STATISTICAL PHYSICS OF LEARNING FROM EXAMPLES: A BRIEF INTRODUCTION*

C. VAN DEN BROECK

Universitaire Campus, Limburgs Universitair Centrum
B-3590 Diepenbeek, Belgium

(Received January 14, 1994)

The problem of how one can learn from examples is illustrated on the case of a student perceptron trained by the Hebb rule on examples generated by a teacher perceptron. Two basic quantities are calculated : the training error and the generalization error. The obtained results are found to be typical. Other training rules are discussed. For the case of an Ising student with an Ising teacher, the existence of a first order phase transition is shown. Special effects such as dilution, queries, rejection , *etc.* are discussed and some results for multilayer networks are reviewed. In particular, the properties of a selfsimilar committee machine are derived. Finally, we discuss the statistics of generalization, with a review of the Hoeffding inequality, the Dvoretzky Kiefer Wolfowitz theorem and the Vapnik Chervonenkis theorem.

PACS numbers: 05.90. +m, 02.50. Le

1. Introduction

How to learn and predict new results on the basis of experience or available data clearly is a problem of tremendous practical importance. Our own survival depends largely on our ability to do so. Some of the learning and prediction tasks have proven to be very hard to tackle on the basis of a sequential program or of an artificial intelligence type of approach. Typical amongst those are the tasks performed by "experts". When asked about the road they have followed to reach a given conclusion, they are usually hard pressed to explain it. Parallel systems, such a neural networks, are offering a promising alternative. They are trained on examples rather than by an analysis of the rules that allow one to perform a given task. In this brief

* Presented at the VI Symposium on Statistical Physics, Zakopane, Poland, September 20–29, 1994.

introduction, we will illustrate how this problem — learning from examples — can be formalized and studied in detail for a few simple scenario's [1, 2, 3]. We will restrict ourselves to a very simple situation, namely that of binary classification. This problem corresponds to the mapping of input patterns into a binary $+/-$ or yes/no classification. The object that performs this mapping will be called a binary classifier. The learning problem can now be formulated as follows. One has at one's disposal a set of training patterns with their corresponding classification. The classifier is trained on this set. This is usually done by tuning the internal parameters $\underset{\sim}{J}$ of the classifier in such a way that a more or less correct classification is reproduced on the training set. The performance of the classifier can be characterized by 2 basic probabilities, namely the training error $\nu(\alpha)$, defined as the fraction of missclassified patterns of the training set, and the generalization error $\varepsilon(\alpha)$, defined as the probability for error on a random new pattern. These quantities typically depend on the amount of training (size of the training set) as indicated by the dependence on the training volume α. Our main purpose will be to either explicitly calculate those functions for specific scenario's or to derive general bounds or properties.

2. The perceptron revisited

2.1. The perceptron

The perceptron is one of the simplest classifiers, originally introduced by Mc Culloch and Pitts as a simplified model of a neuron, and publicized further by Rosenblatt in the late 1950's [4]. The architecture of the perceptron is schematically represented in Fig. 1. It is reminiscent of the structure of a neuron. $\xi_1, \ldots, \xi_n$ correspond to the input values at the n income gates (corresponding to the incoming electrical signals of other neurons). Each of these inputs ξ_i is weighted by its corresponding "synaptic efficiency" J_i, and the resulting signals are added in the "soma" of the perceptron. The total signal is then compared to a threshold value, and the output is set to $+1$ or -1 according to whether the threshold is exceeded or not. For the simple case of a zero threshold, the classification of an input pattern $\underset{\sim}{\xi}$ by a perceptron characterized by the weight vector $\underset{\sim}{J}$ is thus $\xi_{\text{out}} = \text{sgn}(\underset{\sim}{J} \cdot \underset{\sim}{\xi})$. The perceptron is a simple hyperplane classifier since the classification of all patterns $\underset{\sim}{\xi}$ is $+1$ if they lie on the same side as $\underset{\sim}{J}$ of the hyperplane orthogonal to $\underset{\sim}{J}$, and -1 otherwise. This classifier has some nice features. Its simplicity allows for a detailed analysis of various questions regarding learning from examples. Furthermore, it is a parallel classifier. Finally, the classification of a given pattern is not localized on any single weight vector component, hence one talks about distributed memory.

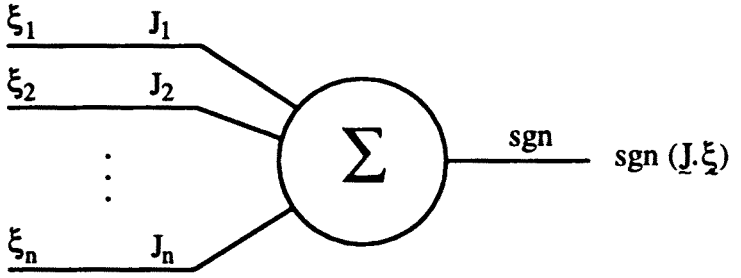

Fig. 1. Schematic representation of the perceptron classifier.

2.2. The Hebb rule

The idea of the Hebb training rule goes back to the principle of Pavlov coincidence training. In his famous experiment, Pavlov noted that when a dog is repeatedly given food at the same time as a light is flashing, he will also start to salivate when only the light is flashed. This indicates that a link has been established between food and light, such that when only the light is present it stimulates the thought about food. The Hebb rule [5] applies this idea of coincidence training right down at the level of individual neurons : it strengthens the connection between two neurons that fire together. When applied to the perceptron, the idea is to increase the connection J_i when both the input channel i (being the output of some other "neuron") and the neuron under consideration are active, i.e. when $\xi_i = +1$ and $\xi_{\text{out}} = +1$. To make the rule mathematically symmetric, we idealize the situation and use the updating rule $J_i^{\text{new}} = J_i^{\text{old}} + \xi_i \xi_{\text{out}}$. Applied to a number of patterns $\underset{\sim}{\xi}^{(\mu)}$, $\mu = 1, ..., p$, with corresponding classification $\xi_{\text{out}}^{(\mu)}$, with a tabula rasa $\underset{\sim}{J} = \underset{\sim}{0}$ starting value, one gets :

$$\underset{\sim}{J} = \sum_{\mu=1}^{p} \underset{\sim}{\xi}^{(\mu)} \xi_{\text{out}}^{(\mu)} . \tag{1}$$

In order to treat all patterns equally, one also assumes that the input vectors have been normalized, *i.e.* $|\underset{\sim}{\xi}^{(\mu)}|^2 = n$, $\forall \mu$. The Hebb rule has a number of advantages. It is explicit: no algorithm has to be applied to find the $\underset{\sim}{J}$-vector. It is additive, new patterns can be easily added. It has to a certain extent a neurophysiological basis. Finally it also has a simple geometric interpretation. Indeed, we mentioned that the perceptron is a hyperplane classifier. The problem to separate the patterns by such a hyperplane into those with $\xi_{\text{out}}^{(\mu)} = +1$ and those with $\xi_{\text{out}}^{(\mu)} = -1$, is equivalent to getting all the "renormalized" patterns $\underset{\sim}{\xi}^{(\mu)} \xi_{\text{out}}^{(\mu)}$ on one side of the hyperplane. A very reasonable prescription is then to take the hyperplane normal

on the center of mass of these vectors. This is precisely what the Hebbian rule is prescribing, *cf.* Eq. (1).

2.3. Learning from a teacher perceptron

We are now ready to attack the central problem in our discussion, namely that of learning from examples. The Hebbian perceptron receives a training set $\{\underset{\sim}{\xi}^{(\mu)}, \xi_{\text{out}}^{(\mu)};\ \mu = 1, \ldots, p\}$ of random patterns with their corresponding classification and constructs its $\underset{\sim}{J}$ vector accordingly, *cf.* Eq. (1). How well is this perceptron doing? Obviously, it is impossible to learn from examples if the underlying problem is random. The best one can do in such a situation is just to store the available data, but generalization is impossible. The storage capacity problem is an interesting problem in its own right, and has received a lot of attention [2, 6]. Here we will investigate another even more interesting situation, namely the one in which the underlying "teacher" that generates the training set has a very regular non-random structure. More precisely, we will assume that the pattern set is provided by a teacher perceptron with teacher vector $\underset{\sim}{T}$ [7]. Consequently, the $\underset{\sim}{J}$ vector of the student has the following form

$$\underset{\sim}{J} = \sum_{\mu=1}^{p} \underset{\sim}{\xi}^{(\mu)} \operatorname{sgn}\left(\underset{\sim}{T} \cdot \underset{\sim}{\xi}^{(\mu)}\right). \tag{2}$$

We also assume here and throughout the text that the pattern $\underset{\sim}{\xi}^{(\mu)}$ are chosen at random and independent of each other. From Eq. (2), we expect that there is a correlation between the orientation of the student $\underset{\sim}{J}$ and that of the teacher $\underset{\sim}{T}$, *i.e.* we expect that the student can learn from examples. In fact the 2 basic quantities that we mentioned in the introduction, the training error and the generalization error, can be calculated analytically in a so-called thermodynamic limit. In many applications, the number of input channels n (which can be thought of, for example, as the number of pixels of the input patterns) is very large. Consequently, one also expects that one needs a rather large number of training patterns, *i.e.* p large, in order to be able to generalize. The limit that one usually considers in the statistical mechanics literature on the subject is :

$$\left.\begin{array}{l} p \to \infty \\ n \to \infty \end{array}\right\} \text{ with } \alpha = \frac{p}{n} \text{ fixed}. \tag{3}$$

The nice feature about this limit is that the training and generalization error are then found to be self-averaging in most cases, *i.e.* independent of the choice of the training set with probability 1. For the above described

scenario of a Hebbian student perceptron trained by a teacher perceptron, the following results were found [8]:

$$\nu(\alpha) = \frac{1}{2} + \int_0^\infty dx \frac{e^{-x^2/2}}{\sqrt{2\pi}} \mathrm{Erf}\left(-x\sqrt{\frac{\alpha}{\pi}} - \frac{1}{\sqrt{2\alpha}}\right)$$
$$\varepsilon(\alpha) = \frac{1}{\pi} \arccos \frac{1}{\sqrt{1 + \frac{\pi}{2\alpha}}} . \tag{4}$$

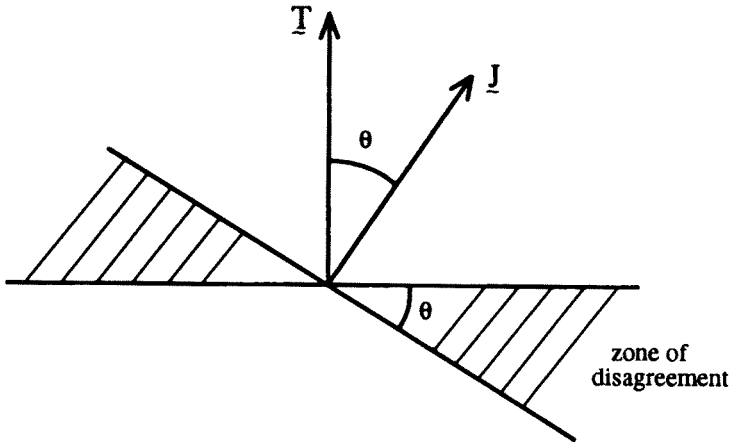

Fig. 2. The generalization error E depends only on the angle between the student perceptron $\underset{\sim}{J}$ and the teacher perceptron $\underset{\sim}{T}$.

To illustrate how one can find such results, we sketch the derivation for ε, the probability for disagreement between student and teacher on a random new question. As is clear from Fig. 2, this probability is equal to the angle between the $\underset{\sim}{J}$ and $\underset{\sim}{T}$ vector, divided by π. Hence

$$\varepsilon = \frac{1}{\pi} \arccos \frac{\underset{\sim}{J} \cdot \underset{\sim}{T}}{| \underset{\sim}{J} || \underset{\sim}{T} |} . \tag{5}$$

Because of the spherical symmetry of the model, one can choose the 1-axis of the n-dimensional input space along the teacher vector, *e.g.* $\underset{\sim}{T} = (1, 0, \ldots, 0)$. Furthermore, the quantities $\underset{\sim}{J} \cdot \underset{\sim}{T}$ and $| \underset{\sim}{J} |$ are found to be self-averaging on the basis of the law of large numbers. For example, one

has that:

$$\underline{J} \cdot \underline{T} = \sum_{\mu=1}^{p} \underline{\xi}^{(\mu)} \cdot \underline{T} \ \mathrm{sgn}(\underline{\xi}^{(\mu)} \cdot \underline{T})$$
$$= \sum_{\mu=1}^{p} \xi_1^{(\mu)} \ \mathrm{sgn} \ \xi_1^{(\mu)} = p\langle|\xi_1^{(\mu)}|\rangle + \text{ small fluctuation}.$$

Here we have used the property that the patterns $\underline{\xi}^{(\mu)}$ are independent of each other. Furthermore, we assume that the patterns are chosen at random. This can be achieved by choosing each component of the pattern vector independently of the others and according to a Gaussian law. From

$$P(\xi_1) = \frac{e^{-\xi_1^2/2}}{\sqrt{2\pi}},$$

we conclude that:

$$\underline{J} \cdot \underline{T} = p\sqrt{\frac{2}{\pi}}. \tag{6}$$

In the same way one finds:

$$| \underline{J} | = np + \frac{2}{\pi} p^2 . \tag{7}$$

Combining Eqs. (5)–(7) we recover the result given in Eq. (4).

One may wonder whether the extremely simplified situation presented so far leads to results which are representative of more realistic situations. It turns out that the results are in fact typical, so that the student teacher perceptron scenario provides a simple and didactic example of learning from examples. The training and generalization error are represented in Fig. 3a. Let us discuss some of their features. For small size of the training set, $\alpha \to 0$, one finds that $\varepsilon(\alpha) - 1/2 \sim \sqrt{\alpha}$. The generalization error starts at $1/2$, which corresponds to random guessing, and then decreases as $\sqrt{\alpha}$. We have conjectured that this is the fastest decrease of the generalization error that can take place for α small (see Section 3.2). Consider now the large α behavior. Both $\varepsilon(\alpha)$ and $\nu(\alpha)$ go to zero as $1/\sqrt{\alpha}$ in this limit. This is consistent with a general bound, based on the Vapnik–Chervonenkis theorem, that guarantees that $| \nu - \varepsilon |$ goes to zero at least as $\sqrt{\ln \alpha/\alpha}$ (*cf.* Section 4.4).

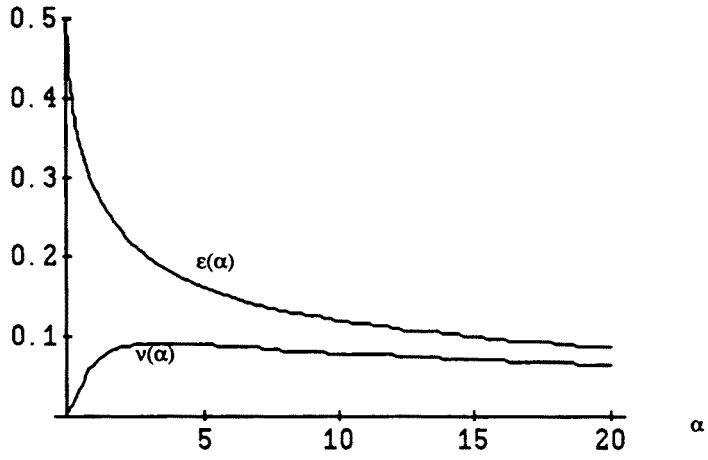

Fig. 3a. Training and generalization error for the Hebbian perceptron trained on random examples from a teacher perceptron.

2.4. Other training rules

One of the disadvantages of the Hebb rule is that it does not lead to a perfect learning of the training set, *i.e.* $\nu \neq 0$. Of course, nothing prevents us from picking at random a student that makes no error on the training set. The space of such students is sometimes called the version space, and the corresponding rule is the so-called Gibbs rule. For the above mentioned scenario, this rule leads to the following results [1, 9]:

$$\begin{aligned} &\nu(\alpha) = 0 \quad \text{(by construction)} \\ &\varepsilon(\alpha) \underset{\alpha\to\infty}{\sim} \frac{.62}{\alpha} \ \text{(Gibbs)}. \end{aligned} \tag{8}$$

Note the very important improvement over the Hebb rule, which is characterized by the weak convergence of ε to zero as $1/\sqrt{\alpha}$. On the other hand, the Gibbs rule is computationally more expensive, since one has to perform a search procedure to find a compatible student. The Gibbs rule should be compared with the best possible result that is available. This is realized by the Bayes rule. It corresponds to finding all the compatible students, letting them vote on the classification of a new pattern, and follow the majority rote [10]. The resulting generalization error is found to be

$$\varepsilon(\alpha) \underset{\alpha\to\infty}{\sim} \frac{.44}{\alpha} \quad \text{(Bayes)}, \tag{9}$$

which is barely better than the Gibbs results (namely by a factor of $\sqrt{2}$). Of course, the Bayes algorithm itself is completely impractical, but moreover

it seems that it is not even worth the effort. In general [11] one can actually prove that the asymptotic difference between Gibbs and Bayes does not exceed a factor of 2.

Finally, one can wonder about the "worst" student in the version space, in other words, what is the generalization error of the worst student which has learned the training set perfectly, $\nu(\alpha) = 0$. In this case, one finds [12]

$$\varepsilon(\alpha) \underset{\alpha\to\infty}{\sim} \frac{3}{2\alpha} \quad \text{(worst case)}. \tag{10}$$

Again, the difference with the Gibbs result is not very large.

Other training rules have been studied in the literature, see *e.g.* clipped Hebbian learning [13], projection rule [14], optimal perceptron [14], and nearest neighbour rule [15].

2.5. The Ising perceptron

We now focus on an interesting variant of the student-teacher perceptron scenario. Consider the situation in which both these perceptrons have weight vector components with binary values $J_i = \pm 1$ and $T_i = \pm 1$, $\forall i$, and let us focus on the Gibbs training scenario. Other scenario's are possible, see *e.g.* [13]. Because of symmetry we can choose $T_i = +1$, $\forall i$. In order to evaluate the generalization error, we have to evaluate the angle between student and teacher. Clearly this angle only depends on the number of weight vector components that are different between teacher and student. Consider those students for which the number of synapses $J_i = -1$ is $k = nx$. One has:

$$\frac{\underset{\sim}{T}\cdot\underset{\sim}{J}}{|\underset{\sim}{T}||\underset{\sim}{J}|} = \frac{-k+n-k}{\sqrt{n}\sqrt{n}} = 1-2x\,. \tag{11}$$

Each of these students has a generalization error given by, *cf.* Eq. (5):

$$\varepsilon(x) = \frac{1}{\pi}\arccos(1-2x)\,.$$

The number of such students is given by:

$$\Omega_0(x) = \binom{n}{k} \sim e^{n[-x\ln x-(1-x)\ln(1-x)]}\,, \tag{12}$$

where we have used Stirling's formula to estimate the asymptotically dominant contribution. One can now easily calculate the number of students $\langle\Omega_p(x)\rangle$ that survive, on average, $p = \alpha n$ random and independent training

examples. Indeed the probability that anyone of these students agrees with the teacher on p random and independent examples is $(1-\varepsilon)^p$. Hence:

$$\langle \Omega_p(x) \rangle = \binom{n}{k}(1-\varepsilon)^p \sim e^{\{n[-x\ln x-(1-x)\ln(1-x)-\alpha\ln[1-\frac{1}{\pi}\arccos(1-2x)]\}} . \tag{13}$$

The intuitive interpretation of this result is at the same time very simple and appealing. The number of students of type x that belong to the version space after $p = \alpha n$ examples is the result of an entropic factor, specifying how many such students are available in the first place, and an error (rather then "energy") factor taking into account that students very different from the teacher (x close to 1) are more likely to be eliminated by the training examples than students close to the teacher (x small). From the above result it is also clear that, in the limit $n \to \infty$, one specific type of student, *i.e.* one value x^*, will exponentially dominate in average numbers over the others, namely that value of x that maximizes the exponent. This value x^* depends on α, an therefore also the resulting observed generalization error $\varepsilon(x^*)$, *cf.* Fig. 3b. One amazing property is that the generalization error undergoes a first order phase transition at the value $\alpha \approx 1.448$, where it jumps abruptly from a value $\varepsilon \approx .257$ to $\varepsilon \equiv 0$. The explanation of this discontinuity is that there are too few students very similar to the teacher to survive the elimination by the training set, so that at $\alpha \approx 1.448$ only the teacher himself remains. The above transition was first reported by Gardner and Derrida [7]. It should however be stressed that we have implicitly assumed that the average $\langle \Omega_p(x) \rangle$ also gives the typical behavior, *i.e.* that $\Omega_p(x)$ is self-averaging. This turns out not to be the case. The correct self-averaging quantity is $\langle \ln \Omega_p(x) \rangle$. It can be evaluated using the replica technique [16]. It is found that the above presented (so-called annealed) calculation is not far off. The phase transition is actually taking place at a critical value $\alpha \approx 1.23$. Note that it is also easy to evaluate in the annealed calculation the behavior of the worst student of the version space. This student corresponds to an x-value for which the exponent in Eq. (13) is equal to zero, so that there is on the average exactly one such student left in the version space. The worst case generalization error is also given in Fig. 3b.

Finally, we comment on the surprising efficiency of the student perceptron in recognizing the teacher. To identify the teacher, one needs at least n bits of information, 1 bit per weight vector component. Each example of the training set provides at most 1 bit. It turns out that the overlap of information between these examples is not very large, since one only needs $p = 1.23n$ of them to reach the first order phase transition point that zooms in on the teacher.

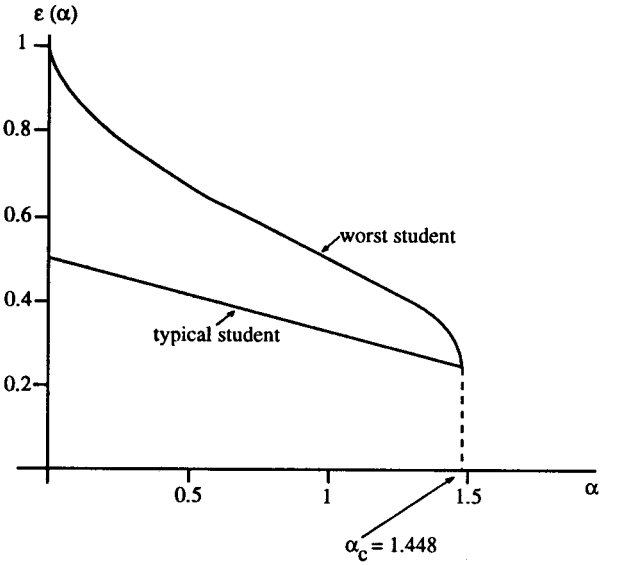

Fig. 3b. Generalization error of the typical and worst student for an Ising perceptron trained by an Ising teacher perceptron.

2.6. Special effects

In practical applications, all kinds of tricks and techniques are applied to improve generalization. Most of these special effects can be studied in the context of the student–teacher perceptron scenario. We mention a few of them here.

The student is allowed to ask its own questions, which are then called queries [17]. The generalization error is found to decrease, but the effect is not very dramatic.

The student is allowed to reject questions. This is of importance in situations in which high reliability is required. In the case of a perceptron $\underset{\sim}{J}$, the patterns $\underset{\sim}{\xi}$ that lie close to the decision boundary, *i.e.* $\underset{\sim}{J} \cdot \underset{\sim}{\xi}$ close to zero, are rejected. The generalization error is then reduced in a monotonous way as the rejection rate (which is the fraction of random questions that lie in the rejection zone around the decision boundary) increases [18].

Another important technique, especially in neural networks, is dilution, also called "optimal brain damage". In this case, the weight vector components that are not very "useful" (*e.g.* the ones with small effect on the output) are removed and the system is retrained [2]. Typically, the generalization error is reduced because the smaller number of parameters in the network reduces the danger of overfitting. This phenomenon can also be understood in the context of the Vapnik–Chervonenkis theorem.

We finally mention that several other modifications of the scenario have been studied, for example that of a non-stationary teacher, that of several generations of teachers, non-random or biased patterns, finite temperature learning, the case of a modest teacher, *etc.*

3. Multi-layer networks

3.1. Introduction

Interest in the perceptron waned towards the end of the 1960' when it was realized that its classification ability is very limited (it cannot implement the XOR), while it looked like it would be very hard to train more complicated networks with a multi-layer structure. With the discovery of the backpropagation algorithm [23] and the formation of the so-called PDP group (parallel distributed processes) the whole field, and in particular the feedforward multi-layer networks, received a great boost. The theoretical analysis of these networks however presents great difficulties, and only a handful of results have been obtained so far. The Vapnik–Chervonenkis theorem can be applied, but it gives bounds that lie way above the results that one obtains in practical applications. Clearly, there is still a lot of room for progress in this field. Instead of reviewing the abundant but rather disperse set of applications and theories, I will focus on two multi-layer networks that have been the object of my own research.

3.2. The self–similar committee machine

Imagine that one is dealing with a given binary classification problem, and that one disposes of a specific classifier with training algorithm. As before, we call $\varepsilon(\alpha)$ the generalization error of this classifier after being trained on a set of size α. Instead of doing this, however, we can partition the training set into $2N+1$ non-overlapping subsets, and use these to train $2N+1$ identical copies of the original classifier. Each of these is now less trained and its generalization error is typically larger and given by $\varepsilon\left(\frac{\alpha}{2N+1}\right)$.

In order to decide on the answer for a *specific input pattern*, we let each subclassifier vote, and we follow the majority rote. The resulting committee machine has a generalization error $\varepsilon'(\alpha)$ which is clearly given by

$$\varepsilon'(\alpha) = \sum_{k=N+1}^{2N+1} \binom{2N+1}{k} \left[\varepsilon\left(\frac{\alpha}{2N+1}\right)\right]^k \left[1-\varepsilon\left(\frac{\alpha}{2N+1}\right)\right]^{2N+1-k} . \tag{14}$$

By repeating this procedure, further test set partitioning followed by majority vote, one constructs a more and more complicated committee machine, with a generalization error that is found by iterating the functional

map (14). In the limit of an infinite number of such iterations, one obtains a self-similar committee machine with a generalization error being a fixed point of the map (14). We denote these "universal" errors by $\varepsilon_N(\alpha)$, with the subscript indicating the number of members in each subcommittee of the iteration. $\varepsilon_N(\alpha)$ obeys the following functional equation:

$$\varepsilon_N(\alpha) = \sum_{k=N+1}^{2N+1} \binom{2N+1}{k} \left[\varepsilon_N\left(\frac{\alpha}{2N+1}\right)\right]^k \left[1-\varepsilon_N\left(\frac{\alpha}{2N+1}\right)\right]^{2N+1-k}. \tag{15}$$

We first note that this equation cannot fix the unit in which α is being measured: when $\varepsilon_N(\alpha)$ is a solution, then also $\varepsilon_N(C\alpha)$ for any value of the constant C. In fact, it turns out that C absorbs all the information that is not specified, namely the generalization error $\varepsilon(\alpha)$ of the original classifier, the unit for α and the question that is being asked. Apart from this degeneracy, and apart from the symmetry $\varepsilon \Leftrightarrow 1-\varepsilon$, there exists a unique smooth solution of Eq. (15) [19]. In the limit $N \to \infty$, the explicit form of the fixed point solution is known:

$$\varepsilon_\infty(\alpha) = \tfrac{1}{2}\,\mathrm{Erfc}\,(\sqrt{\alpha}). \tag{16}$$

We now turn to the approach of the fixed point (also in the limit $N \to \infty$). One can prove that under iteration of Eq. (14), the error $\varepsilon(\alpha)$ of the original classifier will converge to the following universal error:

$$\varepsilon(\alpha) \to \varepsilon_\infty(C\alpha) \tag{17}$$

with

$$C = \lim_{\alpha \to 0} \left(\frac{\varepsilon(\alpha)-.5}{\varepsilon_\infty(\alpha)-.5}\right)^{1/2}. \tag{18}$$

Note that $C = 0$ or $C = \infty$ when the small α behavior of the original classifier does not match the small α behavior $\varepsilon_\infty(\alpha) - .5 \sim \sqrt{\alpha}$ of the universal error curve.

In fact, if one assumes that $\varepsilon(\alpha)$ decays faster than a $\sqrt{\alpha}$ law for small α, then it is found that our self-similar committee machine would reduce the error to zero ($C = \infty, \varepsilon_\infty(\infty) = 0$) !! We believe that this is impossible, leading us to the conjecture that the $\sqrt{\alpha}$-behavior is the best one possible. As we have seen above, this is also the result obtained for the Hebbian perceptron trained by a teacher perceptron!

3.3. Boolean networks

Another interesting example of a multi-layer network is a feedforward Boolean network [20, 21]. This network is built from logic gates like the AND-gate, XOR-gate, NOR-gate, *etc...*, which are linked together in a feedforward but otherwise random way. The input of each of the gates can be taken from the output of a previous gate or from the input nodes to the network. The output of the network is taken to be the output of the last gate. It turns out that the properties of this network are relatively insensitive to the number of logic gates that is used to construct it, once a minimum number of gates is used. The network has several very intriguing properties. Firstly, it displays a scaling law, analogous to Zipf's law [22]. Consider for example the case of 5 input nodes. Each of the input signals can be a 0 or 1, so that there are $2^5 = 32$ possible input signals. For each of these input signals the output can be either 0 or 1. Hence there are 2^{32} possible input-output tables. For every fixed configuration of the network (type and number of gates and their connectivity) the Boolean network realizes one of these tables. If the configurations are chosen at random, one finds that some of these tables appear quite often. This is illustrated in Fig. 4, where the frequency of appearance in function of the table under consideration is plotted. Most frequent are the tables whose output is always 0 or always 1. We rank these tables as $r = 1$ and $r = 2$, and call their frequency of appearance f_r. As one continues checking for the frequencies of less frequent tables, one observes that f_r follows an inverse power law in the ranking $f_r \sim r^{-\alpha}$. This is illustrated in Fig. 4 by a log-log plot of frequency versus rank.

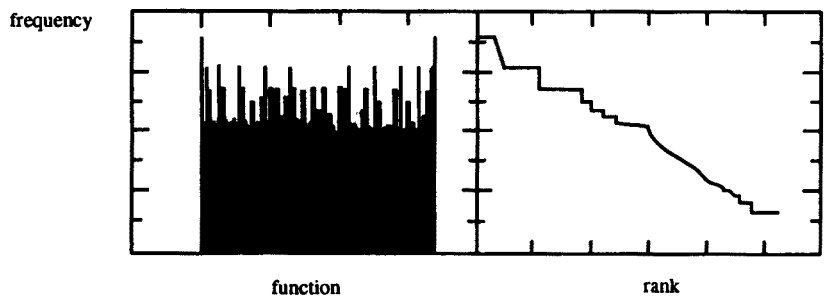

Fig. 4. Zipf's law for the frequency occurrence of Boolean functions in a random feedforward Boolean network. The Boolean functions are ordered according to the decimal representation of their output table, and according to their ranking in function of their decreasing frequency.

Another rather amazing property of the feedforward Boolean networks is their ability to learn a number of more mathematical tasks. These tasks

are typically rather difficult to learn on "conventional" neural networks. In Fig. 5, we give as an example the probabilities of exhaustive learning of the parity problem (output = parity of the inputs that are equal to 0) through the Gibbs rule, in terms of the fraction of examples that is used to train the system (fraction 1 = all examples = $2^{\#\text{inputs}}$). It is seen that, as one increases the number of inputs (in Fig. (5) from 3 to 7), the network learns faster and more abruptly, suggesting the existence of a genuine phase transition in the limit of a large number of inputs. It turns out that one can understand this phenomenon in the same way as that for the Ising perceptron discussed in section (2.5) [21]: the parity problem can be learned rather easily because, referring to the analysis of the Ising perceptron scenario, the parity teacher does not have a lot of students in its close vicinity. Due to their similarity of the teacher, these would actually be the most difficult to eliminate since they agree on many input patterns.

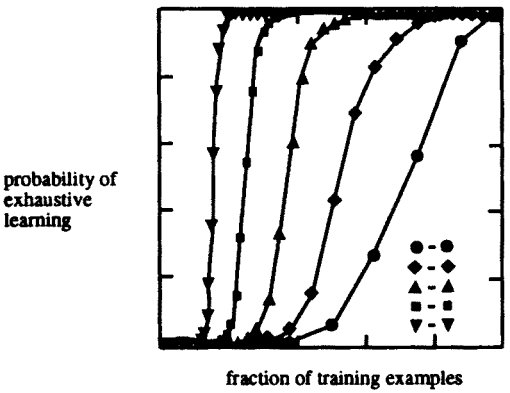

Fig. 5. Probability of learning the parity problem (of respectively 3, 4, 5, 6, and 7 input variables, the corresponding curves go from right, *cf.* the circles, to left, *cf.* the triangles) in function of the fraction of the total number of distinct training examples. Exhaustive learning is defined as the absence altogether of errors in the output table of the selected network configuration.

4. Statistics of generalization

4.1. Hoeffding inequality

One of the basic ways in which we generalize is to check the frequency of occurrence of a given event in a number of trials. In the optimal case these trials should be independent. For example, we grade a student in class on his performance on a number of test questions, and not on the

exhaustive set of questions. We hope that this grade reflects more or less the true capacity of the student. In doing so, we are actually applying the basic theorem of statistics that describes the convergence of frequency of an event to its probability. If the frequency $\nu(p)$ is obtained through a set of p identical and independent tests, then we know that $\nu(p)$ converges to the true probability, which we will denote by ε, in the limit $p \to \infty$. An elegant inequality that describes this convergence was derived by Hoeffding [24]:

$$\text{Prob}\left(\mid \nu(p) - \varepsilon \mid > \mu\right) \leq 2e^{-2\mu^2 p}. \tag{19}$$

More loosely speaking, one finds that $\nu(p) \approx \varepsilon +$ order $(1/\sqrt{p})$, a result which is of course closely related to the central limit theorem. To make the connection with generalization, we just have to state that the event under consideration is disagreement of the answer of the student with the answer of the teacher. ε is then the generalization error and $\nu(p)$ the probability for error on a test set. There is however a basic complication that arises and that prevents a direct application of the Hoeffding inequality in the context of learning from examples. It is not our purpose to evaluate the performance of a single student, but rather to estimate how well each student is doing in a whole class (possibly with a ∞ of students). In fact, we are typically interested in the best student. This student however is not know a priori but is selected using the test set. In other words, the test set becomes a training set. Since our selection of the best student is biased, we cannot directly apply the Hoeffding inequality. Rather we need to evaluate the following probability:

$$\text{Prob}\left(\sup_s \mid \nu_s(p) - \varepsilon_s \mid > \mu\right), \tag{20}$$

where ν_s and ε_s are the test and generalization error of a given student s. If we can bound the worst possible derivation between these quantities, then this bound can also be applied to any student that we decide to select.

4.2. The perceptron in 1d and the Dvoretzky-Kiefer-Wolfowitz theorem

As a first simple example, we consider the class of perceptrons defined on the unit interval [0,1] with threshold $x \in [0,1]$. Such a perceptron classifies an input pattern $y \in [0,1]$ as $\theta(y-x)$, where $\theta(x) = 1$ for $x > 0$ and 0 otherwise. On the other hand, we consider as teacher the classifier whose classification is identically 1 (i.e. it is the perceptron with $x = 0$). Consider now a test set of questions $\{y_i \mid i = 1, \ldots, p\}$. The test and generalization error of perceptron x are obviously given by:

$$\nu_x(p) = \frac{1}{p}\sum_{i=1}^{p} \theta(x - y_i) \tag{21}$$

$$\varepsilon_x = x\,.$$

In order to evaluate the worst case deviation, we would like to estimate the following probability

$$\text{Prob}\left(\sup_{x\in[0,1]} \mid \nu_x(p) - \varepsilon_x \mid > \mu\right), \tag{22}$$

where the probability is with respect to the random variables y_i. We now make the gratifying observation that this problem has been addressed and solved in an entirely different context, namely that of empirical distribution functions [24]. The aim of this theory is to estimate (cumulative) probability densities on the basis of a test sample. Consider a uniformly distributed random variable $x \in [0,1]$. Its cumulative probability is $F_x = x$. On the other hand, one disposes of a set $\{y_i \mid i = 1,\ldots,p\}$ of p independent samplings of this random variable. On this basis the following empirical distribution function can be proposed $E_x(p) = \frac{1}{p}\sum_{i=1}^{p}\theta(x - y_i)$. To estimate how well this distribution approaches the true distribution, one considers the maximum difference between them. There is a long history of research on the properties and statistics of this difference starting with a result by Kolmogorov in the 1930's. One of the most important results is the so-called Dvoretsky–Kiefer–Wolfowitz theorem, which is sometimes called the basic theorem of empirical statistics. It reads:

$$\text{Prob}\left(\sup_{x\in[0,1]} \mid E_x(p) - F_x \mid > \mu\right) \leq 58e^{-2\mu^2 p}\,. \tag{23}$$

Note the similarity of this bound with the Hoeffding inequality. This results guarantees that $E_x(p)$ converges to F_x uniformly $\forall x \in [0,1]$ in the limit $p \to \infty$ with deviations of the order of $1/\sqrt{p}$. It can furthermore be shown rather easily that the probability in the l.h.s. is actually independent of F_x, as long as this distribution does not contain any discontinuities.

Let us now turn back to our generalization problem. It is clear that $\nu_x(p) \equiv E_x(p)$ and $\varepsilon_x \equiv F_x$, hence the Dvoretsky–Kiefer–Wolfowitz bound can also be applied to our generalization scenario. We conclude that the difference between test and generalization error converges to zero roughly as $1/\sqrt{p}$ for the worst student. The test set can then also be used as a training set to select the "good" student, and the $1/\sqrt{p}$ can be applied to this student.

4.3. Vapnik–Chervonenkis classes

The Dvoretsky–Kiefer–Wolfowitz theorem provides a bound for the maximum deviation between test and generalization error for the whole

class of $1d$ perceptrons with threshold. The basic reason why such a result can be derived is that these classifiers are altogether not very different from each other. This rather vague concept of similarity can be made very precise by the introduction of the so-called Vapnik–Chervonenkis classes. Consider a class C of classifiers C. A given set of p patterns can in principle be classified in 2^p different ways (assuming again binary classification). To quantify the classification diversity of the set of classifiers, we can check how many of those 2^p classifications can be implemented. Typically one finds that for p small all classifications can be implemented, at least for one choice of p different patterns. However, when crossing a certain critical value of $p = d_{\mathrm{VC}}$, called the VC-dimension, not a single set of p patterns can be found for which all the classifications can be performed. An example for the hyperplane classifier in 2 dimensions is given in Fig. 6. All the

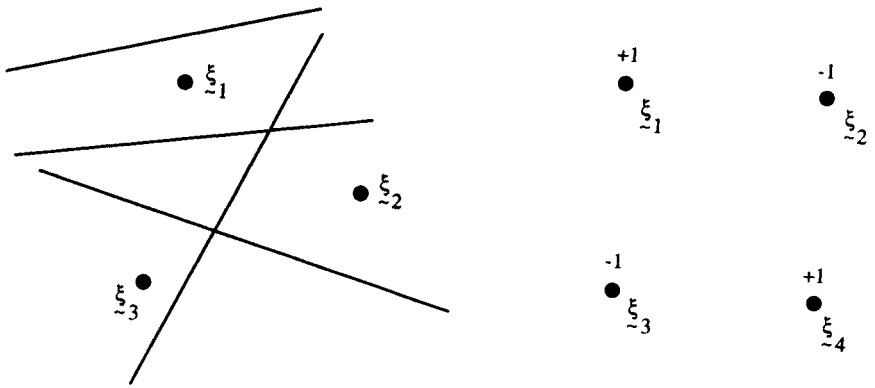

Fig. 6. All 8 classifications of 3 patterns in the plane can be implemented by a hyperplane (= line) classifier, but this is no longer the case for 4 patterns.

classifications of 3 input patterns can be performed, but 2 out of the 16 possible classifications of 4 input patterns can not be realized. Hence the VC dimension is $d_{\mathrm{VC}} = 3$. For perceptrons without threshold (hyperplanes though the origin) one finds that the VC dimension is d_{VC} = dimension n of the input space. For $p > d_{\mathrm{VC}}$ one can say that the classifiers of the class under consideration start resembling each other. In fact one can prove that the number of classifications $\Delta(p)$ that can (at most) be implemented for $p > d_{\mathrm{VC}}$ is bounded by a power law (Sauer's lemma) [25, 26]:

$$\Delta(p) \leq \sum_{k=0}^{d_{\mathrm{VC}}} \binom{p}{k} \leq \left(\frac{ep}{d_{\mathrm{VC}}}\right)^{d_{\mathrm{VC}}} \qquad p \geq d_{\mathrm{VC}}\,. \tag{24}$$

This property is the key ingredient to derive the "uniform" convergence of frequencies to probabilities for such a VC class.

4.4. Vapnik–Chervonenkis theorem

We are now ready to introduce the famous Vapnik–Chervonenkis theorem [25, 27]. Consider a class $\mathcal{C}$ of classifiers with VC-dimension d_{VC}. The following theorem can be proven [27]:

$$\mathrm{Prob}\big(\max_{C\in\mathcal{C}} \mid \nu_C(p) - \varepsilon_C \mid > \mu\big) \leq \Delta(2p)e^{-\mu^2 p}\,, \tag{25}$$

where $\nu_c(p)$ is the observed test error on p questions, and ε_C is the true generalization error of classifier $C \in \mathcal{C}$. Comparing this result with the Hoeffding's inequality, one sees that the main price to be paid for considering a class of classifiers is the prefactor $\Delta(2p)$ reflecting the classification "richness" of this class. Combining the above result with Sauer's lemma, one finds:

$$\mathrm{Prob}\big(\max_{C\in\mathcal{C}} \mid \nu_C(p) - \varepsilon_C \mid > \mu\big) \leq e^{d_{\mathrm{VC}}[\ln(2e\alpha) - \mu^2\alpha]}\,, \tag{26}$$

where we have introduced the scaled variable $\alpha = p/d_{\mathrm{VC}}$ which measures the size of the test training set in terms of the VC-dimension of the set of classifiers. This definition is consistent with the one introduced for the perceptron, where the VC-dimension is precisely equal to the number of inputs ($d_{\mathrm{VC}} = n$). The above result takes on a particularly pleasing form in the limit $d_{\mathrm{VC}} \to \infty$ [29]. Indeed, in this limit, the r.h.s. goes to zero for $\ln e\alpha < \mu^2\alpha$. Hence one can define a sharp accuracy threshold μ_c:

$$\mu_c = \sqrt{\frac{\ln(e\alpha)}{\alpha}} \tag{27}$$

such that with probability 1:

$$\max_{C\in\mathcal{C}} \mid \nu_C(p) - \varepsilon_c \mid \leq \mu_c\,. \tag{28}$$

This general result can immediately be compared with the explicit results for the Hebbian perceptron, trained by a teacher perceptron. In this case, it was found that ν and ε both decrease as $1/\sqrt{\alpha}$ for α large. This is in agreement with the VC bound. The Hebbian perceptron is of course just a specific perceptron, whereas the VC bound applies to any perceptron, and one can wonder whether the perceptron with the largest difference between ε_C and $\nu_C(\alpha)$ is also characterized by a $1/\sqrt{\alpha}$ behavior. It turns out

that this question can be investigated using rather involved "replica symmetry breaking" calculations which indicate that the worst difference goes as $\sqrt{\sqrt{\ln\alpha}/\alpha}$ [12].

The bound (25) is not the most interesting result in the context of generalization. Indeed one typically focuses not on the entire class of classifiers, but rather on those classifiers that perform very well on the training set. If the task that one is learning can indeed be performed by an element of $\mathcal{C}$, than one can always apply the Gibbs rule and pick a student from the version space, that is the subclass of $\mathcal{C}$ for which $\nu \equiv 0$. For this situation, the following variant of the VC-theorem can be proven [28, 29]:

$$\mathrm{Prob}\Big(\max_{C\in\mathcal{C}} \delta^{Kr}_{\nu_C,0}\varepsilon_C > \mu\Big) \leq 4\Delta(2p)2^{-\mu p}. \tag{29}$$

Again, a confidence threshold μ_c^* can be defined in the limit $d_{\mathrm{VC}} \to \infty$:

$$\mu_c^* = \frac{\ln(2e\alpha)}{\alpha \ln 2}, \tag{30}$$

which bounds the generalization error for all the students of the version space with probability 1:

$$\max_{C\in\text{version space}} \varepsilon_C \leq \mu_c^*. \tag{31}$$

This result predicts that the generalization error decreases at least as $\frac{\ln\alpha}{\alpha}$ for large α for students of the version space. This can be compared with the results of the perceptron scenario, where the typical (and in fact also the worst possible) result goes as $1/\alpha$.

5. Discussion and conclusions

We have illustrated how the concept "learning from examples" can be formalized. For specific scenario's, such as the student–teacher perceptron, the quantities of interest — the training and generalization error — can be calculated in detail. On the other hand, more general results can be obtained using theorems from statistics or through constructions and ideas from statistical mechanics and information theory. We stress again that our presentation is a very incomplete and preliminary introduction to the field. Probably, the more important discoveries still lie ahead. As promising fields for further investigation, I would like to cite unsupervised learning (that is in the absence of a teacher), learning in the presence of noisy data, learning through self-organization, and the more dynamic aspects of the learning and generalization process. Another important open question is the explanation

of why neural networks and the sort generalize much better than what can be expected on the basis of the VC-theorem (one observes good generalization for $\alpha \leq 1$!). We believe that recent advances and concepts from statistical mechanics (fractals, chaos, self-organized criticality) may turn out to play a crucial role in a deeper understanding of the phenomenon of "learning from examples".

REFERENCES

[1] H. Sompolinsky, N. Tishby, H.S. Seung, *Phys. Rev. Lett.* **65**, 1683–1686 (1990).

[2] J. Hertz, A. Krogh, R.G. Palmer, *Introduction to the Theory of Neural Computation*, Addison–Wesley, Reading (Massachusetts) 1991.

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

[4] F. Rosenblatt, *Psychol. Rev.* **65**, 386 (1958).

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

[6] E. Gardner, *J. Phys. A* **21**, 257 (1988).

[7] E. Gardner, B. Derrida, *J. Phys. A* **22**, 1983 (1989).

[8] F. Vallet, J.-G. Cailton, *Phys. Rev.* **A41**, 3059 (1990).

[9] G. Gyorgyi, N. Tishby, in *Neural Networks and Spin Glasses*, edited by K. Theumann and R. Köeberle, World Scientific, Singapore 1990.

[10] M. Opper, D. Haussler, in *Computational Learning Theory* Proceedings of the Fourth Annual Workshop, Morgan Kaufmann, San Mateo (California) 1991).

[11] D. Haussler, M. Kearns, R. Schapire, in *Computational Learning Theory* Proceedings of the Fourth Annual Workshop, Morgan Kaufmann, San Mateo (California) 1991.

[12] A. Engel, C. Van den Broeck, *Phys. Rev. Lett.* **71**, 1772 (1993).

[13] C. Van den Broeck, M. Bouten, *Europhys. Lett.* **22**, 319 (1993).

[14] M. Opper, W. Kinzel, J. Kleinz, R. Nehl, *J. Phys. A* **23**, L581 (1990).

[15] M. Bouten, C. Van den Broeck, Nearest neighbour classifier for the perceptron, preprint.

[16] G. Gyorgyi, *Phys. Rev.* **A41**, 7097–7100 (1990).

[17] W. Kinzel, P. Ruján, *Europhys. Lett.* **13**, 473 (1990).

[18] J.M. Parrondo, C. Van den Broeck, *Europhys. Lett.* **22**, 319 (1993).

[19] C. Van den Broeck, J.M. Parrondo, *Phys. Rev. Lett.* **71**, 2355 (1993).

[20] P. Carnevali, S. Patarnello, *Europhys. Lett.* **4**, 1199 (1987).

[21] C. Van den Broeck, R. Kawai, *Phys. Rev.* **A42**, 6210 (1990).

[22] G.K. Zipf, *Human Behavior and the Principle of Least Effort* Addison–Wesley, Reading (Massachusetts) 1949.

[23] D. Rumelhart, J. Mc. Clelland and the PDP Research Group, *Parallel Distributed Processes*, MIT Press, Cambridge (Massachusetts) 1986.

[24] G.R. Shorak, J.A. Wellner, *Empirical Processes with Applications to Statistics*, Wiley, New York 1986.

[25] V.N. Vapnik, A.Y. Chervonenkis, *Th. Prob. Appl.* **16**, 264–280 (1971).

[26] N. Sauer, *J. Comb Th.* **A13**, 145 (1972).

[27] V.N. Vapnik, *Estimation of Dependences Based on Empirical Data* Springer, Berlin 1982.

[28] D. Haussler, N. Littlestone, M.K. Warmuth. Predicting $\{0,1\}$-functions on randomly drawn points. Technical report, University of California, Santa Cruz 1990.

[29] J.M.R. Parrondo, C. Van den Broeck, *J. Phys. A* **26**, 2211 (1993).