Multi-fault classification of Induction motor

A B. Tech Project Report Submitted
in Partial Fulfillment of the Requirements
for the Degree of

Bachelor of Technology

by

Supragya Shrestha, Yashraj Molawade
(170103068, 170103091)

under the guidance of

Prof. Rajiv Tiwari, Dr D.J.Bordoloi

to the

DEPARTMENT OF MECHANICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY GUWAHATI
GUWAHATI - 781039, ASSAM

DECLARATION

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Supragya Shrestha (170103068)

Yashraj Molawade (170103091)

April, 2021

CERTIFICATE

*This is to certify that the work contained in this thesis entitled "**Multi-fault classification of Induction motor**" is a bonafide work of **Supragya Shrestha, Yashraj Molawade** (**Roll No. 170103068, 170103091**), carried out in the Department of Mechanical Engineering, Indian Institute of Technology Guwahati under my supervision and that it has not been submitted elsewhere for a degree.*

Supervisor: **Prof. Rajiv Tiwari, Dr D.J.Bordoloi**

Department of Mechanical Engineering

Indian Institute of Technology Guwahati, Assam

Abstract

The need for smooth and trouble-free operation of induction motors has risen in the current era.This thesis tries to understand the problem and present a way to diagnose faults in motors in real-time. It uses the latest deep learning techniques for identifying the parameters related to faults and trains itself to predict any future occurrences. Ten IM fault conditions are considered for the diagnosis, i.e. four mechanical fault conditions (i.e., the bearing fault, the unbalanced rotor, the bowed rotor and the misaligned rotor), five electrical fault conditions (i.e., the broken rotor bar, the stator winding fault with two severity levels, and phase unbalance with two severity levels), and a healthy IM. In recent years, deep learning has achieved great success in many fields, such as computer vision and natural language processing. Compared to traditional machine learning methods, deep learning has a strong learning ability and can make better use of datasets for feature extraction. Because of its practicability, deep learning becomes more and more popular for many researchers to do research works. These techniques in conjunction with the available data regarding IM faults presents a strong potential in creating a robust mechanism for efficient fault detection algorithms.

Contents

Chapter 1

Introduction

In several manufacturing processes, induction motors are essential components. Therefore, it is becoming increasingly important to track induction motors online. The primary problem in this role is the lack of an appropriate theoretical model to characterise a defective motor with multiple forms of faults, such as winding faults. This leads to unintended faults in various motor components. In their early stages, these defects are normally left ignored, and conclude with disastrous system failure.

1.1 Aim

1. Comprehensive Understanding of continuous and error free operation of Induction Motors

2. Data mining, i.e, obtaining the right amount of data to work with

3. Considering various state-of-the-art techniques to evaluate the merit of the models developed

4. Improvement in performance of fault diagnosis using deep learning

5. Final model evaluation and optimization over further iterations

The process flow begins with pre-processing the data and designing the appropriate model architecture. This is followed by the model's training and testing over the data to get the final predictions. These predictions are then further checked and analysed to better the existing the model.

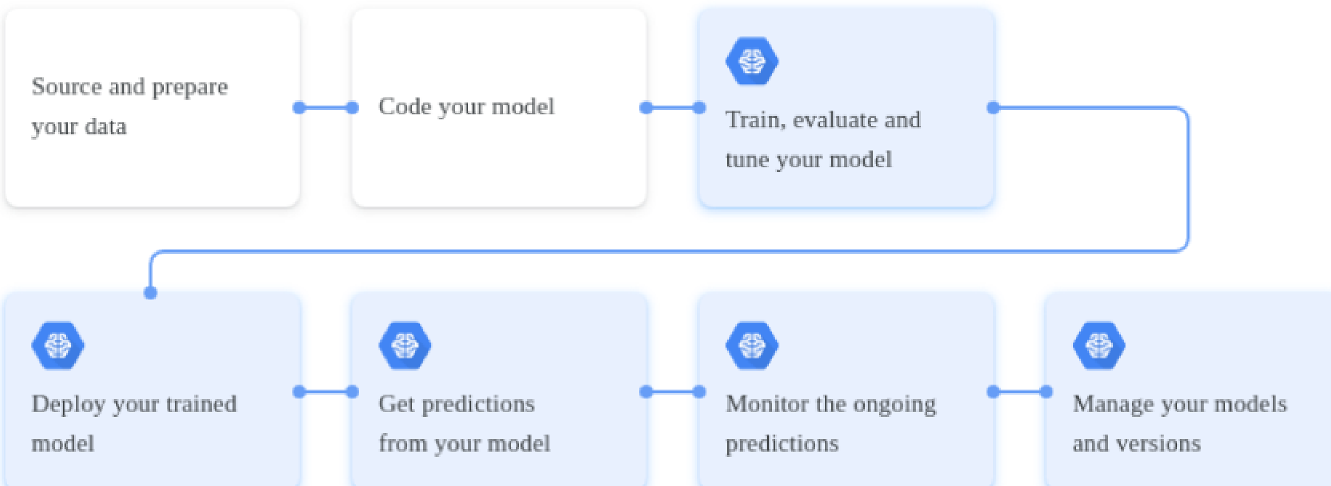

Fig. 1.1: Process Flow (*Source: oralytics.com*)

1.2 Induction Motors

An Induction Motor consists of three main components: the stator, rotor, and bearings.

1. The stator is the outer, stationary component of the IM. It consists of a stator frame,stator core, and stator winding/field winding. The slots on the periphery of the stator core carry three-phase windings. This three-phase winding is supplied by a three-phase AC supply and produces a rotating magnetic field inside the stator.

2. The rotor is the rotating component of the IM. It is connected to the mechanical load through the shaft. Two types of rotors are used in a three-phase IM: squirrel cage rotor and slip ring (or wound) rotor.

(a) The squirrel cage rotor consists of metal bars also known as rotor conductors. These conductors are permanently shorted by the end ring and placed in slots on the periphery of the rotor. The end ring braces the bars to provide mechanical support., forming a complete closed circuit resembling a cage and giving the squirrel cage rotor its name.

(b) The slip ring (wound) rotor consists of a number of slots with rotor winding placed in it. It consists of slip rings permanently connected to three-phase winding. This study uses a three-phase squirrel-cage IM.

3. The bearing used is ball bearing type; supports the rotating shaft.

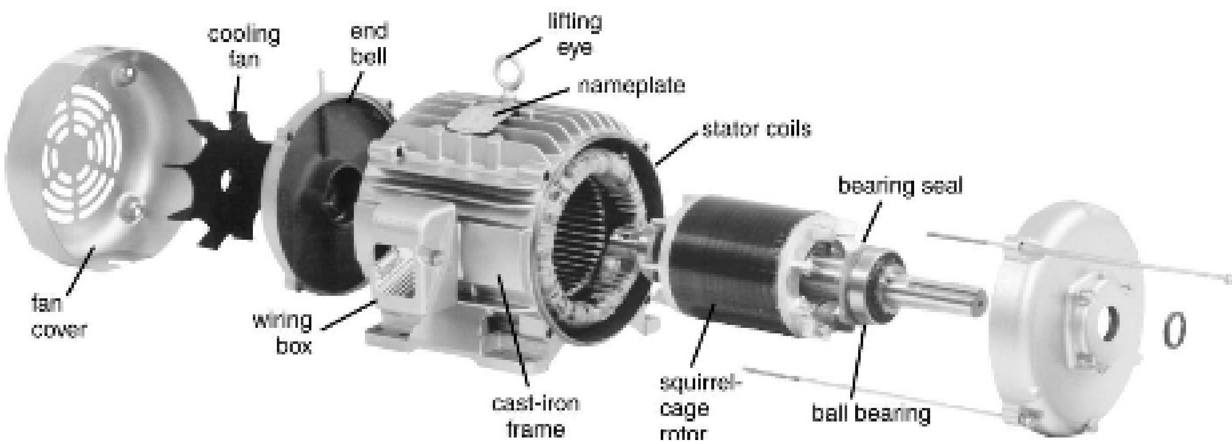

Fig. 1.2: Exploded view of induction motor (*Source: researchgate.com*)

1.3 Types of Faults in Induction Motor

More than 69 of the total faults are accounted for just by the bearing and the stator winding faults. Going by this statistic the contribution of rotor and shaft damages may seem trivial, though a closer look reveals that the majority of bearing damages are due to the shaft unbalance, rotor eccentricity, and misalignment. Although erratic power supply is the main culprit in winding failures sometimes winding is damaged by the broken

and/or misaligned rotor. The electrical faults include the stator winding fault–turn-to-turn fault,coil-to-coil fault, phase-to-phase fault, phase-to-ground fault, and etc., the rotor related faults–such as broken rotor bars, broken end ring, etc.and power supply related faults– such as phase or voltage unbalance, single phasing, and etc.
Faults in IM are broadly classified into two types : Mechanical and Electrical faults.

1.3.1 Mechanical Faults

1. Bearing fault : Most IM use either a ball or roller bearing, which consists of the inner race, outer race, rolling element, and cage. The bearing fault can occur by damage in any of these elements.The bearing fault leads to the eccentricity of the rotor in the stator which causes an unbalanced magnetic pull eventually puts more load on the bearings. The air gap between the rotor and the stator affects the shaft dynamics. Thus the bearing fault leads to aberrant vibrations in the motor. Bearing fault leads to rotor bar failures which ultimately causes total failure of the induction motor.

Fig. 1.3: A bearing with crack in the outer race

2. Rotor related fault: In this fault, the router is not centrally aligned or its axis of rotation is not coincident with the geometrical axis of the stator. this condition is referred to as the air gap eccentricity and causes unbalanced radial forces. large radial forces can cause contact between stator and rotor and can lead to damage.

1.3.2 Electrical Faults

1. Stator winding fault: The stator winding fault (SWF) occurs because of the turn-to- turn, coil-to-coil, phase-to-phase or phase-to-ground fault. Commonly SWFs are the outcome of the growth of undetected turn-to-turn faults. The main culprit for the turn-to-turn fault is prolonged thermal aging and finally insulation failure. The SWF can occur due to a host of reasons: excessive heating (thermal stress), the unbalanced power supply (electrical stress), impact by broken/unbalanced/misaligned rotor bar (mechanical stresses), extreme vibrations, installation failure, and oil contamination.This fault might head to the opening, shorting or grounding windings, excessive heating or total damage of machines.[2]

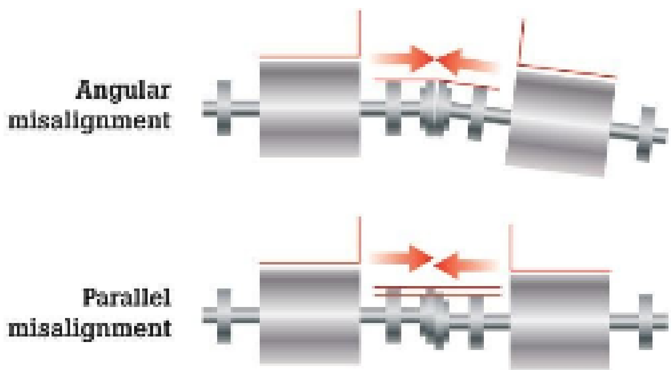

Fig. 1.4: Angular and parallel misalignment

2. Broken rotor bar faults: The failure of rotor bars or end rings is known as a broken rotor bar fault (BRB). It occurs mainly due to the fluctuations in loading and is the main cause of rotor failure. BRB produces frequency both in vibration signals and current signals. it is easier to detect this fault in the current spectrum because these variations are larger than in the vibration spectrum.

3. Phase unbalance and single phasing fault: When the voltages are not equal between the phases it is known as phase unbalance. The small unbalance can cause a severe increase in current in motor windings. Over time, total damage of the motor occurs

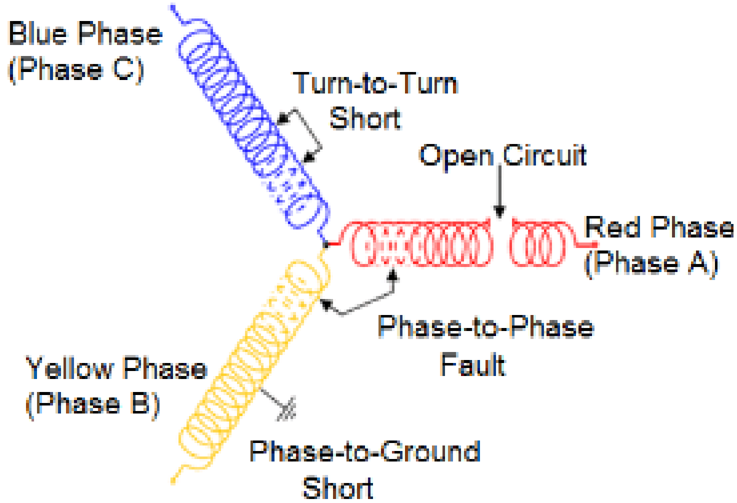

Fig. 1.5: Stator winding faults [6]

due to overheating. The single phasing fault is produced when one of the phases of the motor is abruptly open-circuited during operation.

Fig. 1.6: Broken rotor bar faults [8]

1.4 Literature review

Gangsar P. (2019) found out that induction engines are exposed to various mechanical, electrical, thermal and environmental stresses that lead to multiple faults: phase imbalance, rotor misalignment, bearing faults, gearbox faults, etc. Induction engines are a main inseparable feature in every industry. Industries are prepared to make a substantial investment in condition control and early detection of IM faults .[2]

Yong Oh Lee et al. (2017) found that in a DNN, the maximum number of secret layers is 12, reflecting a very deep network taking more time in the training process. Often the ratio of training data and test data is different, impacting the confidence of the relation of different algorithms .[4]

Nyugen and Lee (2008) successfully used help vector machines to identify faults with high accuracy and precision based on extracted features .[5]

DL systems are typically trained using a trait database that is accessible by business studies. Peiming et al. (2017), however, acknowledged that it is difficult to provide knowledge under all operating circumstances. Intelligent fault detection is also complicated where the evaluation data does not complement the training database used. For subsequent intelligent classification and fault diagnosis, the fault features extracted by deep learning are sufficient. The DNN network will retrieve secret information automatically in the data; they avoid the need for prior diagnosis experience and signal processing technologies and improve the efficiency of feature extraction.[7]

Shen Zhang et al. observed that test accuracy of the DL algorithms applied to bearing fault diagnosis is above 95 % which verifies the utility and effectiveness of applying deep learning in fault diagnostics. Many papers did not ensure a balanced sampling, which means the ratio of data samples selected from the healthy condition and the faulty condition is not close to 1:1. In the case of significant unbalance, accuracy should not be used as the only metric to evaluate an algorithm. Compared with accuracy, other metrics, such as precision, recall, and F1-Score, should be introduced for evaluating the reliability of a fault diagnostic network.[9]

The report will further discuss the experimental setup required for data collection of the given problem. This will be the starting point of data analysis on which we will design the model. We first try to solve the problem of binary classification and then move toward the multi-class classification which is our objective.

Chapter 2

Experimental Setup

The data was collected by conducting experiments on a test-rig with a machine fault simulator (MFS). The test-rig consists of test IMs with different faults, sensors (tri-axial ac-celerometer and AC current probe), a DC power source, and a data acquisition system(DAQ).

2.1 Machine Fault Simulator

The MFS simulates the faults on the IM and bearings. The MFS consists of an IM, a splitbracket bearing housing, a sliding shaft, rotors with split collar ends, a flexible coupling, pulleys, a magnetic brake attached with a gearbox, and a tachometer. In order to acquirevibration and the current signal from test IMs, a tri-axial accelerometer was installed on the top of the test motor near the motor shaft end.

2.2 Measurement Sensors

A tri-axis accelerometer was attached to the top of the motor, close to the bearing atthe shaft end, to obtain vibration signals in three orthogonal directions–axial, radial andtangential. This sensor works the phenomenon of piezoelectricity–generates an electriccurrent

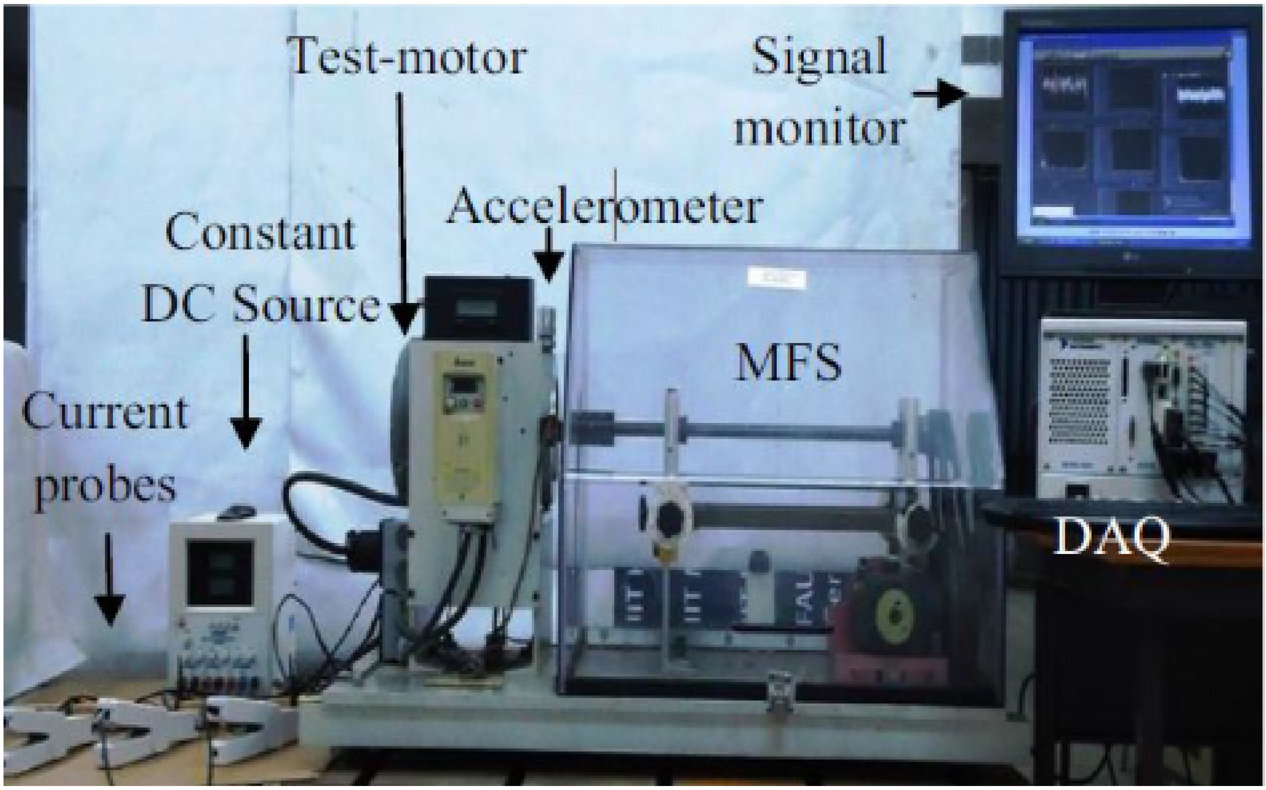

Fig. 2.1: Test Rig

to vibration acceleration.

2.3 AC current probes

The three AC current sensors were clamped to the three cables of IM to acquire statorcurrent signals in all the three phases. AC/DC current probe accurately measures currentsfrom 100 mA to 100 A RMS; DC to 100 kHz AC. It uses a Hall sensor technology formeasurement.

2.4 Tachometer or Photovoltaic Sensor

The tachometer is mounted near the coupling in the MFS; it measures the angular speedof the shaft. It requires a DC power supply to operate.

2.5 Data Acquisition System

The DAQ is an analog-to-digital converter. The analog signal is a continuous voltage signalgenerated by the variations in motor vibration and current to the windings. The analogsignal generated from the sensors is fed as input to the DAQ.

The total data collected was grouped into respective error types with total input data shape being 4498000 rows X 10 columns.

Chapter 3

Overview of Time Series Analysis

3.1 Time series analysis

After careful analysis of the data, it can be observed that rows are related to the description for 1 second of operation containing 2000 samples and 300 such sets exist corresponding to 5 minutes of operation for a given frequency of motor speed. The data is plotted over a specific length of time for particular frequencies with different conditions of load, namely:- 1)No load 2)Light load and 3) Heavy load Such data can be uniquely analysed using techniques analogous to time series forecasting. These techniques help us to precisely determine the outcome of a given parameter corresponding to the time stamp. Formally a time series data can be seen as:- A series of data points indexed in time order. Most commonly, a time series is a sequence taken at successive equally spaced points in time. Thus it is a sequence of discrete-time data.

3.2 Advantages

Time series analysis has various benefits for the data analyst. From cleaning data to understanding it — and helping to forecast future data points — this is all achieved through the application of various time series models, which we'll touch on later. The first benefit of time series analysis is that it can help to clean data. This makes it possible to find the true "signal" in a data set, by filtering out the noise. This can mean removing outliers, or applying various averages so as to gain an overall perspective of the meaning of the data. Another benefit of time series analysis is that it can help an analyst to better understand a data set. This is because the models used in time series analysis help to interpret the true meaning of the data.

3.3 Models for Time series

There are a number of models that can be used to describe and predict data points in a time series. Below mentioned are two of the most basic models: moving averages and exponential smoothing.

3.3.1 Moving averages

A moving average model suggests that an upcoming data point will be equal to the average of past data points. This rudimentary model is powerful in smoothing out data sets so as to observe their overall trend, with little regard for outlying data points. However, it may smooth out the seasonality of some time series.

3.3.2 Exponential smoothing

Exponential smoothing is another model where upcoming data points are predicted based on an exponentially decreasing average of past data points. It's said to be preferable to a moving average model in time series where there is no clear trend or pattern.

3.4 Remarks

Time series analysis is an advanced area of data analysis that focuses on processing, describing, and forecasting time series, which are time-ordered datasets. There are numerous factors to consider when interpreting a time series, such as autocorrelation patterns, seasonality, and stationarity. As a result, a number of models may be employed to help describe time series, including moving averages and exponential smoothing models. More advanced time series analysis models, which have not been discussed in this article, can be used to predict time series behavior with greater accuracy.

Chapter 4

Deep Learning Models

A brief overview of the different models to be used for the task of learning faults from the Induction motor.

4.1 Multilayer Perceptron

A multilayer perceptron (MLP) is a class of feedforward artificial neural networks (ANN). The term MLP is used ambiguously, sometimes loosely to any feedforward ANN, sometimes strictly to refer to networks composed of multiple layers of perceptrons (with threshold activation). Multilayer perceptrons are sometimes colloquially referred to as "vanilla" neural networks, especially when they have a single hidden layer.
An MLP consists of at least three layers of nodes: an input layer, a hidden layer and an output layer. Except for the input nodes, each node is a neuron that uses a nonlinear activation function. MLP utilizes a supervised learning technique called backpropagation for training. Its multiple layers and non-linear activation distinguish MLP from a linear perceptron. It can distinguish data that is not linearly separable.[3]

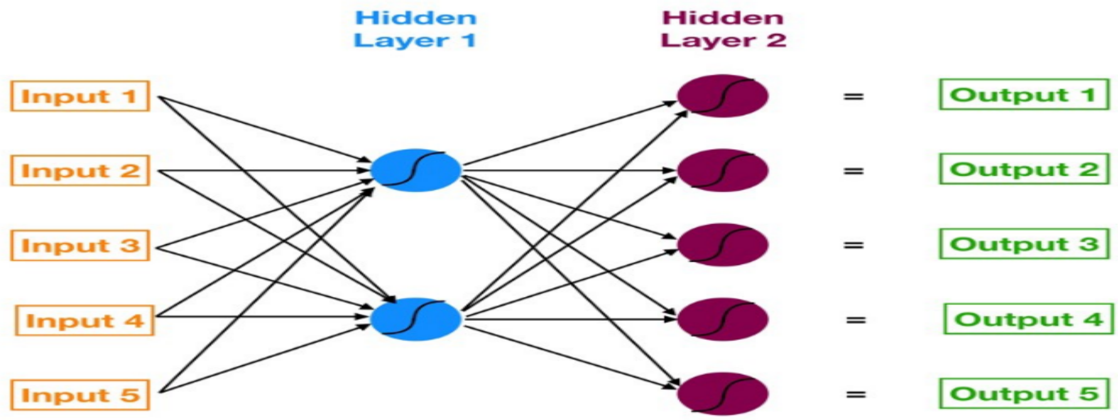

Fig. 4.1: Multilayer Perceptron Image, with 5 input neurons, 2 neurons in first hidden layer and output consists of 5 neurons

4.2 Convolutional Neural Network

In deep learning, a convolutional neural network (CNN, or ConvNet) is a class of deep neural networks, most commonly applied to analyzing visual imagery.They are also known as shift invariant or space invariant artificial neural networks (SIANN), based on their shared-weights architecture and translation invariance characteristics.They have applications in image and video recognition, recommender systems, image classification, medical image analysis, natural language processing, brain-computer interfaces, and financial time series.

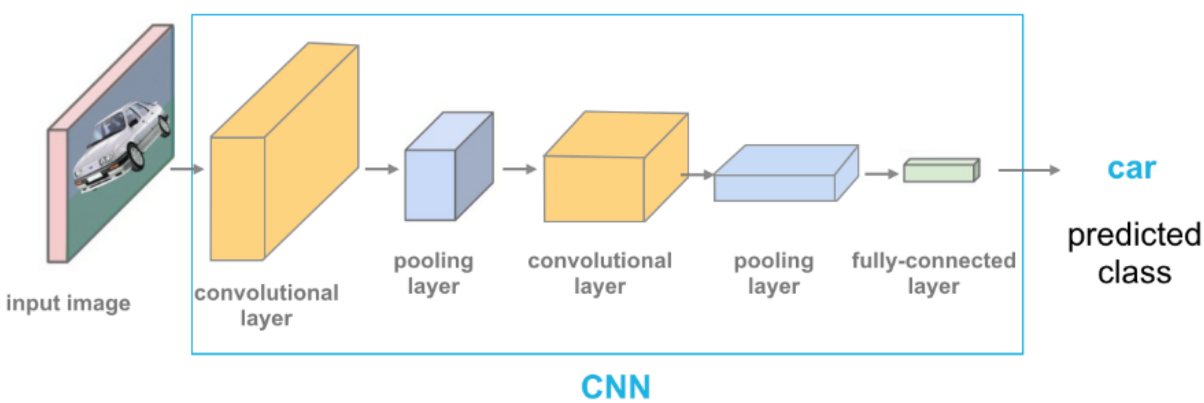

Fig. 4.2: A typical CNN architecture for classification task

4.3 Recurrent Neural Network

The recurrent neural network takes one step further in structuring the data and learning from it. Uses temporal coherence that is the sequence in which the data is, to predict the next label in the time sequence. RNNs are mostly used for speech recognition, predicting the next probable word, autocomplete features and for sequence prediction problems. Input for RNN is the raw time data instead of the domain for frequency level features. With regard to our data set, we intend to derive results by making use of the temporal sequence of data sets.

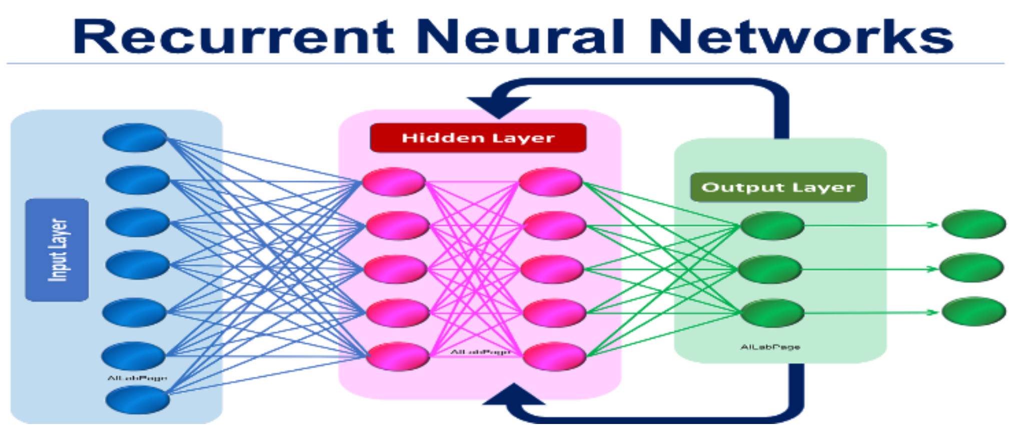

Fig. 4.3: Recurrent Neural Network has the feedback loop suggesting that it is used in sequential data

Chapter 5

Implementation

We take a look at the various libraries that were required to implement our solution. It is important to understand the particular benefits of every tool and how they aid us in a quick, efficient and accurate implementation.

5.1 Libraries used

5.1.1 Numpy

NumPy is the fundamental package for scientific computing with Python. It allows creation and manipulation of n-d arrays and tools for using linear algebra, Fourier transform capabilities. NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

5.1.2 Scikit learn

Scikit Learn has different machine learning capabilities ranging from data collection, storage, washing, rectification, algorithms such as SVMs, gradient boosting, decision trees, clustering of K-means, etc. The functionality of scientific programming libraries such as Numpy, Matplotlib and Scipy is successfully used.

5.1.3 Pandas

Pandas offers scientific computing tools for data structuring and interpretation designed for the Python programming language. It provides time series and table data manipulation operations. It is available free of charge and is published under the BSD Licence.

5.1.4 Tensorflow

Tensorflow is a machine learning system that is open source. It has a robust, scalable ecosystem of software, libraries and community services that enables researchers to quickly create and deploy ML-powered applications to push the state-of-the-art in ML and developers. It is used in a variety of functions for dataflow and segregated programming. Issued under licence from Apache.

5.1.5 Matplotlib

Matplotlib is a plotting library that can be used across platforms within python, IPython notebooks, online apps and UI toolkits, generating statistics of publishing consistency in a range of hardcopy formats and immersive environments. With only a few lines of code, it can generate maps, histograms, power spectra, bar charts, error charts, scatterplots, etc.

5.1.6 Keras

Keras is a Python-written, high-level neural network API capable of running on top of TensorFlow, CNTK, or Theano. It was designed with an emphasis on making it easier to experiment rapidly. It makes prototyping simple and fast (through user friendliness, modularity, and extensibility). As well as variations of the two, it serves both traditional networks and recurring networks and runs smoothly on the CPU and GPU.

5.2 Pre-processing

The data is cleaned by using the method analogous to the one used while creating the dataset for time series analysis. The code snippet given below is a typical way of data mining in time series data. With this method, the data for all frequencies is cleaned and compiled in such a way that the further code will run smoothly without any anomalies.
After removal of the metadata we find that there are 11 rows that are related to the description for 1 second of operation containing 2000 samples and 300 such sets exist corresponding to 5 minutes of operation.
At the end of preprocessing, the data was free of text-containing rows and stored as a tabular csv file to be used by the machine learning model for learning.

5.3 Data Visualization

We generated a few plots to visualize and better understand the data at hand. A heat map (or heatmap) is a graphical representation of data where values are depicted by color. Heat maps make it easy to visualize complex data and understand it at a glance Furthermore we plotted a few graphs depicting the variation of various parameters for both the classes namely No Defect(shown in red) and Defect(shown in green).

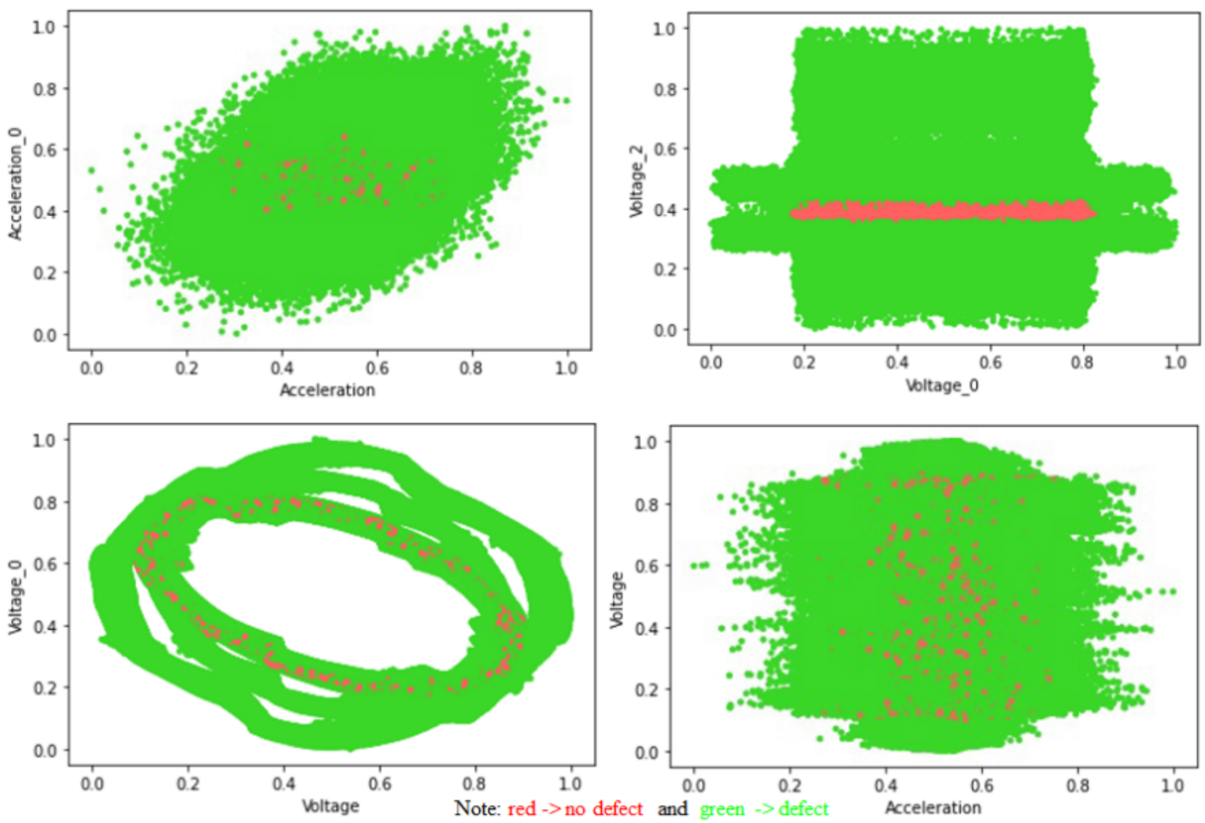

Fig. 5.1: Graphs depicting corelation between voltage and acceleration

Chapter 6

Results and Conclusion

The model shows high accuracy (approx. 98%) for 5 out of 7 different speeds of motor. The data is compiled in such a way that there are 7 faults and 1 non-defect. This leads to the model getting trained to predict a single outcome more number of times.

The higher accuracy can be explained by the fact that the non-defect rows are less in number to defect rows and thus the model improves upon its loss getting the accuracy close to 100%.

Speed(Hz)	Loss	Accuracy
10	0.0302	0.9881
15	0.0066	0.9988
20	0.3477	0.8894
25	0.0061	0.9985
30	0.0055	0.9989
35	0.3478	0.8888
40	0.0051	0.9990

Table 6.1: Accuracy for different speeds

Chapter 7

Limitations and Scope: BTP Sem2

The current model is focused on binary classification where the algorithm tries to predict, given a set of accelerometer and voltage readings the correct label as fault or no fault. Our model doesn't take in account the skewness of our dataset. We intend to add metrics in the future to take this into account. We aim to further extend the implementation of algorithms to convolutional neural networks and recurrent neural networks. Once the baseline accuracy for all the defects in binary classification problem is obtained hyperparameter tuning and optimisation by tweaking learning rate regularization, batch normalisation and momentum features to maximize the learning accuracy. The research over the next semester will be aimed towards multiclass classification that is to predict different labels of faults given the accelerometer and voltage readings.

Chapter 8

Multi Class Classification

This method of classification aims to tell which type of fault is detected, as opposed to just telling whether a fault is detected or not which was done in binary classification. The data label which formerly included only 0 or 1 to indicate for no-defect or faulty,now includes various labels each corresponding to their respective faults. The data given is missing a column in the Phase unbalance and single phasing fault with low resistance type of fault. Similarly another dataset has a column named Voltage3 missing.

We tried to solve this problem using basic neural net, CNN and LSTM. The data used in this classification comprises of all frequencies.

8.1 Neural Network

In this model we have 4 fully connected hidden layers and an output layer, alongwith 2 batch-normalization layers. The input data which goes into the model represents a specific data point corresponding to a particular fault. The input data has 9 features comprising of various voltages and accelerations. We have also added rotational speed of the motor as another feature so that our model is able to predict for any class with just a single training.

The output that we take is one-hot encoded vector depicting 11 classes(10 faults and 1 no defect) each corresponding to the faults.

One-hot encoder:- Encode categorical features as a one-hot numeric array. The input to this transformer should be an array-like of integers or strings, denoting the values taken on by categorical (discrete) features. The features are encoded using a one-hot encoding scheme. This creates a binary column for each category and returns a sparse matrix or dense array (depending on the sparse parameter).

Each dense layer has activation function as ReLU.The rectified linear activation function or ReLU for short is a piecewise linear function that will output the input directly if it is positive, otherwise, it will output zero.

The neural net also contains a layer for regularization. The values need to be regularized so that the loss calculated is not biased with the magnitude of the column. Since the values in a column can be from any range between its whole maximum to minimum, it is important that we regularise or average out each column with respective values.

The last output layer has activation function of softmax. Softmax is a mathematical function that converts a vector of numbers into a vector of probabilities, where the probabilities of each value are proportional to the relative scale of each value in the vector.

The model uses categorical cross-entropy as its loss function. The purpose of loss functions is to compute the quantity that a model should seek to minimize it during training. Categorical crossentropy is a loss function that is used in multi-class classification tasks. It is designed to quantify the difference between two probability distributions.

8.2 CNN

CNN is a deep neural network originally designed for image analysis. Recently, it was discovered that the CNN also has an excellent capacity in sequential data analysis such as natural language processing. CNN always contains two basic operations, namely convolution and pooling. Since the data we have has missing columns, we pre-processed the data in two ways.

1. Type-1: By completely removing the particular feature from the entire dataset. The input data is taken by accumulating consecutive 100 datapoints and is then given to the model as matrix. Therefore the input shape of the matrix is 100 x 9.

2. Type-2: By removing that particular fault from the entire dataset. The input data in this is taken concatenating the 100x10 matrices and using this as one input datapoint. The model in this has similar structure to the previous one.

We have implemented two different kinds of CNN:-

8.2.1 CNN-I

This CNN has 8 layers and takes input of dimensions 100X9 depicting 100 consecutive datapoints having 9 features in total. The layers perform convolution as well as max-pooling for the first 6 layers. The last two layers are fully connected dense layers. The output is the final one-hot encoded vector depicting all the 11 classes.

The convolution operation using multiple filters is able to extract features (feature map) from the data set, through which their corresponding spatial information can be preserved.

The pooling operation, also called subsampling, is used to reduce the dimensionality of feature maps from the convolution operation. Max pooling and average pooling are the most common pooling operations used in the CNN.

Due to the complexity of CNN, relu is the common choice for the activation function to transfer gradient in training by backpropagation.

The optimizer used in this model is the Adam optimzer. Adam is a replacement optimization algorithm for stochastic gradient descent for training deep learning models that can handle sparse gradients on noisy problems.

8.2.2 CNN-II

This CNN has 8 layers and takes input of dimensions 100X90 depicting 100 consecutive datapoints having 9 features in total, but also concatenated horizontally 10 times. This is done to increase the trainable parameters and also to make the model symmetric for better performance whilst training. The layers perform convolution as well as max-pooling for the first 6 layers. The last two layers are fully connected dense layers. The output is the final one-hot encoded vector depicting all the 11 classes.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 32)                320
_________________________________________________________________
dense_1 (Dense)              (None, 32)                1056
_________________________________________________________________
batch_normalization (BatchNo (None, 32)                128
_________________________________________________________________
dense_2 (Dense)              (None, 16)                528
_________________________________________________________________
dense_3 (Dense)              (None, 16)                272
_________________________________________________________________
batch_normalization_1 (Batch (None, 16)                64
_________________________________________________________________
dense_4 (Dense)              (None, 11)                187
=================================================================
Total params: 2,555
Trainable params: 2,459
Non-trainable params: 96
_________________________________________________________________
```

Fig. 8.1: Neural network model summary

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_3 (Conv2D)            (None, 100, 9, 32)        320
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 50, 4, 32)         0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 50, 4, 64)         18496
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 25, 2, 64)         0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 25, 2, 64)         36928
_________________________________________________________________
flatten_1 (Flatten)          (None, 3200)              0
_________________________________________________________________
dense_2 (Dense)              (None, 64)                204864
_________________________________________________________________
dense_3 (Dense)              (None, 11)                715
=================================================================
Total params: 261,323
Trainable params: 261,323
Non-trainable params: 0
_________________________________________________________________
```

Fig. 8.2: CNN-I model summary

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 100, 90, 32)       320
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 50, 45, 32)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 50, 45, 64)        18496
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 25, 22, 64)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 25, 22, 64)        36928
_________________________________________________________________
flatten (Flatten)            (None, 35200)             0
_________________________________________________________________
dense (Dense)                (None, 64)                2252864
_________________________________________________________________
dense_1 (Dense)              (None, 11)                715
=================================================================
Total params: 2,309,323
Trainable params: 2,309,323
Non-trainable params: 0
_________________________________________________________________
```

Fig. 8.3: CNN-II model summary

8.3 LSTM

Long Short Term Memory networks – usually just called "LSTMs" – are a special kind of RNN, capable of learning long-term dependencies. LSTMs are explicitly designed to avoid the long-term dependency problem.

Since the data is sequential, using LSTM is a viable approach to correlate and find patterns over a length of datapoints.

The model structure is similar to that used in the CNN above with the layers of convolution and pooling replaced with layers of lstm.

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (None, 100, 256)          271360
_________________________________________________________________
lstm_1 (LSTM)                (None, 64)                82176
_________________________________________________________________
dropout (Dropout)            (None, 64)                0
_________________________________________________________________
dense (Dense)                (None, 64)                4160
_________________________________________________________________
dense_1 (Dense)              (None, 11)                715
=================================================================
Total params: 358,411
Trainable params: 358,411
Non-trainable params: 0
_________________________________________________________________
```

Fig. 8.4: LSTM model summary

Chapter 9

Results and Discussion

9.1 Neural Network

The neural network model gave the following results during training on validation data. The accuracy starts at 35% and gradually increases with each training uptil 60%. This is also when the model is trained at the optimum number of layers and neurons.

When the model was run on test data, it gave accuracy of 0.6112 or approximately 60%.

Epoch	Loss	Accuracy
1/25	1.6378	0.3506
12/25	1.0660	0.5820
25/25	1.0201	0.5991

Table 9.1: Model accuracy while training

9.2 CNN

The CNN-I model was run on 100X9 dataset. Accuracy gradually increased with each epoch by starting from 52% and stabilizing at 90% for CNN-1 dataset. The CNN-II yielded more accuracy compared to neural network.

Accuracy for CNN-II dataset too increased from 88% to 96%, giving slightly better results than that over the previous model.

Both these models gave similar high value of accuracy when tested:- 89% over CNN-I and 96% over CNN-II. Thus, accuracy was much higher for CNN models in comparison to NN model. And CNN-II performed better than CNN-I as was already assumed based on its structure.

The above results are documented in the table below:

Epoch	Loss	Accuracy
1/100	1.1851	0.5251
50/100	0.4872	0.8237
100/100	0.3353	0.8964

Table 9.2: Model accuracy while training(CNN-1)

Epoch	Loss	Accuracy
1/100	0.2665	0.8888
50/100	0.2491	0.9588
100/100	0.3481	0.9632

Table 9.3: Model accuracy while training(CNN-2)

	Accuracy
CNN-1	0.8964
CNN-2	0.9632

Table 9.4: Accuracy on test data

9.3 LSTM

Results with the LSTM also showed high accuracy over training as well as testing data. The accuracy on test data is 0.9653 0r 96%. This is slightly lesser than that of the CNN. In NN, the f1-score is low for labels 0-6, indicating that the model is not performing

Epoch	Loss	Accuracy
1/100	0.7734	0.6621
50/100	0.1046	0.9562
100/100	0.0770	0.9653

Table 9.5: LSTM accuracy while training

better in predicting those values.

Label	Precision	Recall	f1-Score
0	0.47	0.33	0.39
1	0.71	0.63	0.67
2	0.47	0.51	0.49
3	0.41	0.48	0.44
4	0.52	0.60	0.56
5	0.45	0.33	0.38
6	0.34	0.50	0.41
7	0.91	0.93	0.92
8	0.92	0.90	0.91
9	0.82	0.70	0.76
10	0.87	0.85	0.86

Table 9.6: Classification report for test on NN model

Label	Precision	Recall	f1-Score
0	0.99	1.00	0.99
1	1.00	1.00	1.00
2	0.99	0.99	0.99
3	1.00	1.00	1.00
4	1.00	1.00	1.00
5	0.96	0.99	0.98
6	1.00	0.96	0.98
7	0.95	0.94	0.95
8	0.94	0.94	0.94
9	0.93	0.89	0.91
10	0.88	0.93	0.90

Table 9.7: Classification report for test on LSTM model

In LSTM,the precision, recall and f1-score values for all the labels are close to 1. Similar confusion matrix with high precision, recall and f1-score values is recorded for the CNN model.

Chapter 10

Conclusion and Futurework

Basic neural network gave accuracy much lesser than CNN and LSTM. CNN is specifically used for heavier datasets such images. So in comparison to a simple NN, CNN performs much better as the given data is large(4498000 rows X 10 columns). LSTM is a type of RNN which helps in processing this sequential data and thus performs even better.

The models designed so far have been compiled on a specific set of learning rate, number of epochs, hidden layers and neurons. They have not been trained for different set of such parameters and so we can still do hyper-parameter tuning to gain more insights into our models.

The model has been trained on discrete set of frequency values. We can further collect data having frequency as a time variable parameter and use the same to train our model for better predictions.

Since we got one of best accuracies for LSTM model, which is a sequential model, we can train on state-of-the-art sequential models such as BERT(Bidirectional Encoder Representations from Transformers) [1] to further improve the performance of the model.

References

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[2] Purushottam Gangsar and Rajiv Tiwari. Diagnostics of mechanical and electrical faults in induction motors using wavelet-based features of vibration and current through support vector machine algorithms for various operating conditions. *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, 41(2):71, 2019.

[3] Matt W Gardner and SR Dorling. Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. *Atmospheric environment*, 32(14-15):2627–2636, 1998.

[4] Yong Oh Lee, Jun Jo, and Jongwoon Hwang. Application of deep neural network and generative adversarial network to industrial maintenance: A case study of induction motor fault detection. In *2017 IEEE international conference on big data (big data)*, pages 3248–3253. IEEE, 2017.

[5] Ngoc-Tu Nguyen and Hong-Hee Lee. An application of support vector machines for induction motor fault diagnosis with using genetic algorithm. In *International Conference on Intelligent Computing*, pages 190–200. Springer, 2008.

[6] Abdulkarim Shaeboub. *The Monitoring of Induction Machines Using Electrical Signals from the Variable Speed Drive*. PhD thesis, University of Huddersfield, 2018.

[7] Peiming Shi, Kai Liang, Dongying Han, and Ying Zhang. A novel intelligent fault diagnosis method of rotating machinery based on deep learning and pso-svm. *Journal of Vibroengineering*, 19(8):5932–5946, 2017.

[8] Pan-Pan Wang, Xiao-Xiao Chen, Yong Zhang, Yong-Jun Hu, and Chang-Xin Miao. Ibpso-based music algorithm for broken rotor bars fault detection of induction motors. *Chinese Journal of Mechanical Engineering*, 31(1):1–10, 2018.

[9] Shen Zhang, Shibo Zhang, Bingnan Wang, and Thomas G Habetler. Deep learning algorithms for bearing fault diagnostics-a review. In *2019 IEEE 12th International Symposium on Diagnostics for Electrical Machines, Power Electronics and Drives (SDEMPED)*, pages 257–263. IEEE, 2019.