

Order in the Black Box: Consistency and Robustness of Hidden Neuron Activation of Feed Forward Neural Networks and Its Use in Efficient Optimization of Network Structure

Sandhya Samarasinghe

Abstract Neural networks are widely used for nonlinear pattern recognition and regression. However, they are considered as black boxes due to lack of transparency of internal workings and lack of direct relevance of its structure to the problem being addressed making it difficult to gain insights. Furthermore, structure of a neural network requires optimization which is still a challenge. Many existing structure optimization approaches require either extensive multi-stage pruning or setting subjective thresholds for pruning parameters. The knowledge of any internal consistency in the behavior of neurons could help develop simpler, systematic and more efficient approaches to optimise network structure. This chapter addresses in detail the issue of internal consistency in relation to redundancy and robustness of network structure of feed forward networks (3-layer) that are widely used for nonlinear regression. It first investigates if there is a recognizable consistency in neuron activation patterns under all conditions of network operation such as noise and initial weights. If such consistency exists, it points to a recognizable optimum network structure for given data. The results show that such pattern does exist and it is most clearly evident not at the level of hidden neuron activation but hidden neuron input to the output neuron (i.e., weighted hidden neuron activation). It is shown that when a network has more than the optimum number of hidden neurons, the redundant neurons form clearly distinguishable correlated patterns of their weighted outputs. This correlation structure is exploited to extract the required number of neurons using correlation distance based self organising maps that are clustered using Ward clustering that optimally cluster correlated weighted hidden neuron activity patterns without any user defined criteria or thresholds, thus automatically optimizing network structure in one step. The number of Ward clusters on the SOM is the required optimum number of neurons. The SOM/Ward based optimum network is compared with that obtained using two documented pruning methods: optimal brain damage and variance nullity measure to show the efficacy of

S. Samarasinghe (✉)

Integrated Systems Modelling Group, Lincoln University, Christchurch, New Zealand
e-mail: sandhya.samarasinghe@lincoln.ac.nz

the correlation approach in providing equivalent results. Also, the robustness of the network with optimum structure is tested against perturbation of weights and confidence intervals for weights are illustrated. Finally, the approach is tested on two practical problems involving a breast cancer diagnostic system and river flow forecasting.

Keywords Feed-forward neural networks • Structure optimization • Correlated neuron activity • Self organizing maps • Ward clustering

1 Introduction

Feed forward neural networks are the most powerful and most popular neural network for nonlinear regression [1]. A neural network with enough parameters can approximate any nonlinear function to any degree of accuracy due to the collective operation of flexible nonlinear transfer functions in the network. However, neural networks are still treated as black boxes due to lack of transparency in the internal operation of networks. Since a neural network typically is a highly nonlinear function consisting of a number of elementary functions, it is difficult to summarize the relationship between the dependent and independent variables in a way similar to, for instance, statistical regression where the relationships are expressed in a simple and meaningful way that builds confidence in the model. In these statistical models, coefficients or model parameters can be tested for significance and indicate directly the strength of relationships in the phenomena being modeled. Although neural networks are used extensively and they can provide very accurate predictions, without internal transparency, it is not easy to ensure that a network has captured all the essential relationships in the data in the simplest possible structure in classification or function approximation. Therefore, it is vital for the advancement of these networks that their internal structure is studied systematically and thoroughly. Furthermore, the validity and accuracy of phenomena they represent need thorough assessment. Additionally, any consistency in the activation of neurons can reveal possibilities for efficient optimization of the structure of neural networks.

2 Objectives

The goal of this Chapter is to address in detail the issue of internal consistency in relation to robustness of network structure of feed forward (multiplayer perceptron) networks. Specifically, it has the following objectives:

- To investigate if there is a recognizable pattern of activation of neurons that reveals the required complexity and is invariable under all conditions of network operation such as noise and initial weights.
- To investigate the possibility of efficient optimization of structure (i.e., by pruning) based on internal consistency of neuron activations in comparison to existing structure optimization methods, such as, optimal brain damage and variance nullity measure.
- Apply the above structure optimization approach to multi-dimensional data and practical real-life problems to test its efficacy.
- To test the robustness of a network with the optimum structure against perturbation of weights and develop confidence intervals for weights.

3 Background

Feed forward networks have been applied extensively in many fields. However, little effort has gone into systematic investigation of parameters or weights of neural networks and their inter-relationships. Much effort has been expended on resolving bias variance dilemma (under- or over- fitting) [2] and pruning networks [1, 3–9]. In these approaches, the objective is to obtain the optimum or best possible model that provides the greatest accuracy based on either the magnitude of weights or sensitivity.

A network that under-fits, lacks nonlinear processing power and can be easily corrected by adding more hidden neurons. Over-fitting is more complex and occurs when the network has too much flexibility. Two popular methods for resolving over-fitting are early stopping (or stopped search) and regularization (or weight decay) [10]. In early stopping, a network with larger than optimum structure is trained and excessive growth of its weights is prevented by stopping training early at the point where the mean square error on an independent test set reaches a minimum. Regularization is a method proposed to keep the weights from getting large by minimizing the sum of square weights in the error criterion along with the sum of square error. Pruning methods such as optimal brain damage [5–7] and variance nullity measure [8] make use of this knowledge to remove less important weights. However, they do not reveal if there is a pattern to the formation of weights in networks in general and if they are internally consistent, unique, and robust.

Aires et al. [11–13] in addressing the complexity of internal structure of networks have shown that de-correlated inputs (and outputs) result in networks that are smaller in size and simpler to optimize. This was confirmed by Warner and Prasad [14]. In addressing uncertainty of network output, Rivals and Personnaz [15] constructed confidence intervals for neural networks based on least squares estimation. However, these studies do not address the relationships of weights within a network due to sub-optimum network complexity and uncertainty of response of the simplest structure.

Teoh et al. [16] proposes singular value decomposition (SVD) of hidden neuron activation to determine correlated neuron activations in order to optimize network structure. It is a step toward meaningful investigation into hidden neuron activation space; however, as authors point out, the method requires heuristic judgment in determining the optimum number of neurons. Furthermore, our research, as will be presented in this chapter, revealed that the most meaningful patterns are found not in the hidden neuron activation space but in the weighted hidden neuron activation feeding the output neuron. Xian et al. [17] used an approach based on the knowledge of the shape of the target function to optimize network structure, which is only possible for 2- or 3-dimensional data as target function shape cannot be ascertained easily for high-dimensional data. Genetic and evolutionary algorithms [18, 19] have also been used for identifying network structure, but they typically involve time consuming search in large areas in the weight space and rely on minimum insight from the operation of a network compared to other approaches to network structure optimisation.

In this Chapter, a systematic and rigorous investigation of the internal structure and weight formation of feed forward networks is conducted in detail to find out if there is a coherent pattern to weights formation that reveals the optimum structure of a network that can be easily extracted based on such knowledge. We also greatly expand our previous work presented in [20] for structure optimization.

4 Methodology

A one-dimensional nonlinear function shown in Fig. 1a (solid line) is selected for simplicity of study and interpretation of the formation of weights in detail. This has the form

$$t = \begin{cases} 0.3 \sin x & \text{If } x < 0 \\ \sin x & \text{otherwise} \end{cases} \quad (1)$$

A total of 45 observations were extracted from this function depicted by the solid line in Fig. 1 and these were modified further by adding a random noise generated

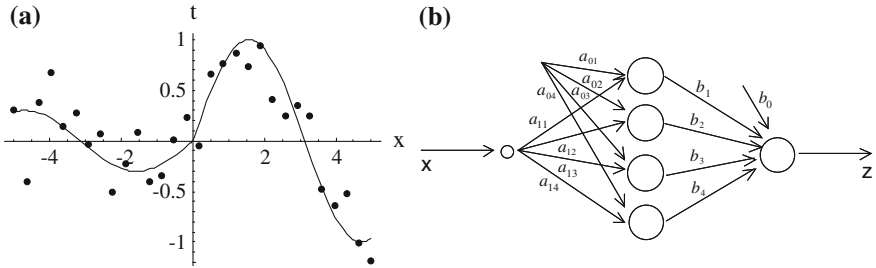


Fig. 1 Data and the model: **a** Target data generator and noisy data (random sample 1) generated from it and **b** network with redundant neurons [1]

from a Gaussian distribution with 0 mean and standard deviation of 0.25 as depicted by dots in Fig. 1. It is worth noting that the size of the extracted data set was purposely kept relatively small with a reasonably large amount of noise to approximate a real situation and to test the robustness of networks rigorously. Also, the fact that the target data generator is known helps assess how the network approaches the data generator through the cloud of rather noisy data.

This data pattern requires 2 neurons to model the regions of inflection. A larger network of 4 hidden neurons, as shown in Fig. 1b is, therefore, used for the purpose of investigation. In this network, the hidden neuron activation functions are logistic, output neuron is linear, the bias and input-hidden layer neuron weights are depicted by a_{0j} and a_{ij} , respectively, and hidden-output weights and the corresponding bias are denoted by b_j and b_0 , respectively. The network is extensively studied in the following sections for patterns of hidden neuron activation as well as robustness of activation patterns and its potential for structure optimization.

5 Consistency of Network Weights

5.1 Consistency with Respect to Initial Weights

It is desirable that there is just one minimal and consistent set of weights that produces the global minimum error on the error surface and that the network reaches that global optimum regardless of the initial conditions. The data set was randomly divided into 3 sets: training, test and validation, each consisting of 15 observations. The network was trained with the training set based on Levenberg-Marquardt method [1] on Neural Networks for Mathematica [21] and test set was used to prevent over-fitting based on early stopping. (Levenberg-Marquardt is a second order error minimization method that uses the gradient as well as the curvature of the error surface in weight adaptation).

Since the network has excessive weights, it is expected that it will experience over-fitting unless training is stopped early. The performance of the optimum network (validation root mean square error RMSE = 0.318) obtained from early stopping is shown in Fig. 2a (solid line) along with the target pattern (dashed line) and training data. It shows that the network generalizes well. The performance of the over-fitted network that underwent complete training until the training error reached a minimum is illustrated in Fig. 2b. Here, the network fits noise as well due to too much flexibility resulting in over-fitting caused by large magnitude weights. The network has 13 weights and their updates during the first 10 epochs are shown in Fig. 2c. Over-fitting sets in at epoch 2. Two weights that increase drastically are two hidden-output weights.

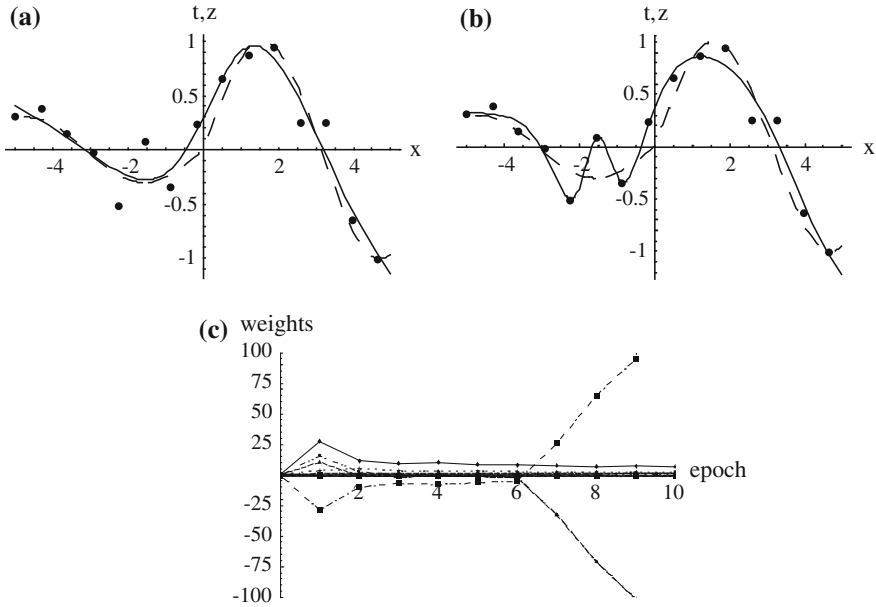
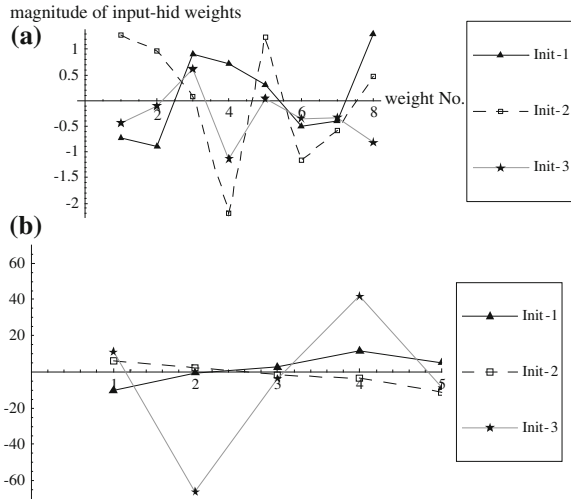


Fig. 2 Network performance: **a** Optimum network performance (*solid line*) plotted with the target data generator (*dashed line*) and training data, **b** over-fitted network performance and **c** Evolution of weights during training [1]

The experiment was repeated twice more with two different random weight initializations. The optimum networks for these two cases produced similar outputs to that shown in Fig. 2a with validation RMSE of 0.364 and 0.299, respectively. However, the first case produced over-fitting with complete training similar to the first weight initialization but the other did not although there were excessive weights. A closer inspection of the evolution of weights for the latter non-overfitting case revealed that 4 of the 13 weights in fact grew to large values after reaching the optimum similar to that shown in Fig. 2c. However, these appear to have pushed the network to operate in the saturated region of the activation functions thereby not affecting the network performance.

Are the optimum weights in these three cases similar? Figure 3a shows the 8 input-hidden neuron weights (a_{01} , a_{11} , a_{02} , a_{12} , a_{03} , a_{13} , and a_{04} , a_{14}) denoted by 1, 2, 3, 4, 5, 6, 7 and 8, respectively, for the three weight initializations and Fig. 3b shows the 5 hidden-output weights (b_0 , b_1 , b_2 , b_3 , b_4) denoted by 1, 2, 3, 4 and 5, respectively. These show that the values for an individual weight as well as the overall pattern across all the weights for the three weight initializations are generally dissimilar. In Fig. 3, the network that did not over-fit was for initialization 3.

Fig. 3 Parallel plots of weights for 3 weight initializations: **a** input-hidden neuron weights and **b** hidden-output neuron weights



5.2 Consistency of Weights with Respect to Random Sampling

A good model must be robust against chance variations in representative samples extracted from the same population. The effect of random sampling on weight structure was tested by training the same network as in Fig. 1b along with the very first set of random initial weights (Init-1 in the previous section) on two more random data sets extracted from the original target function in Fig. 1a. These new datasets are labeled random samples 2 and 3 and the original sample used in the previous section is labeled random sample 1. The optimum network output for sample 2 was similar to that for sample 1 shown in Fig. 2a and had a validation RMSE of 0.270 and produced over-fitting with complete training. Results for sample 3 were interesting in that there was no over-fitting at all with complete training and the weight evolution for this case revealed that weight remained constant after reaching the optimum. The validation RMSE for this case was 0.32. This is natural control of over-fitting by the data and as Siestma and Dow [22] also illustrated, training with properly distributed noise can improve generalization ability of networks.

In order to find out if there is a pattern to the final structure of optimum weights, these are plotted (along with those for the very initial random sample 1 used in the previous section) for comparison in Fig. 4a, b. Here, the weights for random sample 3 stand out in Fig. 4b. Comparison of Figs. 3 and 4 indicate that there is no consistency in the network at this level. However, both non-over-fitted networks- one in Fig. 4b (sample 3) and the other in Fig. 3b (Init-3)-have similar hidden-output weight patterns.

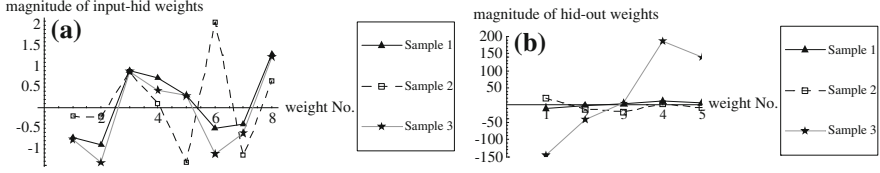


Fig. 4 Final optimum weight structure after training with three random samples: **a** input-hidden weights and **b** hidden-output neuron weights

6 Consistency of Hidden Neuron Activation

Since the weights in Figs. 3 and 4 do not provide any clues as to the existence of an internally consistent pattern, we next explore hidden neuron activation. Activation y_j for each neuron j is a nonlinear transformation of the weighted sum of inputs:

$$y_j = f(a_{0j} + a_{1j}x) = \frac{1}{1 + e^{-(a_{0j} + a_{1j}x)}} \quad (2)$$

The hidden neuron activations for the previous three cases of weight initialization are shown in Fig. 5 as a function of the input x . The Figure reveals an interesting effect. Although actual weights are not identical for the three cases, hidden neuron activations follow some identifiable patterns. For example, the first two cases that produced over-fitting, have a similar pattern of hidden neuron activation whereas the third case that did not produce over-fitting has a unique pattern with only partial similarity to the previous two cases [1].

In the first two cases of weight initialization, early stopping was required to prevent over-fitting. For these, all four activation functions are strongly active in the range of inputs, as indicated by their slopes at the boundary point where the activation y is equal to 0.5, and an external measure is required to suppress their activity. In the third case with no over-fitting however, two of the neurons (solid lines) have low activity and these do not contribute greatly to the output.

Careful observation of Fig. 5 reveals that in the first two cases, there appear to be two sets of neurons each consisting of two neurons of similar activity that could

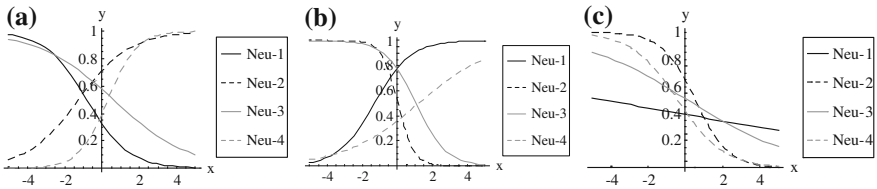


Fig. 5 Activation functions for the 4 hidden neurons for the three weight initializations for random sample 1: **a** Init-1, **b** Init-2 and **c** Init-3 (no-over-fitting)

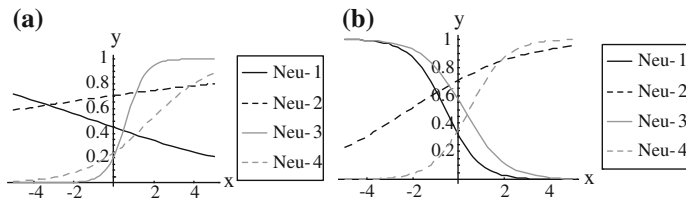


Fig. 6 Hidden neuron function for random samples: **a** sample 2, **b** sample 3 (no over-fitting)

intuitively be interpreted as evidence for redundancy. However, in the third case where there was no over-fitting, there still seem to be two sets of neurons with similar activation but the situation is not so clear. For the case of random sampling, hidden neuron activation patterns for the two networks created from random samples 2 and 3 are shown in Fig. 6. (The patterns for random sample 1 are shown in Fig. 5a).

The activation pattern for sample 3 in Fig. 6b did not produce over-fitting; however, interestingly, this pattern is similar to that for sample 1 which resulted in over-fitting (Fig. 5a, b). Furthermore, now there is a totally new activation pattern for random sample 2 (Fig. 6a) with activation of 3 neurons having a positive slope and one having a negative slope.

Thus, for the six trials (3 weight initializations and 3 random samples), there are 3 distinct activation patterns. In two trial cases, networks did not over-fit but their activation patterns are dissimilar. As for the search for internally consistent weights or activations, there is still ambiguity.

7 Consistency of Activation of Persistent Neurons

It is known that the target data generator used in this study (Fig. 1) requires 2 hidden neurons. The experiment so far indicates that in some cases, there is a persistent 2-neuron structure, such as that in Figs. 5a, b and 6b. In order to confirm if the ones that are persistent point to the optimum, a network with two hidden neurons was tested on the random data sample 1 and with 3 random weight initializations. None of the networks over-fitted even after full training, as expected, since this is the optimum number of neurons. The network produces a closer agreement with the target function and data, similar to that shown in Fig. 2a. However, the hidden neuron functions for the 3 trials produced 3 quite different patterns as shown in Fig. 7.

Figure 7 displays the main features of all previous networks, e.g., activation functions can be all positive, all negative or a combination of positive and negative slopes- and still produce the optimum final output. The figure shows that the optimum network does not have a unique pattern of activation of neurons and still produces the correct output. In order to test the optimality of the two-neuron

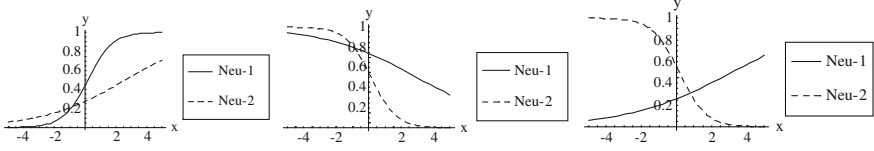


Fig. 7 Hidden neuron activations for the optimum 2-neuron network for 3 random weight initializations for random sample 1

network, another neuron was added and the network trained. This case resulted in a mild form of over-fitting and required early stopping to stop training at the optimum weights. Hidden neuron activation patterns were generally similar to Fig. 5a, b (i.e., 2 functions had negative slope and one had positive slope).

Since the results so far has not yet pointed to a structure that is internally consistent and robust, we next explore hidden neuron contribution to output generation.

8 Internal Consistency of Hidden Neuron Contribution to Output

Contribution of each neuron j to output generation is its weighted activation:

$$y_{\text{weighted}_j} = y_j b_j \quad (3)$$

where y_j is output of hidden neuron j and b_j is the corresponding weight linking neuron j with the output. Returning to our original 4-neuron network, these weighted activation patterns for the first three random weight initializations are presented in Fig. 8.

The plots in Fig. 8 reveal a pattern that is consistent. In each plot, there is one dominant weighted activation pattern with a negative slope. In Fig. 8b, c, the other

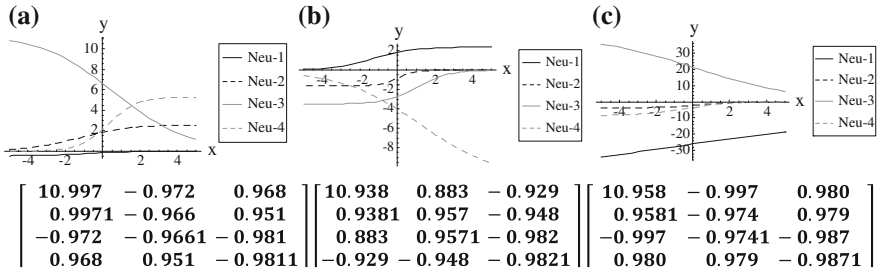


Fig. 8 Weighted hidden neuron activation and correlation matrices for the three random weight initializations for random sample 1: **a** Initialization 1, **b** Initialization 2, and **c** Initialization 3

three patterns are almost parallel to each other. The patterns that are parallel indicate redundancy. In Fig. 8a also, this pattern is obvious but to a lesser extent. These observations for the weighted hidden neuron activation are quite convincing and persistent compared to those observed for the hidden neuron activation in Figs. 5, 6 and 7.

The activation patterns that are parallel can be identified by their strong correlation. This way, it should be possible to eliminate redundant neurons. The correlation matrix for the 3 sets of weighted hidden neuron activation plots are presented below each figure in Fig. 8. The correlation matrices for Fig. 8a, c clearly indicate that neurons 1, 2 and 4 are very highly correlated and all these are inversely correlated with neuron 3 activity. In matrix for Fig. 8b, neurons 1, 2, and 3 are highly correlated and they are inversely correlated with neuron 4 activity. The fact that the correlation coefficients are strong indicate consistency and resilience of the activation patterns. Highly correlated patterns can be replaced by a single representative, leaving two neurons for the optimum network as required. Furthermore, correlation confirms that the optimum network has one neuron with positive weighted activation and another with negative activation for all 3 weight initializations.

The weighted activation patterns and correlations for the network trained with different random samples (samples 2 and 3) are shown in Fig. 9. Results for sample 1 is in Fig. 8a.

Analogous to Fig. 8, highly correlated structure of weighted hidden neuron activation patterns for random samples 2 and 3 is evidenced in Fig. 9 where the left image indicates that neurons 1 and 3 as well as neurons 2 and 4 are highly correlated in an opposite sense. The right image indicates that neurons 1, 2, and 4 are highly correlated with each other and inversely correlated with neuron 3. By replacing the correlated neurons with a representative, an optimum 2-neuron structure is obtained for both these cases.

The above experiment was conducted for 5- and 3-neuron networks with early stopping and results are presented in Fig. 10.

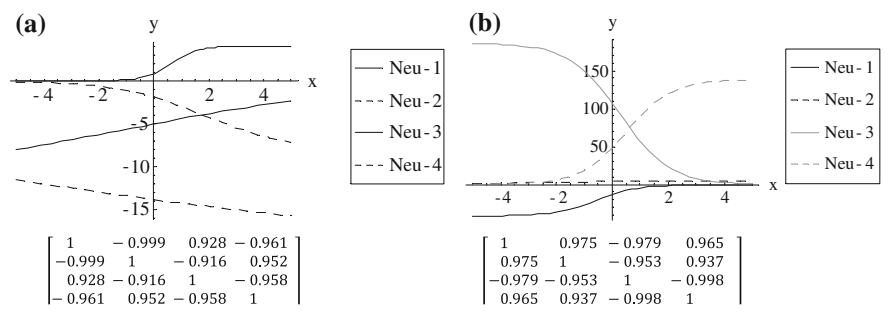


Fig. 9 Weighted hidden neuron activation for the random data samples 2 and 3 and corresponding correlation matrices

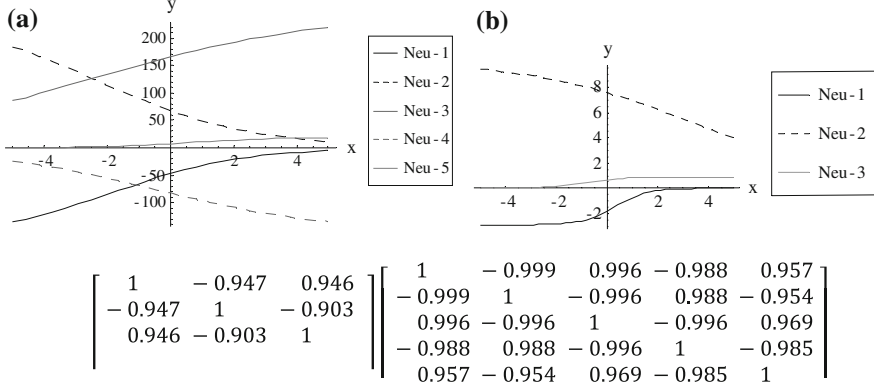


Fig. 10 Weighted hidden neuron activation of a 3-neuron **a** and 5-neuron **b** networks and corresponding correlation matrices

Figure 10 illustrate convincingly that the redundant neurons can be identified by their high correlation. By removing redundant neurons, both networks are left with 2 (optimum number) of neurons.

In summary, the network attains a consistent pattern of weighted hidden neuron activation for this data regardless of the number of hidden neurons, initial weights and random data samples. By replacing highly correlated neurons with similar sign (+ or -) with a single representative, the optimum structure for this example can be obtained with certainty. In what follows, the robustness of weights and hidden neuron activation patterns is further investigated by examining the results obtained from regularization.

9 Internal Structure of Weights Obtained from Regularization

Regularization is another method used to reduce the complexity of a network directly by penalizing excessive weight growth [10]. The amount of regularization is controlled by the parameter δ shown in Eq. 4 where MSE is the mean square error and w_j is a weight in the total set of m weights in the network. In regularization, W is minimized during training.

$$W = MSE + \delta \sum_{j=1}^m w_j^2 \quad (4)$$

The user must find the optimum regularization parameter through trial and error. Too large a parameter exerts too much control on weight growth and too small a value allows too much growth. In this investigation, the original four-neuron

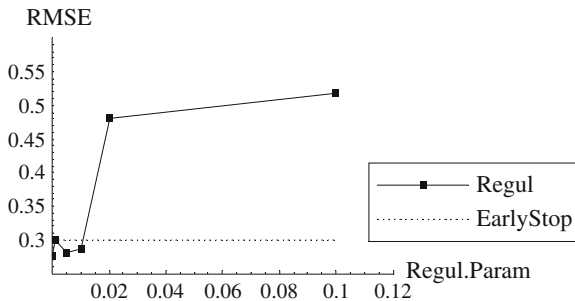


Fig. 11 Comparison of accuracy of networks with weights optimized from regularization and early stopping [1]

network in Fig. 1b with the first set of random initial weights used earlier (Init-1 in Fig. 3) was trained on random sample 1 shown in Fig. 1a. Three regularization parameters, 0.0001, 0.001 and 0.02, were tested. For the first two values, weights initially grew and then dropped to optimum values and from then on they remained constant. For these, the network followed the target pattern closely but the smallest regularization parameter of 0.0001 resulted in the smallest validation MSE of 0.276 which is smaller than that obtained from early stopping (MSE = 0.318). With the parameter value of 0.001, MSE is similar to that obtained from early stopping. For the largest chosen parameter of 0.02, however, weights are controlled too much and therefore, they are not allowed to reach optimum values. In this case, the network performance was very poor.

The experiment was continued further and Fig. 11 shows RMSE for various values of regularization parameter. The horizontal line indicates the RMSE obtained from early stopping. The figure indicates that for this example, regularization can produce networks with greater accuracy than early stopping. However, considering the trial and error nature of regularization, early stopping is efficient. Furthermore, Fig. 11 highlights the sensitivity of RMSE to regularization parameter beyond a certain value.

9.1 Consistency of Weighted Hidden Neuron Activation of Networks Obtained from Regularization

The hidden neuron activations for the two regularization parameters (0.0001 and 0.001) that produced smaller than or similar validation MSE to early stopping are plotted in Fig. 12. They illustrate again that the correlation structure as well as the 2-neuron optimum structure identified in the previous investigations remain persistent.

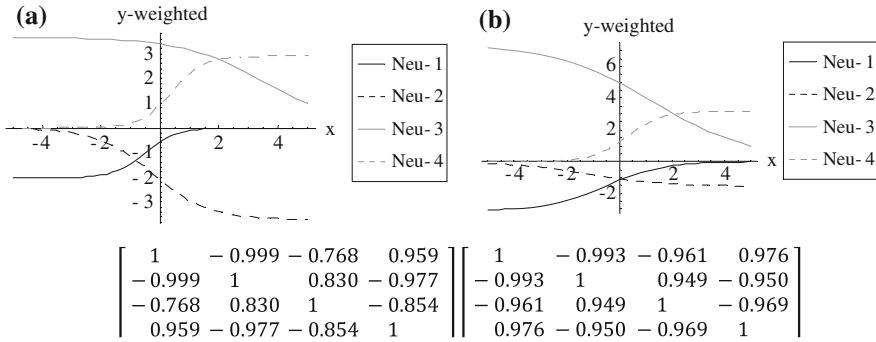


Fig. 12 Weighted hidden neuron activations for regularization parameters 0.0001 **a** and 0.001 **b** and corresponding correlation matrices

10 Identification of Correlated Weighted Hidden Neuron Activations Using Self Organizing Maps

Previous sections demonstrated that the redundant neurons can be identified by the correlation of their weighted neuron activations. It is useful, if these can be identified automatically. In this section, SOM is used to efficiently cluster similar activation patterns. Such approach would be especially useful for larger networks. An input vector to SOM contains weighted hidden neuron activation ($b_j y_j$) for each neuron over the input data. Input vectors were normalized to 0 mean and unit standard deviation and the correlation distance was used as the distance measure [1]. The normalized activation patterns for each network presented so far were mapped to a 2-dimensional SOM [23] (4 neuron map) and the most efficient number of clusters was determined by the Ward clustering [24] of SOM neurons. Ward is an efficient statistical clustering method suitable and effective for relatively small datasets. Figure 13 presents the results for only two networks, one with 4 neurons that was depicted as random weight initialization 2 in Fig. 8b and the other with 5 neurons presented in Fig. 10b. Maps were trained very quickly with default parameter settings of the program [21] indicating the efficiency of clustering highly correlated patterns.

The top two images in Fig. 13 are graphs of Ward likelihood index (vertical axis) against likely number of clusters. The higher the index, more likely that the corresponding number of clusters is the optimum. These images reveal that undoubtedly there are two clusters of activation patterns. The index for other possible cluster sizes is almost zero, which increases the confidence in the two-cluster structure. The bottom images of Fig. 13 show these two clusters on the corresponding SOMs. Here, the two clusters are depicted by brown and black colors, respectively. For example, in the bottom left image, one clusters has 3 correlated patterns distributed in two neurons and the other cluster has one pattern, whereas, in the bottom right image, depicting a 5 neuron network, 2 patterns are grouped into the cluster depicted by the

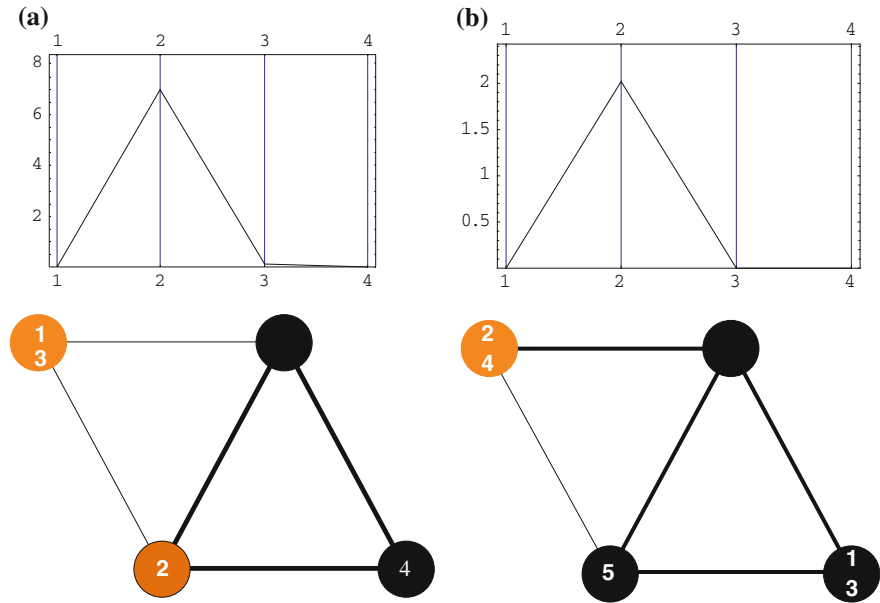


Fig. 13 Self organizing map/Ward clustering of correlated weighted hidden neuron activation patterns. **a** 4-neuron network and **b** 5 neuron network

top left neuron and the other three patterns are spread among the bottom two neurons that form the second cluster. The optimum network structure is obtained by selecting one representative from each cluster and then retraining the network. Similar two-cluster neuron maps were found for all the networks presented previously revealing that, for this example, networks maintain consistent patterns in their internal structure at the level of weighted hidden neuron activation feeding the output neuron. Importantly, the network structure is optimized in one iteration of clustering correlated hidden neuron activation patterns.

11 Ability of the Network to Capture Intrinsic Characteristics of the Data Generating Process

A good model not only should follow the target data but also must capture the underlying characteristics of the data generating process. These can be represented by first and higher order derivative of the generating process. When a network model converges towards the target function, all the derivatives of the network must also converge towards the derivatives of the underlying target function [25, 26]. A new network with two hidden neurons (7 weights in total) was trained and its weighted hidden neuron activation patterns are shown in Fig. 14a that highlights the features already described. The network model is given in Eq. 5 and its first and

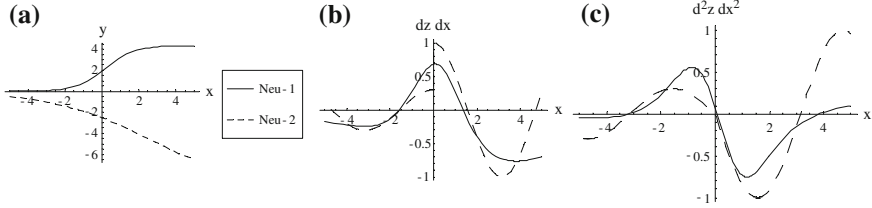


Fig. 14 Optimum network characteristics: **a** weighted hidden neuron activations **b** and **c** gradient and second derivative (solid line), respectively, of network function superimposed on the corresponding values for the target data generator function (dashed line)

second derivatives are superimposed on those of the target function in Fig. 14b, c. They show that, notwithstanding the small sample size and large noise, both derivatives follow the corresponding trends in the target function reasonably well indicating that the trained network is a true representation of the data generating process and can be used for gaining further insight into the process such as sensitivities and errors as well as for further ascertaining the robustness of the weights.

$$z = 0.91 + \frac{4.34}{1 + 1.25e^{-1.27x}} - \frac{9.23}{1 + 2.65e^{-0.37x}} \quad (5)$$

12 Comparison of Correlation Approach with Other Network Pruning Methods

Since it is clear in all the previous experiments that redundant neurons in too flexible networks form highly correlated weighted hidden neuron activation patterns, it is interesting to find out if other pruning methods identify the neurons with the most consistent patterns and prune the redundant ones. A notable feature of the commonly used pruning methods is that they optimize the structure iteratively and require a certain amount of heuristic judgment. In what follows, two pruning methods, Optimal Brain Damage (OBD) [5–7] and Variance Nullity measure (VN) [8] are implemented and compared with the proposed correlation method.

12.1 Network Pruning with Optimum Brain Damage (OBD)

In OBD [5–7], weights that are not important for input-output mapping are found and removed. This is based on a saliency measure of a weight, as given in Eq. 6, that is an indication of the cost of setting it to zero. The larger the s_i , the greater the influence of w_i on error. It is computed from the Hessian (H) which is the matrix containing the

second derivative of error with respect to a pair of weights in the network. This matrix is used in error minimization and weight update by the Levenberg Marquardt method [1]. Since Hessian is nonlocal and computationally intensive, an approximation is used by utilizing only the diagonal entries (H_{ii}) of the Hessian matrix.

$$s_i = H_{ii} w_i^2 / 2 \quad (6)$$

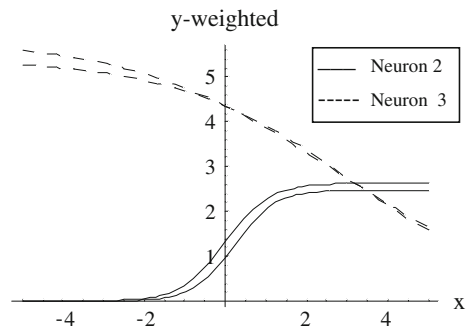
The original 4 neuron network (Fig. 1b) with the first set of initial weights (Init-1) that was trained using a regularization parameter of 0.0001 on the random sample 1 (Fig. 1a) was pruned using saliency measure of the weights. The network was pruned in stages. In the first stage, 5 (or 40 %) of the 13 weights were pruned and the reduced network that retained neurons 2, 3, and 4 was re-trained. The network was further subjected to pruning in the next stage and 2 more weights were removed resulting in a total removal of 7 or (54 %) of the weights from the original network. What remained were neurons 2 and 3 with bias on neuron 2 eliminated leaving 6 weights in the optimum network. The 2 weighted hidden neuron activations for the retrained network are plotted in Fig. 15 by a solid line and a dashed line. (The other set will be discussed shortly). These resemble those of neuron 2 and 3 of the full network in Fig. 8a indicating that the OBD has identified and removed the redundant neurons.

The network performs similarly to that shown in Fig. 2a. Any further pruning resulted in severe loss of accuracy and therefore, the above network was the optimum network obtained from OBD. The output z of the pruned network is [1]

$$z = -5.36 + \frac{2.63}{1 + e^{-1.84x}} + \frac{5.86}{1 + 0.345e^{0.399x}} \quad (7)$$

Equations 5 and 7 are not identical as the network obtained from the proposed correlation method has all 7 weights associated with the hidden and output neurons whereas the one from OBD has only 6 weights. This also reveals that the network can still have redundant bias weights. If a set of weights that are invariant is desired, these redundant weights can be pruned with potentially one extra weight pruning step applied to the trained network with the optimum number of neurons.

Fig. 15 Weighted hidden neuron activation patterns for networks pruned by OBD and Variance nullity



12.2 Network Pruning Based on Variance of Network Sensitivity

In this section, the same original network used in the previous section is pruned using a very different method- variance analysis of the sensitivity of the output of the network to perturbation of weights- as proposed by Engelbrecht [8]. Variance nullity (VN) measure tests whether the variance of the sensitivity of network output over all input-output patterns is significantly different from zero. It is based on a hypothesis test using χ^2 (chi square) distribution to test statistically if the parameter should be pruned. If the sensitivity with respect to a parameter is denoted by S_θ , then the variance of the sensitivity for N patterns can be expressed as

$$\sigma_{S_\theta}^2 = \frac{\sum_{i=1}^N (S_{\theta i} - \mu_{S_\theta})^2}{N} \quad (8)$$

where μ_{S_θ} is the mean sensitivity. This is used to obtain an expression for a variance nullity measure $\gamma_{s\theta}$, that indicates the relevance of a parameter as

$$\gamma_{s\theta} = \frac{(N-1)\sigma_{s\theta}^2}{\sigma_0^2} \quad (9)$$

where σ_0^2 is a value close to zero. The hypothesis that the variance is close to zero is tested for each parameter θ with the null and alternative hypotheses of

$$\begin{aligned} H_0 : \sigma_{s\theta}^2 &= \sigma_0^2 \\ H_1 : \sigma_{s\theta}^2 &< \sigma_0^2 \end{aligned} \quad (10)$$

Under the null hypothesis, $\gamma_{s\theta}$ follows a χ^2 (N-1) where N-1 is the degree of freedom. A parameter is removed if the alternative hypothesis is accepted with the condition $\gamma_{s\theta} \leq \gamma_c$ where γ_c is a critical χ^2 value obtained from $\gamma_c = \chi_{N-1, (1-\alpha/2)}^2$. The α is the level of significance which specifies the acceptable level of incorrectly rejecting null hypothesis. Smaller values result in a stricter pruning algorithm.

The success of the algorithm depends on the value chosen for σ_0^2 . If it is too small, no parameter is pruned. If too large, even relevant parameters will be pruned. Thus, some trial and error is necessary. The method was applied to the original 4-neuron network described in this chapter with an initial value of 0.01 for σ_0^2 at 0.05 significance level. Only one weight was targeted for pruning. When it was increased to 0.1, all six weights associated with neurons 1 and 4 became targets for pruning leaving those for neurons 2 and 3, that are the required neurons, with all 7 corresponding weights. This outcome, in terms of exactly which neurons remain, is similar to that obtained from OBD in the previous section and in both these cases, variance nullity and OBD, considerable subjective judgment is required in setting

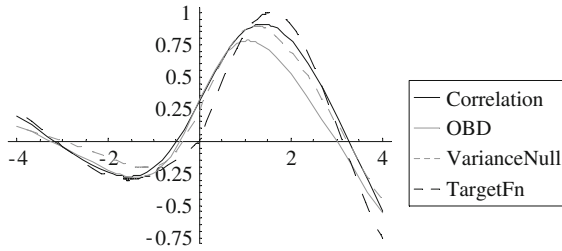


Fig. 16 The pruned network based on correlated activity patterns plotted with networks pruned by OBD and variance nullity and the target function

up the parameters. The weighted hidden neuron activations for the retrained network obtained from variance nullity method are plotted in Fig. 15b indicating that the two methods produce almost identical activation patterns. The corresponding plot for the network obtained from the correlation method was shown in Fig. 14a. Here the shape of the patterns are the same as those for VN and OBD based networks; however, since the initial weights are different in this case, one activation pattern has shifted vertically but this effect is offset by the larger bias weight (0.91 in Eq. 5) on the output neuron when the final output is produced. The important point is that the slopes and trends of the patterns are identical for the 3 methods.

The final network output from variance nullity based weight pruning is

$$z = -5.09 + \frac{2.48}{1 + 1.57e^{-1.99x}} + \frac{5.38}{1 + 0.235e^{0.465x}} \quad (11)$$

which is very similar to that obtained from OBD (Eq. 7). Reason why Eqs. 7 and 11 are similar is that both of them were retrained with the same set of original initial weights that remained on the network after pruning. In the network obtained from the correlation method, new initial weights were used as only one representative from each cluster was used. The three network outputs are superimposed on the target function in Fig. 16.

Figure 16 reveals that the performance of the two full networks with 7 weights obtained from the proposed method and variance nullity method is closer to the target function than that with 6 weights obtained from OBD. The validation RMSE from the proposed correlation method, variance nullity and OBD were, 0.272, 0.272 and 0.285, respectively. However, the proposed method is more efficient and does not require heuristic judgment as in OBD and Variance Nullity. This point applies to other past approaches for structure optimization, such as singular value decomposition as well. The validation RMSE for the correlation based network (0.272) is slightly smaller than that for the best full networks obtained from regularization (0.276) and early stopping (0.318).

13 Robustness and Uncertainty of Networks

13.1 Robustness

The fact that the optimum networks do not match the target pattern perfectly is due to the large noise deliberately added to the data generated from the true function. The noise allows a number of possible outputs that follow the target function closely. Smaller the noise, the tighter the band around the target function within which output of various optimum networks can lie. In order to test this interval, optimum weights were perturbed by randomly adding noise from a Gaussian distribution with 0 mean and standard deviations of 0.01, 0.05, 0.1, to 0.2. Thus there were 4 sets of weights. The network output for these 4 sets of weights showed that the weights are robust against variations up to 0.1 standard deviation which is equivalent to $\pm 30\%$ random perturbation of the weights. The 0.2 standard deviation representing $\pm 60\%$ random perturbations was detrimental to the network performance (see p. 238 of [1]).

13.2 Confidence Interval for Weights

Since the weights are robust against perturbation of at least up to $\pm 30\%$, confidence intervals for weights were developed for a noise level of $\pm 15\%$ (noise standard deviation of 0.05). Ten sets of weights, each representing a network, were drawn by superimposing noise on the optimum weights of the network obtained from the proposed approach based on correlation of weighted hidden neuron activation. Confidence intervals were constructed using methods of statistical inference based on sampling distribution as:

$$(1 - \alpha)CI = \bar{w} \pm t_{\alpha, n-1} \frac{s_w}{\sqrt{n}} \quad (12)$$

where \bar{w} is the mean value of a weight, s_w is the standard deviation of that weight, and n is the sample size. In this case, we have 10 observations. The $t_{\alpha, n-1}$ is the t -value from the t -distribution for $(1-\alpha)$ confidence level and degree of freedom (dof) of $n-1$. The 95 % confidence intervals were constructed for each of the 7 weights and the resulting 95 % Confidence Intervals (CIs) for the network performance are plotted in Fig. 17a with the two solid lines depicting upper and lower limits. In this figure, the smaller dashed line represents the mean and larger dashed line is the target function.

In order to assess all the models developed so far, network outputs from 4 random weight initializations using the proposed method involving correlation of weighted hidden neuron activations were superimposed along with outputs from OBD and variance nullity (6 curves altogether) on the above confidence interval

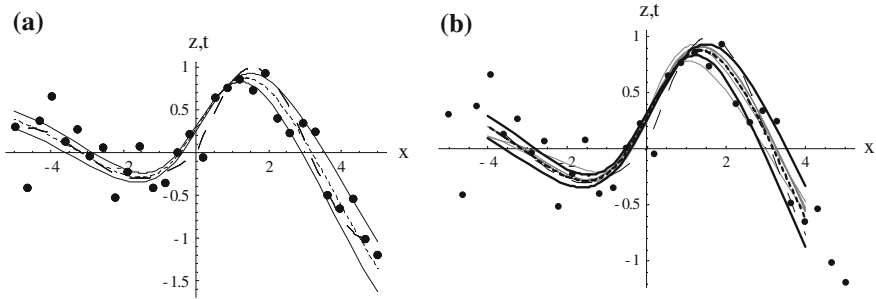


Fig. 17 Confidence Interval (CI) bands and comparison of optimum networks: **a** Mean and 95 % confidence interval limits for the correlation based network and **b** optimum network performance from the 3 methods superimposed on the CIs

plots containing the original data. The results are shown in Fig. 17b that illustrates that all models are within the confidence limits for the entire range of the data and covers most of the target function (larger dashed line). The target function is the combination of two functions (Eq. 1 and also see Fig. 14b) and all networks experience difficulty in the region near the axes origin where the two functions merge.

13.3 2-D Function Approximation

In this section, the correlation of weighted hidden neuron activation is tested on a two-dimensional problem. The function from which 120 data vectors were generated is shown in Fig. 18a. The network was trained with 15 hidden neurons with sigmoid functions and linear output function. Training was done with Levenberg Marquardt method with early stopping to optimise the network. The optimum network output is shown in Fig. 18b. After training, weighted hidden neuron activations were analysed and the correlation matrix is given in Fig. 18c.

The weighted hidden neuron activations were projected onto a 16-neuron SOM and trained SOM weights were clustered with Ward clustering. Figure 19a shows the Ward index plot which clearly indicates 7 clusters as the optimum. The SOM clustered into 7 groups are shown in Fig. 19b.

A new network was trained with 7 hidden neurons and results identical to Fig. 18b was found confirming that the optimum number of hidden neurons is 7. In order to test further, individual networks were trained with hidden neuron numbers increasing from 1 to 10 with a number of weight initializations. Root Mean Square Error (RMSE) plot for these cases are shown in Fig. 20 which clearly indicates that the 7 neurons do provide the minimum error.

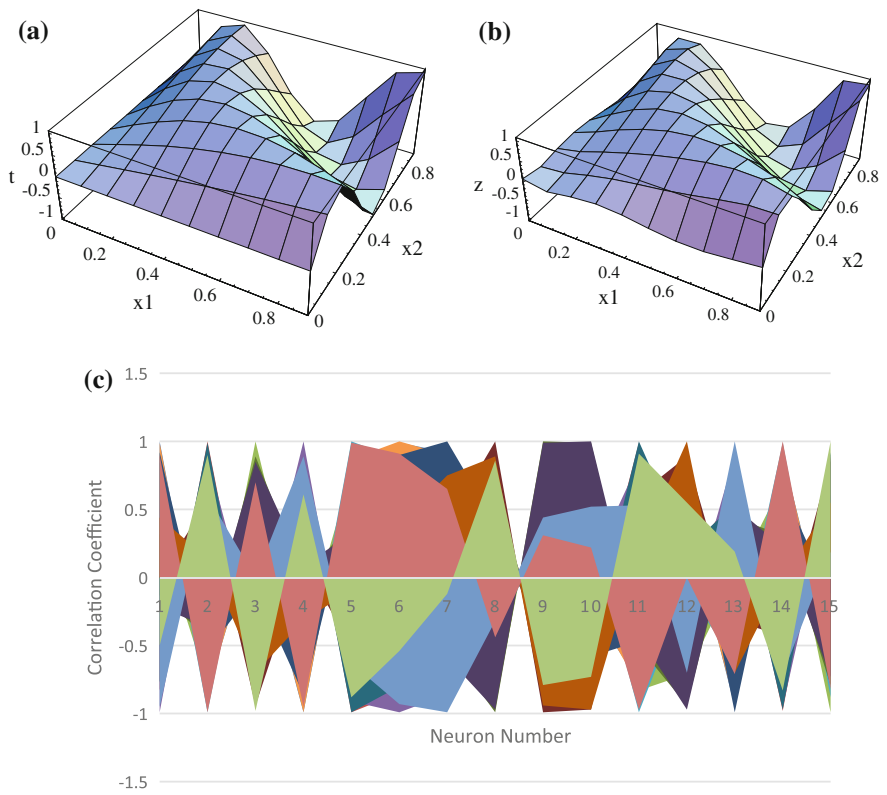


Fig. 18 **a** Two-dimensional target function, **b** and network prediction and **c** correlation of weighted activation of the 15 hidden neurons in the network (colours visually display the general character and strength of correlations across neurons)

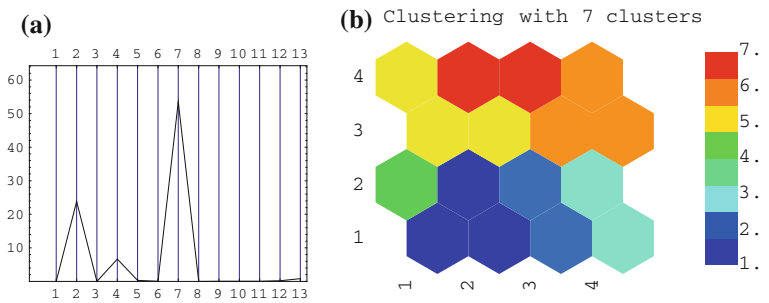


Fig. 19 **a** Ward index against number of clusters; **b** SOM clustered into optimum number of 7 clusters: yellow (neurons 1, 6, 7), red (5, 14), brown (3, 9, 10), cyan (4, 13), pale blue (2, 11), dark blue (8, 15) and green (12)

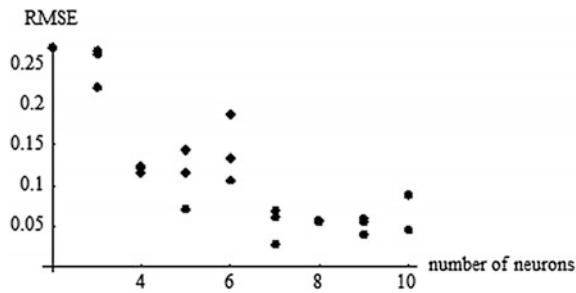


Fig. 20 RMSE for increasing number of hidden neurons trained for a number of random weight initialisations

14 Optimising a Network for Practical Real-Life Problems

14.1 Breast Cancer Classification

Correlation of weighted hidden neuron networks were also tested on a real world problem of breast cancer classification. In this study, a total of 99: 46 malignant and 53 benign, samples obtained from The Digital Database for Breast Ultrasound Image (DDBUI) was used to develop a feed forward network for breast cancer classification. Using a preprocessing and feature selection approach, 6 features were selected with the ability to discriminate between cancer and healthy cases. These were: depth-width ratio of the dense mass and its shape and margin, blood flow, age and a newly identified effective feature called central regularity degree (CRD) that explicitly incorporates irregularity of the mass that has been known to be indicative of malignancy [27].

Networks were developed with sigmoid hidden and output neurons on 70 % and 30 % training and testing data, respectively, and trained with Levenberg Marquardt method and early stopping. First a network with a large number of hidden neurons was developed and then the number of neurons were decreased gradually, every time comparing results with previous results. It turned out that 15 hidden neurons provide optimum results: Training (100 % Sensitivity, Specificity and Accuracy) and Testing (100, 90.9, 95.4 %, respectively, for the above measure). Then we tested the clustering of weighted hidden neuron activation approach on the best network using SOM topology with 20 neurons. The trained SOM results are shown in Fig. 21 where several individual SOM neurons represent a number hidden neurons as indicated by Fig. 21a—hidden neuron groups (4, 7, 10), (6, 12) and (14, 15) each share an SOM neuron. Other neurons are each represented by an individual SOM neuron. The U-matrix in Fig. 21b shows further similarity among the nodes. For example, neurons (8, 1, 9) were found close to each other (blue colour on the top right corner of the map) and were considered as one cluster by Ward clustering that divided the SOM into 9 clusters suggesting that 9 neurons should adequately model the data.

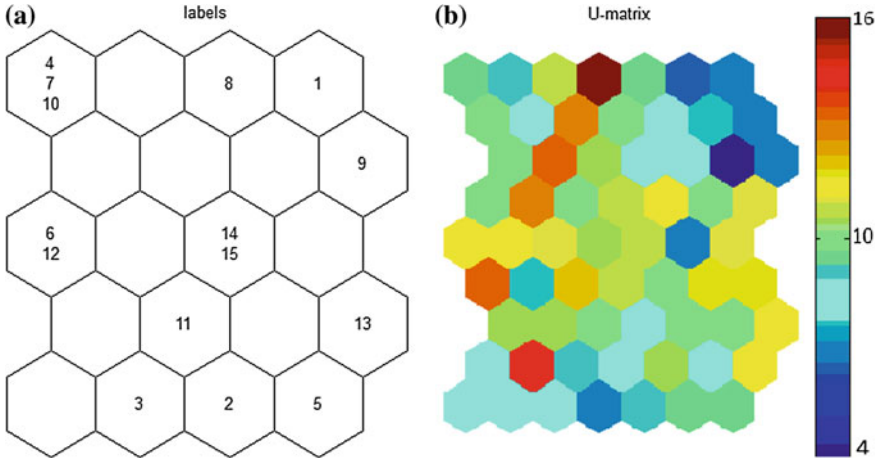


Fig. 21 Twenty neuron SOM representing hidden neuron activation patterns. **a** Distribution of 15 neurons over the map; **b** U-matrix for the 15 hidden neurons (*blue colour indicates similarity/closeness*)

To test this finding, a 9 neuron network was trained and tested on the same data sets as before and the network accuracy (95.4 %), sensitivity (100 %) and specificity (90.9 %) were found to be the same as those for the 15 neuron network. Thus the SOM/Ward reduced the number of neurons without any compromise on the network performance confirming that the redundant weighted hidden neuron activations do form correlated clusters and the number of these clusters indicate the required number of neurons.

14.2 River Flow Forecasting

The efficacy of the correlation method was tested in another complex real world problem of river flow forecasting for a multi-basin river system in New Zealand and the detail results were presented in [28]. Forecasting river flows are very complicated due to the effect of daily, monthly, seasonal and yearly variability of the contributing factors of rainfall and temperature etc. The inputs, selected from an extensive feature selection process, were: previous month's flow, current temperature and a river basin geometric factor and the output was current month's flow. The data were divided into training (70 %) and calibration (30 %) with 1079 and 269 data points, respectively, and validation set with 63 data points. In the original study, it was found that 70 hidden neurons (logistic activation) and one linear output neuron provided the optimum results.

To test this result, a network with 100 hidden neurons with logistic function was trained and the weighted hidden activations of 100 neurons were projected onto a 100 neuron square SOM. Results showed that the 100 patterns were projected onto 59 SOM neurons and the Ward method further clustered these neurons indicating that 2 and 3 neurons provided the highest Ward Likelihood Index, which is much smaller than the original optimum of 70 neurons found by trial and error. A 2-neuron network was trained and the results (training $R^2 = 0.88$; Validation $R^2 = 0.71$) were similar to original results [28]. Results for 59 and 70 neurons were similar.

15 Summary and Conclusions

This chapter presented the results from a systematic investigation of the internal consistency and robustness of feed forward (multi-layer perception) networks. It demonstrated that weighted hidden neuron activations feeding the output neuron display meaningful and consistent patterns that are highly correlated for redundant neurons. By representing each correlated group with one neuron, the optimum structure of the network is obtained. Furthermore, the chapter illustrated that the correlated activation patterns can be mapped on to a self organizing map (SOM) where Ward clustering convincingly revealed the required number of clusters. The chapter also compared the proposed method with two pruning approaches from literature: Optimal Brian Damage (OBD) and Variance Nullity (VN) and demonstrated the efficacy of the proposed correlation based method. A clear advantage of the correlation method is that it does not require heuristic judgment in selecting parameters for optimizing the network as in other methods. Another advantage is that network is optimized in one step of clustering correlated weighted hidden neuron activation patterns thus minimizing the time and effort spent on structure optimization. Yet another advantage is that as the redundant neurons are highly correlated, they cluster easily on the SOM with default network learning parameters and Ward clustering automatically produces the required optimum number of neurons. This chapter used a one-dimensional problem to allow the presentation of a thorough assessment of various modeling issues deemed important and demonstrated that the insights gained are relevant to larger problems as well by successfully applying the concept to multi-dimensional and complex real world problems. These demonstrated that the approach is robust to initial weights, random samplings and for networks with logistic activation function and either linear or logistic output neuron activation function. In future, it will be useful to test the validity of the method for other activation functions and networks.

Appendix: Algorithm for Optimising Hidden Layer of MLP Based on SOM/Ward Clustering of Correlated Weighted Hidden Neuron Outputs

I. Train an MLP with a relatively larger number of hidden neurons

1. For input vector X , the weighted input u_j and output y_j of hidden neuron j are:

$$u_j = a_{0j} + \sum_{i=1}^n a_{ij}x_i$$

$$y_j = f(u_j)$$

where a_{0j} is bias weight and a_{ij} are input-hidden neuron weights. f is transfer function.

2. The net input v_k and output z_k of output neuron k are:

$$v_k = b_{0k} + \sum_{j=1}^m b_{jk}y_j$$

$$z_k = f(v_k)$$

where b_{0k} is bias weight and b_{jk} are hidden-output weights.

3. Mean Square error (MSE) for the whole data set is:

$$MSE = \frac{1}{2N} \left[\sum_{i=1}^N (t_i - z_i)^2 \right]$$

where t is target and N is the sample size.

4. Weights are updated using a chosen method of least square error minimisation, such as Levenberg Marquardt method:

$$w_m = w_{m-1} - \varepsilon R d_m$$

where d_m is sum of error gradient of weight w for epoch m , R is inverse of curvature, and ε is learning rate.

5. Repeat the process 1 to 4 until minimum MSE is reached using training, calibration (testing) and validation data sets.

II. SOM clustering of weighted hidden neuron outputs

Inputs to SOM

An input vector X_j into SOM is:

$$X_j = y_j b_j;$$

where y_j is output of hidden neuron j and b_j is its weight to output neuron in MLP. Length n of the vector X_j is equal to the number of samples in the original dataset.

Normalise X_j to unit length

SOM training

1. *Projecting weighted output of hidden neurons onto a Self Organising Map:*

$$u_j = \sum_{i=1}^n w_{ij}x_i$$

where u_j is output of SOM neuron j and w_{ij} is its weight with input component x_i

2. *Winner selection:* Select winner neuron based on the minimum correlation distance between an input vector and SOM neuron weight vectors (same as Euclidean distance for normalised input vectors)

$$d_j = \sqrt{\sum_{i=1}^n (x_i - w_{ij})^2}$$

3. *Update of weights of winner and neighbours at iteration t :*

Select neighbourhood function $NS(d, t)$ (such as Gaussian) and learning rate function $\beta(t)$ (such as exponential or linear) where d is distance from winner to a neighbour neuron and t is iteration.

$$w_j(t) = w_j(t-1) + \beta(t)NS(d, t)[x(t) - w_j(t-1)]$$

4. *Repeat the process until mean distance D between weights W_i and inputs x_n is minimum.*

$$D = \sum_{i=1}^k \sum_{n \in c_i} (x_n - w_i)^2$$

where k is number of SOM neurons and c_i is the cluster of inputs represented by neuron i

III. Clustering of SOM neurons

Ward method minimizes the within group sum of squares distance as a result of joining two possible (hypothetical) clusters. The within group sum of squares is the sum of square distance between all objects in the cluster and its centroid. Two clusters that produce the least sum of square distance are merged in each step of

clustering. This distance measure is called the Ward distance (d_{ward}) and is expressed as:

$$d_{ward} = \frac{(n_r^* n_s)}{(n_r + n_s)} \|x_r - x_s\|^2$$

where x_r and x_s are the centre of gravity of two clusters. n_r and n_s are the number of data points in the two clusters.

The centre of gravity of the two merged clusters $x_{r(new)}$ is calculated as:

$$x_{r(new)} = \frac{1}{n_r + n_s} (n_r^* x_r + n_s^* x_s)$$

The likelihood of various numbers of clusters is determined by *WardIndex* as:

$$WardIndex = \frac{1}{NC} \left(\frac{d_t - d_{t-1}}{d_{t-1} - d_{t-2}} \right) = \frac{1}{NC} \left(\frac{\Delta d_t}{\Delta d_{t-1}} \right)$$

where d_t is the distance between centres of two clusters to be merged at current step and d_{t-1} and d_{t-2} are such distances in the previous two steps. NC is the number of clusters left.

The numbers of clusters with the highest *WardIndex* is selected as the optimum.

IV. Optimum number of hidden neurons in MLP

The optimum number of hidden neurons in the original MLP is equal to this optimum number of clusters on the SOM.

Train an MLP with the above selected optimum number of hidden neurons.

References

1. S. Samarasinghe, *Neural Networks for Applied Sciences and Engineering-From Fundamentals to Complex Pattern Recognition* (CRC Press, 2006)
2. C. Bishop, *Neural Networks for Pattern Recognition* (Clarendon Press, Oxford, UK, 1996)
3. S. Haykin, *Neural Networks: A comprehensive Foundation*, 2nd edn. (Prentice Hall Inc, New Jersey, USA, 1999)
4. R. Reed, Pruning algorithms-A survey. *IEEE Trans. Neural Networks* **4**, 740–747 (1993)
5. Y. Le Cun, J.S. Denker, S.A. Solla, Optimal brain damage, in *Advances in Neural Information Processing (2)*, ed. by D.S. Touretzky (1990), pp. 598–605
6. B. Hassibi, D.G. Stork, G.J. Wolff, Optimal brain surgeon and general network pruning. *IEEE International Conference on Neural Networks*, vol. 1, (San Francisco, 1992), pp. 293–298
7. B. Hassibi, D.G. Stork, Second-order derivatives for network pruning: Optimal brain surgeon, in *Advances in Neural Information Processing Systems*, vol. 5, ed. by C. Lee Giles, S. J. Hanson, J.D. Cowan, (1993), pp. 164–171
8. A.P. Engelbrecht, A new pruning heuristic based on variance analysis of sensitivity information. *IEEE Trans. Neural Networks* **12**(6), 1386–1399 (2001)

9. K. Hagiwara, Regularization learning, early stopping and biased estimator. *Neurocomputing* **48**, 937–955 (2002)
10. M. Hagiwara, Removal of hidden units and weights for backpropagation networks. *Proc. Int. Joint Conf. Neural Networks* **1**, 351–354 (1993)
11. F. Aires, Neural network uncertainty assessment using Bayesian statistics with application to remote sensing: 1. Network weights. *J. Geophys. Res.* **109**, D10303 (2004). doi:[10.1029/2003JD004173](https://doi.org/10.1029/2003JD004173)
12. F. Aires, Neural network uncertainty assessment using Bayesian statistics with application to remote sensing: 2. Output Error. *J. Geophys. Res.* **109**, D10304 (2004). doi:[10.1029/2003JD004174](https://doi.org/10.1029/2003JD004174)
13. F. Aires, Neural network uncertainty assessment using Bayesian statistics with application to remote sensing: 3. Network Jacobians. *J. Geophys. Res.* **109**, D10305 (2004). doi:[10.1029/2003JD004175](https://doi.org/10.1029/2003JD004175)
14. K. Warne, G. Prasad, S. Rezvani, L. Maguire, Statistical computational intelligence techniques for inferential model development: A comparative evaluation and novel proposition for fusion. *Eng. Appl. Artif. Intell.* **17**, 871–885 (2004)
15. I. Rivals, L. Personnaz, Construction of Confidence Intervals for neural networks based on least squares estimation. *Neural Networks* **13**, 463–484 (2000)
16. E.J. Teoh, K.C. Tan, C. Xiang, Estimating the number of hidden neurons in a feed forward network using the singular value decomposition *IEEE Trans. Neural Networks* **17**(6), (2006)
17. C. Xian, S.Q. Ding, T.H. Lee, Geometrical interpretation and architecture selection of MLP, *IEEE Trans. Neural Networks* **16**(1), (2005)
18. P.A. Castillo, J. Carpio, J.J. Merelo, V. Rivas, G. Romero, A. Prieto, Evolving multilayer perceptrons. *Neural Process. Lett.* **12**(2), 115–127 (2000)
19. X. Yao, Evolutionary artificial neural networks. *Proc. IEEE* **87**(9), 1423–1447 (1999)
20. S. Samarasinghe, Optimum Structure of Feed Forward Neural Networks by SOM Clustering of Neuron Activations. *Proceedings of the International Modelling and Simulation Congress (MODSM)* (2007)
21. *Neural Networks for Mathematica*, (Wolfram Research, Inc. USA, 2002)
22. J. Sietsma, R.J.F. Dow, Creating artificial neural networks that generalize. *Neural Networks* **4**(1), 67–77 (1991)
23. Machine learning framework for Mathematica. 2002 Uni software Plus. www.unisoftwareplus.com
24. J.H. Ward Jr, Hierarchical grouping to optimize an objective function. *J. Am Stat. Assoc.* **58**, 236–244 (1963)
25. K. Hornik, M. Stinchcombe, H. White, Universal approximation of an unknown mapping and its derivatives using multi-layer feedforward networks. *Neural Networks* **3**, 551–560 (1990)
26. A.R. Gallant, H. White, On learning the derivative of an unknown mapping with multilayer feedforward networks. *Neural Networks* **5**, 129–138 (1992)
27. A. Al-yousef, S. Samarasinghe, Ultrasound based computer aided diagnosis of breast cancer: Evaluation of a new feature of mass central regularity degree. *Proceedings of the International Modelling and Simulation Congress (MODSM)* (2011)
28. S. Samarasinghe, Hydrocomplexity: New Tools for Solving Wicked Water Problems
Hydrocomplexité: Nouveaux outils pour solutionner des problèmes de l'eau complexes (*IAHS Publ.* 338) (2010)

Artificial Neural Network Modelling

Shanmuganathan, S.; Samarasinghe, S. (Eds.)

2016, VII, 472 p. 187 illus., 124 illus. in color.,

Hardcover

ISBN: 978-3-319-28493-4