New Design of a Neural Network Algorithm for Detecting and Classifying Transmission Line Faults

S. Vasilic, *Student Member, IEEE,* M. Kezunovic, *Fellow, IEEE*

Abstract--**This paper introduces a new artificial intelligent based approach for detecting and classifying the faults in power system networks. This approach utilizes unique type of neural network specially developed to deal with large amount of input data. A model of an actual power network is implemented in ATP program and used for simulating the fault scenarios on transmission lines. Protective algorithm is implemented in MATLAB and interacts with the network model simulations in ATP. Procedures of generating training and testing patterns are performed carefully to ensure covering of all possible events. Training and testing phases of the neural network algorithm are optimized to improve classification of a variety of previously unseen patterns.**

Keywords--**clustering methods, electromagnetic transients, neural networks, pattern classification, power system faults, protective relaying, testing, training.**

I. Introduction

THIS paper introduces artificial neural network based technique for detecting and classifying faults on transmission lines. Transmission line faults happen randomly, and they are outcome of unpredictable conditions. Several varying parameters: type of fault, fault location, fault impedance, and fault incident time determine the corresponding transient current and voltage waveforms detected by the relays at line ends. The main role of the relaying principle is detecting and classifying the faults, based on three phase voltage and current samples. The new detection and classification approach has to reliably conclude, in a very short time (1-2 cycles), whether and which type of fault occurs under a variety of time-changing operating conditions [1]. Protective relay accordingly performs action, usually disconnecting faulted phase/line and/or initiating some alarm and control signals.

Various applications of neural networks were used in the past to improve the distance relaying of transmission lines [2]. These applications are mainly based on multilayer feed-forward networks. Training of these networks is very slow, needs much larger training sets, and very easily converges on local minima, whenever input patterns with large dimensionality are present as in this particular case. Furthermore, retraining of this type of network with new training data is rather difficult.

Instead of using multilayer neural network, a unique type of neural network may be used for fault classification [3-6]. This network is based on ISODATA clustering algorithm [7] and belongs to a group of special neural networks named Self-Organizing Maps [3]. The adaptive behavior of the neural network is described by Adaptive Resonance Theory [8]. The main aim of this work is to enhance that neural network based clustering algorithm. In the previous version of the algorithm relatively small number of training and testing patterns was used. Training patterns did not cover different values of fault angles and significant performance deterioration became obvious, due to insufficient network training. Number of passes through the stabilization phase was limited (to speed-up the training) and this prevented establishing optimal clustering structure. Classification of test patterns was done based on predetermined number of nearest clusters, instead of selecting optimal number of neighbors for each implementation. Also, fault location classification was performed in three cycles making it challenging for an on-line classification of the fault zone in one cycle.

The new algorithm has to overcome observed deficiencies. It has to show the importance of proper generation of the sets of training and testing patterns and apply extended set of scenarios for improved algorithm design and evaluation. Moreover, some steps in the algorithm training and testing phases may be improved assuring smaller classification error. Additional task is establishing specially devoted simulation environment, where training and testing procedures can be easily controlled and performed. This is done through interfacing different programs, exchanging simulation parameters and results, and using appropriate graphical interface.

A specially developed power network model has been implemented in the electromagnetic transient program ATP [9]. That model has been interfaced to MATLAB package for automatic simulation of a large number of scenarios [10-12]. Simulation outputs are used as a signal generator for the neural

This study was supported by an Army/EPRI contract between EPRI and Carnegie Mellon University, and has been carried out by Texas A&M University under the subcontract titled "Self-Evolving Agents for Monitoring, Control and Protection of Large, Complex Dynamic Systems".

S. Vasilic and M. Kezunovic are with the Department of Electrical Engineering, Texas A&M University, College Station, TX 77843-3128 USA (e-mails respectively: svasilic@ee.tamu.edu, kezunov@ee.tamu.edu).

network algorithm design, implemented in MATLAB.

The paper is organized as follows. Description of the neural network detection and classification algorithm is given in section II. Section III through its subsections shows the selected model of an actual power network, design implementation steps (devoted to pattern generation, algorithm training and testing), and provides the classification results. The conclusion is given at the end.

II. Neural Network Classification Algorithm

Neural networks try to produce a concise representation of system's behavior through identifying natural groupings of data from large data sets. The aim of this procedure, called clustering, is to partition a given set of input data (patterns) into several groups or clusters, so that each pattern is assigned to a unique cluster. Patterns that belong to the same cluster should be as similar as possible, while patterns that belong to different clusters should be as different as possible. Class label is assigned to each cluster, where class symbolizes a group of patterns with a common characteristic.

Self-organizing maps are special type of neural networks, and they map input patterns with similar features into contiguous clusters after enough input patterns have been presented. The similarity between patterns is usually measured by calculating the Euclidean distance between two *n*-dimensional vectors. After training, self-organized clusters represent prototypes of classes of input patterns.

Adaptive Resonance Theory defines forming a new cluster whenever a pattern, sufficiently different from all previously presented patterns, appears. Adaptive resonance architectures are capable of continuos training with non-stationary inputs.

This neural network is without hidden layer and its self-organized structure depends only on the presented input data set. The neural network training consists of unsupervised and supervised learning phases. In the unsupervised learning, patterns are presented without their class labels, and this procedure tries to identify prototypes that can serve as cluster centers. In the supervised learning the class label is associated with each data point. Vigilance parameter is a confidence measure and is being tuned, consecutively decreasing during iterations. It controls the number and size of generated clusters. The large values allow large deviations from the cluster centers and hence lead to a small set of clusters, while small values lead to a large number of tight clusters.

The initial data set, containing all the patterns, is firstly processed using *unsupervised learning*. The outcome of unsupervised learning is a stable family of clusters, defined as hyperspheres in an *n* dimensional space, where *n* denotes the number of input features. Unsupervised learning forms stable family of both homogenous (having patterns with the same class label) and non-homogenous (having patterns with two or more class labels) clusters. It does not require either the initial guess of the number of cluster, or the initial cluster center coordinates. It consists of two steps: initialization and stabilization.

Initialization phase begins with calculating the center of the entire data set. Then Euclidean distances between each pattern and the center are calculated and sorted in an increasing order. Now, the first cluster is formed by taking pattern closest to the center of data set, and with the radius equal to actual value of the vigilance parameter. Furthermore, all the remaining patterns are presented, in order of the sorted distances. Distances between the pattern and existing clusters are calculated. The minimum distance and corresponding (nearest) cluster are found. If the minimum distance is less or equal to the vigilance parameter, then the actual pattern is classified into the nearest cluster and the cluster center is updated by adding new pattern to the cluster. Otherwise, if the minimum distance is greater then the vigilance parameter, the actual pattern forms a new cluster.

During *stabilization phase* the clustering algorithm is reiterated until a stable cluster structure occurs and there are no patterns changing their cluster membership during the iterations. All patterns are presented again. Distance between each pattern and existing clusters are calculated, and minimum distance and corresponding (nearest) cluster are found. Also, a cluster where the pattern was previously classified is found. If pattern was classified into the actual nearest cluster and minimum distance is less or equal to the vigilance parameter then there is no learning (no changes in the cluster structure). If the pattern was not classified into the actual nearest cluster and minimum distance is less or equal to the vigilance parameter then the pattern is moved to the actual nearest cluster and that cluster as well as the cluster where pattern was previously classified are updated. If minimum distance is greater then the vigilance parameter, a new cluster is formed and the cluster where the pattern was previously classified is updated. After processing all patterns, clusters remained without patterns are discarded, because their patterns have been moved to other clusters. Stabilization phase is repeated many times until no pattern changes its cluster membership.

Supervised learning separates non-homogenous clusters from the homogeneous ones. It assigns class labels to the homogeneous clusters, and these clusters and their patterns are extracted from further iterative training process. Set of remaining patterns (patterns in non-homogeneous clusters) is transformed into new, reduced data set of training patterns. If new set of training patterns is not empty, vigilance parameter is decreased, and unsupervised and supervised learning procedures are repeated. Otherwise, if either all actual training patterns are members of only homogeneous clusters, or current value of vigilance parameter is less then specified value, learning is completed.

During the testing phase Euclidean distances between test pattern and established clusters (prototypes) are calculated, and k-nearest neighbor rule [13] is used to classify the pattern. Given a set of classified data, the k-nearest neighbor rule determines the classification of a new pattern based on the most represented class label amongst the *k* nearest clusters, retrieved from the cluster structure adopted during training. The outcome of the testing phase are class labels assigned to testing patterns.

III. Design Implementation

A. Power Network Modeling

A typical 345 kV power system section, from Reliant Energy (RE) HL&P company, was modeled for the testing and simulation studies. The reduced network equivalent was obtained by using the load flow and short circuit data, and verified using both the steady state and transient state results. Fig. 1 shows one-line diagram of the reduced equivalent for the used section. STP-SKY section model has nine buses and contains both short and long transmission lines. This reduced system is convenient for producing fault waveforms to be used for transient testing of protective algorithms.

B. Generation of Training and Testing Patterns

Model of the given power network (Fig. 1) is implemented in Alternative Transient Program (ATP) program, and shown in Fig. 2. This model is used for simulating various fault

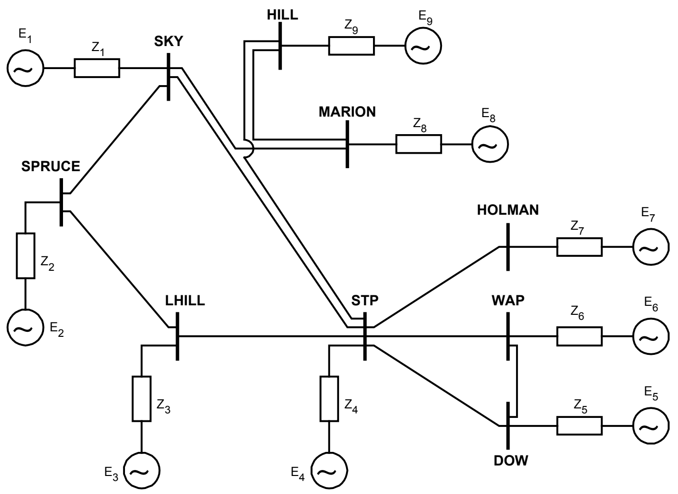

Fig. 1. RE HL&P STP-SKY Power Network Model

scenarios on one of its transmission lines (STP-SKY), by varying fault parameters. An intelligent, neural network based, algorithm is located at the bus SKY, at one end of the selected line (notation AB1 on the scheme in Fig. 2). It takes voltage and current measurements from that end of the line and has to be trained and simulated to recognize the fault on that line. Current and voltage samples obtained through simulations are used for forming training and testing patterns for protective algorithm learning and evaluation.

The MATLAB program interacts with ATP simulations. Generation of patterns may be deterministic or random. The classification algorithm requires deterministic generation of training patterns, by specifying several values for each of the four fault parameters and combining these values to cover diversity of fault cases. This forms prototypes that represent the space of possible events. The number of training patterns is limited and has to roughly cover all important situations in the power network. Random generation is based on the random setting of all fault parameters only constrained by the user. This is the requirement for generating testing patterns for algorithm evaluation in heuristic, previously unseen situations.

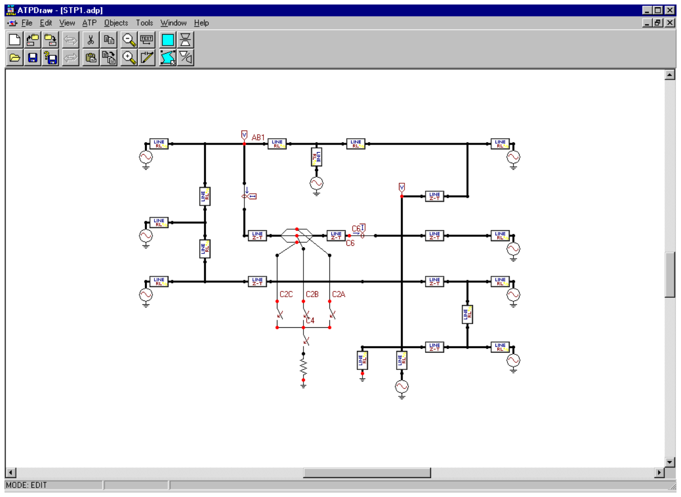

Fig. 2. RE HL&P STP-SKY network model implemented in ATP program

The testing patterns might be very heterogeneous and quite different from the training patterns since there are many operating states and possible events in the power network. They are classified according to their similarity to prototypes adopted during training.

This implementation includes all 11 types of fault (AG, BG, CG, AB, BC, CA, ABG, BCG, CAG, ABC, ABCG) and the normal state. Possible values for fault distance from the SKY bus are anywhere between 0-100 percents of the total line. Fault resistance might be theoretically anywhere between 0 and ∞ Ohms. Fault angle is between 0 and 360 deg. Fault angle is transformed into corresponding fault incident time, where 0 deg is equal to the initial fault incident time, and 360 deg is equal to the initial fault incident time increased for 1 cycle. Parameters are also the initial fault incident time (for angle 0 deg), and simulation step and end times.

Combinations of selected values for fault parameters define total number of simulation cases. After each simulation, output data in specific ATP format containing time and three phase voltage and current variables as well as characteristics of the implemented fault are converted into MATLAB format. The example of the simulation output data for one specific case, phases A to B to ground fault (ABG), is shown in Fig. 3.

Examples of training patterns for different values of fault parameters are shown in Figs. 4 to 8. The algorithm uses training patterns formed by sampling all three phase voltage and/or current measurements in the selected time window. Parameters used for algorithm training in this particular simulation example are: three phase currents selected as the data for training; starting time for taking patterns is 0 seconds after the fault incident time; time window is 16.7 ms or 1 cycle; sampling frequency is 2 kHz (33 samples per cycle). Features of the training patterns are extracted by using simulation data obtained in a desired time window and with selected sampling frequency. Phase A, B, C currents sampled during one cycle after the fault occurs are extracted and placed together in one row (Fig. 9) to form feature vector of 99 components (3 phases with 33 samples each). Then all training patterns are normalized by scaling all features of all patterns to

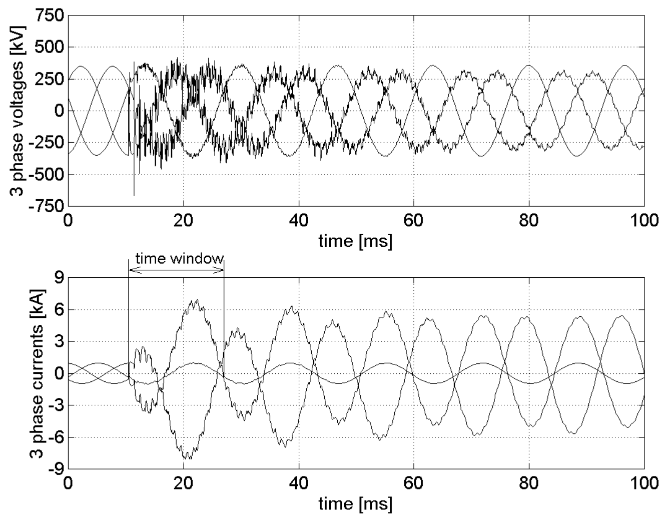

Fig. 3. Voltage and current samples for ABG fault, fault distance 50%, fault impedance 0 Ohm, and fault incident angle 0 deg.

have the mean zero, and variance one, and this scaling value is used later for normalization of the testing patterns. Different values of fault parameters have to be taken into account for training, to avoid misclassification later. Figs. 4-7 show responses during varying type of fault, fault distance, impedance and angle, respectively. Fig. 8 shows how combination of varying parameters may cause misclassification of faults in Zone I and Zone II, supposing that the bound between zones is established at 80% of the line length, counting from SKY bus.

Parameters used for generation of the training patterns in the simulation example given to illustrate the whole algorithm design are: all 11 types of fault and normal state; fault distances 5 to 95 % in increments of 10%; fault resistance (for ground faults) 0, 10, 20 Ohms; fault angle 0 to 330 degrees, in increments of 30 degrees. Total number of training patterns by combining all parameters is 2652, and all training patterns are shown in Fig. 9.

Parameters used for generation of testing patterns are

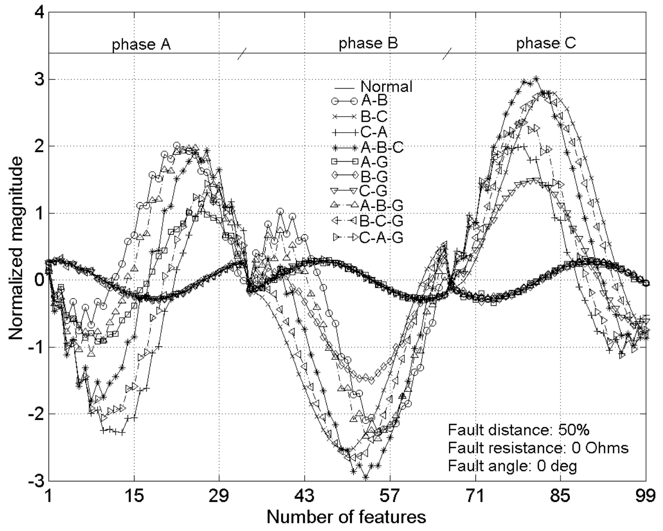

Fig. 4. Example of training patterns for all 11 types of fault. Other fault parameters are constant.

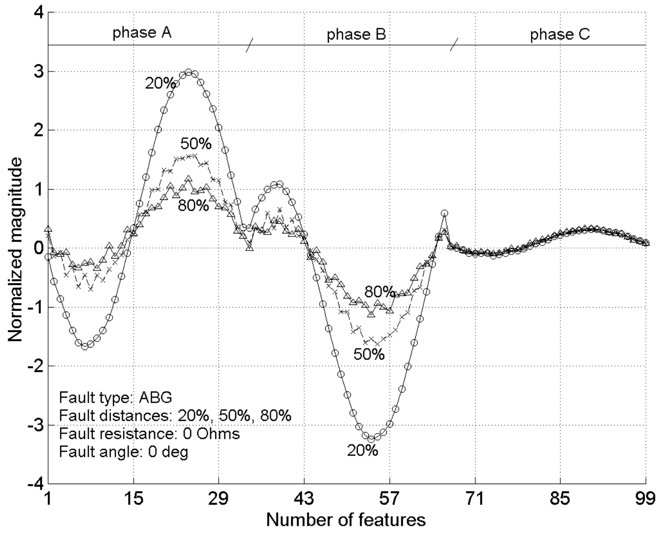

Fig. 5. Example of training patterns for different fault distances. Type of fault is ABG and other fault parameters are constant.

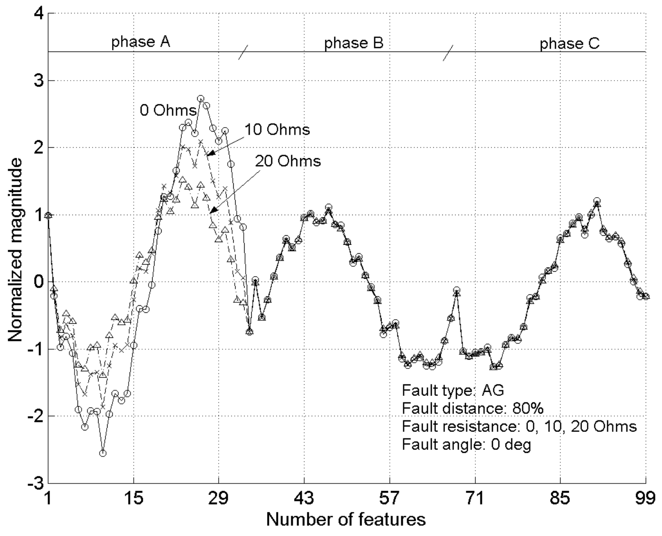

Fig. 6 Example of training patterns for different fault impedances. Type of fault is ABG and other fault parameters are constant.

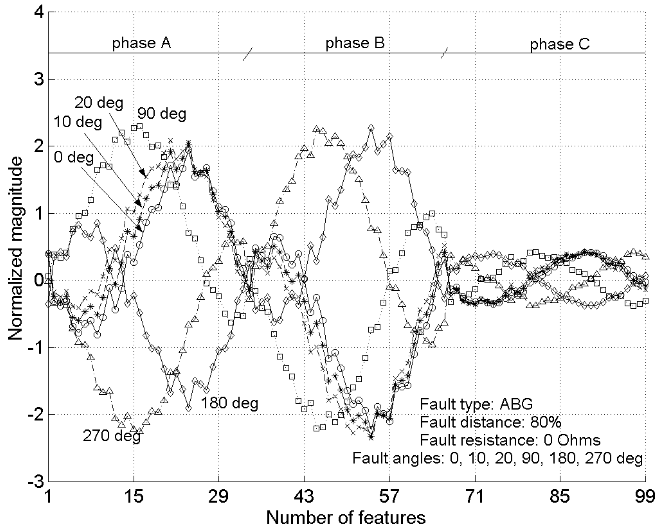

Fig. 7 Example of training patterns for different fault incident time. Type of fault is ABG and other fault parameters are constant.

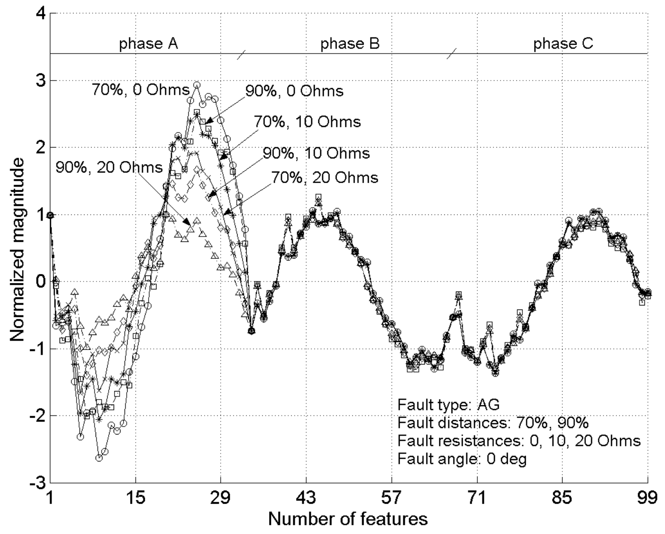

Fig. 8 Example of training patterns for combined different values of fault parameters.

uniformly random selection of fault type, distance between 0 and 100%, angle between 0 and 360 deg, and normally random selection of fault resistance, with mean 0 Ohms and variance 10 Ohms (taking only positive values). Total number of testing patterns is 5000.

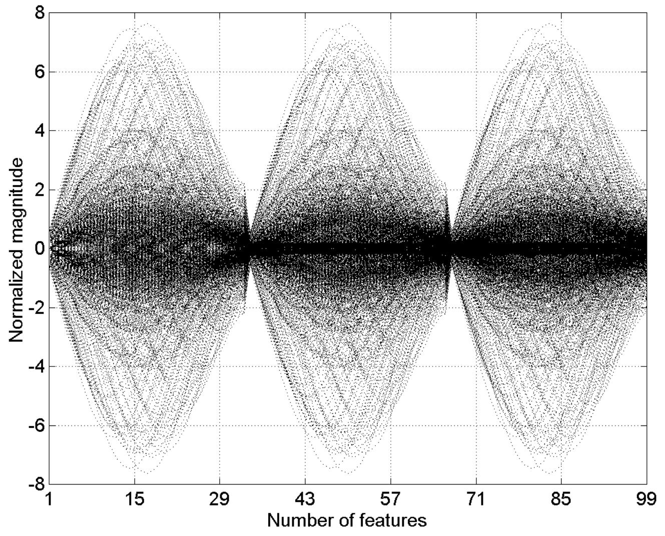

Fig. 9 All training patterns.

C. Algorithm Training

Selection of data for training includes either three phase currents, or three phase voltages, or both the three phase currents and voltages. Vigilance parameter (cluster radius) is defined with its initial (maximal) and minimal values, as well as with the decreasing factor during iterations. Type of classification might be based on detection of fault type (Normal, AG, BG, CG, AB/ABG, BC/BCG, CA/CAG, ABC/ABCG), fault zone (Normal, Zone I, Zone II), fault resistance (Normal, Low, High), or any combination among them. Boundary distances between zones I and II of the fault, and between low and high fault resistance, also have to be specified.

The first step in algorithm training is to extract the training patterns from generated patterns and it is described in the previous section. Two types of classifications were implemented. Training I was performed for establishing the cluster structure capable of recognizing only the type of fault. Training II was performed for establishing the cluster structure capable of recognizing type of fault and zone of fault. Boundary distance between the first and second zone is 80% of the line length. After several hours of iterations, both training procedures terminated successfully. Simulation output of Training I is the cluster structure containing 269 clusters, and of Training II is the cluster structure containing 706 clusters.

D. Algorithm Testing and Classification Results

In the testing phase, input to the neural network is in the form of the “sliding” data window containing samples of phase currents and/or voltages. Classification of testing patterns is performed by using cluster structure established during training and applying the k-nearest neighbor rule. Input parameter for algorithm testing is only the number k of the nearest neighbors for the rule. Testing patterns are extracted from generated patterns using the same procedure as for training patterns and have equal number of features. Testing patterns are normalized by scaling all features of all testing patterns with the same factor used for scaling of training patterns. For each testing pattern, distances to all clusters are computed and sorted in an increasing order. The most frequent class label of k nearest clusters is computed and assigned to actual pattern. If the input pattern belongs to any of the “normal-state” clusters then the input window is "moved" for one sample and the comparison is performed again. If the input pattern does not belong to the “normal-state” clusters then fault is detected and execution of the fault classification logic is initiated. The parameter used to force the neural network to make the final decision is the time. After decision time has expired, if pattern still does not come back to the normal state, neural network will classify fault event according to the fault type detected in that instance.

Average classification error for the entire testing set of patterns for selected values of the nearest neighbors from 1 to 12, is established by comparing the true and computed class labels. It is shown in Fig. 10 for both training cases. The graphic helps in finding optimal values for parameter k in both cases. Optimal value k for Training I is 1, and classification error for optimal k is 0.48%. Optimal value k for Training II is 1, and classification error for that k is 6.34%. Obviously, classifying the zone of fault is much more difficult task then classifying the type of fault.

IV. CONCLUSION

The most important aspect of this research is to show that the proposed neural network approach enables better detection of faults in power system networks and proper action to protect the network. An example of an actual power network was modeled in ATP program and used to simulate various

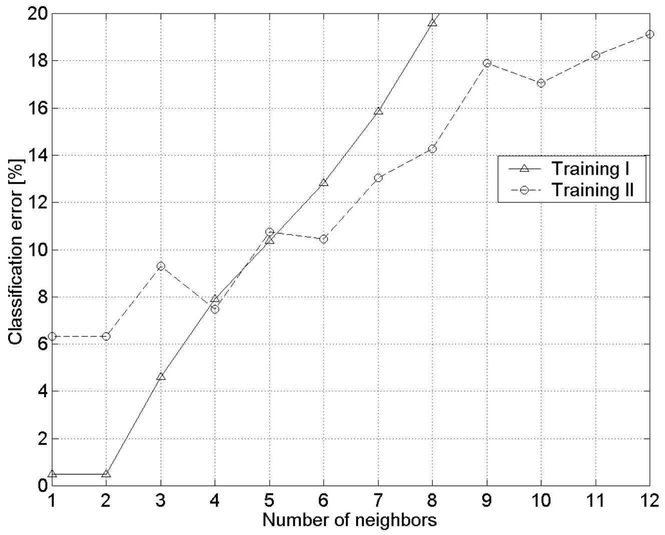

Fig. 10. Results of classification error for Training I and Training II.

fault events in the network. Training and testing patterns are extracted from the measurements. Neural network based clustering algorithm, implemented in MATLAB, is used to form pattern prototypes, homogenous structure of clusters representing various classes of input data set. Testing patterns are classified by combining cluster structure and k-nearest neighbor rule.

This advanced algorithm has several important benefits comparing to the previous version of the algorithm. New algorithm offers easy selection of desired scenarios and algorithm parameters by using MATLAB. Various types of the classification may be selected and combined. Bounds between zones of fault may be easily changed. Libraries of the training and testing patterns, and cluster structures might be generated and combined to achieve better algorithm training and validation. Extended sets of training and testing patterns have been implemented. Since training patterns are generated uniformly, testing patterns are generated randomly to ensure heuristic covering of all possible events. Also, previous version was trained only for particular values of fault angle (0 and 90 deg), while the new algorithm is trained for all possible values of fault angle (0-360 deg). Fault location classification is now performed in one cycle, instead of in three cycles as it was done earlier. Number of passes through stabilization phase is now unlimited and enables forming more realistic prototypes. Tuning of the new algorithm finds optimal value for number of neighbors in k-nearest neighbor rule, while in the previous version only predetermined number of three nearest neighbors was used.

V. References

[1] Power System Relaying Committee, Working Group D5 of the Line Protection Subcommittee, “Proposed statistical performance measures for microprocessor-based transmission line protective relays, Part I and II”, *IEEE Trans. Power Delivery*, vol. 12, no. 1, pp. 134-156, Jan. 1997.

[2] M. Kezunovic, "A Survey of Neural Net Applications to Protective Relaying and Fault Analysis", *Engineering Intelligent Systems*, vol. 5, no. 4, pp. 185-192, Dec. 1997.

[3] Y. H. Pao, *Adaptive Pattern Recognition and Neural Networks*, Reading: Addison Wesley, 1989, p. 309.

[4] Y. H. Pao and D. J. Sobajic, "Combined Use of Unsupervised and Supervised Learning for Dynamic Security Assessment", *IEEE Trans. Power Systems*, vol. 7, no 2, pp. 878-884, 1992.

[5] M. Kezunovic M., I. Rikalo, and D. Sobajic, "High-speed Fault Detection and Classification with Neural Nets", *Electric Power Systems Research*, vol. 34, pp. 109-116, 1995.

[6] M. Kezunovic and I. Rikalo, "Detect and Classify Faults Using Neural Nets", *IEEE Computer Applications in Power*, vol. 9, no. 4, pp. 42-47, 1996.

[7] G. H. Ball and D. J. Hall, "A Clustering Technique for Summarizing Multivariate Data", *Behavioral Science*, vol. 12, pp. 345-370, 1967.

[8] G. A. Carpenter and S. Grossberg, "ART2: self-organization of stable category recognition codes for analog input patterns", *Applied Optics*, vol. 26, no. 23, pp. 4919-4930, Dec. 1987.

[9] CanAm EMTP User Group, *Alternative Transient Program (ATP) Rule Book*, Portland, 1992.

[10] The MathWorks, Inc., *Using MATLAB*, Natick, Jan. 1999.

[11] M. Kezunovic and S. Vasilic, "Advanced Software Environment for Evaluating Protection Performance During Power System Disturbances Using Relay Models", submitted to CIGRE SC 34 Colloquium, Romania, Sep. 2001.

[12] M. Kezunovic and S. Vasilic, "Design and Evaluation of Context-Dependent Protective Relaying Approach", submitted to IEEE Porto Power Tech' 2001 Conference, Portugal, Sep. 2001.

[13] T. M. Cover and P. E. Hart, "Nearest Neighbor Pattern Classification", *IEEE Trans. Information Theory*, vol. IT-13, pp. 21-27, 1967.

VI. Biographies

Slavko Vasilic (S'00) received his B.S. and M.S. degrees in electrical engineering from University of Belgrade in 1993. and 1999., respectively, and currently is a Ph.D. candidate in electrical engineering at Texas A&M University. His research interests are neural networks, fuzzy logic, genetic algorithms, multivariable, robust and adaptive systems, and their implementation in process control and pattern recognition, and especially in power systems control, protection and monitoring.

Mladen Kezunovic (S'77, M'80, SM'85, F'99) received his Dipl. Ing. degree from the University of Sarajevo, the M.S. and Ph.D. degrees from the University of Kansas, all in electrical engineering, in 1974, 1977 and 1980, respectively. He has been with Texas A&M University since 1987 where he is the Eugene E. Webb Professor and Director of Electric Power and Power Electronics Institute. His main research interests are digital simulators and simulation methods for equipment evaluation and testing as well as application of intelligent methods to control, protection and power quality monitoring. Dr. Kezunovic is a registered professional engineer in Texas, and a Fellow of IEEE.