Research Journal of Applied Sciences, Engineering and Technology 6(16): 2972-2975, 2013
DOI:10.19026/rjaset.6.3680
ISSN: 2040-7459; e-ISSN: 2040-7467
© 2013 Maxwell Scientific Publication Corp.
Submitted: January 08, 2013 Accepted: February 18, 2013 Published: September 10, 2013

Research Article

Power System Short-Term Load Forecasting Based on Fuzzy Neural Network

[1]Chao Ge, [2]Lei Wang and [3]Hong Wang
[1]College of Information Engineering, Hebei United University,
[2]Department of Information Engineering, Tangshan College,
[3]College of Qinggong, Hebei United University, Tangshan, China 063009

Abstract: Short-Term Load Forecasting (STLF) is an important operational function in both regulated power systems. This study is concerned with the problem of STLF. Considering the factors such as temperature, date type, weather status, etc, which influence the STLF, a model is set up by dynamic recurrent fuzzy neural network. The fuzzy inference function is realized easily by using a product operation in the network. Introducing local recurrent units to hidden layer, the proposed method can overcome the limit of the traditional BP algorithm. The actual simulation is given to demonstrate the effectiveness of the proposed methods.

Keywords: Fuzzy neural network, dynamic recurrent, load forecasting

INTRODUCTION

Short-Term Load Forecasting (STLF) is an essential part of an electrical power system's operation and has been used from many years ago due to its importance. It provides input data for many operational functions of power systems such as unit commitment, economic dispatch, optimal power flow and security assessment. A more accurate STLF can lead to more economic operating decisions and enhance the security level of the power system.

During the last decade, the STLF of power systems have been received a great deal of efforts by many researchers. For instance, time series models for STLF such as ARMA (Auto-Regressive Moving Average) (Huang and Shih, 2003) and modified ARMA (Amjady, 2001), nonparametric regression (Charytoniuk and Chen, 1998.), Kalman filter (Al-Hamadi and Soliman, 2004) and Neural Network (NN) (McMenamin and Monforte, 1998) have been presented in the literature. Recently, some research works have combined different forecast techniques and proposed more efficient hybrid STLF methods. For instance, a combination of fuzzy linear regression and general exponential smoothing (Song *et al.*, 2006), a two-stage hybrid network composed of Self-Organized Map (SOM) and support vector machine (SVM) (Fan and Chen, 2006), a combination of similar day and NN techniques (Senjyu *et al.*, 2005) and hybridization of forecast aided state estimator with NN (Amjady, 2007) have all been presented for STLF.

Despite the research work performed in the area, more accurate and robust STLF methods, that can be easily adapted to different sites, are still in demand. In this study, a novel Dynamic Recurrent Fuzzy Neural Network (DRFNN) is proposed based on the characteristics of short term load forecast. Multiplication Operation is used in rule layer to ensure the activation degree $\omega_j(k) \leq 1$ for each rule thus easier to realize the fuzzy inference function. Through introducing delay unit into rule layer, static networks have dynamic features. Network activation degree at K times for each rule not only refers to the activation value after computing, but also includes all the activation value contribution at previous times. With the aid of this algorithm, the DRFNN based forecast engine can effectively learn the input/output mapping function of a load time series presenting high STLF accuracy for any power system.

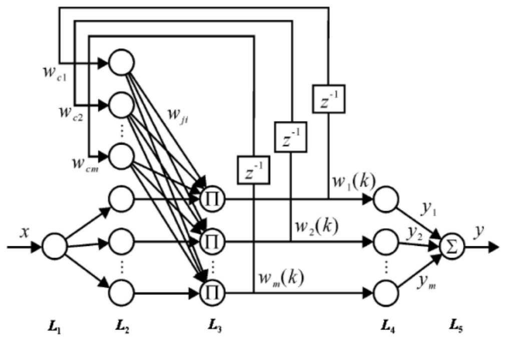

Fig. 1: Frame of dynamic recurrent fuzzy neural network

Dynamic Recurrent Fuzzy Neural Networks (DRFNN): As Fig. 1 show, network topology structure has 5 layers, namely input layer, fuzzy layer, rule layer and output layer. There is recursive layer neurons in rule layer and the neurons are recursive as a result of internal feedback connections. So dynamic response is acquired that can simplify the network model.

L1 is input layer, directly connected to input vector, transmits the input value to next layer.

L2 is fuzzy layer, each node represents a language variable value, such as S_2, S_1, C_E, B_1, B_2 and so on. It is used to compute membership function (The input value belongs to fuzzy set of various language variables). According to the real situation, Gaussian function is chosen as the input membership function:

$$\mu_j = \exp\left[-\left(\frac{x(k)-x_j}{\delta_j}\right)^2\right] \tag{1}$$

where,

x_j	= The center of the j subjection function
δ_j	= The width of the j subjection function
$x(k)$	= The input of the time of k
j	= 1, 2, … , m
m	= The fuzzy divisions

L3 is rule layer, each node, which represents a fuzzy rule, has a function of matching first component of fuzzy rules. and then figure out activation degree $\omega_j(k)$ of each rule. When the recursive layer is introduced, according to the network shown in Fig. 1:

$$\omega_j(k) = \left[\prod_{i=1}^{m} \omega_{ci}(k) W_{ji}\right] \exp\left[-\left(\frac{x(k)-x_j}{\delta_j}\right)^2\right] \tag{2}$$

where, W_{ji} is the connection weight from the j node of recursion layer to the i node of the rule layer, $\omega_{ci}(k)$ is the activity of the i node of the rule layer:

$$\omega_{ci}(k) = \omega_i(k-1), i = 1, 2, \cdots, m \tag{3}$$

The formula (2) and (3) show that the veracity of the network identification is increased and the static network has the dynamic characteristic. Because the recursion layer is introduced in the network structure, the activity $\omega_j(k)$ of each rule in the network at the time of k not only includes the activity value μ_j which is obtained from the current input, but also includes the contribution of the former activity value $\prod_{i=1}^{m} \omega_i\,(k-1)$.

Different from other dynamic recursive networks, this study puts forward multiplication operation in rule layer. In this way, not only is the activation degree for each rule ensured ($0 < \omega_j\,(k) \leq 1$), but also the fuzzy reasoning function is easier to achieve, because the fuzzy rule mode is easy for operators to understand.

L4 is the defuzzification layer, which realized the defuzzification operation. The definition of each node activity in this layer is:

$$\phi = 1 / \sum_{i=1}^{m} \omega_i(k) \tag{4}$$

The output of the i node in this layer is:

$$\overline{\omega_i}(k) = \phi\omega_i(k)\sqrt{b^2 - 4ac} \tag{5}$$

The fifth layer is the output layer. The output of the network at the moment k is:

$$y(k) = \sum_{i=1}^{m} \tilde{y}_i(k)\overline{\omega_i}(k) = \sum_{i=1}^{m} \tilde{y}_i(k)\omega_i(k) / \sum_{i=1}^{m} \omega_i(k) \tag{6}$$

where, $\tilde{y}_i(k)$ is the conclusion of the i fuzzy rule at the moment k

DRFNN learning algorithms: When the fuzzy division is confirmed, the network parameters need to be trained. They are the connection weight W_{ji} from recursion layer to rule layer, the center x_j and width δ_j of the input subjection function, the connection weight $\tilde{y}i$ from hidden layer to output layer.

The objective function can be defined by Mean Square Error (MSE):

$$MSE = \frac{1}{N}\sum_{k=1}^{N} E_p(k) \tag{7}$$

where, N is the sample number, $E_p(k)$ is the instantaneous square error:

$$E_p(k) = \left\| y_d(k) - y(k) \right\|^2 \tag{8}$$

where, $y_d(k)$ is the true sample output at the moment t, y(k) is the output of DRFNN at the moment t.
From formula (2) and (3), we can find:

$$\omega_{cj}(k) = \omega_j(k-1) = \left[\prod_{i=1}^{m} \omega_{ci}(k-1) W_{ji}\right] \times \exp\left[-\left(\frac{x(k-1)-x_j}{\delta_j}\right)^2\right] \tag{9}$$

The formula (9) implies that $\omega_{cj}(k)$ is the dynamic recurrence course which is depended on the foretime connection weight. So the relevant BP algorithm is the dynamic BP algorithm.

The parameters are adjusted by the gradient descent method, which can find the minimum parameter for the object function E_p. Learning algorithm for adjusting specific parameters is as follows:

Connect Weight (W_{ji}) Adjustment from Recursive Layer to Rule Layer:

$$W_{ji}(k+1) = W_{ji}(k) - \eta_\omega \partial E_p / \partial W_{ji}\big|_k \tag{10}$$

where, η_ω is the learning rate coefficient. The big learning rate coefficient can make the algorithm converge fast, but prone to oscillation. The small learning rate coefficient can slow oscillation, but the convergence gets slower and then the principle of their choice must accord to the actual identification. We can get as follow by using chain rule:

$$\frac{\partial E_p}{\partial W_{ji}} = \frac{\partial E_p}{\partial y(k)} \frac{\partial y(k)}{\partial \omega_j(k)} \frac{\partial \omega_j(k)}{\partial W_{ji}} \tag{11}$$

We can get the derivative by formula (8):

$$\partial E_p / \partial y(k) = -(y_d(k) - y(k)) \tag{12}$$

by formula (6):

$$\frac{\partial y(k)}{\partial \omega_j(k)} = \frac{y_j(k) - y(k)}{\sum_{i=1}^{m} \omega_i(k)} \tag{13}$$

by formula (2) and (9):

$$\frac{\partial \omega_j(k)}{\partial W_{ji}} = \left[\prod_{n=1}^{m} \omega_n(k-1) W_{jn}\right] \exp\left[-\left(\frac{x(k)-x_j}{\delta_j}\right)^2\right] \times \left(\frac{1}{W_{ji}} + \frac{1}{\omega_j(k-1)} \frac{\partial \omega_j(k-1)}{\partial W_{ji}}\right) \tag{14}$$

This will constitute the gradient $\partial\omega_j(k)/\partial W_{ji}$ dynamic recurrence relation and this is similar with the time back-propagation learning algorithm.

Input Value Membership Function Centre (x_j) Adjustment:

$$x_j(k+1) = x_j(k) - \eta_x \partial E_p / \partial x_j |_k \tag{15}$$

where, η_x is the learning rate coefficient, the principle of selection is introduced as before. According to chain rule we can get the derived function as follow:

$$\frac{\partial E_p}{\partial x_j} = \frac{\partial E_p}{\partial y(k)} \frac{\partial y(k)}{\partial \omega_j(k)} \frac{\partial \omega_j(k)}{\partial x_j} \tag{16}$$

By Eq. (2) and (9) we can get as follow:

$$\frac{\partial \omega_j(k)}{\partial x_j} = \left[\prod_{i=1}^{m} \omega_i(k-1) W_{ji}\right] \exp\left[-\left(\frac{x(k)-x_j}{\delta_j}\right)^2\right] \times \left[2\left(\frac{x(k)-x_j}{\delta_j}\right) + \frac{1}{\omega_j(k-1)} \frac{\partial \omega_j(k-1)}{\partial x_j}\right] \tag{17}$$

The Eq. (17) will constitute the gradient $\partial\omega_j(k)/\partial x_j$ dynamic recurrence relation. The Eq. (16) can be get by the Eq. (12), (13) and (17).

Input Value Membership Function Width (δj) Adjustment:

$$\delta_j(k+1) = \delta_j(k) - \eta_\delta \partial E_p / \partial \delta_j |_k \tag{18}$$

where, η_δ is the learning rate coefficient. According to chain rule we can get the derived function as follow:

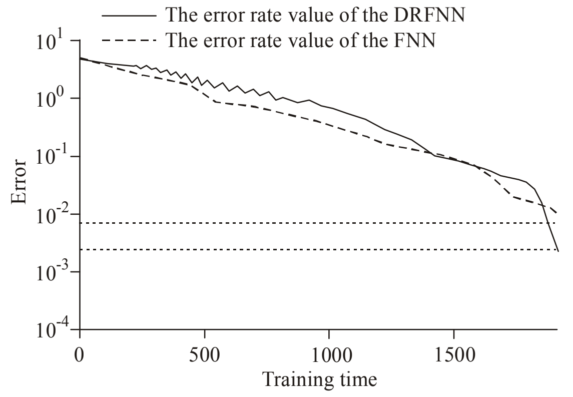

Fig. 2: Training error comparison of DRFNN and FNN

$$\frac{\partial E_p}{\partial \delta_j} = \frac{\partial E_p}{\partial y(k)} \frac{\partial y(k)}{\partial \omega_j(k)} \frac{\partial \omega_j(k)}{\partial \delta_j} \tag{19}$$

$$\frac{\partial \omega_j(k)}{\partial \delta_j} = \left[\prod_{i=1}^{m} \omega_i(k-1) W_{ji}\right] \exp\left[-\left(\frac{x(k)-x_j}{\delta_j}\right)^2\right] \times \left[2\frac{(x(k)-x_j)^2}{\delta_j^3} + \frac{1}{\omega_j(k-1)} \frac{\partial \omega_j(k-1)}{\partial \delta_j}\right] \tag{20}$$

The Eq. (20) will constitute the gradient $\partial\omega_j(k)/\partial \delta_j$ dynamic recurrence relation. The Eq. (19) can be get by the Eq. (12), (13) and (20).

Connect Weight $\tilde{y}_i$ Adjustment from Hide Layer to Output Layer:

$$\tilde{y}_i(k+1) = \tilde{y}_i(k) - \eta_{\tilde{y}} \partial E_p / \partial \tilde{y}_i |_k \tag{21}$$

where, η_y is the learning rate coefficient. According to chain rule we can get the derived function as follow:

$$\frac{\partial E_p}{\partial \tilde{y}_i} = \frac{\partial E_p}{\partial y(k)} \frac{\partial y(k)}{\partial \tilde{y}_i(k)} \tag{22}$$

By Eq. (6) we can get as follow:

$$\frac{\partial y(k)}{\partial \tilde{y}_i(k)} = \omega_i(k) / \sum_{i=1}^{m} \omega_i(k) \tag{23}$$

By Eq. (12) and (23) we can get Eq. (22).

Dynamic recurrent fuzzy neural network of short term load forecast: In order to verify the validity of DRFNN, a computer simulation about DRFNN and traditional FNN is conducted, taking the electric power system data from 1st May to 15th June in 2008 of a North China city as samples. The random number (W_{ji} and $\tilde{y}_i$ is [0, 1] respectively) is chosen. The initial input membership function is Gaussian function. The end condition is as follows: 2000 time's circulation or the error reaches 0.001.

As Fig. 2 shows, according the comparison between DRFNN and FNN training error curve chart, the

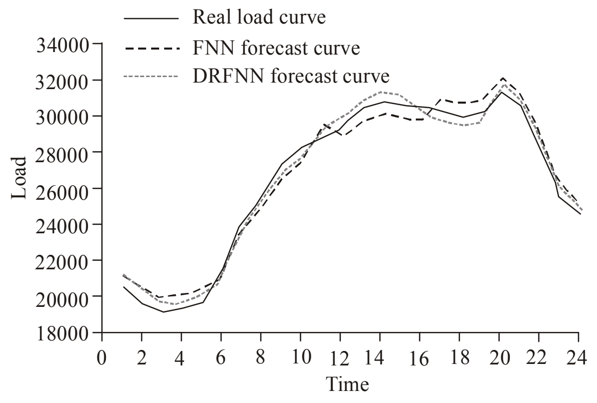

Fig. 3: Prediction comparison of DRFNN and FNN

Table 1: Prediction comparison of DRFNN and FNN

Forecast model	Forecast accuracy/%	Time/S
DRFNN	98.5	74.6
FNN	92.8	72.5

network gains dynamic feature after introducing recursive delay unit. Therefore, the learning ability of neuron network model is improved and the lower training error is reached.

As Fig. 3 shows, DRFNN forecast curve is more similar to the real load curve than FNN. It is proved that FNN forecast is improved by multiplication operation introduced into recursive layer to some extent.

Table 1 shows that the forecast time of the two models are approximately the same. It is proved that DRFNN forecast model ensures forecast efficiency while increasing the forecast accuracy.

CONCLUSION

The short term load forecasting model based on DRFNN in this study can improve the learning ability about the fuzzy neural network effectively. Through Simulation Verification, it is proved that both the convergence rate and forecasting accuracy of DRFNN is increased to some extent compared to traditional FNN, avoiding the problem of partial optimum. This network, which has been proved effective through illustrative examples, can be applied to long term forecast in electric power system after the research methodologies are improved and perfected.

REFERENCES

Al-Hamadi, H.M. and S.A. Soliman, 2004. Short-term electric load forecasting based on Kalman filtering algorithm with moving window weather and load model. Electr. Power Syst. Res., 68: 47-59.

Amjady, N., 2001. Short-term hourly load forecasting using time series modeling with peak load estimation capability. IEEE T. Power Syst., 16: 798-805.

Amjady, N., 2007. Short-term bus load forecasting of power systems by a new hybrid method. IEEE T. Power Syst., 22: 333-341.

Charytoniuk, W. and M.S. Chen, 1998. Nonparametric regression based short-term load forecasting. IEEE T. Power Syst., 13: 725-730.

Fan, S. and L. Chen, 2006. Short-term load forecasting based on an adaptive hybrid method. IEEE T. Power Syst., 21: 392-401.

Huang, S.J. and K.R. Shih, 2003. Short-term load forecasting via ARMA model identification including non-Gaussian process considerations. IEEE T. Power Syst., 18: 673-679.

McMenamin, J.S. and F.A. Monforte, 1998. Short-term energy forecasting with neural networks. Energy J., 19: 43-61.

Senjyu, T., P. Mandal, K. Uezato and T. Funabashi, 2005. Next day load curve forecasting using hybrid correction method. IEEE T. Power Syst., 20: 102-109.

Song, K.B., S.K. Ha, J.W. Park, D.J. Kweon and K.H. Kim, 2006. Hybrid load forecasting method with analysis of temperature sensitivities. IEEE T. Power Syst., 21: 869-876.