On–line Learning with a Perceptron

Michael Biehl*
CONNECT, The Niels Bohr Institute
Blegdamsvej 17, Dk–2100 Copenhagen Ø, Denmark

Peter Riegler+
Institut für theoretische Physik
Julius–Maximilians–Universität Würzburg
Am Hubland, D–97074 Würzburg, Germany

Abstract

We study on–line learning of a linearly separable rule with a simple perceptron. Training utilizes a sequence of uncorrelated, randomly drawn N–dimensional input examples. In the thermodynamic limit the generalization error after training with P such examples can be calculated exactly. For the standard perceptron algorithm it decreases like $(N/P)^{1/3}$ for large P/N , in contrast to the faster $(N/P)^{1/2}$–behavior of the so–called Hebbian learning. Furthermore, we show that a specific parameter–free on–line scheme, the AdaTron algorithm, gives an asymptotic (N/P)–decay of the generalization error. This coincides (up to a constant factor) with the bound for any training process based on random examples, including off–line learning. Simulations confirm our results.

PACS. 87.10, 02.50, 05.90

A very important feature of Feedforward Neural Networks is their ability to learn a rule from examples [1, 2]. Methods known from Statistical Mechanics have been successfully used to study this property [3–5], mainly for the so–called simple perceptron [6, 7].

Usually, learning is interpreted as an optimization process in the space of network parameters or *weights*. The corresponding cost function measures the performance of the trained network (the *student*) on a given set of examples. This is often termed *off–line learning* [1]. In the following, however, we study *on–line learning* of a linearly separable rule with a simple perceptron. Here,

*biehl@physik.uni-wuerzburg.de, + pr@physik.uni-wuerzburg.de

only the latest in a sequence of examples determines the actual change of the student weights in an iterative learning scheme. No explicit storage of a training set is required and the student's performance on earlier examples is not taken into account. One would expect on–line learning to be less efficient in terms of the number of examples needed for good generalization, compared with costly off–line schemes which, for example, minimize the training error. In the course of this letter we will show, however, that this is not necessarily true.

Consider a linearly separable target rule

$$S_{\underline{B}}(\underline{\xi}) = \text{sign}\left(\underline{B}\cdot\underline{\xi}\right), \tag{1}$$

represented by the normalized *teacher vector* $\underline{B} \in I\!R^N$, $\underline{B}^2 = 1$. The student perceptron is trained with a sequence $\{\underline{\xi}^\mu, S^\mu = S_{\underline{B}}(\underline{\xi}^\mu)\}_{\mu=1,\ldots,P}$ of examples for the rule. The aim is to infer as much information as possible from these examples and choose a vector $\underline{J}$, which represents a hypothesis for the unknown rule. The learning algorithm considered in the following is of the generic form

$$\underline{J}^{\mu+1} = \underline{J}^\mu + \frac{1}{N} f\left(S^\mu, h^\mu_{\underline{J}}\right)\underline{\xi}^\mu S^\mu, \tag{2}$$

where the weight function f defines the actual algorithm. It depends only on quantities that are accessible during the process of learning, like the teacher output S^μ and the internal student field $h^\mu_{\underline{J}} = \underline{J}^\mu \cdot \underline{\xi}^\mu$ induced by the current example.

At each 'time step' μ a new, uncorrelated input $\underline{\xi}^\mu \in I\!R^N$ is presented. Its components ξ^μ_j are taken to be *i.i.d.* random numbers with zero mean and unit variance.

From equation (2) one can derive recursion relations for the *order parameters* $Q^\mu = \sqrt{\underline{J}^\mu \cdot \underline{J}^\mu}$ and $R^\mu = \underline{J}^\mu \cdot \underline{B}$. The dependence on the latest training example is only through the internal fields $h^\mu_{\underline{J}}$ and $h^\mu_{\underline{B}} = \underline{B}\cdot\underline{\xi}^\mu$, which, for large N, are correlated Gaussian variables with

$$\left\langle (h^\mu_{\underline{B}})^2 \right\rangle = 1, \qquad \left\langle (h^\mu_{\underline{J}})^2 \right\rangle = (Q^\mu)^2, \qquad \text{and } \left\langle h^\mu_{\underline{J}} h^\mu_{\underline{B}} \right\rangle = R^\mu. \tag{3}$$

We assume selfaveraging properties for the overlaps as $N \to \infty$. In the same limit we can introduce the 'continuous time' $\alpha = \mu/N$ and get a system of differential equations for the order parameters R and Q. Alternatively one can derive the equation for the normalized overlap $\rho = R/Q$. Any two of these equations describe the learning completely, e.g.

$$\frac{d\rho}{d\alpha} = \left\langle \frac{f(S,h_{\underline{J}})S}{Q}\left(h_{\underline{B}} - \frac{\rho}{Q}h_{\underline{J}}\right) - \frac{\rho}{Q^2} f^2(S,h_{\underline{J}})\right\rangle, \tag{4}$$

$$\frac{dQ}{d\alpha} = \left\langle \frac{1}{Q} f(S,h_{\underline{J}}) h_{\underline{J}} S + \frac{1}{2Q} f^2(S,h_{\underline{J}})\right\rangle. \tag{5}$$

Here $\langle \ldots \rangle$ denotes the average over the latest input vector, which can be replaced by an average over the corresponding internal fields $h_{\underline{B}}$ and $h_{\underline{J}}$. We have omitted the index μ here and in the following.

Given a specific choice of the weight function, the averages can be performed and the solution of the resulting differential equations yields the evolution of the order parameters. This method has been applied to various learning scenarios, see [8–11] for examples.

The quantity of interest is the *generalization error*, the probability of 'disagreement' $\mathrm{sign}(\underline{J} \cdot \underline{\xi}) \neq S_{\underline{B}}(\underline{\xi})$ for a new input drawn from the same distribution as the training examples. For the simple perceptron it is directly given by the angle between the student and the teacher weight vector [13]:

$$\epsilon_g(\rho) = \frac{1}{\pi} \arccos(\rho). \tag{6}$$

The choice $f(S, h_{\underline{J}}) = 1$ in the learning prescription (2) yields the well–known Hebbian algorithm [12]. In this case the generalization error is known to decay as [13]

$$\epsilon_g = \frac{1}{\sqrt{2\pi}} \; \alpha^{-1/2} \quad \text{for} \quad \alpha \to \infty. \tag{7}$$

Hebbian learning can be considered as a limiting case of the standard perceptron algorithm [6, 7], which corresponds to the weight function $f(S, h_{\underline{J}}) = \Theta(\kappa Q - h_{\underline{J}} S)$. Note that the student norm Q is clearly an observable quantity in the on–line learning process. The Hebb rule is recovered for infinite stability parameter $\kappa \to \infty$. The corresponding differential equations (4, 5) are given in [10] in a different context.

For any $\kappa \geq 0$, R and Q diverge such that $\rho \to 1$ as $\alpha \to \infty$. Numerical results for $\kappa = 0$ suggest the ansatz $\epsilon_g \propto 1/Q$ in this limit, whereas for positive κ the relation $\epsilon_g \propto 1/Q^2$ seems to hold asymptotically. Using these ansätze, one can solve the differential equations selfconsistently for large values of α and gets

$$\epsilon_g(\alpha \to \infty) = \frac{1}{\pi} \left(\frac{2}{3}\right)^{1/3} \alpha^{-1/3} \qquad \text{for } \kappa = 0 \tag{8}$$

$$\epsilon_g(\alpha \to \infty) = \sqrt{\frac{\Phi(\kappa) - \frac{1}{2}}{[1 - \exp(-\kappa^2)]\pi}} \; \alpha^{-1/2} \quad \text{for } \kappa > 0, \tag{9}$$

where $\Phi(x) = \int_{-\infty}^{x} dt \; e^{-t^2/2}/\sqrt{2\pi}$. Figure 1 shows the evolution of the generalization error for various values of κ.

It is important to note that only for $\kappa = 0$ the asymptotic behavior differs from that of the Hebbian learning. For zero stability the student is only updated if she misclassifies the current input example. This scheme is known to give zero training error, eventually, when applied repeatedly (off–line) to a set of linearly separable input/output pairs [6, 7]. It is somewhat surprising to find that the on–line version, however, performs worse than the Hebb algorithm, which does not

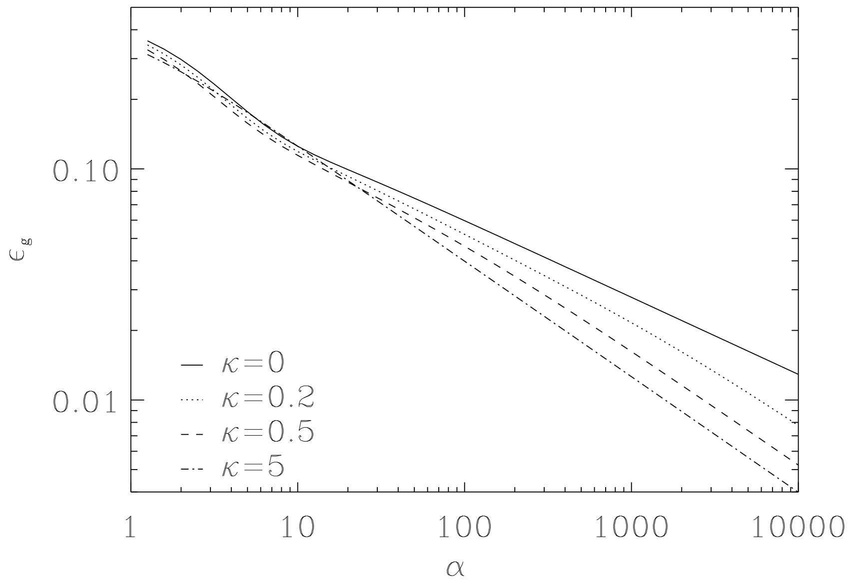

Figure 1: Numerical solution of the generalization error for the perceptron learning rule. Curves are shown for different stabilities κ and initial conditions $R(0) = 0, Q(0) = 1$. The generalization error decays differently on an α–scale for $\kappa = 0$ and $\kappa > 0$. The results for $\kappa = 5$ and Hebbian learning are virtually identical.

have this property. Our result would suggest to use a 'time–dependent' stability $\kappa(\alpha)$ in order to optimize the performance.

Recently, Kinouchi and Caticha [9] proposed an optimal on-line learning process by choosing $f(S, h_{\underline{J}})$ such that it maximizes the expected increase of ρ per learning step. This weight function is given by

$$f_{opt}(S, h_{\underline{J}}) = \frac{Q\sqrt{1-\rho^2}}{\sqrt{2\pi}\,\rho} \; \frac{\exp\left(-\frac{1}{2}\frac{\rho^2}{1-\rho^2}\frac{h_J^2}{Q^2}\right)}{\Phi\left(\frac{\rho}{\sqrt{1-\rho^2}}\frac{h_J S}{Q}\right)}. \tag{10}$$

The authors observe numerically, that the asymptotic generalization error is $\epsilon_g \approx 0.88/\alpha$ for this algorithm. It would clearly outperform Hebbian and perceptron learning. Note, that the weight function depends explicitly on the actual ρ, which –of course– is not available in any realistic learning situation. Therefore, Kinouchi and Caticha suggest in [14] to estimate $\rho(\alpha)$ in the course of the learning process by a 'self-evaluation system'. Such a self-evaluation, however, has to be constructed properly and requires additional computations in the student network besides the updating of its weights $\underline{J}$. Furthermore one might argue, that the use or estimation of the actual generalization error is somewhat opposed to the basic idea behind on–line learning, which is to consider the performance on only the latest example.

However, the result of [9] indicates that it might be possible –even within the restrictions of on–line learning– to achieve the same asymptotic behavior as the optimal off–line scheme. It has been shown [15–17], that for any student $\underline{J}$ which classifies a set of αN random examples correctly, the asymptotic generalization error obeys

$$\frac{0.44}{\alpha} < \epsilon_g(\alpha \to \infty) < \frac{3}{2\alpha}, \tag{11}$$

where the numerical constant in the lower bound is only approximate. We will demonstrate in the following, that a simple, parameter–free algorithm of the form (2) already yields such a $(1/\alpha)$–behavior.

One could try to circumvent the difficulty of not knowing $\rho(\alpha)$ by replacing it in (10) with a constant estimate $\rho_{est} < 1$. This leads to good generalization for intermediate α, but eventually the overlap approaches a value $\rho(\alpha \to \infty) \approx \rho_{est}$ and thus the generalization error remains nonzero (details will be published elsewhere). In the limit $\rho_{est} \to 1$, however, one obtains

$$f(S, h_{\underline{J}}) = -h_{\underline{J}} S \Theta(-h_{\underline{J}} S). \tag{12}$$

This is easily identified as the gradient of the so called AdaTron cost function [18], which would yield the prescription

$$f(S, h_{\underline{J}}) = (\kappa Q - h_{\underline{J}} S)\Theta(\kappa Q - h_{\underline{J}} S) \tag{13}$$

in the more general case of nonzero stability κ. Note, that this is indeed identical with the AdaTron algorithm [19] for the first sweep through a training set. The corresponding differential equation for ρ

$$\begin{aligned} \frac{d\rho}{d\alpha} &= -\frac{\rho}{2\pi}(\kappa^2+1)\arccos\rho - \frac{\rho}{2}\sqrt{\frac{2}{\pi}}\kappa e^{-\frac{\kappa^2}{2}}\Phi\left(\frac{\kappa\rho}{\sqrt{1-\rho^2}}\right) + \\ &\quad + \sqrt{\frac{2}{\pi}}\kappa\Phi\left(\frac{\kappa}{\sqrt{1-\rho^2}}\right) + \frac{1}{\pi}(1-\frac{\rho^2}{2})\sqrt{1-\rho^2}\exp\left(-\frac{1}{2}\frac{\kappa^2}{1-\rho^2}\right) + \\ &\quad - \rho(\kappa^2+1)\int_0^\kappa \frac{dx}{\sqrt{2\pi}}\, e^{-x^2/2}\, \Phi\left(\frac{\rho x}{\sqrt{1-\rho^2}}\right) \end{aligned} \tag{14}$$

turns out to be independent of the norm $Q(\alpha)$. For $\kappa = 0$ the equation simplifies considerably and can be solved asymptotically. The generalization error is exactly given by

$$\epsilon_g = \frac{3}{2\alpha} \qquad \text{for} \quad \alpha \to \infty. \tag{15}$$

We would like to emphasize again that the algorithm (12) requires neither knowledge nor an estimation of the actual generalization error. The $(1/\alpha)$–decay of ϵ_g is realized by a parameter–free algorithm, which at each learning step takes into account the performance on only one example.

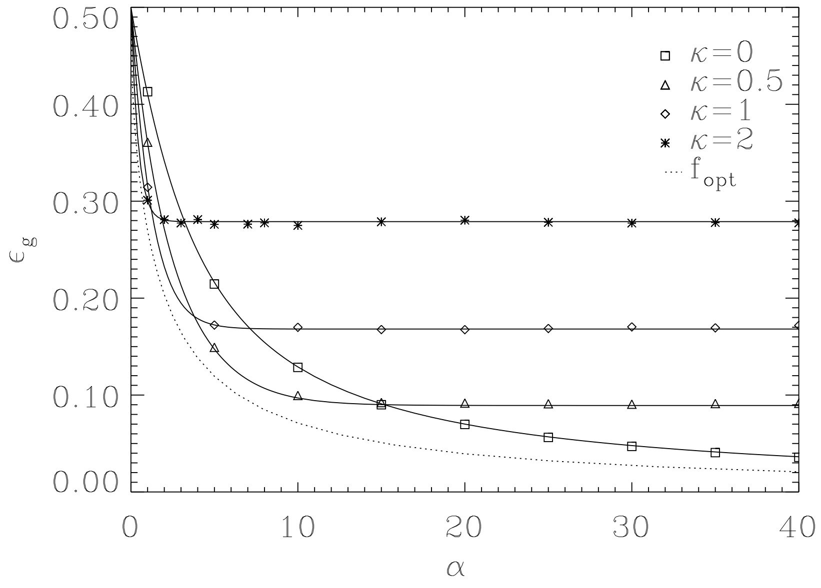

Figure 2: Evolution of the generalization error for the AdaTron learning rule for the same initial conditions as in Fig. 1. The symbols show results obtained by simulations performed for a perceptron of 100 input neurons averaged over 100 independent runs. Standard error bars would be smaller than the size of the symbols. The solid lines represent the numerical solutions of the corresponding differential equation. The dashed line is the numerical solution of the optimal learning process with weight function f_{opt} [9]. Note that the AdaTron learning rule for zero stability performs comparably well.

Fig. 2 shows the generalization error for the optimal weight function and also for the AdaTron algorithm for different stabilities. Simulations confirm our results very well.

We shortly summarize our results for nonzero κ, details will be given in a forthcoming publication. The behavior of the generalization error of the AdaTron algorithm for $\kappa > 0$ is very similar to that of f_{opt} with a fixed $\rho_{est} < 1$. The α–value where ϵ_g becomes a constant is approximately proportional to $1/\kappa$ (see fig. 2). At this value the generalization ability is better than for $\kappa = 0$, indicating that the result for $\kappa = 0$ could be improved by a stability that depends on α and vanishes as $\alpha \to \infty$. A reasonable choice is $\kappa = k/\alpha$. Using the ansatz $\kappa(\alpha) \propto \epsilon_g(\alpha)$ for $\alpha \to \infty$ one can then solve (14) asymptotically, obtaining $\epsilon_g(\alpha \to \infty) = F(k)/\alpha$. For the minimum of $F(k)$ the generalization error decays approximately as $0.97/\alpha$.

A further improvement can be achieved by introducing a learning rate γ in the weight function: $f = \gamma(\frac{k}{\alpha}Q - h_{\underline{J}}S)\Theta(\frac{k}{\alpha}Q - h_{\underline{J}}S)$. An optimization with respect to γ and k yields $\epsilon_g \approx 0.90/\alpha$ which is very close to the result obtained in [9].

Let us conclude with a remark concerning the relevance of our result. Assume that for realizing zero training error with an off–line learning algorithm m sweeps through a set of αN examples were necessary. This would require $m\alpha N$ updates of the form (2) and give at the very best $\epsilon_g \approx 0.44/\alpha$ asymptotically, according to the results of [15, 16]. Yet, only αN updates with on–line–AdaTron (12) yield $\epsilon_g = 3/2\alpha$ already. Clearly, on–line learning is to be preferred with respect to the computational effort, and only if the number of available examples is limited it might pay to employ more sophisticated procedures. It remains an open and important question how these considerations would carry over to on–line training of more complicated, multilayered networks [11].

Acknowledgements
We thank J. Hertz, W. Kinzel, B. Lautrup, M. Opper, H. Schwarze, O. Winther, and in particular G. Reents for useful and stimulating discussions. M. Biehl was supported by the Deutsche Forschungsgemeinschaft and P. Riegler by the Studienstiftung des deutschen Volkes.

References

[1] J.A. Hertz, A. Krogh, and R.G. Palmer, *Introduction to the Theory of Neural Computation*, Addison–Wesley (Redwood City, CA), 1991.

[2] B. Müller and J. Reinhardt, *Neural Networks,* Springer (Berlin), 1990.

[3] H.S. Seung, H. Sompolinsky, and N. Tishby, Phys.Rev. **A45**, 6056 (1992).

[4] T.L.H. Watkin, A. Rau, and M. Biehl, Rev. Mod. Phys. **65**, 499 (1993).

[5] M. Opper and W. Kinzel, *Statistical Mechanics of Generalization*, to appear in *Physics of Neural Networks III,* series eds. E. Domany, J.L. van Hemmen, and K. Schulten, to be published by Springer (Berlin).

[6] F. Rosenblatt, *Principles of Neurodynamics*, Spartan (New York), 1962.

[7] M.L. Minsky and S. Papert, *Perceptrons*, MIT Press (Cambridge, Mass.), 1969 and 1988.

[8] W. Kinzel and P. Rujan, Europhys.Lett. **13**, 473 (1990).

[9] O. Kinouchi and N. Caticha, J.Phys. **A25**, 6243 (1992).

[10] M. Biehl and H. Schwarze, J.Phys. **A26**, 2651 (1993).

[11] M. Biehl and H. Schwarze, *Learning by On–line Gradient Descent*, Lund University preprint LU TP 94–10, 1994.

[12] D.O. Hebb, *The Organization of Behavior*, Wiley (New York), 1949.

[13] F. Vallet, Europhys.Lett. **9**, 315 (1989).

[14] O. Kinouchi and N. Caticha, J.Phys. **A26**, 6161 (1993).

[15] M. Opper and D. Haussler, Phys.Rev.Lett. **66**, 2677 (1991).

[16] T.L.H. Watkin, Europhys.Lett. **21**, 871 (1993).

[17] A. Engel and C. van den Broeck, Phys.Rev.Lett. **71**, 1772 (1993).
A. Engel and W. Fink, J.Phys. **A26**, 6893 (1993).

[18] M. Griniasty and H. Gutfreund, J.Phys. **A24**, 715 (1991).

[19] J.K. Anlauf and M. Biehl, Europhys.Lett. **10**, 687 (1990).