Cent. Eur. J. Comp. Sci. • 3(3) • 2003 • 105-115
DOI: 10.2478/s13537-013-0109-x

VERSITA

Central European Journal of **Computer Science**

Pruning algorithms of neural networks - a comparative study

Review Article

M. Gethsiyal Augasta[1*], T. Kathirvalavakumar[2†]

1 *Department of Computer Applications, Sarah Tucker College, Tirunelveli, 627007, TN, India*

2 *Department of Computer Science, V.H.N.S.N College, ViruduNagar, 626001, TN, India*

Received 27 March 2013; accepted 06 September 2013

Abstract: The neural network with optimal architecture speeds up the learning process and generalizes the problem well for further knowledge extraction. As a result researchers have developed various techniques for pruning the neural networks. This paper provides a survey of existing pruning techniques that optimize the architecture of neural networks and discusses their advantages and limitations. Also the paper evaluates the effectiveness of various pruning techniques by comparing the performance of some traditional and recent pruning algorithms based on sensitivity analysis, mutual information and significance on four real datasets namely Iris, Wisconsin breast cancer, Hepatitis Domain and Pima Indian Diabetes.

Keywords: input and hidden neurons pruning • optimization techniques • classification • feedforward neural networks • data mining

1. Introduction

Neural networks often achieve high classification accuracy but they have some drawbacks related in particular to their long training time requirement and the determination of the most efficient network structure. The power of the neural network depends on how well it generalizes new data following training. Generalization is the ability of the neural network to interpolate and extrapolate the data that it has not seen before [1]. The generalization capabilities of ANNs depend on the size of the training data, training epochs, and architecture of the network. The success of ANNs largely depends on their architecture, which is usually determined by a trial and error process but sometimes by growing method or pruning method. Many algorithms have been used in numerous ways to optimize the network architecture [2–5].

Researchers have proposed a large number of neural network(NN) structures (such as feedforward neural networks(FNN), recurrent neural networks, Hopfield neural networks, etc.) and applied to various real world problems. The

* *E-mail: augasta@yahoo.com*
† *E-mail: Kathirvalavakumar@yahoo.com (Corresponding author)*

feedforward neural network is by far the most popular architecture due to its structural flexibility, good representational capabilities, and the availability of a large number of training algorithms [3]. A typical feedforward neural network contains an input layer, one or more hidden layers and an output layer. The number of nodes in the input layer is equal to the number of attributes of the dataset, the number of nodes in the output layer is equal to the number of target classes of the dataset and the number of hidden layers and the number of nodes in each hidden layer depends on the complexity of the problem [5].

Generally, the FNN with large number of hidden nodes is able to learn fast and avoids local minima as when a network has too many free parameters such as weights and/or nodes, a local minima are more easily avoided [6, 7]. But if the network is too large, it yields more nodes, more weights and more layers than necessary; and thus results in unnecessary arithmetic calculations and high computation cost. Also an oversized network may overfit the training data and has poor generalization ability for the testing data [8]. The better generalization performance can be achieved only by the small networks but their training may require lot of effort since the networks may not have enough processing elements [9]. Moreover a small trained network is easier to interpret and the knowledge can be easily extracted in the form of simple rules [10]. From the implementation point of view small networks only require limited resources in any physical environment. At the same time large networks may exhibit a certain degree of fault tolerance under damage conditions [12]. So both large and small networks exhibit a number of advantages and disadvantages. The optimal architecture is a network that is large enough to learn the problem and is small enough to generalize well. This paper discuss on existing algorithms that finds the optimal architecture by pruning method. The paper is organized as follows: Section 2 discusses various approaches of network pruning, Section 3 provides a survey of existing pruning methods and discusses its advantages and limitations, Section 4 compares the performance of some existing traditional and recent pruning algorithms based on sensitivity, mutual information and significance in terms of pruning percentage, pruning speed and classification accuracy by implementing them on four different real datasets namely iris, breast-cancer, hepatitis and diabetes.

2. Network pruning approaches

Pruning is defined as a network trimming within the assumed initial architecture. This can be accomplished by estimating the sensitivity of the total error to the exclusion of each weight in the network. The weights or neurons which are insensitive to the error changes can be discarded after each step of training. The pruned network is of smaller size and is likely to give higher accuracy than before its trimming [12]. Pruning algorithms are used to remove the redundant connections while maintaining the networks performance. Moreover a simple and comprehensible set of rules can be extracted from the network by removing the input neurons that are not mandatory for solving the problem [13, 14]. Several pruning algorithms are available in the literature to determine irrelevant neurons [15, 16].

Brute-force is the most common approach, to find the neural network with optimal architecture [9]. It is based on successive training of some smaller networks, until the best smallest topology which still fits the data is found. This process is very time consuming since many networks have to be trained. In addition, for initially small networks, the convergence is always not guaranteed [17]. The other common approaches for optimizing neural network architecture are basically growing, pruning and a combination of two strategies namely growing and pruning [5]. The first, also called as constructive methods, start with a minimal network and add new hidden nodes during the training process [18–20]. A drawback of growing methods is that the initial small network can easily be trapped into local minima and the training time may be increased [15]. The second referred as destructive methods, start with a large network and then remove the unimportant nodes or weights [21, 22]. This method combines the advantages of both large and small networks. However it requires the upper bound size of the large networks for the problem at hand, but this is not a serious concern as the methods have been established to determine the upper bounds on the number of hidden nodes [23].

Castellano et al. have shown that, in any case the overall time required for training a large network and then pruning it to a small size compares very favorably with that of simply training a small network [17]. So one can use the larger networks for training and its generalization can be improved by the process of pruning. There are also hybrid methods which can both add and remove [24, 25]. These hybrid methods start training with a small network and incrementally add hidden nodes during training when the network cannot reduce the training error.

3. Survey of pruning algorithms

Researchers have suggested many pruning algorithms [26–29] for optimizing the architecture of neural networks. Based on the techniques used for pruning, the pruning methods can be classified as penalty term methods, cross validation methods, magnitude based methods, Mutual Information (MI) based methods, Evolutionary pruning methods, Sensitivity Analysis (SA) based methods and Significance based pruning methods.

Penalty method (or weight decay method) adds a penalty term to the objective function to be minimized so that the smaller weights can eventually be forced to zero. Rudy Setiono [21] has proposed the penalty function which discourages the use of unnecessary connections and prevents the weights of the connections from taking large values. He also has proposed the simple scheme for removing redundant weights from a network which has been trained to minimize a penalty function. But this approach may eliminate weights that are actually crucial to the over all architecture of the networks and may create additional local minima on the error surface during training. Wan et al. [30] have proposed two algorithms for controlling the hidden layers, one is through adding penalty terms on the error function, in this way when one weight is updated, the effect of other weights are also considered, that is in the updating rule, the derivatives of the added terms are function of not only the current weight, but also other weights. The effect of penalty terms is to make the neural networks robust to the noises in the samples. The second proposed algorithm uses Gauss Schmidt algorithm to determine which nodes are principal nodes in one epoch, the weights connected to these principal nodes are given the chance to be updated in this epoch, while the other weights are remained to be unchanged in this epoch.

In cross validation method, the pruning criterion is still based on the magnitude of each weight but a validation step is additionally used to test the pruned network. The whole dataset is divided into training set and cross validation set and at each phase of pruning the cross validation set is used to validate the pruned network. If the pruned network outperforms the unpruned one, then the pruned network is accepted and the pruning process can be continued. Otherwise the network is restored to the size before the current pruning step. Huynh and Setiono [16] introduced this cross validation method. Sabo and Yu have proposed a new pruning algorithm which combines the advantages of sensitivity analysis method, variance sensitivity analysis method and cross fold validation method. They claim that the use of an additional cross validation set at each phase of the pruning helps to improve the network generalization capacity [8].

The magnitude based pruning (MBP) methods assume that small weights are irrelevant [31, 32]. Hagiwara [31] suggests three simple and effective strategies called Goodness factor, Consuming energy and Weights power for detecting both redundant hidden neurons and weights. These measures are much less sophisticated and require significantly less computational time. However methods based only on weight magnitude often remove important parts of the network also [33].

In the Mutual Information (MI) based methods [34, 35], singular value decomposition is used to analyze the hidden unit activation covariance matrix and the rank of the covariance matrix determines the optimal number of hidden units. Xing-Hu [34] suggests a two phase construction approach for pruning both input and hidden units of MLPs based on mutual information. First all salient input units are determined according to the order of ranking result and by considering their contributions to the network's performance. Then the irrelevant input units are eliminated. Second the redundant hidden units are removed from trained networks, one after another according to a relevance measure. From the view of the information entropy and bionics principle, Zhang and Qiao [35] have proposed a node pruning algorithm based on the neural complexity (PBNC) for feed-forward neural networks. In the process of training, the neural complexity has been acquired by standard covariance matrix of the neural network's connection matrix. The algorithm does not need to train the cost function of the neural network to a local minimum, therefore, it can prune the architecture of the neural network on-line, and the pre-processing neural network weights is avoided before neural network architecture adjustment.

Evolutionary pruning methods use Genetic Algorithms (GA) to prune neural networks. Whitley and Bogart [36] have proposed a method to prune the neural networks using GA terminology. Different pruned networks are created by application of mutation, reproduction and cross-over operators. These pruned networks, being awarded for using fewer parameters and for improving generalization. Benardos Vosniakos [37] suggests a method to optimize the feedforward neural network architecture. It considers the problem as one of multi-objective optimization. The solution space consists of all the different combinations of hidden layers and hidden neurons. Given the complex nature of the problem, a GA

is employed to search the solution space for the best architectures. The basic idea behind the GAs is that the optimal solution will be found in areas of the solution space that contain good solutions and that these areas can be identified through robust sampling.

Many algorithms have been proposed based on sensitivity analysis to optimize the neural network [5, 6, 38, 39]. The sensitivity based approach attempts to find the contribution of each weight or node in the network and then prunes the weight or node that have the least effect on the objective function. The most popular sensitivity based pruning algorithms are OBD [40] and OBS [41]. Lecun et al. [40] have proposed the Optimal Brain Damage (OBD) method that approximates the measure of 'saliency' of a weight by estimating the second derivative of the network output error with respect to that weight. In this method pruning is carried out iteratively on a well trained network to a reasonable level, compute 'saliencies', delete low 'saliency' weights and resume training. This OBD method assumes that the error function is quadratic and that the Hessian is diagonal. By following the same idea used in OBD method Hassibi et al. [41] have proposed the Optimal Brain Surgeon (OBS) method. This method removes the diagonal assumption idea used in OBD, because if the diagonal assumption is inaccurate, it can lead to the removal of wrong weights. However it is impractical for large networks. An early stopping procedure monitors the error on a validation set and halts learning when this error starts to increase. There is no guarantee that the learning curve passes through the optimal point and the final weights are sensitive to the learning dynamics. Today, many pruning algorithms are based on the theory of OBD and OBS and in many ways beyond them [28, 42].

Castellano et al. [17] have proposed an Iterative Pruning (IP) algorithm to find the most appropriate network size. This method solves a linear system by least squares identification algorithm. Indeed for large network size, the output matrix may have deficient rank and for the problems with high norms, infinite solutions may exist. In order to enhance the efficiency of IP, Fangju Ai [2] has proposed the Improved Iterative Pruning algorithm (IIP) based on the simple idea of iteratively removing the nodes of the hidden layers in Feedforward Neural Networks and then adjusting the remaining weights using an Conjugate Gradient Precondition Normal Equation(CGPCNE) algorithm with a view to maintaining the original input-output behavior. So the pruned networks need not retrain. Ponnapelli et al. [43] have suggested that the sensitivities of weights should only be compared with those related with the same node in the same layer. Thus the term Local Relative Sensitivity Index (LRSI) is defined as the ratio of the sensitivity of a particular weight and the sum of all the sensitivities of the weights that are connected to the same node from the previous layer. But this algorithm only considers weight removal; node pruning is not included. Engelbrecht [15] has proposed a modified approach to sensitivity analysis. Instead of using the value of sensitivity directly, Engelbrecht has found the average sensitivity of a network parameter over all the patterns and then he has developed the new measure called variance nullity. This Variance Nullity Pruning (VNP) algorithm allows for pruning of both nodes and weights. Lauret et al., [39] have proposed a new technique to obtain the optimal number of hidden units of a single layer fully connected network. This technique relies on a global Sensitivity Analysis Model Output (SAMO). Fnaiech et al., [29] have proposed the modified version of Variance Nullity Pruning algorithm. In this version, contrarily, to the work of Englebrecht where the pruning is performed on the entire net, the modified algorithm prunes layer by layer with the use of a pruning decision based on a local parameter variance nullity coefficient (LPW). These coefficients are then classified in an ordered list which allows the decision making of coefficients and neurons removal in order to get the best neural network pruned. A global SA method, the Extended Fourier Amplitude Sensitivity Test (EFAST) method, is used to quantify the relevance of the hidden units. Each hidden unit is assigned a ratio that gives their ranking. This quantitative information therefore leads to a suggestion of the most favorable units to eliminate.

Zeng and Yeung [38] have proposed a method to prune the hidden neurons of multilayer perceptron network using a quantified sensitivity measure. It defines the sensitivity of an individual input neuron as the expectation of its output deviation due to expected input deviation with respect to over all inputs from a continuous interval and it estimates the relevance of a neuron by finding the summation of the absolute values of its outgoing weights. Then the method prunes the hidden neurons with the lowest relevance value iteratively. Xua and Hob [6] describes a UB-OBS pruning method, which prunes hidden units in Feedforward NNs. First, it identifies the dependent hidden nodes using QR factorization, and then prunes them and recalculates the output weights of the remaining nodes to maintain the input output behavior of the network.

The Significance based pruning methods calculate the new significant measure based on both the inputs of the network and the outputs of the hidden neurons and consider all the nodes with significance value below the threshold

as insignificant and eliminate them. Belue and Bauer have proposed a method that injects a noisy input parameter into the neural network model and then use statistical tests to decide if the significances of the original neural network parameters are higher than that of the injected noisy parameter [44]. Parameters with lower significances than the noisy parameter are pruned. Augasta and Kathirvalavakumar [45] have proposed a novel pruning algorithm namely N2PS which finds the optimal architecture of multilayered Feedforward neural network by removing both insignificant input nodes and hidden nodes based on a new significant measure that is calculated by the Sigmoidal activation value of the node and all the weights of its outgoing connections.

In this literature many pruning algorithms based on different pruning techniques have been discussed and each algorithm has its own advantages and limitations. Some pruning algorithms [15, 34, 45] prune both irrelevant input neurons and hidden neurons of the network and some algorithms [17, 38, 39] prune irrelevant hidden neurons only. Real-world applications prefer simpler and more efficient methods. But a significant drawback of most standard methods consists in their low efficiency. For example the main weakness of the OBD and OBS techniques are their relatively low computational efficiency. Magnitude based pruning (MBP) methods often remove important parts of the network as they assume that small weights are irrelevant. However small weights may be important compared to very large weights which cause saturation in hidden and output units [15]. Some algorithms [32, 43] require the user to specify the number of problem dependent threshold parameters or tuning parameters. More sophisticated methods [15, 34, 38] reach better results, but the precision is usually compensated by unproportional increase in computation time [1]. Unfortunately, sensitivity analysis based pruning methods are not guaranteed to detect all redundant processing elements as they assume both the inputs of the network and the outputs of the hidden neurons to be mutually independent [1]. When there are dependencies between inputs, the Sensitivity Analysis based method can be ineffective while the Mutual Information based methods and significance based method can successfully avoid this limitation [34].

4. Performance analysis

In this section, the results of some existing pruning algorithms such as N2PS [45], VNP [15], Xing-Hu's method [34], MBP [32], OBD [40] and OBS [41] are compared with each other on four well known real datasets [1] and their performances are evaluated.

4.1. Datasets

The datasets used to test the algorithm are,

1. Iris Plants dataset (iris): Irises are classified into three categories: setosa, versicolor and verginia. Each category has 50 patterns and each pattern possesses four attributes namely sepal length, sepal width, petal length and petal width.

2. Wisconsin-breast-cancer dataset (cancer): This dataset was designed to diagonize breast tumors as either benign or malignant. It contains 699 patterns and each pattern consists of 9 real value attributes as an input vector and two classes as an output vector. Out of 699 patterns 458 are benign patterns and 241 are malignant patterns.

3. Hepatitis Domain dataset (hepatitis) : This dataset contains 155 patterns and each pattern is described using 19 attributes. Hepatitis data are classified into two categories: DIE and LIVE. There are 123 patterns of class DIE and 32 patterns of class LIVE.

4. Pima Indians Diabetes dataset (diabetes): The problem posed here is to predict whether a patient would test positive or negative for diabetes according to the criteria given by World Health Organization (WHO). This is a two class problem with class value 1 and 2 interpreted as negative and positive results for diabetes. There are 500 patterns of class 1 and 268 of class 2. There are 8 attributes for each pattern.

[1] *Available: http://weka.wikispaces.com/Datasets*

Table 1. Properties of 4 real datasets.

	Datasets			
Properties	iris	cancer	hepatitis	diabetes
No. of classes	3	2	2	2
No. of examples	150	699	155	768
No. of training examples	75	350	81	384
No. of testing examples	75	349	74	384
No. of attributes	4	9	19	8

The detailed description of the datasets is shown in Table 1. The training and testing patterns are taken randomly from each class. For example the iris dataset is having 3 classes with 50 patterns for each class. From each class 25 patterns are taken randomly for training and another 25 patterns are taken randomly for testing the network. The N2PS algorithm uses momentum (μ) as 0.5 and the learning rate (η) as 0.1 for all datasets. The network is trained until the error converges to predetermined mean squared error(mse) 0.01 or the prespecified maximum number of iterations 200 has expired, whichever is earlier. The OBD, OBS, MBP, Engelbrecht's and Xing-Hu's method use momentum (μ) as 0.9, the learning rate (η) as 0.0001 and mse as 0.01 for all datasets. For simplicity, all the algorithms are evaluated using single hidden layer FNN. Similar to Engelbrecht's approach [15], the traditional weight-oriented pruning methods (OBS, OBD, and MBP) are compared with the node pruning algorithms (VNP, Xing-Hu and N2PS) because for the weight-oriented pruning methods, a hidden neuron is deleted if all incoming or all outgoing links to that neuron are removed [39]. The performance of the pruning algorithms in this study are evaluated based on the parameters such as required number of iterations for training, classification accuracy, generalization ability and number of pruned nodes. Classification accuracy (ACC_{tst}) is identified by computing the percentage of data examples that are correctly classified by the pruned network on testing dataset. Generalization is the ability to produce accurate results for the inputs that are not included in the training dataset. The number of pruned nodes is identified by finding the difference between the total number of nodes in the original network and the pruned network.

4.2. Comparison of pruning methods

The pruning methods OBS and OBD require additional computation for calculating the Hessian matrix of the system. The efficiency of the MBP method is also low, since it considers only the magnitude of weights to prune the network [15]. The pruning methods OBD, OBS and MBP prune irrelevant hidden neurons only. The sensitivity analysis based method VNP combines both the input units pruning and hidden units pruning of Multi Layer Perceptrons (MLPs) in a single formula and achieves satisfying results, but VNP is not guaranteed to detect redundant neurons as it doesn't consider the mutual dependency between both the inputs of the network and outputs of the hidden neurons. The Xing-Hu's method overcomes this limitation by considering the mutual dependency between them but it performs pruning in two separate phases [34]. The N2PS method combines the advantages of both VNP and Xing-Hu. It performs both the input units pruning and hidden units pruning of MLPs in a single formula as VNP and considers the mutual dependency between the inputs of the network and outputs of the hidden neurons like Xing-Hu's method. Xing-Hu achieves better results than VNP with two separate phases for pruning input units and hidden units respectively while N2PS achieves better results than Xing-Hu in just a single phase for pruning both units. Also N2PS doesn't require any complex computation to find the significant measure of each node [45].

The classification accuracy on testing data (ACC_{tst}) of the most popular pruning algorithms in our study has been compared to evaluate their performance. Experiments were performed 10 times for each dataset by dividing the original dataset into training and testing using a different random seed every time. The average of the results of the 10 runs was calculated for each set. Table 2 shows the comparison results of six pruning methods on four datasets namely iris, cancer, hepatitis and diabetes, a pruned network is only accepted if the deterioration in classification accuracy is less than 5%. For all the classification problems, the N2PS method resulted in better architecture with minimum number of nodes while having the accuracy similar to or better than that of other architectures obtained from other pruning methods. Regarding the classification accuracy, the N2PS and Xing-Hu's method achieve higher accuracy for the maximum of datasets than other methods. Fig. 1 shows the comparison of the classification accuracies achieved by

Table 2. Result Comparision of Six Pruning Methods

Datasets	Unpruned		N2PS		VNP		Xing-Hu		OBD		OBS		MBP	
	NN	ACC_{tst}	NN	ACC_{tst}	NN	ACC_{tst}	NN	ACC_{tst}	NN	ACC_{tst}	NN	ACC_{tst}	NN	ACC_{tst}
iris	5-10-3	96%	3-3-3	98.67%	2-2-3	97.7%	3-2-3	98.67%	4-4-3	98%	4-4-3	98%	4-4-3	98%
cancer	10-10-2	95.4%	3-2-2	97.1%	3-1-2	97.8%	3-3-2	96.78%	9-8-1	92.5%	9-7-1	90%	9-7-1	94.2%
hepatitis	20-25-2	80.2%	2-3-2	86.4%	4-4-2	83.3%	3-8-2	84.62%	19-9-1	78.7%	19-16-1	73.8%	19-18-1	80.3%
diabetes	9-40-2	68.6%	5-3-2	70.3%	6-8-2	69.1%	6-8-2	74.22%	8-16-1	68.6%	8-26-1	65.4%	8-26-1	68.9%

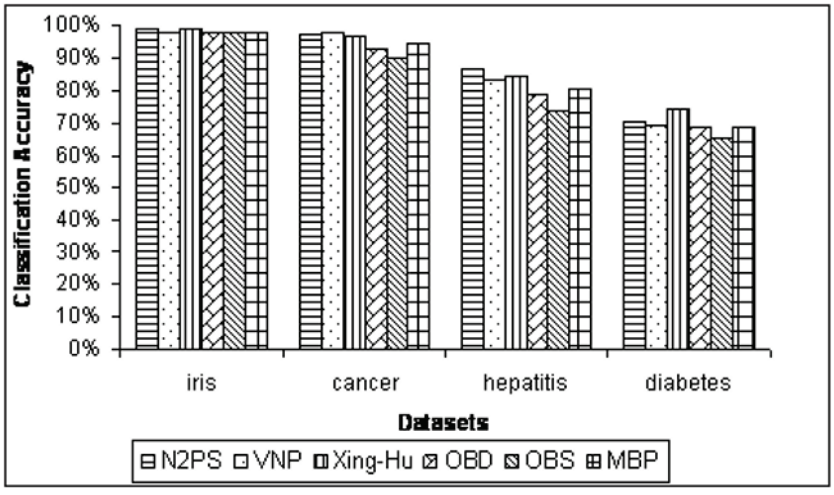

Figure 1. Comparing classification accuracies of Six pruning algorithms.

six pruning methods in study. It shows that the N2PS method achieves higher accuracy for all datasets than OBS, OBD and MBP and achieves maximum or equal accuracy for 3 datasets out of 4 than Xing-Hu and VNP.

Considering the removal of neurons, Fig. 2 compares the six pruning methods by the removal of hidden neurons. It shows that for iris dataset, the number of hidden neurons pruned by N2PS, VNP, Xing-Hu, OBD, OBS and MBP respectively are 7, 8, 8, 6, 6, 6 and for cancer dataset, the number of hidden neurons pruned by N2PS, VNP, Xing-Hu, OBD, OBS and MBP respectively are 8, 9, 7, 2, 3, 3 and so on. The number of pruned hidden neurons are idenified by calculating the difference between the number of hidden neurons in the original network and the number of hidden neurons in the pruned network. While comparing the removal of hidden neurons of each pruning method, VNP and N2PS methods remove equal or more hidden neurons for all datasets than other pruning methods in our comparison. Moreover N2PS achieves better classification accuracy than VNP as it considers the mutual dependency between both the inputs of the network and outputs of the hidden neurons.

In practical real world applications, a better generalization performance is more important than optimal behavior over the training data [8]. The generalization of the reduced networks with respect to the original ones was evaluated based on the classification accuracy computed over the testing data. Fig. 3 shows the generalization ability of all pruning algorithms in this study. It has been observed that the N2PS, VNP and Xing-Hu achieve networks with better generalization as the pruned network gives better classification accuracy than the unpruned network for the testing dataset. Here the pruning algorithms N2PS, VNP and Xing-Hu achieves the better generalization for all datasets while the pruning algorithms MBP, OBS and OBD fail to achieve the generalization for three datasets namely cancer, hepatitis and diabetes.

The time complexity for the pruning is determined by the selection criteria which determines the node or connection for elimination. Let p be the number of training examples and n be the number of connections in the network topology. For MBP, the time complexity of selection is $O(n)$. If one learning step has time complexity $O(pn)$ and e

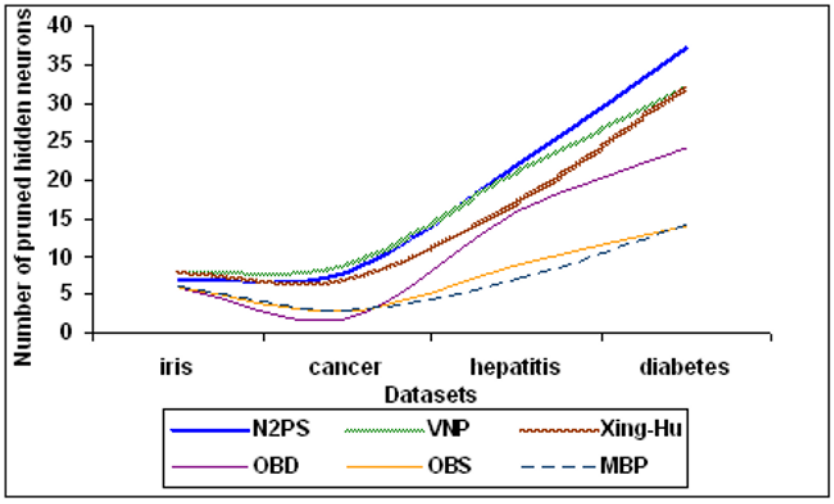

Figure 2. Comparing hidden nodes removal of pruning algorithms.

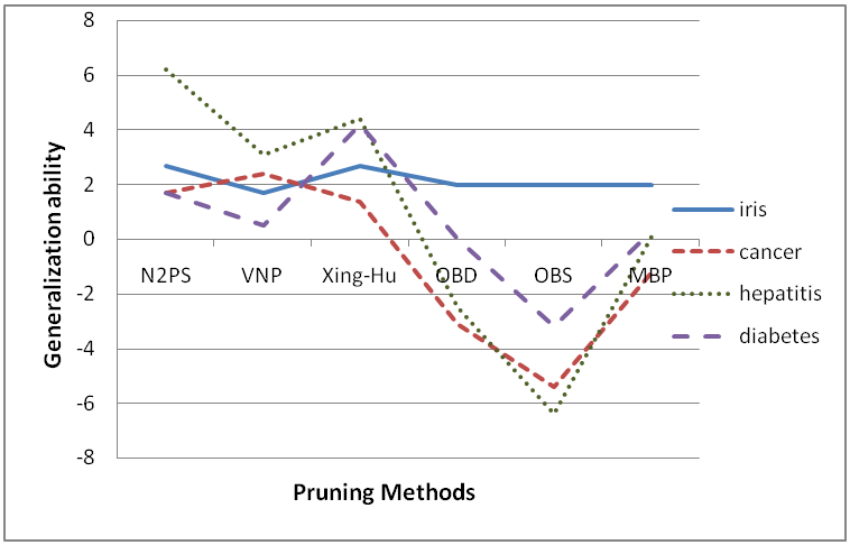

Figure 3. Comparing the generalization ability of pruning algorithms.

be the number of training steps then the time complexity for pruning by MBP is $O(epn)$??. For OBD and OBS the time complexity of selection is $O(pn^2)$. Let $O(n)$ be the time complexity for eliminated connections, then the total time complexity of MBP is $O(epn^2)$ and the total time complexity of OBD and OBS is $O(pn^3)$. While comparing the time complexity of traditionally standard pruning algorithms of this study, MBP has the low time complexity.

Considering the pruning speed of input and hidden nodes pruning algorithms, N2Ps requires maximum 200 iterations for training the initial network while VNP and Xing-Hu's algorithms takes maximum of 10000, 5000, 1000 and 1000 iterations for training the initial networks of iris, hepatities, diabetes and cancer respectively. When a network is pruned, VNP starts retraining the reduced model on new initial random weights which may lead to the increase in number of iterations in each pruning step and decrease in classification accuracy [34]. The pruned network re-trained with the weights inherited from the network obtained from the previous pruning step outperforms the network re-trained directly with new initial random weights on every pruning step [34]. Xing-Hu and N2PS inherit the initial weights from previous step for the retraining process of the pruned network [34, 45]. Unfortunately Xing-Hu requires more number of

pruning and retraining steps for selecting the relevant input units in phase I and for removing the irrelevant hidden units in phase II. But the N2PS requires maximum 3 pruning steps only. The maximum number of pruning steps required by N2PS for four data sets iris, cancer, hepatitis and diabetes respectively are 2, 2, 3 and 2 only while VNP requires 3, 7, 3 and 7. While comparing the maximum number of retraining iterations required by Xing-Hu and N2PS for the pruned network on four data sets iris, cancer, hepatitis and diabetes respectively, Xing-Hu requires 1000, 100, 100 and 100 but N2PS requires only 27, 50, 50 and 50 iterations [34, 46]. The reduction in number of pruning steps and number of retraining iterations of N2PS resulted in a better generalization than the pruned networks of the other pruning algorithms.

In summary, the comparative study consistently indicate that the performance of the significance based algorithms are better than other methods in terms of generalization, input and hidden neurons removal, pruning speed, and classification accuracy.

5. Conclusion

In this paper a survey of pruning algorithms for feedforward neural network have been specified. In addition some of the existing pruning methods based on sensitivity, mutual information and significance are compared for several real datasets with regard to their performance in minimizing network size. The standard techniques such as OBD, OBS and MBP depend heavily on the magnitude of weights. This approach leads to prune minor important weights but may eliminate weights that are actually crucial to the over all architecture of the networks. Thus creates additional local minima on the error surface during training and thereby may increase the minimal network size. Results show that the architecture obtained by those traditional techniques are larger than the resultant architectures of the current pruning algorithms and also the traditional algorithms can prune irrelevant hidden neurons only. The current pruning algorithms in our study namely VNP, Xing-Hu and N2PS combines both the input units pruning and hidden units pruning of feedforward neural network. Section 4 discusses on pruning speed and performance of those algorithms. The size and accuracy of the network pruned by them are comparable with one another. Though VNP and Xing-Hu reach better results, but the precision is usually compensated by unproportional increase in computation time. Both requires more number of iterations and pruning steps to find the optimal network.

The experimental study in section 4 shows that the pruning algorithms based on Mutual Information and Significance can be more efficient in pruning than the methods based on sensitive and magnitude, as they consider the mutual dependency between the inputs of the network and outputs of the hidden neurons. While comparing MI based and Significance based methods, the computation cost and pruning speed of Significance based methods are lower than the MI based methods as they require no sorting for ranking the relevant features and prune each node in a single phase by simply calculating the significance of the node using the mutual information. The performance analysis of six pruning algorithms on four experimental datasets show that the significance based pruning algorithm N2PS achieves better classification accuracy with a better topology in smaller number of iterations than other algorithms.

In a nutshell, the comparative study on pruning algorithms based on sensitivity, MI and significance indicate that the algorithms which are using the significance as a meta heuristic for pruning as in N2PS are the currently available best optimization methods of feedforward neural network for classifying large datasets.

References

[1] P. M. Atkinson, A. R. L. Tatnall, Neural networks in remote sensing, Int. J. Remote Sens. 18 (4), 699, 1997

[2] A. Fangju, A New Pruning Algorithm for Feedforward Neural Networks, Fourth International Workshop on Advanced Computational Intelligence, IEEE Conference Publication, Wuhan, Hubei, China 19-21 October 2011, 286–289

[3] A. Yoan, A. Sorjamaa, P. Bas, O. Simula, C. Jutten, A. Lendasse, 3. OP-ELM: optimally pruned extreme learning machine, IEEE Trans. Neural Networks 21 (1), 158–162, 2010

[4] S. Ahmmed, K. Abdullah-Al-Mamun, M. Islam, A novel algorithm for designing three layered artificial neural networks, Int. J. Soft. Comput. 2 (3), 450–458, 2007
[5] O. Aran, O. T. Yildiz, E. Alpaydin, An incremental framework based on cross validation for estimating the architecture of a multilayer perceptron, Int. J. Pttern. Recogn. Artif. Intell. 23 (2), 159–190, 2009
[6] J. Xua, D. W. C. Hob, A new training and pruning algorithm based on node dependence and Jacobian rank deficiency, Neurocomputing 70, 544–558, 2006
[7] B. Choi, J. HongLee, D.-H. Kim, Solving local minima problem with large number of hidden nodes on two layered feedforward artificial neural networks, Neurocomputing 71, 3640–3643, 2008
[8] D. Sabo, X.-H. Yu, A new pruning algorithm for neural network dimension analysis, IJCNN 2008, IEEE World Congress on Computational Intelligence, In Proc. of IEEE Int. Joint Conference on Neural Networks, Hong Kong, 1-8 June 2008, 3313–3318
[9] R. Reed, Pruning algorithms a survey, IEEE T. Neural Networ. 4 (5), 740–747, 1993
[10] R. Setiono, H. Liu, Understanding Neural Networks via Rule Extraction, In: Proc. of 14th International Joint Conference on Artificial Intelligence, Montreal, Canada, August 20-25 1995, 480–485
[11] M. D. Emmerson, R. I. Damper, Determining and improving the fault tolerance of multi layer perceptrons in a pattern-recognition application, IEEE T. Neural Networ. 4, 788–793, 1993
[12] J. M. Zurada, Introduction to Artificial Neural Systems (Jaisco Publishing House, Mumbai, 2002)
[13] R. Setiono, B. Baesens, C. Mues, A note on knowledge discovery using neural networks and its application to credit card screening, Eur. J. Oper. Res. 192 (1), 326–332, 2008
[14] M. G. Augasta, T. Kathirvalavakumar, Reverse Engineering the Neural Networks for Rule Extraction in Classification Problems, Neural Process. Lett. 35, 131–150, 2012
[15] A. P. Engelbrecht, A new pruning heuristic based on variance analysis of sensitivity information, IEEE T. Neural Networ. 12 (6), 1386–1399, 2001
[16] T. Q. Huynh, R. Setiono, Effective neural network pruning using cross validation, In: Proc. of IEEE Int. Joint Conference on Neural Networks 2, Montreal, 31 July - 4 August 2005, 972–977
[17] G. Castellano, A. M. Fanelli, M. Pelillo, An Iterative Pruning algoritm for feedforward neural networks, IEEE T Neural Networ. 8 (3), 519–530, 1997
[18] S. Marsland, S. U. Nehmzow, J. Shapiro, A self organizing network that grows when required, Neural Networ. 15 (809), 1041–1058, 2002
[19] R. Zhang, Y. Lan, G. B. Huang, Z. B. Xu, Universal approximation of extreme learning machine with adaptive growth of hidden nodes, IEEE T. Neural Networ. Learn. Syst. 23 (2), 365–371, 2012
[20] G. B. Huang, L. Chen, 20. Enhanced random search based incremental extreme learning machine, Neuro Comput. 71 (16-18), 3460–3468, 2008
[21] A. B. Nielsen, L. K. Hansen, Structure learning by pruning in independent component analysis, Neuro Comput. 71 (10-12), 2281–2290, 2008
[22] D. Sabo, X.-H. Yu, Neural network dimension selection for dynamical system identification, In: Proc. of 17th IEEE International Conference on Control Applications, San Antonio, TX, 3-5 September 2008, 972, 977
[23] S. C. Huang, Y. F. Huang, Bounds on the number of hidden neurons in multilayer perceptrons, IEEE T. Neural Networ. 2, 47–55, 1991
[24] H.-G. Han, J.-F. Qiao, A structure optimisation algorithm for feedforward neuralnetwork construction, Neurocomputing 99, 347–357, 2013
[25] P. L. Narasimhaa, W. H. Delashmitb, M. T. Manrya, J. Lic, F. Maldonado, An integrated growing-pruning method for feedforward network training, Neurocomputing 71, 2831–2847, 2008
[26] A. B. Nielsen, L. K. Hansen, Structure learning by pruning in independent component analysis, Neurocomputing, 71 (10-12), 2281–2290, 2008
[27] M. Attik, L. Bougrain, F. Alexandra, Neural Network topology optimization, In: Proceedings of ICANN'05, Lecture Notes in Computer Science, Vol. 3697, 5th International Conference, Warsaw, Poland, 11-15 September, 2005 (Springer, Berlin, Heidelberg, 2005) 53–58
[28] Q. Jun-fei, Z. Ying, H. Hong-gui, Fast unit pruning algorithm for feed-forward neural network design, App. Math. Comput. 205 (2), 662–667, 2008
[29] N. Fnaiech, S. Abid, F. Fnaiech, M. Cheriet, A modified version of a formal pruning algorithm based on local relative variance analysis, First International IEEE Symposium on Control, Communications and Signal Processing,

Hammamet, Tunisia, 21-24 March, 2004, 849, 852
[30] R. Setiono, A penalty function approach for pruning feedforward neural networks, Neural Comput. 9 (1), 185–204, 1997
[31] W. Wan, S. Mabu, K. Shimada, K. Hirasawa, Enhancing the generalization ability of neural networks through controlling the hidden layers, J. Hu, App. Soft Comput. 9, 404–414, 2009
[32] M. Hagiwara, A simple and effective method for removal of hidden units and weights, Neurocomputing, 6, 207–218, 1994
[33] J. Sietsma, Dow RJF, Neural net pruning: why and how, In: Proc. of the IEEE International Conference on Neural Networks, Vol. 1, San Diego, CA, USA, 24-27 July 1988, 325–333
[34] H.-J. Xing, B.-G. Hu, Two phase construction of multilayer perceptrons using Information Theory, IEEE T. Neural Networ. 20 (4), 715–721, 2009
[35] Z. Zhang, J. Qiao, A Node Pruning Algorithm for Feedforward Neural Network Based on Neural Complexity, In: Int. Conf. on Intelligent Control and Information Processing, Dalian, 13-15 August 2010, 406–410
[36] D. Whitley, C. Bogart, The evolution of connectivity: Pruning neural networks using genetic algorithms, In: Int. Joint Conf. Neural Networks, 1 (IEE Press, Washington DC, 1990) 134–137
[37] P. G. Benardos, G.-C. Vosniakos, Optimizing feedforward artificial neural network architecture, Eng. App. Artif. Intelligence, 20, 365–382, 2007
[38] X. Zeng, D. S.Yeung, Hidden neuron pruning of multilayer perceptrons using a quantified sensitivity measure, Neuro Comput. 69, 825–837, 2006
[39] P. Lauret, E. Fock, T. A. Mara, A Node Pruning Algorithm Based on a Fourier Amplitude Sensitivity Test Method, IEEE T. Neural Networ. 17 (2), 273–293, 2006
[40] Y. Le Cun, J. S. Denker, S. A. Solla, In. D. S. Touretzky (Ed.), Optimal brain damage, Advances in neural information processing systems (Morgan Kaufmann, San Mateo, 1990) 2, 598–605
[41] B. Hassibi, D. G. Stork, G. J. Wolf, Optimal brain surgeon and general network pruning, In: Proc. of IEEE ICNN'93, 1, WDS'08 Proceedings of Contributed Papers, Part I, 2008, 293–299
[42] W. U. Jian-yu, H. E. Xiao-rong, DOBD Algorithm for Training Neural Network, Part I. Method, Chinese J. Process Eng. 2 (2), 172–176, 2002
[43] P. V. S. Ponnapallii, K. C. Ho, M. Thomson, A formal selection and pruning algorithm for feedforward artificial neural network optimization, IEEE T. Neural Networ., 10 (4), 964–968, 1999
[44] L. M. Belue, K. W. Bauer, Determining input features for multilayer perceptrons, Neurocomputing 7, 111–121, 1995
[45] G. Augasta, T. Kathirvalavakumar, A Novel Pruning Algorithm for Optimizing Feedforward Neural Network of Classification Problems, Neural Process. Lett. 34 (3), 241–258, 2011
[46] T. Ragg, H. Braun, H. Landsberg, A comparative study of neural network optimization Techniques, In 13th International Conf. on Machine Learning, Norwich, UK, 2-4 April, 1997, Artificial Nets and Genetic Algorithms (Springer, 1997) 341–345