Expert Systems with Applications 36 (2009) 7659–7665

Contents lists available at ScienceDirect

Expert Systems with Applications

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

Differential evolution trained wavelet neural networks: Application to bankruptcy prediction in banks

Nikunj Chauhan, V. Ravi *, D. Karthik Chandra

Institute for Development and Research in Banking Technology, Castle Hills Road #1, Masab Tank, Hyderabad 500057, AP, India

ARTICLE INFO

Keywords:
Wavelet neural networks (WNN)
Differential evolution (DE)
Bankruptcy prediction
Classification
Differential evolution trained wavelet neural network (DEWNN)
Threshold accepting trained wavelet neural network (TAWNN)

ABSTRACT

In this study, differential evolution algorithm (DE) is proposed to train a wavelet neural network (WNN). The resulting network is named as differential evolution trained wavelet neural network (DEWNN). The efficacy of DEWNN is tested on bankruptcy prediction datasets viz. US banks, Turkish banks and Spanish banks. Further, its efficacy is also tested on benchmark datasets such as Iris, Wine and Wisconsin Breast Cancer. Moreover, Garson's algorithm for feature selection in multi layer perceptron is adapted in the case of DEWNN. The performance of DEWNN is compared with that of threshold accepting trained wavelet neural network (TAWNN) [Vinay Kumar, K., Ravi, V., Mahil Carr, & Raj Kiran, N. (2008). Software cost estimation using wavelet neural networks. *Journal of Systems and Software*] and the original wavelet neural network (WNN) in the case of all data sets without feature selection and also in the case of four data sets where feature selection was performed. The whole experimentation is conducted using 10-fold cross validation method. Results show that soft computing hybrids viz., DEWNN and TAWNN outperformed the original WNN in terms of accuracy and sensitivity across all problems. Furthermore, DEWNN outscored TAWNN in terms of accuracy and sensitivity across all problems except Turkish banks dataset.

1. Introduction

The word *wavelet* is due to Grossmann & Morlet, 1984. Wavelets are a class of functions used to localize a given function in both space and scaling (http://mathworld.wolfram.com/wavelet.html). They have advantages over traditional Fourier methods in analyzing physical situations where the signal contains discontinuities and sharp spikes. Wavelets were developed independently in the fields of mathematics, quantum physics, electrical engineering and seismic geology. Interchanges between these fields during the last few years have led to many new wavelet applications such as image compression, turbulence, human vision, radar, chemistry and earthquake prediction.

Taking cue from the locally supported basis functions such as Radial basis function networks (RBFN), a class of neural networks called WNN, which originate from wavelet decomposition in signal processing, have become more popular recently (Zhang, 1997). Wavelet networks employ activation functions that are dilated and translated versions of a single function ψ: $R^d \rightarrow R$, where d is the input dimension as stated in Zhang and Benvniste (1992) and Zhang (1997). This function called the 'mother wavelet' is localized both in the space and frequency domains (Becerra, Galvao, & Abou-Seads, 2005). Based on wavelet theory, the WNN was proposed as a universal tool for functional approximation, which shows surprising effectiveness in solving the conventional problem of poor convergence or even divergence encountered in other kinds of neural networks. It can dramatically increase convergence speed compared to other networks as stated in Zhang et al. (2001).

Since the rapid increase in popularity of WNN researchers around the globe started working more with this robust architecture. Zhang et al. (2001) used WNN to predict programmed-temperature retention values of naphthas. Yu and Chen (2007) used WNN to develop an ECG beat classification system. In this work, WNN is used to discriminate six different beat types in ECG. Avci (2007) developed an expert system based on WNN for texture classification. Dong, Xiao, Liang, and Liu (2008) used fuzzy WNN and rough sets for predicting fault diagnosis accuracy of power transformers. Lung (2007) used WNN for feature selection and recognition of text independent speaker. Here, a wavelet packet feature selection derived by using multilayered neural network for speaker identification is described. Reza and Ghorbani (2007) applied WNN in optimization of skeletal buildings under frequency constraints. The goal was to reduce computational burden for optimum design of steel frames with frequency constraints by approximating frequencies using WNN. A rational function with second order poles (RASP) wavelet was used as a transfer function. Raj Kiran and Ravi (2007) used WNN for software reliability prediction. Dimoulas, Kalliris, Papanikolaou, Petridis, and Kalampakas (2008) used WNN for analysis of bowel sound pattern. Here bowel sounds were

* Corresponding author. Tel.: +91 40 2353 4981x2042; fax: +91 40 2353 5157.
E-mail addresses: vravi@idrbt.ac.in, rav_padma@yahoo.com (V. Ravi).

0957-4174/$ - see front matter © 2008 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2008.09.019

differentiated form other sounds with classification accuracy of 94.84% using WNN. The above-mentioned diverse applications indicate the versatility and increasing popularity of WNN.

WNN uses a gradient descent technique for training. Traditional gradient descent method suffers from well known drawbacks such as entrapment in local minimum, long convergence times and the need of differentiability of the objective function that are associated with calculus based optimization techniques. Consequently, WNN also suffers from these disadvantages. Therefore, Yu, Li, Bai, and Jin (2007) proposed the use of improved chaotic particle swarm optimization (ICPSO) and improved particle swarm optimization (IPSO) to tune both the structure and parameters of the WNN. Recently, Pan, Chen, and Yun (2008) used genetic algorithm to optimize the WNN. Most recently, Vinay Kumar et al. (2008) proposed TAWNN for estimating software development cost. They compared the effectiveness of TAWNN with other techniques such as WNN, multilayer perceptron (MLP), radial basis function network (RBFN), multiple linear regression (MLR), dynamic evolving neuro-fuzzy inference system (DENFIS) and support vector machine (SVM) in terms of mean magnitude relative error (MMRE) obtained on Canadian financial (CF) dataset and IBM data processing services (IBMDPS) dataset. They found that TAWNN outperformed all other techniques except WNN.

Incidentally, Ilonen, Kamarainen, and Lampinen (2003) were the first to apply DE to train feed forward neural network. They used weights as solution vectors and network prediction error as the objective function. However, to the best our knowledge, there is no reported work that employs DE for the training of WNN. In this connection it should be noted that Bhat, Venkataramani, Ravi, and Murty (2006) developed an improved version of DE and called it 'improved differential evolution' (IDE) for efficient parameter estimation in biofilter modeling. The parameter estimation problem is an unconstrained optimization problem. It should be noted that training of a neural network and WNN, in particular, is also an unconstrained optimization problem. Hence, we propose a DE based training algorithm for WNN and call the resulting network as DEWNN.

The remainder of this paper is organized as follows. In Section 2 we describe the WNN. Section 3 describes in brief the metaheuristics used to train WNN. In Section 4, the training of WNN with differential evolution (DEWNN) is clearly explained. In Section 5 we discuss the feature selection module adopted from Garson and later incorporated into DEWNN and TAWNN. Section 6 deals with the area of application of the current research i.e. bankruptcy prediction. Section 7 provides the results and discussions of the work and finally Section 8 contains concluding remarks.

2. Wavelet neural networks

A family of wavelets can be constructed from a function $\psi(x)$, sometimes known as a "mother wavelet," which is confined in a finite interval. "Daughter wavelets" $\psi^{a,b}(x)$ are then formed by translation (b) and dilation (a). Wavelets are especially useful for compressing image data, since a wavelet transform has properties that are in some ways superior to a conventional Fourier transform (http://mathworld.wolfram.com/wavelet.html).

An individual wavelet is defined by

$$\psi^{a,b}(x) = |\alpha|^{-1/2}\psi\left(\frac{x-b}{a}\right) \tag{1}$$

Industrial processes can impose a number of problems upon the structures adopted for neural network dynamic modeling due to varying sampling times, sparse and dense data in different operating regions and the inherent presence of both large and small dynamics. In the case of non-uniformly distributed training data, an efficient way of solving this problem is by learning at multiple resolutions. A higher resolution of input space is used if the data is dense and a lower resolution when it is sparse.

Wavelets, in addition to forming an orthogonal basis, are capable of explicitly representing the behavior of a function at various resolutions of input variables. Consequently, a wavelet network is first trained to learn the mapping at the coarsest resolution level. In subsequent stages, the network is trained to incorporate elements of the mapping at higher and higher resolutions. Such hierarchical, multi resolution training has many attractive features for solving engineering problems, resulting in a more meaningful interpretation of the resulting mapping and more efficient training and adaptation of the network compared to conventional methods. The wavelet theory provides useful guidelines for the construction and initialization of networks and consequently, the training times are significantly reduced (http://www.ncl.ac.uk/pat/neural-networks.html).The structure of the WNN with two input and six hidden nodes is shown in Fig. 1.

The WNN consists of three layers: input layer, hidden layer and output layer. All the units in each layer are fully connected to the nodes in the next layer. The output layer contains a single unit. WNN is implemented here with the Gaussian wavelet function.

The training algorithm (Zhang et al. (2001)) for a WNN is as follows:

1. Select the number of hidden nodes required. Initialize the dilation and translation parameters for these nodes to some random values. Also initialize the weights for the connections between the input and hidden layer and also for the connections between the hidden and the output layer. It should be taken note that the random values should be limited to the interval $(0,1)$.
2. The output of the sample V_k, k=1,...,np, where np is the number of samples, is calculated with the following formula:

$$V_k = \sum_{j=1}^{nhn} W_j f\left(\frac{\sum_{i=1}^{nin} W_{ij}x_{ki} - b_j}{a_j}\right) \tag{2}$$

where k = 1,...,np, nin = number of input nodes and nhn = number of hidden nodes.In (2) when $f(t)$ is taken as a Morlet mother wavelet it has the following form:

$$f(t) = \cos(1.75t)\exp(-t^2/2) \tag{3}$$

and when taken as Gaussian Wavelet it becomes:

$$f(t) = \exp(-t^2) \tag{4}$$

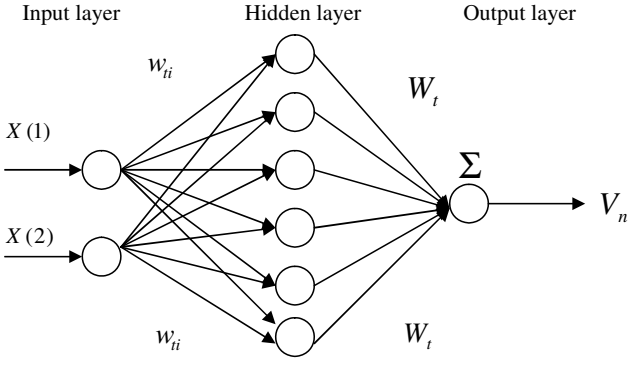


Fig. 1. Wavelet neural network.

3. Reduce the error of prediction by adjusting W_j, w_{ij}, a_j, b_j using ΔW_j, Δw_{ij}, Δa_j, Δb_j (see formulas (5)–(8)). In the WNN, the gradient descend algorithm is employed.

$$\Delta W_j(t+1) = -\eta \frac{\partial E}{\partial W_j(t)} + \alpha \Delta W_j(t) \tag{5}$$

$$\Delta w_{ij}(t+1) = -\eta \frac{\partial E}{\partial w_{ij}(t)} + \alpha \Delta w_{ij}(t) \tag{6}$$

$$\Delta a_j(t+1) = -\eta \frac{\partial E}{\partial a_j(t)} + \alpha \Delta a_j(t) \tag{7}$$

$$\Delta b_j(t+1) = -\eta \frac{\partial E}{\partial b_j(t)} + \alpha \Delta b_j(t) \tag{8}$$

where the error function E is taken as normalized root mean squared deviation (NRMSE) as follows:

$$E = \sqrt{\sum_{k=1}^{np} \frac{(V_k - \hat{V}_k)^2}{V_k}} \tag{9}$$

where η and α are the learning and the momentum rates respectively.

4. Return to step (2), the process is continued until E satisfies the given error criteria, and the whole training of the WNN is completed.

Some problems exist in WNN such as slow convergence, searching space tapping in local minima and oscillation (Pan et al., 2008). We propose DEWNN to resolve these problems.

3. Meta heuristics used to train WNN

3.1. Differential evolution

Differential evolution is a novel approach in evolutionary algorithms. It was proposed by Storn and Price (1997). It is a stochastic, population-based optimization method. Differential evolution algorithm consists mainly of four steps: *initialization, mutation, recombination and selection.* DE differs from other population-based techniques in that it employs differential mutation.

In a population of solutions within an n-dimensional search space, a fixed number of vectors are randomly initialized, then evolved over time to explore the search space and to locate the minima of the objective function. Inside a generation, new vectors are generated by the combination of vectors randomly chosen from the current population (mutation). The vectors so generated are then mixed with a predetermined target vector. This operation is called recombination and produces the trial vector. Finally, the trial vector is accepted for the next generation if and only if it yields a reduction in the value of the objective function. This last operator is referred to as selection. Appendix A depicts the flowchart of the differential evolution algorithm.

3.2. Threshold accepting trained WNN (TAWNN)

Threshold Accepting algorithm, originally proposed by Dueck and Scheuer (1990) is a faster variant of the original simulated annealing algorithm wherein the acceptance of a new move or solution is determined by a deterministic criterion rather than a probabilistic one. Vinay Kumar et al. (2008) proposed TAWNN. In a sense, this work forms the basis for the present paper, where DE replaces TA in training the WNN. For further details on TAWNN, the reader is referred to Vinay Kumar et al. (2008).

4. Differential evolution based wavelet neural network (DEWNN)

Application of DE in training WNN basically modifies steps 3 and 4 of the WNN training algorithm for WNN described in Section 2. Output of a WNN is a function of weights $\boldsymbol{W}$ (weights from input layer to hidden layer), $\boldsymbol{w}$ (weights from hidden layer to output layer), dilation parameters $\boldsymbol{D}$, translation parameters $\boldsymbol{T}$ and input values $\boldsymbol{X}$, i.e. $\boldsymbol{Y} = f(\boldsymbol{X},\boldsymbol{R})$, where $\boldsymbol{Y}$ is the output values vector and $\boldsymbol{R} = (\boldsymbol{D},\boldsymbol{T},\boldsymbol{W},\boldsymbol{w})$. During training phase, both the input vector $\boldsymbol{X}$ and output vector $\boldsymbol{Y}$ are known and synaptic weights $\boldsymbol{W}$ and $\boldsymbol{w}$, dilation parameters $\boldsymbol{D}$ and translation parameters $\boldsymbol{T}$ are predicted and adapted by minimizing network error E to obtain proper relationship from $\boldsymbol{X}$ to $\boldsymbol{Y}$. In DEWNN, the elements involved in the vectors $\boldsymbol{D},\boldsymbol{T},\boldsymbol{W}$ and $\boldsymbol{w}$ are the decision variables.

Vector $\boldsymbol{R}$ consists of

(i) Weight values from input nodes to hidden nodes $\boldsymbol{W} = \{W_{ij}, i = 1,2,\ldots,\text{nin}$, where nin = number of input nodes, $j = 1,2,\ldots,\text{nhn}$, where nhn = number of hidden nodes$\}$
(ii) Weight values from hidden nodes to output nodes $\boldsymbol{w} = \{w_{jk}, j = 1,2,\ldots,nhn$ and $k = 1,2,\ldots,\text{non}$, where non = number of output nodes$\}$
(iii) Dilation parameters $\boldsymbol{D} = (d_1, d_2, \ldots, d_{nhn})$
(iv) Translation parameter $\boldsymbol{T} = (t_1, t_2, \ldots, t_{nhn})$

A population P in each generation consists of M such $\boldsymbol{R}$ vectors where M is the size of population as below:

$$P = (R_1, \ldots, R_M) \tag{10}$$

The initial population is randomly initialized using the user specified lower and upper bounds for weights, dilation and translation parameters as follows:

$$R_i = R_{i\min} + \text{rand}(0,1) * (R_{i\max} - R_{i\min}) \tag{11}$$

Mutation is basically a search mechanism, which, together with recombination and selection, directs the search towards potential areas of optimal solution. In this step, three distinct target vectors R_a, R_b and R_c are randomly chosen from the M vectors of the parent population on the basis of three random numbers a, b and c. One of the vectors R_c is the base of the mutated vector. To this is added the weighted difference of the remaining two vectors, i.e. $(R_a - R_b)$ to generate a noisy random vector, n_i.

$$n_i = R_c + F * (R_a - R_b) \tag{12}$$

where $i = 1,2,\ldots,M$ and a,b,c are random numbers between 1 and M.

F is termed the scaling factor and it is user-supplied. This mutation process is repeated to create a mate for each member of the parent population. Mutation ensures an efficient search of the solution space in each dimension.

In the recombination (crossover) operation, each target vector of the parent population is allowed to mate with a mutated vector. Thus, vector R_i is recombined with the noisy random vector n_i to generate a trial vector t_i. Each element of the trial vector (t_i^j,where $i = 1,\ldots,M$ and $j = 1,\ldots,n$), is determined by a binomial experiment whose success or failure is determined by the user-supplied crossover factor, CR. The parameter CR is used to control the rate at which the crossover takes place.

$$\begin{aligned} t_i^j &= n_i^j \text{ if } \text{rand}(0,1) < CR \text{ or } j = \text{rand}(1,n) \\ &= R_i^j \text{ otherwise} \\ &\text{where } i = 1,2,\ldots,M \text{ and } j = 1,2,\ldots,n \end{aligned} \tag{13}$$

Therefore, trial vector t_i is the child of two parent vectors: noisy random vector n_i and the target vector R_i. DE performs a non-uniform crossover, that determines which trial vector parameters are inherited from which parent.

It is sometimes possible that some particular combinations of three target vectors from the parent population and the scaling factor F would result in noisy vector values, which are outside the bounds set for the decision variables. It is necessary, therefore, to bring such values within the bounds. For this reason, the value of each element of the trial vector is checked at the end of the recombination step. If it violates the bounds, it is heuristically brought back to lie within the bounded region.

It is in the last stage of 'selection' that fitter of the two vectors (trial vector and target vector) survives and proceeds to the next generation. The vector having minimum value of objective function goes to next generation. This procedure is similar to the 'tournament selection' (Deb, 2000). Based on Gaussian wavelet function, network error, NRMSE is calculated for both the trial vector and target vector. The vector giving minimum error goes to the next generation.

After M competitions of this kind in each generation, one will have a new population, which is fitter than the population one started with. This evolution procedure is repeated over several generations until the termination condition is reached, i.e. when the objective function varies by a specified tolerance limit in two consecutive generations or a maximum number of generations is completed, whichever happens earlier. Eventually, we get the final population consisting of vector sets of weights, dilation and translation parameters. Out of these, we choose the best set as the optimal set of decision variables, using which we test the performance of WNN on test data.

5. Feature selection

Feature selection is a process by which a sample in the measurement space is described by a finite and usually smaller set of number classed features. The features become components of the pattern space. Feature selection is regarded as a procedure to determine which variables (attributes) are to be measured either first or last. Guyon and Elisseeff (2003) indicated that there are many potential benefits of feature selection: facilitating data visualization and data understanding, reducing the measurement and storage requirements, reducing training and utilization times, and defying the curse of dimensionality to improve prediction performance.

Nath, Rajagopalan, and Ryker (1997) applied the Garson (1991) algorithm for feature selection while training MLP. We adopted the same algorithm and applied it in the context of WNN. Garson's algorithm for feature selection is presented as follows.

For a two-group classification, consider a neural network with q input nodes, r hidden nodes and one output node. Let w_{ij} ($i = 1,2,\ldots,q$; $j = 1,2,\ldots,r$) represent the weight of the connection from ith input node to jth hidden node. Let w_{ko} ($k = 1,2,\ldots,r$) be the weight of the connection from kth hidden node to the output node.

The method to measure the importance of an input variable is to partition the hidden-to-output connection weights of each hidden node into components associated with each input node. The resulting weight associated with each input is a reflection of its importance.

- Each hidden-to-output weight w_{ko}, irrespective of its sign, is incorporated into the input to- hidden weights W_{ij} using the following expression:

$$W^*_{ij} = \left(|W_{ij}|/S_j\right) * (|w_{ko}|) \tag{14}$$

where $S_j = \sum_{i=1}^{p} |W_{ij}|$, $i = 1,\ldots,q$ and $j = 1,\ldots,r$. $|\ |$ represents the absolute value.

- For each hidden node j, the sum of weights over all input nodes is equal to the hidden-to-output node weight w_{jo}.

$$w_{jo} = \sum_{i=1}^{q} w^q_{ij} \tag{15}$$

- For each input node, the adjusted weights W^*_{ij} are summed over all hidden nodes and converted to a percentage of the total for all input nodes.

$$r_i = (W^*_{ij}/w_{jo}) * 100 \tag{16}$$

This percentage serves as a measure of the importance of the variable represented by the input node.

The features are thus ranked according to their importance and the top six features in each fold are considered. Once we select the top six features in ten folds, we calculate the frequency of occurrence of each feature across all folds and the features with highest frequency of occurrence are considered as important ones. Thus, the six most important features are selected and an optimal feature subset is formed.

6. Bankruptcy prediction

The prediction of bankruptcy for financial firms especially banks has been the extensively researched area since late 1960s by Altman (1968). Creditors, auditors, stockholders and senior management are all interested in bankruptcy prediction because it affects all of them in the same way. The banks are mostly monitored by regulators who conduct on-site examinations on banks' premises every 12–18 months, as stipulated by the Federal Deposit Insurance Corporation Improvement Act of 1991. Regulators indicate the safety and soundness of the institution using a six part rating system. This rating, referred to as the CAMELS rating, evaluates banks according to their basic functional areas: ***C**apital adequacy, **A**sset quality, **M**anagement expertise, **E**arnings strength, **L**iquidity,* and ***S**ensitivity to market risk*. While CAMELS ratings clearly provide regulators with important information, Cole and Gunther (1995) reported that these CAMELS ratings decay rapidly.

Many statistical techniques such as regression analysis, logistic regression etc. have been used to solve the problem of bankruptcy prediction. These techniques make use of the company's financial data to predict its financial state. Bankruptcy prediction problem can also be solved using various other types of classifiers such as case based reasoning (Jo, Han, & Lee, 1997), rough sets (McKee, 2000), support vector machines (Min & Lee 2005) and data envelopment analysis (Cielen, Peeters, & Vanhoof, 2004) to mention a few. Recently, Ravi Kumar, and Ravi (2006a) proposed a fuzzy rule based classifier for bankruptcy prediction. They reported that fuzzy rule based classifier outperformed the well known technique, BPNN in the case of US banks data. Cheng, Chen, & Fu, 2006 combined RBF network with logit analysis learning to predict financial distress. They compared the proposed technique with logit analysis and a backpropagation neural network and found that their method is superior to both the techniques. Ravi Kumar and Ravi (2006b) proposed an ensemble classifier using simple majority voting scheme for the bankruptcy prediction problem based on a host of intelligent techniques such as ANFIS, SVM, RBF, SORBF1, SORBF2, Orthogonal RBF and BPNN. They reported that, ANFIS, SORBF2, BPNN are the most prominent as they appeared in the best ensemble classifier combinations. Ravi, Ravi Kumar, Ravi Srinivas, and Kasabov (2007) proposed a semi-online training algorithm for the radial basis function neural networks (SORBF) and applied it to bankruptcy prediction in banks. Semi Online RBFN without linear terms performed better than techniques such as ANFIS, SVM, BPNN, RBF and Orthogonal RBF. In another work, Ravi Kumar and

Ravi (2007) conducted a comprehensive review of all the works reported using statistical and intelligent techniques to solve the problem of bankruptcy prediction in banks and firms during 1968–2005. It compares the techniques in terms of prediction accuracy, data sources, time line of each study wherever available. Recently, Pramodh and Ravi (2007) employed modified great deluge algorithm to train an auto associative neural network and applied it to bankruptcy prediction. Further, Ravi, Kurniawan, Peter Nwee Kok Thai, & Ravi Kumar, 2008 developed a novel soft computing system for bank performance prediction based on BPNN, RBF, CART, PNN, FRBC and PCA based hybrid techniques.

Most recently, to solve bankruptcy prediction problems, Ravi and Pramodh (2008) proposed a threshold accepting based training algorithm for a novel principal component neural network (PCNN), without a formal hidden layer. They employed PCNN for bankruptcy prediction problems and reported that PCNN outperformed BPNN, TANN, PCA-BPNN and PCA-TANN in terms of area under receiver operating characteristic curve (AUC) criterion. In BPNN and TANN, there is a hidden layer present and in PCA-BPNN and PCA-TANN, PCA is used as preprocessor to BPNN and TANN respectively.

7. Results and discussion

The datasets analyzed by us in this work are three different banks data sets viz. Turkish Banks, Spanish Banks and US Banks datasets and three other benchmark datasets viz., Iris data, wine data and Wisconsin breast cancer data. Turkish banks' dataset is obtained from Canbas, Caubak, & Kilic, 2005 and is available at (http://www.tbb.org.tr/english/bulten/yillik/2000/ratios.xls). Banks association of Turkey published 49 financial ratios. Initially, Canbas applied univariate analysis of variance (ANOVA) test on these 49 ratios of previous year for predicting the health of the bank in present year. However, Canbas et al. (2005) chose only 12 ratios as the early warning indicators that have the discriminating ability (i.e. significant level is <5%) for healthy and failed banks one year in advance. Among these variables, 12th variable has some missing values meaning that the data for some of the banks are not given. So, we filled those missing values with the mean value of the variable following the general approach in data mining. The financial ratios, which are considered as predictor variables are presented at the end of the paper in Table 1. This dataset contains 40 banks where 22 banks went bankrupt and 18 banks are healthy. The Spanish banks' data is obtained from Olmeda and Fernandez (1997). Spanish banking industry suffered the worst crisis during 1977–1985 resulting in a total cost of 12 billion dollars. The considered financial ratios are presented in the end of the paper in Table 1. The ratios used for the failed banks were taken from the last financial statements before the bankruptcy was declared and the data of non-failed banks was taken from 1982 statements. This dataset contains 66 banks where 37 went bankrupt and 29 healthy banks. The US banks' data is obtained from Olemda and Fernandez, 1996 the financial ratios used by them are presented in Table 1. They obtained the data of 129 banks from the Moody's Industrial Manual, where banks went bankrupt during 1975–1982. This 129 US banks dataset contains 65 went bankrupt and 64 healthy banks. Again, the financial ratios used by them are presented in Table 1. The benchmark datasets are taken from UCI repository (http://archive.ics.uci.edu/ml).

The parameters used for WNN were number of hidden nodes, learning rate and momentum rate. Learning rate is taken between 0.1 and 0.9. Momentum rate is taken between 0.01 and 0.2. Parameters used for TAWNN were number of hidden nodes, number of global iterations, number of local iterations, epsilon, pindex, temtol, threshold value and accuracy. Threshold is taken as between 2 and 3.5. The parameter pindex is an odd number between 3 and 21. The parameter epsilon is taken as 10^{-12}. Accuracy is taken in the range 10^{-13} and 10^{-15}. The parameter temtol is taken between 0.01 and 0.03. Local iterations are taken between 150 and 500. Global iterations are taken between 2000 and 8000. Parameters used for DEWNN are number of hidden nodes, scaling factor *F*, crossover factor CR, population size, maximum allowed error and number of generations. *F* is taken in the range 0.5 to 0.9. CR is taken in the range 0.4 to 0.9. Population size is generally around number of parameters divided by two where number of parameters n is calculated as

$$n = (nin + non + 2) * nhn \quad (17)$$

Number of hidden nodes is taken in the range of 3–15 depending on number of input nodes for all the three algorithms.

All the datasets are analyzed with WNN, TAWNN and DEWNN using 10-fold cross validation. The average accuracies over all the folds are computed for the six datasets. The average sensitivities and specificities are computed for datasets with two class problems. The results for bankruptcy datasets are presented in Table 2. It is observed that DEWNN outperformed WNN and TAWNN in terms of accuracies and sensitivities in the case of all datasets except Turkish data. It is also observed that DEWNN is robust in to variations in parameters. In other words, parameter tuning is very simple as compared to the original WNN and TAWNN. Further, to support the claims made in the paper, all the three algorithms are also tested on some benchmark datasets. The results are presented in Table 3. It can be inferred form the empirical experiments conducted that DEWNN surpassed the original WNN and TAWNN.

Then feature subset is extracted from Turkish, Spanish, Wisconsin Breast Cancer and Wine datasets by using Garson's algorithm. Six top features are extracted form Turkish, Spanish and Wisconsin breast cancer datasets where as seven most important features are extracted from wine dataset. The selected features for DEWNN are

Table 1
Financial ratios of the datasets and the selected features

S. No.	Predictor variable name
Turkish banks' data	
1	Interest expenses/average profitable assets
2	Interest expenses/average non-profitable assets
3*	(Share holders' equity + total income)/(deposits + non-deposit funds)
4	Interest income/interest expenses
5*	(Share holders' equity + total income)/total assets
6	(Share holders' equity + total income)/(total assets + contingencies & commitments)
7*	Networking capital/total assets
8	(Salary and employees' benefits + reserve for retirement)/no. of personnel
9*	Liquid assets/(deposits + non-deposit funds)
10*	Interest expenses/total expenses
11	Liquid assets/total assets
12*	Standard capital ratio
Spanish banks' data	
1*	Current assets/total assets
2*	Current assets-cash/total assets
3	Current assets/loans
4*	Reserves/loans
5*	Net income/total assets
6	Net income/total equity capital
7*	Net income/loans
8*	Cost of sales/sales
9	Cash flow/loans
US banks' data	
1	Working capital/total assets
2	Retained earnings/total assets
3	Earnings before interest and taxes/total assets
4	Market value of equity/total assets
5	Sales/total assets

* Features selected by DEWNN based feature selection algorithm.

Table 2
Average results for 10-fold cross validation for Bankruptcy datasets with all features

		DEWNN (%)	WNN (%)	TAWNN (%)
Turkish	Accuracy	95	95	100
	Sensitivity	100	100	100
	Specificity	95	95	100
Spanish	Average	89.99	86.67	88.33
	Sensitivity	91.66	89.67	79.66
	Specificity	93	81	90.5
US	Average	93.33	85.83	90.83
	Sensitivity	97.32	85.82	90.46
	Specificity	89.78	87.5	91.54

Table 3
Average results for 10-fold cross validation for other Benchmark datasets with all features

	DEWNN (%)	WNN (%)	TAWNN (%)
Iris	97.99	94.67	95.99
Wine	97.6	91.76	92.8
WBC	97.05	95.29	95.43

Table 4
Average results for ten fold cross validation for Bankruptcy datasets with reduced features

		DEWNN (%)	WNN (%)	TAWNN (%)
Turkish	Average	90	90	97.5
	Sensitivity	100	100	97.5
	Specificity	80	81.67	100
Spanish	Average	88.33	86.67	88.33
	Sensitivity	94.16	79.5	92.17
	Specificity	86	100	91

Table 5
Average results for ten fold cross validation for other benchmark datasets with reduced features

	DEWNN (%)	WNN (%)	TAWNN (%)
Wine	95.87	90.58	88.19
WBC	97.34	96.32	96.91

Table 6
Comparison of features selected by different techniques

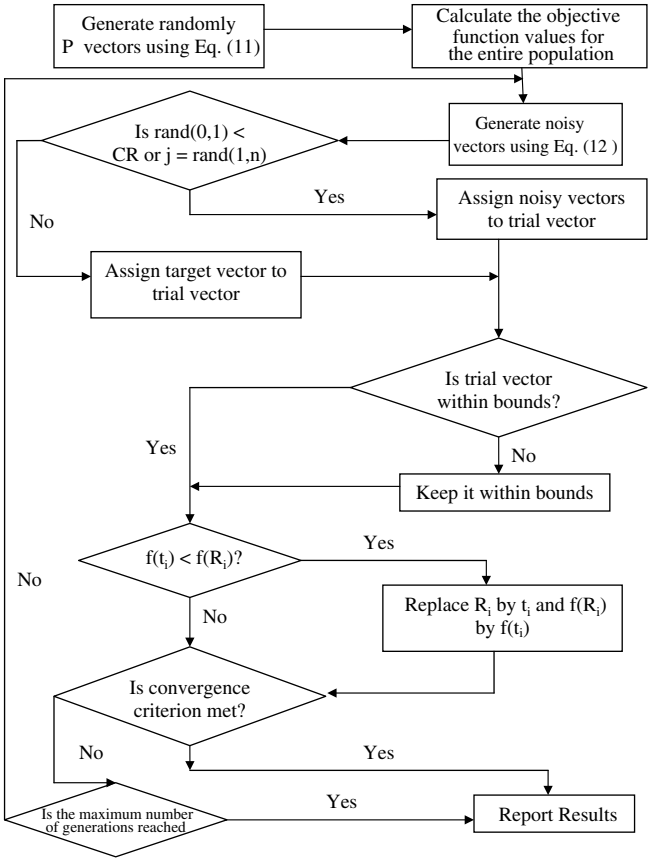
Dataset	DEWNN	WNN	TAWNN
Spanish	1,2,4,5,7,8	1,3,4,5,7,8	4,5,6,7,8,9
Turkish	3,5,7,9,10,12	3,4,5,6,7,12	3,4,5,9,10,12
Wine	1,4,5,7,10,12,13	1,3,4,7,11,12,13	1,3,6,7,10,12,13
WBC	1,2,4,6,8,9	1,2,3,6,7,8	1,2,3,4,6,9

indicated by asterisk (*) symbol in Table 1. Similar kind of experimentation is done with the feature subset also. The results for bankruptcy datasets i.e. Turkish banks and Spanish banks are presented in Table 4. The results indicate the overwhelming supremacy of DEWNN in accuracy and sensitivity as compared to TAWNN and original WNN. The results for other benchmark datasets i.e. wine data and Wisconsin breast cancer data with reduced features are presented in Table 5. DEWNN once again outperformed the other algorithms. In this case also the robustness of the algorithm is proved and the high accuracies show us the impeccable feature selection done by incorporating Garson's algorithm into DEWNN. The features for TAWNN and original WNN are extracted again by incorporating Garson's algorithm into TAWNN and WNN, respectively. Barring a few overlaps, all the algorithms selected different feature subsets as the optimal ones. This is not surprising because the training algorithms are different in each case. The features extracted by the three algorithms are presented in Table 6. Thus, it can be concluded that besides being robust, DEWNN is an effective algorithm for solving classification problems occurring in finance.

8. Conclusions

In this study, DEWNN is developed and compared with TAWNN and the original WNN on benchmark datasets viz. Iris dataset, Wine dataset and Wisconsin Breast Cancer dataset as well as bankruptcy data sets viz. US banks dataset, Turkish banks dataset, Spanish banks dataset and the results indicate that DEWNN can be a very effective soft computing tool for classification problems. In addition, we also adopted the Garson's feature selection algorithm to WNN, DEWNN and TAWNN and top features are extracted from Turkish, Spanish, Wine and Wisconsin breast cancer datasets using the procedure explained in Section 5. The results again indicate the superior performance of DEWNN as compared to TAWNN and the original WNN. Hence, the present research concludes that training WNN with differential evolution solves classification problems with increased accuracy.

Appendix A. Flowchart of differential evolution



References

Altman, E. (1968). Financial ratios, discriminant analysis and the prediction of corporate bankruptcy. *Journal of Finance, 23*, 589–609.

Avci, E. (2007). An expert system based on wavelet neural network-adaptive norm entropy for scale invariant texture classification. *Expert Systems with Applications, 32*, 919–926.

Becerra, V. M., Galvao, R. K. H., & Abou-Seads, M. (2005). Neural and wavelet network model for financial distress classification. *Data Mining and Knowledge Discovery, 11*, 35–55.

Bhat, T. R., Venkataramani, D., Ravi, V., & Murty, C. V. S. (2006). Improved differential evolution method for efficient parameter estimation in biofilter modeling. *Biochemical Engineering Journal, 28*, 167–176.

Canbas, S., Caubak, B., & Kilic, S. B. (2005). Prediction of commercial bank failure via multivariate statistical analysis of financial structures: The Turkish case. *European Journal of Operational Research, 166*, 528–546.

Cheng, C. B., Chen, C. L., & Fu, C. J. (2006). Financial distress prediction by a radial basis function network with logit analysis learning. *Computers and Mathematics with Applications, 51*, 579–588.

Cielen, A., Peeters, L., & Vanhoof, K. (2004). Bankruptcy prediction using a data envelopment analysis. *European Journal of Operational Research, 154*, 526–532.

Cole, R., & Gunther, J. (1995). A CAMEL rating's shelf life. *Federal Reserve Bank of Dallas Review*, 13–20. December.

Deb, K. (2000). An efficient constraint handling method for genetic algorithms. *Computer Methods in Applied Mechanics and Engineering, 186*, 311–338.

Dimoulas, C., Kalliris, G., Papanikolaou, G., Petridis, V., & Kalampakas, A. (2008). Bowel-sound pattern analysis using wavelets and neural networks with application to long-term, unsupervised, gastrointestinal motility monitoring. *Expert Systems with Applications, 34*, 26–41.

Dong, L., Xiao, D., Liang, Y., & Liu, Y. (2008). Rough set and fuzzy wavelet neural network integrated with least square weighted fusion algorithm based fault diagnosis research for power transformers. *Electric Power Systems Research, 78*, 129–136.

Dueck, G., & Scheuer, T. (1990). Threshold accepting: A general purpose optimization algorithm appearing superior to simulated annealing. *Journal of Computational Physics, 90*, 161–175.

Garson, D. G. (1991). Interpreting neural-network connection weights. *A! Expert*, 47–51. April.

Grossmann, A., & Morlet, J. (1984). Decomposition of Hardi functions into square integrable wavelets of constant shape. *SIAM Journal of Mathematical Analysis, 15*, 725–736.

Guyon, B., & Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research, 3*, 1157–1182.

Ilonen, J., Kamarainen, J.-K., & Lampinen, J. (2003). Differential evolution training algorithm for feed-forward neural networks. *Neural Processing Letters, 17*(1), 93–105.

Jo, H., Han, I., & Lee, H. (1997). Bankruptcy prediction using case-based reasoning, neural networks and discriminant analysis. *Expert Systems with Applications, 13*, 97–108.

Lung, S.-Y. (2007). Efficient text independent speaker recognition with wavelet feature selection based multilayered neural network using supervised learning algorithm. *Pattern Recognition, 40*, 3616–3620.

McKee, T. E. (2000). Developing a bankruptcy prediction model via rough set theory. *International Journal of Intelligent Systems in Accounting, Finance, and Management, 9*, 159–173.

Min, J. H., & Lee, Y.-C. (2005). Bankruptcy prediction using support vector machine (SVM) with optimal choice of kernel function parameters. *Expert Systems with Applications, 28*, 603–614.

Nath, J. R., Rajagopalan, B., & Ryker, R. (1997). Determining the saliency of input variables in neural network classifiers. *Computers and Operations Research, Vol. 24*(8), 767–773.

Olmeda, I., & Fernandez, E. (1997). Hybrid classifiers for financial multicriteria decision making: The case of bankruptcy prediction. *Computational Economics, 10*, 317–335.

Pan, C., Chen, W., & Yun, Y. (2008). Fault diagnostic method of power transformers based on hybrid genetic algorithm evolving wavelet neural network. *IET Electric Power Applications, Vol. 2*(1), 71–76.

Pramodh, C., & Ravi, V. (2007). Modified great deluge algorithm based auto associative neural network for bankruptcy prediction in banks. *International Journal of Computational Intelligence Research, 3*(4), 363–370.

Rahimian, E., Singh, S., Thammachote, T., & Virmani, R. (1996). Bankruptcy prediction by neural network. In R. R. Trippi & E. Turban (Eds.), *Neural networks in finance and investing*. Burr Ridge, USA: Irwin Professional Publishing.

Raj Kiran, N., & Ravi, V. (2007). Software reliability prediction using wavelet neural networks. *International Conference on Computational Intelligence and Multimedia Applications*, Sivakasi, Tamilnadu, India.

Ravi, V., & Pramodh, C. (2008). Threshold accepting trained principal component neural network and feature subset selection: Application to bankruptcy prediction in banks. *Applied Soft Computing, 8*(4), 1539–1548.

Ravi, V., Kurniawan, H., Peter Nwee Kok Thai & Ravi Kumar, P. (2008). Soft computing system for bank performance prediction. *Applied Soft Computing, 8*(1), 305–315.

Ravi, V., Ravi Kumar, P., Ravi Srinivas, E., & Kasabov, N. K. (2007). A semi-online training algorithm for the radial basis function neural networks: Applications to bankruptcy prediction in banks. In V. Ravi (Ed.), *Advances in Banking Technology and Management: Impact of ICT and CRM*. USA: Idea Group Inc.

Ravi Kumar, P., & Ravi, V. (2006a). Bankruptcy prediction in banks by fuzzy rule based classifier. In *Proceedings of first IEEE international conference on digital and information management* (pp. 222–227), Bangalore.

Ravi Kumar, P., & Ravi, V. (2006b). Bankruptcy prediction in banks by an ensemble classifier. In *Proceedings of the IEEE international conference on industrial technology* (pp. 2032–2036), Mumbai.

Ravi Kumar, P., & Ravi, V. (2007). Bankruptcy prediction in banks and firms via statistical and intelligent techniques – A review. *European Journal of Operational Research, 180*(1), 1–28.

Reza, M., & Ghorbani, A. (2007). Application of wavelet neural networks in optimization of skeletal buildings under frequency constraints. *International Journal of Intelligent Technology, 2*(4).

Storn, R., & Price, K. (1997). Differential evolution – A simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization, 11*, 341–359.

Vinay Kumar, K., Ravi, V., Carr, Mahil, & Raj Kiran, N. (2008). Software cost estimation using wavelet neural networks. *Journal of Systems and Software, 8*(11), 1853–1867.

Yu, G., Li, G., Bai, Y., & Jin, X. (2007). Tuning of the structure and parameters of wavelet neural network using improved chaotic PSO. In *Proceedings of 26th Chinese control conference*.

Yu, S.-N., & Chen, Y.-H. (2007). Electrocardiogram beat classification based on wavelet transformation and probabilistic neural network. *Pattern Recognition Letters, 28*, 1142–1150.

Zhang, Q., & Benvniste, A. (1992). Wavelet networks. *IEEE Transactions on Neural Networks, 3*(6), 889–898.

Zhang, Q. (1997). Using wavelet network in nonparametric estimation. *IEEE Transactions on Neural Networks, 8*(2), 227–236.

Zhang, X., Qi, J., Zhang, R., Liu, M., Hu, Z., Xue, H., et al. (2001). Prediction of programmed-temperature retention values of naphthas by wavelet neural networks. *Computers and Chemistry, 25*, 25–133.