Motor condition monitoring and fault classification using artificial intelligence in an IEC 61850-based system

C. Kriger[1], R. Tzoneva[2]. P. Barendse[3], A. Apostolov[4]

Cape Peninsula University of Technology, South Africa

krigerc@cput.ac.za[1]; tzonevar@cput.ac.za[2]; paul.barendse@uct.ac.za[3];
alex.apostolov@omicronenergy.com[4];

Introduction

A secure and reliable electric power system is what most end-users expect from their supplier of electricity. In South Africa, scheduled power outages known as load-shedding has been experienced recently as the demand exceeded the generation capacity. This phenomenon is not unique to South Africa and in various parts of the developed world this is an ongoing challenge with the increasing demand of the load versus the generation capacity resulting in power systems that are being operated in a constrained state (Adewole, 2016). The continuous monitoring and detection of any unwanted condition and the mitigation thereof before any mal-operation occurs is the desired response to such a situation.

The power system is comprised of the generation, transmission and distribution sections each containing many different multi-functional protection, automation and control devices known as Intelligent Electronic Devices (IEDs) that have advanced technologically to being able to self-monitor and communicate relevant information to other devices within the system as well as to the control centers or Supervisory Control and Data Acquisition (SCADA) systems. The ability of the IEDs to monitor themselves and the current power system conditions is known as situational awareness. (Apostolov, 2011). The technological advances in the communication, protection and control abilities of the IEDs have however resulted in a much more complex power grid in the pursuit of attaining a smarter grid. This complexity increases the likelihood of errors and reduces reliability of the system in general. The additional challenge of having all these various devices (usually from different vendors) with advanced communication functionality also introduces the issue of interoperability where these devices have to communicate with other devices within the power system network.

The IEC 61850 standard was promulgated to address the issue of interoperability among devices from different vendors being able to communicate and share information among themselves. The IEC 61850 standard has since its initial inception been extended with a second edition and various technical reports as well as to extend its application to other domains. An area of significant interest is that of condition monitoring (IEC 61850-90 Technical Report) as it offers substantial economic and operational benefits in its application within the power system. A condition monitoring system in general has to provide a continuous monitoring function that allows for the early detection of unwanted conditions and mitigate the onset of equipment failure. One of the objectives for this work is therefore the development of a motor condition monitoring system that is applied within an IEC 61850 communication environment.

The condition monitoring functionality is implemented firstly by the detection and classification of four different types of induction motor faults using an Artificial Neural Network (ANN) which is trained and validated using real-time motor data within the MATLAB software environment. The induction motor data is applied as inputs to the ANN using the IEC 61850 analogue Generic Object-Oriented Substation Event (GOOSE) message. The ANN is then implemented onto a Real-Time Automation Controller hardware device which communicates with a motor protection IED using a binary GOOSE message to trip the breaker upon detection of a fault condition within the motor.

The content of the paper is outlined as follows: A background to induction motors and the most common faults are described in the first part together with the laboratory set up for the acquisition of the real-time induction motor data. The second part presents the Neural Network training and validation implemented within the MATLAB software environment with a selected portion of the experimental results. Part three presents the complete system implementation with the Real-Time Digital Simulator (RTDS) and RTAC with the analogue GOOSE message results and binary GOOSE to the motor protection IED. Part four presents the real-time implementation of the neural network on the Real-Time Automation Controller (RTAC) hardware platform and Part five presents the conclusion to this work.

1. Induction Motor Faults and Data Acquisition Lab Test Bench Setup

Although the induction motor is very reliable it is almost inevitable that due its extensive use it could quite likely experience degradation leading to faults and ultimately motor failure. There is therefore a need for the early detection of the onset of these possible faults before it results in the induction motor failing which might lead to processes being halted resulting in financial loss. The following section presents and discusses some of the most common motor faults and focuses on the ones identified in this study. A more detailed survey and discussion of the condition monitoring and protection methods applied to medium voltage induction motors is presented by Zhang, et al., 2011.

1.1. Common motor faults

It is very difficult to find real-time fault data required in a knowledge-based system especially for a developing fault which is not easy for the operator to detect or identify. General motor faults together with possible symptoms are presented in Table 1. The common fault parameter for almost all of the faults is the Current. The current however does not change where there is unbalance or misalignment of the shaft. The faults that pertain to this work include the inter-turn fault (highlighted in yellow), bowed rotor (in red) bearing fault (in green) and the broken rotor bar (highlighted in blue). It is worthwhile noting that for all these faults the common parameter that is affected is the vibration. The next section presents some of the typical faults and methods used to identify them.

Table 1: Motor faults and the associated symptoms

Machine type: Electric motor	Symptom or parameter change												
Examples of faults	Current	Voltage	Resistance	Partial discharge	Power	Torque	Speed	Vibration	Temperature	Coast down time	Axial flux	Oil debris	Cooling gas
Rotor windings	•				•	•	•	•	•		•		•
Stator windings	•							•	•		•		•
Eccentric rotor	•							•			•		
Brush(es) fault	•	•			•	•			•				
Bearing damage	•					•		•	•	•		•	
Insulation deterioration	•	•	•	•									•
Loss of input power phase	•	•						•			•		
Unbalance								•					
Misalignment								•					

• Indicates symptom may occur or parameter may change if fault occurs.

1.1. Bearing Faults

1.1.1. Bearing Faults

Bearings are one of the most important components found in induction motors and more prone to failure than any other component. It is recorded that 40% of all faults in large machines and 90% in small machines are bearing faults (Riera-Guasp, et al.,2015, Supangat, 2008). The bearing structure and components are shown in Figure 1. Bearing faults can be due to race damage inner or outer race), rolling element or cage damage. Bearing faults could also be due to wear where the lifespan is exceeded, the bearing is overloaded, it is incorrectly assembled, due to manufacturer error or there is insufficient lubrication (Ludeca, 2011).

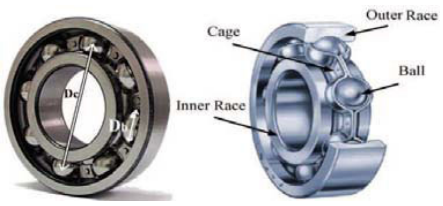

Figure 1: Bearing structure and components (Riera-Guasp, et al., 2015)

Vibration monitoring is one of the most commonly used monitoring techniques used. In order to measure vibration however, additional sensors have to be mounted onto the motor. The vibration frequency can be determined when detecting bearing faults.

$$f_{OD} = \frac{n}{2} f_{rm}\left(1 - \frac{BD}{PD}\cos\phi\right) \tag{1.1}$$

where f_{OD} is the outer race defect frequency, f_{rm} is the rotor speed in revolutions per minute, n is the number of balls, BD is the Ball Diameter, PD is the Pitch Diameter.

The inner race defect frequency is given in Equation (1.2):

$$f_{ID} = \frac{n}{2} f_{rm}\left(1 + \frac{BD}{PD}\cos\phi\right) \tag{1.2}$$

where f_{ID} is the inner race defect frequency.

The ball defective frequency is shown in Equation (3.3):

$$f_{BD} = \frac{PD}{2BD} f_{rm}\left(1 - \left(\frac{BD}{PD}\right)^2 \cos^2\phi\right) \tag{1.3}$$

where f_{BD} is the ball defective frequency.

The induction motor cage defect frequency is specified in Equation (1.4) as:

$$f_{CD} = \frac{1}{2} f_{rm}\left(1 - \frac{BD}{PD}\cos\phi\right) \tag{1.4}$$

where f_{CD} is the cage defect frequency.

1.1.2. Air gap eccentricities

Air gap eccentricity faults can occur as a result of bearing wear or an unbalanced magnetic pull and lead to a non-uniform air gap as shown in Figure 2 resulting in the rotor geometrical centre being displaced with respect to the stator geometrical centre (Dlamini, 2014). These eccentricities can be either static or dynamic. For static eccentricities the air gap has a fixed minimal position and the minimal position rotates with the dynamic eccentricity. Usually a 10% eccentricity is allowed due to manufacturing and assembly or commissioning of the motor (Capolino, 2015).

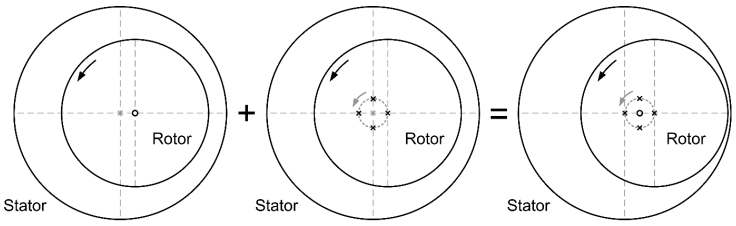

Figure 2: Rotor eccentricities: static (left), dynamic (middle) and mixed (right) Dlamini, 2014)

A combination of static and dynamic eccentricities is referred to as mixed eccentricity shown to the right in Figure 2. The biggest risk with this type of fault is the possibility that there may be mechanical contact between the rotor and the stator which might lead to the induction motor failing. Methods used in the detection of eccentricities usually include Machine Current Signature Analysis (MCSA) that is a frequency-based method that identifies sidebands in the stator line-current spectrum that is related to the eccentricity. There have been shown to be detection problems using the MCSA method given the relationship between the pole-pair number and the rotor slot number. Other more successful detection methods include using the terminal voltage spectrum when the induction motor is switched off (Samonig and Wolbank, 2017).

The fault signature for detecting air gap eccentricities is defined by the following frequency in Equation (1.5):

$$f_{ecc} = f_{rm}\left(1 \pm m\frac{1-s}{p}\right) \quad m = 1,2,3\ldots \tag{1.5}$$

where f_{ecc} is the frequency of eccentricity, s is the slip, m is a constant and p is the number of pole pairs.

1.1.3. Broken rotor bars

Broken rotor bars are responsible for more than 5% of induction motor failures. There are various methods used in the detection of broken rotor bars in induction motors including the hypothesis that the resistance of an induction motor will increase when a rotor bar is broken (Duan, 2005). The rotor resistance is modified as derived from Equation (1.6):

$$r_r = \left(\frac{R_b}{N/3} \frac{1}{k^2} \right) \tag{1.6}$$

where r_r is the rotor resistance referred to stator, N is the total number of rotor bars, n is the number of broken rotor bars, k is the transformation ratio, and R_b is the rotor resistance per bar.

The change in the rotor resistance is given by Equation (1.7):

$$\Delta r = \left(\frac{3n}{N - 3n} \right) r_r \tag{1.7}$$

where r_r is the rotor resistance referred to stator, N is the total number of rotor bars, and n is the number of broken rotor bars (Patel and Bhalja, 2015).

Various reasons are proposed as to why the rotor bars break including magnetic stresses, thermal stresses due to abnormal operating conditions, inadequate casting, manufacturing defects, lack of maintenance, etc. A broken rotor bar results in rotor asymmetries which degrade the motor's overall performance and shortens its lifetime (Toliyat, 2012). Methods used to detect broken rotor bars include Machine Current Signature Analysis (MCSA) which is a method based on the frequency, and also the Zero Crossing Time (ZCT) method (Duan, 2005).

The asymmetrical movement of the rotor results in a backward rotating field at the slip frequency with respect to the forward rotating rotor. This prevents current flow and the magnetic field around this area will not exist. The backward rotating field induces additional frequency harmonics seen as sidebands to the fundamental harmonic component at frequencies given by Equation (1.8):

$$f_{brb} = (1 \pm ms) f \quad m = 1,2,3\ldots \tag{1.8}$$

where f_{brb} is the broken rotor bar frequency, s is the slip, m is a constant and f is the supply frequency.

1.1.4. Shorted stator winding faults in induction motors

The stator winding insulation is very important in induction motors as it prevents electrical short circuits between the motor windings. Stator turn-to-turn faults occur as a result of excessive heating, surge in supply, high humidity, vibration, contamination, amongst others. Stator windings faults represent approximately 38% of reported induction motor faults (as given in Figure 3).

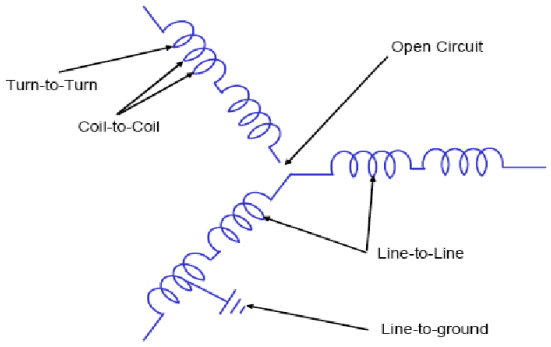

Figure 3: Stator faults (Dlamini, 2014)

Different types of stator faults include:

- Turn-to-turn short circuit faults;
- Coil-to-coil short circuit faults;
- Open circuit faults;
- Phase-to-phase short circuit faults; and
- Phase-to-ground short circuit faults (Duan, 2005).

For the turn-to-turn fault the motor may appear to be operating normally at reduced efficiency. Methods for the detection of stator faults are MCSA where the harmonic components are examined including the sidebands in the stator line-current spectrum. On the onset of a stator fault characteristic fault frequency components in the stator current are produced as defined in Equation (1.9)

$$f_{st1} = f\left(m\frac{(1-s)}{p} \pm v \right),\ m = 1,2,3\ldots \tag{1.9}$$

where f_{st1} is the shorted turn fault frequency 1, f is the supply frequency, m is a constant, s is the slip, p is the number of pole pairs, and v is the order of the stator time harmonics.

Vibration analysis is also an option that is used when the possibility exists to install the sensors without compromising the normal operation of the motor. The shorted stator windings cause an unbalance force at twice the line frequency and multiples with modulation at the rotor speed.

1.2. Lab-scale test setup

A Machine Fault Simulator (MFS) is used to introduce common types of faults which occur in an induction motor during operation. This induction motor simulator is located within the laboratories of the Electrical Engineering Department at the University of Cape Town. For this particular motor fault simulator there are seven different types of 250W, 3-phase AC (Alternating Current) induction motors with the exact same specifications but having different types of faults. For this motor simulator the following faults can be introduced depending on the associated equipment used:

- rotor unbalance;
- rotor misalignment;
- bowed rotor;
- faulted bearings;
- broken rotor bars;
- stator winding faults; and
- voltage unbalance and single phasing (Dlamini, 2014).

For this research work only four types of motor fault conditions are examined together with the healthy condition as summarized in Table 2. The various motors used with each its own unique characteristic are those with faulted bearings, bowed rotor, broken rotor bar, and stator winding (inter-turn) faults. It is important to note that for the healthy condition (no fault) and the inter-turn fault the same motor is used with the option to select either a two or four winding inter-turn fault.

Table 2: The different types of motor faults that are detected

Number	Type of motor faults to be detected
1	Bearing Fault
2	Bowed Rotor
3	Broken Rotor Bar
4	Inter-turn Fault
5	Healthy Condition

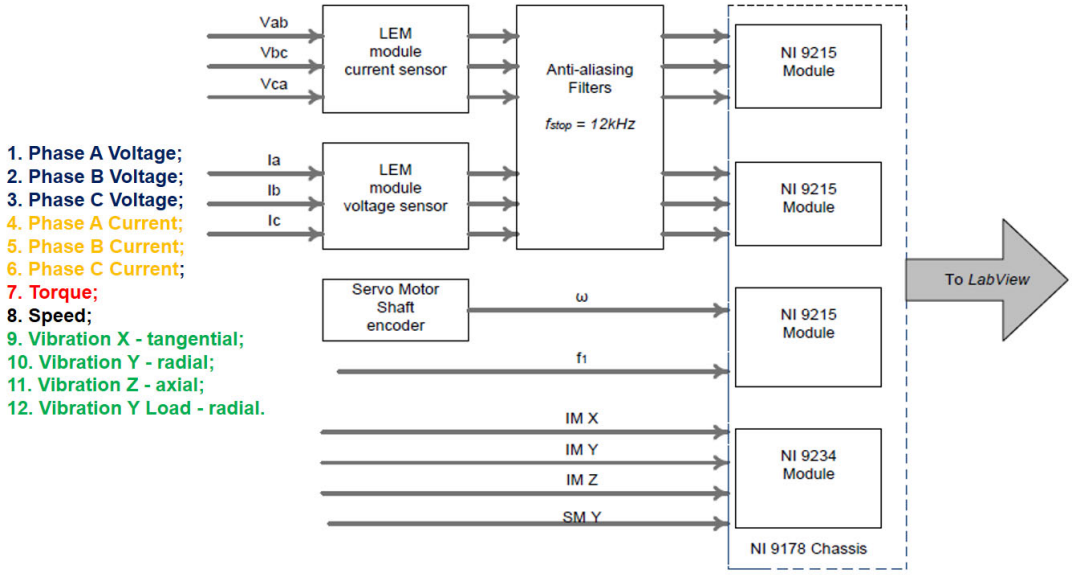

Figure 4: The motor fault simulator DAQ system measured variables (Dlamini, 2014)

There are twelve signals that are connected to various sensors within the data acquisition system as shown in Table 3. These include the 3-phase voltages, 3-phase currents, speed from the load drive, frequency from the Variable Speed Drive (VSD), vibration signals from 3 Accelerometers on the motor drive and one Accelerometer on the load drive (Dlamini et al., 2014). The signal conditioning modules condition the signals to the appropriate voltage levels for the DAQ system that connects to the computer with the LabVIEW software. The current and voltage sensors are connected to the Current LEM and Voltage LEM signal conditioning modules.

Table 3: The input sensory data from the sensors

Number of sensors	Input data from sensors
1	Phase A voltage
2	Phase B voltage
3	Phase C voltage
4	Phase A current
5	Phase B current
6	Phase C current
7	Speed
8	Frequency
9	Vibration X – Tangential
10	Vibration Y – Radial
11	Vibration Z – Axial
12	Vibration Y– Radial (Load)

This section presented some of the typical faults occurring within induction motors and the laboratory setup for the acquisition of the real-time motor data. The following section presents the application of the sensory data from the lab-scale real-time motor data to the input of a neural network in order to detect and classify the correct fault condition. The ANN has 12 inputs from the sensors of the DAQ and 5 outputs which classifies the four specified faults as well as the healthy state.

2. Neural Network implementation within the MATLAB software environment

The techniques for detecting any fault timeously have ranged from time-based methods to frequency based signal processing techniques, and finally to computational intelligent techniques such as neural networks, fuzzy logic and support vector machines for the fault classification and detection. One of the precursors to application of these methods is however the identification of the correct features present in

the monitored data. This is known as feature extraction using methods such as fractals (Maragos and Potamianos, 1999), Fast Fourier Transforms (FFT), wavelets and kurtosis (Marwala and Vilakazi, 2007) (Dlamini, et al., 2014).

This section presents the implementation of the Artificial Neural Network (ANN) with MATLAB software environment. The initial work in the area of neural networks started with the feedforward NN proposed by McCulloch and Pitts (1943). The network topology generally refers to the number of input neurons, output neurons and the number of hidden layer neurons, together with the bias neurons for the respective layers. The main objective of this section is to determine the structure (topology) of the neural network for classification of the four induction motor faults. The design requirements as calculated for the network structure are that it:

- should have twelve inputs;
- should have five outputs (including the no-fault condition);
- should be able to correctly classify four different induction faults including the no-fault (healthy) condition;
- should be able to correctly classify the different induction motor faults, without the network structure being overly complex;
- should implement supervised training with the batch training method where all inputs are presented to the network at the same time;
- should use two gradient descent training algorithms with the structure being implemented whose solution converged fastest and was the least complex;
- should be implemented in real-time on a hardware platform after the training phase has completed using static weight and bias values.

These design requirements are determined by the number of sensors on the induction motor set, faults to be classified as presented in the first part of this paper, real-time implementation as well as other requirements specified for this research project.

2.1. Neural Network topology

Very broadly speaking NN topology could be divided into either feedforward networks where the input is propagated through the network structure in a left to right (forward) direction to the output, or recurrent networks where the output of the network is fed back to the input. There is no specific method to follow when designing the neural network topology. Although authors such as Haykin (1999) propose rules of thumb when selecting the number of input neurons for example, the topology design invariably changes based on a particular application. Careful consideration has to be paid to the number and type of inputs selected as these may influence the ability of the NN to be able to provide the correct classification (decision) boundary between different classes (motor faults).

The single layer perceptron is generally applied to simple pattern recognition tasks (Rosenblatt, 1958). The structure of the artificial neuron which is also known as the perceptron with only one input layer and one output layer together with a bias and an activation function. The disadvantage of the single layer perceptron is that it is unable to solve a problem that is not linearly separable.

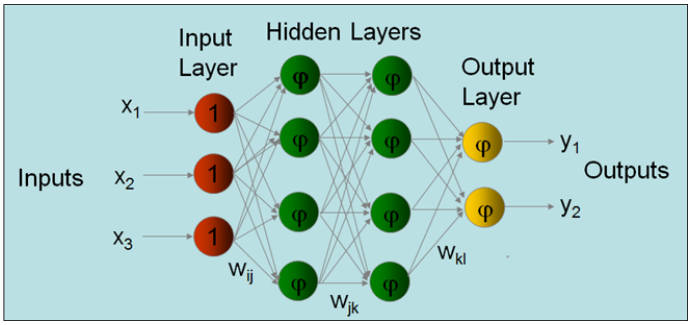

Figure 5: The multilayer perceptron network with two hidden layers (3-4-4-2)

The MultiLayer Perceptron (MLP) has the same features as the single layer perceptron, except that in addition to the input and output layers it has an added layer which is known as the hidden layer. Multilayer perceptron networks can have one or more hidden layers with each layer having its activation function (Figure 5) and any number of neurons in each layer. The main difference between the single layer and multilayer perceptron networks are that the MLP can define a decision boundary even for problems that are not linearly separable. Figure 5 is a 3-4-4-2 network, indicating the number of neurons found in the input, two hidden and output layers respectively.

There are 12 inputs and 5 outputs as presented in the preceding section. In order to determine the number of neurons to place in the hidden layer(s) without the network being overly complex, the number of hidden layers and neurons are increased which effectively means an increase in network complexity. The method employed to determine the optimum number of hidden layers and neurons is to start with the initial structure having only one hidden layer with one hidden neuron and then gradually to increase this number until the network converged to a solution.

2.2. Neural Network training algorithm

Humans learn things by observing others or having a teacher. In the same way NNs learn or are trained either by supervised learning or unsupervised learning. This is accomplished by adjusting the weights if there is a difference between the desired network output and the current network output. For supervised learning the inputs and expected outputs are provided to the network during the training phase. The network weights and biases are changed in order to reduce the performance error during the training phase and obtain the required output.

The Scaled Conjugate Gradient (SCG) algorithm is one of the training algorithms applied in this research work was proposed by Moller (1993) as a way to avoid the line-search procedure of conventional conjugate gradient algorithms.

The Levenberg-Marquardt (LM) algorithm is the second training algorithm used and is designed to specifically minimize a sum-of-squares error.

The Resilient backpropagation (RP) learning rule is the third training algorithm where the weight values are updated in such a way that the total mean squared error of the network is minimized over the entire training set. The backpropagation algorithm has two phases:
Forward pass – In this stage the input is propagated from left to right through the activation functions of all the layers to the output of the network.
Backward pass – In this stage the network error is calculated and used to adjust the network weights. The error is propagated backwards from the output through the network layer by layer. This is iteratively computed with the calculation of the local gradient of each neuron.

Although not explicitly addressed in this paper, issues of generalization and stopping criteria are considered in this work.

2.3. Neural Network Training Results

The inputs and outputs are defined and the training phase starts. The input data is randomly divided into three groups; namely, training (70%), validation (15%) and test (15%) data sets. The stopping criteria for the training phase are minimization of squared error, minimum gradient, increasing validation errors and maximum number of epochs. If any of these conditions exist, the training phase stops and the weights are saved. Figure 6 shows the MLP in MATLAB having 12 input neurons, one hidden layer with 3 neurons, and an output layer with 5 neurons (12-3-5) with the network being trained using the Scaled Conjugate Gradient (SCG) algorithm.

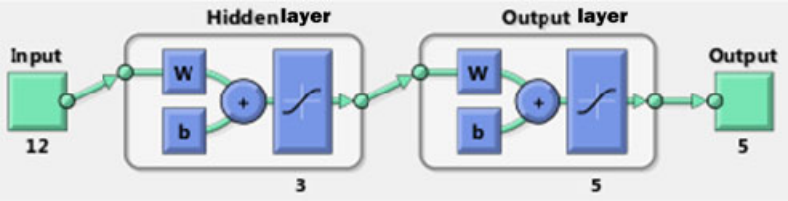

Figure 6: The NN structure (12-3-5) with 12 inputs, 3 hidden neurons and 5 outputs

The NN training software algorithm is executed twelve times with the SAME network structure as per Figure 6 with varying weights and biases for each execution of the algorithm. The classification results for the 12-3-5 NN structure is shown in Table 4 with the highest percentage correct classification being 89.5% (in green) and the lowest percentage correct classification being 29.9% (in red). These results are graphically plotted in Figure 7 with the highest percentage correct classification in green and the lowest percentage correct classification in red. The Scaled Conjugate Gradient (SCG) training algorithm is used.

Table 4: Results of NN Training with 1 Hidden Layer and 3 Hidden Neurons (12-3-5)

Number of hidden layers	Number of hidden neurons	Percentage Correct Classification	Percentage Incorrect Classification	Training Algorithm
1	3	82.8	17.2	SCG
1	3	84.5	15.5	SCG
1	3	84.4	15.6	SCG
1	3	81.6	18.4	SCG
1	3	76.1	23.9	SCG
1	3	59.3	40.7	SCG
1	3	54.2	45.8	SCG
1	3	89.5	10.5	SCG
1	3	81.4	18.6	SCG
1	3	67.5	32.5	SCG
1	3	86.9	13.1	SCG
1	3	29.9	70.1	SCG

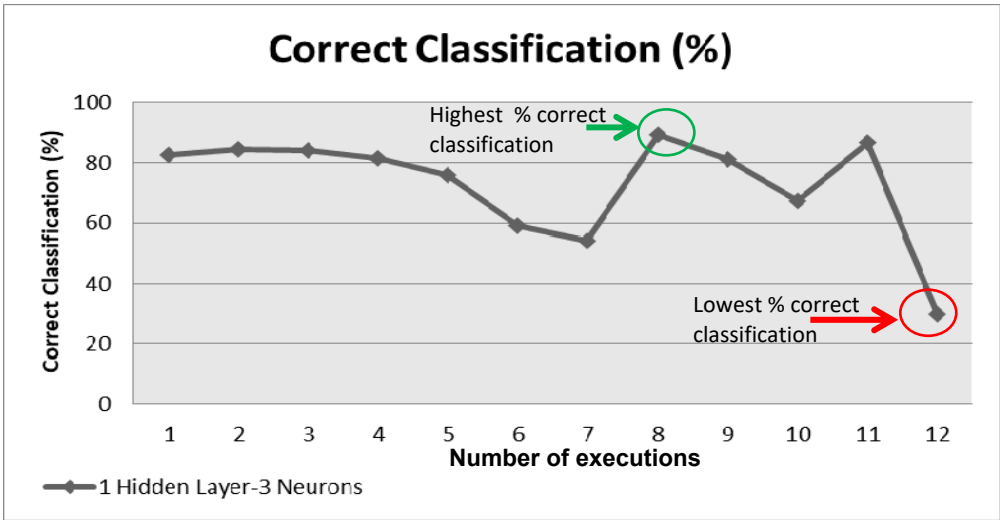

Figure 7: The 12-3-5 NN structure depicting the percentage correct classification results

The Confusion Matrix is shown in Figure 8 for the 12-3-5 NN structure where the overall correct classification of 89.5% and a value of 10.5% for misclassification are obtained. The bearing fault has a correct classification value of 86.8%, the bowed rotor fault has a correct classification value of 87.4%, the broken rotor bar has a correct classification value of 89.2%, the inter-turn fault is correctly classified with a

value of 89% and the no fault condition is correctly classified with a value of 94.9% which is the highest correctly classified condition.

The Receiver Operating Characteristic curves for two out of the twelve algorithm executions are shown in Figure 9. The figure to the left of Figure 9 shows a much better classification than the one on the right with only a 29.9% correct classification. Although the result of cases of correct percentage classification is higher than that with the NN structure having one hidden neuron, it is still not able to fully distinguish between all classes for the stated problem.

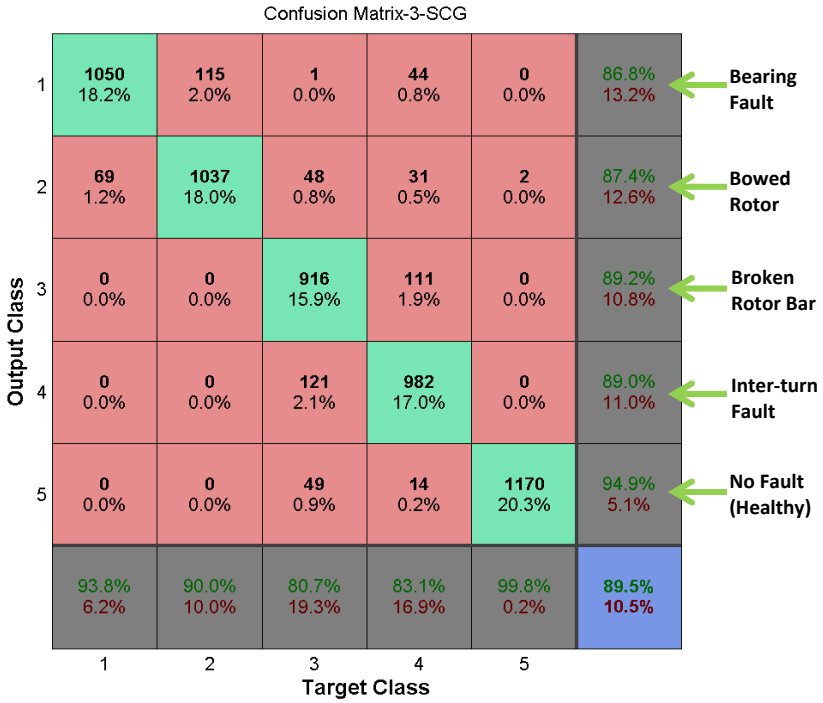

Figure 8: The 12-3-5 NN structure depicting Confusion Matrix with the percentage correct classification results

With an overall value of 10.5% of incorrect classifications, the 3-hidden neuron network structure although adequate in solving the classification problem, there is a possibility that certain classes might be misclassified. As part of the design strategy to determine the optimal number of hidden neurons for classification of the induction motor faults, the number of hidden neurons is now increased to 5 neurons.

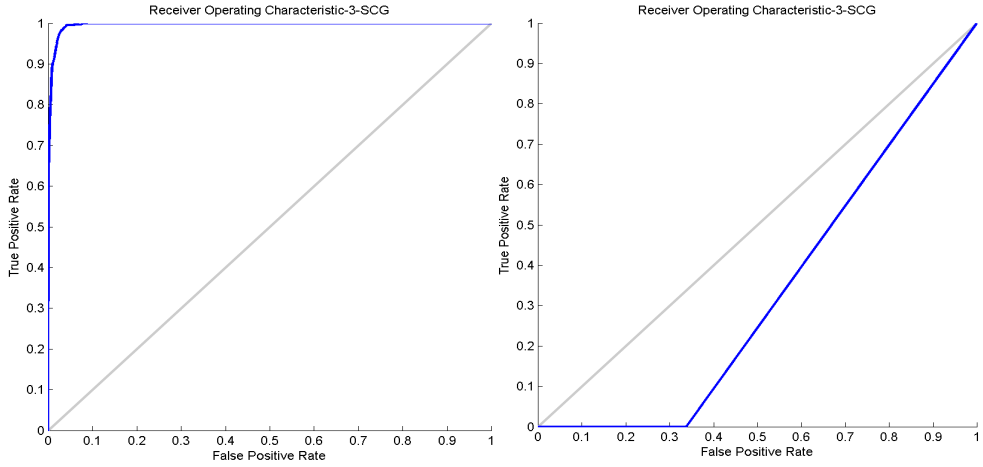

Figure 9: The Receiver Operating Characteristic (ROC) curves for the 12-3-5 NN structure

The results for the 12-5-5 NN structure for all three training algorithms are summarized in Table 5. The highest correct percentage classification is highlighted in green. The LM algorithm obtained a highest correct percentage classification value of 99.3%, the RP algorithm obtained a highest correct percentage classification of 96.6% and the Scaled Conjugate Gradient obtained a highest correct classification percentage of 97.2%.

Table 5: Results of NN Training with 1 Hidden Layer and 5 Hidden Neurons (12-5-5)

Number of hidden layers	Number of hidden neurons	Percentage Correct Classification	Training Time Duration	Training Algorithm	Number of Epochs	Stopping criteria
1	5	98.9	00:02:25	LM	55	VAL
1	5	83	00:13:56	LM	315	VAL
1	5	98.4	00:03:15	LM	73	VAL
1	5	98.9	00:02:13	LM	47	VAL
1	5	99.1	00:02:37	LM	58	VAL
1	5	79.3	00:03:39	LM	80	VAL
1	5	98.9	00:03:28	LM	76	VAL
1	5	63.7	00:04:34	LM	102	VAL
1	5	98.8	00:04:26	LM	99	VAL
1	5	98.9	00:05:51	LM	130	VAL
1	5	96.4	00:02:20	LM	50	VAL
1	5	99.3	00:02:20	LM	54	VAL
1	5	88.9	00:04:30	RP	1508	VAL
1	5	94.5	00:08:58	RP	3000	MAX
1	5	95.8	00:08:58	RP	3000	MAX
1	5	86.9	00:05:22	RP	1824	VAL
1	5	90.1	00:08:50	RP	3000	MAX
1	5	96.3	00:08:00	RP	2645	VAL
1	5	78.5	00:04:33	RP	1436	VAL
1	5	96.6	00:08:43	RP	2880	VAL
1	5	96	00:05:21	RP	1823	VAL
1	5	86.8	00:03:02	RP	947	VAL
1	5	95.9	00:06:34	RP	2262	VAL
1	5	81.5	00:04:50	RP	1538	VAL
1	5	94.8	00:03:33	SCG	583	VAL
1	5	95.8	00:03:07	SCG	503	VAL
1	5	96.7	00:02:32	SCG	393	VAL
1	5	36.9	00:02:23	SCG	391	VAL
1	5	87.1	00:03:44	SCG	596	VAL
1	5	85.8	00:01:56	SCG	297	VAL
1	5	85.9	00:01:48	SCG	300	VAL
1	5	78.4	00:01:53	SCG	290	VAL
1	5	97.2	00:03:10	SCG	499	VAL
1	5	87.5	00:02:43	SCG	459	VAL
1	5	96.6	00:03:41	SCG	586	VAL
1	5	97	00:04:45	SCG	773	VAL

The LM algorithm once again had the highest training time duration (highlighted in yellow) of 13:56 (minutes and seconds) in 315 epochs. The SCG algorithm took 4:45 (minutes and seconds) to reach convergence in 773 epochs. The RP algorithm again reached the maximum number of epochs (3000) epochs in a time of 8:58 (minutes and seconds). Once more the RP algorithm required more time to converge. The algorithms that took longer to converge did not necessarily produce the best classification result as is indicated in Table 4.10. For all cases presented, the LM and SCG algorithms stopped due to the validation error increasing after six consecutive epochs. The RP algorithm algorithms on the other hand also had cases where the maximum number of epochs was reached.

In summary, the RP algorithm once again required more time (in terms of the number of epochs required) to converge than any of the other two training algorithms but the LM algorithm had the highest correct percentage classification although it took a longer time (in minutes and seconds) to converge in fewer

epochs than was the case for any of the other two training algorithms. For all the algorithms other than for the instances where the RP reached the maximum number of iterations, training stopped as the error on the validation set continued to increase.

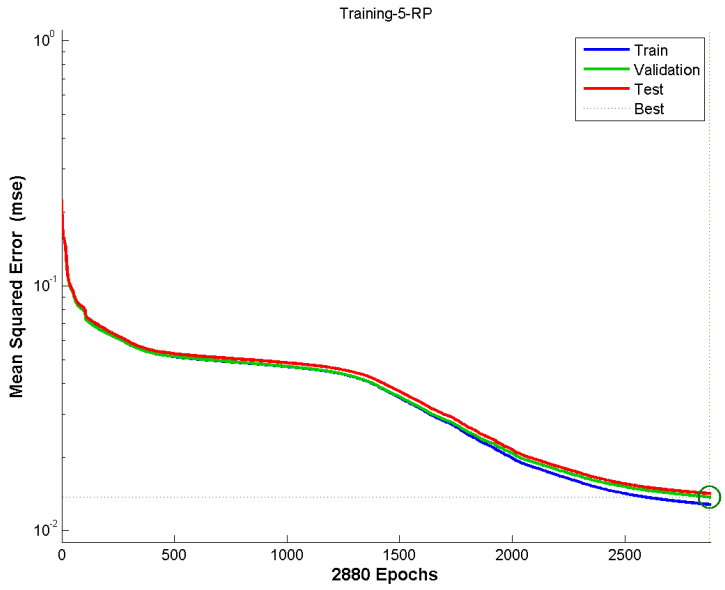

Figure 10: The training result for the RP algorithm for the 12-5-5 NN structure

The training result for the highest percentage correct classification for the RP algorithm is presented in Figure 10. This shows that the training stopped at epoch 2880. The error on the training data set is slightly lower than that on the validation and test sets. The training however stops not on the minimization of the error to zero, but when the validation error continues to increase for six consecutive epochs.

This section presented the MATLAB neural network implementation with a brief discussion of the topology with the feed-forward MLP being chosen where the number of hidden neurons are increased gradually until a suitable classification is obtained. Three training algorithms are implemented: Levenberg-Marquardt, Scaled Conjugate Gradient and Resilient backpropagation. Finally, a selection of results is presented with 3 neurons in the hidden layer which is then increased to 5 neurons. The next section describes the implementation of the NN on a hardware platform.

3. Real-time System implementation

The hardware configuration for the system used for the real-time implementation case study is shown in Figure 11. The Real-Time Digital Simulator (RTDS) rack contains among others a network interface card for connecting to the Local Area Network (LAN). This card is known as the GTNET-GSE (Generic Substation Event) card (blue box in Figure 11) and is used for the publication of the GOOSE message onto the LAN.

The configuration of the GOOSE message is performed within the RSCAD-Draft software within the GTNET block indicated by the purple box in Figure 11. As stated previously this GOOSE message contains twelve analogue data values. The analogue GOOSE message indicated by the yellow arrow is published by the RTDS GTNET (blue box) and the subscribing device is the SEL-3555 RTAC (red box).

The RTAC (red box in Figure 11) is a real-time automation and gateway controller device which supports various communication protocols, e.g IEC 6113-3 (for Programmable Logic Controllers – PLCs) and also IEC 61850 communications, among others. The neural network algorithm is developed within the AcSELerator RTAC software environment using the IEC 6113-3 language and is discussed in Section 4.

The induction motor status is defined in Table 2 in Section 1; i.e. Healthy (no fault), bearing fault, bowed rotor, broken rotor bar, and inter-turn fault. A binary GOOSE message (red arrow on the right) is published from the RTAC device indicating whether a fault is present or not.

The SEL-710 motor protection IED (green box in Figure 11) subscribes to the binary GOOSE message from the RTAC device and publishes a binary GOOSE trip signal (blue feedback arrow) to the RTDS. Within the RSCAD-Runtime software environment a circuit breaker opens when a motor fault is present as indicated within the binary GOOSE from the motor protection relay.

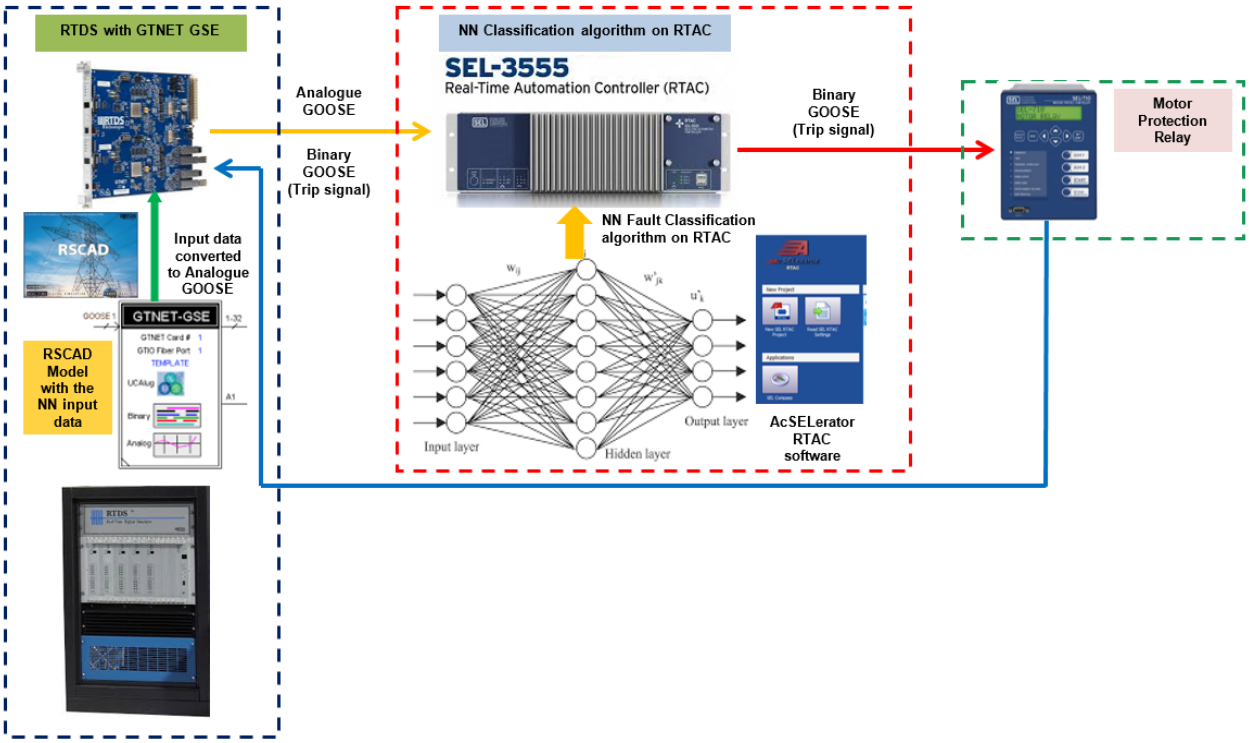

Figure 11: The configuration for the hardware implementation of the NN induction motor fault classification system

The structure and validation of the dataset of the Generic Object-Oriented Substation Event (GOOSE) message is presented in Kriger, et al., 2013. The next section contains the configuration of the Analogue GOOSE message.

3.1. Analogue GOOSE configuration and verification

The four steps that are taken to implement the configuration of the analogue GOOSE message are shown in the flow chart in Figure 12 where the RSCAD software environment is used for the configuration and the GOOSE Inspector software environment is used for the verification of the published Analogue GOOSE message.

Step 1 with the RSCAD Draft implementation of the 12 analogue data inputs is shown in Figure 13 with the 12 inputs being applied using lookup tables and up/down switches.

Step 2 for the configuration of the GTNET-GSE Analogue GOOSE message using the SCL Editor is shown in Figure 14. It is important to note that the version of the editor only supported the use of GGIO Logical Nodes which are used for General Input and Output functionality (shown in the red oval in Figure 14).

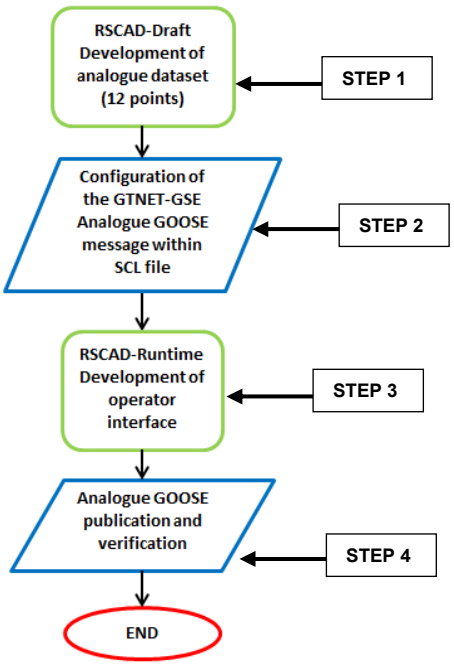

Figure 12: The flowchart detailing the steps for the analogue GOOSE development

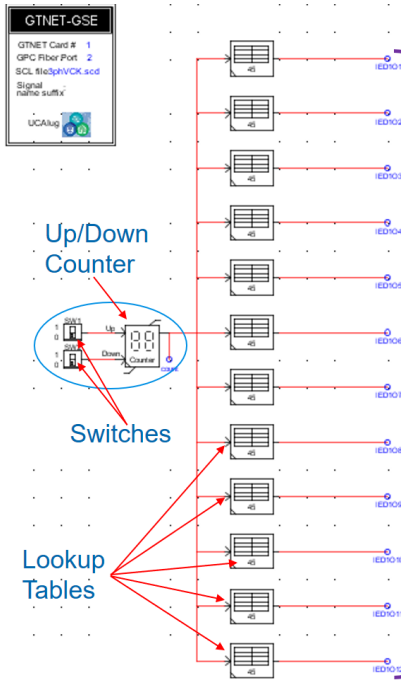

Figure 13: Step 1: RSCAD Draft

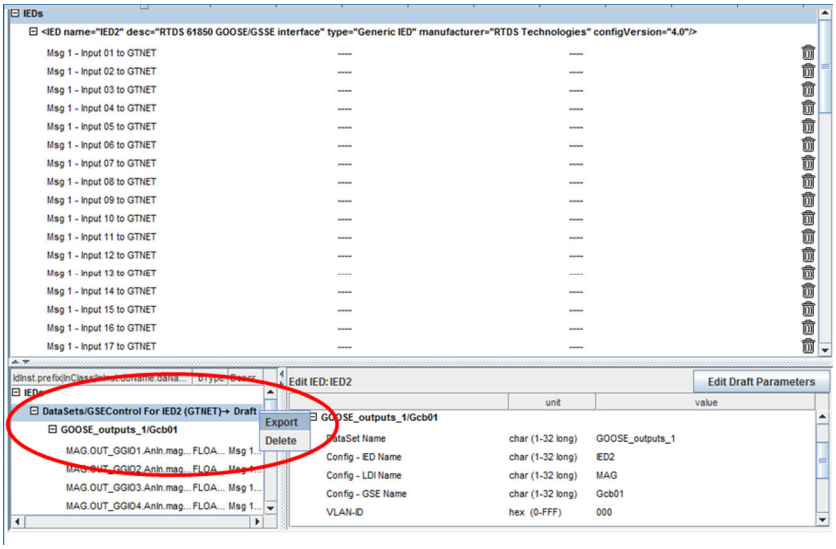

Figure 14: Step 2: SCL Editor

Step 3 where the Runtime development environment is developed is shown in Figure 15 with the 12 Analogue values as described in Table 3 with the up/down switches selecting a new set of input values.

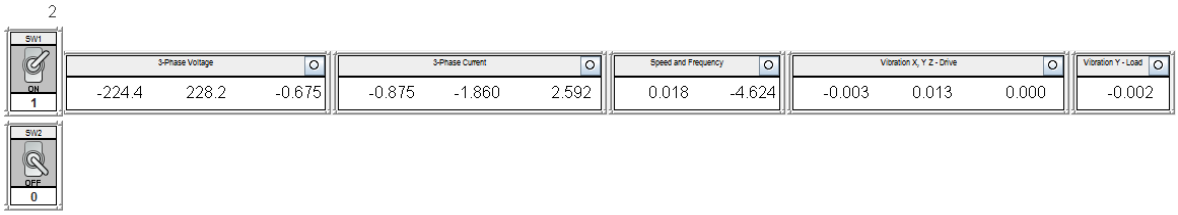

Figure 15: Step 3: RSCAD Runtime

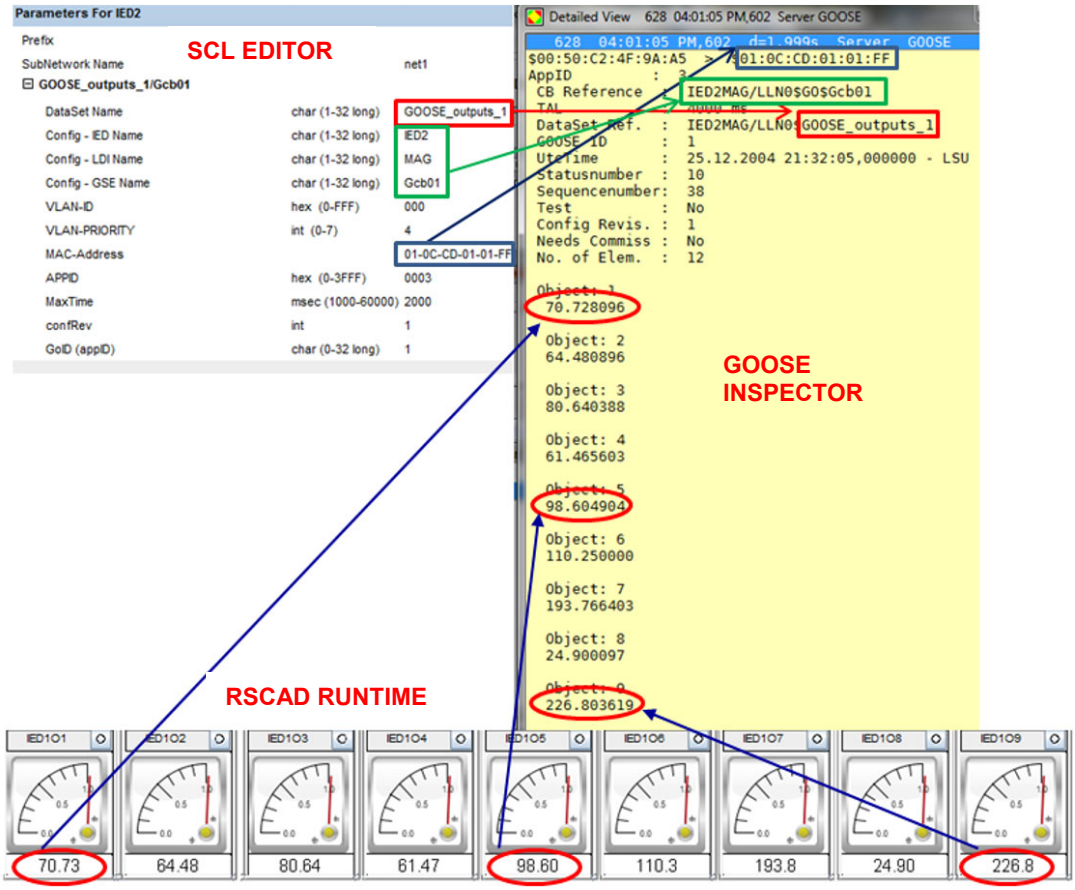

Figure 16: Step 4: RSCAD Draft

Step 4 is where the published Analogue GOOSE message is verified. This is shown in Figure 16 with the data from the SCL Editor (top left) and the data values on the RSCAD Runtime window (bottom) being verified in the published GOOSE message shown in the GOOSE Inspector software (to the right). The green rectangles confirm the GOOSE Control Block Reference, the red rectangles confirm the Dataset Name and the blue rectangles confirm the MAC Address. (IEC 61850-8-1) (Kriger et al., 2013).

3.2. GOOSE message configuration in SEL Architect Software

The Analogue GOOSE message configured using the RSCAD SCL Editor is the publisher and the Real-Time Automation Controller (RTAC) device is the subscribing device. In order for the publisher and subscriber to communicate with each other, the Configured IED Description (CID) files for each device need to be linked using the SEL Architect software. The CID file from the RSCAD SCL Editor is exported as an IED Capability Description (ICD) file and dragged into the upper window of into the SEL Architect Editor as shown in Figure 17. The same is the case for the CID file of the SEL RTAC.

Figure 18 shows the Data objects with the Measured Variable (MV) shown by the green oval at the bottom, being used for the Analogue GOOSE subscription. The GGIO logical node is used (shown by the red oval at the top right) as the publishing device only supports this logical node as indicated earlier.

Figure 19 shows the final step in the configuration process where the publishing device and subscribing device are mapped. With each successful step being indicated with the GREEN LED being lit.

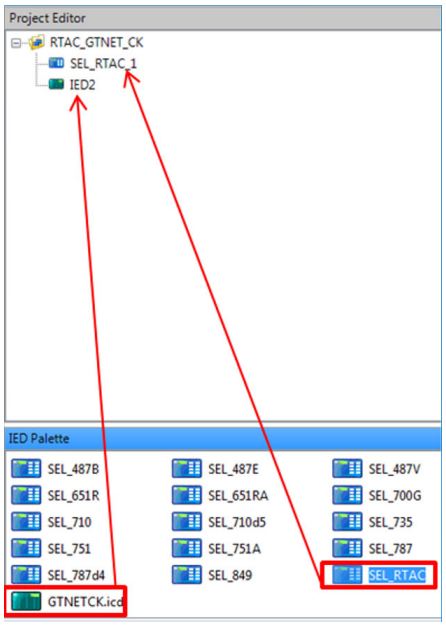

Figure 17: ICD and CID file configuration for the publisher and subscriber

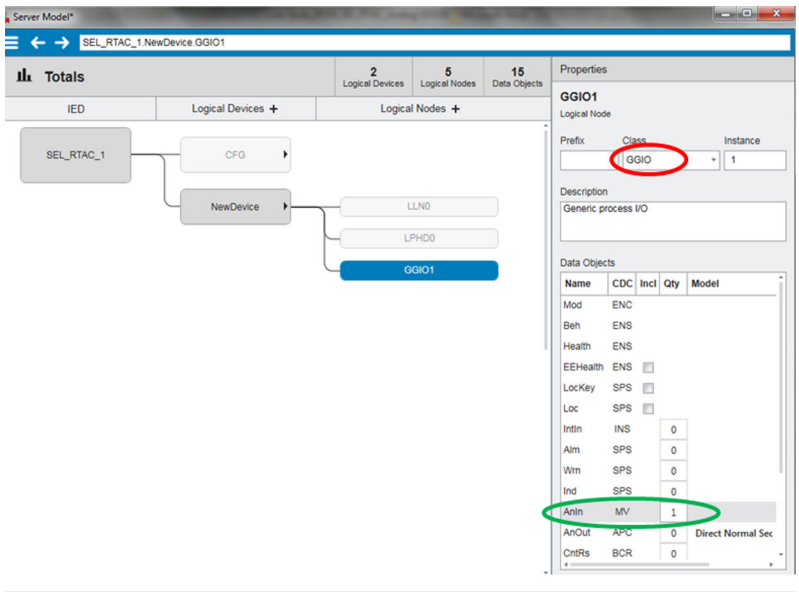

Figure 18: GOOSE configuration of subscriber (RTAC)

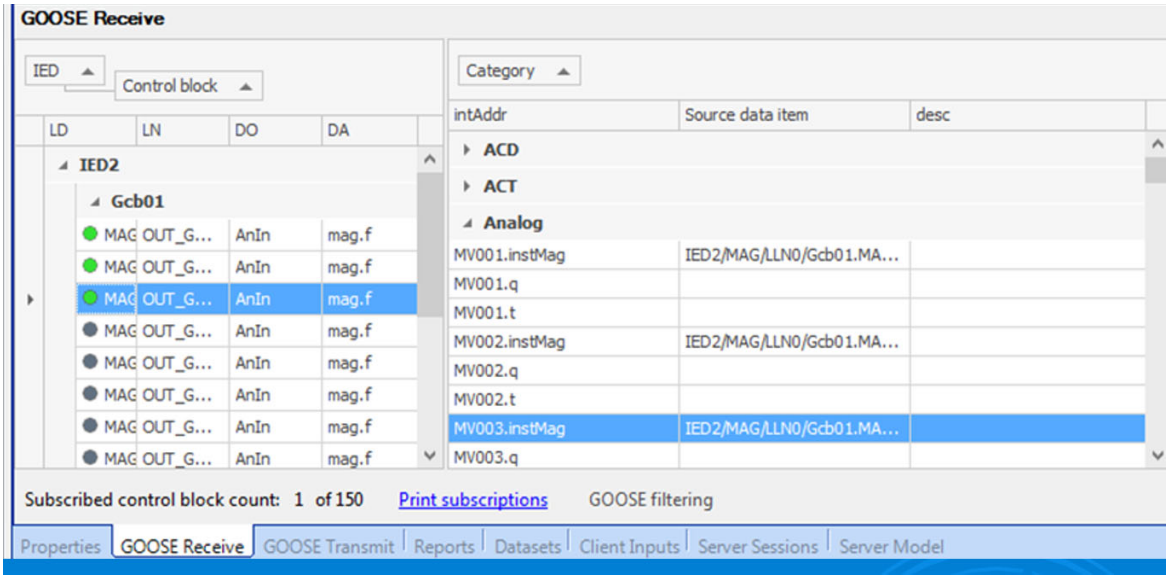

Figure 19: GOOSE configuration – mapping and linking of publisher and subscriber

4. Neural Network implementation on the Hardware Platform

Not all aspects of the hardware implementation of the neural network is described in this section. The mathematical model which is implemented on the Real-Time Automation Controller (RTAC) is shown in Figure 20 for the 12-3-5 NN structure. A very small subset of the software algorithm implemented using the Statement List (STL) component of the IEC 6113-3 standard for Programmable Logic Controllers is

shown in Figure 21 with the top section showing the inputs taken from the Analogue GOOSE message published from the GTNET-GE card within the RTDS.

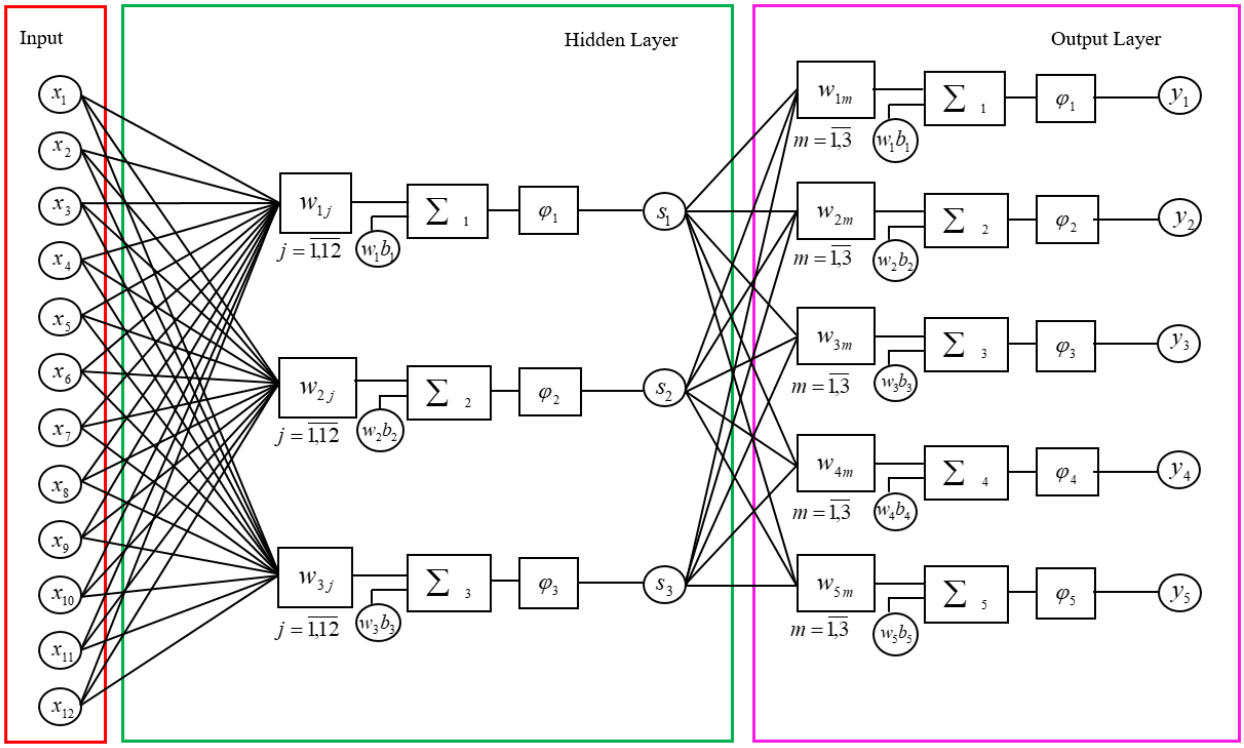

Figure 20: The 12-3-5 NN structure implemented within the RTAC software environment

```
inp[5]:=IED2_MAG_Gcb01.MV005.instMag;
inp[6]:=IED2_MAG_Gcb01.MV006.instMag;
inp[7]:=IED2_MAG_Gcb01.MV007.instMag;
inp[8]:=IED2_MAG_Gcb01.MV008.instMag;
inp[9]:=IED2_MAG_Gcb01.MV009.instMag;
inp[10]:=IED2_MAG_Gcb01.MV010.instMag;
inp[11]:=IED2_MAG_Gcb01.MV011.instMag;
inp[12]:=IED2_MAG_Gcb01.MV012.instMag;

// Product of the 12 Inputs * 10 x Weights
FOR j:=1 TO 10 BY 1 DO
    FOR i:= 1 TO 12 BY 1 DO
    InpWeight[j,i]:=inp[i]*inp_W[j,i];
    END_FOR
END_FOR


//Summing the product of the input and Weight vector
FOR i:=1 TO 10 BY 1 DO
    SumInWeight[i]:=InpWeight[i,1]+InpWeight[i,2]+InpWeight[i,3]+InpWeight[i,4]+InpWeight[i,5]+
    SumInpBias[i]:=SumInWeight[i]+inp_b[i];     // Adding the input bias
END_FOR
```

Figure 21: A subset of the software algorithm for implementation of the 12-3-5 NN structure

The RTAC output shown in Figure 22 showing firstly the detection of a Bearing fault at the top window and a Healthy state (no fault) at the bottom of the window during Runtime execution.

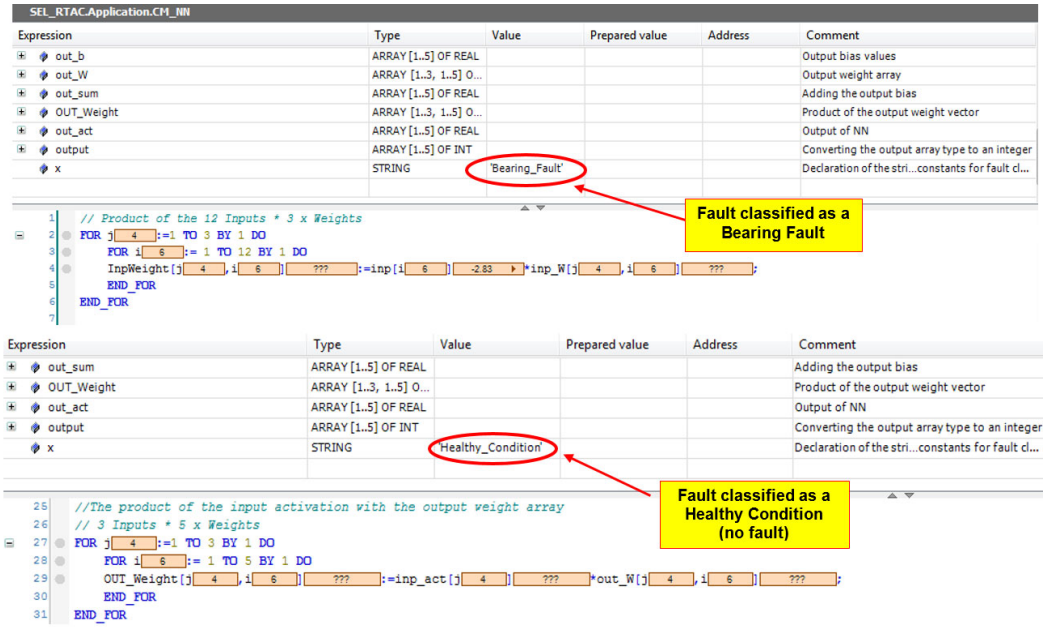

Figure 22: The output window showing the the fault classification for the 12-3-5 NN structure

This section presented the neural network implementation on the RTAC Hardware device within the SEL RTAC software environment. The input to the algorithm is received with the data contained in an Analogue GOOSE message received from the GTNET-GE Publishing device within the RTDS. The binary GOOSE configuration is not covered in this paper. The next section provides the conclusion to this work.

5. Conclusion and Future Work

The paper presented the condition monitoring system implementation using the real-time induction motor data applied as input to an Artificial Neural Network. The training, validation and testing of various neural network topologies in the MATLAB software environment are presented together with three training algorithms. The NN is used to detect and classify four types of induction motor faults including the healthy motor state. The weights of the successfully trained neural networks within the MATLAB software environment are saved and implemented on a real-time hardware device. The input data to the ANN implemented on the Real-Time Automation Controller (RTAC) device, are acquired from an analogue GOOSE message sent from the Real-Time Digital Simulator (RTDS). The ANN algorithm successfully detects and classifies any fault condition existing within the applied input data. A binary GOOSE message communicates with the Motor Protection IED and sends a trip signal for any fault condition. A proof-of-concept lab hardware implementation system is presented which allow for both Analogue and Binary GOOSE messages and allowing for IEC 61850 standard communication between different devices within the system.

Future work include the real-time application of the motor data in the analogue GOOSE message which would ensure a complete real-time implementation of the system. Additional work includes the embedded-based implementation using the IEC 61850.lib library which contain logical nodes from Edition 1 and Edition 2 of the IEC 61850 standard.

6. Acknowledgements

The research work described in the paper is partly funded by the Eskom TESP program under the grant “Intelligent Condition Monitoring of Power System Equipment using the IEC 61850 Communication Standard”.

References

Adewole A.C., 2016. “Voltage Stability Assessment and Wide Area Protection/Control using Synchrophasor measurements”. Doctor Engineering Thesis, Cape Peninsula University of Technology, CPUT.

Apostolov A., 2011. “Condition Monitoring and Protection”, PAC World Magazine Article” September, page 4.

Capolino G., Antonino-Daviu J. A., and Riera-Guasp M., 2015. "Modern Diagnostics Techniques for Electrical Machines, Power Electronics, and Drives," in IEEE Transactions on Industrial Electronics, vol. 62, no. 3, pp. 1738-1745.

Dlamini M., 2014. “Development of an Induction Motor Condition Monitoring Test Rig and Fault Detection Strategies”, Master of Engineering Thesis document, University of Cape Town, UCT.

Dlamini M., Barendse P. S., and Khan A. M., 2014. "Detecting faults in inverter-fed induction motors during startup transient conditions," 2014 IEEE Energy Conversion Congress and Exposition (ECCE), pp. 3131-3138.

Haykin, S. 1999. Neural Networks, A Comprehensive Foundation. Second Edition, Prentice-Hall, Inc. Ontario, Canada.

International Electrotechnical Commission (IEC), IEC 61850-7-4. “*Communication networks and systems in substations* - Basic communication structure for substation and feeder equipment – Compatible logical node classes and data classes”, 2003.

International Electrotechnical Commission (IEC), IEC 61850-8-1. “*Communication networks and systems in substations* - Specific Communication Service Mapping (SCSM) – Mappings to MMS (ISO 9506-1 and ISO 9506-2) and to ISO/IEC 8802-3”, 2003.

Kriger C., Behardien S., Retonda-Modiya J. 2013. “A Detailed Analysis of the GOOSE Message Structure in an IEC 61850 Standard-Based Substation Automation System”, International Journal of Computers Communication and Control, ISSN 1841-9836, pp: 708-721.

Ray, A. 2009 “Application of Signal Processing Techniques for Condition Monitoring of Electrical Equipment” PhD in Electrical Engineering Thesis document. M.S. University of Baroda, Vadodara.

Riera-Guasp M., Antonino-Daviu J. A., and Capolino G., 2015. "Advances in Electrical Machine, Power Electronic, and Drive Condition Monitoring and Fault Detection: State of the Art," in IEEE Transactions on Industrial Electronics, vol. 62, no. 3, pp. 1746-1759.

Rosenblatt, F. 1958. "The perceptron: A probabilistic model for information storage and organization in the brain. Psychological Review, Volume 65, pp. 386-408.

Supangat, R., 2008. “On-line condition monitoring and detection of stator and rotor faults in induction motors”, A thesis submitted for the degree of Doctor of Philosophy, University of Adelaide, Australia.

Zhang P., Y. Du Y., Habetler T. G., and Lu B., 2011. "A Survey of Condition Monitoring and Protection Methods for Medium-Voltage Induction Motors", in IEEE Transactions on Industry Applications, vol. 47, no. 1, pp. 34-46.

Abstract

Induction motors are extensively used in various industry applications in fans, pumps, wind turbines and generators to name but a few applications. Although the induction motor is very reliable it is almost inevitable that due its extensive use it could quite likely experience degradation leading to faults and ultimately motor failure. There is therefore a need for the early detection of the onset of

possible faults before it results in the induction motor failing which might lead to processes being halted which ultimately result in financial loss. The proposed solution to the detection of developing motor faults has to be fast, accurate and able to halt the motor before catastrophic failure.

Various techniques are used in the detection of different types of motor faults where a single method is not always preferred in the detection of multiple motor faults. These methods include signal processing techniques applied in the frequency domain but it has the disadvantage of being applied to static and not real-time dynamic systems. This method although useful cannot be applied in the real-time detection of induction motor faults.

The area of artificial intelligence spans various branches of computer science and engineering where machines are built that perform tasks that would generally require human intelligence. Artificial Neural Networks (ANNs) which try and mimic the way the human brain functions, have been shown to work well when applied to real-time systems. The application areas for the use of neural networks include pattern recognition, classification and prediction. This research proposes the use of artificial neural networks as a single technique in the detection and classification of four different types of induction motor faults in an IEC 61850-based environment.

The artificial neural network is trained in the MATLAB software environment using off-line historical data to detect and classify the four types of induction motor faults. The weights and biases of the trained neural network are stored. These are then used in the neural network software algorithm developed on the Real-Time Automation Controller (RTAC). A real-time case study is presented where a test bed is built using the Real-Time Digital Simulator (RTDS) GTNET-GSE card to publish an Analogue GOOSE message containing the data obtained from the motor fault simulator data acquisition system. The subscribing device is the RTAC that contains the neural network algorithm for fault classification. The output of the neural network algorithm publishes a Binary GOOSE message to a subscribing motor protection IED. The IED in turn publishes a Binary GOOSE message to the GTNET-GSE subscriber to indicate the status of the fault classification algorithm.