New developments of the Z-EDM algorithm

Luís M. Silva, J. Marques de Sá
INEB -Instituto de Engenharia Biomédica
Porto - Portugal
lmsilva@fe.up.pt, jmsa@fe.up.pt

Luís A. Alexandre
IT - Networks and Multimedia Group
Covilhã - Portugal
lfbaa@di.ubi.pt

Abstract

In this paper we address some open questions on the recently proposed Zero-Error Density Maximization algorithm for MLP training. We propose a new version of the cost function that solves a training problem encountered in previous work and prove that the use of a nonparametric density estimator preserves the optimal solution. Some experiments are reported comparing this cost function to the usual mean-square error and cross entropy cost functions.

1. Introduction

The training of multi-layer perceptron (MLP) classifiers in statistical pattern recognition requires the use of objective functions (or cost/loss functions) in order to find the optimal set of parameters. Several proposals have been used ranging from the well known mean-square error (MSE) to a number of information theoretic objective functions [1, 3, 6, 8]. It was based on the latter that we have recently presented the Zero-Error Density Maximization procedure (Z-EDM) [7]. Its basic idea is to update the parameters of an MLP such as to maximize the error density at the origin (details are given in the following section). Some questions and problems remained unclear, such as the need for a larger number of training epochs or the influence of the smoothing parameter of the kernel density estimation. Also, the answer to the fundamental question about the influence in the final solution of the nonparametric density estimator was not known. In this paper, we study and solve these issues while complementing with a set of experiments where Z-EDM, MSE and Cross Entropy (CE) cost functions are compared.

2. The Zero-Error Density Maximization Procedure

We consider an MLP with one hidden layer, an output vector $\mathbf{y}$ and a target vector $\mathbf{t}$ (multi-class problems). We also consider a training set with N pattern vectors, such that the nth sample produces an output $\mathbf{y}(n)$ which is compared with the corresponding target $\mathbf{t}(n)$ to produce the error (or more precisely, the deviation) $\mathbf{e}(n) = \mathbf{t}(n) - \mathbf{y}(n),\ n = 1, \ldots, N$. Target vectors are encoded in an *one-out-of-C* scheme such that $\mathbf{t} = [-1, \ldots, 1, \ldots, -1]$, where the 1 appears at the kth component, is the target for a pattern from class $\mathcal{C}_k$. One can easily see that the errors belonging to each class lie in disjoint hypercubes with the origin as their unique common point. The three-class case is represented in Figure 1.

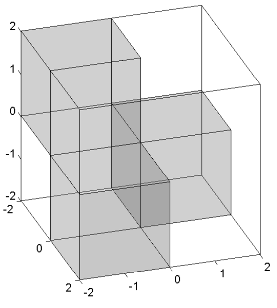


Figure 1. Support space (shadowed cubes) for the error distribution in a three-class problem (with target encoding as described in the text).

We expect, in the training process of an MLP, that the output $\mathbf{y}$ gets closer to the target $\mathbf{t}$ and thus the errors (deviations) will converge to the origin. In a limit scenario one would get $\mathbf{e}(n) = \mathbf{0}\ \forall n$, which amounts to a δ-Dirac distribution of the error variable centered at the origin. This means that, as training evolves, a distribution with a higher peak at the origin is induced in the errors. This idea leads us to the adaptive criteria of adjusting the parameter vector $\mathbf{w}$ (MLP weights) by maximizing the error density value at the origin

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} f(\mathbf{0}; \mathbf{w}) \tag{1}$$

where $\mathbf{w}^*$ is the optimal weight vector for the MLP and f is the error density. In practice, the error distribution is not known and making parametric assumptions would be very restrictive. Thus, we rely on nonparametric density estimation by using the well-known kernel density estimation procedure of Parzen windows [9]. Given a set of errors $\mathbf{e}(1), \ldots, \mathbf{e}(N)$, the estimated density at $\mathbf{e} = \mathbf{0}$ is given by

$$\hat{f}(\mathbf{0}) = \frac{1}{Nh^d} \sum_{n=1}^{N} K\left(\frac{\mathbf{0} - \mathbf{e}(n)}{h}\right), \tag{2}$$

where K is a multidimensional kernel function, h is the smoothing parameter (or kernel bandwidth) and d the dimension of $\mathbf{e}$. By continuity, diferenciability and further reasons stated in section 3, we choose for K the multivariate Gaussian kernel with zero mean and unit covariance [9] giving the final expression to be optimized as

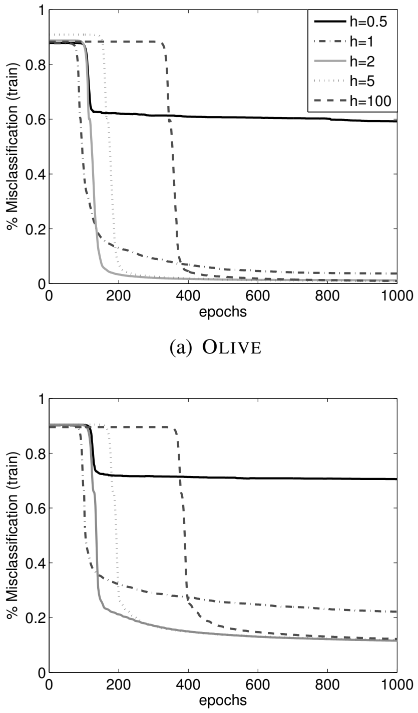
$$\hat{f}(\mathbf{0}) = \frac{1}{Nh^d} \sum_{n=1}^{N} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}\frac{\mathbf{e}(n)^2}{h^2}\right) \tag{3}$$

This new procedure, denoted as Zero-Error Density Maximization (Z-EDM), can be easily plugged in the usual backpropagation scheme. Note that expression (3) depends on the kernel parameter h. This parameter controls the smoothness of the density estimate and consequently the smoothness of the cost function. In order to better understand the influence of h in the training process we conducted several experiments. All datasets used in the experiments of section 4 were trained 100 times (full dataset) using several different values of h. Figure 2 shows the mean training curves for two datasets: OLIVE and CTG16. We found that a value of h smaller than 1 does not work, independently of the number of classes and/or number of training examples. When h is increased, the curve is basically shifted forward and a flat region appears in the earlier epochs. The general behaviour was the same for all datasets, which means that the number of classes (and consequently, the dimension of the error space) and the number of examples available for training are of little influence for the choice of h. Let us now consider, for simplicity sake, the case of a two-class problem (scalar output and target variables). The gradient of (3) is easily derived as

$$\frac{\partial \hat{f}(0)}{\partial w} = -\frac{1}{Nh} \sum_{n=1}^{N} \frac{1}{h^2\sqrt{2\pi}} \exp\left(-\frac{1}{2}\frac{e(n)^2}{h^2}\right) e(n) \frac{\partial e(n)}{\partial w} \tag{4}$$

If we look into this expression we may consider the function

$$\varphi(e) = \frac{1}{Nh^3} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}\frac{e^2}{h^2}\right) e \qquad e \in [-2, 2] \tag{5}$$



(a) OLIVE

(b) CTG16

Figure 2. Mean training curves with Z-EDM for different values of h.

as a weight function of the gradient "particle" $\frac{\partial e}{\partial w}$. Figure 3 shows $\varphi(e)$ for some values of N and h. We can see that gradient particles corresponding to larger values (in absolute value) of e get larger weights and gradient particles corresponding to smaller values of e will have a small contribution to the update value (4) of the parameter w. Of course, if we increase N or h, then $\varphi(e) \to 0$ and this is the reason for the initial flat platforms encountered in the first epochs of the training error (see Figure 2). If we also look to the order of magnitude of the values given by (5), we can conclude that this behaviour is due to the initial work being done by the adaptive learning rate procedure[1] while attempting to compensate those orders of magnitude.

In fact, we can make some modifications to our cost function in order to avoid this problem. Note that minimizing expression (3) is equivalent to the minimization of

$$\sum_{n=1}^{N} h^2 \exp\left(-\frac{1}{2}\frac{\mathbf{e}(n)^2}{h^2}\right) \tag{6}$$

in the sense that the same solutions are encountered, because $\frac{1}{Nh^d\sqrt{2\pi}}$ and h^2 are just positive scaling factors[2]. This

[1] We use an adaptive learning rate procedure as described in [7].

[2] We keep the factor h^2 to allow a simplification of the gradient.

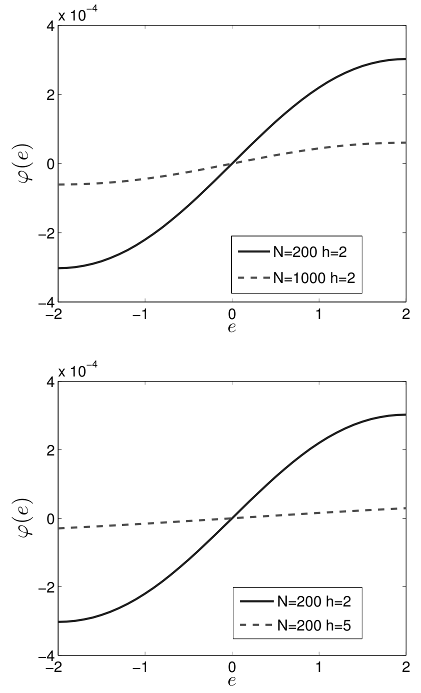


Figure 3. $\varphi(e)$ as in (5) for different values of N and h.

is the new expression used in the experiments of section 4.

3. Preserving the global maximum

The maximum value of $f(\mathbf{0};\mathbf{w})$ in (1) is attained when the distribution of the errors is a δ-Dirac, or equivalently, when $\mathbf{e}(n) = \mathbf{0}, \forall n$. It is important to show that the use of kernel density estimation in (2) preserves this maximum. In this sense, we must evaluate the gradient and the Hessian of (2) at the origin. Let $\bar{\mathbf{e}} = (\mathbf{e}(1), \ldots, \mathbf{e}(N))$. Using the chain rule, the first derivatives are given by

$$\frac{\partial \hat{f}(\mathbf{0})}{\partial \mathbf{e}(k)} = -\frac{1}{Nh^{d+1}} K'\left(-\frac{\mathbf{e}(k)}{h}\right) \tag{7}$$

and

$$\left.\frac{\partial \hat{f}(\mathbf{0})}{\partial \mathbf{e}(k)}\right|_{\bar{\mathbf{e}}=\mathbf{0}} = 0 \Leftrightarrow K'(\mathbf{0}) = 0 \tag{8}$$

Thus, condition (8) sets the property that the kernel must satisfy in order to have a stationary point at the origin. To infer about its nature we compute the second derivatives

$$\frac{\partial^2 \hat{f}(\mathbf{0})}{\partial \mathbf{e}^2(k)} = \frac{1}{Nh^{d+2}} K''\left(-\frac{\mathbf{e}(k)}{h}\right) \tag{9}$$

$$\frac{\partial^2 \hat{f}(\mathbf{0})}{\partial \mathbf{e}(n)\partial \mathbf{e}(k)} = 0 \qquad n \neq k \tag{10}$$

Hence at $\bar{\mathbf{e}} = \mathbf{0}$, we have

$$\left.\frac{\partial^2 \hat{f}(\mathbf{0})}{\partial \mathbf{e}^2(k)}\right|_{\bar{\mathbf{e}}=\mathbf{0}} = \frac{1}{Nh^{d+2}} K''(\mathbf{0}) \tag{11}$$

$$\left.\frac{\partial^2 \hat{f}(\mathbf{0})}{\partial \mathbf{e}(n)\partial \mathbf{e}(k)}\right|_{\bar{\mathbf{e}}=\mathbf{0}} = 0 \qquad n \neq k \tag{12}$$

Hence the Hessian matrix is of the form $\frac{1}{Nh^{d+2}} K''(\mathbf{0})\mathbf{I}$, which means that there exists an unique eigenvalue $\lambda = \frac{1}{Nh^{d+2}} K''(\mathbf{0})$ with multiplicity N. Thus, the origin is a (strict) local maximum if

$$\lambda < 0 \Leftrightarrow K''(\mathbf{0}) < 0 \tag{13}$$

This is the second condition that the kernel function must satisfy. We proceed by showing that a global maximum of (2) is reached when $\mathbf{e}(n) = \mathbf{0}\ \forall n$. We must show that

$$\left.\hat{f}(\mathbf{0})\right|_{\bar{\mathbf{e}}=\mathbf{0}} \geq \left.\hat{f}(\mathbf{0})\right|_{\bar{\mathbf{e}}} \quad \Leftrightarrow \quad NK(\mathbf{0}) \geq \sum_{n=1}^{N} K\left(\frac{\mathbf{0}-\mathbf{e}(n)}{h}\right) \tag{14}$$

Now, if the kernel function is unimodal with the mode centered at the origin, one has

$$\sum_{n=1}^{N} K\left(\frac{\mathbf{0}-\mathbf{e}(n)}{h}\right) \leq N \max_{n} K\left(\frac{\mathbf{0}-\mathbf{e}(n)}{h}\right) \leq NK(\mathbf{0}) \tag{15}$$

Hence, if the kernel function satisfies the following three conditions

1. $K'(\mathbf{0}) = 0$
2. $K''(\mathbf{0}) < 0$
3. unimodal with mode at the origin

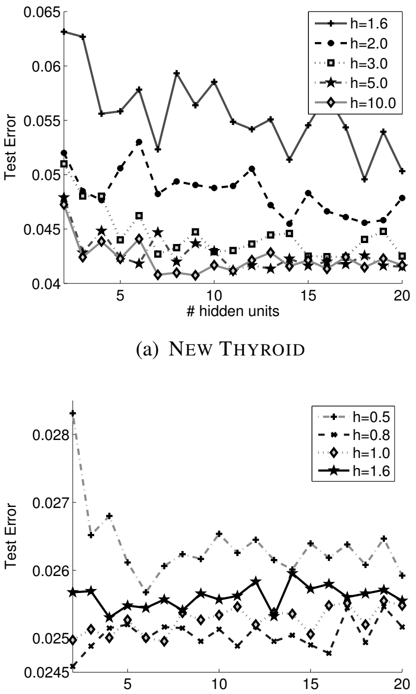
then the use of a nonparametric density estimator (2) to build our cost function (3) preserves the global maximum. It is easy to see that the Gaussian kernel satisfies these conditions and thus the justification for our choice.

4. Experimental Results

To evaluate the generalization ability of MLP's trained with Z-EDM, we conducted a train and test procedure with five datasets. WDBC, NEW THYROID and SONAR were taken from the UCI repository [2], CTG16 is a dataset containing measurements and classification results of cardiotocographic examination of 2126 foetuses (see [5] for a more detailed description) and OLIVE deals with the classification and prediction of the origin of different Italian olive oil samples [4]. Table 1 gives a brief description of the five datasets.

Table 1. Description of the five datasets used in the experiments. The last column reports the value of h used in each dataset.

Datasets	#Samples	#Features	#Classes	h
WDBC	569	30	2	0.8
CTG16	2126	16	10	10
OLIVE	572	8	9	10
SONAR	208	60	2	3
NEW THYROID	215	5	3	10



(a) NEW THYROID

(b) WDBC

Figure 4. Mean error lines (100 repetitions) for different values of h.

Note that we have to set the value of the smoothing parameter h in (6). The strategy was to try several values of h and choose the one that produces better results. We first used $h \in \{1.6,\ 2,\ 3,\ 5,\ 10\}$. Graphs as shown in Figure 4 were produced and used to choose the best value of h for each dataset. For example, in Figure 4(a) we observe that increasing h produces better results. However, we verified an opposite behaviour in WDBC, where smaller values of h seemed to be preferable. Thus, we also tried several values in the range $[0.5, 1.6]$. Figure 4(b) shows the mean errors for some of these values. We verified that decreasing h to a value around 0.8 the mean errors also decreased. However, a large decrease of h caused the errors to increase again (see the mean line for $h = 0.5$ in Figure 4(b)). The last column of Table 1 presents the value of h chosen for each dataset. The following procedure was performed 100 times: divide the data randomly in two subsets, half for training and half for testing; train and test the network; interchange the roles of the training and test sets; perform training and test again. This procedure was applied to several MLP's varying the number of hidden units from 2 to 20. For comparison purposes we also applied the same experimental procedure to MLP's trained with the mean square error (MSE) and cross entropy (CE) cost functions [1]. Also, to have a fair comparison, the same 100 train/test partitions were used for the three algorithms. Figure 5 shows the mean test error lines for these experiments.

We can see that for three datasets, CTG16, OLIVE and SONAR, the mean test error lines are quite similar. However, for WDBC we clearly observe that Z-EDM has a consistently lower mean line than MSE and CE. In NEW THYROID we observe a similar behaviour but in this case CE performs better than MSE and Z-EDM. Table 2 shows the mean test errors and standard deviations (in brackets) for some values of the number of hidden units.

The table confirms that Z-EDM performs better in WDBC while MSE and CE have similar results. This behaviour is inverted in NEW THYROID, where CE performs better than Z-EDM and MSE (which have similar results). The results also highlight some differences in other datasets that are not visible in Figure 5. For example, we can see that in CTG16, Z-EDM and MSE achieve its best result with $hid = 20$ corresponding to an error of 15.7% while CE has 16% as its best result. Also in SONAR we encounter some differences. Z-EDM achieves 21.7% while CE only achieves 22%. Note however that in this case we have higher standard deviations. To verify if the differences encountered are statistically significant, we applied the non-parametric Mann-Whitney test for the location of two independent samples. The test was only applied to the best results of each method listed in the bottom of Table 2. The p-values are shown in Table 3. A p-value smaller than 0.05 (5% level) was considered statistically significant evidencing differences in the results. This is what happens in WDBC. The result of Z-EDM is significantly better than the result of MSE. Comparing with CE the p-value is barely above the threshold 0.05. It would be significant if we relaxed our threshold to the 10% level.

We also encounter significant differences in CTG16. Z-EDM and MSE, performing equally, achieve a significant better result than CE. In the other datasets the differences are not statistically significant. In particular, the difference mentioned above for the SONAR dataset is not significant. It is also interesting to see that the differences encountered

Table 2. Test error rates (%) and standard deviations (in brackets) for the train and test procedure. For each dataset the most significant results were chosen. The top line of each box is the number of hidden units. The last box presents the best results.

WDBC	2	3	4	5	6	7
Z-EDM	2.46(0.49)	2.49(0.51)	2.51(0.49)	2.52(0.52)	2.50(0.53)	2.52(0.53)
MSE	2.59(0.45)	2.61(0.54)	2.64(0.55)	2.59(0.55)	2.59(0.54)	2.62(0.54)
CE	2.59(0.49)	2.65(0.50)	2.62(0.46)	2.64(0.52)	2.62(0.51)	2.64(0.45)

CTG16	15	16	17	18	19	20
Z-EDM	16.2(0.7)	16.0(0.8)	15.9(0.8)	15.8(0.8)	15.8(0.7)	15.7(0.7)
MSE	16.1(0.7)	16.1(0.7)	15.9(0.7)	16.0(0.8)	15.9(0.8)	15.7(0.6)
CE	16.3(0.7)	16.4(0.7)	16.2(0.7)	16.1(0.7)	16.0(0.7)	16.0(0.7)

OLIVE	15	16	17	18	19	20
Z-EDM	5.52(0.70)	5.56(0.68)	5.41(0.75)	5.50(0.66)	5.47(0.68)	5.33(0.71)
MSE	5.55(0.61)	5.45(0.62)	5.50(0.63)	5.48(0.68)	5.45(0.68)	5.45(0.67)
CE	5.57(0.63)	5.56(0.67)	5.44(0.63)	5.43(0.65)	5.48(0.64)	5.44(0.63)

SONAR	12	13	14	15	16	17
Z-EDM	22.2(2.8)	21.8(2.7)	22.1(2.8)	21.7(2.9)	21.7(2.7)	21.9(2.8)
MSE	22.0(3.1)	22.1(2.8)	21.9(2.8)	22.0(3.1)	22.0(2.7)	21.9(2.7)
CE	22.3(2.8)	22.2(2.7)	22.0(2.9)	22.2(2.6)	22.2(2.9)	22.2(2.9)

NEW THYROID	3	7	9	12	14	15
Z-EDM	4.24(1.20)	4.08(1.13)	4.07(1.13)	4.21(1.62)	4.16(1.20)	3.93(1.06)
MSE	4.26(1.24)	4.14(1.23)	4.12(1.23)	4.13(1.23)	4.13(1.17)	4.09(1.20)
CE	4.05(1.14)	3.98(1.05)	3.99(1.06)	3.94(1.04)	3.93(1.12)	3.93(1.06)

Best	Z-EDM		MSE		CE	
	hid	%	*hid*	%	*hid*	%
WDBC	2	2.46(0.49)	2	2.59(0.45)	2	2.59(0.49)
CTG16	20	15.7(0.7)	20	15.7(0.6)	19	16.0(0.7)
OLIVE	20	5.33(0.71)	15	5.45(0.62)	18	5.43(0.65)
SONAR	16	21.7(2.7)	14	21.9(2.8)	14	22.0(2.9)
THYROID	9	4.07(1.13)	15	4.09(1.20)	15	3.93(1.06)

Table 3. p-values of the Mann-Whitney test for differences in the best results.

	WDBC		CTG16		OLIVE		SONAR		THYROID	
	CE	MSE	CE	MSE	CE	MSE	CE	MSE	CE	MSE
Z-EDM	0.058	0.026	0.002	0.898	0.371	0.186	0.455	0.663	0.505	0.981
CE	-	0.719	-	0.000	-	0.629	-	0.725	-	0.535

in Figure 5 and in the results of Table 2 for NEW THYROID are not significant.

5 Conclusion

In this paper we present some new developments on the Zero-Error Density Maximization procedure. One of the problems encountered before was the need of more training epochs than the usual MSE or CE. This was understood to be caused by the scaling factors present in the cost function and thus a new version is here proposed that eliminates this problem. We also show that the use of a nonparametric density estimator in the derivation of the proposed cost function to substitute the (unknown) true density, does not affect the optimal solution. This means that we can simply rely on the data with no *a priori* information needed about the distribution, while guaranteeing that the optimal solution is preserved. The experimental procedure has also shown that there are types of datasets, such as WDBC, where Z-EDM can be a valuable alternative in comparison with MSE or CE. It would be important, in future work, to characterize these datasets in order to have some practical rule that could help us to choose the best approach (CE, MSE or Z-EDM) for each dataset.

References

[1] C. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

[2] C. Blake and C. Merz. UCI Repository of machine learning databases University of California, Irvine, Dept. of Information and Computer Sciences, http://www.ics.uci.edu/∼mlearn/MLRepository.html 1998.

[3] D. Erdogmus and J. C. Principe. An Error-Entropy Minimization Algorithm for Supervised Training of Nonlinear Adaptive Systems. *IEEE Transactions on Signal Processing*, 50(7):1780–1786, 2002.

[4] M. Forina and C. Armanino. Eigenvector projection and simplified non-linear mapping of fatty acid content of italian olive oils. *Ann. Chim. (Rome)*, 50:127–155, 1981.

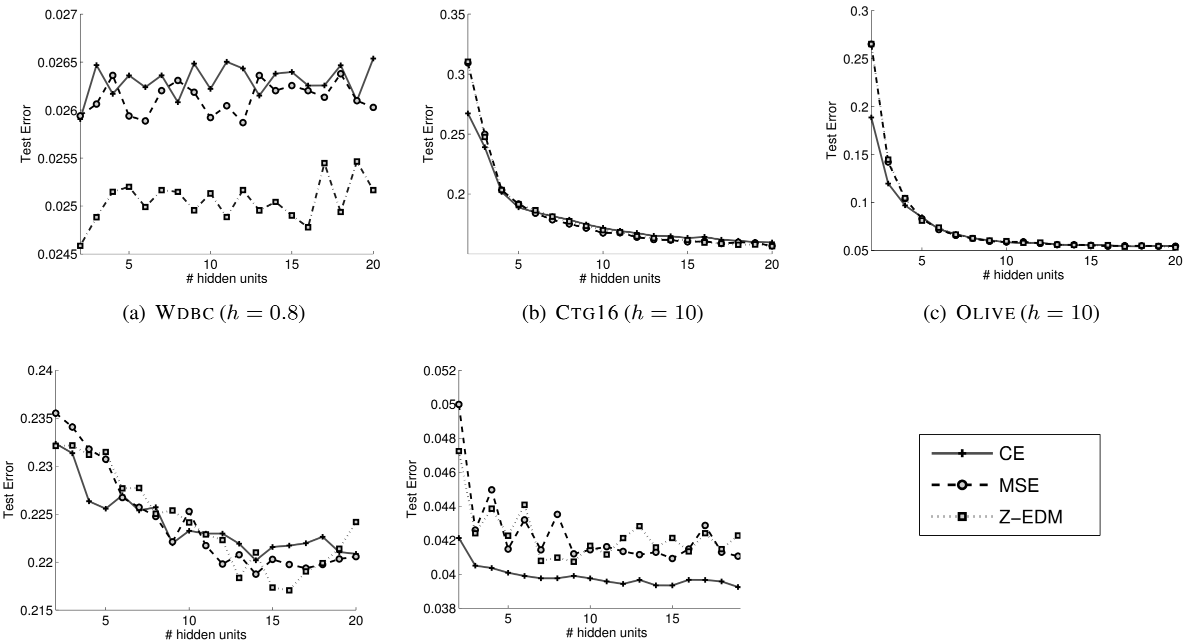
[5] J. Marques de Sá. *Pattern Recognition: Concepts, Methods and Applications*. Springer Verlag, 2001.

[6] J. Santos, L. Alexandre, and J. Marques de Sá. The Error Entropy Minimization Algorithm for Neural Network Classification. In *Int. Conf. on Recent Advances in Soft Computing*, Nottingham, United Kingdom, 2004.

[7] L. Silva, L. Alexandre, and J. Marques de Sá. Neural network classification: Maximizing zero-error density. In *ICAPR 2005, LNCS 3686, 127–135*, 2005.

[8] L. Silva, J. Marques de Sá, and L. Alexandre. Neural Network Classification using Shannon's Entropy. In *European Symposium on Artificial Neural Networks*, 2005.

[9] B. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman & Hall, 1986.



(a) WDBC ($h = 0.8$)
(b) CTG16 ($h = 10$)
(c) OLIVE ($h = 10$)
(d) SONAR ($h = 3$)
(e) NEW THYROID ($h = 10$)

Figure 5. Mean lines of test error obtained in the experimental procedure.