Available online at www.sciencedirect.com
ScienceDirect
Applied Soft Computing 7 (2007) 995–1004
Applied Soft Computing
www.elsevier.com/locate/asoc

Weather analysis using ensemble of connectionist learning paradigms

Imran Maqsood [a], Ajith Abraham [b,*]

[a] Faculty of Engineering, University of Regina, Regina, Saskatchewan S4S 0A2, Canada
[b] IITA Professorship Program, School of Computer Science and Engineering, Yonsei University, 134 Shinchon-dong, Sudaemoon-ku, Seoul 120-749, Republic of Korea

Received 4 April 2003; accepted 12 June 2006
Available online 17 August 2006

Abstract

This paper presents a comparative analysis of different connectionist and statistical models for forecasting the weather of Vancouver, Canada. For developing the models, one year's data comprising of daily temperature and wind speed were used. A multi-layered perceptron network (MLPN) and an Elman recurrent neural network (ERNN) were trained using the one-step-secant and Levenberg–Marquardt algorithm. Radial basis function network (RBFN) was employed as an alternative to examine its applicability for weather forecasting. To ensure the effectiveness of neurocomputing techniques, the connectionist models were trained and tested using different datasets. Moreover, ensembles of the neural networks were generated by combining the MLPN, ERNN and RBFN using arithmetic mean and weighted average methods. Subsequently, performance of the connectionist models and their ensembles were compared with a well-established statistical technique. Experimental results obtained have shown RBFN produced the most accurate forecast model compared to ERNN and MLPN. Overall, the proposed ensemble approach produced the most accurate forecast, while the statistical model was relatively less accurate for the weather forecasting problem considered.

Keywords: Artificial neural networks; Ensemble; Forecasting; Multi-layered perceptron; Statistical model; Radial basis function; Elman recurrent neural network; Simulation; Weather

1. Introduction

Weather forecasts provide critical information about future weather. They help to safeguard life and property, and play a crucial role in planning the activities of government, industry, and the public. Weather forecasting remains a complex business, due to its chaotic and unpredictable nature [21,26]. Forecasting, in general is a process that is neither wholly science nor wholly art [20,39,45–50]. It is known that persons with little or no formal training can develop considerable forecasting skills [14,17]. For example, farmers often are quite capable of making their own short-term forecasts of those meteorological factors that directly influence their livelihood, and a similar statement can be made about pilots, fishermen, mountain climbers, etc. Weather phenomena, usually of a complex nature, have a direct impact on the safety and/or economic stability of such persons. Accurate weather forecast models are important to the regions, where the entire agriculture depends upon weather [1]. It is thus a major concern to identify any trends for weather parameters to deviate from its periodicity, which would disrupt the economy of the country. This fear has been aggravated due to threat by the global warming and green house effect. The impact of extreme weather phenomena on society is growing more and more costly, causing infrastructure damage, injury and the loss of life.

Previously, various methods were developed and used by meteorologists for weather forecasting from relatively simple observation of the sky to highly complex computerized mathematical models [1,2,11,24,28,33,35]. Among them, recently little research has been carried out on the application of artificial neural networks (ANN) techniques for modeling the chaotic behavior of weather [9,19,21,22,26,27,41]. In fact, ANN is very well suited for weather forecasting problems due to two potential reasons. Firstly, they are able to approximate numerically any continuous function to the desired accuracy. In this sense, ANN may be seen as multivariate, non-linear and non-parametric methods. They should be expected to model complex non-linear relationships, such as weather forecasting, much better than the traditional linear models that still form the core of the forecaster's methodology. Secondly, ANN are

* Corresponding author. Tel.: +82 2 820 5352; fax: +82 2 815 7906.
E-mail address: ajith.abraham@ieee.org (A. Abraham).
URL: http://www.softcomputing.net

1568-4946/$ – see front matter
doi:10.1016/j.asoc.2006.06.005

data-driven methods, in the sense that it is not necessary for the researcher to postulate tentative models and then estimate their parameters. Given a sample of input and output vectors, they are able to automatically map the relationship between them; they learn this relationship, and store this learning into their parameters. As these two characteristics suggest, ANN should prove to be particularly useful when one has a large amount of data, but little a priori knowledge about the laws that govern the system that generated the data.

Among ANN techniques, multi-layered perceptron network (MLPN), Elman recurrent neural networks (ERNN) and radial basis function network (RBFN) are most commonly used methods [4,8,17,24,34]. The RBFN is a popular alternative to the MLPN. Although it is not as well suited to larger applications, can offer advantages over the MLPN in some applications [4,8,31]. In comparison, in ERNN, the temporal nature of the data is taken into account. MLPN are capable of modeling non-linearity and the only way to adapt MLPN to temporal data is to provide the entire time series to the network as input at each training cycle. This not only requires networks of immense size, which in turn require a great deal of processing power and time to converge, but also limits the network to fixed-length time series [13,25]. The technique of combining the predictions of multiple networks to produce a single network, called as an ensemble of neural networks, have been investigated by many researchers [7,10,16,32,36,40,42–44]. The resulting network is generally more accurate than any of the individual networks making up the ensemble [29]. Moreover, it is reported that one of the effective combining schemes is to simply average the predictions of the network [3,16,18,22].

The main objective of this study is to investigate applicability of ensembles of neural networks for weather analysis of Vancouver, British Columbia, Canada. To achieve this objective, ANN-based accurate weather forecast models are developed, and performances of the ANN models and their resulting ensembles are computed as well as compared with a classical statistical method. To improve the learning capability of the ANN models, second order error information using the one-step-secant and the Levenberg–Marquardt approaches for MLPN and ERNN are used. Furthermore, a RBFN, which is also a well-established technique for function approximation, is also applied [31]. Subsequently, ensembles of the ANN models are computed and compared with individual ANN and statistical models.

In Section 2, a brief theoretical background of MLPN, RNN, RBFN and ensembles of neural networks is presented. The experimentation setup is described in Section 3, and results and discussions are provided in Section 4. Finally, some conclusions are drawn towards the end in Section 5.

2. Artificial neural networks

Artificial neural networks (ANNs) were designed to mimic the characteristics of the biological neurons in the human brain and nervous system [38]. The network "learns" by adjusting the interconnections (called weights) between layers. When the network is adequately trained, it is able to generalize relevant output for a set of input data. Learning typically occurs by example through training, where the training algorithm, such as one-step-secant and the Levenberg–Marquardt, iteratively adjusts the connection weights (synapses).

2.1. Multi-layered perceptron networks (MLPN)

Typical MLPN is arranged in layers of neurons (nodes), where every neuron in a layer computes the sum of its inputs and passes this sum through a non-linear function (an activation function) as its output. Each neuron has only one output, but this output is multiplied by a weighting factor if it is to be used as an input to another neuron (in a next higher layer). There are no connections among neurons in the same layer. Fig. 1 illustrates a three-layered MLPN used for the weather forecasting.

Activation functions for the hidden layers are needed to introduce non-linearity into the network. Without non-linearity, hidden layers would not make networks more powerful than just simple perceptrons (which do not have any hidden layers, only input and output layers). A composition of linear functions is again a linear function. It is the non-linearity (i.e. the capability to represent non-linear functions) that makes MLPN so powerful. For backpropagation learning, however, it must be differentiable and saturating at both extremes. Sigmoid functions such as the logistic and hyperbolic tangent functions, and the Gaussian function are the most common choices. If the transfer functions were chosen to be linear, then the network would become identical to a linear filter [15].

The training of a network by backpropagation [6] involves three stages: the feedforward of the input training pattern, the calculation and backpropagation of the associated error, and the adjustment of the weights. After training, application of the net involves only the computations of the feedforward phase. In order to train the network, input is shown to the net together with the corresponding known output, and if there exists a relation between the input, ξ_k^{μ}, and the output, O_i^{μ}, the net learns by adjusting the weights until an optimum set of weights that minimizes the network error is found and the network then converges.

2.2. Elman recurrent neural networks (ERNN)

ERNN, also known as partially recurrent neural network, are a subclass of recurrent networks [12,13]. They are multi-layer

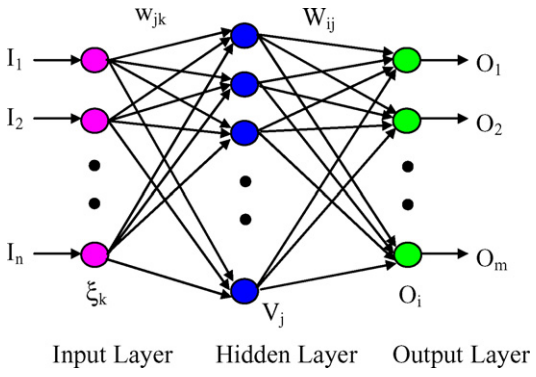

Fig. 1. Architecture of multi-layered perceptron network.

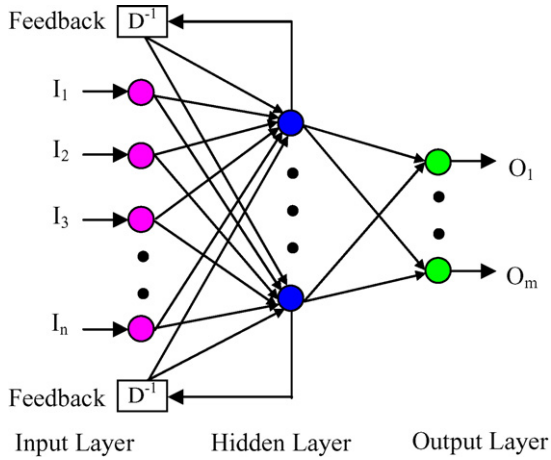

Fig. 2. Schematic diagram of three-layered Elman recurrent neural network.

perceptron networks augmented with one or more additional context layers storing output values of one of the layers delayed by one-step and used for activating this or some other layer in the next time step, as shown in Fig. 2. The ERNN has context units, which store delayed hidden layer values and present these as additional inputs to the network. The ERNN can learn sequences that cannot be learned with other recurrent neural network e.g. with Jordan recurrent neural network (which is a similar architecture with a context layer fed by the output layer) since networks with only output memory cannot recall inputs that are not reflected in the output. Several training algorithms for calculation of error gradient in general recurrent networks exist.

Usually, both hidden and output units have non-linear activation functions. Note that external input at time t does not influence the output of any unit until time $t + 1$. The network is thus a discrete dynamical system.

2.3. Radial basis function network (RBFN)

RBFN network consists of three-layers: input layer, hidden layer, and output layer, as shown in Fig. 3. The neurons in hidden layer are of local response to its input and known as RBF neurons, while the neurons of the output layer only sum their inputs and are called linear neurons [30]. It is well known that neural network training can result in producing weights in undesirable local minima of the criterion function. This problem is particularly serious in recurrent neural networks as well as for MLPN with highly non-linear activation functions, because of their highly

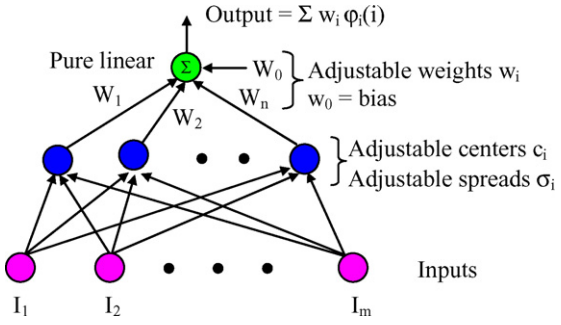

Fig. 3. Architecture of radial basis function network.

non-linear structure, and it gets worse as the network size increases. This difficulty has motivated many researchers to search for a structure where the output dependence on network weights is less non-linear. The RBFN has a linear dependence on the output layer weights, and the non-linearity is introduced only by the cost function for training, which helps to address the problem of local minima. Additionally, this network is inherently well suited for weather prediction, because it naturally uses unsupervised learning to cluster the input data [8,31,34].

There are two basic methods to train an RBFN in the context of neural networks. One is to jointly optimize all parameters of the network similarly to the training of the MLPN. This method usually results in good quality of approximation but also has some drawbacks such as a large amount of computation and a large number of adjustable parameters. Another method is to divide the learning of an RBFN into two steps. The first step is to select all the centers μ in terms of an unsupervised clustering algorithm such as the K-means algorithm proposed by Linde et al. (denoted as the LBG algorithm) [5], and choose the radii σ by the k-nearest neighbor rule. The second step is to update the weights B of the output layer, while keeping the μ and σ fixed. The two-step algorithm has fast convergence rate and small computational burden.

We used a two-step learning algorithm to speed up the learning process of the RBFN. The selection of the centers and radii of RBF neurons can be done naturally in an unsupervised manner, which makes this structure intrinsically well suited for weather prediction. As a result, we adopt below a self-organized learning algorithm for selection of the centers and radii of the RBF in the hidden layer, and a stochastic gradient descent of the contrast function for updating the weights in the output layer. For the selection of the centers of the hidden units, we may use the standard k-means clustering algorithm [8]. This algorithm classifies an input vector x by assigning it the label most frequently represented among the k-nearest neighbor samples. Specifically, it places the centers of RBF neurons in only those regions of the input space, where significant data are present. Once the centers and radii are established, we can make use of the minimization of the contrast function to update the weights of the RBFN.

ANNs offer a variety of advantages, including: (a) an ability to solve complex and non-linear problems, such as weather forecasting, that cannot be described explicitly or that are difficult to compute using traditional methods; (b) the capability of a neural network to learn a given process automatically through a training phase; (c) there is no need to assume or recognize an underlying distribution for the data collected; (d) the compact form in which the knowledge is stored and the comparative ease and speed at which this knowledge can be accessed; (e) robustness in the presence of noise; (f) the high degree of accuracy by any trained network when an ANN solution is used to generalize unseen cases.

2.4. Ensembles of neural networks

No single classification algorithm can be regarded as a panacea. In fact, different neural networks perform differently

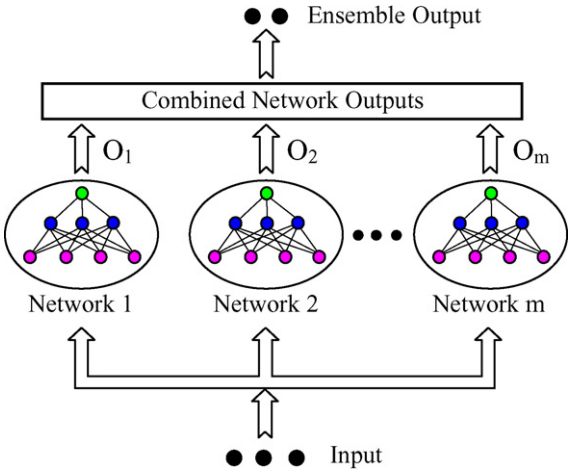

Fig. 4. Architecture of a classifier ensemble of neural networks.

due to different architectures, different learning parameters/algorithms, different initial random weights, different training sets, etc. [5]. Generally, the errors are not fully correlated among neural networks when using multiple networks for a single problem. Particularly, if a negative correlation exists among the multiple networks, then an ensemble approach does better than a single network approach.

Fig. 4 illustrates the basic framework for a classifier ensemble. In this example, neural networks are the basic classification method, though conceptually any classification method can be substituted in place of the networks. Each network in Fig. 4 ensemble (network 1 through network m in this case) is trained using the training instances for that network. Then, for each example, the predicted output of each of these networks (O_i in Fig. 4) is combined to produce the output of the ensemble.

Combining outputs (i.e. ensembles) can be conducted by computing majority vote, arithmetic mean, median, weighted average or trimmed mean of multiple neural networks and/or by setting up another trained neural network (i.e. stacked) [37].

3. Experiment setup and analysis

In the normal case, architecture of the connectionist models is determined after a time-consuming trial-and-error procedure [50]. To circumvent this disadvantage, we use a more systematic way of finding good architectures. A sequential network construction [23] is employed to select an appropriate number of hidden neurons for each of the connectionist model considered. First, a network with a small number of hidden neurons is trained. Then a new neuron is added with randomly initialized weights and the network is retrained with changes limited to these new weights. Next all weights in the network are retrained. This procedure is repeated until the number of hidden neurons reaches a preset limit and then substantially reduces the training time in comparison with time needed for training of new networks from scratch. More importantly, it creates a nested set of networks having a monotonously decreasing training error and provides some continuity in the model space, which makes a prediction risk minimum more easily noticeable.

The concept of forecasting-model consists of: (a) proper model selection of the technique that matches with the local requirements, (b) calculation and update of model parameters, which includes the determination of the network parameters and selection of the method to update the constants values as the circumstance varies (seasonal changes), (c) evaluation of the model performance to validate the model using historical data, also the final validation to use the model in real life conditions. The evaluation terms includes accuracy, ease of use and bad/anomalous data detection and (d) update/modification of the model, if the performance is not satisfactory. Due to sudden variation in weather parameters, the model becomes obsolete and inaccurate. Thus, model performance and accuracy should be evaluated continuously [23]. Sometimes, periodic update of parameters or change of model structure is also required.

We used the weather data from 1 September 2000 to 31 August 2001 for analyzing the connectionist models. For MLPN and ERNN, we used the dataset from 11 to 20 January 2001 for testing and remaining data for training the networks. We also used the dataset 1–15 April 2001 for testing and the remaining for training of RBFN, MLP and ERNN networks. We used this method to ensure that there is no bias on the training and test datasets. We used a Pentium-III, 1 GHz processor with 256 MB RAM and all the experiments were simulated using MATLAB. The following steps were taken before starting the training process: (a) the error level was set to a relatively small value (10^{-4}) that could be decreased to a smaller level, but the results show satisfactory prediction of the required outputs. Also setting the training accuracy to a higher level will take a much longer training time; (b) the hidden neurons were varied (10–80) and the optimal number for each network were then decided as mentioned previously by changing the network design and running the training process several times until a good performance was obtained; (c) when the network faces local minima (false wells), new ones to escape from such false wells replace the whole set of network weights and thresholds. Actually, a random number generator was used to assign the initial values of weights and thresholds with a small bias as a difference between each weight connecting two neurons together since similar weights for different connections may lead to a network that will never learn.

MLPN, ERNN and RBFN were used after deciding the relevant input/output parameters, training/testing data sets and learning algorithms. To decide architectures of the MLPN and ERNN, a trail and error approach was used. Networks were trained for a fixed number of epochs, and the error gradient was observed over these epochs. Performance of the MLPN and ERNN networks were evaluated by increasing or decreasing the number of hidden nodes. Since no significant reduction in error was observed beyond 45 hidden nodes, a single hidden layer network comprising of 45 neurons was identified. The input and output values were scaled between −1 and +1. One-step-secant and Levenberg–Marquardt learning algorithms were used for training the MLPN and ERNN networks. The activation functions for MLPN and ERNN models were chosen to be log-sigmoid and hyperbolic tangent sigmoid for hidden units, respectively, and pureline for the output units. Since, there is no

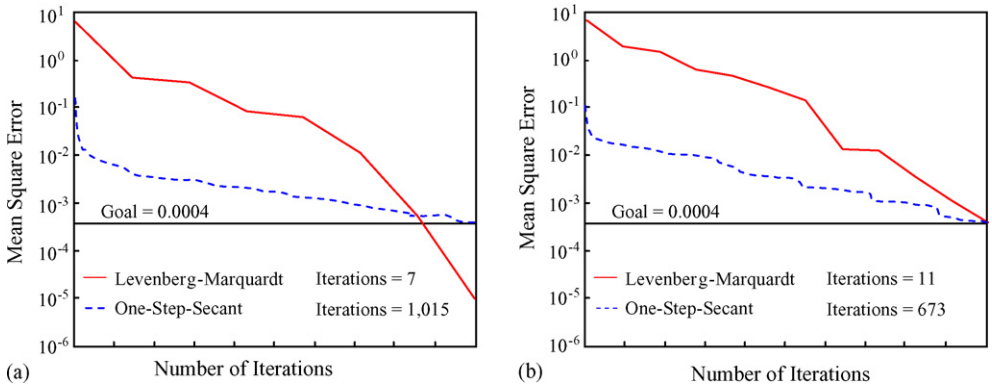

Fig. 5. Convergence of the OSS and LM training algorithms using: (a) MLPN and (b) ERNN.

exact rule for fixing the number of hidden neurons and hidden layers to avoid underfitting or overfitting in MLPN and ERNN, therefore, RBFN is investigated to address this difficulty. In RBFN, the numbers of hidden layers and neurons selected by the model were 2 and 180, respectively; the Gaussian activation function was chosen for hidden units, and the pureline for the output units.

The three strategies adopted for testing the applicability of an ANN in the present work are: (a) to test the capability of an ANN to correctly predict the output for the given input set originally used to train the network (accuracy performance); (b) to test the capability of the ANN to correctly predict the output for the given input sets that were not included in the training set (generalized performance); (c) to develop a neural network model which could be trained faster.

3.1. Weather parameters

3.1.1. Temperature

Our initial analysis of the data has shown that the most important weather parameter is the temperature variable (in degree Celsius units). This variable also represents a strong correlation with other weather parameters. Temperature, in general, can be measured to a higher degree of accuracy relative to any of the other weather variables. Anyhow, forecasting temperature requires the consideration of many factors; day or night, clear or cloudy skies, windy or calm, or will there be any precipitation? An error in judgment on even one of these factors may cause forecasted temperature to be off by as much as 20°. Historical temperature data recorded by a weather station at the prominent meteorological center of the Vancouver, British Columbia is used for the analysis.

Space heating and cooling are the human response to how hot and how cold it 'feels'. Temperature in this case is only one of the contributors to such human response. Factors such as wind speed in the winter and humidity in the summer have to be accounted for when describing how cold or hot it 'feels'. For that purpose, the weather department developed a measure of how cold the air feels in the winter or how hot the air feels in the summer. These two new measurements are called the *wind-chill* and the *heat index*, respectively. Meteorologists use wind-chill equations to calculate the rate at which exposed human skin loses body heat.

3.1.2. Wind speed

It is significant in winter, if the temperature is low, which is roughly below 4.4 °C. When temperature is below freezing point, wind is a major factor determining the cooling rate. The wind speed recorded in Vancouver, British Colombia for the year 2001 is considered for this study.

4. Test results and discussions

4.1. Evaluation of OSS and LM learning algorithms

The training convergence of LM and OSS learning approaches for MLPN and ERNN are illustrated in Fig. 5(a) and (b), respectively.

The optimal network is the one that has the lowest error on test set and reasonable learning time. All the obtained results were compared and evaluated by the maximum absolute percentage error (MAPE), root mean squared error (RMSE), and mean absolute deviation (MAD). Test results (January 2001) for the actual versus predicted temperature and wind speed using MLPN and ERNN with OSS and LM approaches are plotted in Figs. 6 and 7, respectively. Relative percentage error for temperature and wind speed is also illustrated in Figs. 8 and 9, respectively. Empirical results are depicted in Table 1.

In this paper, the assessment of the forecasting performance of the trained networks is done by various forecasting errors. If the training is successful, the network is able to generalize well, resulting in a high accuracy in the forecasting of unknown

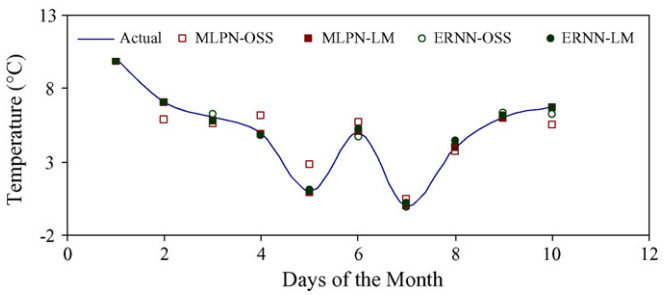

Fig. 6. Comparison of actual and forecasted temperature using OSS and LM approaches for MLPN and ERNN.

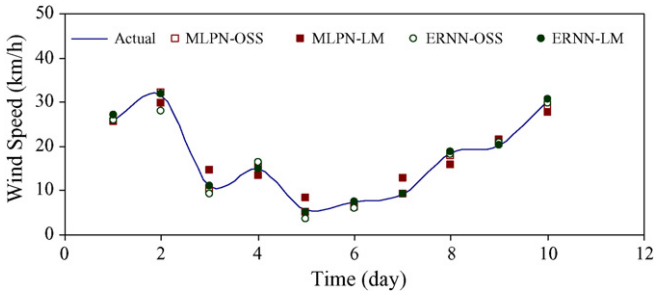

Fig. 7. Comparison of actual and forecasted wind speed using OSS and LM approaches for MLPN and ERNN.

patterns (provided the training data is sufficient representative of the forecasting situation). Various forecasting error measures between the actual and forecasted weather parameters are defined, however, the most commonly adopted by weather forecasters are shown in Table 1.

Table 1 indicates that the LM algorithm generated the lowest errors and higher correlation with respect to the actual values than that of the OSS algorithm. However, the LM algorithm took more training time and had relatively less number of iterations compared to the OSS algorithm. The training time for the LM algorithm ranged from a few minutes to 30 min, while it took only a few seconds to run the models with OSS learning algorithm. In fact, with the improvement of computing speed (and big memory), the training time due to different algorithms may no longer be such a crucial factor if the training record is not too long and the design architecture is not too complicated. The testing accuracy could get worse if the selection of the algorithm to represent the problem is not proper.

4.2. Weather forecasts with multiple neural networks

Test results (April 2001) for the actual versus predicted temperature and wind speed using MLPN (OSS) and ERNN (OSS) and RBFN are illustrated in Figs. 10 and 11.

Table 2 presents empirical results from the MLPN, ERNN and RBFN models. It is shown that the MLPN can achieve useful weather forecasting results in an efficient way and

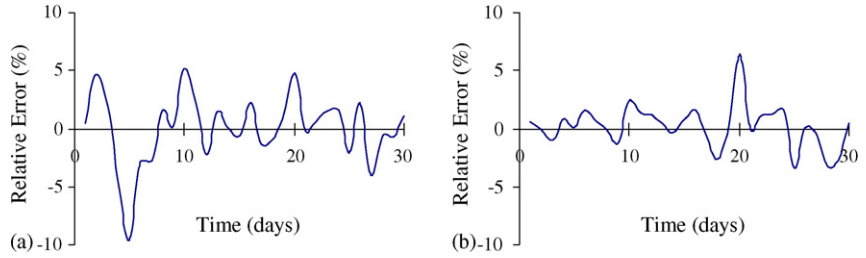

Fig. 8. Relative percentage error between actual and forecasted temperature: (a) MLPN and (b) ERNN.

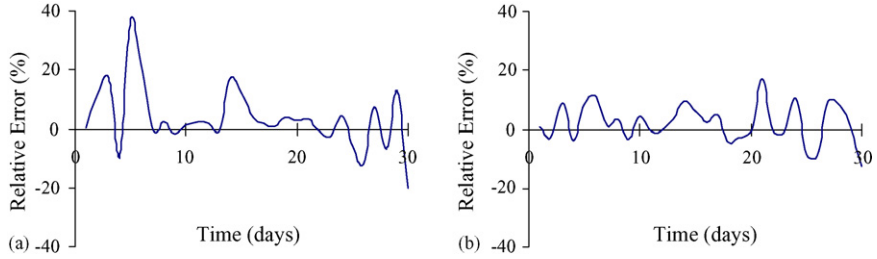

Fig. 9. Relative percentage error between actual and forecasted wind speed: (a) MLPN and (b) ERNN.

Table 1
Performance of OSS and LM for temperature (°C) and wind speed (km/h) forecast

Weather parameter	Performance indicator	OSS learning algorithm		LM learning algorithm	
		MLPN	ERNN	MLPN	ERNN
Temperature	MAPE	0.0170	0.0165	0.0087	0.0048
	RMSE	0.0200	0.0199	0.0099	0.0067
	MAD	0.8175	0.7944	0.4217	0.2445
	R^2	0.9647	0.9457	0.9998	0.9826
	Time (min)	0.4	1.8	30	30
	Iterations	850	1135	7	10
Wind speed	MAPE	0.0896	0.0873	0.0770	0.0333
	RMSE	0.1989	0.0199	0.0162	0.0074
	MAD	0.8297	0.7618	0.6754	0.3126
	R^2	0.9714	0.9886	0.9974	0.9995
	Time (min)	0.3	0.5	1	8
	Iterations	851	1208	8	12

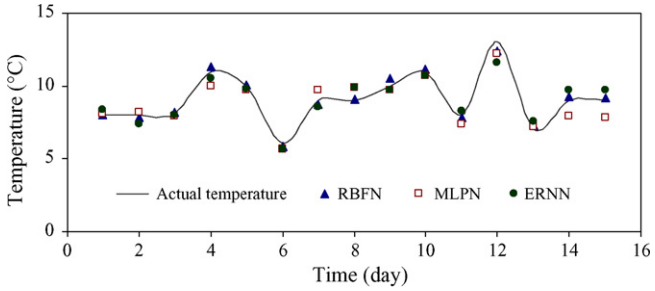

Fig. 10. Comparison of different NN techniques for 15-day ahead temperature forecasting.

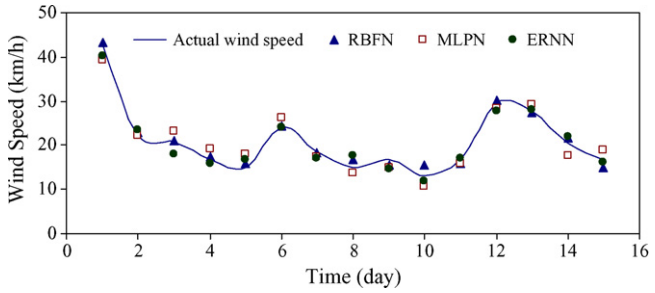

Fig. 11. Comparison of different NN techniques for 15-day ahead wind speed forecasting.

exhibited lower errors. It is capable of representing non-linear functions than the single layered perceptron. However, the learning process of the MLPN algorithm is time-consuming and its performance is heavily dependent on the network parameters like learning rate and momentum. On the other hand, the ERNN, compared to the MLPN, could efficiently capture the dynamic behavior of the weather, resulting in a more compact and natural representation of the temporal information contained in the weather profile. The recurrent network took more training time, but this depends on the data size and the number of network parameters. It can be inferred that the ERNN could yield more accurate results, if good data-selection strategies, training paradigms, and network input and output representations are determined properly. In comparison, the RBFN network gave the overall best results in terms of accuracy and training time. The RBFN is better correlated compared to the MLPN and ERNN. The proposed RBFN network can also overcome several limitations of the MLPN and ERNN such as highly non-linear weight update and the slow convergence rate. Since the RBFN has natural unsupervised learning characteristics and a modular network structure, these properties make it more effective for fast and robust weather forecasting.

Table 2 indicates that the temperature was predicted with the lowest MAPE, RMSE and MAD values by the individual models of MLPN, ERNN and RBFN. In comparison, wind speed predictions represented the least accurate results (i.e. higher MAPE, ERNN and RBFN values) for all the three individual models. The different levels of errors associated with temperature and wind speed could be due to the variability in data and of the modeling structures.

Table 2
Performance comparison of the NN models, the NN ensembles and the statistical model

Model	Parameter	Temperature (°C)	Wind speed (km/h)
MLPN	MAPE	0.0605	0.1000
	RMSE	0.6664	2.1006
	MAD	0.5561	1.9185
	R^2	0.9344	0.9588
ERNN	MAPE	0.0552	0.0740
	RMSE	0.5945	1.6556
	MAD	0.5058	1.4152
	R^2	0.9382	0.9761
RBFN	MAPE	0.0249	0.0573
	RMSE	0.2765	1.1835
	MAD	0.2278	0.9714
	R^2	0.9868	0.9886
NN mean-ensemble	MAPE	0.0265	0.0418
	RMSE	0.3385	1.0045
	MAD	0.2441	0.8051
	R^2	0.9840	0.9915
NN best-ensemble	MAPE	0.0214	0.0352
	RMSE	0.2416	0.7593
	MAD	0.1978	0.6227
	R^2	0.9901	0.9958
Statistical model	MAPE	0.2445	0.3997
	RMSE	2.4339	8.4228
	MAD	1.9647	7.4054
	R^2	0.1955	0.2552

4.3. Ensembles of multiple neural networks

The experimental comparisons of the MLPN, ERNN and RBFN pointed out that no single learning algorithm or model can be regarded as a panacea. Here, the use of ensembles of neural networks (NN) as an alternative approach is proposed. In this study, the results provided by these three neural networks are merged according to two fashions: simple average and weighted average. In simple average method, obviously equal weights were assigned to all the three predicted outputs and the resulting ensemble is here called ‘‘NN mean-ensemble’’. On the other hand, in weighted average method, full weights (i.e. 1) were assigned to the best values that were closest to the actual values and zero weight to the remaining values among the predicted outputs. This led to ‘‘NN best-ensemble’’. Figs. 12 and 13 present comparison of the two ensembles with actual temperature and wind speed, respectively.

Table 2 lists the performance of the ‘‘NN mean-ensemble’’ and ‘‘NN best-ensemble’’. Experimental results point out that the use of neural networks ensembles can constitute a valid alternative to the development of new neural approximators more complex than the present ones. In particular, it is shown that combination of NN results provides function approximation accuracies better than the ones obtained by single approximators after long designing phases. In addition, it is interesting to compare NN ensembles performances with the one of the classical statistical methods, which is being presented in the following subsections.

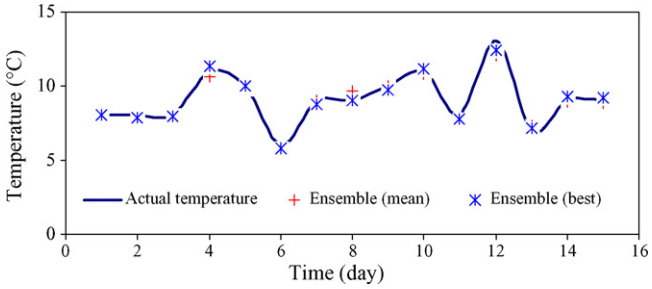

Fig. 12. Comparison between actual temperature and NN ensembles.

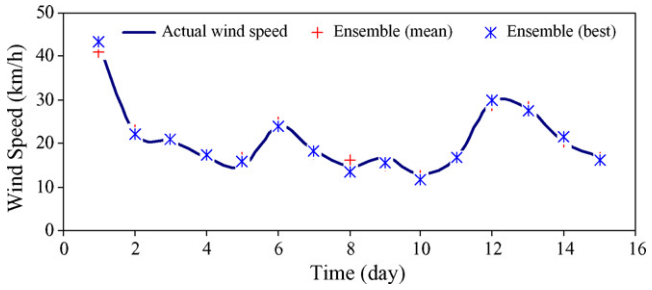

Fig. 13. Comparison between actual wind speed and NN ensembles.

4.4. Statistical forecasts

Regression analysis was applied for weather predictions in addition to the forecasts made by the MLPN, ERNN, RBFN models and their ensembles. In regression analysis, a trendline of polynomial function was drawn and extended beyond the actual data of temperature and wind speed to predict their future values. Figs. 14 and 15 show exponential trendlines for temperature and wind speed, respectively.

The developed equations for the temperature and wind speed trendlines are as follows:

$$T = 0.000004t^3 - 0.002t^2 + 0.292t + 8.513 \quad (1)$$

$$W = 0.0000004t^3 - 0.00005t^2 - 0.038t + 24.415 \quad (2)$$

where T is the temperature (°C), W the wind speed (km/h) and t is the time of forecast (day). Eqs. (1) and (2) were used for temperature and wind speed forecasting, respectively, for 1–15 April 2001. The MAPE, RMSE, MAD and R^2 values obtained through the statistical models are summarized in Table 2. In

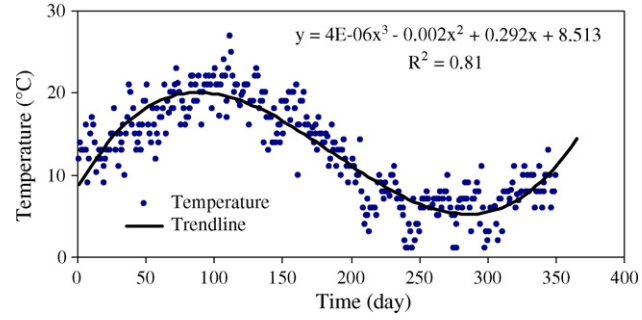

Fig. 14. Actual temperature and statistical forecasting trendline.

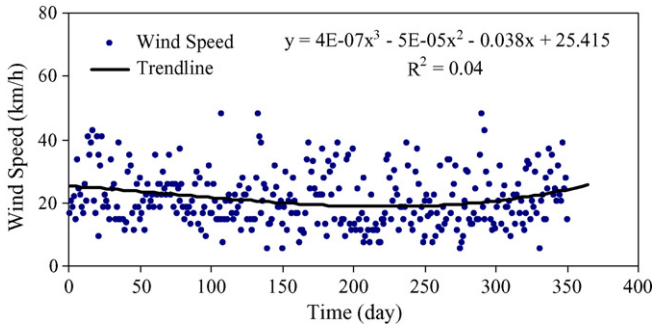

Fig. 15. Actual wind speed and statistical forecasting trendline.

over all, the statistical forecasts produced higher errors both for temperature and wind speed.

4.5. Comparison of neural networks and statistical methods for weather forecasts

Performance of MLPN, ERNN, RBFN, NN mean-ensemble, NN best-ensemble, and statistical models are evaluated by computing a number of measures such as MAPE, RMSE, MAD and R^2, as shown in Table 2. It is indicated that the ''NN best-ensemble'' produced the most accurate results. In comparison, the statistical models forecasted the temperature and wind speed with least accuracy. This illustrates the ''NN best-ensemble'' approach is superior to the statistical model as well as individual neural networks for this weather analysis study.

Figs. 16 and 17 illustrate the relative percentage errors of the NN mean-ensemble and statistical model for temperature and wind speed forecasts, respectively. It is shown that the statistical model generated higher errors compared to the NN mean-ensemble.

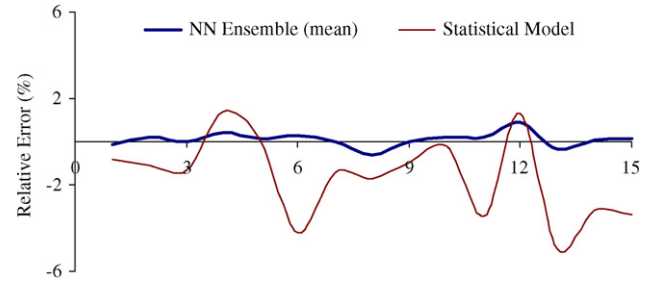

Fig. 16. Relative error of NN mean-ensemble and statistical model for temperature forecast.

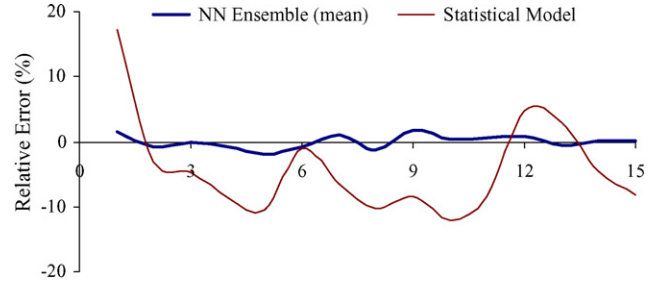

Fig. 17. Relative error of NN mean-ensemble and statistical model for wind speed forecast.

The above six models may be ranked based on their overall performance in descending (from good to not-so-good) order as follows: (1) NN best-ensemble, (2) NN mean-ensemble, (3) RBFN, (4) ERNN, (5) MLPN, and (6) statistical model.

5. Conclusions

In this study, a number of connectionist and statistical models were developed for forecasting the weather of Vancouver, Canada. One year's data comprising of daily temperature and wind speed were used. Performance of multi-layered perceptron neural network (MLPN), Elman recurrent neural network (ERNN) and radial basis functions network (RBFN) were contrasted using several statistical measures. Compared to the MLPN, the ERNN could efficiently capture the dynamic behavior of the weather, resulting in a more compact and natural internal representation of the temporal information contained in the weather profile. ERNN took more training time, however, it is dependent on the training data size and the number of network parameters. It can be inferred that ERNN could yield more accurate results, if good data selection strategies, training paradigms, and network input and output representations are determined properly. Levenberg–Marquardt (LM) approach appears to be the best learning algorithm for mapping the different chaotic relationships. Due to the calculation of Jacobian matrix at each epoch, LM approach requires more memory and is computationally complex while compared to one-step-secant (OSS) algorithm. On the other hand, RBFN gave the best results in terms of accuracy and fastest training time. Empirical results clearly demonstrate that RBFN are much faster and more reliable for the weather forecasting problem considered. The proposed RBFN network can also overcome several limitations of the MLP and ERNN networks such as highly non-linear weight update and slow-convergence rate. Since the RBFN has natural unsupervised learning characteristics and modular network structure, these properties make it a more effective candidate for fast and robust weather forecasting.

Ensembles of the neural networks were generated by linearly combining the MLPN, ERNN and RBFN using arithmetic mean and weighted average methods. Subsequently, performance of the connectionist models and their ensembles were contrasted with a statistical model. Experimental results obtained have shown RBFN produced the most accurate forecast model compared to ERNN and MLPN. In overall, the ensembles of neural networks produced the most accurate forecasts, while the statistical model was relatively less accurate for this weather forecasting problem. Moreover, overall performance of the six models be ranked in descending order as follows: (1) NN best-ensemble, (2) NN mean-ensemble, (3) RBFN, (4) ERNN, (5) MLPN, and (6) statistical model.

Although reasonable accuracy have been achieved through neurocomputing models using 1-year weather data, it will be interesting to study whether the inclusion of other seasonal factors would improve the forecast accuracy.

Acknowledgement

The authors are grateful to the staff of the meteorological department, Vancouver, BC, Canada for the useful discussions and providing the weather data used in this research work.

References

[1] Abraham, S. Philip, B. Joseph, Will we have a wet summer? Long-term rain forecasting using soft computing models, modeling and simulation 2001, in: Proceedings of the 15th European Simulation Multi Conference on Society for Computer Simulation International, Prague, Czech Republic, (2001), pp. 1044–1048.
[2] G. Allen, J.F. Le Marshall, An evaluation of neural networks and discriminant analysis methods for application in operational rain forecasting, Aust. Meteorol. Mag. 43 (1) (1994) 17–28.
[3] E. Alpaydin, Multiple networks for function learning, in: Proceedings of the IEEE International Conference on Neural Networks, San Francisco, (1993), pp. 27–32.
[4] Aussem, F. Murtagh, M. Sarazin, Dynamical recurrent neural networks and pattern recognition methods for time series prediction, application to seeing and temperature forecasting in the context of ESO's VLT astronomical weather station, Vistas Astron. 38 (3) (1994) 357.
[5] W. Baxt, Improving the accuracy of an artificial neural network using multiple differently trained networks, Neural Comput. 4 (1992) 772–780.
[6] C.M. Bishop, Neural Networks for Pattern Recognition, Oxford Press, 1995.
[7] L. Breiman, Bagging predictors, Mach. Learn. 24 (2) (1996) 123–140.
[8] S. Chen, S. MacLaughlin, B. Mulgrew, Complex-valued radial basis function network. Part I: Network architecture and learning algorithm, Signal Process. 35 (1994) 19–31.
[9] J.T. Cholewo, M.J. Zurada, Neural network tools for stellar light prediction, in: Proceedings of the IEEE Aerospace Conference, vol. 3, USA, (1997), pp. 422–514.
[10] R. Clemen, Combining forecasts: a review and annotated bibliography, J. Forecasting 5 (1989) 559–583.
[11] C.A. Doswell, Short range forecasting: mesoscale meteorology and forecasting, Am. Meteorol. Soc. (1986) 689–719 (Chapter 29).
[12] J.L. Elman, Distributed representations, simple recurrent networks and grammatical structure, Mach. Learn. 7 (2/3) (1991) 195–226.
[13] J.L. Elman, Finding structure in time, Cognitive Sci. 14 (1990) 179–211.
[14] S.D. Gedzelman, Forecasting skill of beginners, Bull. Am. Meteorol. Soc. 59 (1978) 1305–1309.
[15] M.T. Hagan, H.B. Demuth, M.H. Beale, Neural Network Design, PWS Publishing, Boston, 1996.
[16] L.K. Hansen, P. Salamon, Neural network ensembles, IEEE Trans. Pattern Anal. Mach. Intel. 12 (10) (1990) 993–1000.
[17] M.R. Khan, C. Ondrusek, Short-term load forecasting with multilayer perceptron and recurrent neural network, J. Electr. Eng. (Slovak Republic) 53 (2002) 17–23.
[18] J. Krogh, J. Vedelsby, Neural network ensembles, cross-validation, and active learning, in: G. Tesauro, D. Touretzky, T. Leen (Eds.), Advances in Neural Information Processing Systems, vol. 7, MIT Press, Cambridge, MA, 1995, pp. 231–238.
[19] S. Kugblenu, S. Taguchi, T. Okuzawa, Prediction of the geomagnetic storm associated D_{st} index using an artificial neural network algorithm, Earth Planets Space 51 (1999) 307–313.
[20] R.J. Kuligowski, A.P. Barros, Experiments in short-term precipitation forecasting using artificial neural networks, Mon. Weather Rev. 126 (2) (1998) 470.
[21] R.J. Kuligowski, A.P. Barros, Localized precipitation forecasts from a numerical weather prediction model using artificial neural networks, Weather Forecasting 13 (4) (1998) 1194.
[22] W. Lincoln, J. Skrzypek, Synergy of clustering multiple back propagation networks, in: D. Touretzky (Ed.), Advances in Neural Information Processing Systems, vol. 2, Morgan Kaufmann, San Mateo, CA, 1989, pp. 650–659.

[23] Y. Linde, A. Buzo, R. Gray, An algorithm for vector quantizer design, IEEE Trans. Commun. 28 (1980) 84–95.
[24] Maqsood, M.R. Khan, A. Abraham, Neurocomputing-based Canadian weather analysis, in: A. Abraham (Ed.), Proceedings of the Second International Workshop on Intelligent Systems Design and Applications, Computational Intelligence and Applications, Dynamic Publishers Inc., USA, 2002, pp. 39–44.
[25] J. Moody, J. Utans, Architecture selection strategies for neural networks: application to corporate bond rating prediction, in: Neural Networks in the Capital Markets, John Wiley and Sons, 1994.
[26] Q.I.M. Sancho, L. Alonso, C.E. Vivaracho, Application of neural networks to weather forecasting with local data, Appl. Inform. 68 (1994).
[27] A.H. Murphy, et al., Probabilistic severe weather forecasting at NSSFC: an experiment and some preliminary results, in: Proceedings of the 17th Conference on Severe Local Storms, American Meteorological Society, 1993, pp. 74–78.
[28] T.R. Neelakantan, N.V. Pundarikanthan, Neural network-based simulation-optimization model for reservoir operation, J. Water Resour. Plan. Manage. 126 (2) (2000) 57–64.
[29] D. Opitz, R. Maclin, Popular ensemble methods: an empirical study, J. Artif. Intel. Res. 11 (1999) 169–198.
[30] M.J. Orr, Regularization in the selection of radial basis function centers, Neural Comput. 7 (3) (1995) 606–623.
[31] J. Park, I.W. Sandberg, Universal approximation using radial basis function, Neural Comput. 3 (2) (1991) 246–257.
[32] M.P. Perrone, L.N. Cooper, When networks disagree: ensemble methods for hybrid neural networks, in: R.J. Mammone (Ed.), Neural Networks for Speech and Image Processing, Chapman and Hall, New York, 1993.
[33] P. Da-Gang Pan, K.P. Lu, Statistical forecast on precipitation over Taiwan area during typhoon invasion using GMS-5 data, in: Proceedings of the 12th Conference on Satellite Meteorology and Oceanography, Long Beach, CA, USA, February 8–12, 2003.
[34] Y. Tan, J. Wang, J.M. Zurada, Non-linear blind source separation using a radial basis function network, IEEE Trans. Neural Networks 12 (1) (2001) 124–134.
[35] D.S. Wilks, Statistical Methods in the Atmospheric Sciences, Academic Press, 1995.
[36] D. Wolpert, Stacked generalization, Neural Networks 5 (1992) 241–259.
[37] L. Xu, A. Krzyzak, C.Y. Suen, Methods of combining classifiers and their applications to handwriting recognition, IEEE Trans. Syst. Man Cybern. 22 (3) (1992) 79–101.
[38] J.M. Zurada, Introduction to Artificial Neural Systems, West Publishing, 1992.
[39] A. Abraham, B. Nath, A neuro-fuzzy approach for forecasting electricity demand in Victoria, Appl. Soft Comput. J. 1/2 (2001) 127–138.
[40] S. Mukkamala, A. Sung, A. Abraham, Intrusion detection using ensemble of soft computing and hard computing paradigms, J. Network Comput. Appl. 28 (2) (2005) 167–182.
[41] I. Maqsood, M.R. Khan, A. Abraham, Neural Network Ensemble Method for Weather Forecasting, Neural Computing and Applications, vol. 13(2), Springer Verlag London Ltd., 2004, pp. 112–122.
[42] S. Chebrolu, A. Abraham, J. Thomas, F. Deduction, Feature deduction and ensemble design of intrusion detection systems, Comput. Secur. 24 (4) (2005) 295–307.
[43] A. Abraham, C. Grosan, S.Y. Han, A. Gelbukh, Evolutionary multi-objective optimization approach for evolving ensemble of intelligent paradigms for stock market modeling, in: A. Gelbukh (Ed.), Proceedings of the Fourth Mexican International Conference on Artificial Intelligence, Mexico, Lecture Notes in Computer Science, Springer Verlag, Germany, 2005, pp. 673–681.
[44] A. Abraham, A. AuYeung, Integrating ensemble of intelligent systems for modeling stock indices, in: J. Mira, J.R. Alverez (Eds.), Proceedings of the Seventh International Work Conference on Artificial and Natural Neural Networks, Lecture Notes in Computer Science, vol. 2687, Springer Verlag, Germany, 2003, pp. 774–781.
[45] A. Jain, A.M. Kumar, Hybrid neural network models for hydrologic time series forecasting, Appl. Soft Comput. 7 (2007) 585–592.
[46] J.N.K. Liu, R.W.M. Kwong, Automatic extraction and identification of chart patterns towards financial forecast, Appl. Soft Comput., in press. http://www.dx.doi.org/10.1016/j.asoc.2006.01.007.
[47] L. Aburto, R. Weber, Improved supply chain management based on hybrid demand forecasts, Appl. Soft Comput. 7 (2007) 136–144.
[48] M.L.M. Lopes, C.R. Minussi, A.D.P. Lotufo, Electric load forecasting using a fuzzy ART&ARTMAP neural network, Appl. Soft Comput. 5 (2) (2005) 235–244.
[49] T. Chen, A fuzzy back propagation network for output time prediction in a wafer fab, Appl. Soft Comput. 2 (3) (2003) 211–222.
[50] A. Abraham, Meta-learning evolutionary artificial neural networks, Neurocomput. J. 56c (2004) 1–38.