MOTOR FAULT DETECTION
USING
NEURAL NETWORKS

Master Thesis

Zehra ŞAHİN

Eskişehir 2018

MOTOR FAULT DETECTION
USING
NEURAL NETWORKS

Zehra ŞAHİN

Master Thesis

Electrical and Electronics
Engineering Program
Supervisor: Assoc. Prof. Dr. Emin GERMEN

Eskişehir
Anadolu University
Graduate School of Sciences
August 2018

FINAL APPROVAL FOR THESIS

This thesis titled “**Motor Fault Detection Using Neural Networks**” has been prepared and submitted by **Zehra ŞAHİN** in partial fullfillment of the requirements in “Anadolu University Directive on Graduate Education and Examination” for the Degree of Master of Science in **Electrical and Electronics Engineering Department** has been examined and approved on 13/08/2018.

Committee Members		**Title Name Surname**	**Signature**
Member (Supervisor)	:	**Assoc. Prof. Dr. Emin GERMEN**	....
Member	:	**Prof. Dr. Atakan DOĞAN**	....
Member	:	**Prof. Dr. Rıfat EDİZKAN**	....

Prof.Dr. Ersin YÜCEL
Director of Graduate School of Sciences

ABSTRACT

MOTOR FAULT DETECTION
USING
NEURAL NETWORKS

Zehra ŞAHİN

Department of Electrical and Electronics Engineering

Anadolu University, Graduate School of Sciences, August 2018

Supervisor: Assoc. Prof. Dr. Emin GERMEN

Since the frequent breakdown of induction motors is a case of industry, the detection of faults is of great importance in order to protect motors and not interrupt vital processes. In this thesis, vibration data from asynchronous motors which are under different loading conditions are classified by using convolutional neural networks. In order to create vibration data, different fault types were deliberately made on asynchronous motors and dataset was created by experimenting with different loading values. The one-dimensional vibration data is transformed into two-dimensional grayscale images and then three-dimensional color images using autocorrelation values, thereby allowing the convolutional neural networks to recognize the vibration data. Different motor faults can be distinguished easily thanks to the convolutional neural network which do not need any feature extraction method.

Keywords: Induction motors, Convolutional Neural Networks, Fault detection

ÖZET

SİNİR AĞLARI KULLANILARAK MOTOR ARIZALARININ BELİRLENMESİ

Zehra ŞAHİN

Elektrik-Elektronik Mühendisliği Anabilim Dalı

Anadolu Üniversitesi, Fen Bilimleri Enstitüsü, Ağustos 2018

Danışman: Assoc. Prof. Dr. Emin GERMEN

Asenkron motorlarının sık arızalanması endüstride karşılaşılan bir durum olmasından dolayı, hataların tespiti hem motorları korumak, hem de hayati önem arz eden süreçleri bölmemek adına büyük önem taşır. Bu tezde farklı yükleme koşulları altındaki asenkron motorlardan gelen titreşim verileri evrişimsel sinir ağları kullanılarak sınıflandırılmıştır. Titreşim verilerini oluşturmak adına, asenkron motorların üzerine farklı arıza tipleri kasıtlı olarak yapılmış ve farklı yükleme değerleriyle deney yapılarak veriseti oluşturulmuştur. Bir boyutlu titreşim verileri otokorelasyon değerleri kullanılarak önce iki boyutlu gri renkli imgelere sonra üç boyutlu renkli imgelere dönüştürülmüş, böylece sinir ağlarının titreşim verilerini tanımasına olanak sağlanmıştır. Evrişimsel sinir ağları sayesinde, herhangi bir özellik çıkarma yöntemine gerek duyulmadan farklı motor hataları kolaylıkla ayırt edilmiştir.

Anahtar Kelimeler: Asenkron motorlar, Evrişimsel Sinir Ağları, Arıza tespiti

To my mother, father and brother

ACKNOWLEDGEMENTS

I would like to express my gratefulness to my advisor, Assoc. Prof. Dr. Emin Germen, not only his technical support, continuous guidance, and advice but also his patience and moral support during the course of my research work. I would like to express all my thanks to him.

Zehra ŞAHİN

August, 2018

13/08/2018

STATEMENT OF COMPLIANCE WITH ETHICAL PRINCIPLES AND RULES

I hereby truthfully declare that this thesis is an original work prepared by me; that I have behaved in accordance with the scientific ethical principles and rules throughout the stages of preparation, data collection, analysis and presentation of my work; that I have cited the sources of all the data and information that could be obtained within the scope of this study, and included these sources in the references section; and that this study has been scanned for plagiarism with “scientific plagiarism detection program” used by Anadolu University, and that “it does not have any plagiarism” whatsoever. I also declare that, if a case contrary to my declaration is detected in my work at any time, I hereby express my consent to all the ethical and legal consequences that are involved.

Zehra ŞAHİN

TABLE OF CONTENTS

TITLE PAGE………………………………………………………………. i

FINAL APPROVAL FOR THESIS………………………………………. ii

ABSTRACT ……………………………………………………………... iii

ÖZET …………………………………………………………………….. iv

ACKNOWLEDGEMENTS ……………………………………………….. vi

STATEMENT OF COMPLIANCE WITH ETHICAL PRINCIPLES AND RULES ……………………………………………………………….. vii

TABLE OF CONTENTS …………………………………………………. viii

LIST OF FIGURES ………………………………………………………. xi

LIST OF TABLES ………………………………………………….......... xiv

LIST OF ACRONYMS ……………………………………………........... xviii

1. INTRODUCTION ……………………………………………………… **1**

1.1. Literature Overview ………………………………………………….. 2

1.2. Thesis Outline ………………………………………………………... 4

2. INDUCTION MOTORS AND DEFINED FAULTS ……………………. **5**

2.1. Induction Motors …………………………………………………….. 5

2.2. Defined Induction Motor Faults ……………………………………… 6

2.2.1 Bearing faults ………………………………………………….... 7

2.2.2 Broken rotor faults …………………………………………….... 8

2.3. Experimental Setup …………………………………………………... 9

2.3.1. Receiving of Vibration Data ……………………………………… 10

3. CONVOLUTIONAL NEURAL NETWORKS ………………………….. 13

3.1. Convolution Layer ………………………………………………………. 16

3.2. Rectifier Linear Unit ……………………………………………………. 21

3.3. Pooling Layer ...…………………………………………………………. 22

3.4. Fully Connected Layer ………………………………………………….. 23

3.5. Output Layer…………………………………………………………….. 24

4. FAULT CLASSIFICATION ……………………………………………… 29

4.1. Formation of Dataset ……………………………………………………. 29

4.1.1. Formation of dataset from 1D vibration data …………………….. 30

4.1.1.1 2D Image representation from 1D vibration data……….... 31

4.1.1.2 3D Image representation from combination of 1D vibration data …………………………………………….. 45

4.2. Classification Result …………………………………………………….. 48

4.2.1. Classification results for images from x-axis of the vibration dataset ………………………………………………………………. 53

4.2.1.1. Classification results for the size of 40x40 images from the x-axis of the vibration dataset……………………….. 53

4.2.1.2. Classification results for the size of 50x50 images from the x-axis of the vibration dataset………………………... 56

4.2.2. Classification results for images from y-axis of the vibration dataset ………………………………………………………… 59

4.2.2.1. Classification results for the size of 40x40 images from the y -axis of the vibration dataset …………………………... 59

4.2.2.2. Classification results for the size of 50x50 images from the y -axis of the vibration dataset…………………………... 62

4.2.3. Classification results for images from z-axis of the vibration dataset ……………………………………………………………….. 65

4.2.3.1. Classification results for the size of 40x40 images from the z -axis of the vibration dataset …………………………... 65

4.2.3.2. Classification results for the size of 50x50 images from the z -axis of the vibration dataset …………………………..... 68

4.2.4. Classification results for color images from of the vibration dataset ……………………………………………………………. 71

4.2.2.1. Classification results for the size of 40x40 color images of the vibration dataset…………………………………………...... 71

4.2.2.2. Classification results for the size of 50x50 color images of the vibration dataset …………………………………………. 74

5. CONCLUSION ……………………………………………………………….. 78

REFERENCES ……………………………………………………………….. 82

RESUME

LIST OF FIGURES

Figure 2.1. Parts of squirrel-cage induction motor ……………………………… 6

Figure 2.2. Faulty and unused bearings respectively ………………………… 7

Figure 2.3. Substantiation broken rotor bar with 5 holes …………………….. 8

Figure 2.4. Metra Mess M108 ……………………………………………... 10

Figure 2.5. Accelerometer and its position ……………………………………… 10

Figure 2.6. Experimental Setup for vibration and acoustic data ……………... 11

Figure 3.1. A simple artificial neural network, consisting of an input layer, an output layer and two hidden layers (Altenberger and Lenz, 2018). 14

Figure 3.2. Architecture of LeNet-5 (LeCun et al.,1998) …………………….. 15

Figure 3.3. An example of 2D Convolution Operation ……………………… 17

Figure 3.4. The operation of the convolution layer (Guoa et al., 2016) ……… 17

Figure 3.5. Sparse connectivity, viewed from below. x_3 is highlighted one input unit, and units are hihglighted the output units in s that are affected by this unit (Goodfellow et al., 2016)…………………. 19

Figure 3.6. Parameter sharing. Black arrow shows the connections that use particular parameter in two distinct models (Goodfellow et al., 2016)…………………………………………………………… 19

Figure 3.7. A visualization of the different hyperparameters of convolutional layers:receptive field, stride and padding (Altenberger and Lenz, 2018)…………………………………………………………… 20

Figure 3.8. The ReLU Operation …………………………………………… 21

Figure 3.9. The example of max pooling (Guoa et al., 2016)…………………. 22

Figure 3.10. The examples of pooling ……………………………………….. 23

Figure 3.11. The operation of Fully Connected Layer (Guoa et al., 2016)……... 24

Figure 3.12. (a) Standard neural net (b) After applying dropout (Srivastava et al., 2014) …………………………………………………….. 26

Figure 3.13. (a) SGD without momentum (b) SGD with momentum (Ruder, 2018) ……………………………………………………………… 27

Figure 3.14. The summary of CNN architecture (Peng et al., 2016) ………….. 28

Figure 4.1. The directions of the vibration data ……………………………… 30

Figure 4.2. The first 1000 vibration data from x- axis of (a) Healthy (b) BFE (c) 5BB (d) BFA (e) 3BB faulty induction motor under 4.1 A …… 32

Figure 4.3. The first 1000 vibration data from y- axis of (a) Healthy (b) BFE (c) 5BB (d) BFA (e) 3BB faulty induction motor under 4.1 A …... 33

Figure 4.4. The first 1000 vibration data from z- axis of (a) Healthy (b) BFE (c) 5BB (d) BFA (e) 3BB faulty induction motor under 4.1 A …... 34

Figure 4.5. An Example of 1D to 2D Vibration Data ………………………… 35

Figure 4.6 Autocorrelation sequences of the vibration data recorded for five different induction motor under 4.1 Amperes load ……………….. 36

Figure 4.7. Autocorrelation sequences of the vibration data recorded for healthy induction motor under different loading conditions……… 37

Figure 4.8 Autocorrelation sequences of the vibration data recorded for BFA under different loading conditions ………………………………... 38

Figure 4.9. 2D Grayscale Images of the x-axis vibratiodata under 4.1 A …….. 40

Figure 4.10. 2D Grayscale Images of the y-axis vibratiodata under 4.1 A …….. 41

Figure 4.11. 2D Grayscale Images of the z-axis vibratiodata under 4.1 A …….. 42

Figure 4.12. 2D Grayscale Images of the x- axis vibration data from the healthy motor under (a) 3.6 A (b) 4.1 A (c) 4.7 A (d) 4.9 A (e) 5 A ……… 43

Figure 4.13. 2D Grayscale Images of the x- axis vibration data from 5BB under (a) 3.6 A (b) 4.1 A (c) 4.7 A (d) 4.9 A (e) 5 A …………………… 43

Figure 4.14. Multiple Image Formation from 2D Grayscale Image of healthy motor under 4.1 A ………………………………………………... 44

Figure 4.15. The formation of color image from the combination of three axes .. 45

Figure 4.16. 3D Color Images under 4.1 A ……………………………………... 46

Figure 4.17. 3D Color Images of BFA under (a)3.6 A (b) 4.1 A (c) 4.7A (d) 4.9 A (e) 5A ………………………………………………………… 47

Figure 4.18. 3D Color Images of 3BB under (a)3.6 A (b) 4.1 A (c) 4.7A (d) 4.9 A(e) 5 A ………………………………………………………… 48

Figure 4.19. (a) Train accuracy (b) Validation accuracy (c) Train loss (d) Validation loss for 40x40 Images of the X-axis of the vibration data classification after 50 epochs using different optimizers…… 54

Figure 4.20. (a) Train accuracy (b) Validation accuracy (c) Train loss (d) Validation loss for 50x50 Images of the X-axis of the vibration data classification after 50 epochs using different optimizers ……. 57

Figure 4.21. (a) Train accuracy (b) Validation accuracy (c) Train loss (d) Validation loss for 40x40 Images from Y-axis of the vibration data classification after 50 epochs using different optimizers ………… 60

Figure 4.22. (a) Train accuracy (b) Validation accuracy (c) Train loss (d) Validation loss for 50x50 Images from Y-axis of the vibration data classification after 50 epochs using different optimizers ………… 63

Figure 4.23. (a) Train accuracy (b) Validation accuracy (c) Train loss (d) Validation loss for 40x40 Images from Z-axis of the vibration data classification after 50 epochs using different optimizers ………… 66

Figure 4.24. (a) Train accuracy (b) Validation accuracy (c) Train loss (d) Validation loss for 50x50 Images from Z-axis of the vibration data classification after 50 epochs using different optimizers ………… 69

Figure 4.25. (a) Train accuracy (b) Validation accuracy (c) Train loss (d) Validation loss for 40x40 Color Images of the vibration data classification after 50 epochs using optimizers …………………... 71

Figure 4.26. (a) Train accuracy (b) Validation accuracy (c) Train loss (d) Validation loss for 40x40 Color Images of the vibration data classification after 300 epochs using different ……………………. 74

Figure 4.27. (a) Train accuracy (b) Validation accuracy (c) Train loss (d) Validation loss for 50x50 Color Images of the vibration data classification after 50 epochs using different optimizers ………… 75

Figure 4.28. (a) Train accuracy (b) Validation accuracy (c) Train loss (d) Validation loss for 50x50 Color Images of the vibration data Classification after 300 epochs using different optimizers ……….. 77

LIST OF TABLES

Table 2.1. List of faulty motors used for experimental setup ………………... 9

Table 2.2. Sound pressure levels of the experiments of the dataset ……….…. 12

Table 4.1. Autocorrelation Values of the vibration dataset ………………….. 39

Table 4.2. The number of samples in the dataset prepared for classification of the induction motor faults ……………………………………… 45

Table 4.3. Parameters tested for the finding best classification of the dataset . 50

Table 4.4. Summary of model of the CNN for the size of 40X40 of 2D Grayscale Images classification ………………………………….. 51

Table 4.5. Summary of model of the CNN for the size of 50X50 of 2D Grayscale Images classification…………………………………… 52

Table 4.6. Test Accuracy for 40x40 Images of X-axis of the vibration data classification after 50 epochs with Various Optimizers ……... 53

Table 4.7. Confusion Matrix for 40x40 Images of X-axis of the vibration data classification using SGD …………………………………… . 55

Table 4.8. Confusion Matrix for 40x40 Images of X-axis of the vibration data Classification using SGD with Nesterov ……………………. 55

Table 4.9. Confusion Matrix for 40x40 Images of X-axis of the vibration data classification using SGD with Momentum ………………….. 55

Table 4.10. Confusion Matrix for 40x40 Images of X-axis of the vibration data classification using SGD with Nesterov and Momentum …… 55

Table 4.11. Confusion Matrix for 40x40 Images of X-axis of the vibration data classification using Adam …………………………………… 56

Table 4.12. Confusion Matrix for 40x40 Images of X-axis of the vibration data classification using Nadam ………………………………….. 56

Table 4.13. Test Accuracy for 50x50 Images of X-axis of the vibration data after 50 epochs with Various Optimizers ……………………..... 57

Table 4.14. Confusion Matrix for 50x50 Images of X-axis of the vibration data classification using SGD …………………………………….. 58

Table 4.15. Confusion Matrix for 50x50 Images of X-axis of the vibration data classification using SGD with Nesterov …………………….. 58

Table 4.16. Confusion Matrix for 50x50 Images of X-axis of the vibration data classification using SGD with Momentum ……………….... 58
Table 4.17. Confusion Matrix for 50x50 Images of X-axis of the vibration data classification using SGD with Nesterov and Momentum …... 59
Table 4.18. Test Accuracy for 40x40 Images of y-axis of the vibration data classification after 50 epochs with Various Optimizers ……….... 59
Table 4.19. Confusion Matrix for 40x40 Images from Y-axis of the vibration data classification using SGD ………………………….. 60
Table 4.20. Confusion Matrix for 40x40 Images from Y-axis of the vibration data classification using SGD with Nesterov ………….. 61
Table 4.21. Confusion Matrix for 40x40 Images from Y-axis of the vibration data classification using SGD with Momentum ……….. 61
Table 4.22. Confusion Matrix for 40x40 Images from Y-axis of the vibration data classification using SGD with Nesterov and Momentum …………………………………………………….. 61
Table 4.23. Confusion Matrix for 40x40 Images from Y-axis of the vibration data classification using Adam……………………….... 61
Table 4.24. Confusion Matrix for 40x40 Images from Y-axis of the vibration data classification using Nadam……………………... 62
Table 4.25. Test Accuracy for 50x50 Images of y-axis of the vibration data classification after 50 epochs with Various Optimizers ……….... 62
Table 4.26. Confusion Matrix for 50x50 Images of Y-axis of the vibration data classification using SGD………………………………….. 63
Table 4.27. Confusion Matrix for 50x50 Images of Y-axis of the vibration data classification using SGD with Nesterov …………………….. 64
Table 4.28. Confusion Matrix for 50x50 Images of Y-axis of the vibration data classification using SGD with Momentum ……………….... 64
Table 4.29. Confusion Matrix for 50x50 Images of Y-axis of the vibration data classification using SGD with Nesterov and Momentum …... 64
Table 4.30. Test Accuracy for 40x40 Images of Z-axis of the vibration data classification after 50 epochs with Various Optimizers ……….... 65
Table 4.31. Confusion Matrix for 40x40 Images of Z-axis of the vibration data classification using SGD ………………………………….... 66

Table 4.32. Confusion Matrix for 40x40 Images of Z-axis of the vibration data classification using SGD with Nesterov …………………….. 67
Table 4.33. Confusion Matrix for 40x40 Images of Z-axis of the vibration data classification using SGD with Momentum ………………..... 67
Table 4.34. Confusion Matrix for 40x40 Images of Z-axis of the vibration data classification using SGD with Nesterov and Momentum …... 67
Table 4.35. Confusion Matrix for 40x40 Images of Z-axis of the vibration data classification using Adam ………………………………... 67
Table 4.36. Confusion Matrix for 40x40 Images of Z-axis of the vibration data classification using Nadam ……………………………….. 68
Table 4.37. Test Accuracy for 50x50 Images of Z-axis of the vibration data classification after 50 epochs with Various Optimizers ……….... 68
Table 4.38. Confusion Matrix for 50x50 Images of Z-axis of the vibration data classification using SGD ……………………………………. 69
Table 4.39. Confusion Matrix for 50x50 Images of Z-axis of the vibration data classification using SGD with Nesterov …………………….. 70
Table 4.40. Confusion Matrix for 50x50 Images of Z-axis of the vibration data classification using SGD with Momentum ………………… 70
Table 4.41. Confusion Matrix for 50x50 Images of Z-axis of the vibration data classification using SGD with Nesterov and Momentum …... 70
Table 4.42. Test Accuracy for 40x40 Color Images of the vibration data classification after 50 epochs with Various Optimizers ………… 71
Table 4.43. Confusion Matrix for 40x40 Color Images of the vibration data classification using SGD ………………………………………. 72
Table 4.44. Confusion Matrix for 40x40 Color Images of the vibration data classification using SGD with Nesterov …………………………. 72
Table 4.45. Confusion Matrix for 40x40 Color Images of the vibration data classification using SGD with Momentum ………………………. 72
Table 4.46. Confusion Matrix for 40x40 Color Images of the vibration data classification using SGD with Nesterov and Momentum ………... 72
Table 4.47. Confusion Matrix for 40x40 Color Images of the vibration data classification using Adam ……………………………………... 73

Table 4.48. Confusion Matrix for 40x40 Color Images of the vibration data classification using Nadam ………………………………………. 73
Table 4.49. Confusion Matrix for 40x40 Color Images of the vibration data classification after 300 epochs using SGD with Momentum and Nesterov ……………………………………………………………….. 73
Table 4.50. Test Accuracy for 50x50 Color Images of the vibration data classification after 50 epochs with Various Optimizers …………. 74
Table 4.51. Confusion Matrix for 50x50 Color Images of the vibration data classification using SGD …………………………………………. 76
Table 4.52. Confusion Matrix for 50x50 Color Images of the vibration data classification after using SGD with Nesterov ……………………. 76
Table 4.53. Confusion Matrix for 50x50 Color Images of the vibration data classification using SGD with Momentum ………………………. 76
Table 4.54. Confusion Matrix for 50x50 Color Images of the vibration data classification using SGD with Nesterov and Momentum ………... 76
Table 4.55. Confusion Matrix for 50x50 Color Images of the vibration data classification after 300 epochs using SGD with Momentum and Nesterov ……………………………………………………………….. 77

LIST OF ACRONYMS

AC	:	Alternating Current
AI	:	Artificial Intelligence
APEC	:	Asia Pasific Economic Cooperation
ARMA	:	Auto-Regressive Moving Average
ANN	:	Artificial Neural Network
BN	:	Batch Normalization
CNN	:	Convolutional Neural Network
CV	:	Computer Vision
ConvNet	:	Convolutional Neural Network
DL	:	Deep Learning
EPRI	:	Electric Power Research Institute
EMF	:	Electromotive Force
FC	:	Fully Connected
FT	:	Fourier Transform
GD	:	Gradient Descent
IEEE	:	Institute of Electrical and Electronics Engineers
IEPE	:	Integrated Electronic Piezoelectric
ML	:	Machine Learning
NLP	:	Natural Language Processing
NN	:	Neural Network
PCA	:	Principal Component Analysis
PL	:	Pooling Layer
ReLU	:	Rectified Linear Unit
RNN	:	Recurrent Neural Networks
RMF	:	Rotating Magnetic Field
SCIM	:	Squirrel Cage Induction Motor

SGD : Stochastic Gradient Descent

SOM : Self-Organizing Map

SPL : Sound Pressure Level

STFT : Short-Time Fourier Transform

SVM : Support Vector Machine

TDNN : Time-Delay Neural Network

1. INTRODUCTION

The energy in the world is mostly used in the industrial sector. In industrially developed nations, electrical motors responsible for a remarkable ratio of total national power consumption (R. Saidur, 2009). Thus, the expense of energy to operate electrical motors is a real concern for industries. In 2008, the consumption rate of electric motors was about 40 % of total national consumption (APEC, 2008). According to the research, electrical motors account for 65% of the electric consumption in industry sector in Turkey (Kaya et al., 2008). In general, electrical motor systems use about 70 % of the electricity in the industry in these days (Almeida et al., 2017). Among whole electrical motor types, squirrel cage induction motors (SCIMs) are the most preferred in the industry. Due to their strong mechanic, simple construction, low cost, self-starting mechanism and long lasting, the SCIMs have found wide applications in the industry. In order to facilitate and accelerate production lines in the industry, the use of SCIMs is increasing day after day (Ojaghi et al., 2018). Because of this rise in usage of SCIMs, the SCIMs are exposed to various stresses which may cause many faults such as magnetic, mechanical, thermal, and environmental stresses (Mehrjou and Mariun, 2011; Bindu and Tomas, 2014). If the incipent motor faults are not determined on time and removed, many undesirable conditions can occur, such as stopping the motor, reduction of efficiency and damage surrounding equipment or nearby people.Therefore, early detection of these faults is crucial and necessary in terms of vital situations and continuity of work.

In this thesis, critical SCIM faults are identified and classified by approaches based on Deep Learning (DL) where the features and the classifiers are together learned from the dataset. As a branch of Machine Learning (ML), DL has been successfully accepted in multiple areas such as computer vision, natural language processing, bioinformatics, audio and speech recognition (Ren et al., 2015; Collobert and Weston, 2008; Hinton et al., 2012; Leung et al., 2014). Up to date, there are many deep learning architectures such as Auto-Encoders (Vincent et al., 2008), Deep Belief Network (Hinton, Osindero and Teh, 2006), Convolutional Neural Networks (Sermanet, Chintala, and LeCun, 2012) and Recurrent Neural Networks (Funahashi and Nakamura, 1993). The success of these deep learning models has proven recently by various researchers in several applications. In this thesis, Convolutional Neural Networks (CNNs or ConvNets), which has been found to be well-suited for visual document analysis tasks (LeCun et al., 1998), have been the preferred method for identifying and classifying SCIM faults. Some researchers have shown that, CNN models are used as basis classifiers (Ciresan, 2011) and they have been demonstrated to learn robust and interpretable image features (Zeiler and Fergus, 2013). Encouraged by positive results in the field of images, the CNN has been used with peace of mind for detection of SCIM faults.

CNNs have demonstrated their achievement in many machine learning problems and computer vision applications such as Handwritten Digit Recognition, Object Recognition (Jarrett et al., 2009), Image Classification (Krizhevsky et al., 2012), Natural Language Processing (NLP) (Collobert et al., 2011; Kim, 2014) and Speech Recognition (Abdel-Hamid et al., 2012). Moreover, CNN models have been shown excellent results in Semantic Parsing (Yih et al., 2014), Search Query Retrieval (Shen et al., 2014) and Sentence Modeling (Kalchbrenner et al., 2014). In addition, the CNN architecture has changed and developed. According to this, there are many types of CNN architecture such as LeNet-5 (LeCun et al., 1998), AlexNet, ZFNet (Zeiler and Fergus, 2013), VGGNet (Simonyan and Zisserman, 2014), GoogLeNet (Szegedy et al., 2014), ResNet (He et al., 2015) and DenseNet (Haung et al., 2017).

1.1. Literature Overview

Asynchronous motors are preferred by the industry due to their usage areas such as blowers, centrifugal fans, pumps, lifts, conveyors, compressors, heavy duty cranes, lathe machines, oil or textile or paper mills. Asynchronous motors are called induction motors because of their working principle. In recent years, the responsibilities of induction motors have increased due to their ease of use and performance. Critical errors have started to appear in the induction motors in proportion to the use of them. These faults can be in places where they seriously affect the function of the induction motors like mechanical and electrical parts of the induction motor. As electrical-related faults are about imbalance supply current or voltage, under or over voltage of current, single phasing, reverse phase sequence, earth fault, overload, short-circuit fault, and crawling;

mechanical-related faults are broken rotor bar, mass unbalance, air gap eccentricity, bearing damage, rotor winding failure, and stator winding failure. Moreover, there is also some environmental-related faults of induction motors such as ambient temperature, external moisture, vibrations of a machine, due to any reason such as installation defect, foundation defect (Karmakar et al, 2016). Therefore, it becomes crucial early detection of possible induction motor faults. In the literature, motor current, vibration, temperature, speed and torque variations, and acoustic noise can be used in order to detect the induction motor faults (Benbouzid, 1999; Germen, Başaran and Fidan, 2014).

Detection of motor faults can be done using 3 major methods: model-based, signal-based and knowledge-based (Ince et al.,2016). In model-based methods, mathematical models, which describe the normal operating conditions of the induction motor, are used (Giantomassi, 2015). In order to monitor consistency, fault diagnosis algorithms are developed between the measured outputs and the predicted outputs (Gao et al.,2015). Next, in the signal-based methods, some signal processing techniques are used such as time-domain analysis and frequency-domain analysis. Famous techniques like Fourier Transform (FT), Short-Time Fourier Transform (STFT) and Auto-Regressive Moving Average (ARMA) can be used to extract features of the data coming from the motor for detection the motor faults. The features such as time domain information energy, local extrema, kurtosis and skewness parameters are used (Günal et al., 2009).

In addition to model-based and signal-based methods, knowledge-based methods are also cover many topics to try detection of the induction motor faults. In knowledge-based methods are both about the qualitative methods and quantitative methods. The qualitative methods which are on basis of symbolic intelligence are fault trees, diagraphs, and expert systems. In quantitative methods, a core of machine-learning intelligence, there are two types of learning systems which are called supervised and unsupervised. Nearest neighbor, principal component analysis (PCA), Self-organizing maps (SOMs), K-means and C-means are in unsupervised-learning systems which have input data and no corresponding output variables. The aim of unsupervised learning is learning more information about the data without any teacher and correct answer. Namely, all data is unlabeled in the unsupervised learning and the systems try to learn inherent structure from the input data. On the other hand, in supervised learning such as artificial neural networks (ANNs), fuzzy logic, support vector machines (SVMs), and hybrid systems; the system has input and the output data. In supervised learning, all data is labeled and the system predicts the output from the input data (Brownlee, 2016).

All the listed methods are selected and applied as appropriate to the fault of the induction motor. In literature, there are multiple research about the detection of induction motor faults like in Kowalski et al., Tung et al., Chen et al., and Ballal et al.

1.2. Thesis Outline

In this thesis, the induction motors and the errors of them to be used are explained in Chapter 2. In addition to this, the working principle and structure of the induction motor are given. Statistical information about the induction motor faults is given and the importance of the solution for the defined errors is grasped. Also, there is an explanation of how the data were taken as an experimental setup to create vibration dataset.

After, there are clear pieces of information about neural network (NN) and then how CNN works in Chapter 3. The layers which make up CNN are explained and information about their functions is given in turn. It is explained that the individual benefits of the parameters that must be set for the experiments.

In Chapter 4, pre-classification studies which are about converting one-dimensional vibration data into two-dimensional grayscale images and three-dimensional color images are given. Moreover, the importance of the correlation values needed to create the image is emphasized. After the pre-classification studies, classification results of the faulty induction motors are given by usage of CNN with changing model parameters. In order to understand which parameters are best suited for the CNN model, some experiments are done. Furthermore, vibration data, which has distinct axis information, are used in order to detect and classify the vibration data of the induction motor faults. Classification performance of different types of optimizers is given. In the last chapter comments on inferences of the thesis are made.

2. INDUCTION MOTORS AND DEFINED FAULTS

2.1. Induction Motors

Induction motor is a kind of asynchronous AC motor where is used many industrial plants. They play a significant role for continuity of an entire industrial process which is in the military, medical, commercial and transportation implementation (Wang, 2007).

Induction motors have multiple useful aspects. To illustrate, they do not require a starting mechanism or any additional power supply. They have also simple construction and low cost. Because of these positive characters, induction motors are preferred mostly in recent years. As it is highly preferred, the failure is unacceptable for vital situations. Thus, in order to prevent induction motor faults, there are many studies recently. Due to the challenging working environment, electrical and mechanical parts of the induction motor is failed especially in machine bearings, rotor bars, and stator windings.

An induction motor includes a magnetic circuit that connects the stator and rotor portion of the machine. As stator is the stationary part of the motor, rotor is the rotating part of it. Thanks to electromagnetic induction, power can pass between the stator and the rotor (Krause, 1986). There are two circuits which are an electric and a magnetic both in the stator and the rotor. As an electric circuit carries current and the magnetic circuit carries the magnetic flux. As it is shown from Figure 2.1. the stator is the outer stationary part of the motor and it consist of the outer cylindrical frame, the magnetic path, and a set of insulated electrical windings (Karkamar et al., 2016). Rotor is placed in the stator and rotates coaxially with the stator. Induction motor is named according to the type of rotor winding like squirrel-cage type induction motor and wound-rotor type induction motor. Due to the fact that the bars of the squirrel-cage type induction motor are made of copper or aluminum or alloy, such motors are very durable. In addition to these parts, there are bearings, end-flanges, shaft, cooling fan and terminal box.

After the stator winding of an induction motor is connected to a three-phase supply, a uniform rotating magnetic field is produced (Say, 2002). This induces electromotive force (EMF) in the rotor and thus, rotor is started to rotate coaxially with the stator core with the help of ball bearings. Due to the induced EMF, currents, which interacts with the magnetic field and thus forms a torque on the rotor, are produced. The torque, which in the direction of the rotating magnetic field ,will cause the rotor to move round in the direction of the stator field. This is the self-starting principle of the induction motors.

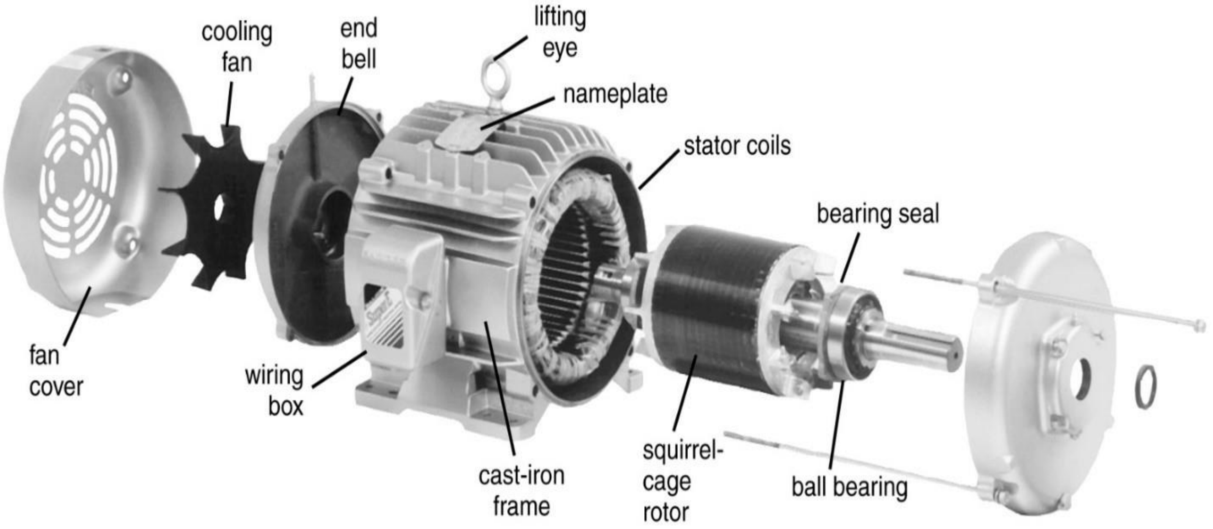


Figure 2.1. *Parts of squirrel-cage induction motor*

2.2. Defined Induction Motor Faults

The common faults of the induction motor occur mostly in bearings, rotor bars and stator which can stop processes that are vital when they come out. Statistics about this subject are made by IEEE (Institute of Electrical and Electronics Engineers) and EPRI (Electric Power Research Institute) (Zhang et al., 2011). In the shade of researches, the most common faults of the induction motor are bearing faults, broken rotor bars and stator winding faults. Since the faults in the induction motors suddenly take place, it is very difficult to detect and to interfere in the fault immediately. Due to these faults, the heat on the motor may rise, the vibration may increase and the torque oscillation may become unbalanced.

2.2.1. Bearing faults

Bearings, which are placed the ends of the rotor of the induction motor, help the rotor to rotate by decreasing the frictions. An inner and an outer ring (called races) are the parts of each bearing. In addition, there are balls in between these two races. Any damage about these parts is termed as bearing faults.

The bearing fault is the mechanical failure and the most common one. About 40-45% of failure in big industrial induction motors because of bearing faults as shown in IEEE and EPRI reports. Environmental conditions such as current distortion and local vibration can damage bearing components. If bearings run for a long time, fatigue failure which increase vibration and noise level is occured (Eschmann, 1958). Moreover, corrosion, contamination, lubricant failure, misalignment of bearings are the other types of the bearing faults, on which it is observed that increasing friction which causes rise in temperature and vibration. According to this, it can be said that temperature and the vibration of the bearings give necessary information about the bearing faults in order to detect and prevent them (Schoen et al., 1995). Since the bearings are an assistance of the rotor, early identification of the fault is of great importance (Gonçalves et al., 2015).

In this thesis, after taking the real faulty bearings which are used for a long time put into the motors in order to get bearing fault data. Figure 2.1 shows that faulty and unused bearings.

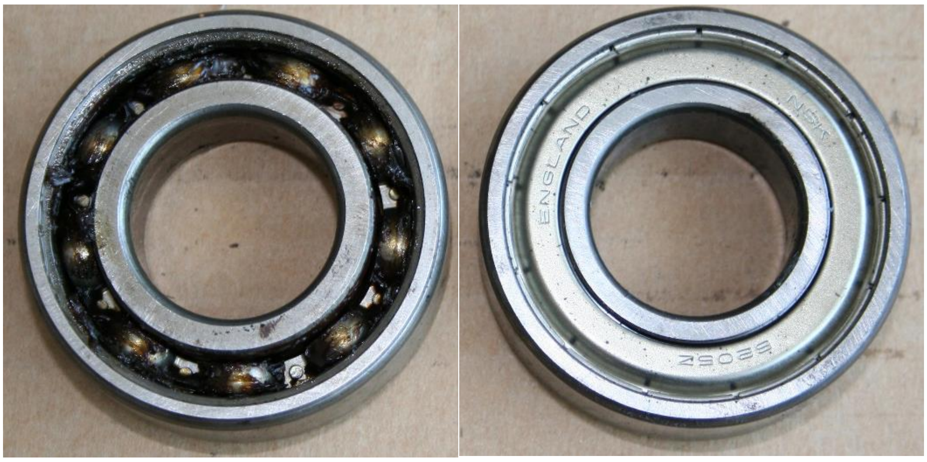


Figure 2.2. *Faulty and unused bearings respectively*

2.2.2. Broken rotor faults

Rotor Faults include at about 10% of all failure. There are two main causes of the rotor failures: broken rotor bar and end ring fault. Unless it can be detected in time it can cause excessing heat, decreasing efficiency or iron core damage. Thus, like bearing faults, early detection of rotor faults is very important to avoid more motor failure (Pons-Linares et al., 2010) (Saleh et al., 2005). In this thesis, broken rotor bars are discussed as rotor faults.

The induction motor broken rotor faults can be caused by many reasons such as manufacturing defect, thermal stresses, mechanical stress, frequent starts of the motor and due to fatigue of metal of the rotor bar.

If there is a broken bar, there is no current through the broken bar and so, there is no magnetic flux around the broken bar which causes asymmetry in the rotor magnetic field (Sonje and Munje, 2011). Rotating field at slip frequency with respect to the rotor is forward since the cage winding is symmetrical. Unfortunately, if asymmetry in the rotor magnetic field occurs, a backward rotating field at slip frequency will emerge. Thus, there will be a twice slip frequency sideband in the air gap flux which finally causes ripples in speed and torque (Finley et al., 1999).

In order to obtain the faults like broken rotor bar for this thesis, the rotor bars have been damaged by means of drill. Different types of faults and their results have to be scrutinized by drilling different amount of bars. Broken rotor bars with 3 holes or 5 holes are created for getting data. Figure 2.2 shows how is formed of the broken rotor bar with 5 holes.

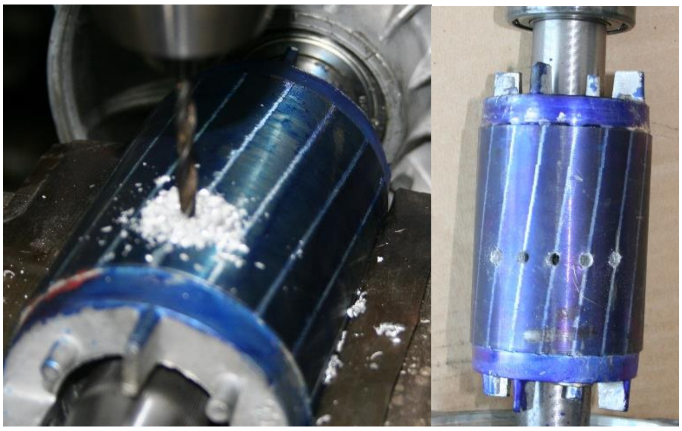


Figure 2.3. *Substantiation broken rotor bar with 5 holes*

2.3. Experimental Setup

Power Systems Laboratory of Anadolu University is used in order to construct a valuable database consisting of current, vibration and acoustic data. Thus, many different experiments are performed by induction motors supplied with the output of the isolation transformer located between AC network and the test rig which is self-designed. In the test rig, there is a single-phase permanent magnet synchronous generator acoupled to a test motor. All test motors are products of trademark GAMAK, 3-phase and 2-pole squirrel cage induction motors rated at 2.2kW, 50Hz, 380 V_{LL}. The resistance values of the resistor load bank is connected to the output of the single-phase 4.2 kVA permanent magnet synchronous generator. The resistance values are regulated in order to adjust the different loading conditions of the test motors which have 0.82 power factor and 4.94 A of rated current. Between the AC network and the test rig, there is a Δ-Y connected 25 kVA isolation transformer.

The most frequently encountered mechanical faults are created synthetically on four of these test motors for the experimental implemantation. One of identical motors is left untouched to compare clearly. In the following Table 2.1 demonstrate that list of fault types and their abbreviation used in this thesis.

Table 2.1. *List of faulty motors used for experimental setup*

Abbreviation	Faulty Motor
Healthy	Untouched Motor
BFE	Motor with Bearing Fault Misalignment
5BB	Motor with 5 Broken Bars
BFA	Motor with Bearing Fault Ball Defect
3BB	Motor with 3 Broken Bars

The test motors are coupled with a single-phase 4.2 kVA permanent magnet synchronous generator which is connected to an adjustable resistive load bank. The different loading conditions are selected as 3.6, 4.1, 4.7, 4.9, 5 Amperes. Before the collecting data, the test motors function under no load conditions in a few minutes in order to reach the steady-state. After this, the resistor is adjusted for loading the motors in pre-planned loading condition levels in a controlled manner.

2.3.1. Receiving vibration data

In this thesis, only the vibration data which come from the squirrel-cage induction motor is used. In order to collect vibration data, five motors which are driven directly from AC are used. For vibration sensing, an integrated electronic piezoelectric (IEPE) accelerometer which is a product of trademark Metra Mess model KS943B.100 is used. In order to amplify vibration signals, the accelerometer is connected to a multichannel signal conditioner that is model M108 of the Metra Mess which can be shown in Figure 2.4. NI 6251 data acquisition card is used to digitize vibration signals at a sampling rate 20kHz with 16-bit vertical resolution.

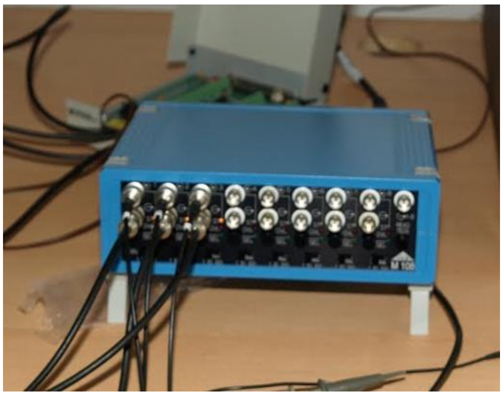


Figure 2.4. *Metra Mess M108*

Vibration data are collected from 5 test motors under 5 different loading conditions for forty seconds. The position of the accelerometer does not change to all test motors. From the Figure 2.5, it can be seen that the accelerometer and its position on fan cover.

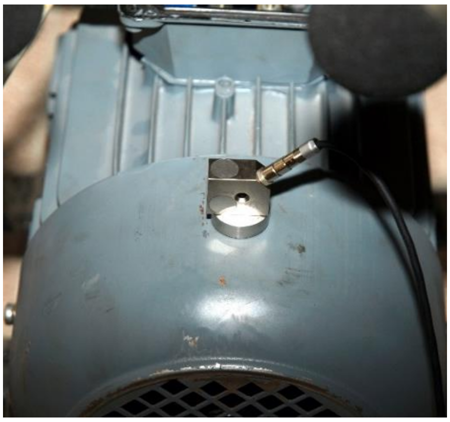


Figure 2.5. *Accelerometer and its position*

These records were repeated 3 times for each test motors (except BFA which is for 5 times, explains why in Chapter 4) to create more reliable and accurate test data such as in real industrial environment with different mounting conditions.

Figure 2.6 shows the experimental setup for collecting the vibration and acoustic data to detection of the induction motor faults.

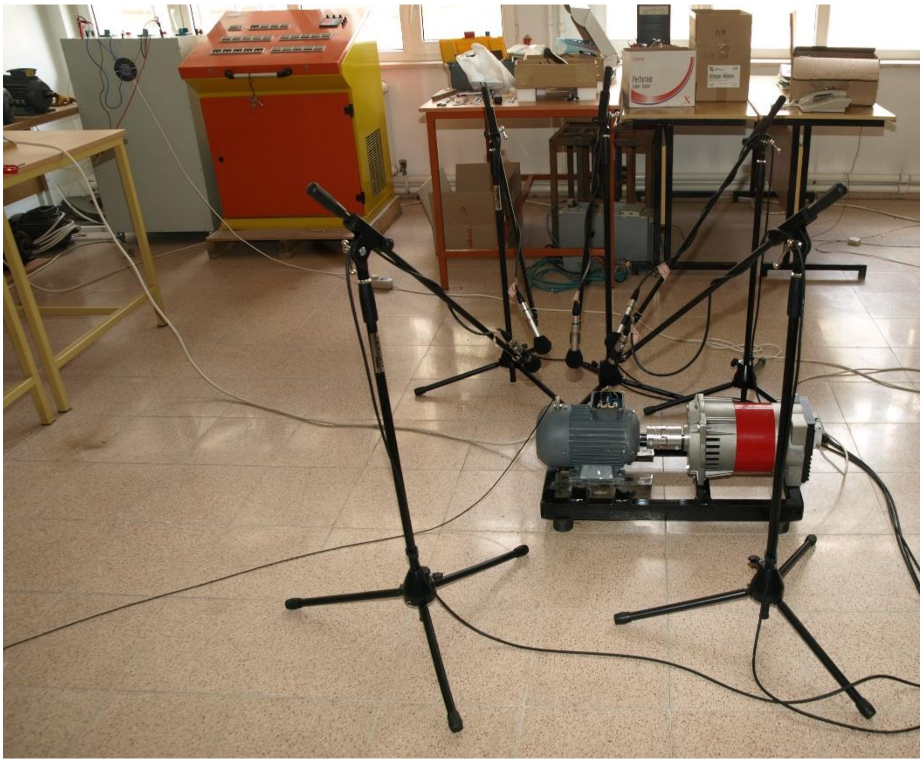


Figure 2.6. *Experimental Setup for vibration and acoustic data*

To mention briefly on receiving of acoustic data, as in Figure 2.6, there are 5 microphones and one is placed 60 cm above from the test rig. Others are located at a lower height and surround the system in a rectangular form. All experiments are carried out at the sound level of a real working environment. According to this, the average ambient sound pressure level (SPL) is recorded as 53.2 dB. The sound pressure levels related to all test motors during the experiments is given below in Table 2.2:

Table 2.2. *Sound pressure levels of the experiments of the dataset*

Environment SPL : 53.2 dB

Sound Pressure Levels of the experiments						
		3.6 A	4.1 A	4.7 A	4.9 A	5.0 A
Healthy	Exp 1	73.8	76.5	78.5	78.5	78.5
	Exp 2	74.9	76.1	74.6	75.1	76
	Exp 3	73.8	73.8	73.8	73.8	73.8
BFE	Exp 1	86.2	84.5	84.2	84.3	84.2
	Exp 2	83.7	83.7	83.2	84.2	83.2
	Exp 3	83.2	83.2	83.4	83.7	83.2
5BB	Exp 1	80	78.8	79.1	79.1	78.5
	Exp 2	78.8	80.1	79.5	79.5	79.5
	Exp 3	76	75.4	76.1	76	76
BFA	Exp 1	84.6	83.7	83.3	83.2	83.2
	Exp 2	83.2	83.2	83.2	83.2	83.2
	Exp 3	83.2	83.7	83.4	83.4	83.8
	Exp 4	85.4	84.8	83.9	84.1	83.7
	Exp 5	83.2	83.2	83.3	83.5	83.7
3BB	Exp 1	75	76.1	77.1	77.1	78.1
	Exp 2	76	74.5	75	75	75.4
	Exp 3	75	74.5	74.1	74.1	74.2

3. CONVOLUTIONAL NEURAL NETWORKS

Nature has become one of the most important inspirations to be able to make scientific progress all the time. The neocortex in the human brain, inspired by artificial intelligence (AI) researchers, recognizes and perceives very complex patterns of any tangible or even abstract form very quickly. This process owes to the fact that neurons, which are about 20 billion small processing units, are miraculously linked and organized to each other (Platek et al., 2007; Kurzweil, 2013). In the field of AI with this inspiration, the brain function has been tried to be imitated with great simplifications and thus the artificial neural networks (ANN) structure has been improved (Altenberger and Lenz, 2018). In this way, ANN has been an important research area in recent times and as a result of this research deep learning (DL) concept has been born (Hinton and Salakhutdinov, 2006; Hou et al., 2016; Yang et al., 2016). The most important reasons why deep learning is so popular in recent times are developed chip processing abilities (e.g. GPU units), lowering the cost of computer equipment, and advancement of machine learning algorithms (Deng, 2014). DL is a method which uses to model data with complex structures that combine different nonlinear transformations. The basis of DL is the neural networks that are combined to perform learning.

Figure 3.1 shows the simple ANN layers which are steps of ANN. It consists of 3 basic parts: Input layer which represent inputs, Hidden Layers which is optional and has detailed features of the inputs, and Output Layer which represents outputs.

The neurons in the same layer usually identify similar patterns, for example, neurons in the hidden layer 1, which is near the input layer, perceive simpler features (called Low-Level features) such as line, edge. As going deeper into the layers, neurons perceive more specific features (called High-Level features) of input. Moreover, all neurons in a layer have connected the neurons in the previous and subsequent layers. These connections from one neuron to another are called *Synapses*. All synapses have *Weights* which determines the importance of the result of the lower level neuron for the outcome of the higher level neuron. To illustrate, if the feature is more distinctive and dominant for the recognition of the induction motor faults, then the weight corresponding to that feature must be that much.Furthermore, each neuron also has a Bias which

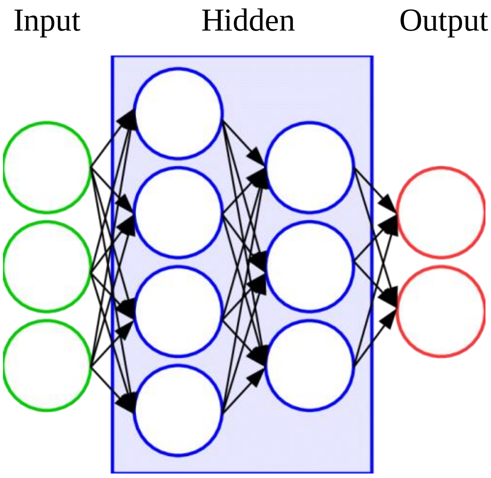


Figure 3.1. *A simple artificial neural network, consisting of an input layer, an output layer and two hidden layers (Altenberger and Lenz, 2018)*

indicates the likelihood of the presence of the corresponding pattern. These weights and biases are called *Parameters* of the network. These parameters are learned during the training process. The result is derived from the input from the setting of these parameters. The equation 3.1 demonstrate simply this procedure. After getting a result, an activation function which is mentioned later is applied to conclude the process.

$$output = (\sum_i input(i) * weight(i)) + bias \tag{3.1}$$

It is necessary to do many repetitions to train the network and these repetitions are evaluated by the loss function (for more information: Janocha and Czarnecki, 2017). The gradient of the loss function is calculated so that the parameters are updated and the updated output layers are transmitted to the previous layers. This process is called *Backpropagation*. In this way, it is intended to give a more accurate result on the next repetition.

One of the main difficulties of deep learning architectures is that different layers can learn at different speeds and adapt accordingly. This brings two problems: vanishing gradient and exploding gradient. Vanishing gradient problem, which is caused by very low gradients, is about especially the layers near the input layer which are learning at very slow speed. On the contrary, exploding gradient problem is caused by high gradients (Nielsen, 2015). It is, of course, possible to avoid these problems, but the manual adjustment of the required parameters is very difficult, especially in the networks with a large dataset. Appropriate choice of activation functions, suitable network parameter initialization, input normalization and gradient-norm clipping are the solutions of these problems. Even if all these parameters are set, the network has to optimize parameters. For this reason, the training process can be prolonged and information density can occur. To avoid such long-running processes, various network architectures have been developed such as convolutional neural networks.

After Hubel & Wiesel found that the cells in the visual cortex in 1959 were responsible for the divergence of light in the receiving areas, Fukushima suggested that neocognitron in 1980 (Hubel and Wiesel, 1968; Fukushima, 1982). Thus, the first seeds of CNN were thrown. The first modern frame of CNN has been the LeNet-5 multi-layered artificial neural network, which is known to be developed to classify handwritten digits. Figure 3.2 shows LeNet-5 as an example of CNN.

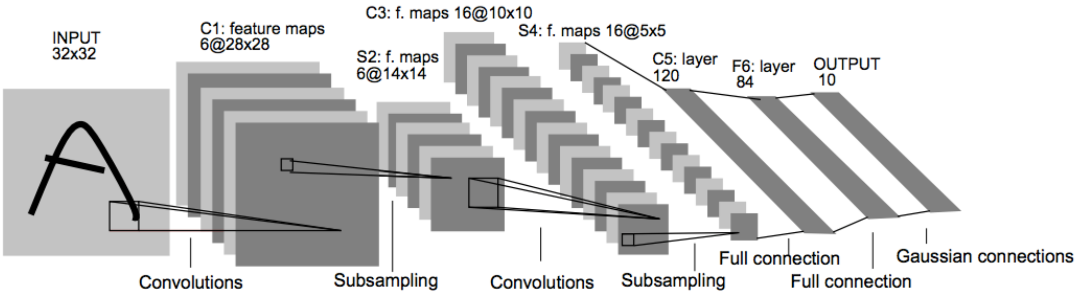


Figure 3.2. *Architecture of LeNet-5 (LeCun et al.,1998)*

Just as in time-delay neural networks (TDNN), CNN shares the weight in a temporal dimension. Thus, convenience is provided in the calculation. Convolution in CNN reduces the complexity of the network by decreasing the number of weights compared to a traditional neural network which is used in matrix calculation. Moreover, since the images from outside can be transferred directly to CNN, the feature extraction procedures in standard learning algorithms may not be needed. It is also the network that requires minimum pre-processing. Sparse interaction (sparse connectivity), parameter sharing and equivalent representation play an important role during the learning stage of the CNN (Bengio and Courville, 2016). Since the relationship between input and output in CNN is applied to the entire image using filters which are smaller than inputs, the calculation is reduced with sparse interaction. Furthermore, parameter sharing is for better performance by learning in each place using only one set of parameters, not a separate set of them. In addition to this, equivalent representation due to parameter sharing means that the output will change when the input is changed.

In short, CNN needs fewer parameters than other classical neural networks, which reduces memory and provides efficiency.

CNN is composed of two main parts: feature extraction and classification. The most important point that separates CNN from other deep learning techniques is the feature extraction part. In this part, CNN extract features directly, without having to manually configure. This section contains convolution layer, rectifier linear unit, and pooling layer in order. A fully connected layer can be considered as part of the classification and the last one is an output layer.

3.1. Convolution Layer

In the convolutional layer, CNN uses some filters called *kernel* to recognize and learn all the images which come in as input and intermediate maps of the input. The input is first convolved with a filter introduced into the system and then, an element-wise non-linear activation function is applied to the obtained convolved results. In this way, a *feature map* or *convolved feature* is obtained. Each neuron of the feature map is connected to a region of neighboring neurons in the previous layer called *receptive field*. The receptive field is meant to be a region of square size only in height and width dimensions. There is no hyperparameter that must be defined for the depth dimension of the receptive field because this dimension is necessary to combine the information as it identifies the different colors of the input and, in addition to this, the default convolution is applied all along the depth. But, when it comes to the definition of depth in the convolution layer, it is related to how much filter is used. Namely, *depth* is the number of the kernel that used for the convolution operation. For example, if the convolved feature has a depth of 5, the number of filters used is 5. Furthermore, the kernel must be used by all locations of the input to generate the feature map. If more feature maps are to be extracted, more filters should be used and thus, CNN becomes better at recognizing patterns.

The input image of size 3X3 and the kernel of size 2X2 are shown in Figure 3.3 to demonstrate one-dimensional convolution operation. In the convolution operation, the kernel is sliding over the input and the overlapping pixel values are multiplied and then all are summed. Next, the filter is shifted one more time and the same process is repeated. Briefly, the convolution operation applies dot product between the local area of the input image and the kernel to produce the feature map as an output of the layer. In addition, a feature map occurs when input and kernel are convolved as shown in Figure 3.4.

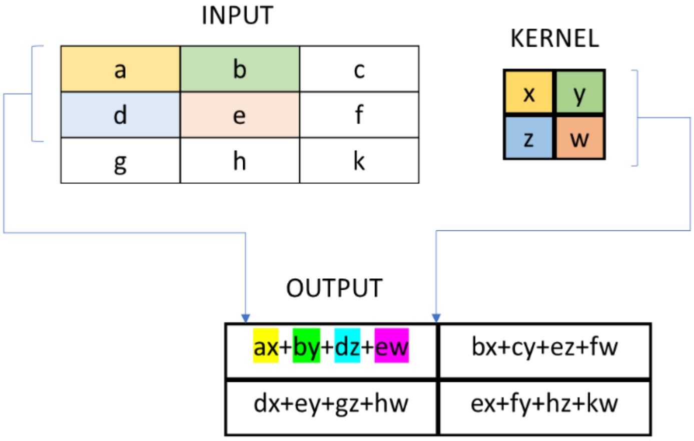


Figure 3.3. *An example of 2D Convolution Operation*

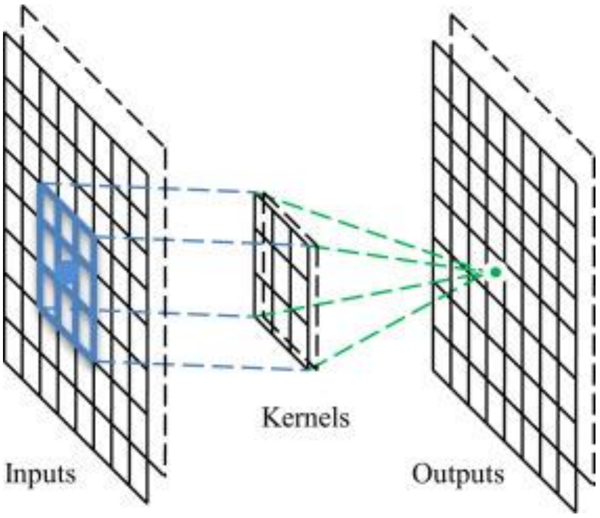


Figure 3.4. *The operation of the convolution layer (Guoa et al., 2016)*

From mathematical point of view, Equation 3.2 and 3.3 show the convolution operation:

$$z(t) = \int x(a)w(t-a)da \tag{3.2}$$

It is also indicated by asterisk as:

$$z(t) = (x * w)(t) \tag{3.3}$$

In CNN, the first function x is usually called as the input and the second argument was the kernel. The output z is the feature map. While input x and the kernel has only integer values, the discrete convolution operation can be defined by Equation (3.3):

$$z(t) = \sum_{a=-\infty}^{\infty} x(a)w(t-a) \tag{3.4}$$

Since the input image I and the kernel K are two dimensional, the operation becomes as in Equation (3.5):

$$z(i,j) = (I * K)(i,j)$$

$$z(i,j) = \sum_m \sum_n I(m,n)K(i-m,j-n) \tag{3.5}$$

Convolution operation has generally three advantages: i) sharing parameters in the same feature map reduce the number of weight, ii) local link learns the correlation between neighboring pixels, iii) invariance to the location of the object (Guoa et al., 2016). These are related to sparse connectivity, parameter sharing, and equivalent representation as mentioned a little earlier. In sparse connectivity, the kernel needs to be smaller than the input so that when the image is processed only small and meaningful edges can be picked and hundreds of unnecessary pixels can be ignored. Figure 3.5 shows the sparse connectivity clearly. As can be seen, the memory requirement is reduced and the efficiency of the network increases with the aid of sparse connectivity. In addition to this, because of parameter sharing, CNN uses a kernel for each member of the input to scroll through the input, unlike a classical neural network used the kernel which is multiplied by only one member of the input and is never used again. Following Figure 3.6 explains the parameter sharing briefly. Moreover, parameter sharing leads to equivalence translation which means that when the input changes, the result of output changes in the same way, which is very useful for classification and recognition of tasks.

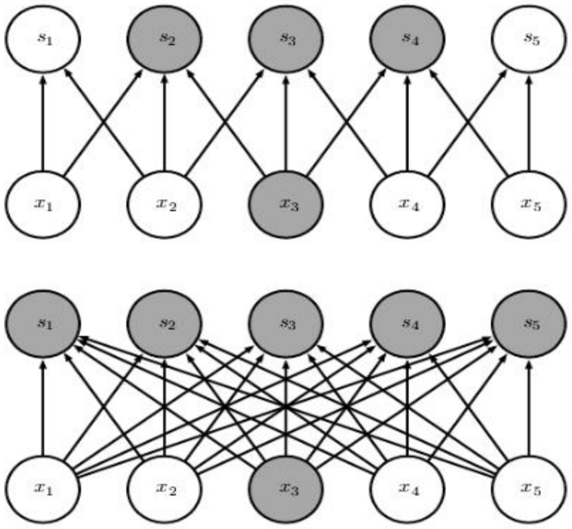


Figure 3.5. *Sparse connectivity, viewed from below. x_3 is highlighted one input unit, and units are hihglighted the output units in s that are affected by this unit. (Top) When s is formed by convolution with a kernel of width 3, only three outputs are affected by x. (Bottom) When s is formed by matrix multiplication, connectivity is no longer sparse, so all the outputs are affected by x_3. (Goodfellow et al., 2016)*

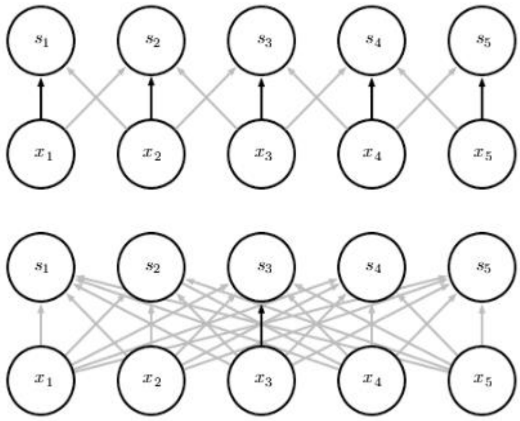


Figure 3.6. *Parameter sharing. Black arrow shows the connections that use particular parameter in two distinct models. (Top)The black arrows indicate uses of the central element of a 3-element kernel in a convolutional model. Because of parameter sharing, this single parameter is used at all input locations. (Bottom)The single black arrow indicates the use of the central element of the weight matrix in a fully connected model. This model has no parameter sharing, so the parameter is used only once (Goodfellow et al., 2016)*

A hyperparameter called *stride* can be defined that specifies how much space between the two scanned zones should be during the convolution operation. In other words, the stride is the number of pixels that is slid the kernel matrix over the input region. Another hyperparameter that can be identified to the network is *padding* which is used for the prevent feature map from shrinking. In order to do this, a zero-valued pixel is inserted to surround the input with zeros (called zero-padding). By this way, it increases the performance. Sometimes padding can also be applied to ensure that the convolution result is of the desired size. Figure 3.7 demonstrate that the distinct hyperparameters of convolutional layers. Following Equation 3.7 displays the output size of the convolution layer:

$$out = \frac{in - receptive\ field + 2 * padding}{stride} + 1 \tag{3.7}$$

It would be useful to talk about a preliminary method, *Batch Normalization*, that may well be done before moving to the rectifier linear unit. In general, all data is normalized to certain values before the CNN process is started, such as the conversion of

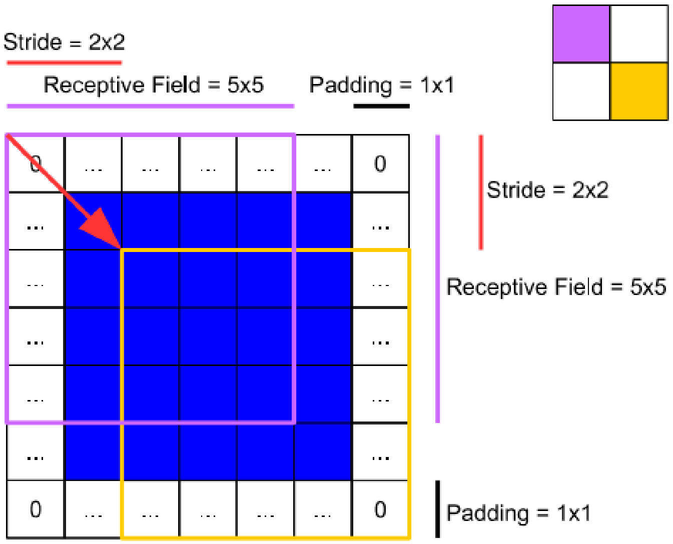


Figure 3.7. *A visualization of the different hyperparameters of convolutional layers: receptive field, stride and padding (Altenberger and Lenz, 2018)*

one-dimensional vibration data values to image pixel values 0 to 255 in order to construct an image. But, while the CNN operation is in progress, as the data travels deeper into the inner layers, it will be distributed and modified so the learning capacity of the network will decrease. To avoid this, a batch normalization (BN) has been developed after each mini-partition far from the entire variation set, in which the mean and variance inputs are calculated (Ioeffe et al., 2015). BN has important advantages such as reducing internal covariant shift, enabling the use of higher learning rate by reducing the dependence of gradients on the scale of the parameters or of their initial values and also reducing the need to *dropout* which will be encountered at the fully-connected layer.

3.2. Rectifier Linear Unit (ReLU)

Rectifier linear unit (ReLU) is one of the most known activation function. ReLU is a function that translates the negative part to zero and protects it as if it were positive (Nair and Hinton, 2010). Although backpropagation part of the network can damage due to the discontinuity over zero in ReLU, this simple operating principle allows ReLU to work faster and empirically than sigmoid and tanh activation functions (Maas et al.; 2013, Zeiler et al., 2013). Figure 3.8 displays the ReLU operation.

$$relu(x) = \max(0, x)$$

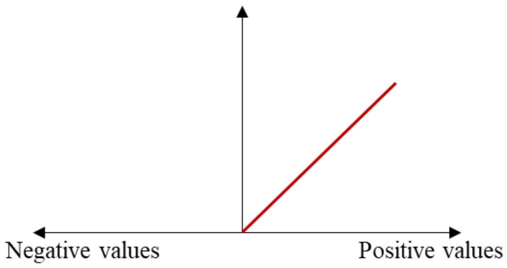


Figure 3.8. *The ReLU Operation*

In this way, the network can approach very complex functions by making non-linear transformations. Researchers recommend using ReLU even at deeper layers to simplify network optimization, accelerate convergence, better expansion, and acceleration of computation (Zeiler et al., 2013).

3.3. Pooling Layer

The Pooling Layer (PL) stands between the convolution operation (usually after ReLU) and the CNN architecture. This layer is sometimes called as a down-sampling or subsampling.

Due to the high dimensions in CNN, a classification process can cause *overfitting* because it will be difficult to maintain *translation invariance* by adding additional filters when the size of the inputs remains equal to the filter of the outputs. Translation invariance allows an object to be identified even if the image changes for different reasons and is therefore important. It can be said that it increases the resistance to noise (Boureau et al., 2010). So, the aim of PL is to reduce the spatial size but not the depth of the CNN representation. Pooling reduces the size of feature maps and network parameters exactly as desired. In this way, the complexity and computation of the network are reduced.

There are different types of pooling operation that is the max, average, sum, etc. The most preferred one is the *max pooling*. Firstly, max pooling was developed to be sub-sampling to the LeNet-5 architecture. From a series of neurons, the maximum one is selected. Figure 3.9 and 3.10 show the max-pooling operation:

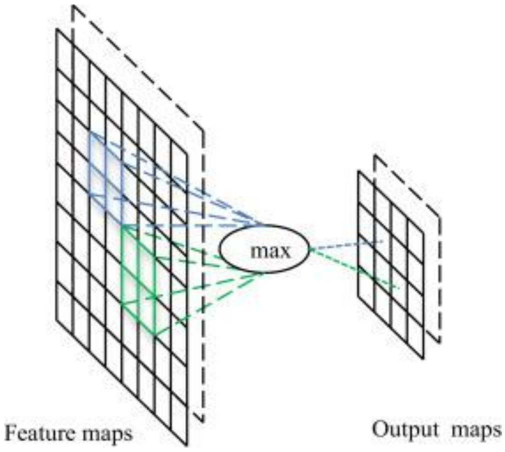


Figure 3.9. *An example of max pooling (Guoa et al., 2016)*

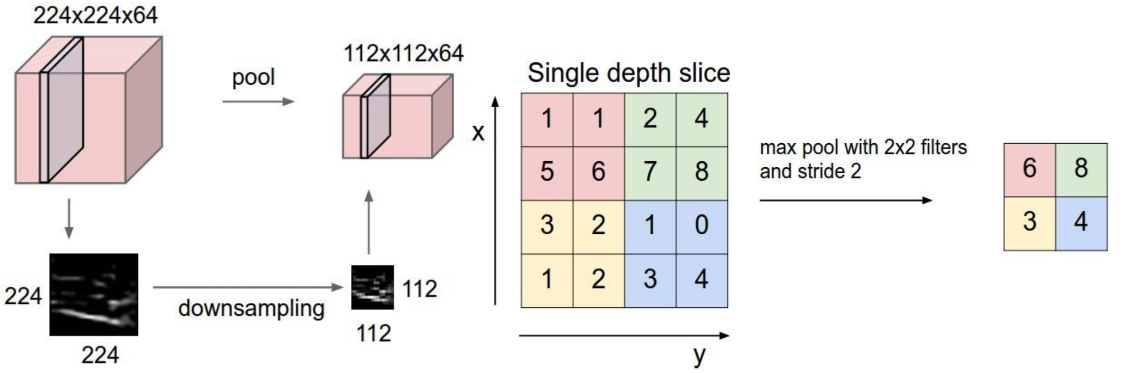


Figure 3.10. *The examples of pooling. Left-side: In this example, the input volume of size [224x224x64] is pooled with filter size 2, stride 2 into output volume of size [112x112x64]. Right-side: the example of max pooling with a stride of 2 (http-2)*

Pooling layer is reduced spatially, independent of the depth of the input dimension independent of the depth of the input dimension. According to left-side example of Figure 3.10, the input volume of size [224x224x64] is pooled with kernel size 2, stride 2 into output volume of size [112x112x64]. In the right side of the Figure 3.10, each max is taken over 4 numbers and small 2X2 square occurs.

3.4. Fully Connected Layer

Fully connected (FC) layer works like a traditional neural network. In the FC layer, 2D feature maps are converted to 1D feature maps, thereby providing more features. Since FC layer contains about 90 percent of the parameters found in CNN, there is a lot of effort in calculations. On the other hand, this layer allows the neural network to advance to a predetermined size and a certain number of categories. In addition, all neurons from the convolution layer (from the feature extraction section) must be flattened (sometimes called *Flatten Layer*) in order to pass to the fully connected layer. Figure 3.11 shows the operation of this layer, each neuron coming from the previous layer are attached to each activation on the following layer in this figure.

Furthermore, FC Layer does not only work in classification but it also aids learning non-linear combinations of the features. Once a few convolutions and pooling layers have been used, one or more FC layers may be used, which is good for a better prediction. Additionally, it is not always necessary to use the FC layer in some cases because it can change with the 1x1 convolution layer (Yin et al., 2014).

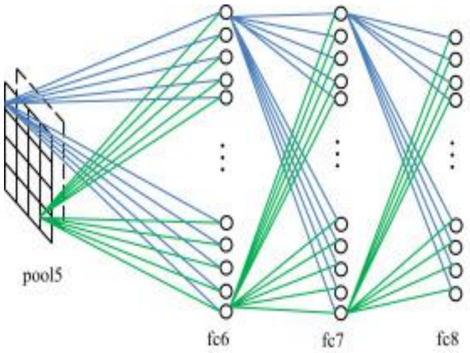


Figure 3.11. *The operation of Fully Connected Layer (Guoa et al., 2016)*

3.5. Output Layer

The last part of the CNN is the output layer. After the FC Layer, the CNN needs a function to classify the input image in this layer. For classification task, softmax and sigmoid functions are generally chosen according to the label of the input. The most important difference between the two is a determination of the number of the label. Softmax function is used for multi-classification tasks, on the other hand, the sigmoid function for binary classification tasks (Polamuri, 2017). Thus, in this thesis, as there are 5 labels of the motors faults, the softmax function is used. Moreover, if a selection of some internal variables in the model is needed, softmax function can also be used in the model.

Softmax Function (or *normalized exponential function)* computes the probabilities of every destination label over each feasible target label. For multi-class classification problems, softmax function yields the probability of all class and the destination one has the higher probability than others. Probability range is between 0 to 1. The sum of the entire the probabilities is equal to 1. In other words, the output layer consists of one neuron per class. Using softmax, each neuron is given values between 0 and 1, which is considered as a probability of belonging to that class.

The mathematical representation of the softmax function is as in Equation 3.8:

$$S(z_p) = \frac{e^{z_p}}{\sum_{t=1}^{K} e^{z_t}} \qquad p = 1..K \qquad (3.8)$$

Equation 3.8 gives the exponent power of the input value divided by the exponential sum of the other values of the input. That is probability distribution over K different possible outcomes. Thus, the range will always be between 0 and 1, and also the sum of the probabilities will be 1. In order to compare the softmax and the sigmoid function, the mathematical representation of the sigmoid function is shown in Equation 3.9:

$$S(t) = \frac{1}{1 + e^{-t}} \qquad (3.9)$$

The sigmoid function is defined for all real input values and each point has a positive derivative. In addition to this, the sum of the possible input values does not have to be 1. Moreover, the output of the sigmoid function is used to determine the probability of yes or no.

Due to the use of deep architects in CNN makes it more advantageous than other traditional networks. But, the size caused by the complexity of the network can lead to a problem called *overfitting*. Overfitting means that CNN is only compatible with the training data, not the test data, and thus, while training stage it gives perfect result above 99% accuracy. That is, with limited training in large networks, many of the complex relationships will only be a result of sampling loudness, which will only be found in the training part, not in the test part. Lately, many methods have been developed to deal with this problem called *Regularization* such as *Dropout* (Srivastava et al., 2014). Dropout means that the remove a unit with incoming and outgoing connections temporarily from the network. Which unit will be removed is a random process. The process can take the form of holding a certain probability of active neurons, releasing others. Figure 3.12 illustrates a simple example of a dropout operation.

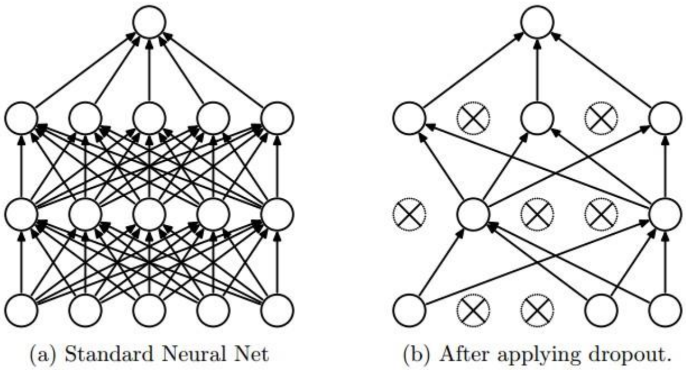


Figure 3.12. *(a) Standard neural net (b)* After *applying dropout (Srivastava et al., 2014)*

In addition to this, some optimization methods have been developed to reduce the loss of the network. In this thesis, Stochastic gradient descent (SGD), Adam and Nadam are discussed as regularization methods. Before coming to these subjects, it has to be known that a large number of repetition needs to be done for the training process and the parameters must be updated each time. This update happens with back-propagation (LeCun, 1992). Updates made for the entire training set, if the training set is large, the calculation of the gradient can be both very difficult and very costly.

Gradient Descent (GD) is the way to reduce the objective function of a model to the lowest. It minimizes by updating the parameters in the opposite direction of the gradient of the objective function. Equation 3.10 shows how this works:

$$\theta = \theta - \eta \cdot \nabla_\theta J(\theta) \qquad (3.10)$$

$J(\theta)$ is the objective function and η is the learning rate which is step taken to reach minimum level.

Stochastic gradient descent (*SGD):* In a classical network, GD is used and the aim is to make an approximate estimate of the cost and gradient taking into account the entire training set. Whereas SGD goes from a simple path, with only one or few training examples to be expected in the update. SGD helps to find better new local minima than the standard one. Unfortunately, the learning rate in SGD is lower than in standard GD. Equation 3.11 displays the different between them:

$$\theta = \theta - \eta \cdot \nabla_\theta J\left(\theta; x^i; y^i\right) \qquad (3.11)$$

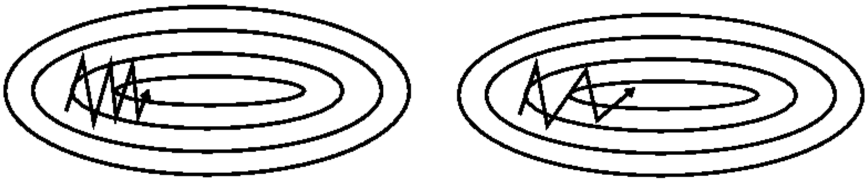
SGD update a parameter for each training instance, x^i label y^i which are a pair from the training set.

In addition to being popular, the SGD learning phase may be too slow and therefore the *Momentum* method has been developed (Polyak, 1964). In order to accelerate learning in the face of more highly-spirited and noisy gradients. The momentum term γ is added to the current update vector as in Equation 3.12:

$$\vartheta\,(t) \;=\; \gamma.\vartheta(\,t-1) \;+\; \eta.\nabla_{\theta}J(\theta) \tag{3.12}$$

$$\theta = \theta - \;\vartheta\,(t) \tag{3.13}$$

The momentum term is assumed to be 0.9 in experiments performed in this thesis. According to Equation 3.12 and 3.13, the convergence becomes faster and the oscillation falls with momentum. Figure 3.13 shows this more clearly:



(a) *(b)*

Figure 3.13. *(a) SGD without momentum (b)* SGD with momentum *(Ruder, 2018)*

Although the momentum with SGD produce fast solution to SGD without momentum, it does not guarantee where to slow down and therefore causes missing local minima. Therefore, another momentum technique called *Nesterov* has been further developed to solve this problem, which can be interpreted as a correction touch of momentum (Nesterov, 1983; Sutskever et al.,2013). The difference between the momentum and the Nesterov momentum is where the training is assessed. The Nesterov momentum gradient is evaluated after the current speed is applied. The goal is to give a rough idea of where the parameters are going to be. The SGD with Nesterov is shown :

$$\vartheta\,(t) \;=\; \gamma.\vartheta(\,t-1) \;+\; \eta.\nabla_{\theta}J(\theta - \;\gamma.\vartheta(\,t-1)) \tag{3.14}$$

$$\theta = \theta - \;\vartheta\,(t) \tag{3.15}$$

Adaptive moment estimation optimizer (*Adam):* Adam is the adaptive moment estimation which computes adaptive learning rates for each parameter. It is only a method that requires gradients from the first order, so it works fast in practice. The adam update-rule is :

$$\theta(t+1) = \theta(t) - \, {}^{\eta}\!/\!\left(\sqrt{\vartheta\,(t)} + \, \varepsilon\right) \; . m(t) \tag{3.16}$$

where $m(t)$ and $\vartheta(t)$ bias-corrected first and second moment estimates. Adam behaves in a similar way with momentum; two of them keep an exponentially decaying average of past gradients (Heusel et al., 2017). However, for example, If the momentum is seen a ball rolling down the slope, Adam is more frictional and acts like a heavy ball.

In addition to this, a method called *Nadam* (Nesterov-accelerated Adaptive Moment Estimation) which consists of combining adam and Nesterov comes out (Dozat, 2016).

In order to observe which optimization technique works better in a model of this thesis, each of these techniques is repeated under the same conditions. The results are in the next section.

To summarize this section, the information is transferred to the following layers in the first step and the properties of the input are obtained with the help of filters. Then, the error between the expected and actual values of the output is introduced to the system and the weight parameters are regulated according to this. In this way, the output is expected to target to the right class. Unlike other classical networks, preprocessing is not used and only CNN is expected to be trained. The nicest, CNN does not need human intervention in extracting features. Figure 3.14 shows a summary of the CNN architecture.

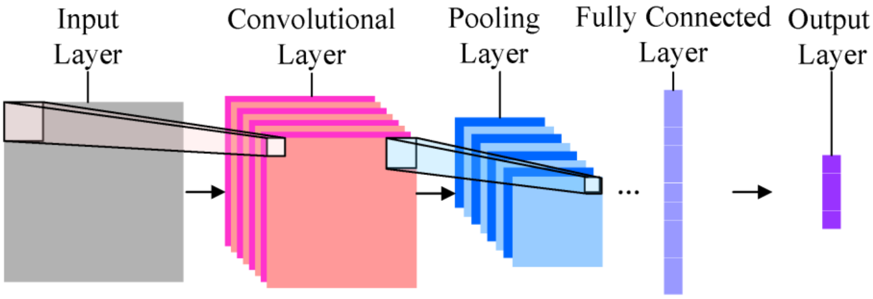


Figure 3.14. *The summary of CNN architecture (Peng et al., 2016)*

4. FAULT CLASSIFICATION

In this thesis, vibration data were selected to classify induction motor faults among acoustic, current and vibration data. Then, it might be good to mention vibration briefly. The fact that the vibration was the first time to enter the life of people with musical instruments (Maggiore, 2010). In fact, vibrations, from humans to machines, are in every area of life. For example, light waves coming to eyes through the vibration, the ear needs vibration to hear and even breathing is due to the periodic vibration of the lungs. At present, vibration is observed for solving more industrial problems and improving systems. Especially the point of interest in the industry is the vibrations of the machines, which is handled in this thesis as well as related to the vibration of an induction motor. The unusal vibration in machines is not wanted because it produced by definitive or undefined problems such as excessive deformation of the motor and wear on motor parts Moreover, it gives a lot of information about the problem types such as imbalance, shaft bow, the conditions of couplings, and misalignment (Tsypkin, 2011). Therefore, it is important to observe the vibration of the induction motor in order to prevent motor and to deal with the problems early.

4.1. Formation of Dataset

IEPE accelerometer, which is a product of trademark Metra Mess model KS943B.100, is used to collect vibration data. The most important point to note when collecting vibration data is the direction. Vibration data were taken from 3 axes, which are horizontal, vertical and axial, for measurement each axis was used separately and in combination to classify the induction motor faults. Three directions are used because it is thought that each direction could give different information according to the types of motor fault. Namely, vibration readings from specific axes provide the best indications of particular problems (Shannon, 2008). To illustrate this, while unbalance can be understood by information from radial (horizontal and vertical) axes, the information from the axial for misalignment is more discriminating. According to the classification method proposed in this thesis, the discriminating properties of each axis are combined together. The directions of the vibration data to be used in this thesis are given in Figure 4.1. Horizontal, vertical, and axial directions are assigned x, y, and z-axis respectively.

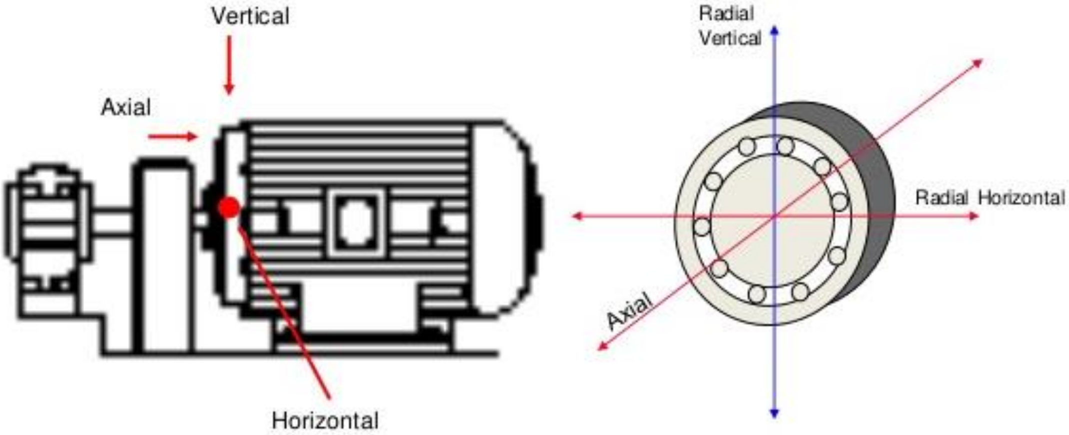


Figure 4.1. *The directions of the vibration data (http-1)*

As explained in Chapter 2, vibration data are collected from 5 test motors under 5 different loading conditions (3.6, 4.1, 4.7, 4.9, 5 Amperes) for forty seconds. These records were repeated 3 times for each test motors (except BFA which is for 5 times, the reason is explained in this chapter).

4.1.1. Formation of dataset from 1D vibration data

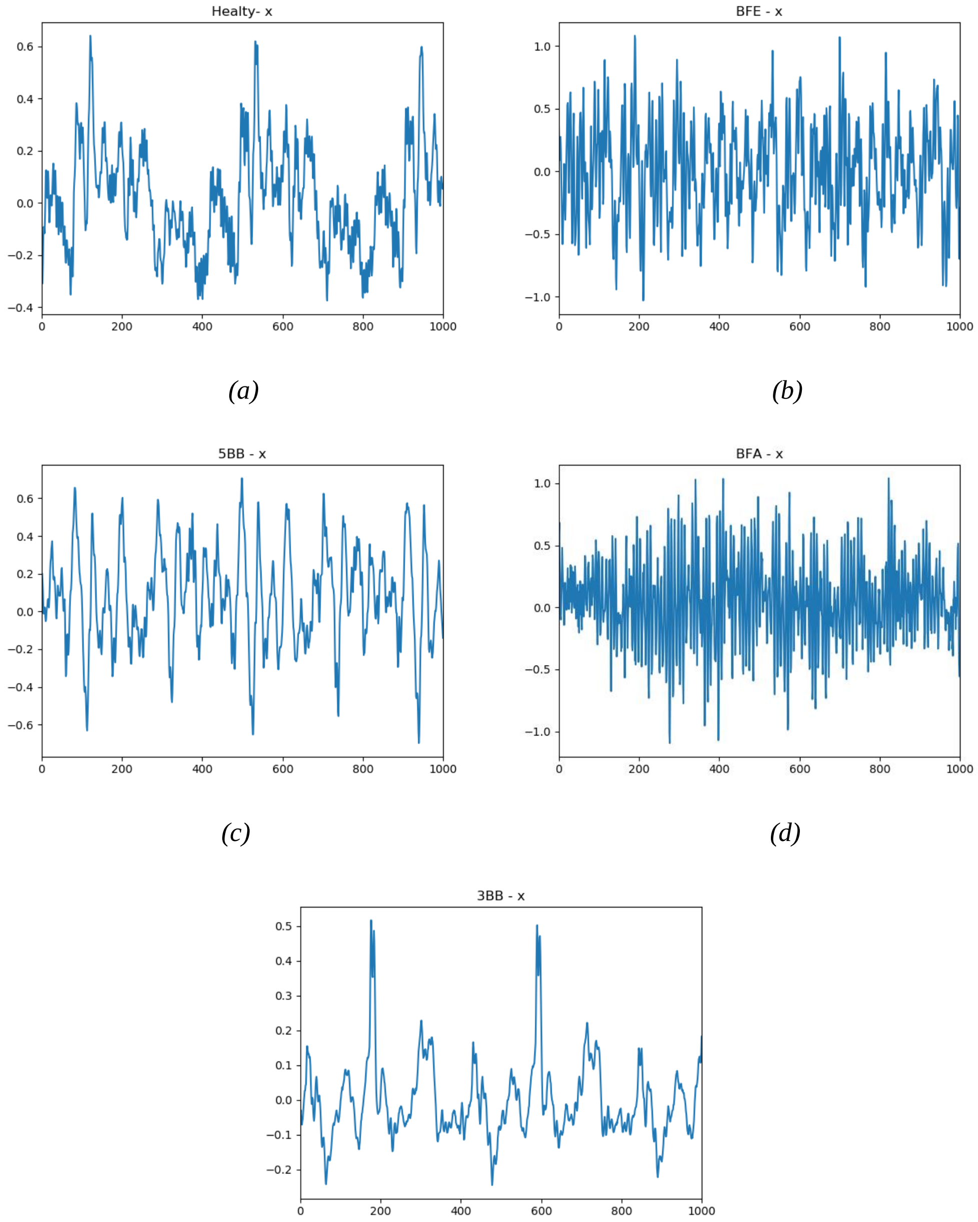
Since the vibration data taken from each axis is one dimensional (1D) and the classification method, CNN, works well in two or three-dimensional (2D, 3D) inputs, all data convert into 2D and 3D dataset respectively. In other words, grayscale (2D) and color (3D) images are created from the collected vibration data and thus prepared for the classification process.

4.1.1.1. 2D Image representation from 1D vibration data

Converting 1D data into the 2D images is a fairly recent method in order to classify (Gerek and Ece, 2004). As the images which show the characteristic of the data have used in many ways, every 1D data like vibration can be converted into the images. In this way, the system is prepared for classification by using CNN.

Figure 4.2 , 4.3 and 4.4 show the first 1000 vibration data from each x, y, and z-axes respectively under 4.1 Amperes of each faulty induction motor from the experiment 1. When looking at the vibration data according to these figures, it has been determined that faults in different induction motors are sufficiently distinguishable.Thus, looking at the visible difference between the vibration data of faulty induction motors, it is thought that a visual method (CNN) for separating them may be useful. Taking into account all the vibration data, since it is observed that there is a distinction between each vibration data of faulty motor, 1D data can be transformed into 2D to construct images. It is generally known that computers see the grayscale (2D) images as a number of pixel values that have the range of 0-255. According to this, vibration data coming from each induction motors is needed to normalize between 0 and 255. After the normalization, the obtained values in the range of 0-255 becomes the pixel intensity of the formed image.

As already mentioned, the induction motor is first fed from the AC network without any load and then gradually loaded with different load values such as 3.6, 4.1, 4.7, 4.9, 5 Amperes for forty seconds. Under the same load conditions, the tests are repeated by removing and then reconnecting the motor and generator. Thus, different experiments are formed. Three experiments per each induction motor (BFA has 5 experiment) for each loading conditions.

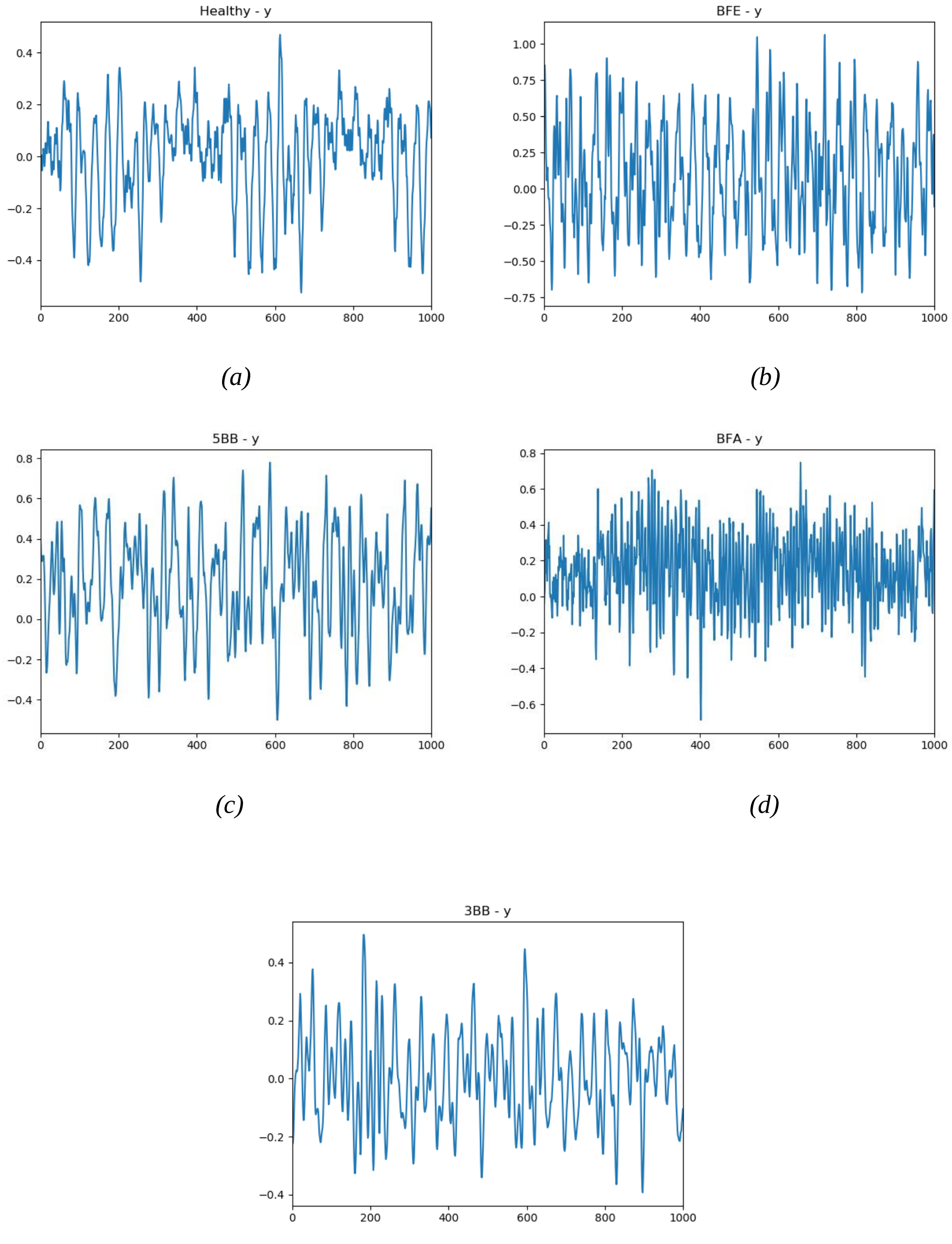


(a) *(b)*

(c) *(d)*

(e)

Figure 4.2. *The first 1000 vibration data from x- axis of (a) Healthy (b) BFE (c) 5BB (d) BFA (e) 3BB faulty induction motor under 4.1 Amperes*

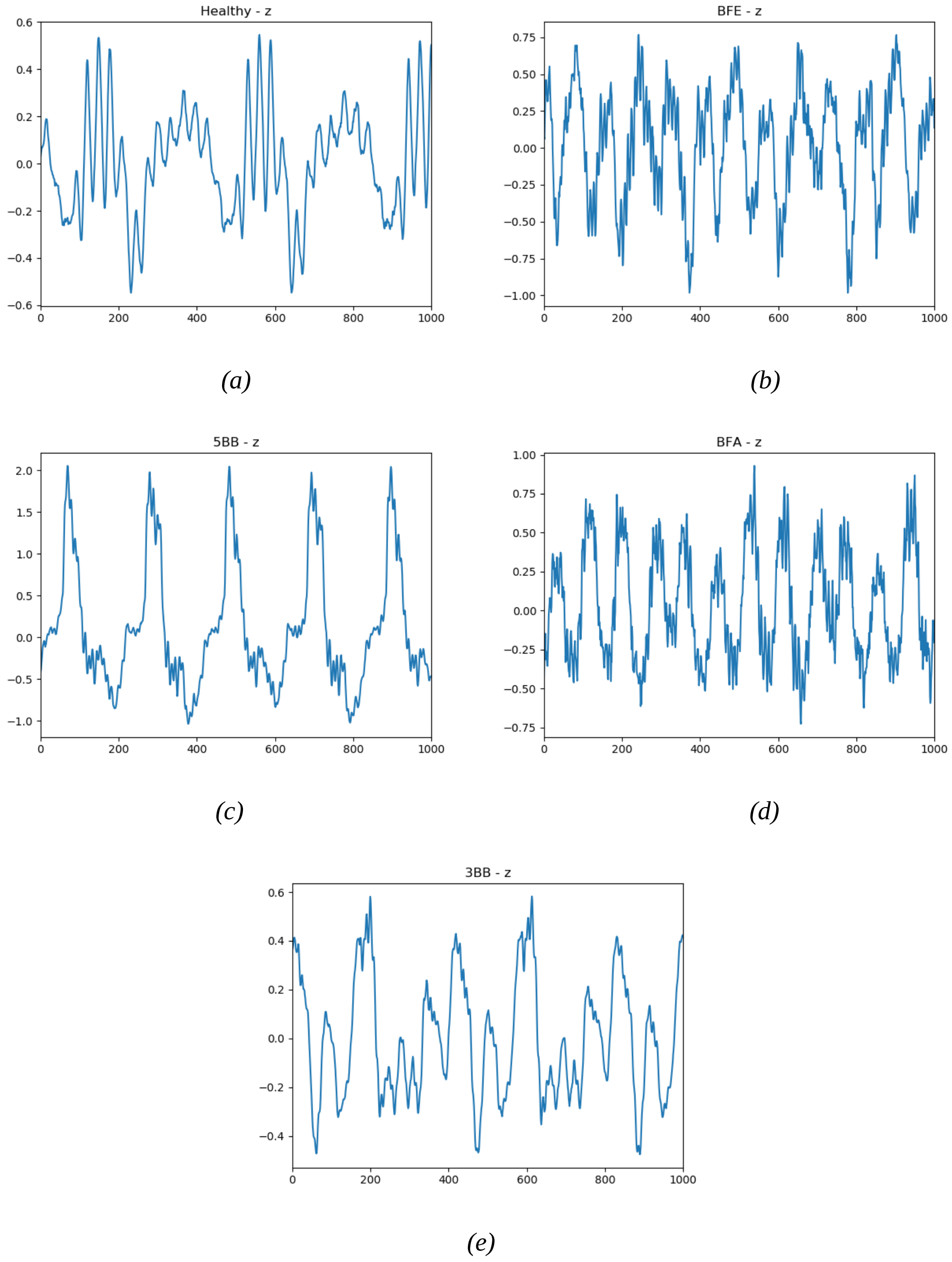


(a) *(b)*

(c) *(d)*

(e)

Figure 4.3. *The first 1000 vibration data from y- axis of (a) Healthy (b) BFE (c) 5BB (d) BFA (e) 3BB faulty induction motor under 4.1 Amperes*



(a) *(b)*

(c) *(d)*

(e)

Figure 4.4. *The first 1000 vibration data from z- axis of (a) Healthy (b) BFE (c) 5BB (d) BFA (e) 3BB faulty induction motor under 4.1 Amperes*

As it is repeated in experiments, when an induction machine without load is fed from the AC network, it rotates slightly more slowly than the synchronous speed. The increasing loading conditions, increases the stator current which will cause the rotation speed to fall (Germen, Başaran, and Fidan, 2014). For example, induction motor rotation speed under 3.6 A is slightly faster than the rotor speed under 4.1 A. Moreover, the type of faults have different influences on the speed of rotation of the motor. Because of all these reasons, the actual rotor frequency, which can vary with each load and fault, must be determined so that motor faults can be distinguished. Since the 2D image will be generated from the vibration data, the determination of rotor frequency has a vital importance for the widths of the images. Because the width of the images is based on a full period of the power signal. Thus, the correct information is obtained from the vibration data. In other words, the 2D image can be created only after a full period size is determined. If each motor creates an image with the same common period value, the imaginary lines can be angularly rotated in the image, which leads to an erroneous information about the vibration data. For the determination of the complete period, *Autocorrelation*, which is a correlation between values of a signal at different times, is used. Since vibration data coming from the induction motor faults are asynchronous, the sample size of the complete period differs due to the fault type. Autocorrelation peaks of each vibration data help to determine the size of the 2D grayscale image.

While the image is generated from the one-dimensional vibration data, the first element of normalized vibration data between 0 and 255 is assigned as the first pixel value, which is the upper left corner of the image. There is only one sample axis of vibration values in Figure 4.5. Apparent values in Figure 4.5 are one of the real x-axis values of the healthy motor recorded under the 3.6A. As shown in the figure, after the normalization, the first value fills the first-pixel position which is pixel [0,0] and goes like this on the row up to the specified width of the image.

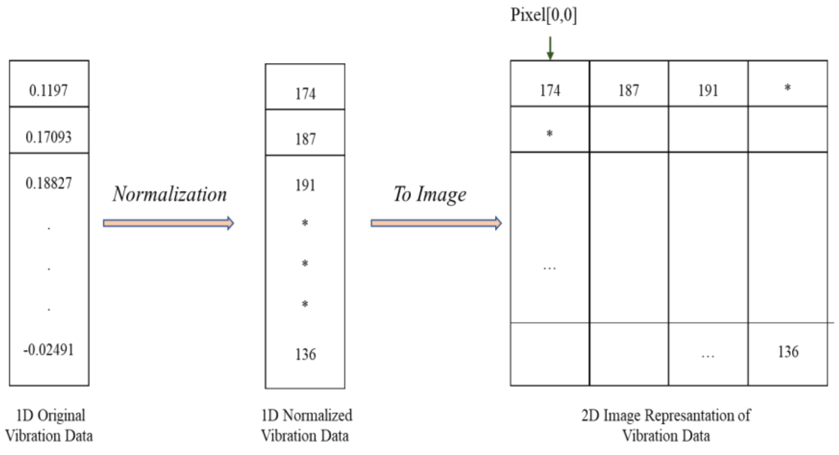


Figure 4.5. *An Example of 1D to 2D Vibration Data*

From the Figure 4.6, which gives the first 2000 samples of the vibration data (recording from Experiment 1), it is obvious that each induction motors under the same amperes load have different autocorrelation values. The healthy motor has 411 as an autocorrelation value, while the last figure has 413. This means that they have to have different size of images.

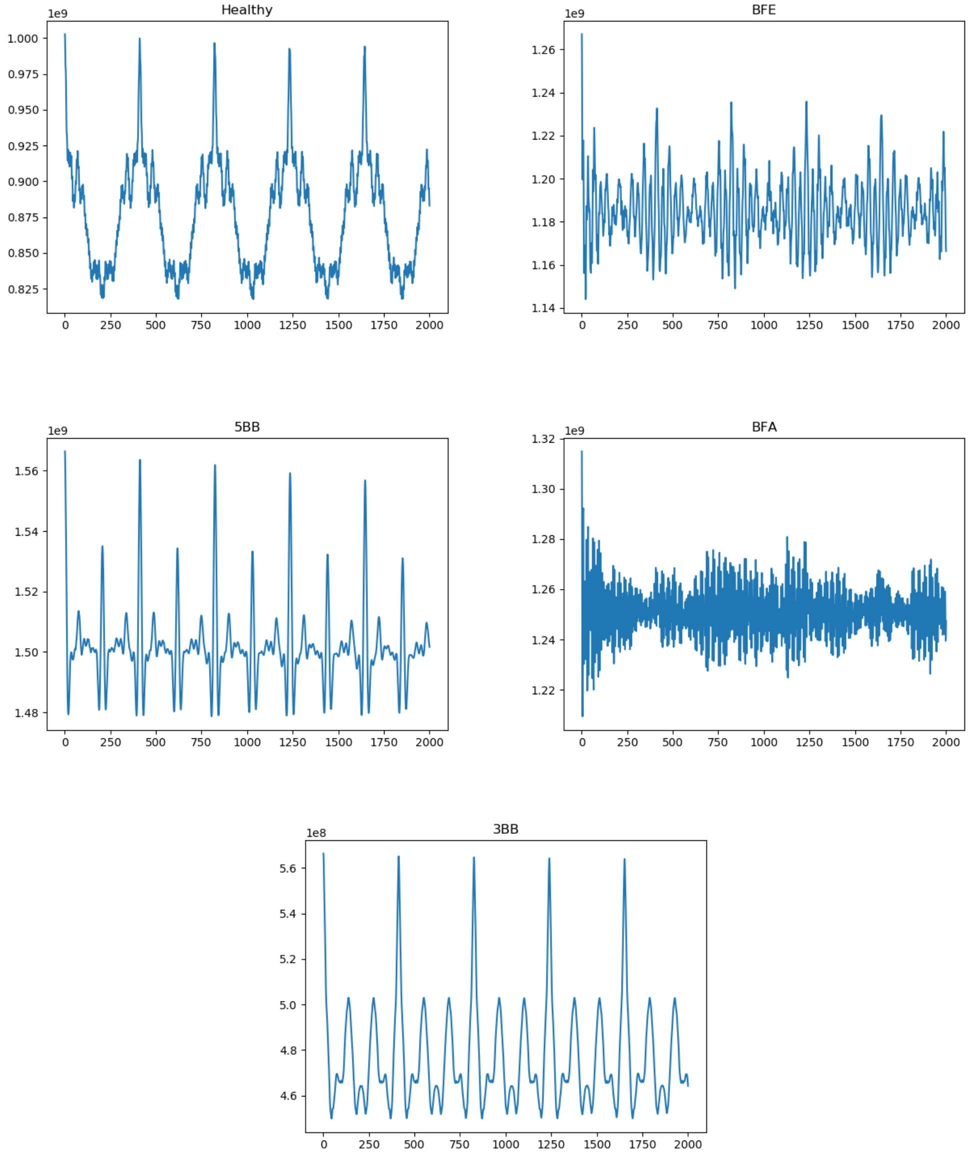


Figure 4.6. *Autocorrelation sequences of the vibration data recorded for five different induction motor under 4.1 Amperes load*

Even though it is the same motor, it can give different autocorrelation values in different loading conditions as shown in Figure 4.7.

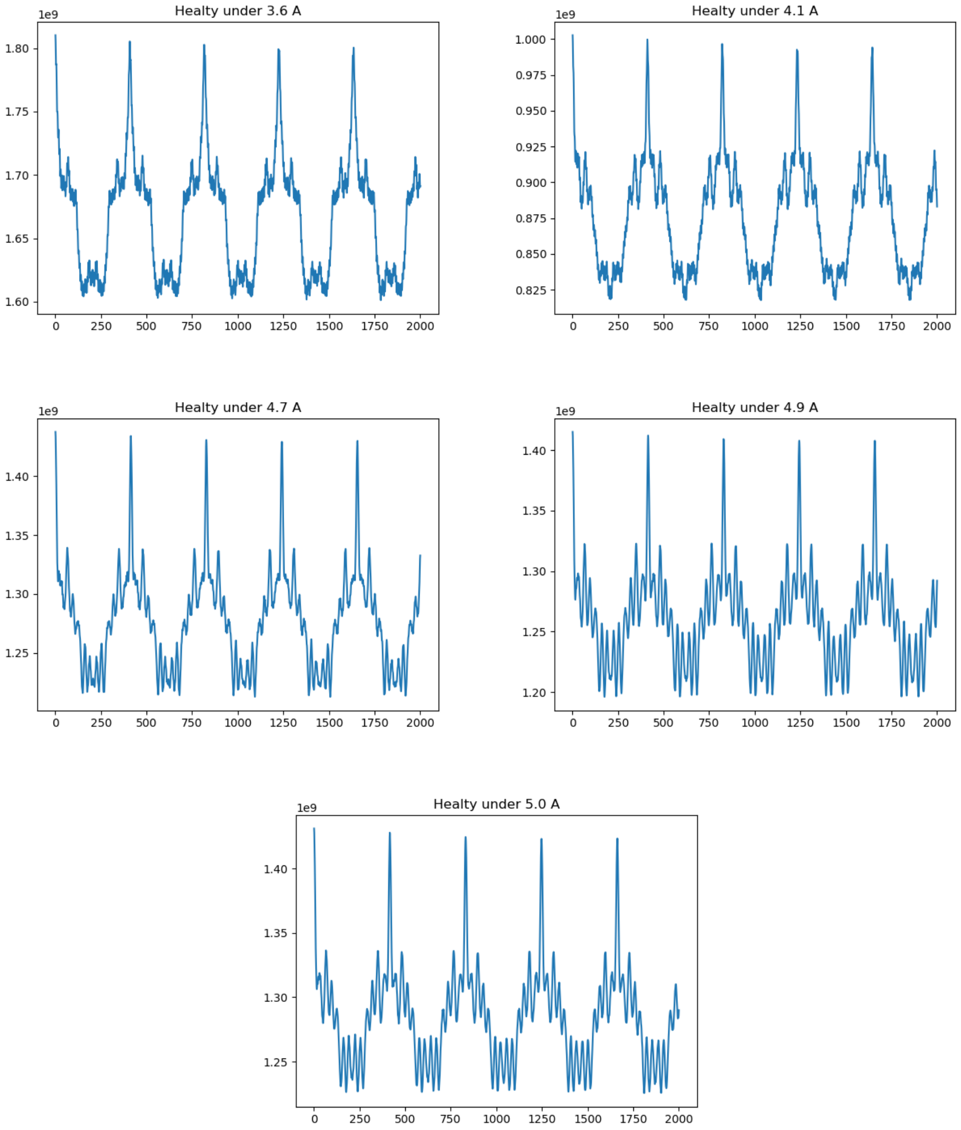
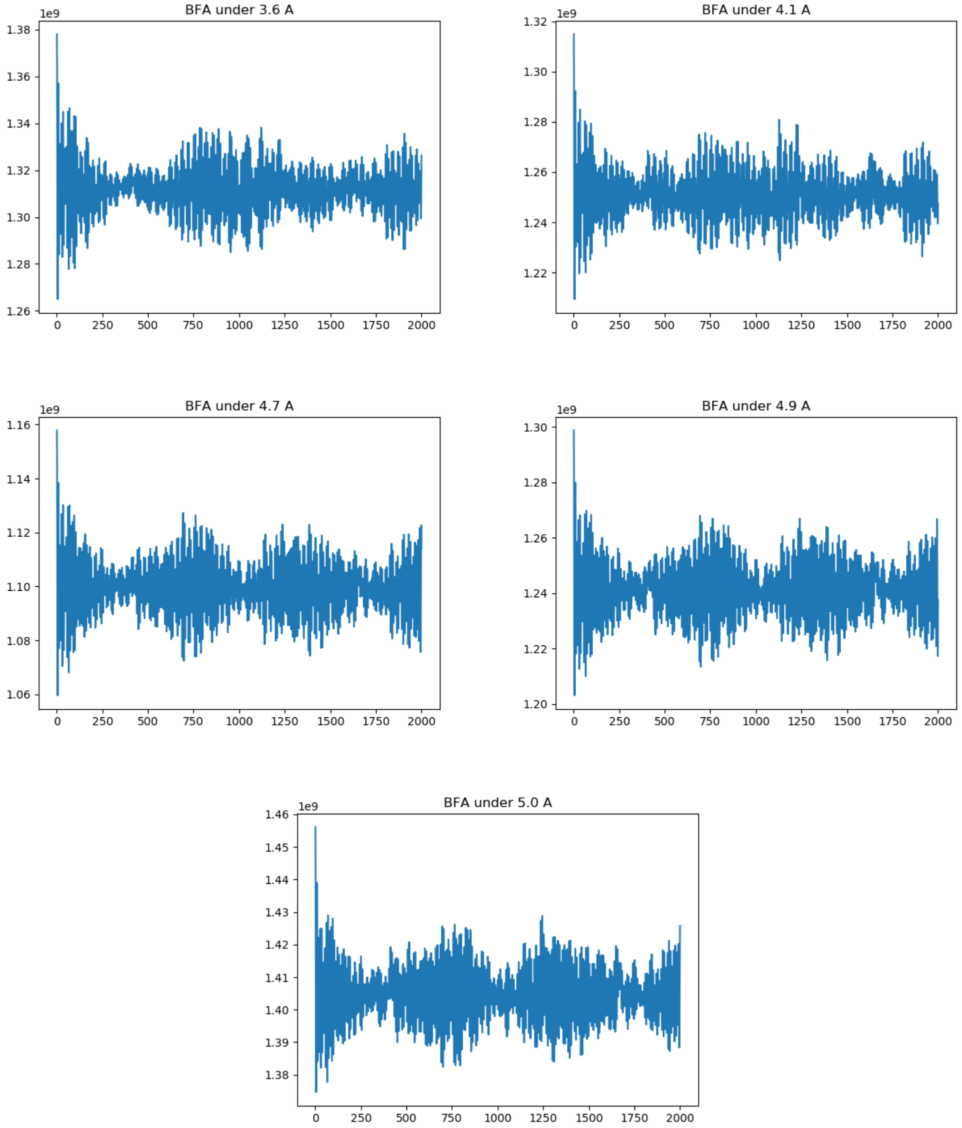


Figure 4.7. *Autocorrelation sequences of the vibration data recorded for healthy induction motor under different loading conditions*

The autocorrelation peaks vary on all induction motor faults as seen from the figures. In addition to this, the autocorrelation peaks in BFA are not shown clearly compared to other fault types as shown in Figure 4.6. It is very difficult to set the complete period for BFA according to the peak values. Because of this autocorrelation peak complexity of BFA, it is thought that even if the images are constructed at a certain size, they may interfere more with other faults. Thus, there are five experiments for BFA under different loading conditions. Figure 4.8 demonstrates the complexity of autocorrelation peaks under different loading conditions clearly.


BFA under 3.6 A

BFA under 4.1 A

BFA under 4.7 A

BFA under 4.9 A

BFA under 5.0 A

Figure 4.8. *Autocorrelation sequences of the vibration data recorded for BFA under different loading conditions*

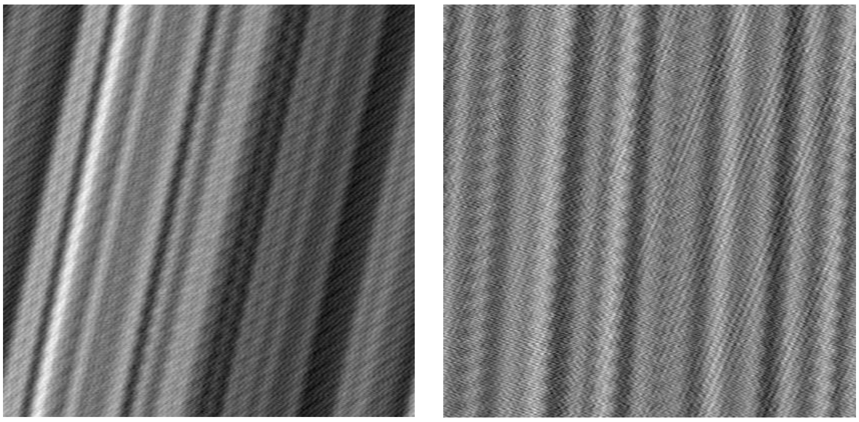
After autocorrelation, the 2D grayscale images can be created. The peak value of autocorrelation is the period of the sample, which is the size of the horizontal axis. All images were created separately according to the axis of the data. For each experiment, it has been observed that the period of each axis of vibration data is the same as expected.

Once the period value is obtained according to the autocorrelation values, a square image is created. Following Table 4.1 shows the autocorrelation values of all induction motors under varied loading conditions and experiments. As shown in the Table 4.1, image database, whose widths vary from 406 to 419 is created according to those autocorrelation values.

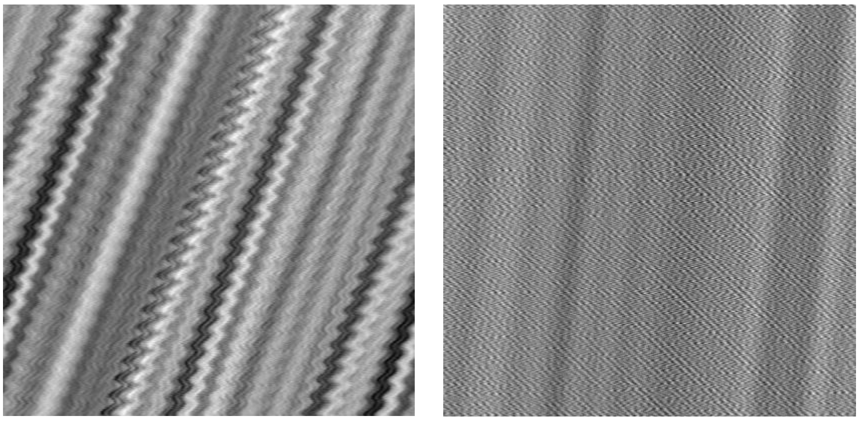
Table 4.1. *Autocorrelation Values of the vibration dataset*

Loading Conditions & Autocorrelation Values						
		3.6 A	4.1 A	4.7 A	4.9 A	5.0 A
Healthy	Exp 1	408	411	414	414	415
	Exp 2	409	411	414	415	416
	Exp 3	408	411	414	415	416
BFE	Exp 1	409	411	413	415	416
	Exp 2	410	413	411	415	416
	Exp 3	410	412	415	416	417
5BB	Exp 1	409	412	415	416	417
	Exp 2	409	412	415	416	417
	Exp 3	409	412	415	416	417
BFA	Exp 1	407	411	414	414	416
	Exp 2	410	411	413	414	416
	Exp 3	409	412	410	416	415
	Exp 4	406	410	412	413	414
	Exp 5	408	411	413	414	415
3BB	Exp 1	410	413	416	417	418
	Exp 2	410	414	417	418	419
	Exp 3	411	414	417	418	419

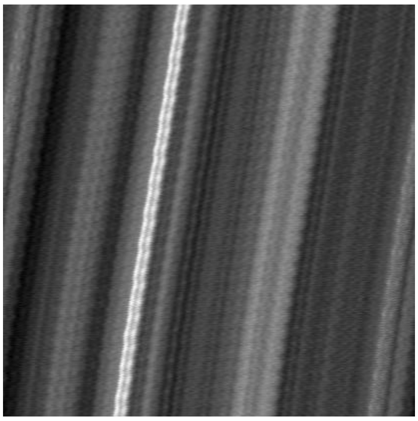
The 2D images of the vibration data obtained from different motors under 4.1 A is shown in Figure 4.9, 4.10 and 4.11. While images are being created, all the axis of the vibration data was considered according to their autocorrelation values. The vibration data obtained from the same induction motor form different images under different loading conditions. However, the characteristics of the images that consist of the same induction motor data are similar. Figure 4.12 and 4.13 show examples of 2D grayscale images taken from x- axis vibration data of healthy motor and 5BB respectively under different loading conditions.



Healthy *BFE*

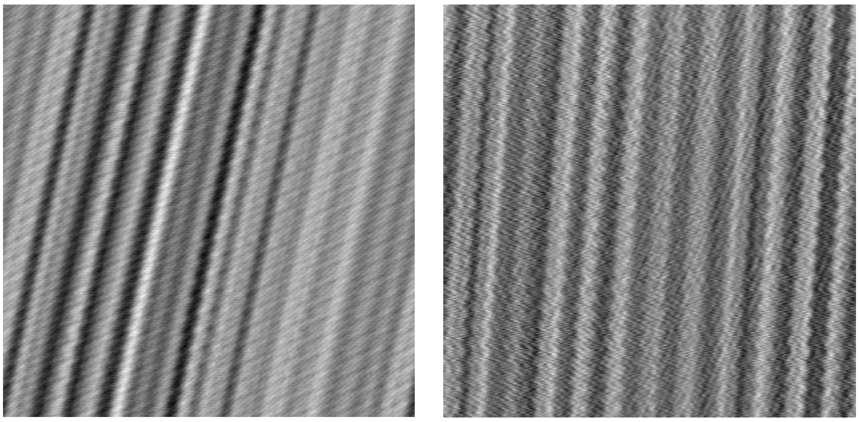


5BB *BFA*

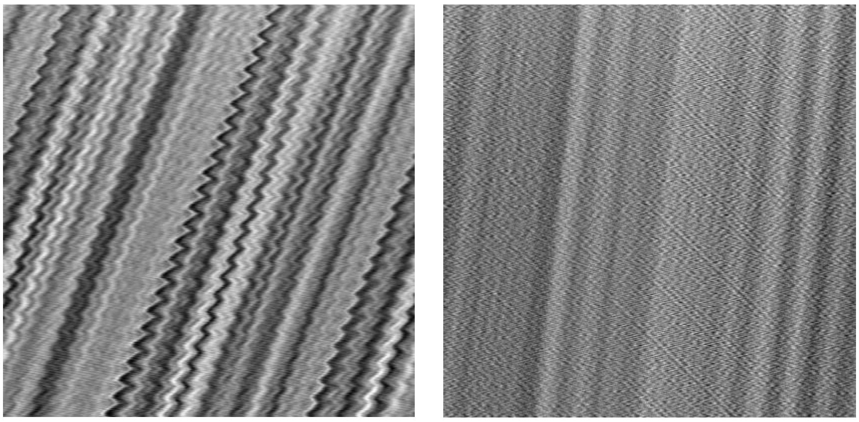


3BB

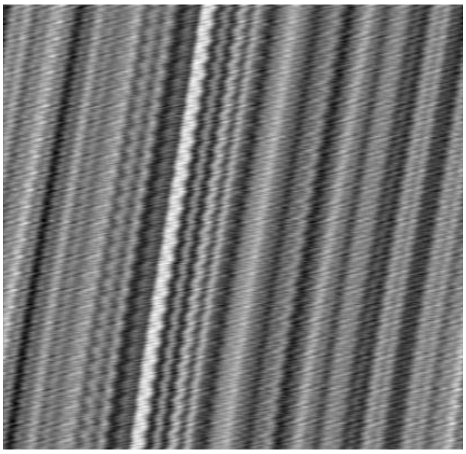
Figure 4.9. *2D Grayscale Images of the x- axis vibration data under 4.1 A*



Healthy *BFE*

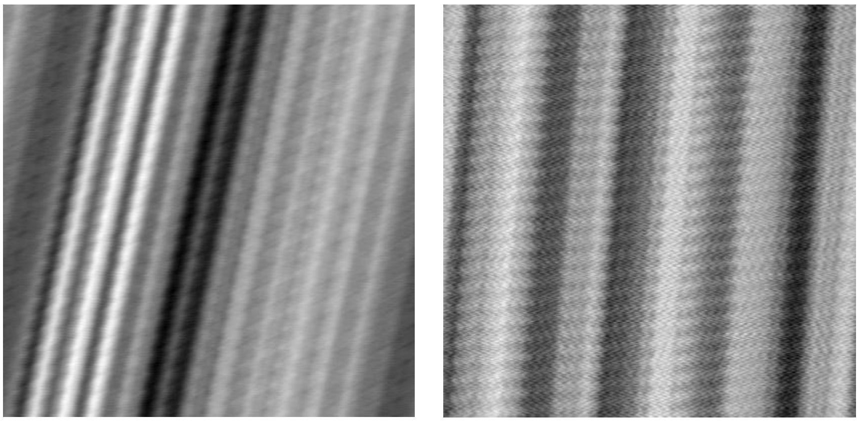


5BB *BFA*

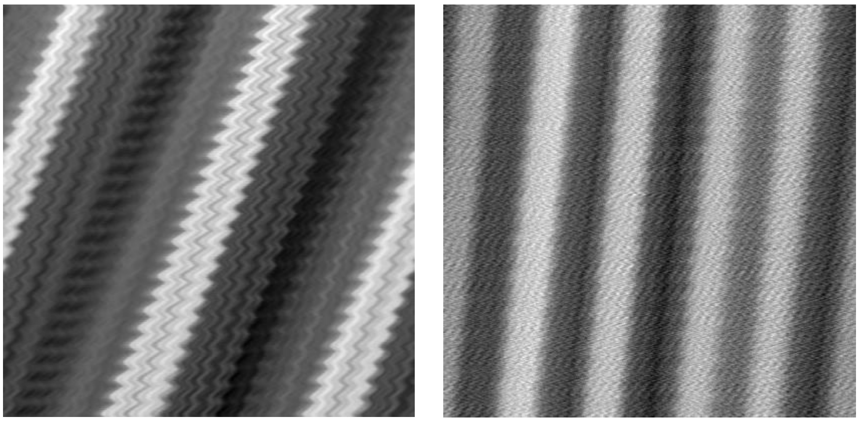


3BB

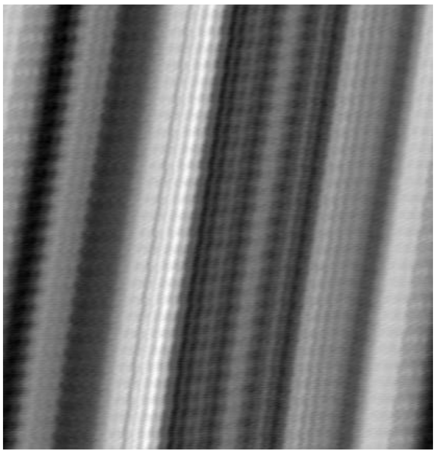
Figure 4.10. *2D Grayscale Images of the y- axis vibration data under 4.1 A*



Healthy *BFE*

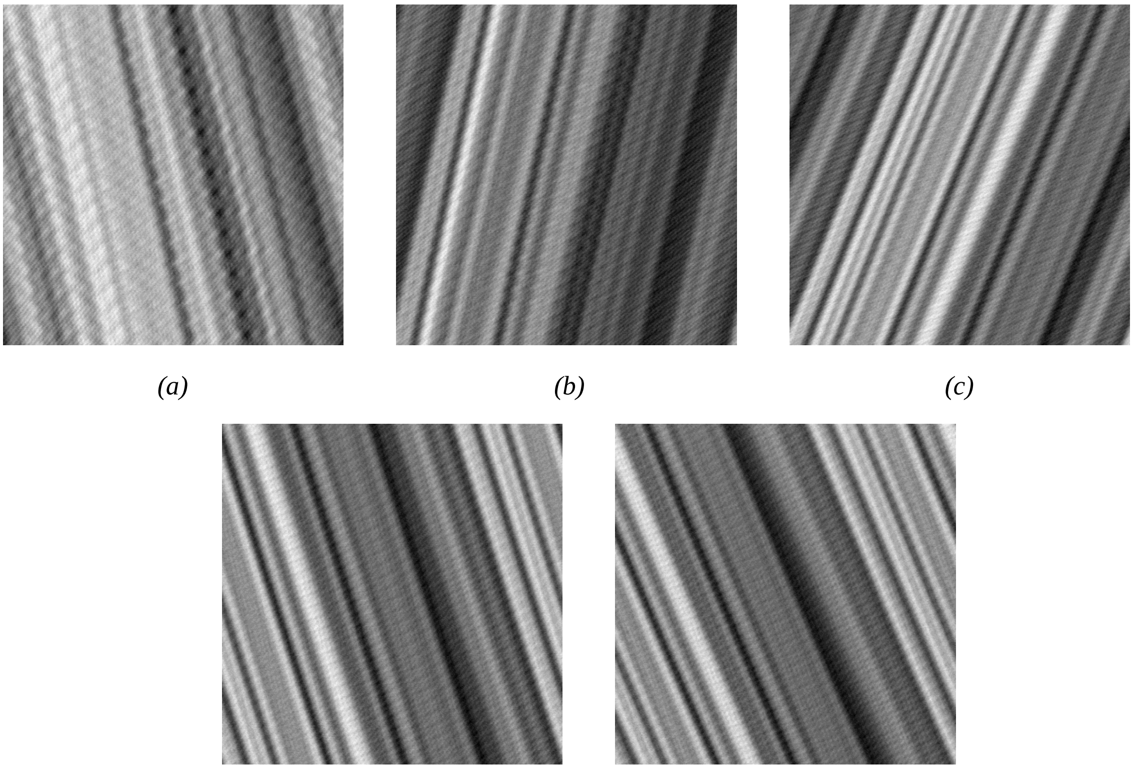


5BB *BFA*



3BB

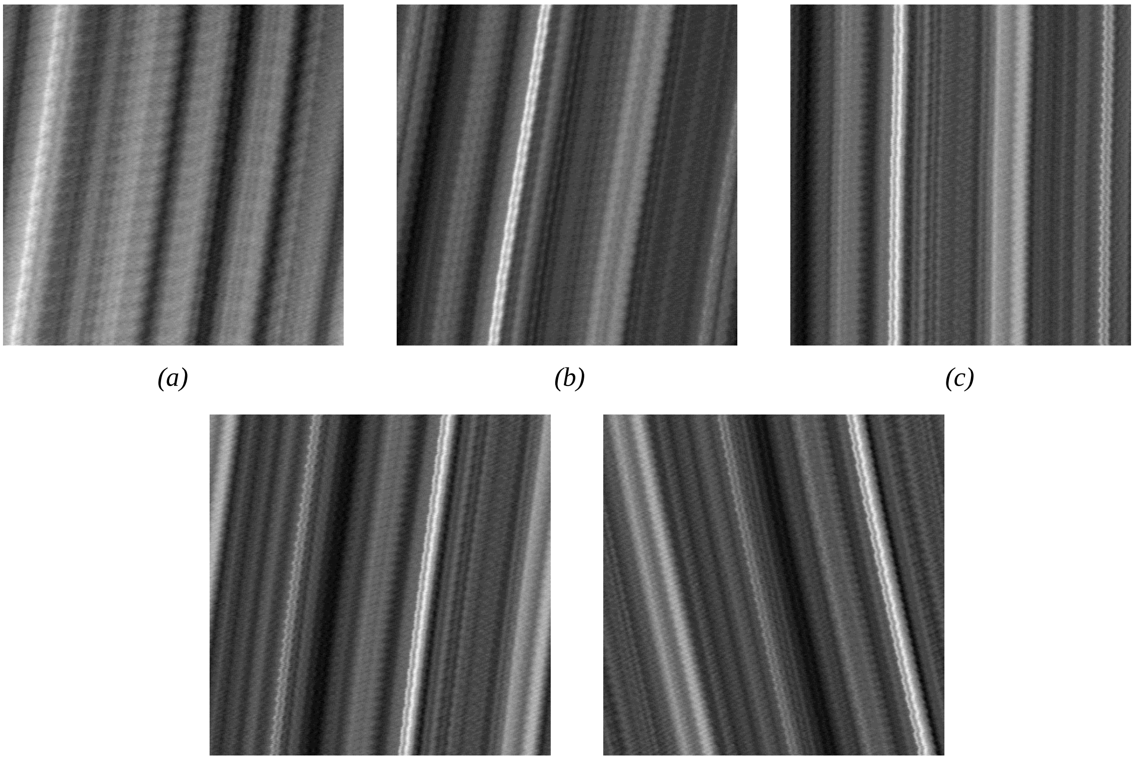
Figure 4.11. *2D Grayscale Images of the z- axis vibration data under 4.1 A*



(a) (b) (c)

(d) (e)

Figure 4.12. *2D Grayscale Images of the x- axis vibration data from the healthy motor under (a) 3.6 A (b) 4.1 A (c) 4.7 A (d) 4.9 A (e) 5 A*



(a) (b) (c)

(d) (e)

Figure 4.13. *2D Grayscale Images of the x- axis vibration data from 5BB under (a) 3.6 A (b) 4.1 A (c) 4.7 A (d) 4.9 A (e) 5 A*

The 5 motors, 5 different load conditions with 3 experiments each (5 of BFA) and 3 different axis produce 85 grayscale images. Those images are not proper for CNN by themselves. Instead each image firstly divided into a smaller part in order to use them for training and test in CNN classification. For this approach two different model have been created with 40X40 images model is used, each gray level image produces 100 small size images. According to this, each induction motor has 1.500 images, except BFA (has 2.500 images).

Figure 4.14 shows multiple image formation from the 1D image. In this process it is obvious that every gray scale image has been segmented 100 different size images which forms the dataset of CNN.

In the second model, the gray level images from vibration data are also divided into 50x50 size for comparison with the model of 40x40 size of the images. In this time, 960 pictures are created from a one motor fault type (each image gives 64 small images), except BFA (has 1600 test images). In this thesis, a comparison of the two datasets coming from 40x40 and 50x50 size of images is given.

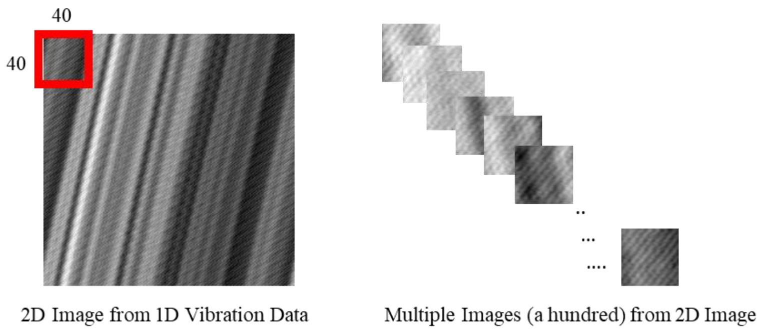
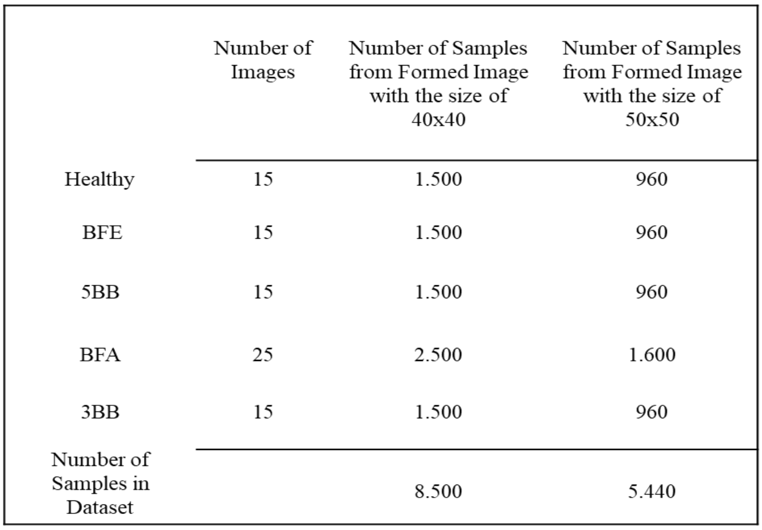

40
40
2D Image from 1D Vibration Data
..
...
....
Multiple Images (a hundred) from 2D Image

Figure 4.14. *Multiple Image Formation from 2D Grayscale Image of healthy motor under 4.1 A.*

The following Table 4.2 displays that the number of samples in an image database prepared for classification of the induction motor faults.

Table 4.2. *The number of samples in the dataset prepared for classification of the induction motor faults.*



	Number of Images	Number of Samples from Formed Image with the size of 40x40	Number of Samples from Formed Image with the size of 50x50
Healthy	15	1.500	960
BFE	15	1.500	960
5BB	15	1.500	960
BFA	25	2.500	1.600
3BB	15	1.500	960
Number of Samples in Dataset		8.500	5.440

4.1.1.2 3D Image represantation from combination of 1D vibration data

In the previous section, 2D images were extracted from one-dimensional data. In this section, 3D images were created using the vibration data of each axis of the induction motor. X, Y and Z axis of the vibration data were recorded during the acquiring the whole data. Thus, these three axes can be combined and converted into a new color image of the existing data. A color image consists of three dimensions that include color information of each dimension and so each pixel. It has red, green and blue component, in this way, x, y and z-axes are appointed to red, green and blue component respectively.

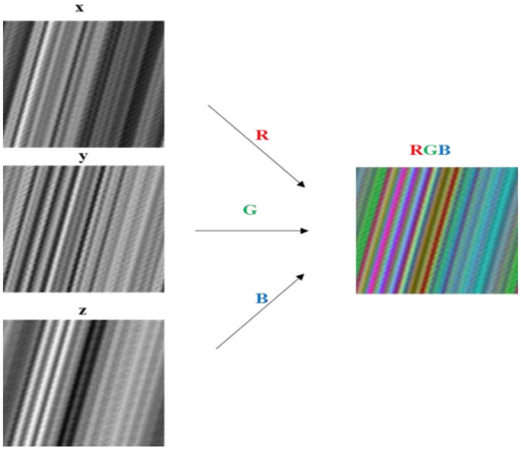
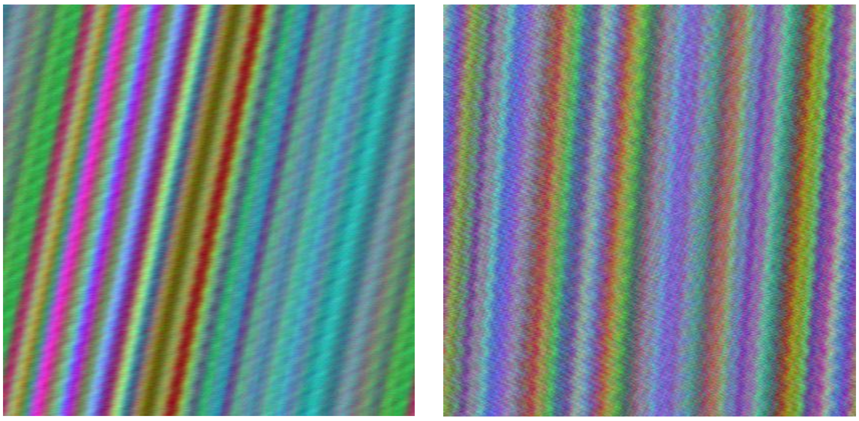
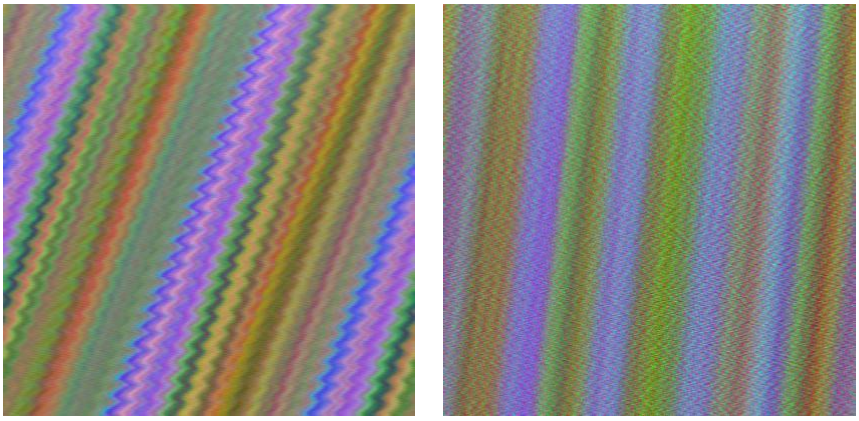


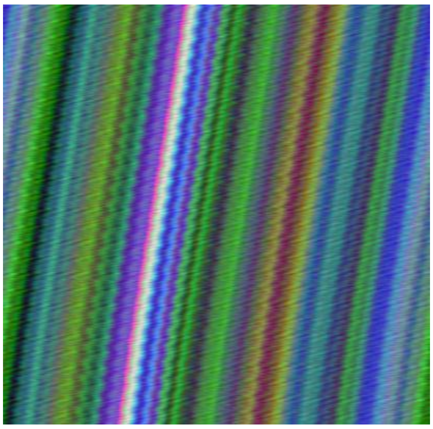
Figure 4.15. *The formation of color image from the combination of three axes*



Healthy *BFE*



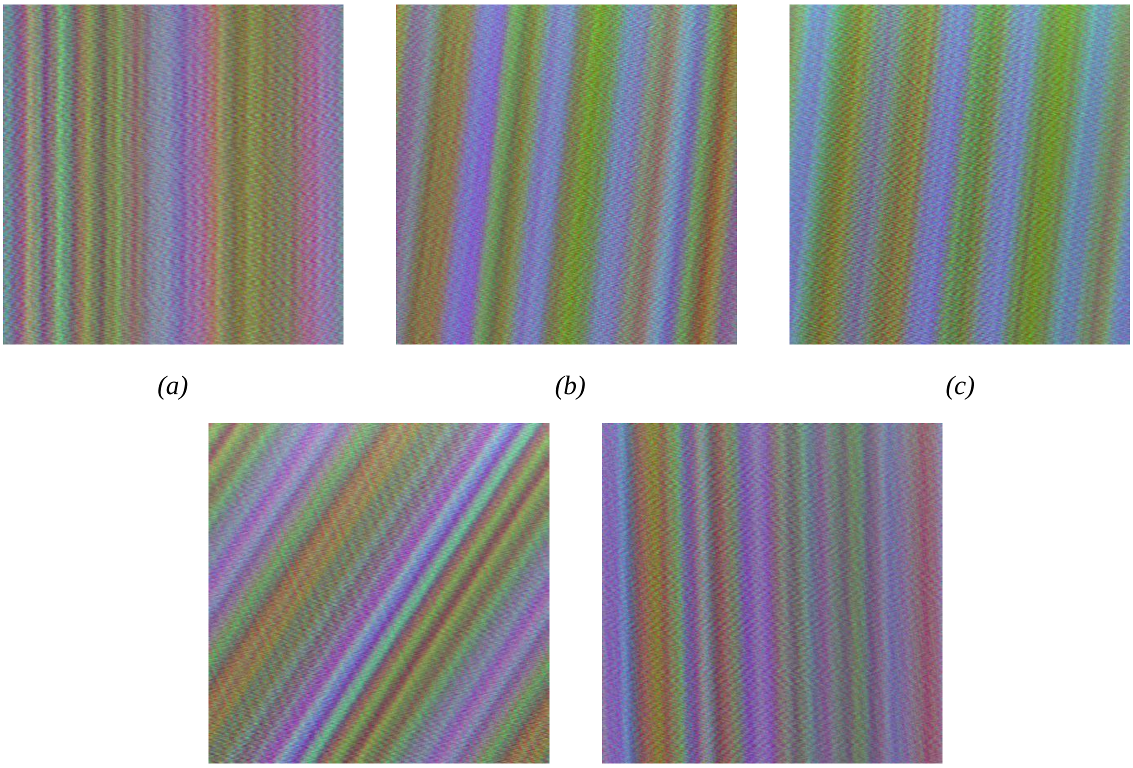
5BB *BFA*



3BB

Figure 4.16. *3D Color Images under 4.1 A*

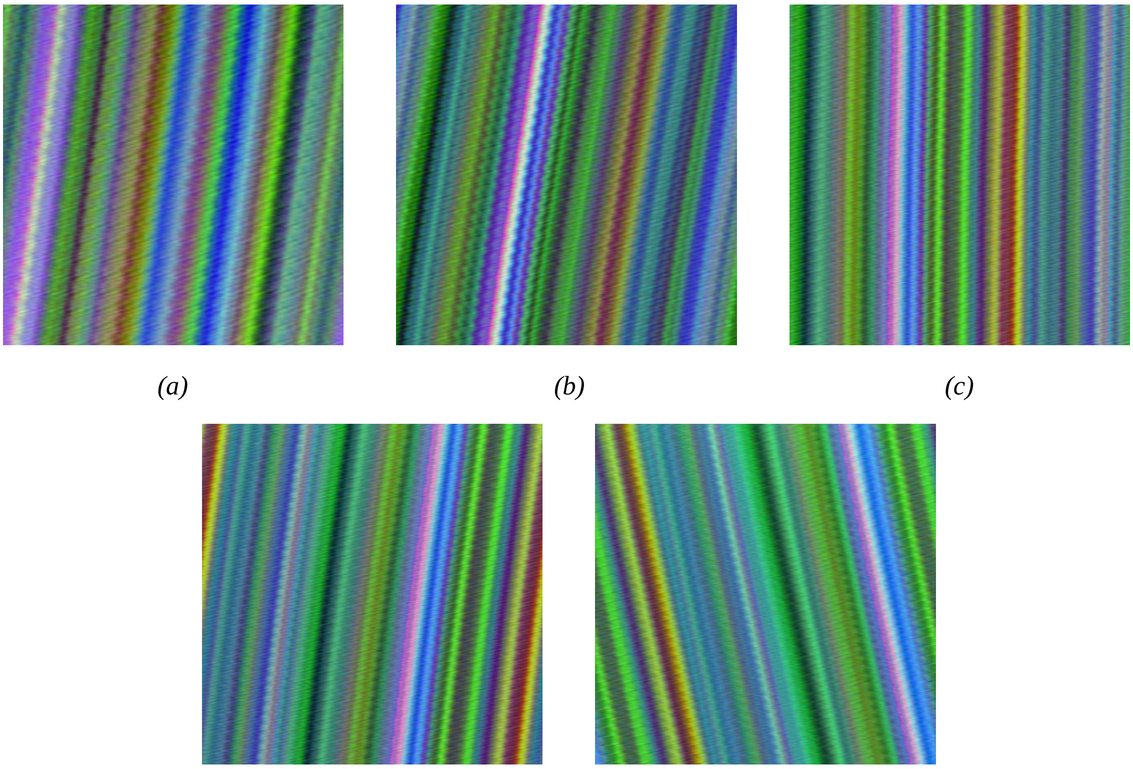
Figure 4.16 is about formed color images according to the combination of the x,y, z-axes values. It shows five distinct data of motor fault types under 4.1 Amperes. Following the previous method, the formed color images were split into 40x40 and 50x50 images in order to compose the dataset of the network. Then, the number of samples in the dataset for colored ones are prepared for classification of the induction motor faults. Moreover, 3D images constructed from the same induction motor data reflect similar character traits, even if they are under different loading conditions, just like the 2D grayscale ones. Figure 4.17 and 4.18 show this by 3D color images which is constructed from BFA and 3BB respectively under distinct loading conditions.



(a) *(b)* *(c)*

(d) *(e)*

Figure 4.17. *3D Color Images of BFA under (a)3.6 A (b) 4.1 A (c) 4.7A (d) 4.9 A (e) 5 A*



(a) *(b)* *(c)*

(d) *(e)*

Figure 4.18. *3D Color Images of 3BB under (a)3.6 A (b) 4.1 A (c) 4.7A (d) 4.9 A (e) 5 A*

4.2. Classification Result

In order to use CNN, many parameters need to be set such as the number of convolution layer, number of the filter, epoch, batch size, kernel size, and optimizer. As already mentioned, CNN needs iterations in order to update training process. These iterations are called *epoch.* Namely, one epoch consists of all the training data set processed back and forths (Sharma, 2017). *Kernel size* is the parameter that determines the height and width of the 2D convolution window (Keras, http). The number of filters is the number of output filters in the convolution. In addition, *batch size* is the number of training samples in one iteration. *Iteration* as a term is different from epoch and batch size. Iteration is the number of batches required to complete an epoch. As mentioned in Chapter 3, optimizers, which are used in this thesis, are SGD, SGD with momentum, SGD with Nesterov, SGD with momentum and Nesterov, Adam and Nadam.

The first approach to classify the vibration data is based on the x-axis dataset. The parameters which are better to classify have been focused on the x-axis dataset for subsequent experiments. Moreover, 80 % of the dataset is left as a train set and remainder set to as a test set during the whole experiments.

As shown in Table 4.3, various experiments were performed on the basis of x-axis image dataset according to different parameters. The number of the filter is seen like 2-32, 2-64 mean that first two Conv2D layers use a number of 32 filters, following two Conv2D layers use a number of 64 filters as shown in Table 4.5. The following experiments on the whole dataset contain 50 epoch, 4 Conv2D layers that 2 of them have 32 filters and the others have 64, each Conv2D layer has (3,3) kernel size. Because, as shown by the Table 4.3, if the number of epoch size and convolution are increased, it is observed that the test accuracy is getting higher. Although the number of epochs is not so large, the test accuracy is not so bad. Conversely, if the kernel size is increased, the test accuracy is decreased. Batch size is also like the kernel size, it is better at low values for classification. But, according to Table 4.3, there is not a big difference in the batch size. In addition, batch size is set to 32 and batch normalization is used for following experiments. Up to the last activation (5 activations), ReLU is used and the last one is Softmax function.

Furthermore, the model was created using these parameters and the Table 4.4 and Table 4.5 show the outline of it for the size of 40x40 and 50X50 of 2D grayscale image dataset respectively. According to Table 4.4 and 4.5, total 4 convolution layers are used. The first two of the convolution layers use the number of 32 filters with the kernel size of (3,3) and the next two convolution layers use the number of 64 filters with the kernel size of (3,3). Batch normalization and activation- ReLU are used after each convolution layer. Note that batch normalization and ReLU have no affect on output shape. After the first two convolution layer, batch normalization and ReLU, the size of (2,2) max pooling is used in order to reduce the output shape.

Table 4.3. *Parameters tested for the finding best classification of the dataset*

Exp	Test Accuracy	# Of Conv 2D	# Of Filter	Kernel Size	Batch Norm.	Optimizer	Epochs	Batch Size
1	0.9064	2	32	3,3	not used	SGD	5	32
2	0.9329	2	32	3,3	not used	SGD	10	32
3	0.9582	2	32	3,3	used	SGD	5	32
4	0.8941	2	64	3,3	used	SGD	5	32
5	0.9535	2	32	5,5	used	SGD	5	32
6	0.9552	4	2-32 2-64	3,3	used	SGD	5	32
7	0.9688	4	2-32 2-64	3,3	used	SGD	10	32
8	0.9976	4	2-32 2-64	3,3	used	ADAM	10	32
9	0.9952	4	2-32 2-64	3,3	used	ADAM	10	64

After the next two of them, (2,2) max pooling is reused. Flatten is used to transmit to the fully connected layer. The first dense function regulates the neurons which come from the flatten part. The last dense operation is appointed according to output labels which are healthy, BFE, 5BB, BFA, and 3BB in this thesis. In addition to this, with the use of last activation, inputs are dispatched to the corresponding labels. There is one dropout operation in the model to make the network less complicated.

As seen in Table 4.4, a different output shape (None, 38, 38, 32) is seen, after the first convolution layer. This shape can be calculated from the equation 3.7 in Chapter 3:

$$out = \frac{in - receptive\ field + 2 * padding}{stride} + 1 \tag{3.7}$$

After the shape of input image (40,40,1) output of convolution layer gives, (None, 38, 38, 32) means that out = 38 and 32 is the number of filters with the size of 3. Padding is the same which means there is no padding in here. So, the padding value is zero. Moreover, the stride is the default value which is 1. According to Table 4.4, the out is:

$$out = \frac{40 - 3 + 2 * 0}{1} + 1 \quad = 38 \tag{4.1}$$

Each filter has 3*3= 9 parameters plus bias. In addition to this, there are 32 filters. According to this after the first convolution layer, there are (9+1) *32 = 320 parameters as seen from the Table 4.4. The same procedure is repeated for the other convolution layers. In addition, (2,2) max-pooling operation reduces the size of the output height and width but not depth, after the activation. Moreover, the reason for the output shape seen in the flatten() is that the max-pooling from the previous layer (None, 7, 7, 64) gives 7*7*64 = 3136.

Table 4.4. *Summary of model of the CNN for the size of 40X40 of 2D Grayscale Images classification*

Layer (type)	**Output Shape**	**Param #**
conv2d_1 (Conv2D)	(None, 38, 38, 32)	320
batch_normalization_1(Batch)	(None, 38, 38, 32)	128
activation_1 (Activation)	(None, 38, 38, 32)	0
conv2d_2 (Conv2D)	(None, 36, 36, 32)	9248
batch_normalization_2 (Batch)	(None, 36, 36, 32)	128
activation_2 (Activation)	(None, 36, 36, 32)	0
max_pooling2d_1 (MaxPooling2)	(None, 18, 18, 32)	0
conv2d_3 (Conv2D)	(None, 16, 16, 64)	18496
batch_normalization_3 (Batch)	(None, 16, 16, 64)	256
activation_3 (Activation)	(None, 16, 16, 64)	0
conv2d_4 (Conv2D)	(None, 14, 14, 64)	36928
batch_normalization_4 (Batch)	(None, 14, 14, 64)	256
activation_4 (Activation)	(None, 14, 14, 64)	0
max_pooling2d_2 (MaxPooling2)	(None, 7, 7, 64)	0
flatten_1 (Flatten)	(None, 3136)	0
dense_1 (Dense)	(None, 512)	1606144
batch_normalization_5 (Batch)	(None, 512)	2048
activation_5 (Activation)	(None, 512)	0
dropout_1 (Dropout)	(None, 512)	0
dense_2 (Dense)	(None, 5)	2565
activation_6 (Activation)	(None, 5)	0
	Total params:	1,676,517
	Trainable params:	1,675,109
	Non-Trainable params:	1,408

As shown in Table 4.5, when the same procedure is applied to the size of 50X50 2D grayscale images, number of parameters are increased because of the size of images. Moreover, the same procedure is also applied to the size of 40X40 and 50x50 color images. However, because of the nature of the color images, number of parameters are changed since depth has been increased by 3.

Table 4.5. *Summary of model of the CNN for the size of 50X50 of 2D Grayscale Images classification*

Layer (type)	**Output Shape**	**Param #**
conv2d_1 (Conv2D)	(None, 48, 48, 32)	320
batch_normalization_1(Batch)	(None, 48, 48, 32)	128
activation_1 (Activation)	(None, 48, 48, 32)	0
conv2d_2 (Conv2D)	(None, 46, 46, 32)	9248
batch_normalization_2 (Batch)	(None, 46, 46, 32)	128
activation_2 (Activation)	(None, 46, 46, 32)	0
max_pooling2d_1 (MaxPooling2)	(None, 23, 23, 32)	0
conv2d_3 (Conv2D)	(None, 21, 21, 64)	18496
batch_normalization_3 (Batch)	(None, 21, 21, 64)	256
activation_3 (Activation)	(None, 21, 21, 64)	0
conv2d_4 (Conv2D)	(None, 19, 19, 64)	36928
batch_normalization_4 (Batch)	(None, 19, 19, 64)	256
activation_4 (Activation)	(None, 19, 19, 64)	0
max_pooling2d_2 (MaxPooling2)	(None, 9, 9, 64)	0
flatten_1 (Flatten)	(None, 5184)	0
dense_1 (Dense)	(None, 512)	2654720
batch_normalization_5 (Batch)	(None, 512)	2048
activation_5 (Activation)	(None, 512)	0
dropout_1 (Dropout)	(None, 512)	0
dense_2 (Dense)	(None, 5)	2565
activation_6 (Activation)	(None, 5)	0
	Total params:	2,725,669
	Trainable params:	2,724,261
	Non-Trainable params:	1,408

The number of parameters that are trainable and non-trainable is also given. Non-trainable parameters are about weights that are not updated by the system during the training stage. The default value of it is zero, 1,408 parameters are kept constant according to this model.

4.2.1. Classification results for images from the x-axis of the vibration dataset

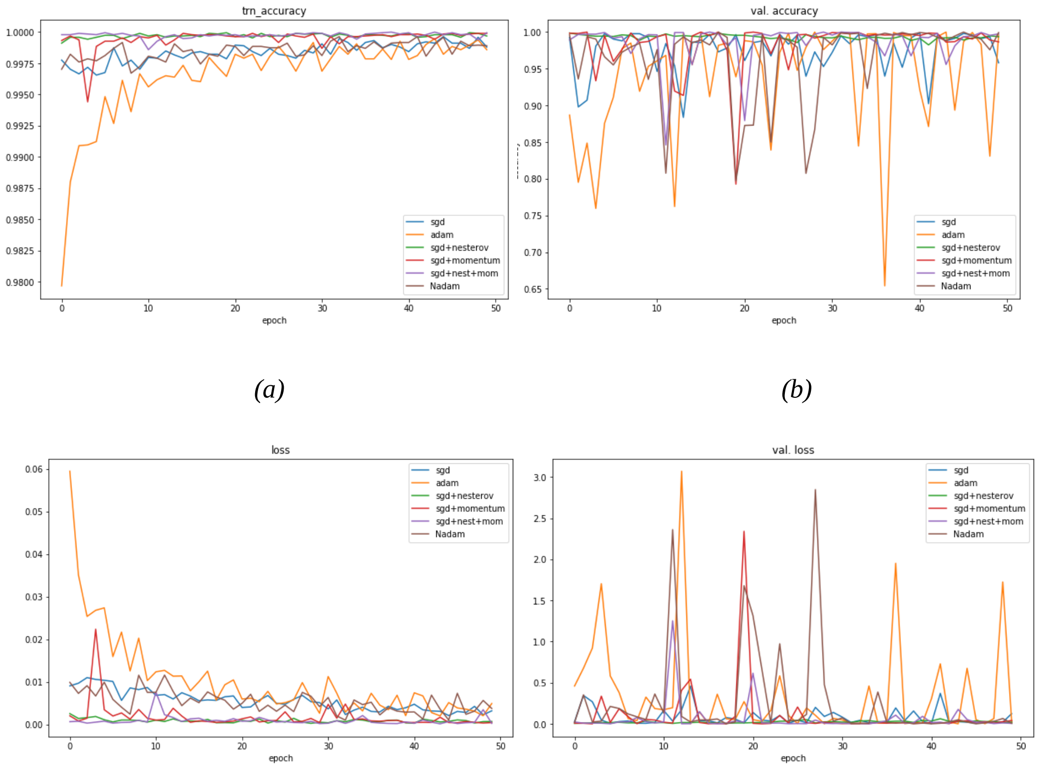
4.2.1.1. Classification results for the size of 40x40 Images from x -axis of the vibration dataset

Classification results for the size of 40X40 images from the x-axis of the vibration dataset regardless of various optimization technique can be seen from the Table 4.6. It seems that SGD and SGD with momentum give lower test accuracy compared to others.

Table 4.6. *Test Accuracy for 40x40 Images formed of x-axis of the vibration data classification after 50 epochs with different optimizer*

Optimization Technique	Test_accuracy
SGD	0.95
SGD + Nesterov	0.99
SGD + Momentum	0.98
SGD + Nesterov +Momentum	0.99
Adam	0.99
Nadam	0.99

Although Nadam and Adam have good test accuracy and training performance, it does not imply a good approximation on their validation accuracy according to Figure 4.19 (a), (b). Training and validation accuracy are similar in SGD Nesterov that is the green line in Figure 4.19 and thus, in here model will give the best result to input images for classification. Unlike test accuracy from the Table 4.6, the SGD has low validation loss. Only SGD with Nesterov and SGD with Nesterov and Momentum fit the model after 50 epochs compared to others which are still underfitting. This could be a small number of epochs. Figure 4.19 (c) and (d) gives training loss and validation loss respectively. By looking at these figures, the learning rate is usually good. The amount of "wiggle" in the loss Figure 4.19 is about to the batch size (http-8).



(a) *(b)*

(c) *(d)*

Figure 4.19. *(a) Train accuracy (b) Validation accuracy (c) Train loss (d) Validation loss for 40x40 Images of the X-axis of the vibration data classification after 50 epochs with different optimizers*

For more detailed information, *confusion matrix* for 40x40 images from the x-axis of the vibration data are given in Table 4.7 to Table 4.12. *Confusion matrix* or *error matrix* is often used to check the output performance of CNN for target and actual values of classification the target qualities. In other words, a confusion matrix gives summary of results of prediction of the model. Thus, by checking confusion matrix, it is easy to understand the similarities of the classification result. For example, two results may have the same test accuracy value under different optimization techniques, but with confusion matrix, it is possible to estimate the output quality of the model in more detail. According to the confusion matrix in Table 4.7, BFE is thought to BFA using SGD in some cases. Moreover, 5BB is confused with BFA. When the Table 4.8 to 4.12 are checked, it can be deducted that SGD with Nesterov and Momentum produce better performance compared to others from point of view of classification.

Table 4.7. *Confusion Matrix for 40x40 Images of the X-axis vibration data classification using SGD*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	284	0	0	0	0
BFE	0	241	0	67	0
5BB	0	0	318	3	0
BFA	0	0	0	518	0
3BB	0	0	0	0	269

Table 4.8. *Confusion Matrix for 40x40 Images of the X-axis vibration data classification using SGD + Nesterov*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	284	0	0	0	0
BFE	0	300	0	8	0
5BB	0	0	320	1	0
BFA	0	0	0	518	0
3BB	0	0	0	0	269

Table 4.9. *Confusion Matrix for 40x40 Images of the X-axis vibration data classification using SGD + momentum*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	284	0	0	0	0
BFE	0	288	0	20	0
5BB	0	0	320	1	0
BFA	0	0	0	518	0
3BB	0	0	0	0	269

Table 4.10. *Confusion Matrix for 40x40 Images of the X-axis vibration data classification using SGD + Nesterov +momentum*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	284	0	0	0	0
BFE	0	304	0	4	0
5BB	0	0	321	0	0
BFA	0	0	0	518	0
3BB	0	0	0	0	269

Table 4.11. *Confusion Matrix for 40x40 Images of the X-axis vibration data classification using Adam*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	284	0	0	0	0
BFE	0	304	0	4	0
5BB	0	0	319	2	0
BFA	0	0	0	518	0
3BB	0	0	1	0	268

Table 4.12. *Confusion Matrix for 40x40 Images of the X-axis vibration data classification using Nadam*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	284	0	0	0	0
BFE	0	304	0	4	0
5BB	0	1	320	0	0
BFA	0	0	0	518	0
3BB	0	0	0	0	269

According to this part, SGD with Momentum and Nesterov gives the best classification results for 40x40 images from the x-axis of the vibration data.

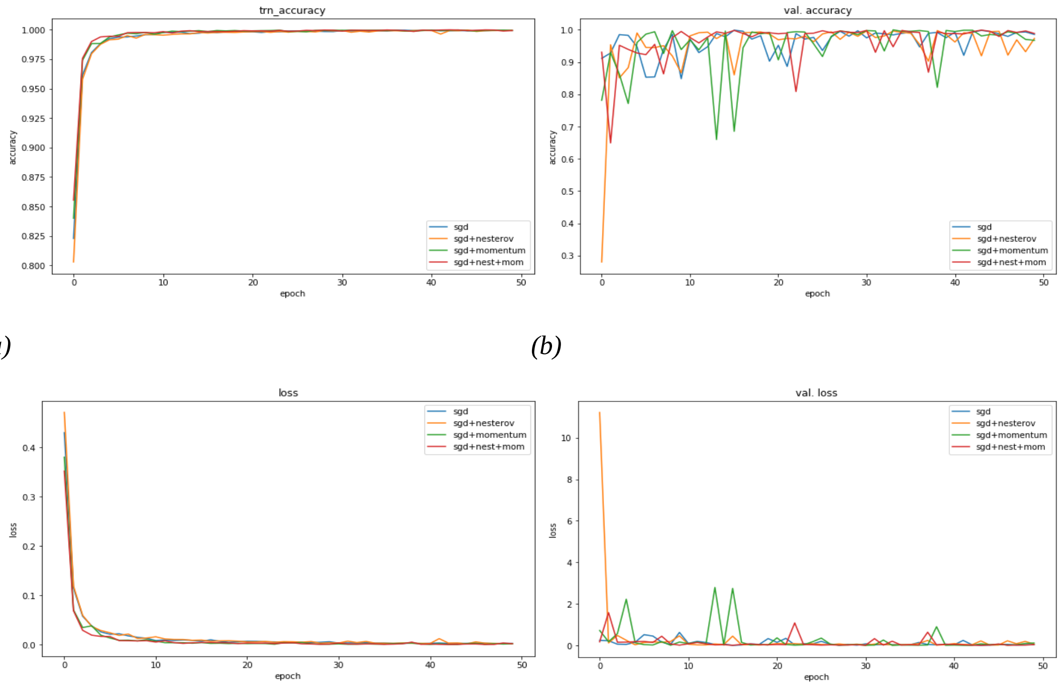
4.2.1.2. Classification results for the size of 50x50 images from x -axis of the vibration dataset

In this section, the classification of 50x50 Images from the x-axis vibration dataset does not include adam and nadam optimizers. The same procedure will be applied to images of 50x50 size in further experiments. By looking at Table 4.13, SGD classifies better with images of size 50x50 than 40x40. For 50X50 images, SGD and SGD with momentum and Nesterov show best test accuracy.

Table 4.13. *Test Accuracy for 50x50 Images of x-axis of the vibration data using different Optimizer*

Optimization Technique	Test_accuracy
SGD	0.98
SGD + Nesterov	0.97
SGD + Momentum	0.96
SGD + Nesterov +Momentum	0.98

As shown in the Figure 4.20 (a) and (b), validation accuracy have ripples in many places compared to training accuracy. According to this, although the test accuracy gives good results, it is possible to deduce the model needs more epochs. Moreover, there is not a big difference between the lines, it is possible to say that there is no overfitting. Figure 4.20 (c) and (d) show that validation loss is nearly the same as the training loss. This gives the learning rate is good enough. So in those experiments it is possible to say that the learning rate is chosen properly.



(a) *(b)*

(c) *(d)*

Figure 4.20. *(a) Train accuracy (b) Validation accuracy (c) Train loss (d) Validation loss for 50x50 Images from X-axis of the vibration data classification after 50 epochs using different optimizers*

When we look at to the confusion matrix which are given in Table 4.14 to Table 4.17, classification using SGD with momentum and Nesterov gives the best result in this time, too. This is about the combination of the Nesterov and momentum works well for the classification. Images of size 40x40 from x-axis vibration data regardless of the SGD with momentum and Nesterov overcomes the images of size 50x50 for classification.

Table 4.14. *Confusion Matrix for 50x50 Images of X-axis of the vibration data classification using SGD*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	198	0	0	0	0
BFE	0	187	0	9	0
5BB	0	2	199	5	0
BFA	0	0	0	285	0
3BB	0	0	0	0	203

Table 4.15. *Confusion Matrix for 50x50 Images of X-axis of the vibration data classification using SGD + Nesterov*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	197	1	0	0	0
BFE	0	173	0	23	0
5BB	0	2	201	3	0
BFA	0	0	0	285	0
3BB	0	0	0	0	203

Table 4.16. *Confusion Matrix for 50x50 Images of X-axis of the vibration data classification using SGD + Momentum*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	198	0	0	0	0
BFE	0	175	0	21	0
5BB	0	13	191	2	0
BFA	0	0	0	285	0
3BB	0	0	0	0	203

Table 4.17. *Confusion Matrix for 50x50 Images of X-axis of the vibration data classification using SGD + Nesterov + Momentum*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	198	0	0	0	0
BFE	0	187	0	9	0
5BB	0	0	203	3	0
BFA	0	0	0	285	0
3BB	0	0	0	0	203

In order to conclude this Section 4.2.1, it is necessary to stress that CNN confuses BFE and BFA, and also SGD with momentum and Nesterov serves the best result for images from x-axis vibration dataset.

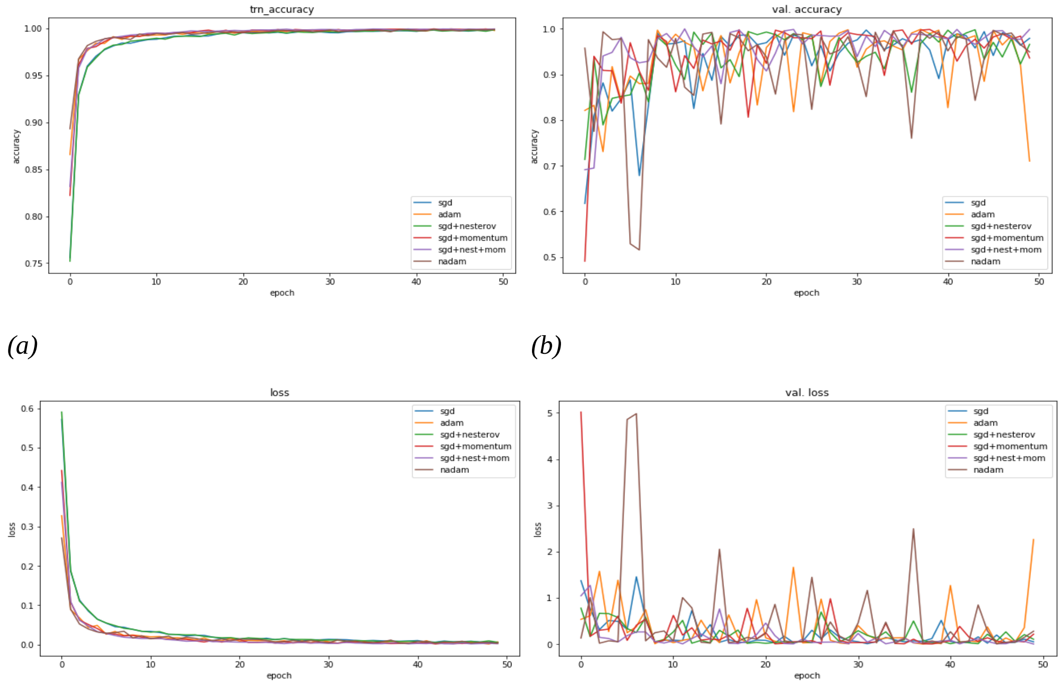
4.2.2. Classification results for images from y-axis of the vibration dataset

4.2.2.1. Classification results for the size of 40x40 images from the y-axis of the vibration dataset

Table 4.20 demonstrates that the result of SGD with momentum and Nesterov comes out on top. Looking at deeper with the Figure 4.21 (a)-(d), its validation accuracy does not fluctuate so much like others and the learning rate is fit better**.**

Table 4.18. *Test Accuracy for 40x40 Images from y-axis of the vibration data classification after 50 epochs with various optimizers*

Optimization Technique	Test_accuracy
SGD	0.98
SGD + Nesterov	0.96
SGD + Momentum	0.93
SGD + Nesterov +Momentum	0.99
Adam	0.71
Nadam	0.94



(a) *(b)*

(c) *(d)*

Figure 4.21. *(a) Train accuracy (b) Validation (c) Train loss (d) Validation loss for 40x40 Images from Y-axis of the vibration data classification after 50 epochs with different optimizers*

A detailed look for classification error is shown in the Table 4.19 to 4.24. Not only there is a confusion between BFE and BFA for images from y-axis vibration, some of fault types are confused.

Table 4.19. *Confusion Matrix for 40x40 Images of Y-axis of the vibration data classification using SGD*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	305	0	0	0	0
BFE	0	251	0	24	0
5BB	0	0	276	0	0
BFA	0	0	0	531	0
3BB	0	0	9	0	304

Table 4.20. *Confusion Matrix for 40x40 Images of Y-axis of the vibration data classification using SGD + Nesterov*

Predicted

	Healthy	BFE	5BB	BFA	3BB
Healthy	298	0	1	6	0
BFE	0	227	0	48	0
5BB	0	0	272	2	2
BFA	0	0	0	531	0
3BB	0	0	0	0	313

Table 4.21. *Confusion Matrix for 40x40 Images of Y-axis of the vibration data classification using SGD + Momentum*

Predicted

	Healthy	BFE	5BB	BFA	3BB
Healthy	305	0	0	0	0
BFE	0	170	0	105	0
5BB	0	0	274	0	2
BFA	0	0	0	531	0
3BB	0	0	0	0	313

Table 4.22. *Confusion Matrix for 40x40 Images of Y-axis of the vibration data classification using SGD + Momentum + Nesterov*

Predicted

	Healthy	BFE	5BB	BFA	3BB
Healthy	305	0	0	0	0
BFE	0	275	0	0	0
5BB	0	0	274	0	2
BFA	0	0	0	531	0
3BB	0	0	0	0	313

Table 4.23. *Confusion Matrix for 40x40 Images of Y-axis of the vibration data classification using Adam*

Predicted

	Healthy	BFE	5BB	BFA	3BB
Healthy	305	0	0	0	0
BFE	22	185	0	68	0
5BB	211	0	50	0	15
BFA	0	0	0	531	0
3BB	177	0	0	0	136

Table 4.24. *Confusion Matrix for 40x40 Images of Y-axis of the vibration data classification using Nadam*

Predicted

	Healthy	BFE	5BB	BFA	3BB
Healthy	296	0	9	0	0
BFE	0	201	0	74	0
5BB	0	0	276	0	0
BFA	0	0	0	531	0
3BB	0	0	3	0	310

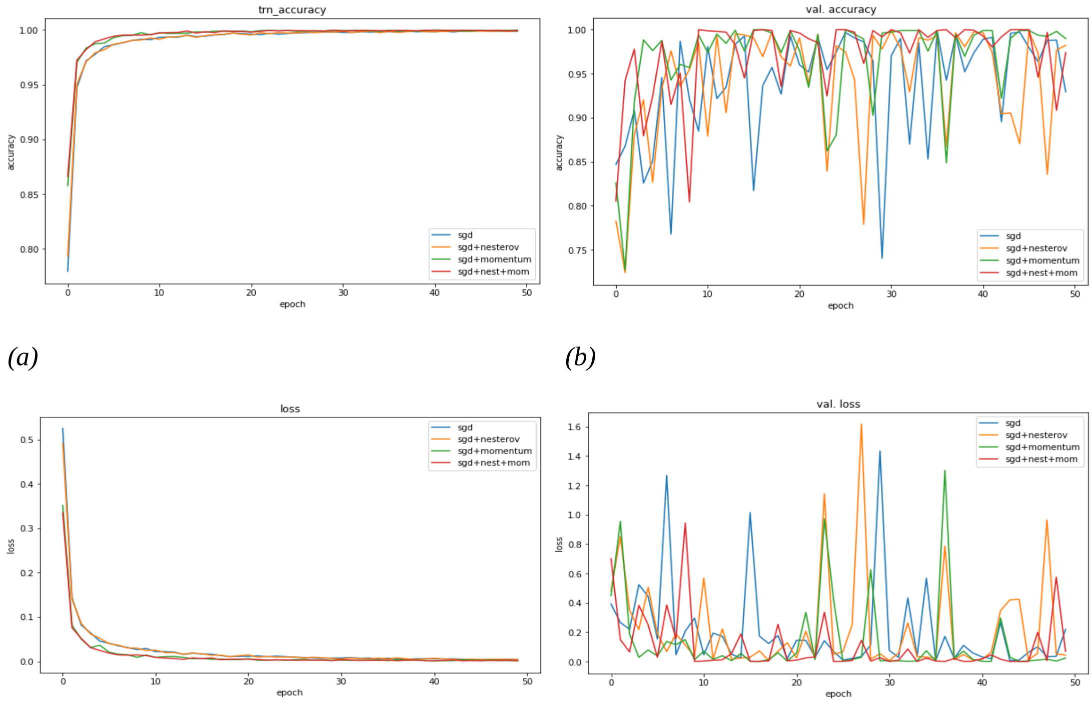
4.2.2.2. Classification results for the size of 50x50 images from the y-axis of the vibration dataset

From Table 4.25, CovnNet works best with the SGD+Momentum and SGD+Nesterov than others.

Table 4.25. *Test Accuracy for 50x50 Images of y-axis of the vibration data classification after 50 epochs with Various Optimizer*

Optimization Technique	Test_accuracy
SGD	0.92
SGD + Nesterov	0.98
SGD + Momentum	0.98
SGD + Nesterov +Momentum	0.97

There are ripples between the values as shown in Figure 4.22 (b). This shows that more epochs would be better for the 50x50 images from y-axis vibration dataset. Furthermore, the learning rate is not fit well as seen from the Figure 4.22 (d).



(a) *(b)*

(c) *(d)*

Figure 4.22. *(a) Train accuracy(b) Validation accuracy (c) Train loss (d) Validation loss for 50x50 Images from Y-axis of the vibration data classification after 50 epochs with various optimizers*

Classification results regardless of all optimizers demonstrate that BFE is still supposed as BFA, even in the use of SGD with momentum which is the best in this part.

Table 4.26. *Confusion Matrix for 50x50 Images of Y-axis of the vibration data classification using SGD*

Predicted

	Healthy	BFE	5BB	BFA	3BB
Healthy	198	0	0	0	0
BFE	2	128	2	72	0
5BB	1	0	187	0	1
BFA	0	0	0	317	0
3BB	0	0	1	0	178

Table 4.27. *Confusion Matrix for 50x50 Images of Y-axis of the vibration data classification using SGD + Nesterov*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	198	0	0	0	0
BFE	0	187	0	18	0
5BB	0	0	188	0	1
BFA	0	0	0	317	0
3BB	0	0	0	0	179

Table 4.28. *Confusion Matrix for 50x50 Images of Y-axis of the vibration data classification using SGD + Momentum*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	198	0	0	0	0
BFE	0	194	0	11	0
5BB	0	0	189	0	0
BFA	0	0	0	317	0
3BB	0	0	0	0	179

Table 4.29. *Confusion Matrix for 50x50 Images of Y-axis of the vibration data classification using SGD + Nesterov + Momentum*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	198	0	0	0	0
BFE	0	176	0	29	0
5BB	0	0	189	0	0
BFA	0	0	0	317	0
3BB	0	0	0	0	179

4.2.3. Classification results for images from z-axis of the vibration dataset

4.2.3.1. Classification results for the size of 40x40 images from the z-axis of the vibration dataset

In Table 4.32, optimization techniques which are SGD with momentum and SGD with momentum and Nesterov have higher test accuracy than the others. Moreover, validation accuracy from using SGD with momentum and Nesterov has nearly the same training accuracy according to Figure 4.23 (a) and (b). In addition to this, it can be deducted from the Figure 4.23 (c) and (d), the learning rate is good enough for SGD with Nesterov, SGD with momentum and, also SGD with momentum-Nesterov.

Table 4.30. *Test Accuracy for 40x40 Images of z-axis of the vibration data classification after 50 epochs with different optimizer*

Optimization Technique	Test_accuracy
SGD	0.95
SGD + Nesterov	0.97
SGD + Momentum	0.98
SGD + Nesterov +Momentum	0.98
Adam	0.92
Nadam	0.92

For a more detailed look, Table 4.31 to Table 4.36 of the confusion matrices give a deeper information. It seems that images in BFE of size 40x40 from z-axis vibration data are also confused with the images in BFA by the system. The top result which is with SGD and momentum shows that healthy motor and 5BB motor are supposed to in 3BB label.

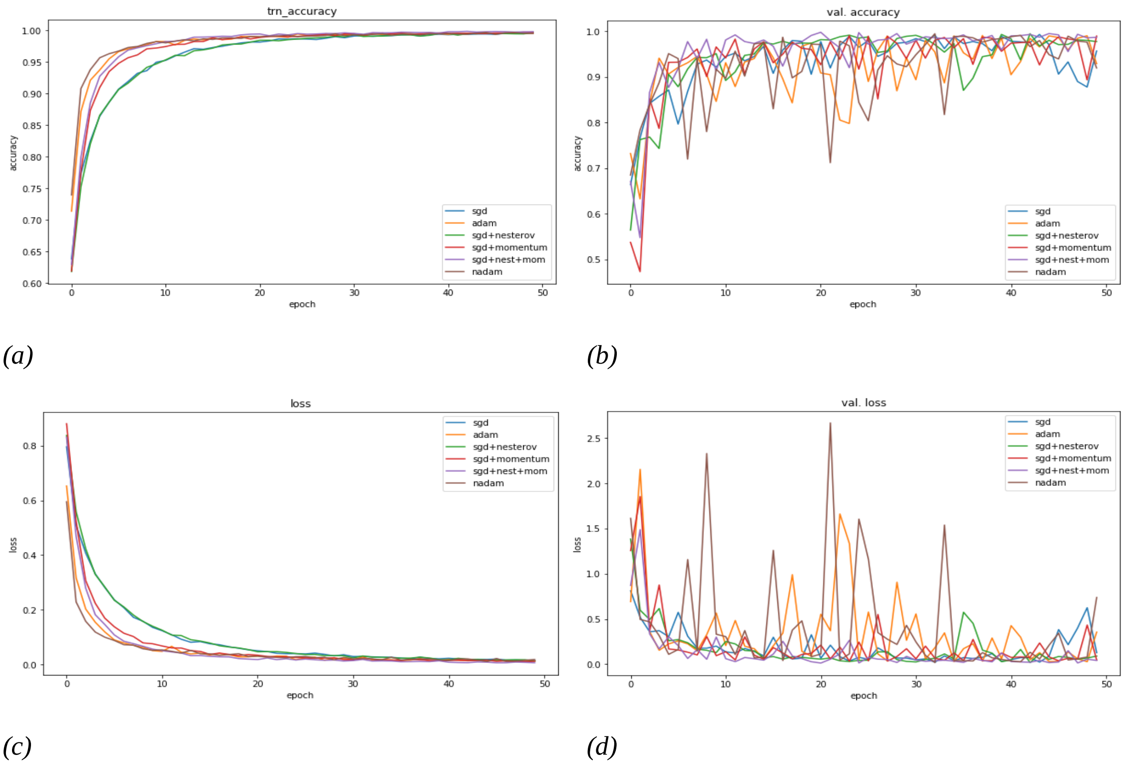


Figure 4.23. *(a) Train accuracy (b) Validation accuracy (c) Train loss (d) Validation loss for 40x40 Images from Z-axis of the vibration data classification after 50 epochs with different optimizers*

Table 4.31. *Confusion Matrix for 40x40 Images from Z-axis of the vibration data classification using SGD*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	303	0	1	0	0
BFE	0	253	0	58	0
5BB	0	0	272	1	12
BFA	0	0	0	515	0
3BB	0	0	4	0	281

Table 4.32. *Confusion Matrix for 40x40 Images of Z-axis of the vibration data classification using SGD + Nesterov*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	304	0	0	0	0
BFE	1	272	0	38	0
5BB	3	0	285	0	0
BFA	0	0	0	515	0
3BB	0	0	1	0	284

Table 4.33. *Confusion Matrix for 40x40 Images of Z-axis of the vibration data classification using SGD + Momentum*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	303	0	0	0	1
BFE	0	311	0	0	0
5BB	1	7	275	0	2
BFA	0	3	0	512	0
3BB	0	0	5	0	280

Table 4.34. *Confusion Matrix for 40x40 Images of Z-axis of the vibration data classification using SGD + Momentum + Nesterov*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	302	0	1	0	1
BFE	0	302	0	9	0
5BB	0	0	275	2	8
BFA	0	0	0	515	0
3BB	0	0	1	0	284

Table 4.35. *Confusion Matrix for 40x40 Images of Z-axis of the vibration data classificationusing Adam*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	243	0	1	0	0
BFE	0	308	0	3	0
5BB	0	0	229	1	55
BFA	0	0	0	515	0
3BB	0	0	0	0	285

Table 4.36. *Confusion Matrix for 40x40 Images of Z-axis of the vibration data classification using Nadam*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	304	0	0	0	0
BFE	0	206	0	105	0
5BB	0	0	258	13	14
BFA	0	0	0	515	0
3BB	1	0	0	0	284

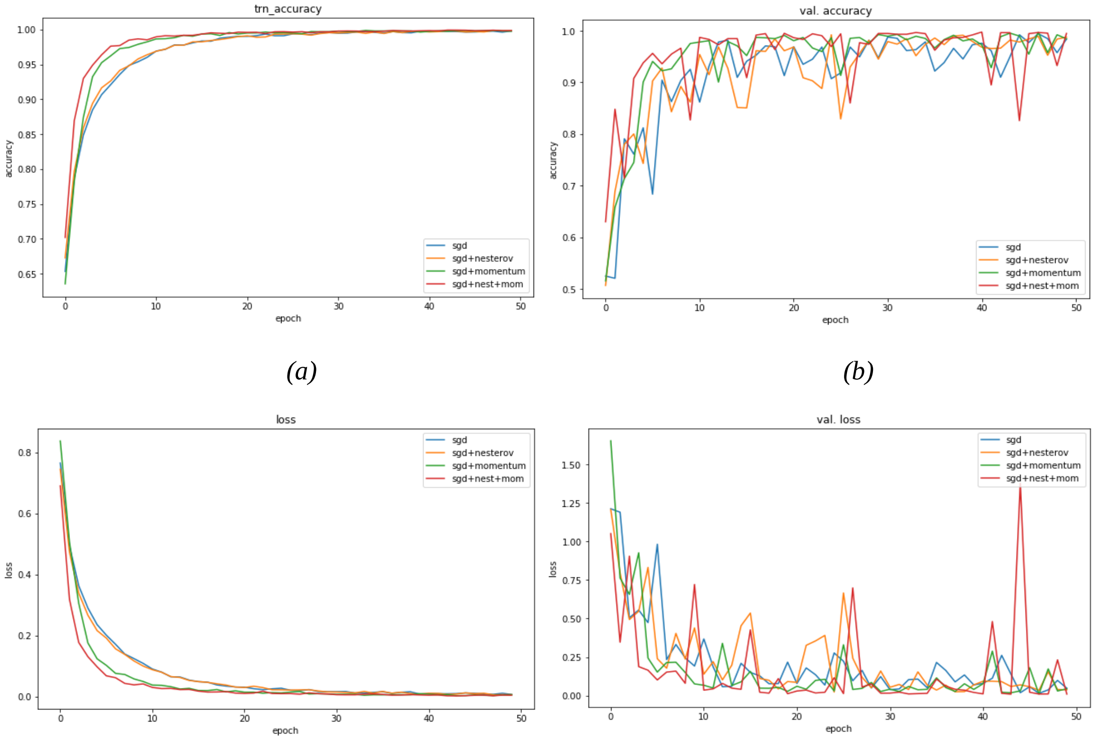
4.2.3.2. Classification results for the size of 50x50 images from the z-axis of the vibration dataset

Just change the size of the image to 50x50 from the z-axis vibration data, classification of the test data results are increased as seen in Table 4.37.

Table 4.37. *Test Accuracy for 50x50 Images of z-axis of the vibration data classification after 50 epochs using different Optimizer*

Optimization Technique	Test_accuracy
SGD	0.98
SGD + Nesterov	0.98
SGD + Momentum	0.99
SGD + Nesterov +Momentum	0.99

By looking Figure 4.24 (a) and (b), although validation accuracy with various optimizers contain fluctuations, SGD with momentum in validation accuracy graph is close to training one. This says that the model fits well. In addition, this situation is also true for the loss charts which gives an information about the learning rate.



(a) *(b)*

(c) *(d)*

Figure 4.24. *(a) Train accuracy (b) Validation accuracy (c) Train loss (d) Validation loss for 50x50 Images from Z-axis of the vibration data classification after 50 epochs using different optimizers*

To look more closely, according to Table 4.41, SGD with momentum and Nesterov gives the best classification result, only with five errors. On the other hand, it is understood from the Table 4.40, 5BB are supposed to 3BB.

Table 4.38. *Confusion Matrix for 50x50 Images of Z-axis of the vibration data classification using SGD*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	174	0	0	0	1
BFE	0	184	0	15	0
5BB	0	0	222	0	1
BFA	0	0	0	292	0
3BB	0	0	1	0	198

Table 4.39. *Confusion Matrix for 50x50 Images of Z-axis of the vibration data classification using SGD + Nesterov*

Predicted

	Healthy	BFE	5BB	BFA	3BB
Healthy	174	0	0	0	1
BFE	0	195	0	4	0
5BB	0	0	219	2	2
BFA	0	0	0	292	0
3BB	0	0	5	0	194

Table 4.40. *Confusion Matrix for 50x50 Images of Z-axis of the vibration data classification using SGD + Momentum*

Predicted

	Healthy	BFE	5BB	BFA	3BB
Healthy	173	0	1	0	1
BFE	0	198	0	1	0
5BB	1	3	210	0	9
BFA	0	1	0	291	0
3BB	0	1	0	0	198

Table 4.41. *Confusion Matrix for 50x50 Images of Z-axis of the vibration data classification using SGD + Momentum and Nesterov*

Predicted

	Healthy	BFE	5BB	BFA	3BB
Healthy	175	0	0	0	0
BFE	0	194	0	5	0
5BB	0	0	223	0	0
BFA	0	0	0	292	0
3BB	0	0	1	0	198

It can be said in total for this section, the images of size 50x50 from z-axis vibration data give more accurate results than the images of the size 40x40.

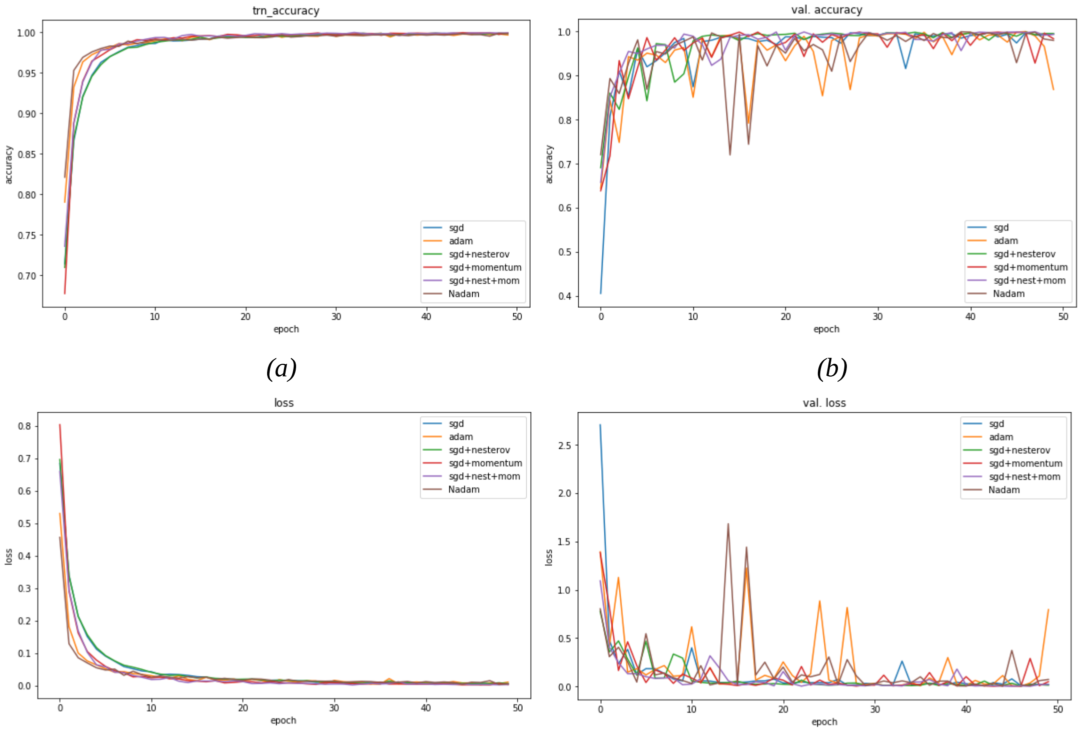
4.2.4. Classification results for color images from of the vibration dataset

4.2.2.1. Classification results for the size of 40x40 color images of the vibration dataset

The results were expected to be more accurate with the color images because of the combination of information of all axes. As seen from the Table 4.42, the result with SGD gives such a high result for the first time. Except for the technique Adam, the other results are also not bad. It can be said by looking at Figure 4.25 (a)-(d) that SGD conforms to the CNN.

Table 4.42. *Test Accuracy for 40x40 Color Images of the vibration data classification after 50 epochs using different Optimizer*

Optimization Technique	Test_accuracy
SGD	0.99
SGD + Nesterov	0.99
SGD + Momentum	0.98
SGD + Nesterov +Momentum	0.99
Adam	0.87
Nadam	0.98



(a) *(b)*

(c) *(d)*

Figure 4.25. *(a) Train accuracy (b) Validation accuracy (c) Train loss (d) Validation loss for 40x40 Color Images of the vibration data classification after 50 epochs regardless of various optimizers*

The network with SGD gives a good result as shown in Table 4.43.

Table 4.43. *Confusion Matrix for 40x40 Color Images of the vibration data classification using SGD*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	290	0	1	0	0
BFE	0	316	0	1	0
5BB	0	0	308	0	1
BFA	0	7	0	497	0
3BB	0	0	0	0	279

Table 4.44. *Confusion Matrix for 40x40 Color Images of the vibration data classification using SGD + Nesterov*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	291	0	0	0	0
BFE	0	312	0	5	0
5BB	1	0	307	1	0
BFA	0	1	0	503	0
3BB	0	0	1	0	278

Table 4.45. *Confusion Matrix for 40x40 Color Images of the vibration data classification using SGD + Momentum*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	291	0	0	0	0
BFE	0	317	0	0	0
5BB	0	0	309	0	0
BFA	0	29	0	475	0
3BB	0	0	0	0	279

Table 4.46. *Confusion Matrix for 40x40 Color Images of the vibration data classification using SGD + Momentum and Nesterov*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	290	0	1	0	0
BFE	0	315	0	2	0
5BB	0	0	306	0	3
BFA	0	2	0	502	0
3BB	0	0	1	0	278

Table 4.47. *Confusion Matrix for 40x40 Color Images of the vibration data classification using Adam*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	291	0	0	0	0
BFE	0	317	0	0	0
5BB	2	2	292	0	13
BFA	0	191	0	312	0
3BB	0	0	0	0	279

Table 4.48. *Confusion Matrix for 40x40 Color Images of the vibration data classification using Nadam*

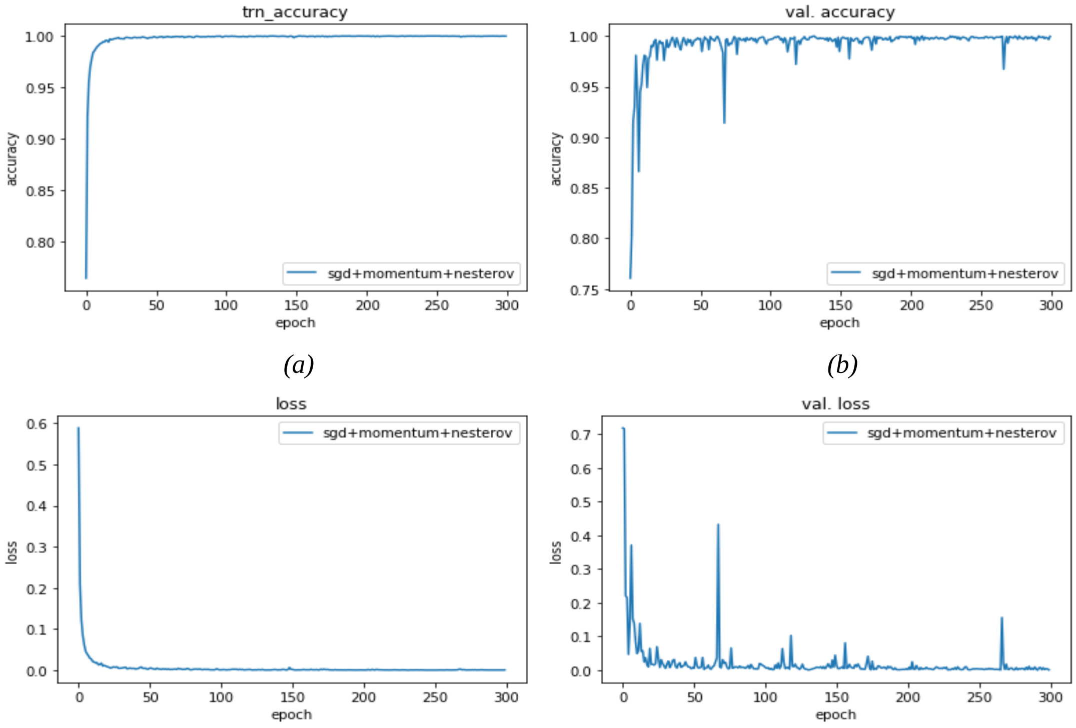
	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	283	0	4	0	4
BFE	0	296	18	3	0
5BB	0	0	305	0	4
BFA	0	7	1	503	0
3BB	0	0	0	0	279

Since better results are expected for the color images, the model is once run with 300 epochs and with only for SGD with momentum and Nesterov which gives the best classification result in total. With the change of this parameter, test accuracy showed 0.999. Thus, Table 4.49 demonstrate the best result up to now, with only one classification error.

Table 4.49. *Confusion Matrix for 40x40 Color Images of the vibration data classification using SGD + Momentum and Nesterov*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	293	0	0	0	0
BFE	0	328	0	0	0
5BB	0	0	280	0	0
BFA	0	0	0	504	0
3BB	0	0	1	0	294

After trying the experiment with 300 epoch seen in the following Figure 4.26, CNN could be learned more stable compared to 50 epochs and so gives the better result.



(a) *(b)*

(c) *(d)*

Figure 4.26. *(a) Train accuracy (b) Validation accuracy (c) Train loss (d) Validation loss for 40x40 Color Images of the vibration data classification after 300 epochs using different optimizers*

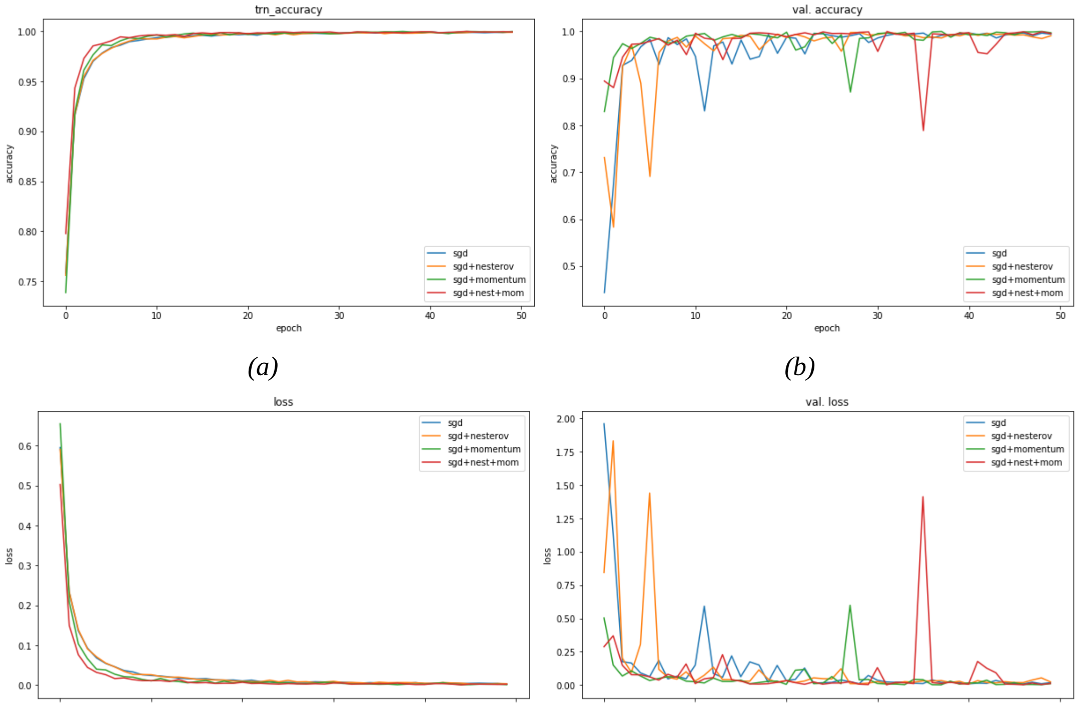
4.2.2.2. Classification results for the size of 50x50 color images of the vibration dataset

Classification using SGD with momentum for the size of 50x50 color images of the vibration data gives more accurate results compared to the size of 40x40 color images from the Table 4.50.

Table 4.50. *Test Accuracy for 50x50 Color Images of the vibration data classification after 50 epochs using different Optimizer*

Optimization Technique	Test_accuracy
SGD	0.99
SGD + Nesterov	0.99
SGD + Momentum	0.99
SGD + Nesterov +Momentum	0.99

Figure 4.24 (a) to (d) indicate that except usage of SGD + Nesterov + Momentum, the CNN fits well in other technique.



(a) *(b)*

(c) *(d)*

Figure 4.27. *(a) Train accuracy (b) Validation accuracy (c) Train loss (d) Validation loss for 50x50 Color Images of the vibration data classification after 50 epochs using different optimizers*

Table 4.51 to 4.54 prove above test accuracy from the point of view classification errors. The model with SGD + Nesterov overcomes the others by having only two errors.

Table 4.51. *Confusion Matrix for 50x50 Color Images of the vibration data classification using SGD*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	185	0	0	0	0
BFE	0	200	0	1	0
5BB	0	2	185	0	2
BFA	0	1	0	326	0
3BB	0	0	0	0	186

Table 4.52. *Confusion Matrix for 50x50 Color Images of the vibration data classification using SGD + Nesterov*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	185	0	0	0	0
BFE	0	201	0	0	0
5BB	0	0	187	0	2
BFA	0	8	0	319	0
3BB	0	0	0	0	186

Table 4.53. *Confusion Matrix for 50x50 Color Images of the vibration data classification using SGD + Momentum*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	185	0	0	0	0
BFE	0	201	0	0	0
5BB	0	0	189	0	0
BFA	0	2	0	325	0
3BB	0	0	1	0	185

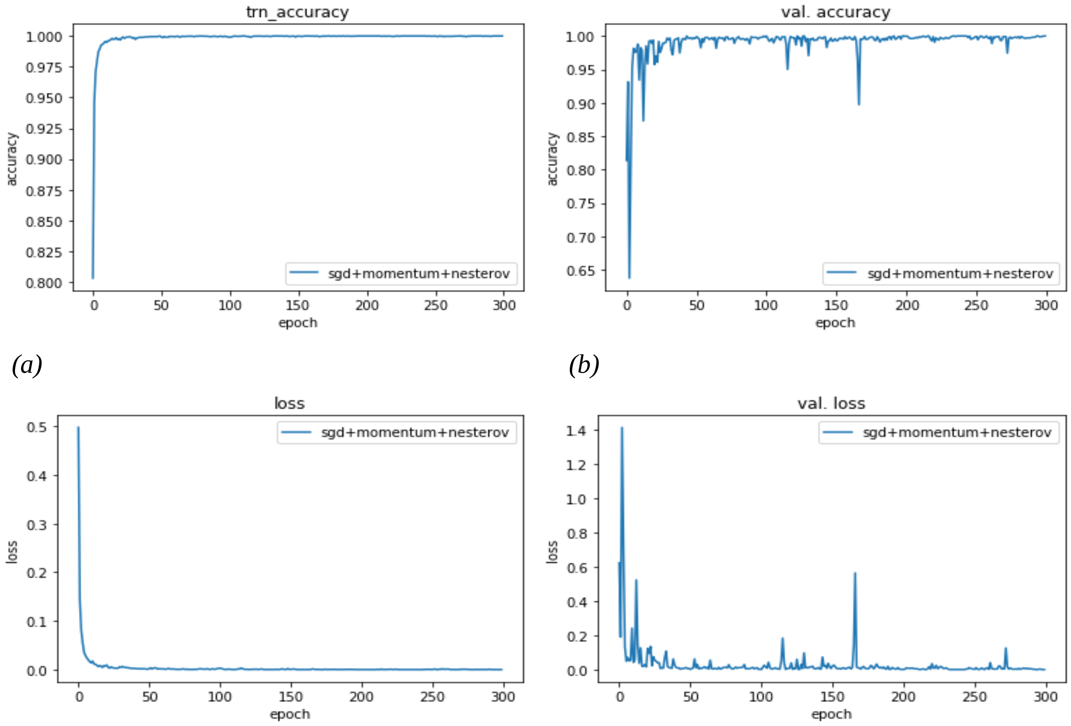
Table 4.54. *Confusion Matrix for 50x50 Color Images of the vibration data classification using SGD + Momentum and Nesterov*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	184	0	0	0	1
BFE	0	201	0	0	0
5BB	0	0	189	0	0
BFA	0	0	0	327	0
3BB	0	0	4	0	182

Like in previous part about the color images of size 40x40, classification of the color images of size 50x50 with 300 epochs is attempted. And finally, the best result is reached: test accuracy is 100%. It can see from the Table 4.55, there are no errors. Moreover, by looking the Figure 4.28, there is no overfitting. In addition to this, after such a long epoch, it can be said that the learning rate is so good.

Table 4.55. *Confusion Matrix for 50x50 Color Images of the vibration data classification after 300 epochs using SGD with Momentum and Nesterov*

	Predicted				
	Healthy	BFE	5BB	BFA	3BB
Healthy	180	0	0	0	0
BFE	0	187	0	0	0
5BB	0	0	197	0	0
BFA	0	0	0	319	0
3BB	0	0	0	0	205



(a) *(b)*

(c) *(d)*

Figure 4.28. *(a) Train accuracy (b) Validation accuracy (c) Train loss (d) Validation loss for 50x50 Color Images of the vibration data classification after 300 epochs using different optimizers*

5. CONCLUSION

Compared to other motor types used in the industry, the induction motors are more preferred because of their robust structure, easy to maintain, easy to use, and cost effectiveness. The fact that the induction motors are used extensively in the industry explain why it is desired to work on tracking the faults in many research. The early detection of faults and preventive maintenance studies are crucial so that industrial processes cannot be interrupted, and vital situations are not experienced. In addition, early detection of the problem also reduces the maintanance and production costs which is another issue that should be considered in the industrial work environment. In this thesis, vibration data are used which are collected in the laboratory environment in order to classifiy induction motor faults which which are frequently encountered in a real industrial environment. Those are synthetically created on test motors. In order to simulate the actual conditions that can be met in the industiral area the induction motors are operated under different load conditions. Then, the vibration data from each test induction motor is collected by the 3-axial accelerometer, and so information of horizontal, vertical and axial axes of the vibration data is formed.

The classification method used in this thesis is a knowledge-based method which is based on deep learning. Before moving on to the classification stage, vibration data based feature extraction technique is applied by studying the characteristics of vibration data. A feature extraction method that has been used frequently has been proposed. Thus, an image-based method is proposed to make the classification possible for a deep neural system. Transformation of one-dimensional vibration data into two-dimensional grayscale images is used by considering the pixel density differences between the vibration data of the induction motor faults. In addition to this, it is very important to determine the fundamental motor frequency of the motor in order to be able to create images. Because the speed of rotation of the motor will be affected by the types of fault under different loading conditions. So, in order to obtain accurate information about the vibration data, the autocorrelation function is used which gives possible fundamental frequencies of the power signal of the vibration data. Then, 2D grayscale square images are created using this information. The classification method will be formed using those images. Here in this phase, the Convolutional Neural Network is preferred to classify the vibration data. CNN, which has recently proven itself in the field of deep learning, has been very well suited for visual documents. By the knowledge of its tremendous performance in both 2D grayscale and 3D color images, 3D color images are also created using the same vibration data in this thesis. In order to use CNN, the images which are created from the vibration data are divided into smaller images, 40x40 and 50x50, and are thus trained and tested by the system. For the formation of 3D color images; horizontal, vertical and axial axes are assigned to red, green and blue respectively.

After both 2D grayscale and 3D color images are created, the induction motor faults differences are visually distinguishable. This motivated the CNN to work well to classify the faulty motors.

During the experiments, using the different parameters in CNN, the classification accuracy of images has been determined, inferences have been made about which engines are more similar to each other and observations have been done about which parameters fit better than others. In this way, a total of eight categories which are compared to each other can be summarized as:

1. The size of 40x40 2D grayscale images of vibration data from the x-axis
2. The size of 50x50 2D grayscale images of vibration data from the x-axis
3. The size of 40x40 2D grayscale images of vibration data from the y-axis
4. The size of 50x50 2D grayscale images of vibration data from the y-axis
5. The size of 40x40 2D grayscale images of vibration data from the z-axis
6. The size of 50x50 2D grayscale images of vibration data from the z-axis
7. The size of 40x40 3D color images of vibration data from the combination of x-y-z-axes
8. The size of 50x50 3D color images of vibration data from the combination of x-y-z-axes

Considering the experiments described in Chapter 4, these conclusions can be drawn:

1. The size of 40x40 2D grayscale images of vibration data from the x-axis: In this section, as well as adaptive learning rate methods, stochastic gradient methods gave good results with Nesterov and momentum. Because, according to the input data, these optimizers are not trapped in the saddle points like SGD. The model produces the most unsuccessful result from the point of view distinguishing BFE. Thus, it can be deducted that motor with

bearing fault misalignment (BFE) can not be deduced using vibration data extracted from the horizontal axis.

2. The size of 50x50 2D grayscale images of vibration data from the x-axis: Changing the size of the model to 50x50 increased the accuracy with SGD, but the issue of mixing the BFE with BFA still exists. This is because of the BFE has similar x-axis vibration data information to BFA. In addition to this, it can be said that SGD learns better with more training parameters from 50x50 images, this is a result confirming the generally known CNN operating principle.

3. The size of 40x40 2D grayscale images of vibration data from the y-axis: If looking at the results of all the optimization techniques, except SGD with momentum and Nesterov, the model confuses the BFE with the BFA on the y-axis. Because BFE does not give enough information from the vertical axis. BFE shows similar characteristic to BFA, because they are similar fault types, bearing faults. On the other hand, interestingly, with the use of SGD with momentum and Nesterov, there is no error on BFE. It means that the model can be learned with the correct parameter. The SGD with momentum and Nesterov was not stuck in saddle points.

4. The size of 50x50 2D grayscale images of vibration data from the y-axis: The situation of BFE, which does not have enough information with x and y-axis, continues in the same way with all optimizers.

5. The size of 40x40 2D grayscale images of vibration data from the z-axis: With the optimizer SGD with momentum and Nesterov, 5BB is confused with the 3BB and BFE. Because 5BB from z-axis has less information 5BB from the x-axis and y-axis. On the other hand, it can be deducted from this part that BFE contains more distinguishing information on the z-axis.

6. The size of 50x50 2D grayscale images of vibration data from the z-axis: It can be said that the test accuracy has increased but the system is still experiencing confusion in the same places.

7. The size of 40x40 3D color images of vibration data from the combination of x-y-z-axes: When it comes to color pictures, the confusion suddenly disappears, as can be understood from the confusion matrix. Because, the model (CNN) is built based on information from all axes, and the resulting system contains more information in order to classify fault types. This emphasizes the importance of combining information and introducing it to the network. When the epoch value is increased to 300, CNN works very

well in the classification with the only one error. So, it can be said that CNN learns very well with enough epoch. If the system can’t learn, it means that it needs more training.

8. The size of 50x50 3D color images of vibration data from the combination of x-y-z-axes: In this part, there are more training parameters and CNN gives a more accurate test result with 50 epochs compared to other. Because, it is well-known fact that even though the best parameters are selected, the number of training parameter always ensures better results in deep learning techniques. In other words, using more parameters can give much more accurate results than choosing the right optimization parameters. When the epoch value is increased to 300, there is no error.

To summarize, the vibration data of the induction motor with bearing fault misalignment (BFE) does not contain distinguishing information from x and y-axes. So, BFE has more oscillation in the z-axis. In addition to this, BFE has similar information to the vibration data of the induction motor with bearing fault ball defect (BFA). Moreover, the vibration data of the induction motor with 5-broken bars have similar information to 3-broken bars, and both of them have more information on radial axes. Furthermore, CNN, which is with SGD plus momentum and Nesterov, works well because the vibration data from induction motor faults probably not too complex and extensive data. If it were so, the Adam would be expected to work as well as SGD with momentum and Nesterov.

Even if the vibration data of the induction motor faults may sometimes look similar, CNN is a promising tool for the classification.

In conclusion, Convolutional Neural Network model proposed in this thesis in order to classify induction motor faults is produced surprising results. Furthermore, it has been observed that CNN with enough data and necessary amount of training cycles, it is possible to obtain 100% success.

This thesis is expected to be an important reference both in the industry and in future research.

REFERENCES

Akcay, H., and Germen, E., (2015) “Subspace-Based Identification of Acoustic Noise Spectra in Induction Motors”. IEEE Transactions on Energy Conversion 30, sy 1:32-40. https://doi.org/10.1109/TEC.2014.2334633.

Abdel-Hamid, O., Mohamed, A.-R., Jiang, H., and Penn, G. (2012), ‘Applying convolutional neural networks concepts to hybrid nn-hmm model for speech recognition,’ in 2012 IEEE international conference on Acoustics, speech and signal processing (ICASSP). IEEE, pp. 4277–4280.

APEC, (2008) Electric Motors—Alignment of Standards and Best Practice Programmes within APEC, Final Report

Albelwi, S., Mahmood, A., (2017) ‘A Framework for Designing the Architectures of Deep Convolutional Neural Networks’, Computer Science and Engineering Department, University of Bridgeport, 5

Altenberger, F., and Claus L., (2018) “A Non-Technical Survey on Deep Convolutional Neural Network Architectures”. ArXiv:1803.02129 [Cs], http://arxiv.org/abs/1803.02129.

Almeida, A., Fong, J., Falkner, H., Bertoldi, P., (2017), “Policy options to promote energy efficient electric motors and drives in the EU”, Renewable and Sustainable Energy Reviews, Elsevier, Vol.74, 1275-1286, t.y.

Bayot, R., Khristopher O. “A Survey on Object Classification Using Convolutional Neural Networks”

Benbouzid, M. (1999), ‘Bibliography on induction motors faults detection and diagnosis,’ IEEE Transactions On Energy Conversion, 14(4), 1065-1074

Bengio, I.G.Y., Courville, A.,(2016) Deep learning, book in preparation for MIT Press [Online]

Bindu, S. and Tomas, V. V., (2014)‘Diagnoses of internal faults of three phase squirrel cage induction motors—A review,’ in Proc. ICAECT, pp. 48–54.

Brownlee, J., (2016), ‘Supervised and Unsupervised Machine Learning Algorithms’, (URL: https://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms/)

Boureau, Y.L., Ponce, J., LeCun, Y., (2010), A theoretical analysis of feature pooling in visual recognition, in: Proceedings of the 27th International Conference on Machine Learning (ICML-10), pp. 111–118.

Ciresan, D. C., Meier, U., Masci, J., Gambardella, L. M. and Schmidhuber, J., (2011), 'High-performance neural networks for visual object classification,' Istituto Dalle Molle di Studi sull'Intelligenza Artificiale (IDSIA), Tech. Rep. IDSIA-01- 11

Chollet, F., (2018), Deep Learning with Python. Shelter Island, New York: Manning Publications Co

Collobert , R. and Weston, J. ,(2008)'A unified architecture for natural language processing: Deep neural networks with multitask learning,' in Proceedings of the 25th international conference on Machine learning. ACM, pp. 160–167.

Collobert,R., Weston, J., Bottou, L., Karlen, M., Kavukcuglu, K., Kuksa., P., (2011), Natural Language Processing (Almost) from Scratch. Journal of Machine Learning Research 12:2493–2537.

Deng, L., (2014), "A Tutorial Survey of Architectures, Algorithms, and Applications for Deep Learning". APSIPA Transactions on Signal and Information Processing 3, https://doi.org/10.1017/atsip.2013.9.

Kingma, D. P., Ba, J., (2015), 'Adam: A Method for Stochastic Optimization', in ICLR

Dozat, T. (2016). Incorporating Nesterov Momentum into Adam. ICLR Workshop, (1), 2013–2016

Eschmann, P., Hasbargen, L., Weigand, K., (1958) Ball and roller bearings: their theory, design and application. K. G. Heyden, London

Finley, W. R., Hodowanec, M. M. and Holter, W. G. (1999), 'An analytical approach to solving motor vibration problems,' Proceedings of the Petroleum and Chemical Industry Conference, 1999. Industry Applications Society 46th Annual, IEEE, 217-232.

Fukushima,K.. Miyake, S., (1982) Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition, in: Competition and cooperation in neural nets, pp. 267–285

Funahashi, K.-I. and Nakamura, Y. (1993), 'Approximation of dynamical systems by continuous time recurrent neural networks,' Neural networks, vol. 6, no. 6, pp. 801–806

Gao, Zhiwei et al. (2015), 'A Survey of Fault Diagnosis and Fault-Tolerant Techniques—Part I: Fault Diagnosis With Model-Based and Signal-Based Approaches.' IEEE Transactions on Industrial Electronics 62,3757-3767.

Germen, E., Başaran, M., and Fidan, M., (2014) "Sound Based Induction Motor Fault Diagnosis Using Kohonen Self-Organizing Map". Mechanical Systems and Signal Processing 46, sy ,45-58. https://doi.org/10.1016/j.ymssp.2013.12.002.

Giantomassi, A., (2015), ‘Electric motor fault detection and diagnosis by kernel density estimation and Kullback–Leibler divergence based on stator current measurements,’ IEEE Trans. Ind. Electron., vol. 62, no. 3, pp. 1770– 1780

Goncalves, M. J. M., Creppe, R.C., Marques, E.G., and Cruz, S.M.A., (2015),“Diagnosis of Bearing Faults in Induction Motors by Vibration Signals - Comparison of Multiple Signal Processing Approaches”, 488-93 IEEE, https://doi.org/10.1109/ISIE.2015.7281516.

Goodfellow,I., Bengio,Y. and Courville,A., (2016), online version (URL: http://www.deeplearningbook.org/)

Gu, J., Zhenhua W., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., Liu, T. vd. “Recent Advances in Convolutional Neural Networks”. Pattern Recognition 77 (Mayıs 2018): 354-77. https://doi.org/10.1016/j.patcog.2017.10.013.

Guoa, Yanming, Yu Liu, Ard Oerlemans, Songyang Lao, Song Wu, ve Michael S. Lew. “Deep Learning for Visual Understanding: A Review”. Neurocomputing 187 (Nisan 2016): 27-48. https://doi.org/10.1016/j.neucom.2015.09.116.

Günal, S., Ece, D. G. and Gerek, Ö. N. (2009), ‘Induction motor fault diagnosis via current analysis on time domain,’ Proceedings of the Signal Processing and Communications Applications Conference, 2009. SIU 2009. IEEE 17th, IEEE, 488-491.8

Harley, A. W., Ufkes, A., (2015), Derpanis, K. G., ‘Evaluation of Deep Convolutional Nets for Document Image Classification and Retrieval,’ in ICDAR

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. (2015). ‘Deep Residual Learning for Image Recognition.’ ArXiv:1512.03385 [Cs], December. http://arxiv.org/abs/1512.03385.

Henao, Humberto, Gerard-Andre Capolino, Manes Fernandez-Cabanas, Fiorenzo Filippetti, Claudio Bruzzese, Elias Strangas, Remus Pusca, Jorge Estima, Martin Riera-Guasp, ve Shahin Hedayati-Kia. “Trends in Fault Diagnosis for Electrical Machines: A Review of Diagnostic Techniques”. IEEE Industrial Electronics Magazine 8, sy 2 (Haziran 2014): 31-42. https://doi.org/10.1109/MIE.2013.2287651.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., & Hochreiter, S. (2017). GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. In Advances in Neural Information Processing Systems 30 (NIPS 2017).

Hinton, G. E., Osindero, S. and Teh, Y.-W., (2006), ‘A fast learning algorithm for deep belief nets,’ Neural computation, vol. 18, no. 7, pp. 1527– 1554.

Hinton,G.E., Salakhutdinov, R.R., “ Reducing the dimensionality of data with neural networks”, Science 313 (5786) (2006) 504–507.

Hinton,G., Deng,L., Yu, D., Dahl, G. E. , Mohamed, A., Jaitly, N. , Senior, A., Vanhoucke, V., Nguyen, P., Sainath T. N. ,et al., (2012), 'Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups,' IEEE Signal Processing Magazine, vol. 29, no. 6, pp. 82–97.

Hou,N., Dong, H., Wang,Z., Ren, W., Alsaadi, F.E., (2016), Non-fragile state estimation for discrete Markovian jumping neural networks, Neurocomputing 179, 238–245.

Hubel,D. H., Wiesel, T. N, (1968), Receptive fields and functional architecture of monkey striate cortex, The Journal of physiology ,215–243.

Huang, G., Liu, Z., Maaten, L., Weinberger, K.,'Densely Connected Convolutional Networks', CVPR 2017, last revised 28 Jan 2018 (v5) https://arxiv.org/abs/1608.06993v5

Janocha,K. and Czarnecki,W. M.,(2017) 'On loss functions for deep neural networks in classification,' arXiv preprint arXiv:1702.05659.

Jarrett, K. , Kavukcuoglu, K., Lecun Y. et al., (2009), 'What is the best multi-stage architecture for object recognition?' in 2009 IEEE 12th International Conference on Computer Vision. IEEE, pp. 2146– 2153.

Kalchbrenner, N., Grefenstette, E., Blunsom.,P., (2014). A Convolutional Neural Network for Modelling Sentences. In Proceedings of ACL

Karmakar, Subrata, Surajit Chattopadhyay, Madhuchhanda Mitra, ve Samarjit Sengupta, (2016), "Induction Motor and Faults", Springer Singapore. https://doi.org/10.1007/978-981-10-0624-1_2.

Kaya D, Yagmur EA, Yigit KS, Kilic FC, Eren AS, Celik C. Energy efficiency in pumps. Energy Conversion and Management 2008;49:1662–73.

Kim, Yoon. "Convolutional Neural Networks for Sentence Classification". ArXiv:1408.5882 [Cs], 25 Ağustos 2014. http://arxiv.org/abs/1408.5882.

Krause, PC (1986) ,"Analysis of electric machinery", Mc-Graw Hill, New York"

Krizhevsky, Alex, Ilya Sutskever, ve Geoffrey E. Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". Communications of the ACM 60, sy 6 (24 Mayıs 2017): 84-90. https://doi.org/10.1145/3065386.

Kowalski, C. T. and Kowalska,T. O,(2003), 'Neural network application for induction motor faults diagnosis,' Math. Comput. Simul., vol. 63, nos. 3–5, pp. 435–448.

Kurzweil,R. , (2013)" How to create a mind: The secret of human thought revealed', Penguin.

LeCun, B. B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D. (1990), 'Handwritten digit recognition with a backpropagation network,' in Advances in neural information processing systems. Citeseer.

Lecun, Y. (1992). A theoretical framework for back-propagation. In P. Mehra, & B. Wah (Eds.), Artificial neural networks: Concepts and theory Los Alamitos, CA: IEEE Computer Society Press.

LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., 'Gradient-based learning applied to document recognition' Proceedings of the IEEE, v. 86, pp. 2278 2324

LeCun, Y., Kavukcuoglu, K., and Farabet, C. (2010), "Convolutional Networks and Applications in Vision", 253-56. IEEE, https://doi.org/10.1109/ISCAS.2010.5537907.

LeCun, Y., Bengio, B., and Hinton, G., (2015), "Deep Learning". *Nature* 521, sy 7553 : 436-44. https://doi.org/10.1038/nature14539.

Leung,M. K. , Xiong, H. Y., Lee, L. J. and Frey, B. J.'Deep learning of the tissue-regulated splicing code,' Bioinformatics, vol. 30, no. 12, pp. i121–i129, 2014.

Li, Fei-Fei, Justin Johnson, ve Serena Yeung. "Lecture 9: CNN Architectures", t.y., 101.

Li, X., Wu, Q, and Nandi, S., (2007) 'Performance analysis of a three-phase induction machine with inclined static eccentricity,' IEEE Trans. Ind. Appl., vol. 43, no. 2, pp. 531–541, Mar./Apr. 2007.

Lin,M. , Chen,Q. ,Yan, S., (2014), Network in network, in: Proceedings of the International Conference on Learning Representations (ICLR).

Liu, Weibo, Zidong Wang, Xiaohui Liu, Nianyin Zeng, Yurong Liu, ve Fuad E. Alsaadi. (2017) "A Survey of Deep Neural Network Architectures and Their Applications". *Neurocomputing* 234 (Nisan 2017): 11-26. https://doi.org/10.1016/j.neucom.2016.12.038.

Maas, A. L., Hannun, A. Y., (2013), Rectifier nonlinearities improve neural network acoustic models, in: Proceedings of the International Conference on Machine Learning (ICML), Vol. 30,.

Maturana, D. and Scherer, S. (2015), 'VoxNet: A 3D Convolutional Neural Network for Real-Time Object Recognition', IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)

Mehrjou, M. R. and Mariun, N. (2011), 'Rotor fault condition monitoring techniques for squirrel-cage induction machine—A review,' Mech. Syst. Signal Process., vol. 25, no. 8, pp. 2827–2848.

Nair,V., Hinton, G. E., (2010), Rectified linear units improve restricted boltzmann machines, in: Proceedings of the International Conference on Machine Learning (ICML), pp. 807–814.

Nesterov, Y. (1983). A method of solving a convex programming problem with convergencerate O(1/k2). Soviet Mathematics Doklady, 27, 372–376

Nielsen, M. A., (2015) 'Neural networks and deep learning'

Ince, T., Kiranyaz, S., Eren, L., Askar, M., and Gabbouj, M., (2016), "Real-Time Motor Fault Detection by 1-D Convolutional Neural Networks". *IEEE Transactions on Industrial Electronics* 63, sy 11,7067-75. https://doi.org/10.1109/TIE.2016.2582729.

Isermann, R. (2005), "Model-Based Fault-Detection and Diagnosis – Status and Applications". *Annual Reviews in Control* 29, sy 1 (Ocak 2005): 71-85. https://doi.org/10.1016/j.arcontrol.2004.12.002.

Ojaghi, M. and Mohammadi, M. 'Unified Modeling Technique for Axially Uniform and Nonuniform Eccentricity Faults in Three-Phase Squirrel Cage Induction Motors' , IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, VOL. 65, NO. 7, JULY 2018

Peng, M., Wang, C., Chen, T., Liu, G., (2016), NIRFaceNet: A Convolutional Neural Network for Near-Infrared Face Identification. Information, 7, 61.

Platek,S., Keenan, J., and Shackelford,T.," Evolutionary cognitive neuroscience", Mit Press, 2007.

Polamuri, S. (2017) Difference Between Softmax Function And Sigmoid Function, Dataspirant Blog, 2017, (URL:http://dataaspirant.com/2017/03/07/difference-between-function-and- sigmoid-function/) (last achieved: 23.07.2018)

Polyak, B. T. (1964). Some methods of speeding up the convergence of iteration methods. USSR Computational Mathematics and Mathematical Physics, 4(5), 1–17. 292

Pons-Llinares, J., Antonino-Daviu, J., Roger-Folch, J., Morinigo-Sotelo, D. and Duque- Perez, O., (2010), 'Eccentricity Diagnosis in Inverter Fed Induction Motors via the Analytic Wavelet Transform of Transient Currents', XIX International Conference on Electrical Machines (ICEM), pp.1-6.

Razavian, A. S., Azizpour, H., Sullivan, J. , Carlsson, S., (2014),"CNN Features off-the-shelf: an Astounding Baseline for Recognition", CVAP, KTH (Royal Institute of Technology) Stockholm, Sweden, arXiv:1403.6382v3 [cs.CV]

Ren, S., He,K., Girshick, R. and Sun, J. (2015), 'Faster r-cnn: Towards realtime object detection with region proposal networks,' in Advances in neural information processing systems, pp. 91–99.

Ruder, S.. (2016), “An Overview of Gradient Descent Optimization Algorithms”. *ArXiv:1609.04747 [Cs]*, 15 Eylül 2016. http://arxiv.org/abs/1609.04747.

Saleh, A., Ausif, M. (2017), “A Framework for Designing the Architectures of Deep Convolutional Neural Networks”. *Entropy* 19, sy 6 (24 Mayıs 2017): 242. https://doi.org/10.3390/e19060242.

Schoen ,R.R., Habetler, T.G., Kamran F., Bartheld, R.G. (1995) Motor bearing damage detection using stator current monitoring. IEEE Trans Ind Appl 31(6):1274–1279

Saidur, R. (2009), ‘A review on electrical motors energy use and energy savings’, Renewable and Sustainable Energy Reviews 14 , Elsevier, 877–898,

Say, M.G. (2002) The performance and design of alternating current machines. M/S Pitman, London. ISBN 81-239-1027-4

Schmidhuber, J., (2015) “Deep Learning in Neural Networks: An Overview”. *Neural Networks* 61 (Ocak 2015): 85-117. https://doi.org/10.1016/j.neunet.2014.09.003.

Seif, G. “Data Science and Machine Learning Interview Questions”

Sergey, I., Christian, S., (2015), “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”, 4-5”. *ArXiv:1502.03167 [Cs]*, 10 Şubat 2015. http://arxiv.org/abs/1502.03167.

Sermanet, P., Chintala, S., and LeCun, Y. (2012), ‘Convolutional neural networks applied to house numbers digit classification,’ in Pattern Recognition (ICPR), 21st International Conference on. IEEE, 2012, pp. 3288–3291.

Shen,Y., He, X., Gao, J., Deng, L., Mesnil. G., (2014), Learning Semantic Representations Using Convolutional Neural Networks for Web Search

Simard, P.Y., D. Steinkraus, ve J.C. Platt. (2003), “Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis”, 1:958-63. IEEE Comput. Soc, https://doi.org/10.1109/ICDAR.2003.1227801.

Simonyan, K., Zisserman, A., (2015), ‘Very Deep Convolutional Networks for Large-Scale Image Recognition’, Visual Geometry Group, Department of Engineering Science, University of Oxford

Sonje, D. M., Munje ,R. K., (2011), ‘Rotor Cage Fault Detection in Induction Motors by Motor Current Signature Analysis’, International Conference in Computational Intelligence (ICCIA),

Sun, Wenjun, Siyu Shao, Rui Zhao, Ruqiang Yan, Xingwu Zhang, ve Xuefeng Chen. (2016), “A Sparse Auto-Encoder-Based Deep Neural Network Approach for Induction Motor Faults Classification”. *Measurement* 89 (Temmuz 2016): 171-78. https://doi.org/10.1016/j.measurement.2016.04.007

Sutskever, I., Martens, J., Dahl, G., and Hinton, G. (2013). On the importance ofinitialization and momentum in deep learning. In ICML, 296,401,408

Szegedy, C., Liu, W., Hill, C., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A., (2014), 'Going deeper with convolutions', arXiv:1409.4842v1 [cs.CV],

Vedaldi, A., ve Karel L., (2014) "MatConvNet - Convolutional Neural Networks for MATLAB". *ArXiv:1412.4564 [Cs]*, 15 Aralık 2014. http://arxiv.org/abs/1412.4564.

Vincent,P. , Larochelle, H., Bengio,Y. , and Manzagol, P.-A., (2008),'Extracting and composing robust features with denoising autoencoders,' in Proceedings of the 25th international conference on Machine learning. ACM, pp. 1096–1103

Theano Development Team, Deep Learning Tutorial Release 0.1, LISA lab, University of Montreal, 51 – 63

Tygert, M., Joan B., Chintala, S., LeCun, Y., Piantino, S. and Szlam. A., (2016), "A Mathematical Motivation for Complex-Valued Convolutional Networks". *Neural Computation* 28, sy 5 (Mayıs 2016): 815-25. https://doi.org/10.1162/NECO_a_00824.

Trivedi, Shubhendu, ve Risi Kondor. "Lecture 6 Optimization for Deep Neural Networks - CMSC 35246: Deep Learning", t.y., 179.

Qian, N. (1999). On the momentum term in gradient descent learning algorithms. Neural Networks : The Official Journal of the International Neural Network Society, 12(1), 145–151.

Wang, L., 'Induction Motor Bearing Fault Detection Using a Sensorless Approach', Ph.D. Dissertation, Graduate Studies of Texas A&M University"

Yang, F., Dong,H., Wang, Z. , Ren, W., Alsaadi,F.E.,(2016), A new approach to non-fragile state estimation for continuous neural networks with time-delays, Neurocomputing 197 (2016) 205–211.

Yu, Y., Wenwu W., and Han, P., (2016), "Localization Based Stereo Speech Source Separation Using Probabilistic Time-Frequency Masking and Deep Neural Networks". *EURASIP Journal on Audio, Speech, and Music Processing* 2016, sy 1 (Aralık 2016). https://doi.org/10.1186/s13636-016-0085-x.

Zeiler, Matthew D., and Fergus, R., (2013), "Visualizing and Understanding Convolutional Networks". *ArXiv:1311.2901 [Cs]*, 12 Kasım 2013. http://arxiv.org/abs/1311.2901.

Zhang, P. J., Du, Y., Habetler, T. G. and Lu, B., (2011), 'A survey of condition monitoring and protection methods for medium-voltage induction motors,' IEEE Transactions on Industry Applications, vol. 47, pp. 3446, Jan-Feb 2011

Zhao,R., Yan, R., Chen, Z., Mao, K., Wang, P., and Gao, R.X., (2015), “Deep Learning and Its Applications to Machine Health Monitoring: A Survey “, JOURNAL OF LATEX CLASS FILES, VOL. 14, NO. 8, AUGUST 2015

http-1:https://slideplayer.com/slide/5784318/ (last achieved: 25.07.2018)
http-2:http://cs231n.github.io/convolutional-networks/#pool (last achieved: 03.08.2018)
http-3:https://www.elprocus.com/induction-motor-types-advantages/ (last achieved: 25.07.2018)
http-4:https://www.electricaleasy.com/2014/02/working-principle-and-types-of.html (last achieved: 25.07.2018)
http-5:https://www.kaggle.com/dansbecker/rectified-linear-units-relu-in-deep-learning (last achieved: 25.07.2018)
http-6:https://ww2.mathworks.cn/en/solutions/deep-learning/convolutional-neural-network.html (last achieved: 25.07.2018)
http-7:https://github.com/shaoanlu/dogs-vs-catsredux/blob/master/opt_experiment.ipynb (last achieved: 25.07.2018)
http-8:https://stackoverflow.com/questions/47312219/what-does-non-trainable-params-mean (last achieved: 25.07.2018)
http-9: https://keras.io/ (last achieved: 25.07.2018)
http-10:http://ufldl.stanford.edu/tutorial/supervised/OptimizationStochasticGradientDescent/ (last achieved: 03.08.2018)
http-11:https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/ (last achieved: 25.07.2018)
http-12:http://cs231n.github.io/convolutional-networks/ (last achieved: 25.07.2018)

RESUME

Name-Surname:	Zehra ŞAHİN
Language:	English (advanced), German (beginner)
Date of birth:	06.01.1988, Yenimahalle/ANK
e-mail:	zerasahin@gmail.com

Education

- Anadolu University, Eskişehir *(2013- 2018)*
  Electrical and Electronics Engineering, Post Graduate

- Middle East Technical University, Ankara (*2006 – 2012)*
  Physics, Undergraduate

Academical Interests

- Artifical Neural Networks
- Deep Learning
- Image Processing
- Machine Vision

Seminar

- Academic Informatics (2017)
- Hacker 101 (2017)
- SAVTEK (2010)
- Nanotech Bilkent

Communities

- LÖSEV (2015-)
- OSA (The Optical Society of America, 2012)

Voluntary Participation

- Design of a Wind Turbine, METU (2012-2013)
- TEGV (*2009)*