Matrix-update for accelerated on-line learning in multilayer neural networks

J. Phys. A: Math. Gen. **31** (1998) L413–L417. Printed in the UK PII: S0305-4470(98)91523-2

LETTER TO THE EDITOR

Matrix-update for accelerated on-line learning in multilayer neural networks

Siegfried Bös[†]

Lab for Information Synthesis, Brain Science Institute, RIKEN, Wako-shi, Saitama 351-01, Japan

Received 10 February 1998, in final form 20 March 1998

Abstract. In this letter we re-examine the emergence of plateaus or symmetric phases in on-line learning of a committee machine. We propose a simple matrix-update in order to avoid the symmetric phase. Simulations show that the length of the plateaus can be considerably decreased. Annealing of the learning rate can then be applied much earlier, making on-line learning more effective.

Learning from examples and generalizing the acquired knowledge to unknown data is the key property of neural networks. Among all learning algorithms, on-line learning has recently attracted considerable attention [1–3]. In on-line learning, the weights are updated by using only the example presented at time t, i.e.

$$\boldsymbol{W}(t+1) = \boldsymbol{W}(t) + \eta\Delta\boldsymbol{W}[\boldsymbol{x}(t), z^*(t); \boldsymbol{W}(t)]. \tag{1}$$

The advantages are rather obvious, the update is very fast and simple, while memory to store all examples is not necessary. It is also possible to enable the system to follow non-stationarities in the data by emphasizing more recent examples.

The learning rate η plays a much more important role in on-line learning than in iterative batch learning. Good results can already be achieved by a simple time-independent learning rate η_0. In unrealizable tasks, which can never be learnt exactly, the following *dilemma* occurs, see [2, 4]. While a large η_0 smaller than the limit $\eta_{\max}$ can accelerate the convergence, it also leads to a large final error. The final error results from fluctuations around the optimum and the variance of these fluctuations is proportional to η_0. Only an asymptotically vanishing learning rate, i.e. $\eta \to 0$, can reach the optimum. This suggests a time-dependent learning rate $\eta(t)$, which is annealed during training. Recently, it has been proved [5, 6] that on-line learning with an optimally annealed learning rate can be asymptotically efficient, i.e. it can be as good as the best possible solution. Until now, this result has only been tested in single-layer models [4]. In studies of on-line learning in multi-layer networks [7, 8], a metastable *symmetric phase* was found, which makes an effective annealing impossible. Several efforts have been made to shorten the time spent in the symmetric phase. However, they are either not practical as they use explicit knowledge about the learning task [9, 10] or very complicated [11, 12]. In this letter we want to show that on-line learning with a rather simple matrix-update can considerably reduce the time spent in the symmetric phase. An effective annealing is then possible.

[†] E-mail address: boes@islab.brain.riken.go.jp

The model we want to use for illustration is the *soft-committee machine*, which was discussed in detail in [7, 8]. The committee machine is a special two-layer network with only one-layer of adjustable weights $\boldsymbol{W}$ between the N input units and the H hidden units. The weights between hidden units and output z are fixed to one, such that

$$z = \sum_{k=1}^{H} g(\boldsymbol{W}_k^T \boldsymbol{x}). \tag{2}$$

A soft-committee machine has continuous outputs by applying a sigmoid function $g(h)$ on the local fields h. We study supervised learning in a *teacher–student scenario*. This means that the correct outputs z^* are given by another network, the teacher. In order to make the task unrealizable, we assume that the teacher network and the student network have identical architectures, but the teacher outputs z_0^* are corrupted by a random Gaussian noise $\epsilon \in \mathbb{N}(0, \sigma)$, yielding $z^* = z_0^* + \epsilon$. For training, the teacher network is invisible to the student, only the target outputs z^* can be facilitated. For the evaluation of the performance, all variables of the teacher can be used.

Supervised training minimizes the difference between the correct output z^* given by the teacher and the actual output z produced by the student, i.e. the loss-function $\text{loss}(z^*, z) := \frac{1}{2}[z^* - z]^2$. The overall performance of the student is measured by the averaged loss over all possible inputs, called *generalization error* $E_{\text{G}} = \langle \text{loss}(z^*, z) \rangle_x$.

In the teacher–student scenario, the generalization error can be calculated if we make an assumption about the distribution of the inputs $\boldsymbol{x}$. For random inputs $\boldsymbol{x}$, the dot-products $h_k^* := \boldsymbol{W}_k^{*T} \boldsymbol{x}$ and $h_l := \boldsymbol{W}_l^T \boldsymbol{x}$ become Gaussians if the input-dimension N becomes large. The correlations between these two Gaussian-variables are $\langle h_k^* h_l^* \rangle_x = S_{kl}$, $\langle h_k^* h_l \rangle_x = R_{kl}$ and $\langle h_k h_l \rangle_x = Q_{kl}$. They define two sets of dynamical *order parameters*,

$$R_{kl} := \boldsymbol{W}_k^{*T} \boldsymbol{W}_l^* \qquad \text{and} \qquad Q_{kl} := \boldsymbol{W}_k^T \boldsymbol{W}_l \tag{3}$$

and the constant task-dependent parameter $S_{kl} := \boldsymbol{W}_k^{*T} \boldsymbol{W}_l^*$.

The generalization error can then be calculated by an average over the correlated Gaussians h_k^* and h_l. By using the error-function $g(h) = \text{erf}(h/\sqrt{2})$, we can find an algebraic expression for the redefined generalization error $E_{\text{G}} := E_{\text{G}} - \frac{\sigma^2}{2}$, which is for $S_{kl} = \delta_{kl}$,

$$E_{\text{G}} = \frac{H}{6} - \frac{2}{\pi} \sum_{k,l=1}^{H} \arcsin \frac{R_{kl}}{\sqrt{2(1 + Q_{ll})}} + \frac{1}{\pi} \sum_{k,l=1}^{H} \arcsin \frac{Q_{kl}}{\sqrt{(1 + Q_{kk})(1 + Q_{ll})}}. \tag{4}$$

Usually the weights are adapted by a *scalar gradient descent update*,

$$\eta \Delta \boldsymbol{W}_k(t) = -\eta \nabla_{\boldsymbol{W}_k} \text{loss}[z^*(t), z(t)]. \tag{5}$$

It is local as it uses, for the update of the hidden unit k, only the information available at the hidden unit k. On-line training of the committee machine with a scalar-update and a fixed learning rate η_0 was studied by Saad and Solla [7, 8]. The characteristic behaviour is shown in figure 1.

For small and large numbers of examples, the dilemma common to online learning with a fixed learning rate becomes apparent. With a larger learning rate it is possible to achieve a faster convergence for small P/N, however the final result for large P/N is worse due to fluctuations. Characteristical for the scalar gradient descent update is the long plateau of the learning curve in the intermediate range, where the generalization error is extremely slowly decreasing. The reason of the appearance of the plateau is the nearly perfect symmetry of the H subperceptrons, which can be seen in the values of $Q_{kl} \simeq Q$.

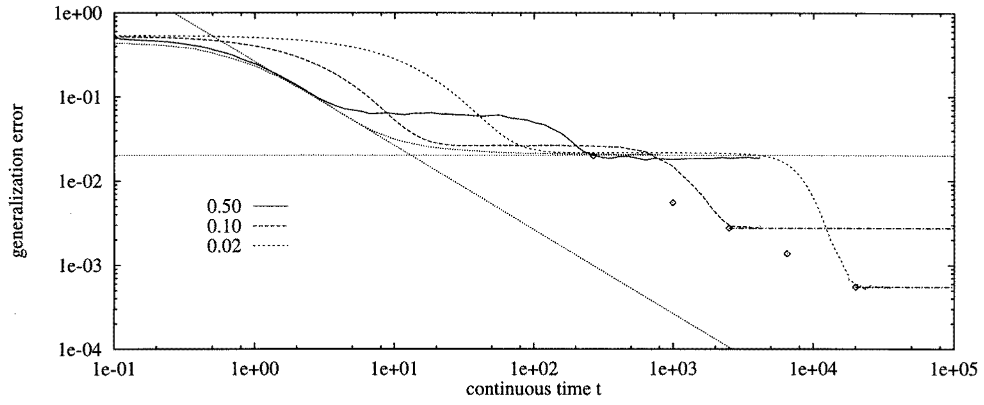

Figure 1. Learning curves for on-line learning of a committee machine with scalar gradient descent update and fixed η_0. The generalization error E_G is shown as a function of the normalized number of examples P/N. Different learning rates η_0 have been used (from 0.5 to 0.02 corresponding to the final theoretical values on the right-hand side). The problem of the fixed learning rate can be seen, a larger η_0 leads to a faster convergence, but also produces a worse final result. The *plateaus* are the intermediate regions, where the curves are horizontal for a long time. The simulation corresponds to the analytical solution of [8, 9]. (Parameters: task $H = 3$ and noise level $\sigma^2 = 0.1$; asymmetric initial conditions $Q_{kl} \simeq kQ_0\delta_{kl}$ with $Q_0 = 0.1$; dotted curves: curved one, see close to $t = 10$, is simple annealing of the scalar update, the diagonal one is the tangent on its fastest decrease, and the horizontal one its final state, which is also the lowest plateau value.) Note, that the scale of both figures is double-logarithmic.

The system remains in this symmetric phase for quite a long time, since the breaking of the symmetry based purely on fluctuations takes a long time. Also the plateaus are left earlier, if the learning rate is larger, see figure 1. From this, we can conclude that the learning rate has to remain considerably large until the symmetric phase has been passed. To start annealing at the beginning, would lead directly into the symmetric phase. Further annealing would probably make it impossible to leave the symmetric phase again (dotted curve in figures).

In order to break the symmetry between the hidden units earlier, a non-local update is necessary. It must facilitate information from other hidden units. This can be achieved by a *matrix-update*, i.e.

$$\eta\Delta \boldsymbol{W}_k(t) = -\eta \sum_{l=1}^{H} (\boldsymbol{G}^{-1})_{kl} \nabla_{\boldsymbol{W}_l} \mathrm{loss}[z^*(t), z(t)]. \tag{6}$$

What should the matrix $\boldsymbol{G}_{kl}$ look like? It should only depend on accessible information, which means it can depend on the Q_{kl}, but not on the unknown R_{kl} or S_{kl}. In this letter, we show that the following choice can yield interesting results,

$$G_{kl} = \frac{2}{\pi}(1 + Q_{kk} + Q_{ll} - 2Q_{kl})^{-\frac{1}{2}}. \tag{7}$$

A theoretical derivation is not yet available and also not within the scope of this letter. The proposed choice (7) was found empirically, starting from the '*natural gradient*' proposed by Amari [5]. The important property of the matrix G_{kl} is its symmetry in the Q_{kl}. The matrix $\boldsymbol{G}$ cannot be inverted in the symmetric phase, when $\boldsymbol{Q}$ is not of full rank. Preliminary results indicate, that similar results can be achieved by using the 'natural gradient' [14, 15].

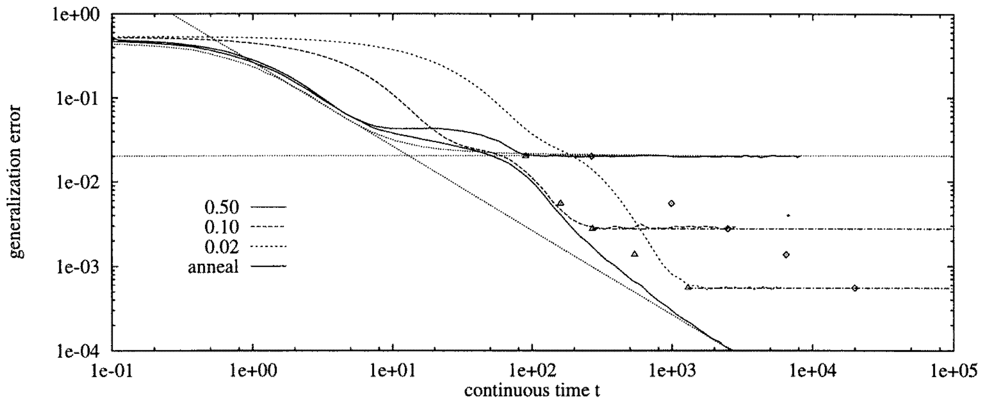

Figure 2. Performance using matrix-update. The dilemma of the fixed learning rate appears again. The plateaus, however, are much shorter and disappear completely for very small learning rates. The same value of η_0 leads to the same final error as in the scalar update, however, it is reached much earlier than in figure 1, also indicated by the triangles versus the diamonds. The lowest full curve shows the results which can be achieved by matrix-update with an optimally annealed learning rate. The system is still attracted by the symmetric phase which slows the learning speed down. However, once the plateau is passed, learning speed can accelerate again and it can catch up with its former fastest descent, shown by the dotted tangent-line. The tangent line is the continuation of the fastest descent between $P/N = 2$ and 8; it is also shown in figure 1. (Parameters and dotted lines as in figure 1. Optimized annealing parameters for the task with $H = 3$ and $\sigma^2 = 0.1$ are $b = 6$, $d = 0.1$, $\tau = 100$, and $c = 0.6$.)

The singularity of $\boldsymbol{G}$ makes the *initial conditions* important. They are often chosen symmetrically, however, they should now be asymmetric. We can choose random initial weights $\boldsymbol{W}(0)$, such that the initial values fulfil, $Q_{kl} \simeq kQ_0\delta_{kl}$ with a certain Q_0. The overlaps with the teacher should remain unaffected and be only of the order of random fluctuations. The initial conditions have an influence on the length of the plateaus, which was studied for symmetric initial conditions in [13]. Some results on asymmetric initial conditions can be found in [14]. To make the comparison between matrix-update and scalar-update fair, we have used the same asymmetric initial conditions in both cases, i.e. in both figures.

The effect of the matrix-update becomes immediately apparent when we use it with a fixed learning rate η_0. Results are shown in figure 2. We can see that the same η_0 leads to the same final error, which is another reason for this choice of $\boldsymbol{G}$. The final error is, however, reached much faster with the matrix-update. A closer look reveals that the plateaus are much shorter than in the case of the scalar-update. For small values of η_0, they can completely disappear.

As the symmetric phase is passed earlier, we can also apply annealing much earlier. To determine the annealing scheme $\eta(t)$, we should specify the initial condition $\eta(t \to 0) = c$ and the asymptotical scaling law $\eta(t \to \infty) = b/t$. Furthermore, the learning rate should remain considerably large until time τ when the plateau is passed, such that $\eta(\tau) = d > 0$. This can be summarized in the following scheme,

$$\eta(t) = \begin{cases} c - (c-d)f(t,\tau) & \text{for } t \leqslant \tau \\ \left(\dfrac{1}{d} + \dfrac{t-\tau}{b}\right)^{-1} & \text{for } t \geqslant \tau. \end{cases} \tag{8}$$

To find a lower bound of the performance, we determine the optimal values of b, d, τ, c

and the function $f(t, \tau)$, which minimize the generalization error. Only the parameter b affects the asymptotical scaling of the generalization error. The parameters d and τ can delay the onset of the asymptotical scaling. And finally c and $f(t, \tau)$ have only an effect on the behaviour above the plateau. A slow decrease of the learning rate above the plateau as $f(t, \tau) = \log(1+t)/\log(1+\tau)$ is better than $f(t, \tau) = t/\tau$.

In practice, when the generalization error is unknown, additional knowledge about the performance is required, which can be provided by a validation scheme such as test-set validation or cross validation. At this point we do not know, whether the optimal value of b can be determined without additional knowledge about the task. This question should be answered by a theory of the matrix-update. The results of optimal annealing are also shown in figure 2.

In this letter, a matrix-update was proposed to accelerate on-line learning in multi-layer neural networks. The proposed approach facilitates the updates of all hidden units and is therefore able to break the symmetry responsible for the plateaus in the learning curves. It is based on a matrix inversion of a $H \times H$ matrix, and is feasible as long as the number of hidden units H is small. Until now, matrix-update has only been proposed in theoretical proofs of the efficiency of on-line learning [5, 6]. It has never been studied in an explicit application to a multilayer neural net. It should be emphasized that we are only at the beginning with our knowledge on matrix-update. The promising results of this short letter should stimulate further interest in this direction.

I would like to thank S Amari and all the participants of the ‘themed week on on-line learning’ at the Newton Institute in Cambridge for stimulating discussions. I thank E Helle for assistance concerning the presentation.

References

[1] Amari S 1967 *IEEE Trans. Elect. Comput.* **EC-16** 299
[2] Murata N 1992 *PhD Thesis* University of Tokyo (in Japanese) unpublished
[3] Biehl M and Schwarz H 1993 Learning drifting concepts with neural networks *J. Phys. A: Math. Gen.* **26** 2651
[4] Bös S, Murata N, Amari S and Müller K-R 1997 The role of the learning rate in on-line learning submitted
[5] Amari S 1998 Natural gradient works efficiently in learning *Neural Comput.* **10** 251
[6] Opper M 1996 Online versus offline learning from random examples: General results *Phys. Rev. Lett.* **77** 4671
[7] Saad D and Solla S A 1995 On-line learning in soft committee machines *Phys. Rev.* E **52** 4225
Saad D and Solla S A 1997 Exact solution for on-line learning in multilayer neural networks *Phys. Rev. Lett.* **74** 4337
[8] Saad D and Solla S A 1997 Learning from corrupted examples in multilayer networks submitted
[9] Vicente R and Caticha N 1997 Functional optimization of online algorithms inmultilayer neural networks *J. Phys. A: Math. Gen.* **30** L599
[10] Rattray M and Saad D 1997 Globally optimal on-line learning rules for multilayer networks *J. Phys. A: Math. Gen.* **30** L771
[11] West A H L and Saad D 1996 *Advances in Neural Information Processing Systems 8 (NIPS'95)* ed D S Touretzky *et al* (Cambridge, MA: MIT) p 321
[12] Yang H H and Amari S 1997 Natural gradient descent for training multi-layer perceptrons submitted
[13] Biehl M, Riegler P and Wöhler C 1996 Transient dynamics of on-line learning in two-layered neural networks *J. Phys. A: Math. Gen.* **29** 4769
[14] Bös S and Amari S 1998 Annealed online learning in multilayer neural networks submitted
[15] Rattray M and Saad D 1998 Incorporating curvature information into online learning submitted