Neural Comput & Applic (1998)7:334–342

Neural
Computing
& Applications

Learning Criteria for Training Neural Network Classifiers

P. Zhou and J. Austin

Advanced Computer Architecture Group, Department of Computer Science, University of York, York, UK

This paper presents a study of two learning criteria and two approaches to using them for training neural network classifiers, specifically a Multi-Layer Perceptron (MLP) and Radial Basis Function (RBF) networks. The first approach, which is a traditional one, relies on the use of two popular learning criteria, i.e. learning via minimising a Mean Squared Error (MSE) function or a Cross Entropy (CE) function. It is shown that the two criteria have different charcteristics in learning speed and outlier effects, and that this approach does not necessarily result in a minimal classification error. To be suitable for classification tasks, in our second approach an empirical classification criterion is introduced for the testing process while using the MSE or CE function for the training. Experimental results on several benchmarks indicate that the second approach, compared with the first, leads to an improved generalisation performance, and that the use of the CE function, compared with the MSE function, gives a faster training speed and improved or equal generalisation performance.

Keywords: Benchmarks; Learning criteria; Multilayer perceptron networks; Pattern classification; Radial basis function networks; Training methods

1. Introduction

In recent years artificial neural networks, especially Multi-Layer Perceptron (MLP) and Radial Basis Function (RBF) networks, have been widely applied to classification problems. Learning criteria or objective functions used for training and testing neural networks are a key factor determining the training speed and generalisation capability. A great deal of research work has been done in this area [1–12]. Several criteria, well known in conventional statistical pattern classification, have been examined for training neural network classifiers. In [1,2] discriminant learning rules are derived for two-class problems. In [3,4] the objective is to minimise a misclassification rate, parameters of nets are adjusted only when a misclassification occurs and less learning is required. However, these complex learning criteria are not adopted as widely as two well-known learning criteria, i.e. minimising a Mean Squared Error (MSE) function and a Cross Entropy (CE) function. The MSE function is often used because of its conceptual simplicity and computational efficiency. This is a regression approach in which a neural network approximates the average of target training data. In [6–8], a net models probability densities of target data conditioned on input data, and the Cross Entropy (CE) function is derived from the probabilistic model. This function is proposed as an alternative to the MSE function, as it seems more appropriate for training neural network classifiers. But, to our surprise, there is no reported comparative study, and experimental results on benchmark data for supporting this claim.

In this work, we study characteristics of the popular MSE and CE learning criteria in training neural networks, and provide comparative experimental results on several benchmarks. In a conventional approach, one of the two functions is used for training and testing neural networks. A problem with such an approach is that it does not necessarily result in a minimal classification error, which is the real concern in pattern classification. Therefore, we propose to overcome this problem simply by introducing a classification criterion in the test process

Correspondence and offprint requests to: Ping Zhou, Advanced Computer Architecture Group, Department of Computer Science, University of York, Heslington, York YO10 5DD, UK. E-mail: zhoup@cs.york.ac.uk

while still using the MSE or CE function for the training. The second approach is appropriate as it can retain good generalisation in neural network classifiers. In practice the classification performance is usually measured by an empirical classification rate, thus it is used as the classification criterion.

We would like to point out the CE learning criterion is given a maximum mutual information interpretation in [9] and different forms of CE functions are also suggested in [10–12]. Neural networks in [8,9] have one-layer of weights, 'radial units' and a softmax activation function. A hard-limiter activation function is used in [10] and a logistic function in [11,12]. Faster training is reported on problems such as XOR, multiplexer and contiguity [10,12]. Neural models investigated in this work include a standard one-hidden layer MLP and RBF nets, different from those in [8–12], and the CE function as one of learning criteria together with the softmax function in output units, as in [8].

The next section discusses the MSE and CE learning criteria and their characteristics in training neural networks classifiers. Section 3 contains a description of four benchmarks and experimental results. Concluding remarks are given in the last section.

2. Learning Criteria for Building Neural Network Classifiers

In pattern classification, a finite set of training data is usually given in the form of $\{\boldsymbol{x},\boldsymbol{t}\}$ where $\boldsymbol{x}$ is a random input vector with d real number components $[x_1 \cdots x_d]$, and $\boldsymbol{t}$ is a random target vector with c components $[t_1 \cdots t_c]$ for representing the class membership of $\boldsymbol{x}$ in 1-of-c format. A neural model can be represented as $\boldsymbol{y}(\boldsymbol{x},\boldsymbol{t};\boldsymbol{\theta})$, i.e. a function of the training data and a set of parameters $\boldsymbol{\theta}$. The neural mode may be realised by a MLP or RBF network. Optimal values of $\boldsymbol{\theta}$ are determined in a learning or training process by minimising some objective function $O(\boldsymbol{\theta})$ over the training data. The classification of an input pattern with a trained net is simply to assign the pattern to class k for $y_k = \max_i \{y_i\}$. The following parts of this section contain a brief introduction to MLP and RBF networks and two different forms of $O(\boldsymbol{\theta})$, and a study of their characteristics in building neural network classifiers.

2.1. MLP and RBF Neworks

The above neural model $\boldsymbol{y}(\boldsymbol{x},\boldsymbol{t};\boldsymbol{\theta})$ is often realised by a MLP or RBF network for pattern classification [7]. For both networks, their outputs, y_i, can be computed as follows:

$$y_i = \varphi\left(\sum_j w_{ij} z_j\right) = \varphi(a_i) \tag{1}$$

where z_j is the output of the jth hidden unit, w_{ij} the weight from hidden unit j to output unit i and a_i is the activation of the ith output unit,

$$a_i = \sum_j w_{ij} z_j \tag{2}$$

and $\varphi(\cdot)$, the transfer function in the output layer, is commonly a logistic function,

$$\varphi(v) = \frac{1}{1 + \exp(-v)} \tag{3}$$

When $\boldsymbol{y}(\boldsymbol{x},\boldsymbol{t};\boldsymbol{\theta})$ is realised by an one-hidden-layer MLP net with the above logistic transfer function in its hidden layer, the output of the jth hidden unit z_j in Eq. (1) is

$$z_j = \varphi\left(\sum_l w_{jl}^0 x_l\right) \tag{4}$$

where w_{jl}^0 is the weight from input l to hidden unit j. Therefore, the MLP network function can be written as

$$y_i = \varphi\left[\sum_j w_{ij} \varphi\left(\sum_l w_{jl}^0 x_l\right)\right] \tag{5}$$

In contrast, when $\boldsymbol{y}(\boldsymbol{x},\boldsymbol{t};\boldsymbol{\theta})$ is realised by an RBF network, z_j represents the output of the jth RBF unit as follows:

$$z_j = \exp\left(-\tfrac{1}{2} \sum_{l=1}^{d} (x - \mu_{jl})^2 / \sigma_{jl}^2\right) \tag{6}$$

In the above the vector $\boldsymbol{\mu}_j = [\mu_{j1} \cdots \mu_{jd}]$ is the centre of the jth RBF unit and $\boldsymbol{\sigma}_j = [\sigma_{j1} \cdots \sigma_{jd}]$ is its width. And the RBF network function is

$$y_i = \varphi\left[\sum_j w_{ij} \exp\left(-\tfrac{1}{2} \sum_{l=1}^{d} (x - \mu_{jl})^2 / \sigma_{jl}^2\right)\right] \tag{7}$$

In general, network parameters, that is the above weights, centres and widths, are donoted by $\boldsymbol{\theta}$. Optimal values of $\boldsymbol{\theta}$ are determined in a learning or training process by minimising some objective function $O(\boldsymbol{\theta})$ over the training data. A gradient descent method with a momentum term and a regularisation term can be used for the optimisation of the parameters, and the corresponding learning rule for adjusting parameters at the nth iteration takes the following form:

$$\Delta\boldsymbol{\theta}(n) = -\eta \frac{\partial O(n)}{\partial \boldsymbol{\theta}(n)} + \alpha\Delta\boldsymbol{\theta}(n-1) - \gamma\boldsymbol{\theta}(n-1) \quad (8)$$

In the above first two terms, η and α are two small positive constants determining learning rate; and in the third regularisation term, γ is a small positive constant determining the amount of weight decay. The following two sub-sections introduce two objective functions, and partial derivatives derived from the two objective functions.

2.2. Learning via Minimising a MSE Function

The MSE function can be used as the objective function $O(\boldsymbol{\theta})$ in Eq. (8). The idea behind this is to model the discriminant function implied in the training data [6]. The optimisation of network parameters is the minimisation of the sum of square difference between the target data and outputs of a network, i.e.

$$\frac{1}{2N}\sum_{n=1}^{N}\sum_{i=1}^{c}[t_i(n) - y_i(n)]^2 = \frac{1}{2N}\sum_{n=1}^{N}E(n) \quad (9)$$

where N is the number of training patterns, $t_i(n)$ is the ith component of the nth target data, and $y_i(n)$ is the ith output of the net for the nth input pattern. When N is large, a net approximates the average of the target training data [6]. Also, when the target data can be modelled by a determinant function with additive Gaussian noise, the MSE function can be derived from the maximisiation of the likelihood of a probabilistic model [7].

Partial derivatives of E with respect to the activation of output units can be evaluated as

$$\frac{\partial E}{\partial a_i} = \frac{\partial E}{\partial y_i}\frac{\partial y_i}{\partial a_i} = (y_i - t_i)y_i(1 - y_i) \quad (10)$$

This result can be used to evaluate partial derivatives of E with respect to weights from the hidden layer to the output layer for both MLP and RBF networks as follows:

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial a_i}\frac{\partial a_i}{\partial w_{ij}} = \frac{\partial E}{\partial a_i}z_j \quad (11)$$

For the one-hidden-layer MLP network described above, partial derivatives of E with respect to w_{jl}^0 can be evaluated as

$$\frac{\partial E}{\partial w_{jl}^0} = \sum_i \frac{\partial E}{\partial z_i}\frac{\partial z_i}{\partial w_{jl}} = \sum_i \frac{\partial E}{\partial a_i} w_{ij}z_j(1 - z_j)x_l \quad (12)$$

For the RBF network, partial derivatives of E with respect to centres and widths of RBF units are

$$\frac{\partial E}{\partial \mu_{jl}} = \sum_i \frac{\partial E}{\partial z_i}\frac{\partial z_i}{\partial \mu_{jl}} = \sum_i \frac{\partial E}{\partial a_i} w_{ij}z_j \frac{(x_l - \mu_{jl})}{\sigma_{jl}^2} \quad (13)$$

$$\frac{\partial E}{\partial \sigma_{jl}} = \sum_i \frac{\partial E}{\partial z_i}\frac{\partial z_i}{\partial \sigma_{il}} = \sum_i \frac{\partial E}{\partial a_i} w_{ij}z_j \frac{(x_l - \mu_{jl})^2}{\sigma_{jl}^3} \quad (14)$$

2.3. Learning via Minimising a CE Function

As an alternative to the above MSE function, the objective function $O(\boldsymbol{\theta})$ in Eq. (8) can be a cross entropy function based on a probabilistic model [7,8]. In this approach it is of interest to model probabilities of class memberships conditioned on input data, that is $p(i|\boldsymbol{x})$, where $i = 1,\cdots,c$, and to determine unknown model parameters via the maximal likelihood estimation. Specifically, let the kth output, y_k, of a net represent the probability of observing target values of a kth class pattern, that is $p(i = k|\boldsymbol{x} \in \omega_k) = y_k$. Assuming training patterns are independent of each other, the likelihood of observing N training data is given by

$$\prod_{n=1}^{N} p(i = k|\boldsymbol{x}(n) \in \omega_k) = \prod_{n=1}^{N} y_k(n) \quad (15)$$

To use the above for supervised training, we can rewrite it to make the target variable $\boldsymbol{t} = [t_1\cdots t_c]$ occur explicitly in the likelihood function

$$\prod_{n=1}^{N}\prod_{i=1}^{c}[y_i(n)]^{t_i(n)} \quad (16)$$

where $t_i = \delta_{ik} = 1$ for $i = k$ and 0 for $i \neq k$. It is more conveniently to minimise the negative logarithm of Eq. (16)

$$\sum_{n=1}^{N}\sum_{i=1}^{c} t_i(n)\ln y_i(n) = \sum_{n=1}^{N} \mathrm{L(n)} \quad (17)$$

The above is a Cross Entropy (CE) function. Since the outputs of the network represent probabilities, that is $y_i \in [0,1]$ and $\Sigma y_i = 1$, the softmax activation function is appropriate for use in output units. This function takes the following form:

$$\varphi(a_i) = \frac{\exp(a_i)}{\sum_j \exp(a_j)} \quad (18)$$

where a_i is the activation of the ith output unit as in Eq. (2). Partial derivatives of L with respect to a_i are given by

$$\frac{\partial L}{\partial a_i} = \sum_j \frac{\partial L}{\partial y_j}\frac{\partial y_j}{\partial a_i} = \sum_j \left(-\frac{t_j}{y_j}\right)(y_j\delta_{ij} - y_iy_j) = y_i - t_i \quad (19)$$

This result is simpler than Eq. (10) for the MSE objective. Substituting the above $\partial L/\partial a_i$ with $\partial E/\partial a_i$ in Eqs (11), (12), (13) and (14), partial derivatives of L with respect to parameters of a MLP net and an RBF net can be obtained.

2.4. Characteristics of MSE and CE Criteria

Having introduced the MSE and CE learning criteria, this section studies their characteristics in training neural networks. The training process can be viewed as a dynamic system as shown in Fig. 1(a), that moves from an initial state towards a final state. The system consists of a net function $f(\cdot)$, which is to be trained, and a learning function or algorithm $g(\cdot)$ that modifies weights of the net. The system's state is described by the network output $\boldsymbol{y}$. Initially, $d\boldsymbol{w} = 0$, the weights are set to some random values and the system has a random state. For each presentation of the input and target data, $\{\boldsymbol{x},\boldsymbol{t}\}$, the net gives the output $\boldsymbol{y}$, and then the learning algorithm updates the weights by the amount $d\boldsymbol{w}$. In this study, the learning algorithm relies on the use of either the MSE or CE objective function.

Consider a simple MLP net in Fig. 1(b), which has only one layer of weights, i.e. w_i and w_b (bias weight). When the MSE objective is used, the learning algorithm updates the weights by the amount proportional to the following:

$$dw_b = -E' = -(y - t)y(1 - y) \tag{20a}$$

$$dw = -(y - t)y(1 - y)x \tag{20b}$$

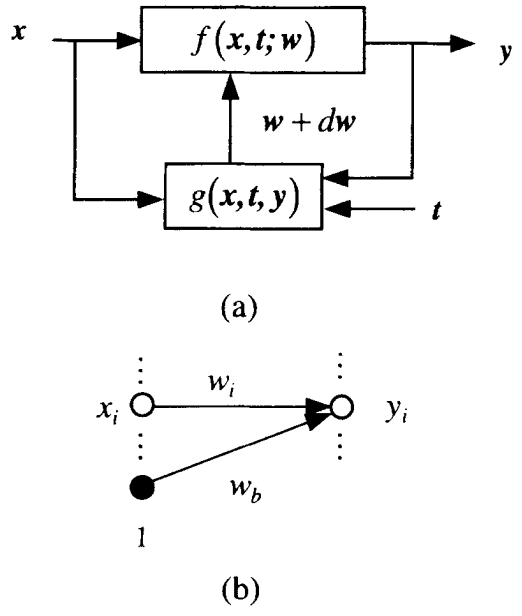

Fig. 1. (a) Viewing the learning process as a dynamic system, (b) a simple MLP net.

The above are derived from Eqs (10) and (11), respectively, and the subscript i is omitted for conciseness. From Eq. (20a), which describes the convergence of the bias, we plot two curves (Fig. 2(a)) to show the convergence, that is $-E' = y(1-y)^2$ for $t = 1$ and $-E' = -y^2(1-y)$ for $t = 0$. For Eq. 20(b) that describes the convergence of the weight, we plot a surface as shown in Fig. 2(b), i.e. $dw = y(1-y)^2x$ for $t = 1$, where the input x is normalised in $[-1,1]$. These curves and suface reveal how the dynamic system behaviours in a training process. When initial weights are set to small random values, the initial state of the system is around $y = 0.5$ (shown as large circles in Fig. 2(a)). In the training process the net moves along the curve $-E'$ for $t = 1$ (or $-E'$ for $t = 0$) towards the final state $y = 1.0$ for $t = 1$ (or $y = 0.0$ for $t = 0$), i.e. crosses for Fig. 2(a). As the net approaches the final state, the converge speed reduces dramatically. When $y > 0.9$ for $t = 1$ (or $y < 0.1$ for $t = 1$) the curve is almost flat and the net learns very slowly. When the initial weights are large, the initial state can be far from the centres, e.g. small circles in Fig. 2(a). In such a situation, the learning is relatively slow or the system may be trapped.

Similarly, for the CE function we have the following:

$$dw_b = -L' = -(y - t) \tag{21a}$$

$$dw = -(y - t)x \tag{21b}$$

and Figs 3(a) and 3(b). Characteristics of training with the CE function are quite different from that of the MSE function. Because of the use of the softmax function in the output units, the initial state of the system can be anywhere on the two straight lines in Fig. 3(a) and the surface in Fig. 3(b). When the system approaches the final state, the learning slows down, but the curve is not flat. When the state is far from the final state, the system moves very quickly. However, this also means that the effects of outliers are much greater. This may affect the performance of a MLP network but an RBF network in which RBF units response locally.

2.5. Two Training Approaches

In a traditional approach, a neural network is trained with either the MSE or CE learning criterion for an epoch (the presentation of the whole set of training data); then the net is tested on the test data; the net is retained if it performs better than in previous epochs. In pattern classification the real concern is the probability of misclassification rather than the mean squared error (in Eq. (9)) or the likelihood

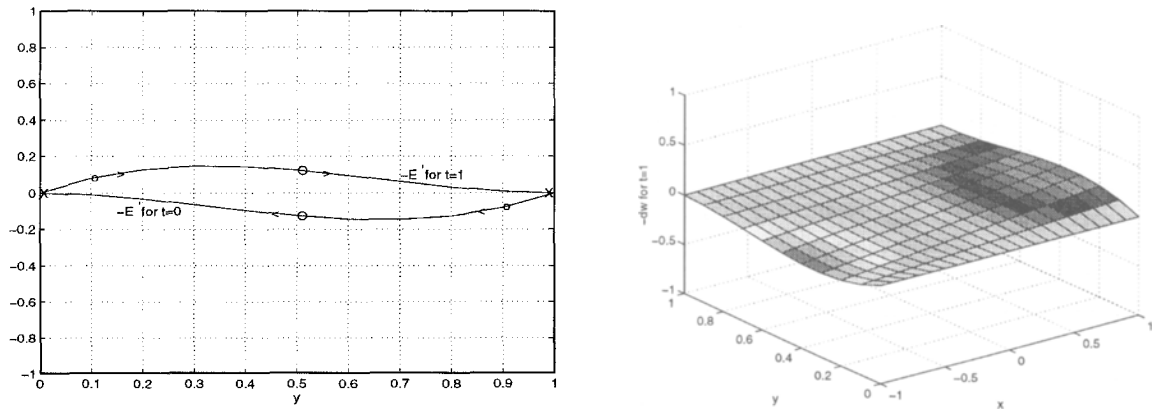

Fig. 2. (a) $-E'$ versus y for $t = 1$ and 0, (b) dw versus x and y for $t = 1$.

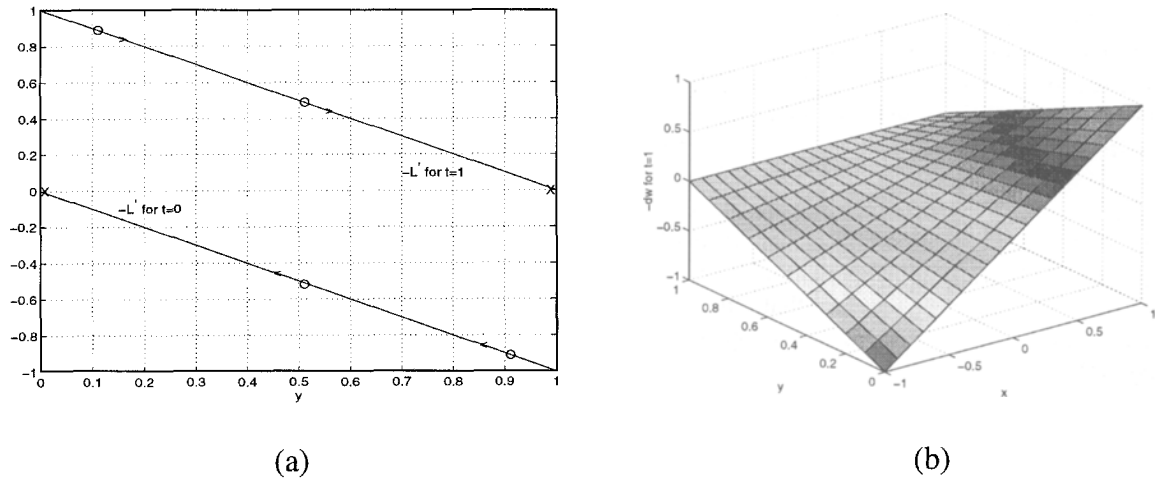

Fig. 3. (a) $-L'$ versus y for $t = 1$ and 0, (b) dw versus x and y for $t = 1$.

(Eq. (17)). A problem with this approach is that a trained net, which gives a minimal mean squared error or a likelihood, does not necessarily produce a minimal classification error. This can be illustrated by a simple hypothetical example in Fig. 4, in which the density $p(c_1|x)$ is modelled by two trained nets A and B, respectively. The net A gives a smaller MSE or models the overall density better, but it gives a larger classification error; the net B produces a smaller classification error as it models the classification boundary more precisely. Therefore, the net B is preferred for classification tasts.

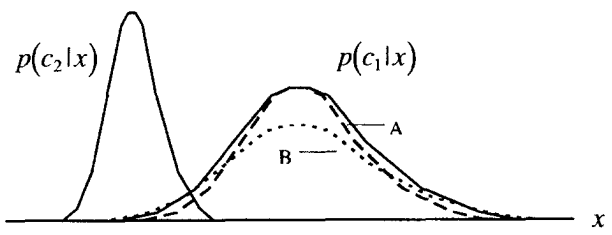

Fig. 4. An example of modelling densities of the target data.

The above problem can be overcome simply by introducing a classification criterion for the test process while using MSE or CE criterion for the training only. Such an approach can retain nets with minimal classification errors on the test data (what we wish to achieve). In practice, the classification error is usually measured by a Classification Rate (CR), i.e. the ratio of the number of correctly classified patterns over the total number of patterns, thus it is appropriate to use the CR classification criterion.

Table 1. Four benchmarks used in the experiments.

Benchmark	Inputs	Outputs	Training patterns	Test patterns
British vowel	11	10	528	462
Sonar signal	60	2	104	104
Radar return	34	2	200	151
Satellite image	36	6	4000	2000

3. Experimental Results

Four benchmark domains used in the experiments were obtained from the CMU Neural Network Benchmark Databases [13] and UCI Repository of Machine Learning Databases [14]. Table 1 contains input and output dimensions and numbers of training and test patterns in each benchmark. The first benchmark [15] is related to speaker independent recognition of the eleven steady state of vowels of British English. The second data set [16] is for the classification of sonar signals bounced off a metal cylinder and a roughly cylindrical rock. The third set of benchmark data is for the classification of radar returns from the ionosphere [17]. The fourth benchmark is a small part of a multi-spectral satellite image [18].

In experiments described below, we studied the two approaches to training MLP nets on four benchmarks; as the results indicated that the second approach was more promising, we adopted it for training RBF nets.

MLP nets were trained using the two approaches described in the last section. The first approach used only the MSE or CE learning criterion; the second used either the MSE function together with the Classification Rate (CR), denoted as the MSE-CR criterion, or the CE function with the CR, termed as the CE-CR criterion. During a training process the generalisation performance of a net was evaluated on a test set. The performance was recorded if it was better than that obtained in previous evaluation. MLP nets had one layer of hidden nodes and employed the logistic transfer function in Eq. (3) when using the MSE objective, or the softmax transfer function in Eq. (18) while using the CE objective. The number of hidden nodes was optimised manually for individual data sets. Learning parameters, such as the learning rate η, momentum α and initial weights, were also determined experimentally to obtain the best performance for individual sets of data. For MLP networks the results were sensitive to initial weights, which were set to random numbers in the range [–0.05, +0.05] for vowel and satellite image data, and the range [–0.5, +0.5] for sonar and ionosphere data. The learning rate η and momentum α were relatively smaller (0.05–0.1) for the satellite image data than for the other three data sets (0.1–0.3). The appropriate value of the weight decay factor γ in Eq. (8) was determined as 10^{-5} experimentally.

Experimental results on the vowel benchmark are given in Table 2, on the sonar data in Table 3, the ionosphere data in Table 4 and the satellite image data in Table 5. The generalisation performance was measured using the classification rate, i.e. the ratio of the number of patterns correctly classified over all patterns in a test set. These tables include maximal and minimal classification rates, mean and standard deviation of classification rates. Results were accumulated over 10 runs. The experimental results indicated that on all four data sets the second approach using either the MSE-CR or CR-CR criterion gave a better generalisation performance, that is higher maximal, minima and mean classification rates, and lower standard deviation in classification rates. On the vowel data the improvement in the mean classification rate was over 2%; on the rest three data sets, the improvement was about 0.5–1%.

It is also interesting to note the CE-CR criterion performed better than the MSE-CR criterion. To further study them, we adopted the more promising

Table 2. Classification rates by MLP nets on British vowel test data.

Method	Hidden-nodes	Mean	S.D.	Min.	Max.
MLP with MSE	88	0.5731	0.0219	0.5303	0.6017
MLP with MSE-CR	88	0.5961	0.0080	0.5801	0.6104
MLP with CE	88	0.5725	0.0171	0.5454	0.5952
MLP with CE-CR	88	0.6111	0.0149	0.5909	0.6407

Table 3. Classification rates by MLP nets on sonar test data.

Method	Hidden-nodes	Mean	S.D.	Min.	Max.
MLP with MSE	12	0.8782	0.0161	0.8462	0.8942
MLP with MSE-CR	12	0.9070	0.0075	0.8942	0.9231
MLP with CE	12	0.9116	0.0115	0.8942	0.9327
MLP with CE-CR	12	0.9287	0.0104	0.9135	0.9423

Table 4. Classification rates by MLP nets on ionosphere test data.

Method	Hidden-nodes	Mean	S.D.	Min.	Max.
MLP with MSE	10	0.9647	0.0082	0.9536	0.9801
MLP with MSE-CR	10	0.9702	0.0053	0.9603	0.9801
MLP with CE	10	0.9735	0.0084	0.9536	0.9868
MLP with CE-CR	10	0.9768	0.0072	0.9669	0.9868

Table 5. Classification rates by MLP nets on satellite image test data.

Method	Hidden-nodes	Mean	S.D.	Min.	Max.
MLP with MSE	48	0.9092	0.0027	0.9035	0.9135
MLP with MSE-CR	48	0.9112	0.0027	0.9045	0.9145
MLP with CE	48	0.9058	0.0022	0.9025	0.9080
MLP with CE-CR	48	0.9098	0.0026	0.9060	0.9140

second approach to training RBF nets, which are termed as supervised RBF (S-RBF) nets. They had the logistic function in their output units when using the MSE objective, and the softmax function for the CE objective. The number of hidden nodes in an RBF net was optimised manually. A K-mean clustering method [19] was used to determine initial centre positions of RBF units. Initial widths of an RBF unit were set to the average of Euclidean distances to its two nearest neighbours. Weights from RBF units to output units were initialised to random numbers in the range [−0.5, +0.5]. After such an initialisation, RBF nets were trained using supervised training method in Section 2. The learning rate η and momentum α for updating weights were 0.3, for centres 0.06 and for widths 0.03. Experimental results on the vowel benchmark are given in Table 6, on the sonar data in Table 7, the ionosphere data in Table 8 and the satellite image data in Table 9.

Now let us compare results from using MSE-CR and CE-CR for training MLP and RBF nets. On the vowel data the CE-CR criterion led to higher classification rates. For instance, increases in the mean classification rates were 1.5% with MLP nets (Table 2) and 2.9% with RBF nets (Table 9). Improvements in the maximal classification rates using the CE-CR were a 3% with MLP nets and a 4.3% with RBF nets. Therefore, the CE-CR criterion performed significantly better than the first approach on the vowel data. On the sonar and ionosphere data and CE-CR criterion gave 0.3–2.2% increases in mean classification rates and 0–1.9% increases in maximal classification rates. The CE-CR criterion performed better on the two data sets. On the satellite image data for MLP nets (Table 5), the CE-CR criterion gave less than a 0.14% lower mean classification rate, and for RBF nets (Table 9) it gave a 0.84% higher mean classification rate. Therefore, the performance difference of the MSE-CR and CR-CR criteria on this data set was very small.

It was observed that much less time was needed for training neural networks using the CE-CR objective than using the MSE-CR objective. The time ratio with MLP nets was 0.22 for vowel data, 0.65 for the sonar data, 0.77 for the ionosphere data, and 0.57 for the satellite image data.

In summary, on the four benchmarks the MSE-CR criterion performed better than the MSE criterion, and the CE-CR also performed better than the CE criterion. This has confirmed our expectation (in Section 2.5) that the second approach can retain

Table 6. Classification rates by RBF nets on British vowel test data.

Method	Hidden-nodes	Mean	S.D.	Min.	Max.
S-RBF with MSE-CR	33	0.6835	0.0132	0.6667	0.7013
S-RBF with CE-CR	33	0.7128	0.0191	0.6818	0.7446

Table 7. Classification rates by RBF nets on sonar test data.

Method	Hidden-nodes	Mean	S.D.	Min.	Max.
S-RBF with MSE-CR	12	0.9319	0.0086	0.9135	0.9423
S-RBF with CE-CR	12	0.9351	0.0083	0.9231	0.9423

Table 8. Classification rates by RBF nets on ionosphere test data.

Method	Hidden-nodes	Mean	S.D.	Min.	Max.
S-RBF with MSE-CR	6	0.9834	0.0068	0.9735	0.9934
S-RBF with CE-CR	6	0.9868	0.0059	0.9735	0.9934

Table 9. Classification rates by RBF nets on satellite image test data.

Method	Hidden-nodes	Mean	S.D.	Min.	Max.
S-RBF with MSE-CR	30	0.9030	0.0021	0.9005	0.9065
S-RBF with CE-CR	30	0.9114	0.0025	0.9075	0.9145

good generalisation in neural network classifiers. Compared with the MSE-CR criterion, the CE-CR criterion resulted in better classification rates and faster training speed for the first three benchmarks. On the satellite image data, which are a relative large set, the two criteria gave the similar classification accuracy and the CE-CR criterion used 43% less training time. These experimental observations are consistent with our analysis of characteristics of the MSE and CE criteria in Section 2.4.

4. Conclusions

In this work we have studied two learning criteria and two approaches to using them for building neural network classifiers. Compared with the CE criterion, characteristics of the MSE criterion include smaller outlier effects and slower learning speed, especially when the outputs of a net approach the target data. The traditional training approach gives a minimal MSE or likelihood. However, this does not guarantee a minimal classification error. We extend this approach simply by introducing an empirical classification criterion (classification rate) for the test process while using the MSE or CE function for the training. Comparative experimental results on the four benchmark data show that the second approach has resulted in an improved generalisation performance, and that the use of the CE criterion has led to a faster training speed and improved or equal generalisation performance.

References

1. Juang B, Katagiri S. Discriminative learning for minimum error classification. IEEE Trans Signal Processing 1992; 40(12): 3043–3053.
2. Nedljković V. A novel multilayer neural networks training algorithm that minimises the probability of classification errors. IEEE Trans Neural Networks 1993; 4(4): 650–659
3. Miller D, Rao AV, Rose K, Gersho A. A global optimisation technique for statistical classier design. IEEE Trans Signal Processing 1996; 44(12): 3108–3122.
4. Telfer BA, Szu HH. Energy function for minimising

misclassification error with minimum-complexity networks. Neural Networks 1994; 7 (5): 809–818.
5. Hampshire JB, Waibel AH. A novel objective function for improved phoneme recognition using time-delay neural networks. IEEE Trans Neural Networks 1990; 1(2): 2160–2228
6. Hey H. On the probabilistic interpretation of neural network classifiers and discriminative training criteria. IEEE Trans Pattern Analysis and Machine Intelligence 1995; 17(2): 107–119
7. Bishop CM. Neural networks for pattern recognition. Clarendon Press, Oxford, 1995
8. Bridle JS. Probabilistic interpretation of feedforward classification networks outputs, with relationships to statistical pattern recognition. In: Soulie F, J. Herault J (eds) Neuro-computing: Algorithms, architectures and applications, Springer-Verlag 1990, pp. 227–236
9. Bridle JS. Training stochastic model recognition algorithm. In: Touretzky D (ed), Advances in Neural Information Processing Systems, Vol 2, MIT Press, 1990, pp. 211–217
10. Bichsel M, Seitz P. Minimum class entropy: a maximum information approach to layered neural networks. Neural Networks 1989; 2(2): 133–141
11. Solla SA, Levin E, Fleisher M. Accelerate learning in layered neural networks. Complex Systems 1988; 22: 625–640
12. Holt MJJ, Semnani S. Convergence of back-propagation in neural networks using a log-likelihood function. Electronic Letters 1990; 26(23): 1965–1965
13. The Carnegie Mellon University Collection of Neural Net Benchmarks, from http://www.cs.cmu.edu:80/afs/cs/project/connect/bench/, or from ftp.cs.cmu.edu – afs/cs/project/connect/bench/
14. University of California, Irvine, UCI Repository of Machine Learning Databases, from http://www.ics.uci.edu/~mlearn/MLOther.html, or from ftp.ics.uci.edu – pub/machine-learning-databases
15. Deterding D. Speaker normalisation for automatic speech recognition. PhD Thesis, University of Cambridge, 1989
16. Gorman RP, Sejnowski TJ. Analysis of hidden units in a layered network trained to classify sonar targets. Neural Networks 1988; 1(1): 75–89
17. Sigillito VG, Wing SP, Hutton LV, Baker KB. Classification of radar returns from the ionosphere using neural networks. Johns Hopkins APL Technical Digest 1989; 10: 262–266
18. Michie D, Spiegelhalter DJ, Taylor CC. Machine learning, neural and statistical classification, Ellis Horwood, 1994
19. Moody J, Darken CJ. Fast learning in networks of locally tuned processing units. Neural Computation 1989; 1(2): 281–294

Nomenclature

$\boldsymbol{x}$	random input vector with d real number components $[x_1 \cdots x_d]$
$\boldsymbol{t}$	random target vector with c binary components $[t_1 \cdots t_c]$
$\boldsymbol{y}(\cdot)$	neural network function or output vector
$\boldsymbol{\theta}$	parameters of a neural model
η	learning rate
α	momentum
γ	decay factor
O	objective function
E	mean sum-of-squares error function
L	cross entropy function
n	nth training pattern
N	number of training patterns
$\varphi(\cdot)$	transfer function in a neural unit
z_j	output of hidden unit-j
a_i	activation of unit-j
w_{ij}	weight from hidden unit-j to output unit-i
w_{jl}^0	weight from input unit-l to hidden unit-j
$\boldsymbol{\mu}_j$	centre vector $[\mu_{j1} \cdots \mu_{jd}]$ of RBF unit-j
$\boldsymbol{\sigma}_j$	width vector $[\sigma_{j1} \cdots \sigma_{jd}]$ of RBF unit-j
$p(\cdot\|\cdot)$	conditional probability function