Learning Algorithms, Input Distributions and Generalization

Ronny Meir*
Department of Electrical Engineering
Technion
Haifa 32000, Israel
rmeir@ee.technion.ac.il

June 1993

Abstract

We study the interaction between input distributions, learning algorithms and finite sample sizes in the case of learning classification tasks. Focusing on the case of normal input distributions, we use statistical mechanics techniques to calculate the empirical and expected (or generalization) errors for several well-known algorithms learning the weights of a single-layer perceptron. In the case of spherically symmetric distributions within each class we find that the simple Hebb algorithm is optimal. Moreover, we show that in the regime where the overlap between the classes is large, algorithms with low empirical error do worse in terms of generalization, a phenomenon known as over-training.

1 Introduction

The problem of pattern recognition can formally be stated as follows (Vapnik 1982): in a certain environment which is characterized by a probability density $P(\vec{x})$, instances $\vec{x}$ appear randomly and independently. The instructor classifies these instances into one of k classes, using the conditional probability distribution function $P(y|\vec{x})$, where $y = 0, 1, \ldots, k-1$ is a class label. Neither the properties of the environment $P(\vec{x})$ nor the decision rule $P(y|\vec{x})$ is known. In the remainder of the paper we restrict ourselves to *two-class* classification, denoting the class labels by $y = \pm$.

In the case of *parametric* classification one considers a class of parameterized functions $f_w(\vec{x}) = \mathrm{sgn}(h_w(\vec{x}))$, depending on a parameter vector $\vec{w}$. The objective of pattern recognition is then to estimate the value $\vec{w}^*$ of the parameters $\vec{w}$, which minimize the probability of misclassification for an input instance $\vec{x}$ drawn randomly according to the environmental probability distribution $P(\vec{x})$. This quantity, which (following Vapnik 1982) we term the *expected error*, is given by

$$\varepsilon(\vec{w}) = \int d\vec{x}dy\ P(\vec{x}, y) I_y^c(f_w(\vec{x})), \tag{1}$$

*Research supported in part by the Ollendorf center of the electrical engineering department at the Technion and by the Lady Davis foundation.

where the complementary indicator function $I_y^c(z)$ is 0 if $y = z$ and 1 otherwise. It should be noted that the expected error is sometimes termed generalization or prediction error by other authors. In typical situations one is exposed to a set m training pairs $D^m = \{(\vec{x}^1, y^1), \ldots, (\vec{x}^m, y^m)\}$, each drawn independently at random according to the unknown probability distribution $P(\vec{x}, y)$. It is then common to define the *empirical error*

$$\nu(\vec{w}, D^m) = \frac{1}{m} \sum_{l=1}^{m} I_{y^l}^c(f_w(\vec{x}^l)), \tag{2}$$

which is a finite sample approximation to the expected error. It can be shown that under a wide range of conditions (Vapnik 1982) the empirical error converges uniformly to the expected error almost surely, when the sample size m increases without bound. The rate at which this occurs however, is a complicated and problem-specific issue, which is at the heart of the learning problem. In fact, it is the objective of efficient learning strategies to ensure that this convergence occur with the highest possible rate. In particular, one is often interested in learning algorithms which produce the lowest possible expected error for finite sample sizes. It has been pointed out by Vapnik (1982), that there is no reason to expect that the parameter value, w^m, minimizing the empirical risk for a finite sample size m, is indeed the best approximation to the true minimizer $\vec{w}^*$. Moreover, from a practical point of view, minimizing the empirical error in the pattern recognition problem may be problematic as well. As can be seen from Eq. (2), the empirical error is a piece-wise constant function, rendering any gradient-based method useless.

One possible solution to these problems is choosing the parameter value $\vec{w}$ to minimize an auxiliary function termed the *training error*,

$$\varepsilon_t(\vec{w}, D^m) = \frac{1}{m} \sum_{l=1}^{m} V(y^l, h_w(\vec{x}^l)). \tag{3}$$

Here $V(y^l, f_w(\vec{x}^l)$ is a *differentiable* distance measure between the desired and actual outcome of the classifier. One then usually considers *gradient-based* learning algorithms, which utilize the information available from the gradient vector in order to minimize the function $\varepsilon_t(\vec{w}, D^m)$. The widely used back-propagation algorithm is a special case of this strategy (at least when applied to classification problems). As we show in section 3, it is actually possible for finite sample sizes to obtain a lower expected error by choosing $\vec{w}$ to minimize the training, rather than the empirical, error.

A general framework for judging the performance of any classifier is Bayesian theory (Duda and Hart 1973). Denoting by $P(\vec{x}|y)$ the conditional probability for input $\vec{x}$ given that it belongs to class $y = \pm$, one can show that the optimal classifier is obtained by the decision rule

$$F(\vec{x}) = \operatorname{sgn}\left[\ln P(\vec{x}|+) - \ln P(\vec{x}|-) + \ln \frac{p_+}{p_-}\right], \tag{4}$$

where $p_\pm$ are the prior probabilities for class $\pm$. In the remainder of this work we will be interested in the important case of *normal* probability distributions $P(\vec{x}|y)$, for which the optimal Bayes classifier is given by the *quadratic classifier*

$$F(\vec{x}) = \operatorname{sgn}\left[\frac{1}{2}(\vec{x} - \vec{u}_+)^T \Sigma_+^{-1} (\vec{x} - \vec{u}_+) - \frac{1}{2}(\vec{x} - \vec{u}_-)^T \Sigma_-^{-1} (\vec{x} - \vec{u}_-) + \frac{1}{2} \ln \frac{|\Sigma_+|}{|\Sigma_-|} - \ln \frac{p_+}{p_-}\right]. \tag{5}$$

In the above equation $\vec{u}_\pm$ and $\Sigma_\pm$ stand for the mean vectors and covariance matrices, respectively, of the normal distributions $P(\vec{x}|\pm)$, and $|\Sigma_\pm|$ stand for the determinants. As can be readily seen from Eq. (5), in the case where the covariance matrices $\Sigma_\pm$ are equal, the optimal Bayes classifier reduces to a *linear* discriminant function (single-layer perceptron),

$$F(\vec{x}) = \mathrm{sgn}\left[(\vec{u}_+ - \vec{u}_-)^T\Sigma^{-1}\vec{x} + \frac{1}{2}\left(\vec{u}_+^T\Sigma^{-1}\vec{u}_+ - \vec{u}_-^T\Sigma^{-1}\vec{u}_-\right) - \ln\frac{p_+}{p_-}\right]. \tag{6}$$

We restrict ourselves in the remainder of the paper to the analysis of the single layer case, both in the case where it is optimal (equal covariance matrices) and otherwise. We are able then to calculate analytically the empirical, training and expected errors for a variety of choices for the training error.

In performing the calculations for the different learning models we follow the line of research pioneered by E. Gardner (1988), which enables the analytic calculation of many relevant quantities. Much of the earlier work along these lines has focused on rather simple situations such as uniform distributions and computationally unfeasible learning algorithms (see Watkin et. al. 1993 for a review). More recently Griniasti and Gutfreund (1991) and Meir and Fontanari (1992) have studied more practical gradient-based learning algorithms, while Biehl and Mietzner (1993) and Barkai et. al. (1993) have focused on more realistic input distributions of the type discussed in this paper. In fact, it is the goal of this paper to combine the above extensions, thus prompting a theoretical investigation into the performances of several well-known learning algorithms, and their dependence on the task being solved.

The remainder of the paper is organized as follows. In section 2 we specialize the above general framework to the particular case studied in this work. Section 3 then introduces several training error functions, corresponding to distinct learning algorithms, while in section 4 we present the analysis of the models under various circumstances. Finally, in section 5 we summarize our findings and list several open problems. The mathematical details of the derivation, based on the work of E. Gardner (1988), are briefly reviewed in the appendix but are not essential for an understanding of the paper.

2 The Linear Classifier

We consider a single-layer perceptron with inputs $\vec{x} \in R^d$ and output

$$o = \mathrm{sgn}(h_w(\vec{x})) = \mathrm{sgn}(\vec{w}\cdot\vec{x}). \tag{7}$$

We have set the bias term to zero for simplicity and thus, without loss of generality, impose the normalization condition $\|\vec{w}\| = 1$. We further assume that the conditional probability distribution of input vector $\vec{x}$ given class label $y = \pm$ is Gaussian,

$$P(\vec{x}|y) = \left(2\pi\sigma_y^2\right)^{-d/2}\exp\left(-\|\vec{x}-\vec{u}_y\|^2/2\sigma_y^2\right). \tag{8}$$

Here we have assumed the covariance matrices Σ_y to be multiples of the unit matrix in d dimensions, i.e. $\Sigma_\pm = \sigma_\pm^2 I$. The geometrical meaning of this assumption is that the pattern distribution around each class center is *spherically-symmetric*. To simplify the calculations we follow Barkai et. al. (1993) and take the mean vectors $\vec{u}_y$ to be orthogonal, $\vec{u}_+\cdot\vec{u}_- = 0$, and of equal magnitude, $\|\vec{u}_+\| = \|\vec{u}_-\| = u$. Furthermore, we will assume throughout that the prior class probabilities, $p_\pm$, are equal to 1/2.

The probability of error for a linear classifier of weight vector $\vec{w}$ and input distribution as in Eq. (8) is readily evaluated, yielding

$$\varepsilon(\vec{w}) = \frac{1}{2}\left[H\left(\frac{\vec{w}\cdot\vec{u}_+}{\sigma_+}\right) + H\left(-\frac{\vec{w}\cdot\vec{u}_-}{\sigma_-}\right)\right], \tag{9}$$

where we have used $H(y) = \int_y^\infty dx\ e^{-x^2/2}/\sqrt{2\pi}$. As mentioned in section 1, in the case where the variances are equal, i.e. $\sigma_+ = \sigma_- = \sigma$, the linear discriminant function is optimal. The optimal weight $\vec{w}^*$ can be read directly from Eq. (6), yielding

$$\vec{w}^* = (\vec{u}_+ - \vec{u}_-)/(\sqrt{2}u), \tag{10}$$

giving rise to the minimal expected error

$$\varepsilon(\vec{w}^*) = H(u/\sqrt{2}\sigma). \tag{11}$$

We note that the factor $\sqrt{2}$ in Eq. (10) guarantees the normalization $\|\vec{w}\| = 1$ (keeping in mind the orthogonality condition $\vec{u}_+ \cdot \vec{u}_- = 0$). Obviously if one knew $\vec{u}_\pm$ in advance one could plug them directly into Eq. (10) and obtain $\vec{w}^*$. The problem of learning is approximating $\vec{w}^*$ as well as possible from a limited data set.

3 Minimizing the Training Error

As discussed in section 1 the goal of learning is to calcualte parameter values, $\vec{w}^*$, which yield minimal expected error. However, with access to only a finite set of input/output pairs, D^m, the exact calculation of the expected error is impossible. This fact, together with the non-differentiability of the empirical error led us to consider minimizing the training-error, Eq. (3).

For the case of single-layer perceptrons it is useful to define the *stability*, given by

$$\Delta^l = y^l \vec{w} \cdot \vec{x}^l. \tag{12}$$

The complementary indicator function $I^c_{y^l}(f_w(\vec{x}^l))$ of Eq. (2) is then replaced by $\Theta(-\Delta^l)$ where the Heaviside function $\Theta(x)$ is zero for negative x and 1 otherwise. The empirical error is then given by

$$\nu(\vec{w}, D^m) = \frac{1}{m}\sum_{l=1}^m \Theta(-\Delta^l). \tag{13}$$

With a slight abuse of the notation for $V(\cdot)$, Eq. (3), the training error for our problem then takes the form (Griniasti and Gutfreund 1991),

$$\varepsilon_t(\vec{w}, D^m) = \frac{1}{m}\sum_{l=1}^m V(\Delta^l)\ . \tag{14}$$

In particular we focus here on three different functions $V(\Delta)$, which give rise to distinct learning algorithms. The first function studied is the one giving rise to the well-known perceptron learning algorithm,

$$V_{\mathrm{P}}(\Delta) = -\Delta\Theta(-\Delta). \tag{15}$$

Gradient descent learning using this function gives rise to the perceptron learning rule (in batch mode). We note in passing that the well known result on the non-convergence of the perceptron learning algorithm in the non-separable case (Minsky and Papert 1988), does not apply in the batch mode since the algorithm always converges to a local minimum, as long as the gradient descent is done properly. Another related function, recently proposed by Frean (1993),

$$V_{\mathrm{F}}(\Delta) = [1 - \exp(\lambda\Delta)]\Theta(-\Delta), \tag{16}$$

was motivated by the desire to construct an efficient learning algorithm in the non linearly-separable case. The motivation for this proposal was the observation that in the perceptron error function, Eq. (15), the price paid for strongly 'wrong' patterns (large negative Δ) is very high (proportional to $|\Delta|$). Thus, outliers are expected to wreak havoc with this learning algorithm. In the newly proposed algorithm (i.e. gradient-descent dynamics on the fuction (3) with $V = V_{\mathrm{F}}$) outliers are suppresses due to the exponential decay. The Frean function interpolates nicely between the perceptron function for small λ and the empirical error, i.e. the fraction of misclassifications (Eq. (2)), when λ becomes large. Note that the functions V_{P} and V_{F} are continuous, in spite of the step-functions appearing in their definition.

Finally, in the Hebb learning algorithm (recently studied in a very similar context by Barkai et. al. 1993) one has

$$\vec{w} = \frac{1}{\gamma\sqrt{d}} \sum_{l=1}^{m} y^l \vec{x}^l, \tag{17}$$

where γ assures the normalization $||\vec{w}|| = 1$. One should note that this algorithm can be derived as the gradient-descent dynamics performed on the training-error ε_t with

$$V_{\mathrm{H}}(\Delta) = -\Delta, \tag{18}$$

provided initially $\vec{w} = \vec{0}$. We note in passing that the Hebb rule is different from the other rules studied in that it is not an error-correcting rule, i.e. all patterns contribute to the weight modification, whether they are correctly classfied or not.

In this work we do not solve the dynamical process resulting from the gradient descent procedure, but rather compute the properties of the global minima of the above energy functions. In the absence of local minima, any gradient descent procedure will converge to a global minimum. In particular, in section 4 we present the empirical and expected errors predicted by minimizing the different training errors discussed in this section.

4 Analysis

The analytic results we derive are obtained in the so called *thermodynamic limit*: $d \to \infty$, $m \to \infty$ and $\alpha = m/d < \infty$, and utilize the frameowrk developed by Gardner (1988) and Gardner and Derrida (1988). We will be particularly interested in the case where the class centers, $\vec{u}_{\pm}$, as well their widths, $\sigma_{\pm}$, are of order unity, so that the inputs from the two classes overlap considerably, and thus the minimum expected error is non-zero (see for example Eq. (9)).

Before discussing the results it is important to make the following observation. For small values of α, the normalized sample size, both the perceptron and the Frean error functions yield zero empirical error. This results from the fact that for a small number of training examples (small α) a hyper-plane can always separate the data without error, yielding $\Delta^l > 0$ for all values of l. In this paper we focus on values of α larger than this critical size, since all error-correcting algorithms are identical below this point. This observation does not hold for the Hebb rule, which yields non-zero empirical error even for small α. Using the equations described in the appendix allows us to analyze the empirical and expected errors produced by the various learning algorithms studied. Following the notation of Barkai et. al. (1993) we define the variables $R_\pm$ through

$$\vec{w}\cdot\vec{u}_\pm = uR_\pm. \tag{19}$$

In the thermodynbamic limit, the empirical and expected errors can be simply expressed in terms of these variables, as explained in the appendix. Denoting by $w^m(D^m)$ the value of w minimizing the training error, we obtain the following results for the average empirical and expected errors, where the average is taken with respect to the probaility distribution of the data D^m. The average empirical error is given by the following remarkably simple expression,

$$\nu = \frac{1}{2}\left[H(\tau_+ + uR_+/\sigma_+) + H(\tau_- - uR_-/\sigma_-)\right], \tag{20}$$

where the parameters $\tau_\pm$ and $R_\pm$ are algorithm dependent. Their specific values can be obtained by solving the equations given in the appendix. The average expected error is similarly given by the expression

$$\varepsilon = \frac{1}{2}\left[H(uR_+/\sigma_+) + H(uR_-/\sigma_-)\right]. \tag{21}$$

In the case of the Hebb rule, the overlap variables $R_\pm$ can be written out explicitly (see also Barkai et. al.1993),

$$R_+ = \frac{1}{\sqrt{2}\sqrt{1+\sigma^2/\alpha u^2}} \qquad \text{(Hebb rule)}, \tag{22}$$

with $R_- = -R_+$ and $\sigma^2 = \sigma_+^2 + \sigma_-^2$. We note that for consistencey we must have $\tau_\pm \to 0$ when $\alpha \to \infty$, so that the empirical error converges to the expected error as it should.

4.1 Equal covariance matrices

In this case we take $\sigma_+ = \sigma_- = 1$ in which case the linear discriminant function is optimal. Moreover, as can clearly be seen in Eq. (6), the optimal bias vanishes (keeping in mind $\|\vec{u}_+\| = \|\vec{u}_-\|$). For large α we find, for all training functions considered, that the expected error dacays towards the optimal value, $\varepsilon_{\min}$, according to a power law

$$\varepsilon(\alpha) - \varepsilon_{\min} \approx \frac{c}{\alpha}, \tag{23}$$

where the constant c is again algorithm specific. However, since for the perceptron and Frean error functions the replica symmetric solution is unstable (see appendix) it is not inconceivable that the decay rate is somewhat different. For the Hebb rule, Eq. (17), the solution is exact and the value c is given by Barkai et. al. (1993). It is interesting that choosing $\vec{w}$ to minimize the empirical error $\nu(\vec{w})$, (the so called zero temperature Gibbs algorithm) yields an expected error which decays asymptotically like $1/\sqrt{\alpha}$, as observed by Barkai et. al. (1993). This implies that minimizing the empirical error is a sub-optimal strategy in this case. The dependence of the empirical and expected errors on the sample size α is plotted in figure 1 for the learning algorithms studied, in the case $u = 1$ and $\sigma_\pm = 1$. As can be seen in figure (1a), the Hebb rule yields the highest empirical error, being followed by the perceptron and Frean error functions. It should be noted however that the Hebb rule gives rise to the lowest expected error, as can be seen in figure (1b). We have added to figure (1b) results of numerical simulations performed with $d = 50$ and averaged over 100 cases. As can be seen, there is a small discrepency between the analytic and numerical results for the

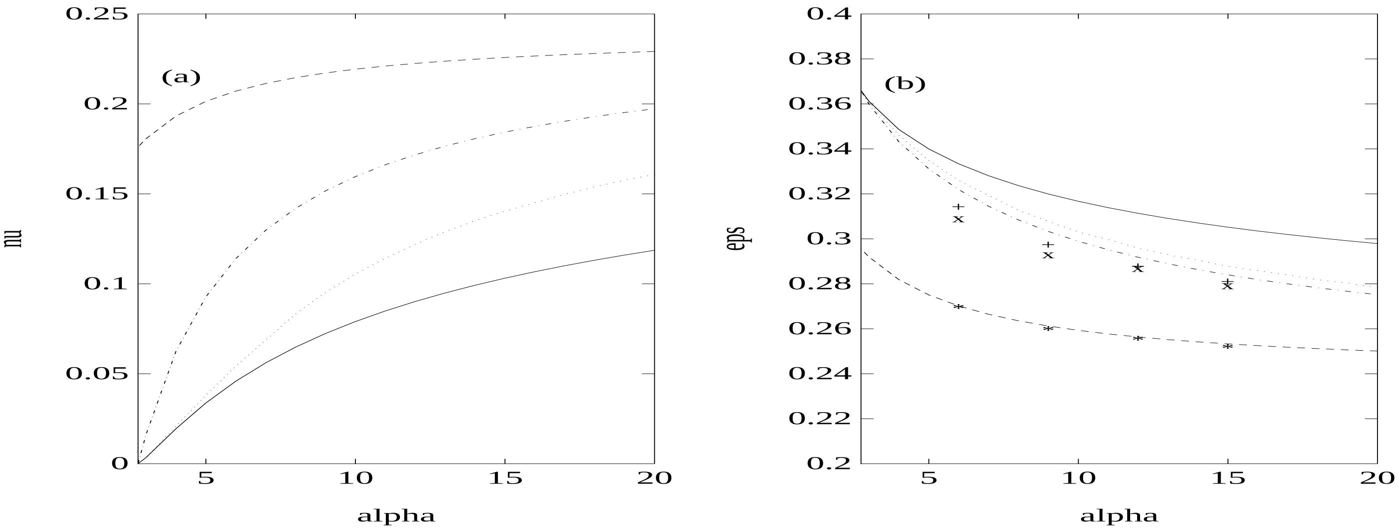

Figure 1: (a) Empirical error for a mixture of Gaussians with $u = 1$ and $\sigma_\pm = 1$. The lowest possible empirical error is given by the solid line, the dotted line is the result of the Frean algorithm (with $\lambda = 1$), the dashed-dotted line is that of the perceptron algorithm while the Hebb algorithm is given by the dashed line. (b) Same as (a), but for the expected error. Results of simulations for 50 inputs are presented by crosses for the perceptron algorithm, by + for Frean's algorithm and by $*$ for the Hebb rule. The size of the error bars is approximately $\pm(0.01 - 0.02)$.

perceptron and Frean functions. After having increased the system size to 200 and observing no noticable change we concluded that the difference is probably due to replica symmetry breaking effects, as discussed in the appendix. All numerical results were obtained using the conjugate gradient method to minimize Eq. (3) (keeping in mind the constraint $\|\vec{w}\| = 1$).

This superiority of the Hebb rule over the others is in fact not surprising, as can be seen from the following argument. As was claimed above, in the present case the optimal Bayes classifier is given in Eq. (10) by the difference between the Gaussian centers $\vec{u}_+$ and $\vec{u}_-$. Now, the Hebb rule can be written in the form

$$w_i = \frac{1}{\gamma\sqrt{d}} \left(\sum_{l\in+} \vec{x}^l - \sum_{l\in-} \vec{x}^l \right) , \tag{24}$$

where $l \in \pm$ refers to those inputs arising from class $\pm$ respectively. Now, it is well known that the sample average is the optimal estimate for the mean of a normal random variable, in the sense of maximum likelihood (Duda and Hart 1973). Thus, we expect that in this situation the Hebb-rule, which simply calculates the sample-mean for each class, will indeed be optimal. These results are an illustration of the observation of Vapnik (1982) that the weight value minimizing the empirical error is not necessarily optimal for finite sample sizes. In fact, in the present model it turns out that minimizing the empirical error is the worst strategy among those studied (for finite sample sizes).

We note in passing that the result for the Hebb rule can be derived without recourse to statistical mechanics, by using simple probabilistic arguments. For details of a similar calculation the reader is referred to Watkin (1992).

4.2 Unequal covariance matrices

As remarked in section 4.1, it would seem that the simplest learning algorithm, namely the Hebb rule, is optimal for the situation where the covariance matrices are both equal to the unit matrix. This prompted us to consider the slightly more complex situation described by Eq. (8) with $\sigma_+ \neq \sigma_-$, as a more realistic scenario. Since in this case the linear classifier is no longer Bayes optimal, it behooves us to find the *best* linear classifier. A surprisingly simple result (Fukunaga 1990) demonstrates that under the condition that $h_w(\vec{x}) = \vec{w} \cdot \vec{x}$ is normal, the best linear classifier is given by

$$\vec{w}^* = (a\Sigma_- + (1-a)\Sigma_+)^{-1} (\vec{u}_+ - \vec{u}_-), \tag{25}$$

where a is determined by the specific optimlaity criterion used. We note that in the limit of high dimension (large d) $h_w(\vec{x})$ is expected to be Gaussian under a wide range of conditions (central limit theorem), and thus the above result should be of wide applicability. In the case studied here, namely $\Sigma_\pm = \sigma_\pm^2 I$, we find that again the optimal linear classifier is proportional to $\vec{u}_+ - \vec{u}_-$, as in the case of equal variances. In this case however, the optimal bias term is no longer zero, as could be surmised on geometrical grounds. This would lead us to conjecture, based on our arguments in section 4.1, that the Hebb rule is optimal (for linear classifiers) in this case as well. This is in fact borne out by the the calculations, and can be seen clearly in figure 2 where we plot the expected error for several learning algorithms in the case $u = 1$, $\sigma_+ = 1$ and $\sigma_- = 0.3$. It is interesting to note that in this case the perceptron learning algorithm in fact produces slightly worse results (i.e. higher expected error) than those obtained by minimizing the empirical error (at least for the range of α appearing in the figure). In fact, we expect that as relative width of the two Gaussian centers decreases (keeping their centers fixed) the empirical error offers a continually improving estimate, since in this case the overlap between the two classes become negligible. We note also from figure (2b) that the Frean error function yields a lower expected error than that obtained by minimizing the empirical error, if the sample size is large enough. However, this effect is small and may be an artifact of the instability of the replica symmetric solution.

We conclude from the above results, that the Hebb rule is probably the optimal learning rule in the case where the input distribution within each class is a spherically symmetric Gaussian, i.e. $\Sigma_\pm = \sigma_\pm^2 I$. As can be seen from Eq. (25), in the case where $\Sigma_\pm$ are not spherically-symmetric we expect a different choice of weights to be optimal. Unfortunately, the analytic calculation of the learning curve in the case of a general covariance matrix $\Sigma_\pm$ becomes much more complicated, and will not be pursued in this paper.

5 Conclusion

We have studied the interaction between data distributions, learning algorithms and finite sample size. In particular, we have looked at the classification of mixtures of Gaussians with spherically symmetric covariance matrices, of possibly different variances. Comparing several learning algorithm for single-layer perceptrons, we have found the Hebb rule to be the best choice under the above conditions. In fact, we conjecture that the Hebb algorithm is optimal for spherically symmetric mixtures of Gaussians, as argued in section 4.

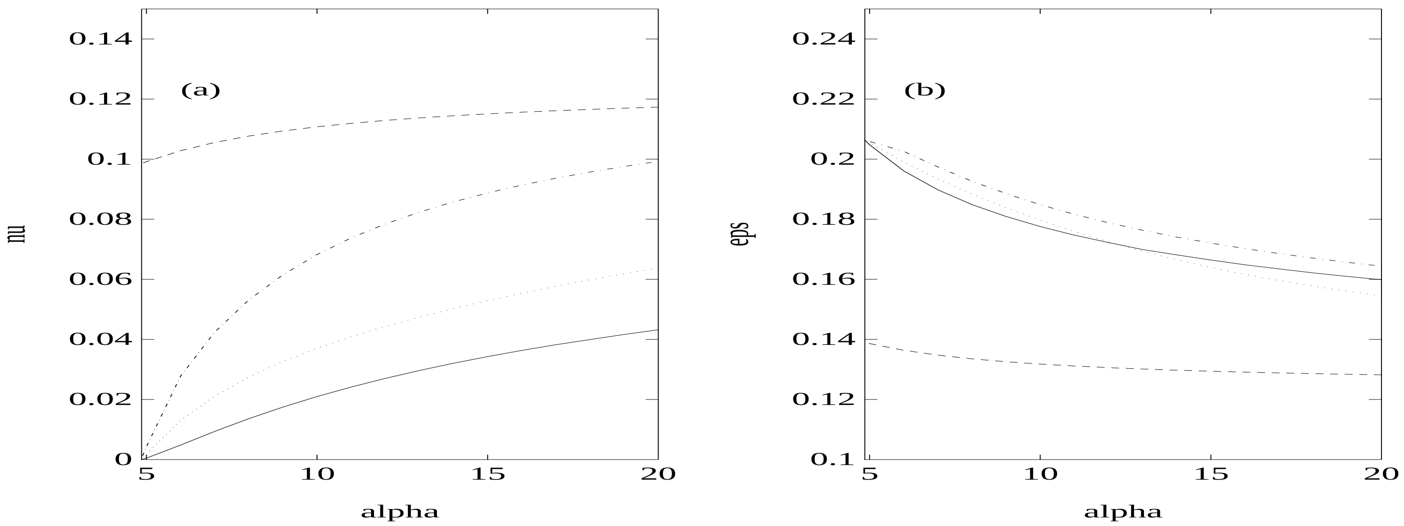

Figure 2: (a) Empirical error for a mixture of Gaussians with $u = 1$, $\sigma_+ = 1$ and $\sigma_- = 0.3$. The lowest possible empirical error is given by the solid line, the dotted line is the result of the Frean algorithm (with $\lambda = 1$), the dashed-dotted line is that of the perceptron algorithm while the Hebb algorithm is given by the dashed line. (b) Same as (a), but for the expected error.

An interesting result of our work is that algorithms yielding low empirical error do worse in terms of the expected error. This result is perhaps not surprising, since we have focused on situations where the data arising from the two classes overlap considerably, giving rise to problems of overfitting. The important case of optimal algorithms for general covariance matrices requires further study. In any case, it is clear that the Hebb rule is not optimal under these conditions, since it does not even converge asymptotically to the optimal weight matrix (see Eq. (6)). It would thus be worthwhile to further investigate how the other algorithms, which are more sensitive to the input distribution, behave under these conditions. A related line of research would be the investigation of quadratic classifiers, which are known to be optimal for arbitrary mixtures of Gaussians.

Acknowledgement The author is grateful to J.F. Fontanari for many helpful discussions and to M. Biehl and H.S. Seung for sending him copies of their work prior to publication.

Appendix

In this appendix we briefly describe the mathematical technique used to obtain the analytical results presented in this paper.

In order to calculate the minimal training error we first define a *partition function*, $Z(D^m)$, given by

$$Z_m(D^m) = \int d\vec{w}\ \delta(\|\vec{w}\|^2 - 1)e^{-\beta m \varepsilon_t(\vec{w}, D^m)}, \tag{26}$$

from which the average minimal training error is easily seen to be given by

$$\varepsilon_{\min} = -\frac{1}{m} \lim_{\beta\to\infty} \frac{d}{d\beta} \langle \ln Z \rangle_{D^m} . \tag{27}$$

In this expression, $\langle \ln Z_m \rangle_{D^m}$ stands for an average over the distribution of the data D^m. Thus, all results derived will be average case results. Without going into the mathematical details, which are lengthy but unilluminating, we present the general equations needed in order to calculate the overlap functions $R_\pm$ defined in Eq. (19), which are shown to be relevant for the calculation of the behavior of the system. For simplicity we present results for the case where the prior probabilities of the classes are equal, i.e. $p_+ = p_- = 1/2$. Our calculations were done using the elegant formulation of Griniasti and Gutfreund (1991), where details of a similar calculation can be found (see also Meir and Fontanari 1992). We focus on the regime above α_c, which is the value of α above which zero empirical error is impossible. All algorithms, except for the Hebb rule, yield identical results for $\alpha \le \alpha_c$, since they all lead to zero empirical (and training) error in this regime. In order to calculate $R_\pm$ we need to solve the following three equations for $R_\pm$ and the auxiliary variable x.

$$\begin{aligned}
&1 - (R_+^2 + R_-^2) - \frac{\alpha}{2}\left[\sigma_+^{-2}\int Dt\,(\lambda_+ - \sigma_+ t - uR_+)^2 + \sigma_-^{-2}\int Dt\,(\lambda_- - \sigma_- t + uR_-)^2\right] = 0 \\
&R_+ - \frac{\alpha u}{2\sigma_+^2}\int Dt\,(\lambda_+ - \sigma_+ t - uR_+) = 0 \\
&R_- + \frac{\alpha u}{2\sigma_-^2}\int Dt\,(\lambda_- - \sigma_- t + uR_-) = 0.
\end{aligned} \tag{28}$$

Here $Dt = e^{-t^2} dt/\sqrt{2\pi}$ is a Gaussian measure and $\lambda_\pm$ are the minima, respectively, of the functions

$$F_\pm(\lambda, x, R) = V(\lambda) + \frac{(\lambda - \sigma_\pm t \mp uR_\pm)^2}{2x\sigma_\pm^2}, \tag{29}$$

where the form of $V(\lambda)$ is given for the three error functions studied in this paper in section 2. The variables $\tau_\pm$ appearing in the empirical error results of Eq. (20) are given by $\sqrt{2x}$ in the case where the empirical error is minimized, by $\sigma_\pm x$ in the case of the perceptron function and by $2R_+\sigma_\pm/\alpha u$ for the Hebb algorithm (notice that in this case $R_- = -R_+$ irrespective of whether $\sigma_+ = \sigma_-$ or not). In all cases x is obtained through a solution of Eqs. (28). The expression for $\tau_\pm$ in the case of the Frean error function is much more complex, requiring the solution of several other equations, and will not be presented here.

The calculations reported above were done using the so called replica symmetric assumption (Mezard et. al. 1987). We have found however that this ansatz is incorrect in the regime considered (except for the Hebb rule, which can anyway be solved without recourse to replicas (Watkin et. al. 1992)). In spite of this fact, it has been found in many similar problems (see Watkin et. al. 1993 for a review) that the replica symmetric solution provides a good approximation even in the regime where replica symmetry breaking takes place. The numerical results presented in section 4 lend further credence to this claim.

References

Barkai N., H.S. Seung and H. Sompolinsky, 1993. Scaling laws in learning classification tasks, preprint Hebrew University.

Biehl M. and A. Mietzner, 1993. Statistical mechanics of unsupervised learning, preprint Universität Würzburg.

Duda R.O. and P.E. Hart 1973. *Pattern Classification and Scene Analysis*, Wiley, New-York.

Gardner E. The space of interactions in neural network models. *J. Phys. A* 21: 257-270.

Gardner E. and B. Derrida. Optimal storage properties of neural networks, *J. Phys. A* 21: 271-284.

Griniasti M. and H. Gutfreund, 1991. Learning and retrieval in attractor neural networks above saturation, *J. Phys. A* 24: 715.

Frean M, 1992. A "thermal" perceptron learning rule, *Neural Computation* 4:9 46-957.

Fukunaga K., 1990. *Introduction to Statistical Pattern Recognition*, Academic Press, San Diego, U.S.A.

Meir R. and J.H. Fontanari 1992. Calculation of learning curves for inconsistent algorithms, *Phys. Rev. A*: 45(12): 8874-8884.

Mezard M., G. Parisi and M.A. Virasoro, 1987. *Spin Glass Theory and Beyond*, World Scientific, Singapore.

Minsky M. and S. Papert, 1988. *Perceptrons*, MIT Press, Cambridge, MA.

Vapnik V.N. 1982. *Estimation of dependences based on empirical data*, Springer-Verlag, Berlin.

Watkin T.L.H., A. Rau, D. Bolle and J. van Mourik 1992. Learning multi-class classification problems, *J. Phys. (Paris)* I2: 167-180.

Watkin T.L.H., A. Rau and M. Biehl, 1993. The statistical mechanics of learning a rule, *Rev. Mod. Phys.*, in press.