On-line learning in the Ising perceptron

2000 J. Phys. A: Math. Gen. 33 7277

(http://iopscience.iop.org/0305-4470/33/41/302)

J. Phys. A: Math. Gen. **33** (2000) 7277–7287. Printed in the UK PII: S0305-4470(00)14593-7

On-line learning in the Ising perceptron

Michal Rosen-Zvi
Minerva Centre and the Department of Physics, Bar-Ilan University, Ramat Gan, 52900 Israel

Received 6 June 2000

Abstract. On-line learning of both binary and continuous rules in an Ising space is studied. Learning is achieved by using an artificial parameter, a weight vector $\vec{J}$, which is constrained to the surface of a hypersphere (spherical constraint). In the case of a binary rule the generalization error decays to zero super-exponentially as $\exp(-C\alpha^2)$, where α is the number of examples divided by N, the size of the input vector, and $C > 0$. Much faster learning is obtained in the case of continuous activation functions where the generalization error decays as $\exp(-e^{|\lambda|\alpha})$. The number of steps required for perfect learning is estimated for both scenarios and compared with simulations.

1. Introduction

The application of methods of statistical mechanics to neural networks has yielded a wealth of results. Analysing the prototype architecture, the perceptron produces several types of behaviour such as phase transitions [1–3]. Here a learning algorithm for the one-layer network in the case of Ising weights is presented. The goal is to find the student's weight vector, $\vec{W}^{\mathrm{S}}$, that minimizes the generalization error $\epsilon_{\mathrm{g}}(\vec{W}^{\mathrm{S}})$. The error is determined relative to a network that generates for any input $\vec{\xi}$ an output S, according to some rule, $S = F(\frac{\vec{W}^{\mathrm{T}}\cdot\vec{\xi}}{\sqrt{N}})$. Here $\vec{W}^{\mathrm{T}}$ is the teacher's weight vector and N is the size of the input. In the following we will discuss both binary and continuous activation functions. These two types of activation function have different characteristics from the on-line scenario point of view as well as that of the off-line scenario. In the on-line scenario the learning in each step is only the latest from a sequence of examples. Such an algorithm drastically reduces the computational effort compared with batch learning and no explicit storage of a training set is required [4]. In the batch scenario an effort is made to generalize according to a fixed set of αN patterns. The quantity that determines the quality of the learning is the generalization error ϵ_{g}, which is calculated by taking the average over the examples of the mismatch between teacher and student. It was shown that ϵ_{g} in the case of perceptrons depends on the overlap between teacher and student, $R = \frac{1}{N}\vec{W}^{\mathrm{S}}\cdot\vec{W}^{\mathrm{T}}$, and the student norm, $Q = \frac{1}{N}\vec{W}^{\mathrm{S}}\cdot\vec{W}^{\mathrm{S}}$.

The binary Ising model has been studied from the batch point of view [5–9]. In the case of a perceptron an off-line learning procedure was proved to exhibit a first-order transition from a state of poor learning to a state of perfect learning at $\alpha \sim 1.25$ [9]. However, there was no practical algorithm showing this phase transition or even coming near to it. A new line of research was established by Van den Broeck and Bouten [10] and was expanded later on in [11]. They suggested a learning rule which is based on a clipping of a continuous perceptron. Having an artificial continuous weight vector enables smooth learning; clipping it results in a binary student $\vec{W}^{\mathrm{S}}$, whose components are close to those of the teacher. The algorithm does not show a phase transition but nevertheless exhibits an exponential decay of the generalization

0305-4470/00/417277+11$30.00

error to zero. A first-order phase transition was found in batch learning of the continuous Ising model as well [12]. All in all, off-line learning in Ising weight space of all transfer functions that have been studied results in a first-order phase transition.

The clipping method has been used in a related field, dealing with capacity calculations. The weight-space of the binary perceptron was examined by comparing it to a clipped continuous network by Penney and Sherrington [13]. An example of further development along this line is [14].

Turning to on-line learning, it has been shown [15] that in the case of on-line learning in an Ising weight space there is no mapping that updates $\vec{W}^{\mathrm{S}}$ in the following manner: $\vec{W}'^{\mathrm{S}} = g(\vec{W}^{\mathrm{S}}, \vec{\xi}, S(\vec{\xi}, \vec{W}^{\mathrm{T}}))$. The examples, $\vec{\xi}$, are drawn from a Gaussian distribution and the N-dimensional weight vector $\vec{W}^{\mathrm{S}}$ is constrained to take values in a discrete state space, L^N. Generalization is only possible if L is large enough, of the order of $\sqrt{N}$. Using an artificial continuous vector ($L \to \infty$) and clipping it enables generalization. Some aspects of on-line learning in the case of a binary Ising perceptron have been studied as one can find in [16].

However, there are still some unsolved problems. First, learning was proved to improve in the case of a binary rule; what happens in the case of a continuous rule? Secondly, the clipping method was a 'trick' used independent of the updating of the student's weights. The student who wants to find out the teacher's weights treats her as if she were continuous. The fact that the teacher is binary is ignored during the updating procedure. The clipping is performed only at the end or at any point along the way. What happens if the student tries to update his continuous weights according to the output which is a result of his clipped weights? Only in on-line learning can one change the learning algorithm during the learning itself. Perhaps this is the preferred alternative. Thirdly, the analytical calculations are made in the case of infinite N, although simulations are made in large but finite N. What are the finite-size effects?

The remainder of the paper is organized as follows. In section 2 we present an extensive study of the qualities of *on-line learning* in an Ising space using a continuous artificial weight vector. Although it has already been introduced, it was done only indirectly. We prefer to start by introducing the clipping method using the on-line neural network language and then proceed emphasizing the on-line aspects. In section 3 we demonstrate the results in the case of a binary rule whereas in section 4 we present results in the case of a continuous rule. Non-trivial scaling relations between the number of steps required for perfect learning in both cases are introduced in section 5. Finally, in section 6 we summarize our results and present some concluding remarks.

2. Clipping in the case of on-line learning

In order to learn the rule given by a binary perceptron, the student has to use an artificial vector, a continuous one, $\vec{J}$, as was explained. The ith component of the clipped weight vector, $\vec{W}^{\mathrm{S}}$, is determined from the ith component of $\vec{J}$ according to

$$W_i^{\mathrm{S}} = \mathrm{sign}(J_i). \tag{1}$$

The learning algorithm for J may be any algorithm that has been used in the spherical case [4, 17–20]. In general, in the clipping method one updates the continuous student, $\vec{J}$. The generic form of the learning algorithm is

$$\vec{J}^{\mu+1} = \vec{J}^{\mu} + \frac{\eta}{\sqrt{N}} f(S^{\mu}, x_J^{\mu}) \vec{\xi}^{\mu} S^{\mu} \tag{2}$$

which means that at each learning set μ, the current weight vector $\vec{J}^{\mu}$ is updated according to the new example, $\vec{\xi}^{\mu}$. Here x_J is the student's local field, $x_J \equiv \frac{1}{N} \vec{J} \cdot \vec{\xi}$. Assuming self-averaging

properties for the following overlaps:

$$R_J \equiv \frac{1}{N}\vec{J}\cdot\vec{W}^{\mathrm{T}} \tag{3}$$

$$Q_J \equiv \frac{1}{N}\vec{J}\cdot\vec{J} \tag{4}$$

and using equation (2), one can find a system of differential equations for the order parameters R_J and Q_J and the 'continuous time', $\alpha = \frac{\mu}{N}$. The clipped order parameter, the one that determines the learning curve, is given by the overlap of the Ising-student and the teacher,

$$\rho \equiv \frac{1}{N}\vec{W}^{\mathrm{S}}\cdot\vec{W}^{\mathrm{T}}. \tag{5}$$

The norm, $Q \equiv \frac{1}{N}\vec{W}^{\mathrm{S}}\cdot\vec{W}^{\mathrm{S}}$, is obviously equal to unity by definition.

The order parameter, ρ, in the binary machine as a function of $\rho_J \equiv \frac{R_J}{\sqrt{Q_J}}$ was shown in [11] to be

$$\rho(\rho_J) = \mathrm{erf}\left(\frac{1}{\sqrt{2}}\frac{\rho_J}{\sqrt{1-\rho_J^2}}\right). \tag{6}$$

As was explained there, this relation was obtained on the grounds of the reasonable assumption that the multiplication $J_i W_i$ generally depends only on the order parameters R_J and Q_J. We will demonstrate that this assumption is no longer valid when one uses a training rule which cannot be described by equation (2). In principle one can use the above mapping for any result known in the continuous space and find the learning curve which describes the learning procedure in the clipped machine. Yet, one can update the artificial vector according to the quantities related to the binary student. The adaptation of the artificial continuous weight vector according to the Ising weight vector's local field, x_W, is described in general by the following equation:

$$\vec{J}^{\mu+1} = \vec{J}^{\mu} + \frac{\eta}{N} f(S^{\mu}, x_W^{\mu})\vec{\xi}^{\mu} S^{\mu}. \tag{7}$$

Note that the development of the weight vector $\vec{J}$ and hence the development of R_J and Q_J directly depends on ρ. This means that equation (6) is no longer valid.

In order to be able to interpret the updating rule in equation (7) to a system of differential equations one has to obtain the local field distribution, $P(x_W, y_W, x_J, y_J|R_J, Q_J)$. The teacher's local field in the continuous procedure is $y_J \equiv \frac{1}{N}\vec{W}^{T}\cdot\vec{\xi}$, and there is the obvious relation $P(y_J|y_W, A) = \delta(y_J - y_W)$ Using the simple relation for conditional probability, the product rule, $P(x, y|A) = P(x|y, A)P(y|A)$ and taking the one to one relation between R_J, Q_J, and ρ, assembling everything, we have

$$P(x, y, x_J, y_J|R_J, Q_J) = P(x_J, y_J|R_J, Q_J)P(x, y|\rho)\frac{\delta(y_J - y)}{P(y_J)} \tag{8}$$

where we have eliminated the subscript W over the local fields' symbols. The distribution of the teacher's local field is $P(y) = \exp(-\frac{1}{2}y^2)/\sqrt{2\pi}$ and the conditional probabilities are

$$P(x_J, y_J|R_J, Q_J) = \frac{1}{2\pi}\frac{1}{\sqrt{Q_J - R_J^2}}\exp\left(-\frac{1}{2}\frac{x^2 - 2R_J xy + Q_J y^2}{Q_J - R_J^2}\right) \tag{9}$$

$$P(x, y|\rho) = \frac{1}{2\pi}\frac{1}{\sqrt{1-\rho^2}}\exp\left(-\frac{1}{2}\frac{x^2 - 2\rho xy + y^2}{1-\rho^2}\right). \tag{10}$$

Finding the equations of motion of the order parameters, R_J, Q_J is performed in the standard rigorous way, only now one has to average over the above-mentioned distribution equation (8). The result will be two coupled equations, $\frac{dR_J}{d\alpha} = g(R_J, Q_J, \rho)$, $\frac{dQ_J}{d\alpha} = g(R_J, Q_J, \rho)$. Since the development of the order parameters directly depends on ρ, it is obvious that ρ must be not only a function of ρ_J but a function of η as well. Hence, one cannot find $\rho(\rho_J, Q_J)$. Finding $\frac{d\rho}{d\alpha}$, an equation for the rate of change in ρ as a function of R_J, Q_J and η is impossible as well, since W_S depends on J in a discontinuous manner, which makes it impossible to obtain a continuous equation for $\frac{d\rho}{d\alpha}$.

Generally speaking, we found that learning according to the last procedure results in a slower decay of the generalization error. We analysed this training rule from the on-line point of view but of course one can use it in batch learning. However, updating according to the continuous vector's quantities (equation (2)) during most of the learning procedure and turning to the last procedure (equation (7)) at the very last steps of the learning (where the generalization error is small enough) ends in a faster decay. The ability to change the training rule in the learning procedure in such a way is unique to on-line learning. The last result is obtained (in the specific cases that we covered as written in section 5) in simulations for large but finite N.

In order to describe the details we have to be more specific concerning the rule, F (equations (11) and (28)), and the learning algorithm, f (equations (13) and (22)). All of the above is described in the following paragraphs.

3. Binary rule

We analyse the learning procedure in the case of a binary rule,

$$S = \mathrm{sign}(x) \tag{11}$$

for a given local field, x. In this case the generalization error as a function of ρ is known to be

$$\epsilon_g = \frac{1}{\pi}\cos^{-1}(\rho). \tag{12}$$

Although it was shown that using the 'expected stability' algorithm that maximizes the generalization gain per example leads to an upper bound for the generalization ability [17], we choose to concentrate on the so-called AdaTron or relaxation learning algorithm. This latter algorithm for zero stability, $\kappa = 0$, performs comparably well and unlike the 'expected stability' algorithm does not require additional computations in the student network besides the updating of its weights [4].

We update the artificial continuous weight vector, $\vec{J}$, similarly to the procedure of equation (2), according to the above learning rule:

$$J_i^{\mu+1} = J_i^{\mu} - \frac{\eta}{\sqrt{N}}\left(\frac{\vec{J}^{\mu}\cdot\vec{\xi}^{\mu}}{\sqrt{N}}\right)\xi_i^{\mu}\theta\left(-\frac{\vec{J}^{\mu}\cdot\vec{\xi}^{\mu}}{\sqrt{N}}S\right). \tag{13}$$

Hence, the equation of motion for ρ_J in the case of $\eta = 1$ results in the following expression [4]:

$$\frac{d\rho_J}{d\alpha} = -\frac{\rho_J}{2\pi}\cos^{-1}(\rho_J) + \frac{1}{\pi}\left(1 - \frac{\rho_J^2}{2}\right)\sqrt{1-\rho_J^2}. \tag{14}$$

The solution of equation (14) only describes the curve of the artificial, continuous perceptron. The transformation to the clipped order parameter, ρ, is given by equation (6) and the generalization error is a function of ρ according to equation (12). This results in the Gaussian decay of the generalization error as $\alpha \to \infty$

$$\epsilon_g \propto \frac{\exp\left(-\frac{\alpha^2}{9\pi^2}\right)}{\alpha^{\frac{1}{2}}}. \tag{15}$$

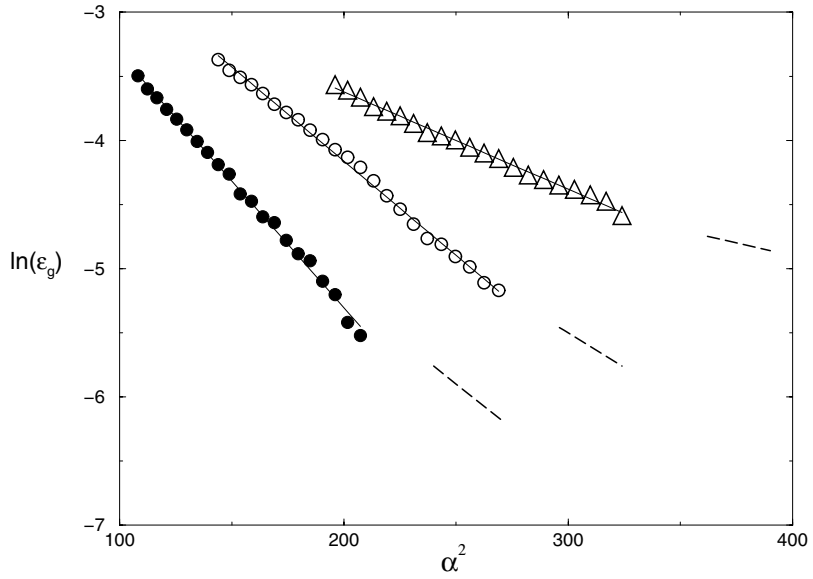

Figure 1. Simulation results of $\ln \epsilon_g$ versus the square of the number of steps is plotted in the case of $\eta = 1$ (○), $\eta = 1.5$ (●) and $\eta = 2.5$ (△). Curves are least-squares fits to numerical data. Error bars are not visible at the scale of the figure. Dashed lines are the analytical slopes.

In comparison, the result for the Hebb learning [10] is

$$\epsilon_g \propto \frac{\exp\left(-\frac{\alpha}{2\pi}\right)}{\alpha^{\frac{1}{4}}}. \tag{16}$$

The asymptotic decay is much slower here, reflecting the differences in the decay in the spherical case [4].

The rate of convergence to zero error clearly depends on the learning rate, η. In order to find the optimal learning rate, which results in a faster decay at large number of learning steps, one needs to calculate the generalization error in the $\alpha \to \infty$ limit as a function of η,

$$\epsilon_g \propto \exp\left[-\frac{\alpha^2\eta^2(2-\frac{2}{3}\eta)^2}{(4\pi)^2}\right]. \tag{17}$$

The optimal learning rate is $\eta = 1.5$. We performed simulations in the case of $N = 1000$, averaging each point over 100 samples. We demonstrated the super-Gaussian behaviour in the following rates: $\eta = 1, 1.5, 2$. The slopes of the linear curves should be -0.011, -0.014, -0.004 respectively. Figure 1 shows that there is good agreement between the simulation results and the analytical predictions. In section 5 we discuss the finite-size effects.

One can update the artificial continuous weight according to the quantities in the clipped machine instead of those of the artificial one. The equations of motion in this case are calculated in the same manner as explained in the previous chapter. The result is

$$\frac{\mathrm{d}\rho_J}{\mathrm{d}\alpha} = \frac{\eta}{\pi}\left[(1-\rho_J^2)\frac{\sqrt{1-\rho_W^2}}{\sqrt{Q_J}} - (1-\rho_J^2)\frac{\rho_W}{\sqrt{Q_J}}\cos^{-1}(\rho_W)\right]$$
$$-\frac{\eta^2}{2\pi}\frac{\rho_J}{Q_J}\left[\cos^{-1}(\rho_W) - \rho_W\sqrt{1-\rho_W^2}\right] \tag{18}$$

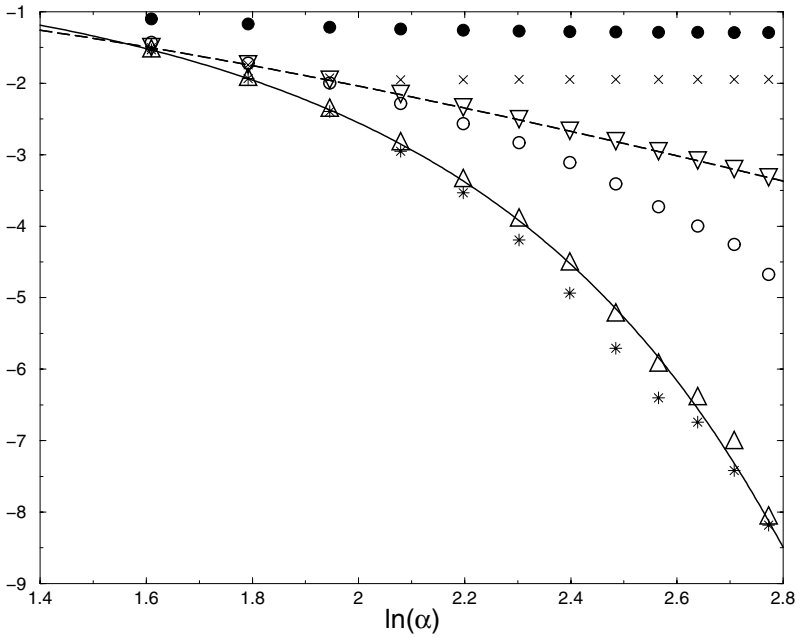

Figure 2. Simulation results of $\ln(1-\rho_W)$ and $\ln(1-\rho_J)$ are plotted in the following three cases: (1) updating according to the continuous quantities (ρ_W, $\triangle$, ρ_J, $\triangledown$). (2) Updating according to the clipped vector quantities (ρ_W, $\circ$, ρ_J, $\bullet$). (3) Updating as in case (1) and alternating to update as in case (2) as far as $\rho_W > 0.9$, (ρ_W (*) and ρ_J (x)). ρ_J according to the analytical results in the first case is the dashed curve whereas ρ_W is the solid curve. All plots are versus $\ln(\alpha)$.

$$\frac{\mathrm{d}Q_J}{\mathrm{d}\alpha} = \frac{2\eta}{\pi}\rho_J\sqrt{Q_J}\left[\sqrt{1-\rho_W^2} - \rho_W\cos^{-1}(\rho_W)\right] + \frac{\eta^2}{\pi}\left[\cos^{-1}(\rho_W) - \rho_W\sqrt{1-\rho_W^2}\right]. \quad (19)$$

The last equations constitute an incomplete set of differential equations. Therefore it is impossible to obtain analytical curves from these equations in figure 2. Furnishing an equation for $\frac{\mathrm{d}\rho}{\mathrm{d}\alpha}$ is impossible, as was noted in section 2.

It is very clear that as long as $\rho_W = 1$ the right-hand side of both equations is zero, i.e. this is a fixed point independent of the R_J, Q_J values. In other words, although the continuous student did not learn the teacher quantities perfectly, the clipped one did. Using the knowledge of binary weights in such a way at the initial stages of the learning procedure is too puzzling. That is why more steps are needed using the latter algorithm then the former in order to obtain perfect learning. However, the latter algorithm is found to be more economical in the very last stages since instead of wasting information in the effort of minimizing the angle between the continuous vector and the teacher, one only concentrates on the Ising student vector.

We made simulations with a learning rate $\eta = 1$, and $N = 1000$. Each point in the simulation results presented in figure 2 is averaged over 100 samples. Initial conditions in all carried simulations are $R_J = \rho_W = 0$, $Q_J = 0.5$. Figure 2 shows simulation results according to the last procedure, that ends in a slower convergence of the overlap ρ_W to 1. An example of simulation results which updates the student according to the continuous quantities in the initial stages of the learning procedure and alternates at the end of the learning to updating according to the discrete quantities is given in figure 2. The analytical calculations according to equations (6) and (13) are plotted in figure 2 as solid and dashed curves. Comparison is made between the development of the 'standard' order parameter ρ_J and the new one, ρ in each

case. Learning in the case of updating the continuous vector according to its own quantities results in a faster convergence then in the case of updating according to the clipped quantities. Even faster learning is gained in the alternating case as one can see in the plot.

The well known result of a power law decay of the generalization error in the case of continuous weights reappears as soon as a finite fraction, P, of the teacher weight vector's components are continuous. In that case

$$\rho(R_J, Q_J) = \frac{(1-P)\mathrm{erf}\left(\frac{1}{\sqrt{2}}\frac{\rho_J}{\sqrt{1-\rho_J^2}}\right) + PR_J}{\sqrt{1-P+PQ_J}} \tag{20}$$

and the generalization error decreases to zero according to

$$\epsilon_g \sim \frac{\sqrt{P}A}{\alpha}. \tag{21}$$

Here A is characterized by the specific on-line learning algorithm: in the case of optimal learning, for instance, its value is 0.88 [17]. Having a finite fraction $1-P$ of binary components decreases the factor above while $P = 0$ results in an exponential decay.

4. Continuous rules

We now study the case of continuous perceptrons with Ising weights. As long as one uses a continuous activation function, the generalization error decreases exponentially (see for instance [18–20]). In order to learn a rule which is defined by a binary vector, we used a spherical vector for the student weight vector, $\vec{J}$, and clipped it in order to have a binary student weight vector $\vec{W}^S$. The updating of the spherical student weight vector is done according to the gradient descent method as usual:

$$\vec{J}^{\mu+1} = \vec{J}^{\mu} - \frac{\eta}{\sqrt{N}}\nabla_{\vec{J}}\epsilon(\vec{J}^{\mu}, \vec{\xi}^{\mu}). \tag{22}$$

The error $\epsilon(\vec{J}^{\mu}, \vec{\xi}^{\mu})$ measures the deviation of the student from the teacher's output for a particular input $\vec{\xi}$. The generalization error of a student is defined as the averaged error

$$\epsilon_g = \langle \tfrac{1}{2}[S(\vec{J}, \vec{\xi}) - S(\vec{W}, \vec{\xi})]^2 \rangle_{\vec{\xi}}. \tag{23}$$

Note that in the case of continuous rules there is the same variety of possibilities as in the case of binary rules, i.e. one can update the student weight vector according to a gradient of an error which measures the deviation of the clipped student from the teacher's output (as was shown in equation (7)).

In general, one can show that the generalization error, ϵ_g, is explicitly dependent only on the two order parameters R_J and Q_J. The learning curves of the continuous version are the same as if there were a rule defined by a continuous teacher (having a limitation of two available values for the components is merely a special case of the spherical constraint). Under the limitation of a small enough learning rate, η, $R_J = 1$, $Q_J = 1$ is an attractive fixed point. Linearizing the equations of motion around this fixed point results in the following general form:

$$R_J = 1 - \frac{c_1}{\det V}V_{22}\exp(\lambda_1\alpha) + \frac{c_2}{\det V}V_{12}\exp(\lambda_2\alpha) \tag{24}$$

$$Q_J = 1 + \frac{c_1}{\det V}V_{21}\exp(\lambda_1\alpha) - \frac{c_2}{\det V}V_{11}\exp(\lambda_2\alpha). \tag{25}$$

The two eigenvalues of V, λ_1, λ_2, are both negative. c_1, c_2 are some initial conditions that are determined from the numerical solution of the equations of motion. Note that the

last two equations depend on the actual transfer function in detail but not in principle. The generalization error ϵ_g can be calculated as a function of ρ_W, the overlap between the binary weight vectors. Since the local field distribution in the last case is the same as that in the continuous case it results in the same function as that already given for the specific activation function in the continuous space (only now $Q = 1$ by definition, and the term is a function not of R_J but of ρ). Using the simple relation between ρ and ρ_J, equation (6) [11] in the $\alpha \to \infty$ limit, results in the following form:

$$\rho_{\alpha\to\infty} = 1 - \sqrt{\frac{1}{\pi C_0}} \exp\left(\frac{\lambda_1 \alpha}{2}\right) \exp\left(-C_0 e^{-\lambda_1 \alpha}\right). \tag{26}$$

Here $C_0 \equiv \frac{\det V}{2c_1(2V_{22}+V_{21})}$, λ_1, λ_2 are both negative and λ_1 absolute value is assumed to be smaller than λ_2. Hence, the generalization error decays as

$$\epsilon_g \sim \exp\left(\frac{\lambda_1 \alpha}{2}\right) \exp\left(-C_0 e^{-\lambda_1 \alpha}\right). \tag{27}$$

To examine the analytical predictions we carried out simulations on perceptrons with a 'sin' activation function [20]

$$S = \sin\left(\frac{\vec{W}^{\mathrm{T}} \cdot \vec{\xi}}{\sqrt{N}}\right). \tag{28}$$

In this case for $\eta = 1$, one obtains perfect learning, that means $R_J = 1$, $Q_J = 1$ is an attractive fixed point. Linearization around these fixed points ends with the following quantities: $\lambda_1 = -0.3$, $\lambda_2 = -0.7$,

$$V = \begin{pmatrix} -1.03 & 0.51 \\ 0.05 & -1 \end{pmatrix}. \tag{29}$$

The generalization error decreases to zero as

$$\epsilon_g \sim \exp\left(-\frac{0.3\alpha}{2}\right) \exp\left(-C_0 e^{0.3\alpha}\right). \tag{30}$$

When $\eta = 0.1$ perfect learning is still a fixed point, of course, but the eigenvalues are much smaller, $\lambda_1 = -0.03$, $\lambda_2 = -0.109$. Simulation results are presented in figure 3. The generalization error, ϵ_g, the teacher–continuous student overlap divided by the norm, ρ_J, and the teacher–student overlap, ρ_W, are plotted as a function of α. The initial conditions are $R_J = \rho_W = 0$, $Q_J = 0.5$. The simulations are made in the case of $N = 1000$, and the results are averaged over 100 samples.

We compared the numerical results of the development of the order parameters to the simulation results. As usual, at the beginning of the learning the overlap between the continuous vector and the teacher is larger than the discrete one, $\rho_J > \rho_W$.

5. Finite-size effects

Simulation results in the case of off-line learning under the spherical constraint showed that ϵ_g, the generalization error, is linear in $1/N$ for any α and N large enough that the second-order corrections in $1/N$ are negligible [21]. The learning in the Ising space is characterized by a cutoff. The one step before perfect learning is well defined where $\rho = 1 - \frac{2}{N}$, or more generally in the very last steps $\rho = 1 - \frac{A}{N}$. Hence, it is possible to calculate the influence of having a large but finite N. The small generalization error, right before perfect learning, will be, according to the last argument, $\epsilon_g \propto \frac{a}{\sqrt{N}}$. The number of steps required for absolute

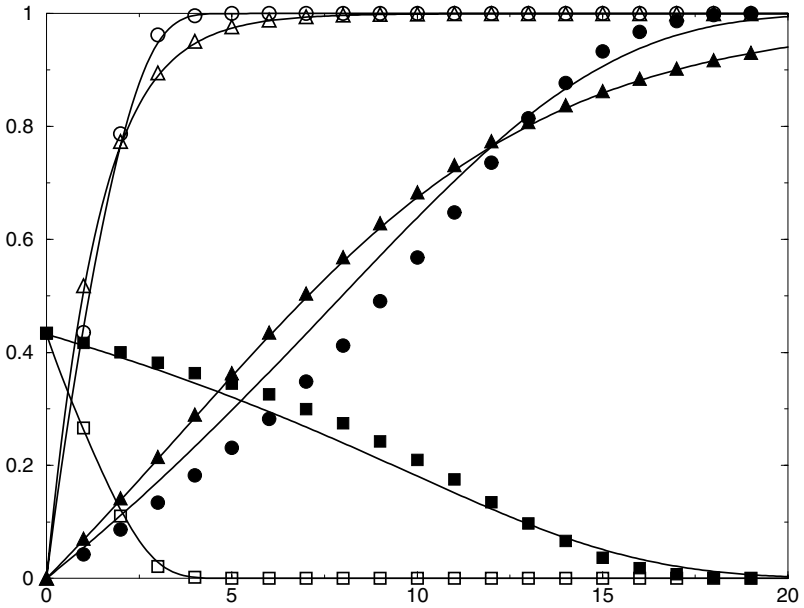

Figure 3. The order parameter ρ_J is the open triangles in the case of $\eta = 1$ and the filled triangles in the case of $\eta = 0.1$. The generalization error ϵ_g is the open squares in the case of $\eta = 1$ and the filled ones in the case of $\eta = 0.1$. The teacher–student overlap, ρ_W, is the open circles and the filled ones in the cases of $\eta = 1$ and $\eta = 0.1$, respectively. Analytical results are the solid curves.

learning, α_f, in finite dimension, is smaller than that predicted by the theory as a result of the finite-size corrections reflecting the lower generalization error in finite-dimension learning.

In the case of the binary perceptron the generalization error decays super-exponentially and one can find, as a result, that α_f scales as $\sqrt{\ln N}$ by setting $\epsilon_g = \frac{A}{\sqrt{N}}$ in equation (17). After neglecting higher-order terms one can obtain a very different scaling relation from the above-mentioned spherical constraint case,

$$\alpha_f \sim \frac{4\pi}{\eta(2 - \frac{2}{3}\eta)}\sqrt{\ln N}. \tag{31}$$

In our simulations we determined the number of steps α_f in which the overlap between the clipped student and the teacher was exactly unity, averaged over $M(N)$ training sets. Values of $M(N)$ ranged from 1000 to 50 in accordance with N, which is varied between 100 and 10 000. To obtain results in lower dimension, N, we averaged over a larger number of simulations, M. The obtained values of $\alpha_f(N, \eta)$ are presented in figure 4 as a function of $\sqrt{\ln N}$. As was shown theoretically, the time required to perfect learning α_f is linear in $\sqrt{\ln N}$. In the thermodynamic limit $N \to \infty$, $\alpha_f \to \infty$ as expected. The slope is determined according to equation (31) for a given learning rate, η.

The results for three different learning rates are presented in figure 4: the optimal learning rate $\eta = 1.5$, a smaller one $\eta = 1$ and a larger one $\eta = 2.5$; the slopes are determined to be 5.9, 6.7 and 10.7, respectively. The simulation results show an agreement with the predictions of linearity in $\sqrt{\ln N}$, although there is a clear influence of higher-order terms.

In the case of a continuous activation function it is also possible to give an estimation of the number of steps (αN) required to learn the rule perfectly. In the above-mentioned example if the constants c_1 and c_2 as well as the matrix V contain elements that are of the order of unity,

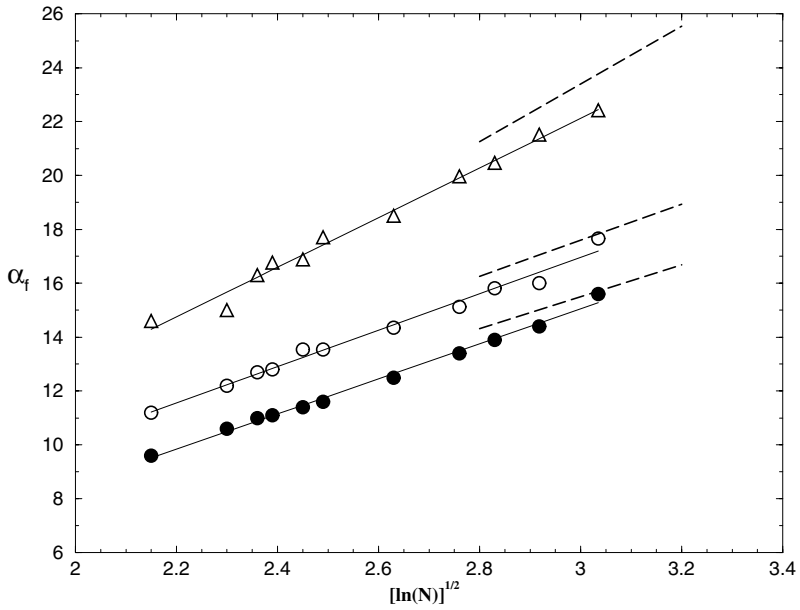

Figure 4. Average number of steps required for absolute learning versus $\sqrt{\ln N}$ are plotted in the case of $\eta = 1.5$ (•), $\eta = 1$ (○), $\eta = 2.5$ (△). Error bars are not visible at the scale of the figure. Lines are least-squares fits to numerical data. Dashed lines are the analytical slopes as calculated in equation (31).

$C_0 \sim 1$, the estimation holds also for finite but large enough N:

$$\alpha_f \sim \frac{1}{-\max(\lambda_1, \lambda_2)} \ln\left(\ln\left(\frac{N}{2}\right)\right)\left(1 - \frac{1}{2\ln(\frac{N}{2})}\right). \tag{32}$$

Both of the eigenvalues λ_1 and λ_2 are negative and the decay in the limit $\alpha \to \infty$ is dominated by the larger one. The eigenvalues are proportional to the learning rate as $\eta \to 0$ and hence $\alpha \sim \frac{1}{\eta}\ln(\ln(\frac{N}{2}))$, which means that the number of steps required for full learning increases as η decreases.

6. Discussion

In summary, we have studied the generalization properties of a perceptron with Ising weights and binary/continuous activation functions in an on-line scenario. The analysis, which is based on on-line updating of an artificial continuous vector and then clipping it, yields two different results. The continuous functions, in general, exhibit a very fast decrease of the generalization error, unlike the case of 'sign' activation function, which results in a slower decrease, an exponential decay.

We demonstrated several ways of using the idea of clipping in order to achieve fast generalization. We mainly examined two algorithms, clipping according to the continuous quantities or according to the discrete. Of course one can produce a combination of the two scenarios. We showed that clipping according to the continuous quantities results at the beginning with ρ_W which is smaller than ρ_J, but at some point in the learning it overtakes it. Clipping according to the discrete quantities results at the beginning of the procedure with over-pay and slow development of both overlaps, ρ_W, and ρ_J. Hence, it is better to alternate

between different algorithms and the freedom to do this is unique to on-line learning.

The clipping method above can be generalized to any number of values L not just the Ising (± 1) case. The same asymptotic behaviour is achieved [22].

Acknowledgments

The author is indebted to I Kanter for his helpful comments. The author thanks M Biehl and W Kinzel for a fruitful on-going dialogue.

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] Watkin T L H, Rau A and Biehl M 1993 *Rev. Mod. Phys.* **65** 499
[3] Kinzel W 1998 *Phil. Mag.* B **77** 1455
[4] Biehl M and Riegler P 1994 *Europhys. Lett.* **28** 525
[5] Gardner E and Derrida B 1989 *J. Phys. A: Math. Gen.* **22** 1983
[6] Sompolinsky H, Tishby N and Seung H S 1990 *Phys. Rev. Lett.* **65** 1683
[7] Györgi G 1990 *Phys. Rev.* A **41** 7097
[8] Opper M and Haussler D 1991 *Phys. Rev. Lett.* **66** 2677
[9] Seung H S, Sompolinsky H and Tishby N 1992 *Phys. Rev.* A **45** 6056
[10] Van den Broeck C and Bouten M 1993 *Europhys. Lett.* **22** 223
[11] Schietse J, Bouten M and van den Broeck C 1995 *Europhys. Lett.* **32** 279
[12] Kwon C, Park Y and Oh J-H 1993 *Phys. Rev.* E **47** 3707
[13] Penney R W and Sherrington D 1993 *J. Phys. A: Math. Gen.* **26** 6173
[14] Bouten M, Reimers L and Van Rompaey B 1998 *Phys. Rev.* E **58** 2378
[15] Kinzel W and Urbanczik R 1998 *J. Phys. A: Math. Gen.* **31** L27–30
[16] Solla S A and Winther O 1998 *On-line Learning in Neural Networks (Publications of Network Institute)* ed D Saad (Cambridge: Cambridge University Press) and references therein
[17] Kinouchi O and Caticha N 1992 *J. Phys. A: Math. Gen.* **25** 6243
[18] Biehl M and Schwarze H 1995 *J. Phys. A: Math. Gen.* **28** 643
[19] Saad D and Solla S A 1995 *Phys. Rev. Lett.* **74** 4337
Saad D and Solla S A 1995 *Phys. Rev.* E **52**
[20] Rosen-Zvi M, Biehl M and Kanter I 1998 *Phys. Rev.* E **58** 3606
[21] Buhot A, Torres Moreno J-M and Gordon M B 1997 *Phys. Rev.* E **55** 7434
[22] Kanter I and Rosen-Zvi M 2000 unpublished