Learning multi-class classification problems

Timothy L. H. Watkin, Albrecht Rau, Desiré Bollé, Jort van Mourik

▶ **To cite this version:**

Timothy L. H. Watkin, Albrecht Rau, Desiré Bollé, Jort van Mourik. Learning multi-class classification problems. Journal de Physique I, 1992, 2 (2), pp.167-180. 10.1051/jp1:1992131 . jpa-00246470

HAL Id: jpa-00246470
https://hal.science/jpa-00246470

Submitted on 4 Feb 2008

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Classification
Physics Abstracts
87.10 — 02.50 — 64.60

Learning multi-class classification problems

Timothy L. H. Watkin ([1]), Albrecht Rau ([1]), Desiré Bollé ([2]) and Jort van Mourik ([2])

([1]) Theoretical Physics, Oxford University, 1 Keble Road, GB-Oxford OX1 3NP, G.B.
([2]) Inst. voor Theor. Fysica, K. U. Leuven, B-3001 Leuven, Belgium

(received 26 August 1991, accepted in final form 20 October 1991)

Abstract. — A multi-class perceptron can learn from examples to solve problems whose answer may take several different values. Starting from a general formalism, we consider the learning of rules by a Hebbian algorithm and by a Monte-Carlo algorithm at high temperature. In the benchmark "prototype-problem" we show that a simple rule may be more than an order of magnitude more efficient than the well-known solution, and in the conventional limit is in fact optimal. A multi-class perceptron is significantly more efficient than a more complicated architecture of binary perceptrons.

1. Multi-class perceptrons for multi-class problems.

One recent success in the rapidly expanding field of neural networks has been the learning of a "rule" from examples (correct associations of "questions and answers") [1–7]. Analysis so far, however, has been restricted to small classes of possible rules and in particular to those in which there are only two answers, unlike the generality of real engineering problems [8–10]. In this paper we seek to relax this restriction in a way which preserves the natural symmetries of the problem.

Historically a formal neuron has been allowed two Ising-like states, ± 1, by analogy with the on/off nature of a biological neuron [11, 12]. A binary perceptron, for example, consists of N Ising inputs which determine one Ising output and may learn a set of random N-vector questions, with Ising components, and Ising answers associated according to some rule. In this way algorithms have been devised [1, 2] to teach a perceptron "linearly-separable" rules.

Clearly it is a matter of considerable engineering importance to discover how the success of learning schemes changes if the restrictions in this formulation are relaxed. In [5, 6] for example, successive questions are chosen not at random but on the basis of what has already been learnt, and recently a study has been made of how well perceptrons learn problems which are not linearly separable [6]. Here we consider another important generalisation to problems in which each digit of a question may take Q values and in which the answer to a problem is one of Q' possibilities.

Many scientific applications exist. Bohr and his collaborators [8], for example, have trained a network to predict with 70% accuracy the "secondary structure" of proteins from their local sequence of amino acids. In our notation the Q input states are the amino acids and the $Q' = 4$

output states correspond to the local stereo-chemical form of the amino acid chain: α-helix, β-turn, β-sheet or random coil. Another technically relevant issue is fault detection, where a network is expected to distinguish classes of technical defects [9]. There is also the classic neural network problem of classifying phonemes from frequency analysis of human speech [10].

May not many outputs be represented by the output combinations of several binary perceptrons? Previous studies have assumed that this is how multi-class classification would proceed, but there are two disadvantages. Firstly, a binary perceptron divides the input space in two and so must separate the possible outputs into two classes, which violates the symmetry of the problem: without *a priori* knowledge to the contrary we should assume that all answers are equivalent. Secondly, if a greater number of neurons is used more connections are required at some engineering cost; we would rather a single neuron performed a more complicated function of its inputs.

Such a neuron already exists in the literature and has been extensively studied in the very different problem of storing patterns [13–15]. Instead of two states in the output there are Q', each of which has the same relationship to each of the others, like the Q' vertices of a $Q'-1$ dimensional tetrahedron. Similarly there are Q equivalent states for each input. At every time step the neuron calculates a local "field" for each output state using a different, constant function of its inputs and enters the state with the highest field. We shall refer to such a neuron as a "multi-class perceptron". (In engineering a similar network is called a "linear machine", although it does not perform a linear function of its inputs, or a "winner-take-all" machine and is attributed to Nilsson [16]. Physicists have prefered the term "Potts-perceptron" [15].) This *multi-class* perceptron is not to be confused with a *multi-level* or *graded-response* neuron, which only possesses levels of activity between on and off; these states have a completely different symmetry (like a ladder) from that of the multiclass perceptron.

In this paper we will review the theory of learning from examples, explain the formalism of the multi-class perceptron and briefly describe the ways in which the two may be combined. We then apply multi-class perceptrons to the benchmark problems of learning a rule and of classifying prototyes. For the prototype problem we shall show that a simple learning rule (even with a binary perceptron) is better than the well-known previous solution, and in the conventional limits is actually optimal. One lesson will be that geometrical arguments may let us avoid the extremely difficult algebra of an algebraic statistical mechanics formulation. We shall generally find that in multi-class problems a multi-class perceptron is significantly more efficient than a more complicated architecture of binary perceptrons, but will finally discuss problems in which an even better choice of neuron may be appropriate.

2. The theory of multi-class learning.

2.1 BINARY PERCEPTRON LEARNING. — A conventional binary perceptron has N Ising inputs $\{S_i = \pm 1\}$, $i = 1, . \ , N$ and one Ising output S_o given by

$$S_o = U(\mathbf{S}) = \mathrm{sgn}(\mathbf{J} \cdot \mathbf{S}), \tag{1}$$

where we have defined the scalar product of two N-vectors $\mathbf{x}$ and $\mathbf{y}$ as $\mathbf{x} \cdot \mathbf{y} = 1/N \sum_i x_i y_i$. The N-vector $\mathbf{J}$ of weights defines the perceptron. If we choose $J_j \in \mathcal{R}$ it is a spherical binary perceptron, or if $J_j \in \{\pm 1\}$ an Ising binary perceptron. $\mathbf{J}$ is normalised to $\mathbf{J} \cdot \mathbf{J} = 1$, so that it lies on the surface of the unit sphere.

The perceptron learns from p pairs of N-vector questions $\{\boldsymbol{\xi}^\mu\}$, $\mu = 1, . \ , p$ and answers $\{\xi_o^\mu\}$, associated according to a rule V, defined as $\xi_o^\mu = V(\boldsymbol{\xi}^\mu)$. Each ξ_j^μ is chosen independently and randomly from the set $\{\pm 1\}$ and $p = \alpha N$, where α remains constant as $N \to \infty$. A

linearly-separable Boolean function is of the form

$$V(\boldsymbol{\xi}^\mu) = \mathrm{sgn}(\mathbf{B} \cdot \boldsymbol{\xi}^\mu), \tag{2}$$

where **B** is a "teacher" vector of unit magnitude. An Ising **J** is typically used to learn an Ising **B**, and a spherical **J** to learn a spherical **B** (assumed to be randomly chosen). If this is not the case and e.g. an Ising perceptron has to learn a spherical perceptron, the problem becomes unlearnable and has recently been analysed by [7]. From the examples we wish to construct a **J** which finds the $\xi_o^{\mu+1}$ for a random new question $\boldsymbol{\xi}^{\mu+1}$, and it is clear that $\mathbf{J} = \mathbf{B}$ fulfills this. The chance that **J** generates the right answer is the *generalisation ability* G (clearly a randomly chosen **J** will give $G = 1/2$ if $\xi_o^{\mu+1}$ has equal chance of being ± 1), and thus the goal of the training process is to minimize the *generalisation error* $\epsilon_g = 1 - G$. This, as has been recently pointed out [20, 23], is not necessarily equivalent to minimizing the number of examples which the perceptron gets wrong, the "training error"

$$E(\mathbf{J}) = \sum_\mu \Theta\left(-\xi_o^\mu \mathbf{J} \cdot \boldsymbol{\xi}^\mu\right). \tag{3}$$

The simplest learning algorithm [1] sets **J** according to a variant of the Hebb rule

$$\mathbf{J} = \frac{1}{\gamma\sqrt{N}} \sum_\mu \xi_o^\mu \boldsymbol{\xi}^\mu, \tag{4}$$

where γ is just a factor chosen to normalise **J** to 1. We will now present a simple geometric argument to analyse how the rule works. Figure 1 shows a projection of the N-dimensional space containing **J** and **B**. Randomly chosen $\{\boldsymbol{\xi}^\mu\}$ have a component, y, in any direction **y** (i.e. $y = \sqrt{N}\boldsymbol{\xi} \cdot \mathbf{y}$) which is Gaussian distributed with mean zero and standard deviation 1. The components of $\{\xi_o^\mu \boldsymbol{\xi}^\mu\}$ in the **B** direction add to **J** constructively, while those in any perpendicular direction do so randomly. Thus after presenting p examples the component of **J** in the **B** direction is $(\alpha/\gamma)\langle |\, y_B \,| \rangle = (\alpha/\gamma)\sqrt{2/\pi}$ (in the large N limit), while that in each of the perpendicular directions is $\sqrt{\alpha/N}/\gamma$ — the sum of a random walk of αN steps of length $1/N$. However, there are $N - 1$ directions perpendicular to **B** and to each other, and so the sum of all the components not in the **B** direction, the component of **J** perpendicular to **B**, is

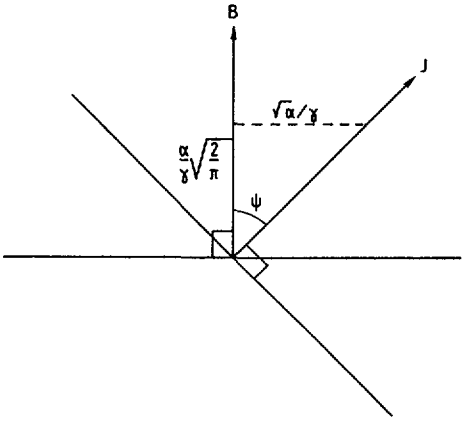

Fig.1. — Plane containing **B** and **J**, which lie at angle ψ.

found from Pythagoras to be $\sqrt{\alpha}/\gamma$. Thus, by Pythagoras, $\gamma = \sqrt{\alpha + \alpha^2 2/\pi}$. A well-known result is that $\epsilon_g = \frac{\psi}{\pi} = \frac{1}{\pi}\cos^{-1}(\mathbf{B}\cdot\mathbf{J})$, since, as may be seen from Figure 1, this is the fraction of the space whose overlap with $\mathbf{B}$ and $\mathbf{J}$ has different sign. It follows that

$$\epsilon_g = \frac{1}{\pi}\cos^{-1}\left(\frac{\alpha\sqrt{2/\pi}}{\sqrt{\alpha+\alpha^2 2/\pi}}\right) = \frac{1}{\pi}\tan^{-1}\sqrt{\frac{\pi}{2\alpha}}, \tag{5}$$

a result derived at some length in [1]. It implies that as $\alpha \to \infty$, the asymptotic behaviour of ϵ_g is $\epsilon_g \sim 1/\sqrt{\alpha}$. An alternative approach ("zero-temperature" learning) is to search the N-dimensional $\mathbf{J}$-space to find the volume which stores every pattern with a stability larger than κ, that is, for spherical $\mathbf{J}$,

$$\text{Volume} = \int \mathrm{d}\mathbf{J}\,\delta(\mathbf{J}\cdot\mathbf{J}-1)\prod_{\mu}\Theta(\xi_o^\mu\sqrt{N}\mathbf{J}\cdot\boldsymbol{\xi}^\mu - \kappa). \tag{6}$$

As κ is increased to a critical value the volume shrinks continuously to zero, centered on the $\mathbf{J}$-vector with "maximum stability", and algorithms exist which will construct this $\mathbf{J}$, notably the MinOver algorithm [19]. Reference [2] has shown that as $\alpha \to \infty$, ϵ_g tends to zero as $1/\alpha$. For Ising $\mathbf{J}$, however, for which the $\mathbf{J}$-space integral is replaced by a trace over all $\mathbf{J}$ configurations, a first-order transition (the thermodynamic transition) to perfect generalisation has been predicted within the framework of the "replica symmetric approximation" to occur at $\alpha_c = 1.245$. However the "one-step replica symmetry broken solution" raises the spinodal line for the vanishing of the metastable spin glass phase to $\alpha_{\mathrm{RSB}} = 1.628$ [17].

Analysis of the zero-temperature algorithm is complicated, however, and may be treated under the "annealed approximation", which approximates the training error by p times the generalisation error. If a Monte-Carlo algorithm is used to minimize the "training error" at a temperature T (with inverse β), (i.e. we use the error as an energy and generate states with Boltzmann probability $\exp\{-\beta E(\mathbf{J})\}$), then in the high temperature limit ("high-temperature learning") the annealed approximation is exact and the algorithm is equivalent to minimizing a free energy given by [3]

$$\beta f = \tilde{\alpha}\epsilon_g - s, \tag{7}$$

where $\tilde{\alpha} \equiv \alpha/T$ and s is the entropy (to find the appropriate entropy see, for example [7]). Since we are working at a high temperature a number of examples scaling with T is required, but otherwise the same qualitative behaviour has been observed as at low temperatures [3] (gradual learning for spherical perceptrons and first order transitions to perfect generalisation for Ising ones). Theoretical predictions made with the high-T formulation are found to agree with simulations for temperatures as low as $T \approx 5$. It has been noted [7] that in problems which are not linearly separable, and so may not be learnt exactly by a perceptron, high temperature learning may be the best way to avoid "overfitting" — giving undue significance to unrepresentative examples.

A fundamentally different sort of rule was studied in [20], the "prototype-problem". Instead of a teacher $\mathbf{B}$, we begin with p_0 random, uncorrelated Ising N-vector prototypes $\{\boldsymbol{\eta}^\mu\}$, $\mu = 1,\ldots,p_0$, where $p_0 = \alpha N$, each of which has a random Ising output $\{\eta_o^\mu\}$. The correct answer for any input $\mathbf{S}$ is now the correct output of the prototype closest in Hamming distance to $\mathbf{S}$. The rule is learnt from examples of each prototype $\{\boldsymbol{\xi}^{\mu l}\}$, $l = 1,\ldots,p$, chosen at random but with the constraint that $\boldsymbol{\eta}^\mu \cdot \boldsymbol{\xi}^{\mu l} = m$. For an extensive number of examples this problem is clearly unlearnable, since no plane can divide the input space to correctly answer every possible input. Instead [20] searched the $\mathbf{J}$-space to find the $\mathbf{J}$ minimizing the number of incorrectly answered examples, the training error, and considered only the limit of m small, which implies

large p, since p should be rescaled as $\tilde{p} = m^2 p/(1-m^2)$, which remains of order 1. For $\tilde{p}$ less than a critical value, $\tilde{p}_c$, a $\mathbf{J}$ may be found which makes the training energy zero; in this range the consequent generalisation error falls from 1/2 as $\tilde{p}$ rises, but then climbs slightly as $\tilde{p} \to \tilde{p}_c$, since the only $\mathbf{J}$ which correctly learns all examples has overfitting. For $\tilde{p} > \tilde{p}_c$ the training error rises smoothly and the generalisation error falls and both tend to the same value $\epsilon(m)$ as $\tilde{p} \to \infty$. Training with a finite training error (equivalent to a finite temperature) eliminates the problem of overfitting and the asymptotic behaviour of the generalisation error is $\epsilon_g - \epsilon(m) \sim \tilde{p}^{-1}$

2.2 THE MULTI-CLASS PERCEPTRON. — Only when the concept of a binary perceptron was generalised to many states did it become clear how many assumptions lie hidden in the ± 1 notation. We shall work within the formalism recently derived in [15] to define a multi-class perceptron with N Q-state inputs $\{\sigma_j\}; j = 1, \ldots, N$ and each $\sigma_j \in \{1, \ldots, Q\}$ and one Q'-state output $\sigma' \in \{1, \ldots, Q'\}$. The local field for output state s' is given by

$$h_{s'} = \sum_{j,s} J_j^{s's} m_{s,\sigma_j}, \tag{8}$$

where $m_{a,b} \equiv Q\delta_{a,b} - 1$ is the Q-fold *multi-class operator*, so that the synaptic matrix $J_j^{s's}$ is the weight of a signal coming from the input j which is in state s on the state s' of the processing unit. σ' is set equal to the state with the highest local field, i.e.

$$\sigma' = \left\{ s'_o | h_{s'_o} > h_{s'} \; \forall s' \neq s'_o \right\}. \tag{9}$$

Clearly the operations of this perceptron are invariant under the transformation

$$J_j^{s's} \to J_j^{s's} + u_j^s \; ; \; \forall s' \tag{10}$$

for any u_j^s, since it alters all the fields by the same amount and thus there exists a gauge freedom which we shall usually fix by enforcing

$$\sum_{s'} J_j^{s's} = 0, \; \forall \, s, j \, , \tag{11}$$

although all choices of fixing the gauge are of course equivalent. We shall also choose

$$\sum_{s} J_j^{s's} = 0, \; \forall \, s', j \, , \tag{12}$$

since any other choice is equivalent to adding a threshold to the field at state s'. Thus in the case of binary inputs, $Q = 2$, $J_j^{s'1} = -J_j^{s'2}$ for all s', j, so that taking $\sigma_j = \pm 1$ we may rewrite the field simply as

$$h_{s'} = \sum_j J_j^{s'} \sigma_j. \tag{13}$$

If the output is binary as well ($s' = \pm 1$) then, renaming $\mathbf{J}^1$ as $\mathbf{J}$ gives

$$h_{s'} = s' \sum_j J_j \sigma_j \tag{14}$$

and the s' maximising this expression is $\mathrm{sgn}(\mathbf{J} \cdot \boldsymbol{\sigma})$.

A *spherical* multi-class perceptron has the $\{J_j^{s's}\}$ chosen to be real numbers such that each vector $\{\mathbf{J}^{s's}\}$ is normalised to 1, but what is the natural multiclass analogue of an Ising binary perceptron? Should we allow the $\{J_j^{s's}\}$ to take many integer values? This point is considered further in the conclusion. In our present work, however, in which we shall only briefly use quantised interactions, we will choose each $J_j^{s's}$ to just be ± 1, which, for Q or Q' odd, meaṅs we must relinquish (11) and (12). We shall call the result an *Ising multi-class perceptron.*

The generalisation of a multi-class perceptron to N' outputs or to a fully connected network, for which $N' = N$ and $Q' = Q$, as in [13, 14], is straightforward.

2.3 MULTI-CLASS PROBLEMS. — A multi-class problem is one in which the answer may take several values. It is straightforward to extend the well-known proof for binary perceptrons [21] to show that a two-layer network of multi-class perceptrons may perform any logical function of its inputs. Here we will be concerned only with problems in which questions $\{\boldsymbol{\xi}^\mu\}$ — strings, N digits long with $\xi_j^\mu \in \{1, \ldots, Q\}$ — are associated with answers ξ_o^μ, where $\xi_o^\mu \in \{1, \ldots, Q'\}$.

A single multi-class perceptron can learn exactly problems which are the analogues of "linearly-separable" problems, i.e. there exists a $B_j^{s's}$ which associates the questions $\{\boldsymbol{\xi}^\mu\}$ with the answers ξ_o^μ via the dynamical rule (8,9), with $\{\mathbf{J}^{s's}\}$ replaced by $\{\mathbf{B}^{s's}\}$, and the $\mathbf{B}^{s's}$ obeying the same gauge fixings (11,12). We will use a spherical multi-class perceptron, when the $\{\mathbf{B}^{s's}\}$ are spherical and an Ising perceptron when the $B_j^{s's} = \pm 1$.

Generally multi-class problems are not linearly-separable, of course. A good example is the proximity-problem, in which prototypes may be associated with more than two outputs (this is discussed in more detail in section 4), and multi-class analogues also exist of all the unlearnable problems discussed in [6].

As the multi-class analogue of (4), following [13], we introduce:

$$J_j^{s's} = \frac{1}{\gamma^{s's}\sqrt{N}} \sum_\mu m'_{s',\xi_o^\mu} m_{s,\xi_j^\mu}, \tag{15}$$

where $m'_{a,b}$ is the Q'-fold multi-class operator and $\gamma^{s's}$ is introduced to normalise each $\mathbf{J}^{s's}$ to 1. For the case of $Q = 2$ this reduces to

$$\mathbf{J}^{s'} = \frac{1}{\gamma^{s'}\sqrt{N}} \sum_\mu m'_{s',\xi_o^\mu} \boldsymbol{\xi}^\mu. \tag{16}$$

These rules obey the gauge constraints (11) and (12), since they are true for every term of the sum over μ. However, since we know that the output is unaffected by the gauge fixing we can equivalently, for $Q = 2$, analyse

$$\mathbf{J}^{s'} = \frac{1}{\gamma^{s'}\sqrt{N}} \sum_\mu \delta_{s',\xi_o^\mu} \boldsymbol{\xi}^\mu, \tag{17}$$

which violates (11) but makes it clear that the $\{\boldsymbol{\xi}^\mu\}$ affect only the $\mathbf{J}^{s'}$ with $s' = \xi_o^\mu$. Thus noise in the $\{\mathbf{J}^{s'}\}$ is uncorrelated for different s'. Gauge constraint (11) can be enforced after learning by subtracting, from each $\mathbf{J}^{s's}$, $1/(Q'-1)$ times the sum of the other $\{\mathbf{J}^{s'}\}$, and the set $\{\mathbf{J}^{s'}\}$ should then be renormalised.

In the rest of this work we shall confine ourselves to $Q = 2$ (binary inputs) and consider varying Q'. This is for simplicity and in the belief that problems with many answers are more interesting than those in which inputs take several values (many-valued inputs may be represented anyway by combinations of binary inputs). However, it is straightforward to extend the arguments of the next two sections to higher values of Q.

3. Learning a learnable rule.

An algebraic evaluation of learning with rule (4) is rather difficult, but may be avoided by a generalisation of the geometric argument proposed in the last section to derive (5) . However, there is a different distribution of overlaps between $\mathbf{B}^{s'}$ and those $\{\xi^\mu\}$ whose answer is s', as may be seen from figure 2 which, for $Q' = 3$, shows the plane containing the $\{\mathbf{B}^{s'}\}$: examples must fall closer to $\mathbf{B}^{s'}$ to be assigned to it. The average of this quantity is

$$A(Q') \equiv Q' \Big\langle B^{s'} \ \xi^\mu \prod_{\tilde{s}' \neq s'} \Theta(B^{s'} \ \xi^\mu - B^{\tilde{s}'} \ \xi^\mu) \Big\rangle, \tag{18}$$

where $\langle . \rangle$ indicates the pattern average. This can be evaluated using the integral representation of the Heavyside function to give

$$A(Q') = \frac{Q'(-a)^{3/2}}{(1-a)} \int_0^\infty \left(\prod_{s'>1}^{Q'} \frac{d\lambda_{s'}}{\sqrt{2\pi}} \right) \left(\sum_{s'>1}^{Q'} \lambda_{s'} \right) \exp \left\{ -\frac{1}{2} \sum_{s'>1}^{Q'} \lambda_{s'}^2 - \frac{a}{2(1-a)} \left(\sum_{s'>1}^{Q'} \lambda_{s'} \right)^2 \right\}, \tag{19}$$

where $a \equiv \mathbf{B}^{s'} \ \mathbf{B}^{\tilde{s}'} = 1/(1-Q') \ \forall \tilde{s}' \neq s'$. The components of $\{\xi^\mu\}$ in perpendicular directions remain the same, however, in the large N limit. It follows that $\mathbf{J} \cdot \mathbf{B} = A\alpha/Q'$ (since only one example in Q' contributes to each $\mathbf{B}^{s'}$), where using Pythagoras the modulus is again given by $\gamma^2 = (A\alpha/Q')^2 + \alpha/Q'$. The components of $\{\mathbf{J}^{s'}\}$ perpendicular to the teacher vectors are, as explained, not correlated and so are effectively perpendicular in a high dimensional space. However, enforcing the gauge constraint, as explained at the end of section 2.3, correlates these other directions and gives

$$\mathbf{J}^{s'} \ \mathbf{B}^{\tilde{s}'} = \frac{\delta_{s',\tilde{s}'} + a(1-\delta_{s',\tilde{s}'})}{\sqrt{1 + \frac{Q'-1}{A^2\alpha}}} \tag{20}$$

with

$$\mathbf{J}^{s'} \ \mathbf{J}^{\tilde{s}'} = \mathbf{B}^{s'} \ \mathbf{B}^{\tilde{s}'} = \delta_{s',\tilde{s}'} + a(1-\delta_{s',\tilde{s}'}) \tag{21}$$

by symmetry. We define $R \equiv \mathbf{J}^{s'} \ \mathbf{B}^{s'}$ for all s'.

How are we to work out the generalisation error? The geometrical argument in section 2 comparing areas on a plane, which it was possible to generalise in [7] to areas on the surface of a sphere, is hard to apply here since the sphere containing $\{\mathbf{B}^{s'}\}$ and $\{\mathbf{J}^{s'}\}$ is $2(Q'-1)$

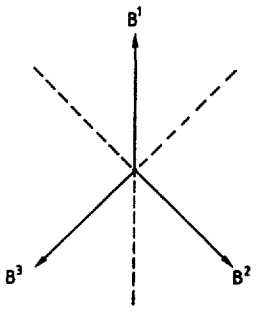

Fig.2. — The plane of the three teachers for $Q' = 3$. The dashed lines show the barriers between their regions.

dimensional, even after enforcing (11). We resort to an algebraic treatment of the sort used to derive (19), giving the generalisation error

$$\begin{aligned}\epsilon_g &= 1-\left\langle \sum_{a'}\left(\prod_{b'\neq a'}\Theta\left(\mathbf{B}^{a'}\ \mathbf{S}-\mathbf{B}^{b'}\ \mathbf{S}\right)\prod_{c'\neq a'}\Theta\left(\mathbf{J}^{a'}\ \mathbf{S}-\mathbf{J}^{c'}\ \mathbf{S}\right)\right)\right\rangle_{\mathbf{S}} \\ &= 1-\int_0^\infty\left(\frac{\mathrm{d}x}{\sqrt{2\pi}}\right)^{2(Q'-1)}(u^2-v^2)^{\frac{1-Q'}{2}}\exp\left\{-\frac{\mathbf{x}\cdot T\cdot\mathbf{x}}{2Q'(u^2-v^2)}\right\}, \end{aligned} \tag{22}$$

where T is a $2(Q'-1)\times 2(Q'-1)$ matrix given by

$$T=\left(\begin{array}{ccc|ccc} (Q'-1)u & & -u & -(Q'-1)v & & v \\ & \ddots & & & \ddots & \\ -u & & (Q'-1)u & v & & -(Q'-1)v \\ \hline -(Q'-1)v & & v & (Q'-1)u & & -u \\ & \ddots & & & \ddots & \\ v & & -(Q'-1)v & -u & & (Q'-1)u \end{array}\right) \tag{23}$$

(i.e. 4 $(Q'-1)\times(Q'-1)$ blocks), where $u\equiv 1-a$ and $v\equiv R(1-a)$. The results of this scheme are plotted in figure 3 for $Q=2,3,4$. In all cases $\epsilon_g\sim 1/\sqrt{\alpha}$ as $\alpha\to\infty$.

Although the Q' outputs could be represented by combinations of only the next largest integer to $\log_2(Q')$ binary outputs, it would require Q' binary perceptrons to learn the problem exactly, since the input space must be divided by Q' planes, lying along the planes of the multi-class perceptron. It should be noted that the Hebb rule is unable to generate a set of planes to learn the problem perfectly, however, since, as may be seen from figure 1, it is not clear from a given example which planes should be changed.

A better way to teach the multi-class perceptron would be with the multiclass maximum stability rule, which can be generated by an algorithm given in [15], but analysis of the consequences is cumbersome for $Q'>2$. We would expect qualitatively similar results to those above, however, but with $\epsilon\sim 1/\alpha$, as in [2]. To verify this we analyse high temperature learning, as explained in section 2, which in learnable problems is believed to be of qualitatively similar form to the low temperature result [3]. For simplicity we will assume (21), so that there is a single order parameter $R\equiv\mathbf{B}^{s'}\cdot\mathbf{J}^{s'}$ (taken to be the same for all s'), the cosine of the angle between the space containing $\{\mathbf{J}^{s'}\}$ and that containing $\{\mathbf{B}^{s'}\}$. We will further assume that the $\mathbf{J}^{s'}\ \mathbf{B}^{\tilde{s}'}$ are the same for all $s'\neq\tilde{s}'$. We would not expect our choice for the overlaps between the $\{\mathbf{J}^{s'}\}$, which can easily be enforced by adding a term to the training energy, to make a difference to the dynamics, since allowing different overlaps will surely increase ϵ_g, without making an extensive contribution to the entropy. Equation (21) should thus be obeyed naturally. In problems where the angle between the teacher vectors is not known in advance, the multi-class perceptron should also naturally evolve to have the best angle between its planes. The entropy required for the free energy (7) may be obtained by considering the $(Q'-1)$-dimensional spaces in which the $\{\mathbf{J}^{s'}\}$ and $\{\mathbf{B}^{s'}\}$ lie; let us say these spaces are at an angle ϕ. If we choose any $(Q'-1)$ orthogonal vectors in the B-space and for each choose, independently, a vector in the full N-dimensional space at an angle ϕ, then these new vectors (in the large N limit) will be effectively orthogonal and so the space spanned by these new vectors is at an angle ϕ to that of the $\{\mathbf{B}^{s'}\}$, and thus could be that of the $\{\mathbf{J}^{s'}\}$. We know (from [7]) that the entropy associated with choosing a new vector at an angle ϕ to a given vector is (to within a constant) $\frac{1}{2}\log(1-R^2)$, where $R=\cos\phi$. Selection of the vectors

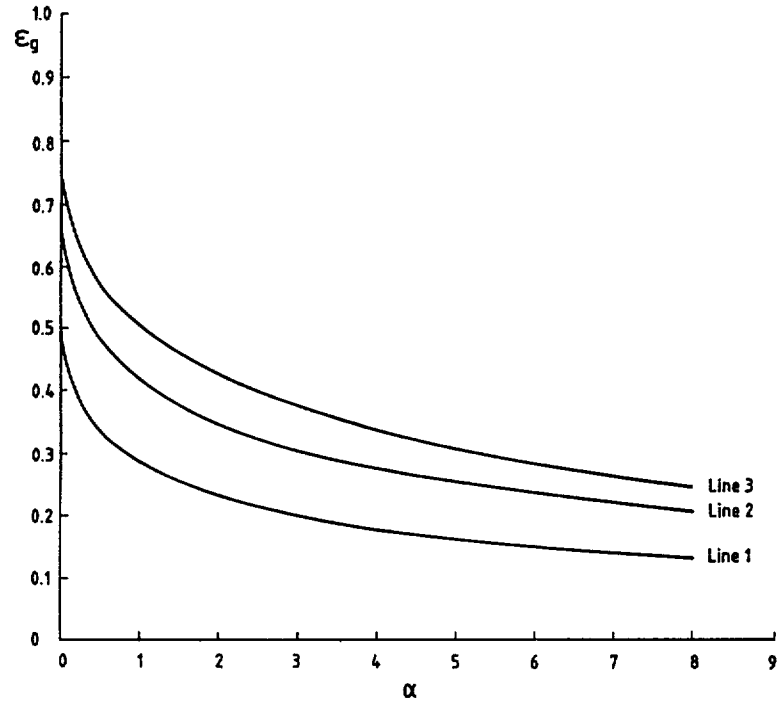

Fig.3. — Learning with the Hebb rule. This shows the generalisation error using $Q = 2, 3$ and 4, as lines 1, 2 and 3 respectively.

within the B-space (i.e. the orientation of the J-space and the B-space) does not generate an extensive entropy, so the total entropy for (7) is $(Q' - 1)/2 \log(1 - R^2)$.

The numerical results (for $Q' = 2, 3$ and 4), where the line for $Q' = 2$ is the same as in [3], show, as expected, that the spherical multi-class perceptron learns smoothly with asymptotic $\epsilon_g \sim 1/\tilde{\alpha}$. The multi-class Ising percep ton, at least for $Q' = 2, 3$, has a first order transition to perfect generalisation. The result for $Q' = 3$ is similar to the one for $Q' = 2$. The generalisation error $\epsilon(\tilde{\alpha})$ decreases with increasing $\tilde{\alpha}$ from 2/3 smoothly and jumps at $\tilde{\alpha} \approx 2.24$ discontinously to zero. To derive this high temperature result we had to determine the Ising entropy, which is the logarithm of the maximal number of states compatible with the constraints (20,21). Using a simple counting argument we find for the entropy

$$s(Q' = 3) = a_+ \ln a_+ + a_- \ln a_- + b_+ \ln b_+ + b_- \ln b_- + 4\tilde{a} \ln \tilde{a}, \tag{24}$$

where $a_\pm \equiv (1 \pm R)/8 - \tilde{a}$, $b_\pm \equiv 3(1 \pm R)/8 - \tilde{a}$ and $\tilde{a}$ is given by the solution of $\tilde{a}^3 - 10(1 - R^2)\tilde{a}^2/64 + 3(1 - R^2)\tilde{a}/64 - [3(1 - R^2)/64]^2 = 0$, for $(1 - R)/8 > \tilde{a} > 0$.

It is possible to teach the same problem to Q' binary neurons in the same way. In fact enforcing their weight vectors to have the same overlaps with each other as the $\{\mathbf{J}^{s'}\}$ makes the learning equivalent to that above, but of course at the expense of more neurons and a more complicated output representation.

4. The proximity problem.

Section 2.1 introduced one variety of unlearnable rule: the proximity problem of classifying inputs according to their Hamming distance from $p_o = \alpha N$ prototypes $\{\boldsymbol{\eta}^\mu\}$. We learn the problem using noisy examples $\boldsymbol{\xi}^{\mu l}$, such that $\boldsymbol{\eta}^\mu \cdot \boldsymbol{\xi}^{\mu l} = m$. Reference [20] assumed that the best learning algorithm would be to search the J-space for the weight vector with the smallest training error; they were able to solve the model in the limit of small m, which implies that a large number of examples of each prototype must be presented.

However, it is possible to deduce the optimal learning algorithm using the very original approach of [18], which was developed for a different problem. They considered a highly diluted neural network storing patterns $\boldsymbol{\xi}^\mu$ and minimised the output error of the stored patterns after the first time step when the input patterns were ensembles of their noisy versions with overlap m with the clean patterns. From the work of [24] we know that the output error at the next time step is given by (nelecting additive constants) $H(m\Lambda/\sqrt{1-m^2})$, where $\Lambda = \sqrt{N}\boldsymbol{\xi}_o \mathbf{J} \cdot \boldsymbol{\xi}$, $H(z) \equiv \int_{-\infty}^{z} \mathrm{D}z$ and $\mathrm{D}z \equiv \frac{1}{\sqrt{2\pi}} \exp(-z^2/2)\mathrm{d}z$. [18] showed that the $\mathbf{J}$ which minimises this expression for m small, is given by the Hebb rule (4). For m close to 1 the optimal $\mathbf{J}$ is given by a maximum-stability rule (MSR). For intermediate m the optimal $\mathbf{J}$ has to be found numerically using a Maxwell construction.

Reference [20] shows that the generalisation error of the binary perceptron in the case of the proximity problem is given by $H(m\Lambda/\sqrt{1-m^2})$, where $\Lambda = \sqrt{N}\boldsymbol{\eta}_o \mathbf{J} \cdot \boldsymbol{\eta}$. Using the insight outlined above we see that only for high m is the MSR (used in [20]) optimal; the $\mathbf{J}$ which minimises the generalisation for small m (the case [20] mainly considers) is given by the Hebb rule.

This is because the MSR generates narrow, deep valleys in the energy surface around each example presented, so that these examples are well stored. The Hebb rule, by contrast, generates wider, but shallower, valleys so that each example may not be perfectly stored, but its influence extends over a wider region. If the noise is high, so examples fall a long way from prototypes, the second rule is to be prefered, to generate a valley around prototypes, so in this limit, the one solved by [20], the Hebb rule is optimal for binary perceptrons (as we will verify). We will assume the same result applies to multi-class perceptrons.

We shall again choose to analyse rule (17) (optionally enforcing (11) afterwards, as in Sect. 2), which implies that each $\mathbf{J}^{s'}$ is affected only by examples $\boldsymbol{\xi}^{\mu l}$ such that $s' = \eta_o^\mu$. If, after learning, the alignment between $\mathbf{J}^{s'}$ and one such $\boldsymbol{\eta}^\mu$ is called Λ then, from [20], the overlap z between $\mathbf{J}^{s'}$ and a new example $\boldsymbol{\xi}^{\mu l}$ has a distribution

$$\Pr(z) = \frac{1}{\sqrt{2\pi(1-m^2)}} \exp\left\{-\frac{(z-m\Lambda)^2}{2(1-m^2)}\right\} \tag{25}$$

The distribution of overlaps between $\boldsymbol{\xi}^{\mu l}$ and every other $\mathbf{J}^{s'}$ is an independent Gaussian with zero mean and unit variance, independent for different s'. The chance that $\boldsymbol{\xi}^{\mu l}$ will be incorrectly classified, ϵ_g, is the chance that the correct field is higher than the $Q'-1$ others, and so

$$\epsilon_g = 1 - \int \frac{\mathrm{d}z}{\sqrt{2\pi(1-m^2)}} \exp\left\{-\frac{(z-m\Lambda)^2}{2(1-m^2)}\right\} \left[\int_{-\infty}^{z} \mathrm{D}y\right]^{Q'-1} \tag{26}$$

It should be noted that this formula, for the case of $Q' = 2$ (for which one integral may be performed), differs by a factor of $\sqrt{2}$ from that of [20]. This is because their Λ is defined as overlap with $\mathbf{J} = \frac{1}{\sqrt{2}}(\mathbf{J}^1 - \mathbf{J}^2)$, the normalised binary form. The normalisation factor is $\frac{1}{\sqrt{2}}$, since $\mathbf{J}^1$ and $\mathbf{J}^2$ are, by (17), uncorrelated and thus, in a high dimensional space, effectively

perpendicular. An expression of similar form to (26) was derived [13] for the very different problem of storing many-valued patterns at zero-temperature in Hopfield-like networks.

To calculate the distribution of Λ, consider that as far as a prototype (without loss of generality prototype 1, $\boldsymbol{\eta}^1$) is concerned , the $\mathbf{J}^{s'}$ with $\eta_o^1 = s'$, is the sum of three components: i) The correct prototype $\boldsymbol{\eta}^1$; ii) noise from the examples of that prototype; iii) noise from other prototypes whose correct output is also s'. Each example has a component m in the direction of its prototype, so (i) is $pm\boldsymbol{\eta}^1/\gamma^{s'}$, and $\sqrt{1-m^2}$ in a different random direction (effectively perpendicular for different examples since $p \ll N$) so that term (ii), by Pythagoras, is $\sqrt{p(1-m^2)}/\gamma^{s'}$ The magnitude of the sum of these terms is $(p^2m^2 + p(1-m^2))^{1/2}/\gamma^{s'}$ (iii) is the sum of p_o/Q' independent vectors of this kind, and, since $p_o \propto N$, we add them in the same way as the noise terms of the Hebb rule in section 3, to obtain $\gamma^{s'} = \sqrt{p_o/Q'}(p^2m^2 + p(1-m^2))^{1/2}$, the factor by which $\mathbf{J}^{s'}$ must be normalised. Then Λ, the overlap between a pattern and $\mathbf{J}^{s'}$, is the sum of term (i) and the random variable which is the component of (iii) in the direction $\boldsymbol{\xi}$, a Gaussian of width $\sqrt{p_o}$,

$$\Pr(\Lambda) = \sqrt{\frac{Q'}{2\pi\alpha}} \exp\left\{-\frac{(\Lambda - pm/\gamma)^2}{2\alpha/Q'}\right\} \tag{27}$$

which, rescaling with $\tilde{p} \equiv pm^2/(1-m^2)$ and $\alpha_0 \equiv \alpha(1-m^2)/m^2$, gives

$$\Pr(\Lambda) = \sqrt{\frac{Q'(1-m^2)}{2\pi m^2 \alpha_o}} \exp\left\{-\frac{\left(\Lambda - \frac{\sqrt{Q'(1-m^2)}}{m\sqrt{\alpha_o(1+1/\tilde{p})}}\right)^2}{2\alpha_o m^2/Q'(1-m^2)}\right\}. \tag{28}$$

As $m \to 0$ and with $\alpha_o = O(1)$ we obtain $Pr(\Lambda) = \delta\left(\Lambda - \sqrt{Q'}/(m\sqrt{\alpha_o(1+1/\tilde{p})})\right)$, which gives

$$\epsilon_g = 1 - \int Dz\ H^{p-1}\left(z + \frac{\sqrt{Q'}}{\sqrt{\alpha_o(1+1/\tilde{p})}}\right). \tag{29}$$

For $Q' = 2$ this reduces to $\epsilon_g = 1 - H\left(1/\sqrt{\alpha_o(1+1/\tilde{p})}\right)$.

Figure 4 shows a graph from [20] with $\alpha_o = 1.6$. Line 1 is their zero-temperature result (with overfitting) and line 2 is their result allowing a training error. The results for Hebb learning with a binary perceptron are shown as line 3; they tend to the same minimum generalisation error but the Hebb rule typically requires one and a half orders of magnitude fewer examples than the algorithm of [20] to obtain the same generalisation error. Calling $\epsilon(m)$ again the minimum training error, we see that as in [20], $\epsilon_g - \epsilon(m) \sim 1/\tilde{p}$ as $\tilde{p} \to \infty$, but clearly the coefficient of $1/\tilde{p}$ is very much lower for Hebb learning. The points show the results of a single numerical simulation using $N = 5000$ and $m = 0.1$.

Figure 5, which has a horizontal scale linear in $\log(1+\tilde{p})$, compares several values of Q' using the same $\alpha_o = 1.6$. The problem becomes harder for a larger number of output states because inputs must be classified into a greater number of areas; however, the number of prototypes in each class is smaller. It is found that as Q' rises the minimal generalisation error first rises with it and then, at a critical $Q'(\alpha_o)$, falls back, tending to zero as $Q' \to \infty$.

We can compare how the four-output problem is learnt by two binary neurons, encoding the four states as the possible combinations of the two spins (state 1 is $\{1,1\}$, 2 is $\{1,-1\}$, etc.). This representation is clearly not unique and breaks the symmetry of the answers. Each of the binary perceptrons learns to give one answer for patterns near two of the prototypes and

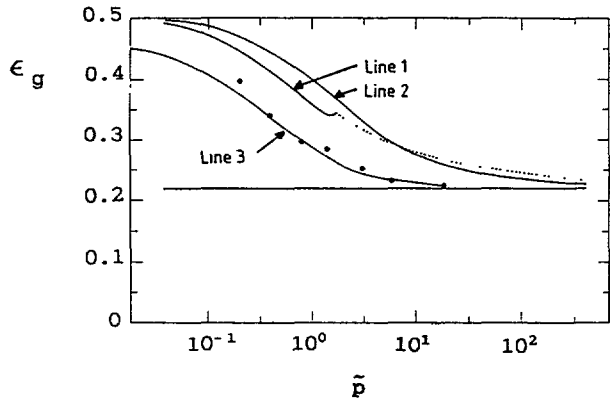

Fig. 4. — Learning the proximity problem for $\alpha_0 = 1.6$. Lines 1 and 2, from [20], show respectively, the effects of learning by minimising the training error and by fixing it at a finite value. Line 3 is the Hebb result, with experimental points shown for comparison.

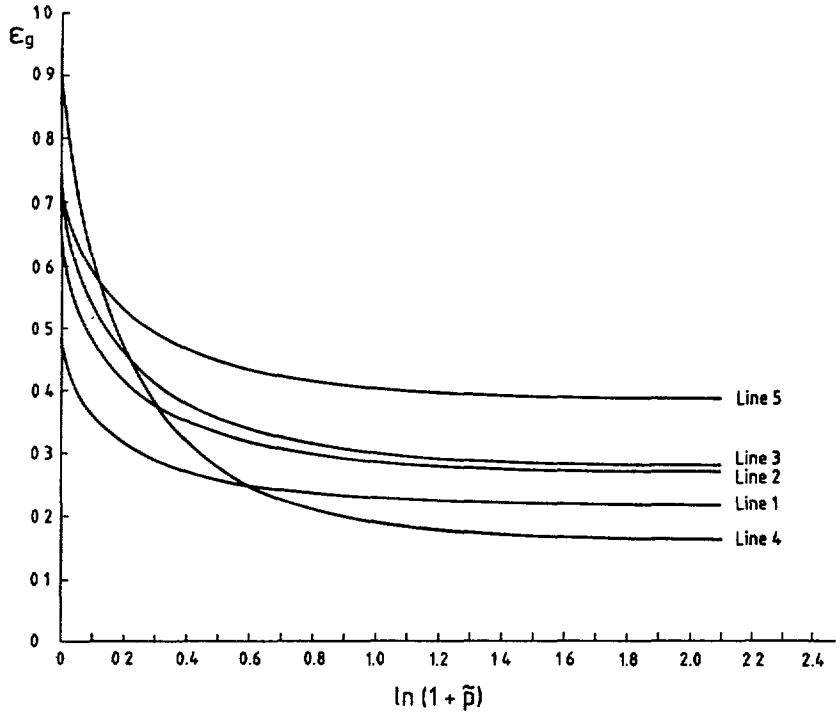

Fig.5. — Learing the proximity problem for $\alpha_0 = 1.6$ using Q' equal to 2 (line 1, as in figure 4), 3 (line 2), 4 (line 3) and 12 (line 4). Line 5 is Hebb learning of the $Q' = 4$ problem using two binary perceptrons.

the other answer for patterns near the other two. The consequences of Hebb learning can be calculated in a manner similar to the one used above and are shown in figure 5 as line 5. Two binary perceptrons are always less efficient and tend to a higher minimum generalisation error. This is related to the fact that binary neurons have only two planes with which to divide the space, against the six of a $Q' = 4$ multi-class perceptron. As Q' rises a combination of binary perceptrons will do progressively worse, compared to the multi-class method. It might be argued that a $Q' = 4$ multiclass perceptron should do better than a combination of two binary perceptrons, since it has more degrees of freedom (3 independent J vectors, instead of 2), and of course, we could use many more binary perceptrons, with enough planes to divide the space as well as the multi-class perceptron (at some engineering cost). This would be unnatural, however, and give many more unwelcome alternatives for the representation of answers as combinations of binary digits.

5. Conclusion.

We have shown that in multi-class problems, which form a large proportion of the obvious neural network applications, a multi-class perceptron is a far more natural choice than a combination of binary perceptrons, because without *a priori* knowledge we would expect the possible answers to a question to be equivalent, and also has advantages in efficiency. This seems to be another instance of the rule suggested by [22], that the simplest network which can solve a problem is also the most efficient. It remains to be seen how more complicated architectures would perform. In many problems whose solutions are thought to require a complicated, many-layer network of perceptrons it may be possible to make significant improvement in efficiency by using more complicated neurons. Such a choice might be natural for such a problem.

We have seen again, as was indicated in [7], how critical the choice of energy function is to find the best solution to a problem. The Hebb rule, which seemed such a poor choice for some problems, has been shown to be optimal for other, especially unlearnable, problems.

This study has mainly used spherical multi-class perceptrons, but, as with binary perceptrons, this may not be the best if, for example, the values of the components of the teacher vectors, $\{B_j^{s's}\}$ are discrete. It is not clear, however, what the analogue of the binary Ising perceptron is. Should we allow each $J_j^{s's}$ to take several real values? If so, these values will of course be a ladder, not obeying the multi-class symmetry. We might conceivably overcome this by allowing the interactions to take complex values, for example the three complex roots of 1, but this seems a shade gratuitous and runs into difficulties for more than a few states. However, if we allow the interactions to be just Ising (± 1), then the gauge fixings (11,12) cannot be enforced for Q or Q' odd. Ultimately our guide must be the values each $B_j^{s's}$ may take.

It is interesting, finally, to point out that there are multi-class problems for which a multi-class perceptron is not the best choice: ones in which the possible answers have a different form of symmetry. An example is the classification of levels of quality, which have a ladder symmetry corresponding to different values of one parameter: "goodness". In this case a *multi-level*, or *graded response* neuron, whose states have the same symmetry, would presumably be better. Alternative solutions using a multi-class perceptron or a combination of binary perceptrons are possible, but only if we are allowed to introduce thesholds into the perceptrons; in both cases the relative orientation of the many planes which divide the space is an unwanted freedom, which presumably lowers the efficiency of learning. This type of problem will be the subject of another paper.

Acknowledgements.

Two of us, (T.L.H.W.) and (A.R.), are very grateful to the group at the Katholieke Universiteit Leuven for their kind hospitality during our stay there, to the SERC for its financial support, and to K.Y.M. Wong for many invaluable discussions. One of us (A.R.) thanks the *Studienstiftung des deutschen Volkes* and *Corpus Christi College, Oxford* for the award of two scholarships. We would like to thank Marc Mézard and Jean-Pierre Nadal for drawing our attention to the problem of learning multi-class classification problems, while one of us (A.R.) was staying at the Ecole Normale Supérieure.

References

[1] Vallet F. *Europhys. Lett.* **8** (1989) 747.

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

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

[4] Györgyi G., *Phys. Rev.* **A 41** (1990) 7097.

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

[6] Watkin T. L. H., Rau A. Selecting Examples for Perceptrons, in print *J.Phys.A* (1991).

[7] Watkin T. L. H., Rau A. How to Learn the Unlearnable, accepted for publication in *Phys.Rev.A* (1991).

[8] Bohr H., Bohr J., Brunak S., Cotterill R.J.M., Fredholm H., Lautrup B., Peterson S.B., *FEBS Letters* **261** (1990) 43.

[9] Gallant S.I., *IEEE Trans. Neural Net.* **1** (1990) 179.

[10] Kohonen T., *Neural Network Architectures (Kogan Page, London*, 1988), p. 26.

[11] Minsky M.L., Papert S.A., Perceptrons (MIT Press, Cambridge, 1969).

[12] Hopfield J.J. *Proc. Natl. Acad. Sci. (USA)* **79** (1982) 2554.

[13] Kanter I., *Phys. Rev.* **A37** (1988) 7.

[14] Bollé D., Dupont P. and van Mourik J., *J. Phys. A*, **24**, (1991) 1065;
Bollé D. and Dupont P., in Statistical Mechanics of Neural Networks, L. Garrido Ed. (Springer, Berlin, 1990), p365;
Bollé D., Dupont P. and Huyghebaert J., Thermodynamic Properties of the Q-state Potts-glass Neural Network, *Phys. Rev. A* (1991) in press.

[15] Nadal J.-P., Rau A., *J. Phys. I France* **1** (1991) 1109.

[16] Nilsson N.J., Learning Machines (McGraw-Hill, New York 1965).

[17] Seung S., Sompolinsky H., Tishby N. Statistical Mechanics of Learning from Examples II., preprint (1991).

[18] Wong K.Y.M., Sherrington D. *J. Phys.* **A 23** (1990) 4659.

[19] Krauth W., Mézard M. *J.Phys.* **A 20** (1987) L745.

[20] Hansel D., Sompolinsky H., *Europhys. Lett.* **11** (1990) 687.

[21] Müller B., Reinhardt J., Neural Networks - An Introduction (Springer Verlag, Berlin) (1991).

[22] Schwarze H., *Diplomarbeit*, Universität Gießen (1991).

[23] Györgyi G., *Phys.Rev.Lett.* **64** (1990) 2957.

[24] Krauth W., Mézard M., Nadal J.-P., *Complex Systems* **2** (1988) 387.