thus more robust in the face of non-linear processes, non-Gaussian behavior and noise. Many domain models change with time. The adaptive nature of neural networks permits them to respond to and adapt with changes in the external world state. This evolutionary feature is lacking in conventional statistical techniques. Neural networks do not require detailed rules about domain behaviors as they are trained by presenting examples of input-output pairs.

1.4 Focus and Structure of Pape

Bond rating is the problem of assigning ratings (such as AAA, AA, etc.) to corporate bonds. Such ratings are typically given by private agencies using proprietary analyses. The model used for determining these ratings is not known. This paper explores the capability of neural networks to learn the domain model for bond rating from examples, and to generalize to new bonds. Prior research in finance has attempted to predict bond ratings using various conventional mathematical (regression) models. However, they have obtained limited success in their results. We use a neural model to predict bond ratings, and compare our results with those obtained by using multi-variate regression models. Neural networks can be considered as systems which map vectors in one space (for us, the space of financial characteristics of a company) onto another space (here, the bond ratings). The advantage of the neural network model is that there is no need to specify any particular functional model for mapping the input vector space onto the output vector space. This is in contrast to statistical techniques which require the apriori specification of a mathematical model for performing this mapping. Such characteristics indicate that neural networks can be useful for the domain of bond rating.

Most of the earlier applications of neural networks [1,3,15,16,17,18,24,25,30,31,34] have been to recognition problems. Neural networks are being applied more recently to generalization problems [6,7,9,10,12,21,29]. This study adds to the study of the application of neural networks for generalization problems.

The structure of this paper is as follows. The next section introduces the domain of corporate bond rating, gives a precise definition of the problem under consideration and describes prior research in bond rating. Section 3 introduces the neural network model and compares the use of statistical models with the neural network model. Details of the data collection and experimentation procedures, together with an analysis of the obtained results are given in section 4. Section 5 concludes the paper.

2. Corporate Bond Rating

This section formulates the problem of bond rating and describes relevant prior research in this area.

2.1 Introduction

The issuance of corporate bonds is an important source of cash for companies. A bond is, in simple terms, a loan which is to be paid back by the company at a fixed rate of interest according to a pre-determined schedule. The default risk of a bond is the possibility that the promised coupon and par values of a bond will not be paid back by the company to the bond holders.The default risks of the most actively traded bonds are rated by various independent organizations. Standard and Poor's and Moody's are the largest of these rating agencies concentrating upon corporate and municipal issues. Table 1 summarizes some of the ratings given by S&P and Moody's. The purpose of these ratings [26] "is to provide the investor with a simple system of gradation by which the relative investment qualities of bonds may be noted". These ratings are often used to define allowable bond purchases by certain investors. For example, the Comptroller of Currency has stated that bank investments must be of investment grade (i.e., in the top four rankings). Bond ratings have a significant effect on the offering yield on bond issue. A lower rated bond shall certainly have to offer a higher rate of interest. These ratings are also used by institutional portfolio managers as a metric to reflect the risk of their investment in bonds.

Table 1 about here

To evaluate a bond's potential for default, rating agencies rely upon a committee analysis of the issuer's ability to repay, willingness to repay, and protective provisions for an issue. It is not known what model, if any, do these rating agencies use for rating the various bond issues. All aspects analyzed by the ratings committee are also not known completely, and some features such as *willingness to repay* are affected by a number of variables which are difficult to characterize precisely. Thus, it is difficult to accurately define a mathematical model to perform the rating of bonds. Formulation of such a rating model is further complicated by the fact that the ratings of bonds are not fixed, but tend to fluctuate over time in response to changes in the domain. It is difficult to develop a rule-based expert system for rating bonds as very few experts are available and most knowledge about the process of ratings is confidential. Developing a model for rating

corporate bonds is important as it enables an independent assessment of the default risk of bond investments.

2.2 The Problem Statement

The task of assigning ratings to the bond issues of companies is an example of the generalization in the classification problem. The essence of classification problems is :

Given a set of classes and a set of input data instances, each described by a suitable set of features, assign each input data instance to one of the classes.

For our study, the different bond issues form the set of input data instances, and the various bond ratings (AA, B, etc.) form the set of possible classes to which the input bonds can belong. Each bond instance can be described by a set of features which represent important financial information about the company issuing the bond. To formalize the problem statement:

Let B represent the space of n bonds, $B_1, B_2, ..., B_n$, and R be the set of possible (mutually exclusive) m bond ratings, $R_1, R_2, ..., R_m$. Let F represent the k dimensional feature space, $F_1, ..., F_k$, describing each of the bonds. Thus each bond B_i can be considered as a k-tuple $F1_{Bi}XF2_{Bi}X....XFk_{Bi}$ in the Cartesian space $F_1XF_2X \ldots XF_k$. Our aim is to find the one to one mapping function f

$$f: F_1 \times F_2 \times ... \times F_k \rightarrow R$$

The mapping produced by this function f, i.e., the ratings assigned to the various bonds is determined by the rating agencies, but a precise functional form or a mathematical model of this function f is not known. It is also not clear exactly what feature space F to use. There is partial consensus on financial features that can affect the rating of a company's bonds and this is reflected in the regression variables used in prior research (see Section 2.3). As the relative effect of factors affecting the function f can change with time, a static statistical model has obvious limitations. These factors (of f being unknown and dynamic, and F being partially defined) make it difficult to formulate accurate regression models. A neural network approach has important advantages in such a scenario. A neural network does not require the function f to be specified apriori, but learns f autonomously from examples of companies and their associated ratings. In this paper, we choose a

feature space F comprising of relevant financial features (as evident from research in finance) and see if the network can successfully model the function f. The input vector space is given by $F_1XF_2X \ldots XF_k$ and the output vector space is R.

A neural network trained for predicting bond ratings is only useful if it is able to generalize. Solving the recognition problem in this context is not very important as financial data available about companies is generally crisp (and not noisy). However, the ability to predict the bond rating for new bond issues (i.e., generalize) is very important.

2.3 Review of Prior Research in Finance and Accounting

While rating bonds, agencies use both the financial data of the company and other qualitative factors, such as their subjective judgement concerning the future prospects of the firm ("worst potentialities in the visible future" [26]). Due to the difficulty of quantifying subjective analyses and qualitative judgements, researchers in finance and accounting have concentrated upon quantifiable historical data for the firm and provisions of the bond issue. The typical financial variables used include proxies for liquidity, debt capacity, debt coverage, size of issue etc.

Horrigan [20] regressed coded ratings with fifteen financial ratios. Subject to the magnitudes of cross-correlations, he chose six ratios out of the fifteen, which had the highest correlation with ratings. These were total assets, working capital over sales, net worth over total debt, sales over net worth, profit over sales, and subordination. Another regression between the ratings and the new set of variables, gave a model. This model was correct for 58% of Moody's ratings during the period 1961-1964. West [35] has a similar approach, using logarithmic forms of nine variables including earning variability, solvency period, debt equity ratio and outstanding bonds. His model correctly predicted 62% of Moody's rating during 1953. Pogue and Soldofsky [28] used a regression model with a dichotomous (0-1) dependent variable, which represents the probability of group membership in one group of pairs. They ran separate regressions for each pair of successive ratings (e.g. Aaa and Aa, Aa and A, A and Baa etc.) with the following independent variables: debt over total capital, income over assets, income over interest charge. Dummy variables were used for broad industry effects. This approach involved at least (n - 1) regressions for n-rating groups. A bond is assigned to the group in which its probability of occurrence is the highest. This method predicted 8 bonds out

of 10 in a hold out sample from the period 1961-1966. Pinches and Mingo [27] screened the initial 35 variables via factor analysis, and used multiple discriminant analysis to develop the final model. They used the following variables: subordination (0-1), years of consecutive dividend, issue size, income over assets, income over interest charge, and debt over assets. Bonds were classified on the probability of group membership. This model predicted roughly 65% and 56% of the Moody's ratings for hold out samples in the periods 1967-1969. Research on bond rating models have also been done by Kaplan and Urwitz [22], Ang and Patel [2], Belkaoui [4], Ederington [11], and Gentry et. al. [14].

2.4 Limitations of Statistical Models

Sections 2.1 and 2.2 described some of the problems in formulating regression models for bond rating. Thus it is not surprising that prior research using regression models has had limited success in predicting bond ratings, even after considering as many as 35 financial variables and performing a large number of iterative regressions (n-1 iterations for n ratings). This demonstrates the limited applicability of conventional mathematical techniques in domains with poorly defined models. Statistical techniques require the assumption of a certain functional form for relating dependent variables to independent variables. When the assumed functional form is not correct, statistical techniques merely confirm that, but do not predict the right functional form. Neural networks provide a more general framework for determining relationships in the data and do not require the specification of a explicit functional form. More details are provided in section 3.

3. The Neural Network Model

This section introduces the neural network model used in our experiments and compares regression and neural network models.

3.1 Introduction

A variety of neural network classifiers are described in the literature. Figure 7 (adapted from reference [23]) depicts the major types of networks classified on the basis on the nature of inputs (binary or continuous) and the employed training procedure (supervised or unsupervised). Most domains have a certain associated natural representation for the inputs (binary or continuous). In the domain of bond

rating, certain constraints (see section 4.2) necessitate the use of continuous-valued inputs. It is also possible to conduct supervised learning in our chosen domain as the assigned ratings are known for the bond issues in the training sample. Thus the multi-layered perceptron was chosen as the appropriate neural network for our experiments. The multi-layered perceptron is a popular neural network model and has been successfully used for applications in various domains [23].

Figure 7 about here

Figure 8 depicts a simple multi-layered perceptron. There are three layers in the network. The first layer is the "input layer" of four neurons. Four (continuous) input signals, I_1-I_4, come into these four neurons. The output of each of the neurons in the input layer is fed as input to each of the neurons in the next layer, the "hidden layer" (called so because the neurons in this layer are hidden from the input and the output). The outputs of each of the neurons in the hidden layer are in turn fed as input to the two neurons in the highest layer, the "output layer". There are two continuous output signals (O_1 and O_2) from the network. The neurons in the various layers are numbered for ease of reference. Though not shown in Figure 8, each connection between two neurons has an associated numerical number called the "weight" of the connection. For conceptual simplicity, they can be thought of as indicating the strengths of the connections between two neurons. During the training of the network (see section 3.2 below), these weights are varied according to some learning procedures. The input to the network comes via the lowest input layer and the output from the network comes from the highest output layer.

Figure 8 about here

Figure 9 gives a schematic representation of the processing occurring within an arbitrarily selected neuron, #22. There are four inputs (O_{11}-O_{14}) to the neuron, which are the outputs from each of the four neurons in the input layer. There is one output from the neuron, O_{22}, which is fed as input to the two neurons in the output layer. There are weights on the connections entering and leaving the neuron. The notation adopted to represent these weights is as follows: $_{xx}W_{yy}$ represents the weight on the connection (emanating) from neuron #xx and ending on neuron #yy. Note that (for clarity of presentation) all weights are not shown on the connections in Figure 9. There are two basic computation processes occurring within the neuron. First, a

weighted sum, Y_{22}, is computed using the incoming input signals and the corresponding connection weights. In the case of neuron #22, this is given by:

$$Y_{22} = (O_{11})({}_{11}W_{22}) + (O_{12})({}_{12}W_{22}) + (O_{13})({}_{13}W_{22}) + (O_{14})({}_{14}W_{22})$$

Figure 9 about here

This weighted sum is then passed through a transfer function to produce the neuron output O_{22}. The transfer function simply transforms the weighted sum of the inputs signals to produce the output signal. Different kinds of transfer functions are used and each results in different kinds of properties for the network. The most commonly used transfer function is the sigmoid transfer function shown in Figure 10. The output is a continuous monotonic function of the input. Both the function and its derivatives are continuous everywhere. Its popularity is largely due to these nice mathematical properties which enable the derivation of sound properties for networks employing such transfer functions.

Figure 10 about here

3.2 Learning in Neural Networks

Training in perceptrons is supervised (see Figure 7). The training set consists of various "input-output" example pairs (see Figures 1, 2, and 4). To learn the domain model, the network has to execute a particular algorithm. Different training algorithms are available in the literature. We used the back-propagation algorithm, which is in essence, an iterative gradient descent algorithm designed to minimize the mean square error between the actual output of a multi-layer perceptron and the desired output. The basic steps of the back-propagation algorithm are given below.

- **Step 1**: Assign small, random weights on the various inter-neuron connections in the network.
- **Step 2**: Feed to the input layer of the network a set of inputs from an input-output training example.
- **Step 3**: Compute the output of the network given the current connection weights and the inputs of step 2. The computation inside every individual neuron occurs as depicted in Figure 9.

- **Step 4**: Compare the output of the network (step 3) with the desired output (from the input_output example of step 2) and compute the error.
- **Step 5**: Use a recursive algorithm starting at the output nodes and working back to the input layer nodes to adjust the weights on the various connections in a manner so as to reduce the error (step 4).
- **Step 6**: Repeat steps 2-5 with the input_output pairs from the training set till all exemplar pairs in the training set are passed through the network.
- **Step 7**: Repeat steps 2-6 till no changes are required in the weights on the various connections (step 5) for any input_output example pair. Thus, the algorithm cycles through the entire training set repeatedly till the weights on the various connections converge to a stable value.

The recursive algorithm used to adjust the weights (step 5) is fundamentally a gradient search technique. The mathematical details are dependent on the actual implementation of the neuron transfer functions and is omitted here (see references [1,15,17,24,30,34]). The back-propagation algorithm has been tested in a number of different domains [1,15,17,24,30,34] such as speech recognition and synthesis and visual pattern recognition and has yielded good results. It is a popular choice for multi-layer perceptron implementations.

3.3 Decision Regions Solved By Perceptron Models

The problem solving capabilities of neural networks vary with the number of layers in the network. Lippman [23] mentions that most general decision region solvable by one layer perceptrons (consisting of only one neuron) is a half plane (bounded by a hyper-plane) as shown on the left in Figure 11. This is not very interesting and thus one layer perceptrons are rarely used. Two layer perceptrons do not have a hidden layer (with the input nodes directly connected to the output nodes) and the most general decision region they can solve is a closed or open convex region as shown in the middle of Figure 11. Rumelhart [24,29] has indicated that a two-layered network is inadequate when the similarity structures of the input and output patterns are very different (as in the XOR problem). In such cases a three-layered network is required. Three layered networks (with one hidden layer) are more powerful and are able to solve arbitrarily shaped decision regions as shown on the right in Figure 11. There is a recoding of the input patterns in the hidden units of a three-layered network such that the network can support any required mapping from the input to the output units. Hornik et. al. [19] have demonstrated that perceptron models with a

hidden layer have the "universal approximation" property, i.e., the network can provide an approximation to any function (mapping the input vectors onto the output vectors) likely to be encountered, provided the number of hidden units is large enough. The classification power of multi-layer perceptron networks arises from the non-linearity used within nodes. If the nodes were linear elements, a single layer network with appropriately chosen weights could exactly duplicate the capabilities of a multi-layer network [23].

Figure 11 about here

3.4 Regression Analysis vs Neural Network Models

The decision region of the problem of bond rating consists of many closed regions (corresponding to the different possible ratings) in the multi-dimensional space of the input variables. Figure 12 depicts the different decision regions for the case of a two dimensional feature space F_1XF_2. Linear regression models are similar to single layer perceptron models and thus the most general type of decision regions generated by them are half planes bounded by a hyperplane (see Figure 11). Linear regression models are inadequate for solving the bond rating problem accurately as a half-plane cannot discriminate amongst the decision regions of the various ratings. However, if the decision problem is posed as recognizing only one type of bond rating, a linear regression model may be adequate. For example, in Figure 12, a linear regression model (or a one layer perceptron model) can be devised to generate the shaded planar decision region (which discriminates between AAA bonds and other ratings).

Figure 12 about here

The decision region generated by a two layer neural network is an open or closed convex region. Thus two layer neural networks are also inadequate for the general problem of bond rating (i.e., for discriminating amongst all the ratings) and can be used profitably only when the decision problem is posed as recognizing a single bond rating. There are similarities in the decision regions generated by certain non-linear regression models and two layered perceptron models. We did not consider non-linear regression models in our analysis as we did not have sufficient guidelines for formulating the non-linear functional form to be employed. It can be seen from

Figure 12 that two layer neural networks can yield a better model than linear regression as the decision region can "encapsulate" any one rating decision region (e.g, the rating "A" in Figure 12). A linear regression model would also necessarily enclose the decision regions of other ratings while covering the region of the rating "A". Three layer neural networks are capable of generating arbitrary decision regions and thus can be used for the general problem of bond rating.

Regression can give us the parameters of a functional form but cannot determine the correct functional form. In contrast, a neural network autonomously determines the functional form from among a larger set. The functional form represented by a fixed architecture of a neural network can be written in closed form as a composition of sigmoids and weighted summations. However this function can represent a larger set of functions than the linear and logarithmic functional form used by regression. The network tunes the functional form and the weights (on the arcs) to fit the learning examples, as closely as desired. We can specify both, the desired size (complexity) of the model (neural network) and the error tolerance for fitting the model. This gives us a more general framework for discovering relationships existing in data.

However, note that statistical models are useful for determining the right set of independent variables, which determine the dependent variable to the largest extent. The hidden layers (intermediate layers) of a neural network extract the higher order features from the input variables, and thus automatically discriminate among the independent variables, but the output does not indicate which of the input variables are more important in determining the output. The weights on the connections for a two layered network can give the relative importance of the input variables, but for any three or higher layered network, such a discrimination from the weights on the connections becomes very difficult.

4 Data Collection and Experimentation

4.1 Selection of Variables

Based on the results of prior researchers in bond rating, we selected ten financial variables for predicting bond ratings. The assumed influence of a variable on bond rating and the ease of availability of data were the primary factors in the selection of the variables. The selected variables are listed in Table 2.

Table 2 about here

As the feature space of the input vector is not known accurately in the problem of bond rating (see section 2.2), we ran two sets of experiments to verify whether choosing a subset of the initially selected feature space would significantly affect our results. Our first experiment used all ten variables in predicting the bond rating. Then we used only the first six variables (#1 - #6) to predict the bond ratings.

4.2 Data Collection

Values of the necessary input variables were collected from the April 1986 issues of the Valueline Index and the S&P Bond Guide. Bond issues of forty seven companies were selected at random. Thirty companies were used as the learning sample, i.e., to train the neural network (learn the weights on the different connections) and obtain the regression coefficients. The rest seventeen companies were used to test the neural network and the regression performance. All the selected bonds had approximately the same maturity date (1998 - 2003).

For the purposes of our experiment, the symbolic ratings of bonds were converted into numerical values as shown in Table 3. For enhancing discrimination in the output, we mapped similar ratings (e.g., AA+, AA and AA-) onto the same numerical value (e.g., 0.9). The varying difference in the numerical values corresponding to adjacent rating groups (e.g., AA and A) reflect our interpretation (from reading the S&P Bond Guide and Moody's Bond Record) of the changes in the investment quality of bonds from one rating group to the next. We could have used a binary scale to convert the bonds, (e.g., bonds rated A = 1; all other bonds = 0) but such a model would not be accurate as there is an ordering in the rating of the bonds with respect to the quality of the bonds (i.e., AAA is the best, and the quality decreases progressively with lower rated bonds). This information is not captured by the binary model.

Table 3 about here

4.3 Regression Model

We used the Berkeley ISP [5] for a multi-variate linear regression analysis. The t-statistics were significant for every regression coefficient. The regression coefficients were then used to predict the ratings of both the learning sample (to see how well the regression model fitted the learning sample) and the testing sample of new bond issues (to test how well the regression coefficients predicted the ratings of the test bonds). We did not use a non-linear regression model due to a lack of knowledge about the correct functional form to use for obtaining the regression coefficients.

4.4 Neural Network Model

The multi-layer perceptron model considered had 10 (6) input nodes - one node for each of the 10 (6) input variables - and 1 output node specifying the corresponding bond rating. We experimented with different neural network configurations (2 layered, 3 layered, different number of hidden nodes in 3 layered neural networks, etc.). The total permissible tolerance was kept constant for different network configurations. We compared the performance of the various neural networks against regression analysis and the performance of the various neural network configurations against themselves. Finally, we repeated all the above experiments with a smaller number of variables (variables #1-#6 of the ten initially chosen). Our results are summarized in the next sub-section.

4.5 Results

Tables 4 through 8 summarize the results of our experiments. The models were expected to recognize if a given bond belongs to class of all bonds with AA rating. Such a formulation of the decision region enables a reasonably fair comparison of the linear regression and neural network models because there was only 1 bond with a AAA rating in our sample (see Section 3.4). This problem naturally classifies the responses of prediction models into four different categories, described by the pairs of (actual, predicted) columns shown in Table 4. The column *Actual* is *Accept* if and only if the given bond is actually rated AA by S&P. For an ideal model, our test results should only belong to the first and fourth rows of Table 4 where the ratings given by S&P and the model coincide. Rows 2 and 3 of Table 4 are the undesirable cases and represent false negatives and false positives respectively.

Table 4 about here

Table 5 condenses the results of the learning phase and Table 6 summarizes the results of the testing phase. The % entries in Tables 5 and 6 represent the % of correctness of prediction of the regression and neural network models. We also list the absolute error as tot_sq_err for each model to give an idea of goodness of fit by various models. The tot_sq_err gives the sum of the squares of the errors in prediction in all the cases.

Tables 5 & 6 about here

4.6 Interpretation of Results

Neural networks consistently outperform the regression model in predicting bond ratings from the given set of financial ratios. Both in the training and learning samples the total squared error for regression analysis is about an order of magnitude higher than that for neural networks (see Tables 5 and 6). Also, the success rate of prediction for neural networks is considerably higher than that for regression analysis, e.g., the success rate during the testing phase for the two layered neural network (10 variables) is 88.3% as compared to 64.7% for the regression model. There is also no appreciable change in the prediction accuracy of the regression model from the learning to the testing phases. This is also indicative of the poor performance of the regression model.

Comparing the performance of 2 and 3 layered networks, we observe that the performance of 3 layered networks is better during the learning phase (the accuracy of prediction is marginally better but the total error is much lower). But surprisingly, the performance of 2 layered networks is superior during the testing phase. This is due to "over-fitting" to data while using 3 layered networks [17]. Using a larger network requires the setting of more weights (parameters) and can lead to over-fitting. The resulting network will be good for the training data but generalize poorly on new test data as compared to a network that gives a good fit on the training data. This again illustrates the importance of choosing the right network for the problem. For the problem under consideration (recognize AA bonds), two layered neural networks are adequate (see Figures 11 and 12). Using a three layered network leads to inferior generalization.

Tables 7 and 8 describe the errors in the ratings given by the neural network and regression models during the testing phases. It can be observed that the neural network models usually generated errors of a single rating category (i.e., an AA bond was rated as an A bond or a BB bond was rated as a BBB bond and so on). In contrast, regression models usually generated errors of two or more rating categories. As explained in sections 3.3 and 3.4, the decision region generated by the regression model is a half-plane. A half-plane cannot discriminate effectively between several ratings within the half-plane. A two (or three) layered network can "encapsulate" the decision region of one rating and provide better discrimination.

Tables 7 & 8 about here

The results obtained during the testing phase by our regression model are comparable to those obtained by prior researchers (see section 2.3). The poor performance of the regression models indicates that the linear multi-variate model is inadequate for explaining the rating of bonds. This is the case when we have selected a relatively simple formulation of the decision region (recognizing the category of AA bonds). The performance of regression models would degrade further for the general problem of bond rating (i.e., discriminating amongst all bond ratings).

5 Conclusion

The research described in this paper was first reported in an earlier paper [10]. This section compares our results with some other recent approaches to the problem of bond rating, and provides some concluding comments.

5.1 Comparisons with Related Research

Some novel approaches have recently been used for the problem of corporate bond rating. Srinivasan and Bolster [32] have use the Analytical Hierarchy Process to model the process of assigning corporate bond ratings. They applied their model successfully to the examples of two large American motor companies. Garavaglia [13] and Utans and Moody [33] have used neural networks for the task of bond rating. Garavaglia [13] has used the counter-propagation neural network for bond rating. In her results, while the neural network gave poor results (maximum accuracy 55%) for

predicting any one particular bond rating, the accuracy of neural networks increased substantially (maximum of 92%) if the the problem was simplified to detecting three broad classes of bonds (investment, speculative and poor quality). Utans and Moody [33] conducted experiments in the domain of bond rating to select appropriate neural network architectures. They used 196 companies and found a maximum accuracy of prediction of 28% for the prediction of any one rating. The success rate increased to 67% for errors of one or less rating.

5.2 Comments on Generalization with Neural Networks

Mathematical and heuristic models are commonly used for problem solving. The construction of mathematical models demands a good understanding of the underlying domain model. Heuristic rule-based systems also require a fair knowledge about the relevant domain model (which is usually acquired incrementally during knowledge engineering). Both these modelling techniques are not very applicable for domains where knowledge about the underlying domain model is poor and knowledge engineering is difficult. Neural networks represent an alternative modelling technique which is applicable to problem solving and modelling in such unstructured domains. Neural networks autonomously learn the domain model from examples and do not require the apriori specification of the domain model.

This paper has considered the generalization capabilities of neural networks in the domain of bond rating. The poor applicability of mathematical models in this domain can be seen from the mediocre performance of the regression models (note that prior researchers with more sophisticated regression models also achieved similar low success rates). The results obtained using neural networks, while not perfect, are certainly very encouraging. They point to the potential of neural networks for providing decision support in unstructured domains. The results reported in this paper are from one relatively simple experiment. An improvement in the results can probably be expected by grouping bonds on the basis of some domain knowledge (e.g., only considering one industry sector). (Note that we selected the various companies (in the training and testing samples) at random.) More experimentation with different financial variables should be beneficial and may lead to better results. The sample size of companies should also be enlarged. However, it is useful to note that both our results and those of Garavaglia [13] indicate that neural networks can play an important role in predicting bond ratings if the problem is suitably posed (as

the determination of three broad bond categories - in Garavaglia's experiment - as opposed to distinguishing between individual categories).

References

1. Anderson J. A., and Rosenfield E., (Eds.), Neurocomputing: Foundations of Research, MIT Press, Cambridge, 1988.
2. Ang J. S. and Patel K. A., Bond Rating Methods: Comparison and Validation, Journal of Finance, Vol. 30, No.2, pp. 631-640, May 1975.
3. Bechtel W., and Abrahamsen A., Connectionism and the Mind, Blackwell, 1991.
4. Belkaoui A., Industrial Bond Ratings: A New Look, Financial Management, Autumn, pp. 44-51, 1980.
5. Berkeley Interactive Statistical Package, Department of Statistics, University of California at Berkeley, 1988.
6. Caudill M., Using Neural Networks: Making an Expert Network, AI Expert, pp. 41-45, July 1990.
7. Collins E., Ghosh S., and Scofield C.L., An Application of Multiple Neural Networks to Emulation of Mortgage Underwriting Judgement, in Proceedings of IEEE International Conference on Neural Networks, Vol. II, pp. 459-466, San Diego, CA, July 1988.
8. Denker J., Schwartz D., Wittner B., Solla S., Howard R., Jackel L., and Hopfield J., Large Automatic Learning, Rule Extraction and Generalization, Complex Systems, 1, pp. 877-922, 1987.
9. Dutta S. , Shekhar S., and Wong, W.Y., Decision Support in Non-Conservative Domains: Generalization with Neural Networks, forthcoming in Decision Support Systems, North-Holland, 1992.
10. Dutta S. and Shekhar S., Bond-Rating: A Non-conservative application of neural networks, in the Proceedings of the IEEE International Conference on Neural Networks, Vol. II, pp. 443-450, San Diego, CA, July 1988.
11. Ederington L., Classification Models and Bond Ratings, The Financial Review, November, pp. 237-262, 1985.
12. Gallant S.I., Connectionist Expert Systems, in the Communications of the ACM, Vol. 31, No. 2, pp. 152-168, Feb. 1988.
13. Garavaglia S., An application of a Counter-Propagation neural network: Simulating the Standard & Poor's corporate bond rating system, in the

Proceedings of the 1st International Conference on Artificial Intelligence Applications on Wall Street, pp. 278-287, 1991.

14. Gentry J.A., Whitford D.T., and Newbold P., Predicting Industrial Bond Ratings with a Probit Model and Funds Flow Component, BEBR Working Paper No. 1198, University of Illinois, Urbana-Champaign, 1985.
15. Hecht-Nielsen R., Neurocomputing, Addison Wesley, 1989.
16. Hecht-Nielsen R., Neurocomputing: Picking the Human Brain, IEEE Spectrum, Vol. 25, No. 3, pp. 36-41, March 1988.
17. Hertz J., Krogh A., and Palmer R.G., Introduction to the theory of neural computation, Addison Wesley, 1991.
18. Hopfield J. J. and Tank D. W., Neural Computation of Decision in Optimization Problems, Biological Cybernetics, Vol. 52, No. 2, pp. 141-152, Springer Verlag, 1985.
19. Hornik K., Stinchcombe M., and White H., Multi-layer Feedforward Networks are Universal Approximators, Neural Networks 2, pp. 359-366, 1989.
20. Horrigan J.O., The Determination of Long Term Credit Standing with Financial Ratios, Empirical Research in Accounting: Selected Studies, Journal of Accounting Research, pp. 44-62, 1966.
21. Irani A., Matts J.P., Long J.M., and Slagle J.R., Using artificial neural nets for statistical discovery: Observations after using back-propagation, expert systems, and multiple-linear regression on clinical trial data, University of Minnesota Technical Report, 1989.
22. Kaplan R., and Urwitz G., Statistical Models of Bond Ratings: A Methodological Inquiry, Journal of Business, 52, pp. 231-262, 1979.
23. Lippman R.P., An Introduction to Computing with Neural Networks, IEEE ASSP Magazine, pp. 4-22, April 1987.
24. McClelland J. L., and Rumelhart D. E., Explorations in Parallel Distributed Processing, MIT Press, Cambridge, MA, 1988.
25. McCord Nelson M., and Illingworth W. T., A Practical Guide to Neural Networks, Addison Wesley, 1991.
26. Moody's Bond Record, Moody's Corporation, NY. 1988.
27. Pinches G. E. and Mingo K. A., A Multivariate Analysis of Industrial Bond Ratings", Journal of Finance, 28, pp. 1-18, 1973.
28. Pogue T.F. and Soldofsky R. M., What is in a Bond Rating?, Journal of Financial and Quantitative Analysis, 4, pp. 201-228, June 1969.
29. Rumelhart D.E., Brain Style Computation: Learning and Generalization, Academic Press, 1990

30. Rumelhart K., McClelland J., and the PDP Research Group, Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Bradford Books, Cambridge, MA, 1986.
31. Sejnowski T., and Rosenberg C.R., NETtalk: A Parallel Network that Learns to Read Aloud", Johns Hopkins Univ. Tech. Report JHU/EECS-86/01, 1986.
32. Srinivasan V. and Bolster P.J., An industrial bond rating model based on the Analytic Hierarchy Process, European Journal of Operational Research, 48, pp. 105-119, 1990
33. Utans J. and Moody J., Selecting neural network architectures via the Prediction Risk: Application to Corporate Bond Rating Prediction, in the Proceedings of the 1st International Conference on Artificial Intelligence Applications on Wall Street, pp. 35-41, 1991.
34. Wasserman P.D., Neural Computing: Theory and Practice, Van Nostrand Reinhold, New York, 1989.
35. West R.R., An Alternative Approach For Predicting Corporate Bond Ratings, Journal of Accounting Research, pp. 118-127, Spring 1970.

Table and Figure Captions

Moody's	Standard & Poor's	Definition*
Aaa	AAA	The highest rating assigned. Capacity to pay interest and principal very strong.
Aa	AA	Very strong capacity to pay interest and principal. Minor differences from highest rated issue.
A	A	Strong economic capacity to repay interest and principal, but may be susceptible to adverse changes in economic conditions.
Baa	BBB	Adequate protection to repay interest and principal but more likely to have weakened capacity in periods of adverse economic conditions.
B	B	Moderate default risk.
Caa	CCC	High default risk. May be in default or in severe danger of default.
Ca	CC	Highly speculative and likely to be in default. For Moody's, the lowest rated bonds with poor prospects of gaining any real investment standing.
	D	In default. Payment of interest and/or principal in arrears.

* Adapted from S&P's Bond Guide and Moody's Bond Record

Table 1: Definitions of ratings

Var. #	Definition	Var. #	Definition
1	Liability/(Cash + Assets)	2	(Total funded debt)/(net property)
3	Sales/Net worth	4	Profit/Sales
5	Financial strength of company as given in the Valueline Index	6	Number of times available earnings cover fixed charges
7	Past 5 year revenue growth rate	8	Next 5 year projected revenue growth rate
9	Working capital/Sales	10	Subjective prospects of company (from Valueline)

Table 2: Financial variables used to predict bond ratings

Bond Rating	Numerical Value	Bond Rating	Numerical Value
AAA	0.95	BBB+, BBB, BBB-	0.5
AA+, AA, AA-	0.9	BB+, BB, BB-	0.35
A+, A, A-	0.7	B+, B, B-	0.25

Table 3: Conversion of bond ratings to numerical values

Actual	Prediction	Description
Accept	Accept	S&P rating is AA and rating of model is also AA
Accept	Reject	S&P rating is AA and rating of model is not AA
Reject	Accept	S&P rating is not AA and rating of model is AA
Reject	Reject	S&P rating is not AA and rating of model is also not AA

Table 4: Possible classification of responses

Number of Variables	Neural Network		Regression
	2-layered	3-layered	
6	80% tot_sq_err = 0.236	80% tot_sq_err = 0.175	63.33% tot_sq_err = 1.107
10	80% tot_sq_err = 0.224	92.4% tot_sq_err = 0.054	66.7% tot_sq_err = 0.924

Table 5: Summary of results from the learning phase

Number of Variables	Neural Network		Regression
	2-layered	3-layered	
6	82.4% tot_sq_err = 0.198	76.5% tot_sq_err = 0.194	64.7% tot_sq_err = 1.528
10	88.3% tot_sq_err = 0.164	82.4% tot_sq_err = 0.228	64.7% tot_sq_err = 1.643

Table 6: Summary of results from the testing phase

Testing Phase 10 Variables	Number of ratings by which the output is in error 1 rating	2 ratings	3 ratings	4 ratings
2 Layer Neural Network	5	0	0	0
3 Layer Neural Network	6	1	0	0
Regression	3	8	1	1

Table 7: Analysis of errors in the output ratings (10 variables)

Testing Phase 6 Variables	Number of ratings by which the output is in error 1 rating	2 ratings	3 ratings	4 ratings
2 Layer Neural Network	5	1	0	0
3 Layer Neural Network	5	1	0	0
Regression	3	9	0	1

Table 8: Analysis of errors in the output ratings (6 variables)

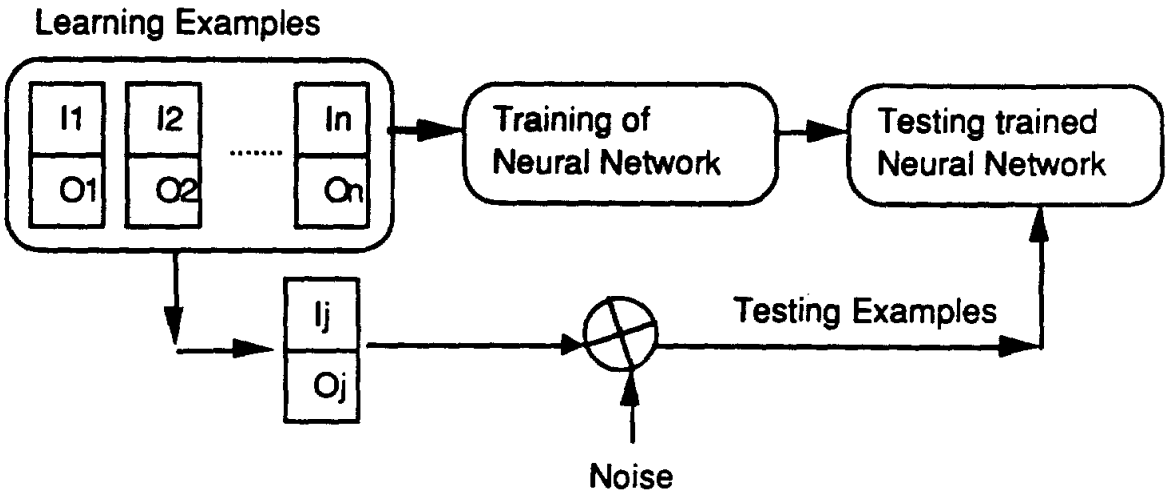
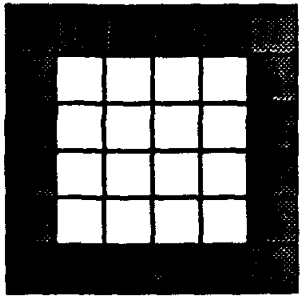


Figure 1: The recognition problem



Input 6X6 Pixel Pattern

0

Corresponding
Output_label

Figure 2: An (input, output_label) pair

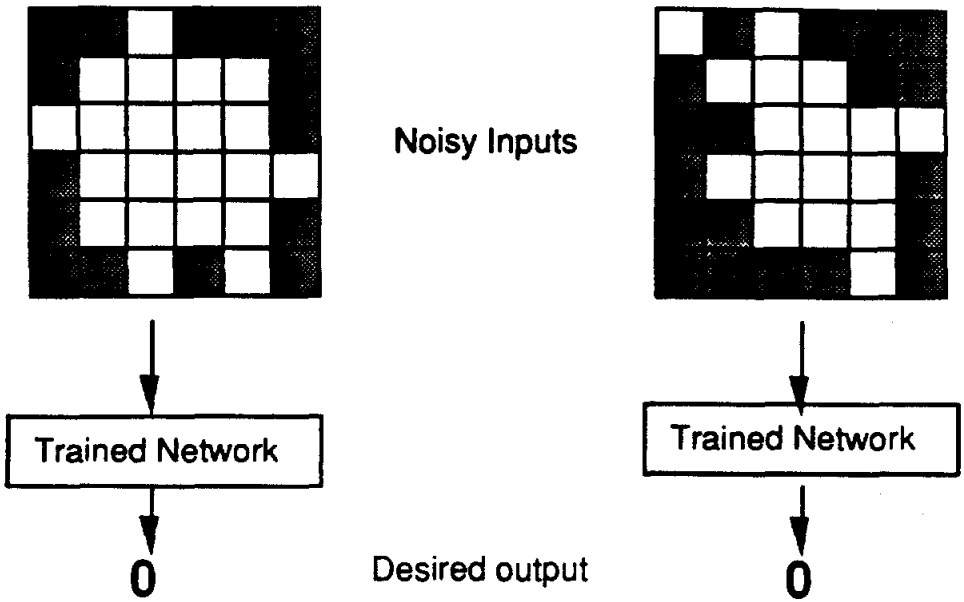


Figure 3: Recognizing noisy patterns

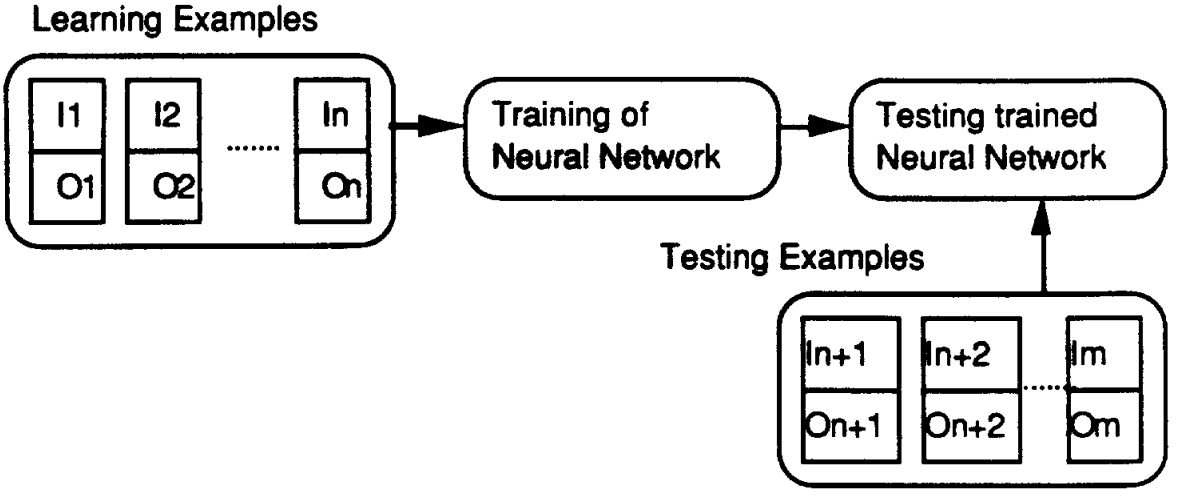


Figure 4: The generalization problem

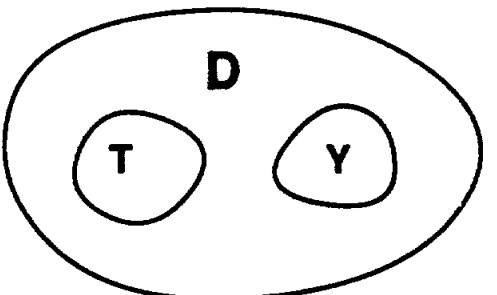


Figure 5: A pictorial interpretation of generalization

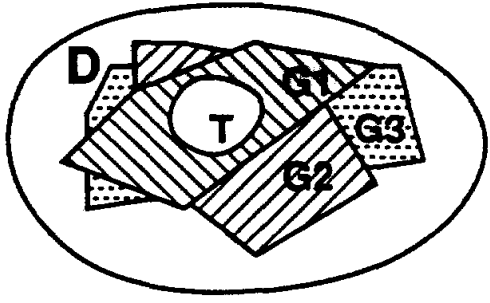


Figure 6: Different possible ways to generalize

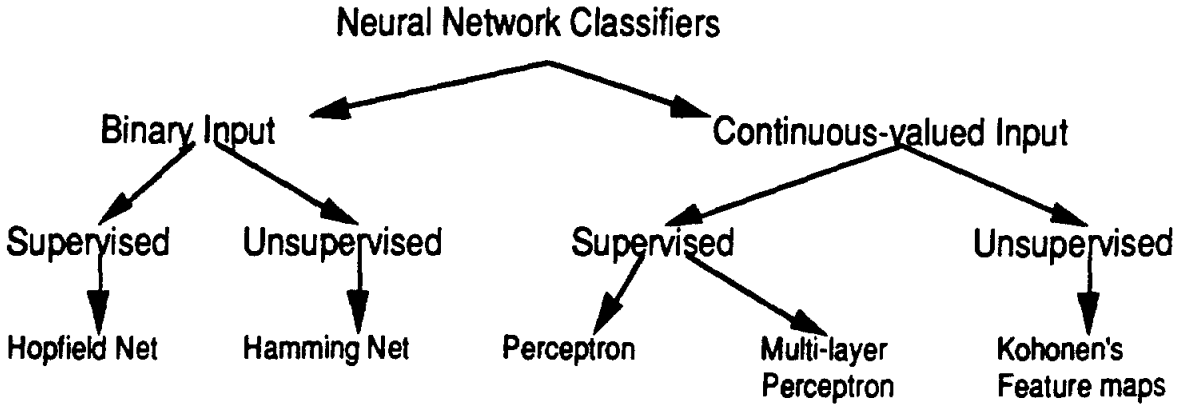


Figure 7: Types of neural networks

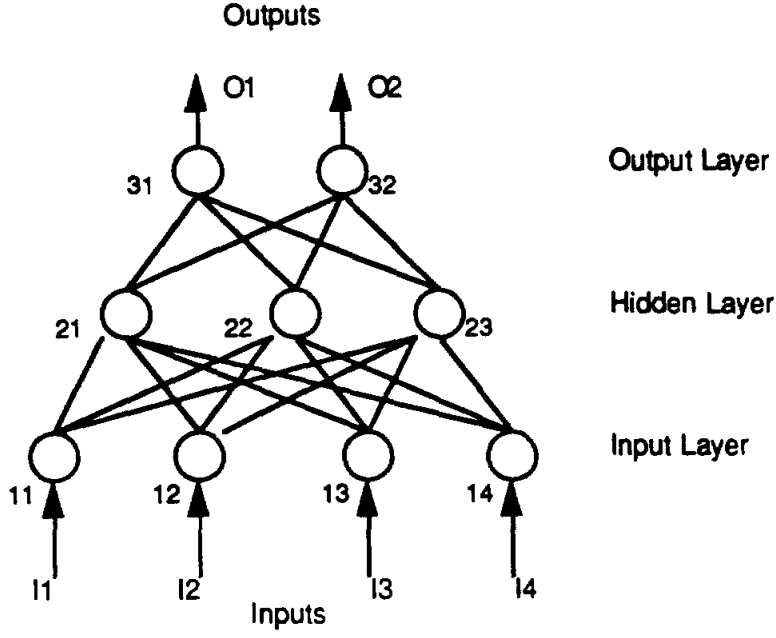


Figure 8: A simple three layered perceptron model

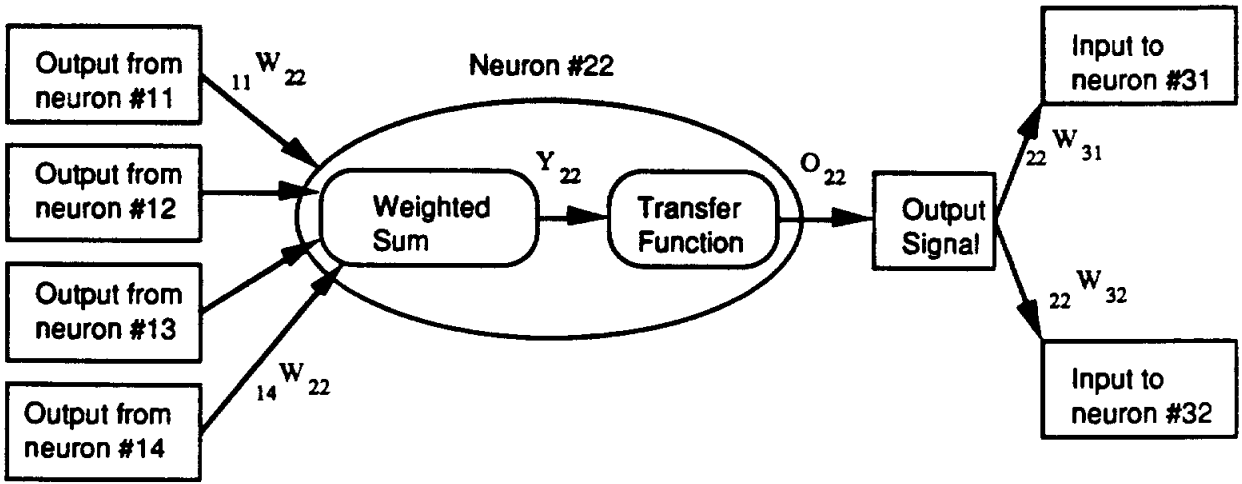


Figure 9: Schematic representation of processing within a neuron

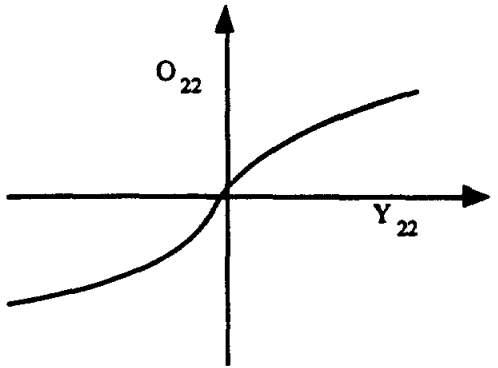


Figure 10: The sigmoid transfer function

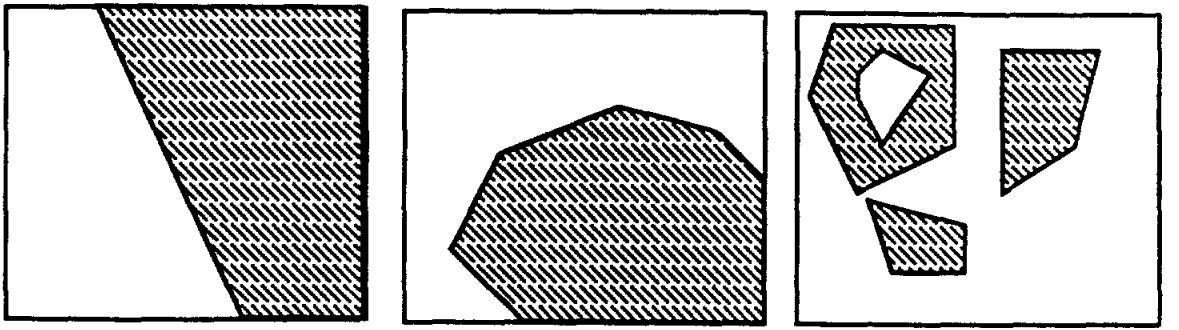


Figure 11: Decision regions solved by multi-layered perceptron models

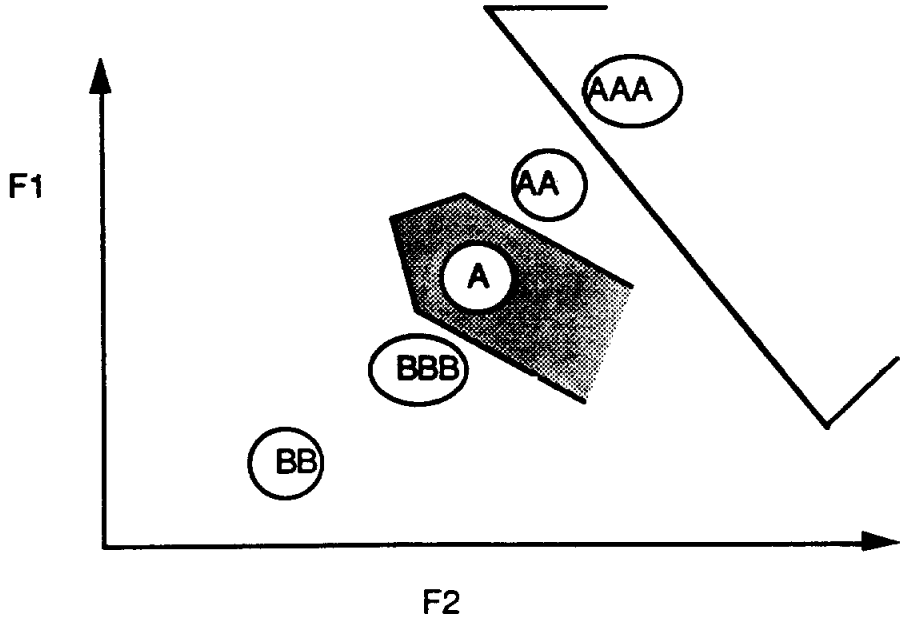


Figure 12: Decision regions in the bond rating problem