Analytical interpretation of feed-forward nets outputs after training *

Lluis Garrido and Sergio Gómez †

Dept. d'Estructura i Constituents de la Matèria,
Facultat de Física, Universitat de Barcelona,
Diagonal 647, 08028 Barcelona, Spain,
and
Institut de Física d'Altes Energies,
Universitat Autònoma de Barcelona,
08193 Bellaterra (Barcelona), Spain.

Abstract

The minimization quadratic error criterion which gives rise to the back-propagation algorithm is studied using functional analysis techniques. With them, we recover easily the well-known statistical result which states that the searched global minimum is a function which assigns, to each input pattern, the expected value of its corresponding output patterns. Its application to classification tasks shows that only certain output class representations can be used to obtain the optimal Bayesian decision rule. Finally, our method permits the study of other error criterions, finding out, for instance, that absolute value errors lead to medians instead of mean values.

Published in *International Journal of Neural Systems* **7** (1996) 19–27.

*This research is partly supported by the 'Comissionat per Universitats i Recerca de la Generalitat de Catalunya' and by EU under contract number CHRX-CT92-0004

†Present address: Dept. d'Enginyeria Informàtica (ETSE), Univ. Rovira i Virgili, Tarragona (Spain)

1 Introduction

During the last few years neural networks have become increasingly popular. Their ability to classify and predict from examples is what has made them most interesting. Among the different types of networks, the ones which have found more applications are the feed-forward nets whose training is based in the minimization of a squared error criteria, to the greatest extent using some version of the famous back-propagation algorithm.

From the point of view of statistics, supervised learning is just a synonymous of regression, and it is well-known that the regression 'line' which minimizes the quadratic error criterion is the function formed by the expectation values of the outputs conditioned to the inputs. In this work we make use of functional derivatives to find this unconstrained global minimum, which easily allows for the minimization of more involved error criterions. As an example, we have solved the error functions of the form of an absolute value to any positive integer power.

Once the theoretical unconstrained global minimum has been found, it is possible to investigate the role played by the representation given to the training output patterns, specially whenever the number of different possible outputs is finite (e.g. in classification tasks). For instance, we will show that unary output representations are the simplest choice to achieve Bayesian decisions, but not the only one, and that binary output representations should never be used, since they lead to a loss of information which induces errors in the interpretation of the final outputs.

It must be stressed that our results will be derived with the only assumption that global minima are possible to be calculated, without any reference to the intrinsic difficulty of this problem nor to its dependence on the shape of the net; in fact, it need not be a neural network. That is, the word 'net' should be understood as a short for 'big enough family of functions'.

2 Analytical interpretation of the net output

Let $(\xi, \zeta) \in \mathcal{X} \times \mathcal{Z}$ denote a certain pair of input-output patterns which has been produced with a given experimental setup. Since the sets $\mathcal{X}$ and $\mathcal{Z}$ are arbitrary, it is convenient to represent each pattern by a real vector in such a way that there is a one-to-one correspondence between vectors and feature patterns. We will make use of the vectors $\boldsymbol{x} \in \mathbb{R}^n$ for the input patterns and $\boldsymbol{z}(\boldsymbol{x}) \in \mathbb{R}^m$ for the output ones.

If $\{(\boldsymbol{x}^\mu, \boldsymbol{z}^\mu),\ \mu = 1, \ldots, N\}$ is a representative random sample of pairs input-output, our goal is to find the net

$$\boldsymbol{o} :\ \boldsymbol{x} \in \mathbb{R}^n \longmapsto \boldsymbol{o}(\boldsymbol{x}) \in \mathbb{R}^m \tag{2.1}$$

which closely resembles the unknown correspondence process. The least squares

estimate is that which produces the lowest mean squared error $E[\boldsymbol{o}]$, where

$$E[\boldsymbol{o}] \equiv \frac{1}{2N} \sum_{\mu=1}^{N} \sum_{i=1}^{m} \lambda_i(\boldsymbol{z}^\mu, \boldsymbol{x}^\mu) \; (o_i(\boldsymbol{x}^\mu) - z_i^\mu)^2 \; . \tag{2.2}$$

Usually the λ_i functions are set to 1. However, there are times in which it is useful to weight each contribution to the error with a term depending on the pattern. For instance, if the values of the desired outputs are known with uncertainties $\sigma_i(\boldsymbol{z}^\mu, \boldsymbol{x}^\mu)$, the right fitting or 'chi-squared' error should be

$$E[\boldsymbol{o}] \equiv \frac{1}{2N} \sum_{\mu=1}^{N} \sum_{i=1}^{m} \left(\frac{o_i(\boldsymbol{x}^\mu) - z_i^\mu}{\sigma_i(\boldsymbol{z}^\mu, \boldsymbol{x}^\mu)} \right)^2 \; . \tag{2.3}$$

A typical example occurs when we want to estimate a model using a histogram with k^μ events in the μ-th bin, $\mu = 1, \ldots, N$. In this case, the least squares method [3] consists in obtaining the model from the minimization of

$$\chi^2(\boldsymbol{\theta}) = \sum_{\mu=1}^{N} \frac{(o(\boldsymbol{x}^\mu; \boldsymbol{\theta}) - k^\mu)^2}{k^\mu} \; , \tag{2.4}$$

where $\boldsymbol{\theta}$ are the parameters to be estimated. (In (2.4) we have taken $\sigma(k^\mu, \boldsymbol{x}^\mu) \approx \sqrt{k^\mu}$, which is a valid approximation for large values of k^μ).

Under the three hypothesis that: (i) the different measurements ($\mu = 1, \ldots, N$) are independent (i.e., viewed as probability theory objects, they define independent random variables), (ii) the underlying probability distribution of the differences $o_i(\boldsymbol{x}^\mu) - z_i^\mu$ has zero mean, $m_\mu = 0 \, , \; \forall \mu$, and (iii) that, for instance, the mean square deviations σ_μ are uniformily bounded, $\sigma_\mu < K$, for all $\mu = 1, 2, \ldots, N$ (actually, in order to make use of Kolmogorov's theorem it is enough that $\displaystyle\sum_{\mu=1}^{N} \frac{\sigma_\mu}{\mu^2} < +\infty$, for any N), the *Strong Law of Large Numbers* applies (see e.g. [4, 7]). It tells us that, with probability one (i.e., in the usual almost-everywhere convergence, common to the theory of functions and functional analysis) the limiting value of (2.2) for large N is given by

$$\begin{aligned} E[\boldsymbol{o}] &= \frac{1}{2} \sum_{i=1}^{m} \int_{\mathbb{R}^n} d\boldsymbol{x} \int_{\mathbb{R}^m} d\boldsymbol{z} \, p(\boldsymbol{z}, \boldsymbol{x}) \, \lambda_i(\boldsymbol{z}, \boldsymbol{x}) \, [o_i(\boldsymbol{x}) - z_i]^2 \\ &= \frac{1}{2} \sum_{i=1}^{m} \int_{\mathbb{R}^n} d\boldsymbol{x} \, p(\boldsymbol{x}) \int_{\mathbb{R}^m} d\boldsymbol{z} \, p(\boldsymbol{z}|\boldsymbol{x}) \, \lambda_i(\boldsymbol{z}, \boldsymbol{x}) \, [o_i(\boldsymbol{x}) - z_i]^2 \end{aligned} \tag{2.5}$$

where $p(\boldsymbol{z}, \boldsymbol{x})$ is the joint probability density of the random variables $\boldsymbol{z}$ and $\boldsymbol{x}$ in the sample, $p(\boldsymbol{x})$ stands for the probability density of $\boldsymbol{x}$, and $p(\boldsymbol{z}|\boldsymbol{x})$ is the

conditional probability density of $\boldsymbol{z}$ knowing that the former random variable has taken on the value $\boldsymbol{x}$.

Notice that the first two of the conditions for the validity of the strong law of large numbers are naturally satisfied in most cases. In fact, while the first one is equivalent to the usual rule that the practical measurements must be always done properly (which is generally assumed), the second just tells us that the net is also to be constructed conveniently in order to fulfill the goal of closely resembling the unknown correspondence process (see above). But we also take for granted that we will be always able to do this, in the end. The third condition, however, is of a rather more technical nature and seems to be difficult to realize from the very begining (or even at the end, in a strict sense!). In practice, the thing to do is obviously to check *a posteriori* that it is fulfilled for N large enough, and to convince ourselves that there is no reason (in the case treated) for it to be violated at any value of N. We do think that this condition prevents one from beeing able to consider the use of the strong law of large numbers as something that can be 'taken for granted' in general, in the situation described in this paper. This comment should be considered as a warning against the apparently indiscriminate application of the law which can be found sometimes in the related literature.

Assuming no constraint in the functional form of $\boldsymbol{o}(\boldsymbol{x})$, the minimum $\boldsymbol{o}^*(\boldsymbol{x})$ of E is easily found by annulling the first functional derivative:

$$\begin{aligned}
\frac{\delta E[\boldsymbol{o}]}{\delta o_j(\boldsymbol{x})} &= \sum_{i=1}^{m} \int_{\mathbb{R}^n} d\boldsymbol{x}'\, p(\boldsymbol{x}') \int_{\mathbb{R}^m} d\boldsymbol{z}\, p(\boldsymbol{z}|\boldsymbol{x}')\, \lambda_i(\boldsymbol{z},\boldsymbol{x}')\, [o_i(\boldsymbol{x}') - z_i]\, \delta_{ij}\, \delta(\boldsymbol{x}-\boldsymbol{x}') \\
&= p(\boldsymbol{x}) \int_{\mathbb{R}^m} d\boldsymbol{z}\, p(\boldsymbol{z}|\boldsymbol{x})\, \lambda_j(\boldsymbol{z},\boldsymbol{x}) [o_j(\boldsymbol{x}) - z_j] \\
&= p(\boldsymbol{x})\, [o_j(\boldsymbol{x}) \langle \lambda_j(\boldsymbol{z},\boldsymbol{x}) \rangle_{\boldsymbol{x}} - \langle \lambda_j(\boldsymbol{z},\boldsymbol{x})\, z_j \rangle_{\boldsymbol{x}}] = 0 \qquad (2.6)
\end{aligned}$$

implies that the searched minimum is

$$o_j^*(\boldsymbol{x}) = \frac{\langle \lambda_j(\boldsymbol{z},\boldsymbol{x})\, z_j \rangle_{\boldsymbol{x}}}{\langle \lambda_j(\boldsymbol{z},\boldsymbol{x}) \rangle_{\boldsymbol{x}}}\,,\ \forall \boldsymbol{x} \in \mathbb{R}^n \text{ such that } p(\boldsymbol{x}) \neq 0\,,\ j = 1, \ldots, m\,, \qquad (2.7)$$

where we have made use of the *conditional expectation values*

$$\langle f(\boldsymbol{z},\boldsymbol{x}) \rangle_{\boldsymbol{x}} \equiv \int_{\mathbb{R}^m} d\boldsymbol{z}\, p(\boldsymbol{z}|\boldsymbol{x})\, f(\boldsymbol{z},\boldsymbol{x}) \qquad (2.8)$$

i.e. the average of any function f of the output vectors $\boldsymbol{z}$ once the input pattern $\boldsymbol{x}$ has been fixed. Eq. (2.7) is the key expression from which we will derive the possible interpretations of the net output. An alternative proof can be found for instance in [11].

From a practical point of view unconstrained nets do not exist, which means that the achievable minimum $\tilde{\boldsymbol{o}}(\boldsymbol{x})$ is in general different to the desired $\boldsymbol{o}^*(\boldsymbol{x})$.

The mean squared error between them is written as

$$\varepsilon[\tilde{\boldsymbol{o}}] \equiv \frac{1}{2} \sum_{i=1}^{m} \int_{\mathbb{R}^n} d\boldsymbol{x}\, p(\boldsymbol{x}) \int_{\mathbb{R}^m} d\boldsymbol{z}\, p(\boldsymbol{z}|\boldsymbol{x})\, \lambda_i(\boldsymbol{z},\boldsymbol{x})\, [\tilde{o}_i(\boldsymbol{x}) - o_i^*(\boldsymbol{x})]^2 \,. \tag{2.9}$$

However, it is straightforward to show (using eq. (2.7)) that

$$E[\boldsymbol{o}] = \varepsilon[\boldsymbol{o}] + \frac{1}{2} \sum_{i=1}^{m} \int_{\mathbb{R}^n} d\boldsymbol{x}\, p(\boldsymbol{x}) \int_{\mathbb{R}^m} d\boldsymbol{z}\, p(\boldsymbol{z}|\boldsymbol{x})\, \lambda_i(\boldsymbol{z},\boldsymbol{x})\, [z_i - o_i^*(\boldsymbol{x})]^2 \,, \tag{2.10}$$

and, since the second term of the sum is a constant (it does not depend on the net), the minimizations of both $E[\boldsymbol{o}]$ and $\varepsilon[\boldsymbol{o}]$ are equivalent. Therefore, $\tilde{\boldsymbol{o}}(\boldsymbol{x})$ is a minimum squared-error approximation to the unconstrained minimum $\boldsymbol{o}^*(\boldsymbol{x})$.

In the rest of this paper we will limit our study to problems for which it is satisfied that $\lambda_i(\boldsymbol{z},\boldsymbol{x}) = 1\,,\ \forall i\,,\ \forall \boldsymbol{z}\,,\ \forall \boldsymbol{x}$. In fact, they cover practically all the applications of back-propagation, since the training patterns are most of the times implicitly regarded as points without error bars. Then, eq. (2.7) gains the simpler form

$$o_j^*(\boldsymbol{x}) = \langle z_j \rangle_{\boldsymbol{x}} = \int_{\mathbb{R}^m} d\boldsymbol{z}\, p(\boldsymbol{z}|\boldsymbol{x})\, z_j \,,\ j = 1, \ldots, m \,, \tag{2.11}$$

whose meaning is that the unconstrained minimum of

$$E[\boldsymbol{o}] = \frac{1}{2N} \sum_{\mu=1}^{N} \sum_{i=1}^{m} \left(o_i(\boldsymbol{x}^\mu) - z_i^\mu\right)^2 \,, \tag{2.12}$$

is, for large N, the *conditional expectation value* or *mean* of the output vectors in the training sample for each input pattern represented by $\boldsymbol{x} \in \mathbb{R}^n$.

As a particular case, if the output representation is chosen to be discrete, say

$$\boldsymbol{z}(\boldsymbol{x}) \in \{\boldsymbol{z}^{(1)}, \boldsymbol{z}^{(2)}, \ldots, \boldsymbol{z}^{(a)}, \ldots\} \,, \tag{2.13}$$

then eq. (2.11) reads

$$o_j^*(\boldsymbol{x}) = \sum_a P(\boldsymbol{z}^{(a)}|\boldsymbol{x})\, z_j^{(a)} \,,\ j = 1, \ldots, m \tag{2.14}$$

where $P(\boldsymbol{z}^{(a)}|\boldsymbol{x})$ is the probability of $\boldsymbol{z}^{(a)}$ conditioned to the knowledge of the value of the input vector $\boldsymbol{x}$.

3 Unary output representations and Bayesian decision rule

It is well known that nets trained to minimize (2.12) are good approximations to Bayesian classifiers, provided a *unary representation* is taken for the output

patterns [6, 12, 14]. That is, suppose the input patterns have to be separated in C different classes $\mathcal{X}_a$, $a = 1, \ldots, C$, and let

$$\boldsymbol{z}^{(a)} \equiv (\overset{1}{0}, \ldots, \overset{a-1}{0}, \overset{a}{1}, \overset{a+1}{0}, \ldots, \overset{m}{0}) \tag{3.1}$$

be the desired output of any input pattern $\boldsymbol{x} \in \mathcal{X}_a$. This assignment specializes each output component to recognize a distinct class ($m = C$). Substituting (3.1) in eq. (2.14) we get

$$o_a^*(\boldsymbol{x}) = \sum_b P(\boldsymbol{z}^{(b)}|\boldsymbol{x}) z_a^{(b)} = P(\boldsymbol{z}^{(a)}|\boldsymbol{x}) \,, \tag{3.2}$$

i.e. the a-th component of the net output turns out to be a minimum squared approximation to the conditional probability that pattern $\boldsymbol{x}$ belong to class $\mathcal{X}_a$. Therefore, if $\tilde{\boldsymbol{o}}(\boldsymbol{x})$ is the output of the net once the learning phase has finished, a good proposal for an almost *Bayesian decision rule* would be:

> $\boldsymbol{x}$ is most likely a member of class $\mathcal{X}_b$, where $\tilde{o}_b(\boldsymbol{x})$ is the largest component of the output vector $\tilde{\boldsymbol{o}}(\boldsymbol{x})$.

The applicability of eq. (3.2) goes beyond classifications. For example, suppose that you have a certain Markov chain $\{\boldsymbol{s}_t,\, t \in \mathbb{N}\}$ of discrete states with constant transition probabilities, and you train a net to learn $\boldsymbol{s}_t$ as a function of $\boldsymbol{s}_{t-1}, \ldots, \boldsymbol{s}_{t-\tau}$. Hence, the output of the net will tend to give these transition probabilities $P(\boldsymbol{s}_t|\boldsymbol{s}_{t-1}, \ldots, \boldsymbol{s}_{t-\tau})$, which by hypothesis do not depend on t.

4 Other discrete output representations

In the previous section we showed how nets can solve, among others, classification problems through the use of unary output representations. The role played by these representations is fundamental, not because they easily give the right solution but because the output contains all the information needed to make a Bayesian decision. In fact, it is easy to find other representations for which some of the information will be unavoidably losen. For instance, consider a *binary representation* for a four classes classification task:

$$\begin{cases} \boldsymbol{z}^{(1)} \equiv (0,0) \\ \boldsymbol{z}^{(2)} \equiv (1,0) \\ \boldsymbol{z}^{(3)} \equiv (0,1) \\ \boldsymbol{z}^{(4)} \equiv (1,1) \end{cases} \tag{4.1}$$

Then, eq. (2.14) leads to

$$\begin{cases} o_1^*(\boldsymbol{x}) \equiv P(\boldsymbol{z}^{(2)}|\boldsymbol{x}) + P(\boldsymbol{z}^{(4)}|\boldsymbol{x}) \\ o_2^*(\boldsymbol{x}) \equiv P(\boldsymbol{z}^{(3)}|\boldsymbol{x}) + P(\boldsymbol{z}^{(4)}|\boldsymbol{x}) \end{cases} \tag{4.2}$$

with the normalization condition

$$\sum_{a=1}^{4} P(\boldsymbol{z}^{(a)}|\boldsymbol{x}) = 1\,. \tag{4.3}$$

Eqs. (4.2) and (4.3) constitute an indeterminated linear system of three equations with four unknown conditional probabilities. The situation will be the same whenever a binary output representation is taken. Thus, they should be avoided if useful solutions are required.

Of course, unary representations are not the only possible choice to find useful solutions. For example, a 'thermometer' representation [5] for the same problem could be

$$\left\{\begin{array}{lcl} \boldsymbol{z}^{(1)} & \equiv & (0,0,0) \\ \boldsymbol{z}^{(2)} & \equiv & (1,0,0) \\ \boldsymbol{z}^{(3)} & \equiv & (1,1,0) \\ \boldsymbol{z}^{(4)} & \equiv & (1,1,1) \end{array}\right. \tag{4.4}$$

which has as solution

$$\left\{\begin{array}{lcl} P(\boldsymbol{z}^{(1)}|\boldsymbol{x}) & = & 1 - o_1^*(\boldsymbol{x}) \\ P(\boldsymbol{z}^{(2)}|\boldsymbol{x}) & = & o_1^*(\boldsymbol{x}) - o_2^*(\boldsymbol{x}) \\ P(\boldsymbol{z}^{(3)}|\boldsymbol{x}) & = & o_2^*(\boldsymbol{x}) - o_3^*(\boldsymbol{x}) \\ P(\boldsymbol{z}^{(4)}|\boldsymbol{x}) & = & o_3^*(\boldsymbol{x}) \end{array}\right. \tag{4.5}$$

The interest towards these representations comes from the need of discretizing continuous output spaces. Simulations showed that binary representations were more difficult to be learnt than thermometer-like ones. However, it is not so clear that those who selected them have interpreted their results in the proper way, putting the outputs in terms of conditional probabilities, and deciding as true output the class of maximal probability.

The final conclusion which could be learnt from what has been said is that, in the discrete and finite case, it is always possible to make an approximated Bayesian decision. To do it, it is needed to obtain all the $P(\boldsymbol{z}^{(a)}|\boldsymbol{x})$ as functions of all the $o_j^*(\boldsymbol{x})$, $j = 1, \ldots, m$, from eq. (2.14). This requires a matrix inversion, which implies that the representation $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(C)}\}$ should be chosen in such a way that the linear system

$$\left\{\begin{array}{l} \displaystyle\sum_{b=1}^{C} P(\boldsymbol{z}^{(b)}|\boldsymbol{x})\, z_a^{(b)} = o_a^*(\boldsymbol{x})\,,\ a = 1, \ldots, d\,,\ d \in \{C-1, C\} \\ \displaystyle\sum_{b=1}^{C} P(\boldsymbol{z}^{(b)}|\boldsymbol{x}) = 1 \text{ needed if } d = C-1 \end{array}\right. \tag{4.6}$$

has a non null determinant (which guarantees the existence of a solution). Usually, the number of output neurons is taken as $C - 1$, since the normalization

condition ($\sum_{b=1}^{C} P(\boldsymbol{z}^{(b)}|\boldsymbol{x}) = 1$) has to be fulfilled. However, it is possible also to use C output neurons if care is taken so as the resulting outputs are properly normalized to be interpreted as probabilities (for instance, due to the approximations introduced by the net, the normalization condition may not be satisfied, thus invalidating their interpretation).

5 An example of learning with different discrete output representations

In order to compare what happens when different discrete output representations are considered we have designed the following example, which from now on we will refer to as the 'four gaussians problem'. Suppose we have one-dimensional real patterns belonging to one of four possible different classes. All the classes are equally probable, $P(\text{class } a) = 1/4,\ a = 1, \ldots, 4$, and their respective distributions $p(\boldsymbol{x}|\text{class } a)$ are normal $N(m, \sigma)$ with averages m and standard deviations σ given in Table 1 (see Fig. 1). Knowing them, the needed conditional probabilities are given by the Bayes theorem:

$$P(\text{class } a|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|\text{class } a)}{\sum_{b=1}^{4} p(\boldsymbol{x}|\text{class } b)}\ ,\ a = 1, \ldots, 4. \tag{5.1}$$

We have trained three neural networks to classify these patterns using as many different output representations: unary, binary and real, as defined in Table 2. All the networks had one input unit, two hidden layers with six and eight units respectively, and four output units in the unary case, two in the binary case and one in the real case. The activations functions were sigmoids, and back-propagation was not batched, i.e. the weights were changed after the presentation of each pattern.

In Fig. 2 we have plotted the expected Bayesian classification (solid line), which according to eq. (3.2) should coincide with the minimum using unary output representation, together with a solution given by the back-propagation algorithm using that representation (dashed line). Both lines are almost the same, but the net assigns the forth class to patterns lower than -5.87 when it should be the third one. This discrepancy is easily understood if one realizes that the probability of having patterns below -5.87 is about 4.92×10^{-8}, which means that the number of patterns generated with such values is absolutely negligible. Thus, the net cannot learn patterns which it has not seen! This insignificant mistake appears several times in this section, but will not be commented any more. The conclusion is, then, that prediction and simulation agree fairly well,

and since the theoretical output is the Bayes classifier, neural nets achieve good approximations to the best solution provided the different classes are encoded using a unary output representation.

To show that neural nets really approach eq. (3.2) we have added Fig. 3, which is a plot of both the predicted conditional probability and the learnt output of the first of the four output units. It must be taken into account that to make approximations to Bayesian decisions you just have to look at the unit which gives the largest output, but you do not need their actual values. However, in the Markov chain example previously proposed, that figures would be necessary.

When using binary and real representations, one has to decide which outputs should go to each class. For instance, the most evident solution for the binary case is to apply a cut at 0.5 to both output units, assigning 0 if the output is below the cut and 1 otherwise. For the real representation a logical procedure would be the division of the interval $[0, 1]$ in four parts of equal length, say $[0, 0.25]$, $[0.25, 0.50]$, $[0.50, 0.75]$ and $[0.75, 1]$, and then assign 1/8, 3/8, 5/8 and 7/8 respectively. These interpretations have been exploited in Fig. 4, where we have plotted the expected results for the four gaussians problem in the three cases of Table 2, according to eq. (2.14). The three lines coincide only in the input regions in which the probability of one class is much bigger than that of the rest. Both binary and real cases fail to distinguish the first class in the interval $[-0.76, 0.29]$. Moreover, the real case incorrectly assigns the third class in the interval $[-2.27, -0.84]$ instead of the forth. The narrow peak at $[2.20, 2.27]$ of the binary representation is just an effect of the transition between the third and the second classes, which are represented as $(1, 0)$ and $(0, 1)$ respectively, making very difficult that both output units cross the cut simultaneously in opposite directions.

Figs. 5 and 6 show predicted and neural networks classifications when binary and real representations are employed. Two examples for the binary neural network outputs are shown, with the expected peaks around 2.23, either as a transition through $(1, 1)$, as in NN1, or through $(0, 0)$, as in NN2.

6 Continuous output representations

Discrete and finite output representations arise quite naturally in the treatment of classification problems. For each class, an arbitrary but different vector is assigned and, taking into account the system (4.6), the best way of doing it so is by imposing linear independence of this set of vectors. Now, with the aid of a sample of patterns, we will be able to determine good approximations to the conditional probabilities that the patterns belong to each class and, knowing them, Bayesian decisions will be immediate.

On the other hand, prediction and interpolation tasks usually amount to finding the 'best' value of several continuous variables for each given input. One

possible but unsatisfactory solution is the discretization of these variables, which has to be made carefully in order to skip various problems. If the number of bins is to big, the number of training patterns should be very large. Otherwise, if the size of the bins is relatively big, the partitioning may fail to distinguish relevant differences, specially if the output is not uniformly distributed. Therefore, it may be stated that a good discretization needs a fair understanding of the unknown output distribution!

Fortunately, neural nets have proven to work well even when the output representation is left to be continuous, without any discretization. For instance, feed-forward networks have been applied to time series prediction of continuous variables, outperforming standard methods [15]. The explanation to this success lies precisely in eq. (2.11), which reveals the tendency of nets to learn, for each input, the *mean* of its corresponding outputs in the training set. Thus, the net is automatically doing what everyone would do in the absence of more information, i.e. substituting the sets of points with common abcise by their average value.

To illustrate that learning with neural networks really tends to give the minimum squared error solution given by eq. (2.11) we have trained them with a set of patterns distributed as the dots in Fig. 7. The solid line is the theoretical limit, while the dashed lines are two different solutions found by neural nets. The first one has been produced using the ordinary sigmoidal activation function, while in the second they have been replaced by sinusoidal ones (scaled between 0 and 1). In most of the input interval the three curves are very similar. However, the sigmoidal one fails to produce the peak located at about -0.5. This is in favour of recent results [13] which show that sinusoidal activations can solve difficult task which sigmoidal cannot, or can but with much more epochs of training.

7 Study of other error criterions

In all the previous sections the study has been concentrated in the minimization of the quadratic error function eq. (2.12). However, other quantities may serve for the same purpose, such as

$$E_q[\boldsymbol{o}] \equiv \frac{1}{qN} \sum_{\mu=1}^{N} \sum_{i=1}^{m} |o_i(\boldsymbol{x}^\mu) - z_i(\boldsymbol{x}^\mu)|^q \ . \tag{7.1}$$

For instance, in [10] the authors modify the error measure during the learning in order to accelerate its convergence, changing in a continuous way from $E_2[\boldsymbol{o}]$ to $E_1[\boldsymbol{o}]$.

Repeating the scheme of Sect. 2, the large N limit of eq. (7.1) is

$$E_q[\boldsymbol{o}] = \frac{1}{q} \sum_{i=1}^{m} \int_{\mathbb{R}^n} d\boldsymbol{x}\, p(\boldsymbol{x}) \int_{\mathbb{R}^m} d\boldsymbol{z}\, p(\boldsymbol{z}|\boldsymbol{x}) \ |o_i(\boldsymbol{x}) - z_i|^q \ , \tag{7.2}$$

and its unconstrained minimum $\boldsymbol{o}^*(\boldsymbol{x};q)$ is found by annulling the first functional derivative. The solutions for the different values of q satisfy the following equations:

$$\begin{aligned}
&\sum_{k=0}^{q-1}(-1)^k\binom{q-1}{k}\,o_j^*(\boldsymbol{x};q)^{q-k-1}\left\langle(z_j)^k\right\rangle_{\boldsymbol{x}}=0 && \text{if } q \text{ even},\\
&\sum_{k=0}^{q-1}(-1)^k\binom{q-1}{k}\,o_j^*(\boldsymbol{x};q)^{q-k-1}\left\langle\mathrm{sign}(o_j^*(\boldsymbol{x};q)-z_j)\,(z_j)^k\right\rangle_{\boldsymbol{x}}=0 && \text{if } q \text{ odd}.
\end{aligned} \tag{7.3}$$

The most interesting case is that of $q = 1$ due to the fact that eq. (7.3) acquires the simplest form

$$\left\langle\mathrm{sign}(o_j^*(\boldsymbol{x};1)-z_j)\right\rangle_{\boldsymbol{x}}=0\,, \tag{7.4}$$

which may be written as

$$\int_{-\infty}^{o_j^*(\boldsymbol{x};1)}dz_j\,p_{(j)}(z_j|\boldsymbol{x})=\int_{o_j^*(\boldsymbol{x};1)}^{\infty}dz_j\,p_{(j)}(z_j|\boldsymbol{x})\,,\ j=1,\ldots,m, \tag{7.5}$$

where $p_{(j)}(z_j,\boldsymbol{x})$ is the j-th marginal distribution of $p(\boldsymbol{z},\boldsymbol{x})$. Therefore, the minimization of $E_1[\boldsymbol{o}]$ has as solution the function that assigns, for each input $\boldsymbol{x}$, the *median* of its corresponding outputs in the training set.

The usefulness of this result lies in that both the median and the mean are, among others (e.g. the *mode*), different estimators of the centre of a distribution. The sample mean is the most obvious and the most oftenly used of these estimators. However, if the distribution is not normal, this is not the best estimator, and in some cases (e.g. the double-exponential distribution) the median becomes the best choice [3].

8 Conclusions

We have proved that the squared error criterion which is used in the back-propagation algorithm has an unconstrained global minimum which, from a statistical point of view, is the *'line' of the means* of the outputs for each input pattern. Thus, neural networks tend to give this solution, since their ability to form very precise and complex maps is well-known. In fact, neural nets can approximate any sufficiently well-behaved function provided the number of units is large enough (see [8, 2, 9, 1] for several theorems on the approximation of functions with neural networks).

From the previous result it is easy to recover the possibility to achieve estimates to the optimal *Bayesian classifiers*, as a particular case in which the representation of the output classes is unary. Other output representations have been

investigated and compared with the unary one, showing that only a certain set of all the possible representations have the capability of making approximations to the Bayesian decisions. Moreover, we show that binary output representations should be avoided in order to obtain correct solutions.

Finally, other error criterions have been studied, the most interesting one being the sum of absolute values instead of their squares. For it, the unconstrained global minimum is the *'line' of the medians*, which is a centrality measure as good as the mean or even better for many output distributions.

References

[1] A.R. Barron, "Universal approximation bounds for superpositions of a sigmoidal function," *IEEE Trans. Information Theory* **39** , 930 (1993).

[2] G. Cybenko, "Approximations by superpositions of a sigmoidal function," *Math. Contr. Signals, Syst.* **2** , 303 (1989).

[3] W.T. Eadie, D. Drijard, F.E. James, M.Roos and B. Sadoulet, "Statistical Methods in experimental Physics", North-Holland (1971).

[4] W. Feller, "An introduction to probability theory and its applications," Wiley, New York (1971).

[5] S. Gallant, "Perceptron-based learning algorithms," *IEEE Trans. Neural Networks* **1** , 179 (1990).

[6] Ll. Garrido and V. Gaitan, "Use of neural nets to measure the τ polarization and its Bayesian interpretation," *Int. J. of Neural Systems* **2** 221 (1991).

[7] B. Gnedenko, "The theory of probability," Mir, Moscow (1978).

[8] R. Hecht-Nielsen, "Neurocomputing," Addison-Wesley, Reading MA (1991).

[9] K. Hornik, M. Stinchcombe and H. White, "Multi-layer feedforward networks are universal approximators," *Neural Networks* **2** , 359 (1989).

[10] N.B. Karayiannis and A.N. Venetsanopoulos, "Artificial neural networks: learning algorithms, performance evaluation an applications," Kluwer Academic Publishers, Boston (1993).

[11] A. Papoulis, "Probability, random variables and stochastic processes," McGraw-Hill, New York (1965).

[12] D.W. Ruck, S.K. Rogers, M. Kabrisky, M.E. Oxley and B.W. Suter, "The multilayer perceptron as an approximation to a Bayes optimal discriminant function," *IEEE Trans. Neural Networks* **1** , 296 (1990).

[13] J.M. Sopena and R. Alquezar, "Improvement of learning in recurrent networks by substituting the sigmoid activation function," Proceedings of ICANN'94, Sorrento (Italy), 417 (1994).

[14] E.A. Wan, "Neural network classification: a Bayesian interpretation," *IEEE Trans. Neural Networks* **1** , 303 (1990).

[15] A.S. Weigend, D.E. Rumelhart and B.A. Huberman, "Backpropagation, weight elimination and time series prediction," In R.Touretzky, J. Elman, Sejnowsky and G. Hinton, *Connectionist Models*, Proceedings of the 1990 Summer School, Morgan Kaufmann.

Table captions

- Table 1: Averages and standard deviations of the normal probability densities for the four gaussians problem.
- Table 2: Three representations for the four gaussians problem.

Class	m	σ
1	0.0	0.997
2	0.8	0.878
3	4.0	2.732
4	-0.8	1.333

Table 1: Averages and standard deviations of the normal probability densities for the four gaussians problem.

Class	Unary	Binary	Real
1	(1,0,0,0)	(0,0)	(1/8)
2	(0,1,0,0)	(0,1)	(3/8)
3	(0,0,1,0)	(1,0)	(5/8)
4	(0,0,0,1)	(1,1)	(7/8)

Table 2: Three representations for the four gaussians problem.

Figure captions

- Figure 1: Probability densities for the four gaussians problem.
- Figure 2: Predicted and neural network classifications for the four gaussians problem using unary output representations.
- Figure 3: First output unit state for the four gaussians problem using the unary output representation.
- Figure 4: Predicted classifications for the four gaussians problem. Only the unary output representation achieves the desired Bayes classification, whereas both binary and real representations give the wrong answer for certain values of the inputs.
- Figure 5: Predicted and neural network classifications for the four gaussians problem using binary output representations.
- Figure 6: Predicted and neural network classifications for the four gaussians problem using real output representations.
- Figure 7: Predicted and neural network outputs. Two different nets have been tested, one with ordinary sigmoidal activation functions and the other with sinusoidal ones.

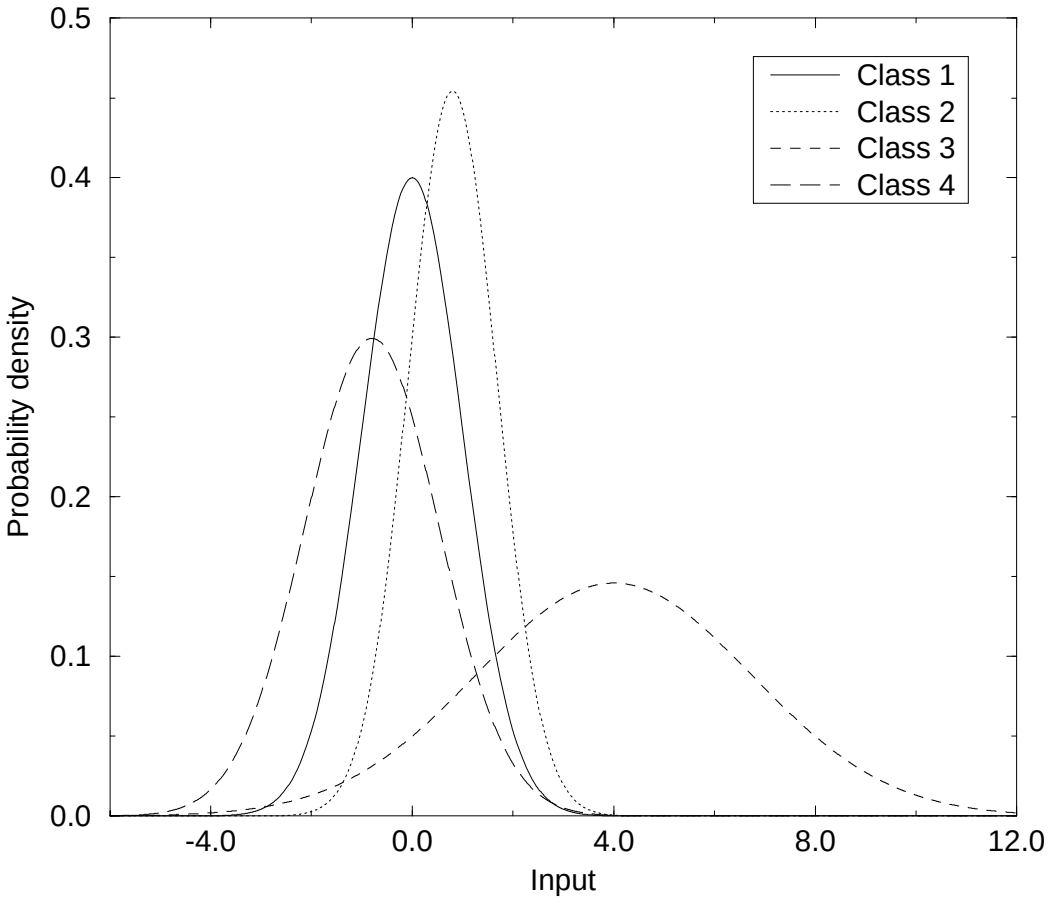

Figure 1: Probability densities for the four gaussians problem.

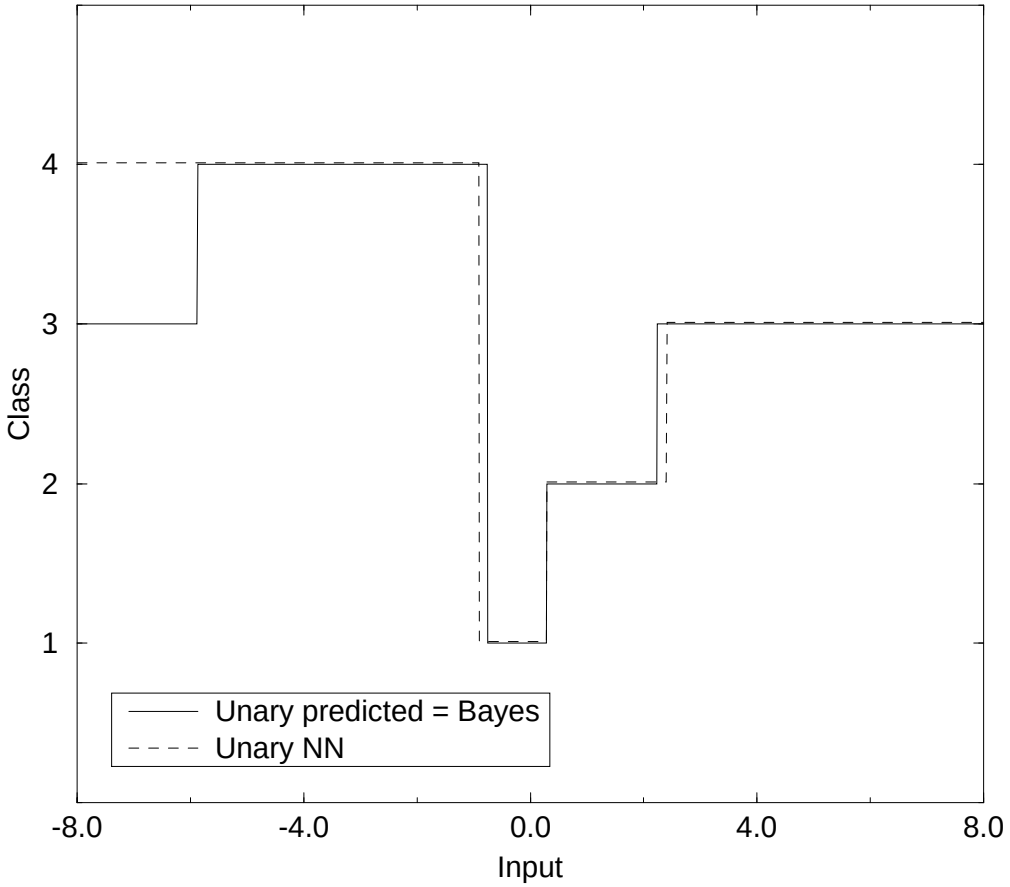

Figure 2: Predicted and neural network classifications for the four gaussians problem using unary output representations.

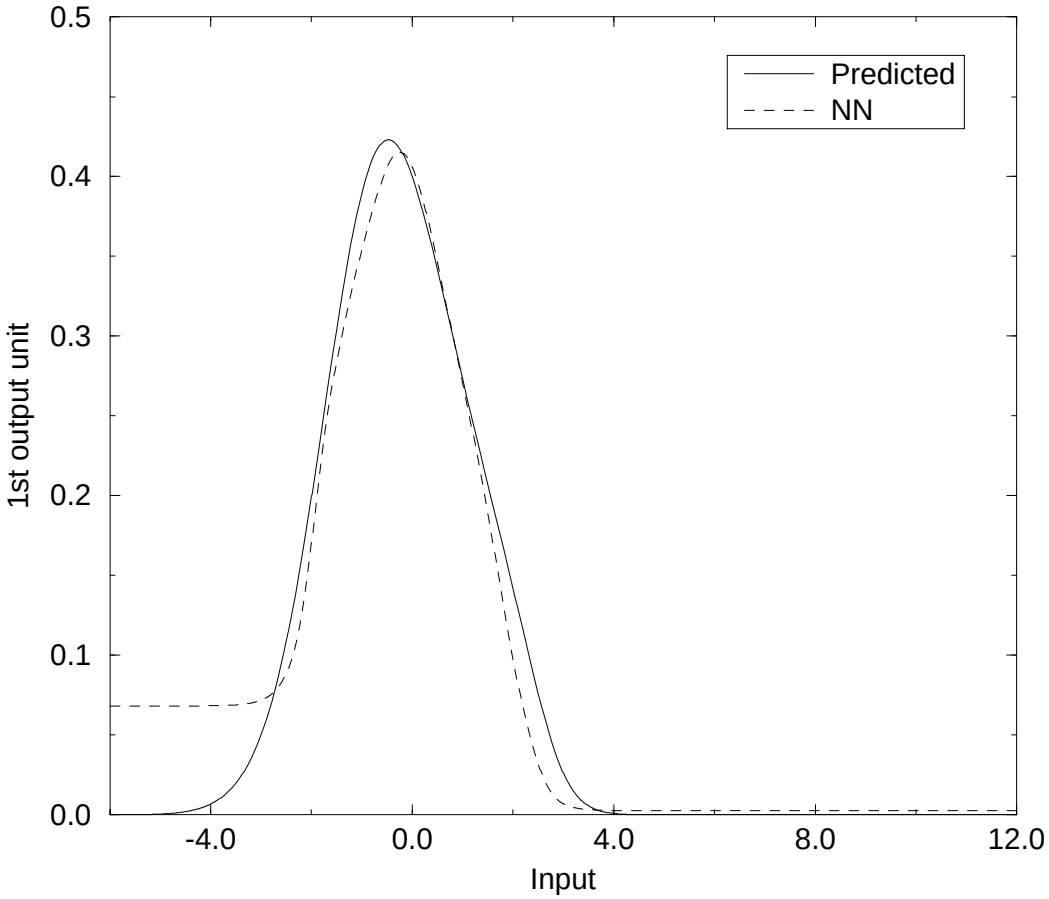

Figure 3: First output unit state for the four gaussians problem using the unary output representation.

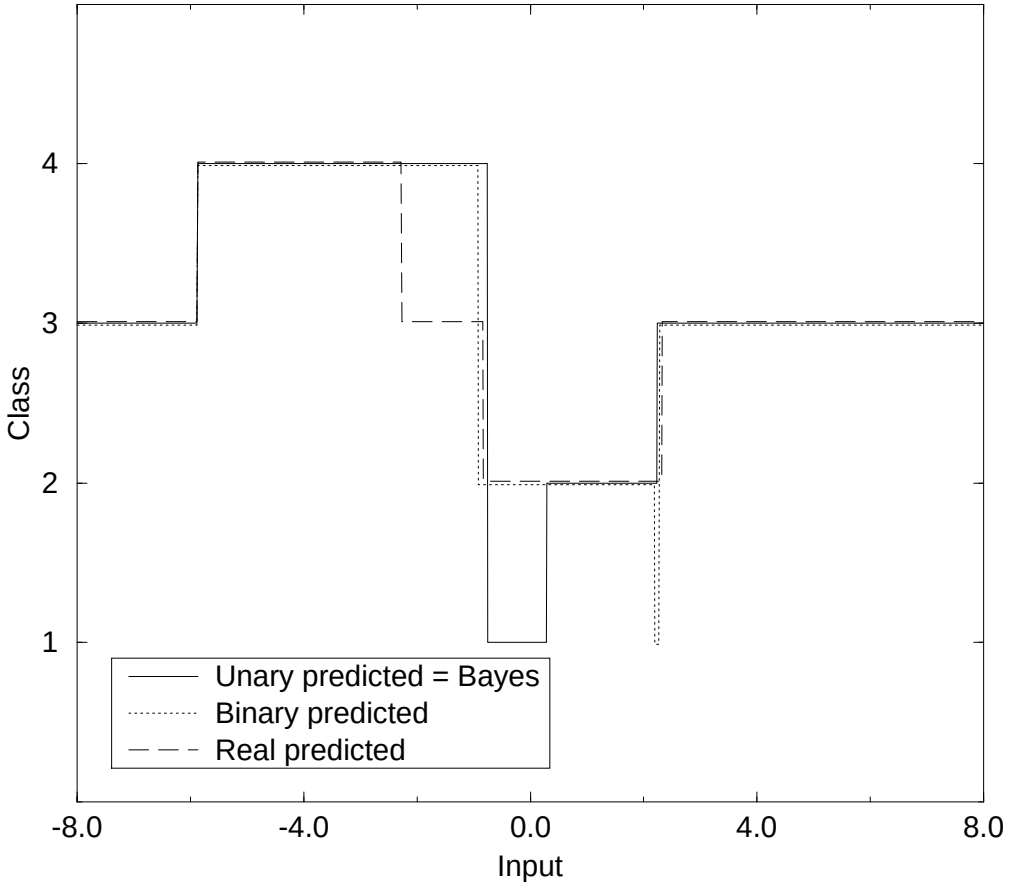

Figure 4: Predicted classifications for the four gaussians problem. Only the unary output representation achieves the desired Bayes classification, whereas both binary and real representations give the wrong answer for certain values of the inputs.

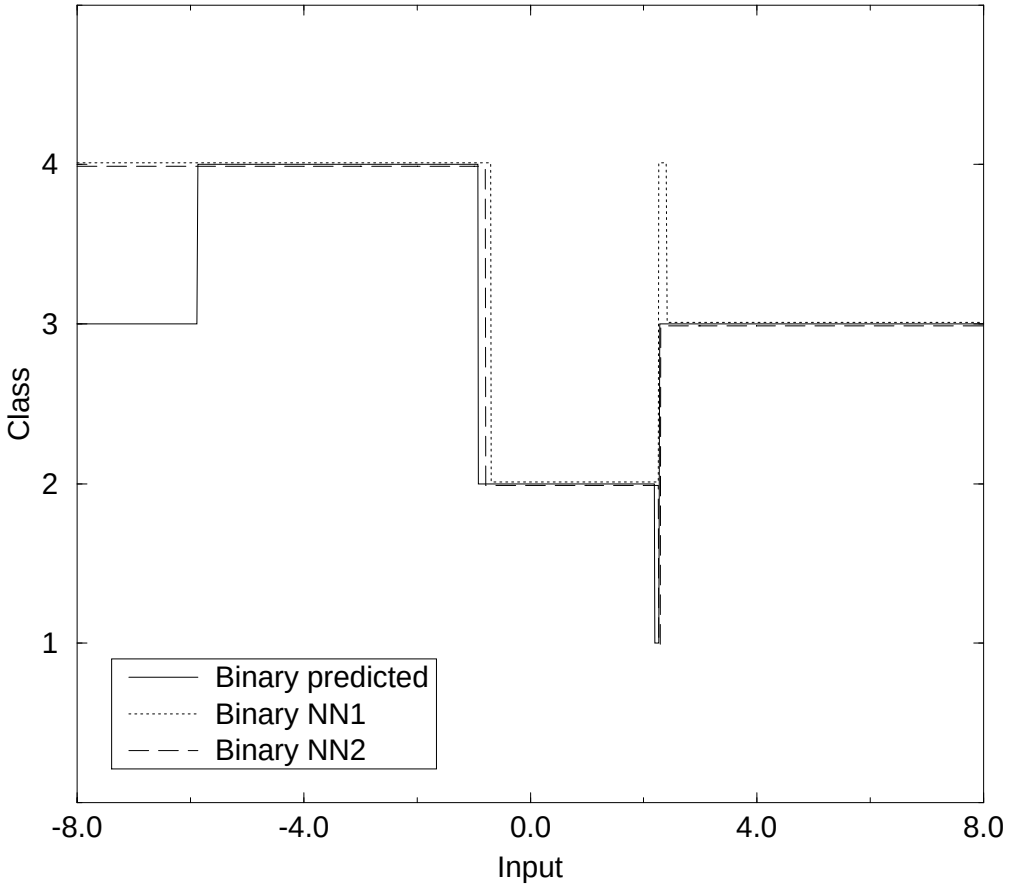

Figure 5: Predicted and neural network classifications for the four gaussians problem using binary output representations.

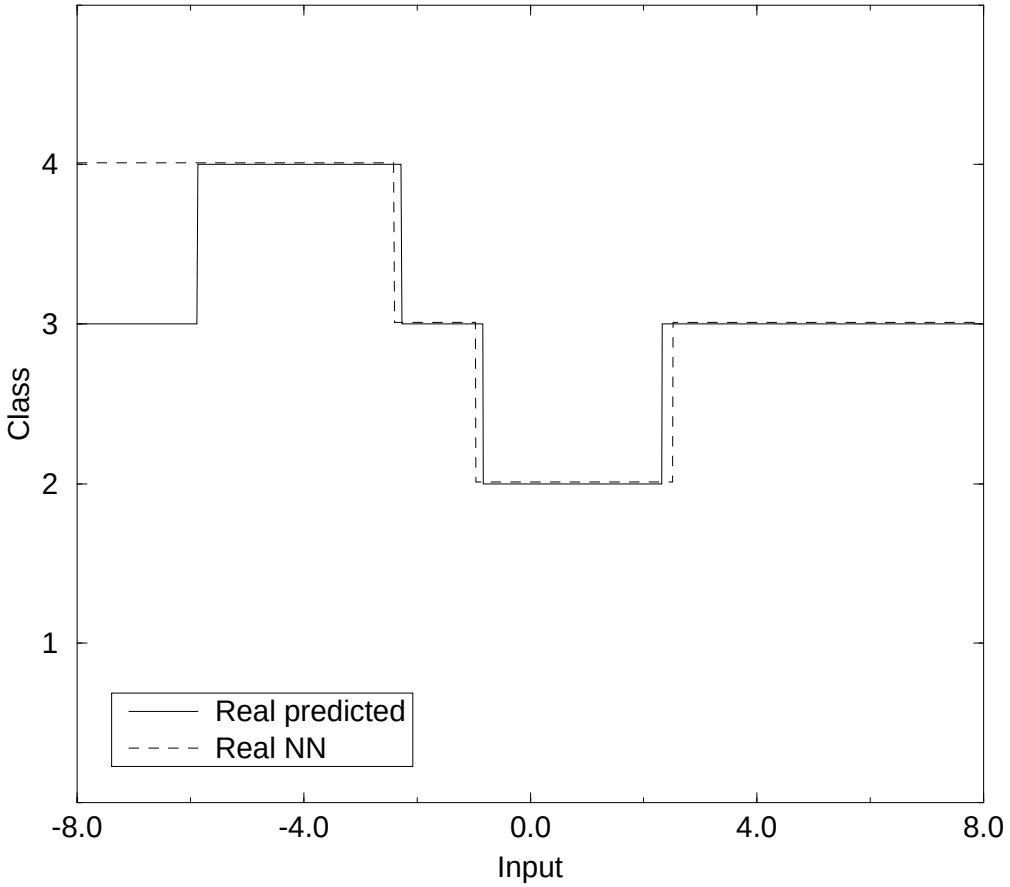

Figure 6: Predicted and neural network classifications for the four gaussians problem using real output representations.

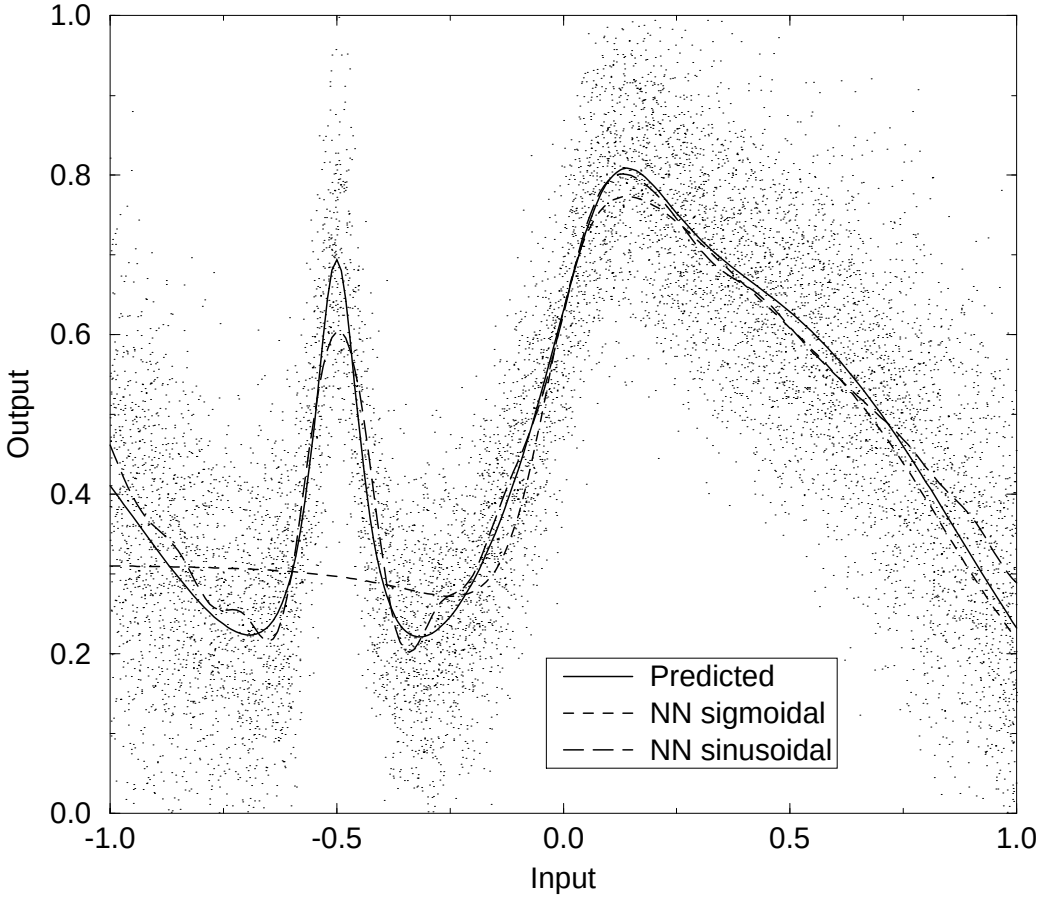

Figure 7: Predicted and neural network outputs. Two different nets have been tested, one with ordinary sigmoidal activation functions and the other with sinusoidal ones.