A Comparative Study of Neural Network Optimization Techniques

T. Ragg, H. Braun and H. Landsberg
Institute of Logic, Complexity and Deduction Systems
University of Karlsruhe, 76128 Karlsruhe, Germany,
Email: {ragg,braun}@ira.uka.de

Abstract

In the last years we developed ENZO, an evolutionary neural network optimizer which we compare in this study to standard techniques for topology optimization: optimal brain surgeon (OBS), magnitude based pruning (MbP), and unit-OBS, an improved algorithm deduced from OBS. The algorithms are evaluated on several benchmark problems. We conclude that using an evolutionary algorithm as meta-heuristic like ENZO does is currently the best available optimization technique with regard to network size and performance. We show that the time complexity of ENZO is similar to magnitude based pruning and unit-OBS, while achieving significantly smaller topologies. Standard OBS is outperformed in both size reduction and time complexity.

1 Optimization Techniques

Optimizing the topology of neural networks is an important task when one aims to get smaller and faster networks, as well as a better generalization performance. Moreover, automatical optimization avoids the time consuming search for a suitable topology. Techniques for optimizing neural network topologies can be divided into destructive algorithms like pruning or optimal brain surgeon and constructive algorithms, e.g. cascade correlation. Our evolutionary algorithm ENZO [2, 7] is a mixture of both methods, since it allows for reduction as well as for growing structures.

We will show that ENZO surpasses magnitude pruning as well as unit-OBS by evolving topologies with significantly less size while using nearly the same amount of computing time. Moreover, the size of the evolved topologies is even smaller as constructed by the very time consuming incremental optimal brain surgeon algorithm, i.e., we get a better network size reduction using several orders of magnitudes less computing power.

The main criteria for optimizing the network topology is the size of the network, furthermore the time needed for the optimization and the classification error. A problem of pure network reduction algorithms lies in the fact that the smallest network achieving a learning error below a given error limit has not the best generalization performance in general. The tradeoff between network size and generalization capability can be balanced by ENZO using both criteria in the fitness function while MbP and OBS do not consider the generalization and achieve therefore worse generalizing networks. In order to keep the comparison clear, we do not evaluate this essential advantage of ENZO. Therefore, the only criteria is the size of the achieved neural network under the constraint that the learning error (and thereby the classification error) remains under a given error bound. The size of the network is measured by three parameters: number of input units, number of hidden units and number of weights. In typical applications of the multilayer perceptron model there are some redundant or even irrelevant input units which may decrease the generalization capability. Therefore, we are interested in both: reducing the input size and the network size. The first gives hints about the salient parameters in the input representation, the second speeds up the network evaluation and both together improve the generalization capability.

All destructive techniques are used in the same way. The networks are trained until they reach a local minimum, i.e. the mean square error is below a given bound or a maximal number of epochs is exceeded. A candidate c (either a link or a unit) is deleted and then the network gets retrained. If it is not possible to learn the patterns with the given constraints (#epochs mean square error), optimizing stops and the network before the last operation is restored. Different strategies are used to select the candidates:

G. D. Smith et al., *Artificial Neural Nets and Genetic Algorithms*
© Springer-Verlag Wien 1998

Pruning: Magnitude based pruning (MbP) searches the weight with the smallest absolute value. This weight is the candidate to delete.

Optimal Brain Surgeon: Optimal brain surgeon (OBS) was introduced by Hassibi and Stork [4]. It uses the Hessian matrix H (second derivate of the error function with respect to the weights) to compute the less important weight, i.e., the one which causes the lowest increases in error (ΔE), if pruned. This is the weight w_q which minimizes $\frac{1}{2}\Delta w H \Delta w$ such that $(\Delta w)_q = -w_q$. A special case of OBS is optimal brain damage [3], which makes the assumption that the Hessian matrix is diagonal. This simplification causes it to delete wrong weights sometimes. Thus it is inferior to OBS and is not further considered here.

Optimal Brain Surgeon with Units: A promising variant to speed up the OBS-algorithm is to prune units or several weights at a time. This variant, unit-OBS, is described in detail in [9]. Deleting several weights at once leads to a generalized equation for $w_{q_1} w_{q_2} \ldots w_{q_m}$ minimizing a term similar as for OBS such that $(\Delta w)_{q_i} = -w_{q_i}, i = 1 \ldots m$.

2 Evolutionary Algorithm

The basic principles of evolution as a search heuristic may be summarized as follows: the search points or candidate solutions are interpreted as individuals. The optimization criterion has to be one-dimensional and is called the fitness of the individual. Constraints can be embedded in the fitness function as additional penalty terms. New candidate solutions, called offspring, are created by a mutation operator using current members of the population, called parents. Weights (respectively units) can be removed or added depending on their ranking in a list, where the list is sorted with respect to a criterion reflecting the relevance of the weight respectively unit. Several criteria can be chosen, which differ in their accuracy, e.g. size of a weight or information content of an input unit. The selection of the parents is random but biased, preferring the fitter ones, i.e. fitter individuals produce more offspring. Each new inserted offspring replaces another population member with lower fitness in order to maintain a constant population size.

Evolution as a search heuristic offers the possibility of a balance between exploration and exploitation. On the one hand parallelizing the search using a population of search points and by stochastic search steps allows for a explorative search. On the other hand this explorative search is biased towards exploitation by biasing the selection of the parents, preferring the fitter ones.

3 Comparing the Strategies for Minimizing the Topology

Comparing MbP and OBS we may state that both select single connections for elimination. Whereas MbP uses a simple heuristic (select the smallest weight), this selection is done much more carefully by OBS: It approximates the error function by a Taylor approximation of second degree (quadratic polynomial) and computes for this approximation the optimal connection, i.e. the connection with smallest increase of learning error caused through its elimination. As a positive side effect, the according weight matrix is also computed which achieves the minimal increase of learning error on the Taylor approximation which reduces the number of necessary training steps drastically. A problem of this single weight elimination step is its short sightedness: it cannot handle the synergetic effect of eliminating several connections in one step but only the singular effect of eliminating just one weight. This is improved by unit-OBS, where the effect of eliminating whole groups of connections is evaluated.

All three algorithms (MbP, OBS, unit-OBS) are greedy and cannot backtrack from an unfavorable elimination. Since there are many single elimination steps any error resulting from the approximations may cut off favorable areas of the search space and therefore does not necessarily lead to an optimal reduction of the topology. These problems can be handled by ENZO: it does not only eliminate the most promising unit (or connection) but λ promising variants and evaluates their actual increase in error. Moreover, the short sightedness is reduced by following not only one line of development but a whole population of promising lines which are extended by chance according to their fitness.

4 Time Complexity

All four algorithms (MbP, OBS, unit-OBS and ENZO) proceed in the same way: they produce iteratively an offspring from a parent by deleting a unit or some connections and optimize it by gradient descent. The time complexity for producing an offspring can be subdivided into time complexity for the mutation and for the training.

The time complexity for the mutation is deter-

mined by the selection operator which specifies the unit or connections for elimination (or addition using ENZO, respectively). For MbP and ENZO, the time complexity of selection is $O(n)$ (with n = number of connections in the topology). Since one learning step has time complexity $O(pn)$ (with p = number of learning patterns), the time complexity for training maximizes the selection and therefore the time complexity for producing one offspring is $O(epn)$ (e = number of learning steps). For OBS and unit-OBS the time complexity of the selection operator is $O(pn^2)$ (cf. [4, 9]). Therefore it maximizes the time complexity of training ($O(epn)$) if $e < n$. This is true since both OBS and unit-OBS compute not only the connection (or unit respectively) with the smallest increase of the learning error but also the weight matrix, which achieves this smallest learning error. Nevertheless some learning steps are necessary, caused by the approximation error of the quadratic polynomial. In our experimental investigations the number of learning steps were typically less than ten. Therefore we may conclude that the time complexity of producing an offspring is $O(pn^2)$ for both OBS and unit-OBS.

Comparing the time complexity for all four algorithms, we have to take into account the number of offspring which are produced. Of course, this depends upon how many units or connections can be eliminated. On a minimal topology none of the algorithms can produce an improved offspring. In typical applications, however, the number of connections can be drastically reduced by less than half. Therefore we may assume, that the number of eliminated connections is $O(n)$. The elimination of a unit can be simulated by the elimination of all its connections. If we consider a weight matrix with $n = mm$ weights, then the elimination of a unit is the elimination of its according row and column, i.e. about $2m$ weights are eliminated. Therefore a unit elimination can be simulated by the elimination of $O(\sqrt{n})$ connections. Consequently we may conclude that we need $O(\sqrt{n})$ eliminations of units. Both ENZO and unit-OBS eliminate first units and then connections. Since the time complexity is proportional to the size of the network, the time consumed for producing the first offsprings is much larger than the time for eliminating single connections when the topology is already nearly minimal (such that no unit can be eliminated).

Therefore, the time complexity is dominated by the elimination of units and the elimination of connections may be neglected for both unit-OBS and ENZO. Summarizing, we get for all four algorithms the time complexities as shown in Table 1.

5 Generalization Behavior

It is well known that the generalization capability may decrease through intensively optimizing on the learning set. This overfitting effect can be handled by limiting the degree of freedom of the network topology [1]. There are two types of limitations: firstly we can reduce the number of free parameters (weights) and secondly we may penalize the size of these parameters by a weight decay factor. Both approaches are important for good generalization behavior. The reduction of the number of weights is automatically obeyed since we reduce the network topology. The penalization of the size (weight decay) has to be optimized beforehand. The weight decay factor has to be chosen so large that the overfitting effect is just suppressed. A larger weight decay factor decreases again the generalization behavior, because in this case the gradient descent priorises the minimization of the size of the weights instead of the learning error. Therefore we optimized in our experimental investigations the weight decay factor beforehand according to the above considerations.

6 Results

For all benchmarks we used fully-connected networks with shortcut connections between all layers, RPROP with weight-decay as learning algorithm [6] using 1.0 as maximal step size. The weight decay is adjusted manually in a series of training runs by increasing its value until overfitting on the validation set does not occur any more. The number of generations, the population size and the number of offspring each generation for ENZO are 30, 20 and 7, respectively.

The purpose of using an artificial benchmark like

Table 1: Time complexity of the four examined optimization algorithms. e is the number of training epochs, p the number of patterns, n the number of weights and λ the number of offspring.

MbP	OBS	unit-OBS	ENZO
$O(epn)O(n)$	$O(pn^2)O(n)$	$O(pn^2)O(\sqrt{n})$	$O(epn)O(\lambda\sqrt{n})$
$= O(epn^2)$	$= O(pn^3)$	$= O(pn^{2.5})$	$= O(\lambda epn^{1.5})$

the TC-Problem, is solely to examine if the optimization algorithm is able to find a minimal topology that still solves the problem with a 100% correctness and the probability to do so, i.e., to determine the dependency from the initialization. The training set consists of all possible T's and C's on a 4×4 pixel matrix, i.e. 17 T's and 23 C's. The topology is 16-16-1 with 288 weights in total. Table 2 shows that only unit-OBS and ENZO were able to determine the minimal topology, where the average results of ENZO are still better.

Breast Cancer is a real world benchmark from the UCI benchmark collection. A tumor is to be classified as benign or malignant based on cell description, e.g. the cell size and shape, gathered by microscopic examination. The network topology is 9-8-4-2 with weights. The data consists of 350 training patterns, 174 validation and 175 test patterns, from which 65.5% are benign. We needed only about 700 training epochs to get satisfactory results and a classification error between 2% and 3%. We used a weight decay of 10^{-2}. Other parameters were set as above.

Thyroid Gland is also a real-world benchmark available at the UCI repository. This task requires a very good classification, because 92% of the patterns belong to one class, thus useful networks should have a classification error of less than 2%. The data consists of 1886 training patterns, 1886 validation and 3428 test patterns. The topology is 21-10-3, with 304 weights. The classification error was for all methods between 1% and 2%, which is in the same range as reported in [8] for an evolved network (284 weights, 1.4% classification error). The smallest network was evolved by ENZO using 15 weights achieving a classification error of 1.0%.

Table 2: Results of the optimization processes for the TC-Problem. The first two columns give the results for the median from 21 runs, i.e., the 11th best run. The other two rows are the values for the best run.

	Pruning	OBS	unit-OBS	ENZO
median topology	14-5-1	13-5-1	11-4-1	9-2-1
median #weights	33	29	23	22
best topology	12-2-1	12-4-1	8-2-1	8-2-1
best #weights	20	22	22	18

Table 3: Results of the optimization processes for breast cancer classification.

	Pruning	OBS	unit-OBS	ENZO
median topology	8-6-2	8-3-2	5-2-2	3-2-2
median #weights	29	29	17	10
best topology	7-6-2	8-3-2	5-1-2	3-1-2
best #weights	26	26	12	8

Table 4: Results of the optimization processes for thyroid gland classification.

	Pruning	OBS	unit-OBS	ENZO
median topology	7-10-3	9-6-3	8-6-3	6-1-3
median # weights	45	42	29	21
best topology	6-10-3	7-4-3	6-3-3	5-1-3
best # weights	26	26	16	15

7 Discussion

Optimization techniques were compared for several benchmarks with regard to their performance in minimizing the network size. Our results show that standard techniques depend heavily on the initialization of the network, as well as on the weight decay term. On the one hand, using weight decay makes it easier to prune minor important weights and thus leads to better results. On the other hand it decreases the degree of freedom for fitting the weights and thereby may increase the minimal network size.

The time consuming optimal brain surgeon algorithm is outperformed by unit-OBS with respect to network size reduction and computing time. Moreover, our results show that ENZO surpasses both magnitude pruning and unit-OBS by evolving topologies with significantly less size while all algorithms have about the same time complexity.

We conclude that using evolutionary algorithms as a meta-heuristic as done in ENZO is concurrently the best available optimization method, with regard to network size and performance. Furthermore, it is scalable in time and simple to parallelize [5]. Extra constraints on network topology or performance are easily integrated into the fitness function. Thus evolution combined with learning appears to be the best framework with which to train neural networks.

References

[1] C. M. Bishop. *Neural Networks for Pattern Recognition.* Oxford Press, 1995.

[2] H. Braun and T. Ragg. ENZO – Evolution of Neural Networks, User Manual and Implementation Guide, http://i11www.ira.uka.de. Technical Report 21/96, Universität Karlsruhe, 1996.

[3] Y. L. Cun, J.S. Denker, and S.A. Solla. Optimal Brain Damage. In *NIPS 2*, 1990.

[4] B. Hassibi and D. G. Stork. Second order derivatives for network pruning: Optimal Brain Surgeon. In *NIPS 4*, 1992.

[5] T. Ragg. Parallelization of an Evolutionary Neural Network Optimizer Based on PVM . In *Parallel Virtual Machine - EuroPVM'96*, Lecture Notes in Computer Science 1156, 1996.

[6] M. Riedmiller and H. Braun. A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP Algorithm. In *Proceedings of the ICNN*, 1993.

[7] J. Schäfer and H. Braun. Optimizing classifiers for handwritten digits by genetic algorithms. In *Artificial Neural Networks and Genetic Algorithms*, D. W. Pearson, N. C. Steele, R. F. Albrecht (editors), pages 10–13, Wien New York, 1995. Springer-Verlag.

[8] W. Schiffmann, M. Joost, and R. Werner. Application of genetic algorithms to the construction of topologies for multilayer perceptrons. In *Artificial Neural Networks and Genetic Algorithms*, . D. W. Pearson, N. C. Steele, R. F. Albrecht (editors), pages 675–682, Wien New York, 1993. Springer-Verlag.

[9] A. Stahlberger and M. Riedmiller. Fast network pruning and feature extraction by removing complete units. In *NIPS 9*. MIT Press, 1997.