Pattern Recognition 43 (2010) 1984–1992

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

A new convex objective function for the supervised learning of single-layer neural networks

Oscar Fontenla-Romero [*], Bertha Guijarro-Berdiñas, Beatriz Pérez-Sánchez, Amparo Alonso-Betanzos

Laboratory for Research and Development in Artificial Intelligence (LIDIA), Department of Computer Science, Faculty of Informatics, University of A Coruña, Campus de Elviña s/n, 15071 A Coruña, Spain

A R T I C L E I N F O

Article history:
Received 8 June 2009
Received in revised form
27 October 2009
Accepted 26 November 2009

Keywords:
Single-layer neural networks
Global optimum
Supervised learning method
Least squares
Convex optimization
Incremental learning

A B S T R A C T

This paper proposes a novel supervised learning method for single-layer feedforward neural networks. This approach uses an alternative objective function to that based on the MSE, which measures the errors before the neuron's nonlinear activation functions instead of after them. In this case, the solution can be easily obtained solving systems of linear equations, i.e., requiring much less computational power than the one associated with the regular methods. A theoretical study is included to proof the approximated equivalence between the global optimum of the objective function based on the regular MSE criterion and the one of the proposed alternative MSE function.

Furthermore, it is shown that the presented method has the capability of allowing incremental and distributed learning. An exhaustive experimental study is also presented to verify the soundness and efficiency of the method. This study contains 10 classification and 16 regression problems. In addition, a comparison with other high performance learning algorithms shows that the proposed method exhibits, in average, the highest performance and low-demanding computational requirements.

1. Introduction

For a single-layer feedforward neural network, with linear activation functions, the weight values minimizing the mean-squared error function (MSE) can be found in terms of the pseudo-inverse of a matrix [1,2]. Furthermore, it can be demonstrated that the MSE surface of this linear network is a quadratic function of the weights [3]. Therefore, this convex hyperparaboloidal surface can be easily traversed by a gradient descent method. However, if nonlinear activation functions are used then local minima can exist in the objective function based on the MSE criterion [4–6]. In [7] it was shown that the number of such minima can grow exponentially with the input dimension. Only in some specific situations it is guaranteed the lack of local minima. In the case of linearly separable patterns and a threshold MSE criterion, it was proved the existence of only one minimum in the objective function [8,9]. Nevertheless, this is not the general situation.

The contribution of this work is to present a new convex objective function, equivalent to the MSE, that does not contains local minima and the global solution is obtained using a system of linear equations. This system can be solved, for each output, with a complexity of $O(N^2)$, where N is the number of parameters of the network.

The problem of local minima for one-layer networks was rigorously demonstrated in [5], where an example with a sigmoid transfer function, for which the sum of squared errors presents a local minimum, is given. They pointed out that the existence of local minima is due to the fact that the error function is the superposition of functions whose minima are at different points. In this situation, a closed form solution is no longer possible.

Previous approaches, during the last decades, have been presented to overcome the problems emerged by the presence of these stationary points in single-layer neural networks. In [10], a globally convergent natural homotopy mapping is defined for single-layer perceptrons by deformation of the node nonlinearity. This homotopy tracks a possibly infinite number of weights by transforming coordinates and characterizing all solutions by a finite number of distinct and unique solutions. Although this approach ensures computation of a solution, it does not provide global optimization [1]. At the same time, these authors proposed in [11] a method for both the a posteriori evaluation of whether a solution is unique or globally optimal and for a priori scaling of desired vector values to ensure uniqueness, through analysis of the input data. Although these approaches are potentially helpful for evaluating optimality and uniqueness, the minima are characterized only after training is complete. In addition, other authors have proposed methods for different criteria from the MSE to avoid the problem of local minima in the objective

[*] Corresponding author.
E-mail addresses: ofontenla@udc.es (O. Fontenla-Romero), cibertha@udc.es (B. Guijarro-Berdiñas), bperezs@udc.es (B. Pérez-Sánchez), ciamparo@udc.es (A. Alonso-Betanzos).

0031-3203/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2009.11.024

nction to minimize. In this sense, in [12] it was proposed an on-line additive learning method for matching cost functions based on the Bregman divergence.

Pao [2] proposed the functional link approach that obtains an analytical solution of the weights establishing a system of linear equations $Xw = z$, where X is a matrix formed by the input patterns, w is the weight vector and z is another vector formed by the inverse of the activation function applied over the desired output. The dimensions of matrix X are $S \times N$, where S is the number of training patterns and N is the number of weights. As Pao mentions in his work, if $S = N$ and the determinant of X is not zero then the solution can be obtained by $w = X^{-1}z$. However, this is not the common situation, because in real data sets usually $S > N$ or $S < N$. For these last cases, Pao analyzes the situation separately. In the case $S < N$, a large number of solutions could be obtained (perhaps an infinite number of them) which is not obviously desired. He proposed a partition of X to avoid in some way this problem. In the other case, $S > N$, an infinitely large number of orthonormal functions could be generated, and then a method based on the pseudoinversion is proposed ($w = (X^T X)^{-1} X^T z$). However, as he already mentions, this formulation could be often unacceptable as is indicated by the high error value at the end of the learning process.

Some studies for multilayer feedforward neural networks have used similar results to the one proposed in [2] for the back-propagation of the desired output or for the learning of the weights of the output layer. Specifically, there have been heuristic proposals of least-squares initialization and training approaches [13–17]. Of special interest is [15], where three least-squares initialization schemes were compared for speed and performance. Nevertheless, these methods did not rigorously consider the transformation of the desired output through the nonlinear activation functions as they did not take into account the scaling effects of the slopes of nonlinearities in the least squares problem. This is an important issue as it will be discussed later.

Lastly, in a previous paper [18], a new learning method, for single-layer neural networks, based on a system of linear equations was presented. This approach is possible due to the use of a new objective function that measures the sum of the squared errors *before* the nonlinear activation functions instead of *after* these functions, as it is usually done. Although the experimental results presented in this previous work support the validity and soundness of the proposed method, some theoretical research was still necessary to proof the equivalence between the global optimum of the objective function based on the MSE after the nonlinearities and the proposed objective function (minimization of the MSE before the nonlinear functions). This paper completes the mentioned research presenting a theoretical analysis and considering in the objective function the scaling effects of the slope of the nonlinear transfer function. Besides, a new set of linear equations, to obtain the optimal weights for the problem, are derived.

2. Description of the proposed method

The architecture of the considered neural network is shown in Fig. 1. The inputs are denoted as x_{is} and outputs as y_{js} being $i = 0, 1, \ldots, I$; $j = 1, 2, \ldots, J$ and $s = 1, 2, \ldots, S$. The numbers I, J and S represent the number of inputs, outputs and training samples, respectively. The network contains only a single layer of J output neurons with nonlinear activation functions $f_1, f_2, \ldots, f_J$. The set of equations relating inputs and outputs is given by

$$y_{js} = f_j(z_{js}) = f_j\left(\sum_{i=0}^{I} w_{ji} x_{is}\right), \quad j = 1, 2, \ldots, J, \quad s = 1, 2, \ldots, S, \tag{1}$$

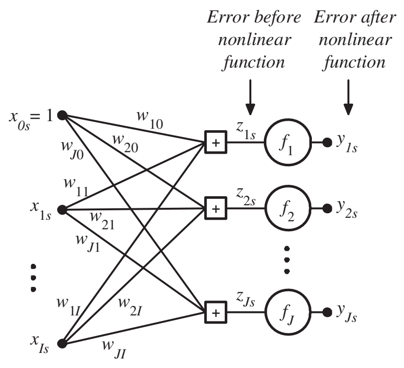

Fig. 1. Architecture of a single-layer feedforward neural network.

where w_{j0} and w_{ji}, $i = 1, 2, \ldots, I$, are, respectively, the bias and the weights associated with neuron j (for $j = 1, 2, \ldots, J$). The system presented in (1) has $J \times S$ equations and $J \times (I+1)$ unknowns. In practice, since the number of data is large ($S \gg I+1$), this set of equations does not have a solution, and consequently, it cannot be solved analytically.

Thus, the widely employed approach to obtain the optimal weights is based on the optimization, by means of an iterative procedure, of an objective function that measures the errors obtained by comparing the real output of the network and some desired response.

2.1. *Regular objective function: mean-squared error after nonlinearities*

Currently, different objective functions have been proposed being one of the most used that based on the mean-squared error (MSE) criterion. This is the function considered in this work. Thus, the usual approach is to consider some errors, ε_{js} measured *after* the nonlinearities. Therefore, the set of equations relating inputs and outputs is now defined as

$$\varepsilon_{js} = d_{js} - y_{js} = d_{js} - f_j\left(\sum_{i=0}^{I} w_{ji} x_{is}\right), \quad j = 1, 2, \ldots, J, \quad s = 1, 2, \ldots, S, \tag{2}$$

where d_{js} is the desired output for neuron j and the training pattern s. To estimate (learn) the weights, the sum of squared errors defined as

$$MSEA = \sum_{s=1}^{S} \sum_{j=1}^{J} \varepsilon_{js}^2 = \sum_{s=1}^{S} \sum_{j=1}^{J} \left(d_{js} - f_j\left(\sum_{i=0}^{I} w_{ji} x_{is}\right)\right)^2 \tag{3}$$

is minimized. There exists many gradient descent methods that can be used to obtain a saddle point of this function.

It is important to note that, due to the presence of the nonlinear activation functions f_j, the function in (3) is nonlinear in the weights. In this situation, the absence of local minima in *MSEA* is not guaranteed, as was demonstrated in [5]. Therefore, a gradient descent method can be stuck in a local minimum instead of achieving the global optimum of the objective function.

2.2. *New objective function: mean-squared error before nonlinearities*

In order to avoid the problems mentioned in the previous section, a new approach for the supervised learning of single-layer feedforward neural networks is proposed. This method is based on the use of an alternative objective function that measures the s *before* the nonlinear activation functions, that is, using the z_{js} instead of the y_{js} variables as it is graphically illustrated in Fig. 1.

In a previous work [18], these alternative errors were already used for the learning of single-layer neural networks. However, two main theoretical issues were not taken into account:

- The influence on the slope of the nonlinear activation functions in the errors considered in the objective function. In this new research, it will be demonstrated that the scaling effects of these functions can be calculated to obtain a most accurate solution when the variability of the error is small.
- The equivalence between the solution obtained considering the errors after and before the nonlinear function. This analysis is presented in this paper, taking into account the effects mentioned in the previous item.

Other authors have used the errors before the nonlinear activation function for the initialization or learning of multilayer feedforward neural networks [13–17]. However, all these previous approaches did not also take into account the scaling effects of the slope of the nonlinear activation functions in the errors measured before these functions. In this work, a new objective function, that considers the influence of the slope, and a novel learning method based on this objective function are presented. In addition, it is rigorously proved that the solution obtained is approximatively the same as using the objective function based on the regular mean-squared error. This fact is showed in Theorem 1. Given that, for each output j, the weights w_{ji}, $i=0,1,\ldots,I$ are related only to the output y_{js} it is clear that the problem of learning the weights can be separated into J independent problems (one for each output j). Thus, for notational simplicity, in what follows only one of these problems (for a fixed j) will be dealt with.

Theorem 1. *Let x_{is} $(i=1,\ldots,I,\ s=1\ldots,S)$ be the inputs of a single-layer feedforward neural network, d_{js} and y_{js} be the desired and real output for the output neuron j and the pattern s, w_{ji} be the weights, and f_j, f_j^{-1}, f_j' be the nonlinear function, its inverse and its derivative. Then, the minimization of the MSE between d_{js} and y_{js} at the output of the nonlinearity is approximately equivalent, up to first order of a Taylor series, to minimizing the MSE before the nonlinearity, i.e., between $z_{js}=\sum_{i=0}^{I} w_{ji}x_{is}$ and $\bar{d}_{js}=f_j^{-1}(d_{js})$ weighted according to the value of the derivative of the nonlinearity at the corresponding operating point. Mathematically, this property can be written as*

$$\min_{w_{ji}} MSEA_j = \sum_{s=1}^{S}(d_{js}-f_j(z_{js}))^2 \approx \min_{w_{ji}} MSEB_j = \sum_{s=1}^{S}(f_j'(\bar{d}_{js})(\bar{d}_{js}-z_{js}))^2 \tag{4}$$

or equivalently

$$\min_{w_{ji}} MSEA_j = \sum_{s=1}^{S}\left(d_{js}-f_j\left(\sum_{i=0}^{I} w_{ji}x_{is}\right)\right)^2 \approx \min_{w_{ji}} MSEB_j = \sum_{s=1}^{S}\left(f_j'(\bar{d}_{js})\left(f_j^{-1}(d_{js})-\sum_{i=0}^{I} w_{ji}x_{is}\right)\right)^2. \tag{5}$$

Proof. Considering the regular MSE minimization problem:

$$\min_{w_{ji}} MSEA_j = \sum_{s=1}^{S}(d_{js}-y_{js})^2 \tag{6}$$

$$= \sum_{s=1}^{S}\left(d_{js}-f_j\left(\sum_{i=0}^{I} w_{ji}x_{is}\right)\right)^2 \tag{7}$$

and employing $y_{js}=f(z_{js})$ and $d_{js}=f_j(\bar{d}_{js})$ in the right-hand side of Eq. (6), the following equivalent minimization problem can be obtained:

$$\min_{w_{ji}} MSEA_j = \sum_{s=1}^{S}(d_{js}-y_{js})^2 = \sum_{s=1}^{S}(f_j(\bar{d}_{js})-f_j(z_{js}))^2. \tag{8}$$

Let $\bar{d}_{js}=f_j^{-1}(d_{js})$ be the desired output before the nonlinear functions, thus the error before the output neuron can be defined as $\bar{\varepsilon}_{js}=\bar{d}_{js}-z_{js}=f_j^{-1}(d_{js})-\sum_{i=0}^{I} w_{ji}x_{is}$. If the variability of this error $(var(\bar{\varepsilon}_j))$ is small then the following first order Taylor series approximation on each component of the output vector can be used:

$$f_j(z_{js})=f_j(\bar{d}_{js}-\bar{\varepsilon}_{js})\approx f_j(\bar{d}_{js})-f_j'(\bar{d}_{js})\bar{\varepsilon}_{js}. \tag{9}$$

Substituting the right-hand side of Eq. (9) in Eq. (8), it can be obtained the desired result:

$$\min_{w_{ji}} MSEA_j = \sum_{s=1}^{S}(d_{js}-f_j(z_{js}))^2 \approx \min_{w_{ji}} MSEB_j = \sum_{s=1}^{S}(f_j'(\bar{d}_{js})\bar{\varepsilon}_{js})^2 \tag{10}$$

or

$$\min_{w_{ji}} MSEA_j = \sum_{s=1}^{S}\left(d_{js}-f_j\left(\sum_{i=0}^{I} w_{ji}x_{is}\right)\right)^2 \approx \min_{w_{ji}} MSEB_j = \sum_{s=1}^{S}\left(f_j'(\bar{d}_{js})\left(f_j^{-1}(d_{js})-\sum_{i=0}^{I} w_{ji}x_{is}\right)\right)^2. \quad \square \tag{11}$$

The previous theorem is an important result because it establishes that the minimization of the mean-squared error at the output of the network is equivalent (up to first order) to the minimization of the error before the output neurons scaled by a factor $f_j'(\bar{d}_{js})$. This factor ensures that each error, corresponding to an input–output pair of the training set, is weighted appropriately according to the value of the nonlinear activation function at the corresponding point of the desired response. All the previous works failed to take into account this scaling factor, because they simply reflected the desired response to the input of the nonlinearity. This is not the best approximation if the variability of the error is small, as it is demonstrated in Theorem 1, because the scaling effect of the nonlinearity on the variance of the error should be considered.

As a result of this theorem, it is possible to use, alternatively, any of the minimization problems shown in (5) for the learning process of the neural network. This result is the basis of the method proposed in this work. The only requirement for this alternative minimization problem is that the activation functions must be invertible. However, this is not a very restrictive condition because most used activation functions in neural networks, like for example the sigmoid function, are invertible. The important consequence of the described theorem is that if the minimization problem on the right-hand side of Eq. (5) is used, then its solution can be obtained easily because it is a convex optimization problem. Hence, the absence of local minima is assured in this situation. In this case, the system of equations described previously in Eq. (2) can be rewritten in the following way:

$$\bar{\varepsilon}_{js}=\bar{d}_{js}-z_{js}=f_j^{-1}(d_{js})-\sum_{i=0}^{I} w_{ji}x_{is}, \quad s=1,2,\ldots,S, \ j=1,2,\ldots,J, \tag{12}$$

that measures the errors in terms of the inputs (x_{is}). It is important to note that in Eq. (12), the unknowns (weights of the network) are not affected by the activation function (f_j) and, thus, the error is linear with respect to the parameters of the network. Then, the aim is to minimize the following alternative objective function $MSEB_j, \forall j = 1, \ldots, J$:

$$MSEB_j = \sum_{s=1}^{S} (f_j'(\overline{d}_{js})\overline{\varepsilon}_{js})^2 = \sum_{s=1}^{S} \left(f_j'(\overline{d}_{js}) \left(f_j^{-1}(d_{js}) - \sum_{i=0}^{I} w_{ji} x_{is} \right) \right)^2. \quad (13)$$

The global optimum of this convex objective function can be easily obtained computing its derivative with respect to the weights of the network and setting the derivative to zero:

$$\frac{\partial MSEB_j}{\partial w_{jp}} = -2 \sum_{s=1}^{S} \left(f_j'(\overline{d}_{js}) \left(f_j^{-1}(d_{js}) - \sum_{i=0}^{I} w_{ji} x_{is} \right) \right) x_{ps} f_j'(\overline{d}_{js}) = 0,$$
$$p = 0, 1, \ldots, I, \quad (14)$$

that can be rewritten as follows:

$$\sum_{i=0}^{I} A_{pi} w_{ji} = b_{pj}, \quad p = 0, 1, \ldots, I, \quad (15)$$

where $A_{pi} = \sum_{s=1}^{S} x_{is} x_{ps} f'^2_j(\overline{d}_{js})$, $b_{pj} = \sum_{s=1}^{S} \overline{d}_{js} x_{ps} f'^2_j(\overline{d}_{js})$ and $\overline{d}_{js} = f_j^{-1}(d_{js})$.

For the output j, the system (15) has $I+1$ linear equations and unknowns and, thereby, there exists only one real solution (except for ill-conditioned problems) which corresponds to the global optimum of the objective function.[1] Several computationally efficient methods can be used to solve this kind of systems with a complexity of $O(N^2)$, where $N = I+1$ is the number of weights of the network for the output j [19,20].

One of the main differences of this method and the one presented previously by Pao [2] is the dimension of the system of linear equations. The approach by Pao uses a nonsquared system ($S \times N$) while in this case the system is squared ($N \times N$), where S is the number of patterns and N the number of weights. This is an important advantage because, independently of number of training patterns S, the system always has an unique solution. As was explained in Section 1, if the system is not squared the underdetermined and overdetermined cases can generate non-accurate and multiple solutions.

Finally, other interesting characteristics of the proposed learning method are its incremental and distributed nature. This can be clearly checked by observing Eq. (15). The coefficients (A_{pi} and b_{pj}) are calculated as a sum of terms in function of the points (input–desired output) of the data set; therefore, and due to the commutative and associative properties of the sum, the same solution is obtained independently of the order and occurrence of the data points. Also, M networks can be trained in different computers with partial data sets ($D_1, D_2, \ldots, D_M$) and afterwards a single network representing the union of the M networks could be obtained. This can be achieved by summing the corresponding M matrices of coefficients of all the networks and obtaining the weights for this new matrix.

3. An illustrative example

In this section, an example is presented to illustrate the characteristics of the proposed objective function (MSEB), compared with the regular mean-squared error (MSEA), and the behavior of the derived learning method. In order to do this, a simple topology was used, consisting of a one-layer neural network with one input (x), one output (y) with a logistic activation function, one weight (w_1) and one bias (w_0). The goal of this network is to learn the desired function $d = x + 3\sin(x) + \log(x)$ normalized in the interval [0.05,0.95]. Fig. 2 shows, for this example, the 3D-surface of the MSEA objective function and the one for the alternative MSEB. As can be seen, the new function is convex and its global optimum is approximately at the same point as that of the MSEA objective function. Also, in this figure is represented the error trajectory along training of the Levenberg–Marquardt (LM) algorithm, using the MSEA objective function, from two different initial states:

- In the first case, represented by the symbol (1), the method begins from the initial point defined by $w = -0.65$ and $b = 0.9$. In this situation the learning algorithm is capable of descending in the error surface to get a value near the global optimum.
- In the second case, indicated by the symbol (2) and the triangle, the same method begins from the initial point defined by $w = -1.1$ and $b = 0.1$. As can be observed, the method is not able to get the global minimum but is trapped in a plateau of the error function.

Fig. 3 depicts the level curves for the MSEA objective function and its global optimum (represented by a $\times$-mark). This figure

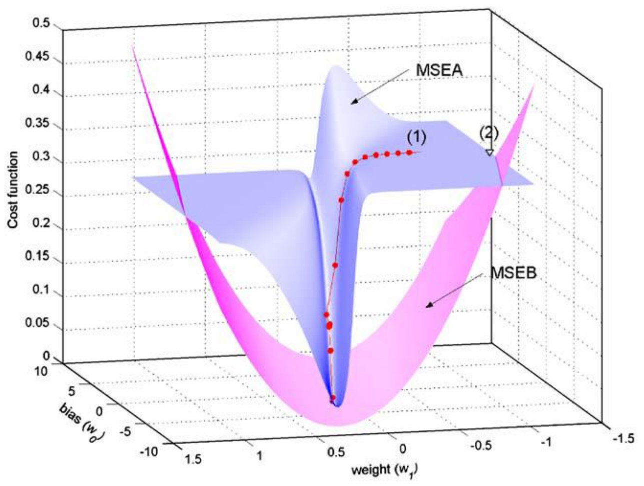

Fig. 2. Example of the regular objective function based on the MSE and the proposed objective function.

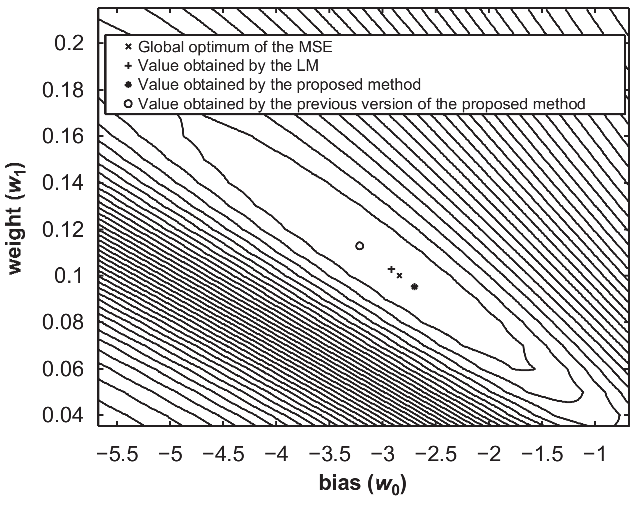

Fig. 3. Level curves near the global minimum of the MSE function.

[1] MATLAB® demo code available at http://www.dc.fi.udc.es/lidia/downloads/LLM

includes the point obtained by the LM method and the proposed learning method using the MSEB. As can be seen, the proposed method obtains a close approximation to the global optimum of the MSEA, and it improves the value obtained by the approach using the MSEB without the scaling term of the slopes of the activation functions, as was previously presented in [18].

4. Simulations

Several experiments were carried out to check the efficacy of the proposed method and its speed to get the global optimum. In order to accomplish this analysis several data sets were used for classification and regression problems and the proposed method was compared to some other relevant techniques. Although single-layer neural networks can use nonlinear activation functions in their unique layer (output layer), their discriminating capability is still linear as it was demonstrated by Minsky and Papert [21]. Thus, all the approaches were selected so as to have similar capacities to a single-layer neural network in terms of linear classification. In order to estimate the true error, a 10-fold cross-validation was employed for all cases. In addition, 50 random simulations were performed to obtain a mean value and a standard deviation for each case. The simulations were performed using MATLAB® and the method used to solve the system of linear equations in Eq. (15) was a general triangular factorization computed by Gaussian elimination with partial pivoting (LU matrix factorization).

4.1. Comparative analysis with the previous approach

The first experiment was carried out using 16 regression data sets. The aim in this kind of problems is to predict a future data point of a time series employing some previous samples. In order to perform this, each time series was preprocessed to transform the one-dimensional signal into a set of I-dimensional input patterns and one-dimensional desired output. This was accomplished employing a sliding window procedure and for all data sets I was equal to six. The topology of the one-layer neural network is determined by the dimension of the input and output data. Two different activation functions were employed: the hyperbolic tangent sigmoid function and the logarithmic sigmoid function. Although the results obtained for both were slightly different, the comparative study among the different approaches is equivalent for the two cases. For that reason, only the results obtained using hyperbolic tangent sigmoid function are shown in tables below.

Table 1 includes the benchmarking data sets employed in this experiment and the number of samples employed for each one. The first five data sets were obtained at Eric's Home page,[2] the next three were from StatLib Datasets Archive,[3] the Annulus data set from Eric Weeks's Chaotic Time Series repository[4] and the last ones from the Time Series Data Library.[5]

In order to carry out the comparative study, the weights of the single-layer neural network were obtained using the proposed method and the previous version presented in [18]. In addition, the least squares support vector machine (LS-SVM) [22], with linear kernels and regularization parameter equal to 0.1, was included in the comparative analysis, as it is one of the state-of-the-art approaches in the machine learning field and also leads to a convex optimization problem. The optimal value of the regularization

[2] Available at http://www.cse.ogi.edu/~ericwan/data.html
[3] Available at http://lib.stat.cmu.edu/datasets
[4] Available at http://www.physics.emory.edu/~weeks/research/tseries1.html
[5] Available at http://www-personal.buseco.monash.edu.au/~hyndman/TSDL

Table 1
Characteristics of the regression data sets.

Data set	Samples
Henon (chaotic time series generated by code)[2]	3000
Lorenz (chaotic time series generated by code)[2]	3000
Mackey-Glass (chaotic time series generated by code)[2]	2000
Ikeda (chaotic time series generated by code)[2]	2000
Laser K.U. Leuven (laser time series from Santa Fe Competition)[2]	1900
Dow-Jones (Dow-Jones Industrial Average closing values)[3]	5000
Saubts (recording of the height of ocean waves as a function of time)[3]	4096
Lmpavw (vertical ocean shear time series)[3]	6875
Annulus (chaotic time series from a rotating annulus sketch)[4]	3000
Kobe (seismograph of the Kobe earthquake, recorded at Tasmania)[5]	3042
Imports (monthly Australian imports from Japan)[5]	334
Co2 (monthly measurements of carbon dioxide above Mauna Loa, Hawaii)[5]	378
Oscillation (monthly measurements of differences in sea-surface air pressure between Darwin and Tahiti)[5]	1229
Blowfly (bi-daily blowfly population in glass jar)[5]	355
TreeRings (normalized tree-ring widths in dimensionless units, years: 525-1983)[5]	1453
Waves (forces on a cylinder suspended in a tank of water)[5]	314

parameter was obtained using cross-validation. Specifically, the LS-SVMlab Matlab Toolbox[6] (version 1.5) implemented by Pelckmans et al. was employed. Table 2 shows the mean *training* MSE and standard deviation obtained for each data set over the 50 simulations. Table 3 shows the same results but for the mean *test* MSE (true MSE estimated by cross-validation) and standard deviation obtained for each data set. The boxes in these tables indicate the best result comparing only the proposed and the previous approach. The black down triangles indicate the cases where the LS-SVM obtains worse results than the proposed method. Analyzing these tables, the following issues can be observed:

- The proposed method obtains, in general, better results (10 cases of 16) than the previous version. The mean percentage of improvement in the test MSE is 19.19%, for those cases in which the proposed method performs better, while the degradation percentage is only of 3.66% in the other cases. The cases (6 of 16) in which the previous approach is working better than the new one is due to the relative high magnitude of the error. It is worth noting that the smaller the term $\overline{\varepsilon}_{js}$ (error measured before the activation function), the more accurate the Taylor series approximation used in Eq. (9). Thus, when the error is relatively high the approximation is not so exact. As can be observed in Tables 2 and 3, these are the cases in which the previous approach is working better. Specifically, analyzing the test errors (Table 3) the proposed method achieves better results when the error is approximately lesser than $2.224e^{-2}$ (see the TreeRings compared to Imports and Waves data sets). So, it seems that around that value is the threshold that determines which method is the most accurate. Anyway, in the six cases in which the previous approach is working better in half of them the differences are very small (see TreeRings, Oscillation and Mackey data sets).
- In 9 of 16 cases the proposed method obtains better results than the LS-SVM with a mean percentage of improvement in the test MSE of 86.33%. Therefore it can be considered as an interesting method in the field of machine learning. In addition, it allows incremental and distributed training, in contradistinction to many of the standard algorithms. Thus, it can be used in a real time learning environment or

[6] Available at http://www.esat.kuleuven.ac.be/sista/lssvmlab/

ble 2
Mean training MSE $\pm$ standard deviation using hyperbolic tangent activation functions.

Data set	Proposed	Previous	LS-SVM
Henon	$2.083e^{-1} \pm 9.81e^{-6}$	$\boxed{1.867e^{-1} \pm 1.14e^{-5}}$	$1.891e^{-1} \pm 8.85e^{-6}$
Lorenz	$\boxed{6.281e^{-5} \pm 1.69e^{-8}}$	$7.700e^{-5} \pm 3.26e^{-8}$	$6.872e^{-4} \pm 1.23e^{-8}$ ▼
Mackey	$3.007e^{-2} \pm 2.42e^{-6}$	$\boxed{2.914e^{-2} \pm 2.43e^{-6}}$	$3.188e^{-2} \pm 2.13e^{-6}$ ▼
Ikeda	$1.716e^{-1} \pm 1.38e^{-5}$	$\boxed{1.648e^{-1} \pm 1.36e^{-5}}$	$1.643e^{-1} \pm 1.38e^{-5}$
Laser	$\boxed{4.202e^{-4} \pm 3.76e^{-8}}$	$4.816e^{-4} \pm 5.13e^{-8}$	$1.914e^{-3} \pm 5.27e^{-8}$ ▼
DowJones	$\boxed{2.974e^{-4} \pm 5.17e^{-9}}$	$4.484e^{-4} \pm 1.12e^{-8}$	$2.860e^{-5} \pm 3.37e^{-10}$
Saubts	$\boxed{8.022e^{-4} \pm 2.22e^{-8}}$	$9.125e^{-4} \pm 3.70e^{-8}$	$1.109e^{-3} \pm 1.94e^{-8}$ ▼
Lmpavw	$\boxed{5.327e^{-4} \pm 1.54e^{-8}}$	$6.874e^{-4} \pm 2.74e^{-8}$	$3.534e^{-4} \pm 1.07e^{-8}$
Annulus	$\boxed{3.461e^{-4} \pm 1.94e^{-8}}$	$4.109e^{-4} \pm 2.39e^{-8}$	$5.399e^{-4} \pm 4.80e^{-9}$ ▼
Kobe	$\boxed{5.963e^{-4} \pm 3.53e^{-8}}$	$6.394e^{-4} \pm 4.67e^{-8}$	$4.446e^{-3} \pm 1.24e^{-7}$ ▼
Imports	$\boxed{1.166e^{-2} \pm 1.02e^{-5}}$	$1.256e^{-2} \pm 1.09e^{-5}$	$8.829e^{-3} \pm 4.66e^{-6}$
Co2	$\boxed{1.866e^{-3} \pm 5.43e^{-7}}$	$2.239e^{-3} \pm 8.87e^{-7}$	$7.154e^{-3} \pm 8.79e^{-7}$ ▼
Oscillation	$3.584e^{-2} \pm 3.88e^{-6}$	$\boxed{3.561e^{-2} \pm 3.91e^{-6}}$	$3.564e^{-2} \pm 3.63e^{-6}$
Blowfly	$5.137e^{-2} \pm 2.66e^{-5}$	$\boxed{4.896e^{-2} \pm 2.49e^{-5}}$	$5.076e^{-2} \pm 1.95e^{-5}$
TreeRings	$2.205e^{-2} \pm 1.96e^{-6}$	$\boxed{2.198e^{-2} \pm 1.97e^{-6}}$	$2.217e^{-2} \pm 1.88e^{-6}$ ▼
Waves	$\boxed{1.106e^{-2} \pm 4.20e^{-6}}$	$1.110e^{-2} \pm 4.85e^{-6}$	$2.012e^{-2} \pm 5.92e^{-6}$ ▼

Boxes indicate the best result comparing only the proposed and the previous approach and black triangles indicate the cases where the LS-SVM obtains worse results than the proposed one.

Table 3
Mean test MSE $\pm$ standard deviation using hyperbolic tangent activation functions.

Data set	Proposed	Previous	LS-SVM
Henon	$2.090e^{-1} \pm 1.39e^{-4}$	$\boxed{1.877e^{-1} \pm 1.79e^{-4}}$	$1.900e^{-1} \pm 1.65e^{-4}$
Lorenz	$\boxed{6.428e^{-5} \pm 3.13e^{-7}}$	$7.944e^{-5} \pm 5.85e^{-7}$	$6.899e^{-4} \pm 8.87e^{-7}$ ▼
Mackey	$3.030e^{-2} \pm 4.01e^{-5}$	$\boxed{2.938e^{-2} \pm 4.42e^{-5}}$	$3.210e^{-2} \pm 4.46e^{-5}$ ▼
Ikeda	$1.726e^{-1} \pm 2.10e^{-4}$	$\boxed{1.664e^{-1} \pm 3.01e^{-4}}$	$1.656e^{-1} \pm 2.56e^{-4}$
Laser	$\boxed{4.234e^{-4} \pm 6.69e^{-7}}$	$4.868e^{-4} \pm 1.16e^{-6}$	$1.920e^{-3} \pm 1.49e^{-6}$ ▼
DowJones	$\boxed{2.979e^{-4} \pm 8.47e^{-8}}$	$4.499e^{-4} \pm 2.40e^{-7}$	$2.862e^{-5} \pm 7.19e^{-9}$
Saubts	$\boxed{8.045e^{-4} \pm 3.69e^{-7}}$	$9.165e^{-4} \pm 7.06e^{-7}$	$1.111e^{-3} \pm 4.75e^{-7}$ ▼
Lmpavw	$\boxed{5.344e^{-4} \pm 2.61e^{-7}}$	$6.909e^{-4} \pm 6.91e^{-7}$	$3.542e^{-4} \pm 1.96e^{-7}$
Annulus	$\boxed{3.474e^{-4} \pm 3.19e^{-7}}$	$4.136e^{-4} \pm 6.59e^{-7}$	$5.408e^{-4} \pm 2.80e^{-7}$ ▼
Kobe	$\boxed{5.999e^{-4} \pm 6.14e^{-7}}$	$6.444e^{-4} \pm 1.05e^{-6}$	$4.466e^{-3} \pm 4.31e^{-6}$ ▼
Imports	$\boxed{1.247e^{-2} \pm 1.66e^{-4}}$	$1.373e^{-2} \pm 2.44e^{-4}$	$9.112e^{-3} \pm 8.45e^{-5}$
Co2	$\boxed{1.925e^{-3} \pm 9.28e^{-6}}$	$2.332e^{-3} \pm 1.69e^{-5}$	$7.257e^{-3} \pm 2.38e^{-5}$ ▼
Oscillation	$3.624e^{-2} \pm 6.39e^{-5}$	$\boxed{3.612e^{-2} \pm 8.05e^{-5}}$	$3.602e^{-2} \pm 5.92e^{-5}$
Blowfly	$5.401e^{-2} \pm 4.19e^{-4}$	$\boxed{5.228e^{-2} \pm 4.96e^{-4}}$	$5.275e^{-2} \pm 3.28e^{-4}$
TreeRings	$2.226e^{-2} \pm 3.05e^{-5}$	$\boxed{2.224e^{-2} \pm 4.32e^{-5}}$	$2.237e^{-2} \pm 2.94e^{-5}$ ▼
Waves	$\boxed{1.160e^{-2} \pm 7.30e^{-5}}$	$1.176e^{-2} \pm 8.22e^{-5}$	$2.074e^{-2} \pm 1.11e^{-4}$ ▼

Boxes indicate the best result comparing only the proposed and the previous approach and black triangles indicate the cases where the LS-SVM obtains worse results than the proposed one.

for large-scale data sets for which a batch learning is not possible. Also, it is important to remark that the solution achieved by the method using an on-line or incremental learning is exactly the same obtained in a batch mode.

In addition, Table 4 includes the mean CPU training time (in milliseconds), and the associate standard deviation, for each data set. The time required by the proposed method to obtain the solution is considerably lesser than the LS-SVM method. This is to the fact that it is not an iterative algorithm but it gets the global optimum of the objective function in just a step. Thereby, the proposed method provides an interesting combination of performance and speed.

4.2. Comparative study with other standard algorithms

In this experiment, the performance of the proposed method was checked over 10 different binary classification problems and compared to a representative set of learning methods. Table 5 shows the number of attributes and instances for each data set. The first five data sets were obtained from the UCI Machine Learning Repository [23] and the last five from the Data Mining Institute of the University of Wisconsin [24]. Again, the topology of the network in each problem is determined by the dimension of the input and output data. The same activation functions mentioned in the previous subsection were used but for the same reason only one of them was included, in this case the logarithmic sigmoid function.

The proposed method was compared with the following efficient classifiers from the pattern recognition field:

- The LS-SVM method described in the previous section.
- The linear version of the proximal support vector machine (PSVM) [25]. Although this algorithm is not so well-known as the LS-SVM, its interest is given by the fact that it uses a system of linear equations and exhibits good performance for classification problems.
- The well-known Fisher linear discriminant (FLD) method. In this study the implementation provided by the Statistical Pattern Recognition Toolbox for Matlab [26] was used.

Table 6 shows the mean *training* accuracy (in percentage) and standard deviation obtained for each data set over 50 different simulations. Moreover, the number in brackets is the ranking obtained by the method for each data set together. The last row of the table contains the mean results considering all the data sets. Table 7 shows the same results but for the mean *test* accuracy (true accuracy estimated by cross-validation) and standard deviation obtained for each data set. The best result for each data set is underlined in each row of the table. The last row shows the number of times that the proposed method wins and ties against each of the other algorithms. Additionally, Table 8 contains the CPU training time (in milliseconds) employed by each method for each fold of the 10-fold cross-validation.

Regarding the performance analysis, the following issues can be observed:

- Considering the test accuracy of all the algorithms, it achieves the greatest number of wins (4 of 10 data sets). In an individual comparative analysis it obtains 7 wins and 1 tie in the best case (Fisher) and 4 wins and 2 ties in the worst situation (PSVM). In addition, it achieves the best mean accuracy (86.43%) and the best mean ranking (2.0) for the test sets. These results are only closely similar for the PSVM but this last usually requires considerably more CPU time in order to get the outcome of the learning process. Therefore, the proposed method can be considered as a potential alternative to the regular learning algorithms.
- The Fisher discriminant method obtains bad results in some cases, e.g., Iono, Curie and Forest data sets. This is due to problems induced by ill-conditioned matrices for those data sets.

Table 4
Mean training time (in milliseconds) $\pm$ standard deviation.

Data set	Proposed	Previous	LS-SVM
Henon	21.406 ± 6.425	8.531 ± 4.033	9182.781 ± 52.210
Lorenz	20.969 ± 6.179	9.719 ± 4.706	5766.750 ± 68.715
Mackey	15.469 ± 4.653	6.563 ± 5.051	4115.469 ± 39.118
Ikeda	15.437 ± 5.608	6.500 ± 4.344	4094.219 ± 31.606
Laser	14.000 ± 5.403	5.438 ± 3.644	2883.594 ± 47.643
DowJones	24.750 ± 5.588	10.875 ± 5.310	18387.906 ± 180.961
Annulus	20.094 ± 6.894	9.500 ± 4.605	6478.344 ± 131.236
Kobe	20.906 ± 6.177	8.844 ± 4.856	9408.875 ± 67.977
Imports	3.375 ± 3.099	1.656 ± 2.592	139.000 ± 8.370
Co2	4.781 ± 3.871	2.562 ± 2.579	163.594 ± 8.263
Oscillation	8.594 ± 4.685	3.312 ± 2.972	1587.062 ± 20.166
Blowfly	3.188 ± 2.544	2.063 ± 2.197	158.687 ± 7.962
TreeRings	12.094 ± 7.368	5.063 ± 3.937	2157.219 ± 50.177
Waves	4.344 ± 3.972	1.969 ± 2.599	128.969 ± 7.893
Lmpavw	10.514 ± 2.671	4.680 ± 3.260	23015.296 ± 143.183
Saubts	8.299 ± 3.342	3.058 ± 1.916	7673.752 ± 49.278

Table 5
Characteristics of the classification data sets.

Data set	Attributes	Instances
Wisconsin Diagnostic Breast Cancer	30	569
Pima data	8	768
Housing data	13	506
Iono data	34	351
Forest CoverType	55	5000
Curie data	499	22
Dim data	14	4192
Mush data	22	8124
Musk "Clean2" database	168	6598
Bright data	14	2462

Table 6
Training mean accuracy $\pm$ standard deviation for classification problems using logarithmic sigmoid activation functions.

Data set	Proposed	PSVM	Fisher	LS-SVM
BreastCancer	96.54 ± 0.06 [4]	95.65 ± 0.06 [2]	$\underline{96.85 \pm 0.05}$ [1]	95.58 ± 0.05 [3]
Pimadata	$\underline{77.94 \pm 0.07}$ [1]	77.93 ± 0.08 [2]	76.73 ± 0.09 [4]	77.82 ± 0.08 [3]
Housingdata	86.20 ± 0.13 [3]	$\underline{87.09 \pm 0.16}$ [1]	86.21 ± 0.11 [2]	86.07 ± 0.11 [4]
Ionodata	$\underline{90.73 \pm 0.16}$ [1]	90.28 ± 0.18 [2]	35.90 ± 0.00 [4]	89.69 ± 0.13 [3]
Curiedata	$\underline{100.00 \pm 0.00}$ [1]	$\underline{100.00 \pm 0.00}$ [1]	46.60 ± 16.36 [4]	99.21 ± 1.99 [3]
Dimadata	$\underline{93.50 \pm 0.02}$ [1]	92.60 ± 0.17 [3]	93.47 ± 0.02[2]	92.25 ± 0.01 [4]
Mushdata	80.87 ± 0.10 [3]	80.90 ± 0.10 [2]	62.05 ± 5.42 [4]	$\underline{81.04 \pm 0.07}$ [1]
Musk2	$\underline{94.44 \pm 0.01}$ [1]	94.43 ± 0.01 [3]	94.27 ± 0.02 [4]	$\underline{94.44 \pm 0.01}$ [1]
Brightdata	97.35 ± 0.01 [4]	97.42 ± 0.04 [2]	$\underline{97.84 \pm 0.01}$ [1]	97.42 ± 0.01 [2]
Forest	76.03 ± 0.03 [2]	$\underline{76.05 \pm 0.04}$ [1]	50.54 ± 0.00 [4]	75.87 ± 0.04 [3]
Mean results	89.36 [2.1]	89.24 [1.9]	74.05 [3]	88.94 [2.7]

The number in brackets indicates the ranking of each learning method.

ble 7
Test mean accuracy $\pm$ standard deviation for classification problems using logarithmic sigmoid activation functions.

Data set	Proposed	PSVM	Fisher	LS-SVM
BreastCancer	95.63 $\pm$ 0.19 [2]	95.06 $\pm$ 0.19 [4]	96.07 $\pm$ 0.22 [1]	95.19 $\pm$ 0.24 [3]
Pimadata	77.17 $\pm$ 0.38 [1]	77.13 $\pm$ 0.37 [2]	76.04 $\pm$ 0.36 [4]	77.09 $\pm$ 0.31 [3]
Housingdata	84.81 $\pm$ 0.57 [3]	85.91 $\pm$ 0.54 [1]	84.81 $\pm$ 0.54 [3]	84.88 $\pm$ 0.46 [2]
Ionodata	86.60 $\pm$ 0.56 [2]	86.47 $\pm$ 0.63 [3]	35.91 $\pm$ 0.02 [4]	86.82 $\pm$ 0.58 [1]
Curiedata	79.27 $\pm$ 4.35 [1]	79.27 $\pm$ 4.35 [1]	50.40 $\pm$ 12.47 [4]	70.10 $\pm$ 6.28 [3]
Dimadata	93.38 $\pm$ 0.06 [1]	92.53 $\pm$ 0.23 [3]	93.37 $\pm$ 0.06 [2]	92.21 $\pm$ 0.06 [4]
Mushdata	80.77 $\pm$ 0.12 [3]	80.79 $\pm$ 0.11 [2]	62.02 $\pm$ 5.50 [4]	80.90 $\pm$ 0.12 [1]
Musk2	93.91 $\pm$ 0.06 [1]	93.91 $\pm$ 0.07 [1]	93.66 $\pm$ 0.07 [4]	93.91 $\pm$ 0.06 [1]
Brightdata	97.27 $\pm$ 0.06 [4]	97.33 $\pm$ 0.09 [3]	97.76 $\pm$ 0.06 [1]	97.37 $\pm$ 0.05 [2]
Forest	75.47 $\pm$ 0.10 [2]	75.51 $\pm$ 0.08 [1]	50.54 $\pm$ 0.00 [4]	75.38 $\pm$ 0.13 [3]
Mean results	86.43 [2.0]	86.39 [2.1]	74.06 [3.1]	85.39 [2.3]
Wins/ties of the proposed vs …		4 / 2	7 / 1	5 / 1

The number in brackets indicates the ranking of each learning method.

Table 8
Mean training time (in milliseconds) $\pm$ standard deviation for classification problems.

Data set	Proposed	PSVM	Fisher	LS-SVM
BreastCancer	14.0 $\pm$ 4.7	163.8 $\pm$ 4.8	10.4 $\pm$ 3.7	2295.6 $\pm$ 45.6
Pimadata	6.0 $\pm$ 4.0	167.3 $\pm$ 4.9	3.9 $\pm$ 3.3	1234.7 $\pm$ 21.0
Housingdata	5.4 $\pm$ 3.7	173.8 $\pm$ 5.2	4.4 $\pm$ 3.5	1189.4 $\pm$ 23.0
Ionodata	19.8 $\pm$ 4.9	157.6 $\pm$ 4.6	22.4 $\pm$ 5.7	545.6 $\pm$ 8.9
Curiedata	4684.4 $\pm$ 24.7	137.3 $\pm$ 4.7	425.0 $\pm$ 8.5	21.8 $\pm$ 4.4
Dimadata	27.7 $\pm$ 4.3	188.3 $\pm$ 5.2	21.0 $\pm$ 3.8	28637.2 $\pm$ 206.4
Mushdata	53.6 $\pm$ 5.4	199.8 $\pm$ 5.6	76.4 $\pm$ 8.5	253610.3 $\pm$ 9282.8
Musk2	551.3 $\pm$ 28.9	649.0 $\pm$ 15.3	683.3 $\pm$ 30.2	4567293.2 $\pm$ 28634.8
Brightdata	21.6 $\pm$ 5.5	175.3 $\pm$ 7.4	13.3 $\pm$ 4.9	38398.0 $\pm$ 1028.2
Forest	82.7 $\pm$ 5.8	219.7 $\pm$ 9.0	112.0 $\pm$ 5.7	238498.8 $\pm$ 6027.1

- Concerning the CPU time, the proposed method is usually the fastest method only matched by the Fisher discriminant method. Comparing the performance showed in Tables 6 and 7, it is observed that in general the proposed method obtains better results than the FLD in similar CPU times.
- Compared with the LS-SVM. The proposed method presents a similar performance to this approach but the training times are much lesser. Of special interest is the result obtained for the Musk2 data set, that contains a large number of examples and attributes, for which the proposed method obtains a mean performance equal to the LS-SVM, but using less than 1 s of CPU time, while LS-SVM needs more than 1 h. Thereby, the proposed method provides an interesting combination of performance and speed.

5. Conclusions

Current methods for learning the weights in single-layer neural networks have the following shortcomings:

- The objective function to be optimized can have several (usually many) local optima. This implies that the users cannot know whether or not they are in the presence of a global optimum, and how far from it the local optimum resulting from the learning process is.
- The learning process requires a large amount of computational time for large data sets and networks.

In this work a new supervised learning method for single-layer feedforward neural networks was proposed. This approach is based on a new objective function that measures the errors before the nonlinear function instead of after this function, as usual, and scales the errors according to the corresponding operation point of the nonlinearity. This leads to a convex objective function which global optimum can be easily obtained using a square system of linear equations. It was experimentally verified that this method can improve, in many cases, the performance of the previous approaches that do not scale the errors measured before the activation functions.

The main features of the proposed learning method are:

- The global optimum obtained is approximately equivalent to the one of the MSE measured after the activation functions. This equivalence was proved employing a first order Taylor series approximation. In general, for more accurate results one may want to use more terms in the Taylor series expansion. However, this brings in the higher order moments of the error, which makes impossible the use of a system of linear equations. Anyway, as it was verified in the experimental simulations, the proposed approach obtains a close approximation to the global optimum of the regular MSE. If for any reason, it would be necessary to get a more exact solution, the weights provided by the proposed algorithm could be used as an initial state of any standard learning algorithm, for instance, the Scaled Conjugate Gradient or the Levenberg–Marquart method.
- The employed alternative objective function based on the MSE, measured before the nonlinear activation functions, is strictly convex and thus the absence of local minima is ensured.
- It allows an incremental and distributed learning.
- It exhibits a fast operation, with a complexity of $O(N^2)$ being N the number of weights of the network.

Acknowledgments

This work was supported in part by the Spanish Ministry of Education and Science, Grant code TIN2006-02402 (partially supported by FEDER funds), and by the Xunta de Galicia, Grant code 08TIC012105PR.

References

[1] C.M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, New York, 1995.
[2] Y.H. Pao, Adaptive Pattern Recognition and Neural Networks, Addison-Wesley Publishing Company, Inc., Reading, MA, 1989.
[3] B. Widrow, M.A. Lehr, 30 years of adaptive neural networks: perceptron, madeline and backpropagation, in: Proceedings of the IEEE, vol. 78, 1990, pp. 1415–1442.

M. Brady, R. Raghavan, J. Slawny, Back propagation fails to separate where perceptrons succeed, IEEE Transactions on Circuits and Systems 36 (1989) 665–674.

[5] E. Sontag, H.J. Sussmann, Backpropagation can give rise to spurious local minima even for networks without hidden layers, Complex Systems 3 (1989) 91–106.

[6] M. Budinich, E. Milotti, Geometrical interpretation of the back-propagation algorithm for the perceptron, Physica A 185 (1992) 369–377.

[7] P. Auer, M. Hebster, M.K. Warmuth, Exponentially many local minima for single neurons, in: D.S. Touretzky, M.C. Mozer, M.E. Hasselmo (Eds.), Advances in Neural Information Processing Systems, vol. 8, The MIT Press, Cambridge, MA, 1996, pp. 316–322.

[8] E. Sontag, H.J. Sussmann, Back propagation separates where perceptrons do, Neural Networks 4 (1991) 243–249.

[9] M. Gori, A. Tesi, On the problem of local minima in backpropagation, IEEE Transactions on Pattern Analysis and Machine Intelligence 14 (1) (1992) 76–85.

[10] F.M. Coetzee, V.L. Stonick, On a natural homotopy between linear and nonlinear single-layer networks, IEEE Transactions on Neural Networks 7 (2) (1996) 307–317.

[11] F.M. Coetzee, V.L. Stonick, On the uniqueness of weights in single layer perceptrons, IEEE Transactions on Neural Networks 7 (2) (1996) 318–325.

[12] J. Kivinen, M.K. Warmuth, Relative loss bounds for multidimensional regression problems, Machine Learning 45 (2001) 301–329.

[13] R.S. Scalero, N. Tepedelenlioglu, A fast new algorithm for training feedforward neural networks, IEEE Transactions on Signal Processing 40 (1) (1992) 202–210.

[14] F. Biegler-Konig, F. Barnmann, A learning algorithm for multilayered neural networks based on linear least squares problems, Neural Networks 6 (1993) 127–131.

[15] M. Di Martino, S. Fanelli, M. Protasi, Exploring and comparing the best "direct methods" for the efficient training of MLP-networks, IEEE Transactions on Neural Networks 7 (6) (1996) 1497–1502.

[16] Y. Yam, T. Chow, A new method in determining the initial weights feedforward neural networks, Neurocomputing 16 (1) (1997) 23–32.

[17] S.-Y. Cho, Z. Chi, W.-C. Siu, A.C. Tsoi, An improved algorithm for learning long-term dependency problems in adaptive processing of data structures, IEEE Transactions on Neural Networks 14 (4) (2003) 781–793.

[18] E. Castillo, O. Fontenla-Romero, A. Alonso-Betanzos, B. Guijarro-Berdiñas, A global optimum approach for one-layer neural networks, Neural Computation 14 (6) (2002) 1429–1449.

[19] G. Carayannis, N. Kalouptsidis, D. Manolakis, Fast recursive algorithms for a class of linear equations, IEEE Transactions on Acoustics, Speech and Signal Processing ASSP-30 (2) (1982) 227–239.

[20] A. Bojanczyk, Complexity of solving linear systems in different models of computation, SIAM Journal on Numerical Analysis 21 (3) (1984) 591–603.

[21] M. Minsky, S. Papert, Perceptrons: An Introduction to Computational Geometry, MIT Press, Cambridge, MA, 1969.

[22] J. Suykens, T.V. Gestel, J.D. Brabanter, B.D. Moor, J. Vandewalle, Least Squares Support Vector Machines, World Scientific, Singapore, 2002.

[23] D. Newman, S. Hettich, C. Blake, C. Merz, UCI repository of machine learning databases 〈http://www.ics.uci.edu/∼mlearn/MLRepository.html〉, 1998.

[24] O.L. Mangasarian, R. Ramakrishnan, Data Mining Institute, Computer Sciences Department, University of Wisconsin 〈http://www.cs.wisc.edu/dmi〉, 1999.

[25] G. Fung, O.L. Mangasarian, Proximal support vector machine classifiers, in: F. Provost, R. Srikant (Eds.), Proceedings KDD-2001: Knowledge Discovery and Data Mining, August 26–29, 2001, San Francisco, CA, Association for Computing Machinery, New York, 2001, pp. 77–86.

[26] V. Franc, V. Hlaváč, A new feature of the statistical pattern recognition toolbox, in: S. Scherer (Ed.), Computer Vision, Computer Graphics and Photogrammetry—A Common Viewpoint, Österreichische Computer Gesellschaft, 2001, pp. 143–150 〈http://cmp.felk.cvut.cz/cmp/software/stprtool/index.html〉.

About the Author—OSCAR FONTENLA-ROMERO was born in Ferrol, Spain, in 1974. He received his B.S., M.S. and Ph.D. degrees in computer science from the University of A Coruña, in 1997, 1998, and 2002, respectively. He works as an Associate Professor at the Department of Computer Science of University of A Coruña since 2004. His current research interests include new linear learning methods and noise immunity for neural networks and functional networks. His main current areas are neural networks, functional networks and nonlinear optimization.

About the Author—BERTHA GUIJARRO-BERDIÑAS was born in A Coruña, Spain, in 1969. She received her B.S., M.S. and Ph.D. degrees in computer science from the University of A Coruña, in 1990, 1992, and 1998, respectively. She works as an Associate Professor at the Department of Computer Science of University of A Coruña since 1994. Her current research interests include theoretical works on neural networks, knowledge-intensive systems, hybrid systems and intelligent agents with applications to medicine, intrusion detection and fire forest management and prediction.

About the Author—BEATRIZ PÉREZ-SÁNCHEZ was born in A Coruña, Spain, in 1980. She received her M.S. degree in computer science from the University of A Coruña, in 2005. She works as an Assistant Lecturer at the Department of Computer Science of University of A Coruña since 2007. Her current research interests include theoretical works on neural networks and new linear learning methods and learning optimization.

About the Author—AMPARO ALONSO-BETANZOS (M'88) was born in Vigo, Spain, in 1961. She graduated with the degree in chemical engineering from the University of Santiago, Spain, in 1984. In 1985 she joined the Department of Applied Physics, Santiago de Compostela, Spain, where she received the M.S. degree for work in monitoring and control of biomedical signals. In 1988, she received the Ph.D. (cum laude and premio extraordinario) degree for work in the area of medical expert systems. From 1988 through 1990, she was a postdoctoral fellow in the Department of Biomedical Engineering Research, Medical College of Georgia, Augusta. She is currently a Full Professor in the Department of Computer Science, University of A Coruña. Her main current areas are hybrid intelligent systems, intelligent multi-agent systems, linear optimization methods and entropy based cost functions for neural networks and functional networks. Dr. Alonso-Betanzos is a member of various scientific societies, including the ACM and IEEE.