Induction Machine Fault Detection Using Support Vector Machine Based Classifier

V.N. Ghate  
Electrical Engineering Department  
Government College of Engineering, Amravati (MS)  
INDIA  
vng786@rediffmail.com http://www.gcoeia.in

S.V. Dudul  
Applied Electronics Department  
Sant Gadge Baba Amravati University, Amravati  
INDIA  
dudulsv@rediffmail.com

Abstract: Industrial motors are subject to various faults which, if unnoticed, can lead to motor failure. The necessity of incipient fault detection can be justified by safety and economical reasons. The technology of artifical neural networks has been successfully used to solve the motor fault detection problem. This paper develops inexpensive, reliable, and noninvasive NN based fault detection scheme for small and medium sized induction motors. Detailed design procedure for achieving the optimal NN model and Principal Component Analysis for dimensionality reduction is proposed. Overall thirteen statistical parameters are used as feature space to achieve the desired classification. Generalized Feed Forward (GFFDNN) and Support Vector Machine (SVM) NN models are designed and verified for optimal performance in fault identification on experimental data set of custom designed 2 HP, three phase 50 Hz induction motor.

Keywords: Induction motor, Fault detection, Neural Network, GFFDNN, SVM, PCA

1 Introduction

Induction machines play a crucial role in certain industries, such as manufacture, transportation, etc. They offer the core capabilities for industrial success and the maintenance of them is essential and profitable to most electrical industrial processes. A lack of coherent maintenance strategy may lead to the loss of individual items of a plant, and a heavy capitalized losses burden. As it is not economical to introduce redundant backup machines, online monitoring for induction machines is important for safe operation and production quality. In order to keep machines in good condition, techniques such as fault monitoring, detection, classification, and diagnosis have become increasingly essential [1]–[3]. There are invasive and noninvasive methods for machine fault detection [4], [5], [7]. The noninvasive methods are more preferable than the invasive methods because they are based on easily accessible and inexpensive measurements to diagnose the machine conditions without disintegrating the machine structure. Recently, artificial intelligence (AI) techniques have been proposed for the noninvasive machine fault detection [4], [6], [8]. They have several advantages over the traditional model-based techniques [6], [9]. They require no detailed analysis of the different kinds of faults or modeling of the system. These AI-based techniques include expert systems, neural network, and fuzzy logic. An expert system is able to manage knowledge-based production rules that model the physical system [11], [12]. Neural network approaches can be considered as “black-box” methods as they do not provide heuristic reasoning about the fault detection process [4], [5], [10]. Fuzzy logic systems can heuristically implement fault detection principles and heuristically interpret and analyze their results [13], [14], [15]. In this paper, neural network type approach is used because generalized feed forward neural network (GFFDNN) is able to provide an accurate fault diagnostic classification.

2 Feature Extraction

The main problems facing the use of ANN are the selection of the best inputs and how to choose the ANN parameters making the structure compact, and creating highly accurate networks. For the proposed
system, the feature selection is also an important process since there are many features after feature extraction. Many input features need significant computational efforts to calculate, and may result in a low success rate. In order to collect data at different conditions i.e. healthy condition, under inter turn fault, Eccentricity and both i.e. inter turn and Eccentricity specially designed 2 HP, 4 pole, 415V, 50 Hz, three phase induction motor is used. Three AC current probes were used to measure the stator current signals. From the time waveforms, as shown in Fig.1. no conspicuous difference exists among the different conditions.

There is a need to come up with a feature extraction method to classify faults. To classify the different faults the statistical parameters are used. To be precise, ‘sample’ statistics will be calculated for current data. Overall thirteen parameters are calculated as input feature space. Minimum set of statistic to be examined includes the root mean square (RMS) of the zero mean signal (which is the standard deviation), the maximum, and minimum values the skewness coefficient and kurtosis coefficient. Pearson’s coefficient of skewness, $g_2$ defined by:

$$g_2 = \frac{3(\bar{x} - \tilde{x})}{S_x}$$

(1)

Where $\bar{x}$ denotes mean, $\tilde{x}$ denotes median and $S_x$ denotes the sample standard deviation. The sample coefficient of variation $v_x$ is defined by:

$$v_x = \frac{S_x}{\bar{x}}$$

(2)

The $r^{th}$ sample moment about the sample mean for a data set is given by:

$$m_r = \frac{1}{n} \sum_{i=1}^{n} (x_i - \bar{x})^r$$

(3)

$m_2$ denotes to spread about the center, $m_r$ refers to skewness about the center; $m_4$ denotes to how much data is massed at the center. Second, third and fourth moments are used to define the sample coefficient of skewness, $g_3$, and the sample coefficient of kurtosis, $g_4$ as follows.

$$g_3 = \frac{m_3}{(\sqrt{m_2})^3}$$

(4)

$$g_4 = \frac{m_4}{(\sqrt{m_2})^4}$$

(5)

The sample covariance between dimensions $j$ and $k$ is defined as:

$$c_{jk} = \frac{\sum_{i=1}^{n} (x_{ij} - \bar{x}_j)(x_{ik} - \bar{x}_k)}{(n-1)}$$

(6)

The ordinary correlation coefficient for dimensions $j$ and $k$, $r_{jk}$ is defined as:

$$r_{jk} = \frac{c_{jk}}{S_j S_k}$$

(7)

3 Generalized Feed Forward NN

Generalized feed forward network is a generalization of the MLP such that connections can jump over one or more layers. In this network it is allowed to cross the hidden layer, i.e. output layer will get the input from the hidden layer and directly from input layer also.

Generalized feed forward Neural Network is proposed as fault classifier. Number of input Processing Elements (PE) must be equal to that of number of input statistical parameters so 13 input Processing Elements are used in input layer. Four Processing Elements are used in output layer for four conditions of motor namely Healthy, Inter turn fault, Eccentricity and Both faults. For data processing MATLAB7.1, Neuro Solution 5.0 and XLSTAT is used. General learning algorithm used is as follows:

Initialization of Weights:
Step 1: Initialize the weights to small random values
Step 2: While stopping condition is false, do step 3-10
Step 3: For each training pair do steps 4-9

Feed forward:
Step 4: Each input unit receives the input signal $x_i$ and transmits this signals to all units in the hidden layer
Step 5: Each hidden unit ($z_j , j=1,\ldots,p$) sums its weighted input signals
\[ z_{inj} = v_{oj} + \sum_{i=1}^{n} x_i v_{qi} \] \hspace{1cm} (8)

Applying the activation function \( Z_j = f(z_{inj}) \) here the activation function is \( \tanh(x) = (e^x - e^{-x}) / (e^x + e^{-x}) \) and sends this signal to all units in output units.

Step 6: Each output unit \((y_k, k=1,\ldots,m)\) sums its weighted input signals,
\[ y_{inj} = w_{ok} + \sum_{j=1}^{p} z_j w_{jk} + \sum_{j=1}^{p} x_i v_j \] \hspace{1cm} (9)
And applies its activation function to calculate the output signals \( Y_k = f(y_{inj}) \) here the activation function is
\[ \tanh(x) = (e^x - e^{-x}) / (e^x + e^{-x}) \] \hspace{1cm} (10)

**Back Propagation Error:**

Step 7: Each output unit \((y_k, k=1,\ldots,m)\) receives a target pattern corresponding to an input pattern error information term is calculated as
\[ \delta_k = (t_k - y_k) f'(y_{inj}) \] \hspace{1cm} (11)

Step 8: Each hidden unit \((z_j, j=1,\ldots,p)\) sums its delta inputs from units in the layer above
\[ \delta_{inj} = \sum_{k=1}^{m} \delta_k w_{jk} \] \hspace{1cm} (12)
The error information term is calculated as
\[ \delta_j = \delta_{inj} f'(z_{inj}) \] \hspace{1cm} (13)

Updation of weight and Biases:

Step 9: Each output unit \((y_k, k=1,\ldots,m)\) updates its bias and weights \((j=0,\ldots,p)\)
\[ w_{jk} (t+1) = w_{jk} (t) + \alpha \delta_k z_j + \mu [w_{jk} (t) - w_{jk} (t-1)] \] \hspace{1cm} (14)
Where \( \alpha \) is learning rate and \( \mu \) is momentum factor

And each hidden unit \((z_j, j=1,\ldots,p)\) updates its bias and weights \((i=0,\ldots,n)\)
\[ v_{ij} (t+1) = v(t) + \alpha \delta_j x_i + \mu [v_{ij} (t) - v_{ij} (t-1)] \] \hspace{1cm} (15)

Step 10: Test the stopping condition

**Selection of Error criterion:**

Supervised learning requires a metric, a measure of how the network is doing. Members of the Error Criteria family monitor the output of a network, compare it with some desired response and report any error to the appropriate learning procedure. In gradient descent learning, the metric is determined by calculating the sensitivity that a cost function has with respect to the network’s output. This cost function, \( J \), is normally positive, but should decay towards zero as the network approaches the desired response. The literature has presented several cost functions, in which \( p \) is to be define such as \( p=1, 2, 3, 4\ldots 8 \) criterion is \( L_1, L_2, L_3, L_4 \ldots L_8 \).

Components in the ErrorCriteria family are defined by a cost function of the form:
\[ J(t) = \frac{1}{2} \sum_{i=1}^{p} (d_i(t) - y_i(t))^p \] \hspace{1cm} (16)
and error function:
\[ e_i(t) = (d_i(t) - y_i(t)) \] \hspace{1cm} (17)
Where \( d(t) \) and \( y(t) \) are the desired response and network’s output, respectively. To select the correct error criterion various error criterion has been tested and results are shown in Fig. 2, Fig.3 and Fig. 4.

![Fig.2 Variation of Average Minimum MSE with Error Criterion](image)

![Fig.3 Variation of Average Classification Accuracy with Error Criterion](image)

![Fig.4 Variation of Average MSE with Error Criterion](image)
observed that network with single hidden layer and 5 PEs in hidden layer gives the better results as shown in Fig.5(a) and Fig.5(b).

Fig.5 (a) Variation of Average MSE with Number of PEs in Hidden Layer

Fig.5 (b) Variation of Average Minimum MSE with Number of PEs in Hidden Layer

Various transfer functions and learning rules namely Momentum (MOM), Conjugate-Gradient (CG), Quick Propagation (QP), Delta Bar Delta (DBD), Levenberg-Marquardt (LM) and Step (STP) are verified for training and testing the network. Average minimum MSE on training and CV data and classification accuracy on testing, CV and training data is compared in Table 1 and Table 2 (Appendix)

The parameters of the hidden layer and output layer i.e. step size and momentum are selected by comparing average minimum MSE. In Hidden layer optimum value of Step size is 0.13 and momentum is 0.6 and for output layer Step size is 0.05 and momentum is 0.08. Performance is shown in Fig.6, Fig. 7 and Fig.8.

Fig.6 Variation of Average Minimum MSE with Step size in Hidden Layer

Fig.7 Variation of Average Minimum MSE with Momentum rate in Hidden Layer

Fig.8 Variation of Average Minimum MSE with Step size and Momentum rate in output Layer

From above experimentation, selected parameters for GFFD-NN are given below.

GFFD- NN (13-5-4), Number of epochs = 5000,
Exemplars for training = 70%,
Exemplars for cross validation = 15%,
Exemplars for Testing = 15%
Number of Hidden Layers: 01
T.F.: Tanh Learning Rule: Momentum
Step size: 0.13 Momentum: 0.6
Output Layer:
T.F.: Tanh Learning Rule: Momentum
Step size: 0.05 Momentum: 0.08
Number of connection weights: 146
Time Elapsed per epoch per exemplar: 1.001 msec.

Different datasets are formed using variable split ratios and leave-N-out cross validation technique. Proposed NN is trained and tested five times on various datasets and later validated carefully so as to ensure that its performance does not depend on specific data partitioning scheme. The performance of the NN should be consistently optimal over all the datasets with respect to MSE and classification accuracy. To check the learning ability and classification accuracy the total data is divided in four groups. First two groups (50% data) are tagged as Training data and third and forth group (each 25%) is tagged for Cross Validation and Testing (1234:1,2-TR, 3-CV, 4-Test). Similar 18 combinations are prepared and network is train and test for each group. Results are shown in Fig.9 to Fig.12.
The support vector machine (SVM) is a new kind of classifier that is motivated by two concepts. First, transforming data into a high-dimensional space can transform complex problems (with complex decision surfaces) into simpler problems that can use linear discriminant functions. Second, SVMs are motivated by the concept of training and using only those inputs that are near the decision surface since they provide the most information about the classification.

It is a kind of learning machine based on statistical learning theory. The basic idea of applying SVM to pattern classification can be stated as follows: first map the input vectors into one features space, possible in higher space, either linearly or non-linearly, which is relevant with the kernel function. Then, within the feature space from the first step, seek an optimized linear division, that is, construct a hyperplane which separates two classes. It can be extended to multi-class. SVMs training always seek a global optimized solution and avoid over fitting, so it has ability to deal with a large number of feature.

Kernel Adatron algorithm:

For N dimensional space data \( x_i \) \((i = 1...N)\) this algorithm can be easily extended to network by substituting the inner product of patterns in the input space by the kernel function, leading to the following quadratic optimization problem:

\[
J(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j d_{ij} G(x_i - x_j, 2\sigma^2) \quad (16)
\]

Subject to

\[
\sum_{i=1}^{N} d_i \alpha_i = 0 \quad \alpha_i \geq 0, \forall \alpha \in \{1,...N\} \quad (17)
\]

where \( G(x, s^2) \) represents a Gaussian function, \( N \) is the number of samples, \( \alpha_i \) are a set of multipliers (one for each sample),

\[
J(x_i) = d_i \left( \sum_{i=1}^{N} d_i \alpha_j G(x_i - x_j, 2\sigma^2) + b \right) \quad (18)
\]

and

\[
M = \min_i g(x_i) \quad (19)
\]

and choose a common starting multiplier \( \alpha_i \), learning rate \( \eta \), and a small threshold. Then, while \( M > t \), we choose a pattern \( x_i \) and calculate an update \( \Delta \alpha_i = \eta(1 - g(x_i)) \) and perform the update

If \( \alpha_i (n) + \Delta \alpha_i > 0 \)

\[
\alpha_i (n + 1) = \alpha_i (n) + \Delta \alpha_i (n)
\]

\[
b(n + 1) = b(n) + d_i \Delta \alpha_i \quad (20)
\]

And if \( \alpha_i (n) + \Delta \alpha_i \leq 0 \)

\[
\alpha_i (n + 1) = \alpha_i (n)
\]

\[
b(n + 1) = b(n) \quad (21)
\]

After adaptation only some of the \( \alpha_i \) are different from zero (called the support vectors). They correspond to the samples that are closest to the boundary between classes. This algorithm can be considered the "on-line" version of the quadratic
optimization approach utilized for SVMs, and it can find the same solutions as Vapnik's original algorithm for SVMs. It is easy to implement the kernel Adatron algorithm since \( g(x_i) \) can be computed locally to each multiplier, provided that the desired response is available in the input file. In fact, the expression for \( g(x_i) \) resembles the multiplication of an error with an activation, so it can be included in the framework of neural network learning. The Adatron algorithm essentially prunes the RBF network so that its output for testing is given by,

\[
f(x) = \text{sgn}\left( \sum_{i \in \text{support vectors}} d_i \alpha_i G(x_i - x, 2\sigma^2) - b \right)
\]  

(22)

And cost function in error criterion is

\[
J(t) = \frac{1}{2} \sum_{i=1}^{N} \left( d_i(t) - (\tanh(y_i(t))) \right)^2
\]

(23)

For selection of step size randomize data is fed to the neural network and is retrained five times with different random weight initialization. It is observed that 0.8 step size gives the optimal result. Number of connection weights: 472

Time Elapsed per epoch per exemplar: 0.934 ms

Using the similar datasets SVM classifier is tested retaining five times and results are shown in Fig.15 to Fig.18

Fig. 13 Variation of Average Minimum MSE with Step size

Fig. 14 Variation of Average MSE with Step size

Fig. 15 Variation of average Minimum MSE with Test on Testing and Training dataset and percent data tagged for training

Fig.16 Variation of average Classification Accuracy with testing on Testing and Training dataset and percent data tagged for training

Fig.17 Variation of average MSE with Test on Testing, CV and Training dataset with CV rows shifted (n)

Fig. 18 Variation of average Minimum MSE with Training and CV with group of Dataset
5 Dimensionality Reduction Using Principal Component Analysis

One problem appears after the feature extraction. There are too many input features that would require a significant computational efforts to calculate, and may result in low accuracy of the monitoring and fault diagnosis. The potential improvements which can be achieved by first mapping the data into a space of lower dimensionality. Reduction in dimensionality of the input space and hence the network can be achieved by Principal Component Analysis (PCA). PCA is performed by Pearson rule. The Fig.12 is related to a mathematical object, the eigenvalues, which reflect the quality of the projection from the 13-dimensional to a lower number of dimensions.

Using the results of Principal Component Analysis, dimensions of GFFD-NN can be reduced. Number of inputs are reduced to five. By similar experimentations, the optimum GFFD-NN classifier is designed with the following changes;
Number of Inputs = 5, Number of PEs in Hidden Layer = 5, Number of epochs = 5000, Exemplars for training = 70%, Exemplars for cross validation = 15%, Exemplars for Testing = 15%.
Number of Hidden Layers: 01
T.F.: Tanh Learning Rule: Momentum
Step size: 0.09 Momentum: 0.6
Output Layer:
T.F.: Tanh Learning Rule: Momentum
Step size: 0.05 Momentum: 0.08
Number of connection weights: 74
Time Elapsed per epoch per exemplar: 0.786 ms
Training and testing results for new model is as shown in Fig.20 to Fig.25
Using the results of PCA, dimensions of the support vector machine classifier are also reduced. It is found that number of inputs, reduced to five and step size of 0.7 gives the optimal results. 
Number of connection weights: 264 
Time Elapsed per epoch per exemplar: 0.693 ms

Variation of average minimum MSE and average classification accuracy with number of PCs as input is shown in Fig. 26 and Fig. 27, and training and testing results are shown in Fig. 28 and Fig. 29.

6 Robustness of Classifier to Noise

Since the proposed classifier is to be used in real time, where measurement noise is anticipated, it is necessary to check the robustness of classifier to noise. To check the robustness Uniform and Gaussian noise with mean value zero and variance varies from 1 to 20% is introduced in input and output and average classification accuracy on testing data i.e. unseen data is checked. It is observed that in GFFDNN, average classification accuracy is not affected by both noise in input and output and in SVM classification accuracy is consistent with noise of variance up to 15%. Comparative results are shown in Table 3.

7 Results and Discussion

In this paper, the authors evaluated the performance of the developed GFFD NN and Support Vector Machine (SVM) based classifier for detection of four conditions of three phase induction motor and examined the results. After completion of the training, the learned network is able to detect different types of faults. For GFFDNN various learning rules and transfer functions are investigated for different number of hidden layers and processing elements in hidden layer. It is observed that Momentum learning rule and Tanh transfer function gives the optimal results in hidden and output layer.
By varying the step size optimum results are obtained in SVM classifier. By performing Principal Component Analysis, number of inputs are reduced from 13 to 5 and thus significant reduction in dimension is achieved. From the analysis, it is seen that dimensionally reduced support vector machine(SVM-DR) based classifier works as an elegant classifier for fault diagnosis of three phase induction motor, in the sense that, average MSE on testing and cross validation samples is consistently observed as reasonably low such as 0.0591 and 0.0619, respectively. In addition, average classification accuracy on testing as well as cross validation instances is obtained as 99.61% and 98.72%, respectively indicating a reasonable classification. Also proposed classifier is enough robust to the noise, in the sense that classifier gives consistent results for Uniform and Gaussian noise with 12% variance in input and with 20% variance in output. Comparative results are shown in Table 4.

References:


### APPENDIX

#### Table 1
VARIATION OF AVERAGE MINIMUM MSE AND AVERAGE CLASSIFICATION ACCURACY WITH TRANSFER FUNCTIONS

<table>
<thead>
<tr>
<th>TF</th>
<th>Average Minimum MSE on Training</th>
<th>Average Classification Accuracy CV</th>
<th>Test</th>
<th>CV</th>
<th>TR</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Max.</td>
<td>Min.</td>
<td>Avg.</td>
<td>SD</td>
<td>Max.</td>
</tr>
<tr>
<td>Tanh</td>
<td>0.0024</td>
<td>0.0016</td>
<td>0.0021</td>
<td>0.0003</td>
<td>0.1997</td>
</tr>
<tr>
<td>L-Tanh</td>
<td>0.0645</td>
<td>0.0035</td>
<td>0.0158</td>
<td>0.0273</td>
<td>0.2742</td>
</tr>
<tr>
<td>L-sig</td>
<td>0.3731</td>
<td>0.3548</td>
<td>0.3668</td>
<td>0.0084</td>
<td>0.3734</td>
</tr>
<tr>
<td>Sig</td>
<td>0.3569</td>
<td>0.3557</td>
<td>0.3564</td>
<td>0.0005</td>
<td>0.3659</td>
</tr>
<tr>
<td>Lin</td>
<td>0.405</td>
<td>0.3897</td>
<td>0.3979</td>
<td>0.0059</td>
<td>0.4103</td>
</tr>
<tr>
<td>BAX</td>
<td>0.2985</td>
<td>0.2985</td>
<td>0.2985</td>
<td>0</td>
<td>0.3035</td>
</tr>
<tr>
<td>AX</td>
<td>0.1248</td>
<td>0.1222</td>
<td>0.1233</td>
<td>0.0001</td>
<td>0.4304</td>
</tr>
<tr>
<td>Soft</td>
<td>0.0184</td>
<td>0.0144</td>
<td>0.0158</td>
<td>0.0017</td>
<td>0.0484</td>
</tr>
</tbody>
</table>


#### Table 2
VARIATION OF AVERAGE MINIMUM MSE AND AVERAGE CLASSIFICATION ACCURACY WITH TRANSFER FUNCTIONS

<table>
<thead>
<tr>
<th>TF</th>
<th>Average Minimum MSE on Training</th>
<th>Average Classification Accuracy CV</th>
<th>Test</th>
<th>CV</th>
<th>TR</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Max.</td>
<td>Min.</td>
<td>Avg.</td>
<td>SD</td>
<td>Max.</td>
</tr>
<tr>
<td>Moment</td>
<td>0.0024</td>
<td>0.0016</td>
<td>0.0021</td>
<td>0.0003</td>
<td>0.1997</td>
</tr>
<tr>
<td>DBD</td>
<td>0.016</td>
<td>0.0109</td>
<td>0.0132</td>
<td>0.0019</td>
<td>0.0128</td>
</tr>
<tr>
<td>QP</td>
<td>0.0026</td>
<td>0.0025</td>
<td>0.0026</td>
<td>8E-05</td>
<td>0.2124</td>
</tr>
<tr>
<td>CG</td>
<td>0.0294</td>
<td>0.0025</td>
<td>0.0086</td>
<td>0.0116</td>
<td>0.3015</td>
</tr>
<tr>
<td>Step</td>
<td>0.0037</td>
<td>0.0026</td>
<td>0.0029</td>
<td>0.0004</td>
<td>0.1657</td>
</tr>
<tr>
<td>LM</td>
<td>0.0085</td>
<td>0.0006</td>
<td>0.0075</td>
<td>0.001</td>
<td>0.1827</td>
</tr>
</tbody>
</table>

DBD: Delta Bar Delta         QP: Quick Propagation  CG: Conjugate-Gradient  LM: Levenberg-Marquardt
Table 3
EFFECT OF NOISE ON AVERAGE CLASSIFICATION ACCURACY WHEN CLASSIFIER TESTED ON TESTING DATA

<table>
<thead>
<tr>
<th>NN-Model</th>
<th>GFFDNN</th>
<th>GFFDNN-DR</th>
<th>SVM</th>
<th>SVM-DR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Noise in</td>
<td>Input</td>
<td>Output</td>
<td>Input</td>
<td>Output</td>
</tr>
<tr>
<td>Type of Noise</td>
<td>G</td>
<td>U</td>
<td>G</td>
<td>U</td>
</tr>
<tr>
<td>% Variance</td>
<td>Average Classification Accuracy on Testing on Testing Data i.e. unseen Data</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>75</td>
<td>100</td>
<td>100</td>
<td>50</td>
</tr>
<tr>
<td>2</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>3</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>4</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>75</td>
</tr>
<tr>
<td>5</td>
<td>100</td>
<td>100</td>
<td>50</td>
<td>100</td>
</tr>
<tr>
<td>6</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>7</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>8</td>
<td>100</td>
<td>75</td>
<td>100</td>
<td>75</td>
</tr>
<tr>
<td>9</td>
<td>100</td>
<td>87.5</td>
<td>100</td>
<td>75</td>
</tr>
<tr>
<td>10</td>
<td>100</td>
<td>75</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>11</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>75</td>
</tr>
<tr>
<td>12</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>13</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>14</td>
<td>100</td>
<td>100</td>
<td>75</td>
<td>100</td>
</tr>
<tr>
<td>15</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>16</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>17</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>18</td>
<td>100</td>
<td>100</td>
<td>75</td>
<td>100</td>
</tr>
<tr>
<td>19</td>
<td>100</td>
<td>100</td>
<td>75</td>
<td>100</td>
</tr>
<tr>
<td>20</td>
<td>100</td>
<td>87.5</td>
<td>100</td>
<td>100</td>
</tr>
</tbody>
</table>

U: Uniform Noise
G: Gaussian Noise
<table>
<thead>
<tr>
<th>N-N Model</th>
<th>Performance</th>
<th>Testing on Test Data</th>
<th>Testing on CV Data</th>
<th>T</th>
<th>W</th>
<th>% W</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Max. Observed</td>
<td>Min. Observed</td>
<td>Average</td>
<td>SD</td>
<td>Max. Observed</td>
<td>Min. Observed</td>
</tr>
<tr>
<td>GFFDNN</td>
<td>MSE</td>
<td>0.2435</td>
<td>0.00135</td>
<td>0.05273</td>
<td>0.072</td>
<td>0.16856</td>
</tr>
<tr>
<td></td>
<td>Percent Correctness</td>
<td>100</td>
<td>75</td>
<td>94.79</td>
<td>9.216</td>
<td>100</td>
</tr>
<tr>
<td>GFFDNN</td>
<td>MSE</td>
<td>0.28437</td>
<td>0.00022</td>
<td>0.05653</td>
<td>0.077</td>
<td>0.21960</td>
</tr>
<tr>
<td></td>
<td>Percent Correctness</td>
<td>100</td>
<td>75</td>
<td>98.5</td>
<td>5.997</td>
<td>100</td>
</tr>
<tr>
<td>SVM</td>
<td>MSE</td>
<td>0.84364</td>
<td>0.00529</td>
<td>0.04527</td>
<td>0.120</td>
<td>0.28433</td>
</tr>
<tr>
<td></td>
<td>Percent Correctness</td>
<td>100</td>
<td>83.33</td>
<td>97.41</td>
<td>5.249</td>
<td>100</td>
</tr>
<tr>
<td>SVM DR</td>
<td>MSE</td>
<td>0.09926</td>
<td>0.05086</td>
<td>0.05915</td>
<td>0.011</td>
<td>0.094134</td>
</tr>
<tr>
<td></td>
<td>Percent Correctness</td>
<td>100</td>
<td>88.88</td>
<td>99.611</td>
<td>1.944</td>
<td>100</td>
</tr>
</tbody>
</table>

T - Time Elapsed per epoch per exemplar for training in ms.
W - Number of weight connections.
%W - Percent number of weight connections reduced.
<table>
<thead>
<tr>
<th>N-N Model</th>
<th>Performance</th>
<th>HLTY</th>
<th>BOTH</th>
<th>INT</th>
<th>ECE</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>GFFDNN (13-5-4)</td>
<td>MSE</td>
<td>0.0072</td>
<td>0.0023</td>
<td>0.0160</td>
<td>0.0109</td>
<td>0.0091</td>
</tr>
<tr>
<td></td>
<td>Percent Correctness</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>GFFDNN DR (5-5-4)</td>
<td>MSE</td>
<td>0.0009</td>
<td>0.0719</td>
<td>0.0024</td>
<td>0.0243</td>
<td>0.0249</td>
</tr>
<tr>
<td></td>
<td>Percent Correctness</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>SVM</td>
<td>MSE</td>
<td>0.0001</td>
<td>0.0003</td>
<td>0.0002</td>
<td>0.0006</td>
<td>0.0003</td>
</tr>
<tr>
<td></td>
<td>Percent Correctness</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>SVM DR</td>
<td>MSE</td>
<td>0.0036</td>
<td>0.0353</td>
<td>0.0025</td>
<td>0.0294</td>
<td>0.0177</td>
</tr>
<tr>
<td></td>
<td>Percent Correctness</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Desired Output of Network</th>
</tr>
</thead>
<tbody>
<tr>
<td>H B I E</td>
</tr>
<tr>
<td>H B I E</td>
</tr>
<tr>
<td>0 0 0 1</td>
</tr>
<tr>
<td>0 0 0 1</td>
</tr>
<tr>
<td>0 0 1 0</td>
</tr>
<tr>
<td>0 1 0 0</td>
</tr>
<tr>
<td>0 0 1 0</td>
</tr>
<tr>
<td>0 1 0 0</td>
</tr>
<tr>
<td>0 0 0 1</td>
</tr>
<tr>
<td>0 0 0 1</td>
</tr>
<tr>
<td>1 0 0 0</td>
</tr>
<tr>
<td>0 0 1 0</td>
</tr>
<tr>
<td>0 0 1 0</td>
</tr>
<tr>
<td>1 0 0 0</td>
</tr>
<tr>
<td>0 0 0 1</td>
</tr>
<tr>
<td>0 0 1 0</td>
</tr>
<tr>
<td>1 0 0 0</td>
</tr>
<tr>
<td>0 0 0 1</td>
</tr>
<tr>
<td>0 0 1 0</td>
</tr>
<tr>
<td>0 0 0 1</td>
</tr>
</tbody>
</table>

Table 5
SAMPLE RESULTS OF NETWORK FOR EACH FAULT

Table 6
SAMPLE DESIRED AND ACTUAL OUTPUT OF NETWORK