Learning drifting concepts with neural networks

1993 J. Phys. A: Math. Gen. 26 2651

(http://iopscience.iop.org/0305-4470/26/11/014)

J. Phys. A: Math. Gen. 26 (1993) 2651–2665. Printed in the UK

Learning drifting concepts with neural networks

Michael Biehl† and Holm Schwarze‡

† Physikalisches Institut, Julius-Maximilians-Universität, Am Hubland, D-8700 Würzburg, Federal of Republic Germany

‡ CONNECT, The Niels Bohr Institute, Blegdamsvej 17, DK-2100 Copenhagen Ø, Denmark

Received 27 November 1992

Abstract. The learning of time-dependent concepts with a neural network is studied analytically and numerically. The linearly separable target rule is represented by an N-vector, whose time dependence is modelled by a random or deterministic drift process. A single-layer network is trained online using different Hebb-like algorithms. Training is based on examples which are chosen randomly and according to a query strategy. The evolution of the generalization error can be calculated exactly in the thermodynamic limit $N \to \infty$. The rule is never learnt perfectly, but can be tracked within a certain error margin. The generalization performance of various learning rules is compared and simulations confirm the analytic results.

1. Introduction

A feedforward neural network [1] can serve as a tool for *classification*: it assigns an output to any possible input configuration. Learning is the adaption of the network parameters aiming at the realization of a specific input–output relation. Typically, this relation is defined by an unknown classification scheme or *rule*, and the correct output is available only for certain example inputs. The extraction of the unknown concept by learning from these training examples is called *generalization*.

So far the investigation of this attractive feature of neural networks has focused, in particular, on the learning of a fixed, given rule (e.g. [2, 3]). Principally for the simplest feedforward system, the single-layer perceptron [4, 5], several sophisticated training procedures have been studied, such as stochastic minimization of proper cost functions [2, 3] or deterministic iterative algorithms (e.g. [6–9]). These strategies would usually require the explicit storage of examples in a separate device, and the computational effort per example can be relatively high.

In this paper we address the more general problem of a network in a time-varying environment [10–13], where the target rule changes during the process of learning. A natural approach to this kind of situation is called *online* or *real-time learning* [1]. Assume a sequence of novel input–output pairs is provided, each of which is used only once to adjust the parameters in the learning network (or *student*, for short). Its performance on previous examples is not taken into account by the learning procedure. Thus, the explicit storage of examples is not necessary.

An illustrative example of such a situation is the sequence of items passing by on an assembly line, which have to be classified as 'okay' or 'defective'. Some of them have already been marked correctly by a *teacher* according to the quality requirements, which change in time. The student network is supposed to generalize, that is to classify the unmarked products in good agreement with the current rule.

0305-4470/93/112651+15$07.50

In the following we present an exactly solvable model for the online learning of drifting concepts. We consider a single-layer perceptron and various training algorithms. In section 2 we describe this type of network, the training procedure, the target rule and two possible time dependences thereof. Section 3 deals with the first dependence, a random drift in rule space. We outline the solution in general and apply it to the simple Hebb rule, an algorithm optimal with respect to the achieved generalization, and the so-called perceptron algorithm. Furthermore, learning from *queries* rather than from randomly chosen examples is considered. The results for the Hebb rule have recently been reported in [13], but are included here for completeness. In the fourth section the corresponding analysis for a deterministically changing rule is given and the results are presented. The last section gives a brief summary and discussion.

2. The model

We study an exactly solvable model for the learning of a time-dependent rule with a single-layer perceptron. The network is represented by an N-dimensional weight vector $\boldsymbol{J} \in \mathbb{R}^N$, and for each input vector $\boldsymbol{\xi} \in \mathbb{R}^N$ its output is given by

$$S_J(\boldsymbol{\xi}) = \operatorname{sign}\left(\sum_{j=1}^{N} J_j \xi_j\right). \tag{1}$$

The perceptron is trained to implement a linearly separable target rule

$$S_B(\boldsymbol{\xi}) = \operatorname{sign}\left(\sum_{j=1}^{N} B_j \xi_j\right) \tag{2}$$

where $\boldsymbol{B} \in \mathbb{R}^N$ is normalized to 1. The training process is based on examples consisting of input–output pairs $(\boldsymbol{\xi}^\mu, S_\mu = S_B(\boldsymbol{\xi}^\mu))$, $\mu = 1, \ldots, p$ of the target rule. Instead of explicitly storing the entire set of training examples we study a class of Hebb-like algorithms [14, 9] which use each example only once. After the presentation of the μth example the student vector is updated according to the rule

$$J_i^{\mu+1} = \left(1 - \frac{\lambda}{N}\right) J_i^\mu + \frac{1}{N} f(S_\mu, h_J^\mu) \xi_i^\mu S_\mu. \tag{3}$$

Here h_J^μ is the internal field in the student network, $h_J^\mu = \boldsymbol{J}^\mu \cdot \boldsymbol{\xi}^\mu$. Each training example is weighted by a function $f(S_\mu, h_J^\mu)$. An additional weight decay term $-\lambda J_i^\mu / N$ reduces the norm of the student vector at each learning step [1, 15, 16]. Every choice of f and the parameter λ defines a specific algorithm. They allow us to introduce the effect of 'forgetting' as will be explained later. Note that the components of both $\boldsymbol{J}$ and $\boldsymbol{B}$ are of order $\mathcal{O}(1/\sqrt{N})$.

The quality of the network performance may be measured by the generalization error: the probability that a new, randomly drawn input is misclassified by the student. By geometrical arguments (e.g. [3]) the generalization error of a simple perceptron is determined by the angle between the student and the teacher vector. It is given by the expression

$$\epsilon_g(\rho) = \frac{1}{\pi} \cos^{-1} \rho \tag{4}$$

where $\rho = R/Q$ is the overlap of the two weight vectors, $R = \boldsymbol{J} \cdot \boldsymbol{B}$, normalized by the length of the student vector, $Q = \sqrt{\boldsymbol{J} \cdot \boldsymbol{J}}$. An independently drawn, random $\boldsymbol{J}$ gives $\rho = 0$ and thus $\epsilon_g = 1/2$, which corresponds to randomly 'guessing' the output.

Our specific goal is to calculate the generalization error for a linearly separable but time-dependent target function S_B. The time dependence may be modelled by a teacher vector which changes after each learning step. We study two different types of such drift processes. First we consider a weight vector which performs a random walk on the N-dimensional unit sphere. As an example of a deterministic drift, we study a teacher which seeks to minimize its overlap with the student vector.

3. Learning a randomly varying rule

We first consider the learning of a rule, which changes according to a stochastic drift process. After each presentation of a training example μ the weight vector of the target rule changes randomly subject to the conditions

$$\boldsymbol{B}^{\mu+1} \cdot \boldsymbol{B}^{\mu} = 1 - \frac{\eta}{N} \qquad \text{and} \qquad \boldsymbol{B}^{\mu+1} \cdot \boldsymbol{B}^{\mu+1} = 1. \tag{5}$$

The drift parameter η determines the angle between two consecutive teacher vectors, which for large N is given by $\sqrt{2\eta/N}$. It relates the time scales of the drift process and the presentation of training examples. For binary weights, $B_i = \pm 1/\sqrt{N}$, condition (5) means that at each time step a finite number $\eta/2$ of randomly chosen bits is flipped. For non-integral values of $\eta/2$ this condition can only be satisfied on average, but in the thermodynamic limit ($N \to \infty$) it is self-averaging. One may think of the drift process as a fixed sequence of teacher vectors randomly drawn in advance, independently of the learning process.

Using (5) and the learning rule (3) we obtain difference equations for the overlaps $R^{\mu} = \boldsymbol{B}^{\mu} \cdot \boldsymbol{J}^{\mu}$ and $(Q^{\mu})^2 = \boldsymbol{J}^{\mu} \cdot \boldsymbol{J}^{\mu}$, to order $1/N$ given by

$$\begin{aligned} R^{\mu+1} - R^{\mu} &= \frac{1}{N}[f(S_{\mu}, h_J^{\mu}) h_B^{\mu} S_{\mu} - (\lambda + \eta) R^{\mu}] \\ (Q^{\mu+1})^2 - (Q^{\mu})^2 &= \frac{2}{N}[f(S_{\mu}, h_J^{\mu}) h_J^{\mu} S_{\mu} + \tfrac{1}{2} f^2(S_{\mu}, h_J^{\mu}) - \lambda (Q^{\mu})^2]. \end{aligned} \tag{6}$$

These equations have to be averaged over the distribution of training examples. The dependence on the training inputs is only through the internal fields $h_B^{\mu} = \boldsymbol{B}^{\mu} \cdot \boldsymbol{\xi}^{\mu}$ and $h_J^{\mu} = \boldsymbol{J}^{\mu} \cdot \boldsymbol{\xi}^{\mu}$ which, for large N, are correlated Gaussian variables. If the components of the training inputs $\boldsymbol{\xi}^{\mu}$ are drawn independently from a distribution with zero mean and unit variance, the joint distribution of h_B^{μ} and h_J^{μ} is given by

$$P(h_B, h_J) = \frac{1}{2\pi} \frac{1}{\sqrt{Q^2 - R^2}} \exp\left(-\frac{1}{2} \frac{(h_J)^2 + Q^2 (h_B)^2 - 2R h_J h_B}{Q^2 - R^2} \right). \tag{7}$$

Here and in the following we suppress the index μ. In the thermodynamic limit ($N \to \infty$), we may furthermore assume that R and Q^2 are self-averaging with respect to the averages over the training examples and different random walks [9, 21]. We therefore replace them with their average values.

In the same limit, (6) can be written as differential equations, if we introduce the continuous 'time' $\alpha = \mu/N$, the number of training examples per weight learnt so far.

Note that this is only possible for the scaling of the drift parameter given in (5). If, for instance, the angle between two consecutive teacher vectors was of order 1, the drift term would dominate the behaviour of (6); no learning would be possible. If, on the other hand, the angle was of higher order in $1/N$, the drift would have no effect. Despite the $1/N$ scaling of the drift parameter, learning the considered rule is not an easy task, because the average overlap of $\boldsymbol{B}$ with the initial vector decays exponentially fast on an α-scale, $\langle \boldsymbol{B}(\alpha) \cdot \boldsymbol{B}(0) \rangle \propto \mathrm{e}^{-2\alpha\eta}$.

We now get coupled differential equations for R and Q^2 as functions of α

$$\frac{\mathrm{d}R}{\mathrm{d}\alpha} = \langle f(S, h_J) h_B S \rangle - (\lambda + \eta) R(\alpha)$$

$$\frac{\mathrm{d}[Q^2]}{\mathrm{d}\alpha} = 2\langle f(S, h_J) h_J S + \tfrac{1}{2} f^2(S, h_J) \rangle - 2\lambda Q^2. \tag{8}$$

Here $\langle \ldots \rangle$ denotes the average over the distribution (7). Instead of R one can also use the differential equation for the normalized overlap ρ, given by

$$\frac{\mathrm{d}\rho}{\mathrm{d}\alpha} = \left\langle \frac{f(S, h_J) S}{Q} \left(h_B - \frac{\rho}{Q} h_J \right) - \frac{1}{2} \frac{\rho}{Q^2} f^2(S, h_J) \right\rangle - \eta\rho. \tag{9}$$

For specific learning rules, defined by different choices of the weight function f and the parameter λ, the average over (7) can be performed. The solution of the resulting differential equations finally yields the evolution of the generalization error and its asymptotic behaviour. This will be described in the following sections.

3.1. The Hebb rule

For the choice $f(h_J, S) = 1$ the general learning algorithm (3) reduces to the simple Hebb rule. In this case the averages in (8) can easily be performed yielding

$$\frac{\mathrm{d}R}{\mathrm{d}\alpha} = \sqrt{\frac{2}{\pi}} - (\lambda + \eta) R \qquad \frac{\mathrm{d}[Q^2]}{\mathrm{d}\alpha} = 2\sqrt{\frac{2}{\pi}} R + 1 - 2\lambda Q^2. \tag{10}$$

For the Hebb rule without weight decay ($\lambda = 0$) the solution of (10) is simply given by

$$R(\alpha) = \sqrt{\frac{2}{\pi}} \frac{1}{\eta} (1 - \mathrm{e}^{-\eta\alpha})$$

$$Q^2(\alpha) = \left(1 + \frac{4}{\pi\eta} \right) \alpha + \frac{4}{\pi\eta^2} \mathrm{e}^{-\eta\alpha} + \left(1 - \frac{4}{\pi\eta^2} \right) \tag{11}$$

if we assume an initially normalized student vector ($Q(0) = 1$), which is uncorrelated to the teacher ($R(0) = 0$). For small values of α, the corresponding generalization error $\epsilon_g(\alpha)$ decreases as shown in figure 1. However, if the drift parameter η does not vanish, ϵ_g passes through a minimum and its asymptotic increase is given by $\epsilon_g \approx 1/2 - \mathcal{O}(1/\sqrt{\alpha})$. Hence, for long training times the network fails to generalize and randomly 'guesses' the output. Note that the asymptotic behaviour does not depend on the specific choice of the initial

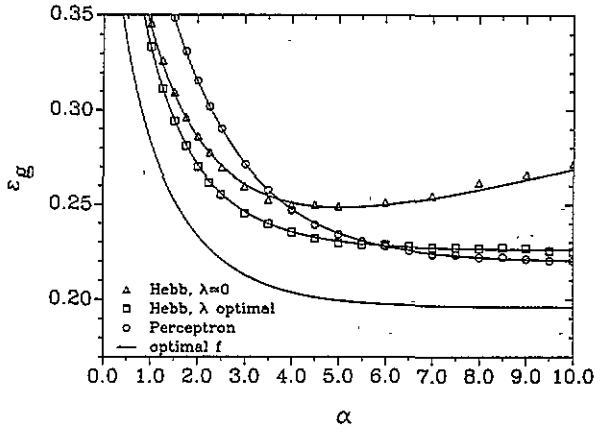

Figure 1. Evolution of the generalization error for the learning of a randomly drifting concept with $\eta = 0.1$ from independently drawn examples. Note the offset on the ϵ_g axis. The learning curves are shown for the Hebb rule with $\lambda = 0$ and with optimal weight decay respectively, for the simple perceptron algorithm ($\kappa = \lambda = 0$), and for learning with an optimal weight function f. Except for the latter, simulations were performed for a system of $N = 200$ input neurons. Averages were calculated over 200 independent runs and the standard error bars would be approximately the size of the symbols.

conditions. Even a student vector, which initially is perfectly aligned with the teacher, eventually loses all correlations ($\rho_\infty = 0$).

The reason for this failure is that the Hebb rule weights all the training examples with the same factor. It overemphasizes the old examples which contain little information about the current rule. As a result the norm of the student vector in (11) is unbounded for large α, while the overlap R to the teacher remains finite.

More efficient learning rules adjust the attention paid to different examples. As described in the following sections, this may be done by a non-trivial choice of the weight function f. Here we study the effect of a simple weight decay $\lambda > 0$ on the generalization ability of the Hebb rule. For non-vanishing λ, the solution of (10) remains finite for all α and the normalized overlap ρ shows an exponential approach to a stationary non-vanishing value, given by

$$\rho_\infty = 2\sqrt{\lambda}\sqrt{(\eta+\lambda)[\pi(\eta+\lambda)+4]}. \tag{12}$$

The stationary solution may be regarded as the 'working phase' of the network, which describes its performance after an initial training process. Therefore we are particularly interested in optimizing the network's generalization ability for $\alpha \to \infty$. By adjusting the weight decay parameter λ for a given drift η, we can maximize the asymptotic overlap (12) and obtain

$$\rho_\infty^{\text{opt}} = \frac{2}{\sqrt{\pi\eta}+\sqrt{\eta\pi+4}} \qquad \text{for} \qquad \lambda_{\text{opt}} = \sqrt{\eta\left(\frac{4}{\pi}+\eta\right)}. \tag{13}$$

Figure 1 shows the evolution of the generalization error for a given value of η using the optimal weight decay λ_{opt}. Simulations were performed for $N = 200$ and they confirm the results very well. In figure 2 the asymptotic generalization error for the optimal λ is

plotted against the drift parameter η. For a very slowly drifting rule the η-dependence of the asymptotic error is given by

$$\epsilon_g^{\mathrm{opt}}(\alpha \to \infty) \approx \eta^{1/4}/\pi^{3/4}. \tag{14}$$

Hence, employing weight decay in the Hebb rule dramatically improves the generalization ability for a time-dependent rule. Instead of an algebraic decay of the generalization ability, we find an exponential approach to a working phase with a finite overlap with the teacher. The weight decay isotropically reduces the length of the student vector before each learning step and weights the information about more recent locations of the teacher vector more strongly. Learning rules such as (3) have already been studied in the context of attractor networks working as 'forgetful memories' storing recently embedded patterns (e.g. [15, 17–19]). Here, forgetting unreliable old information allows the teacher to be followed to a certain degree, although a complete approach and therefore perfect generalization cannot be achieved due to the drift process.

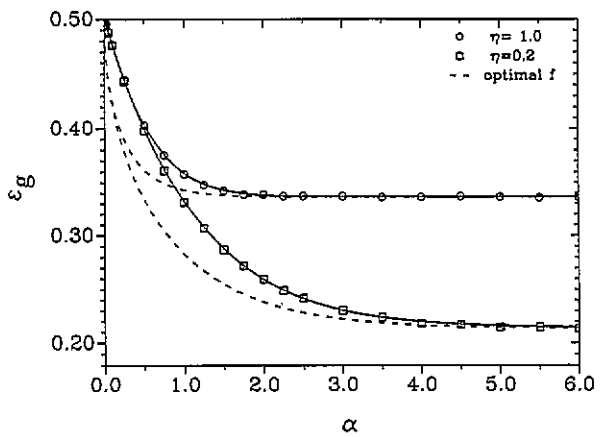

Figure 2. The asymptotic generalization error for the learning of a randomly drifting concept as a function of the drift parameter. The upper four curves show the dependence in the case of independently drawn examples for the Hebb rule with optimal weight decay, simple perceptron learning, perceptron learning with optimized parameters κ and λ, and for the use of the optimal weight function. The full curve corresponds to the learning from queries.

3.2. The optimal weight function f

In the previous section it has been shown that the generalization performance of the simple Hebb rule can be greatly improved by suppressing the influence of old training examples using weight decay. A more general way of adjusting the attention payed to different examples is to introduce a non-trivial weight function $f(S, h_J)$ in (3). An additional weight decay could formally be incorporated into the definition of f as an explicit dependence upon α [15]. Therefore, we will restrict the following discussion to the case $\lambda = 0$. Each choice of f defines a different learning rule, and one would like to find a form, which yields the best possible generalization ability. Recently, Kinouchi and Caticha [9] considered an optimal choice of f, which maximizes the decrease in generalization error per example or,

equivalently, maximizes $\mathrm{d}\rho/\mathrm{d}\alpha$. Taking the functional derivative of (9) one obtains for the optimal weight function

$$f^*_{\mathrm{opt}} = S\left(h_B \frac{Q}{\rho} - h_J\right). \tag{15}$$

This form still depends on the internal fields h_B in the teacher network, which are not observable during training except for their signs S. Therefore, Kinouchi and Caticha [9] suggest averaging f^*_{opt} over the distribution

$$P(h_B|S,h_J) = \frac{P(h_B,h_J)\Theta(Sh_B)}{\int_{-\infty}^{+\infty} \mathrm{d}h_B\, P(h_B,h_J)\Theta(Sh_B)} \tag{16}$$

of h_B for given S and h_J, yielding

$$f_{\mathrm{opt}} = \frac{Q\sqrt{1-\rho^2}}{\sqrt{2\pi}\rho} \exp\left[-\frac{1}{2}\frac{\rho^2}{1-\rho^2}\frac{h_J^2}{Q^2}\right] \Big/ \Phi\left(\frac{\rho}{\sqrt{1-\rho^2}}\frac{h_J S}{Q}\right) \tag{17}$$

where $\Phi(x) = \int_{-x}^{\infty} Dt$ and $Dt = \mathrm{e}^{-t^2/2}\mathrm{d}t/\sqrt{2\pi}$. Note that f_{opt} does not explicitly depend on the drift parameter η. Using this weight function in (9) and performing the average over h_J yields a differential equation for ρ

$$\frac{\mathrm{d}\rho}{\mathrm{d}\alpha} = \frac{1}{2\pi}\frac{(1-\rho^2)^{3/2}}{\rho} \int_{-\infty}^{+\infty} \frac{\mathrm{d}x}{\sqrt{2\pi}} \frac{\exp[-\frac{1}{2}(1+\rho^2)x^2]}{\Phi(\rho x)} - \eta\rho. \tag{18}$$

This equation can be solved numerically, and the corresponding generalization error as a function of α is shown in figure 1. From the stationary solution of (18) the asymptotic generalization error $\epsilon_g(\alpha \to \infty)$ can be obtained (figure 3). Its small-η behaviour is given by

$$\epsilon_g(\alpha \to \infty) \approx \left(\frac{2}{\pi^2 A}\eta\right)^{1/3} \approx 0.45\eta^{1/3} \qquad \text{where} \qquad A = \int_{-\infty}^{\infty} \frac{\mathrm{d}x}{\sqrt{2\pi}} \mathrm{e}^{-x^2}/\Phi(x) \tag{19}$$

which is a considerable improvement compared with the Hebb rule. However, in order to utilize the optimal weight function as given in (17), one needed to know ρ, the true overlap with the teacher. Since the teacher is unknown, this quantity could only be estimated from monitoring the frequency of errors during training. Nevertheless, the optimal learning rule provides a lower bound for the generalization error, which can be achieved with an algorithm of the form (3).

3.3. The perceptron learning rule

As an example of a non-trivial weight function f we consider the perceptron learning rule [4, 5], which can be obtained from (3) by the choice $f(S,h_J) = \Theta(\kappa - h_J S/Q)$. κ is called

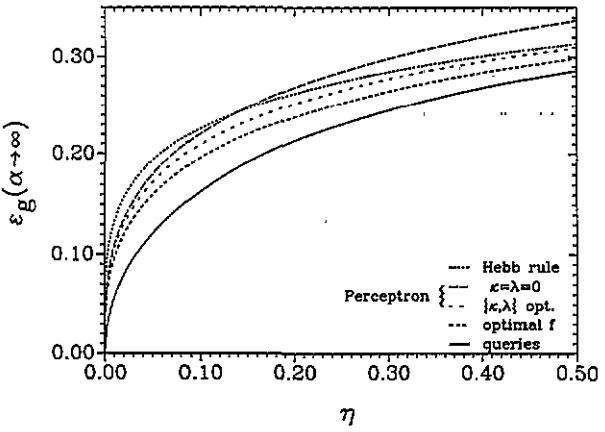

Figure 3. Learning from queries for two values of η. The full curves correspond to the Hebb rule with optimal weight decay, the broken curves show the result for the optimal f procedure. Both approach the same asymptotic generalization error. (Simulations as in figure 1.).

the stability parameter. After a straightforward calculation the average over (7) yields the following coupled differential equations for ρ and Q

$$\frac{\mathrm{d}\rho}{\mathrm{d}\alpha} = \sqrt{\frac{2}{\pi}}\frac{1-\rho^2}{Q}\Phi\left(\frac{\kappa}{\sqrt{1-\rho^2}}\right) - \frac{1}{2\pi}\frac{\rho}{Q^2}\cos^{-1}\rho - \frac{\rho}{Q^2}\int_0^\kappa Dz\Phi\left(\frac{\rho z}{\sqrt{1-\rho^2}}\right) - \eta\rho$$

$$\frac{\mathrm{d}[Q^2]}{\mathrm{d}\alpha} = 2\sqrt{\frac{2}{\pi}}\rho Q\Phi\left(\frac{\kappa}{\sqrt{1-\rho^2}}\right) + \frac{1}{\pi}\cos^{-1}\rho + 2\int_0^\kappa Dz\Phi\left(\frac{\rho z}{\sqrt{1-\rho^2}}\right)$$
$$- 2\sqrt{\frac{2}{\pi}}Q\Phi\left(\frac{\kappa\rho}{\sqrt{1-\rho^2}}\right)\mathrm{e}^{-\kappa^2/2} - 2\lambda Q^2. \tag{20}$$

We first consider the case of zero stability without weight decay to obtain a simple learning prescription without externally adjustable parameters. In this case, (20) reduces to

$$\frac{\mathrm{d}\rho}{\mathrm{d}\alpha} = \frac{1}{\sqrt{2\pi}}\frac{1-\rho^2}{Q} - \frac{1}{2\pi}\frac{\rho}{Q^2}\cos^{-1}\rho - \eta\rho \tag{21}$$

$$\frac{\mathrm{d}[Q^2]}{\mathrm{d}\alpha} = \frac{1}{\pi}\cos^{-1}\rho - \sqrt{\frac{2}{\pi}}Q(1-\rho).$$

From the numerical solution of these equations we obtain the generalization error (figure 1) and its stationary value $\epsilon_g(\alpha\to\infty)$ as shown in figure 2. It is remarkable that for small η the perceptron rule has the same η-dependence as learning with an optimal weight function, except for a slightly greater prefactor

$$\epsilon_g(\alpha\to\infty) \approx (4\eta)^{1/3}/\pi \approx 0.51\eta^{1/3}. \tag{22}$$

For larger drift parameters, on the other hand, this simple form of perceptron learning performs increasingly worse, even compared with the Hebb rule with an optimized weight decay. However, in contrast to both the Hebb rule with optimized weight decay and optimal

learning, the perceptron learning rule does not require the setting of external parameters. Using the optimal weight decay (13) in the Hebb rule requires knowledge of the drift parameter. Alternatively, one could monitor the frequency of errors and adjust λ online so as to minimize the observed frequency. The perceptron learning rule works without online adjustments or an *a priori* knowledge about the drift process, its simple form yields good results for both a stationary teacher and a slowly drifting rule.

If we include a non-vanishing stability $\kappa \neq 0$ and weight decay $\lambda > 0$, we can numerically optimize the asymptotic behaviour with respect to these parameters. From figure 2 we see that this additional effort results in an improvement of the generalization error particularly for large η.

3.4. Learning from queries

So far we have only considered the situation where the training examples are drawn from a given (uniform) distribution. More generally one might allow this distribution to vary in time as well [10].

One special form of such a time dependence is the generation of training inputs, which carry—together with the teacher output—much information. As an illustration one might think of a student who asks *intelligent questions* in order to speed up learning. Several such *query strategies* have been studied in the context of neural network learning [20–23]. Clearly, the choice of a new example should take into account the current properties of the student network. For the simple perceptron, Kinzel and Rujan [21] suggest choosing training inputs perpendicular to the actual student vector $\boldsymbol{J}$. The corresponding output is completely uncertain for the learning network, and therefore the correct answer should provide much information about how to change the weights.

Following this idea we consider examples drawn randomly, but subject to the condition $h_J = 0$. This restriction can easily be incorporated into the analysis outlined above. Of course the weight function f in the learning algorithm (3) does not depend on h_J any more. Note that, for instance, the perceptron algorithm as defined in section 3.3 reduces to simple Hebbian learning in the query scheme. Averages in equation (8) are now to be performed over the probability distribution

$$P_{\text{query}}(h_B) = \frac{P(h_B, 0)}{\int \mathrm{d}h_B\, P(h_B, 0)} = \frac{1}{\sqrt{2\pi(1-\rho^2)}} \exp\left[-\frac{(h_B)^2}{2(1-\rho^2)}\right] \tag{23}$$

with $P(h_B, h_J)$ from equation (7).

3.4.1. The Hebb rule. We consider the simple Hebb rule first. After inserting $h_J = 0$ and $f = 1$ in equation (3) and averaging over P_{query} one obtains the differential equations

$$\frac{\mathrm{d}R}{\mathrm{d}\alpha} = \sqrt{\frac{2}{\pi}}\sqrt{1-\frac{R^2}{Q^2}} - (\lambda+\eta)R \qquad \frac{\mathrm{d}[Q^2]}{\mathrm{d}\alpha} = 1 - 2\lambda Q^2 \tag{24}$$

which have to be solved under the initial conditions $R(0) = 0$ and $Q(0) = 1$.

Again, for non-zero drift without weight decay, the residual error is $\epsilon_g(\alpha \to \infty) = 1/2$, due to the assignment of an equal weight to all the examples and the unboundedness of $Q^2(\alpha) = 1 + \alpha$. For positive λ the solution of the second equation is

$$Q^2(\alpha) = \frac{1}{2\lambda} + \left(1 - \frac{1}{2\lambda}\right)\exp[-2\lambda\alpha] \tag{25}$$

and the remaining equation for R can be solved numerically. The asymptotic normalized overlap is calculated exactly from the condition $\mathrm{d}R/\mathrm{d}\alpha(\alpha \to \infty) = 0$ using $Q^2 \to 1/2\lambda$. The result is

$$\rho_\infty = 2\sqrt{\frac{\lambda}{\pi}} \Big/ \sqrt{(\lambda+\eta)^2 + \frac{4\lambda}{\pi}} > 0 \tag{26}$$

which is maximal for

$$\lambda_{\mathrm{opt}} = \eta \qquad \text{with} \qquad \rho_\infty^{\mathrm{opt}} = \frac{1}{\sqrt{1+\eta\pi}}. \tag{27}$$

The residual error for very slowly drifting rules ($\eta \to 0$) is given by

$$\epsilon_{\mathrm{g}}^{\mathrm{opt}}(\alpha \to \infty) \approx \sqrt{\eta/\pi}. \tag{28}$$

Obviously the query procedure improves generalization compared with learning from random examples. For small values of η this advantage is particularly pronounced. In figure 2 $\epsilon_{\mathrm{g}}^{\mathrm{opt}}(\alpha \to \infty)$ is plotted against the drift parameter, figure 3 shows the evolution of the generalization error for two different values of η, together with the results of simulations for a system of 200 neurons.

3.4.2. Optimal f. The differential equations for $Q^2(\alpha)$ and $\rho(\alpha)$, when learning from queries with a non-trivial weight function f and zero weight decay, read

$$\frac{\mathrm{d}Q^2}{\mathrm{d}\alpha} = \langle f^2 \rangle \qquad \frac{\mathrm{d}\rho}{\mathrm{d}\alpha} = \left\langle \frac{f|h_B|}{Q} - \frac{f^2\rho}{2Q^2} \right\rangle - \eta\rho \tag{29}$$

with the averages to be taken over $P_{\mathrm{query}}(h_B)$.

Again, the calculation of the function f which maximizes $\mathrm{d}\rho/\mathrm{d}\alpha$ is the same as in the work of Kinouchi and Caticha [9] for the learning of a static rule. The result is

$$f_{\mathrm{opt}} = \sqrt{\frac{2}{\pi}} Q \frac{\sqrt{1-\rho^2}}{\rho}. \tag{30}$$

Inserting this weight function in equations (29) and performing the averages gives

$$\frac{\mathrm{d}\rho}{\mathrm{d}\alpha} = \frac{1}{\pi}\frac{1-\rho^2}{\rho} - \eta\rho \tag{31}$$

independent of Q. It is solved by

$$\rho(\alpha) = \sqrt{\frac{1-\exp[-2\alpha(\eta+1/\pi)]}{1+\eta\pi}}. \tag{32}$$

The student vector approaches the teacher exponentially fast (note that this still holds for $\eta = 0$ as pointed out in [9]). See figure 3 for a comparison with optimized Hebbian learning. However, both algorithms approach the same stationary value resulting in an identical asymptotic generalization error (28). Hence, the Hebb rule is already optimal with respect to the behaviour during the working phase.

4. Learning a deterministically varying rule

In this section we study how deterministic changes in the target rule alter the above considerations. The worst drift process possible in our model is one where at each time step the target vector 'moves away' from the current student vector as far as possible. Clearly, any other drift, that produces the same angle between consecutive teacher vectors (see below), would be easier to track for the student. In this sense, the following results provide lower bounds on the generalization error that can be achieved by an algorithm of type (3).

At time step μ we choose the new target vector such that it minimizes the overlap $\boldsymbol{B}^{\mu+1} \cdot \boldsymbol{J}^{\mu}$ subject to the conditions

$$\boldsymbol{B}^{\mu+1} \cdot \boldsymbol{B}^{\mu} = 1 - \tilde{\delta} \qquad \text{and} \qquad \boldsymbol{B}^{\mu+1} \cdot \boldsymbol{B}^{\mu+1} = 1. \tag{33}$$

Here $\tilde{\delta}$ determines the size of the steps the teacher is allowed to perform. The solution to this problem is the linear combination

$$\boldsymbol{B}^{\mu+1} = a\boldsymbol{B}^{\mu} - b\boldsymbol{J}^{\mu} \tag{34}$$

where

$$b = \sqrt{\frac{2\tilde{\delta} - \tilde{\delta}^2}{(Q^{\mu})^2 - (R^{\mu})^2}} \qquad \text{and} \qquad a = 1 - \tilde{\delta} + bR^{\mu}. \tag{35}$$

Inserting this vector $\boldsymbol{B}^{\mu+1}$ in equation (6), we see that $\tilde{\delta} = \delta/N^2$ is the correct scaling by the same argument as in section 3.

Obviously the drift parameter must be a factor of $\mathcal{O}(1/N)$ smaller than in the previous case, the angle between $\boldsymbol{B}^{\mu}$ and its successor is only $\sqrt{2\delta}/N$ for large N here. This is due to the fact that steps towards and away from the student were equally probable in the random walk. Another difference is that the above sequence of vectors $\boldsymbol{B}$ cannot be generated 'in advance', it depends explicitly on the actual evolution of the student. One might think of $\boldsymbol{B}$ being chosen by an *adversary* [12] at each time step, obeying the restrictions (33).

In the limit $N \to \infty$ we obtain the following differential equations

$$\frac{\mathrm{d}R}{\mathrm{d}\alpha} = \langle f|h_B|\rangle - \lambda R - \sqrt{2\delta(Q^2 - R^2)} \tag{36}$$

$$\frac{\mathrm{d}[Q^2]}{\mathrm{d}\alpha} = 2\langle f h_J S + \tfrac{1}{2} f^2\rangle - 2\lambda Q^2 \tag{37}$$

or, alternatively,

$$\frac{\mathrm{d}\rho}{\mathrm{d}\alpha} = \left\langle \frac{fS}{Q}\left(h_B - \frac{\rho h_J}{Q}\right) - \frac{1}{2}\frac{\rho f^2}{Q^2}\right\rangle - \sqrt{2\delta(1-\rho^2)}. \tag{38}$$

Only the terms directly connected to the drift have changed compared with equations (8) and (9). Thus the analysis proceeds completely analogously to the procedure in the previous sections and we summarize and discuss the results in the following. In figure 4 the evolution of $\epsilon_g(\alpha)$ is shown for one specific value of the drift parameter, simulations confirm the results. Figure 5 shows the residual error against δ for the different learning schemes.

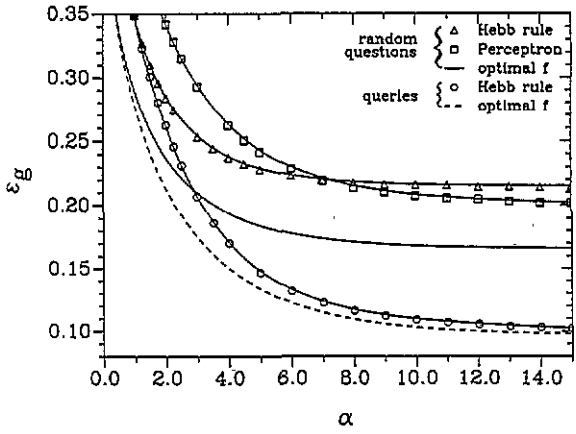

Figure 4. Evolution of the generalization error for the learning of a deterministically drifting concept according to the worst case prescription with $\delta = 0.005$. The upper curves correspond to training with independently drawn examples, according to the Hebb algorithm with an optimal value of λ, the simple perceptron procedure and the optimal choice of the weight function f. The lower two curves show the generalization error of the optimized Hebb rule and the optimal f procedure when learning from queries. (Simulations as in figure 1.).

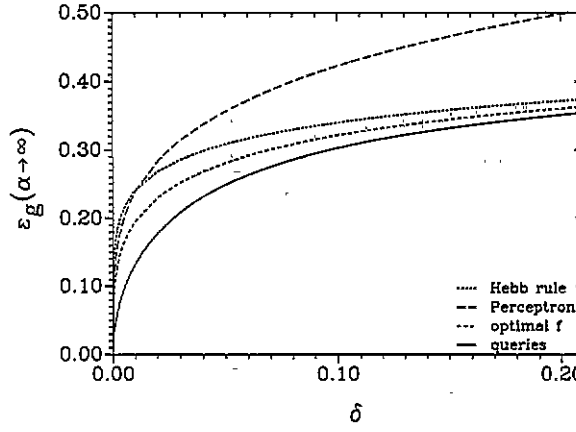

Figure 5. Dependence of the asymptotic generalization error on the parameter δ in the worst case drift. The curves correspond to simple perceptron learning, the Hebb rule with optimal weight decay, and the procedure with the optimal weight function. The full curve is for the learning from queries.

4.1. Random examples

4.1.1. The Hebb rule. Inserting $f = 1$ in equations (36)–(38) and performing the averages over the distribution (7), one finds that the optimized asymptotic overlap is given by the real solution of the cubic equation

$$0 = 2\delta(\rho_\infty^{\mathrm{opt}})^2 - \frac{1}{\pi^2}\left(1 - (\rho_\infty^{\mathrm{opt}})^2\right)^3 \tag{39}$$

which is achieved when using the corresponding weight decay

$$\lambda_{\rm opt} = \frac{2}{\pi}\frac{1-(\rho_\infty^{\rm opt})^2}{(\rho_\infty^{\rm opt})^2} - \sqrt{2\delta}\sqrt{\frac{1-(\rho_\infty^{\rm opt})^2}{(\rho_\infty^{\rm opt})^2}}. \tag{40}$$

For very small drift parameters we find

$$\rho_\infty^{\rm opt} \approx 1 - \tfrac{1}{2}\left(2\pi^2\delta\right)^{1/3} \qquad \text{and} \qquad \epsilon_{\rm g}^{\rm opt}(\alpha\to\infty) \approx \frac{(2\delta)^{1/6}}{\pi^{2/3}}. \tag{41}$$

4.1.2. Optimal f procedure and perceptron algorithm. The optimal choice of a weight function f is not modified compared with section 3.2, because $f_{\rm opt}$ in (17) does not explicitly depend on the drift process. In the limit $\delta \to 0$ we find a residual generalization error

$$\epsilon_{\rm g}^{\rm opt}(\alpha\to\infty) \approx \sqrt{\frac{2}{\pi A}}(2\delta)^{1/4} \approx 0.63\delta^{1/4} \tag{42}$$

with A from equation (19).

Perceptron learning with zero stability ($f = \theta(-h_J S/Q), \lambda = 0$) gives in the same limit

$$\epsilon_{\rm g}(\alpha\to\infty) \approx \sqrt{\frac{2^{5/4}}{\pi}}\delta^{1/4} \approx 0.76\delta^{1/4} \tag{43}$$

which is already optimal with respect to the δ-dependence. Figure 4 shows the evolution of the generalization error for a given drift parameter and in figure 5 the dependence of $\epsilon_{\rm g}(\alpha\to\infty)$ on δ is plotted.

For $\delta \gtrsim 0.2$ the simple perceptron even gives asymptotic values for $\epsilon_{\rm g} > 1/2$, which is, in fact, worse than random guessing. Similar effects can be observed for the Hebb algorithm with suboptimal weight decay. This is merely an artifact of our specification of the drift process: the minimum of $\boldsymbol{J}\cdot\boldsymbol{B}$ subject to (33) can be negative for large enough δ. This is somewhat unphysical: a 'clever' student could 'wait' until $\rho = -1$, then change the signs of all weights and generalize perfectly. One could include an additional restriction $\boldsymbol{J}\cdot\boldsymbol{B} \geqslant 0$ in the definition of the drift, but we are mainly interested in the results for small δ, where this problem does not occur.

Helmbold and Long [11] recently derived, for general rules of a certain complexity (Vapnik–Chervonenkis-dimension $d_{\rm VC}$ [1]), lower bounds on the generalization error given only the amount of drift. Apart from constant factors, equations (42) and (43) coincide with their result for the linearly separable case ($d_{\rm VC} = N$). For slow drifts they find the relation

$$\epsilon_{\rm g}(\alpha\to\infty) \geqslant \frac{2^{1/4}}{\sqrt{\pi}\,{\rm e}^2}\delta^{1/4} \approx 0.09\delta^{1/4}. \tag{44}$$

This inequality holds for any possible learning procedure and arbitrary types of drift. Consequently, the basic dependence on the drift parameter for the considered evolution of $\boldsymbol{B}$ could not be improved by allowing more complicated schemes than the online algorithm of type (3) for the learning from random examples.

Kuh *et al* [12] find a similar relation for a two-dimensional ($N = 2$) perceptron tracking a drifting rule. Apparently, the structure of our results does not depend on the thermodynamic limit $N \to \infty$, but seems to hold more generally.

4.2. Queries

Training with examples that satisfy the condition $h_J = 0$ leads to a further improvement in the asymptotic generalization error. Asking intelligent questions allows us to exceed the bound (44), which was obtained for a fixed distribution of examples. As in the case of a random drift process, both the Hebb rule with optimal weight decay ($\lambda_{\mathrm{opt}} = 2\pi\delta$) and the optimal f procedure (section 3.2) yield the same asymptotic value. Here we find

$$\rho_\infty^{\mathrm{opt}} = 1/\sqrt{1 + 2\pi^2\delta} \tag{45}$$

which gives an error

$$\epsilon_g^{\mathrm{opt}}(\alpha \to \infty) \approx \sqrt{2\delta} \tag{46}$$

in the limit $\delta \to 0$, see figure 5.

5. Summary

We have studied the learning of a time-dependent rule in a single-layer neural network for two different drift processes of the target vector. For various online learning algorithms we investigated the generalization performance using both randomly selected examples and examples generated by a query strategy. None of the algorithms is able to learn the drifting rules perfectly. However, introducing the effect of forgetting allows us to track the rules within certain error margins. This was demonstrated by the use of weight decay and explicit weight functions in the different learning algorithms. The simple perceptron learning rule was found to work without adjusting parameters and thus without prior knowledge about the drift process. Remarkably, for small drifts its generalization error is already optimal up to a constant.

For the deterministic worst-case drift of section 4 the generalization error of the perceptron learning rule and the optimal f procedure reaches, apart from prefactors, the lower bounds found by Helmbold and Long for any algorithm [11]. Hence, the online learning scheme studied here provides an efficient tool for the tracking of drifting concepts. For its worst-case behaviour we do not expect a considerable improvement from using memory-based algorithms [12], since they also have to satisfy the bounds of [11]. However, for a general drift process a memory-based prescription may be useful, e.g. to detect a deterministic drift component along a given direction of phase space.

Compared with algorithms using randomly drawn training examples, the generalization performance can be further improved by generating examples according to a query strategy. The generalization ability achieved by such a strategy was found to exceed the bounds for learning rules relying on randomly chosen examples.

It might be interesting, and in a way more realistic, to consider deterministically varying concepts, where the drift is along a certain direction on the N-dimensional sphere. Obviously, the trackability of such a rule should somewhat interpolate between the random drift and the worst case.

Furthermore one might consider the presentation of patterns which are semantically or spatially correlated [24, 25]; the possible outcome of our theory for this case is unclear.

Acknowledgments

We would like to thank W Kinzel, P Kuhlmann, A Mietzner, M Opper and G Reents for helpful discussions. The simulations were done on the Cray Y–MP EL of the Rechenzentrum der Universität Würzburg. H Schwarze was supported by the EC under the Science programme and by the Danish Natural Science Council and the Danish Technical Research Council through CONNECT. He acknowledges the hospitality at the Physikalisches Institut der Universität Würzburg.

Note added in proof. After submission of this paper we received a preprint by Kinouchi and Caticha [26], in which basically the same results are obtained for the randomly drifting rule. Furthermore the authors suggest a method for estimating and monitoring the current generalization ability of the student.

References

[1] Hertz J A, Krogh A and Palmer R G 1991 *Introduction to the Theory of Neural Computation* (Redwood City, CA: Addison–Wesley)
[2] Seung H S, Sompolinsky H and Tishby N 1992 *Phys. Rev.* A **45** 6056
[3] Watkin T L H, Rau A and Biehl M 1993 The statistical mechanics of learning a rule *Rev. Mod. Phys.*
[4] Rosenblatt F 1962 *Principles of Neurodynamics* (New York: Spartan)
[5] Minsky M L and Papert S 1969 *Perceptrons* (Cambridge, Ma: MIT Press)
[6] Vallet F 1989 *Europhys. Lett.* **9** 315
[7] Opper M, Kinzel W, Kleinz J and Nehl R 1990 *J. Phys. A: Math. Gen.* **23** L581
[8] Watkin T L H 1992 *Preprint* Oxford University
[9] Kinouchi O and Caticha N 1992 *J. Phys. A: Math. Gen.* **25** 6243
[10] Heskes T M and Kappen B 1992 *Phys. Rev.* A **45** 8885
[11] Helmbold D P and Long P M 1991 *Proc. Fourth Ann. Workshop on Computational Learning Theory* ed M K Warmuth and L G Valiant (San Mateo: Morgan Kaufmann)
[12] Kuh A, Petsche Th and Rivest R L 1992 *Advances in neural Information Processing Systems III* ed R P Lippmann, J E Moody and D S Touretzky (San Mateo: Morgan Kaufmann)
[13] Biehl M and Schwarze H 1992 *Europhys. Lett.* **20** 733
[14] Hebb D O 1949 *The Organization of Behavior* (New York: Wiley)
[15] Derrida B and Nadal J-P 1987 *J. Stat. Phys.* **49** 993
[16] Krogh A and Hertz J A 1992 *Advances in Neural Information Processing Systems IV* ed J E Moody, S J Hanson and R Lippmann (San Mateo: Morgan Kaufmann)
[17] Mezard M, Nadal J-P and Toulouse G 1986 *J. Physique* **47** 1457
[18] van Hemmen J L, Keller G and Kühn R 1988 *Europhys. Lett.* **5** 663
[19] Kinzel W and Opper M 1991 *Models of Neural Networks* ed E Domany, J L van Hemmen and K Schulten (Berlin: Springer)
[20] Baum E B 1991 *IEEE Trans. on Neural Networks* **2** 5
[21] Kinzel W and Rujan P 1990 *Europhys. Lett.* **13** 473
[22] Watkin T L H and Rau A 1992 *J. Phys. A: Math. Gen.* **25** 113
[23] Seung H S, Opper M and Sompolinsky H 1992 *Proc. Fifth Ann. ACM Workshop on Computational Learning Theory* (New York: ACM)
[24] Monasson R 1992 *J. Phys. A: Math. Gen.* **25** 3701
[25] Tarkowski W and Lewenstein M 1992 *Phys. Rev.* A **46** 2138
[26] Kinouchi O and Caticha N 1992 *Preprint* Universidade São Paulo