International Journal of Electrical, Electronic and Communication Sciences
ISSN: 2517-9438
Vol:8, No:9, 2014

In Search of a Suitable Neural Network Capable of Fast Monitoring of Congestion Level in Electric Power Systems

Pradyumna Kumar Sahoo, Prasanta Kumar Satpathy

***Abstract*—**This paper aims at finding a suitable neural network for monitoring congestion level in electrical power systems. In this paper, the input data has been framed properly to meet the target objective through supervised learning mechanism by defining normal and abnormal operating conditions for the system under study. The congestion level, expressed as line congestion index (LCI), is evaluated for each operating condition and is presented to the NN along with the bus voltages to represent the input and target data. Once, the training goes successful, the NN learns how to deal with a set of newly presented data through validation and testing mechanism. The crux of the results presented in this paper rests on performance comparison of a multi-layered feed forward neural network with eleven types of back propagation techniques so as to evolve the best training criteria. The proposed methodology has been tested on the standard IEEE-14 bus test system with the support of MATLAB based NN toolbox. The results presented in this paper signify that the Levenberg-Marquardt backpropagation algorithm gives best training performance of all the eleven cases considered in this paper, thus validating the proposed methodology.

***Keywords*—**Line congestion index, critical bus, contingency, neural network.

I. Introduction

THE problems faced by present day electrical power systems are numerous, as a result of which finding a suitable solution for healthy and reliable operation has gradually turned out to remain even more critical than ever. In this paper, the authors have focused more on one such critical issue, i.e. congestion of transmission corridors, which could prove to be fatal for the utility and the consumers of electricity, if ignored beyond a certain margin. Congestion level in transmission lines is primarily governed by several factors such as physical structure of the network, available generation/transmission capacities for support of active/reactive power, unprecedented abnormal hike in the demand for electricity, and last but not the least the unforeseen contingency conditions that might occur anytime, anywhere in the system beyond the knowledge and schedule of the operators on job. The issues related to transmission congestion of electrical power systems and scopes for the relief have been well addressed in the literature. Daniel et al. [1] tried in their research to find the contribution of individual generators in a system while calculating the system performance in order of the loads and hence to determine the line flows. Bansilal et al. [2] proposed an expert system mechanism for alleviating network overloads with a view to protect the system from hitting the congestion limits. Several researchers [3], [4] in their proposed work have raised various key issues on the background of transmission congestion concerning to smooth transmission dispatch in emerging and competitive energy markets aided by unbundled operation, service identification and minimum number of adjustments in preferred schedules in the face of constraints imposed by contingency conditions. In a way to circumvent the impacts of transmission congestion on the power utilities, several studies have been performed such as performance evaluation [5], coordination in the deregulated power market [6], and preventive analysis for relieving overloading of the system [7]. A. Kumar et al. in their research paper [8] presented a thorough survey on most of the important works carried out in the area of transmission congestion almost over a period of two decades.

Prasanta Satpathy is with the Department of Electrical Engineering, College of Engineering and Technology, Bhubaneswar, Odisha 751003, India (phone: +91-674-238-6075; fax: +91-674-238-6182; e-mail: pksatpathy_ee@cet.edu.in).
Pradyumna Sahoo is with the Department of Electrical Engineering, ITER, S'O'A University, Bhubaneswar, Odisha 751003, India (e-mail: pradyumnakumar_sahoo@yahoo.co.in).

Yet, the problem remained a matter of significant worry and deep concern for every active researcher in this area. The major trouble that surfaced in the process of evaluation of the transmission line congestion level is that it requires execution of a systematic load flow program for every time frame in the dynamic scenario of events and schedules. Such a stringent requirement manifests further inhibitions in a realistic way. The authors of this paper feel that application of NN to power systems could possibly bring some improvements in reducing time delay that occurred in the conventional approach. Thus, preventive and precautionary measures would be more functional and effective if the repeat execution of the load flow program could be possibly bypassed with implementation of the proposed NN approach.

In view of this, the authors have tried a more fundamental way of approach of deriving the line congestion index (*LCI*) once for all in an exhaustive way, covering most of the possible events and situations that would possibly come the way of power system operation. The idea behind this collection of information is to prepare an exhaustive set of input/target data to be presented before the NN for imparting a thorough training through successive supervisory learning mechanism, as it is believed that a thoroughly trained NN becomes capable of learning from these events so that the trained NN could be exposed to any set of input data in a future time for necessary validation and testing purpose. Thus, the operator's burden of repeat execution of the load flow program for monitoring the critical conditions in a dynamic time frame could be relieved in a significant way. Secondly, in order to improvise the performance of the proposed feed forward neural network, the authors have tried a comparative analysis for eleven types of back propagation learning rules (i.e. gradient descent backpropagation, gradient descent backpropagation with momentum, variable learning rate backpropagation, resilient backpropagation, conjugate gradient backpropagation with Fletcher-Reeves update, conjugate gradient backpropagation with Polak-Ribiere update, conjugate gradient backpropagation with Powell-Beale restarts, scaled conjugate gradient backpropagation, quasi-Newton BFGS backpropagation, quasi-Newton one step secant backpropagation, and Levenberg-Marquardt backpropagation) in search of deriving the best NN structure for this purpose. In consideration of these facts, the paper is organized as follows.

Section II highlights the methodology behind formation of the line congestion index (*LCI*) as a function of real power handled by each line in the network. Section III presents the background of NN structure and its applicability to this problem. In this section, the authors have considered multiple layered feed forward NN supplemented with eleven types of backpropagation algorithms while considering various combinations of hidden layers and number of neurons in those layers. In Section IV, a detailed case study is presented through implementation of the proposed methodology in the standard IEEE 14-bus test system and it is observed that the algorithm works well, as evident from the convergence results during training offering minimal error goal to reach the specific target. From the comparative analysis of Section IV, it is inferred that the proposed feed forward neural network along with the Levenberg-Marquardt backpropagation algorithm would perform the best by scaling out the rest of the backpropagation schemes.

II. Calculation of Line Congestion Index

The methodology adopted in this work is based on the most fundamental principles of electrical power system studies, such as, characteristics of transmission lines, their performance and load flow studies. While evaluating the line flows for a particular network configuration and an underlying situation, the focus is primarily made so as to maintain the voltage at the candidate load buses within specified limits as referred by the grid code. In a particular power system case as shown in Fig. 1, the load current is governed mostly by the impedance of the load connected at the receiving end of the line. While, the load impedance matches exactly with the characteristic impedance or surge impedance of the line, the line gets eventually terminated by its own characteristic impedance resulting in an infinite line and the then power supplied to the load through the line is called surge impedance loading (*SIL*) of the line. A comparison between the actual value of real power (P_l) being transmitted in a particular line with its own surge impedance loading (*SIL*) could be established as a measure or an indicator of the congestion level in that particular line.

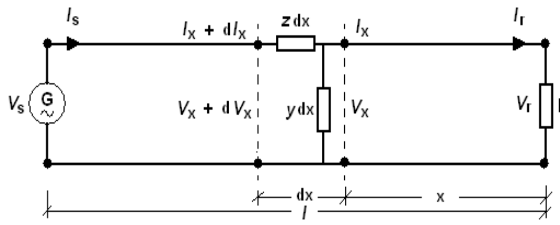


Fig. 1 Distributed representation of a long transmission line

Thus, the line congestion index (*LCI*) is treated as the ratio of the two powers as given in (2).

$$SIL = \frac{|V_l|^2}{Z_c} \tag{1}$$

$$LCI = \frac{P_l}{\left(\frac{|V_l|^2}{Z_c}\right)} \tag{2}$$

The index offers an insight on many aspects such as loading level of lines as a function of limiting value of real power transaction possible, level of transmission loss taking place in the lines and degree of temperature rise associated thereof. In order to accomplish the objective of the proposed scheme, the authors have implemented the same in the Newton-Raphson based load flow algorithm supported by the Matpower computing platform for computation of *LCI*.

In general, the performance of electrical power utilities remains almost stable during base case loading scenario. However, in a complex dynamic system like this, the operating conditions remain hardly static. Therefore, it becomes mandatory to monitor the system's performance with contingent conditions so as to assess critical issues like line congestion level and margin to voltage instability during such worse situations. In this paper, the load demand at the load buses are increased simultaneously above the base loading with additional step loading. This is accomplished by use of a multiplying factor (MF=λ). Assuming that the complex load demand at bus-i during base case is denoted by '$S_{i,base}$', the load increasing scenario for any future time 'S_i' is expressed as a function of '$S_{i,base}$' as indicated in (3). It may be noted here that a zero value for the load multiplying factor refers to base case loading condition and non-zero positive values greater than zero refer to higher loading beyond the base case.

$$S_i = (1 + \lambda)S_{i,base} \tag{3}.$$

III. Problem Formulation with Neural Networks

Biological neural networks in general refer to a portion of the biological structure of the nervous system in living beings, which perform the wonderful job of mapping input signals with a set of reference signals in order to produce an output signal that activates the stimuli accordingly and generates control signals for various activities that take place in the living organism. This concept when applied to non-living entities (dynamic systems consisting of processing plants) for decision making, it is termed as Artificial Neural Networks (ANN). The literature indicates successful application of ANN in solving complex real world problems with ease and it has also been widely accepted by the researchers in the area of electrical power systems [9]-[14]. Being motivated by this fact, the authors of this paper have tried to implement the NN principles in this present work for obtaining approximate and faster results in computing the congestion level of transmission lines without compromising much on the accuracy and at the same time developing a comparison on the performance of various NN structures.

Neural networks are capable of analyzing complex mappings accurately and rapidly, without any particular involvement of functional relationships existing between the independent and dependent variables of the system or the process. A generalized structure of the NN is presented in Fig. 2, which comprises of an input stage, another output stage and few hidden layers. The hidden layers contain neuron like elements having interconnectivity which by and large determine the network function. Each connection is associated with an index called weight parameter that modulates/transforms the input in accordance with the weighting index in order to present a specific output. Such networks also have the ability to synthesize the internal structure of the neurons through assignment and adjustment of weights, based on the exposure, experience and learning skill they acquire through training.

In this paper the authors have used '*newff*' algorithm of MATLAB NN-toolbox for the feed forward network, which considers transfer functions 'tan-sigmoid (*tansig*)' and 'linear (*purelin*)' respectively for the hidden layers and the output layer of the NN, as illustrated in Fig. 3. A single line diagram of a simple neuron model is shown in Fig. 4, so as to illustrate the functional mechanism of adjustments in the weighted connections (w) while transforming a given input (p) into a corresponding output (a) with or without biasing (b). The bias is much like the weight except the fact that bias is always assigned a constant unit value (b=1). Fig. 4 (a) illustrates the mechanism considering weights only and Fig. 4 (b) shows the same, in presence of both weights and biases. Equations (4) and (5) represent the mathematical form of the output for the network architectures shown in Figs. 4 (a) and (b) respectively. Though, there is absolutely no restriction in making a suitable selection of the transfer function (f), yet it could be preferably any of the mathematical functions such as hard-limiting, linear, logarithmic, sigmoid, or bell functions.

$$a = f(wp) \tag{4}$$
$$a = f(wp + b) \tag{5}$$

The process by which the NN learns is called training process. The training objective may be set to obtain goals like approximation of functions and pattern association or pattern classification of data sets. In order to start with the training, it is desirable that the weights and bias associated with the neurons be properly initialized. The learning rules for the training are of two types such as supervised learning and unsupervised learning. In case of supervised learning, the NN is trained with the help of a training set comprising of actual inputs and their corresponding correct outputs (treated as the target). However, in unsupervised learning, the NN is trained in response to network inputs only as the target outputs are not available.

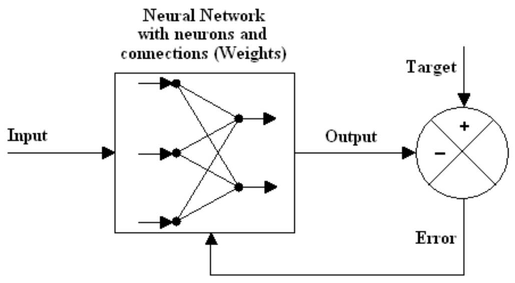


Fig. 2 Generalized representation of a Neural Network

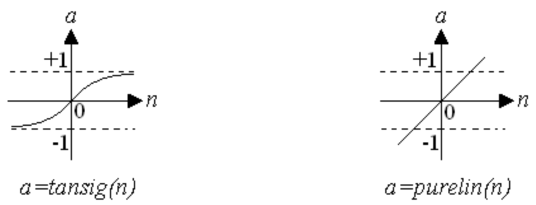
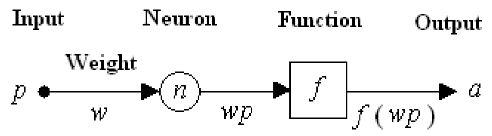
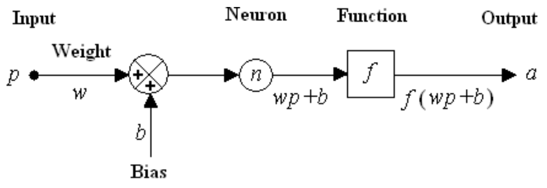


Fig. 3 Transfer functions for feed forward Neural Network



(a) NN with weights only



(b) NN with weights and bias

Fig. 4 (a) NN with weights only, and (b) NN with weights and bias

In this paper, the former training algorithm (i.e. supervised learning rule) has been followed by considering multiple layered feed forward NN architecture supported with the standard gradient descent backpropagation algorithm [15] and Widrow-Hoff learning rule [16]. A brief discussion on the eleven types of backpropagation schemes considered in this paper is presented next.

A. Gradient Descent Backpropagation (GDB)

The gradient descent training function drags the network weights along the negative/reverse direction of the gradient corresponding to the performance function by calculating the derivative of the transfer functions assigned to the neurons in the network's layers. As the number of layers increase in a network, learning of complex relationships between inputs and output becomes faster. A successfully trained NN becomes capable of producing reasonable outputs in response to newer inputs. In mathematical form the logic of backpropagation technique may be expressed as shown in (6):

$$Z_{k+1} = Z_k - \alpha_k g_k \tag{6}$$

where, Z_{k+1} is the vector of weights and bias values to be calculated with respect to current vector Z_k by utilizing current gradient g_k and learning rate α_k.

B. Gradient Descent Backpropagation with Momentum (GDBM)

The gradient descent training function with momentum helps a neural network to learn in response to recent trends over the error surface in addition to the local gradient. This could be achieved through selection of a momentum parameter in the range of zero and unity. A zero assignment corresponds to no momentum, while a unity value attributes for highest possible momentum. The advantage of assigning a momentum to the gradient enhances the ability of the network to ignore smaller features over the error surface, thereby enabling the learning process to come over the points of local minimum without getting stuck in the shallow there.

C. Variable Learning Rate Backpropagation (VLRB)

Unlike the gradient descent algorithm (where the learning rate is maintained constant throughout the training), this algorithm is supplemented with the provisions of acquiring an adaptive learning rate that could change in accordance to changes occurring over the error surface during the training process, while keeping the learning stable throughout. A simple logic specified by the algorithm monitors the errors calculated during each epoch. If it is observed that the current error has exceeded the previous one by a specific pre-defined margin, the algorithm thereby prompts for repeat calculation of weights and bias with a lower learning rate than the current one despite continuing with the currently calculated weights and bias values instead.

D. Resilient Backpropagation (RB)

In case of muli-layered neural networks with sigmoid function in the hidden layers, the gradient descent backpropagation often suffers from slower convergence due to smaller changes observed in weights and bias parameters during the epochs. This problem could be accelerated by propelling the learning with a resilient function that monitors the sign of the partial derivative of the performance function with respect to that of the weights over two consecutive iterations during the epochs of the training process. If, it is observed that same sign is maintained over any two consecutive iterations, the updated weights and bias values are lowered by a specific factor. However, if the derivative results in zero, the update is maintained as such.

E. Conjugate Gradient Backpropagation with Fletcher-Reeves Update (CGBFRU)

Yet, in search of finding a faster convergence as compared to all the backpropagation schemes discussed earlier, the conjugate gradient scheme utilizes the mechanism of adjusting the step size of the updates during iterations by making a search along the conjugate gradient direction that would minimize the performance function along that direction. In this Fletcher-Reeves method, the updates are calculated as per (7) through (9):

$$Z_{k+1} = Z_k + \alpha_k p_k \tag{7}$$

where

$$p_k = -g_k + \beta_k p_{k-1} \tag{8}$$

and

$$\beta_k = \frac{g_k^T g_k}{g_{k-1}^T g_{k-1}} \tag{9}$$

F. Conjugate Gradient Backpropagation with Polak-Ribiere Update (CGBPRU)

As a subset of conjugate gradient backpropagation scheme, (7) and (8) remain valid for this Polak-Ribiere method. However, corresponding updates are calculated as per the expression given in (10):

$$\beta_k = \frac{\Delta g_{k-1}^T g_k}{g_{k-1}^T g_{k-1}} \tag{10}$$

G. Conjugate Gradient Backpropagation with Powell-Beale Restarts (CGBPBR)

As a subset of conjugate gradient backpropagation scheme, (7), (8) and (10) also remain valid for this Powell-Beale method. However, corresponding updates are calculated as per the condition of (11):

$$|g_{k-1}^T g_k| \geq 0.2\|g_k\|^2 \tag{11}$$

H. Scaled Conjugate Gradient Backpropagation (SCGB)

Each component of conjugate gradient backpropagation scheme considers a line search trend during the iterations, which could be computationally expensive. Therefore, this method uses a scaling mechanism by combining the model-trust region approach [17].

I. Quasi-Newton BFGS Backpropagation (QNBFGSB)

The Newton's BFGS method of backpropagation is an improved version of conjugate gradient method. It makes use of a Hessian matrix $[A_k]^{-1}$ consisting of the performance index for the current weights and bias parameters as shown in (12):

$$Z_{k+1} = Z_k - A_k^{-1} g_k \tag{12}$$

J. Quasi-Newton One Step Secant Backpropagation (QNOSSB)

One major limitation of Newton's BFGS backpropagation method is that it consumes lot of memory due to storage of the Hessian matrix. This limitation is overcome by adding the secant approximation method, which reduces the need for storing the complete Hessian matrix. Hence, this scheme becomes capable of faster convergence while ensuring limited memory usage.

K. Levenberg-Marquardt Backpropagation (LMB)

In this method, the Hessian matrix is approximated instead being computed in each of the iterations, thus making this scheme most competitive. Equations (13) through (15) indicate the procedure being adopted in this method.

$$Z_{k+1} = Z_k - H^{-1}h \tag{13}$$

where

$$H = J^T J + \mu I \tag{14}$$

and

$$h = J^T e \tag{15}$$

Among the other unknowns, 'J' is the jacobian matrix containing first derivatives of errors with respect to weights and bias values, 'e' is the error of the network, and 'μ' is a scalar in the range of zero and higher.

L. Application of Feed Forward Backpropagation for Finding Training Performance

It may be noted here that the weights and bias parameters could be adjusted during training process in order to minimize the network performance function until the net output falls within close tolerable/acceptable proximity to the target set by the operator. It could take few iterations/epochs during each cycle of training before the NN gets properly trained so as to rationalize newer inputs through mapping by utilizing the experience acquired during earlier training. The process of exposing the trained NN to a predefined input dataset is termed as validation and testing, which is essential for validating the correctness of the trained NN. The performance of NN to adapt to particular or random inputs could be very well assessed from graphical plots such as error surface (ES) plots and performance plots.

Error surface indicates the error associated with the neurons over a range of weight and bias values. The shallowest point of the error surface corresponds to best values of weights and bias parameters as the calculated error remains the least at that point. Performance plots highlight the paths of training, validation and testing schemes during the process of convergence between the NN outputs and the assigned targets subject to the constraint imposed in the form of acceptable tolerance limit for the error, hence showing best validation during the iterations/epochs of the learning process.

IV. Implementation and Result Analysis

The proposed methodology has been implemented on standard IEEE-14 bus test system. Fig. 5 indicates a pie chart with percentage division of input data into three components; training data (*TrD*), validation data (*VaD*), and test data (*TeD*). In this paper, percentage ratio of data division is considered as; $TrD : VaD : TeD = 60 : 20 : 20$.

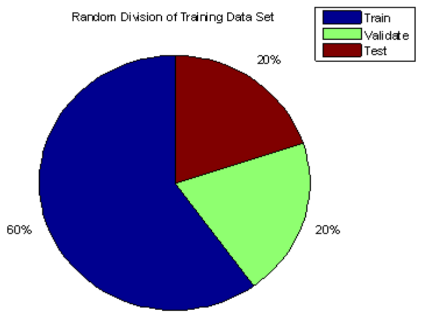


Fig. 5 Percentage division of input data; (*TrD*), (*VaD*), and (*TeD*)

The approach followed in this paper for preparing the input data set for the test system under study is described here. It contains thirty two situations under two distinct categories comprising of sixteen situations each for sixteen different loading scenarios described by load multiplying factor (λ=0 through 1.5 with a step rise of 0.1 per step). The initial sixteen situations for category-I corresponds to the *LCI* results for the twenty lines of the IEEE 14-bus test system as obtained from the Newton-Raphson load flow corresponding to initial voltage setting of flat 1.0 p.u. at all the system buses.

The remaining sixteen situations for category-II corresponds to similar load flow results corresponding to initial bus voltage settings as specified by the load flow inputs. The line congestion index (*LCI*) of all the twenty branches/lines of IEEE-14 bus test system corresponding to the above mentioned thirty two situations have been obtained from successive execution of the Newton-Raphson power flow program and are presented to form the input data for the proposed training of the NN. Thus, the size of the input matrix is observed as (20×32).

While doing so, the voltage states of the buses are continuously monitored to identify if the voltage at any bus had a grid code violation (i.e. beyond ±5% of initial base case voltage). If the number of violations for a particular situation exceeded a predefined limit (in this paper, the limit is being set at 30% of total buses present in the system), the target for that situation is set as '1', else which the target is set as zero. Thus the resulting target matrix had a dimension of (1×32) for this case. Once the structure of the NN, input and target data were finalized, the proposed feed forward NN algorithm with eleven types of backpropagation training rules is performed to obtain the numerical results and those in form of graphs such as error surface (ES) plots, and performance plots. During the trial sessions of training, it has been observed that arbitrary selection of parameters (error tolerance limit, limiting number of epochs assigned during training cycles, number of hidden layers and number of neurons in the hidden layers) bear lots of significance towards successful convergence of training cycles. Some of the parameters and results pertaining to the gradient descent backpropagation scheme are presented in Table I. It is observed from the results of Table I that the NN offered limitations leading to either non-convergence or delayed convergence in some cases with higher assignment of epoch limit and selection of lower error tolerance limit.

However, convincingly good results (faster convergence with lesser number of training cycles) are also observable in some cases, having limiting value of epochs equal to 10, with negligible impact on the speed of convergence subject to variations in selection of error tolerance limit.

In order to stage a comparative analysis for selection of a suitable NN, all the eleven cases of backpropagation have also been dealt in this paper with epoch setting in the range of 10, 100, and 1000, with a fixed error tolerance setting of 0.001. The convergence results for all the eleven cases of backpropagation have been presented in Table II. It is observed from the findings of Table II that for the proposed feed forward neural network, the Levenberg-Marquardt backpropagation algorithm stood significantly up in the list of eleven cases of backpropagation algorithm in order to be treated as the 'most consistent and faster algorithm' for monitoring line congestion levels in the system under study, irrespective of the assignments for the epoch setting. Other findings of case study include error surface plots (Fig. 6), and performance plots (Fig. 7), corresponding to the numerical results under case-11 of Table II with an epoch setting of 10 epochs per training cycle. These plots justify that the criteria of training perfection are very well met by the proposed feed forward neural network and the corresponding Levenberg-Marquardt backpropagation algorithm and learning methodology.

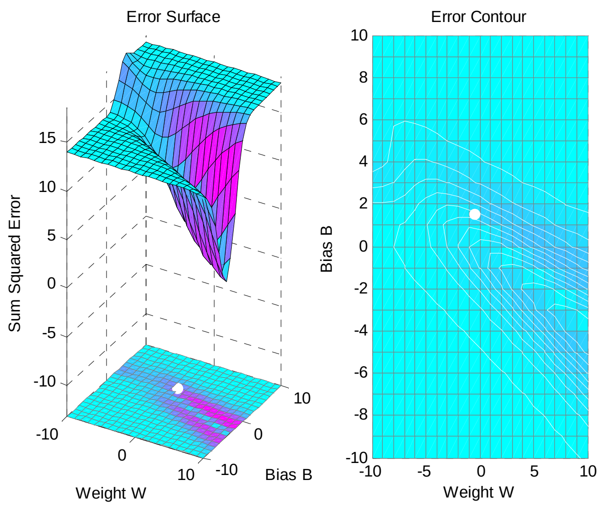


Fig. 6 Error Surface/Error Contour plots showing least error point

TABLE I
TRAINING RESULTS FOR CONVERGENCE

Assigned Parameters (feed forward, gradient descent backpropagation) (a) No. of hidden layers (b) No. of neurons per layer (c) Max. epochs per cycle *(d)* Error tolerance				Training Results at Convergence (e) Trg. Cycles performed (f) Epochs in the last cycle (g) Time taken for Convergence (in seconds)		
(a)	(b)	(c)	(d)	(e)	(f)	(g)
0	0	100	0.001	5889	6	5929
0	0	100	0.01	5889	6	5464
0	0	100	0.1	1261	47	1020
0	0	10	0.001	101	10	40
0	0	10	0.01	81	10	36
0	0	10	0.1	81	10	35
1	5	100	0.001	1829	100	2460
1	5	100	0.01	1363	100	1401
1	5	100	0.1	179	48	156
1	5	10	0.001	4	10	5
1	5	10	0.01	4	10	5
1	5	10	0.1	4	10	5
2	5	100	0.001	265	100	297
2	5	100	0.01	265	100	286
2	5	100	0.1	58	38	55
2	5	10	0.001	14	10	9
2	5	10	0.01	14	10	9
2	5	10	0.1	14	10	9
3	5	100	0.001	>9999	Did not converge	
3	5	100	0.01	>9999	Did not converge	
3	5	100	0.1	140	21	139
3	5	10	0.001	5	10	5
3	5	10	0.01	5	10	5
3	5	10	0.1	5	10	5
4	5	100	0.001	8690	100	11284
4	5	100	0.01	5718	18	6544
4	5	100	0.1	202	39	213
4	5	10	0.001	3	10	4
4	5	10	0.01	3	10	4
4	5	10	0.1	1	10	3
5	5	100	0.001	7917	30	9517
5	5	100	0.01	1820	84	2107
5	5	100	0.1	59	100	70
5	5	10	0.001	14	10	10
5	5	10	0.01	14	10	10
5	5	10	0.1	14	10	10

TABLE II
COMPARATIVE ANALYSIS OF CONVERGENCE RESULTS

Assigned Parameters (feed forward, gradient descent backpropagation) No. of hidden layers=4 No. of neurons per layer=5 Error tolerance for convergence=0.001			Training Results (*Consistently Less cycles used) (#Consistently Faster convergence) ($Best Backpropagation Scheme)		
Sl. No	Type of backprop trg. rule	Epoch limit/per trg.cycle	Training cycles done	Epochs during last cycle	Time for converge nce (s)
		10	3	10	4
1	GDB	100	8690	100	11284
		1000	>9999	Did not converge	
		10	5	10	5
2	GDBM	100	>9999	Did not converge	
		1000	>9999	Did not converge	
		10	5	10	5
3	VLRB	100	>9999	Did not converge	
		1000	>9999	Did not converge	
		10	764	6	354
4	RB	100	764	6	354
		1000	764	6	354
		10	61	6	35
5	CGBFRU	100	61	6	35
		1000	61	6	35
		10	2336	6	1163
6	CGBPRU	100	2336	6	1261
		1000	2336	6	1502
		10	265	6	147
7	CGBPBR	100	265	6	147
		1000	265	6	147
		10	61	6	35
8	SCGB	100	61	6	35
		1000	61	6	35
		10	489	8	351
9	QNBFGSB	100	489	8	351
		1000	489	8	351
		10	265	5	147
10	QNOSSB	100	265	5	147
		1000	265	5	147
		10	55*	**7**	**31#**
11	LMB	100	55*	**7**	**31#**
		1000	55*	**7**	**31#**

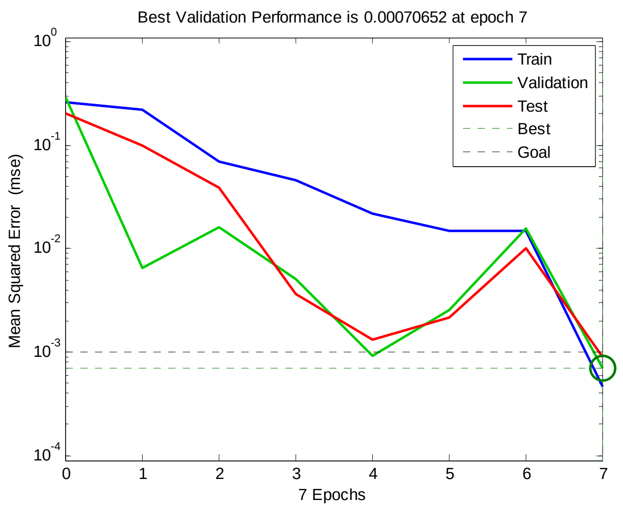


Fig. 7 Performance plots showing Training, Validation, and Testing

V. CONCLUSIONS

The basic objective of finding a successful application of neural networks to power system in order to monitor line congestion level in the transmission lines has been met quite fruitfully in this paper. In the beginning, an exhaustive set of load flow results are obtained for each set of operating condition, which covers base case condition, steady and gradual loading around the network and contingencies as well. In this paper, all these aspects have been considered with proper coordination of issues relating to preparation of input and target data sets, selection of NN structure including number of hidden layers and neurons in the layers, and assignment of weights/bias to the neurons so as to obtain successful training of the feed forward neural network with backpropagation algorithm based on supervised learning. The major advantage of this application would be to provide an alternative (stable and faster) tool for the power system utilities so that frequent use of load flow calculations could be avoided. This objective has been worked out with the proposed methodology in support of MATLAB based Matpower computing platform and the results are validated through the case studies conducted on standard IEEE 14-bus test system. The results of the case study conducted on a standard IEEE-14 bus test system justifies the validity of the findings which also satisfy all the required conditions for a perfectly trained neural network.

REFERENCES

[1] Daniel, K., Allan, R., and Goran, S., "Contribution of Individual generators to loads and flows," IEEE Trans. on Power Systems, Vol. 12, No. 1, 1997.
[2] Bansilal, Thukaram, D., and Parthasarathy, K., "An expert system for alleviation of network overloads," Elect. Power Syst. Research, Vol. 40, pp. 143-153, 1997.
[3] Glavitsch, H., and Alavardo, F., "Management of Multiple Congested Conditions in Unbundled Operation of a Power System," IEEE Trans. on Power Systems, Vol. 13, No. 3, pp. 1013-1019, 1998.
[4] Alomoush, MI., and Shahidehpour, SM., "Contingency-constrained Congestion Management with a Minimum Number of Adjustments in

Preferred Schedules," Elect. Power and Energy Syst., Vol. 22, No. 4, pp. 277-290, 2000.

[5] Goncalves, MJD., and Zita, AV., "Evaluation of Transmission Congestion Impact in Market Power," IEEE Bologna Power Tech. Conference, Vol. 4, pp. 6-11, 2003.

[6] Yamina, HY., and Shahidehpour, SM., "Congestion Management Coordination in the Deregulated Power Market," Elect. Power Syst. Research, Vol. 65, No. 2, pp. 119-127, 2003.

[7] Lobato, L., Rouco, T., Gomez, F., Echaverren, M., Navarrete, MI., Casanova, R., and Lopez, G., "Preventive analysis and solution of overloads in the Spanish electricity market," Elect. Power Syst. Research, Vol. 68, pp. 185-192, 2004.

[8] Kumar, A., Srivastava, SC., and Singh, SN., "Congestion Management in Competitive Power Market: A Bibliographical Survey," Elect. Power Syst. Research, Vol. 76, pp. 153-164, 2005.

[9] Kalra, PK., Srivastava, A., and Chaturvedi, DK., "Artificial neural nets applications to power systems operation and control," Elect. Power Syst. Research, Vol. 25, pp. 83-90, 1992.

[10] Dillon, TS., "Artificial neural nets applications to power systems and their relationships to symbolic methods," Elect. Power and Energy Syst., Vol. 13, pp. 66-72, 1991.

[11] Sharkawi, El., Marks, MA., and Weerasoritya, RJ., "Neural Networks and their Application to Power Engineering," Control Dynamic Syst., Advances in theory and Appln., Vol. 41, 1991.

[12] Ranaweera, DK., "Comparison of neural network models for fault diagnosis of power systems," Elect. Power Syst. Research, Vol. 29, No. 2, pp. 99-104, 1994.

[13] Hagan, MT., and Menhaj, M., "Training feed-forward networks with the Marquardt algorithm," IEEE Trans.on Neural Networks, Vol. 5, No. 6, pp. 989-993, 1994.

[14] Haykin, SS., "Neural Networks: a comprehensive foundation," Prentice Hall, India, 1999.

[15] Hagan, M., Demuth, H., and Beale, M., "Neural Network Design," PWS Publ., Boston, 1996.

[16] Widrow, B., and Sterns, SD.. "Adaptive Signal Processing," Prentice Hall, New York, 1985.

[17] Moller, MF., "A scaled conjugate gradient algorithm for fast supervised learning," IEEE Trans. on Neural Networks, Vol. 6, pp. 525-533, 1993.