LU TP 92-18

June 1992

Pattern Discrimination Using Feed-Forward Networks - a Benchmark Study of Scaling Behaviour

Thorsteinn Rögnvaldsson[1]

Department of Theoretical Physics, University of Lund,
Sölvegatan 14 A, S-223 62 Lund, Sweden

Submitted to **Neural Computation**

Abstract:
The discrimination powers of Multilayer perceptron (MLP) and Learning Vector Quantisation (LVQ) networks are compared for overlapping Gaussian distributions. It is shown, both analytically and with Monte Carlo studies, that the MLP network handles high dimensional problems in a more efficient way than LVQ. This is mainly due to the sigmoidal form of the MLP transfer function, but also to the the fact that the MLP uses hyper-planes more efficiently. Both algorithms are equally robust to limited training sets and the learning curves fall off like $1/M$, where M is the training set size, which is compared to theoretical predictions from statistical estimates and Vapnik-Chervonenkis bounds.

[1] denni@thep.lu.se

Introduction

The task of discriminating between different classes of input patterns has proven to be well suited for artificial neural networks (ANN). Standard methods, like making cuts or discriminant analysis, are repeatedly being outperformed by non-linear ANN algorithms, where the most extensively used algorithms are the feed-forward Multilayer Perceptron (MLP) [1] and the Learning Vector Quantisation (LVQ) [2]. Both algorithms have shown good discrimination and generalisation ability, although some confusion prevails on their performance on realistic large sized problems, especially concerning their parsimony in parameters to fit data - an important issue when algorithms are transferred to hardware.

This paper compares, analytically and with Monte Carlo simulations, the discrimination power of the MLP and LVQ algorithms on separating two Gaussian distributions. Two classes are sufficient since the results carry over to problems with more classes. The problem is designed to resemble "real-life" situations with many input nodes and overlapping distributions, making the classification fuzzy. Discrimination is thus only possible down to a minimum error; the *Bayes limit* [3]. It is found, in contrast to previous results [4, 5], that the MLP is more efficient than the LVQ algorithm on heavily overlapping distributions. The sensitivities of the two algorithms to limited training data are also examined and compared to theoretical predictions.

The Problem

The problem [4] consists of two overlapping Gaussian distributions, P_1 and P_2, of dimensionality d and normalised to unity with standard deviations $\sigma_1 = 1.0$ and $\sigma_2 = 2.0$. Two versions are generated; one where the distributions have the same mean, referred to as the "hard" case, and one where their means are separated by a distance ξ, referred to as the "easy" case (notation follows [4]):

$$P_1(\vec{r}) = (\sigma_1\sqrt{2\pi})^{-d}\exp[-\frac{r^2}{2\sigma_1^2}] \quad (1)$$

$$P_2(\vec{r}) = (\sigma_2\sqrt{2\pi})^{-d}\exp[-\frac{(\vec{r}-\vec{\xi})^2}{2\sigma_2^2}] \quad (2)$$

where $\vec{\xi} = \vec{0}$ for the "hard" case, and $\vec{\xi} = (2.32, 0, 0, \ldots, 0)$ for the "easy" case. Bayes limit equals $\int \min[P_1, P_2]$ which is easily calculated for Gaussian distributions (see table 1).

dimension d	Bayes limit "hard" case	Bayes limit "easy" case
2	26.4%	16.4%
3	21.4%	13.8%
4	17.6%	11.6%
5	14.8%	9.8%
6	12.4%	8.4%
7	10.6%	7.2%
8	9.0%	6.2%

Table 1: Bayes limit for the "hard" and "easy" cases for dimensions $2 \leq d \leq 8$.

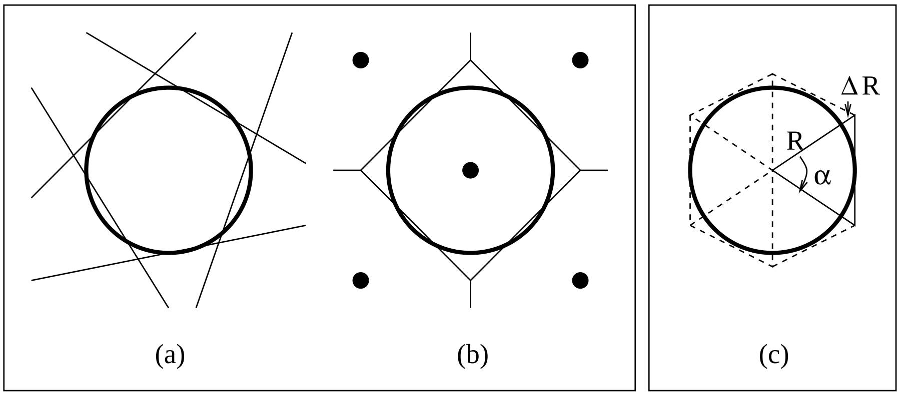

Figure 1: **(a)** The MLP approximation to a sphere and **(b)** the LVQ tessellation for the same sphere (dots correspond to reference vectors). **(c)** The resulting polyhedron and quantities used in the text.

Analytical Results

The optimal classification boundary between the two Gaussian distributions is in both the "hard" and "easy" cases the surface of a d-dimensional hyper-sphere, defined by $P_1(\vec{r}) = P_2(\vec{r})$. This is the surface the networks try to reproduce. The MLP uses "hyperplanes" (see fig. 1a) and is able to "cut corners" if a sigmoidal transfer function is used. The LVQ network divides the space by means of reference vectors, so-called "tessellation", and if the number of reference vectors is moderate the LVQ algorithm will result in one reference vector being inside the sphere and the others outside (fig. 1b). Both algorithms thus produce a polyhedral-like reproduction of the sphere (fig. 1c) and the generalisation error E can be expressed as

$$E \approx B + \Delta V \Delta P \tag{3}$$

where B is Bayes limit, ΔV is the deficit volume of the polyhedron as compared to the hyper-sphere, and ΔP the average absolute difference between the two distributions inside that volume.

This error can be estimated for the "hard" case, where the polyhedron is assumed to be spherically symmetric: If the number of planes bounding the polyhedron is N, it will consist of N conical sectors with opening angle α (see fig. 1c). If $N \gg d$ the end-cap of the cone is a $(d-1)$-dimensional hexagon, approximately a $(d-1)$-dimensional hyper-sphere, and α will be given by

$$\frac{A_d}{NA_{d-1}} \approx \int_0^{\alpha/2} d\theta \sin^{d-2}\theta \approx \frac{(\alpha/2)^{d-1}}{d-1} \tag{4}$$

where $A_d = \frac{2\pi^{d/2}}{\Gamma(d/2)}$, and Γ is the Gamma function[2]. It is assumed in the last step that N is so large that α is small.

The volumes of the cone and the sector are given by

$$V_{cone} = \frac{A_{d-1}}{(d-1)d} R^d \tan^{d-1}(\alpha/2) \tag{5}$$

$$V_{sector} = \frac{A_d}{dN} R^d \tag{6}$$

where $R = \sqrt{2d \ln[\sigma_2/\sigma_1]\sigma_1^2\sigma_2^2/(\sigma_2^2-\sigma_1^2)}$ is the radius of the optimal classification hyper-sphere.

The deficit volume is approximated by expanding in power series

$$\begin{aligned}\Delta V &= N(V_{cone} - V_{sector}) \\ &\approx N\left[\frac{A_{d-1}}{A_d}\frac{(\alpha/2)^{d-1}}{(d-1)}\left(1+\frac{(\alpha/2)^2}{3}\right)^{d-1} - 1/N\right]\frac{A_d R^d}{d} \\ &\approx \frac{(\alpha/2)^2}{3} A_d R^d \end{aligned} \tag{7}$$

where eq. (4) has been used in the last step.

If ΔV is not too large, the density of patterns inside can be assumed constant. For $\sigma_1 < \sigma_2$ and $\Delta R \ll R$ one gets

$$\begin{aligned}\Delta P &\approx |P_1 - P_2|_{R+\Delta R/2} \approx P_R \left|\exp[-R\Delta R/2\sigma_1^2] - \exp[-R\Delta R/2\sigma_2^2]\right| \\ &\approx P_R R^2 \frac{\Delta R}{2R}\left(\frac{1}{\sigma_1^2}-\frac{1}{\sigma_2^2}\right) \approx P_R R^2 \frac{(\alpha/2)^2}{8}\left(\frac{1}{\sigma_1^2}-\frac{1}{\sigma_2^2}\right)\end{aligned} \tag{8}$$

[2] A_d comes from the surface area $A = A_d r^{d-1}$ of a hyper-sphere of dimension d and radius r.

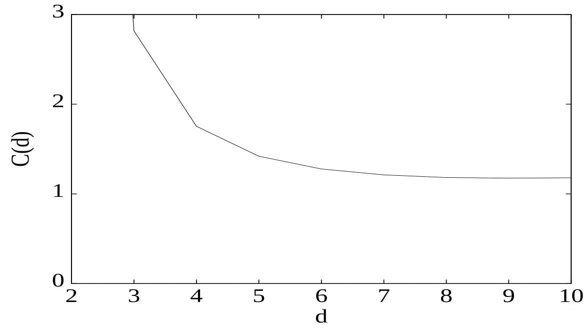

Figure 2: The factor $C(d)$ for $2 < d < 10$ when $\sigma_1 = 1.0$ and $\sigma_2 = 2.0$.

where P_R is the value of the distributions at the border.

Inserting eqs. (7) and (8) into (3) gives

$$\frac{E - B}{B} \approx C(d) d N^{-4/(d-1)} \propto d N^{-4/(d-1)} \tag{9}$$

where

$$C(d) = \frac{A_d \ln(\sigma_2/\sigma_1)}{12B} R^d P_R \left(\frac{(d-1)A_d}{A_{d-1}} \right)^{4/(d-1)} \tag{10}$$

is approximately constant for $4 < d < 10$ (see fig. 2).

The true value of $C(d)$ is not exactly given by eq. (10). Equation (9) should instead be considered as a general scaling relation for heavily overlapping distributions of dimension $d < 10$, under the condition that $N \gg d$ and α can be considered small. For comparison with Monte Carlo simulations a value of $C(d) = 0.5$ was used.

Deriving a similar expression for the "easy" case would be arduous; the polyhedron is not spherically symmetric and the simplifications above are not possible. The generalisation error is however bounded from above by expression (9): If $\xi \to \infty$ the distributions are perfectly separated by one hyper-plane, with E equal to zero, and if $\xi \to 0$ the "hard" case is recovered. Hence, the "easy" problem never scales worse than the corresponding "hard" problem.

Each reference vector in an LVQ network with N_{LVQ} computational units has at most $N_{LVQ} - 1$ nearest neighbours. For moderate N_{LVQ}, with a setup similar to figure 1b, the number of hyper-planes bounding the polyhedron is $N = (N_{LVQ} - 1)$. On the other hand, more than one reference vector will be inside the sphere if N_{LVQ} is large and the number of hyper-planes bounding the polyhedron will be $N < (N_{LVQ} - 1)$. The LVQ generalisation error will thus scale like eq. (9) for moderate N_{LVQ} and worse for large N_{LVQ}.

Each hidden unit in an MLP with a Heaviside transfer function corresponds to one hyper-plane. Hence, an MLP with N_{MLP} hidden units has $N = N_{MLP}$. A smooth sigmoidal transfer function allows the MLP to "cut corners" in the polyhedron, and the generalisation error subsequently scales better than (9). How much better is difficult to predict, but analytical results [6] imply that a sigmoid allows a decrease of the hidden units with at least a factor of two, compared to a Heaviside.

Monte Carlo Studies

The dimension d of the Gaussian distributions varied between 2 and 8, and 12 different architectures of MLP and LVQ networks were set up for each value of d. An ensemble of 100 networks was trained for each configuration to measure the worst, best, and average discrimination performances.

The LVQ networks had d input units and N_{LVQ} processing units, where $N_{LVQ} \in \{d, d+4, d+8, \ldots, d+4\cdot 11\}$. The weights were initialised with the on-line *k-means* clustering algorithm [7] for a duration of 150 epochs, with randomly selected starting points. LVQ updating was then applied for an additional 150 epochs, while the learning rate was lowered geometrically from $\eta = 0.1$ down to $\eta = 0.001$. One epoch corresponded to 1000 patterns. This choice of parameters and initialisation was made to match those of ref. [4].

The MLP networks had d input units and one hidden layer with N_{MLP} hidden units, where $N_{MLP} \in \{d, d+2, d+4, \ldots, d+2\cdot 11\}$. The initial weights were randomly picked from a flat distribution $\omega \in [-\omega_0, \omega_0]$, with $\omega_0 = 0.1/(\text{maximum "fan-in"})^3$. On-line backpropagation with summed square error was used to train the networks (in the "hard" case a Langevin form was used; $\Delta\omega = -\eta\nabla E +$ "noise", to avoid local minima). The learning rates were scaled in inverse proportion to the "fan-in" of the layer, $\eta \propto 1/$"fan-in", and dynamically changed with the "bold driver" method; increasing η if the network is improving and decreasing it otherwise. The momentum parameter was kept constant at $\alpha = 0.5$.

The largest architectures were also trained with 7 different training sets, with sizes $M \in \{10, 10^{3/2}, 10^2, 10^{5/2}, 10^3, 10^{7/2}, 10^4\}$, to estimate the learning curve. For each value of M the number of epochs was changed to keep the total number of presentations constant at $4\cdot 10^5$ for the MLP and $3\cdot 10^5$ for the LVQ networks.

All simulations were performed with the network program package `JETNET 2.0` [8].

[3] The "fan-in" of a unit is the number of units feeding to it.

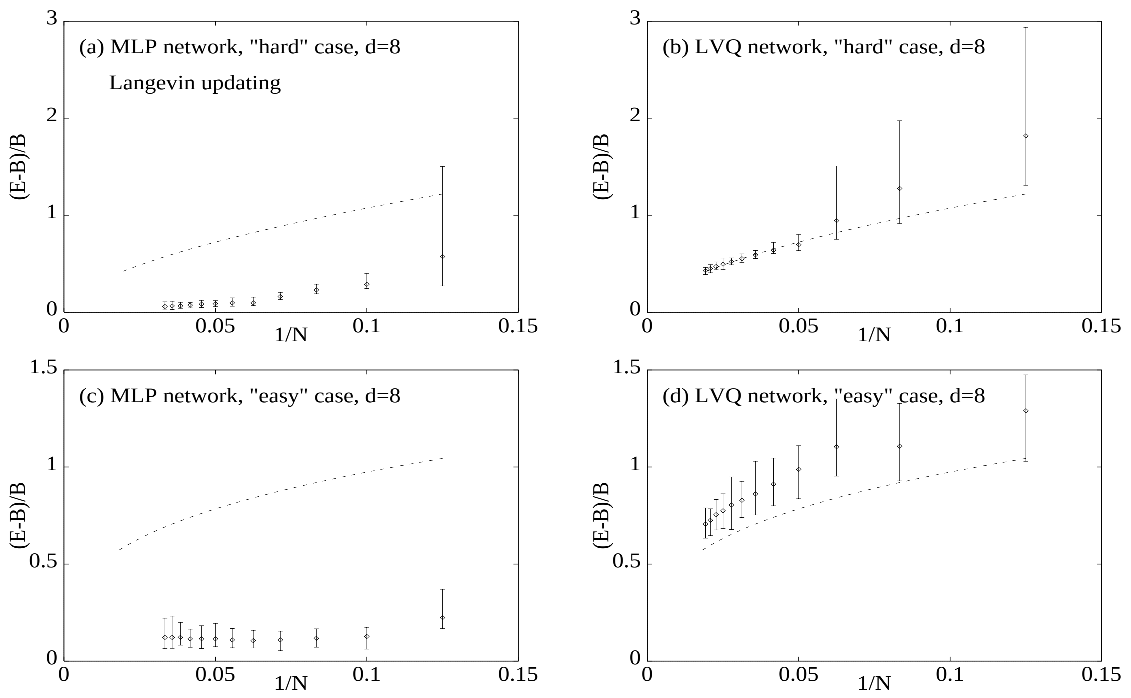

Figure 3: Minimum, maximum and average values of the scaled generalisation error for $d = 8$ using **(a)** MLP and **(b)** LVQ on the "hard" task, and **(c)** MLP and **(d)** LVQ on the "easy" task. The dashed line in (a) and (b) is the predicted $(E - B)/B \propto dN^{-4/(d-1)}$, whereas in (c) and (d) it is a fitted $(E - B)/B \propto dN^{-5/2d}$

Figure 3 shows the $E(N)$-behaviour for $d = 8$, training on continuously generated data (figures for $d < 8$ are similar). Indicated for the "hard" case is the predicted scaling relation of eq. (9) with $C = 0.5$. The LVQ errors follow the predicted behaviour well, whereas the MLP errors scale better, as expected. For the "easy" case the LVQ error scales like $(E - B)/B \propto dN^{-5/2d}$.

The worst case upper bound generalisation error for this kind of binary classification problem is expected to be $E \leq d_{VC}/M$, where d_{VC} is the Vapnik-Chervonenkis dimension of the network [9]. Figure 4 shows that this bound is indeed well above the actual learning curves if the number of weights is used as an approximate value of d_{VC}. The learning curves are also well described by $E \propto 1/(M + M_0)$, predicted by statistical learning theories for problems with continuous generalisation spectrum [10], and the algorithms are equally robust to limited training sets.

An attempt was also made to estimate the prior generalisation spectrum. The generalisation ability of 10,000 untrained networks was tested and the learning curves numerically calculated from

$$E(M) = 1 - \frac{\langle g^{M+1} \rangle_0}{\langle g^M \rangle_0} \tag{11}$$

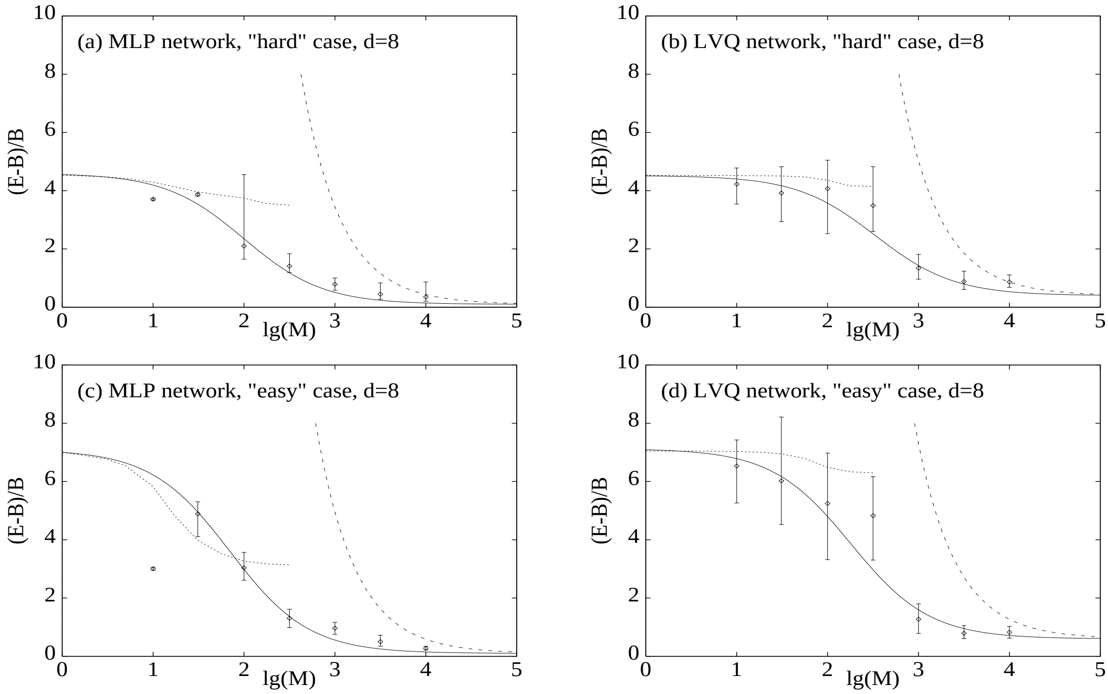

Figure 4: The "hard" task learning curves for **(a)** MLP and **(b)** LVQ, and the "easy" task learning curves for **(c)** MLP and **(d)** LVQ. The solid line is a fitted $(E-B)/B \propto 1/(M + \text{constant})$ and the dashed line shows W/M where W is the number of weights. The dotted lines are the curves numerically calculated from the prior generalisation spectrum.

where $\langle g^M \rangle_0$ is the M^{th} moment of the prior generalisation distribution [10]. The resulting curves (dotted lines in fig. 4) do not follow the true learning curves. Statistical errors, however, are large already for $\lg M \approx 1.5$ and higher statistics was not pursued due to the CPU consumption involved.

Conclusions and Discussion

The results demonstrate that the MLP architecture, with sigmoidal transfer functions, is superior to LVQ for discriminating between heavily overlapping distributions with convex borders. This is in contrast to previous results comparing LVQ and MLP architectures [4, 5], but in agreement with results for Boltzmann [4] and Mean Field machines [11]. The MLP method is highly efficient and the more viable alternative for problems with many inputs. Furthermore, both algorithms are equally robust to limited training data and their learning curves follow a $1/M$ behaviour.

These results are also valid for binary problems where more than two classes of input patterns are present. An MLP with n output units can be used for a n-class problem.

The output values will then correspond to the Bayesian *a posteriori* likelihoods that the input pattern belongs to the specific class [12]. A discrimination decision can therefore be made from these output values and the number of hidden units will in the worst case just be multiplied by n. The number of units in the LVQ network will for moderate N_{LVQ} also scale multiplied by n and the relative performances of MLP and LVQ will hence be approximately unchanged. LVQ may of course be more efficient than MLP for some pathological problems with very large $n \gg d$ and where each class only needs one or two reference vectors.

The reason the MLP performs well already for low N_{MLP} is clearly its sigmoidal transfer function, making it possible to smooth the corners of the polyhedron. Allowing direct input to output connections in the MLP would increase the discrimination power further [6, 11].

The LVQ algorithm can be augmented with more advanced evaluations of the output signals [5, 13] and thus be made to perform better. With such superstructures one can achieve discrimination performances comparable to the MLP, but at the price of many more parameters.

Acknowledgements

This work has benefited from discussions with Bo Söderberg.

References

[1] D. Rumelhart and J. McLelland (eds.) *Parallel Distributed Processing* (Vol. 1), MIT Press (1986)

[2] T. Kohonen, *Self-organization and Associative Memory*, 3^{rd} ed., Springer-Verlag, Heidelberg (1990)

[3] See e.g. R. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*, John Wiley & Sons (1973).

[4] T. Kohonen, G. Barna and R. Chrisley, "Statistical Pattern Recognition with Neural Networks: Benchmarking Studies", *IEEE* 2^{nd} *Int. Conf. on Neural Networks* **I**, 61-68, San Diego, CA: SOS Printing (1988)

[5] G. Barna and K. Kaski, "Stochastic vs. Deterministic Neural Networks for Pattern Recognition", *Physica Scripta* **T33**, 110-115 (1990)

[6] E. Sontag, "Feedforward Nets for Interpolation and Classification", to appear in *Journal of Computer and System Sciences*, (1992)

[7] J. MacQueen. "Some Methods for Classification and Analysis of Multivariate Observations", in L. M. LeCam and J. Neyman (eds.), *Proc. 5th Berkeley Symposium on Math. Stat. and Prob.*, U. California Press, Berkeley (1967).

[8] L. Lönnblad, C. Peterson and T. Rögnvaldsson, "Pattern Recognition in High Energy Physics with Artificial Neural Networks - JETNET 2.0", *Computer Physics Communications* **70**, 167-182 (1992)

[9] See e.g. D. Cohn and G. Tesauro, "How Tight are the Vapnik-Chervonenkis Bounds?", *Neural Computation* **4**, 249-269 (1992)

[10] D. Schwartz, V. Samalam, S. Solla and J. Denker, "Exhaustive Learning", *Neural Computation* **2**, 374-385 (1990)

[11] C. Peterson and E. Hartman, "Explorations of the Mean Field Theory Learning Algorithm", *Neural Networks* **2**, 475-494 (1989)

[12] M. Richard and R. Lippmann, "Neural Networks Estimate Bayesian *a posteriori* Probabilities", in *Neural Computation* **3**, 461-483 (1991)

[13] I. Scabai, F. Czakó and Z. Fodor, "Combined Neural Network - QCD Classifier for Quark and Gluon Jet Separation", *Nuclear Physics* **B374**, 288-308 (1992)