A Novel Neural Network Training Technique based on a Multi-Algorithm Constrained Optimization Strategy

D.A. Karras and I.E. Lagaris

14-96

Preprint no. 14-96/1996

Department of Computer Science
University of Ioannina
45 110 Ioannina, Greece

A Novel Neural Network Training Technique based on a Multi-Algorithm Constrained Optimization Strategy

D.A. Karras and I.E. Lagaris

Indexing term: Multilayer Perceptron Training.

A novel methodology for efficient offline training of multilayer perceptrons (MLPs) is presented. The training is formulated as an optimization problem subject to box-constraints for the weights, so as to enhance the network's generalization capability. An optimization strategy is used combining variable metric, conjugate gradient and no-derivative pattern search methods that renders the training process robust and efficient. The superiority of this approach is demonstrated by direct application to two real world benchmarks and the parity-4 problem.

Introduction: Minimizing the MLP error function in realistic problems is a difficult task since the many layers, the multitude of training patterns and the variety of categories cast a very complex landscape with wide plateaus and narrow valleys [1]. There is no single algorithm that can be used as a panacea to solve such optimization problems. Algorithms that use gradient information perform well only at regions of the parameter space where the function is smooth, while algorithms using only function values may be effective at regions

where the derivatives are not well defined. The other competitive objective is to train the MLPs so as to obtain high quality generalization. Unconstrained optimization algorithms often permit some weights to become arbitrarily large, which results in a deterioration of the network's generalization since the other weights don't play any role to it. From this point of view, the main weak points of the existing MLP training procedures are the use of a single optimization algorithm and the lack of control on the weight growth. To address these issues we propose a novel multi-algorithm box-constrained optimization procedure governed by a strategy that exploits the virtues and strengths of the participating algorithms. This renders the procedure efficient and robust and although it is an established approach in the field of optimization [2], it has never been employed in MLP training before. To control weight growth and thus to ensure a high level of generalization we suggest box constraints, so that the weights assume values inside a desired reasonable range. Previous attempts to improve generalization ability relied on regularization schemes, like "weight decay/ elimination" [1], that merely add a penalty term to the error function to discourage large weight values. This is an indirect and rather inefficient way to treat constraints and moreover, it does not provide explicit weight-control. To implement our method we used the MERLIN /MCL package [2,3] that offers an environment with several powerful optimization algorithms, the ability to impose box constraints and a language to create strategies.

Constrained Optimization Strategy for the Training Problem: The suggested procedure uses three different algorithms, specifically the quasi-Newton BFGS [1], the Polak-Ribiere (PR) conjugate gradient algorithm [1] and a pattern search method (ROLL [2]) that uses only function values. Pattern search methods have not been used in MLP training so far. Since the above algorithm employs no derivatives it is expected to be effective at the regions with plateaus of the weight space where the BFGS and PR techniques that use gradient information fail to perform. Since the ROLL method is not widely known we provide a brief description for it. Let $E(W_1, W_2, \ldots, W_n)$ be the error function in MLPs with W_j corresponding to the weight variables. Let, also, $W^c = (W_1^c, W_2^c, \ldots, W_n^c)$ be the current point in the optimization process of E and $E_c = E(W^c)$. Finally, let S_i be a step associated with each free variable W_i.

1. Pick a trial point: $W_j^t = W_j^c$ for all $j \neq i$ and $W_i^t = W_i^c + S_i$
2. Calculate $E_+ = E(W^t)$.
3. if $E_+ < E_c$ set $W^c = W^t, E_c = E_+$ and $S_i = aS_i$. Then, go to step 8.
4. if $E_+ >= E_c$ pick another trial point as : $W_j^t = W_j^c$ for all $j \neq i$ and $W_i^t = W_i^c - S_i$
5. Calculate $E_- = E(W^t)$.
6. if $E_- < E_c$ set $W^c = W^t, E_c = E_-$ and $S_i = -aS_i$. Then, go to step 8.

7. if $E_- >= E_c$ calculate an appropriate step by: $S_i = -\frac{1}{2}\frac{(E_+ - E_-)}{(E_+ + E_- - 2E_c)} S_i$.

8. Proceed with step 1 for the next value of i.

In the above, $a > 1$, is a user set factor (in our experiments $a = 3.0$). If after looping over all variables there is no progress, a line search is performed in the direction $S = (S_1, S_2, \ldots, S_n)$. The above procedure is repeated until a preset number of calls to the objective function is reached.

Box constraints: Let $[a_i, b_i]$ be the desired range for the weight W_i. Then in the ROLL method the W_i^t is forced inside this range via the following statements.

If $W_i^t < a_i$ set $W_i^t = \frac{a_i + W_i^c}{2}$

if $W_i^t > b_i$ set $W_i^t = \frac{b_i + W_i^c}{2}$

The BFGS and the PR methods proceed via line searches. The feasible segment along the direction of search S is determined first, by simply calculating with analytical geometry the sections of S with the hyper-box defined by the given box-constraints, and then it is used to bracket the line search.

In what follows we give a rather detailed account of the proposed strategy that was coded in MCL [3]. (It is worth mentioning that the actual code is less than 70 statements).

Initialization: Pick at random an initial set of weights all in $[-1, 1]$.

Set the initial range for the weights to be $[-b, b]$ (with $b = 1$).

Set the maximum allowed number of calls to the error function.

Set the range enhancement α.(We used $\alpha = 1.1$).

Set the maximum allowed range to be $[-d, d]$ (We used $d = 6$).

Set the target value (a satisfactory value for the error function) E_0.

Set the value for the rate of progress r. (We used $r = \frac{1}{100}$).

Step (1): Test the number of calls to decide whether to stop or not.

Step (2): Determine and fix the non-influential weights. These weights w have the property $|\frac{\partial E}{\partial w}| < \epsilon$, where $\epsilon > 0$ a small preset value, i.e the error function is not very sensitive to changes in these weights. This step adds efficiency since at this point these weights are not important.

Step (3): Apply in succession the BFGS and the ROLL algorithms (this adds efficiency and robustness since these two methods are succesful for different types of landscapes).

Step (4): Redetermine the non-influential weights and fix them (temporarily fixing non-influential weights is beneficial since, due to dimensionality reduction, the optimization problem becomes easier).

Step (5): Test if E_0 has been reached to decide whether to stop or not

Step (6): If the relative rate of progress per call $\frac{1}{Noc}\frac{\Delta E}{E} \leq r$ enhance the weight range as $b = min(d, b\,\alpha)$ (Noc = Number of calls).

Step (7): Apply the PR method (Usually it is less efficient than BFGS, but performs less bookkeeping operations).

Step (8): Repeat from step (1).

Experimental results: To demonstrate the efficiency of our approach in MLP

training we considered two real problems and the parity-4 benchmark, since the increasing demand for high performance neural networks in real world applications renders obsolete any research based only on artificial benchmarks like XOR etc. Both real problems were selected from the Proben1 real world benchmark collection [4], since they are considered especially difficult and hence suitable for testing. In the first problem the approval of a credit card to a customer should be predicted, while in the second the diabetes of Pima Indians should be diagnosed. There are 51 (8) inputs, 2 outputs and 690 (768) examples divided randomly three times in 345 (384), 173 (192) and 172 (192) patterns for training, validating and testing respectively, hence forming card1, card2 and card3 (diabetes1, diabetes2 and diabetes3) tasks. In Table 1 we compare the results obtained in these six tasks by our Multi-Algorithm Constrained Optimization (MACO) methodology against to those obtained by the offline Backpropagation (Off-BP) ($learning - rate = 0.01, momentum = 0.05$) and the Polak-Ribiere Conjugate gradient method (PR-BP), according to Proben1 specifications [4] concerning architectures, error measures and number of runs. MACO clearly outperforms the other methods as well as the RPROP algorithm used in Proben1 in terms of training and generalization average error reduction (notice an improvement of 16-90% for the former and 4-16% for the latter regarding the best results obtained in Proben1 [4] with no-shortcut architectures). The significant reduction in average training error obtained

by MACO is worth noting despite the stringent maximum bounds set to the weights ($W_i \in [-6, 6]$ for all i). Fig.1 clearly illustrates the superiority of the multi-algorithm training strategy in the notoriously difficult parity-4 benchmark. It is shown that a 4-4-1 MLP trained via a strategy (the same as MACO but with PR substituting BFGS) using first the PR method and subsequently, when a slow progress is encountered, the ROLL method achieves a mean square error of only $1.6\ 10^{-18}$. On the other hand the same network trained with either PR or BFGS alone, achieves the significantly higher mean square error values of 0.5 and 2.5 respectively.

Conclusions: A novel multi-algorithm constrained optimization training strategy for MLPs using box constraints was presented, exhibiting superior performance in both the generalization ability and in the reduction of the training error in several difficult real and artificial benchmarks.

D.A. Karras and I.E. Lagaris (*Department of Computer Science, University of Ioannina, GR-45 110, Ioannina, Greece*)

One of us (I. E. L.) acknowledges partial support from the Greek General Secretariat of Research and Technology under contract PENED 91 ED 959.

References

1 HAYKIN, S.: 'Neural Networks A comprehensive introduction', *Macmillan Publishing Company*, 1994

2 EVANGELAKIS, G.A. AND RIZOS, J.P. AND LAGARIS, I.E. AND DEMETROPOULOS, I.N.: 'MERLIN - A portable system for multi-dimensional minimization', *Computer Physics Communications*, 1987, **46**, pp. 401-415

3 CHASSAPIS, C.S. AND PAPAGEORGIOU, D.G. AND LAGARIS, I.E.: 'MCL - Optimization Oriented Programming Language', *Computer Physics Communications*, 1989, **52**, pp. 223-239

4 PRECHELT, L: 'PROBEN1 - A set of Neural Network Benchmark Problems and Benchmarking Rules', *Technical Report 21/94*, September 30, 1994, *Fakultat fur Informatik, Universitat Karlsruhe, Germany*

Table 1:

Problem	Average (60 runs) training/validation/test error			
	MACO	Proben1	PR-BP	Off-BP
card1	**1.06/8.24/9.83**	8.86/8.69/10.35	8.83/8.75/10.40	9.10/8.88/10.55
card2	**0.82/10.39/14.25**	7.18/10.87/14.94	8.47/10.95/15.10	8.12/11.10/15.22
card3	**0.85/8.15/12.73**	7.13/8.62/13.47	7.50/8.58/13.40	7.96/9.12/13.75
diabetes1	**12.12/13.98/14.25**	14.36/15.93/16.99	14.10/15.80/16.81	14.98/16.40/16.97
diabetes2	**10.45/15.10/16.04**	13.04/16.94/18.43	13.32/17.05/18.40	12.88/17.20/18.60
diabetes3	**10.19/15.97/14.18**	13.52/17.89/16.48	13.79/17.95/16.35	14.01/18.43/16.60

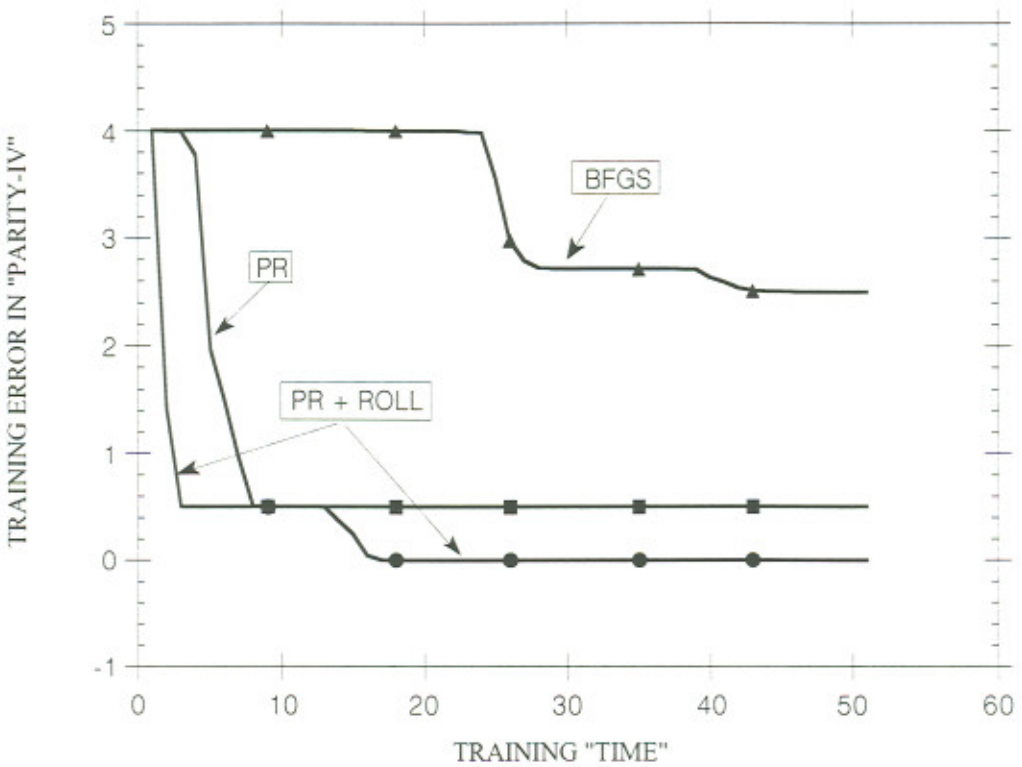
TRAINING ERROR IN "PARITY-IV"
5
4
3
2
1
0
-1
0
10
20
30
40
50
60
TRAINING "TIME"
BFGS
PR
PR + ROLL

Figure captions:

Figure 1: The impressive training error reduction in a 4-4-1 MLP employing PR+ROLL, compared to the one obtained by using either PR or BFGS, in the parity-4 benchmark.

Table captions:

Table 1: The significant improvements obtained in the training/ validation/ generalization average (60 runs) error per pattern and output node (multiplied by 100) in the MLPs when using MACO.