Abstract—Interest in neural networks has expanded rapidly in recent years. Selecting the best structure for a given task, however, remains a critical issue in neural-network design. Although the performance of a network clearly depends on its structure, the procedure for selecting the optimal structure has not been thoroughly investigated. It is well known that the number of hidden units must be sufficient to discriminate each observation correctly. A large number of hidden units requires extensive computational time for training and often times prediction results may not be as accurate as expected. This study attempts to apply the principal component analysis (PCA) to determine the structure of a multilayered neural network for time series forecasting problems. The main focus is to determine the number of hidden units for a multilayered feedforward network. One empirical experiment with sunspot data is used to demonstrate the usefulness of the proposed approach.

I. INTRODUCTION

INTEREST in neural networks has expanded rapidly in recent years. The Parallel Distributed Processing (PDP) Research Group has greatly contributed to the resurgence of neural-network models [29]. Much of the success of neural networks is due to such characteristics as parallel processing, nonlinear processing, and nonparametric and distributed representation, etc. [14], [18].

While there have been considerable efforts to develop various neural-network models and learning algorithms [9], [17], [19], [21], [28], [30], [37], the design for the optimal structure of a network for a given task has still remained a problem. Designing an optimal structure involves finding the structure with the smallest size network which produces minimal errors for trained cases as well as for untrained cases. The size of the network is defined as the number of parameters to be estimated. Although it is often cited as a rule of thumb that the number of weights should be less than one tenth of the number of training patterns [34], it is crucial to have an objective way to determine the network structure which produces the minimal errors with the smallest size.

Most previous studies on selecting the optimal structure for a network have been done in the context of discrete classification problems [7], [32]. This paper concentrates on the issue in the context of continuous cases, in particular time series forecasting issues.

The paper is organized as follows: Section II gives a brief review on previous relevant studies. A procedure to determine the structure of a neural network is proposed in Section III. A real example is employed to demonstrate the application of the proposed method in Section IV. Section V concludes this study.

II. REVIEW OF PREVIOUS STUDIES

Previous studies on selecting a network structure can be categorized into four approaches: the ad hoc approach, the dynamic approach, the distribution approach, and others. By taking an ad hoc approach, an experimenter usually has a clear idea of the problem and decides on the structure of the network subjectively based on past experience. One justification for this approach is the robustness of the neural network, that is, the network structure may not affect the performance of the network significantly [33]. Although neural networks are quite robust with respect to a specific structure to some extent, often it is found the performance of a neural network model still depends on the model structure (see Morgan and Bourlard [23]).

Unlike selecting the structure on an ad hoc basis, the dynamic approach automates the structure selection procedure. The idea is to increase or decrease the number of hidden nodes dynamically during the training process based on system errors. The procedure is repeated until there is no change in the system errors. Typical examples are the method proposed by Hirose et al. [12], the cascade-correlation method proposed by Fahlman and Lebiere [6], Frean’s upstart algorithm [8], and the weight-elimination method by Weigend et al. [34], [35]. Hanson and Pratt [11] and Chauvin [5] extend Weigend et al.’s weight-elimination method.

The distribution approach is an attempt to either find or assume the distribution function of estimates and tries to derive a statistic to measure the performance of the network. For example, White [36] proposes a technique to test whether a current network ignores any nonlinear components, i.e., hidden units. Although his technique is mathematically sound, it requires a series of assumptions to test the hypothesis. The technique does not guarantee that there does not exist a network which has an optimal structure with a smaller number of hidden units. Fogel’s [7] method is another example of the distribution approach. Under the assumption that the input to the final node has a normal distribution, he proposes a...
statistic called the final information statistic (FIS), which is a derivative of Akaike's information criterion (AIC) [1]. To select the optimal number of hidden nodes, the method requires a network designer to calculate the FIS for each proposed model and then to choose the model which has the smallest FIS value. It is applied only to the discrete classification problem. To guarantee finding the optimal structure by the FIS method, it is required to examine a large class of network structures.

Along with the approaches mentioned above, there are several alternative techniques developed for determining the structure of a neural network. Sietsma and Dow propose a pruning approach [31], [32]. Pruning refers to a process that identifies unnecessary units in a solution and removes them from the network [31]. An overview of pruning algorithms is provided in [27]. Another technique is Kung and Hwang's algebraic projection (AP) method [20] which checks the regularity of the training data set and determines the number of hidden nodes based on the regularity. Gutiérrez et al. [10] try to estimate the number of hidden nodes for the binary classification problem by investigating the number of patterns to be stored with a set of given nodes. Brent [4] proposes a method that trains a network faster than the backpropagation algorithm. It first constructs a decision tree and later simulates the decision tree with a neural network. Because the number of hidden layers and the number of hidden nodes per layer are determined by the level of the decision tree and the number of nonterminal nodes in the tree, it eliminates the problem of specifying the number of hidden nodes in advance. Ho [13] investigates the effect of adding a layer versus adding a node.

Although many researchers have made efforts to find the optimal structure for a network, success has been very limited. To fully utilize the reviewed method, analysts need to make strong assumptions on the distribution function of estimates or the error function, or to go through variations of trial and error process. Therefore, it is critical to develop a method which helps neural-network users to build the internal structure of a network systematically and efficiently. In contrast to the dynamic approach and the pruning approach that attempt to use information of system errors or weight changes to improve the structure of a network, the main idea of the proposed procedure is to use statistical information to guide the search for a better network structure. In the next section, the paper proposes a method to find a structure for a network. The method is applied to cases which deal with continuous variables as well as discrete variables.

III. A METHOD TO DETERMINE THE STRUCTURE OF A NETWORK

Structure determination in this paper refers to the questions: 1) how many layers are required and 2) how many nodes are required in each layer. In this study, structure determination does not include the selection procedure of input nodes and output nodes.

As Lippmann [22] points out, the two-layered network1 can describe a fairly complex nonlinear relationship as well as a linear one between the input variable and the output variable. Although the three-layered network is more flexible in describing a complicated relationship, it has a drawback, i.e., an increase in processing time. Hornik et al. [15] and Hornik [16] prove that the two-layered network with a sufficiently large number of hidden nodes can represent any functional relationship between the input variable and the output variable. Therefore, this study concentrates on the second issue: how many nodes are required in the hidden-layer in a two-layered network. It does not, however, address the question of whether or not a network of more than two layers might have a smaller total number of nodes in the hidden layer.

A. Proposed Technique

The idea of the proposed method is to check whether there is any redundant information on the outputs of the hidden nodes and, if there exists any, the method eliminates this redundancy by using principal component analysis (PCA)2 so that a fewer number of nodes can describe the relationship without losing information. The steps of the proposed method are as follows:

1) Initially train a network with an arbitrarily large number of hidden nodes,
2) Obtain the covariance matrix3 \((p \times p)\) of outputs of the hidden nodes. \(P\) is the number of hidden nodes in the initial network.
3) Obtain the eigenvalues of the matrix.
4) Count the number \((p^*)\) of eigenvalues whose value is greater than one.4
5) a) If \(p^*\) is less than \(p\), pick \(p^*\) nodes out of \(p\) by examining the correlation between the hidden nodes and the selected principal components.
b) Otherwise, it confirms that there is no redundant information. It is not guaranteed, however, that the current network has the optimal number of hidden nodes. The system may need more hidden nodes to improve its performance. In this case, there are two ways to check the optimal size of the hidden nodes. One is simply to increase the number of hidden nodes and reapply this method. The other is to apply the method proposed by White [36] to test whether the system needs additional hidden nodes. This should not happen, however, as long as the

---

1 The two-layered network means a network which has only one hidden layer.
2 Applying the PCA method to a neural-network model is not new. For instance, Almeida and Silva [2] apply the PCA method to input data with the goal of reducing training time, and Oja [25] uses a neural-network model to calculate the principal components. Plumley [26] demonstrates the stability conditions of principal component algorithms on unsupervised learning. This study, however, is the first attempt to use the method to determine the network architecture.
3 When there is a scale problem, it is recommended that a correlation matrix instead of a covariance matrix be used for PCA. It is not necessary, however, to consider the problem at this stage because each hidden node is already a linear combination of input variables. In other words, the scaling problem must be addressed when the input variables are selected.
4 When the number of hidden nodes is determined, in addition to the eigenvalues, other statistics such as the proportions explained by each component, cumulative proportions, the reduction of the eigenvalues, etc., may be considered.
initial network has a sufficiently large number of hidden nodes.

6) After determining the number of hidden nodes, there are two ways to retrain the network with a new structure: one is to randomly generate the initial weights again and to retrain the network from the start. The other approach is to use the estimated weights, especially the ones between the selected hidden nodes from Step 5a) and the output node as starting values. The second approach is expected to reduce the number of iterations required for training.

This method examines the covariance matrix of outputs of the hidden nodes to test the optimal structure of the network. Each node in the hidden layer has as input the linear sum of the input variables and produces as output the sigmoidal transformation of the input. If there is any redundant information, for example, a hidden node can be represented by another hidden node or a set of hidden nodes, the rank of the covariance matrix will be less than the number of hidden nodes. To calculate the rank of the matrix, the redundant information, for example, a hidden node can be considered as the equations to define a hyper-region and \( p^{*} \) is the number of equations required to define the region.

5) Unlike other methods which require many experiments with different structures, the proposed method requires training of only two network structures. Another advantage is that it may not require new learning when the principal components are highly correlated with the selected hidden nodes. Because the reduced number of nodes have most of the information possessed by the original nodes, it may not require any retraining of the network.

IV. EXAMPLES

This section illustrates how the proposed method, especially PCA, can be used to select the network structure. An experiment is done with yearly sunspot data from 1770–1869. This series has also been analyzed by Box and Jenkins [3] and is identified as having the following AR(2) model

\[
y_t = c + \theta_1 y_{t-1} + \theta_2 y_{t-2} + e_t.
\]

A 2 x 11 x 1 network which has \( Y_{t-1} \) and \( Y_{t-2} \) as input nodes, 11 hidden nodes, and one output node is initially trained to predict a one-step-ahead forecasted value \( Y_t \). After training

the initial network, the covariance matrix of the outputs of the hidden units is calculated to determine the optimal structure of the network. The eigenvalues produced by the matrix are 8.3093, 2.5978, 0.0568, 0.0359, 0.0001, 0.0001, 0, 0, 0, 0, and 0. Two principal components can explain up to 99.15% of the total variation. Therefore, two units are selected as an optimal size for the hidden layer and a network with two hidden nodes is retrained. For comparison the same data set is trained by four other different network models as well. Table I shows the training result from each model. The experiment is repeated 10 times with different random seeds. All networks are trained with a learning rate of 0.3 and a momentum factor of 0.7. As illustrated in Table I, the proposed network has the smallest training error. When the same data set is estimated by the AR(2) model in (1), the MSE from the AR(2) model is 222.9554.

The result shows that the proposed network not only produces the smallest training error among the networks but also has a smaller error than the AR(2) model. Fig. 1 shows the plots of the forecasted values from both the optimal network model and the AR(2) with actual observed data.\(^6\)

\(^6\)Weigend et al. [34] also demonstrate their weight-elimination method with sunspot data from 1700–1920. They start with a 12 x 5 x 1 network as the initial network and train the network with the weight-elimination method. They find that a network with three hidden nodes is the optimally structured network. The authors apply the PCA method proposed in this paper to the same data set with the same initial network and find that the proposed method also produces the exact same result, three hidden nodes. This can be viewed as additional evidence which validates the proposed PCA method. Moreover, this PCA method not only finds the optimal structure correctly but also converges more quickly than the weight-elimination method. The PCA method requires only 451 epochs but the weight-elimination method requires 5000 epochs.

<table>
<thead>
<tr>
<th>Network Structure</th>
<th>Mean Square Error (MSE)</th>
<th>Mean Absolute Error (MAE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 x 11 x 1</td>
<td>156.2792</td>
<td>9.8539</td>
</tr>
<tr>
<td>2 x 3 x 1</td>
<td>154.4879</td>
<td>9.4056</td>
</tr>
<tr>
<td>2 x 2 x 1</td>
<td>151.9368</td>
<td>9.2865</td>
</tr>
<tr>
<td>2 x 1 x 1</td>
<td>230.2508</td>
<td>12.2454</td>
</tr>
<tr>
<td>2 x 0* x 1</td>
<td>233.8252</td>
<td>12.1660</td>
</tr>
<tr>
<td>AR(2)</td>
<td>222.9554</td>
<td>11.8726</td>
</tr>
</tbody>
</table>

* indicates the proposed optimal structure.
** represents a network without hidden units.

<table>
<thead>
<tr>
<th>Retraining Method</th>
<th># of Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>using weights randomly generated</td>
<td>427</td>
</tr>
<tr>
<td>using weights previously estimated</td>
<td>129</td>
</tr>
</tbody>
</table>

The speed of convergence of each approach.
The number of iterations (epochs) are counted to compare the speed of reaching a stable state when the network uses the estimated weights from the previous network as initial values for retraining and when the network generates the weights randomly. As mentioned in Step 6) of the proposed technique, among initial 11 hidden nodes hidden node 4 has the highest correlation (0.99113) with the first principal component and hidden node 11 has the highest correlation (0.99428) with the second principal component. Therefore, the weight between hidden node 4 and the output node, and the weight between hidden node 11 and the output node are used as starting values of the weights in the $2 \times 2 \times 1$ structure. Table II shows the number of iterations required to reach a stable condition with these two methods.

The number of iterations for the first method is the average value of the number of iterations for 10 experiments. Each experiment starts from a different random seed so that it gives a more accurate measure for the speed of convergence. As expected, the latter approach greatly reduces the number of iterations.

So far the series have been examined only in terms of training error. It is also important, however, for a model to have a capability for accurately predicting values outside of the training data set. To compare the forecasting power of this model with a time-series model as well as with other network models, the AR(2) model expressed in (1) and four different models are considered. In each case, the first 88 observations from 100 observations of sunspot data are used for the training data set, and the remaining 12 observations are stored to examine the forecasting errors. This experiment is also repeated 10 times with different random seeds. The result in Table III shows that the selected neural network model, $2 \times 2 \times 1$, has the smallest forecasting errors among the network models for the testing data. The $2 \times 2 \times 1$ model performs better than the AR(2) model as well.
For this sunspot data series, it is concluded that the neural-network model proposed by this paper produces better forecasted values for the training data set as well as for the outside training data set than the other structured network models and the AR(2) model.

Another finding is that when a network has more hidden nodes than the proposed size, the MSE difference is not as significant as when a network has a fewer number of hidden nodes than the proposed size. This is consistent with the findings of other neural-network research on the so-called "robustness" of the structure [33]. Therefore, it is concluded that the hidden-node size determined by the proposed method provides the minimum boundary for the network.

V. CONCLUSION

This paper reviewed and proposed a procedure to determine a structure of a neural-network model for a given task. It concentrated on a method to determine the minimal number of nodes in a single hidden layer. The sunspot example demonstrates that the proposed method produces reliable results. When the method is implemented into a DSS (decision support system), it will help users who may or may not be familiar with neural-network models to easily construct a network for a given task. The method proposed in this paper is limited only to the feedforward multilayered model but the idea of this method may be applied to other models as well. It is also limited in that the proposed method reduces the number of hidden units based on linear relationships rather than nonlinear ones. It is noteworthy, however, that the method provides a way to determine the minimal bound of the number of hidden units required for processing.

REFERENCES