Statistical Mechanics of On–line Learning and Generalization

Michael Biehl[(a)] and Nestor Caticha[(b)]
(a) Universität Würzburg - Institut für Theoretische Physik - Am Hubland, D-97074 Würzburg -Germany
(b) Universidade de São Paulo - Instituto de Física CP66318 São Paulo 05389-970 - Brazil

Short title: On–line learning and Generalization

Correspondence:

Michael Biehl
Institut für Theoretische Physik
Julius–Maximilians–Universität
Am Hubland
D–97074 Würzburg, Germany

Phone: +49–931–888–5865
Fax: +49–931–888–5141
email: biehl@physik.uni-wuerzburg.de

INTRODUCTION

In trying to understand how artificial neural networks (ANN) learn from examples, a variety of questions can be addressed which may require very different approaches. When asking about the typical properties of large ANN, the framework of Statistical Mechanics (SM) provides the natural set of tools as it was developed in order to obtain macroscopic properties emerging from microscopic interactions among a large number of units.

The aim of this survey is to introduce the reader to the SM theory of on-line training of feed–forward neural networks and their generalization ability. The main characteristic of on–line learning is that training examples are dealt with one at a time, as opposed to off–line or memory based methods, where learning is guided by the minimization of a cost function which incorporates possibly all of the data. From a statistical physics point of view, the distinction is between systems which can be thought of as being in a state of thermal equilibrium and off–equilibrium situations where the network is not allowed to extract all possible information from a set of examples. The equilibrium SM of off–line learning and generalization is reviewed in (Engel and Zippelius; Opper, this volume).

While on–line learning is an intrinsically stochastic process, the restriction to large networks, togther with assumptions about the statistical properties of the inputs, permits a concise description of the dynamics in terms of coupled ordinary differential equations. These deterministic equations govern the average evolution of quantities that completely define the macroscopic state of the ANN. The average is taken with respect to the data, which is straightforward if the presented examples are statistically independent.

The study of this off-equilibrium situation can be carried out analytically for a variety of network architectures and environments. The type of question this framework has allowed to investigate includes (i) the determination of learning curves for a large class of models, (ii) the derivation of upper bounds to the typical performance and how these bounds could be approached, (iii) dynamical symmetry breaking in hidden layer neurons which can model rapid transitions into new levels of generalization ability, and (iv) the effect of different types of noise, or changes in the environment. In particular this last issue arises in real world applications and shows the realm where on–line learning is to be preferred. The performance of on–line algorithms competes well with sophisticated off–line prescriptions. However, the computational cost and memory requirements are significantly lower. Off–line training is not capable of dealing with environment drifts, even less so with sudden changes, since no distinction is made between old, possibly irrelevant, and more recent data. This is treated most naturally by on–line methods, see (Vicente et al., 1998) and

references therein.

THE DYNAMICS OF ON-LINE LEARNING

In the following we will illustrate the theoretical treatment of on–line learning in terms of a specific class of neural networks, for the sake of simplicity. Note, however, that many of the restrictions made here can be relaxed without complicating the analysis very much.

We consider student networks with N real valued inputs, K units in a hidden layer and one single output. Denote by $\mathbb{J} = (\mathbf{J}_1, \mathbf{J}_2, ...\mathbf{J}_K)$ the set of adaptive student weights which connect the input layer to the hidden units. Here and in the following, boldface vectors are in $I\!R^N$. The state of a hidden unit is taken to be a function $\sigma_J^k = f_{hid}(x_k)$ of the projection $x_k = \mathbf{J}_k \cdot \boldsymbol{\xi}$ and the total output is given as $\Sigma_J(\boldsymbol{\xi}) = f_{out}(\sigma_J^1, \sigma_J^2, \ldots, \sigma_J^K)$. All units may have boolean (± 1) or continuous (e.g. linear or sigmoidal) transfer functions.

In general, the functions f_{hid}, f_{out} can be specified through a set of additional adaptive parameters. For simplicity we will concentrate here on cases where only $\mathbb{J}$ is to be determined in the course of learning and f_{out} is fixed a priori. Learning a finite number of second layer weights or thresholds, for instance, is an easier and faster task than the adaptation of the $K\,N$ first layer weights $\mathbb{J}$ (Riegler and Endres, 1999).

The student network is to learn a rule from examples. For the purpose of modeling it is convenient to parameterize the unknown rule in terms of a teacher network. Let $\mathbb{B} = (\mathbf{B}_1, \mathbf{B}_2, \ldots \mathbf{B}_M)$ denote its M hidden unit weight vectors with $y_m = \mathbf{B}_m \cdot \boldsymbol{\xi}$. This teacher net implements the mapping $\Sigma_B(\boldsymbol{\xi}) = f_{out}(\sigma_B^1, \sigma_B^2, \ldots \sigma_B^M)$ where $\sigma_B^m = f_{hid}(y_m)$. For simplicity we will assume that the activation functions are the same in student and teacher.

We also restrict our attention to the thermodynamic limit, where $N \to \infty$, but K and M remain finite. Note that in this simplified frame it is still possible to model a variety of scenarios, including the case of matching complexity ($K = M$) where the rule is perfectly learnable. Obviously, the student can only approximate the rule for $K < M$, whereas the network is

over–sophisticated if $K > M$.

The learning of a linearly separable rule with a so–called single layer perceptron,

has served as a prototype model in the development of the entire field (see Statistical Mechanics of Generalization). This scenario is recovered for $K = M = 1$ with threshold hidden activation $f_{hid}() = \text{sign}()$ and with the output transfer function f_{out} being the identity.

A set of weights $\mathbb{J}^{\mu}$ is chosen after presentation of $\mu - 1$ examples of the form $(\boldsymbol{\xi}_{\nu}, \Sigma_{\nu})$. Here $\boldsymbol{\xi}_{\nu}$ is the input vector drawn at discrete time ν from a distribution $P(\boldsymbol{\xi}_{\nu})$ and Σ_{ν} is the corresponding training label. The latter can coincide with the true rule output $\Sigma_B(\boldsymbol{\xi}_{\nu})$ if reliable example data is available, or it is a noisy version thereof in the presence of stochastic corruption. The aim of learning is to determine $\mathbb{J}^{\mu}$ such that the teacher and student outputs are close in some respect for a novel, random input $\boldsymbol{\xi}$. A quite natural choice of an error measure is the quadratic deviation from the correct rule output $e(\boldsymbol{\xi}) = (\Sigma_J(\boldsymbol{\xi}) - \Sigma_B(\boldsymbol{\xi}))^2/2$. The expected value of this quantity with respect to $P(\boldsymbol{\xi})$ is termed the generalization error $e_G = \langle e(\boldsymbol{\xi}) \rangle_{\xi}$. Achieving small values of e_G is aimed at by making small changes in $\mathbb{J}^{\mu}$ at each time step.

Although there is quite a liberty in the choice of the training prescription, some common features are shared by all reasonable algorithms. Here, we will consider updates of the student vectors which can be written in the form

$$\mathbf{J}_k^{\mu} = \mathbf{J}_k^{\mu-1} + \frac{1}{N} F_k \cdot \boldsymbol{\xi}_{\mu} \tag{1}$$

upon presentation of example $(\boldsymbol{\xi}_{\mu}, \Sigma_{\mu})$. The central quantities in defining the algorithm are the *modulation functions* F_k which determine the amount of correction in the direction of the latest example. Often, the update term $F_k \, \boldsymbol{\xi}_{\mu}$ can be written as the gradient w.r.t. $\mathbf{J}_k$ of an instantaneous cost function defined for the latest example. Because of the similarity to pioneering ideas about learning in biological neural networks, algorithms of type (1) have been termed (generalized) *Hebbian learning.* Modulation functions which vary between different components of the student vectors (*matrix update*) can also be considered, see for example (Murata; Bös and Amari, in: Saad, 1998).

The modulation F_k can be a function of several quantities. It utilizes information about the current example and the actual student weights, usually. More sophisticated algorithms may depend on several other variables, which play the role of hyper–parameters. In general they reflect the fact that the amount of information contained in an example depends strongly on the state of the student network. Learning can be introducing so–called hyper–parameters to the algorithm which estimate, among other things, the performance of the network, noise levels affecting the data, or parameters that describe the input distribution. Learning will then be efficient, in some cases even as fast as optimal off–line training.

In order to obtain a macroscopic description of the dynamics one can project the iteration equation

(1) onto $\mathbb{J}$ and $\mathbb{B}$, which immediately yields a set of equations for the overlaps

$$Q_{ij} = \mathbf{J}_i \cdot \mathbf{J}_j$$

and $\mathrm{R}_{in} = \mathbf{J}_i \cdot \mathbf{B}_n$.(2)The set of (usually) non–dynamical parameters $T_{mn} = \mathbf{B}_m \cdot \mathbf{B}_n$ is also introduced here for future reference. In the simplest models, all considered inputs are independent random vectors with uniform density on the sphere $\boldsymbol{\xi}^2 = N$ (referred to as isotropic data in the following). Then, no further directions in $I\!R^N$ and therefore no other projections are relevant.

Performing the projections brings up great simplifications: first, the number of equations drops from $N\,K$ to only $(K^2 + K)/2 + KM$. Secondly, by means of the law of large numbers, the overlaps (), but not their changes, become *self–averaging* quantities in the thermodynamic limit for a large class of algorithms and input distribution. Mean values, as observed in a computer simulation for instance, will approach the theoretical prediction with increasing N. In many example scenarios, the deviation is found to be on the order $1/N$, yielding excellent results for fairly small systems with, say, $N \sim 100$ already. At the same time, the width of the distribution of quantities (), emerging from the randomness in the data, tends to zero with $N \to \infty$. This property is a common feature of so–called order parameters describing disordered systems in SM. A more detailed discussion of self–averaging in on–line learning is given in (Reents and Urbanczik, 1998).

Finally, the iteration equations for the overlaps reduce to first order ODE, where the continuous parameter $\alpha = \mu/N$ measures the learning time. Averaging over the last example input $\boldsymbol{\xi}_\mu$ and the possibly noisy label Σ_μ leads to the set (indices μ omitted)

$\dot{R}_{in} =< y_n F_i >, \qquad \dot{Q}_{ij} =< x_i F_j + x_j F_i + F_i F_j >,$ (3)where the dot denotes a derivative w.r.t. α. The r.h.s. reduce to multi–dimensional Gaussian integrals if the dependence of the F_k on $\boldsymbol{\xi}_\mu$ is only through the projections $x_k^\mu = \mathbf{J}_k^{\mu-1} \cdot \boldsymbol{\xi}_\mu$ and $y_m^\mu = \mathbf{B}_m \cdot \boldsymbol{\xi}_\mu$. By means of the Central Limit Theorem, these quantities become correlated Gaussian random numbers in the thermodynamic limit. Hence, their statistics is completely determined by first and second moments, which read in the case of isotropic data

$$\langle x_i \rangle = \langle y_m \rangle = 0, \ \langle x_i x_j \rangle = Q_{ij}, \ \langle y_m y_n \rangle = T_{mn} \quad \text{and} \quad \langle x_i y_n \rangle = R_{in}. \quad (4)$$

The set of differential equations () can now be integrated, at least numerically, and yields the evolution of R_{ij}, Q_{mn} from arbitrary initial conditions at $\alpha = 0$. Another advantage of the projection is that the generalization error $e_G = \langle e(\boldsymbol{\xi}) \rangle_{\xi}$, can be written as a function of the overlaps (). The particular form depends on the transfer functions, distribution of examples and the considered architectures, obviously.

The formalism outlined here allows to address a variety of problems: What is the typical behavior of standard algorithms such as Rosenblatt's Perceptron prescription or Backpropagation in the soft–committee? Which is the best possible decrease of the generalization error and by what choice of F is it achieved?

We will discuss a few selected results briefly in the following. The next section addresses the learning of classification schemes.

Then, regression by means of on–line gradient descent and related methods is discussed and finally we summarize some of the many other topics that have been investigated.

LEARNING OF CLASSIFICATION SCHEMES

The study of binary classifiers presents one of the simplest approaches to categorization. On-line learning of classifications has been investigated for several architectures ranging from the single layer perceptron learning a linearly separable rule, to fully connected feed–forward machines with $(K, M > 1)$, but also for density estimation and unsupervised learning. We will restrict ourselves to the case of supervised learning of a Boolean rule in student/teacher scenarios.

For binary outputs $\Sigma_{J,B} = \pm 1$ it is natural to choose the measure $e(\boldsymbol{\xi}) = (\Sigma_J - \Sigma_B)^2/4 \in \{0, 1\}$ which differs from the above introduced only by a factor 1/2. Its expected value e_G is the probability for disagreement between student and teacher w.r.t. a novel random input. As an example, for the single layer perceptron $(K = M = 1)$ one obtains $e_G = \arccos(R/\sqrt{Q})/\pi$ for isotropic inputs.

Instead of a detailed review of results for each particular network architecture and learning scenario we will describe the properties that are common to successful learning algorithms in terms of the single layer perceptron. The modulation function F, the central quantities in this analysis, serves as an estimate of the value of the information carried by the example. Different algorithms make different attributions to this value which leads to differences in performance.

The simplest choice is pure *Hebbian learning*, where $F = \Sigma$ is given by the training label, independent of the actual student weights. This recipe seems to work in the sense that for isotropic inputs the generalization error decreases as $e_G \propto \alpha^{-1/2}$ if the rule is linearly separable. This hides the fact that it will fail to learn completely for non–uniform distributions. The reason can be tracked to the equal treatment of correctly and incorrectly predicted examples. Rosenblatt's Perceptron rule, by only paying attention to errors and

disregarding correctly predicted examples, goes further. The resulting algorithm, $F\,\Sigma_B(\boldsymbol{\xi}) = \Theta(-x\Sigma(\boldsymbol{\xi}))$, i.e. 1 (if wrong) or 0 (if correct), is able to deal with non–uniform distributions, although in the singular isotropic case it is not as fast, $e_G \propto \alpha^{-1/3}$ for large α, as the pure Hebbian prescription (Biehl and Riegler, 1994).

The Perceptron rule can be improved by noting two facts. First, not all errors are equally important. Those made on examples far from the decision border should be penalized more than for examples near it. The on-line relaxation algorithm with $F = -x\,\Theta(-x\Sigma(\boldsymbol{\xi}))$ leads to a dramatic improvement with $e_G \propto \alpha^{-1}$ (Biehl and Riegler, 1994). The second fact is that correctly predicted examples may also contain useful information. Their inclusion is, however, not always desired. At the beginning of a learning session, they carry useful information and lead to improved performance, however their importance decreases as the student net starts to approach the rule. The lesson is that successful algorithms F should be performance dependent.

The effect of noise in the data has been intensely investigated, see (Copelli and Caticha, in: Saad, 1998) for references. The analysis above suggests that algorithms work by constructing a modulation function from information about the *surprise* carried by an incorrectly predicted example and by deciding whether that surprise is expected or not, given the state of the learning process. This strategy is most useful if determined by an estimate of the performance. In the presence of noise a new ingredient enters: the suspicion that a misclassification is due to noise in the data should suppress large modulations. There is therefore the need for a crossover from a regime of surprise to one of *student confidence* where blame attribution for prediction errors shifts to the data. This idea is, for instance, implemented in the Online Gibbs Algorithm (Kim and Sompolinsky, 1996) through a cut–off the function F at large absolute values of x.

For those cases where learning conditions permit the use of queries, Kinzel and Rujan (1990) have shown that improved performance is achieved by selecting examples at the very edge of the student decision border. To our knowledge, their work presents the first description of on–line learning and generalization by means of ODE. Appropriate choices of the actual algorithm F lead their query method to an exponential decay of e_G.

All these ingredients have been introduced during the past decades to the art of building algorithms. They can also be obtained from first principles for a set of model problems by the variational optimization of e_G w.r.t. to the modulation function. The variational method was introduced in (Kinouchi and Caticha, 1992) with an application to the pure perceptron problem, where $|de_G/d\alpha|$ is to be maximized. It has been extended by several authors but is restricted to working in the limit $N \to \infty$. The obtained optimal modulation function depends typically on the actual performance e_G of the student. Further, it may require knowledge of other

unavailable quantities describing the noise or the input distribution, for instance. Obviously, this limits the direct applicability of the prescription in practice. However, the procedure does (a) suggest which are the relevant features of a successful algorithm and (b) enable to establish lower bounds to $e_G(\alpha)$ in model situations.

The key result in this context is that on–line learning of classifications can be as efficient as the much more involved

memory based training (Kinouchi and Caticha, 1992; Opper, 1996; Van den Broeck and Reiman, 1996). For the single layer perceptron, e.g., the asymptotic decay of $e_G \propto 1/\alpha$ with $\alpha \to \infty$ differs only by a factor of 2 from the result for optimal off–line learning if examples are noiseless or corrupted by random inversion of the labels. In the case of continuous corruption, e.g. through additive noise in the inputs, the optimal performances of off–line and on–line training coincide exactly and $e_G \propto \alpha^{-1/2}$ (see Statistical Mechanics of Generalization). Recently the relation of the variational approach with Bayesian analysis was clarified (Opper, 1996; Solla and Winther; Opper; in: Saad, 1998).

LEARNING IN CONTINUOUS NETWORKS

In applications, ANN with continuous activation are frequently used to solve regression problems. But also for classification it is common practice to either approximate the rule by a *soft* or *probabilistic* version thereof. One reason for doing so is that apealing, simple training schemes are available for multilayered architectures. In networks with differentiable activations the error measure itself is a differentiable function of the network parameters. Therefore, off–line training can be based on gradient descent or similar minimization procedures. The most popular prescription of this kind has been termed *Backpropagation of Error* and its success has boosted the interest in ANN applications.

As an example architecture, the so–called soft–committee machine has been considered (Saad and Solla, 1995), with sigmoidal $f_{hid}(x) = \mathrm{erf}(x/\sqrt{2})$ and a single linear output with weights fixed to one. Training is based on the quadratic error for the latest example, i.e.

$$F_k \, \boldsymbol{\xi}_\mu \;=\; -\,\eta \;\nabla_{J_k}$$

e$(\boldsymbol{\xi}_\mu)$ $_{(\mu-1)} in Eq.$ (1)$, where the gradient$

is evaluated with the weights at time $\mu - 1$. The dynamics () as well as the generalization error can be worked out analytically for arbitrary K and M (but with $K, M \ll N$).

The learning rate η appears quadratically in the equations of motion (), hence its non–trivial influence on the behavior of the system. For example, a critical value η_c exists in matching scenarios, above which the perfect solution $\mathbb{J} = \mathbb{B}$ becomes unstable and even very small deviations from the perfect solution would increase in the training process. For $K < M$ the asymptotic value $e_G(\alpha \to \infty)$ depends explicitly on η, and only in the limit $\eta \to 0$ the optimal approximation of the rule is achieved.

One of the most interesting phenomena in on–line gradient descent is the breaking of permutation symmetry during learning. It can be observed for $K = M = 2$ already: Fig. 1 shows a typical learning curve obtained in this scenario when training is based on reliable examples with isotropic inputs. Obviously , the learning process is dominated by a pronounced plateau state in which hardly any progress is made while the number of examples increases. Only after an extended period of time, the system leaves the plateau and approaches its asymptotic state exponentially fast.

The occurrence of plateaus can be related to fixed points of the ODE (). In the case displayed in Fig. 1, the system is very close to a perfectly symmetric configuration with $\mathbf{J}_1 = \mathbf{J}_2$, i.e. $R_{ij} = R$ and $Q_{ij} = Q$ for all i, j. The existence of such a stationary state reflects an underlying symmetry of the problem: the student output and hence the error measure is invariant under permutations of hidden units. Whereas this property leads to phase transitions in equilibrium off–line training (see Statistical Mechanics of Generalization), the effect is here that the system approaches a symmetric state from generic initial conditions.

Properties of such plateaus can be investigated in detail by linearizing the dynamics close to the fixed point. The configuration is unstable, i.e. weakly repulsive in a few directions in overlap space. Consequently, the system stays in its vicinity for a long time before leaving the plateau, eventually. The observed plateau length

depends on the corresponding eigensystem and, in a subtle way, on the initial state. If the student weights are uncorrelated with the teacher before training one expects random $R_{ij}(0) = \mathcal{O}(N^{-1/2})$ in a finite system. As a consequence, the plateau length and hence the characteristic learning time will be $\alpha \propto \ln N$, demonstrating the relevance of the phenomenon to the practical treatment of high–dimensional data (Biehl et al., 1996).

The occurrence of plateaus in learning curves is by no means a feature of the particular scenario considered here. Weakly repulsive fixed points are found for any $K \geq 2$, independent of M. They exist also in systems where the invariance w.r.t. permutations is replaced by more complicated symmetries. Further, the number of fixed points increases with both K and M, while only the most symmetric states seem to dominate the dynamics from generic initial conditions (Saad and Solla, 1995; Biehl et al., 1996). An interesting observation is the possible existence of attractive fixed points with $e_G > 0$ when $K > M$. Successful training can be impossible if the student is highly over–sophisticated for the task to learn.

The deeper reason for the occurence of plateaus is the fact that learning requires the specialization of hidden units. After presentation of only a few examples, it is only reasonable that the student's hypothesis about the rule is as simple as possible. In this sense, unspecialized plateaus can be considered a necessary phase of learning and should be expected quite generally. However, from a practical point of view the question arises how the length of this phase can be reduced, potentially to its minimum. To this end, variational methods have been applied to the soft–committee machine as well. Note

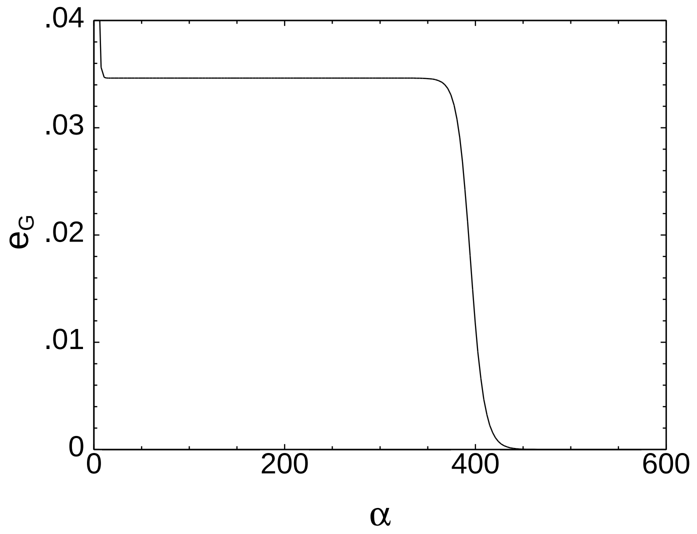

Figure 1: The learning curve of the $K = M = 2$ soft–committee scenario (as described in the text) with $T_{mn} = \delta_{mn}$, $\eta = 1.5$ and initial values $Q_{ii}(0) = 0.5$. All $R_{in}(0)$ as well as $Q_{12}(0)$ were different numbers of order $\mathcal{O}(10^{-12})$.

that the situation is more complicated here than it is in the perceptron or non–overlapping architectures: e_G is a function of many order parameters and it is not a priori clear that the local maximization of $|de_g/d\alpha|$ yields a successful strategy. Alternatively, the global optimization of e_G achieved after presentation of a certain number of examples has been considered. Both approaches predict that an efficient decrease of the plateau length should be possible (Saad and Rattray, 1997; Vicente and Caticha, 1997). The latter is also true for methods which explicitly incorporate curvature information into the learning algorithm. The so–called Natural Gradient method of Amari deserves particular attention in this context, see (Rattray and Saad; Bös and Amari, in: Saad, 1998). Its main idea is to base learning on a distance measure which is invariant under re–parameterization of the student output.

DISCUSSION

The SM approach to on–line learning has allowed for a theoretical understanding of the dynamics of various adaptive systems. Here, only selected examples for the success of this method could be discussed. We would like to conclude by mentioning only a few topics that we could not cover. For a more complete overview of the field and recent developments see e.g. (Saad, 1998).

The question of on–line learning in systems with discrete degrees of freedom has been addressed recently (Solla and Winther, in: Saad, 1998). A treatment of network parameters which can be trained on different time scales is discussed in (Riegler and Endres, 1999).

The framework of on–line learning and many of the phenomena discussed here carry over immediately to unsupervised learning in high–dimensional spaces, see (Biehl et al., in: Saad, 1998) for further references.

In a related, alternative approach to on–line learning one preserves the stochastic nature of the Markov process and refrains from taking $N \to \infty$ beforehand (e.g. Heskes and Wiegerinck; Leen, in: Saad,1998). By means of appropriate expansions and approximations one can then, for example, investigate the dynamics of finite systems. The convergence of on–line algorithms is discussed within the framework of stochastic approximation theory for instance in (Bottou, in: Saad, 1998).

A question of great importance is that of correlations among the training examples and their influence on the learning dynamics. In this context, the training from a limited pool of fixed examples (Barber and Sollich; Coolen and Saad; Heskes and Wiegerinck, in: Saad, 1998) remains one of the theoretical challenges with great practical relevance.

REFERENCES

Amari, S., 1967, A theory of adaptive pattern classifiers, IEEE Trans. Elect. Comp. EC–16, 299

Biehl, M. and Riegler, P., On–line learning with a perceptron, 1994, Europhys. Lett. 28, 525

Biehl, M., Riegler, P., and Wöhler, C., 1996, Transient dynamics of on–line learning in two–layered neural networks, J. Phys. A 29, 4769

Kim, J.W. and Sompolinsky, H., 1996, On–line Gibbs learning, Phys. Rev. Lett. 76, 3021

Kinouchi, O. and Caticha, N., 1992 , Optimal generalization in perceptrons, J. Phys. A 25, 6243

Kinzel, W. and Rujan, P. 1990, Improving a network generalization ability by selecting examples, Europhys. Lett. 13, 2878

Opper, M., 1996, On–line versus off–line learning from random examples: general results, Phys. Rev. Lett. 76, 4671

Reents, G. and Urbanczik, R., 1998, Self–averaging and On–line learning, Phys. Rev. Lett. 80, 5445

Riegler, P. and Endres, 1999, D. Learning dynamics on different timescales, J. Phys. A 32, 8655

Saad, D. (ed.), 1998, On–line learning in Neural Networks, Cambridge (UK): Cambridge University Press

Saad, D. and Solla, S.A., 1995, On–line learning in soft committee machines, Phys. Rev. E E 52, 4225

Saad, D. and Rattray, M., 1997, Globally optimal parameters for on–line learning in multilayer neural networks, Phys. Rev. Lett. **79**, 2578

Van den Broeck, C. and Reimann, P. 1996, Unsupervised learning by examples: on–line vs. off–line, Phys. Rev. Lett. **76**, 2188

Vicente, R. and Caticha, N., 1997, Functional optimization of online algorithms in multilayer neural networks, J. Phys. A 30, L599

Vicente R., Kinouchi, O., and Caticha, N., 1998 Statistical Mechanics of Online Learning of Drifting Concepts: a Variational Approach, Machine Learning 32, 179