# Real-Time Implementation of Cellular Neural Networks for Smart Grid Applications

Bipul Luitel, *Student Member, IEEE*
Advisor: Dr. Ganesh K Venayagamoorthy, *Senior Member, IEEE*

***Abstract*—Cellular neural network (CNN) consisting of multilayer perceptrons (MLP) on each of its cells is called a cellular multilayer perceptron (CMLP). In this research, CMLP is used to develop a wide area monitor for a Smart Grid. The study is carried out in personal computer (PC) using MATLAB, a non real-time platform, as well as on a digital signal processor (DSP), a real-time platform. The non real-time results obtained by CMLP are also compared with that of an MLP. The study is unique in its implementation of a CMLP for given application, in its training approach and real-time implementation by interfacing the real-time digital simulator with the DSP. The results show scalability of the CMLP architecture for Smart Grid applications and the feasibility of its use in a real-time application.**

***Index Terms*—Backpropagation, CNN, Cellular Multilayer Perceptron, Hardware Implementation, MIMO, Power system, real-time, Smart Grid, Wide Area Monitor**

## I. Introduction

Cellular neural networks (CNN) was introduced by Chua and Yang in 1988 [1]. It consists of individual units (cells) connected to each of its neighbors on a cellular structure. Each cell of the CNN is a computational unit. Most of the applications of this architecture of CNN are focused on pattern recognition [2], [3] and image processing [4], [5]. Another form of CNN that exists in literatures is the structure containing one or more forms of neural network (NN) architectures at each cell. Such CNN can be symmetric with each cell consisting of the same type and size of NN or asymmetric where different cells may consist of different types and sizes of neural networks. A CNN containing a multilayer perceptron (MLP) in each of its cells is called as cellular MLP (CMLP). CNN structure consisting of simultaneous recurrent neural networks (SRN) at each cell is called CSRN and is studied in [6]–[8]. In CNN, each cell is connected only to its adjacent cells. In CMLP and CSRN, the connection of different cells to each other is determined by the kind of application. In [9], each cell represents one bus and it is connected to other cells in the same way as the connection of the buses. In this study, CMLP has been used for speed deviation predictions of generators in a multimachine power system, where each cell represents a generator and connection of the cells is based on 'nearest-$n$ neighbors' topology.

Bipul Luitel and Ganesh K Venayagamoorthy are with Real-Time Power and Intelligent Systems Laboratory, Missouri University of Science & Technology, Rolla, MO 65401 USA, Contact: iambipul@ieee.org, gkumar@ieee.org

The financial support provided by the Walter Karplus graduate student research grant provided by the IEEE Computational Intelligence Society (CIS) for carrying out this research is gratefully acknowledged.

Implementation of different types of neural networks on hardware has been studied in literatures in the past [10]–[12]. Implementation of CNN on hardware has also been studied in literatures [13], [14]. In [15], graphics processing units (GPU) have been utilized as hardware platforms for parallelizing CNN operation. However, most of these studies are related to single unit based cell and implementation of structures with each cell representing a neural network remains to be explored. In this work, a CMLP has been implemented on a digital signal processor (DSP), trained and tested in real-time by interfacing with the real-time digital simulator (RTDS). Similar studies are also performed in non real-time using MATLAB. The results of the CMLP implemented on PC using MATLAB are compared with that of an MLP to show scalability. The remaining sections of the paper are arranged as follows: Smart Grid application of CMLP is described in Section:II. Section III describes the proposed design of WAM using CNN. CNN training approach is described in Section IV. Results and discussions are presented in Section V and conclusions are given in Section VI.

## II. Smart Grid Application of CMLP

Stability of electric power system depends on proper functioning of various power system components. Since power system is a massively distributed network, wide area monitoring is essential to assess the current state of these components. Based on this assessment, related control action is taken on the power system components in order to keep the system in stability. Therefore, wide area monitoring and control system (WAMCS) is an integral part in transitioning from the traditional power system to a Smart Grid. However, wide area monitoring becomes challenging as the size of the electric power grid, and consequently the number of components to be monitored, grows.

Applications of wide area monitoring systems (WAMS) in power system for state estimation, disturbance identification and wide area PSS have been reported in literature [16], [17]. Various design aspects of WAMCS are studied in [18]. Unlike traditional methods of data acquisition and control, typically supervisory control and data acquisition (SCADA) methods that relied on remote terminal units (RTU) for data, WAMS utilize phasor measurement units (PMU) for collecting data from the power system on a faster time-scale and hence can be used to monitor transient and dynamic response of the system [16]. A substation based dynamic state estimator has been used as WAMS in [19] that provides abilities to predict instabilities before they occur. Although these various techniques are being used and developed for wide area monitoring, there are still major challenges in their use for control. These challenges are related to extracting dynamics of the system without knowing the system model, mining and interpreting huge amount of data available from monitoring devices and assessment of the overall dynamics of the system based on wide area information [16]. It is even bigger challenge to make reliable control decisions under real-time constraints.

Computational intelligence (CI) techniques have shown promises in the field of wide area monitoring and control [20]. Since neural networks (NN) can be used to represent the dynamics of the system by training on the historical data of the system without having to know its actual model, they have shown promises in predictive control applications. NNs have been successfully implemented as state predictors and neurocontrollers [21] in the areas of wide area monitoring and control. Simultaneous recurrent neural network (SRN) and echo state network (ESN) based wide area monitor (WAM) has been demonstrated to be quite effective in performing predictive neuroidentification of distributed power systems for the purposes of accurate control [21], [22]. Radial basis function networks have been used for wide area monitoring with an adaptive critic designs based control in [23]. However, these feed-forward and feedback neural network architectures do not scale up to handle the growing complexity of the Smart Grid for wide area monitoring and control. As the number of variables increases, the number of neurons in the NN increases and so does the computational complexity. Therefore, it becomes challenging for the NN training algorithms to correctly learn the non-linear system dynamics. In this study, CMLP implemented on a hardware platform is presented as a way to provide scalability in the development of a WAM for Smart Grid in a real-time platform.

## III. Design of a CMLP based WAM

This study is carried out in three phases. Phase I refers to the implementation on a non real-time platform (Intel Core 2 CPU with 2 GB RAM running at 2.13 GHz programmed using MATLAB) where a WAM is developed using a CMLP and compared with that using a multiple-inputs multiple-outputs (MIMO) MLP. Phases II and III refer to the implementation on a real-time platform (160 MHz TMS320M6701 DSP programmed using C) where a WAM is developed using a CMLP on a DSP interfaced with the RTDS. In Phase I, the performance of CMLP is compared with an MLP. Fig. 1 shows the three phases of this study for CMLP implementation.

Test system used in the study is the two-area four-machine system shown in Fig. 2 [24]. It consists of four generators, two in each area. The WAM is developed to predict the speed deviations ($\Delta\hat{\omega}$) of each generator in the system at time instant $k+1$ based on speed deviations ($\Delta\omega$) and deviation of the reference voltage ($\Delta V_{ref}$) (shown in Fig. 3) of the generators at time instant $k$ as the inputs. The WAM is implemented using a CMLP. In WAM developed using a CMLP, each cell is used to predict the speed deviation of one generator in the power system. The cells are connected to each other based on 'nearest-$n$ neighbors' topology, which means previous sample outputs of $n$ nearest neighbors of each cell are connected to the inputs of that cell. The "nearness" is defined as the electrical distance between the generators and is measured based on the length of the transmission lines separating the two generators. In this study, two nearest neighbors are considered for designing the CNN. For example in Fig. 2, two nearest neighbors of generator G1 are generators G2 and G4. This is represented in the CNN by connecting the outputs of the cells C2 and C4 to the inputs of the cell C1. Similarly for G4, two nearest neighbors are G2 and G3 and hence outputs of the cells C2 and C3 are connected to the inputs of the cell C4. This topology allows for the scalability of the CNN by keeping the size of the MLP in each cell to a minimum. The MLP in each cell consists of an input layer with four neurons, a hidden layer with six neurons and an output layer with a single neuron. The four inputs to the MLP in each cell consist of $\Delta V_{ref}(k)$ and $\Delta\omega(k)$ associated with the generator represented by the cell and $\Delta\hat{\omega}(k)$ associated with the generators represented by the two nearest neighboring cells. The output of the CNN is $\Delta\hat{\omega}(k+1)$ of the generator associated with the cell, where $k$ is the sample index of the signal. This is explained in Fig. 2.

Fig. 4 shows the implementation of the WAM using a three layered feed-forward MLP for predicting the speed deviations of the three generators in the two-area four-machine system. It consists of eight neurons in the input layer, 15 neurons in the hidden layer and four neurons in the output layer, one output representing the step-ahead predictions of speed deviation for each generator. The eight inputs to the network are the two inputs ($\Delta\omega$,$\Delta V_{ref}$) going into the WAM from each generator.

## IV. CMLP Training

Two types of training approaches are carried out depending on real-time or non real-time studies. In non real-time studies carried out in Phase I, the neural networks are trained online using backpropagation algorithm [25]. In this approach, weights of the neural network are updated after every sample is passed through the network. After all the samples are covered, this process is repeated for as many passes through the network as required to achieve better convergence, as explained in [25]. Values of various parameters involved in training are listed in Table I. The training data is collected from the test system designed on RSCAD and simulated on a Real-time Digital Simulator [26]. During the forced training, all of the generators are simultaneously perturbed using a pseudo-random binary signal (PRBS) (shown in Fig.

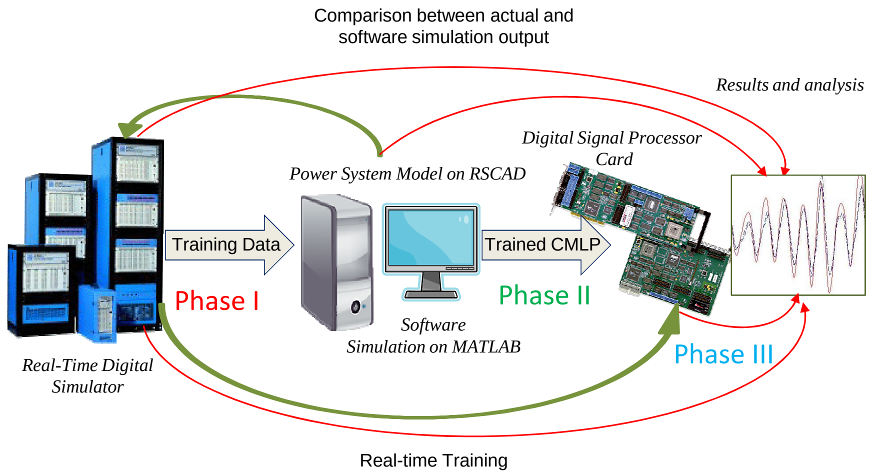

Comparison between actual and software simulation output
Results and analysis
Digital Signal Processor Card
Power System Model on RSCAD
Training Data
Trained CMLP
Phase I
Phase II
Phase III
Software Simulation on MATLAB
Real-Time Digital Simulator
Real-time Training

Fig. 1. Three phases of the study.

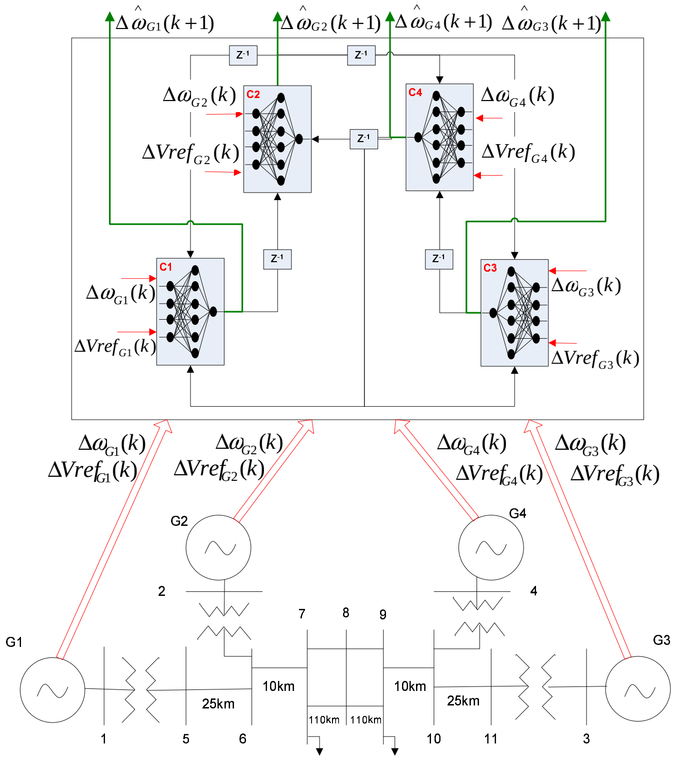


Fig. 2. CNN based WAM for two-area four-machine system.

5) applied to the excitation system of the generators (shown in Fig. 3). The deviation of the generator speed as a result of the PRBS perturbation is recorded along with the reference voltage applied to the generator excitation system ($\Delta V_{ref}$ in Fig. 3). The MIMO MLP is trained using these two signals as the inputs.

In case of CMLP, each cell is treated as an "object" and therefore, all of the cells are simultaneously trained with similar parameters. The parallel training approach of each cell object of the CMLP is explained further.

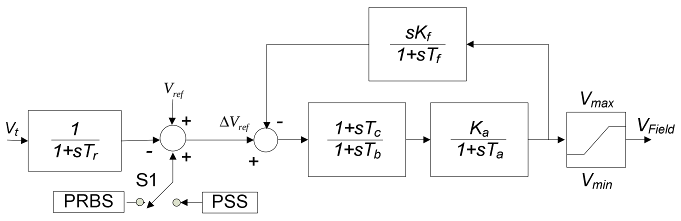


Fig. 3. Generator excitation system (showing application of PRBS and $\Delta V_{ref}$).

TABLE I
PARAMETERS USED FOR TRAINING MLP

| | |
|---|---|
| Trials | 50 |
| Number of passes | 100 |
| Learning Rate ($\mu$) | 0.005 |
| Momentum Gain ($\delta$) | 0.001 |

Each cell consists of four inputs viz. actual reference voltage applied to the generator $\Delta V_{ref}(k)$, actual speed deviation of the generator $\Delta\omega(k)$ and the predicted speed deviations of the nearest two generators, $\Delta\hat{\omega_{k1}}(k)$ and $\Delta\hat{\omega_{k2}}(k)$. For every sample of the input data $I(k)$, each cell produces a step-ahead predicted output $O(k+1)$. Therefore, at any input data of size $1, 2, \ldots, k, \ldots, K$ discrete samples, and $W_n$ and $V_n$ be the input and output weight matrices respectively of the MLP in $n^{th}$ cell, then the output of each cell is given by:

$$\begin{aligned} O_n(k) &= \Delta\hat{\omega}_n(k) \\ &= f\left(I_n(k-1), W_n(k), V_n(k)\right) \end{aligned} \quad (1)$$

Thus, $I_n(k) = [\Delta V_{refn}(k) \quad \Delta\omega_n(k) \quad \Delta\hat{\omega}_{n1}(k) \quad \Delta\hat{\omega}_{n2}(k)]$ uses the predicted output of the previous sample in case of the neighboring cells $n1$ and $n2$. This helps the parallelization of the cell objects, as long as the calculation of each sample output is synchronized among the different cells. In MATLAB, this is achieved by training each cell sequentially

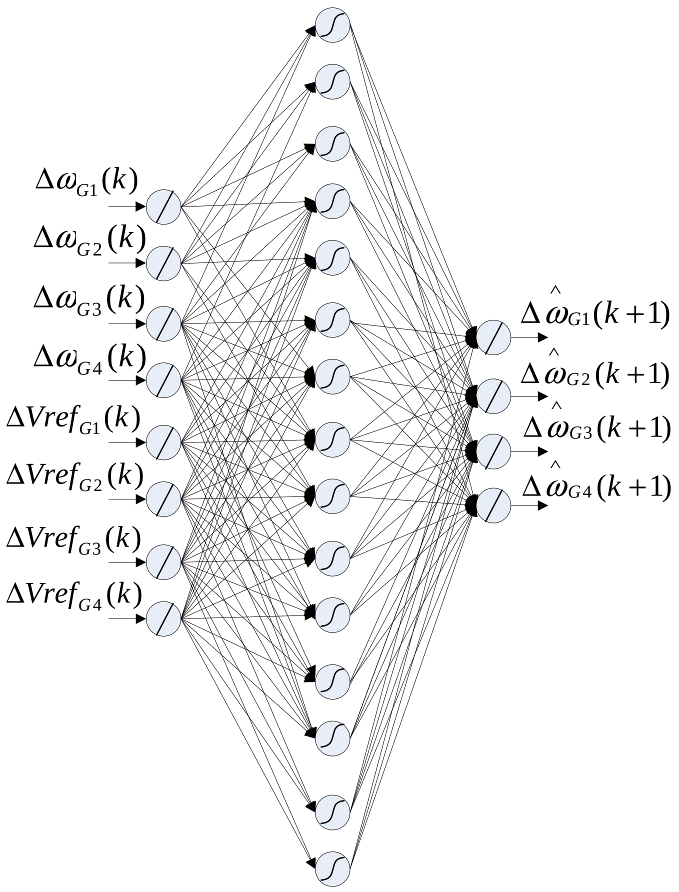


Fig. 4. Implementation of WAM using MLP for two-area four-machine system.

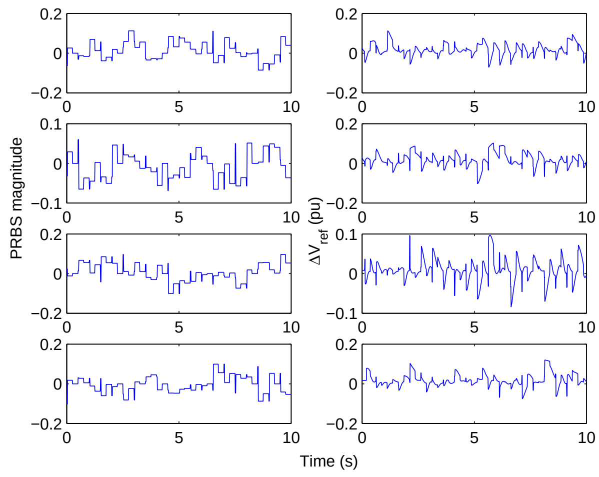
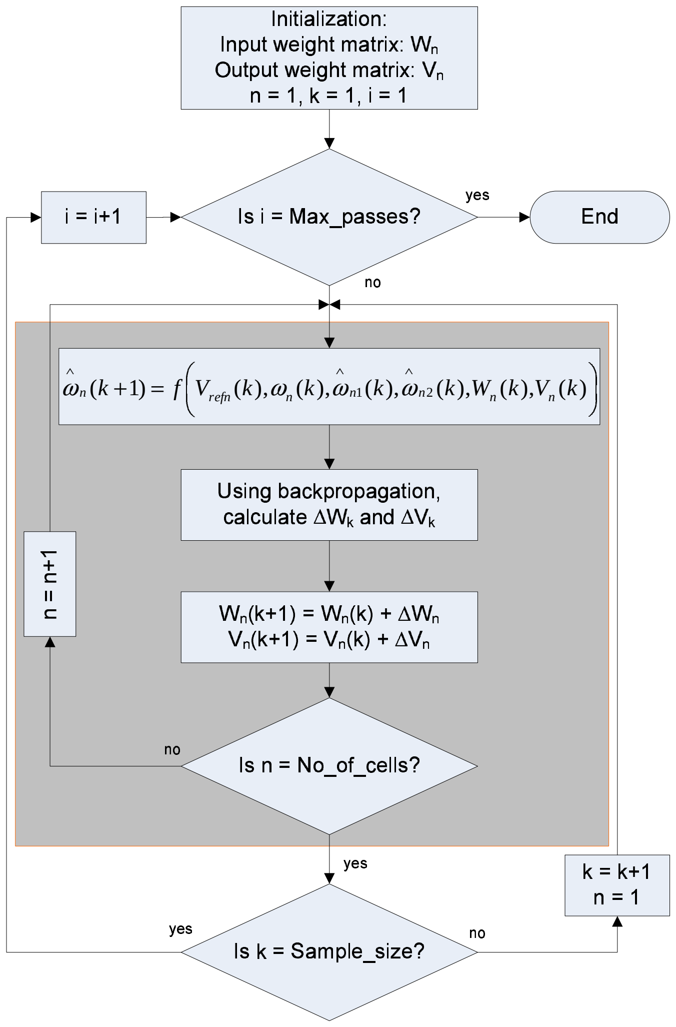


Fig. 5. PRBS signals applied to the four generators and the resulting $\Delta V_{ref}$.

for every sample. After the output is calculated for each cell, the weights of each cell is updated before calculating the output for the next sample. This process of online training of a CMLP using backpropagation is shown in the flowchart of Fig. 6. The part in the flowchart surrounded in dark box shows the process that can be implemented in parallel irrespective of the number of cells when a suitable platform is available.



Initialization:
Input weight matrix: $W_n$
Output weight matrix: $V_n$
n = 1, k = 1, i = 1
Is i = Max_passes?
yes
End
i = i+1
no
$\hat{\omega}_n(k+1) = f\left(V_{refn}(k), \omega_n(k), \hat{\omega}_{n1}(k), \hat{\omega}_{n2}(k), W_n(k), V_n(k)\right)$
Using backpropagation, calculate $\Delta W_k$ and $\Delta V_k$
$W_n(k+1) = W_n(k) + \Delta W_n$
$V_n(k+1) = V_n(k) + \Delta V_n$
n = n+1
no
Is n = No_of_cells?
yes
Is k = Sample_size?
yes
no
k = k+1
n = 1

Fig. 6. Flowchart for training of CMLP using backpropagation.

The real-time study is carried out in two phases. In Phase II, the CMLP trained using MATLAB is implemented on the DSP and is used for predicting the speed deviation of the generators in the test system by interfacing with the RTDS. In Phase III, the CMLP implemented on a DSP is trained online similar to Phase I using similar parameters and approaches. The only difference in this phase is that the inputs are available in real-time and the CMLP sees each input only once as they arrive (a single pass through the network). This makes learning in CMLP carried out in Phase II a much harder problem than non real-time online training carried out in Phase I.

## V. Results and Discussions

### A. Phase I

*1) Test System:* Two-area four-machine system consisting of four generators is considered as the test system for this study. This is implemented by a CMLP consisting of four cells and is compared against a MIMO MLP. Training and testing data are obtained for different operating points. The training data is obtained for $OP_1$. After the training is complete, it is tested on three operating points, $OP_1$, $OP_2$ and $OP_3$. These operating points are different to each other in the amount of power transfer between the two areas. Testing data is also obtained for operating point $OP_4$ by causing a 10-cycle 3-phase to ground fault on bus 8 of the test system during $OP_1$ steady state conditions. Similarly, operating point $OP_5$ is obtained by causing a line outage on one of the two transmission lines between the buses 7 and 8 in the test system. The different operating points and their properties are shown in Table II.

TABLE II
FIVE OPERATING POINTS CONSIDERED IN THE STUDY

| | $OP_1$, $OP_4$, $OP_5$ | $OP_2$ | $OP_3$ |
|---|---|---|---|
| Load 1 (MW) | 950 | 556 | 950 |
| Load 2 (MW) | 1650 | 1469 | 944 |
| $P_{area1 \Leftrightarrow area2}$ (MW) | 253.2 | 302.9 | 80.45 |
| $Q_{area1 \Leftrightarrow area2}$ (MVar) | 22.68 | 57.2 | -38.02 |
| $P_{G1}$ (MW) | 705.6 | 573.8 | 579.5 |
| $Q_{G1}$ (MVar) | 163.5 | 117.2 | 53.89 |
| $P_{G2}$ (MW) | 705.5 | 537.7 | 579.1 |
| $Q_{G2}$ (MVar) | 296 | 234.6 | 81.12 |
| $P_{G3}$ (MW) | 441.5 | 309.5 | 314.4 |
| $Q_{G3}$ (MVar) | 68.8 | 49.79 | -31.56 |
| $P_{G4}$ (MW) | 705.6 | 537.7 | 578.6 |
| $Q_{G4}$ (MVar) | 169.8 | 140.1 | -59.53 |

Fig. 7 shows the convergence diagram for the four outputs of the MIMO MLP. Similar convergence diagram for the CMLP is shown in Fig. 8. These diagrams show how the mean squared error (MSE) between the actual and the predicted outputs decreases over multiple passes of the training data through the network. The testing outputs obtained from the CMLP for the five operating points are shown in Figs. 9 to 13. The comparison of absolute errors obtained using MLP and CMLP for $OP_1$ to $OP_5$ are shown in Figs. 14 to 18, respectively. The average and standard deviation of the mean absolute error (MAE) obtained by the two networks during testing on five operating points over 50 trials are shown in Table III.

## B. Phase II

In this phase, the CMLP trained in Phase I using MATLAB is implemented on a DSP by initializing the trained weights. This network is then used to predict the outputs (speed deviation of the generators of the test system) generated by the power system simulated on the RTDS in real-time. The outputs of four cells of CMLP represent the speed deviations of four generators and are shown in Fig. 19. During the time when the CMLP on the DSP is predicting the outputs, operating points are varied by causing a 10-cycle 3-phase line to ground fault ($OP_4$) and transmission line outage on the system ($OP_5$). The predicted outputs of the CMLP compared

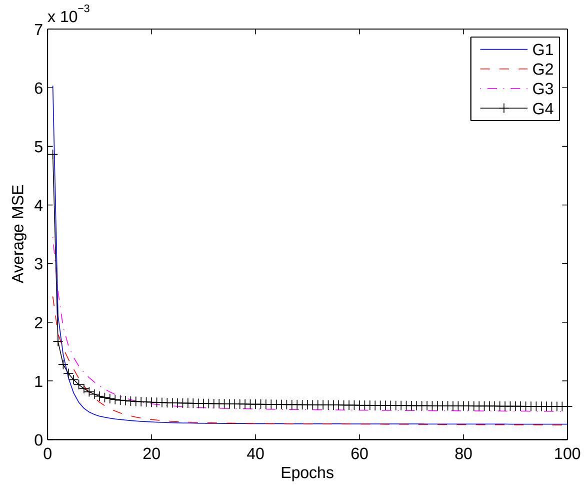


Fig. 7. Convergence of individual outputs of the MIMO MLP during training.

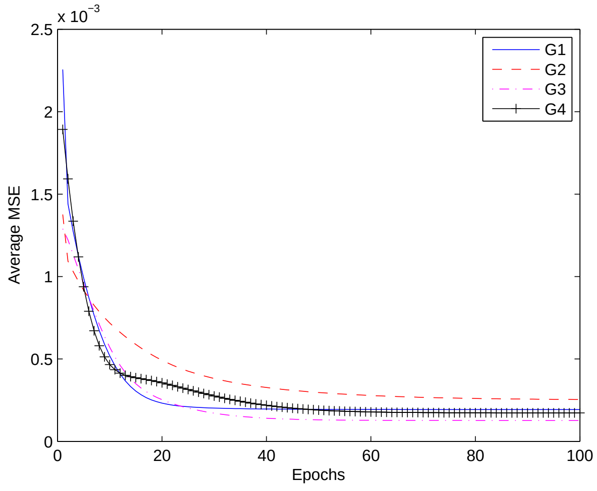


Fig. 8. Convergence of individual cells of the CMLP during training.

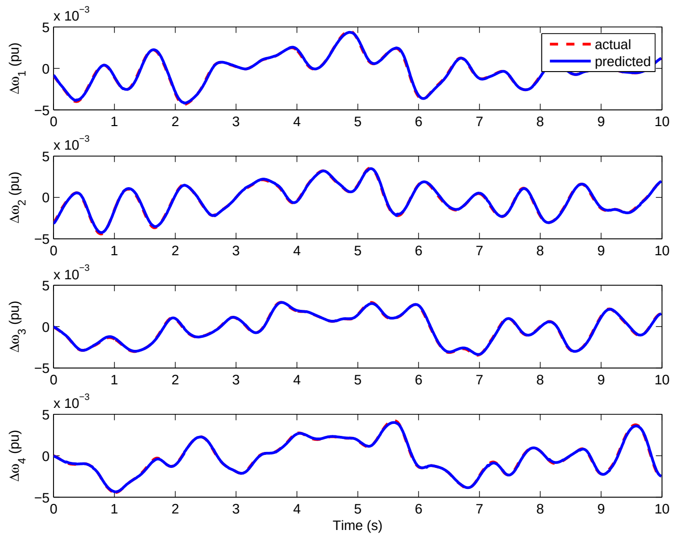


Fig. 9. Testing output of CMLP for operating point I.

TABLE III
COMPARISON OF MLP AND CMLP FOR DIFFERENT OPERATING POINTS IN PHASE I

| | | G1 | | G2 | | G3 | | G4 | |
|---|---|---|---|---|---|---|---|---|---|
| | | MLP | CMLP | MLP | CMLP | MLP | CMLP | MLP | CMLP |
| $OP_1$ | Avg. | **0.010533** | 0.011064 | **0.012169** | 0.013711 | 0.010201 | **0.009517** | 0.013105 | **0.010210** |
| | Std. | 0.000225 | **0.000202** | **0.000491** | 0.000587 | 0.000499 | **0.000088** | 0.000431 | **0.000242** |
| $OP_2$ | Avg. | 0.012927 | **0.008843** | 0.014253 | **0.011057** | **0.015672** | 0.016951 | 0.019904 | **0.013778** |
| | Std. | 0.000560 | **0.000330** | **0.000638** | 0.001145 | 0.001045 | **0.000227** | 0.000825 | **0.000245** |
| $OP_3$ | Avg. | 0.013716 | **0.012222** | 0.014582 | **0.013466** | 0.013608 | **0.011166** | 0.013869 | **0.011375** |
| | Std. | **0.000050** | 0.000252 | **0.000145** | 0.000626 | 0.000521 | **0.000135** | 0.000320 | **0.000130** |
| $OP_4$ | Avg. | 0.003171 | **0.001850** | 0.006353 | **0.005029** | 0.007390 | **0.001142** | 0.008319 | **0.002843** |
| | Std. | **0.000069** | 0.000152 | **0.000155** | 0.000638 | 0.000122 | **0.000023** | 0.000737 | **0.000213** |
| $OP_5$ | Avg. | **0.002383** | 0.004535 | 0.003849 | **0.002275** | 0.002516 | **0.001739** | 0.008739 | **0.006740** |
| | Std. | **0.000581** | 0.000649 | **0.000358** | 0.000814 | 0.000590 | **0.000084** | 0.000467 | **0.000322** |
| Winner | | 1 | 4 | 1 | 4 | 1 | 4 | 0 | 5 |

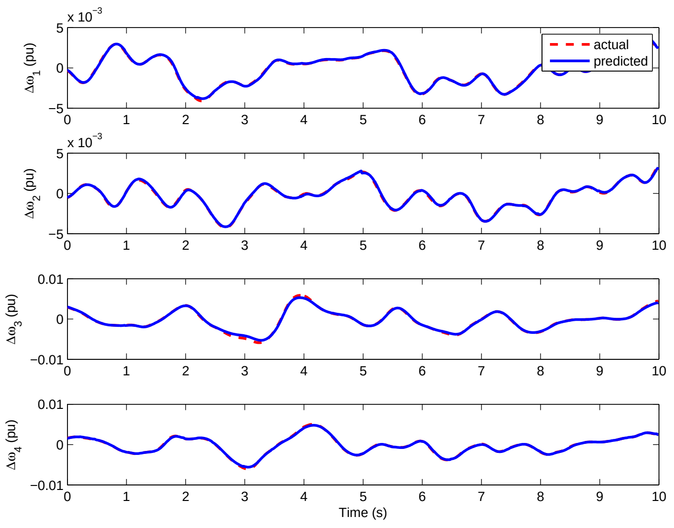


Fig. 10. Testing output of CMLP for operating point II.

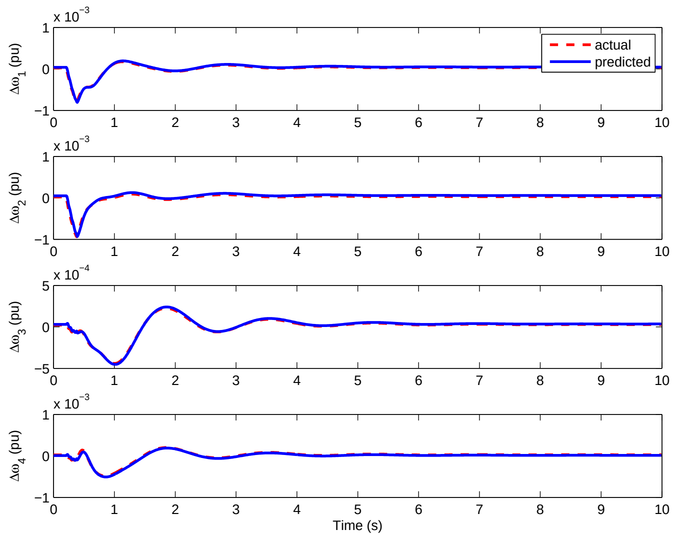


Fig. 12. Testing output of CMLP for operating point IV.

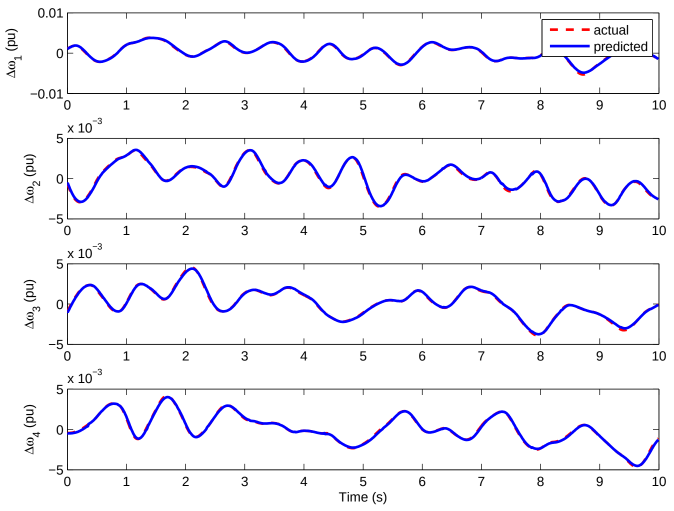


Fig. 11. Testing output of CMLP for operating point III.

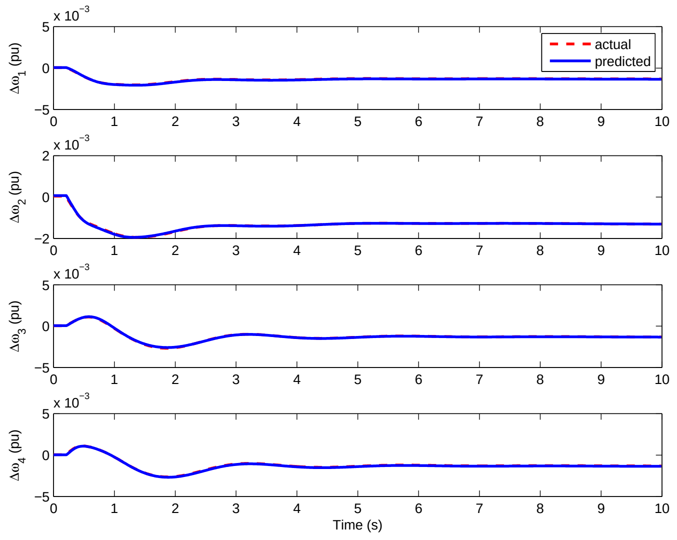


Fig. 13. Testing output of CMLP for operating point V.

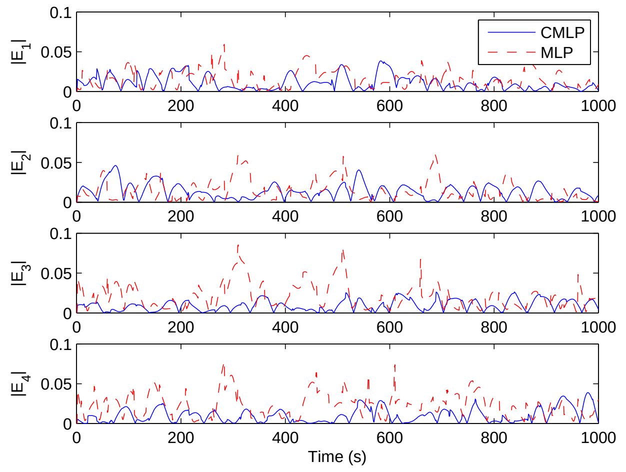


Fig. 14. Comparison of absolute errors obtained by MLP vs. CMLP for operating point I.

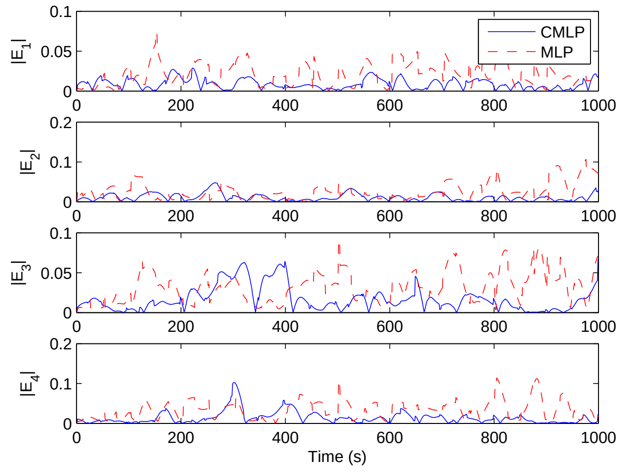


Fig. 15. Comparison of absolute errors obtained by MLP vs. CMLP for operating point II.

with the actual outputs for these tests are shown in Figs. 20 and Figs. 21.

## C. Phase III

Phase III is a combination of the other two phases. In this phase, a CMLP is developed on a DSP using C programming language and is interfaced with the RTDS where the test system is simulated. The set up is similar to Phase I except that the inputs to the DSP in this study are fed in real-time from the output of the RTDS. The CMLP developed in DSP is trained in real-time to predict the speed deviations of the generators of the test system. The weights of the CMLP are continuously updated after each sample of the input is received and the whole training process and prediction is completed before the next sample arrives. The results shown in Fig. 22 show the outputs of the CMLP during the real-time

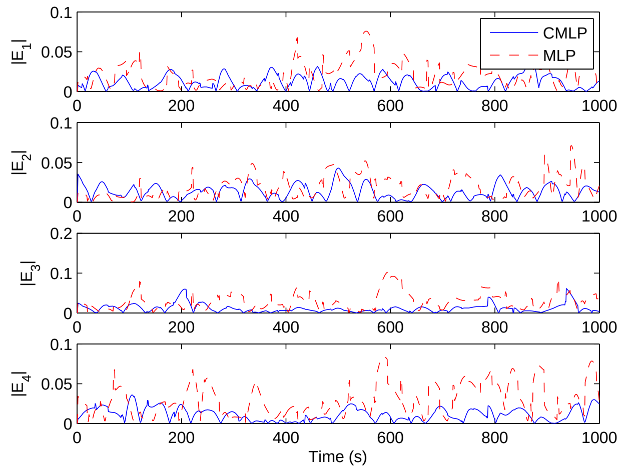


Fig. 16. Comparison of absolute errors obtained by MLP vs. CMLP for operating point III.

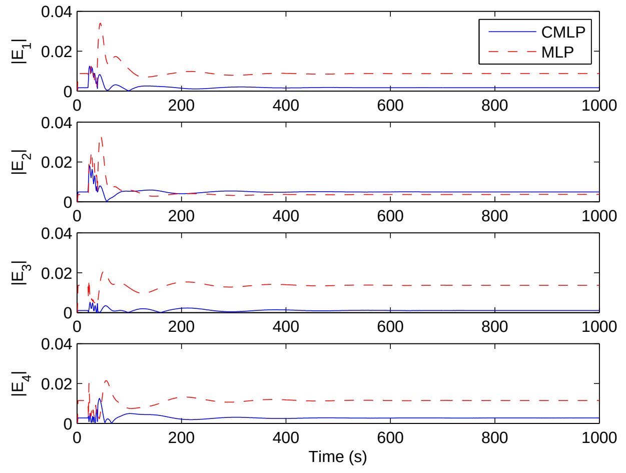


Fig. 17. Comparison of absolute errors obtained by MLP vs. CMLP for operating point IV.

online training.

## D. Result Analysis

Learning in CNN is a challenging task because of their connectivity. Since the predicted output from one cell is used as input(s) to other neighboring cell(s), errors due to poor training and hence false predictions of the NN in one cell can ripple through all of the cells and deteriorate the performance of the CNN. On the other hand it is also arguable that the NNs get trained even better due to the connectivity because the errors propagate through the network and each cell is trained actively (through its own training) and passively (through the training of its neighbors) as training algorithm on each cell tries to minimize the error at its output. This way, knowledge of the actual dynamics of the system is preserved not only on the individual neural networks at each

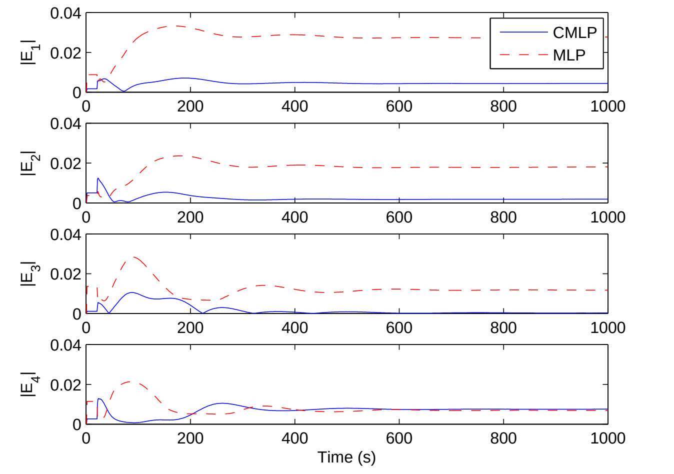


Fig. 18. Comparison of absolute errors obtained by MLP vs. CMLP for operating point V.

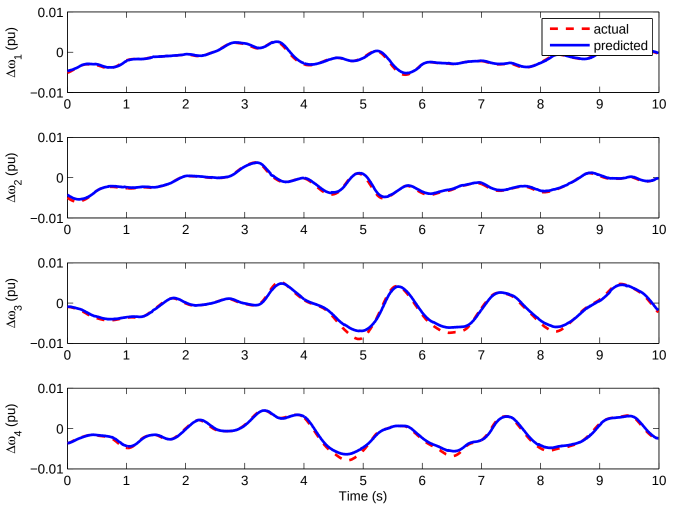


Fig. 19. Outputs of CMLP implemented on a DSP (Phase II) for $OP_1$.

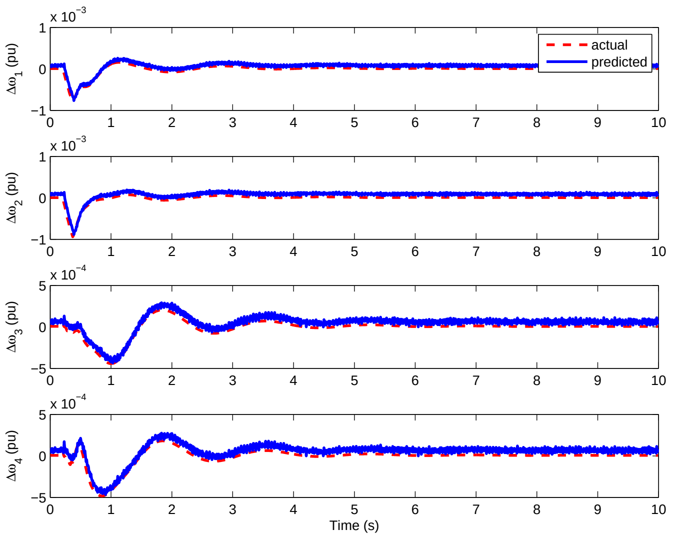


Fig. 20. Outputs of CMLP implemented on a DSP (Phase II) for $OP_4$.

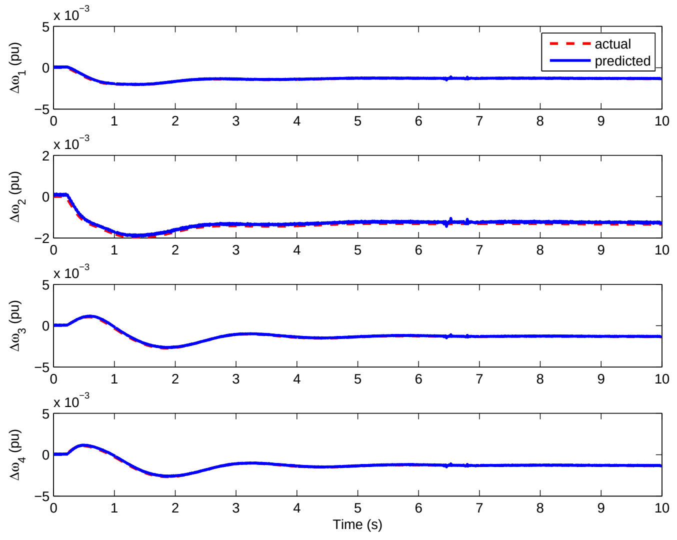


Fig. 21. Outputs of CMLP implemented on a DSP (Phase II) for $OP_5$.

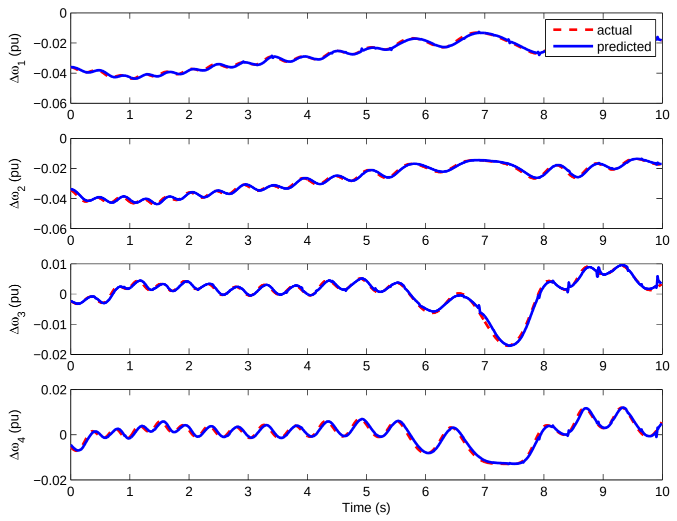


Fig. 22. Outputs of CMLP implemented on a DSP (Phase III) during real-time training.

cell, but also on the connectivity between the different cells of the CNN during training. Moreover, the advantage of CNN comes from its ability to scale up to a much larger system without significant impact on the performance. When the size of the network grows, the number of cells increases but the size of an MLP on each cell remains the same (as long as the nearest-$n$ topology remains the same). For a CMLP with $m$ cells with each cell having an MLP with $N$ weights, the total number of weights in the network is $mN$. This is in contrast to a MIMO MLP where the number of neurons in the hidden layer needs to be increased significantly in order to obtain a satisfactory performance when the number of inputs and outputs increases. This causes the number of weights in an MLP to increase drastically as the size of the network grows, thus increasing the computational complexity leading to poor training and testing results. However, training of CMLP takes a long time if implemented sequentially on desktop PCs. Therefore, the second and third phase of the study were carried out on a dedicated hardware platform. Although the implementation was still sequential in C, the fast computations carried out in DSP allowed for the real-time implementation of the training and testing of CMLP up to four cells considered in the studies. However, as the number of cells increases in larger systems, the computation time also increases and can only be justified if implemented on a parallel hardware and/or software platforms such as general purpose GPU clusters, FPGAs or shared memory architectures.

The tabulated data also show that the performance of CMLP is better than that of MLP. The lower values of average MAE show better performance of CMLP over MLP and the lower values of standard deviation show its consistency in maintaining that performance. The 'winner' row in the table quantitatively expresses the results to show which NN architecture has better performance. The lower values of average MAE is given priority in deciding the winning architecture for each output. If the difference of average MAE is less than 5% of each other, then it is considered as a tie and hence the standard deviation is considered to be the tie-breaker. For the given test system, CMLP has better performance in four operating points for outputs G1, G2 and G3 and in all five operating points for output G4.

## VI. Conclusion

In this study, CMLP is used as state predictor for implementing wide area monitor for a multi-machine power system. The different cells of the CMLP are connected to each other using 'nearest-$n$ neighbors' topology so that the size of MLP in each cell is reduced. This ensures that the complexity of the network increases in linear proportion to the size of the network being implemented. Therefore, a CMLP becomes highly scalable. The study is carried out on non real-time as well as real-time platforms. Results obtained from these different studies are presented and are shown to be comparable to or better than implementation of WAM using a single MIMO MLP structure in terms of performance as well as number of weights. Real-time training and implementation of CMLP is a challenging problem and this study has shown some light in this area. However, much study remains to be done in the area including implementation of the studied approaches in a larger system as well as on a parallel hardware/software platforms. Advanced learning techniques for real-time CMLP training and performance enhancement in larger system using parallel hardware platforms will be the future work in this direction.

## References

[1] L. Chua and L. Yang, "Cellular neural networks," in *IEEE International Syposium on Circuits and Systems*, vol. 2, 1988, pp. 985–988.

[2] P. Tzionas, A. Thanailakis, and P. Tsalides, "A hybrid cellular automaton/neural network classifier for multi-valued patterns and its vlsi implementation," *Integration, the VLSI Journal*, vol. 20, no. 2, pp. 211 – 237, 1996.

[3] T.-J. Su, Y.-Y. Du, Y.-J. Cheng, and Y.-H. Su, "A fingerprint recognition system using cellular neural networks," may 2005, pp. 170 – 173.

[4] L. Chua and L. Yang, "Cellular neural networks: Theory," *IEEE Transactions on circuits and systems*, vol. 35, no. 10, pp. 1257–1272, 1988.

[5] I. Aizenberg, N. Aizenberg, J. Hiltner, C. Moraga, and E. M. zu Bexten, "Cellular neural networks and computational intelligence in medical image processing," *Image and Vision Computing*, vol. 19, pp. 177 – 183, 2001.

[6] D. Wunsch, "The cellular simultaneous recurrent network adaptive critic design for the generalized maze problem has a simple closed-form solution," vol. 3, 2000, pp. 79 –82 vol.3.

[7] R. Ilin, R. Kozma, and P. Werbos, "Cellular SRN trained by extended Kalman filter shows promise for ADP," in *Proc. int. joint conf. neural networks, July*, 2006, pp. 16–21.

[8] Y. Ren, K. Anderson, K. Iftekharuddin, P. Kim, and E. White, "Pose invariant face recognition using cellular simultaneous recurrent networks," june 2009, pp. 2634 –2641.

[9] L. Grant and G. Venayagamoorthy, "Cellular multilayer perceptron for prediction of voltages in a power system," in *15th International conference on intelligent system application to power systems (ISAP'09)*, 2009.

[10] W. Qinruo, Y. Bo, X. Yun, and L. Bingru, "The hardware structure design of perceptron with fpga implementation," vol. 1, oct. 2003, pp. 762 – 767 vol.1.

[11] M. Krid, D. Masmoudi, and M. Chtourou, "Hardware implementation of bfnn and rbfnn in fpga technology: Quantization issues," dec. 2005, pp. 1 –4.

[12] M. Nuno-Maganda, M. Arias-Estrada, C. Torres-Huitzil, and B. Girau, "Hardware implementation of spiking neural network classifiers based on backpropagation-based learning algorithms," june 2009, pp. 2294 – 2301.

[13] T. Szirnyi, "Texture recognition using a superfast cellular neural network vlsi chip in a real experimental environment," *Pattern Recognition Letters*, vol. 18, no. 11-13, pp. 1329 – 1334, 1997.

[14] K. Kayaer and V. Tavsanoglu, "A new approach to emulate cnn on fpgas for real time video processing," july 2008, pp. 23 –28.

[15] T. Ho, P. Lam, and C. Leung, "Parallelization of cellular neural networks on GPU," *Pattern Recognition*, vol. 41, no. 8, pp. 2684–2692, 2008.

[16] Y. Xue, "Some viewpoints and experiences on wide area measurement systems and wide area control systems," in *Power and Energy Society General Meeting - Conversion and Delivery of Electrical Energy in the 21st Century, 2008 IEEE*, 2008, pp. 1–6.

[17] I. Kamwa, J. Béland, G. Trudel, R. Grondin, C. Lafond, and D. McNabb, "Wide-area monitoring and control at Hydro-Québec: Past, present and future," in *Power Engineering Society General Meeting, 2006. IEEE*. IEEE, 2006, p. 12.

[18] M. Zima, M. Larsson, P. Korba, C. Rehtanz, and G. Andersson, "Design aspects for wide-area monitoring and control systems," *Proceedings of the IEEE*, vol. 93, no. 5, pp. 980 –996, May 2005.

[19] S. Meliopoulos, G. Cokkinides, R. Huang, E. Farantatos, S. Choi, and Y. Lee, "Wide area dynamic monitoring and stability controls," in *Bulk Power System Dynamics and Control (iREP) - VIII (iREP), 2010 iREP Symposium*, 2010, pp. 1–8.

[20] G. K. Venayagamoorthy, "Potentials and promises of computational intelligence for smart grids," in *Proc. IEEE Power & Energy Society General Meeting (PES '09)*, 26–30 July 2009, pp. 1–6.

[21] S. Ray and G. Venayagamoorthy, "Real-time implementation of a measurement-based adaptive wide-area control system considering communication delays," *IET Generation Transmission and Distribution*, vol. 2, no. 1, pp. 62–70, 2008.

[22] G. Venayagamoorthy, "Online design of an Echo State Network based wide area monitor for a multimachine power system," *Neural Networks*, vol. 20, no. 3, pp. 404–413, 2007.

[23] S. Mohagheghi, G. Venayagamoorthy, and R. Harley, "Optimal wide area controller and state predictor for a power system," *IEEE Transactions on Power Systems*, vol. 22, no. 2, pp. 693–705, 2007.

[24] M. Klein, G. Rogers, and P. Kundur, "A fundamental study of inter-area oscillations in power systems," *IEEE Transactions on Power Systems*, vol. 6, no. 3, pp. 914–921, 1991.

[25] P. Werbos, "Backpropagation through time: what it does and how to do it," *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550 –1560, oct. 1990.

[26] RTDS, "Real time digital simulator tutorial manual (rscad version)," RTDS Technologies, March 2008.