TRAINING AN ARTIFICIAL NEURAL NETWORK TO DISCRIMINATE BETWEEN MAGNETIZING INRUSH AND INTERNAL FAULTS

LUIS G. PEREZ Member, IEEE
ALFRED J. FLECHSIG Senior Member, IEEE
JACK L. MEADOR Member, IEEE
ZORAN OBRADOVIC Member, IEEE

School of Electrical Engineering and Computer Science
Washington State University

Abstract - A feed forward neural network (FFNN) has been trained to discriminate between power transformer magnetizing inrush and fault currents. The training algorithm used was *back-propagation*, assuming initially a sigmoid transfer function for the network's processing units ("neurons"). Once the network was trained the units' transfer function was changed to hard limiters with thresholds equal to the biases obtained for the sigmoids during training. The off-line experimental results presented in this paper show that a FFNN may be considered as an alternative method to make the discrimination between inrush and fault currents in a digital relay implementation.

Keywords - Inrush Current, Power Transformer Protection, Digital Relays, Neural Networks.

1. INTRODUCTION

Any power transformer protective scheme has to take into account the effect of magnetizing inrush currents, since this effect could cause miss-operation of the relays[1]. The two classic methods used to avoid undesired tripping due to inrush currents are: 1) implementation of delays in the protective devices, and 2) restraining or blocking the relay operation according to the harmonic content of the measured current. The first solution has been used for primary overcurrent protection and in differential schemes. However, this option is undesirable because of the potential danger of delaying the tripping time during a real internal fault. The second solution is based on the fact that the second harmonic component of the inrush current is considerably larger than in a typical fault current[2], and schemes based on the detection of the second (sometimes the fifth) harmonic were proposed and implemented in both analog and digital differential relays with good results [3,4,5,6,7].

In a recent paper [8] it was reported that, in certain cases, harmonics are generated during internal faults in transformers and therefore the detection of the second and/or fifth harmonic is not a sufficient index to determine if the measured overcurrent is an inrush or an internal fault.

93 WM 037-2 PWRD A paper recommended and approved by the IEEE Power System Relaying Committee of the IEEE Power Engineering Society for presentation at the IEEE/PES 1993 Winter Meeting, Columbus, OH, January 31 - February 5, 1993. Manuscript submitted Augsut 6, 1992; made available for printing November 10, 1992.

In that paper, a method based on the use of one of the primary phase voltages as the control signal is proposed. There are also methods for transformer protection that are implicitly immune to magnetizing inrush [9,10]; and some other effective ways for detecting inrush have been proposed [11,12].

The method described in this paper detects inrush current based on recognizing its wave shape, more precisely, in differentiating its wave shape from the fault wave shapes. This discrimination can be done in different ways, one of the most common methods uses the harmonic analysis of the signal, while others are based on the use of neural networks. The work reported here is a description of an experimental demonstration that a feed forward neural network could be used as an alternative method to discriminate between inrush magnetizing current and internal faults in power transformers. The network and its training process were adapted to the goal of implementing the algorithm in a digital protective relay.

2. NETWORK CHARACTERISTICS

2.1. General

The general architecture of a feed forward neural network is shown in Figure 1. The most important characteristics of this network are: 1) processing units are grouped by layers, and 2) the processor interconnect is

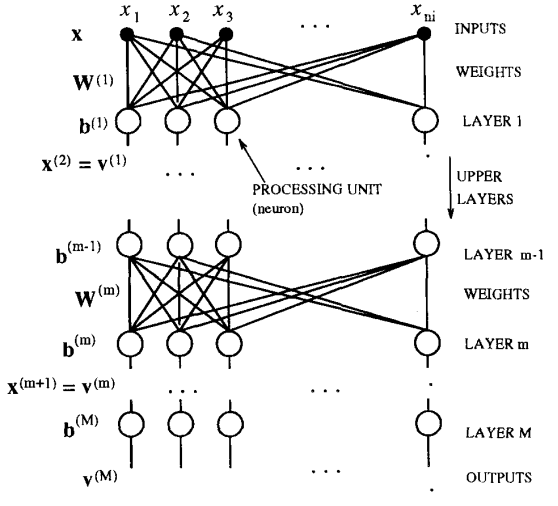

Figure 1. General architecture of a FFNN.

organized such that all inputs to a layer come exclusively from outputs originating in some previous layer (the specific FFNN used does not have connections skipping layers). The basic equations that define the way this FFNN computes its output given an input vector $\mathbf{x}$ are presented in Appendix 1. The FFNN training process consists of determining the weights $\mathbf{W}^{(m)}$ and the units' biasing $\mathbf{b}^{(m)}$, in order to make the network respond in a given way. The training method used in this work was the well known ***back-propagation algorithm***[13,14,15,16]. Appendix 2 gives a basic description of the training process using back-propagation, and defines the training matrices.

2.2. Input - Output

The FFNN implementation used is constrained in part by the nature of the current classification problem and also in part by existing digital relay system organization. Based on this, the following criteria can be stated:

i) Digital relays base their operation on samples of the measured quantity (current, in this case). The sampling rate and the data window varies depending on the application. For this particular case, since the NN must recognize wave shapes, it seems to be logical to use a one cycle length window.

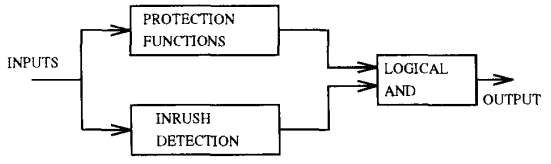

FIGURE 2.a. INRUSH DETECTION AS A PERMISSIVE

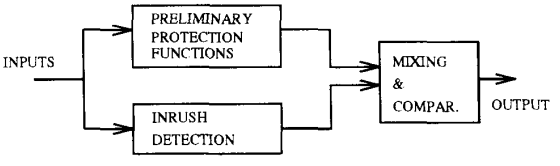

FIGURE 2.b. INRUSH DETECTION RESTRAINING OPERATION

Figure 2. Simplified block diagram of a protective relay using two possible implementations of the inrush detection function.

ii) As shown in Figure 2, the function of recognizing inrush is accessory, i.e., it can be assumed that the digital relay has a different function to implement the power transformer protection (for example, primary overcurrent or differential principle), inrush detection can be used as permissive for the relay operation, or to restrain the relay operation. Independently of the way this combination is made, both functions (inrush detection and protection) may use the same sampling rate. A sampling rate of 12 samples per cycle (720 Hz), was chosen in view of reported experience on different digital relay designs [17].

iii) The fact mentioned above bounds the problem as one of designing a FFNN that, given a sequence of samples of the transformer current, it can distinguish the two wave shapes. A good approach allows to have a network capable of determining if the input current is or is not an inrush, this means that the number of outputs of the neural network (number of units in the output layer) NN must be 1, necessary to indicate "true" and "false," or *"0"* and *"1"*, as indicated in section 3.2. In this case it was chosen that the network's output be *0* when the applied current is an inrush, and *1*, when it is not an inrush.

2.3. Basic Architecture

A functional block representation of the scheme used to achieve the goals stated in the former sections is shown in Figure 3. This kind of network is sometimes called time-delay neural network [16] and has been used recently in another power system application [18]. Feed forward networks have also been applied to the detection of high impedance faults with remarkable results [19,20]. Notice that the scheme given in Figure 3 is equivalent to the scheme of Figure 1, if the input x_k is equal to the k^{th} sample of input $x(t)$. In other words, the network will receive the 12 samples of each window every sample period, and must make a decision based on these 12 samples (i.e. one cycle), which implies the number of inputs must be 12. The number of layers and the number of units on each layer is determined by the heuristic process described in section 4.1 (the only layer completely defined at this point is the output layer).

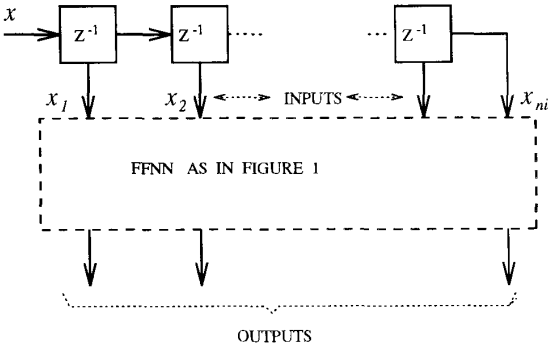

Figure 3. Time-delay network.

3. TRAINING STRATEGY

3.1. Training Examples

Given that the network has to distinguish between two kind of signals, two sets of examples signals (cases) were prepared for that purpose: the inrush cases and the fault cases.

The inrush cases were measured in the laboratory energizing at random a small power transformer of 50 VA, 120/240 V. The cases taken to train the network were chosen such that they represented an acceptable range of inrush current shapes (the six cases described in [21] were considered) Those signals were stored in a PC by means of a program developed at WSU [22]. The sampling rate at which the inrush signal examples were measured was originally 84 samples per 60 Hz cycle, and then re-sampled (at a rate 7 times lower) to get the de-

sired 12 samples/cycle. Figure 4 show some of the inrush training signals.

The fault cases were generated by computer, some of these using an electromagnetic transients program [23], others by simple generation of fault-like signals (response of an R-L circuit), and a third group of fault-like currents infected with second, third and fifth harmonics.

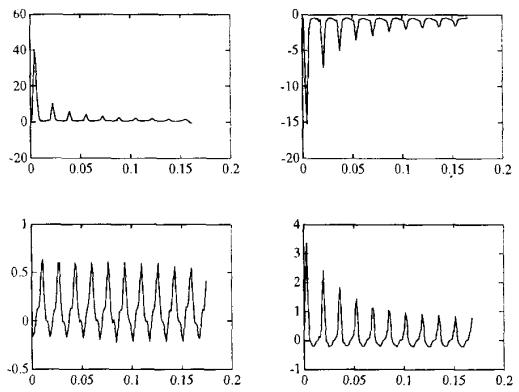

Figure 4. Four inrush training cases.

Two special cases grouped with the inrush cases were the zero current (indicating the transformer is de-energized) and the load current (a simple sinusoidal wave with a magnitude equal to the transformer load current).

For each case, signals were sampled at 12 samples per cycle (with a window length equal to one cycle) and the total of samples was limited to have about 100 windows (i.e. 1.67 seconds per case).

3.2. Data Windowing and Training Matrices

The training matrices were built in such a way that the network was trained to produce a *0* output when the presented signal was an inrush, a zero current or a full load condition; and a *1* output for the other cases (the authors do not insinuate the method can be used to do all the protection function, since additional research will be needed to prove it).

Vectors $\mathbf{p}_j$ were built from the cases in the following way (see Appendix 2 for the definition of these vectors). Suppose the first of the cases' signals is characterized by the sequence:

$$i = [i_1, i_2, \ldots i_k, \ldots i_{112}] \qquad (1)$$

this means, a sequence of 112 samples. The first vector $\mathbf{p}_1$ elements corresponds to the first 12 samples of i, the vector $\mathbf{p}_2$ entries are the 12 samples from i_2 to i_{13} and so on, until completing the first 100 columns of example matrix $\mathbf{P}$ defined in eqn. (A-11) and (A-13). The same process was repeated for each cases' signal until having an example matrix of 936 columns (the number of examples, n_e). This matrix was built such that the first 600 examples correspond to inrush examples (desired output=0), and the other 336 examples correspond to faults (desired output=1). Since the FFNN has only one output, matrix $\mathbf{D}$ became a horizontal vector of 936 elements, the first 600 entries are 0's and the other 336 are 1's. In practice, a target tolerance of 0.1 was used, meaning that the network was trained to produce a response of 0.9 or greater to represent one class and 0.1 or less to represent the other class. This is necessary because the nature of the nonlinear squashing function is such that it can never assume the precise values of 0.0 or 1.0.

3.3. Training Process

Once the training matrices $\mathbf{P}$ and $\mathbf{D}$ were defined, the back-propagation algorithm was applied to the problem. The MATLAB Neural Network Toolbox [24] was used for such a purpose. Function TRAINBP was used with sigmoid transfer functions and a learning rate variable between 0.01 and 0.1. The admissible error (sum of square errors in an epoch) was 0.1, for which it was necessary to present the training matrices between 200 and 600,000 times, depending on the network size (number of units) and the learning rate employed.

4. IMPROVING THE NETWORK

4.1. Network Testing and Pruning

As mentioned in section 2.2 it is desired that the network be applied in a digital relay implementation. This implies a compromise between speed and accuracy. As is well known, the FFNN classification time (the required time to produce an output given an input) depends on the number of units in the network, so it is very important to have the lowest number of units but without jeopardizing the quality of the classification. Figure 5 shows the architectures tested for this purpose.

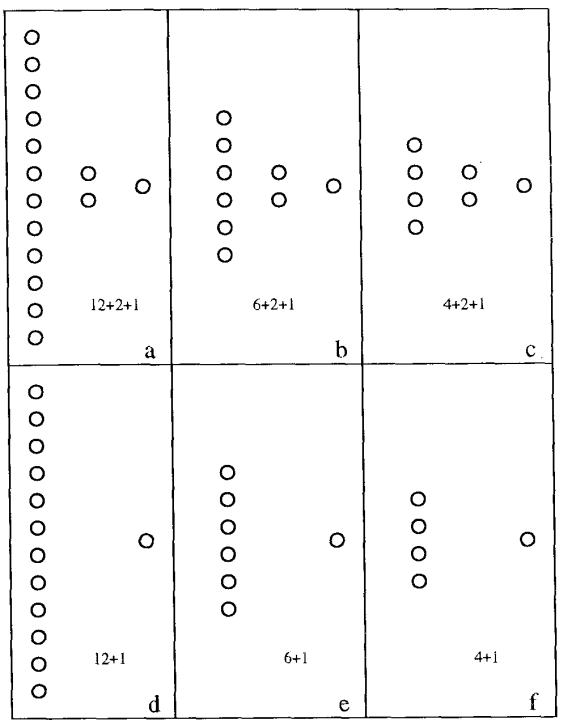

Figure 5. Tested architectures.

The training process was first tried in a 3-layer (2-hidden), 12+2+1 FFNN as shown in Figure 5a. The result was tested with another set of examples (different than the set used for training) which gave successful results. To achieve the error limit with this network, only about 200 epochs were necessary, which encouraged the authors to try a smaller net. In fact, by a process of *pruning* the network, the architectures shown in Figure 5 were trained and tested, until having inadmissible results for the 2-layer (1 hidden), 4+1 FFNN of Figure 5f. By inadmissible results the authors mean it was practically impossible to reach the low error established as a criterion to stop training.

4.2. Hard Limiter Units

Once the networks with sigmoid units were trained and tested with good results (see Figure 6a), the units transfer functions were changed to hard-limiters (see Appendix 1). This change increases the NN computation speed considerably since it takes less computation time to implement a hard limiter (unit step) than a sigmoid (see equations (A-6) and (A-7)). In fact, a hard-limiter is equivalent to an IF statement. The implicit risk in this change is that in certain cases the approximation of the sigmoid by a unit step could produce an inadmissible error. With hard limiters, the output of one of the units is obtained by comparing the unit's input:

$$h = \sum_{k=1}^{L_m} w_{ik}^{(m)} x_k^{(m)} \qquad (2)$$

with the unit's bias b_i.

5. RESULTS

Figures 6 and 7 show the results obtained after testing the networks with non-malicious testing examples (i.e. examples which are different from the ones used for training, but have the same form).

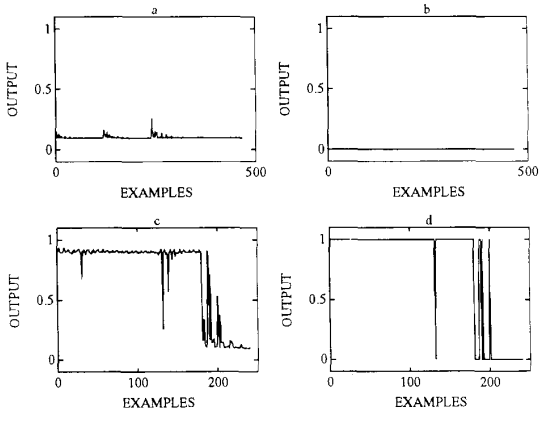

Figure 6. Results of testing a trained 6+2+1 FFNN: a) Sigmoid units, inrush examples, b) HL units, inrush examples, c) sigmoid units, fault examples, d) HL units, fault examples.

Two testing matrices (similar to matrix **P**) were formed: one with only inrush cases, containing examples of inrush currents different than the ones used for training, and the other with fault examples plus and an artificially created inrush-like signal, at the end of the matrix. This las set of examples can be considered *malicious*, since the fault examples were mixed with a set of examples obtained from a wave shape similar to the inrush current, but created by means of a mathematical equation.

Note from Figures 6 and 7 the network responds in a very adequate way, performing the discrimination function between inrush and fault currents correctly for most cases. In fact, for the 6+2+1 network, the classification was 100% correct for the inrush examples and 97.1% correct for the fault plus inrush artificial examples. The percentage of correct classification for the 6+1 network was 99.6% for the inrush examples and 94.6% for the fault plus inrush artificial examples. This percentages are measured by counting example by example (every sample in time), so it does not mean the network fails. The miss-operation of an eventual relay using the algorithm, can be avoided by using an integrator immediately after the relay output or, alternatively, after the inrush detector. This scheme has been used in analog and digital relays to insure correct operation.

Table 1 displays some of the important numbers related to the networks shown in Figure 5. Networks were classified as having either *good* or *poor* performance (by good performance the authors mean that the network could be trained to achieve the error goals and responded adequately to the test). Note from the table that the network with good performance and the minimum number of units was the 2-layer 6+1 network shown in Figure 5e. This implies that, taking into account that the weights and biases are fixed (can be previously loaded in the system's memory), a digital relay implementation (with 12 samples per cycle), would have to be capable of 78 scalar multiplications, 71 additions and 7 comparisons (IF statements) in order to recognize the inrush current. Therefore the relay should be able to execute all these operations and the protective function in a twelfth of a cycle, which is 1.4 msec. This is reasonable for modern microprocessor based systems.

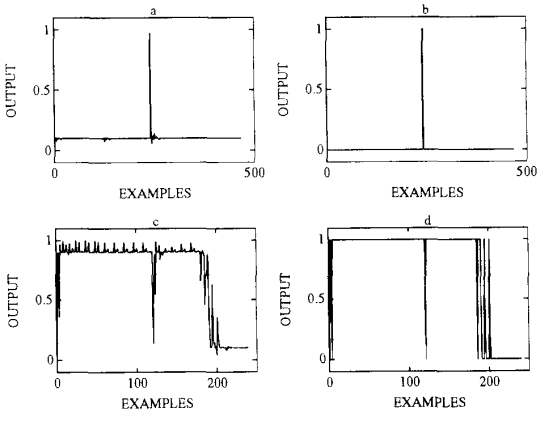

Figure 7. Results of testing a trained 6+1 FFNN: a) Sigmoid units, inrush examples, b) HL units, inrush examples, c) sigmoid units, fault examples, d) HL units, fault examples.

TABLE 1

Net. #	# of layers	# of units	# of mult.	# of addit.	# of IF	Performance
1	3	12+2+1	170	155	15	good
2	2	12+1	156	143	13	good
3	3	6+2+1	86	77	9	good
4	3	4+2+1	58	51	7	poor
5	2	6+1	78	71	7	good
6	2	4+1	52	47	5	poor

6. DISCUSSION

6.1. Quality of Generalization

Quality of generalization is a critical point in any neural network application. In this case the question that arises has to do with the example signals. Both sets of inrush examples, the training and the testing examples, were measured for the same kind of transformers. This could produce inconclusive results, since it is impractical to train the network with inrush examples measured on the specific transformer where the network is going to be applied. However, as mentioned before, the network responded well when an artificially created inrush-like signal was presented to it, which experimentally indicates that the NN generalizes based on the inrush current shape, more than on the transformer design. As a preliminary effort the training does not need to be so specific, but it would be a good procedure to use the method proposed here in a relay to protect transformers with similar characteristics to the transformer whose inrush was used to train the network.

It should be noted that signal examples representing the differential current obtained in the event of an external fault when one of the current transformers saturates were not used. This kind of examples can be included with the "non-inrush" cases since there is no reason for which the network can not be trained. The authors recognize that the current wave shape in this case is closer to the inrush case.

No consideration was given to the effect due to energization of adjacent transformers, but there is no reason to say that the network can not be trained to take into account this effect. This, of course, implies an increase on the examples set, and will affect the training time. This issue and other practical details, like the possiblity of using variable sampling rate, are left for a mature implementation of the algorithm.

6.2. Time Considerations

With the proposed method the necessary time to detect the inrush current could be longer than the protection algorithm itself (for example in a digital differential relay which calculates the current's magnitude based on the full-cycle Fourier algorithm). This could be a weak point of the NN method, but with the speed of modern microprocessors, it is not a great problem. The faster hardware allows the algorithms for digital relay design focus more on the security aspect of the protection, rather than on minimizing the number of steps of the algorithm. In an eventual practical application, the operation time will be determined also by the hardware employed (output circuits, and other components).

6.3. Physical Meaning of the Weights

One of the problems related to the design of systems based on experimentally trained neural networks is the lack of physical meaning of the network parameters (weights and biases). The NN studied here appears to be a good example of it. The authors have tried to find a relationship between the FFNN parameters and digital filter coefficients given the similarity between eqn. (2) and digital filter equations. The frequency responses of digital filters with deference equation as eqn. (2) have been (and continue being) studied but no pattern has been found.

7. CONCLUSION

It has been shown experimentally that a feed forward neural network can be used to discriminate between magnetizing inrush and fault currents in power transformers. The neural network can be trained using sigmoid units and then implemented with hard limiter units in order to improve the computation speed of the network.

The practical application of the proposed method is dependent on the following:

1. The quality of generalization: inrush and fault examples must be representative of the transformer and system where the relay is going to be installed.

2. The speed of the microprocessor system used to implement the relay.

Even though at present the possible application of neural networks to achieve some protective relay functions is a matter of research, this work exemplifies one of those applications in which neural networks could be considered. Future work will serve to clarify aspects related to the physical meaning of the weights and the applicability of the method.

ACKNOWLEDGEMENTS

The authors wish to acknowledge FUNDAYACUCHO, LASPAU, Simón Bolívar University and Schweitzer Engineering Laboratories for their partial financial support. They also wish to thank Dr. Richard Baker for his help in obtaining inrush measurements in the Power Laboratory, and the Software Toolworks for making available an early version of the MATLAB Neural Network Toolbox software used in this work.

APPENDIX 1

This appendix is mainly intended to serve as a guide for the notation and terminology used in the paper. With reference to Figure 1, the *input to a given unit i*, on layer m is defined as [15,16]:

$$h_i^{(m)} = [\mathbf{w}_i^{(m)}]^T \mathbf{x}^{(m)} = \sum_{k=1}^{L_m} w_{ik}^{(m)} x_k^{(m)} \qquad (A-1)$$

where

$$\mathbf{x}^{(m)} = [\, x_1^{(m)} \quad x_2^{(m)} \quad \ldots \quad x_{L_m}^{(m)} \,]^T \qquad (A-2)$$

is the vector whose elements $x_k^{(m)}$ are the inputs to units in layer m, i.e., *the input vector to layer m*,

$$\mathbf{w}_i^{(m)} = [\, w_{i1}^{(m)} \quad w_{i2}^{(m)} \quad \dots \quad w_{iL_m}^{(m)} \,]^T \qquad (A-3)$$

is the vector whose entries w_{ik} represent the connection weight from input $x_k^{(m)}$ to unit i on layer m, L_m is the number of inputs to this layer, and the superscript ^{T} denotes matrix or vector transposition.

The vectors $\mathbf{w}_i^{(m)}$ can be grouped in a single matrix $\mathbf{W}^{(m)}$, as follows:

$$\mathbf{W}^{(m)} = \begin{bmatrix} [\mathbf{w}_1^{(m)}]^T \\ [\mathbf{w}_2^{(m)}]^T \\ [\mathbf{w}_3^{(m)}]^T \\ \vdots \\ [\mathbf{w}_{L_m}^{(m)}]^T \end{bmatrix} \qquad (A-4)$$

The output of unit i on layer m is defined as:

$$v_i^{(m)} = f_i(h_i - b_i) \qquad (A-5)$$

where $f(\bullet)$ is a non-linear function, and b_i is the unit bias. Depending on the application, this function can take different forms. In this work only two of this forms are used, the sigmoid function, defined as:

$$f(h-b) = \frac{1}{1+e^{-(h-b)/2}} \qquad (A-6)$$

and the hard-limiter function (unit step, or Heaviside function) u_{step}:

$$f(h-b) = u_{step}(h-b) \qquad (A-7)$$

see Figure (A-1) for a graphic description of both functions.

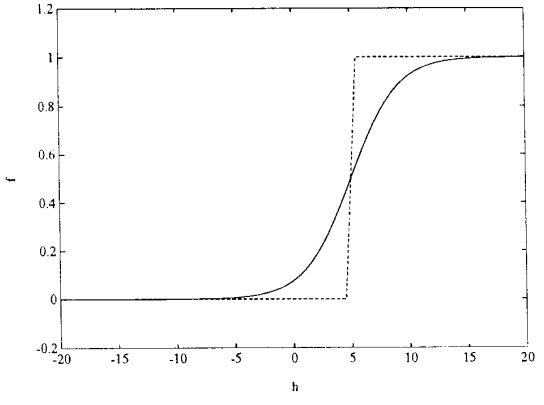

Figure A-1. Sigmoid and hard limiter functions plotted for b=5

These equations can be grouped in a matrix equation as follows:

$$\mathbf{v}^{(m)} = \mathbf{f}(\mathbf{W}^{(m)}\mathbf{x}^{(m)} - \mathbf{b}^{(m)}) \qquad (A-8)$$

and

$$\mathbf{x}^{(m+1)} = \mathbf{v}^{(m)} \qquad (A-9)$$

where $\mathbf{v}^{(m)}$ is the vector representing the outputs of layer m units. In this paper $\mathbf{x}^{(1)} = \mathbf{v}^{(0)} = \mathbf{x}$, is the network's input.

$\mathbf{f}(\bullet)$ is a non-linear vector function, with elements defined as in eqns. (A-6) and (A-7); and

$$\mathbf{b}^{(m)} = [\, b_1^{(m)} \quad b_2^{(m)} \quad \dots \quad b_{L_m}^{(m)} \,]^T \qquad (A-10)$$

is a vector which contains layer m units' biases.

Thus, an M-layer-n_i-input FFNN has an output equal to equation (A-8) evaluated at $m = M$, i.e. there exists an output vector $\mathbf{v}^{(M)}$ for each input vector $\mathbf{x}$.

APPENDIX 2

Basically, back-propagation consists of the presentation of a set of examples $\mathbf{p}_j$, $(j = 1, 2, \dots, n_e)$ and the desired outputs $\mathbf{d}_j$. The weights and biases are updated for each j, according to a rule (back-propagation[13,14,15, 16]) which is minimizes the square distance between the predicted and the desired outputs. The set of examples and desired outputs can be grouped in two matrices, $\mathbf{P}$ and $\mathbf{D}$ (training matrices), as follows:

$$\mathbf{P} = [\, \mathbf{p}_1 \quad \mathbf{p}_2 \quad \dots \quad \mathbf{p}_{n_e} \,] \qquad (A-11)$$

$$\mathbf{D} = [\, \mathbf{d}_1 \quad \mathbf{d}_2 \quad \dots \quad \mathbf{d}_{n_e} \,] \qquad (A-12)$$

or, in expanded notation,

$$\mathbf{P} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & \cdots & p_{1n_e} \\ p_{21} & p_{22} & p_{23} & \cdots & p_{2n_e} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ p_{n_i1} & p_{n_i2} & p_{n_i3} & \cdots & p_{n_in_e} \end{bmatrix} \qquad (A-13)$$

$$\mathbf{D} = \begin{bmatrix} d_{11} & d_{12} & d_{13} & \cdots & d_{1n_e} \\ d_{21} & d_{22} & d_{23} & \cdots & d_{2n_e} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ d_{n_o1} & d_{n_o2} & d_{n_o3} & \cdots & d_{n_on_e} \end{bmatrix} \qquad (A-14)$$

where n_i, n_o and n_e are the number of inputs, outputs and examples, respectively.

REFERENCES

[1] IEEE Std. C37.91-1985. *IEEE Guide for Protective Relay Applications to Power Transformers.*

[2] Sonnemann, W. K., Wagner, C. L., Rockefeller, G. D., "Magnetizing Inrush Phenomena in Transformer Banks," *AIEE Transactions*, Vol. 77, part. III, pp 884-892. Oct. 1958.

[3] Blackburn, J. L. *Protective Relaying. Principles and Applications*, Elect. Eng. and Electronics Series, Marcel Dekker, INC, NY, 1987.

[4] *Applied Protective Relaying Reference Book*, Westinghouse Electric Corporation, Relay Instrument Division, 1976.

[5] Hayward, C. D., " Harmonic-Current-Restrained Relays for Transformer Differential Protection," *AIEE Transactions*, Vol. 60, 1941, pp. 377-382.

[6] Sharp, R. L., Glassburn, W. E., "A Transformer Differential Relay with Second Harmonic Restraint," *AIEE Transactions*, Vol. 77, Dec. 1958, pp. 913-918.

[7] Phadke, A. and Thorp, J. *Computer Relaying for Power Systems*, John Wiley & Sons, NY, 1988.

[8] Liu, P., Malik, O. P., Chen, D., Hope, G. and Guo, Y. "Improved Operation of Differential Protection of Power Transformers for Internal Faults". IEEE PES Winter Meeting, NY, Jan. 1992. Paper No 92WM206-3 PWRD.

[9] Phadke, A. and Thorp J. "A New Computer Based, Flux Restrained, Current Differential Relay for Power Transformer Protection," *IEEE Transactions in Power Apparatus and Systems*, Vol. PAS-102, No 11, Nov. 1983, pp 3624-3629.

[10] Schweitzer, E.O., Larson, R.R. and Flechsig A.J. "An Efficient Inrush Detection Algorithm for Digital Computer Relay Protection of Transformers," IEEE PES Summer Meeting, Mexico City, Paper No 77 510-1, July 1977.

[11] Sidhu T. S. and Sachdev M. S. "On Line Identification of Magnetizing Inrush and Internal Faults in Three-phase Transformers" IEEE PES Winter Meeting, NY, Jan. 1992. Paper No 92WM205-5PWRD.

[12] Sidhu, T. S., Sachdev, M., Wood, H. C., Nagpal, M., "Design, Implementation and Testing of a Microprocessor Based High-Speed Relay for Detecting Transformer Winding Faults," *IEEE Transactions on Power Delivery*, Vol. 7, No 1, Jan. 1992, pp. 108-117.

[13] Rumelhart, D.E., Hinton, G.E., and Williams, R., "Learning Internal Representations by Error Propagation", in D. Rumelhart and J. McClelland (Eds.), *Parallel Distributed Processing*, (pp. 318-362). Cambridge, MA: MIT Press, 1986.

[14] Werbos, P. J., *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*, Doctoral Dissertation, Applied Math, Harvard University, Nov. 1974.

[15] Widrow, B., Lehr, M. A., "30 Years of Adaptive Neural Networks: Perceptron, Madaline, and Backpropagation," *Proceedings of the IEEE*, Vol. 78, No. 9, Sept. 1990, pp. 1415-1442.

[16] Hertz, J., Krogh, A., Palmer, R., *Introduction to the Theory of Neural Computation*, Addison-Wesley, 1991.

[17] Sachdev, M. S. (Co-ordinator), "Microprocessor Relays and Protection Systems," *IEEE Tutorial Course Text*, Publication No. 88EH0269-1-PWR, 1988.

[18] Mori, H., Itou, K., Uematsu, H., Tsuzuki, S. "An Artificial Neural-Net Based Method for Predicting Power System Voltage Harmonics," *IEEE Trans. on Power Delivery*, Vol. 7, No 1, Jan. 1992, pp. 402- 409.

[19] Ebron, S., Lubkeman, D. and White, M. "A Neural Network Approach to the Detection of Incipient Faults on Power Distribution Feeders." *IEEE Transactions on Power Delivery*, Vol. 5, No 2, April 1990.

[20] Sultan, A., Swift, G., Fedirchuk, D. "Detection of High Impedance Arcing Faults Using a Multi-Layer Perceptron". IEEE PES Winter Meeting, NY, Jan, 1992. Paper No 92 WM 207-1 PWRD.

[21] Stigant, S. A. and Franklin, A. C. *The J & P Transformer Book*. John Wiley & Sons, NY, 1973.

[22] Becia, W. W.*PC METER: A Data Acquisition & Control System for the Power System Laboratory*, M. Sc. Thesis, Washington State University, 1992.

[23] Wall, R. W., *PC-EMTP User's Manual*, Power Products ucts Inc., July 18, 1989.

[24] Demuth, H. and Beale M. *Neural Network Toolbox User's Guide*, Jan., 1992.

BIOGRAPHIES

Luis G. Pérez (M) was born in Valle de la Pascua, Venezuela in 1957. He received his Electrical Engineer degree from Simón Bolivar University (USB) in 1979 and his M. Sc. degree in EE from the Central University of Venezuela in 1982. He joined the Dept. of Energy Conversion and Delivery at USB in 1983, and has held a tenured position in this department since 1986. He was the Chairman between 1986 and 1989. Mr. Pérez has taught in and researched the areas of Power System Protection and High Voltage Substation Design since 1983, and has participated in several projects related to these areas in Venezuela since 1979. He has been a graduate student working toward his doctoral degree in the School of EE and Computer Science at WSU since January 1991.

Alfred J. Flechsig (SM) joined the Electrical Engineering Department at Washington State University in 1959. He was born in Tacoma, Washington in 1935. He received his BSEE and MSEE degrees from WSU in 1957 and 1959, and his Ph. D. in EE from Louisiana State University in 1970. His primary interests are instruction and research in Electric Power Systems. His research activities have included analysis of Housing Systems, Power Systems, Protective Relaying, Residential Energy Conservation and Solar Assisted Heat Optimization. He is a registered professional Engineer in Washington and Louisiana and a Professor in the School of EE and Computer Science at WSU.

Jack L. Meador (M) was born in Amarillo Texas in October, 1956. He received the BS, MS, and Ph.D degrees in EE from Washington State University in 1979, 1981, and 1987 respectively while he worked as a computer systems manager and an independent consultant. He is currently an assistant professor in the School of EE and Computer Science at Washington State University where he teaches and conducts research in neural networks and VLSI design. His current research interests include pulse coded neural network theory and implementation and neural network applications to mixed-signal integrated circuit test. He is the current chair of the IEEE Circuits and Systems Society Technical Committee on Neural Systems and Applications and is also a member of the International Neural Network Society and the ACM.

Zoran Obradović (M) received the B.S. degree in Applied Mathematics, Information and Computer Sciences in 1985; the M.S. degree in Mathematics and Computer Science in 1987, both from the University of Belgrade; and the Ph.D. degree in Computer Science from The Pennsylvania State University in 1991. He was a systems programmer at the Department for Computer Design and Development at the Institute "Boris Kidric," Vinca-Belgrade, from 1984 to 1986, and has been a research scientist at the Mathematical Institute of the Serbian Academy of Sciences and Arts, Belgrade, since then. At present, he is an Assistant Professor in the School of EE and Computer Science at Washington State University. His research interests include neural networks, parallel processing, machine learning, pattern recognition, and algorithms design and analysis.

Discussion

M.A.Rahman, B.Jeyasurya (Memorial University of Newfoundland, St.John's, NF, Canada): The authors must be congratulated for presenting the use of feed forward neural network to discriminate between inrush and fault currents in a transformer. We have a few comments and questions on this paper.

1. The training examples for inrush were obtained by energizing a transformer and the examples for fault cases were obtained by simulation. It is not clear whether the training examples for fault had significant harmonic components. A spectral analysis of the fault signals would have been useful.

2. Did the authors consider obtaining the training data for the fault cases on the laboratory transformer by applying a fault, with suitable back-up protection for the transformer? It is likely that the converged values of the weights and biases may be different if all the training examples for both the inrush and fault cases were obtained from the test transformer.

3. As the authors have pointed out, the computational requirements for the FFNN implementation is very large even for a single phase transformer. This is very significant considering that it is possible to implement the major protective functions for a three phase transformer in a single digital signal processing board [A]. Due to the wide range of the weights, the proposed method will require a Math coprocessor for accurate computations within one sampling interval.

4. The proposed architecture of the FFNN (Figure 5e) requires a data window of half a cycle. If the input to the network contains some of fundamental signal samples and some of faulted signal samples, as in the instant of fault occurrence, or in the case of an inrush followed by internal fault, can one depend on the decision of the neural network?

We commend the authors for presenting one of the first applications of artificial neural networks for digital relays.

Reference

[A] I.Hermanto, Y.V.V.S.Murty, M.A.Rahman, "A Stand-Alone Digital Protective Relay for Power Transformers", IEEE Transactions on Power Delivery, Vol. 6, No. 1, January 1991, pp. 85-95.

L. G. Pérez, A. J. Flechsig, J. L. Meador, Z. Obradović *(School of Electrical Engineering and Computer Science, W. S. U., Pullman, WA)*: The authors wish to thank the discussers for their interest in the paper and for their valuable comments. These comments will be raised in the same order as they were written in the discussion.

1. The fault examples employed were based on computer simulations. The instantaneous values of the fault examples were scaled properly to the per-unit base of the transformer used to obtain the inrush examples. In some cases, the fault examples were obtained as the response of a simple R-L circuit contaminated purposely with up to 40 % of second harmonic component. This was done with the intention of training the network such that it was capable of distinguishing between fault currents with a high second harmonic component and inrush currents.

2. The discussers are correct when they say that the resulting weights and biases could be different if the fault training examples were obtained as they suggest. We considered that possibility, however, we thought that for this preliminary result the computer simulated fault examples were sufficient to give us the security that the method would work. We believe that in a practical implementation the training must be done using a combination of field measured and computer created fault examples.

3. We agree with the discussers on this point. However, we anticipate that with the present developments in hardware for DSP applications, the requirements of speed and accuracy for complete transformer protection can be achieved succesfully, including the inrush detection as suggested in the paper.

4. All architectures shown in Figure 5 were trained and tested using a one-cycle data window and a sample rate of twelve samples per window. In that figure only the layers containing processing units (neurons) are shown (the layer corresponding to the inputs is not shown in Figure 5). Small unfilled circles represent processing units, as indicated in Figure 1. In a more complete representation, the network of Figure 5e looks like the one shown in Figure C-1.

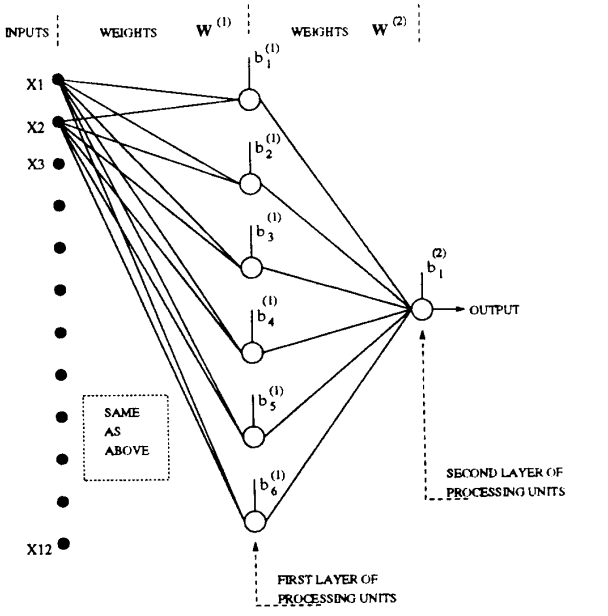

Figure C-1. *Detailed scheme of the network shown in Figure 5e.*

Manuscript received March 29, 1993.