Assessment and Comparative Study of Different Enhanced Artificial Neural Networks Based Power Flow Solutions

Hassan Kubba
Dept. of Electrical Engineering, College of Engineering, Baghdad University, Aljadriyia, Baghdad, Iraq

Abstract **- This paper presents the development of Neural Networks based fast load flow solution method, which can be used for such real time applications. A feed-forward model of the neural network based on back propagation algorithm (BP) and a radial basis function neural network (RBFNN) are proposed to solve the load flow problem under different loading/contingency condition for computing voltage magnitudes and angles of a power system. A comparative study is presented to assess the performance of different models of ANNs. The RBFNN has many advantageous features such as optimized system complexity, minimized learning, less computation time for training and simulation and recall times as compared to single layer and multi-layer perceptron models. The effectiveness of the proposed ANNs models for on-line application is demonstrated by computation of buses voltage magnitudes and voltages angles for different loading/contingency conditions in three typical test systems also, the Iraqi National Grid load flow problem is solved by two efficient ANN models.**
The proposed models (RBFNN) have been found to provide sufficiently accurate results and a robustness fast load flow solution which can be efficiently applied to on-line (real-time) implementation.

Keywords: Load Flow Analysis; Contingency Conditions; Newton-Raphson Method; Neural Networks; Radial Basis Function Neural Networks.

I. NOMENCLATURE

ANNs: Artificial Neural Networks
LF: Load Flow
LM: Levenberg-Marquardt Algorithm
BP: Back-propagation Algorithm
ING: Iraqi National Grid
MLP: Multilayer Perceptron
N-R: Newton Raphson
PCA: Principal Component Analysis
PQ: Load Busbar
PV: Generator Busbar
RBFN: Radial Basis Function Network
SLFE: Static Load Flow Equations
VLSI: Very Large Scale Integration
B: Imaginary part of nodal admittance matrix
G: Real part of nodal admittance matrix
H, L, M, N: Jacobian submatrices
k: Busbar Index
ΔP: active power mismatch
ΔQ: reactive power mismatch
θ_k: voltage phase angle at bus k
V_k: voltage magnitude at bus k
α: Momentum parameter
η: Learning rate parameter

II. INTRODUCTION

The load flow calculation is one of the most basic problems in power engineering. Load flow or Power flow studies are conducted to determine the steady state operating condition of a power system, by solving the static load flow equations (SLFE) that mathematically are represented by a set of non-linear algebraic equations for a given network. The main objective of load flow (LF) studies is to determine the bus voltage magnitude with its angle at all the buses, real and reactive power flow (line flows) in different branches, and the transmission losses,…etc. It is the most frequently carried out study by power utilities and is required to be performed almost all the stages of power system planning, optimization, operation and control. During last four decades, almost all the known methods of numerical analysis methods for solving a set of non-linear algebraic equations have been applied in developing load flow algorithms. One or more desirable features to compare the different LF methods can be speed of solution, memory storage requirement, accuracy of solution and the robustness or reliability of convergence. However not all but only a particular combination of the various features is what will be needed in a given situation. For example, the memory requirement may be important only to small computers having low storage space. But, with the advent of modern digital computers, memory requirement is no more a limiting factor. Robustness or reliability of convergence of the methods is required in all types of applications. But, the speed of the solution is more important for on-line applications as compared to the off-line studies. The repetitive solution of a large set of linear equations in the load flow problem is one of the most time consuming parts of power system simulations. A straightforward implementation of these methods becomes inefficient, when large-scale networks exist, resulting in additional memory requirement and computing time.

For contingency selection, fast direct method, but iterative in nature approximate load flow methods, such as DC load flow method, linearised AC load flow, decoupled load flow, fast decoupled load flow methods are used,

which provide results having significant inaccuracies. Full AC load flow methods are accurate but become unacceptable for on-line implementation due to high computational time requirements. With the advent of artificial intelligence, in recent years, experts systems, pattern recognition, decision tree, neural networks and fuzzy logic methodologies have been applied to the security assessment problem. Amongst these approaches, the applications of artificial neural networks (ANNs) have shown great promises in power system engineering due to their ability to synthesize complex mappings accurately and rapidly. The artificial Neural Networks (ANNs) are gaining popularity in many engineering and scientific application due to their high computational rates, ability to handle nonlinear functions and a great degree of robustness. A single-layer ANN, separate MLP model based on Levenberg-Marquardt method have been used for computation for bus voltage magnitude and for angle at each bus of power system, and a radial basis neural network are proposed in this paper for on-line load flow studies. For the purpose of estimating the performance of the different types for the ANN algorithm, it has been tested on various scale test systems and practical system.

III. LOAD FLOW PROBLEM SOLUTION

The objective of power flow study is to determine the steady state conditions of a power system. For the purpose of power flow studies, it is assumed that the three-phase power system is balanced and also mutual coupling between elements is neglected. Variable associated with each bus of the power system include four quantities which are voltage magnitude V_k, its phase angle θ_k, real power P_k, and reactive power Q_k.

- Newton-Raphson method

The most widely used numerical method in solving the load flow problem is the Newton-Raphson method. The Newton-Raphson load flow equations are [1],

$$\Delta P_k = P_k^{sp} - V_k \sum_{m \in k} (G_{km}\cos\theta_{km} + B_{km}\sin\theta_{km})\, V_m \qquad (1)$$

$$\Delta Q_k = Q_k^{sp} - V_k \sum_{m \in k} (G_{km}\cos\theta_{km} - B_{km}\sin\theta_{km})\, V_m \qquad (2)$$

Where, $\vec{E_k} = V_k e^{j\theta_k}$, $\vec{I_k} = |I_k| e^{j\alpha_k}$, $\theta_{km} = \theta_k - \theta_m$, $\vec{E_m} = V_m e^{j\theta m}$, and $Y_{km} = |Y_{km}| e^{j\emptyset_{km}}$

$$\begin{bmatrix} \Delta P^t \\ \Delta Q^t \end{bmatrix} = \begin{bmatrix} H^t & N^t \\ M^t & L^t \end{bmatrix} \begin{bmatrix} \Delta\theta^{t+1} \\ \dfrac{\Delta V^{t+1}}{V^t} \end{bmatrix} \qquad (3)$$

$$\begin{bmatrix} H^t & N^t \\ M^t & L^t \end{bmatrix} = \begin{bmatrix} \dfrac{\partial \Delta P}{\partial \theta} & \dfrac{\partial \Delta P}{\partial V} \\ \dfrac{\partial \Delta Q}{\partial \theta} & \dfrac{\partial \Delta P}{\partial V} \end{bmatrix} \qquad (4)$$

Where the sub-matrices H, N, M and L form the Jacobian matrix and t is the iteration index. When the voltage corrections $\Delta\theta$ and ΔV are solved in (3) the new voltages are found from:

$$V_k^{t+1} = V_k^t + (\Delta V)^{t+1}$$
$$\theta_k^{t+1} = \theta_k^t + \Delta\theta_k^{t+1} \qquad (5)$$

The solution of (3) provides the correction vector i.e. $\Delta\theta$'s for all the PV and PQ type buses and ΔV's for all the PQ type buses, which are used to update the earlier estimates of θ's and V's. This iterative process is continued till the mismatch vector i.e. ΔP's for all the PV and PQ type buses and ΔQ's for all the PQ buses become less than a pre-assigned tolerance value (ε).

IV. ARTIFICIAL NEURAL NETWORKS

The intelligence of ANN and its capability to solve hard problems emerges from the high degree of connectivity that gives neurons its high computational power through its massive parallel-distributed structure. The current resurgent of interest in ANN is largely because ANN algorithms and architectures can be implemented in VLSI technology for real time applications [3]. The development of ANN involves two phases: training or learning phase and testing phase. Training of ANN is done by presenting the network with examples called training patterns. During training, the synaptic weights get modified to model the given problem. Soon as the network has learnt the problem it may be tested with new unknown patterns and its efficiency can be checked (testing phase). Depending upon the training important, ANN can be classified as supervised ANN or unsupervised ANN.

A. The One Layer Neural Network

A one layer neural network is characterized by a layer of input neurons and layer of output neurons
interconnected to one another by weights to be determined by the training process. This process is illustrated in (1).

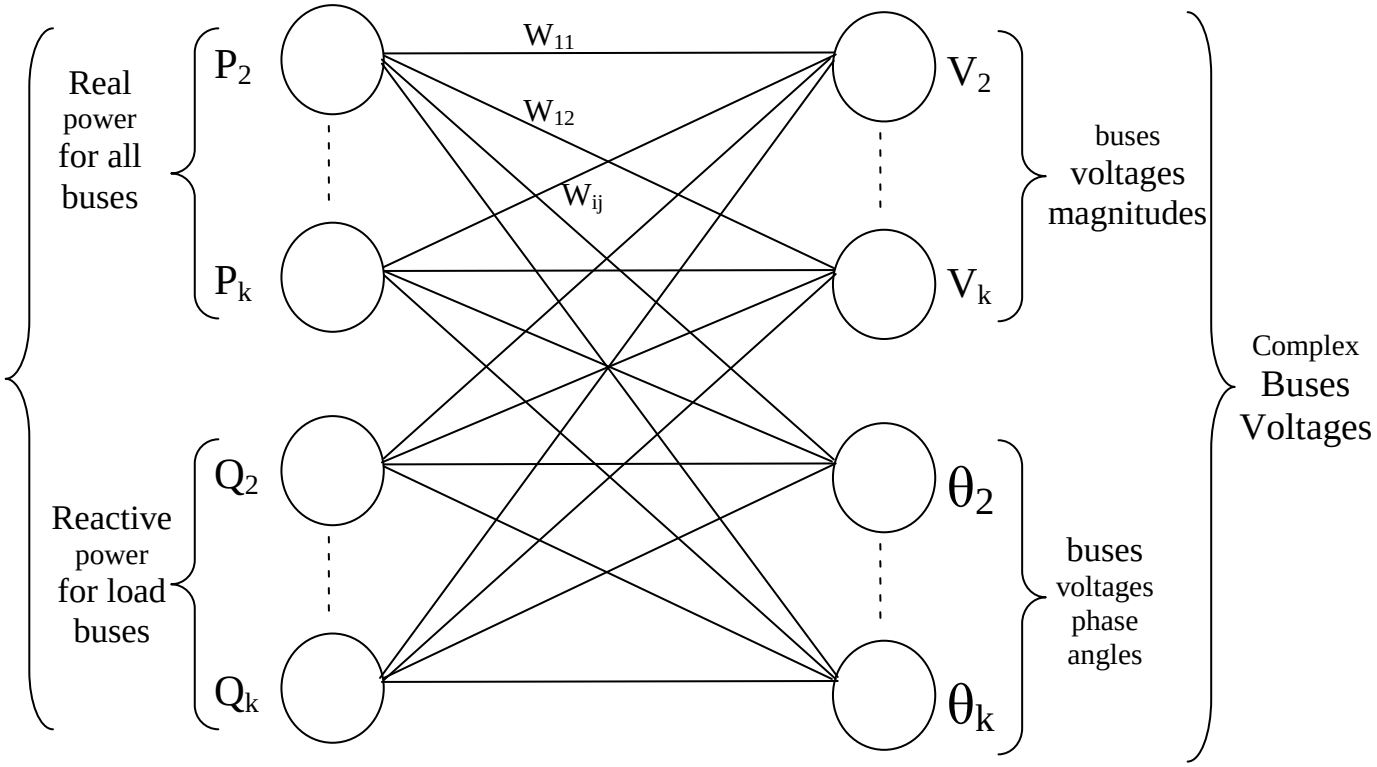

Figure (1) One-Layer Neural Network

A few configurations of the neural network are experimented with, and the best results are achieved with a single-layer feed forward neural network with nonlinear feedback. Using the trained neural network, an approximate solution of power flow can be obtained almost immediately. For application to power flow, the power system is linearised and then modeled by one layer of the forward neural network, as shown in (2). The input data are selected by using active and reactive loads added to diagonal elements of the bus admittances (G, B) respectively, and the output data are the complex bus voltages. Single layer neural network represents a linear system and it is obvious that results obtained for a nonlinear system such as a power system can be accurate. One possible solution is to introduce additional input layers to generate second and higher order nonlinear terms. This approach however, will result in significant increase of the size of a neural network and it will be impractical for large power systems to be analyzed.

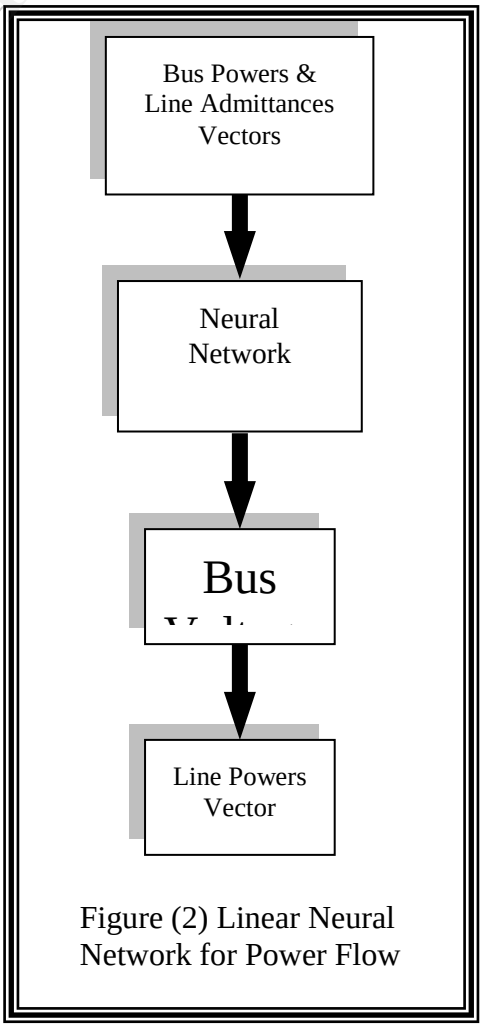

Figure (2) Linear Neural Network for Power Flow

A possible approach to increase accuracy is to use a feedback loop, as shown in (3). Line power vector can be directly computed from bus voltages and line impedances. Using simple summation with complex arithmetic, the input vector IN_F (bus powers) can obtained from line powers summation. At the initial state, the vector of line powers S_L is zero and there is no feedback – IN_F is zero. Therefore in the first step the input vector IN alone, is applied to the neural network and an approximate initial vector of bus voltages V_B is obtained. In the second step the difference between input vector IN and feedback vector IN_F is computed from line powers S_L and bus voltages V_B. Therefore the neural network operates on the difference (error) and the vector of the line powers is corrected.

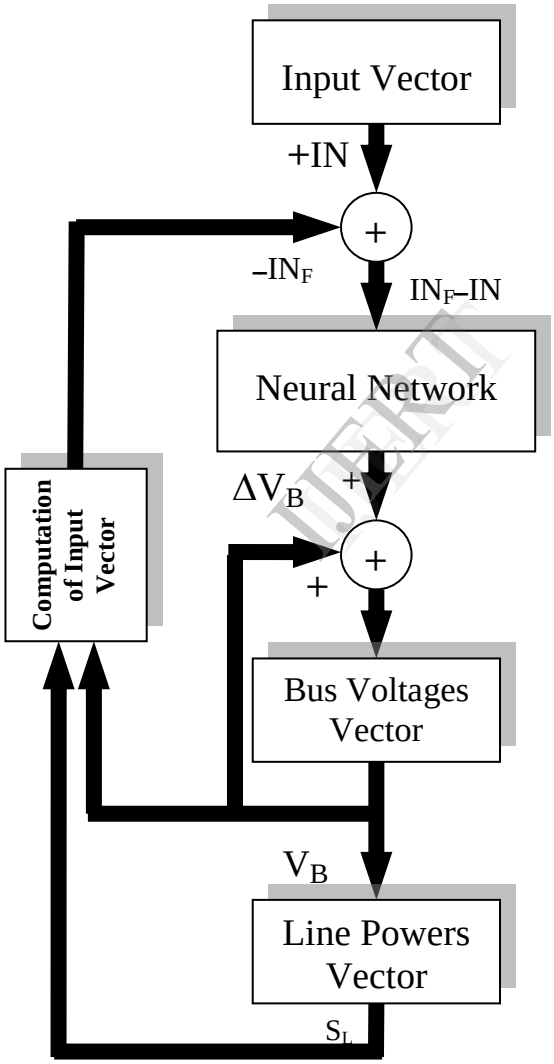

Figure (3) Neural Network with Feedback for power Flow Analysis

By adding the non-linear feedback, we can obtain significant improvement over the case with no feedback. Usually a few iterations are enough to obtain convergence as shown in the results section. The results are very much

comparable with those from a rigorous mathematical analysis, but the computational effort is negligibly smaller in comparison.

B. Multilayer Perceptron (MLP) Model Based Back Propagation Algorithm

All types of networks (MLP) discussed in the following sections use feed forward network architecture, consisting of an input layer, one or more hidden layer(s) and an output layer. Initially a random weight (usually in the range of -1 to +1) is assigned to each connection. These weights are then adjusted as learning progresses. The main difference between the network types lies in the type of activation function used by the hidden neurons. In MLPs, a common type of activation function used by the hidden neurons has a sigmoid transfer function. This sigmoid function divides a high-dimensional input space into two halves, with a high output in one half and a low output in the other, as illustrated in Figure (4). The back-propagation algorithm uses an objective function, which is defined to be the summation of square errors between the desired outputs and the network outputs [4]. It then employs a steepest-descent search algorithm to seek the minimum of the objective function. Training the MLP NN by using the standard (BP) Algorithm can be performed according to the following algorithm:

1. Initialization the network synaptic weights values.
2. Repeat the following steps until some criterion is reached:

For each training input-output pairs:

- Do a Forward Pass.
- Do a Back Pass.

Update weights.

Test network generalization, and Run the train network.

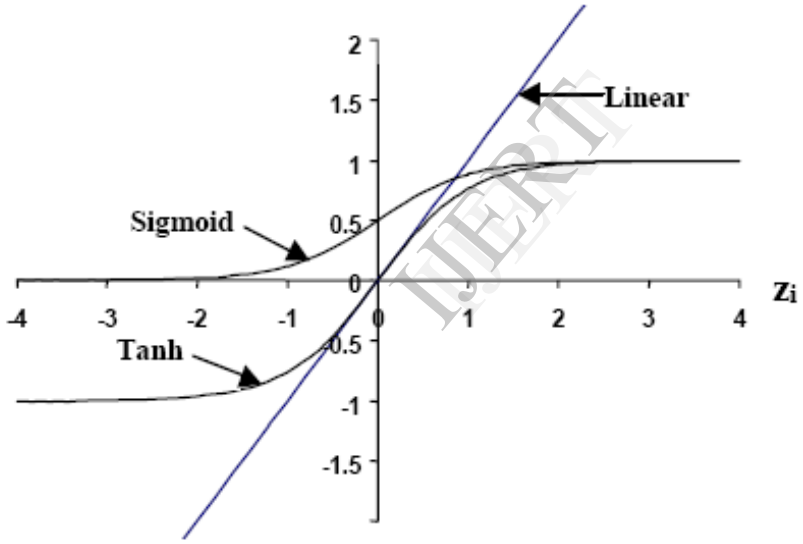

Figure (4) The Transfer Function of hidden nodes in MLP nets. ″The output layer neurons are sometimes linear″

C. Radial Basis Function Neural Network (RBFNN)

The RBFNN, whose structure is a three-layer ANN. The input vectors are transformed in vectors of an n-dimensional space by the n non-linear units (called basis functions) of the hidden layer. The weights of the output layer are easily computable by linear regression. Therefore, the input-output relationship is approximated by a linear combination of non-linear functions.

In RBF networks, hidden neurons usually have a Q Gaussian basis functions G(X,cj) at the center cj:

$$G(X,cj) = G(\|X-cj\|) = \exp\left(-\frac{\| X - c_j \|^2}{2\sigma^2}\right) \quad (6)$$

Here cj indicates the centre of the basis functions of the neuron and σ is its width. This function is selective to a small portion of the input space, as illustrated in (5). Where ||X-cj|| is the Euclidean distance between the input vector X and the cj, and σ is estimated by the following empirical formula.

$$\sigma = d_{max} / \sqrt{Q} \quad (7)$$

Where, dmax is the maximum Euclidean separation between the RBFN centers. And Q is the number of the RBFN centers [5].

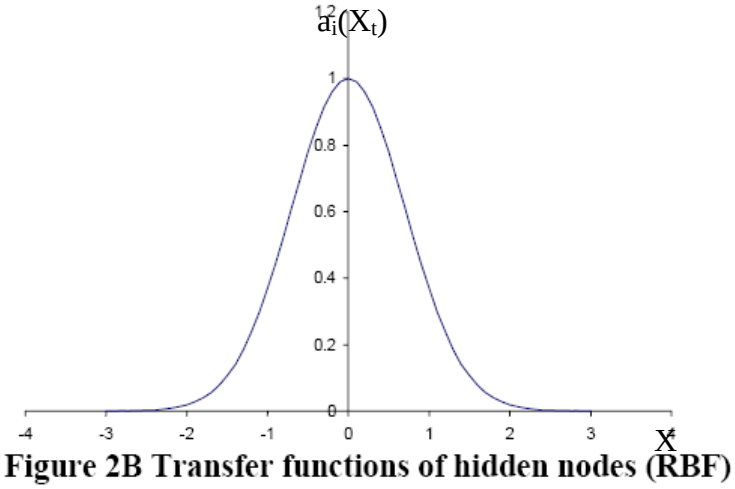

Figure (5) Transfer function of hidden nodes in RBF nets

The parameters of the RBF units are determined in three steps of the training activity. First, the unit centers are determined by some form of clustering algorithm. Then the widths are determined by a nearest neighbor method. Finally weights connecting the RBF units and the output units are calculated using multiple regression techniques. Euclidean distance based clustering technique has been employed to select the number of hidden (RBF) units and unit centers. The normalized input and output data are used for training of the RBF neural network. For the commonly used neural networks such as Multi-layer Perceptrons (MLP), the design of the network involves all the layers of the network simultaneously. The design of the hidden and output layer of an RBFNN can be carried out separately, at different points of time. The hidden layer applies non-linear transformation from the input space to the hidden space. The output layer is a linear combination of the activations in the hidden layer. The weights in the output layer are found by using linear optimization techniques. As described in the next section, the centers of the RBFNN for selected contingencies are chosen by using a sequential learning strategy. The optimal output weights are found for different contingencies, which linearly combine the activations of the same hidden layer to give the desired output for different contingencies [10].

1. Unsupervised Learning to select data centers of the training patterns

The well-known clustering algorithm is used to find center of desired number of clusters for the training patterns for each contingency. The steps of process of the unsupervised learning are described in the preceding section.

2. Selection of centers for Basis Function using sequential learning strategy

After data centers for the training patterns for the base case and the selected contingencies are found by k-means clustering algorithm. Let be the data center for the training pattern for the rth contingency; r = 1,2,…,g, where g is the number of selected contingencies; r=0 correspond to the base case. Number of data centers chosen to represent the training data set for the rth contingency is qr. The data centers are updated for each contingency by using a sequential learning strategy as described below.

Starting with the data centers for the base case, a new data center is added for a contingency, if the Euclidean distance of the particular data center from the nearest one in the existing set of data centers is more than a specified value α, which is set by experimentation. The steps for updating the data centers are summarized below:

a) The data centers for the base case topology are chosen as the initial centers for the RBFN. Let

the initial set of centers be designated by, S = , where, Q(0) denotes the number of centers at the beginning.

b) A new data centre c is added for rth contingency to the overall set of centers if the following

criterion is satisfied.

$$\min \|c - cj\ (r-1)\| \geq \alpha, \quad k=1,\ldots,qr; \ j=1,\ldots..Q(r-1) \quad (8)$$

c) The update set of centers is S= , where Q(r) is the number of centers after considering rth

contingency. The above steps are repeated till all the contingencies are considered in the overall set of centers.

3. Off-line Training of the RBFNN

Once the hidden layer is designed for the RBFNN by choosing desired number and locations of the centers of basis functions, the network can be trained with sample patterns for different contingencies.

Let { Xi,di} be the training patterns, where Xi is the vector of real and reactive load powers at buses and di is the corresponding the complex voltages, for any system topology. The optimal weight vector between the hidden layer and the output of the RBFNN is determined by linear optimization, which is described later. The same RBFNN is trained separately for different contingencies and the corresponding optimal weight vectors are recorded in the output weight vector matrix, W_M.

$$W_M = [\, w_0, w_1, \ldots..,w_c] \qquad (9)$$

The output value ym of the mth output node is given as:

$$y_m = \sum_{i=1}^{Q} w_{im} a_i (X_i) + w_{om} \qquad (10)$$

Where, Q is the number of the hidden layer, w_{om}: biasing term at mth output node, and ai(Xi), the output in the hidden layer for the input patterns.

Figure (6) shows the architecture of the proposed radial basis function neural network but without the synaptic weights between the input and hidden layers, these weights are used in case of MLP networks.

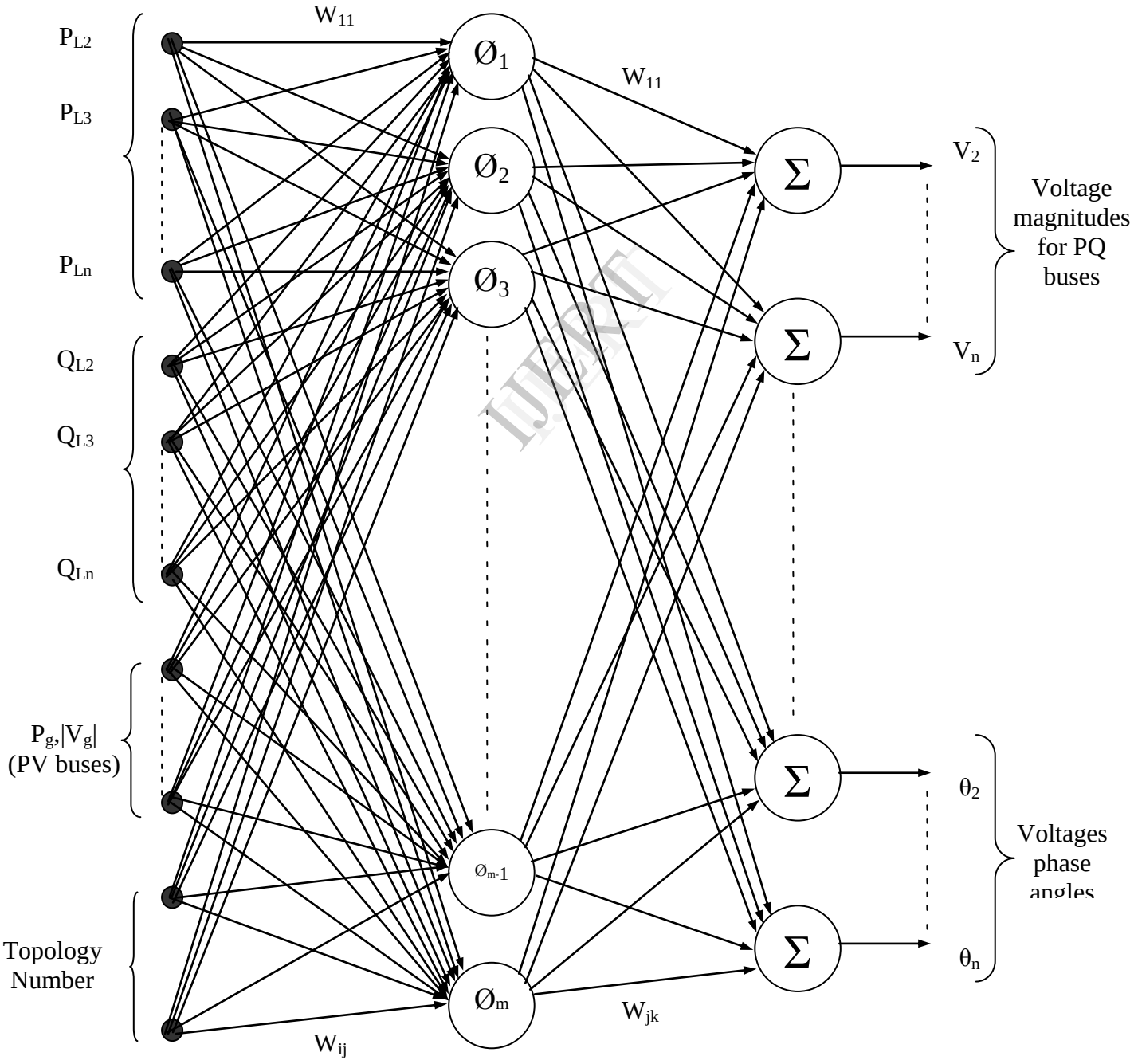

Figure (6) Radial Basis Function Neural Network or MLP Network model in Load Flow Solution

V. CHOICE OF INPUT PARAMETERS

Figures (7) & (8) show the architectures of the two models for the neural networks. The composition of the input variables for the proposed neural network has been selected to emulate the solution process of a conventional load flow program.

Input features to the MLP models shown in (7)

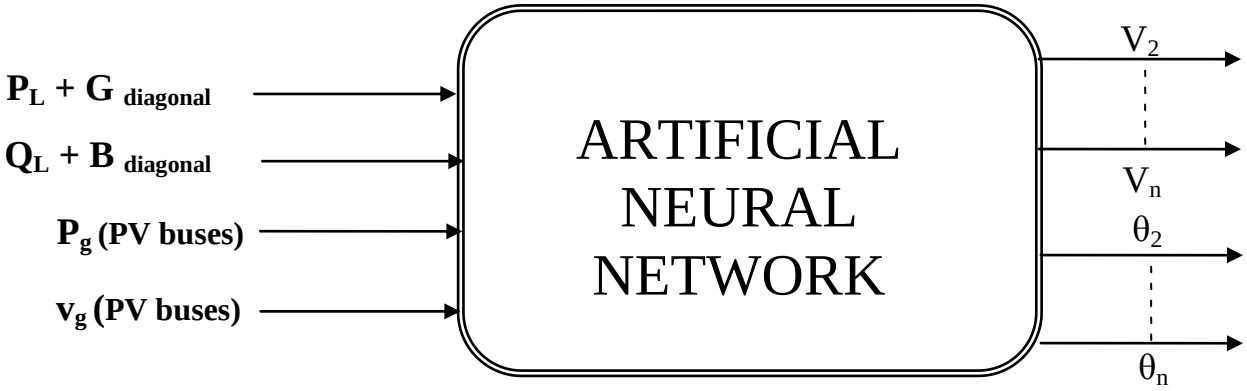

Figure (7) Model No.1 Proposed ANNs Architecture

- Input features to the RBF models shown in (8)

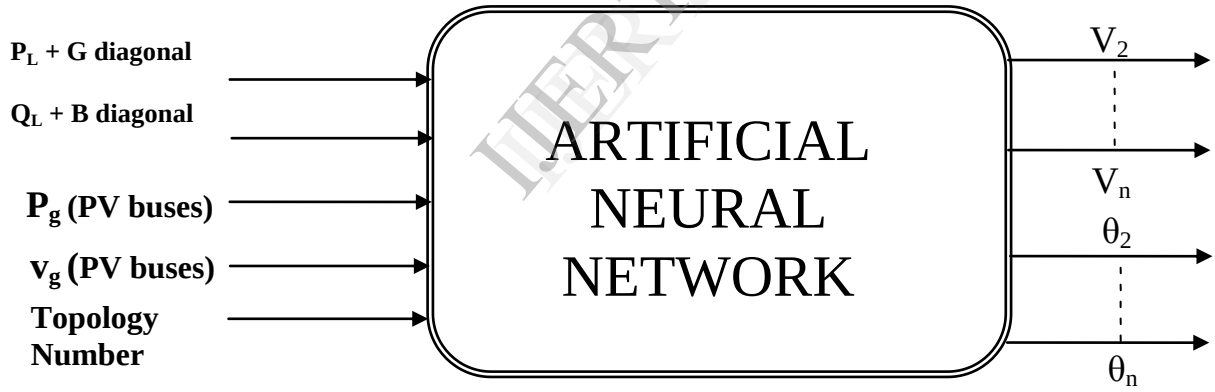

Figure (8) Model No.2 Proposed ANNs Architecture

The input consists of the electric network parameters represented by the diagonal elements of the bus conductance and susceptance matrix, voltage magnitudes V_g of generation, the active power generations Pg of PV buses. In order to speed up the neural network training, the conductance and susceptance are normalized between 0.1 and 0.9. Since, only one RBFNs with multi-output node is designed to predict the bus voltages for the base case as well as for the line outage cases, a topology number in the form of bipolar digits (+1 or –1) is used as an input to the RBFNs to represent the corresponding case. For example, the base case is represented by a bipolar string (-1 -1 -1 -1 -1) and the first line outage by (-1 -1 -1 -1 +1).

VI. DATA PROCESSING AND POST-PROCESSING

In general, the performance of a neural network is strongly dependent on the preprocessing that is performed on the training data [6]. The neural network training process can also be made more efficient if certain preprocessing steps are carried out on the input patterns and target values. That is, many times the "raw" data are not the best data to use for training a neural network. The preprocessing and post-processing for input data of artificial neural networks are as follows:

A. Data Scaling

The training data can be amplitude-scaled in basically two ways: so that the values of the pattern lie between -1 and 1.

B. Dimensionality Reduction

Dimensionality reduction is another area, which reduces the number of patterns required for network training and hence network complexity. Using statistical analysis and dimensional analysis and combining the number of variables to a smaller set of input variables are useful methods for optimizing the number of input and output parameters. Principal Component Analysis (PCA) can be used to "compress" the input training data set (or reduce the dimension of the inputs). The resulting "compressed" input vectors will have elements that are uncorrelated.

C. Removing Data Outside and Data Selection for the ANN Training

In order to ensure that the network has properly mapped input training data to the target output, it is essential that the set of patterns presented to the network is appropriately selected to cover a good sample of the training domain. A well trained network is one which is able to respond to any unseen pattern within an appropriate domain. At present NNs. are not good at extrapolating information outside the training domain.

D. Training Modes

Training a neural network involves gradual reduction of the error between neural network output and the target output. Generally, there are two different modes of training neural networks: batch mode and pattern mode. In a batch mode, when an epoch is completed (i.e. when the entire set of training data is presented to the network) a single average error is calculated and the weights in the network are adjusted according to that error. In a pattern mode, the error is calculated after each pattern is presented to the network, and network weights are adjusted.

VII. IMPLEMENTATION AND RESULTS

The effectiveness of the proposed ANNs models are demonstrated by computation of bus voltage magnitudes and angles following different loading/contingency conditions in the following test systems, 5 buses test system [12], IEEE 14-bus system, IEEE 30-bus system, and 362-bus practical system, Iraqi National Grid (ING).

A. Solution Algorithm of the Robustness Proposed Method

The solution algorithm for load flow problem using RBF networks is as follows:

1. A large number of load patterns are generated randomly by spreading the load at all the buses, real power generation at the generator buses, voltage magnitudes at *PV* and slack buses.
2. Principal Component Analysis (PCA) is applied to compress the input training data set thus, reduce the input and output variables.
3. AC load flow (NR) programs are run for all the load patterns and also for contingency cases to calculate bus voltage magnitudes at all the *PQ* type buses and voltage angles at all the *PV* and *PQ* type buses.
4. Input features for RBF (P_i and Q_i) are selected on the basis of entropy gain, voltage magnitude at PV and real power generations at PV buses.
5. The number of hidden (RBF) units and unit centers are determined using Euclidean distance based clustering technique. Then width of the RBF unit is determined. While MLP model, the number of hidden nodes could be decided using some trial and error method.
6. For training of the ANNs, initialize all the connection weights between the hidden nodes and output nodes.
7. Calculate the output of the ANNs.
8. Calculate the Mean Squared Error e_p for the p^{th} pattern using

$$e_{p=}\frac{1}{2}\cdot\frac{1}{no}\sum_{j=1}^{no}(T_{jp}-L_{jp})^2 \qquad (11)$$

Where, no = number of neurons in output layer
T_{jp} = target value at j^{th} neuron of output layer
L_{jp} = actual output at j^{th} neuron for p^{th} pattern

9. Repeat steps (6) & (7) for all the training patterns.
10. Calculate the error function E_k using the following equation:

$$E_k=\sum_{p=1}^{P_{max}} e_p=\sum_{p=1}^{P_{max}}\frac{1}{no}\sum_{j=1}^{NO}(T_{jp}-L_{jp})^2 \qquad (12)$$

11. The connection weights w_{ji} between the hidden nodes and -output nodes at K^{th} iteration are updated using equations

$$w_{ji}(k+1)=w_{ji}(k)+\Delta w_{ji}(k) \qquad (13)$$

Where,

$$\Delta w_{ji}(k)=\eta(k).\delta_j.A_i+\alpha.\Delta w_{ji}(k-1) \qquad (14)$$

$\delta_j = T_j - Wj\ .A_i$
$\eta(k)$ = learning rate or adaptive size at K^{th} iteration
δ_j = error signal for unit j, α = Momentum term
$T_j = [\ t_{j1}, t_{j1,\ldots\ldots\ldots}, t_{jp}^{max}]$
$W_j = [\ w_{j1}, w_{j1,\ldots\ldots\ldots}, w_{jo}]$
for i =1,2,……Q+1, Q = number of hidden layer (RBF) nodes.
T_i = target value at j^{th} neuron of output layer,
w_i = The weights between the hidden layers and output layers
The procedure is continued till the error becomes negligible.

Two RBFNNs were developed in this work, one (RBFNN1) for computation of bus voltage magnitude at all the PQ type buses, while the other (RBFNN2) for computation of bus voltage angle at PV type and PQ type buses are shown in (9 & 10). After training, the knowledge about the training patterns in form of voltage magnitudes at all the PQ buses and voltage angles at different PV and PQ buses in various contingency cases and different system operating conditions are stored in structured memory by the trained RBFNNs.

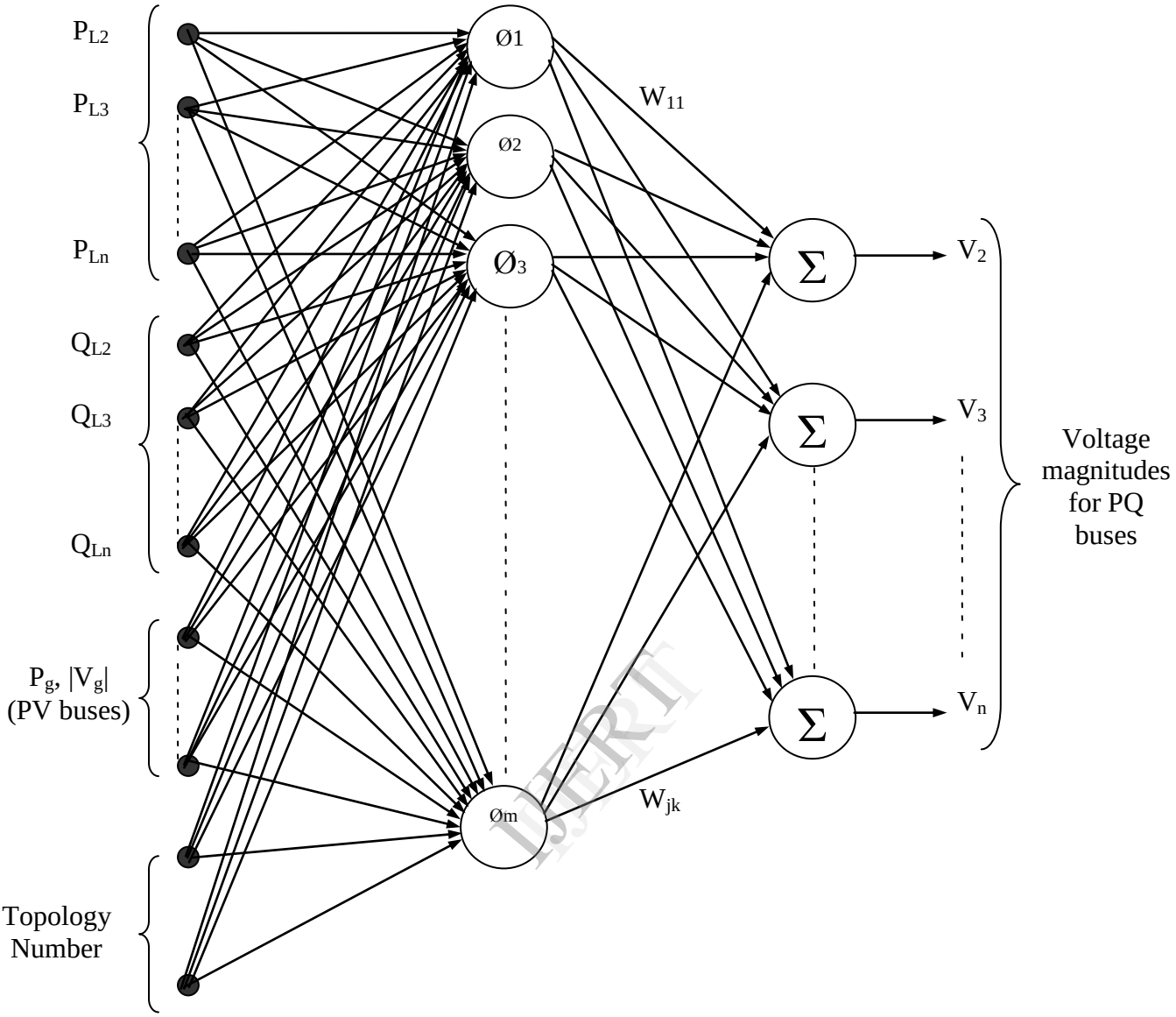

Figure (9) RBFNN1 model in Load Flow Solution (Voltage Magnitudes for PQ buses)

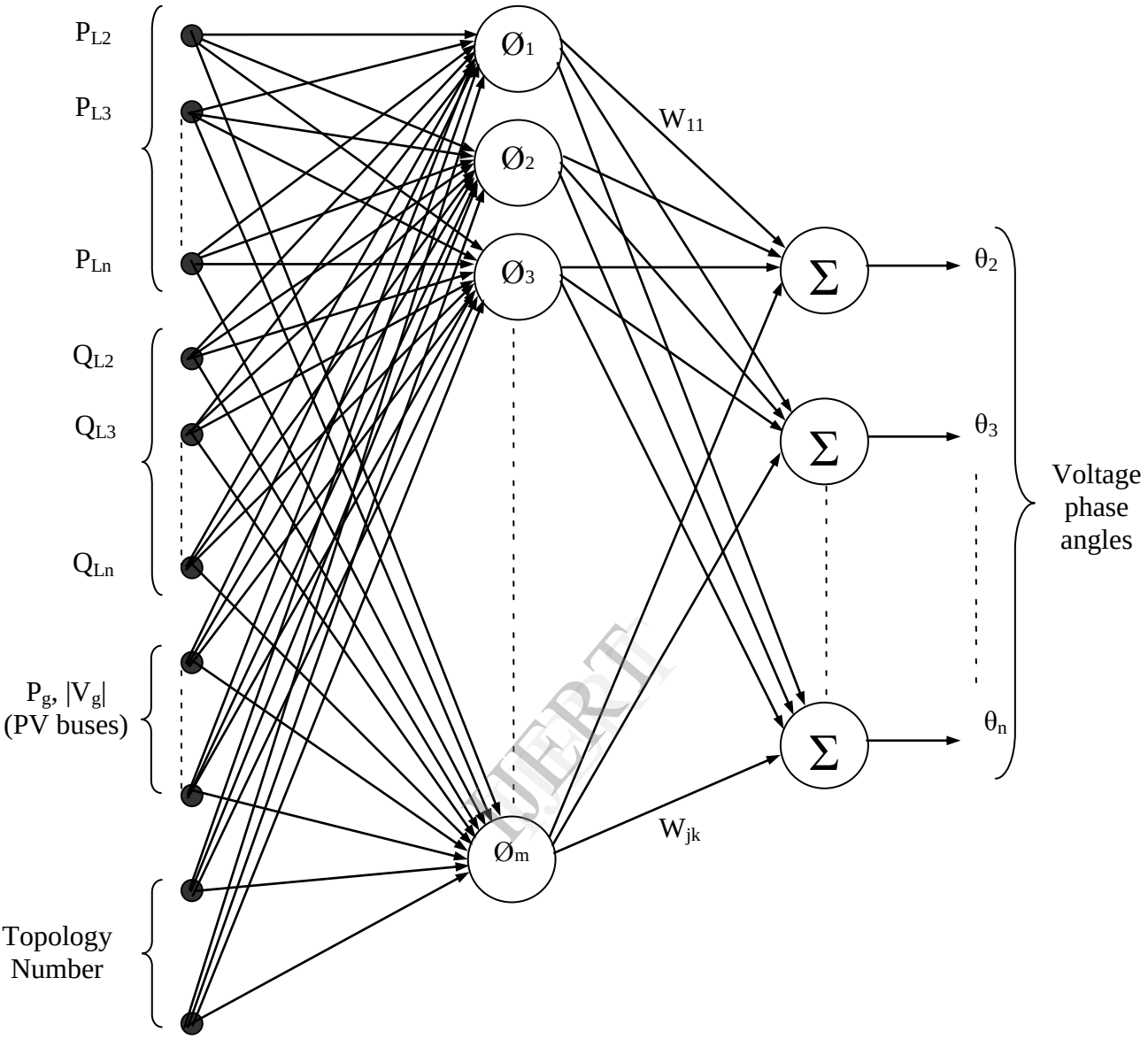

Figure (9) RBFNN2 model in Load Flow Solution (Voltage Angles)

B. Training and Testing Patterns of the ANN Models

For training and testing of ANNs, It is changed the load at each bus randomly from 60% to 140% of their base values, PV bus voltage magnitudes between 0.9 to 1.1 and real power generation in the range of 80% to 120%. Single-line outages were considered as contingencies as shown in (1).

Table (1) Training and Testing of ANN for different type of systems

Type of System	Training sets	Testing sets	Total of Patterns
5-bus	160	40	200
IEEE 14-bus system	418	95	513
IEEE 30-bus system	836	190	1216
362-bus (Iraqi National Grid)	1500	500	2000

For the purpose of estimating the performance of the different types for the ANN algorithm, IEEE 14-bus test system is used, which is composed of 14 buses and 20 lines, the data for IEEE14-bus system were taken from [8]. Table (2) shows the Load-Flow Solution for ″IEEE 14-Busbar″ test system using single layer with and without nonlinear feedback.

Table (2) Load-Flow Solution for ″IEEE 14-Busbar″ Test System
Using single layer with and without nonlinear feedback

Bus No.	Bus* Type	Load flow solution Without Feedback		Load flow solution With feedback		Load flow solution N-R Method	
		V (p.u.)	θ (deg.)	V (p.u.)	θ (deg.)	V (p.u.)	θ (deg.)
1	1	1.06	0	1.06	0	1.06	0
2	2	1.045	-4.942	1.045	-4.94	1.045	-4.955
3	2	1.01	-12.604	1.01	-12.62	1.01	-12.625
4	0	1.0278	-10.356	1.0271	-10.36	1.0271	-10.377
5	0	1.0343	-8.935	1.0334	-8.94	1.0334	-8.955
6	2	1.07	-14.820	1.07	-14.83	1.07	-14.880
7	0	1.0459	-13.415	1.0453	-13.42	1.0453	-13.459
8	2	1.09	-13.410	1.09	-13.459	1.09	-13.459
9	0	1.0285	-15.032	1.0281	-15.1	1.0281	-15.077
10	0	1.0283	-15.276	1.0279	-15.325	1.0279	-15.325
11	0	1.0455	-15.162	1.0451	-15.31	1.0451	-15.217
12	0	1.0533	-15.660	1.0531	-15.721	1.0531	-15.721
13	0	1.0465	-15.681	1.0463	-15.740	1.0463	-15.740
14	0	1.0181	-16.347	1.0177	-16.35	1.0177	-16.399
Computation Time without feedback = 0.0677 second Computation Time with feedback = 0.0270 second							

* Numbers appearing in this column, being as follows:
(0) Stands for PQ buses, (1) Stands for Slack busbar, and (2) Stands for PV buses.
The solid rectangular row represents the PV buses and the others buses represent the PQ buses.

C. Application of Multilayer (MLP) NN Model for Load Flow Analysis

Solution of load flow problem had been done using the feed-forward neural network based on the Levenberg-Marquardt (LM) back propagation Algorithm. Training is a procedure used to minimize the difference between outputs of MLP and
the desired values by adjusting the weights of the network. Sets of input vectors are presented to the network until training is completed. Then the network's weights are "frozen" in the trained state and the new input data are presented to the network to determine the appropriate output.

1. Network Topology with One Hidden Layer (MLP)

A single hidden layer with an optimum number of neurons will be sufficient for modeling to solve load flow problems. Table (3) shows Load-Flow Solution for ″14-Busbar″ IEEE test system using MLP with Single Hidden layer

Table (3) Load-Flow Solution for ″14-Busbar″ IEEE Test System
Using MLP with Single Hidden layer

Bus No.	Bus Type	Load flow solution			Load flow solution		
		V (p.u.)	V (p.u.)	Absolute Error	θ (deg.)	θ (deg.)	Absolute Error
		N-R Method	BP-LM Method		N-R Method	BP-LM Method	
1	1	1.06	1.06	Slack	0	0	Slack
2	2	1.045	1.045	PV-Bus	-4.955	-4.8433	0.1117
3	2	1.01	1.01	PV-Bus	-12.6258	-12.724	0.0982
4	0	1.0271	1.0252	0.0019	-10.3777	-10.2805	0.0972
5	0	1.0334	1.0319	0.0015	-8.9559	-8.5309	-0.425
6	2	1.07	1.07	PV-Bus	-14.8809	-14.9779	0.097
7	0	1.0452	1.0486	0.0034	-13.4591	-12.9801	0.479
8	2	1.09	1.09	PV-Bus	-13.4591	-13.4099	0.0492
9	0	1.028	1.0261	0.0019	-15.078	-15.1977	0.1197
10	0	1.0279	1.0269	0.001	-15.3251	-15.3771	0.052
11	0	1.0451	1.0489	0.0038	-15.2179	-14.972	0.2459
12	0	1.053	1.0527	0.0003	-15.7213	-15.7314	0.0101
13	0	1.0463	1.0437	0.0026	-15.7407	-15.5156	0.2251
14	0	1.0177	1.0182	0.0005	-16.3991	-16.3775	0.0216

Input neurons=32
Output neurons=22
Neurons in hidden layer=72
Momentum = 0.6
Training Patterns = 418
Test Patterns = 95

Total number of epochs = 180
Time of Training = 533.81 sec
Time of Simulation = 0.023 sec

2. Network Topology with Two Hidden Layer (MLP)

A neural network with one hidden layer was tried first, but was found hard to converge. Thus, a neural network with two hidden layers was selected for further analysis. This network converges quickly and is more accurate than the single hidden layer. Hyperbolic tangent sigmoid transfer functions are used for the hidden layers and a linear transfer function is used for the output layer as shown in (4).

Table (4) Load-Flow Solution for ″14-Busbar″ IEEE Test System Using MLP with Two Hidden Layers

Bus No.	Bus Type	Load flow solution			Load flow solution		
		V (p.u.)	V (p.u.)	Absolute Error	θ (deg.)	θ (deg.)	Absolute Error
		N-R Method	BP-LM Method		N-R Method	BP-LM Method	
1	1	1.06	1.06	Slack	0	0	Slack
2	2	1.045	1.045	PV-Bus	-4.955	-4.9416	0.0134
3	2	1.01	1.01	PV-Bus	-12.6258	-12.6003	0.0255
4	0	1.0271	1.0267	0.0004	-10.3777	-10.3617	0.016
5	0	1.0334	1.033	0.0004	-8.9559	-8.9418	0.0141
6	2	1.07	1.07	PV-Bus	-14.8809	-14.8509	0.03
7	0	1.0452	1.0448	0.0004	-13.4591	-13.4344	0.0247
8	2	1.09	1.09	PV-Bus	-13.4591	-13.4319	0.0272
9	0	1.028	1.0276	0.0004	-15.078	-15.0507	0.0273
10	0	1.0279	1.0275	0.0004	-15.3251	-15.2975	0.0276
11	0	1.0451	1.0446	0.0005	-15.2179	-15.1908	0.0271
12	0	1.053	1.0525	0.0005	-15.7213	-15.6941	0.0272

13	0	1.0463	1.0458	0.0005	-15.7407	-15.7135	0.0272
14	0	1.0177	1.0172	0.0005	-16.3991	-16.3704	0.0287

Input neurons = 32	Training Patterns = 418
	Test Patterns = 95
Output neuron = 22	Total number of epochs = 51
Neurons in hidden layer 1 = 10	Time of Training = 87.112 sec
Neurons in hidden layer 2 = 10	Time of Simulation = 0.07 sec
Momentum = 0.6	

D. Application of RBF Neural Network Model for On-Line Load Flow Analysis

Two RBF neural networks are developed in this work, one (RBFN1) for computation of bus voltage magnitude at all PQ type buses, while the other (RBFN2) for computation of bus voltage angle at PV type and PQ type buses. The bus voltage magnitudes and angles are affected by several parameters of the power system. Some of them are having larger effect and some are having lesser impact. It is not necessary to use all the available variables to train the RBFN. It will increase the number of input nodes and will result in a complex structure, requiring large training time.

An approach based on system entropy has been used to identify the input features, i.e. real and reactive loads affecting the bus voltage most. The term entropy has been used to describe the degree of uncertainty about an event. A large value of entropy indicates high degree of uncertainty and minimum information about an event.

A topology number in the form of five bipolar digits (+1 or –1) is used as an input to the RBFNs to represent the corresponding case. For example, the base case is represented by a bipolar string (-1 -1 -1 -1 -1) and the first line outage by (-1 -1 -1 -1 +1). Thus the total input features used to train the RBFN are 25 and 27 in RBFN1 and RBFN2 respectively. Two RBFNs were developed, one for computation of bus voltage magnitudes at 9 PQ type buses, while the other for computation of bus voltage angles at 4 PV type buses and 9 PQ type buses (total 13). The optimum structures of the neural networks were found to be 25-284-9 for RBFN1 and 27-273-13 for RBFN2.

The different models of ANNs had been tested on various scale test systems and practical system. Specially, the effectiveness of different approaches of ANNs was examined through three test systems as well as the practical system (Iraqi National Grid). The size of the test systems varies from a few buses up to about 362 buses. The following tables show input-output of ANNs, number of epochs and time of training for all typical test systems. The training of ANNs and simulations were implemented on a Pentium 4 personal computer, 3 GHz processor, 2 Gbytes RAM with 1 Gbyte internal cache memory.

Table (5) Feature Selection for IEEE 14-Bus System in RBFNN1

Feature Selection method	No. of Feature Selected	Features
Entropy Reduction Method	20	$P_2,P_3,P_6,P_9,P_{10},P_{13},P_{14}$ $Q_4,Q_5,Q_9,Q_{10},Q_{11},Q_{12},Q_{13},Q_{14}$ Pg_2, $\|Vg_2\|,\|Vg_3\|,\|Vg_6\|,\|Vg_8\|$

Table (6) Feature Selection for IEEE 14-Bus System in RBFNN2

Feature Selection method	No. of Feature Selected	Features
Entropy Reduction Method	22	$P_2,P_3,P_4,P_5,P_6,P_9,P_{10},P_{11},P_{12},P_{13},P_{14}$ $Q_2,Q_3,Q_6,Q_9,Q_{10},Q_{13}$ Pg_2, $\|Vg_2\|,\|Vg_3\|,\|Vg_6\|,\|Vg_8\|$

Table (7) Load-Flow Solution for ″14-Busbar″ IEEE Test System
Using RBFN1 (25-284-9) & RBFN2 (27- 273 -13)

Bus No.	Bus Type	Load flow solution			Load flow solution		
		V (p.u.)	V (p.u.)	Absolute Error	θ (deg.)	θ (deg.)	Absolute Error
		N-R Method	RBFN1		N-R Method	RBFN2	
1	1	1.06	1.06	Slack	0	0	-
2	2	1.045	1.045	PV-Bus	-4.955	-4.932	0.023
3	2	1.01	1.01	PV-Bus	-12.6258	-12.573	0.0528
4	0	1.0271	1.0268	0.0003	-10.3777	-10.3321	0.0456
5	0	1.0334	1.033	0.0004	-8.9559	-8.9165	0.0394
6	2	1.07	1.07	PV-Bus	-14.8809	-14.821	0.0599
7	0	1.0452	1.0449	0.0003	-13.4591	-13.4036	0.0555
8	2	1.09	1.09	PV-Bus	-13.4591	-13.4055	0.0536
9	0	1.028	1.0277	0.0003	-15.078	-15.0162	0.0618
10	0	1.0279	1.0276	0.0003	-15.3251	-15.2618	0.0633
11	0	1.0451	1.0447	0.0004	-15.2179	-15.1547	0.0632
12	0	1.053	1.0526	0.0004	-15.7213	-15.6572	0.0641
13	0	1.0463	1.0459	0.0004	-15.7407	-15.6769	0.0638
14	0	1.0177	1.0173	0.0004	-16.3991	-16.3319	0.0672

Input neurons = 25(RBFN1),27(RBFN2), Test Patterns = 95
Output neuron = 9(RBFN1),13(RBFN2), Total number of epochs = 250
Neurons in hidden layer RBFN1 = 284, Time of Training (RBFN1) = 16.39 sec
Neurons in hidden layer RBFN2 = 273, Time of Training (RBFN2) = 15.88 sec
Momentum = 0.9 Time of Simulation = 0.016 sec
Training Patterns = 418

Table (8) Network Topology with One Hidden Layer (MLP).

Type of System	Input	Output	Structure	No. of epochs	Time of Training (sec)
5-Bus	11	8	11-25-8	90	33.845
14-Bus IEEE	32	22	32-72-22	180	533.81
30-Bus IEEE	63, (PCA)22	53, (PCA)5	22-45-5	588	307.13

Table (9) Network Topology with Two Hidden Layer (MLP).

Type of System	Input	Output	Structure	No. of epochs	Time of Training (sec)
5-Bus	11	8	11-10-10-8	51	87.112
14-Bus IEEE	32	22	32-15-10-22	126	156.38
30-Bus IEEE	63	53	(PCA)22-25-10-5	111	265.9
362-Bus ING (Angles)	716	361	(PCA)38-36-36-53	1300	3069.55
362-Bus ING (Voltage Mag.)	716	332	(PCA)38-36-36-53	1476	4448.89

Table (10) Network Topology for Radial Basis Function Neural Network.

Type of System	Input	Output	Structure	No. of epochs	Time of Training (sec)
14-Bus IEEE (RBFN1)	25	9	25-284-9	250	16.39
14-Bus IEEE (RBFN2)	27	13	27-273-13	250	15.88
30-Bus IEEE (RBFN1)	38	24	38-595-24	575	226.18
30-Bus IEEE (RBFN2)	42	29	42-463-29	450	142.24
ING-RBFN1	322	332	322-1016-332	1000	1350.55
ING-RBFN2	303	361	303-1470-361	1450	2053.2

VIII. DISCUSSION

Tables (3) and (4) prove that multi-layer perceptron (MLP) NN with two hidden layer is better than MLP NN with one hidden layer. For both techniques error back-propagation learning strategy with Levenberg-Marquardt minimization technique, application of update momentum, and sigmoid transfer function are used. It is better according to the following criteria: a) less absolute errors or more accurate results, b) time of training is less, c) time of simulation or real-time implementation is less, d) number of epochs for an efficient learning strategy and NN generalization is much less. The multi-layer perceptron feed-forward neural network model based back-propagation (BP) training algorithm uses standard numerical optimization techniques. Three types of these numerical optimization techniques are Conjugate gradient, Levenberg-Marquardt, and Quasi-Newton algorithms. All these minimization algorithms were used and tested. We found that Levenberg-Marquardt algorithm is the best in back-propagation training method because it is faster than the other algorithms and can converge from ten to one hundred times faster than the other mentioned algorithms.

Since ING is a large and practical power system so, it is very important and efficient to simplify the NN architecture by reducing the input and output neurons through the use of principal component analysis (PCA) by applying the algorithm of entropy gain and dimensionality reduction. Table (9) shows that the input neurons (nodes) and the output neurons were 716 and 332 neurons respectively without PCA application while, they became 38 and 53 neurons respectively with PCA application.

The numbers of hidden nodes in RBF networks are determined by the clustering algorithm but in MLP it is difficult to decide about the number and size of hidden layers so, a trial method is used. Radial basis function networks can require more neuron than standard feed-forward back-propagation networks, but often they can be designed in a fraction of the time it takes to train standard feed-forward networks. They work best when many training vectors are available. Radial basis networks need more neurons than a comparable feed-forward networks, this is because sigmoid neurons can have outputs over a large region of the input space, while radial basis function neurons only respond to relatively small regions of the input space. The result is that the larger the input space (in terms of number of inputs, and the ranges those inputs vary over) the more radial basis function neurons required.

IX. CONCLUSIONS

In this research, the solution of the load flow problem using artificial neural networks was achieved in a very short computing time for all systems of various sizes under different contingencies. The ANN has been trained only once, will operate for any load condition operation with no outages as well as for operating conditions under contingencies of generator and line outages, very accurate results could be obtained without the need for changing the topology of the network under contingencies.

Neurocomputing has attractive features, such as its ability to tackle new problems which are hard to define or difficult to solve analytically, its robustness in dealing with incomplete or "fuzzy" data, its processing speed, its flexibility and ease of maintenance.

Radial basis neural network have been developed to solve load flow problem in an efficient manner and reduce the possibility of ending at a local minima. In the commonly used multi-layer perceptron feed-forward neural network model based back-propagation (BP) algorithm, this usually suffers from local minima and over-fitting problems. The training process of MLP is slow, and its ability to generalize a pattern-mapping task depends on the learning rate and the number of neurons in the hidden layer. On the other hand training of a radial basis neural network is very fast, at the same time the generalization capability of the RBFN network allows it to produce an accurate output even when it is given an input vector that is partially incomplete or partially incorrect. The RBFNN has many advantageous features such as optimized system complexity, minimized learning and recall times as compared to multilayer perceptron model.

The proposed method (RBFNN) can be applied for on-line (real-time) load flow solution for both small and large-scale power systems with high accurate results.

REFERENCES

[1] Kubba, H. A. and Krishnaparandhama, T., "Comparative Study of Different Load Flow Solution Methods", *Al-Muhandis, Refereed Scientific Journal of Iraqi Engineers Society*, Vol. 107, December 1991, pp. 25-46.

[2] Dhar R.N., "computer Aided Power System Operation and Analysis", *Mc-GRAW-HILL Publishing Company Limited*, 1982.

[3] Tarafdar Haque M., Kashtiban A. M., ″Application of Neural Networks in Power Systems; A Review″, *Trans. on Eng., Computing and Tech.* Vol.6, 2005, ISSN 1305-5313.

[4] Haykin S., "Neural Networks-A Comprehensive Foundation", *2nd edition, Upper Saddle River, NJ: Prentice Hall*, 1999.

[5] Rafiq M.Y., Bugmann G. and Easterbrook D.J., "Neural Network Design for Engineering Applications", *International Journal of Computers & Structures*, Vol. 79/17, Sept. 2001, pp. 1514-1552.

[6] Ranaweer D.K., Hubele N.F., and Papalexopoulos, ″Application of radial basis Function neural network model for short term load forecasting″, *IEE Proc. Gener. Transm. Distrib.* Vol. 142, No.1, January 1995.

[7] Moody J. and Darken C.J., ″Fast Learning in networks of locally tuned processing units″, *Neural Computing*, 1989, pp. 281-294.

[8] Freris, L.L. and Sasson, A.M., "Investigation of the load-load problem", *Proc. IEE, Vol. 115, No.10*, October 1968, pp. 1459-1470.

[9] Ham F.M. and Kostanic I. ,"Principles of Neurocomputating for Science and Engineering ", *International Edition, McGraw – Hill, Book Company*, 2001.

[10] Jain T., Srivastava L., and Singh S.N.," Parallel Radial Basis Function Neural Network Based Fast Voltage Estimation for Contingency Analysis", *IEEE Inter. Conference on Electric Utility Deregulation*, Hong Kong, April 2004.

[11] Nitin Malik, "Artificial neural networks and their applications", *National conference on Unearthing Technological Developments' GLA ITM, Mathura*, India 17-18 April 2005.

[12] Stagg G.w. and Al-Abiad A., ″Computer methods in power system analysis″, *Mc-Graw Hill book company*, 1968.

APPENDIX A

Table A.1 Load Flow Solution Results Using Newton-Raphson Method for IEEE 14-Bus system for power Mismatch = 0.001p.u (0.1 MW/MVAR)

Bus Number	Bus Type	Voltage Mag.	Voltage Ang. (Deg.)
1	1	1.060	0
2	2	1.045	-4.955
3	2	1.01	-12.6258
4	0	1.0271	-10.3777
5	0	1.0334	-8.9559
6	2	1.07	-14.8809
7	0	1.0452	-13.4591
8	2	1.09	-13.4591
9	0	1.028	-15.078
10	0	1.0279	-15.3251
11	0	1.0451	-15.2179
12	0	1.053	-15.7213
13	0	1.0463	-15.7407
14	0	1.0177	-16.3991