Neural Network Pruning and Pruning Parameters

G. Thimm and E. Fiesler
Institut Dalle Molle d'Intelligence Artificielle Perceptive (IDIAP)
C.P. 192, Rue de Simplon 4, CH-1920 Martigny, Switzerland.
Email: Thimm@idiap.ch and EFiesler@idiap.ch
WWW: http://www.idiap.ch/neural/network.html

ABSTRACT

The default multilayer neural network topology is a fully interlayer connected one. This simplistic choice facilitates the design but it limits the performance of the resulting neural networks. The best-known methods for obtaining partially connected neural networks are the so called *pruning methods* which are used for optimizing both the size and the generalization capabilities of neural networks. Two of the most promising pruning techniques have therefore been selected for a comparative study.

It is shown that these novel techniques are hampered by having numerous user-tunable parameters, which can easily nullify the benefits of these advanced methods.

Finally, based on the results, conclusions about the execution of experiments and suggestions for conducting future research on neural network pruning are drawn.

Keyworks: neural network, pruning, parameters, neural network optimization, network size, generalization

INTRODUCTION

Various neural network pruning techniques have been developed in the aim to find neural networks of suitable sizes for a given to certain problem. The multitude of techniques and the ongoing research can be explained by the difficulties to find a minimal neural network for a specific application [1].

Looking at the history of neural network pruning methods, one remarks that the first methods have been mainly concerned with which connection or neuron to remove from a network. The most famous among these methods are: smallest weight removal, the smallest variance [6], the optimal brain damage method [4]. Overviews on these and several other methods can be found in [5] and [10].

More recent approaches are also aimed at to estimating the best moment for removing units and how many units to prune, as well as to optimize the performance of the resulting network, as well as its size and training time. These optimizations are often outweighed by the introduction of additional parameters, for which the respective authors do or do not specify values, whose optimal values are usually specific for a certain data set. Hence, a setting found to be good for one data set may be unsuitable for another. Furthermore, these training parameters are subject to changes if they are applied to new neural network architectures. A logical way to adapt such methods to a new network architecture is to perform experiments with the original neural network architecture in the aim to find suitable parameter settings and to gain experience with these methods in order to facilitate the adaption to the new neural network model. This was done for to high order perceptrons (compare [8] for information on high order perceptrons). Hence, the focus of this report is on the experiments performed with multilayer perceptrons for which the methods discussed later have been developed.

THE METHODS

This research is concentrated on two very promising methods, where one is a further development of the other. The first method was proposed by W. Finnoff *et al.* Their method uses a test statistic to estimate whether a certain weight will become zero during a future training step. This statistic is based on the weight changes during the backpropagation (compare [2]).

In the original work of W. Finnoff *et al.*, the pruning follows a fixed schedule: in the first pruning step 35% of the connections and in the following steps 10% is removed. In order to compare objectively the criteria for when to prune, the pruning schedule of L. Prechelt is integrated into the algorithm of W. Finnoff *et al.*. In this schedule all connections for which the test statistic evaluates to a value below the mean of all connections multiplied by a function $\lambda(GL)$ are removed during a pruning step, where $\lambda(Gl)$ is defined as:

$$\lambda(Gl) = \lambda_{max}(1 - \frac{1}{1 + \frac{Gl}{\alpha}}).$$

For this function, L. Prechelt determined experimentally suitable values $\lambda_{max} = 2/3$ and $\alpha = 2$. The *generalization loss* GL is defined as $100 \cdot (\frac{E_{va}}{E_{opt}} - 1)$, with E_{va} the current error and E_{opt} the smallest error the observed during the training for a test data set distinct from the training set. The function $\lambda(Gl)$ is close to zero if the generalization loss is small and approaches $2/3$ if it increases. Simplified, this means that the number of removed connections increases if the generalization of the network gets worse. Note, that beside the constants used this function, also a strip length for the averaging of GL is introduced. Further, the training data has to be split into two distinct sets.

The difference between the two methods is the decision of when to prune: W. Finnoff *et al.* remove connections when the error for the training set is inferior to a fixed threshold. L. Prechelt bases this decision on the error for a separate data set: basically, his method removes connections, when the generalization loss for the current training step exceeds a certain limit.

EXPERIMENT OVERVIEW

The experiments were performed using 5 real-world data sets, on which further documentation can be found in [9] or [7]. Only the outcome of the experiments is therefore documented here.

The experiments follow a fixed scheme: in a first phase, for each data set a few simulations have been performed in the aim to find suitable settings for the numerous parameters. Then, using these optimized parameter settings a certain amount of simulations was performed and the confidence interval for the average generalization performance and network size calculated. These confidence intervals then were used for the comparison for the pruning methods. This resulted in the following outcome:

Solar data: for this data set, L. Prechelt's method performed better than the method W. Finnoff *et al*. The networks found have on average a better generalization performance and are smaller in size.

Auto-mpg data: W. Finnoff's method found networks with a better generalization but similar network size as compared to the method of L. Prechelt.

Servo data: the method W. Finnoff *et al.* produced bigger networks with better generalization.

Wine data: both algorithms failed to reduce the size of the network considerably, although it is possible: the method of E. D. Karnin reduces the same network to approximately 20% of its original size [3].

Digits data: both pruning methods hardly improve the generalization of a fully connected multilayer perceptron. Both methods remove only few connections (with a little advantage for W. Finnoff's method). However, as the amount of connections is considerably below the results obtained with the pruning method of E. D. Karnin, this is a poor result.

The disappointing result for the wine and the digits data is probably due to an non-successful search for the optimal parameter settings. However, this search was considerably longer than for the other data sets, and in a real application such an extensive is not feasible. Further, a comparison with other pruning methods, as for example the method of E.D. Karnin, which permits the rating of the network is usually not possible. The user is therefore not able to rate the performance and size of a network, which is necessary in order to decide whether the search for optimal training parameter settings has to be extended.

Discussion of the Experiments

From the outcome of the experiments it can be concluded that the two methods perform on average equally good. Furthermore, both methods have weaknesses for certain data sets.

The reader familiar with the publication of Prechelt might have noticed that his results are somewhat different: he showed that his method performs better than the method of W. Finnoff *et al*. At this point, the authors would like to state that he showed conclusively that for certain parameter settings and several data sets his method performs better.

The problem is that there are many different parameters to adapt and the unexperienced user is not able to find optimal values for them. This phenomena does not only apply to the comparison of the methods of L. Prechelt and W. Finnoff *et al.*, but to many others in neural network science and beyond.

Conclusion

Several conclusions can be drawn: first, in an application oriented environment, as for example in industry, the efficiency of the methods of W. Finnoff *et al.* and L. Prechelt in terms of number of removed connections is comparable due to the difficulty in finding good parameter settings.

Second, experiments should ideally be repeated with different data sets by someone, who is not familiar with the method in question. If this is not done, the results can be easily biased, as the developer of a certain method has more insight of the influence of a certain parameter than others.

Third, research in neural networks should not only aim at optimizing training procedures, but in eliminating user defined parameters, even at the cost of finding often suboptimal neural networks. In most cases the optimal performance will not be reached in most applications as in most cases a user does not have the experience to find the optimal settings of the parameters or does not have the time to search for them, or both.

References

[1] E. Fiesler. Minimal and high order neural network topologies. In *Proceedings of the Fifth Workshop on Neural Networks, number 2204 in SPIE* Proceedings, pages 173–178, San Diego, California, 1993. Simulation Councils, Inc. / The Society for Computer Simulation.

[2] William Finnoff, Ferdinand Hergert, and Hans Georg Zimmermann. Improving model selection by nonconvergent methods. *Neural Networks*, 6:771–783, 1993.

[3] E. D. Karnin. A simple procedure for pruning back-propagation trained neural networks. *IEEE Transactions on Neural Networks*, 1(2):239–242, June 1990.

[4] Yann Le Cun, John S. Denker, and Sara A. Solla. Optimal brain damage. In David S. Touretzky, editor, *Advances in Neural Information Processing Systems (NIPS)*, volume 2, pages 598–605, San Mateo, California, 1990. IEEE, Morgan Kaufmann.

[5] Russell Reed. Pruning algorithms — a survey. *IEEE Transactions on Neural Networks*, 4(5):740–747, September 1993.

[6] J. Sietsma and R. J. F. Dow. Creating artificial neural networks that generalize. *Neural Networks*, 4(1):67–69, 1991.

[7] Georg Thimm and Emile Fiesler. High order and multilayer perceptron initialization. Technical Report 94-07, IDIAP, 1994.

[8] Georg Thimm and Emile Fiesler. Evaluating pruning methods. In *1995 International Symposium on Artificial Neural Networks (ISANN'95)*, pages A2 20–25, National Chiao-Tung University, Hsinchu, Taiwan, Republic of China, December 18-20 1995.

[9] Georg Thimm and Emile Fiesler. High order and multilayer perceptron initialization. Accepted for publication in *IEEE Transactions on Neural Networks*, 1996.

[10] Mike Wynne-Jones. Constructive algorithms and pruning: Improving the multi layer perceptron. In R. Vichnevetsky and J. J. H. Miller, editors, *Proceedings of the 13th IMACS World Congress on Computation and Applied Mathematics*, volume 2, pages 747–750. International Association for Mathematics and Computers in Simulation, 1991.