AN ARTIFICIAL NEURAL NETWORK METHOD TO FORECAST ELECTRIC LOAD

Acad. **Paul Dan CRISTEA**, PhD, Prof. **Rodica TUDUCE**, PhD, Assoc. Prof. **Dumitru Iulian NĂSTAC**, PhD

University Politehnica of Bucharest, Biomedical Engineering Center

REZUMAT. Lucrarea prezintă un sistem care utilizează o rețea neurală (RN) pentru învățarea eficientă a unei secvențe nestaționare. Sistemul are potențialul de a învăța, clasifica și prezice orice semnale staționare sau nestaționare, în particular curbele de variație în timp a sarcinii din sistemele electro-energetice, la diferite scale de timp. O mare varietate de aplicații de clasificare și predicție a secvențelor nestaționare pot fi rezolvate într-un mod eficient utilizând această tehnică de reantrenare adaptivă combinată cu arhitectura în două trepte a sistemului.

Cuvinte cheie: predicția curbelor de sarcină, rețele neurale artificiale, analiza componentelor principale, reantrenare

ABSTRACT. The paper presents a system using an artificial neural network (ANN) to efficiently learn the shape of a non stationary sequence. The system has the potential to learn, classify and predict any stationary or non stationary signals, in particular the chronological load curves of power systems, at various time scales. A large variety of classification and forecasting applications to non stationary sequences can be efficiently solved by using this adaptive retraining technique combined with the two step system architecture.

Key words: load forecasting, Artificial Neural Networks, Principal Component Analysis, retraining technique

1. INTRODUCTION

A safe short-term load forecasting is needed for the efficient operation of power systems. There is a considerable interest in the development of reliable forecasting models. During the last decades we found in the literature a series of so called classical approaches [1] - [3] together with an increasing number of other methods and techniques based on artificial intelligence models [4] - [16]. The first applications of artificial neural networks (ANNs) to short-term load forecasting were in the beginning of 1990s [17], [18] and since then many different applications of ANNs to load forecasting were presented.

ANNs have been widely applied to forecasting problems [19] - [26] as long as they are less susceptible to the problem of misspecification as compared to most parametric models. ANNs are also more noise tolerant, having the ability to learn complex systems with incomplete or corrupted data. In this paper, the electric load forecasting is inherently noisy and nonstationary. The nonstationary characteristic implies that the distribution of the time series changes over time. Furthermore, some gradual changes in the dependency between the input and output variables may appear. In other words, the recent data points could provide more important information than the distant data points. Therefore, we propose to use the adaptive retraining mechanism [27] to take this characteristic into account and prove its worth on the electric load forecasting.

This paper is organized as follows. Section 2 presents the problem that concerns the model structure and data preprocessing. In the next section, we introduce the adaptive retraining technique and explain our approach. The main features of the experimental results are given in sections 4. Finally, Section 5 concludes the paper.

2. GENERAL STRUCTURE APPROACH

Usually, time-delays are frequently encountered in financial or technical systems. It is well known that feedback control in the presence of time/spatial delay leads to particular difficulties, since a delay places a limit on the time/spatial interval.

In order to cover a wide number of possible applications, we start to consider a system with s inputs and u outputs. In Fig. 1 we present our idea of training a feedforward ANN such that the latter becomes a predictor. We use delayed rows of the input data to simulate the current states. For learning purposes, the network inputs involve many blocks with several time-delayed values of the system inputs, and fewer blocks with system delayed outputs. Each ANN target-output consists of the current value of the corresponding sequence. Therefore, the system tries to match the current values of the output, by properly adjusting a function of the past values of the inputs and outputs.

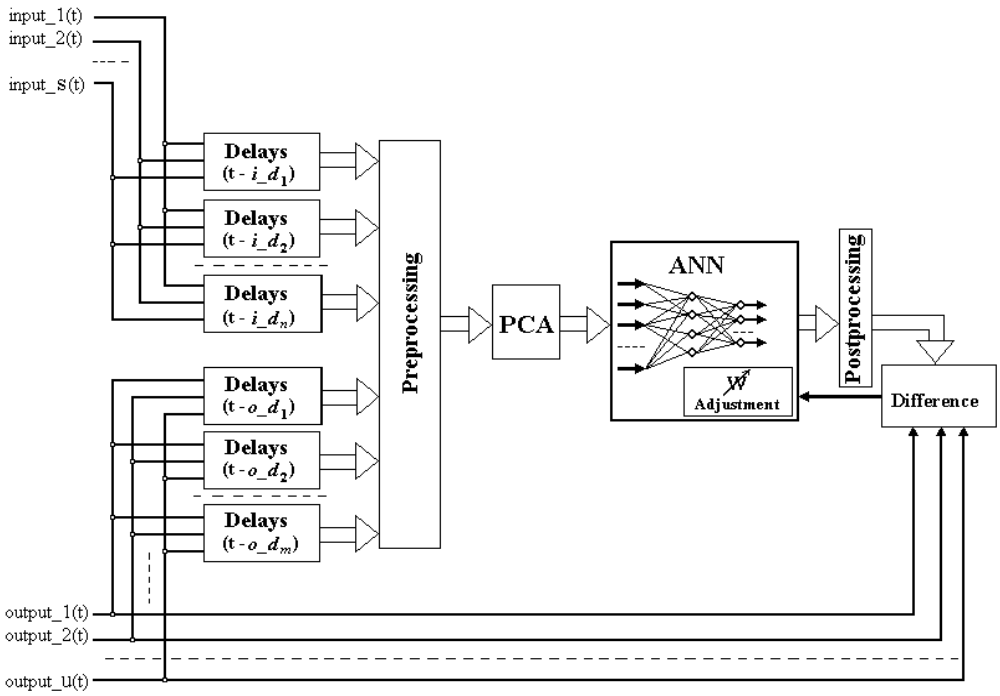

Fig. 1. Forecasting architecture. Training process.

At the current moment t, the output *output_k* (k = 1..u, Fig. 1) is affected by the inputs at different previous time/space steps (*t-i_d1, …, t-i_dn*), and also by the outputs at other previous time/space steps (*t-o_d1, …, t-o_dm*), respectively. We denote by so called *In_Del* and *Out_Del*, two *delay vectors* that include the delays that we take into account:

$$In\_Del = [i\_d_1, i\_d_2, ..., i\_d_n] \quad (1)$$

and

$$Out\_Del = [o\_d_1, o\_d_2, ..., o\_d_m] \quad (2)$$

where typically n > m.

The distribution of the vector elements is usually (but not compulsory) similar to the Gamma distribution. The elements of each vector are ascendingly ordered. Consequently, the maximum values of any delay vector are $\boldsymbol{i\_d_n}$ or $\boldsymbol{o\_d_m}$, respectively. Usually, $\boldsymbol{i\_d_n}$ significantly exceeds $\boldsymbol{o\_d_m}$.

The recurrent relation performed by the model that predicts the *output_k* is as follows:

$$y_k(t+1) = F(X(t+1-In\_Del(i)), Y(t-Out\_Del(j))) \quad (3)$$

where *X* is the input vector, Y the corresponding output vector, y_k the value of output_k, $i = 1,\ldots,n$ and $j = 1,\ldots,m$.

We use feedforward ANNs with two hidden layers in order to achieve a good approximation function, based on our preliminary research, where we have obtained better results in case of two hidden layers than in case of one hidden layer, however maintaining a similar ratio (approx. 5/1) between the number of the training samples and the total number of the weights. The ANN models, depicted in Fig. 1, use training sets of V-i_d_n input-output pairs for model adaptation (see next section), where V is the initial time steps interval employed for training purpose.

Once we have established all the influences on the output, at moment t, we apply Principal Component Analysis (PCA) [24], [28], [29] to reduce the dimensionality of the input space and to un-correlate the inputs. Before applying PCA, we had preprocessed the inputs and outputs, by using: replacement of missing values (NaN); detection and replacement (peak-shaving) of outliers; and, finally, normalization. Data preprocessing prepares raw data for the forecasting model and transforms it into a format that will be easier and more effectively processed. We have applied the reverse process of normalization, in order to denormalize the simulated outputs. Data preprocessing and data postprocessing are essential steps of the knowledge discovery process, in real world applications, and they greatly improve the network's ability to capture valuable information, if they are correctly carried out.

3. METHOD DESCRIPTION

The feature of "universal functional approximator" [30] adds the power and flexibility of neural networks to the process of learning complex patterns and relationships. However, the potential risk of using the universal approximator is the overfitting problem, since it is often easy to train a large network model to learn the peculiarities, as well as the underlying relationship. Therefore, the balance between the learning capability and the generalization power is very important in neural network forecasting applications.

As the basic training algorithm, we use the Scale Conjugate Gradient (SCG) algorithm [31]. In order to avoid the overfitting phenomenon, we apply the early stopping method (validation stop) during the training process.

Next, the adaptivity of the result is performed (and improved), by applying the retraining technique [27], [32], in a special way. This technique is a mechanism for extracting practical information directly from the weights of a reference ANN that had been already trained in a preliminary phase. The retraining procedure reduces the reference network weights (and biases) by a scaling factor γ, $0<\gamma<1$. The reduced weights are used further as the initial weights of a new training sequence, with the expectation of a better accuracy. The data that we have used in our model consist of V-i_d_n input-output pairs during each training (or retraining) phase, where V is the initial time/spatial steps interval employed for training purpose. As the splitting criterion, we randomly choose approximately 85% of the data (V-i_d_n) for training set, and the rest for validation. Furthermore, we imposed the supplementary condition:

$$E_{val} \leq \text{threshold} \cdot E_{tr} \tag{4}$$

to avoid a large difference between the error of the training set (E_{tr}) and the error of the validation set (E_{val}). The threshold is set so that the validation error does not exceed too much the training error (e.g. with no more than 20% extra). This way, the overfitting phenomenon on the test set will be considerably reduced. In our approach the validation set acts at the same time as a kind of test set, even though there is a real and separate test set of T different timesteps (where $T<<V$).

Next, we describe the five steps that we have taken to adapt (retrain) our model:

1. Firstly, we decided the proper number of hidden neurons for each hidden layer (N_{h1} and N_{h2}) by testing several pyramidal ANN architectures. We chose the best model with respect to the smallest error between the desired and the simulated outputs. This error (E_{tot}) was calculated for V-i_d_n data that included both training and validation sets.

2. Secondly, we predicted T values of the outputs (during the interval *(V+1) - (V+T)*), in a sequential mode. We use an *always real inputs* (ARI) approach [23], which employed the real previous outputs at the input of the model.

Then, we computed the error ERR [27] that represents the accuracy of the approximation of the output data, within the forecasting horizon of T steps:

$$ERR = \frac{100}{T} \sum_{p=1}^{T} \frac{\left| O_{Rkp} - O_{Fkp} \right|}{\left| O_{Rkp} \right|} \cdot \frac{T}{T+p} \tag{5}$$

where T = number of times/space steps, O_{Rkp} = real *output_k* at step p, and O_{Fkp} = forecasted *output_k* at step p.

3. Thirdly, we applied the retraining technique for a shifted interval of timesteps between (*Shift*+1) and (*Shift*+V), where *Shift* $\leq T$. Here we used the ANN architecture that resulted at the end of the previous step. We applied this technique for each value of γ (γ = 0.1, 0.2, …, 0.9), keeping the neural network (weight distribution) that achieved the minimum error, as the reference network. We repeated this step five times, and we randomly reconstructed the training and validation sets each time.

4. Fourthly, we predicted T values of the outputs (during the interval of timesteps (*Shift+V+1*) - (*Shift+V+T*)), in the same sequential mode as in step 2.

5. We repeated L times the steps 3 and 4 on successive shifted intervals of V timesteps for retraining processes and T timesteps for sequential forecasting. Each time the intervals were ascendingly repositioned with *Shift* time/space steps.

The retraining technique allows us to continuously improve the model, at times, by using new (shifted) databases. For a single combination of the delay vectors, we obtained (and used) a model with its associate adaptive behavior. The next two sections show how the above-mentioned steps were applied for a short-term load forecasting application.

4. EXPERIMENTAL RESULTS

Usually, short-term load forecasting aims to predict electric loads for a period of minutes, hours, days or weeks. The particular system, which resulted by using our approaches, describes the relationship between ten input variables and one output variable that models the evolution of the electric load. The raw data consist of more than 30,000 rows (time steps) – one data row every hour during 4 years (from January 2008 till December 2011). We performed the steps described in Section 3 for two combinations of delay vectors. The inputs of the system are information concerning data from different suppliers of electricity (coal, hydro, oil, nuclear, wind, import) together with dates (hour, month calendar, year). The goal is to predict energy consumption in two cases: in advance of one hour and six hours in advance.

In Table 1 we present the values of test error (ERR according to (5)) for iterative simulations of the output, computed at the end of first training and, then, after each successive retraining phase (L = 149), when using Case I (one hour in advance when using: *In_Del* = [1 2 4 8] and *Out_Del* = [0 1 4]) and Case II (six hours in advance when using: *In_Del* = [6 7 9 13] and *Out_Del* = [5 6 9]).

We carried out the simulations under the following assumptions: V = 8805 timesteps (more than one year) are enough for first training phase and then for each retraining phase; $\boldsymbol{T}$=24×7=168 hours (one week) represent the prediction interval; and *Shift* = 168 is the shifting time for the next retraining. It is worth to mention that the values of these parameters can be easily changed. Choosing the number of samples for training is an open issue: not too small to have enough data (more than five times the number of samples versus the number of weights), but not too large especially in a nonstationary environment.

The first step, when we choose the ANN architecture, needs about one day of intensive computation. We used a series of multiple imbricate loops, where we vary the number of neurons of each layer. If we extend the searching possibilities, then the computational effort will increase accordingly, but with a greater expectation for a better solution. The most inner loop of this searching algorithm concerns the starting of the training from different uniformly distributed weights of the ANN architecture. We choose to restart five times the last loop for each ANN architecture, and finally select the one that prove the minimum error and also respect the condition (4). This ANN is used to predict the first time horizon.

The obtained architecture at the first step remains unchanged and it is successively retrained with one week shifted database. Each retraining phase uses nine values of the scaling factor ($\gamma = 0.1 \div 0.9$) to recreate the initial values of the weights for the learning process, and it is repeated five times before predicting the corresponding time horizon. A complete retraining phase lasts about half an hour.

Table 1.

Evolution of Test Error (ERR) for the Case I and Case II during the training and retraining phases when T=24×7=168 hours (one week)

			ERR	
	Training / retraining interval	**Test interval**	**Case I**	**Case II**
First training	1 - 8805	8806 - 8973	1.8654	3.4278
Retraining 1	169 - 8973	8974 - 9141	8.1743	2.2377
Retraining 2	337 - 9141	9142 - 9309	7.9477	3.5993
Retraining 3	505 - 9309	9310 - 9477	7.7424	2.3580
Retraining 4	673 - 9477	9478 - 9645	8.2210	8.2234
…				
Retraining 145	24361 - 33165	33166 - 33333	1.7575	2.6037
Retraining 146	24529 - 33333	33334 - 33501	0.9210	1.8045
Retraining 147	24697 - 33501	33502 - 33669	1.0063	2.8682
Retraining 148	24865 - 33669	33670 - 33837	0.9877	1.8759
Retraining 149	25033 - 33837	33838 - 34005	0.9112	1.6272

In Table 1, it seems that the delay vectors have been well chosen, since, finally, there has been a decrease in the test error, when successive retraining phases were performed. An example of the error trends is showed in Fig. 2 for the Case I and in Fig. 3 for the Case II. One may note that the abscissa represents the numbers of the successive retraining phases and the first value 0 is associated with the first training.

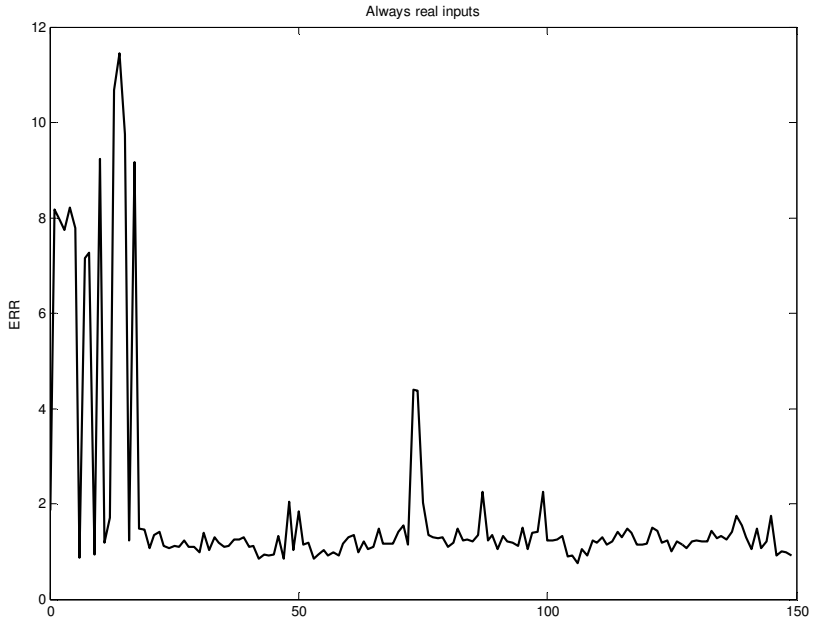

Fig. 2. ERR trend (Case I) of test sets for the first training and L = 149 successive retraining phases.

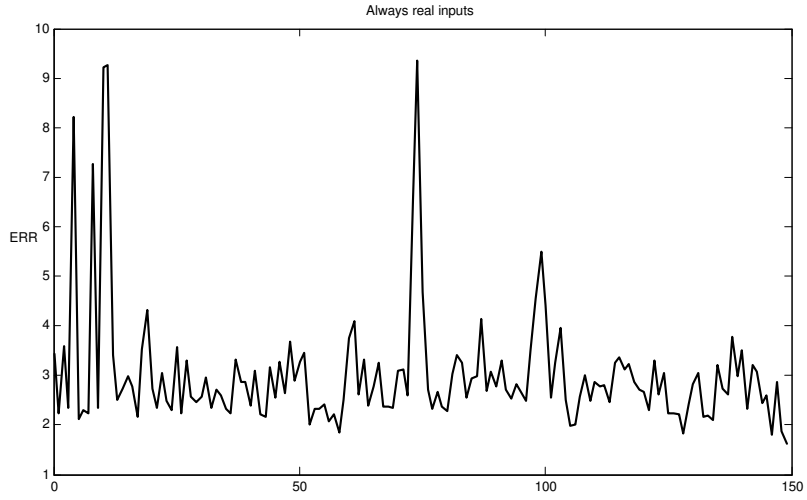

Fig. 3. ERR trend (Case II) of test sets for the first training and L = 149 successive retraining phases.

The quality of the predictions is graphically analyzed (Fig. 4), by enforcing a tube around the real outputs, given by a function like the one below:

$$f(n) = A + n \cdot q \tag{6}$$

Here, A is an acceptable prediction error, q is an increasing factor and n is the number of predicted timesteps. The predicted output values should lay in the interval *output(n) +/- f(n)*, represented with dotted lines in Figure 4 that shows the graphs of the energy consumption for the test interval of retraining 106. The real data are represented with thin lines and the neural network output values with thick lines. There is a "tube" (dotted lines) around the real data, given by the function f(n)=200+0.0005·n (where n = 1 … 168). The abscissa shows the index number of the corresponding lines in the database when predictions are performed.

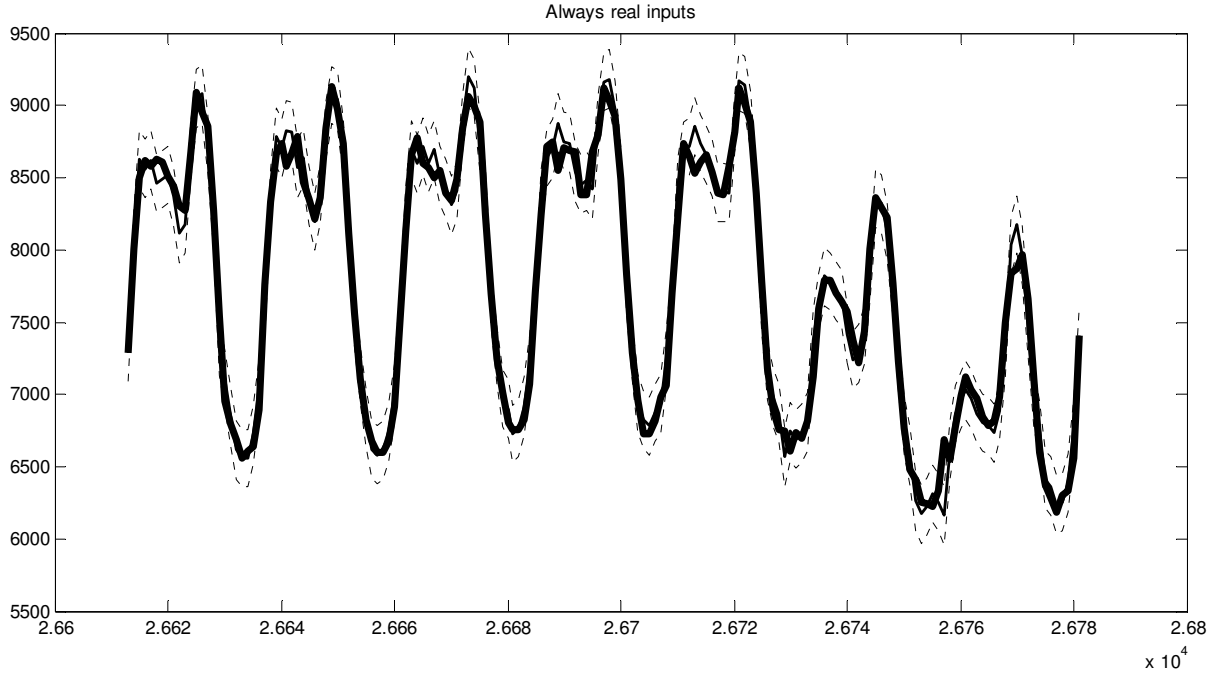

Fig. 4. Data forecasting for the test interval of retraining 106 (Case I).

ERR = 0.7671

We noticed that, except for a few cases, in almost all the graph representations of the predictions the trends were well captured (even outside the "tube") by using our approach.

5. CONCLUSIONS

The paper presents an adaptive predictive system that can be used for non-stationary sequences in time or space domains. It is crucial to start with an optimal system in what concerns the architecture and the choice of delay vectors. This is the reason that our approach must take a relatively long time in the first phase, for searching among many possible solutions. The updating mechanism is the adaptive retraining technique, which has to be performed periodically (at intervals that depend on the application). The SCG algorithm has been used for (re)training the ANN, even if it is not the fastest one. Its great advantage is that it works very efficiently for networks with a large number of weights, does not require large computational memory, has a good convergence and is very robust. Furthermore, as we always use the early stopping method (validation stop) during the training, it is better to avoid algorithms that converge too rapidly, such as Levenberg-Marquardt (LM). The SCG is well suited for validation stop method. Nevertheless, it is quite easy to replace the SCG algorithm with another one, since the adaptive retraining technique is flexible and independent of the basic training algorithm.

The ANNs ability to extract significant information, from its training data, provides a valuable framework for the representation of relationships that are present in the structure of the data. This allows both the interpolation among the a priori defined points, as well as the extrapolation outside the range bordered by the extreme points of the training set.

The evaluation of test error shows that the adaptive retraining technique can gradually improve, on average, the achieved results. There was a clear difference between the first training process, which needed a long time to search for the best architecture, and the retraining on the other hand. It can be quite easy to retrain a good ANN architecture, several times, by using a shifted training set. The great advantage of the retraining technique is that some relevant aspects are preserved (remembered) not only from the immediate previous training phase, but also from the previous but one phase, and so on. A kind of slow forgetting process also occurs, thus it is much easier for the ANN to remember specific aspects of the previous training instead of the first training. This means that the old information accumulated during the older trainings will be slowly forgotten and the learning process will be adapted to the newest evolutions of the process.

Current research targets to analyze other combinations of delay vectors and theirs effect on the short-term load forecasting.

ACKNOWLEDGEMENTS

We are grateful to Prof. Paul Ulmeanu for the data and helpful discussions. The work was supported by the Sectorial Operational Programme Human Resources Development (SOP HRD), financed from the European Social Fund and the Romanian Government under the contract number SOP HRD/89/1.5/S/59758.

REFERENCES

[1] **Taylor**, **J.W.,** *Short-Term Load Forecasting With Exponentially Weighted Methods*, IEEE Transactions on Power Systems, vol. 27 , no. 1, pp. 458 – 464, 2012.

[2] **Huang, S.J., Shih, K.R.,** *Short-term load forecasting via ARMA model identification including non-Gaussian process considerations*, Power Systems, IEEE Transactions on, Volume: 18, Issue: 2, pp. 673 - 679, 2003.

[3] **Amjady, N.,** *Short-term hourly load forecasting using time-series modeling with peak load estimation capability*, Power Systems, IEEE Transactions on, Volume: 16, Issue: 3, pp. 498 - 505, 2001.

[4] **Catalao, J.P.S., Pousinho, H.M.I., Mendes, V.M.F.,** *Hybrid intelligent approach for short-term wind power forecasting in Portugal*, IET Renew. Power Gener., vol. 5, no. 3, pp. 251–257, 2011.

[5] **De Felice, M., Yao, X.,** *Short-Term Load Forecasting with Neural Network Ensembles: A Comparative Study*, IEEE Computational Intelligence Magazine, Volume: 6 , Issue: 3, pp. 47 – 56, 2011.

[6] **Ruta, D., Gabrys, B., Lemke, C.,** *A Generic Multilevel Architecture for Time Series Prediction*, IEEE Transactions on Knowledge and Data Engineering, Vol. 23 , no. 3, pp. 350 – 359, 2011.

[7] **Nose-Filho, K., Lotufo, A.D.P., Minussi, C.R.,** *Short-Term Multinodal Load Forecasting Using a Modified General Regression Neural Network*, IEEE Transactions on Power Delivery, Vol. 26 , no. 4, pp. 2862 – 2869, 2011.

[8] **Anvari Moghaddam, A., Seifi, A.R.,** *Study of forecasting renewable energies in smart grids using linear predictive filters and neural networks*, IET Renewable Power Generation, Vol. 5 , Issue: 6, pp. 470 – 480, 2011.

[9] **Khosravi, A., Nahavandi, S., Creighton, D.,** *Construction of Optimal Prediction Intervals for Load Forecasting Problems*, IEEE Transactions on Power Systems, Vol. 25 , no. 3, pp. 1496 – 1503, 2010.

[10] **Wu, L., Shahidehpour, M.,** *A Hybrid Model for Day-Ahead Price Forecasting*, IEEE Transactions on Power Systems, Vol. 25, no. 3, pp. 1519 – 1530, 2010.

[11] **Areekul, P., Senjyu, T., Toyama, H., Yona, A.,** *A Hybrid ARIMA and Neural Network Model for Short-Term Price Forecasting in Deregulated Market*, IEEE Transactions on Power Systems, Vol. 25 , no. 1, pp. 524 – 530, 2010.

[12] **Fuchs, E., Gruber, C., Reitmaier, T., Sick, B.,** *Processing Short-Term and Long-Term Information With a Combination of Polynomial Approximation Techniques and Time-Delay Neural Networks*, IEEE Transactions on Neural Networks, Vol. 20 , no. 9, pp. 1450 – 1462, 2009.

[13] **Qi, M., Zhang, G.P.,** *Trend Time–Series Modeling and Forecasting With Neural Networks*, IEEE Transactions on Neural Networks, Vol. 19 , no. 5, pp. 808 – 816, 2008.

[14] **Alves da Silva, A.P., Ferreira, V.H., Velasquez, R.M.G.,** *Input space to neural network based load forecasters*, International Journal of Forecasting, Volume 24, Issue 4, October-December 2008, Pages 616-629.

[15] **Senjyu, T., Takara, H., Uezato, K., Funabashi, T.,** *One-hour-ahead load forecasting using neural network*, IEEE Transactions on Power Systems, Vol. 17 , no. 1, pp. 113 – 118, 2002.

[16] **Hippert, H.S., Pedreira, C.E., Souza, R.C.,** *Neural networks for short-term load forecasting: a review and evaluation*, IEEE Transactions on Power Systems, Vol. 16 , no. 1, pp. 44 – 55, 2001.

[17] **Peng, T., Hubele, N., Karady, G.,** *An adaptive neural network approach to one-week ahead load forecasting*, IEEE Trans. Power Syst., vol. 8, no. 3, pp. 1195–1203, 1993.

[18] **Djukanovic, M., Babic, B., Sobajic, D., Pao, Y.-H.,** *Unsupervised/supervised learning concept for 24-house load forecasting*" IEE Proc. Gener., Trans. Distrib., vol. 140, no. 4, pp. 311–318, 1993.

[19] **Huang, J.Q., Lewis, F.L.,** *Neural-network predictive control for nonlinear dynamic systems with time-delay*, IEEE Transactions on Neural Networks, vol. 14, pp. 377-389, 2003.

[20] **Costea, A., Nastac, I.,** *Assessing the Predictive Performance of ANN Classifiers Based on Different Data Preprocessing Methods*, Internat. Journal of Intelligent Sys. Acc. Fin. Mgmt. vol. 13, no. 4, pp. 217-250, 2005.

[21] **Nastac, D.I., Cristea, P.,** *DNA Sequences Forecasting using an Adaptive Retraining Procedure*, Proceedings of International Workshop on Genomic Signal Processing (GSP 2005), Bucharest, 11-13 July 2005, pp. 121-126, 2005.

[23] **Nastac, I., Dobrescu, E., Pelinescu, E.,** *Neuro-Adaptive Model for Financial Forecasting*, Romanian Journal of Economic Forecasting, Vol. 4, No. 3, pp. 19-41, 2007

[24] **Sousa, S.I.V., Martins, F.G., Alvim-Ferraz, M.C.M., Pereira, M.C.,** *Multiple linear regression and artificial neural networks based on principal components to predict ozone concentrations*, Environmental Modelling & Software, Volume 22, Issue 1, Pages 97-103, 2007.

[25] **Pino, R., Parrenoa, J., Gomeza, A., Priore, P.,** *Forecasting next-day price of electricity in the Spanish energy market using artificial neural networks*, Engineering Applications of Artificial Intelligence, Volume 21, Issue 1, pp. 53-62, 2008.

[26] **Nastac, D. I., Cristea, P.D.,** *An ANN - PCA Adaptive Forecasting Model*, in Proceedings of IWSSIP 2012, Vienna, Austria, 11-13 April 2012, pp. 532-535, 2012

[27] **Nastac, D.I.,** *An Adaptive Retraining Technique to Predict the Critical Process Variables*, TUCS Technical Report, No. 616, June 2004, Turku, Finland. [Online] Available: www.tucs.fi/research/series/serie.php?type=techreport&year=2004

[28] **Jolliffe, I.,** *Principal component analysis*, 2nd edition, Springer, NY, 2002

[29] **Jackson, J.E.,** *A user guide to principal components*, John Wiley, New York, 1991.

[30] **Hornik, K., Stinchcombe, M., White, H.,** *Multilayer feedforward networks are universal approximators*, Neural Networks, vol.2, pp. 359-366, 1989.

[31] **Moller, M.F.,** *A scaled conjugate gradient algorithm for fast supervised learning*, Neural Networks, vol. 6, pp. 525-533.

[32] **Nastac, D.I., Matei, R.,** *Fast retraining of artificial neural networks*, in Rough Sets, Fuzzy Sets, Data Mining and Granular Computing, Wang et al. (Eds.), Springer-Verlag in the series of Lecture Notes in Artificial Intelligence (LNAI 2639), pp. 458-462, 2003.

About the authors

Acad. **Paul Dan CRISTEA**, PhD
University Politehnica of Bucharest
Biomedical Engineering Center
email: pcristea@dsp.pub.ro

Paul Dan Cristea graduated the Faculty of Electronics and Telecommunications (UPB - University "Politehnica" of Bucharest, 1962) and the Faculty of Physics (University of Bucharest, 1969). He obtained a Ph.D. in Technical Physics (PUB, 1970). Since then his research and teaching activities covered an extended area of Electrical Engineering and related domains including topics like Digital Signal and Image Processing, Genomic Signals, Neural and Evolutionary Systems, Computerized Medical Equipment, Evolutionary Intelligent Agents, Intelligent e-Learning Environments. He is member of the Romanian Academy, the most prestigious recognition institution of Romania, Member of Honor of the Romanian Scientists Academy, Fellow IEEE, director of the Bio-Medical Engineering Center of UPB and director of the Romanian Bioinformatics Society.

Prof. **Rodica TUDUCE**, PhD
University Politehnica of Bucharest
Biomedical Engineering Center
email: trodica@dsp.pub.ro

Rodica Tuduce is Professor of Electrical Engineering and Applied Computer Science at the University “Politehnica” of Bucharest (UPB), Faculty of Engineering in Foreign Languages. Rodica Tuduce graduated the Faculty of Electronics and Telecommunications (UPB) in 1965, and she holds a Ph.D. in Electrical Engineering from UPB, 1983. Her research and teaching activities cover an extended area of Electrical Engineering and related domains, including topics like Signal Theory, Digital Signal Processing and Economic Data and Signal Processing and Biological Signal Processing.

Assoc. Prof. **Dumitru Iulian NĂSTAC**, PhD
University Politehnica of Bucharest
Biomedical Engineering Center
email: nastac@ieee.org

Dumitru Iulian Nastac received the M.Sc. degree in electronics in 1993 and the Ph.D. degree in 2000, both from the University “Politehnica” of Bucharest, Romania. He was with the Turku Centre for Computer Science, Finland, as a Post-doc Researcher from 2002 to 2004. His research interests are in neural networks, adaptive systems and data forecasting. He has published four books as author and co-author, and over 40 papers in journals and conferences. He was one of the winners of the International Competition on Artificial Intelligence EUNITE 2003.