NEURAL NETWORK APPLICATIONS TO REAL-TIME AND OFF-LINE FAULT ANALYSIS

Mladen Kezunovic, Igor Rikalo
Dept. of Electrical Engineering
Texas A&M University
College Station, TX 77843-3128
U. S. A.
ph: 1-409-845-7509
fax: 1-409-845-7161
e-mail: kezunov@ee.tamu.edu
rikalo@ee.tamu.edu

Dejan J. Sobajic
Electric Power Research Institute
3412 Hillview Avenue
Palo Alto, CA 94304-1395
U. S. A.
ph: 1-415-855-8537
fax: 1-415-855-2954
e-mail: dsobajic@msm.epri.com

Abstract: This paper is concerned with application of Neural Networks (NNs) to fault analysis for both the real-time applications such as protective relaying of transmission lines and the off-line applications such as post-mortem study of fault events recorded with Digital Fault Recorders (DFRs). A supervised learning NN of the same type is utilized for both applications. It has been demonstrated that the NN approach reaches performance of the existing techniques in both application areas and yet shows some additional benefits.

Keywords: *Fault analysis, neural networks, supervised clustering, EMTP simulations*

INTRODUCTION

Application of intelligent systems to power system problems has been a research area with growing interest [1,2]. The fault analysis problem was one of the widely studied topics [3]. Most of the advanced solutions where related to application of expert systems to off-line analysis of fault events [4,5]. The most recent developments in this area suggest further benefits from using combination of different intelligent system approaches to solving this problem [6,7].

Most recently, NN techniques were used to solve fault analysis problems. Various approaches were proposed for analysis of fault data collected by SCADA systems [8] and DFRs [6]. Also, some applications to directional relaying [9] and high impedance fault detection [10 - 13] were reported. Those studies have shown some potential benefits of NN utilization in the mentioned applications.

This paper concentrates on somewhat unique application where the same NN architecture is utilized for both real-time and off-line implementation. The real-time application is related to protective relaying of HV transmission lines, while the off-line application relates to analysis of faults recorded by substation DFRs.

The specific reason for investigating both the real-time and the off-line applications in the same implementation framework is motivated by some advanced fault monitoring concepts and system implementation approaches. The real-time monitoring of power systems may be quite useful in verifying correctness of the protective relaying system operation in substations. This in turn may be further utilized for a fast, hierarchical off-line system monitoring executed both at the substation and the control center level. If the implementation approach is the same, combining of the two processing modes in one system solution may be easier. Further more, if the neural network algorithms used are the same, fast dedicated hardware implementation of both processing tasks may be realized in the same hardware/software. This again facilitates combining of these tasks in one processing system.

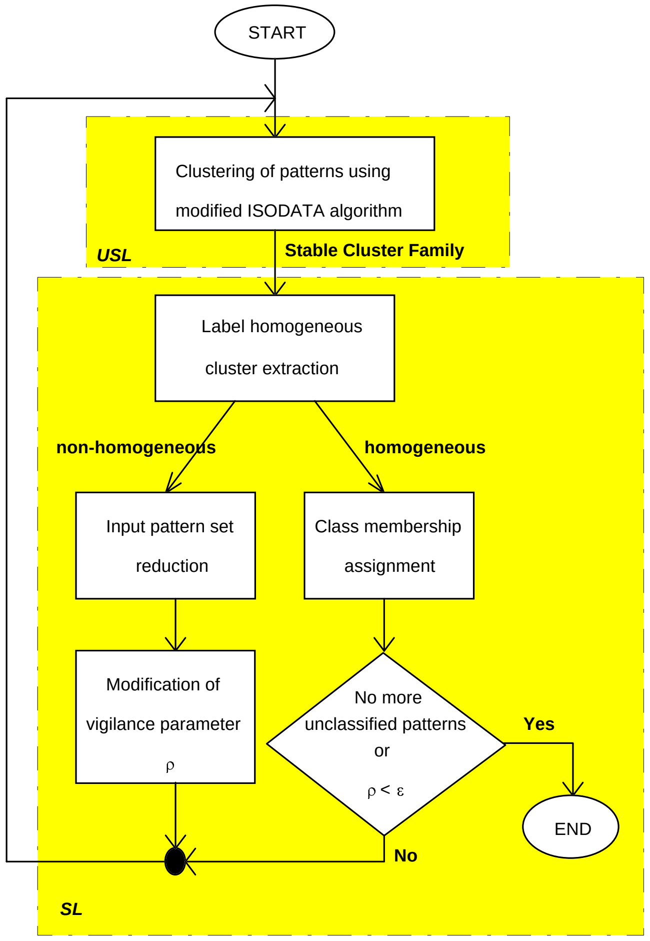
The protective relaying application has quite stringent time response requirements. It has been demonstrated that the selected NN approach meets those requirements. The analysis of DFR files has been implemented earlier using an expert system approach [5] so it was interesting to investigate if the NN technique will produce some additional benefits [6]. Results from both of the studies are summarized in this paper.

The first part of the paper gives a brief summary of the NN approach used. The real-time application is described next, followed by the off-line application. Conclusions are given at the end.

NEURAL NET ALGORITHM

The NN algorithm used in this study has been introduced by one of the authors earlier and has been successfully used for several power system applications [9, 14-17]. The algorithm has shown good performance and some interesting properties that prompted further feasibility study related to the fault analysis problem. The results of this study are reported in this paper.

Neural network algorithm described in this paper implements a supervised "follow the leader" approach. Current learning methods used in the NNs, can be classified into two categories (supervised learning and unsupervised learning) although aspects of each may coexist in a given architecture. Figure 1 shows the block diagram of the learning process that contains both unsupervised (USL) and supervised (SL) part. Unsupervised learning self-organizes presented data and discovers its collective properties. Initially, the whole data set, containing all patterns, is processed using unsupervised clustering algorithm similar to ISODATA self-organization procedure (detailed description of the clustering algorithm is given in Appendix at the end of the paper). The output is stable family of clusters, defined as a hyperspheres in N dimensional space, where N denotes number of features in each pattern. In the supervised part, non-homogeneous clusters are separated from homogeneous (clusters containing only label uniform patterns). Class membership is assigned to homogeneous clusters.


START
Clustering of patterns using modified ISODATA algorithm
USL
Stable Cluster Family
Label homogeneous cluster extraction
non-homogeneous
homogeneous
Input pattern set reduction
Class membership assignment
Modification of vigilance parameter ρ
No more unclassified patterns or ρ < ε
Yes
END
No
SL
Figure 1. Neural network learning process

Training data set is, then, reduced to contain only patterns from non-homogeneous clusters. Vigilance parameter ρ is decreased, and the whole procedure is reiterated.

After completion of the training procedure, all generated clusters contain uniform data patterns, and are characterized by their centroids, corresponding radii (i.e., vigilance parameter ρ), and inherited class membership. Figure 2. shows schematic illustration of the outcome of the training process in the feature space. It can be observed that cluster topology is not uniform, and that two or more clusters may have the same class membership.

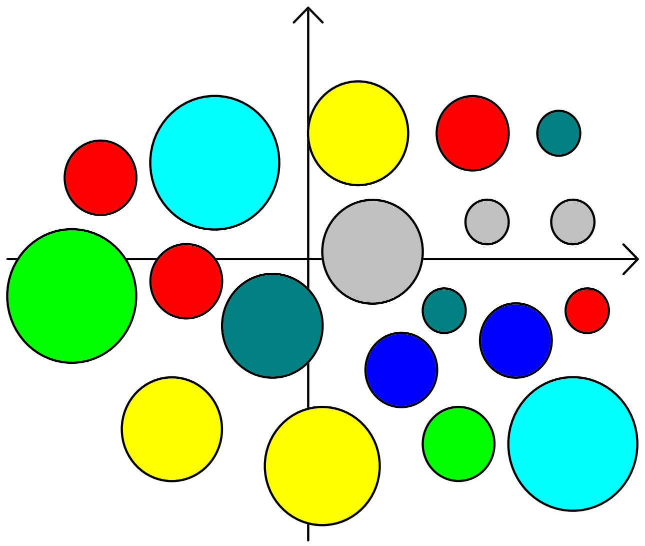


Figure 2. Schematic illustration of the outcome of the training process

APPLICATION CASES AND TESTING SET-UP

Two different applications of the proposed algorithm have been studied. One of them is implementing neural network as a transmission line protective relay. The other is implementation of the neural network for the off-line fault detection and classification, which can be used as an aid to the operators in the substations for fast characterization of the Digital Fault Recorder (DFR) event files. Table I shows main features of both applications. Electromagnetic Transient Program (EMTP) simulation [18] is used to obtain fault patterns for network training and testing for both applications.

Table I. Simulation characteristics of real-time and off-line applications

	Real-time application	**Off-line application**
Source of data	EMTP simulation	EMTP simulation
Pattern length	1 cycle	variable
Sampling frequency	2 kHz	2 kHz
Sample types	3-phase currents	3-phase currents and/or 3-phase voltages
Fault types	all 11 phase faults	all 11 phase faults
Number of patterns used for training	200	325
Input during operation	sliding window 1 cycle long	snapshot corresponding to the pattern length used in training
Number of patterns used for testing	80	295

One line diagram of the modeled 161 kV power system is given in Figure 3. This model is part of an actual system with short mutually coupled transmission lines. Faults are simulated on the line between buses 2 and 3.

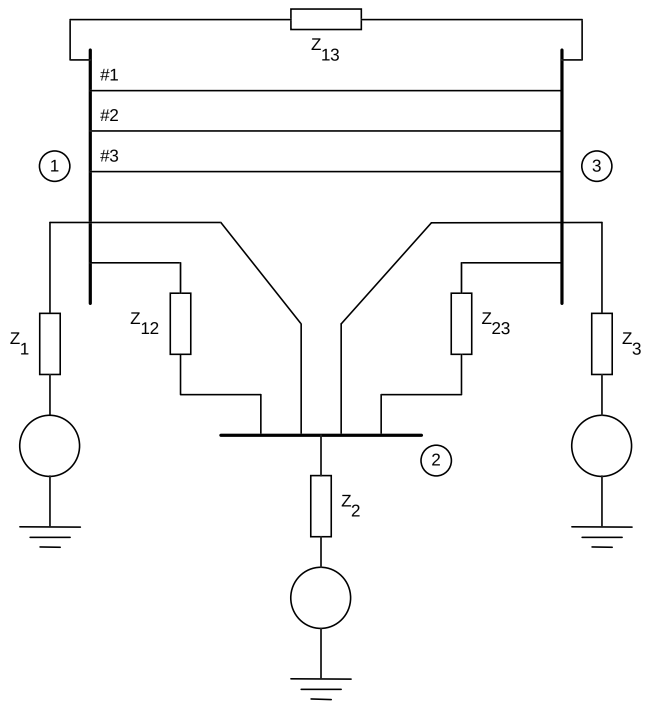


Figure 3. One line diagram of the 161 kV power system used for EMTP modeling

An extensive simulation was conducted, resulting in more than 600 fault cases. All 11 possible transmission line fault types are simulated (a-g, b-g, c-g, a-b, a-c, b-c, a-b-g, a-c-g, b-c-g, a-b-c and a-b-c-g faults) and the following three parameters are varied within each simulation:

- fault location (0.0, 0.14, 0.80, and 1.0, where 1.0 corresponds to the whole length of the transmission line)
- fault resistance (0 Ω, 3 Ω, 6 Ω, and 50 Ω)
- incidence angle of the fault occurrence (0°, 45°, and 90°)

Roughly, half of these cases were used for training of the neural network, and the other half was used for NN testing.

Real-Time Application

The main functions that neural network has to perform in this application are:

- Fault detection
- Fault type classification

Execution of the above tasks is extremely time restricted. For transmission systems it is expected that protective relays perform fault detection and fault type classification in less than 1 cycle (16.67 ms). Conventional high speed relay can perform both functions as fast as 1/4 cycle (less than 5 ms).

Neural network training is conducted using 200 fault patterns. Trained network is then used for detection and classification of the fault events in the power system.

In order to test the performance of the NN based relay, 80 fault events were generated using the EMTP simulation. Total length of simulated waveforms for each event was 0.5 s (30 cycles) and simulation time step was 0.5 ms. Input to the NN based relay is in the form of the sliding data window containing only samples of phase currents. Window length is fixed and 33 samples long (approx. window length is 1 cycle). Example of the sliding data window technique is shown in Figure 4, where arrow points to the sliding direction. Sliding motion is obtained by putting every new sample at the end of the window and removing the first sample from the beginning of the window.

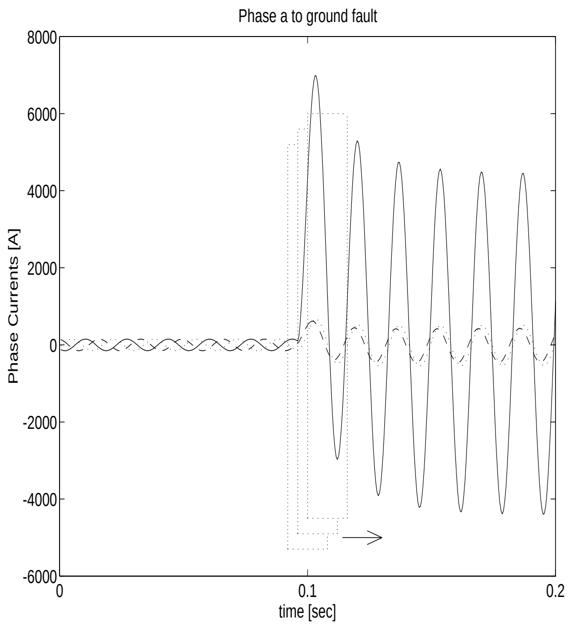


Figure 4. Sliding data window input into the neural network

Block diagram of the fault detection and fault classification functions that are part of the neural network is shown in Figure 5.

Fault detection logic is implemented as follows:

- Input pattern is compared to the cluster representing power system steady state.
- If input pattern "belongs" to the "normal-state" cluster (i.e. Euclidean distance from cluster centroid is less than radius of that cluster) then slide the input window for one sample and compare again.
- If input pattern "doesn't belong" to the "normal-state" cluster (i.e. Euclidean distance from cluster centroid is greater than radius of that cluster) then fault is detected and execution of the fault classification logic is started.

Fault classification logic is implemented as follows:

- Input pattern is now compared to every cluster obtained during training of the neural network.
- If input pattern "belongs" to a particular cluster then NN classifies fault event according to the class membership for that cluster.
- If input pattern "doesn't belong" to any of the clusters then relay slides input window for one sample and does the comparison again. Decision Time (DT) is parameter used to force NN to make final decision. That means after DT [ms] from disturbance detection, if pattern still doesn't belong to any of the clusters, NN will classify fault event according to the class membership of the nearest cluster. Nearest cluster is cluster whose centroid is closest to the input pattern according to the Euclidean distance.

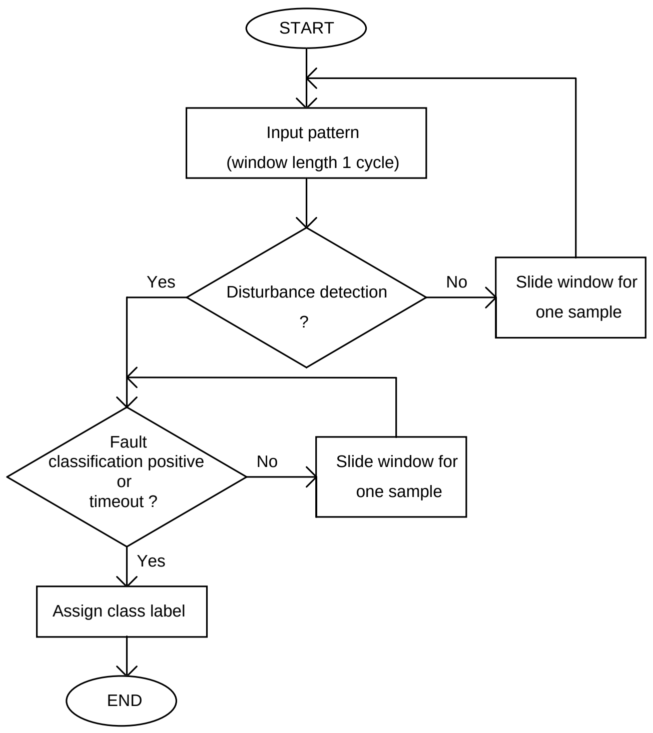

START
Input pattern
(window length 1 cycle)
Disturbance detection
?
Yes
No
Slide window for one sample
Fault classification positive or timeout ?
No
Slide window for one sample
Yes
Assign class label
END

Figure 5. Block diagram of the disturbance detection and fault classification logic

The neural network algorithm was implemented and tested on IBM PC compatible computer with Intel 486 DX-2 microprocessor (66 MHz). Computational time needed for comparison is:

- for fault detection logic approx. 0.2 ms
- for fault classifiaction logic approx. 15 ms

Taking into consideration that data sampling frequency was 2 kHz (0.5 ms), it can be concluded that fault detection logic was operating in real time, while fault classification logic required more time. Further work is being done to evaluate possible enhancement of the neural network computational performance by using special digital signal processors (DSPs) for real-time embedded applications (e.g., TI TMS320C4x family).

Table II shows relay operation for different length of the decision time (DT). Neural network was trained using fault patterns with incidence angle α = 0° and α = 90°. It can be noticed that relay performance deteriorates in the case of faults with incidence angle α = 45° due to the insufficient network training. Sampling frequency in this simulation was 2 kHz, which means that DT=22 samples corresponds to 11 ms.

Times (i.e., number of samples needed to make a decision) for fault detection logic and fault classification logic for different cases are presented in Table III.

The following is the assessment of the performance of the NN based relaying logic:

- Overall performance for all three fault incidence angles and for DT = 22 samples (11 ms) expressed in percentage is 71.6 %. The longest time for fault detection and classification was 32 samples (corresponds to 16 ms) which is under 1 cycle.

Table II. Neural Network Performance

Decision Time (DT) [samples]	Incidence Angle [degrees]	# of Correct Classifications		# of Incorrect Classifications		Correct operation [%]
		Within Cluster	Outside Cluster	Within Cluster	Outside Cluster	
22	0°	6	13	1	7	70.37
24	0°	10	13	1	3	85.19
28	0°	25	0	2	0	92.59
22	45°	11	3	6	7	51.85
22	90°	25	0	2	0	92.59

Table III. Timing for Fault Detection and Classification Logic

Incidence Angle [degrees]	Fault Detection Logic			Fault Classification Logic		
	Min. Time	Max. Time	Average	Min. Time	Max. Time	Average
0°	1.5 ms or 3 samples	5 ms or 10 samples	2.5 ms or 5 samples	1 ms or 2 samples	Depending on the DT (22, 24, or 28 samples)	10.5 ms or 11 samples
45°	1 ms or 2 samples	6 ms or 12 samples	2.5 ms or 5 samples	1.5 ms or 3 samples	22	7.5 ms or 15 samples
90°	1.5 ms or 3 samples	5 ms or 10 samples	3 ms or 6 samples	3 ms or 6 samples	22	8 ms or 16 samples

- Overall performance for all three fault incidence angles and for DT = 24 samples (12 ms) expressed in percentage is 76.54 %. The longest time for fault detection and classification was 32 samples (corresponds to 16 ms) which is under 1 cycle.
- Overall performance for all three fault incidence angles and for DT = 28 samples (14 ms) expressed in percentage is 79.01 %. The longest time for fault detection and classification was 32 samples (corresponds to 16 ms) which is under 1 cycle.

Off-Line Application

This is a pattern recognition problem, where Neural Network, based on the sample set of fault pattern, first reorganizes and classifies training patterns according to patterns collective properties. After completing of the training procedure, NN is capable to classify new "unseen" patterns.

A number of EMTP simulations of various fault events were performed for generating fault patterns to be used for training and testing of the neural network. The maximum simulation time was 50 ms, and the sampling frequency was f_S= 2 kHz.

Total number of 619 fault patterns were generated in this way. Also 1 pattern labeled as normal state is generated using EMTP to represent the steady state (no fault state) of the power system.

Total number of fault patterns used for training was 324. In addition to that, one steady state pattern was provided during the training. Distribution of the patterns according to the fault type is given below in Table IV.

Table IV. Distribution of Training Fault Patterns

Fault Type	Fault Location		Fault resistance		No. of patterns
	Local	Remote	Low	High	
a-g	6	18	16	8	24
b-g	12	24	24	12	36
c-g	12	24	24	12	36
a-b	12	18	20	10	30
a-c	12	21	22	11	33
b-c	12	21	22	11	33
a-b-g	10	20	24	6	30
a-c-g	10	20	24	6	30
b-c-g	10	20	24	6	30
a-b-c	8	14	14	8	22
a-b-c-g	8	12	18	2	20
Total	**112**	**212**	**232**	**92**	**324**

Several different input formats into ANN are considered during design and testing. These input formats are summarized in Table V. Input formats 1, 2 and 3 consist of both prefault and postfault samples, while input formats 4, 5 and 6 contain only postfault samples. Since the proposed neural network algorithm contains no hidden layers (flat net) the network structure depends on the type of the input format. Number of neurons in the input layer is determined with the length of the input vector (e.g., for the input format 1 number of neurons is 600, while for the input format 5 that number is 99).

Table V. Neural Network Input Formats

Input format	Pattern length		Sample types
	Cycles	# Samples	
1	3	600	3-phase voltages and 3-phase currents
2	3	300	only 3-phase currents
3	3	300	only 3-phase voltages
4	1	198	3-phase voltages and 3-phase currents
5	1	99	only 3-phase currents
6	1	99	only 3-phase voltages

Results of training (clustering) for every input format are given in Table VI. Class membership of each cluster is determined during the supervised part of training. Cluster is considered homogenous if it contains fault patterns of particular type (a - g, b - c - g, a - b - c, etc.).

Table VI. Neural Network Training Results

Input format	# of training patterns	# of clusters after training
1	325	131
2	325	132
3	325	105
4	325	128
5	325	128
6	325	118

During the testing, NN was presented with 295 new fault patterns. Neural network never "saw" these patterns and the task was to classify new patterns based solely on the previous experience (i.e. using the information "learned" during the training).
The results are summarized according to the NN input formats in Table VII. Considering pairs of input formats 1 and 2, and 4 and 5 it can be noticed that NN performance is not deteriorating if only samples of phase currents are taken as pattern features. On the other hand, computational effort and time are greatly reduced if the voltage samples are disregarded.

CONCLUSIONS

Results presented in this paper demonstrate that:

- the NN selected gives good performance results for both real-time and off-line applications
- the supervised learning process significantly increases the performance
- the NN approach is quite beneficial since it allows for on-line learning which is not available in conventional techniques.
- the use of the same NN algorithms for both the real-time and the off-line fault analysis enables combining of these two tasks in one hierarchical system solution designed for fast fault analysis.

Table VII. Neural Network Classification Results

Input Format	Correct Classification			Incorrect Classification			Correct [%]
	Within Cluster	Outside Cluster	Total	Within Cluster	Outside Cluster	Total	
1	214	68	282	7	6	13	95.59
2	218	65	283	7	5	12	95.93
3	195	86	281	1	13	14	95.25
4	209	67	276	11	8	19	93.56
5	207	71	278	11	8	19	93.56
6	199	80	279	1	15	16	94.58

REFERENCES

[1] *Proceedings of the Symposia on Expert System Applications to Power Systems*: First - Stockholm/Helsinki, 1988; Second - Seattle, 1989; Third - Tokyo/Kobe, 1991; Fourth- Melbourne, 1993.

[2] *Special Issue on Knowledge-Based Systems in Electric Power Systems*, Proceedings of the IEEE, Vol. 80, No. 5, May 1992

[3] M. Kezunovic, "*Implementation Framework of an Expert System for Fault Analysis*", Third Symposium on Expert System Applications to Power Systems, Tokyo/Kobe, Japan, April 1991.

[4] M. Kezunovic, et. al., "*An Expert System for Substation Event Analysis*", IEEE Trans. on Power Delivery, Vol. 8, No. 4, October 1993.

[5] M. Kezunovic, P. Spasojevic, "*An Expert System for DFR File Classification and Analysis*", 4th Symposium on expert System Applications to Power Systems, Melbourne, Australia, February 1993.

[6] M. Kezunovic, et. al., "*Automated Fault Analysis Using Neural Network*", 9th Annual Conference for Fault and Disturbance Analysis, Texas A&M University, College Station, Texas, March 1994.

[7] M. Kezunovic, "*New Approaches to Expert System Utilization for Fault Diagnosis*", 26th Annual Frontiers of Power Conference, Stillwater, Oklahoma, October 1993.

[8] R. Khosla, T. S. Dillon, "*Combined Symbolic-Artificial Neural Net Alarm Processing System*", 11th PSCC, Avignon, France, September 1993.

[9] T. Dalstein, et. al., "*Neural Network Approach to Fault Direction Identification in Electric Power Systems*", North American Power Symposium 1993, October, Washington, D.C.

[10] R. D. Christie, et. al. "*High Impedance Fault Detection in Low Voltage Networks*", Paper 92 SM 507-4 PWRD, presented at the IEEE PES Summer Meeting, Seattle, WA, July 12-16, 1992

[11] A. F. Sultan, et. al., "*Detection of High Impedance Arcing Faults Using a Multi-Layer Perceptron*", IEEE Trans. on Power Delivery, Vol. 7, No. 4, October 1992.

[12] S. R. Fernando, K. L. Watson, "*High Impedance Fault Detection Using a Time-Delay Neural Network*", Intelligent Engineering Systems Through Artificial Neural Networks, ASME Press, Vol. 2, New York 1992.

[13] F. Islam, et. al., "*A Self-Organizing Neural Network Architecture to Detect HI Faults in Power Distribution Lines*", Intelligent Engineering Systems Through Artificial Neural Networks, ASME Press, Vol. 2, New York 1992.

[14] D. J. Sobajic, Y. H. Pao, "*On-Line Transient Stability Evaluation by Associative Dichotomous Classification*", 11th PSCC, Avignon, France, September 1993.

[15] D. J. Sobajic, Y. H. Pao, "*Artificial Neural-Net Based Security Assessment for Electric Power Systems*", IEEE/PES Winter Meeting, January, 1988, Paper #88WM-211-05.

[16] Y. H. Pao, D. J. Sobajic, "*Autonomous Feature Discovery for Critical Clearing Time Assessment*", Symposium on Expert Systems Applications to Power Systems, Stockholm, Sweden, August 22-26, 1988.

[17] D. J. Sobajic, et. al., "*On-line Monitoring of Power Systems Operating Conditions Using Artificial Neural Networks*", IEEE International Symposium on Circuit and Systems, Portland, OR, May 8-11, 1989

[18] *Electromagnetic Transient Program - Workbook*, Electric Power Research Institute, Palo Alto, California, September 1986.

APPENDIX

Following algorithm implements unsupervised learning technique which is a neural-net implementation of the ISODATA clustering algorithm.

Given is a set of P (p=1, 2, ..., P) patterns $\underline{x}^{(p)}$ where

$$\underline{x}^{(p)} = \left[x_1^{(p)}, x_2^{(p)}, \ldots, x_N^{(p)}\right]^T$$

Initialization run

Step 1: $\rho > 0$ is chosen.

Step 2: Find a center $\underline{b}$ of the entire data set

$$\underline{b} = \frac{1}{P}\sum_{p=1}^{P} \underline{x}^{(p)}$$

Find Euclidean distances between each pattern $\underline{x}^{(p)}$ and the center $\underline{b}$ and rank them in an increasing order.

Step 3: Form cluster no. 1

$\underline{b}_1(1) = \underline{x}^{(1)}$ (Meaning cluster C_1 with centroid $\underline{b}_1$ contains 1 pattern.)

Step 4: If $\left(\underline{x}^{(2)} - \underline{b}_1\right)^T \left(\underline{x}^{(2)} - \underline{b}_1\right) \le \rho^2$ then adapt $\underline{b}_1$ as

$$\underline{b}_1(2) = \underline{b}_1(1) + \frac{1}{2}\left(\underline{x}^{(2)} - \underline{b}_1(1)\right)$$

If $\left(\underline{x}^{(2)} - \underline{b}_1\right)^T \left(\underline{x}^{(2)} - \underline{b}_1\right) > \rho^2$ then form cluster 2 as

$$\underline{b}_2(1) = \underline{x}^{(2)}$$

In doing so, after presenting $q < P$ patterns the situation is as follows:

m - clusters exists, their centroids $\underline{b}_m$ are known and we know how many patterns belong to each cluster n_m.

When we present next pattern $q+1$ we first allocate the closest cluster τ, by

$$\min_j \left\{ \left(\underline{x}^{(q+1)} - \underline{b}_j\right)^T \left(\underline{x}^{(q+1)} - \underline{b}_j\right) \right\} = r_\tau^{\,2}$$

and then compare $r_\tau^{\,2}$ and ρ^2:

If $r_\tau^{\,2} \le \rho^2$ then adapt cluster as

$$\underline{b}_\tau(n_\tau + 1) = \underline{b}_\tau(n_\tau) + \frac{1}{n_\tau + 1}\left(\underline{x}^{(q+1)} - \underline{b}_\tau(n_\tau)\right)$$

If $r_\tau^{\,2} > \rho^2$ then form new cluster as

$$\underline{b}_{m+1}(1) = \underline{x}^{(q+1)}$$

This procedure is repeated until the entire set of patterns is processed once.

Stabilization run

Step 5: We present every pattern, $\underline{x}^{(p)}$, again.

Let say presently pattern p belongs to cluster C_k. The shortest distance between $\underline{x}^{(p)}$ and all existing centroids $\underline{b}_j$ is found

$$\min_j \left\{ \left(\underline{x}^{(p)} - \underline{b}_j\right)^T \left(\underline{x}^{(p)} - \underline{b}_j\right) \right\} = r_\tau^{\,2}$$

Then,

1. If $\tau = k$ and $r_\tau^2 \le \rho^2$

then no learning occurs; check next pattern $p+1$.

2. If $\tau \ne k$ and $r_\tau^2 \le \rho^2$

then adapt $\underline{b}_\tau$ and $\underline{b}_k$ as

$$\underline{b}_\tau(n_\tau + 1) = \underline{b}_\tau(n_\tau) + \frac{1}{n_\tau + 1}\left(\underline{x}^{(p)} - \underline{b}_\tau(n_\tau)\right)$$

$$\underline{b}_k(n_k - 1) = \underline{b}_k(n_k) - \frac{1}{n_k - 1}\left(\underline{x}^{(p)} - \underline{b}_k(n_k)\right)$$

for $n_k > 1$

Cluster $\underline{b}_k(n_k - 1)$ is discarded if $n_k = 1$

3. If $r_\tau^2 > \rho^2$ form new cluster C_m,

$\underline{b}_m(1) = \underline{x}^{(p)}$ and adapt "previous" centroid $\underline{b}_k$ as

$$\underline{b}_k(n_k - 1) = \underline{b}_k(n_k) - \frac{1}{n_k - 1}\left(\underline{x}^{(p)} - \underline{b}_k(n_k)\right)$$

for $n_k > 1$

Cluster $\underline{b}_k(n_k - 1)$ is discarded if $n_k = 1$