2014 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)
July 6-11, 2014, Beijing, China

A Comparison of Computational Intelligence Techniques for Energy Time Series Forecasting

Abbas Namdar / MBA & Data miner
Department of Economical and Administrative Science
University of Mazandaran
Babolsar, Iran
a.namdar@umz.ac.ir

Hamid Berenji / IEEE Fellow
Intelligence Inference Systems Corp
MS: 566/108, NASA Research Park
Moffett Field, California, 94035
berenji@iiscorp.com

***Abstract*— Energy time series forecasting plays a crucial role in the process of energy planning. This topic has been, and is still attracting vast research activities that are performed by researchers in the academia and energy companies. Various techniques exist for energy time series forecasting, and the selection of the most suitable forecasting algorithm is not an easy process. For a clear application of such techniques in energy time series forecasting, there should be a clear distinction between these techniques. This paper compares the overall performance of the Time Delay Neural Network (TDNN), Neuro Fuzzy Inference System and Support Vector Regression (SVR). The efficiency of these techniques is compared in energy time series forecasting, and the performances of them are tested. The results of our analysis indicate that the Time Delay Neural network (TDNN) shows the best performances overall.**

Keywords—Comparison; Energy; Computational Intelligence; Forecasting; Time series.

I. INTRODUCTION

One of the important criteria of the economic development is energy consumption. Industries changes strongly affect energy consumption. Consumption of energy has increased with growing population and prominent standards of living [1],[2]. Energy Information Administration has forecasted that consumption of energy will raise by 49% as of 2007 to 2035 [3]. Energy consumption has considerably and definitely affected economic growth [4]. Consequently, an extremely accurate model for forecasting energy consumption must be developed. Many models and tools are currently available to forecast. The reason of forecasting is to meet prospect necessities, decrease unexpected cost and offer possible information to decision making [1]. Energy segment established huge interest from individuals and countries as it leads to comfortable living. With economic development energy consumption has become a life-sustaining index [5]. With regards to limitation of energy resources on earth, researchers focused on developing better approaches for predicting the future demand for energy to meet future supply, which will assist countries to plan their development activities appropriately, thus, avoiding under-or over-planning of future supply. If we are able to forecast energy consumption time series more accurately, we can make the world's resources allocate to a right place that avoids wasting natural resource. Hence, providing accurate forecasts of energy consumption time series is of crucial importance for decision-makers, such as investment planning. Based on such a model, energy decision makers can also execute an energy conservation strategy or apportion of a certain amount of energy.

There are several different approaches to time series modeling. Traditional statistical models including moving average, exponential smoothing, and ARIMA are linear in that predictions of the future values, Although linear regression is frequently utilized to forecast time-series data, but it is imprecise when observations are few or do not convince statistical assumptions [6].

Several techniques have evolved over the years for energy time series forecasting, and the use of computational intelligence models for forecasting is slowly beginning to replace traditional statistical methods, as the efficiency and faster convergence rates makes them more desirable.

Lee and Tong [7] claimed that the traditional linear time-series model (ARIMA) cannot easily be used to fit nonlinear time-series data and consequently they developed a heuristic model to improve the precision of residual series. There have been several studies comparing the performance of energy time series forecasting models. Some studies [7] have compared the forecasting accuracy of different models, they presented models more accurate than ANN in forecasting or classification problems [8],[9],[10]. Togun and Baysec found that genetic programming performs as well as ANN in forecasting energy consumption [10].

Many researchers have developed Neuro Fuzzy models to improve their forecasting precision. The Neuro Fuzzy model has been adopted in many forecasting studies, usually, real-world data sets are difficult to collect and data sets contain a few observations [11]. Some significant algorithms have been proposed to train the neuro fuzzy models. The pioneers, Takagi and Sugeno, presented an adaptive algorithm for their fuzzy inference system [12].

This work compares forecasting models which is applied to the same time-series data, to ensure high forecasting precision. In this paper, we compare strong computational intelligence techniques to forecast energy consumption time series. Energy

978-1-4799-2072-3/14/$31.00 ©2014 IEEE

consumption time series data of International Institute of Energy Studies was used to test models.

II. COMPUTATIONAL TECHNIQUES TO FORECAST

Different methods, including computational intelligence based approaches have been proposed for energy consumption time series forecasting. This section describes three models that are used in forecasting energy consumption.

A. Time Delay Neural Network

Time-delayed neural network architecture is used to process the time-varying input data. The first input is delayed sequentially by a time-delay component in each input phase of the network, therefore, the time-delay produces the customized learning-rule such that the connection-weight will change only if the time-delayed input and current output are activated rather than if the current input and current output are activated concurrently. In other words, the output is associated with the earlier input rather than the current input, which can be expressed mathematically as follows:

$$\Delta w(t) = x(t)y(t) \quad \textit{(1)}$$

The relationship between the input and output is given by:

$$y(t) = w(t)x(t) \quad \textit{(2)}$$

Architecture of the time-delayed neural network showing the relationships between the time-delayed input, $x(t-k\Delta t)$, connection-weights, $w_k(t,k\Delta t)$ and their output, $y_k(t)$.

Let $x(t)$ and $y(t)$ denote the input and output data at time t, correspondingly, and $w(t,\tau)$ denotes the connection-weight between them with a lag-time, τ, then the modified time-delayed learning-rule is given by:

$$\underline{\text{if}}\ x(t-\tau) \neq 0 \underline{\text{and}}\ y(t) \neq 0$$
$$\underline{\text{then}}\ \Delta w(t,\tau) \neq 0$$
$$\underline{\text{else}}\ \Delta w(t,\tau) = 0$$

Where $x(t-\tau)$ denotes the input data delayed by the lag-time, τ, and $\Delta w(t,\tau)$ denotes the change in connection-weight (or the weight-change). Therefore, a continuous-time, time-delayed learning-rule is given by extending Eq. 1:

$$\Delta w(t,\tau) = x(t-\tau)y(t) \quad (3)$$

For implementation, we use discrete lag-times (i.e., $\tau = k\Delta t$), in integral, k, multiples of Δt to delay the input data by multiple delay-tap lines, thus, the time-delayed associative learning-rule at the k-th delay-line is given by:

$$\Delta w_k(t,k\Delta t) = x(t-k\Delta t)y_k(t) \quad (4)$$

where k is an integer constant, $\Delta w_k(t,k\Delta t)$ is the change in the k-th connection-weight between the time-delayed input, $x(t-k\Delta t)$, and the k-th output, $y_k(t)$.

A time-series data is used as the input to the network. This time-delayed input is cascaded into multiple branches as inputs to successive neurons to provide the inputs for the modified learning-rule (Eq. 4) to update the corresponding connection-weights. The network would produce as many outputs as there are discrete time-delays. The k-th output of the network is resulted from:

$$y_k(t) = w_k(t,k\Delta t)x(t-k\Delta t) \quad (5)$$

Alternatively, each of the delay-tap lines can be considered as feeding into a pseudo-neuron as the first (pseudo) layer of the network. This first layer can be considered as a pseudo-layer for the network because it does not perform further calculation, except for conceptualization of the equivalent neural network architecture.

The output of the k-th time-delayed pseudo-input neuron (in the first pseudo-layer) can be expressed in terms of the initial input data by:

$$x_k(t) = x(t-k\Delta t) \quad (6)$$

The main cause why we represent the network in this equivalent architectural form is that now the layer of time-delayed inputs is a parallel layer rather than a cascaded sequential input layer. In other words, it transforms the data sequential time-series input into parallel inputs by the delay-lines, which allows for simultaneous parallel processing rather than sequential processing. This represents the spatiotemporal transformation of the input data clearly by the alternate network architecture, although they are equivalent absolutely.

Such a network would have a single sequential input, $x(t)$, branched into $(k+1)$ parallel lines by k discrete delays. It will also have k outputs, $y_k(t)$. The k-th output of the network is given by:

$$y_k(t) = \Delta w_k(t,k\Delta t)x_k(t) \quad (7)$$

These outputs can be further combined into a single output, $y(t)$, to form a network produces a single output data rather than multiple outputs. This results in the output of the network that computes the weighted-sum of all k time-delayed data mathematically:

$$\begin{aligned} y(t) &= \sum_{i=0}^{k} y_i(t) \\ &= \sum_{i=0}^{k} \Delta w_i(t,i\Delta t)x_i(t) \quad (8) \\ &= \sum_{i=0}^{k} \Delta w_i(t,i\Delta t)x(t-i\Delta t) \end{aligned}$$

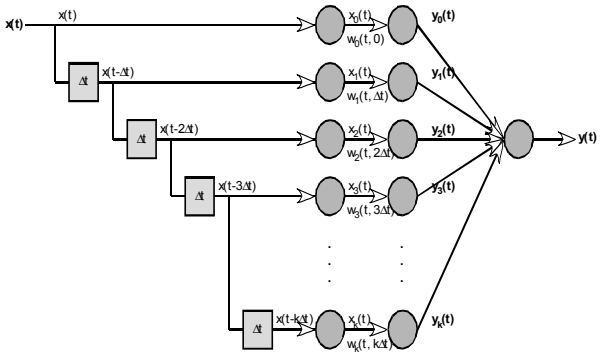

Fig. 1. Structure of the Time Delay Neural Network model

Figure one showing the architecture of the single-input and single-output network that computes the weight-sum of time-delayed input data.

Therefore, this network architecture will present a single input and a single output to process the time-series data using a pseudo-input layer. This design satisfies the key goal of creating a neural network that correlates two time-series data, $x(t)$ and $y(t)$, using a set of time-delayed associative learning-rules.

It will be shown below that the cross-correlation coefficients are computed by the weight-sum of the time-delayed inputs by the output neuron at the k-th connection-weight after consecutive iterative training.

When the network is trained with n iterations of the discrete time step, Δt, the resulting connection-weight is given by:

$$w_k(n\Delta t, k\Delta t) = w_k(0, k\Delta t) + \sum_{j=0}^{n} \Delta w_k(j\Delta t, k\Delta t) \quad (9)$$

for $t = n\Delta t$ and $\tau = k\Delta t$. The continuous-time time-delayed learning-rule of Eq. (3) can be re-expressed in terms of the discrete-time step (for $t = j\Delta t$) as:

$$\Delta w_k(j\Delta t, k\Delta t) = x(j\Delta t - k\Delta t)\, y(j\Delta t). \quad (10)$$

The resulting connection-weights after iterating n discrete time steps become:

$$w_k(n\Delta t, k\Delta t) = w_k(0, k\Delta t) + \sum_{j=0}^{n} x(j\Delta t - k\Delta t)\, y(j\Delta t) \quad (11)$$

B. Neuro Fuzzy Inference System

A neuro-fuzzy model is a fuzzy system drawn in a neural network structure, combining the learning, parallel processing and generalization capabilities of neural networks and logicality, transparency and use of a priori knowledge in fuzzy systems. The mathematical description of neuro fuzzy model which is the most general formulation will be described in this section. The fuzzy inference system is constructed by fuzzy rules of the following type:

$$Rule_i: \ If\ u_1 = A_{i1}\ And\ ...\ And\ u_p = A_{ip}\ then\ \hat{y} = f_i(u_1, u_2, ..., u_p) \quad (12)$$

Where $i = 1...M$ and M is the number of fuzzy rules. $u_1, ..., u_p$ are the inputs of network, each A_{ij} denotes the fuzzy set for input u_j in rule i and $f_i(.)$ is a crisp function which is defined as a linear combination of inputs in most applications

$$\hat{y} = \omega_{i0} + \omega_{i1}u_1 + \omega_{i2}u_2 + \ldots + \omega_{ip}u_p \quad (13)$$

Matrix form $\hat{y} = a^T(\underline{u}) \cdot W$

therefore the output of this model can be calculated by

$$\hat{y} = \frac{\sum_{i=1}^{M} f_i(\underline{u})\mu_i(\underline{u})}{\sum_{i=1}^{M} \mu_i(\underline{u})} \quad ; \quad \mu_i(\underline{u}) = \prod_{j=1}^{p} \mu_{ij}(u_j) \quad (14)$$

Where $\mu_{ij}(u_j)$ the membership function of jth is input in the ith rule and $\mu_i(\underline{u})$ is the degree of validity of the ith rule. This system can be formulated in the basis function realization. The basis function will be

$$\phi_i(\underline{u}) = \frac{\mu_i(\underline{u})}{\sum_{j=1}^{M} \mu_j(\underline{u})} \quad (15)$$

as a result

$$\sum_{j=1}^{M} \phi_j(\underline{u}) = 1 \quad (16)$$

This neuro fuzzy model has two sets of adaptable parameters; first the antecedent parameters, which belong to the input membership functions such as centers and deviations of Gaussians, second the rule resulting parameters such as the linear weights of output in equation (13). It is commonly to optimize only the rule resulting parameters. This can be simply done by linear techniques like least squares [2]. A linguistic interpretation to determine the antecedent parameters is usually sufficient. Nevertheless, one can opt to use a more powerful nonlinear method to optimize all parameters together. Gradient based learning algorithms can be used in the optimization of resulting linear parameters. Supervised learning is aimed to minimize the following loss function (mean square error of estimation):

$$J = \frac{1}{N}\sum_{i=1}^{N} (y(i) - \hat{y}(i))^2 \quad (17)$$

Where N is the number of data samples. According to the matrix form of (13) this loss function can be extended in the quadratic form

$$J = W^T R W - 2W^T P + Y^T Y / N \quad (18)$$

Where $R = (1/N)A^T A$ is the autocorrelation matrix, A is the $N \times p$ solution matrix whose ith row is $a(\underline{u}(i))$ and $P = (1/N)A^T y$ is the p dimensional cross correlation vector. From

$$\frac{\partial J}{\partial W} = 2RW - 2P = 0 \quad (19)$$

the following linear equations are obtained to minimize J:

$$RW = P \quad (20)$$

and W is simply defined by pseudo inverse calculation. One of the simplest local nonlinear optimization

techniques is the steepest descent. In this method the direction of changes in parameters will be opposite to the gradient of cost function

$$\Delta W(i) = -\frac{\partial J}{\partial W(i)} = 2P - 2RW(i) \quad (21)$$

and

$$W(i+1) = W(i) + \eta \cdot \Delta W(i) \quad (22)$$

Where η is the learning rate. In this paper, we have used a different approach from the usual neuro fuzzy method as used in [22] [23] [24]. The network structure of a neuro fuzzy model is illustrated by Figure 2.

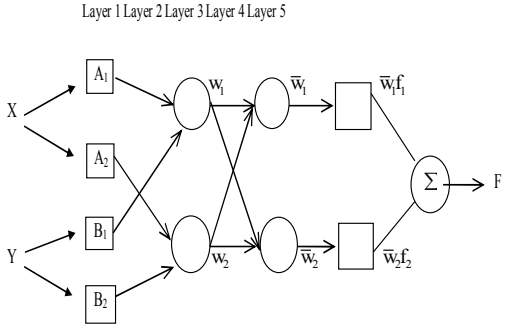

Fig. 2. Structure of the Neuro Fuzzy model

C. Support Vector Regression

Consider a set of training data $\{(x_1, y_1),....,(x_\ell, y_\ell)\}$, where each $x_i \subset R^n$ denotes the input space of the sample and has a corresponding target value $y_i \subset R$ for *i=1,..., l* where *l* corresponds to the size of the training data **Error! Reference source not found.**, [21]. The scheme of the regression problem is to establish a function that can estimated future values precisely.

The generic SVR estimating function takes the form:

$$f(x) = (w \cdot \Phi(x)) + b \quad (23)$$

where $w \subset R^n, b \subset R$ and Φ denotes a non-linear transformation from R^n to high dimensional space. Our aim is to find the value of w and b such that values of x can be determined by minimizing the regression risk:

$$R_{reg}(f) = C\sum_{i=0}^{\ell} \Gamma(f(xi) - yi) + \frac{1}{2}\|w\|^2 \quad (24)$$

where $\Gamma(\cdot)$ is a cost function, C is a constant and vector w can be written in terms of data points as:

$$w = \sum_{i=1}^{\ell} (\alpha_i - \alpha_i^*)\Phi(x_i) \quad (25)$$

By substituting equation (25) into equation (23), the generic equation can be rewritten as:

$$f(x) = \sum_{i=1}^{\ell} (\alpha_i - \alpha_i^*)(\Phi(x_i) \cdot \Phi(x)) + b$$

$$= \sum_{i=1}^{\ell} (\alpha_i - \alpha_i^*)k(x_i, x) + b \quad (26)$$

In equation (26) the dot product can be replaced with function $k(x_i, x)$, recognized as the kernel function. Kernel functions enable dot product to be performed in high-dimensional feature space using low dimensional space data input without knowing the transformation Φ. All kernel functions must satisfy Mercer's condition that corresponds to the inner product of some feature space. The radial basis function (RBF) is generally used as the kernel for regression:

$$k(x_i, x) = \exp\left\{-\gamma |x - x_i|^2\right\} \quad (27)$$

The ε-insensitive loss function is the most widely used cost function [21].The function is in the form:

$$\Gamma(f(x) - y) = \begin{cases} |f(x) - y| - \varepsilon, & for \quad |f(x) - y| \geq \varepsilon \\ 0 & otherwise \end{cases} \quad (28)$$

By solving the quadratic optimization problem in (29), the regression risk in equation (24) and the ε-insensitive loss function (28) can be minimized:

$$\frac{1}{2}\sum_{i,j=1}^{\ell} (\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j)k(x_i, x_j) - \sum_{i=1}^{\ell} \alpha_i^*(y_i - \varepsilon) - \alpha_i(y_i + \varepsilon)$$

subject to

$$\sum_{i=1}^{\ell} \alpha_i - \alpha_i^* = 0, \quad \alpha_i, \alpha_i^* \in [0, C] \quad (29)$$

The Lagrange multipliers, α_i *and* α_i^*, represent solutions to the above quadratic problem that act as forces pushing predictions towards target value y_i. Only the non-zero values of the Lagrange multipliers in equation (29) are useful in forecasting the regression line and are known as support vectors. For all points inside the ε-tube, the Lagrange multipliers equal to zero do not contribute to the regression function. Only if the requirement $|f(x) - y| \geq \varepsilon$ is fulfilled, Lagrange multipliers maybe non-zero values and used as support vectors.

The constant C introduced in equation (24) determines penalties to estimation errors. A large C assigns higher penalties to errors so that the regression is trained to minimize error with lower generalization while a small C assigns fewer penalties to errors; this allows the minimization of margin with errors, therefore superior generalization capability. If C goes to considerably large, SVR would not allow the happening of any error and result in a complex model, while C goes to zero, the result would tolerate a large amount of errors and the model would be less complex.

Now, we have solved the value of w in terms of the Lagrange multipliers. For the variable b, it can be computed by applying Karush-Kuhn-Tucker (KKT) conditions which, in this case, implies that the product of the Lagrange multipliers and constrains has to equal zero:

$$\alpha_i(\varepsilon + \zeta_i - y_i + (w, x_i) + b) = 0$$
$$\alpha_i^*(\varepsilon + \zeta_i^* + y_i - (w, x_i) - b) = 0 \quad (30)$$

and

$$(C-\alpha_i)\zeta_i = 0$$
$$(C-\alpha_i^*)\zeta_i^* = 0 \quad (31)$$

Where ζ_i and ζ_i^* are slack variables used to measure errors outside the ε-tube. Since $\alpha_i, \alpha_i^* = 0$ and $\zeta_i^* = 0$ for $\alpha_i^* \in (0, C)$, b can be computed as follows:

$$b = y_i - (w, x_i) - \varepsilon \quad for\ \alpha_i \in (0, C)$$
$$b = y_i - (w, x_i) + \varepsilon \quad for\ \alpha_i^* \in (0, C) \quad (32)$$

Putting it all together, we can use SVR without knowing the transformation.

III. EXPERIMENTAL RESULTS

In this section we compare models by using time series data of International Institute of Energy Studies (IIES) (the same dataset which were used by CIFE competition (2012)). The data set is made up of 8 energy time series.

A. Implement Models for Indexes Prediction

In the first stage, we normalize data over range, then, we implement TDNN and then NFIS also SVR to compare the best model with the least error.

B. Performance Analysis

For the purpose of evaluating accuracy of models, we will compare their outcomes with each other. The performance evaluations of the various techniques were based on their ability to forecast the true parameters accurately. Specifically, the evaluations of the proximity of estimated and true values were based on three performance measures. To carry out the comparison we use a common evaluation statistic called mean absolute percentage error (MAPE), root mean absolute error (RMSE) and total error (TE). For the first 5 time series, only the MAPE and RMSE, were computed and because of result similarity, one of them was chosen that is shown in table 1, while for the last 3 time series MAPE, RMSE and TE were considered.

$$MAPE = 100\frac{1}{N}\sum_{i=1}^{N} \left|Y_i - P_i\right|^2 \Big/ Y_i \quad (33)$$

$$TE = \frac{1}{5}\sum_{i=1}^{5}(MAPE) + \frac{1}{3}\sum_{i=1}^{3}(MAPE/D) \quad (34)$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(Y_i - P_i)^2} \quad (35)$$

Where Y_i is the actual value and P_i is the forecasted value of ith test data obtained from the models and N is the number of test data. Summary of evaluations in comparison of methods is shown in Table 1.

The first column specifies the type of model used. The remaining columns list the TE, MAPE and RMSE performance measures for the estimators in each technique. Regarding the computational results indicated in Table 1, TDNN has the best prediction accuracy of energy indexes and outperforms the other methods. So it can be considered as a promising technique for energy time series forecasting problem.

TABLE I. COMPARISON OF VARIOUS TECHNIQUES IN FORECASTING

Techniques Name	Specification	MAPE (Daily)	TE	RMSE
Time Delay Neural Network	36 neurons in hidden layer	0.48	12.92	5. 2
Adaptive Neuro Fuzzy Inference System	10 rules and 165 epochs	0.59	16.41	7.4
Support Vector Regression	linear kernel function with ε=0.01 and C=1000	0.63	17.95	8.9

The result of predicting the indexes of energy consumption by TDNN is presented in figures below.

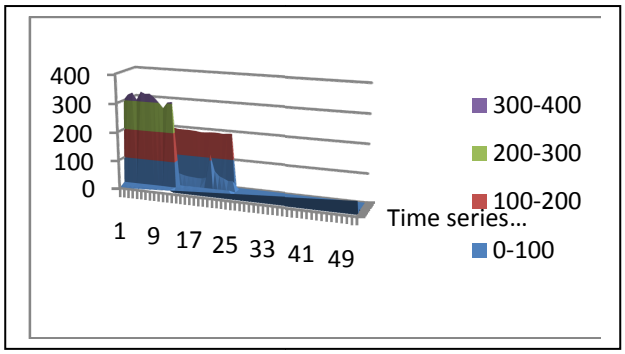

Fig. 3. Forecasting the Time Series using TDNN

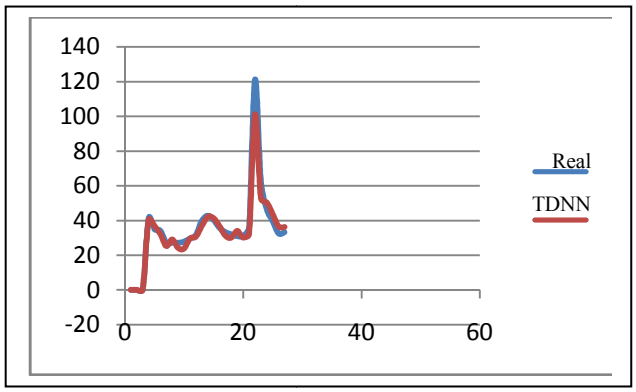

Fig. 4. Forecasting the Hourly Series using TDNN

IV. CONCLUSION

In this paper, we have experimentally compared the overall performance of the time delay neural network, neuro fuzzy and SVR, with each other in the context of energy consumption time series forecasting. There have been several studies comparing the performance of energy time series forecasting models. The use of computational intelligence models, specifically time delay neural network, because of the delicate use of time index in the calculation which entail the estimation of parameters by optimization procedure, is, yet, comparatively recent. The numerical experiments were performed and the

results of our analysis indicate that the time delay neural network model consistently outperforms the neuro fuzzy and SVR. As this paper showed, it is very crucial to use TDNN in all time series forecasting problems. The results of this paper can be extended to other forecasting problems, such as, the time series of stock exchange market and other economical time series.

ACKNOWLEDGMENT

The authors would like to thank from Miss. Salma Mousavi for editing and support. Next, we extend our thanks to International Institute of Energy Studies (IIES) for providing the data.

REFERENCES

[1] Ekonomou L. Greek long-term energy consumption prediction using artificial neural networks. Energy 2010;35:512–7.
[2] Pao HT. Forecasting energy consumption in Taiwan using hybrid nonlinear models. Energy 2009;34:1438–46.
[3] US Energy Information Administration. <http://www.eia.doe.gov/oiaf/ieo/highlights.html>.
[4] Lu W, Ma Y. Image of energy consumption of well off society in China. Energy Convers Manage 2004;45:1357–67.
[5] Li J, Dong X, Shangguan J, Hook M. Forecasting the growth of China's natural gas consumption. Energy 2011;36:1380–5.
[6] Lee Yi-Shian, Tong Lee-Ing. Forecasting nonlinear time series of energy consumption using a hybrid dynamic model. Applied Energy 2012;94:251–256.
[7] Lee YS, Tong LI. Forecasting time series using a methodology based on autogressive integrated moving average and genetic programming. Knowl- Based Syst 2011;24:66–72.
[8] Lee YS, Tong LI. Forecasting energy consumption using a grey model improved by incorporating genetic programming. Energy Convers Manage 2011;52:147–52.
[9] Ong CS, Huang JJ, Tzeng GH. Building credit scoring models using genetic programming. Expert Syst Appl 2005;29:41–7.
[10] Togun N, Baysec S. Genetic programming approach to predict torque and brake specific fuel consumption of a gasoline engine. Appl Energy 2010;87:3401–8.
[11] Jang J.R. (1993) ANFIS: Adaptive network based fuzzy inference system, IEEE Tran. On systems, Man and Cybernetics, 23(3), pp. 665-685.
[12] Takagi T., Sugeno M. (1985), Fuzzy identification of systems and its applications to modeling and control, IEEE Tran. On systems, Man and Cybernetics, vol. 15, pp. 116-132.
[13] D. C. Tam. A hybrid time-shifted neural network for analyzing biological neuronal spike trains. In: Progress in Neural Networks, (O. Omidvar, ed.) Ablex Publishing Corporation: Norwood, New Jersey, vol. 2, pp.129-146, 1994.
[14] D. C. Tam. Computation of cross-correlation function by a time-delayed neural network. In: Intelligent Engineering Systems through Artificial Neural Networks. (Cihan H. Dagli, Laura I. Burke, Benito R. Fernández, Joydeep Ghosh, eds.), American Society of Mechanical Engineers Press, New York, NY, vol. 3, pp. 51-55, 1993.
[15] Jazbi A., Lucas C. (1999) Intelligent control with emotional learning, 7th Iranian Conference on Electrical Engineering, ICEE'99, Tehran, Iran, pp. 207-212.
[16] Brown M., Harris C.J. Neuro fuzzy adaptive modeling and control, Prentice Hall, 1994.
[17] Bossley K.M. Neurofuzzy Modeling Approaches in System Identification, PhD thesis, University of Southampton, Southampton, UK, 1997.
[18] Azadeh, A, Asadzadeh, SM., Ghanbari, A. An adaptivenetwork-based fuzzy inference system for short-term natural gas demande stimation: Uncertain and complex environments, Energy Policy. 38, 1529–1536, 2010.
[19] C. H. Wu,, D. C Su, J. Chang, C. C. Wei, J. M. Ho, K. J. Lin, and D.T. Lee , "An Advanced Traveler Information System with Emerging Network Technologies", to appear in the 6th Asia-Pacific Conference on Intelligent Transportation Systems, 2003
[20] C. H. Wu,, D. C Su, J. Chang, C. C. Wei, K. J. Lin, and J. M. Ho, "The Design and Implementation of Intelligent Transportation Services", to appear in IEEE Conference on E-commerce, 2003
[21] K.R. Muller, A. Smola, G. Ratch, B. Scholkopf, J. Kohlmorgen,and V Vapnik, "Using Support Vector Support Machines for Time Series Prediction", Image Processing Services Research Lab, AT&T Labs
[22] H. Berenji, Y. Wang, D. Vengerov, R. Langari and M. Jamshidi, "Using Gated Experts in Fault Diagnosis and Prognosis", FUZZ-IEEE, Budapest, Hungary, July 2004.
[23] Berenji, H.R., An Architecture for Designing Fuzzy Controllers using Neural Networks, Readings in Fuzzy Sets for Intelligent Systems, Edited by: D. Dubois, Henri Prade, and Ronald Yager, pages: 368-380, Morgan Kaufmann, 1993.
[24] Berenji, H.R., Neural Networks for Fuzzy Logic Inference, Second IEEE International conference on Fuzzy Systems, San Francisco, CA, March 1993.