Cape Peninsula
University of Technology

FAULTS DETECTION AND CLASSIFICATIONS FOR GRID CONNECTED PV SYSTEM USING DEEP LEARNING METHODS

by

THOMAS LIONEL MAKOSSO

Dissertation submitted in fulfilment of the requirements for the degree

Doctor of Engineering in Electrical Engineering

in the Faculty of Engineering and Built Environment

at the Cape Peninsula University of Technology

Supervisor: Dr Ali Almaktoof

Co-supervisor: Prof Khaled Aboalez

Bellville campus

Date submitted (June 2025)

DECLARATION

I, Thomas Lionel MAKOSSO, declare that the contents of this dissertation represents my own unaided work and that the dissertation has not previously been submitted for academic examination towards any qualification. Furthermore, it represents my own opinions and not necessarily those of the Cape Peninsula University of Technology.

22 May 2025

Signed **Date**

Glossary

BPA: Back propagation algorithm

LSTM: Long short term memory

PV: Photovoltaic

DG: Distribution grid

MATLAB: Simulation software

RoC: Receiver over characteristics

SCADA: Supervisory Control and Data Acquisition

DA: Distribution automation

SG: Smart Grid

SAIDI: Average duration of network interruption

CAIDI: Customer average interruption duration index

Python : Programming Software

LV: Low voltage

MV: Medium voltage

HV : High voltage

Hif : High impedance faults

Lif : Low impedance faults

DFT : Discrete Fourrier Transform

STFT : Short time Fourier Transform

DWT: Discrete Wavelet Transform

CWT : Continuous Wavelet Transform

WT: Wavelet Transform

ANN: Artificial Neural Network

MLP: Multiple layer perceptrons

CNN: Convolutional neural network

RNN: Recurrent neural network

DRNN: Deep recurrent neural network

NLP: Natural language processing

DNN: Deep neural network

DRL: Deep reinforcement learning

BPA: back propagation algorithm

DN: Distribution Network

MSE: Mean square error

PLL: Phase lock loop

PCC: Point of common coupling

DWT: Voltage source inverter

GCI: Grid connected inverter

DC: Direct current

AC: Alternative current

GB: Gradient boost

IEEE: Institute of electrical and electronics engineers

ABSTRACT

This study provides a thorough approach for fault detection and classification in a 200 kW grid-connected photovoltaic (PV) system using Long Short-Term Memory (LSTM) neural networks. As PV systems and other renewable energy sources are being included into modern power networks, stability and reliability must be maintained. One of the biggest problems is recognizing and classifying short-circuit failures, which can result in serious damage and downtime if ignored. Conventional fault detection techniques frequently fall short in addressing their limitations in practical settings due to their inability to adjust to a variety of fault types and system dynamics. Traditional methods struggle with detection accuracy because system dynamics and fault types vary greatly. LSTM networks, which are great at managing time-series data and detecting temporal relationships in electrical signals, are used in the proposed method. By analysing voltage and current waveforms, the LSTM model effectively detects and classifies a wide variety of short-circuit fault types, including single-line-to-ground, line-to-line, and three-phase faults. The grid-connected PV system is simulated using Python, and synthetic fault data is generated for the training and validation of the LSTM network. By creating a solid dataset with a variety of fault scenarios, this study also fills an empirical gap and allows for a more thorough assessment of the model's performance. The model is trained on a large dataset that includes both usual operating conditions and different fault scenarios. Data augmentation approaches, like the Synthetic Minority Over-Sampling Technique (SMOTE), are also utilized to address the imbalance in the fault data so that the model can accurately categorize less prevalent problem types. The effectiveness of the LSTM-based fault detection and classification system is evaluated using confusion matrix measurements and receiver over the curve (ROC). The simulation's results demonstrate that the LSTM model outperforms conventional techniques in terms of accuracy, defect detection rates, and response times. In contrast to MATLAB, which offered configured blocks that were typically set to predict the best performance and did not permit investigation because poor results scenarios could not be evaluated and upgraded as the study that was conducted, Python allowed for a great understanding of the development of a deep learning model.

AKNOWLEDGEMENTS

I would like to thank my supervisor, Dr. Ali Almaktoof, for his guidance and support, and the co-supervisor, Prof. Khaled. Appreciation is also extended to friends and colleagues in the department, especially Dr. Ndeke, Dr. Kangaji, Dr. Showers, and Dr. Melamu, for their assistance and inspiration. Lastly, I dedicate this work to my mother, Francine MAKOSSO, and heartfelt thanks go to my siblings for their unwavering support and encouragement.

TABLE OF CONTENTS

List of Figures

List of Tables

CHAPTER ONE
INTRODUCTION

1.1 Introduction

Distributed fault detection and classification is crucial for the safety, effectiveness, and dependability of grid-connected photovoltaic (PV) systems (Asmar et al;2022). Since distributed PV systems are decentralized and can cover large areas, close monitoring is required for prompt problem identification and resolution (Biju et al; 2024). Early fault identification prevents minor issues from developing into more significant ones that could damage equipment, disrupt power output, or even jeopardize the grid's stability (Wu et al., 2020). Common faults, such as arc, short circuit, or ground faults, can significantly reduce performance or create hazardous conditions(Naidu et al; 2022).

Given the challenges in detecting faults accurately due to the varying fault types and system dynamics (Kumar et al., 2020), effective monitoring is crucial for ensuring the system's dependability and safety (Sarita et al., 2020). Early detection of defects is key to preventing minor issues from escalating into more serious ones that could compromise machinery or grid stability (Li et al, 2023).

To address these challenges, techniques for fault detection and classification have become crucial in order to overcome these obstacles and guarantee the safe and effective functioning of grid-connected PV (De-Jesús-Grullón et al; 2024) . In comparison to conventional approaches, recent developments in deep learning techniques have demonstrated considerable promise in terms of increased accuracy and dependability (Etukudor et al; 2021). In order to improve performance, dependability, and safety, this research project focuses on the creation and deployment of deep learning defect detection approaches for grid-connected PV systems (Hung et al, 2022).Several simulation scenarios were conducted, starting with voltage and current measurements on the distribution side and IEEE 13 test bus (Reilly, 2020). The study examined the impact of incorporating different Distributed Generators (DGs) into the distribution feeder, with a PI Controller developed to adjust the current and voltage before reaching the step-up transformer and IEEE 13 bus. This simulation was modeled using MATLAB/Simulink (Vardham et al., 2021).

Throughout the course of this research, a deep learning model based on LSTM networks was developed to assess, detect, and classify four types of short circuit faults: single-phase, two-phase, three-phase-to-phase, and three-phase-to-ground faults. The study was divided into two parts. In the first phase, a model was created on

MATLAB to simulate fault occurrences on the distribution side of the system. A faulty condition was defined as having a voltage of 0V and a current exceeding 50A. Various scenarios were tested, including evaluating the impact of the number of epochs, ranging from 500 to 1000; analyzing the effect of varying fault probabilities from 20% to 80%; and investigating how different learning functions impacted the model. Fault detection and classification were then applied, categorizing faults into four classes: no fault, single-phase, two-phase, and three-phase faults, as well as a specific class for three-phase-to-ground faults. The ReLU and Softmax functions were used for the input and output of the network, respectively.

In the second phase, measurements from the IEEE 13 bus were exported to Python, where an LSTM model was developed with the same structure as the MATLAB model to track, detect, and classify faults. The purpose of this approach in Python was to demonstrate that, in sequence-based data like this, issues such as over- or under-sampling could affect the training process and degrade model performance. To address these challenges and improve classification of less common fault types, data augmentation techniques, such as the Synthetic Minority Over-Sampling Technique (SMOTE), were employed to rectify data imbalances. The performance of the LSTM-based fault detection and classification system was assessed using the receiver over the curve (ROC) and confusion matrix.

The results of the simulation revealed that, compared to traditional methods, the LSTM model achieved higher accuracy, better fault detection rates, and faster response times. In the distributed fault detection scenario, the Levenberg-Marquardt algorithm performed best, with a mean square error of 5.5×10^-12, and the fault probability ratio of 0.8 helped distribute the different fault scenarios more evenly. An increase in the number of epochs significantly enhanced the model's overall performance. Out of 1000 simulated fault scenarios, the model demonstrated exceptional results, achieving 90% true positive and false negative predictions.

1.2 Research problem

The integration of renewable energy sources into traditional power networks is being advanced in large part by grid-tied PV installations. However, stable power distribution and grid reliability are challenged by their vulnerability to faults such as open circuits, ground faults, partial shading, and short circuits. The requirement for effective fault management to guarantee uninterrupted and secure operation is growing along with the use of PV systems. The flexibility needed to manage the complicated fault patterns and fluctuating operating circumstances in PV systems is frequently lacking in traditional fault detection solutions, which rely on physical sensors and rule-based

algorithms. The efficacy of these techniques in big, grid-connected PV systems may be limited by their inability to provide automated, real-time fault responses.

Deep learning techniques, which can learn from enormous volumes of data and recognize minor fault patterns on their own, have become viable substitutes for fault detection and classification in PV systems. It is still difficult to create reliable, accurate deep learning models specifically for grid-connected photovoltaic systems. The use of deep learning in this field is hampered by problems including model complexity, data needs, and the necessity for accurate, quick categorization under shifting load and weather conditions. By examining and applying deep learning methods that can identify different kinds of faults in real time, this study seeks to close these gaps. The goal is to provide a model that, in grid-connected contexts, improves fault detection accuracy, reduces response times, an acceptable level of reliability and offers a greater energy quality and decrease the rate of faults. .

1.3 Research objectives

The research has two levels of objectives:

i. To detect and classify faults in real time using deep learning in grid-connected PV systems, ensuring reliable operation.
ii. This enhances energy quality, safeguards equipment, and reduces the risk of power outages.

The secondary objectives aim to:

i. Address limitations of conventional fault detection methods in terms of accuracy and robustness.
ii. Introduce a rigorous method for evaluating synthetic fault data.
iii. Detect short-circuit faults on the AC side of a 200kW grid-tied PV system.
iv. Improve performance using a modified oversampling strategy based on SMOTE.

1.4 Research questions

This research investigates the solution to the questions that follow:

- ✓ What are the recent methods, and how can they be categorized?
- ✓ How can the analysis of the fault be automated and avoid using complex mathematical calculations and manual segmentation?
- ✓ How can data be used, combined and transformed with innovative deep learning techniques to instantly detect faults

1.5 Methodology

A particular technique was established for the best theoretical underpinnings, simulation models, and commercial software in order to address the study problems. The objectives include:

- ✓ Conducting a systematic literature review on fault location techniques in PV-integrated grids and their limitations.
- ✓ Modeling a PV-integrated network with various customers in MATLAB/Simulink.
- ✓ Analyzing performance using the IEEE 13-Bus system.
- ✓ Developing an LSTM-based fault detection and classification model in Python.
- ✓ Enhancing model performance with a modified SMOTE algorithm.
- ✓ Identifying research gaps and suggesting directions for future work.

1.6 Outline of the dissertation

The structure of the dissertation has a composition of seven chapters; the details of what each chapter entails are as follows:

1.6.1 Chapter One

This chapter is the overall introduction of the research, where the study's background is described, highlighting the purpose, the aim and objective, and the whole dissertation structure.

1.6.2 Chapter Two

A survey of the literature on the distribution network's general structure opened this chapter. A chapter on the basic components of an electrical distribution network typically covers the infrastructure required to safely and efficiently move power from substations to end users. Transformers, which adjust voltage levels to suitable distribution values; distribution lines, which transport power over various distances; and protective devices, including fuses and circuit breakers, which prevent network failures, are crucial components. The deep learning idea is then expanded, which further highlights the limitations of the current approaches and illustrates how the proposed method is a contribution of this research that is different from the current approaches.

1.6.3 Chapter Three

This chapter tackled the overview of the distribution system and protection. Then a deep study of three different distribution networks will be undertaken. Finally, the Structural configuration of the PV system and the battery modelling will be presented.

1.6.4 Chapter Four

This chapter analysed the mathematical modelling of a three phase grid connected PV system, and also description of the tools used in the proposed LSTM model developed in the algorithm

1.6.5 Chapter Five

The detection and classification of low and medium voltage faults is the focus of this chapter. The incorporation of a deep learning-based fault detection technique that uses LSTM to identify problems on both the IEEE bus side and the distributed side (PV inverter side). The application of a suggested improved SMOTE approach to enhance performance follows.

The outcomes of numerous case studies were thoroughly examined and discussed on a variety of forums.

1.6.6 Chapter Six

The main contributions of the research are enumerated, explaining the impact of the results and their importance to the industry. Further recommendations for future research are made.

CHAPTER TWO

LITERATURE REVIEW

2.1 Introduction

The primary function of an electricity distribution network is to deliver power from one point to another, typically operating in a radial mode with one-way power flow. Distributed generation (DG) has emerged due to energy privatization, the push to reduce greenhouse gas emissions, and advancements in small-scale generation(Sistani et al. 2023).

DG technologies include micro turbines, fuel cells, photovoltaic systems, and biomass. These systems use various generators: synchronous generators for combustion engines and solar thermal, power electronic converters for solar, fuel cells, and micro turbines, and induction generators for wind farms. Modern wind turbines also feature variable-speed operation and integrated power electronics. (Harrou et al. 2019).

Distribution networks is often affecting while relying on fossil fuels and critical choice of the grid architecture. Reason why this chapter gives an overview of the differents architectures and their applications in order to provide the most appropriate one for grid connected systems.

2.2 Overview of a Distribution System

The distribution substation serves as the link between the power distribution system and the upstream power delivery system. The substation is equipped with a range of equipment to guarantee precise measurement, switching, and protection (Hategekimana et al. 2024). The sub-transmission voltage level is reduced by the sub-station transformer (HV/MV) to a value appropriate for primary distribution lines, also called feeders, and then stepped down to a level appropriate for final consumption (400V/230V) by the distribution transformer (MV/LV) (De-Jesús-Grullón et al. 2024). Energy is delivered to consumers via low voltage secondary distribution lines, which are primarily single-phase with a few three-phase circuits.Figure 2.1 depicted a typical power distribution system configuration.

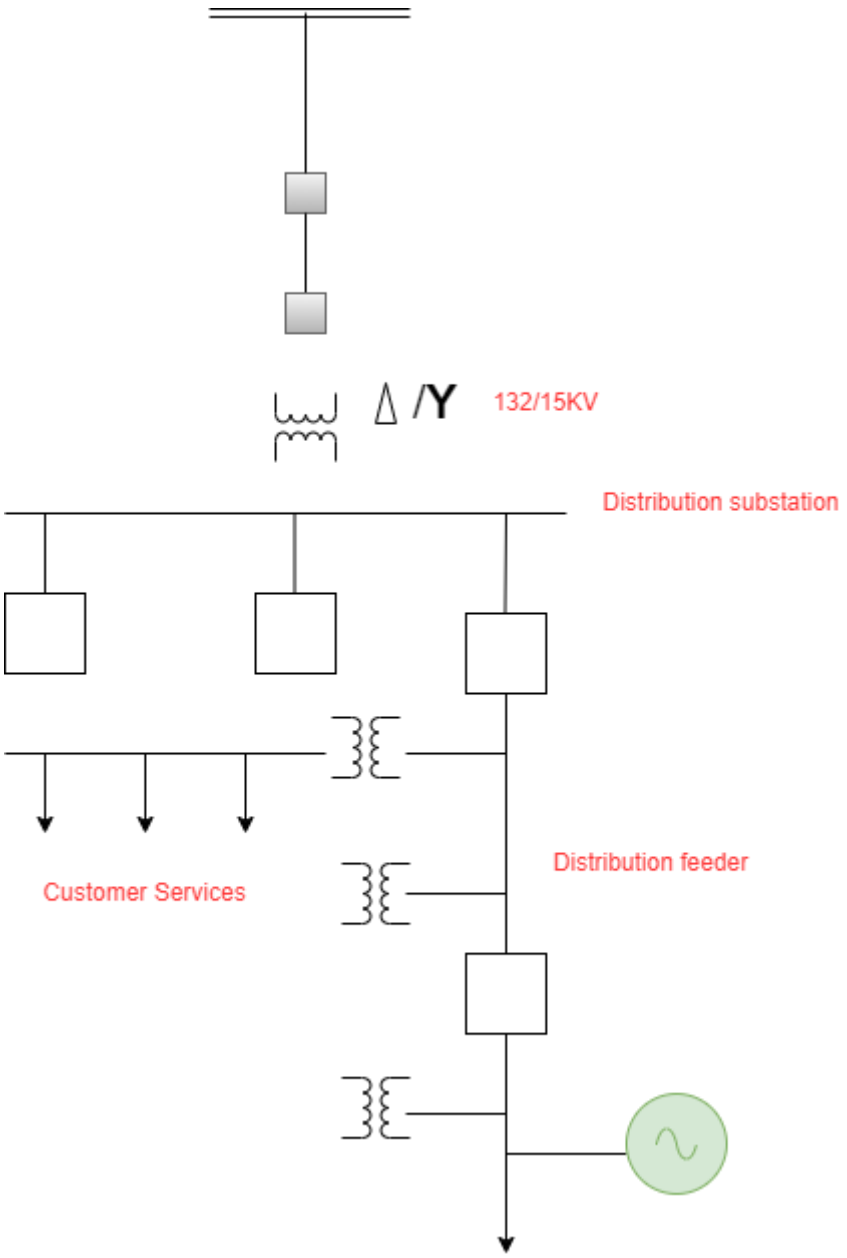

Figure 2.1: Power distribution system configuration (Uddin Khan et al. 2019)

2.3 Substation of a distribution system

Certain important factors are considered when designing a distribution substation, such as load level, auxiliary equipment required, anticipated reliability, and system type (urban, suburban or rural).

2.3.1 Rural Substation

A suburban substation typically includes one high-voltage bus and a separate medium-voltage bus for each transformer. However, some utilities opt for a single medium-voltage bus shared by all transformers to achieve better load balance. Due to higher

load demands in suburban areas compared to rural ones, multiple transformers are necessary. Medium-voltage buses are interconnected using a normally open tie-switch, which also supports nearby rural substations. A rural substation is depicted in Figure 2.2. When a transformer fails, the tie switch is activated, and the load served by the failed transformer is taken over by another in-service transformer.

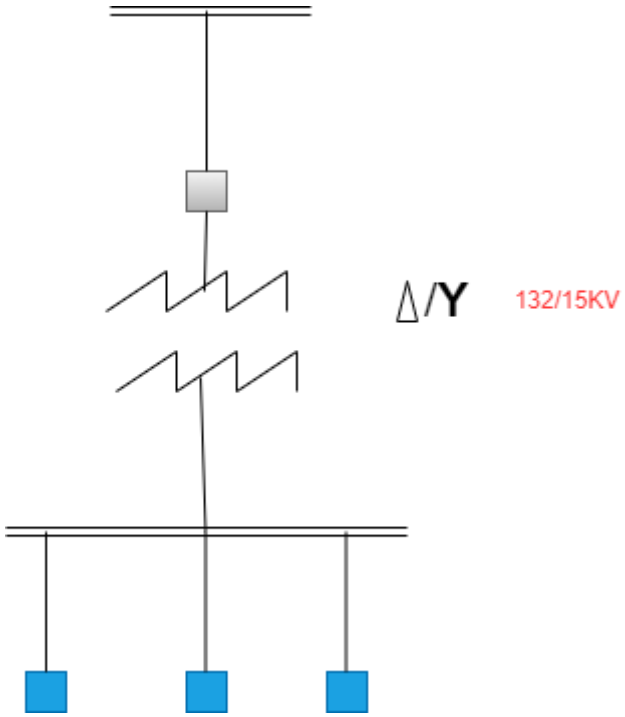

Figure 2.2: Rural substation (Khan et al. 2019)

2.3.2 Urban Substation

Urban substations commonly use ring-bus and breaker-and-a-half configurations. In the ring-bus design, medium-voltage buses form a closed loop separated by circuit breakers, allowing connections at the midpoint between breakers. In the breaker-and-a-half design, branches with three circuit breakers link two medium-voltage buses, enabling connections to distribution lines or transformer secondaries. Both designs support flexible load transfer and maintenance through configuration adjustments.

2.4 Distribution System Components

The distribution network in electricity delivers power from generation sources to consumers, involving transformers, lines, circuit breakers, fuses, reclosers, switchgear, and metering systems. These components ensure reliable energy

distribution, ensuring efficient and uninterrupted electricity supply to residential, commercial, and industrial users (Salehimehr et al. 2024).

2.4.1 Transformer

A transformer is a device that consists of a primary and secondary winding that are connected by electromagnetic fields. It transfers power from one winding to another by changing voltage level (Lee et al. 2020). Transformers reduce the voltage in stages to provide the appropriate voltage level required by each section of the distribution system. The transformer operation is illustrated by equation 2.1.

$$\frac{V_p}{V_s} = \frac{n_p}{n_s} \tag{2.1}$$

Where V_p= potential difference (voltage) input on the primary coil, V_s= potential difference (voltage) output on the secondary coil, n_p = number of turns (coils) of wire on the primary coil and n_s = number of turns (coils) of wire on the secondary coil

2.4.2 Current Transformer

It's key equipment used for measurement of alternating current by conversion of high current levels to lower and proportional values. In power generation, transmission and distribution, it plays a significant role by facilitating current measurement and stem protection .

2.4.3 Lines

A line serves as a conduit for the transmission of electrical energy from one place to another (Rezapour et al. 2024). This might be done above ground or beneath. In contrast to underground lines, which are often cables composed of polymer-insulated materials like Cross-Linked Polyethylene (XLPE) and Ethylene Propylene Rubber, overhead lines are typically made of bare aluminium, typically of the Aluminium Conductor Steel Reinforced (ACSR). The rated voltage and current carrying capacity of distribution lines are important characteristics (Han et al. 2019).

2.4.4 Circuit Breaker

A circuit breaker's purpose is to safeguard other equipment and itself by automatically isolating a circuit at a specific overcurrent value. It is merely a switching element made to shut and open a circuit using manual methods (Batmani, Takhtabnus, and Mirzaei 2022). It is separated into three sections: power, op-amp, and relay.

2.4.5 Relay

Relays are low-power switching components used in distribution networks to turn on high-power components. By instructing the associated circuit breakers to trip, a relay

protects feeders and equipment by cutting off the current created by a failure (Nair and Costa-Castello 2020).

2.4.8 Recloser

A recloser is an intelligent switching device that can detect overcurrent, stop fault current, and automatically close again to reenergize the line. When a persistent failure arises, the recloser locks open after three or four operations (often programmed), separating the malfunctioning portion of the system from the remainder of the system (Ayiad, Leite, and Martins 2021).

2.4.9 Fuse

An overcurrent or short-circuit current will cause a strip of wire within a fuse to melt. The melting and clearing times are determined by the time-current curves of the fuse. In distribution systems, laterals, secondary circuits, and low power transformers are protected by fuses, often of the K and T kinds (Sadeghkhani et al. 2018).

2.4.10 Sectionalizer

A sectionalizer is a safety feature that works in conjunction with source-side safety features to automatically isolate faulty electrical distribution system portions. Reclosers or circuit breakers are examples of such source side protection devices(Pillai et al. 2023). The source-side safety mechanism opens to deenergize the circuit when the sectionalizer detects current flow in excess of a certain amount (Kumar et al. 2020). The sectionalizer then counts the overcurrent interruption.

2.4.11 Switch

A switch is a tool used to isolate a system component for maintenance or repair. It must be capable of carrying and severing currents under typical working circumstances. A switch may have the ability to cause faults, but it is not intended to break fault current. It may operate under specific abnormal conditions, such as those of a short circuit, and carry currents for a specific period of time (Sistani et al. 2023).

2.4.12 Voltage regulator

A transformer with an on-load tap changer that has a nominal transformation ratio of 1:1 is a voltage regulator. At the halfway point of long primary lines, voltage regulators are added to make up for the voltage drop that is caused (Sistani et al. 2023).

2.4.13 Supervisory Control and Data Acquisition (SCADA)

A communication system called SCADA is utilized to monitor the distribution network in real-time. Users and programs can access the database where SCADA keeps the

data gathered throughout the system. Circuit breakers and switches can be controlled remotely by SCADA, which makes system operations flexible and speeds up switching actions.

2.5 Distribution automation

Automation involves performing tasks automatically with precision and efficiency(Ahmad et al. 2021). In the power sector, it plays a key role in transforming traditional grids into smart grids through the integration of information and communication technology (ICT). Smart grids incorporate smart appliances, smart meters, energy-saving tools, and renewable energy sources, creating an advanced and efficient electricity infrastructure.(Haddadi et al. 2020). "A system that allows an electric utility to view, manage, and control components of a distribution system in a real-time fashion from remote locations," according to IEEE, is what Distribution Automation (DA) is defined as (Hategekimana et al. 2022). Distribution Automation (DA) enhances control, consistency, and efficiency in power distribution by enabling real-time monitoring, fault detection, and automatic switching to isolate issues and restore supply. While it doesn't fully eliminate the need for manual intervention, DA improves system performance by providing accurate, timely data for decision-making. Utilities invest in DA to boost reliability, meet regulatory standards, and increase revenue through improved service quality. Although DA is mainly applied upstream, it's increasingly used downstream near customer sites when commercially justified or contractually required. Advances in cost-effective technologies and deregulation have made DA more financially viable and widely adopted.

2.6 Power quality

Distribution automation enhances utility performance by improving power quality in line with standards like NRS 048-2:2007. While many electronic devices have built-in protection against poor-quality power, automation helps maintain consistent supply quality. It also enables active voltage regulation through remote control of capacitor banks and voltage regulators.(Forouzesh et al. 2021).

2.7 Availability and reliability

Network quality is closely linked to the reliability of the electrical distribution system, focusing on outages and customer interruptions during normal operations. It's important to differentiate between normal and standby conditions, with standby referring to temporary network states following faults or maintenance. Various events, such as contingencies, open circuits, faults, and both momentary and sustained interruptions, can affect reliability and disrupt normal operations(Caicedo et al. 2022)..

The key difference between a momentary and sustained interruption is the duration of the outage. However, distinguishing between the two has become increasingly challenging as network restoration times continue to improve, resulting in progressively shorter momentary interruptions (Cao et al., 2024). The Institute of Electrical and Electronics Engineers (IEEE) defines a momentary interruption as an outage lasting up to five minutes, though some countries, like Spain, have reduced this threshold to three minutes.

In addition to power quality and reliability, electrical network performance is also evaluated by the probability of components remaining energized. Harmonics can negatively impact the system by reducing efficiency, accelerating equipment aging, and causing malfunctions in various components.

2.8 Smart Distribution System

The smart grid (SG) is considered the next-generation energy system due to its controlled emissions, defined demand, enhanced efficiency, and cost-effectiveness. It offers greater reliability, safety, environmental sustainability, and security compared to traditional electricity systems.

(Salehimehr et al. 2024). According to the European Regulators Group for Electricity and Gas, "a power system that can effectively coordinate the conduct and activities of all stakeholders linked to it - generators and/or consumers - to ensure commercially viable, workable energy system with commensurate quality levels and less losses with a safe and secure supply" is referred to as a system of gauges.

They also enable a variety of activities that are trustworthy and secure in the distribution system. The process of the smart grid's self-healing may control the various network situations, leading to optimization and caution under normal conditions and fault analysis under faulty conditions, ultimately leading to network reconfiguration and the restoration of the power supply. In life-threatening situations, self-healing allows for disconnection from the main grid. In this situation, distributed generating and power storage technology are used to ensure supply continuity. Distribution network flaws can be either transient or persistent(Sistani et al. 2023). If the fault is persistent, the relevant protective relays permanently disconnect the indicated faulty portions or equipment. So, it's crucial that a fault's location be known or able to be projected with logically high precision. As a result of the quicker restoration of the power supply, the cost and time for inspection and restoration are reduced, and better services are also provided. Customers are more sensitive to power interruptions in modern societies and with the advent of the "smart power grid." Thus, improved methods for locating, identifying, and monitoring faults are needed(Batmani, et al. 2022). This will lead to an

enhanced method for restoring power supplies, a decrease in the overall length of power outages, an increase in customer satisfaction, and a decrease in operating costs.

Reliability in distribution systems focuses on minimizing customer interruptions and equipment outages. Under normal conditions, all equipment (except standby) is energized, and customers receive uninterrupted power. However, planned and unplanned events can disrupt this. Enhancing reliability aims to ensure continuous power supply, reduce outage frequency and duration, minimize impacts, meet standards, evaluate system performance, and identify outage causes to implement corrective measures—ultimately reducing the high cost of outages for customers (Haddadi et al. 2020).

Power Systems Reliability Indices are measures of the dependability and accessibility of power supply from the utility in relation to the experience of outages suffered by different customers. The performance reliability indices measure interruptions with respect to the frequency, duration, number of customers affected and that of the installed plant (transformers) impacted by the events on the network. Each index is defined by different equations as described in the following subsections(Ahmad et al. 2021).

2.9 Sustainable interruption indices

The primary objective of these indices is to assess network reliability from the user's perspective**,** where the user refers to the energy consumer within the system (Jayamaha et al., 2019). By calculating these indices, it is possible to determine the overall quality of the electrical network, which depends not only on the continuity of supply but also on waveform quality and the commercial relationship between customers and the utility provider**.**

2.9.1 SAIFI

The SAIFI index aims to assess the frequency with which a user experiences a sustained interruption over a set period of time. This index is a ratio of the number of users affected by the interruption to the total number of network users.

$$\text{SAIFI} = \frac{1}{N}\sum_{i=1}^{n} I_i \tag{2.3}$$

Where, I_i is the number of interruptions from user i and N is the total number of users.

2.9.2 SAIDI

The SAIDI index determines the average duration of network interruptions over a given period of time. The unit of measurement is the hour or minute per year. As a result,

this index defines the relationship between total interruption duration and total number of users.

$$\text{SAIDI} = \frac{1}{N}\sum_{i=1}^{n} D_i \tag{2.4}$$

where, D_i is the total duration of the user i and N is the total number of users.

2.9.3 CAIDI

The Customer Average Interruption Duration Index (CAIDI) determines how long it takes to restore the network on average.

$$\text{CAIDI} = \frac{SAIDI}{SAIFI} = \sum_{i=1}^{n} D_i \left(\sum_{i=1}^{n} I_i\right)^{-1} \tag{2.5}$$

The index reflects the network's restoration time from the user's perspective and incorporates SAIDI and SAIFI metrics. Its value is typically lower than the actual restoration time, as some customers regain power through alternative circuits during the recovery process.

2.9.4 CTAIDI

The Customer Total Average Interruption Duration Index (CTAIDI) displays the total amount of time spent during the information period. Unlike the CAIDI index, this index only counts the users who have experienced an outage once(Salehimehr et al. 2024). In other words, if a user has experienced several interruptions, the index will only count one time within the total number of users who have experienced an interruption.

$$CTAIDI = \sum_{i=1}^{n} D_i \left(\sum_{i=1}^{n} U_j\right)^{-1} \tag{2.6}$$

where, U_j takes the value 1 if the user j has suffered one or more interruptions or the value 0 if there has not suffered anything. Thus, the affected user is counted one time.

2.10 Conventional distribution network protection

The primary goal of power system protection is to ensure the safety of the crew, equipment, and people while the power system is operating. When creating an effective protection system, the following rules ought to be followed:

i. Reliability: A security system needs to be dependable. The ability to differentiate between mistakes and situations that don't require any action is referred to as "correct operation."
ii. Selectivity: A protection system should cut off the smallest possible segment of the network in order to isolate the breakdown.
iii. Speed: A fault must be addressed as soon as possible to avoid dangerous situations for humans and damage to equipment.

Different network architectures require different protection techniques. The fundamental principal of overcurrent prevention is that network problems typically result in very large currents as compared to nominal load values. Additionally, because the impedance between the power supply and the fault point varies, failures in various sections of the network result in currents of various strengths (Kumar et al. 2020). In order to minimize disconnected plant and provide acceptable service continuity and selectivity, coordination of protection devices along a graded path is crucial. Devices are synchronized so that the one closest to the fault activates first in the event of a fault. In this manner, the issue is isolated and as many unaffected loads as feasible are still receiving electricity. This coordination is quite simple because the power flow is unidirectional.

This coordination is relatively simple to do because power flow is unidirectional and faults further down the feeder have lower fault current due to increased impedance, among the source and the defect (Forouzesh et al. 2021).

Overcurrent protection devices commonly used in distribution networks include overcurrent relays, fuses, reclosers, and sectionalizers. Overcurrent relays operate based on time/current characteristics and are typically classified into three types:

i. Definite-current relays, which operate instantly once a set current threshold is exceeded.
ii. Definite-time relays, which operate after a fixed time delay when the current surpasses a threshold.
iii. Inverse-time relays, where the operating time decreases as the fault current increases (Zaben et al. 2024).

Figure 2.2, as an illustration, displays the coordinated operating characteristics of four overcurrent relays in a radial feeder. A time inverse element and an instantaneous element are provided for each relay. Their temporal coordination is configured so that, in the event of a problem, the closest device to the fault opens first.

The bulk of faults (between 80% and 90%) in distribution networks are transient, which has a significant impact on the design of the protection systems. In order to allow for the clearing of the problem without permanently disconnecting the defective section of the network, it is usual practice to repeatedly detach and re-connect the affected circuit. A recloser, a device that can detect and stop an overcurrent before automatically closing to re-energize the feeder, is used to enable this practice. For rapid and slow operation, reclosers typically have various inverse time/current operating characteristics (Dai et al. 2019).

If the fault is permanent, sectionalizers and fuses coordinate with the recloser to isolate the faulted section selectively.

To design and coordinate protection systems, voltage levels are categorized using standard classifications. It's important not to mix standards, so both Institute of Electrical and Electronics Engineers (IEEE) and International Electrotechnical Commission (IEC) definitions are shown separately below:

IEEE Voltage Classification:

- Low Voltage (LV): < 1,000 V
- Medium Voltage (MV): 1,000 V ≤ V < 100,000 V
- High Voltage (HV): 100,000 V ≤ V ≤ 230,000 V

IEC Voltage Classification (IEC Standard No. 38):

- Low Voltage (LV): 100 V – 1,000 V (commonly 400 V, 690 V, 1,000 V at 50 Hz)
- Medium Voltage (MV): 1,000 V – 35,000 V (standard levels include 3.3 kV, 6.6 kV, 11 kV, 22 kV, and 33 kV)
- High Voltage (HV): 35,000 V – 230,000 V (with common ratings such as 45 kV, 66 kV, 110 kV, 132 kV, 150 kV, and 220 kV) (Xie et al. 2023).

These classifications guide the selection and coordination of protective equipment suitable for specific voltage ranges in power distribution networks.

2.11 Substation and Distribution System

Sub-transmission lines transport substantial amounts of power at intermediate voltages from bulk power substations to the primary distribution substations in the vicinity of the point of consumption (Jayamaha et al. 2019). Direct medium voltage distribution centres within residential neighbourhoods or major industrial and commercial consumers are served by the medium voltage network, which receives electricity from the main distribution substations. Transformers for step-down distribution receive electricity from distributor centres via primary feeders. The main switchboard of the building receives electricity from the distribution transformer via a three-phase, 380 V, four-wire secondary cable (Ali et al. 2021).

2.11.1 Generating station

Generating Station refers to the location where three-phase alternators and generators connected in tandem produce electricity(Nougain et al. 2023). The usual generating voltage might range from 11 kV to 11.5 kV to 12 kV to 13 kV. However, it is beneficial economically to use a step-up transformer to increase the produced voltage to 132 kV, 220 kV, or 500 kV or more (power Transformer).

2.11.2 Primary transmission

The overhead transmission system transmits the electric supply (in 132 kV, 220 kV, 500 kV or larger) to the load centre.

2.11.3 Secondary Transmission

Secondary transmission refers to a remote area with a line connection to a receiving station. At the receiving station, step-down transformers drop the voltage level to 132 kV, 66 kV, or 33 kV. Electric power is then transmitted to various sub stations via a three-phase, three-wire overhead system.

2.11.4 Primary Distribution

Step-down transformers at substations reduce secondary transmission voltages of 132 kV, 66 kV, or 33 kV down to 11 kV. Electricity is then supplied to large consumers requiring 11 kV, with dedicated substations managing and distributing the power accordingly (Naidu et al., 2022). These substations receive electricity from secondary transmission or primary distribution networks at 132 kV, 66 kV, or 33 kV to meet high-demand industrial consumers. The voltage is further stepped down within their own substations for specific applications, such as electric traction and other large-scale operations.

2.11.5 Secondary Distribution

The distribution substation receives electricity from the primary distribution line (11 kV) and is located near consumer areas. Within these substations, 440V step-down transformers are used to reduce the voltage level (Wang et al., 2021). These are three-phase, four-wire distribution transformers, providing 230V (single-phase supply) between the neutral and phase (live) wires, and 400V (three-phase supply) between any two phases. While three-phase loads can be connected directly to the three-phase lines, residential loads (such as fans, lights, and televisions) are typically connected to any phase and the neutral wire.

2.12 Radial Distribution

Radial distribution is widely used for electricity delivery due to its simplicity, affordability, and efficient design, making it ideal for most micro grids. Globally adopted and responsible for 90% of power distribution in South Africa, it connects generation points to homes through a single path. While easy to implement, its main drawback is limited stability, as power flows in only one direction. If a fault occurs along the line, all connected consumers lose power. Radial, ring, and mesh networks represent evolving configurations, with the radial ring as a variation and the mesh network as a more complex extension. Figure 2.3 depicted the radial distribution.

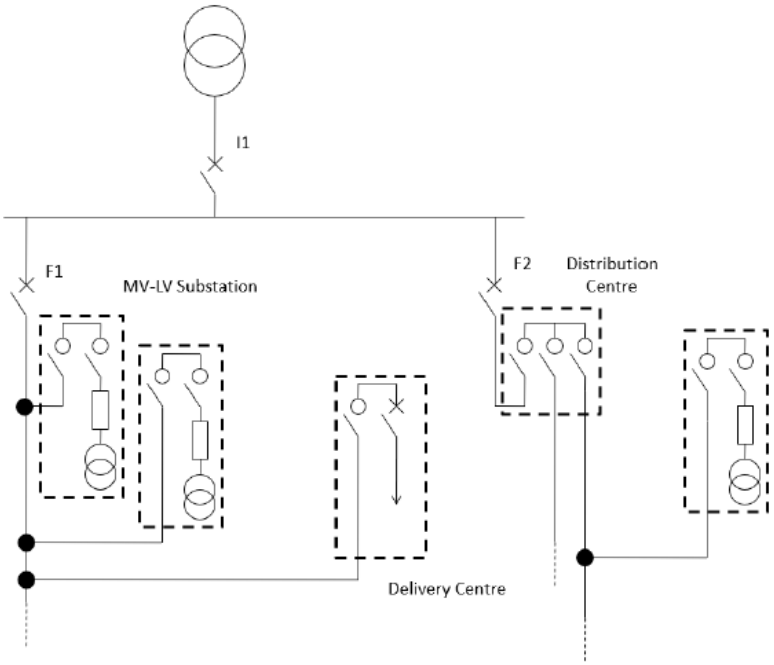

Figure 2.3: The radial distribution(Ahmad et al., 2021)

Figure 2.4 presents the radial distribution with the switching areas. With this architecture the network is assimilated to a straight line, thus allowing limited actions in case of fault.

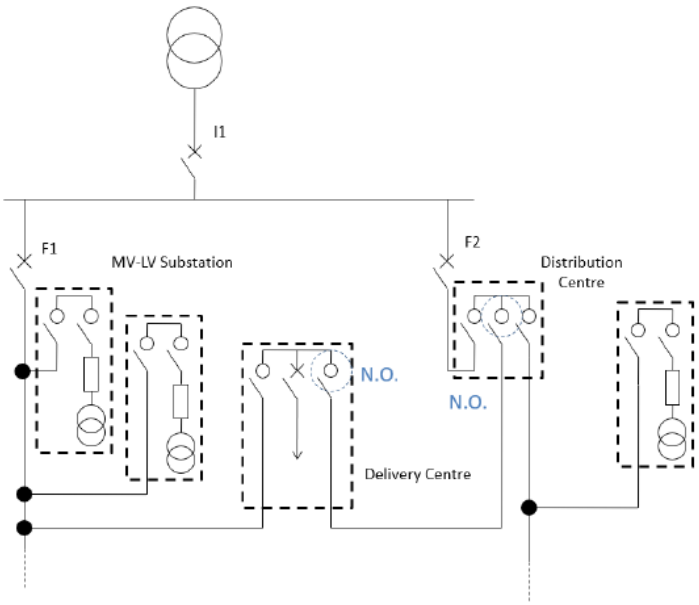

Figure 2.4 :The radial distribution with the switching points(Ahmad and Asar 2021)

2.13 Loop distribution

The loop distribution system is increasingly considered for microgrids as an alternative to radial systems, especially in Europe. It delivers power in both directions, enhancing reliability and efficiency. If a fault occurs, power can still reach consumers via alternate paths, either automatically or manually. However, loop systems are more costly due to the need for additional switches and conductors compared to radial systems.(Cao et al. 2024).

2.14 Network Distribution

To distinguish fault surge characteristics from those specific to capacitor switching, load switching, and arc-producing devices, the harmonic analysis method looks at the higher harmonic currents (Rezaei et al. 2022). These frequency features propagate in both directions from the fault point. These waves encounter disturbances as they travel along the feeder, and as a result, they are reflected back to the fault area and cluster at the relay terminal to generate a signal that is highly interconnected. The initial levels of the waves depend on the frequency, location on the feeder, and route resistance of the fault. The Figure 2.5 presents the loop distribution (Mazibuko et al. 2024).

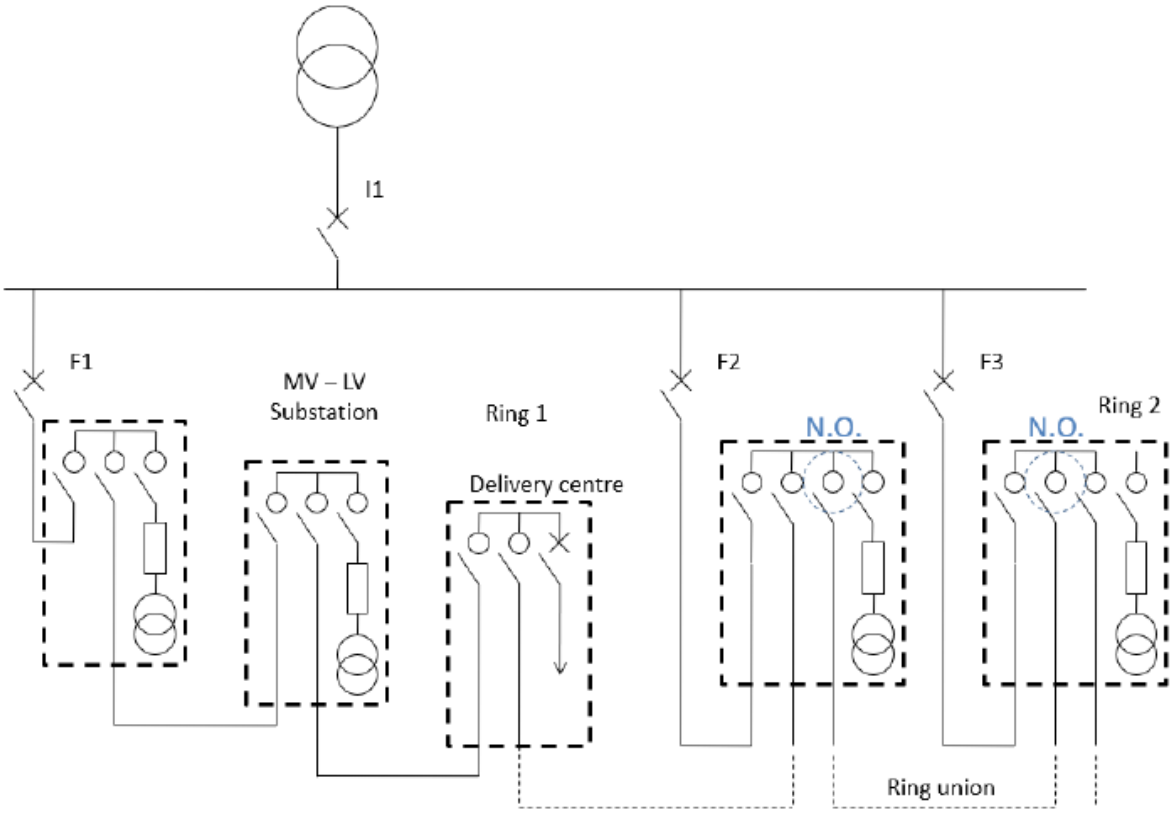

Figure 2.5: The loop distribution (Ahmad and Asar 2021).

2.15 Ring distribution

Ring distribution networks offer improved fault recovery due to the presence of two switch-disconnectors between each network frame. When a fault occurs, the system can be quickly reconfigured and restored using alternate paths fed by separate substations. Unlike radial distribution, MV-LV substations in ring networks have switch-disconnectors at both input and output, allowing flexible connection and isolation of network nodes. Typically, one switch-disconnector remains open, and in case of a fault, only the affected feeder responds due to circuit breakers. This design ensures that voltage loss is limited to the faulted section, maintaining supply in the rest of the ring. Figure 2.6 depicts the ring distribution.

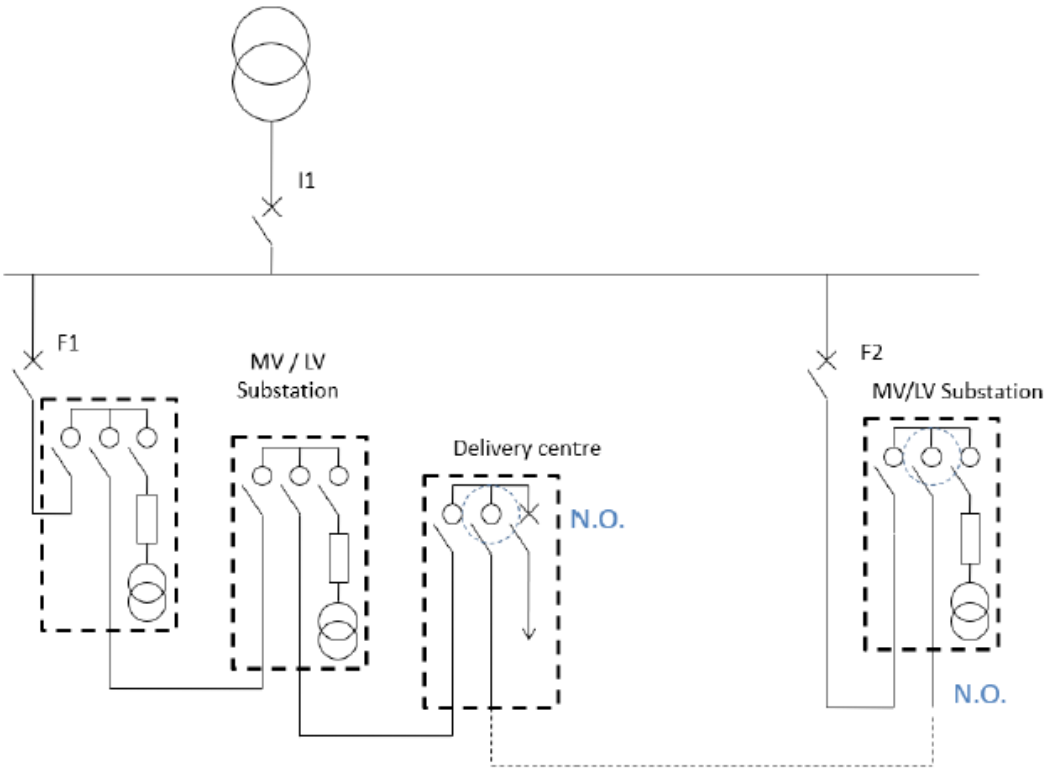

Figure 2.6: The ring distribution (Ahmad et al. 2021).

2.16 Summary

Understanding the electricity distribution network is essential for delivering energy reliably, safely, and efficiently. Key components like transformers, circuit breakers, and switchgear help manage power flow and maintain system integrity. This chapter highlights how failures in any part of the network can affect the whole system,

emphasizing the importance of regular maintenance, technological upgrades, and effective monitoring. It also discusses how advancements in metering and load management support performance and demand control. With growing energy demand and renewable integration, enhancing the resilience and adaptability of distribution networks is vital for future power delivery improvements.

CHAPTER THREE

FAULT ANALYSIS IN THE DISTRIBUTION NETWORK

3. 1 Introduction

Faults in electrical power systems are inevitable and can occur randomly, disrupting the system temporarily or permanently. These faults can be caused by various factors, such as trees falling on power lines, wind damage, vehicle collisions, lightning, or vandalism. While traditional grids take longer to clear faults, smart grids use intelligent electronic devices (IEDs) and algorithms to resolve faults more quickly. Faults in distribution networks (DN) lead to power quality issues like voltage sags, outages, and increased operating costs. Traditional fault-location methods are time-consuming and rely on surveys and basic data collection at substations. Fault detection and management have long been challenges in power distribution systems(Zhang et al. 2018).

3.2 Categories of faults

Understanding fault categories is essential for effective fault detection and recovery in power systems. Faults disrupt system operation and can cause significant damage. They are classified into three types: active faults, which involve unintended connections; transient faults, which are temporary and resolve after removal; and permanent faults, which require intervention and may cause extended outages. Faults in power systems can be symmetrical, affecting all phases equally with high current levels, or asymmetrical. Effective detection and mitigation strategies are crucial for system resilience and reliable power delivery. Faults are classified into various categories, though many share similar subtypes.

3.2.1 Active Faults

An "Active" fault occurs when there is current flow between conductors (phase-to-phase) or between a conductor and the ground (phase-to-earth). It is further divided into two types: solid and incipient faults. A solid fault is caused by a sudden failure of insulation, such as when an excavator damages a buried cable. An incipient fault starts small, like an incomplete discharge in the insulation, and grows over time, potentially evolving into a solid fault due to factors like pollution or high-resistance contact causing tracking.

3.2.2 Passive faults

Passive faults refer to conditions that increase stress on a system, potentially leading to active faults over time. Examples include:

a) Overloading: Overheating insulation, which deteriorates quality and reduces lifespan.
b) Overvoltage: Excessive pressure on insulation.
c) Under frequency: Causing plants to malfunction.
d) Power swings: Occurring when generators fall out of sync with each other.

3.2.3 Transients and Permanents Faults

They do not cause permanent damage to the insulation but allow the network to safely come back online after a short period of time. A common example is an insulator flashover caused by a lightning strike, which could be cleared successfully and then reclosed automatically if circuit breakers are opened (Kalimuthukumar et al. 2021). Transient faults occur primarily on outdoor equipment where the core-insulating channel is the air. A permanent fault occurs when there is permanent damage to the insulation. When this occurs, there must be no room for reclosing, and the equipment must be repaired as needed.

3.2.4 Symmetrical & Asymmetrical Faults

A symmetrical fault is a balanced fault with sinusoidal waves that are equivalent around their axes and represent a steady state condition. A symmetrical fault has the same and equal effect on all three phases (Li et al. 2023). An asymmetrical fault is a momentary unbalanced fault that exhibits a direct current offset before decomposing to the steady state of the symmetrical fault after a certain time duration. An asymmetrical fault does not have the same and equal effect on all three phases. The offset quantity is determined by the electric system's X/R (power factor), and the ratio of the first peak to the steady state value can be as high as 2.55 (i.e. 2.55 times the steady state value) (Li et al. 2023).

3.2.5 Balanced Three-Phase Unsymmetrical Faults

When all three phases of the transmission line or the machine's terminals come into contact with one another, it results in balanced three-phase unsymmetrical faults. The phase currents at the generator's terminals are caused by balanced three-phase unsymmetrical faults. Figure 3.1 illustrates the sequence networks of three-phase Faults.

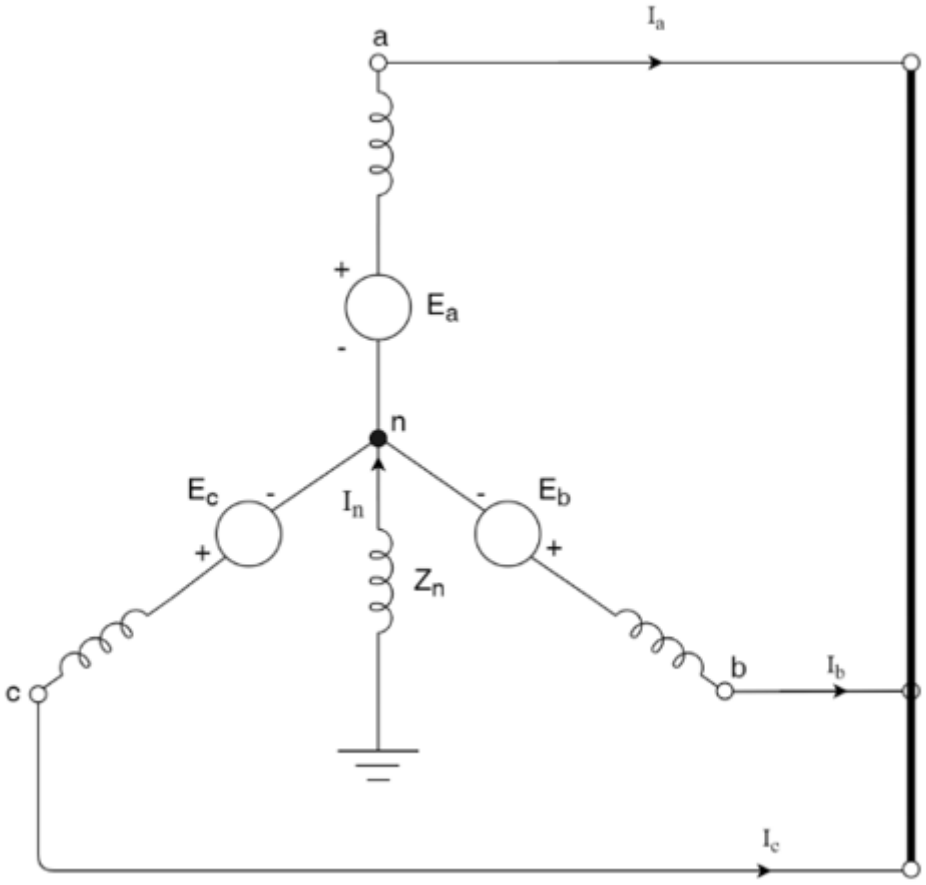

Figure 3.1: Sequence networks of three-phase faults (Singh et al.2022)

The equations below illustrated how the sequence network are interconnected for different types of faults and served as foundation for algorithms structures:

$$V_{a1} = E_a - I_{a1}Z_1 \tag{3.1}$$

$$V_{a2} = -I_{a2}Z_2 \tag{3.2}$$

After applying the Ohm Law , the voltage obtained was

$$V_{a0} = -I_{a0}Z_0 = I_{a0}(Z_{go} + 3Z_n) \tag{3.3}$$

From the matrices it changed to

$$V_{a0} = 0[as\ Z_0\ is\ Finite] \tag{3.4}$$

$$I_{a2} = 0 \tag{3.5}$$

$$E_a - I_{a1} \times Z_1 = 0 \tag{3.6}$$

$$I_{a1} = \frac{E_a}{Z_1} \tag{3.7}$$

3.2.5.1 Fault types

A fault is any abnormal condition in a power system. The power system operates in a steady state when the three AC phases are balanced. This condition can be subject of disruption due to changes in the system (Hung et al.2022). A short circuit, or fault, occurs when the insulation of the system breaks down at one or more points or when a conducting object comes into contact with a live point. The types of faults studied in this research include single line-to-ground faults, double line-to-ground faults, and line-to-line faults.

In the case of a single line-to-ground fault, we consider a generator where the fault occurs at one phase and is connected to ground through a fault impedance Z_f. This type of Line to ground fault is illustrated in Figure 3.2.

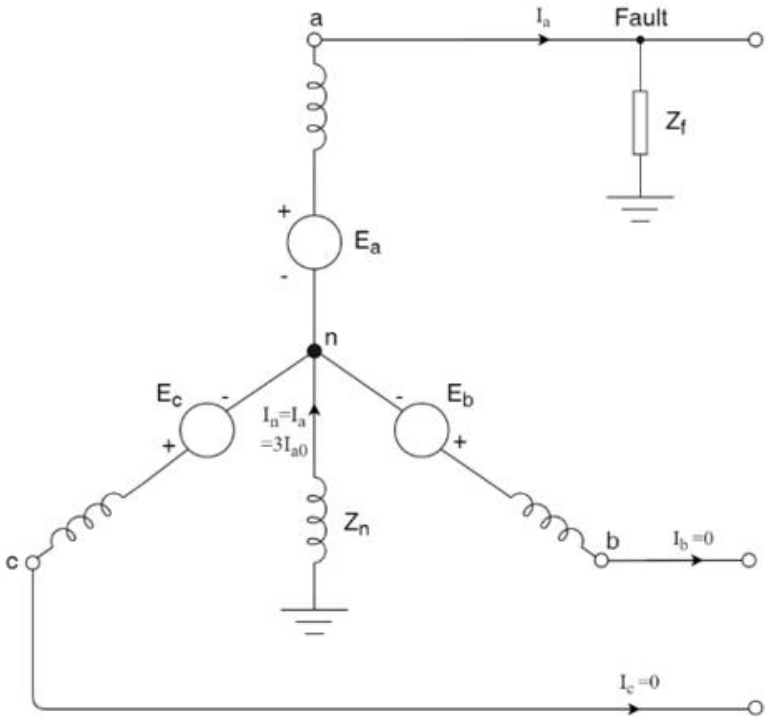

Figure 3.2: Line to ground fault (Singh et al.2022)

For this case, the following assumptions are made:

$$I_b = 0; I_c = 0 \; and \; V_a = I_a \times Z_f \tag{3.8}$$

Here the use the symmetrical components and represents the fault currents by substituting the value is

$$\begin{pmatrix} I_{a0} \\ I_{a1} \\ I_{a2} \end{pmatrix} = \frac{1}{3}\begin{pmatrix} 1 & 1 & 1 \\ 1 & a & a^2 \\ 1 & a^2 & a \end{pmatrix}\begin{pmatrix} I_a \\ 0 \\ 0 \end{pmatrix} \quad (3.9)$$

Then it gives

$$I_{a0} = I_{a1} = I_{a2} = \frac{1}{3} I_a \ldots (i) \quad (3.10)$$

$$V_a = I_a \times Z_f = 3I_{a1} \times Z_f \ldots (ii) \quad (3.11)$$

And the voltage phases are obtained with:

$$V_{a2} = -I_{a2} Z_2 \quad (3.12)$$

$$V_{a0} = -I_{a0} Z_0 \quad (3.13)$$

$$V_{a0} = E_a - I_{a1} \times Z_1 \quad (3.14)$$

$$V_{a0} = -I_{a1} Z_2 \quad (3.15)$$

The three-phase voltage is the sum of the different phases

$$V_a = V_{a0} + V_{a1} + V_{a2} = I_a \times Z_f \quad (3.16)$$

$$V_{a0} = -I_{a0} Z_0 \quad (3.17)$$

The analysis of such faults requires the use of symmetrical components. It's configuration can be adjusted to facilitate calculations. As viewed in Figure 3.3 Equivalent line to ground fault.

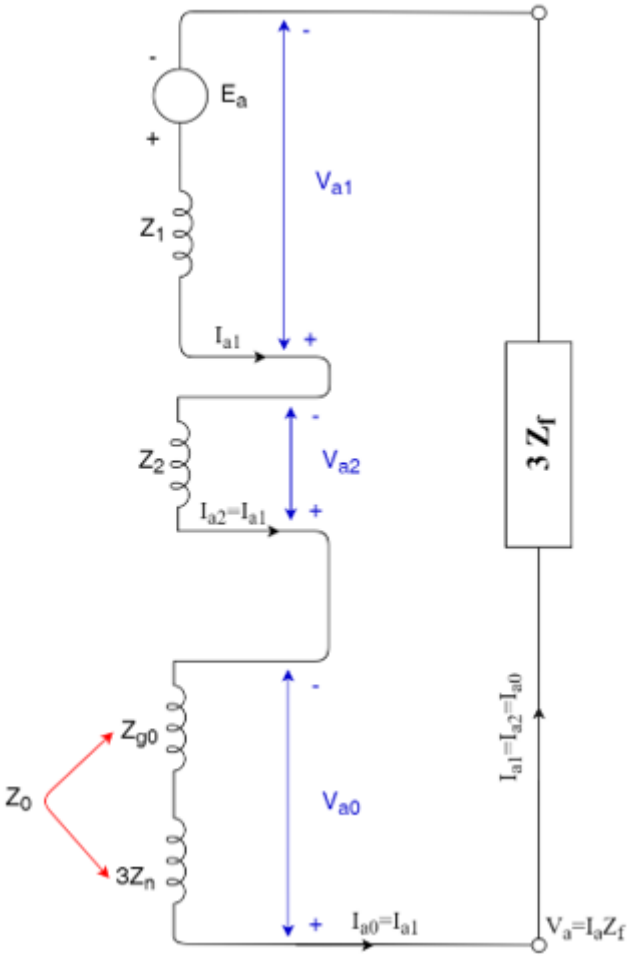

Figure 3.3: Equivalent Sequence network of the line to ground fault (Singh et al.2022)

After applying Kirchhoff current law, the following equation was found:

$$I_{a1} = \frac{E_a}{Z_1+Z_2+Z_0+3Z_f} \tag{3.19}$$

3.2.5.2. Lines to lines faults

A generator is considered where the line-to-lines unsymmetrical faults occur on phase a, b and c with a fault impedanceZ_f. TheLine-to-Line Fault is represented by Figure 3.4.

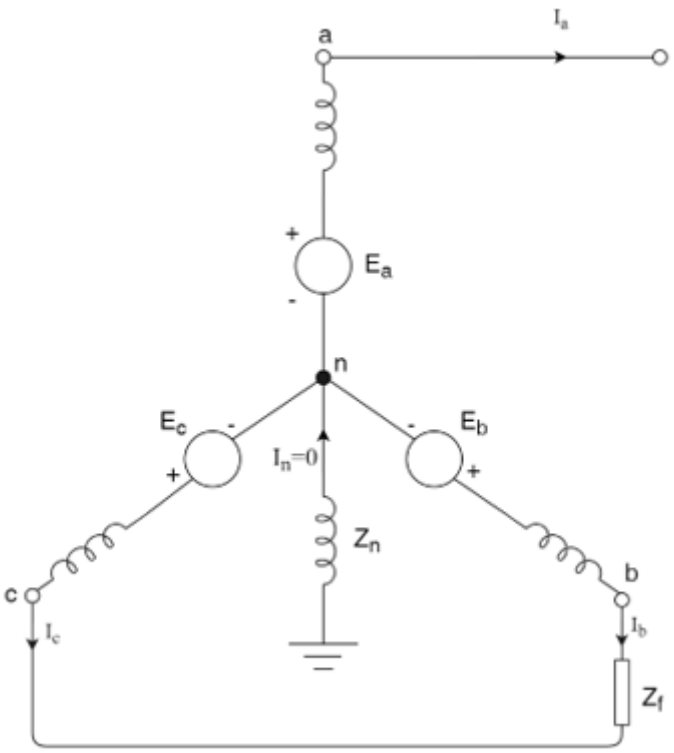

Figure 3.4: Line to Line Fault (Singh 2022)

Under the condition of line to lines faults ,the current from phase one is deducted as follow:

$$I_{a1} = \frac{E_a}{Z_1+Z_2+Z_0+3Z_f} \quad (3.20)$$

$$I_a = 0 \quad (3.21)$$

$$I_b = -I_c \quad (3.22)$$

$$V_b = V_c + Z_f \times I_b \quad (3.23)$$

$$I_b = -I_c = I_{a0} + a^2 I_{a1} + aI_{a2} \quad (3.24)$$

Under the condition of fault

$$I_a = 0 \quad (3.25)$$

$$V_b = V_c = (I_b + I_c)Z_f \quad (3.26)$$

The symmetrical components of the voltage are

$$\begin{pmatrix} I_{a0} \\ I_{a1} \\ I_{a2} \end{pmatrix} = \frac{1}{3}\begin{pmatrix} 1 & 1 & 1 \\ 1 & a & a^2 \\ 1 & a^2 & a \end{pmatrix}\begin{pmatrix} 0 \\ I_b \\ -I_b \end{pmatrix} \quad (3.27)$$

Then we have

$$I_{a1} = \frac{1}{3}[0 + a + a^2]I_b \tag{3.28}$$

$$I_{a2} = \frac{1}{3}[0 + a + a^2] \times (-I_b) \tag{3.29}$$

So, $I_{a1} = -I_{a2}$ and $I_{a0} = 0$ (3.30)

Similarly, the voltages in terms of symmetrical components are

$$\begin{pmatrix} V_{a0} \\ V_{a1} \\ V_{a2} \end{pmatrix} = \frac{1}{3}\begin{pmatrix} 1 & 1 & 1 \\ 1 & a & a^2 \\ 1 & a^2 & a \end{pmatrix}\begin{pmatrix} V_a \\ V_b \\ V_b - I_b \times Z_f \end{pmatrix} \tag{3.33}$$

The voltage can be extracted as follow

$$V_{a1} = \frac{1}{3}\left[V - a + aV_b + a^2\left(V_b - I_b \times Z_f\right)\right] \tag{3.34}$$

$$= \frac{1}{3}\left[V + (a + a^2)V_b + a^2\left(V_b - I_b \times Z_f\right)\right] \tag{3.34}$$

$$= \frac{1}{3}\left[V + (a + a^2)V_b + a^2\left(V_b - I_b \times Z_f\right)\right] \tag{3.35}$$

$$= \frac{1}{3}\left[V + (a + a^2)V_b + aI_b \times Z_f\right] \tag{3.36}$$

$$3(V_{a1} - V_{a2}) = \left(a - a^2 I_b \times Z_f\right) = j\sqrt{3}I_b \times Z_f \tag{3.37}$$

The equations become

$$V_{a1} - V_{a2} = j\frac{\sqrt{3}}{3} \times \left(-j\sqrt{3}I_{a1}Z_f\right) = I_{a1} \times Z_f \tag{3.38}$$

In this case, due to the absence of ground connection , the zero sequence network remained unused, only the positive and negative sequence were active and carry currents in opposite directions. Figure 3.5 illustrated the sequence network of line-to-line fault.

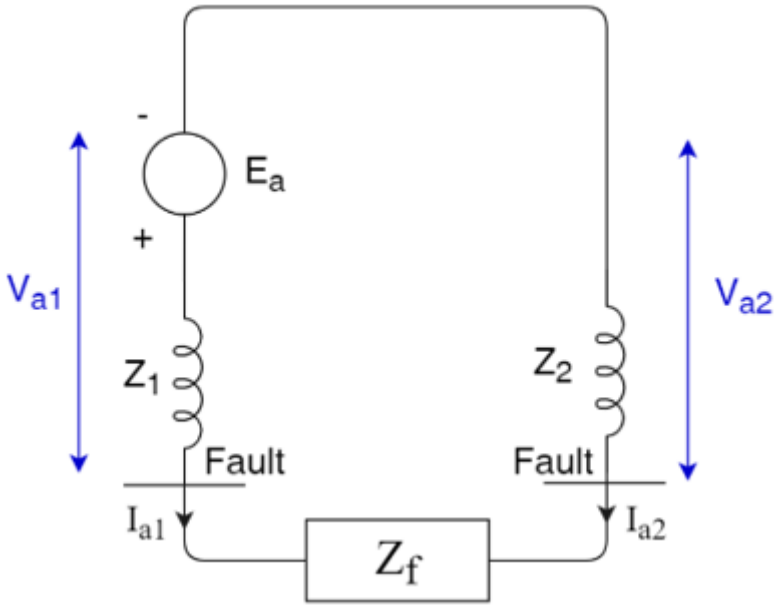

Figure 3.5: Sequence network of line-to-line fault (Singh et al. 2022).

The following equation was obtained by applying Kirchoff laws on the phase one and it was explicated as follow:

$$I_{a1} = \frac{E_a}{Z_1 + Z_2 + Z_f} \tag{3.39}$$

3.2.5.3 Double line to ground faults

In this instance, it considers a generator. Between points a, b, and c, there are double line to ground faults. Figure 3.6 shows the Double Line to Ground Fault Sequence Network.

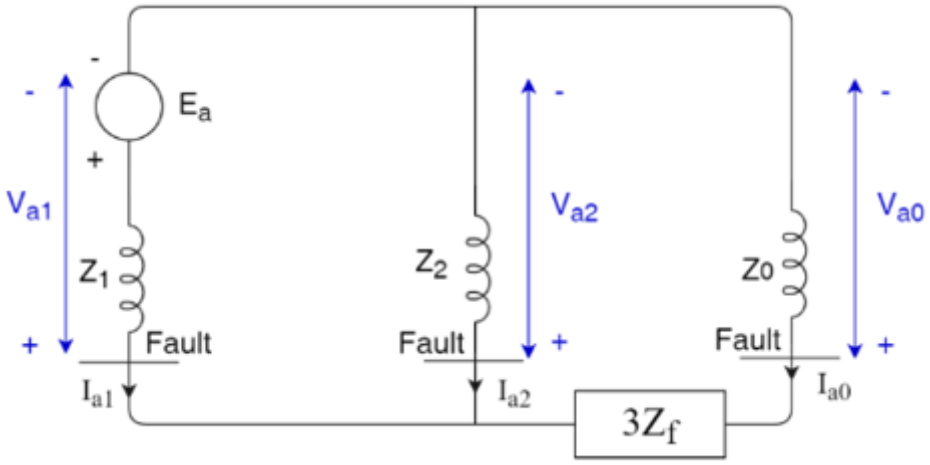

Figure 3.6: Sequence Network of Double Line to Ground Fault (Singh et al., 2022).

It happens when the active conductor of a transmission line comes accidentally in contact with each other. Figure 3.7 represents the termination of a three-phase access port with double-line faults. Then we obtain the following equations:

$$I_a = 0 \text{ and } I_f = I_b + I_c \tag{3.40}$$

$$V_b = V_c = I_f \times Z_f \tag{3.41}$$

$$= (I_b + I_c) \times Z_f = 3I_0 \times Z_f \tag{3.42}$$

$$\begin{pmatrix} V_{a0} \\ V_{a1} \\ V_{a2} \end{pmatrix} = \frac{1}{3} \begin{pmatrix} 1 & 1 & 1 \\ 1 & a & a^2 \\ 1 & a^2 & a \end{pmatrix} \begin{pmatrix} V_a \\ V_b \\ V_c \end{pmatrix} \tag{3.43}$$

From the above equation

$$V_{a1} = V_{a2} = \frac{1}{3}[V_b + (a + a^2)V_b] \tag{3.44}$$

$$\{as, a + a^2 = 1\} = \frac{V_a - V_b}{3} \tag{3.45}$$

$$\text{And } V_{a0} = \frac{V_a + 2V_b}{3} \tag{3.46}$$

Now subtracting $V_{a1}\ and\ V_{a0}$

$$V_{a0} - V_{a1} = \frac{1}{3}[2V_b + V_b] = V_b = 3I_0 \times Z_f \tag{3.47}$$

The sequence analysis through symmetrical components facilitated the fault calculation , from one phase to another. Figure 3.7 showed the double line to ground fault

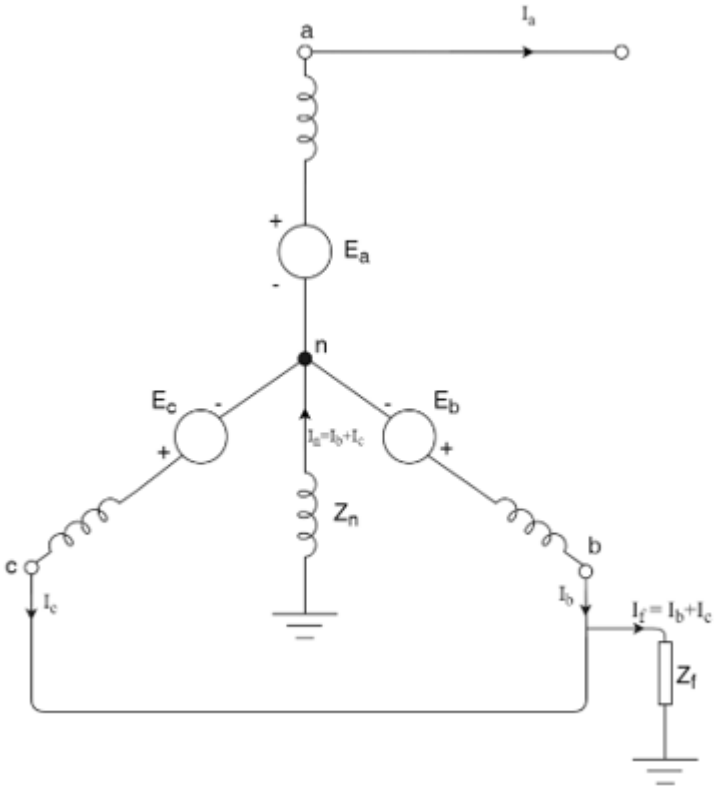

Figure 3.7: Double line to Ground voltage (Singh 2022)

In the terms of Thevenin equivalent Circuits, the currents are:

$$I_{a1} = \frac{E_a}{Z_1 + Z_2 \dfrac{Z_0 + 3Z_f}{Z_2 + Z_0 + 3Z_f}} \tag{3.44}$$

In the case of direct short circuits from all the above faults cases, take $Z_f \; as \; 0$.

3.2.5.4 Transmission line fault analysis

Transmission line fault analysis is essential for ensuring the stability and safety of power systems. Fault locators play a crucial role in identifying the location of faults quickly and accurately. These devices often use data from standard recording equipment such as digital relays and Intelligent Electronic Devices, integrated with traditional protection schemes. Typically, bus phase voltages, line voltages, and current data are utilized, and sometimes information from circuit breakers is also incorporated (Thomas, et al., 2020).

Although both protective relays and fault locators are used to safeguard transmission lines, they operate in distinct ways:

Accuracy: Protective relays are generally designed to identify a broader zone where the fault may have occurred. In contrast, fault locators employ advanced algorithms

that can pinpoint the exact fault location with minimal error margins (Chen, et al., 2021).

Speed: Conventional relays must operate at high speed to react promptly to fast-spreading faults in the grid. However, using high-speed communication systems and breakers may compromise the relay's security and selectivity. Fault locators, on the other hand, can quickly identify fault locations using algorithm-based analysis, offering faster results without compromising accuracy.

Data Control: Relays manage fault data using a fault interval, which defines the size of the data window from the fault's occurrence to its clearance by circuit breakers. This process often spans multiple fundamental frequency cycles. Fault locators analyze this same data to calculate the fault position more precisely.

Complexity: Both relays and fault locators can be straightforward in terms of their fundamental calculations. However, fault locators can become more advanced when enhanced with artificial intelligence tools such as neural networks and fuzzy logic, offering more refined and adaptive fault detection capabilities.

This analytical comparison highlights the complementary roles of relays and fault locators in modern transmission line protection systems, with fault locators offering improved precision and adaptability for post-fault analysis.

3.2.6 High Impedance Faults (HIFs)

HIFoccurs when an electrical contact is made between a naked conductor (carrying current) and an insulated object, i.e. when a conductor (carrying current) breaks and comes into physical contact with a high-impedance surface (Kim et al. 2019). Such a surface could be grass, sand, or a road, and it could pose a risk to humans and their surroundings. Alternatively, the conductor (with current flowing) may not break but come into contact with an insulated, grounded object.

3.2.7 Low Impedance Faults (LIFs)

LIFs refer to conventional short-circuit or shunt faults in power systems. These faults are characterized by low resistance paths that allow high fault currents to flow. Common types of LIFs include:

1. Single Line-to-Ground (L-G) Fault
2. Line-to-Line (L-L) Fault
3. Double Line-to-Ground (L-L-G) Fault
4. Three-Phase Fault
5. Balanced Three-Phase-to-Ground Fault

These faults are typically easier to detect and isolate due to the high current levels involved, and they form the basis for most traditional fault analysis and protection schemes.

3.3 Causes of faults

The factors responsible for interruptions of power supply to customers are classified into different categories, namely, natural factors, human factors and equipment factors.

3.3.1 Natural factors

Natural factors include events such as harsh weather, vegetation, and animals coming into physical contact with current-carrying conductors. These are known as natural disasters and are beyond human control.

3.3.2 Human Factors

Theft and vandalism are examples of human factors. It is practically impossible for utilities to prohibit humans from interfering with all distribution system apparatuses. The vandals' primary targets are copper, aluminium, and transformer oil(Chen et al.2020). Theft and vandalism are largely driven by increased global market demand for these products, which has led to increased markets for aluminium and copper in the international metals market, fuelled by the industrialization of China and India.

3.4 Fault type classification and detection

Researchers have focused on developing reliable and accurate fault classification algorithms for power distribution systems, as fault-type classification is crucial for protective relays. Most methods use classifier models based on statistical learning theory, while others rely on logic flows derived from practical knowledge and observed data. The growth in this research has significantly impacted pattern recognition and machine learning, particularly supervised learning algorithms. Fault classification techniques can be categorized into three types: conventional, intelligent and hybrids methods.

3.4.1 Conventional methods (Impedance-based methods)

The impedance-based fault detection method identifies faults by measuring current and voltage before and after a disturbance at one or both feeder terminals. These measurements are then used in mathematical calculations to determine the line's characteristics and locate the fault. The fault distance is determined with the following equations:

$$V_s = (x.Z_l.I_s) + R_F I_F \quad (3.45)$$

$$I_M\left(\frac{V_s}{I_s}\right) = I_m \times (x.Z_l) = x.x_l \tag{3.46}$$

$$x = \frac{I_M\left(\frac{V_s}{I_s}\right)}{x_l} \tag{3.47}$$

where x stands for the distance to the fault position, I_s and V_s for the line's producing terminal's current and voltage and for the current difference between the pre- and post-fault states (also known as the fault component current), and I_F for fault current. Power line reactance is represented by x_l and imaginary component byI_M.

By using both pre-fault and fault data, the author (Jeh, et al., 2019) presented an enhancement to the reactance method for fault location in a single-end approach. This method improves upon the simple reactance method by reducing the impact of load flow and fault resistance. The update is represented as follows:

$$x = \frac{I_m(V_s.I_s^*)}{I_m(Z_l.I_s.I_s^*)} \tag{3.48}$$

where, V_s and I_s stand for the voltage and current of the sending terminal, respectively, where x is the distance from the fault spot. Z_l stands for the electric network's impedance per unit length, I_m for the imaginary component, and * for the conjugate component.

The modified Takagi approach is an additional impedance-based methodology that does not require pre-fault information because it employs zero sequence current.

Girgis published equations for various fault types occurring at main feeders and single-phase lateral lines in distribution networks (Muhammad et al. 2022). The issue of various fault locations obtained from mathematical calculations was resolved by updating the voltage and current vectors using the model of static impedance load. The Discrete Fourier Transform (DFT) was used to filter harmonics.

3.4.2 Hybrid methods (High-Frequency Components and Travelling Wave Methods)

The study of higher frequency signals outside the fundamental frequency, such as harmonic, travelling wave, and natural oscillation frequency, is referred to as high-frequency components and travelling wave schemes (Cai et al. 2022). Due to their modest magnitudes and overwhelming background noise, these may be challenging to distinguish from the fault composite signal. The program uses either the traveling wave method or the analysis of naturally occurring oscillating frequencies and harmonics. To distinguish fault surge characteristics from those specific to capacitor switching, load switching, and arc-producing devices, the harmonic analysis method looks at the higher harmonic currents (Rezaei et al. 2022). These frequency features

propagate in both directions from the faults, they encounter disturbances as they travel along the feeder, and as a result, they are reflected back to the fault area and cluster at the relay terminal to generate a signal that is highly interconnected. The initial levels of the waves depend on the frequency, location on the feeder, and route resistance of the fault. The fault distance (x) based on the travelling wave technique for a double ended line is described with the following equations:

$$x = \frac{l + k_C(t_A - t_B)}{2} \tag{3.49}$$

The length of the cable is l while the propagation speed of the travelling wave is kc; the difference between the arrival times of the travelling waves of each side is $t_A - t_B$. A Short-Time Fourier Transform (STFT) technique is used to analyse the phase content and frequency of local signal sections by dividing the signal, selecting time-constrained windows, and analysing each section (Vadivel et al. 2021). However, STFT uses fixed-size windows, which can be limiting. Wavelet analysis addresses this by using flexible-sized windows, offering better accuracy for low-frequency signals. Wavelet Transform (WT) divides a signal into frequency bands for temporal analysis, creating wavelets from a "mother wavelet" function (Tariq et al. 2022). Wavelet Transform has two types: Discrete Wavelet Transform (DWT) and Continuous Wavelet Transform (CWT), with CWT involving the summation of a signal over time and shifting versions of the wavelet function. The CWT signal is given by

$$\text{CWT}(a,b) = \int_{-\infty}^{\infty} x(t)_{\varphi_{a,b}}(t)dt \tag{3.50}$$

The CWT provides a time-frequency representation of a signal, offering better time and frequency localization compared to the Fourier Transform making it ideal for analysing non-stationary signals (Kavitha et al., 2021). The DWT analyzes signals at different resolutions and frequency bands by decomposing the signal into coarse approximations and detail coefficients using wavelet functions and scaling. DWT varies from WT in that its analysis scale changes by a factor of two:

$$\text{DWT}\ (m,n) = \frac{1}{\sqrt{2}}\sum_k f(k)\,\varphi\left(\frac{n-k2^m}{2^m}\right) \tag{3.51}$$

The signal analysis involves the use of DWT and multi-resolution analysis for fault detection and identification in power systems. DWT decomposes signals into coefficients, with scale and shift variables, to analyse high-frequency components and detect HIFs. Methods like directional wave, correlation, and wavelet tests were evaluated for MV distribution systems, with techniques implemented in MATLAB for

fault detection in unbalanced, noisy networks. Simulations using ATP-EMTP software were applied to analyse faults in underground systems, where wavelet transforms helped in recognizing fault signals and fault locations by analysing high-frequency and transient signals. A defect localization approach based on high-frequency signals was suggested (Vanam et al., 2022).

The plan relied on the WT's unique characteristics to distinguish between faults occurring in different parts of the main feeder that were equally far from the main substation. Various methods have been proposed for detecting HIFs and classifying faults in distribution systems using WT and multi-resolution analysis. These methods involve capturing voltage and current features in high-frequency mode, analysing peak separations, and using moving window strategies for continuous monitoring. MATLAB models were developed for fault events like capacitor switching and HIFs. Techniques for fault localization, detection, and forecasting cable life were explored, with some approaches utilizing ANN for pattern identification. Other methods used cross-correlation for fault distance computation, and WT was employed to identify faults like single-line-to-ground faults in primary distribution systems.

The fault distance from the transmitting terminal was calculated using the initial peak time of the bus at fault, and the technique was validated with various fault types, locations, and inception angles. Fault signals were simulated at a 200 kHz sample rate using ATP/EMTP. In another study, an algorithm was developed for identifying broken feeders using both transient and steady-state signals, with wavelet packet analysis used to extract fundamental and high-frequency components to identify the faulty feeder in a grounded Peterson-coil setup.

3.4.3 Artificial Neural Network with intelligent methods

ANN are inspired by the human brain's data processing and consist of basic neurons arranged in layers. The inputs to the ANN are linked to neurons in previous layers, and each neuron processes inputs non-linearly to produce an output. Training an ANN involves adjusting the weights attached to the neurons based on the training data (Sarangi et al., 2021). (Abdullah et al., 2022). Figure 3.8 shows the architecture of a feed forward ANN.

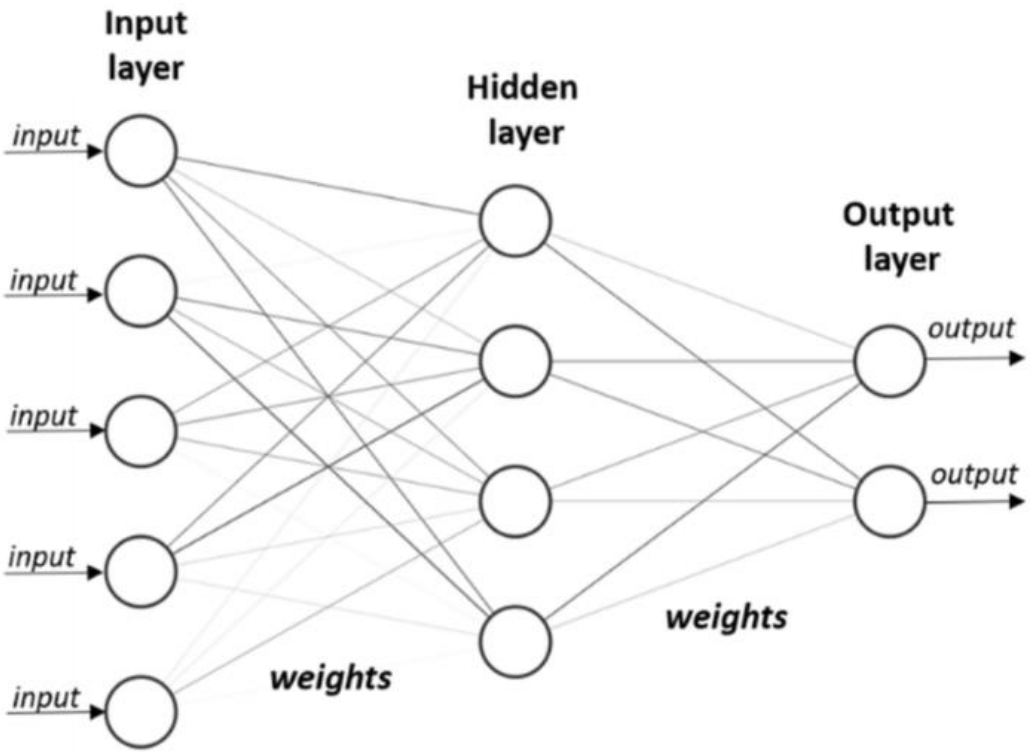

Figure 3.8: The architecture of a feed forward ANN (Manmai et al. 2021).

ANNs learn by adjusting node weights in response to inputs, using a training data set. Training involves fine-tuning these weights to match the training set(Biswal et al., 2022). The back-error-propagation algorithm (BPA), a supervised learning technique, reduces errors by gradually adjusting the weights to minimize the difference between actual and desired outputs. A basic neuron model computes outputs based on inputs. (Rahman et al. 2021),(Figure 3.9 shows the model of a neuron).

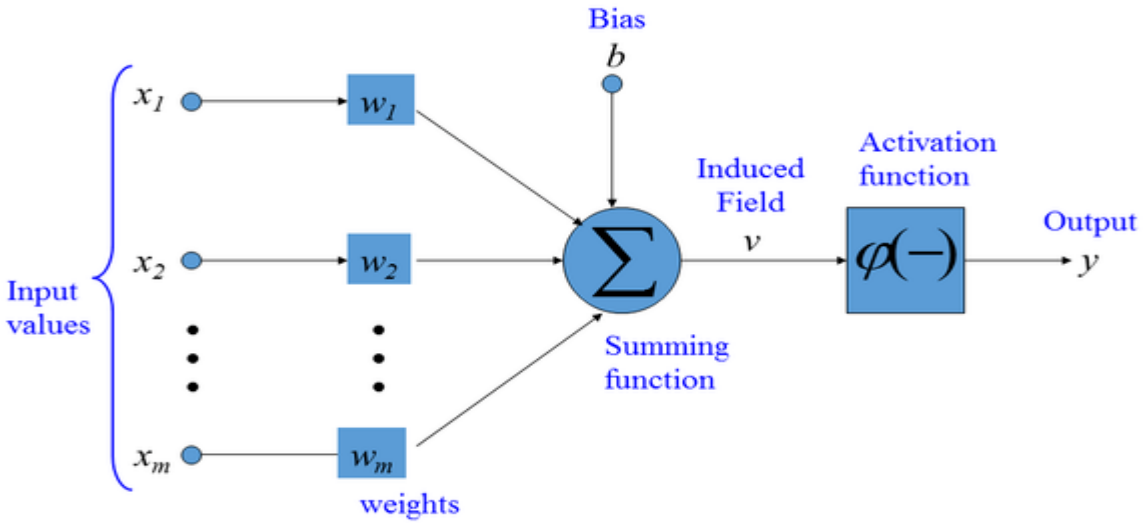

Figure 3.9: Model of a neuron

The output of the neuron is expressed as:

$$y = f(\varphi) = f\left(\sum_{i=0}^{No} w_i\, a_i\right) \tag{3.52}$$

where, $w_i a_i$ represents the threshold value (polarization), $f(\varphi)$ stands for the neuron activation function, φ depicts the summation output signal and y is the neuron output

$$\varphi = W^T.A \tag{3.53}$$

where

$$W = [w_0 w_1 \ldots . w_k] \quad A = [a_0 a_{1 \ldots} a_{No}] \tag{3.54}$$

Based on the needs of the application, f(), the suitable activation function is chosen. The f() represents the strength of the neuron's output based on the total of its inputs(Jagadanand et al. 2015). The f() can have many different shapes, some of which are as follows:

The step activation function is expressed as

$$f(\varphi) = \begin{cases} 1\ if\ \varphi \geq 0 \\ 0\ if\ \varphi < 0 \end{cases} \tag{3.55}$$

The Piecewise Linear Activation Function is defined as

$$f(\varphi) = \begin{cases} 1\ if\ \varphi > 1 \\ -1\ if\ \varphi < -1 \\ if\ |\varphi| < -1 \end{cases} \tag{3.56}$$

Sigmoid Unipolar Activation Function is described as follows

$$f(\varphi) = \frac{1}{1 - e^{-\beta\varphi}} \tag{3.57}$$

Sigmoid Bipolar Activation Function equation is

$$f(\varphi) = \tan(\beta\varphi) = \frac{1 - e^{-\beta\varphi}}{1 + e^{-\beta\varphi}} \tag{3.58}$$

Regarding how a model's neurons are connected, it may generally be divided into two categories: feed forward neural networks and feedback neural networks, often known as recurrent neural networks(RNN) (Mohammadi et al. 2021).

3.4.4 Feed-forward Neural Network

Looping nodes are absent from feed-forward networks, which are artificial neural networks. This type of neural network is also known as a multi-layer neural network since all input is only transmitted forward.

In a feed-forward neural network, data flows from input nodes through hidden layers to output nodes, with no feedback connections that allow data to return from the output. The network approximates functions by processing data in this one-way flow. $y = f(x)$ is used the algorithm for the classifier calculations

3.4.5 Working principle

The feed-forward neural network might resemble a single layer perceptron when it is simplified as presented in Figure 3.10.

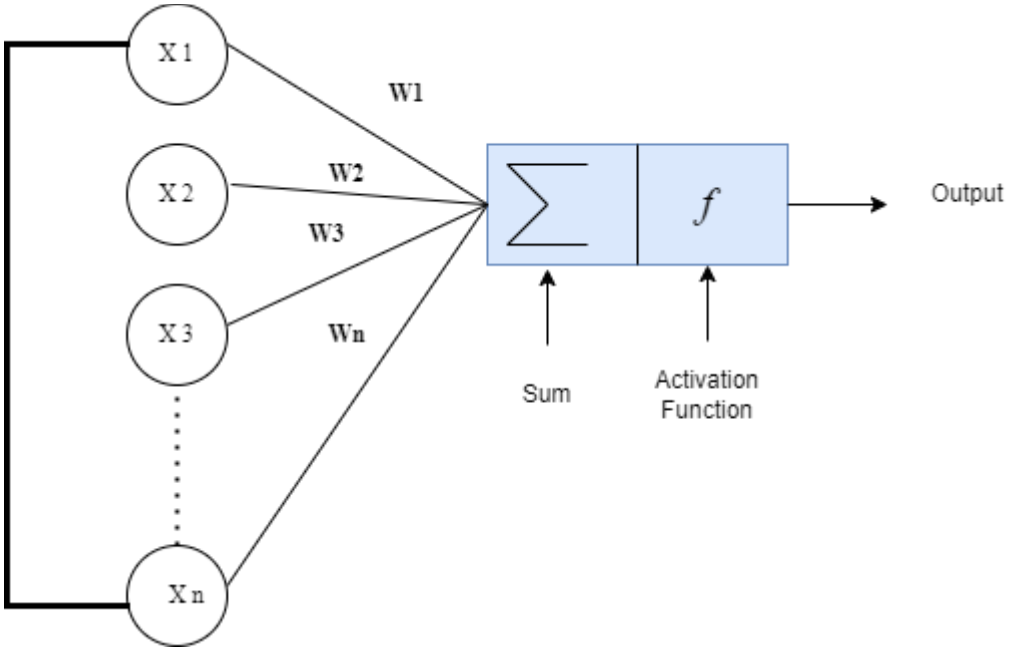

Figure 3.10: Model of a neural network.

In this model, inputs are passed through a layer where they are multiplied by corresponding weights. The resulting weighted values are then summed up. The output is typically 1 if the sum exceeds a set threshold, usually zero, and -1 if it falls below that threshold. A common feed-forward neural network used for classification is the single-layer perceptron, which can also integrate machine learning. Neural networks are trained to adjust their weights using a method known as the delta rule, allowing them to compare their outputs with the desired results. Training leads to gradient descent, and perceptrons with multiple layers adjust their weights through a

process called back-propagation. In this case, the weights in the network's hidden layers are updated based on the output values from the final layer.

3.4.6 Layers of the feed-forward neural network

It is made of different elements such as**:** Input, Output and hidden layer, Rectified linear unit, Neurons, Sigmoid and activation function.

3.4.7 Input layer

This layer's neurons take in information and transmit it to the other layers of the network. The dataset's feature or attribute counts must correspond to the number of neurons in the input layer.

3.4.8 Output layer

According to the type of model being built, this layer represents the forecasted feature.

3.4.9 Hidden layer

Hidden layers serve to separate the input and output layers. The number of hidden layers can vary depending on the type of model. In these layers, multiple neurons modify the input before it is passed on to the next layer. Weights in this network are continuously adjusted to improve the accuracy of predictions.(Hamdani et al.2021).

3.4.10 Activation function

Activation functions in neural networks introduce non-linearity, allowing the model to learn and model complex patterns. When neurons receive inputs, they compute a weighted sum and apply an activation function to decide whether to activate. This process transforms linear combinations of inputs into non-linear outputs, capturing intricate relationships (Rabbi et al. 2023). Several types of activation functions are available, each with unique characteristics suited to different tasks. Sigmoid and tanh functions map inputs to values between 0 and 1 or -1 and 1, respectively. The ReLU function outputs zero for negative values and input values for positives, allowing faster training and mitigating vanishing gradient problems. The choice of activation function significantly impacts the model's performance.

3.5. Multilayer Perceptrons

Multilayer Perceptrons (MLPs), a particular family of layered feed-forward networks, have come to be designed using the back-propagation algorithm. A multilayer perceptron model consists of two main layers: an input layer with source nodes and an output layer with computation neurons. It typically includes one or more hidden layers, where hidden neurons extract important features from the input data (X. Liu et al. 2023). Figure 3.11 shows the Model of multiple-layer perceptrons**.**

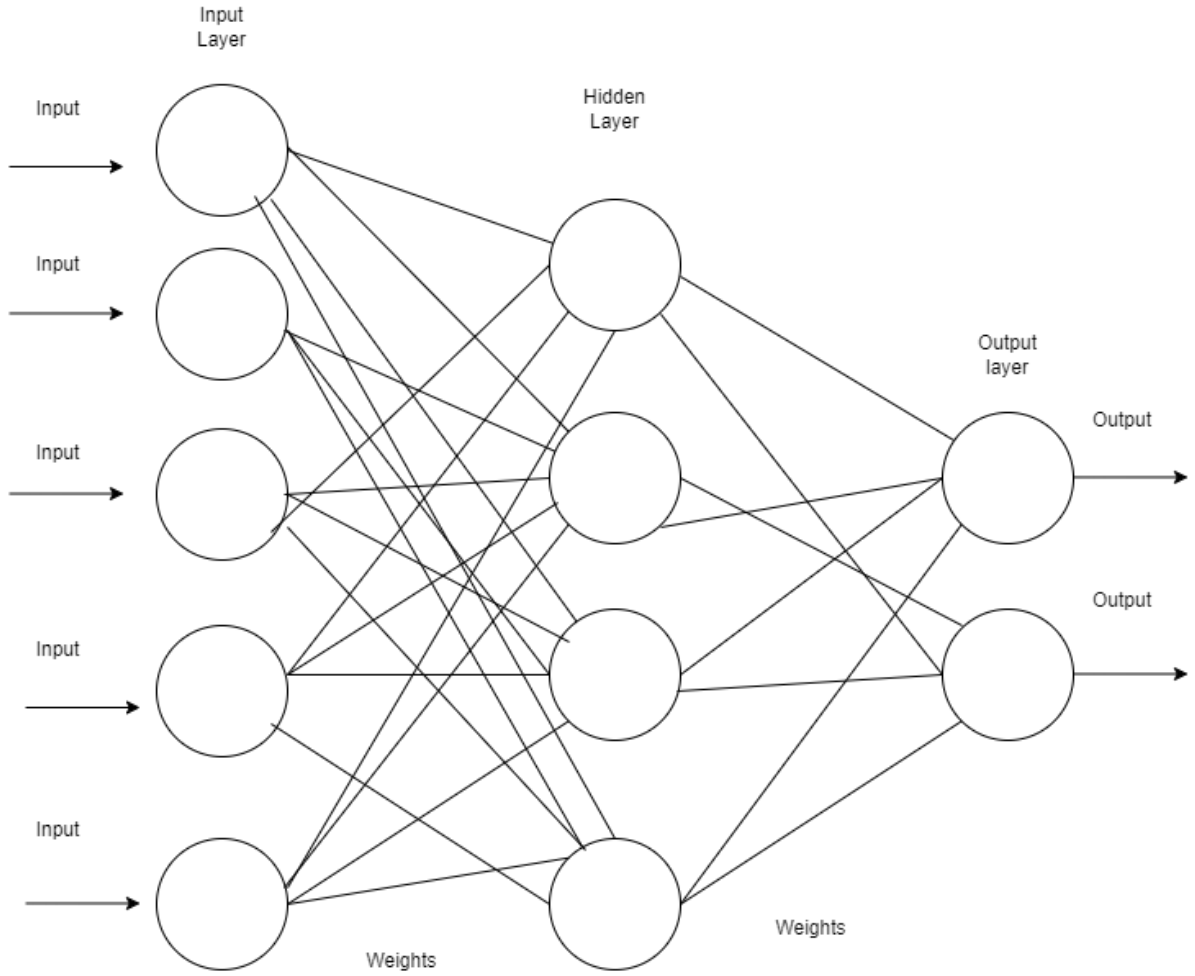

Figure 3.11: Model Multiple layer perceptrons

3.6 Convolutional neural network (CNN)

As the filter passes over the input, it determines the average value in the receptive field and sends it to the output array. The convolutional layer serves as the starting point in a convolutional neural network. While additional convolutional or pooling layers can come afterwards, the network typically ends with a fully connected layer. As the data moves through each layer, the CNN becomes increasingly sophisticated, recognizing broader areas of the image (Babu et al., 2022). Early layers focus on detecting simple elements like colors and edges, while deeper layers progressively identify larger patterns and eventually the target object.

3.6.1 Convolutional layer

The convolutional layer is the core element of a CNN and is responsible for most of its computations. It operates using input data, filters, and a resulting feature map. When dealing with a color image as input, the data is typically represented as a 3D matrix of pixels (Hu et al., 2022), consisting of height, width, and depth—similar to the channels

in an image. The feature detector, also known as a kernel or filter, moves across the image's receptive fields to detect the presence of specific features.

This filter is essentially a 2D array of weights that scans parts of the image. Although filter sizes can differ, a common choice is a 3x3 matrix, which determines the receptive field's dimensions. As the filter slides over the image, it performs a dot product between the filter values and the corresponding input pixels. The result is stored in an output array. This operation is repeated across the entire image as the filter shifts according to a defined stride. The collection of these outputs forms what is known as the feature map, activation map, or convolved feature (Madhubabu et al., 2021).

During training, parameters like filter weights are updated using techniques such as back propagation and gradient descent. However, before training begins, three key hyper parameters must be set, as they significantly influence the size of the output volume. The number of filters affects the output depth, with each filter producing a separate feature map. Stride refers to the distance the kernel moves across the input matrix, and a longer stride results in a smaller output. Zero padding is used when filters don't fit the input image, filling non-input areas with zeros to maintain or increase the output size.

3.6.1.1 Pooling layer

Down sampling, also known as pooling, is used to reduce the dimensions of the input and decrease the number of parameters in the network. Similar to convolutional and pooling layers slide a filter over the input data; however, unlike convolutional filters, pooling filters do not have weights. Instead, they apply an aggregation function, such as taking the maximum or average, within each receptive field to generate values for the output array. The two primary forms of pooling are:

i. Max pooling as it moves over the input, the filter selects the input pixel with the greatest value to deliver to the output array. In addition, this approach is used more often than typical pooling.
ii. Average pooling as the filter passes over the input, it determines the average value in the receptive field and sends it to the output array.
iii. Fully connected layer, the fully connected layer functions just as its name suggests. Unlike partially connected layers, where input pixel values are not directly linked to the output layer, the fully connected layer ensures that each node in the output layer is directly connected to every node in the preceding layer.(Lozanov et al., 2023). Figure 3.12 shows the architecture of a sequential CNN.

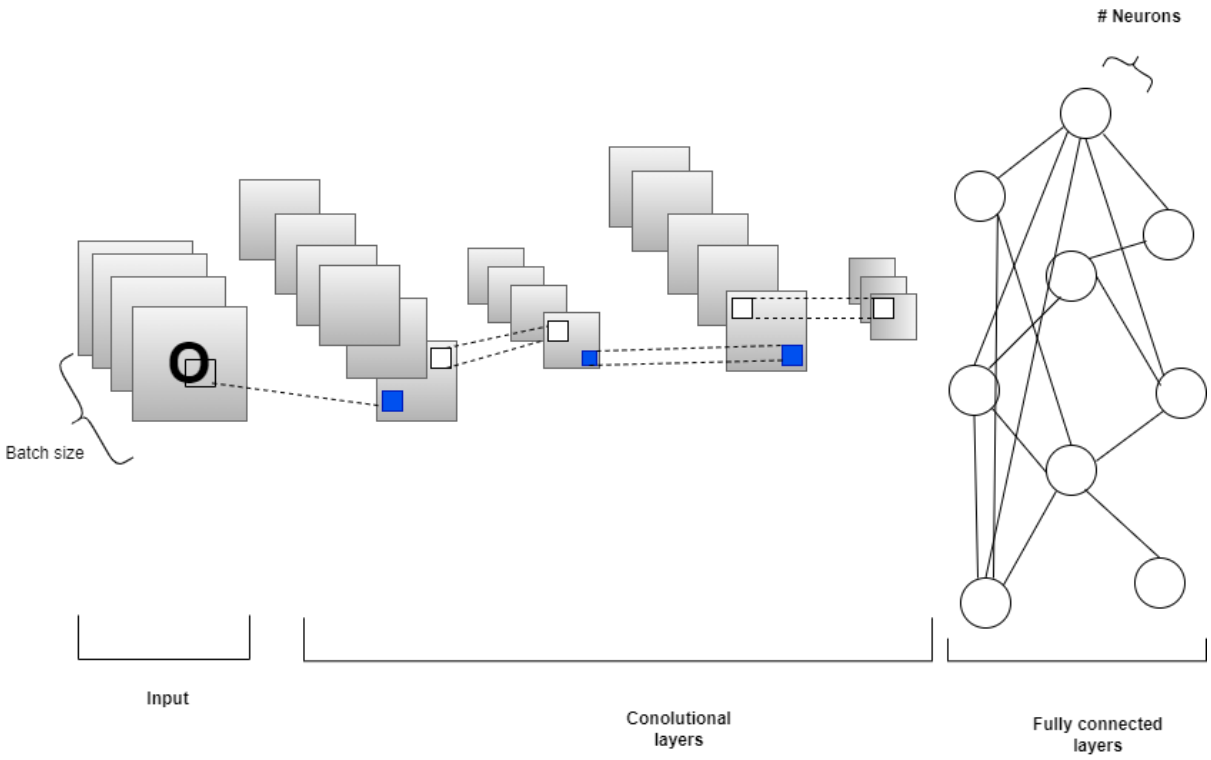

Figure 3.12: The architecture of a sequential CNN (Muralidhar and Rajasekar 2021).

3.6.2 Recurrent Neural Network

RNNs extend feed-forward networks by adding recurrent edges, giving them a sense of time. These connections allow inputs from previous time steps to influence current outputs. Weight matrices along with biases, guide learning in hidden layers. Models like Jordan and Elman networks enable memory through context units. RNNs are trained using back propagation through time , but face challenges like vanishing and exploding gradients, especially over long sequences. Techniques like truncated BPTT and regularization help mitigate these issues. Before LSTM's introduction in 1996, training RNNs for long-term dependencies often failed to yield meaningful results. RNNs have become common due to successful training methods. Sutskever and Martens applied Hessian-Free, or truncated Newton, training to an RNN that generates text one character at a time. This method avoids saddle points, unlike Newton's method, and has shown improved performance on recurrent networks. However, Newton's method requires calculating the Hessian, which is computationally expensive for large networks (M. and A. 2021). Table 3.2 represents the different uses of Deep Recurrent Neural Networks.

Table 3.2: Different use of Deep Recurrent Neural Networks.

Date	References	Description
2019	(Chen et al., 2019)	This paper discussed using Deep Recurrent Neural Networks (DRNNs) to predict user behaviour in Tor networks. A Tor server and client (Deep Web browser) were set up, with the client sending browsing data through the Tor network. The data was collected using Wireshark, and DRNNs were applied to predict outcomes, showing the model's effectiveness in predicting user behaviour in Tor networks.
2023	(Minic et al.,2023)	In this article, a new approach using neural networks to identify sound events in a fog computing environment is proposed. Applications developed for smartphones and PCs are presented, and experimental results confirm the effectiveness of the suggested method.

3.6.3 Auto encoder and generational encoders

In an encoder-decoder model, the encoder transforms the input sequence into a hidden vector, and the decoder generates the output sequence from this vector. The model is jointly trained to maximize the conditional probabilities of the target sequence given the input. Figure 3.13 presents a general encoder.

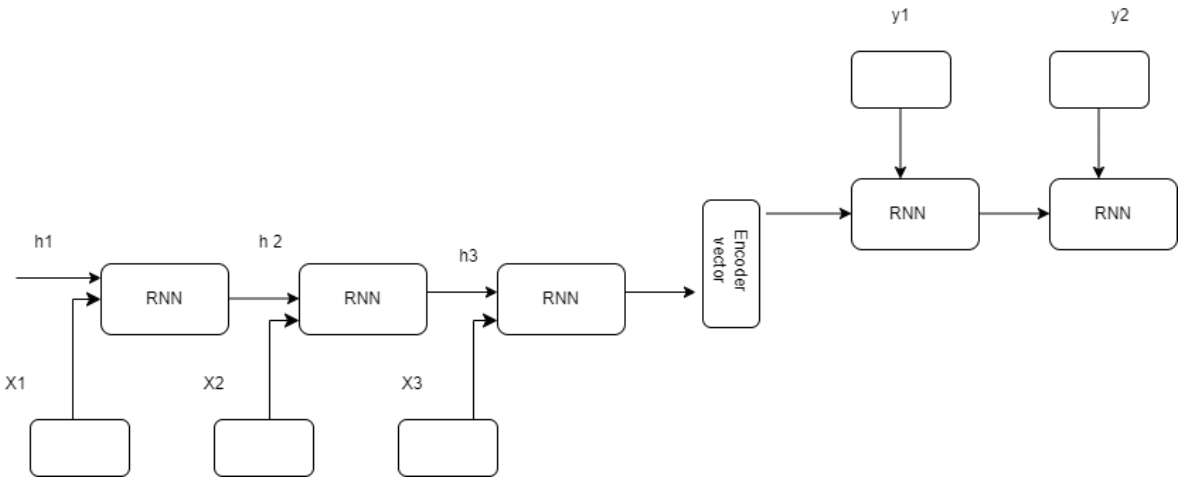

Figure 3.13: General Encoder.

Autoencoders are neural networks used for unsupervised learning tasks such as dimensionality reduction, data compression, and anomaly detection. They consist of an encoder that compresses input data and a decoder that reconstructs it, aiming to minimize the difference between the original and reconstructed data. Auto encoders are useful for data denoising and feature extraction and can effectively compress complex visual data. Variants like convolutional and recurrent auto encoders are tailored for specific data types, while generative variations can create new data samples similar to the input. c Regarding the given example, this occurs for all four stages. a stack of multiple recurrent units, each of which takes one element from the input sequence, gathers data for that element and propagates it forward (LSTM or GRU cells for better performance). All of the words from the question make up the input sequence in the question-answering problem. Every word is represented by the symbol x, i where i is the word's order.

3.6.4 Encoder vector

An encoder vector is a compressed representation in neural networks, used in models like auto encoders, sequence-to-sequence models, and transformer architectures. It captures essential features of input data in a reduced-dimensional form, known as the latent space or bottleneck layer. This helps distill complex data into a more manageable format, removing redundancies and focusing on core information for tasks like decoding or classification (Tangirala et al., 2021). In language translation, the encoder converts an input sentence into an encoder vector, which is then used by the decoder to generate the translated sentence.

3.6.5 Decoder

A decoder is a crucial component in neural networks that reconstructs or generates output data from a compressed representation, often called the latent vector or encoder vector. They are used in models like auto encoders, sequence-to-sequence networks, and transformers to transform the simplified representation back into a form that resembles the original input and fulfil a specific task. In natural language processing (NLP), decoders play a vital role in tasks like machine translation and text generation. The decoder's structure and effectiveness are critical, as it must interpret and expand on abstracted data while maintaining fidelity to the intended output structure and details.

The output sequence in the question-answering problem is an assemblage of every word from the solution. It is applied using:

$$h_t = f\left(W^{(hh)} h_{t-1}\right) \tag{3.60}$$

With as h_t output sequence and W weight matrice.

The output layer in a neural network produces predictions or classifications based on data processed by previous layers. Its structure depends on the task, with classification networks having one neuron per class and regression tasks having a single neuron. The output layer translates learned information into a usable result, calculates the network's loss during training, and adjusts its weights through back propagation to make accurate predictions based on data (Minic et al., 2023).

3.6.6 Deep Reinforcement Learning

The Dual Q-Network estimates Q-values using a Dual Neural Network based on the Bellman equation and principles of Q-learning. It offers advantages over traditional Q-networks, such as improved performance and decision-making. DQN can use screen images to learn how to play video games effectively and maximize scores. By gathering data on "what screen," "with which operation," "how the score will change," and "what will be the next screen," you can learn from this. Using this analogy, an algorithm is created. You will obtain reward and the subsequent state for any pair that has the action and the current state (Tang et al., 2021).

A model can be trained to its highest potential if a large enough dataset is available. However, it's not feasible to gather data for every possible state-action pair (s, a). The success of the Deep Q-Network largely comes from two key features: experience replay and the target network. Experience replay stores past state transitions, actions, and rewards, which are crucial for Q-learning. These stored experiences are then broken into mini-batches to update the neural network based on prior interactions. On the other hand, the target network helps calculate the target Q-value, which is used to determine the loss for each action during the training phase.

3.6.7 Double Deep Network

The originator of the double dual Q network (DDQN) was "Hado Van Hasselt." (Raj et al., 2015). By breaking down the max operation in the target to action selection, it helps to reduce the overestimation issue. It is the result of combining Dual Neural and Q Network. This technique was developed to address the issue of Q values in previously discussed models being overestimated. Since the action with a higher Q value is known to have the best chance of leading to the next state, the accuracy of the Q value depends on the actions we take, the results we obtain, and the next state of this trial (Liang et al., 2023).

At the beginning of the experiment, we do not have enough Q values to estimate the best possibility. At this stage, as there are fewer Q values to estimate choosing, the highest Q value from the limited values may lead you to a false action towards the

target (Sharma et al., 2021). To overcome this problem, double dual Q network (DDQN) is used. Here, we use two dual Q, one for the selection of the Q value and the other calculates the target Q value for choosing that specific action using the target network (Sarita et al., 2020). This DDQN helps to minimize the overestimation of the Q values, which helps in reducing the training time.

3.6.8 Duelling Q Network

A duelling Q network is a deep reinforcement learning model that separates the estimation of state values and action advantages from traditional Q-learning approaches. It decomposes the action-value function into two streams: one estimating the state value and the other estimating the advantage function of each possible action within that state (Srivastava et al., 2021).

3.6.9 Deep reinforcement relevance network

A new architecture called the Deep Reinforcement Relevance Network (DRRN) is developed to handle action and state spaces in sequential text-based games. Based on the DRRN uses two neural networks—one for encoding states and the other for actions—combined through an interaction function to compute the Q-value in continuous space. Unlike standard deep reinforcement learning (DRL), DRRN focuses on understanding text rather than memorizing it. It effectively captures the relevance between states and actions, achieves higher average rewards, and converges faster than traditional approaches.

3.7 Learning technique in Feed Forward Neural Network

This study uses only the feed forward neural network for simulation due to its simplicity and well-defined learning methods. It is a basic neural network where information flows in a single direction. The computation process for the i^{th} layer is developed as follows:

$$p^{(i)} = f^{(i)}\left(W^{(i)} g^{(i-1)}\right) \tag{3.61}$$

Where i is the number of input signals;

$$p^{(i)} = \left[p_1^{(i)} p_2^{(i)} \ldots . p_N^{(i)}\right] \tag{3.62}$$

p is the signal vector at the i^{th} layer output; and

$$W^{(i)} = \begin{bmatrix} W_{10}^{(i)} & W_{12}^{(i)} & W_{1N}^{(i)} \\ W_{21}^{(i)} & W_{22}^{(i)} & W_{2N}^{(i)} \\ W_{31}^{(i)} & W_{32}^{(i)} & W_{3N}^{(i)} \end{bmatrix} \tag{3.63}$$

W is the weighing matrix between

$$g^{(i-1)} = \begin{cases} A & for\ i = 1 \\ \begin{bmatrix} 1 \\ g^{(i-1)} \end{bmatrix} & for\ i = 2,3 \ldots., R \end{cases} \quad (3.64)$$

In a neural network,A represents the input vector, R is the number of processing levels, and $g(i)$is the activation function. The assumption is that all neurons in a layer are identical, and the network may have multiple hidden layers depending on the specific goal of the network. The output vector y represents the output of the processed NN as:

$$y^{(i)} = \left[y_1^{(i)} y_2^{(i)} \ldots . y_N^{(i)}\right] \quad (3.65)$$

Learning or training in neural networks involves adjusting weights to achieve a desired outcome. The learning method depends on the network's architecture and falls into three main categories: supervised, unsupervised, and reinforcement learning. These are classified based on the presence of a teacher and the type of data provided, and further grouped by the rules they follow. Figure 3.14 presents the Classification of learning algorithms.

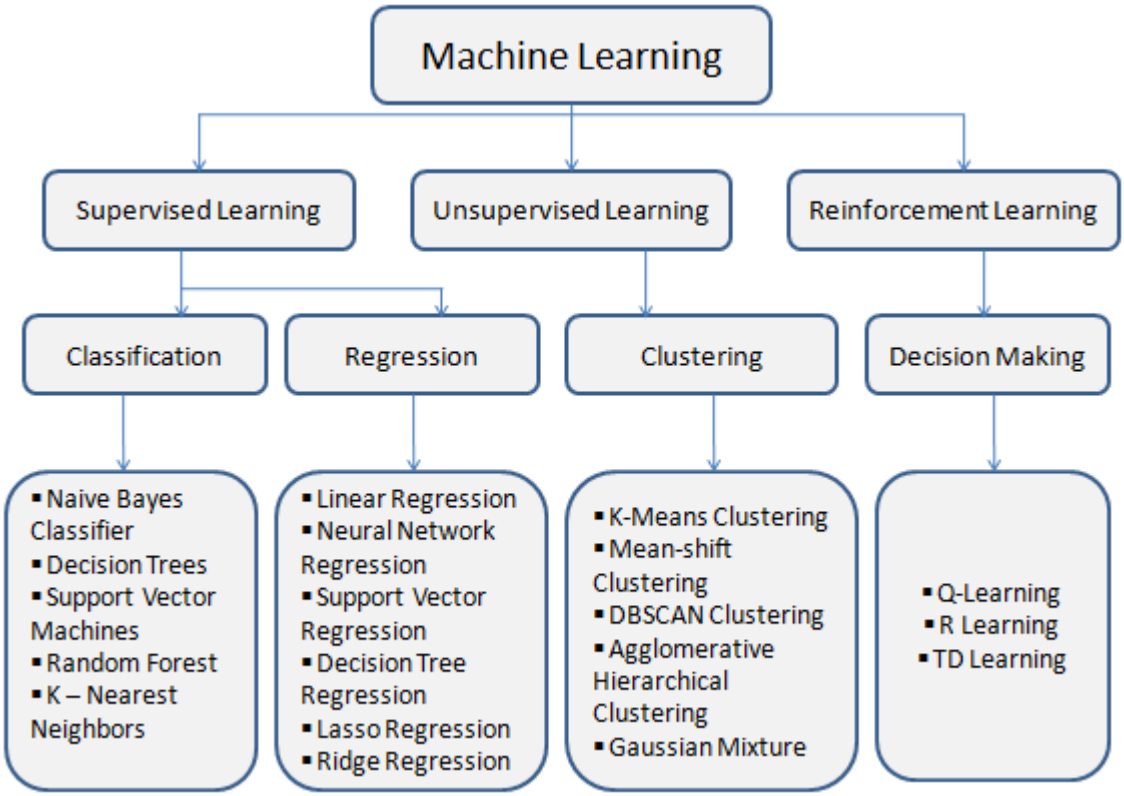

Figure 3.14: Classification of learning algorithms.

In supervised learning, a teacher provides expected results, and the network adjusts its weights to minimize the error between inputs and target values. Both inputs and target values are known before training. In unsupervised learning, no teacher is present, and the network is not given target values. The network is trained using only input data, with the relationship to the target values unknown before training (Hasan et al. 2023); (Hasan et al. 2023).

Feed-forward networks are trained using supervised learning, where input-output sets are collected through simulations or measurements. During training, weights are adjusted based on the error "e" between the network's outputs and the expected results, with the weights being iteratively updated (see equation (2.52)) (Thomas et al. 2020). Figure 3.15 depicts a supervised learning scheme.

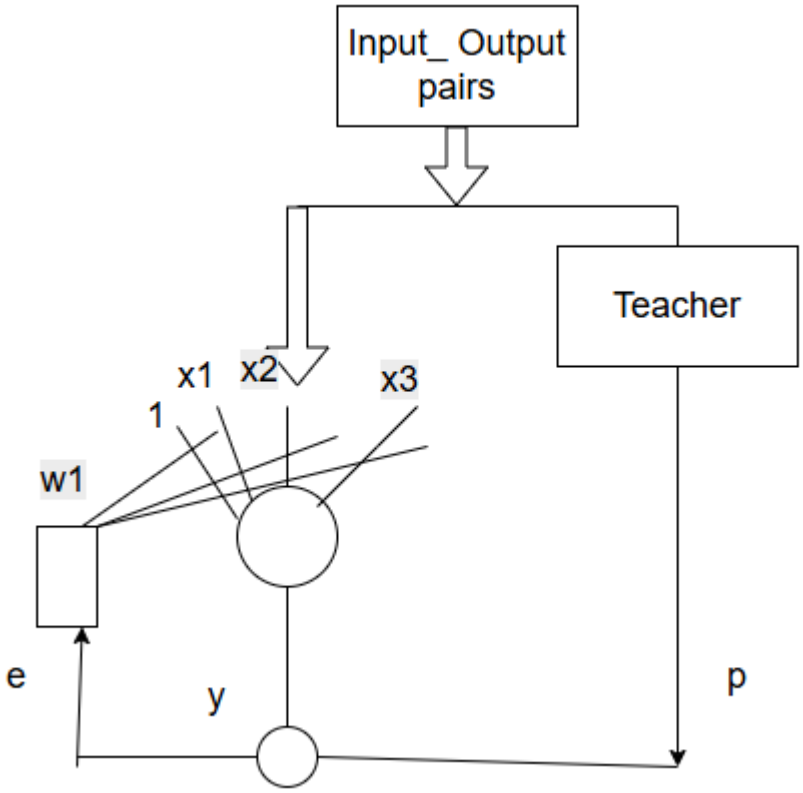

Figure 3.15: Supervised learning scheme

This scheme made use of vector predictor and output values. During training the data are inserted in a function that linked the data to the desired one. An excellent application allowed prediction with random and unseen data. Equations 3.66illustrated the weights changes during training

$$w_{ji}\,(n+1) = w_{ji}\,(n) + \Delta w_{ji}\,(n) \tag{3.66}$$

Where the modified weights are coupled between the connecting layers, and $w_{ji}\,(n+1)$ stands for the previous weights. The alteration factor is $w_{ji}\,(n)$, where n is the number of iterations. The training efficacy of the jth neuron in a one-layer NN is increased by minimizing the difference between its actual and desired output. Let $y_{jN}^{(i)}$ and $p_j^{(i)}$ reflect the jth neuron's actual and intended outputs in the nth iteration. The BPA is a training method where neural network nodes are assigned random weights, input-output pairs are fed in, and errors are calculated by propagating them backward from the output layer. Weights are then updated, and the process is repeated using the full training dataset until the network converges to the desired targets. This layer-by-layer backward process is commonly used in various applications, including computing Jacobian and Hessian matrices and applying different error functions. Figure 3.16 presents the back propagation error.

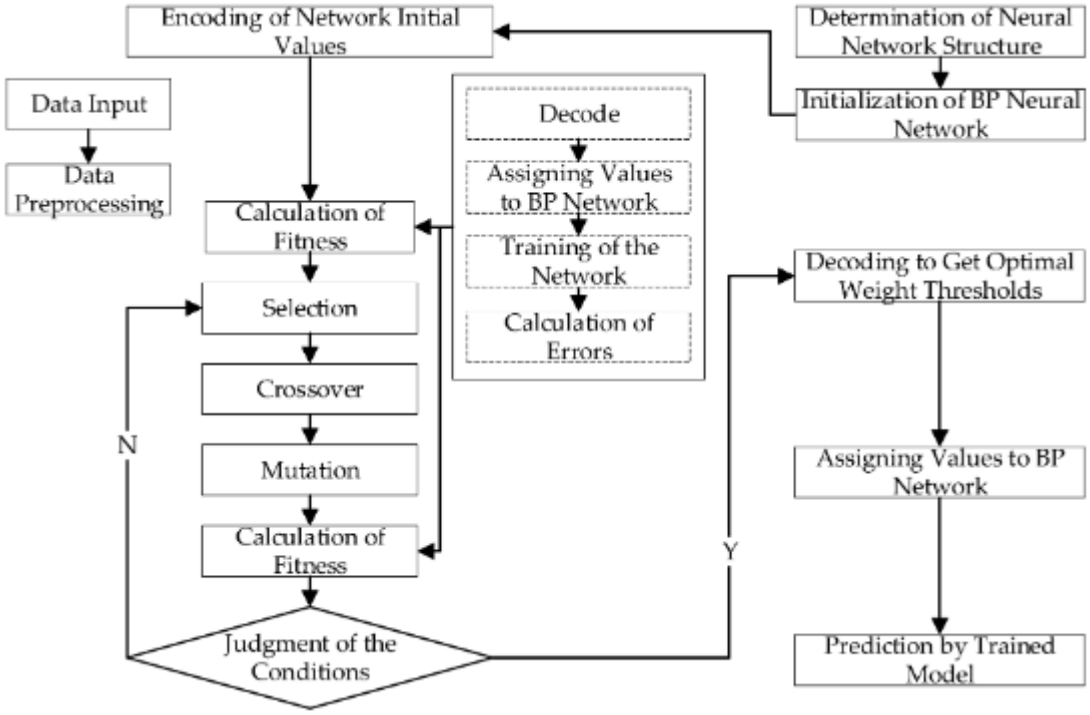

Figure 3.16: The back error propagation.

Although the BPA has various implementations, its core concept remains the same. What differs is the method of updating weights layer by layer during back propagation. These weight updates are also applied in training recurrent neural networks, and the learning rate can be assessed by comparing correction values across iterations.

3.7.2 Framework of Fault Recognition Scheme

The primary objective of this strategy is to design, develop, test, and implement a fault localization, type identification, and fault recognition system. Even though the fundamental idea behind how relays work hasn't changed, smart grid technology has brought about several improvements over traditional electromechanical relays and drastically modified how relays function. The training and testing data sets are the two sets into which the entire generated data is subsequently divided.

Fault detection is the initial step in the process, which is then followed by determining the type of fault based on the distribution line's faulty phases and, ultimately, pinpointing the problem's location and even more. For fault recognition, a back-propagation-based NN has been used, along with others of a like nature for fault type identification and fault localization.

3.7.3 The Neural Network Training Process

The process by which the NN learns from the input sets and updates its weights accordingly is known as training (Kim et al. 2019). It is necessary to enter the training data set (a collection of input-output pairs) into the neural network in order to train it. This teaches the neural network what output to expect when feeding it input. The training data set is gradually learned, the capacity to generalize from the input is gradually improved, and eventually, an output is generated when it receives fresh data. The user-defined performance function is the one that feed forward networks commonly utilize, and it is updated by the NNs as they train with the aim of reducing Mean Square Error (MSE). MSE is used as a performance metric in this study (Phan et al.,2024). For the aim of training the neural networks for various steps, sequential 74 feeding of input and output pairs is used.

3.7.4 Neural Network Testing Process

Following training in a neural network application, the next step is to test the neural network. To make sure the trained dataset generalizes well and generates the desired results when it receives fresh data. Plotting the best linear regression fit between the actual NN's outputs and the anticipated targets is one way to evaluate the performance of the trained network (Rahman et al. 2021).

The performance of an Artificial Neural Network (ANN) is evaluated by comparing its output to expected results (Liu et al. 2023),. As the correlation value rrr nears 1, performance may decline. Tools like confusion matrices and percentage error calculations using new datasets help assess accuracy. A low error and high classification rate indicate good performance. The dataset is divided into training, validation, and test sets in MATLAB Simulink. Training adjusts network weights, validation monitors over-fitting, and testing evaluates final performance (Hu et al.

2022). Training stops when the validation error increases to avoid overfitting. If the test set performs best at a different point than the validation set, the model may be unreliable(Babu et al., 2022).

3.8 Summary

The chapter on fault analysis highlights the need to understand fault types and detection methods in distribution networks. Traditional techniques, such as impedance-based and threshold methods, often fall short in complex systems. Hybrid approaches that incorporate digital signal processing improve detection and classification accuracy. These methods combine fault data analysis with real-time monitoring, making them suitable for modern grid needs. Intelligent techniques using AI and machine learning can accurately identify fault types, locations, and features while enabling predictive analysis for proactive maintenance and quicker responses. By integrating conventional, hybrid, and intelligent approaches, fault analysis becomes more robust and efficient, improving the resilience of modern distribution networks.

CHAPTER FOUR

MODELLING AND DATA ANALYSIS

4.1 Introduction

This chapter discusses the modelling and data analysis phase as well as the procedures for creating trustworthy models from actual data. It discusses feature selection, pre-processing data, and developing deep learning models tailored to the IEEE 13-bus system (A. Toubal et al. 2018). Additionally, the efficacy of the created LSTM model is assessed in this context. This chapter attempts to give a detailed knowledge of how deep learning might transform fault detection and classification in PV-integrated power systems through extensive modelling and meticulous data analysis. By addressing both theoretical and practical aspects, we offer valuable insights for researchers, practitioners, and policymakers seeking to enhance the resilience and efficiency of modern electrical grids (J. Wang et al. 2022).

4.2 Modelling of the Power Distribution Network

The IEEE-13 test feeder is a widely used benchmark model for medium-voltage power distribution system research. It features a radial network topology with around 8,500 nodes, starting from a single substation and branching to various loads. The model includes distributed generation (DG) units, allowing for the study of renewable energy integration and its impact on system operations. It also includes: The feeder model includes various node types such as source, load, and intermediate nodes, representing power sources, consumer loads, and distribution elements like transformers, switches, and capacitors. It also incorporates different fault scenarios, including short circuits and ground faults, to assess how protection systems and fault detection algorithms perform. Additionally, the model includes feeder variants to

evaluate the impact of these faults on the distribution system and the behaviour of protection systems.

4.2.1 Load modelling

The IEEE test bus feeder is a heavily loaded system with single- and three-phase loads in delta or Y configurations, characterized by constant PQ, I, or Z. In Simulink, unbalanced loads are modelled using single-phase dynamic load blocks, while balanced loads use three-phase dynamic load blocks. The active and reactive power are given by the following equations:

$$P = P_{O\left(\frac{U}{U_o}\right)^{np}} \tag{4.1}$$

$$Q = Q_{O\left(\frac{U}{U_o}\right)^{nq}} \tag{4.2}$$

where P_O, Q_O and U_O are the initial values of active power, reactive power and voltages. The constant np and nq are controlling the type of load as PQ, I or Z. The IEEE 13 BUS has 12 loads; the details of the distributed loads are given in Appendix. The load types are either constant power PQ, voltage (V) or current (I).

4.2.2 Transformer modelling

Three-phase transformer characteristics of 4.16kV/480kV are used in the IEEE13 bus, with the following specifications shown in Table 4.1.

Table 4.1: Three-phase transformer characteristics

	kV	kV -High	$kV - Low$	R-%	X-%
Substation	5000	115-D	4.16 Gr.W	1	8
XFM-1	500	4.16 Gr.W	0.48 Gr.W	1.1	2

4.2.3 Line modelling

The PI-line model as suggested in (J. Wang et al. 2022) is utilized to simulate the distribution feeder lines. You may find the IEEE-specified line configuration, impedance, and admittance data in (Weng et al. 2023).

4.2.4 Voltage regulation

The aim of this work is to use DGs to simplify the micro gird model without sacrificing accuracy. In order to accomplish this, the IEEE-13 Bus system is modelled without a voltage regulator. Since DGs can change the grid's voltage level

4.2.5 Short circuit analysis

Short circuit analysis is useful in both network planning and addressing operational issues. In the planning phase, it helps assess equipment ratings and estimate the maximum and minimum current levels (Z. Li et al., 2023). This information is critical for selecting appropriate switchgear, determining component ratings, and confirming the effectiveness of the protection strategy. When it comes to operational challenges, short circuit analysis is used to accurately evaluate fault current levels—for example, to identify whether a protection device malfunctioned due to incorrect settings or an actual failure(Ladjici and Tiguercha 2023). MATLAB supports the IEC 60909 standard using two methods for short-circuit calculations: the equivalent voltage source method (using Thevenin's voltage at the fault location) and the superposition method (considering pre-fault voltage). Both methods support LU factorization to improve performance and reduce memory usage, especially for large grids. They can calculate both the initial symmetrical short-circuit current I''_{kl} and the peak short-circuit current I''_{kll}.

Those two currents are combined into the short-circuit current:

$$I''_k = I''_{kl} + I''_{kll} \tag{4.3}$$

All voltage sources are substituted at the problem location with a single equivalent voltage source in order to do the short-circuit calculation using the equivalent voltage source.

4.2.5.1 Equivalent voltage source

The minimum and maximum voltage correction factor, C_{min} and C_{max}, are defined for short-circuit currents based on the network's nominal voltage, with differences in tolerance between networks with a 6% and 10% tolerance. Table 4.2 depicts the Voltage Level during the fault occurrence according to IEC60909.

$$V_Q = \begin{cases} \frac{c.V_N}{\sqrt{3}} \\ \frac{c.V_N}{2} \end{cases} \tag{4.4}$$

Table 4.2: The Voltage Level during the fault occurrence according to IEC60909

Voltage Level	C_{min}	C_{max}
<1kV	0.95	1.05

>1kV	1	**1.10**

- The admittance matrix is a fundamental tool while doing a fault analysis. Each element represents how voltage and current are related to buses. It is calculated with the following equations:

$$\begin{bmatrix} \underline{Y_{11}} & \cdots & \cdots & \underline{Y_{n1}} \\ \vdots & \ddots & \vdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \underline{Y_{1n}} & \cdots & \cdots & \underline{Y_{nn}} \end{bmatrix} \begin{bmatrix} \underline{V_1} \\ \vdots \\ \underline{V_{QJ}} \\ \vdots \\ \underline{V_n} \end{bmatrix} = \begin{bmatrix} \underline{0} \\ \vdots \\ \underline{I''_{kIJ}} \\ \vdots \\ \underline{V_n} \end{bmatrix} \tag{4.5}$$

With I''_{kIJ} as current source contribution .To figure it out, the matrix is multiplied using the impedance matrix (inverted nodal point admittance matrix):

$$\begin{bmatrix} \underline{V_1} \\ \vdots \\ \underline{V_{QJ}} \\ \vdots \\ \underline{V_n} \end{bmatrix} = \begin{bmatrix} \underline{Z_{11}} & \cdots & \cdots & \underline{Z_{n1}} \\ \vdots & \ddots & \vdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \underline{Z_{1n}} & \cdots & \cdots & \underline{Z_{nn}} \end{bmatrix} \begin{bmatrix} \underline{0} \\ \vdots \\ \underline{I''_{kIJ}} \\ \vdots \\ \underline{V_n} \end{bmatrix} \tag{4.6}$$

Then

$$I''_{kIJ} = \frac{V_{QJ}}{Z_{jj}} \tag{4.7}$$

With V_{QJ} as voltage contribution Inverting the nodal point admittance matrix and adding a fault impedance to the diagonal allows for the simultaneous calculation of all buses and short-circuit currents.

$$\begin{bmatrix} \underline{V_{Q1}} & \cdots & \underline{V_{n1}} \\ \vdots & \ddots & \vdots \\ \underline{V_{1n}} & \cdots & \underline{V_{Qn}} \end{bmatrix} = \begin{bmatrix} \underline{Z_{11}} & \cdots & \underline{Z_{n1}} \\ \vdots & \ddots & \vdots \\ \underline{Z_{1n}} & \cdots & \underline{Z_{nn}} \end{bmatrix} \begin{bmatrix} \underline{I''_{kIJ}} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \underline{I''_{kIJ}} \end{bmatrix} \tag{4.8}$$

$$\begin{bmatrix} I''_{kIJ} \\ \vdots \\ I''_{kIn} \end{bmatrix} = \begin{bmatrix} \frac{V_{Q1}}{Z_{11}} \\ \vdots \\ \frac{V_{Qn}}{Z_{nn}} \end{bmatrix} \tag{4.9}$$

Short-circuit currents can be calculated simultaneously by inverting the nodal point admittance matrix and adding a fault impedance to the diagonal, resulting in the calculation of all buses.

$$\begin{bmatrix} I''_{kIJ} \\ \vdots \\ I''_{kIn} \end{bmatrix} = \begin{bmatrix} \frac{V_{Q1}}{Z_{11}+Z_{Fault}} \\ \vdots \\ \frac{V_{Qn}}{Z_{nn}+Z_{Fault}} \end{bmatrix} \tag{4.10}$$

The current source component of short circuit current is calculated by short-circuiting all voltage sources and considering only current sources, resulting in bus currents.

$$\begin{bmatrix} I_1 \\ \vdots \\ I_m \\ \vdots \\ \underline{I_n} \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ \underline{I''_{kIJ}} \\ \vdots \\ 0 \end{bmatrix} = \begin{bmatrix} I''_{kC1} \\ \vdots \\ \underline{I''_{kcJ}} \\ \vdots \\ I''_{kcn} \end{bmatrix} = \begin{bmatrix} -I''_{kC1} \\ \vdots \\ \underline{I''_{kIJ}} + \underline{I''_{kcJ}} \\ \vdots \\ I''_{kcn} \end{bmatrix} \tag{4.11}$$

Where I''_{kC1} are the short circuit fed by the converter and the network equations are given as:

$$\begin{bmatrix} \underline{V_1} \\ \vdots \\ 0 \\ \vdots \\ \underline{V_n} \end{bmatrix} = \begin{bmatrix} \underline{Z_{11}} & \cdots & \cdots & \cdots & \underline{Z_{n1}} \\ \vdots & \ddots & \vdots & \vdots & \vdots \\ \vdots & \vdots & Z_{jj} & \ddots & \vdots \\ \underline{Z_{1n}} & \cdots & \cdots & \cdots & \underline{Z_{nn}} \end{bmatrix} \begin{bmatrix} -I''_{kC1} \\ \vdots \\ \underline{I''_{kIJ}} + \underline{I''_{kcJ}} \\ \vdots \\ -I''_{kcn} \end{bmatrix} \tag{4.12}$$

$$0 = \underline{Z_{jj}} \times \underline{I''_{kIJ}} - \sum_{m=1}^{n} \underline{Z_{jm}} \times \underline{I''_{kIJ}} \tag{4.13}$$

$$I''_{kIJ} = \frac{1}{Z_{jj}} \sum_{m=1}^{n} Z_{jm} \times I_{kcm} \tag{4.14}$$

The matrix equation can be used to calculate all short circuit currents for faults at each bus simultaneously.

$$\begin{bmatrix} I''_{kII1} \\ \vdots \\ I''_{kIIn} \end{bmatrix} = \begin{bmatrix} \underline{Z_{11}} & \cdots & \cdots & \underline{Z_{n1}} \\ \vdots & \ddots & \vdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \underline{Z_{1n}} & \cdots & \cdots & \underline{Z_{nn}} \end{bmatrix} \begin{bmatrix} \frac{I''_{kC1}}{\vdots} \\ \underline{Z_{11}} \\ \vdots \\ \frac{I''_{kCn}}{\underline{Z_{nn}}} \end{bmatrix} \tag{4.15}$$

The peak short circuit current is calculated as:

$$\begin{bmatrix} ip,1 \\ \vdots \\ ip,n \end{bmatrix} = \sqrt{2} \left(\begin{bmatrix} k_1 \\ \vdots \\ k_1 \end{bmatrix} + \begin{bmatrix} I''_{kI,1} \\ \vdots \\ I''_{kI,n} \end{bmatrix} + \begin{bmatrix} I''_{kII,1} \\ \vdots \\ I''_{kII,n} \end{bmatrix} \right) \tag{4.16}$$

Where k is the peak factor. In radial networks k is given as:

$$k = 1.02 + 0.98\, e^{-3R/X} \quad (4.17)$$

Where is the ratio R/X the equivalent short circuit of the impedance Z_K at the fault location.

4.3 Proposed LSTM for fault recognition approach

To use the LSTM algorithm for fault detection on an IEEE bus, time-series data of power, voltage, and current must first be collected, labelled, normalized, and split into training, validation, and test sets. An LSTM model is then built with input, LSTM, and dense layers, and trained using optimization methods like Adam. After tuning hyper parameters with the validation set, the model's accuracy is evaluated on the test set. Once trained, the model can detect and classify faults in real-time on the IEEE bus. Figure 4.3 presents the LSTM fault recognition algorithm.

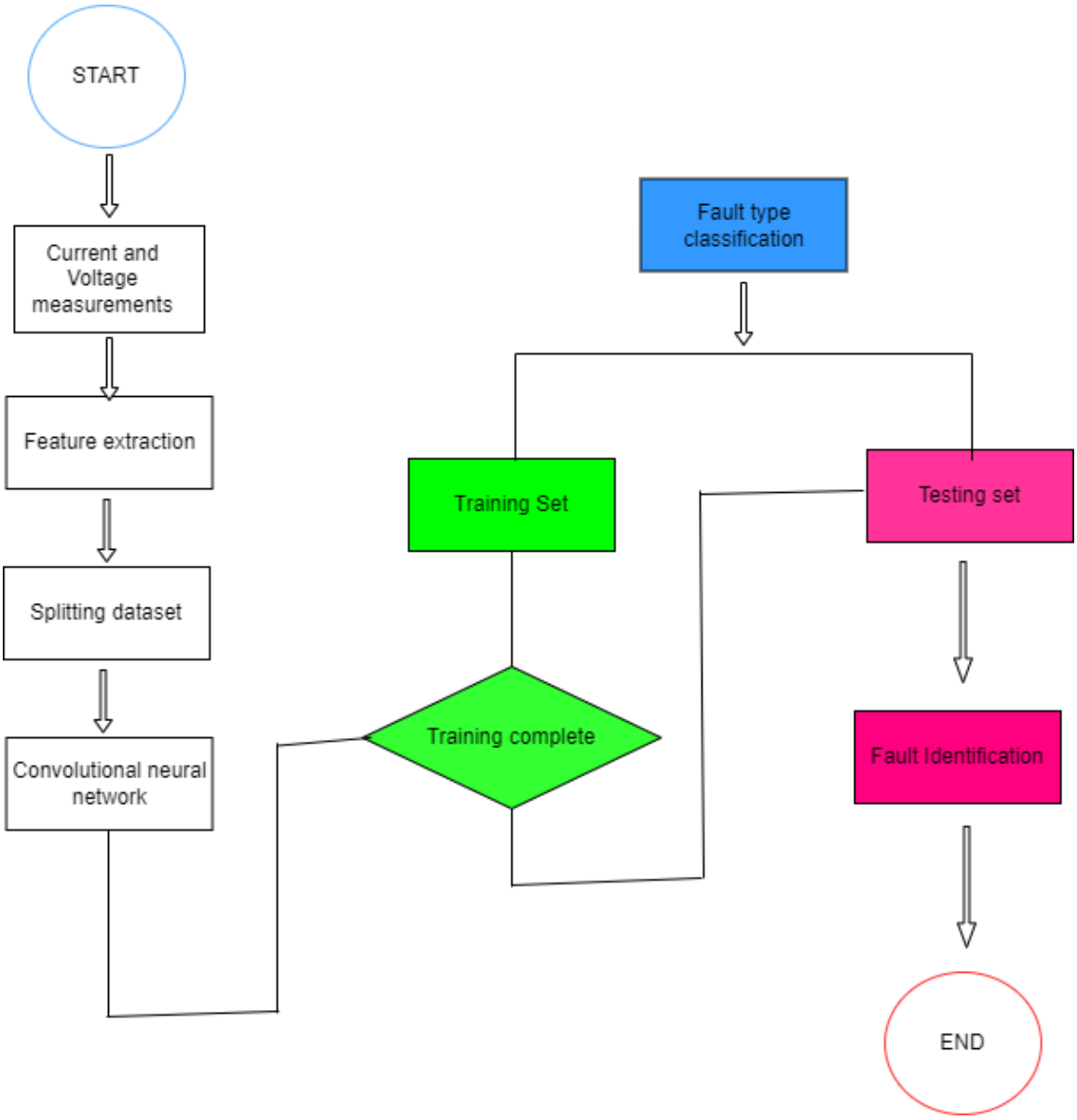

Figure 4.3 The Long Short-Term Memory (LSTM) fault recognition algorithm.

This fault recognition algorithm relied on current and voltage measurements from the bus IEEE13. Then the data was splitted into training and testing data and it all came from the same neural network. For each fault simulated the algorithm identified if there is a fault or not and eventually the fault.

4.3.1 Sensors

For the general situation of n conductors, the magnetic field near a transmission or distribution line can be found. In (Y. Zhang et al. 2022) the author provides the necessary analysis; however, to account for variations in the conductor current magnitudes, some changes have been made to this synopsis of their work. Sensors model are the computational representation on how it reacts to conversion of a given signal. It display the relationship between the input and the output by incorporating parameters like the linearity or sensitivity. Figure 4.3 depicts the sensor model.

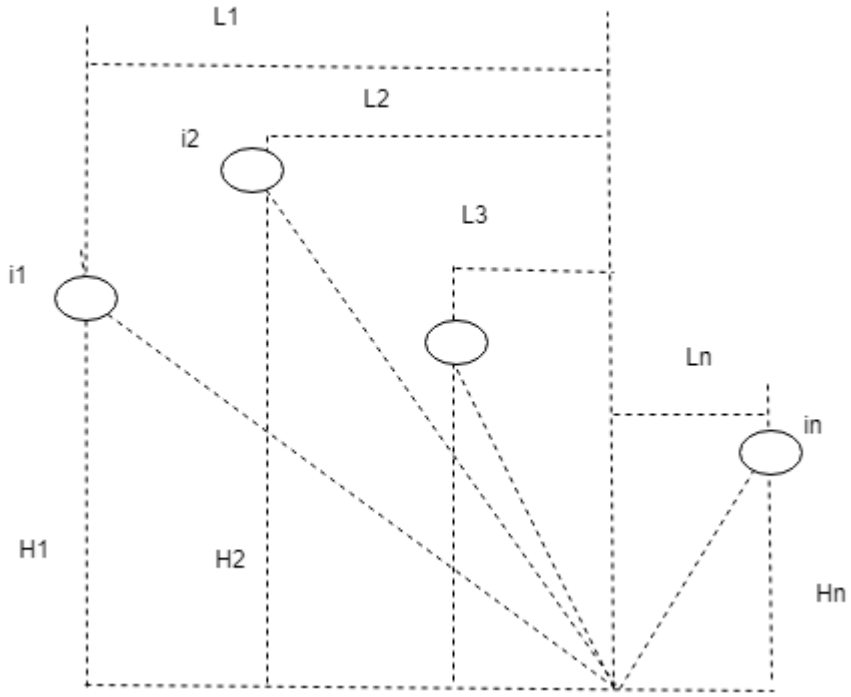

Figure 4.3: The sensor model.

The magnetic field due to any given conductor is:

$$H_{x,k} = D_k . i_k \quad \text{and} \quad H_{y,k} = Q_k . i_k \tag{4.18}$$

where

$$D_k = \frac{\text{Cos}\, y_k^2}{2\pi h_k} \text{ and } \quad D_k = \frac{\text{Sin}\, y_k^2 \,\text{Cos}\, y_k}{2\pi h_k} \tag{4.19}$$

$$y_k = \tan\left(\frac{L_k}{h_k}\right)^{-1} \tag{4.20}$$

The combination of sine and cosine represents the main current such as:

$$i_k = \sqrt{2}\,[X_k.\sin(wt) + Y_k.\cos(wt)\,] \tag{4.21}$$

where X_k and Y_k are constant values

Since

$$X_k.\sin(wt) + Y_k.\cos(wt) = I_k \sin(wt + \varphi_k) \tag{4.23}$$

where φ_k is the phase shift of the current and I_k is the RMS value of the current

$$\varphi_k = \tan\left(\frac{Y_k}{X_k}\right)^{-1} \tag{4.24}$$

Both terms X_k and Y_k can be explained as:

$$X_k = \frac{I_k}{\sqrt{1 + \tan\varphi_k^2}} \tag{4.25}$$

And

$$Y_k = \frac{I_k \tan\varphi_k}{\sqrt{1 + \tan\varphi_k^2}} \tag{4.26}$$

Using the trigonometry identity, the equations are:

$$1 + \tan(\varphi)_k^2 = \sec(\varphi)_k \tag{4.27}$$

With the following values of X_k and Y_k

$$X_k = I_k \cos(\varphi_k) \tag{4.28}$$

$$Y_k = I_k \sin(\varphi_k) \tag{4.29}$$

The currents and their coefficients can be defined as:

$$[i] = \begin{bmatrix} i_1 \\ i_2 \\ \vdots \\ \vdots \\ i_n \end{bmatrix} \text{ and } [XY] = \begin{bmatrix} I_1 & 0 \\ X_1 & Y_1 \\ X_2 & Y_2 \\ \vdots & \vdots \\ X_n & Y_n \end{bmatrix} \quad (4.30)$$

The magnetic field due to all field is:

$$[H] = [P][i] \quad (4.31)$$

Where the magnetic field matrix is

$$[H] = \begin{bmatrix} H_x \\ H_y \end{bmatrix} \quad (4.32)$$

And the position matrix is

$$[P] = \begin{bmatrix} D_1 & D_2 & \cdots & \cdots & D_n \\ Q_1 & Q_2 & \cdots & \cdots & Q_n \end{bmatrix} \quad (4.33)$$

The magnetic field can thus be described as

$$H_x = \sqrt{2}\,[A_x \sin(wt) + B_x \cos(wt)] = \hat{H}_x \sin(wt + \emptyset) \quad (4.34)$$

$$H_y = \sqrt{2}\,\left[A_y \sin(wt) + B_y \cos(wt)\right] = \hat{H}_y \cos(wt + \emptyset) \quad (4.35)$$

4.4 PV Plant

In MATLAB/SIMULINK, a 200 kW PV module is modelled using the Maximum Power Point Tracking (MPPT) approach to optimize power efficiency. Among various MPPT algorithms, Perturb and Observe (P&O) is the most widely used due to its 98% tracking efficiency and simplicity. P&O operates by comparing voltage and power levels to identify the maximum power point, and it does not require system parameters for implementation.

Two common PV models, the single-diode and double-diode models, are frequently used in research. These models are widely adopted for simulating the behaviour of PV systems (Polytechnique, 2013). The PV Cell electrical model is shown in Figure 4.4. The power output of the PV generator is calculated using:

$$I^{m/p} = I_{ph_{m/p}} - I_{o\frac{m}{p}}\left[exp\left(\frac{\left(V^{\frac{m}{p}} + R_{s\frac{m}{p}} \times I^{\frac{m}{p}}\right)}{nV_{T\frac{m}{p}}}\right) - 1\right] - \left(V^{\frac{m}{p}} + R_{s\frac{m}{p}} \times I^{\frac{m}{p}}\right)/R_{p\frac{m}{p}} \quad (4.36)$$

$$I^{m/p} = I_{ph_{m/p}} - I_{o1\frac{m}{p}}\left[exp\left(\frac{\left(V^{\frac{m}{p}} + R_{s\frac{m}{p}} \times I^{\frac{m}{p}}\right)}{n_1 V_{T\frac{m}{p}}}\right) - 1\right] - \cdots I_{o2\frac{m}{p}}\left[exp\left(\frac{\left(V^{\frac{m}{p}} + R_{s\frac{m}{p}} \times I^{\frac{m}{p}}\right)}{n_2 V_{T\frac{m}{p}}}\right) - 1\right] - \frac{\left(V^{\frac{m}{p}} + R_{s\frac{m}{p}} \times I^{\frac{m}{p}}\right)}{R_{p\frac{m}{p}}} \quad (4.37)$$

$$\text{Where} \quad P^{m/p} = V^{m/p}.I^{m/p} \quad (4.38)$$

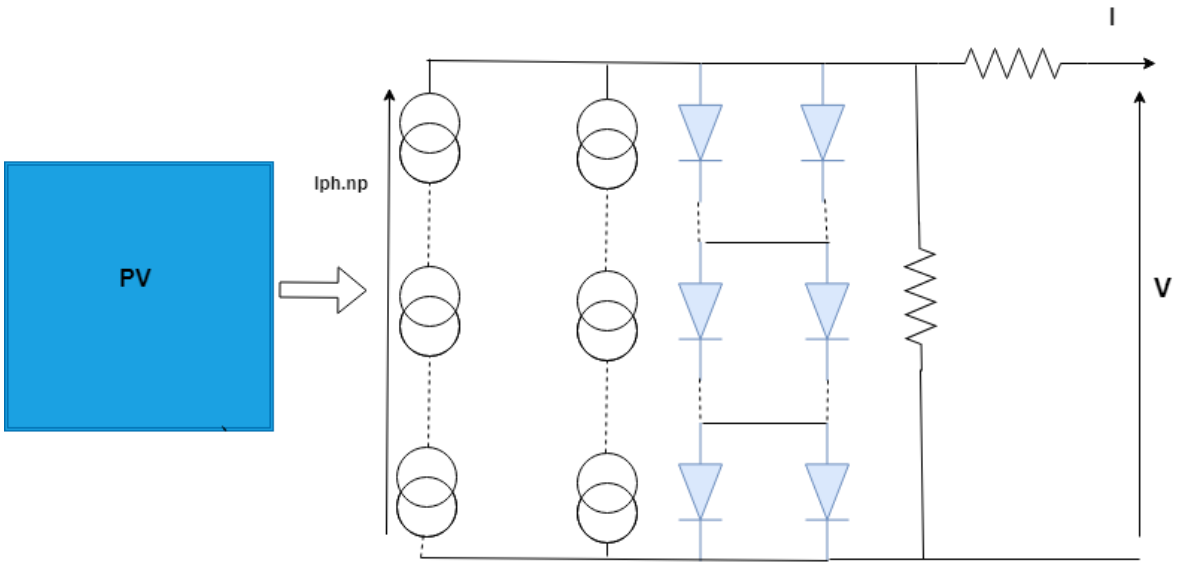

Figure 4.4: PV Cell electrical model.

The single-diode model, inspired by the saturation currents of diodes and an idealistic factor n, has been shown to effectively match experimental data in recent years (Tan et al., 2022). Its representation is given by equation (2.1). In the described equation, there is 5 main parameters to determine the photocurrent $I_{ph_{m/p}}$, saturation current $I_{o\frac{m}{p}}$, series and parallel resistance $R_{s\frac{m}{p}}$ and $R_{p\frac{m}{p}}$, coefficient of ideality n.

The double diode model offers higher accuracy for PV systems but involves more complex calculations, while the simpler and more commonly used single-diode model is easier to work with. A PV system consists of cells connected in series and parallel, which are described by the following equation:

$$I_{PV} = N_P.I_{phm} - N_P.I_{om}\left[exp\left(\frac{V_{pvm} + I_{pvm}.R_{sm}\left(\frac{N_S}{N_P}\right)}{nN_S V_{Tm}}\right) - 1\right] - \frac{V_{pvm} + I_{pvm}.R_{sm}\left(\frac{N_S}{N_P}\right)}{R_{sm}\left(\frac{N_S}{N_P}\right)} \quad (4.39)$$

4.4 Relationship between the conditions of operation and the PV cell

The known and unknown parameters of a PV system model change based on real-world operating conditions, such as light intensity and temperature, which commonly affect PV panels (De Jesus et al., 2020). Therefore, it is crucial to understand how these parameters relate to environmental factors, especially sunlight and temperature. Consequently, parameter extraction relies heavily on the specific weather conditions during the PV system's operation. Figure 4.5 shows the PV Cell operating conditions relationship.

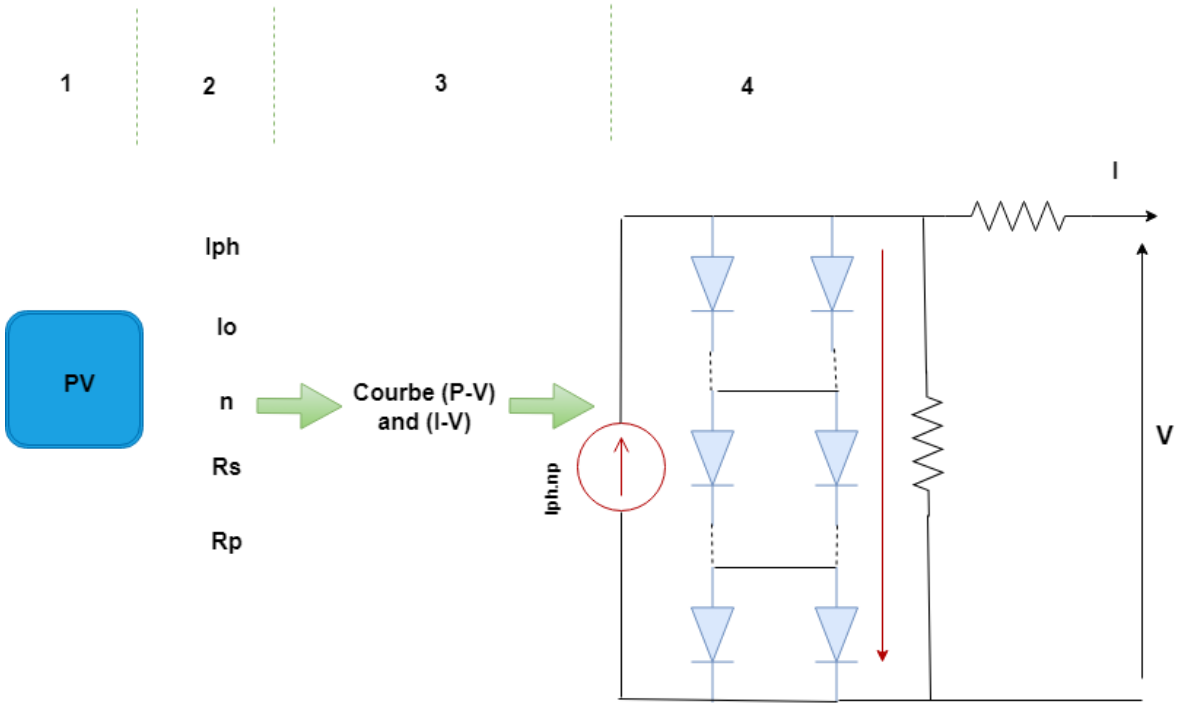

Figure 4.5: The PV Cell operating conditions relationship.

4.4.2 Standard conditions

Equation 1 can be reduced to:

$$I^{m/p} = \left(I_{ph_{m/p}}\right)_{ref} - \left(I_{0_{m/p}}\right)_{ref}\left[exp\left(\frac{\left(V^{\frac{m}{p}} + \left(R_{S_{m/p}}\right)_{ref} \times I^{\frac{m}{p}}\right)}{n\left(V_{T_{m/p}}\right)_{ref}}\right) - 1\right] - \left(V^{\frac{m}{p}} + \left(R_{S_{m/p}}\right)_{ref} \times I^{\frac{m}{p}}\right) / \left(R_{S_{m/p}}\right)_{ref} \quad (4.40)$$

With $\left(I_{ph_{m/p}}\right)_{ref}, \left(I_{0_{m/p}}\right)_{ref}, \left(R_{S_{m/p}}\right)_{ref}, \left(R_{p_{m/p}}\right)_{ref}$ and n the photon current, saturation current, series and parallel resistance, and diode factor. The following

equations can be established, respectively, based on the assessment of the currents at specific short-circuit points:

$$\left(I^{m/p}\right)_{\boldsymbol{sc,ref}} = \left(I_{ph_{m/p}}\right)_{ref} - \left(I_{0\frac{m}{p}}\right)_{ref} . \left[exp\left(\frac{\left(\left(R_{S_{m/p}}\right)_{ref} \times \left(I^{m/p}\right)_{\boldsymbol{sc,ref}}\right)}{n\left(V_{T_{m/p}}\right)_{ref}}\right) - 1\right] - \left(\left(R_{S_{m/p}}\right)_{ref} \times \left(I^{m/p}\right)_{\boldsymbol{sc,ref}}\right) / \left(R_{S_{m/p}}\right)_{ref} \quad (4.41)$$

$$\left(I_{ph_{m/p}}\right)_{ref} - \left(I_{0\frac{m}{p}}\right)_{ref} . \left[exp\left(\frac{\left(\left(V^{m/p}\right)_{\boldsymbol{Oref}}\right)}{n\left(V_{T_{m/p}}\right)_{ref}}\right) - 1\right] - \left(\left(V^{m/p}\right)_{\boldsymbol{O,ref}}\right) / \left(R_{S_{m/p}}\right)_{ref}$$

$$\left(I^{m/p}\right)_{\boldsymbol{max}} = \left(I_{ph_{m/p}}\right)_{ref} - \left(I_{0_{m/p}}\right)_{ref} \left[exp\left(\frac{\left(V^{\frac{m}{p}}{}_{\max Ref} + \left(R_{S_{m/p}}\right)_{ref} \times I^{m/p}{}_{\max Rref}\right)}{n\left(V_{T_{m/p}}\right)_{ref}}\right) - 1\right] - \left(V^{\frac{m}{p}}{}_{\max Ref} + \left(R_{S_{m/p}}\right)_{ref} \times I^{m/p}{}_{\max Rref}\right) / \left(R_{p_{m/p}}\right)_{ref} \quad (4.42)$$

The derivative of the one-diode model (2.2) in circuit open, then short circuit, under standard test circumstances with

$$\left(R_{S_{m/p}}\right)_{ref} = -\frac{dV^{\frac{m}{p}}}{dI^{\frac{m}{p}}}\Big| I^{\frac{m}{p}} = 0, V^{\frac{m}{p}} = \left(V_{T\frac{m}{p}}\right)_{Oref} - \frac{1}{X_V} \quad (4.43)$$

$$\text{With } X_V = \frac{I_{Oref}}{V_{th,ref}} exp\left(\left(V^{\frac{m}{p}}\right)_{0ref} / \left(V_{T_{m/p}}\right)_{ref}\right) \quad (4.44)$$

$$\frac{dV^{\frac{m}{p}}}{dI^{\frac{m}{p}}}\Big| I^{\frac{m}{p}} = \left(I^{m/p}\right)_{\boldsymbol{sc,ref}}, V^{\frac{m}{p}} = 0 = -\frac{1}{\left(R_{p_{m/p}}\right)_{ref}} \quad (4.45)$$

4.5 Characteristics of the proposed distributed system

The proposed system that will be connected to the IEEE 13 bus is mostly made of solar energy. The structure of the system is composed of the following: PV module, three-phase inverter, LCL filter and controller.

4.5.1 PV module

The specifications of the PV module utilized in the simulation are described in full in Table 4 These properties change with temperature and irradiation, resulting in variations in the curves (I-V) and (P-V) as a function of both temperature and irradiation.

Table 4.3: The specifications of the PV module

Cell	Polycrystalline silicon
No of cells and connections	36 in series
Open circuit voltage	21.6V
Short circuit voltage	5.16A
Maximum Power Voltage at Pmax	21.65V
Maximum Power Current	4.75A
Maximum Power	80W (+10%/−5%)
Module efficiency	12.40%
Maximum system voltage	600V DC
Series Fuse rating	10A
Type of Output Terminal	Junction Box
Temperature Coefficient of Isc	$0.65e{-}3 \pm 0.015/o_C$
Temperature coefficient of Voc	$-160 \pm 20mV\%/o_C$
Temperature coefficient of Power	$-0.5 \pm 0.05\%/o_C$

4.5.2 LCL Filter

The following system parameters mentioned in Table 4.4 are considered for designing the requisite LCL filter:

Table 4.4: Nominal system parameters

f_g	Grid frequency
f_{sw}	PWM carrier frequency
w_{res}	Resonance frequency
P_n	Rated active power
E_n	Line to line RMS voltage
V_{DC}	DC link voltage
L_1	Inverter side inductor
L_2	Grid side inductor
C	Capacitor
R_d	Damping resistance
V_g	Grid voltage

4.5.3 LCL Filter design process

In typical three-phase inverters, the three phases are identical and operate independently from one another (Li et al., 2023). This results in three separate systems, each functioning similarly to a single-phase inverter. When using an LCL filter in the inverter design, each phase includes three key components: two inductors and one capacitor, along with associated series resistances (De Jesus et al., 2020). Each inductor is wound on a separate, isolated core. In Table 4.5, the parameters used in filter calculation are presented.

Table 4.5: Parameters used in Filter calculation

Parameters	Nomenclature	Values
Voltage line to line	V_{LL}	380V
Voltage at DC link	V_{DC}	600V
Resonance frequency	f_w	30000Hz
Nominal Power	P_n	210000kW

$$Z_b = \frac{V_{LL}^2}{P_n} \tag{4.46}$$

$= \frac{380^2}{210} = 687.61\Omega$

$$Z_b = \frac{1}{Z_b \times \omega_n} \tag{4.47}$$

$= \frac{1}{2\pi \times 50 \times 687.61}$

$$C_b = 462\ \mu F$$

The inverter side inductor mainly minimizes the ripple current of the MOSFET inverter.

$$X_{LPU} = \frac{L\omega_n}{Z_b} \tag{4.48}$$

$$= \frac{Z_b \times X_{LPU}}{\omega_n} \tag{4.49}$$

$= \frac{0.05 \times 686}{2\pi \times 50} = 10.91$ mH

$$\Delta I_{Lmax} = \frac{V_{DC}}{8 f_\omega L_i} \tag{4.50}$$

$= \frac{600}{8 \times 30 \times 109.1}$

$\Delta I_{Lmax} = 2.29\ A$

$$L_g = rL_i \quad (4.51)$$

$$L_{Total} = \frac{X_{L\_Total} \times Z_b}{\omega_n} \quad (4.52)$$

$= \frac{686.6 \times 0.09}{2\pi 50} =$ 19.67mH

$$\text{So } L_g = L_{Total} - L_i \quad (4.53)$$

$L_g = 19.67 - 10.91 = 8.67$ mH

$$C_f = \beta . C_b \quad (4.54)$$

$= 0.05 \times 462 = 23.1\mu F$

The filter resonance frequency can be calculated with:

$$f_{res} = \frac{1}{2\pi}\sqrt{\frac{L_i + L_g}{L_i \times L_g \times C_f}} \quad (4.55)$$

$= \frac{1}{2\pi}\sqrt{\frac{19.97}{8.67.10^{-3} \times 10.91.10^{-3} \times 23.1 \times 10^{-6}}}$

$f_{res} =$ 2055 Hz

$\omega_{res} =$ $2\pi \times 2055 = 12911.9\, rad/sec$

The requirement of about 5% is perfectly matched. The damping resistor value is calculated as follow:

$R_f = \frac{1}{3 \times 2\pi \times 2055 \times 23.1 \times 10^{-6}}$

$R_f = 1.12\, \Omega$

There are key parameters that will be used for the software implementation phase in MATLAB/ Simulink .Table 4.6 presents the LCL filter parameters found after calculation.

Table 4.6: LCL filter parameters

Parameter	Name	Value
Grid side inductor	L_g	8.67 mH
Inverter side inductor	L_i	10.91
Filter capacitance	C_f	$23.1\mu F$
Damping resistor	R_f	1.12Ω

Resonance frequency	ω_{res}	$12911.9\ rad/sec$

4.6 Control Scheme of the PV plant

The proposed model is supposed to work in grid connected mode. The power flow in the micro grid is managed by the converter controllers that are suggested in each distributed generating model. Furthermore, the design of LCL filters, current, and voltage controllers ensures dependable power quality.

4.6.1. Three-phase MOSFET

The developed model is a three-stage bidirectional DC to AC inverter using Pulse width modulation (PWM) control. It utilizes a six-switch, three-phase inverter topology, where each phase consists of two switches connected in series(Wang et al. 2021); (Wang et al. 2020). The system converts DC power to AC at the desired voltage and frequency.MOSFET switches with anti-parallel diodes are used to allow bidirectional current flow. The inverter is controlled using Sinusoidal Pulse Width Modulation (SPWM), which generates switching signals by comparing sinusoidal reference waves with a triangular carrier wave. This configuration is widely used due to its simplicity and adaptability for adding advanced inverter features.(Yan & Xu 2020). Table 4.8 represents the PWM switching pattern. Equations represent the voltage wave form.

$$\text{Va\_ref} = A\sin(2\pi ft + \theta) \tag{4.56}$$

$$\text{Vb\_ref} = A\sin\left(2\pi ft + \theta - 120^{\circ}\right) \tag{4.57}$$

$$\text{Vc\_ref} = A\sin\left(2\pi ft + \theta + 120^{\circ}\right) \tag{4.58}$$

Energy losses are minimized with smooth ad accurate control devices. Where A represents the amplitude, f frequency, θ phase shift. The switches go on and off depending on the value of the voltage. Table 4.7 shows the PWM switching pattern.

Table 4.7: PWM switching pattern

Switch	Applicable Sinewave	$V_S > V_T$	$V_S < V_T$
S_1	V_a	ON	OFF
S_2		OFF	ON
S_3	V_b	ON	OFF
S_4		OFF	ON
S_5	V_c	ON	OFF

S_6		OFF	ON

There can be a short output over the DC source, which could harm the switches or the total inverter when there is concurrent exchanging. In any case, exchanging is performed dynamically; the mix of the two corner to corner switches gives the unfiltered yield voltage(Khotsirwong et al. 2022). A comparing numerical inverter was introduced to depicting the power electronic voltage source inverter(VSI)design to build up the control framework. The regulators 'factors are the energy stockpiling components.

4.6.2 Output filter

It was provided to smooth the current and voltages delivered from the converter to the grid. In grid tied PV, there is a considerable level of noise and losses due to the switching nature of power electronic equipment. This filter was represented with differential equations of each current phases based on its design parameters as shown below:

$$i_a = -\frac{R}{L}i_a - \frac{1}{L}e_a + \frac{v_{pv}}{3L}(2S_a - S_b - S_c) \quad (4.59)$$

$$i_b = -\frac{R}{L}i_b - \frac{1}{L}e_b + \frac{v_{pv}}{3L}(S_a + 2S_b - S_c) \quad (4.60)$$

$$i_c = -\frac{R}{L}i_c - \frac{1}{L}e_c + \frac{v_{pv}}{3L}(S_a - S_b + 2S_c) \quad (4.61)$$

For high performance and reliable renewable energy systems provide high switching speed and compact design. Figure 4.6 presents the three phase MOSFET inverter as well as the DC source which is the PV system.

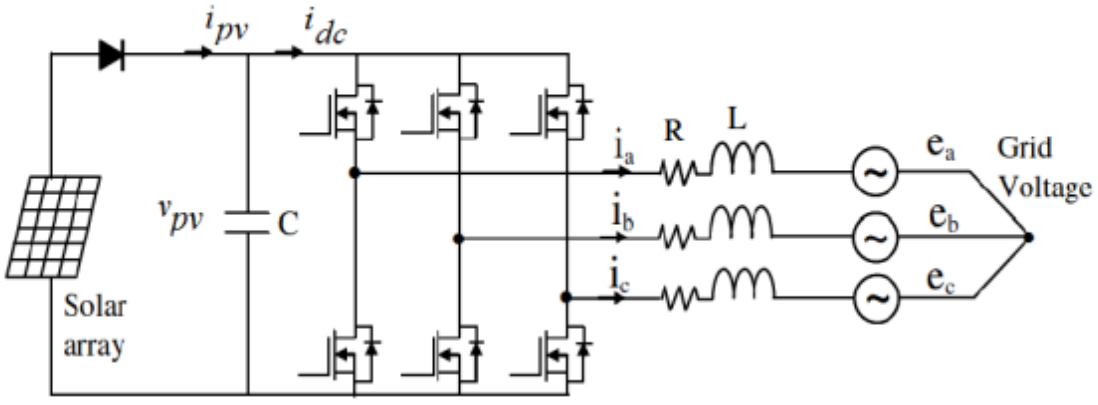

Figure 4.6: Tree-phase MOSFET inverter

Then , the voltage was obtained through:

$$i_{dc} = i_a S_a + i_b S_b + i_c S_c \quad (4.62)$$

$$\rightarrow v_{dc} = \frac{1}{C} i_{dc} - \frac{1}{C}(i_a S_a + i_b S_b + i_c S_c) \quad (4.63)$$

4.6.3 Three phase grid connected inverter synchronization

The Park transformation is used for two main reasons. First, the dimensions of the mathematical model of the inverter is too extensive. And last, facilitate the control process of the entire system. We opted for the synchronous reference frame due to the fact that it deals perfectly with non-ideal conditions of the grid.

4.6.4 Grid synchronization using PLL (phase lock loop) Technique

The PLL (Phase-Locked Loop) system acts like a tool that synchronizes an output signal with a reference input signal in both frequency and phase. The current control loop uses a PI (Proportional-Integral) controller, with the PLL providing the necessary signals. The PLL detects the phase, frequency, and voltage of the grid (Wang et al., 2021). It consists of phase detection and a loop filter, with the phase detection performed through an abc-to-dq transformation in a three-phase system. The loop filter controls system dynamics, and its bandwidth balances filter performance with response time. The quality of the PLL's lock and overall performance depend on the parameters of the loop filter (Yan & Xu, 2020). Figure 4.7 introduces the PLL control loop.

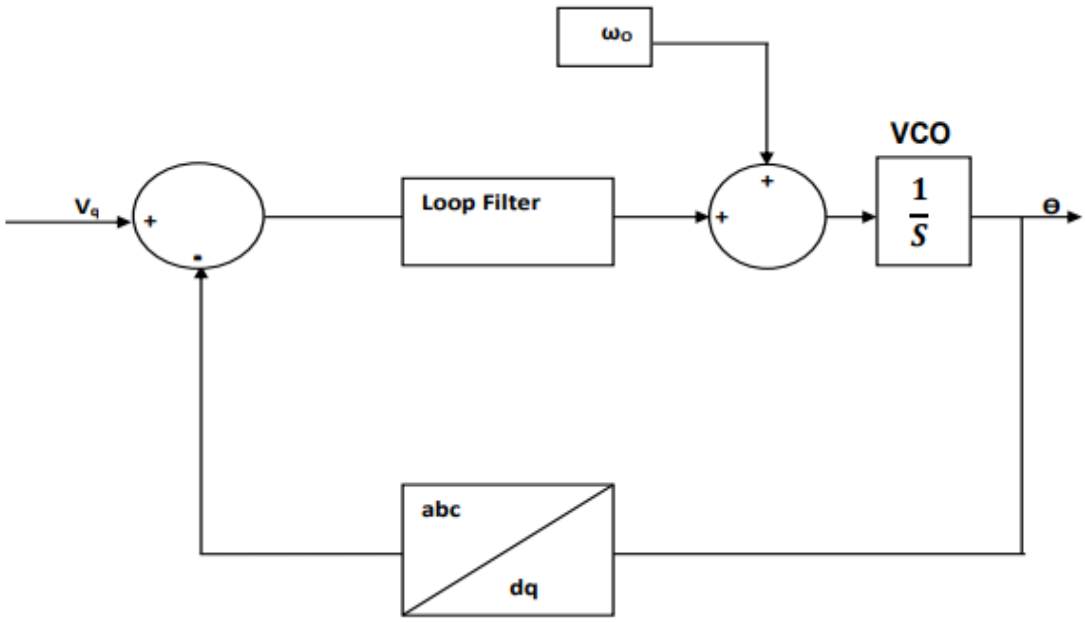

Figure 4.7: PLL control loop (Battacharya & Kumar, 2019).

4.6.5 Design of synchronous reference PLL

The performance of Phase-Locked Loop (PLL) systems in grid-connected applications is affected by distortions in the utility network. Fast dynamic designs enable quick synchronization, while slower dynamics can offer more precise control. For control synchronization, a moderate response is preferred, whereas rapid dynamics are ideal for grid monitoring and fault detection (Jayamaha & Rajapakse, 2019; Li et al., 2023). A tuning method for the dq-PLL controller is introduced, allowing adjustment of system parameters like settling time and damping ratio. This helps in designing either fast or slow dynamic systems based on application needs (Lacerda et al., 2020). The transfer function of the closed loop dq PLL system is represented as follow:

$$H_C(s) = \frac{\theta_O(s)}{\theta_i(s)} \tag{4.64}$$

$$H_C(s) = \frac{K_f(s)}{s+K_f(s)} \tag{4.65}$$

$$H_e(s) = \frac{\sigma(s)}{\theta_i(s)} \tag{4.66}$$

$$H_e(s) = \frac{s}{s + K_f(s)} \tag{4.67}$$

$$K_f(s) = K_p\left(\frac{1+s\tau}{s\tau}\right) \tag{4.68}$$

$$H_C(s) = \frac{2\zeta\omega_n s + \omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{4.69}$$

where

$$\omega_n = \frac{s^2}{s^2+2\zeta\omega_n s+\omega_n^2} \tag{4.70}$$

$$\text{where} \quad \omega_n = \sqrt{\frac{K_p}{\zeta}} \tag{4.71}$$

$$\zeta = \frac{K_p}{2\omega_n} \tag{4.72}$$

4.6.6 The three-phase connected VSI method

By and large, two existing procedures can be used, which are voltage strategy and current control technique. The DC precariousness is certainly not a significant issue when planning the grid connected inverter (GCI) as this can be overwhelmed by executing a quick current regulator(Han et al. 2019). The Air conditioner current via

the structure of control is limited. Likewise, the stage point and the plentifulness of the line current for the voltage controls the genuine and responsive force with regard to the PCC voltage. Thus, the (voltage source inverter) VSI is not under the danger of over-burden circumstances because of current control framework. The current regulator additionally has some different advantages such as strength against jumble in boundaries of the (voltage source inverter)VSI and AC framework, an exceptional dynamic of execution and sophisticated control accuracy (Zhang et al. 2022).

The proposed controller was built using a MATLAB script, so the purpose is to keep a constant voltage that matches the voltage of the grid under permissible limits. Table 4.2 shows the voltage controller parameters.

Table 4.8: Voltage Controller parameters

Parameters	**Description**
Voltage PV	PV array voltage (V)
Voltage Grid	Grid voltage (V)
Voltage Reference	Vgrid= Vref (same as the grid)
K_p = 0.1	Proportional gain
K_i = 0.01	Integral gain
T_s = 0.01	Sampling time (s)
T_{sim} = 10	Simulation time (s)

The voltage controller parameter ensures quick responses to errors as well as disturbances. Accurate current control is needed in order to reach efficient power conversion and components protection. Table 4.9 shows the main parameters used in the Current controller parameters script:

Table 4.9: Current controller parameters

Parameters	**Description**
Current PV	PV array current (V)
Current Grid	current voltage (V)
Current Reference	I_V= I_{Grid} (same as the grid)
K_p = 0.1	Proportional gain
K_i = 0.01	Integral gain
T_s = 0.01	Sampling time (s)
T_{sim} = 10	Simulation time (s)

The Current controller is key component of the solar system since it provides safety and reliability. Based on the error between the actual and desired current, it determines the output respond through the proportional gain.

4.7 Data analysis

Data preparation and pre-processing is one of the most crucial steps in data analysis. The raw data must be transformed into a format that is appropriate for additional analysis, such as the use of machine learning models, through a number of different processes. There are several steps in this process, starting with the preliminary data exploration and ending with the data being fed into the models(Tariq et al. 2022).

These procedures involve, among other things, formatting the data appropriately before using machine learning models (e.g., standardization/normalization) and handling inconsistent, duplicate, or missing values (e.g., imputation). This section discusses the primary pre-processing and data preparation stages and highlights how these steps were applied in this dissertation.

Data exploration: The initial investigation of data to comprehend the data and its properties and reveal as much information as possible is one of the first phases in data analysis. This stage can assist in defining the issue that needs to be resolved and offers guidance on how to do it.

Dealing with categorical and missing values: Numerical values are typically required as inputs for machine learning models. But before the study can continue, there are several categorical variables that must be addressed. For instance, n new variables with the values 0 and 1 can be used to represent a categorical variable with n possible values, signifying the existence or absence of an event.

The initial data exploration can identify missing data, and how the missing data is handled can have a significant impact on the analysis's subsequent steps. The data examples with missing values can be used for analysis (after the empty values have been substituted by other values in a process known as imputation) or discarded, depending on the application and the missing data. There are several approaches to missing data imputation. The most common value, the mean value of the relevant variable, the value of the previous data point, etc., can all be used to replace the missing value.

Numerical values are typically required as inputs for machine learning models. But before the study can continue, there are several categorical variables that must be addressed. For instance, n new variables with the values 0 and 1 can be used to represent a category variable with n possible values, indicating whether the event is present or absent.

Training and testing datasets: The dataset needs to be divided into a training and test set when some or all of the preceding procedures have been completed and it contains no missing or inconsistent values. This is necessary before using a machine learning model on the dataset. In this study, the predictive models were trained using 80% of the data, and their performance was tested using the remaining 20%.

4.8 Anomaly detection

Finding rare or infrequent occurrences that are expressed by data points that appear to act significantly differently from the bulk of the examined data is known as anomaly detection. Similar to novelty detection, anomaly detection identifies anomalous behaviour in data that should be avoided, whereas novelties, once identified, represent previously unnoticed behaviour that can be broadly regarded as a component of normal behaviour.

These abnormalities in the context of power systems may be linked to a rare operation that doesn't necessarily endanger the network or they may be tied to a really troubling network operation. For instance, they may indicate that a phase was abruptly shut down to zero or that there were atypical consumption patterns for a certain time of day or season as a result of an uncommon or unforeseen occurrence. It affects the supply's security and quality.

Both of the case studies in Chapters 6 and 7 centre on the third stage of the suggested technique, which entails determining the distribution network's fault or disturbance predictor. In order to create a prediction model using machine learning, this stage makes use of the data that was gathered in the first and second phases. This step includes choosing the input variables and machine learning approaches that are most suited for each application.

4.8.1 Gradient Boost

After considering several classification methods based on their performance, the choice fell on gradient boosting.

Boosting is a machine learning technique that applies a weak classifier successively to frequently altered versions of the data, combining numerous simple models into an ensemble that outperforms the individual models(Wang et al. 2021). The AdaBoost algorithm, short for adaptable boosting, was the first to successfully apply boosting. Gradient Boost (GB), an AdaBoost generalization, creates very basic, broad classifications using multiple single predictors (usually trees or rules) and then weights the outputs based on their expected error to create an overall classifier with more predictive power than any one of the component predictors(Maniraju & Kumaran 2024).

Beginning with a rudimentary classifier, which generates predictions that are just marginally better than guesswork, the classification process is carried out. The inaccuracy made by the model thus far is then predicted using later models. The models concentrate on the hard-to-predict data samples after being trained in a sequential manner(Aziz et al. 2020). These classifiers' goal is to minimize the loss—that is, the discrepancy between a training example's actual and predicted class values. Gradient descent is used in this strategy to minimize this loss. Each model is assessed and assigned a weight, and the weighted total of the models in the collection determines the classifier's overall prediction.

Though the specifics of implementation would vary depending on the kind of neural networks employed, the general notion of iteratively fitting models to rectify the flaws of the prior models would still apply. In general, the steps are as follows:

Initialization it helps to compute the mean of the targeted value as follow

$$F_0(x) = argmin \sum_{i=1}^{n} L(y_i, c) \tag{4.73}$$

Iterative Procedure for m=1

$$r_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F=F_m-1} \tag{4.74}$$

Output of the model Equation presents the formula used to display the final output

$$F_M(x) = F_0(x) + \sum_{m=1}^{M} vh_m(x) \tag{4.75}$$

4.8.2 Performance evaluation

Many performance measures are available in the field of data analysis that are used to evaluate the outcomes obtained from applying a machine learning algorithm to specific data, but not all of them are appropriate for all applications.

At times, evaluating a method's performance for a particular application might be done only based on its accuracy in categorization. Even while the classification accuracy can be high, there are situations in which the classifier may perform poorly. For this reason, this statistic alone could not always provide a whole picture.

The following dataset example clarifies this: the examples fall into one of two classes, with the bulk of the cases falling into one of the classes. A classification algorithm trained on a dataset such as this one might be able to identify most samples as belonging to the highly populated class. In such scenario, even though every passed example is assigned to the class with the highest population density, the classification

accuracy can still be high (in this case, 90%) when some additional examples are fed to the learnt model, of which only a small percentage (10% for example) belong to one class.

A table called a confusion matrix is frequently used to assess how well a classification model is performing. It displays an overview of the model's predictions in comparison to the dataset's actual ground truth labels. In the matrix, the examples in a predicted class are represented by each column, and the occurrences in an actual class are represented by each row. With TP as true positive which is how many occurrences are accurately anticipated to be in the positive class. With FP as false positive which is the quantity of cases that belong to the negative class but are mistakenly expected to belong to the positive class. With TN as true negative which is the quantity of cases that the negative class is accurately projected to include. With FN as false negative which is the quantity of cases that belong to the positive class but are mistakenly anticipated to be in the negative class.

4.9 Deep learning main elements

When applied to a layer output, an activation function (f) is a non-linear, differentiable function. Its primary function is to add non-linearity between layers so that the network as a whole is not factorized into a single linear process. Remember that the tensor algebra is associative. Most usual functions are the following:

- *represents the ReLu or Rectified Linear Unit*

$$f(x) = x' = \max(0, x) \tag{4.76}$$

- Equation (4.77) represents the *Sigmoid*

$$f(x) = x' = \frac{1}{1 + e^{x}} \tag{4.77}$$

- Equation (4.78) represents the *Tanh*

$$\frac{e^{-x} + e^{x}}{1 + e^{x}} \tag{4.78}$$

- Equation (4.79) represents the *Softmax*

$$f(X)_i = x'_i = \frac{e^{\frac{x_i}{T}}}{\sum_{j=1}^{D} e^{\frac{x_i}{T}}} \tag{4.79}$$

With x considered as a scalar and x' the output

- *Loss functions*

Typical loss functions for neural network training include:

- Mean square error

$$g(X_L, Y) = \frac{1}{Y}\sum_{i=1}^{Y}\left(x_{L,i} - y_i\right)^2 \tag{4.80}$$

- Cross entropy

$$g(X_L, Y) = -\frac{1}{Y}\sum_{i=1}^{Y} y_i log\left(x_{L,i}\right) \tag{4.81}$$

- Binary cross entropy

$$g(X_L, Y) = -\frac{1}{Y}\sum_{i=1}^{Y} y_i log\left(x_{L,i}\right) + (1 - y_i) log\left(1 - x_{L,i}\right) \tag{4.82}$$

- Hinge loss

$$g(X_L, Y) = -\frac{1}{Y}\sum_{i=1}^{Y} max\left(0, 1 - y_i . x_{L,i}\right) \tag{4.83}$$

The initial term for one of these losses was Mean Square Error (MSE). The L_2 distance between the intended target and the deep neural networ output is measured in a very intuitive manner. When the error is small, one of the main issues with employing MSE is that it tends to slow down the training process.

- *Layers:*

It is a combination of a linear and activation function f. So it generates an output X_{l+1} based on the variations of the bias and weights using the following formula:

$$X_{l+1} = f(h(X_l, W_l) + B_l) \tag{4.84}$$

with B_l as the bias and W_l as the weights.

- *Learning process:*

Reducing the loss of a specific architecture on the training set is the goal of the learning process. This is usually processed in two steps: feed forward (or inference), where the output is computed for each input, and back propagation, where the weights are updated.

- *Feed forward propagation:*

$$\bar{g} = \frac{1}{M}\sum_{m=1}^{M} g(X_L^m, Y^m) \tag{4.86}$$

- *Back propagation:*

$$w^{new} = w^{old} - \alpha \frac{\delta g}{\delta w^{old}} \tag{4.87}$$

4.10 Summary

This chapter presents the mathematical modelling of a fault detection algorithm and the design of a grid-tied PV system. The system uses LSTM neural networks to handle sequential data and detect temporal dependencies. The mathematical formulation starts with pre-processing voltage and current waveforms to extract relevant features and establishes an LSTM network architecture for short-circuit fault detection. Key equations governing the LSTM layers are derived and explained. The PV system design is modelled to simulate real-world operating conditions, including a 200 kW grid-connected array and its distribution-side components. Synthetic fault scenarios are integrated to replicate diverse conditions. This combination of algorithmic modelling and system design provides a robust framework for fault detection and classification, addressing critical challenges in PV system monitoring.

CHAPTER FIVE

RESULTS AND DISCUSSIONS

5.1 Introduction

This chapter presents the simulation results of fault detection and classification in grid-connected PV systems using deep learning methods, particularly LSTM networks, demonstrate the potential for enhanced accuracy and reliability. LSTM, a type of RNN, excels in learning temporal patterns, making it highly suitable for detecting and classifying dynamic faults in PV systems. To further improve the model's performance, the SMOTE has been innovatively applied. SMOTE addresses the issue of imbalanced datasets by generating synthetic fault data, ensuring the model accurately classifies both common and rare fault types. The combination of LSTM and SMOTE leads to a robust fault detection framework that not only improves prediction accuracy but also reduces false positives. The simulation results showcase the model's ability to detect and classify various faults, contributing to increased grid stability and system reliability in real-world PV applications.

The simulation results are presented as follows:

i. 200KW PV system development and interconnection to the grid
ii. Proposed LSTM networks proposed fault detection and classification strategy
iii. Proposed modified SMOTE technique for performance improvement
iv. Overview of a comparative analysis between MATLAB and PYTHON

5.2 Software Environments

This study used both MATLAB and PYTHON to design and evaluate a fault detection system for grid tied PV system. This integration enabled efficient simulation and intelligent analysis in order to provide more accurate results.

5.2.1 MATLAB Software Environment

MATLAB SIMULINK provides a flexible platform for power system modelling. Its block diagram framework makes it simple to create complicated systems and makes simulating different fault scenarios in the IEEE 13-bus system easier. This expertise is essential for creating fault detection algorithms and for comprehending the

Fault Modelling: SIMULINK is widely used to represent various faults, including open circuits, ground faults, and short circuits, in power systems. To better understand how these failures affect the power system and create effective detection techniques, researchers can model these defects.

5.2.2 Python Software Environment

The LSTM-based fault detection models' performance and effective implementation depend heavily on the deep learning frameworks we use. The frameworks that we have chosen offer strong functionality, user-friendliness, and wide community support. Here are thorough explanations of the main frameworks that were employed in the study:

- ✓ Keras: The main application programming interface was used because it supports time series data like voltage and current.
- ✓ Pytorch: provided the framework to custom LSTM architecture.
- ✓ Scikit: learn helped for the preliminary steps, pre-processing, feature and extraction. It easier to build the fault detection pipeline with their features.

5.3 200kW PV system development and interconnection to the grid

Several optimization strategies can be used to find the best position and size of DG units in a power distribution system. Utilizing MATLAB to integrate the power distribution system and optimize the positioning and dimensions of the distributed generation units is one method for achieving specific goals, including reducing losses, enhancing power quality, or increasing the share of renewable energy sources. A network of buses, each representing a node in the system where energy is created, consumed, or transferred, represents the power distribution system in the IEEE 13 Bus System. In this case, the PV system was installed close to the main load at the bus number that is supposed to supply.

The model is composed of a PV system of 200kW connected to the modified IEEE Bus with two stable loads. Figure 5.1 presents the 200 kW grid-tied PV systems with balanced loads. Then, after the fault implementations, the data is sent through a data acquisition process.

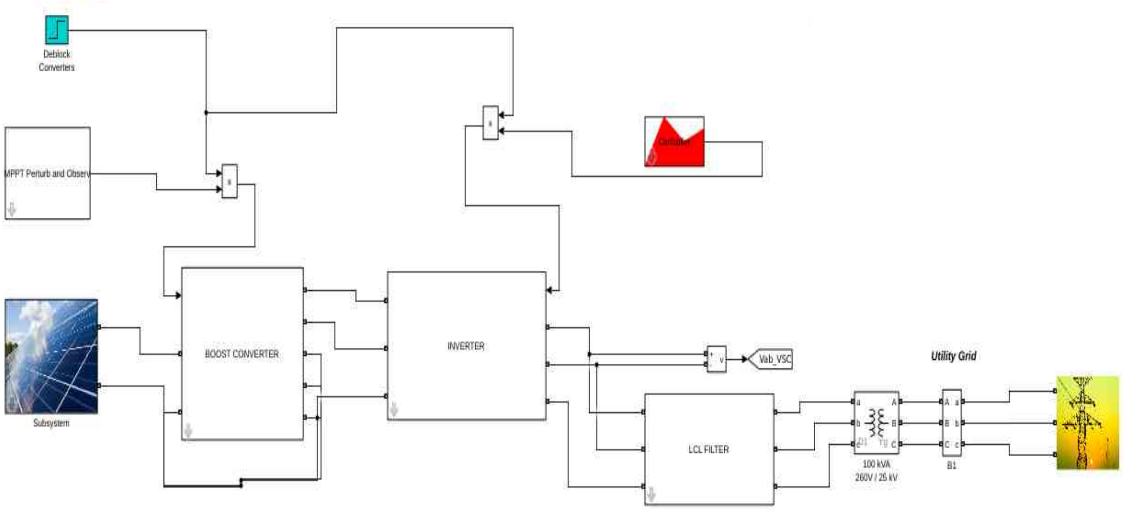

Figure 5.1: 200kW grid-tied PV systems.

The grid was actually made of an IEEE13 Bus system. Over the simulations, the loads have been modified from the actual Bus system, in order to test the overall system under different conditions.

The objective is to study the behaviour of the system under different conditions. The MPPT Algorithm presented in Appendix 3 allows the tracking of the maximum power.

5.4 Scenarios definition

In this section, three scenarios were proposed to assess the effectiveness of the fault detection model. At first, there is a case were the system is balanced, and one,two and three phase faults are introduced. Second scenarios the system is unbalanced and the same three types of short circuits faults are implemented. Voltage imbalance helps to see the resilience of the system. Then the third one, is a data augmentation approach calls SMOTE to resample the minimum data under unbalance voltage that are available and find similar faults in the same voltage range

5.4.1 Scenario 1: Balance three-phase system

Figure 5.2 provides the specification of the PV array. The MATLAB window showed how the PV data were computed with 66 and 10 parallel and series strings.

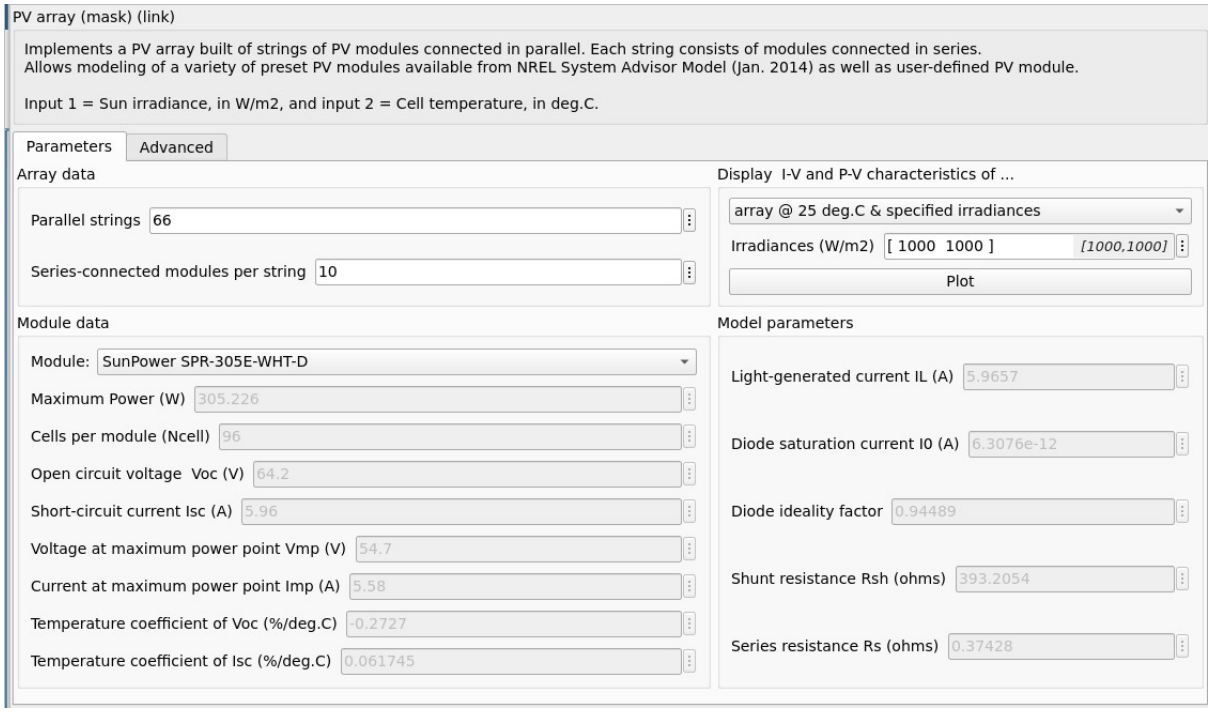

Figure 5.2: Specification of the PV array.

Figure 5.3 showed the PV characteristics curve. From this curve, the power and irradiance increased progressively until it reached 200kW then dropped to 0. The voltage and power and dependant on the irradiance power. As the irradiance move, voltage and irradiance moved as well.

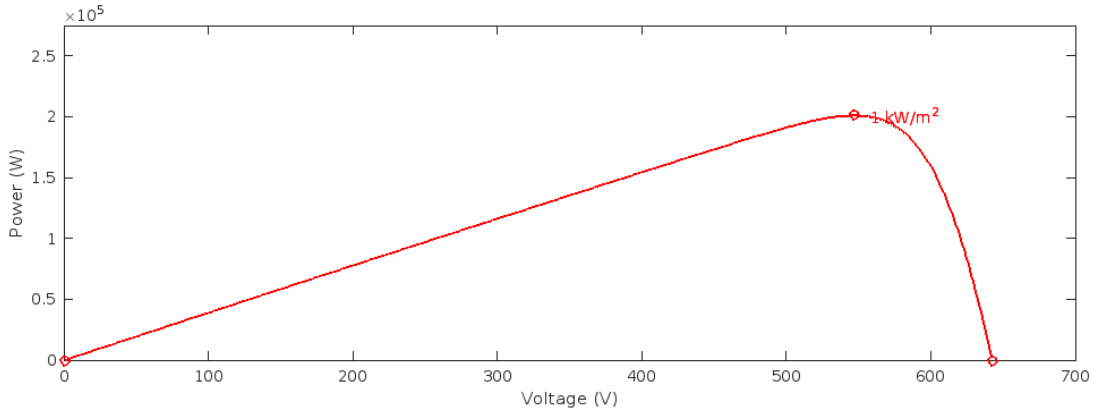

Figure 5.3. P-V Characteristics

The irradiation variations changed depending on the weather conditions, as depicted in Figure 5.4. The Module selected was a Sun Power SPR-305E-WHT-D. And a maximum power point of almost 180kW close to 200kW was expected from the PV system model.

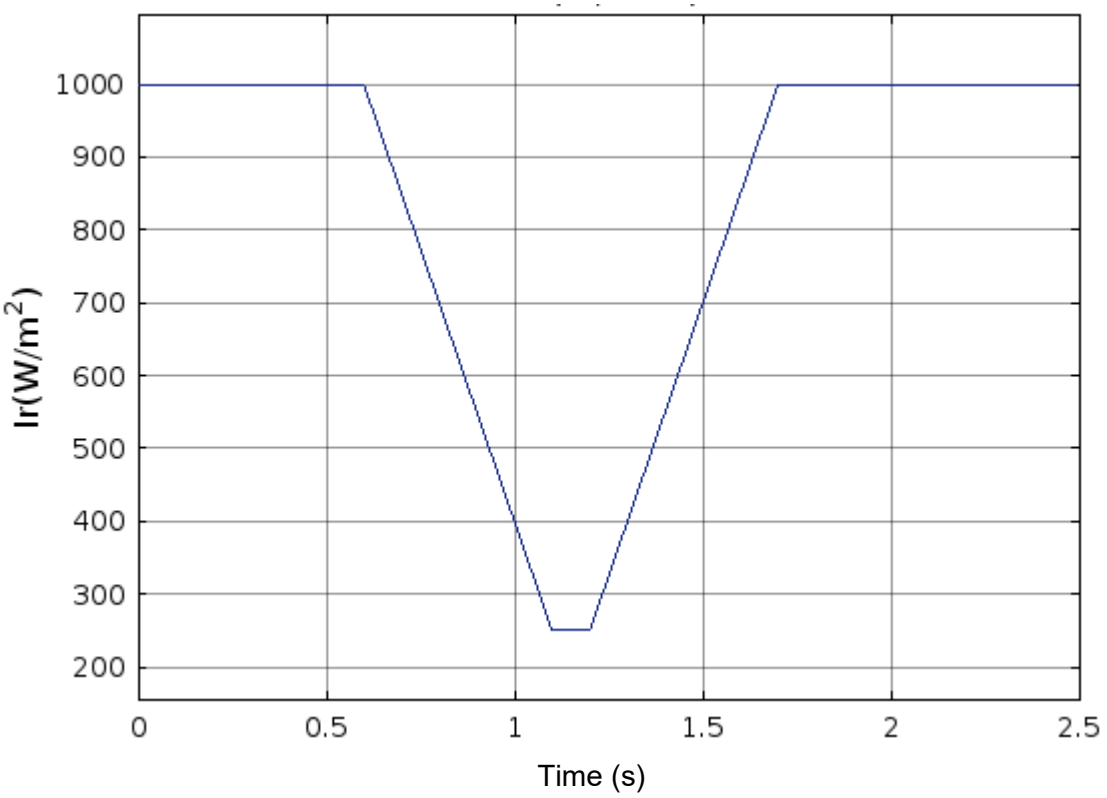

Figure 5.4: Irradiance variations.

5.4.1.1 Irradiance

The effectiveness of the studied PV system is obtained under variations of the irradiance**.** The simulation was carried out to analyse the performance of the PV system via changing the system temperature (T) from about 50 °C, while letting the solar irradiation (G) vary to 1000 [W/m2/m2]. However, at time t = 0.25 s, sudden variations in irradiance happened from 1000 to 250[W/m2/m2] under this situation; the subsequent output power (P), current (I), and voltage (V) of the adopted PV system are provided in Figure 5.4. From Figure 5.5, it can be observed that the PV output power increased in a range from 150kW, then went back up to be steady at almost 200kW MW at system temperature variations of about 50 °C. This demonstrates that the presented PV system is operating at its MPP during the sudden variation in irradiance

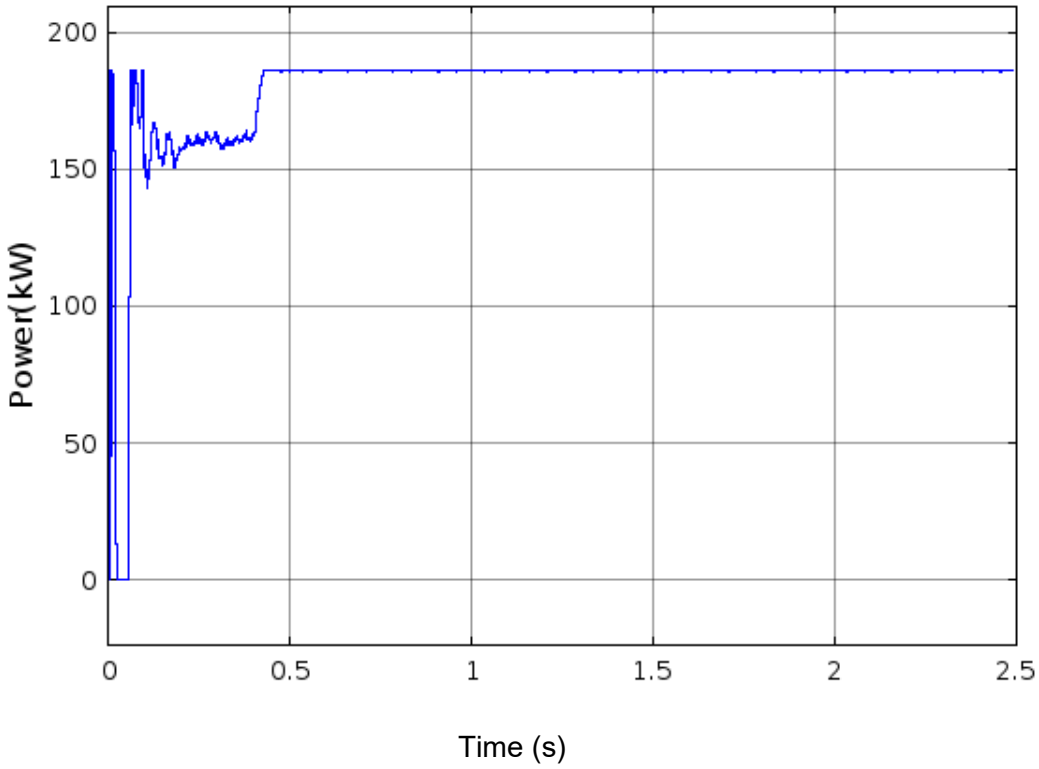

Figure 5.5: PV Power Output.

5.4.1.2 PV voltage

Figure 5.6 depicts the PV Voltage output; it can be seen that the waveform follows the flow of the irradiance. The same fluctuation that was observed in the Irradiance occurred then the MPPT implemented started to operate to bring the maximum voltage to 500V. Then from 0.2s it dropped as the irradiance dropped to 250[W/m2/m2], as soon as it went back up to 1000[W/m2/m2], the voltage followed and reached back the previous threshold.

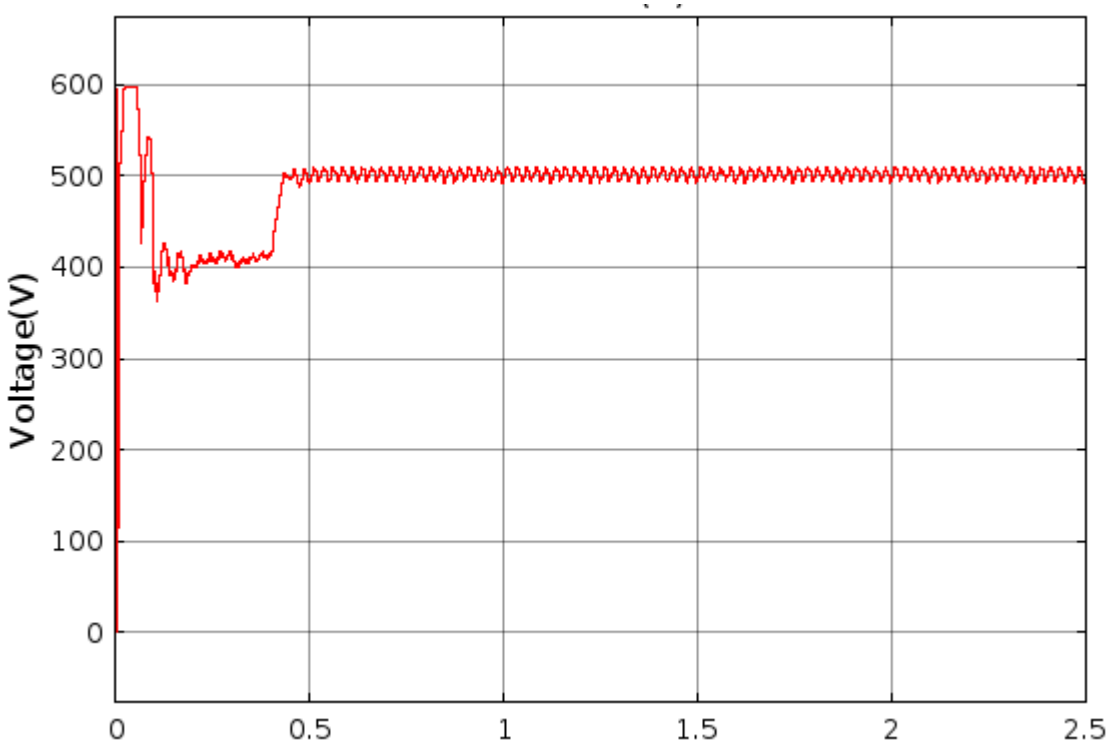

Figure 5.6: Voltage after MPPT

5.4.1.3 Inverter Voltage

The conversion of power was carried out from DC to AC using a three-phase inverter using pulse width modulation. The inverter voltage, once filtered, was sent to the step-up transformer to inject power into the medium voltage side of the IEEE 13 Bus. Figure 5.7 shows the Inverter Voltage output.

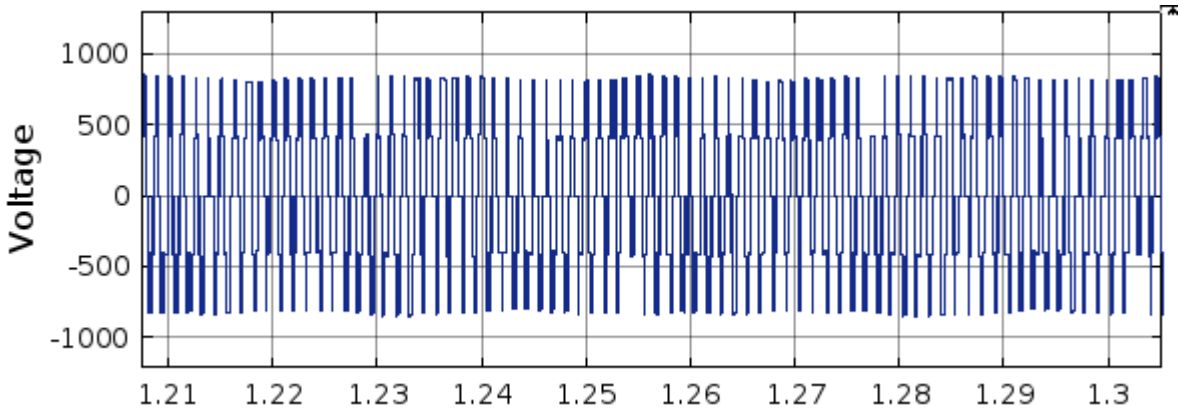

Figure 5.7: Inverter Output

Phasor current (Id) is depeicted in Figure 5.8. It had for objective to control the active power coming out of the 200kW PV system and inject the optimal amount to the IEEE 13 Bus. So, the Id has to match the Id, depending on the MPPT.

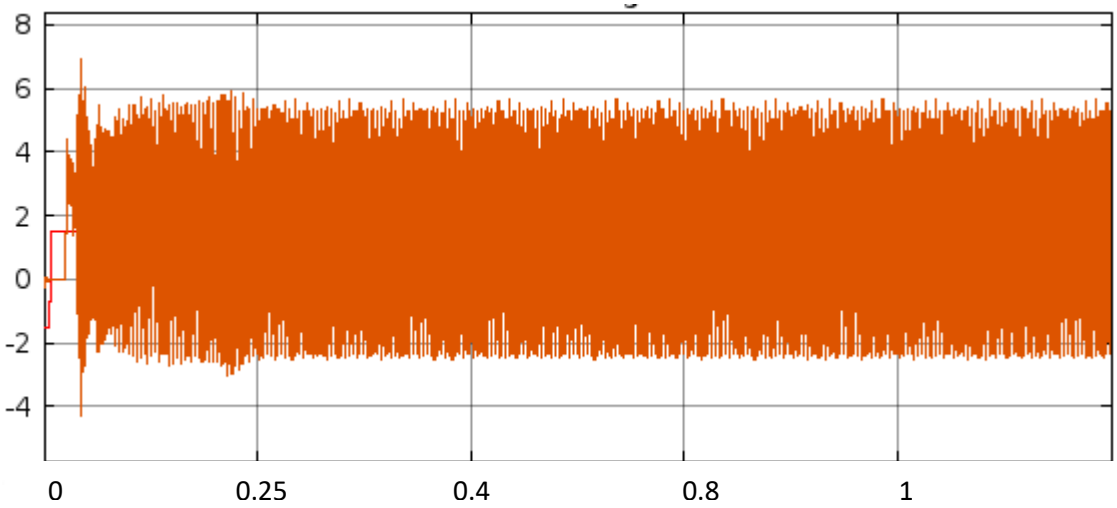

Figure 5.8: Phasor current (Id) current axis control

Phasor current (Iq) is presented in figure 5.9. Relatively closer to 0 since we are not injecting any reactive power to the IEEE 13 Bus. So, maintaining it to zero is because the objective is to get a power factor of 1. Depending on how the load changes, the Iq might change to negative if the grid requires substantial reactive power.

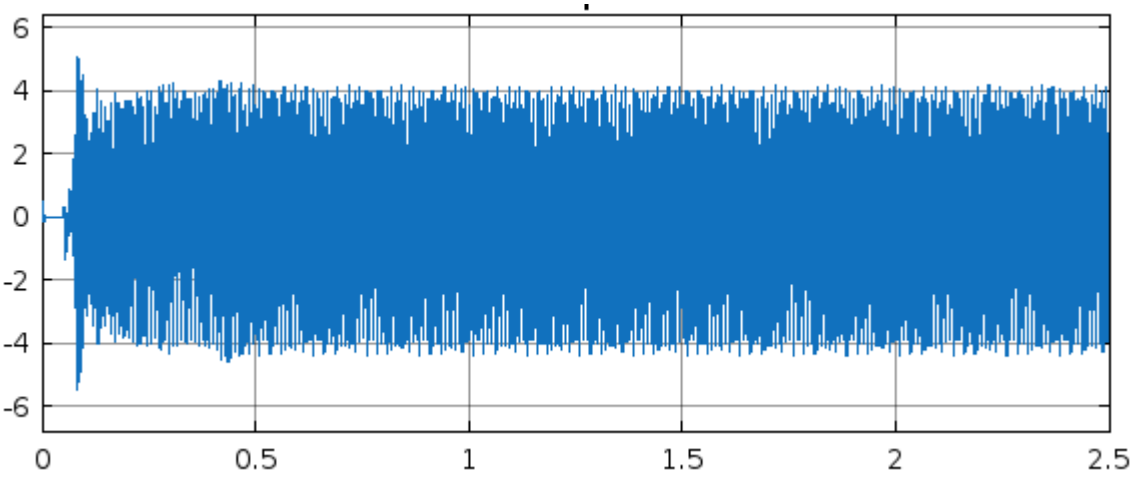

Figure 5.9: Phasor current (Iq) current axis control

5.4.2 Scenario 2: Unbalanced system

This case introduces an unbalanced three-phase voltage system, which causes overheating of equipment and possibly an increase in total harmonic distortion. It can be observed that Voltages fluctuate from Phase 1, Phase 2, and Phase 3, respectively, ranging from 2kV,4kV and 6kV. Figure 5.10 depicts the Grid Voltage under an unbalanced system.

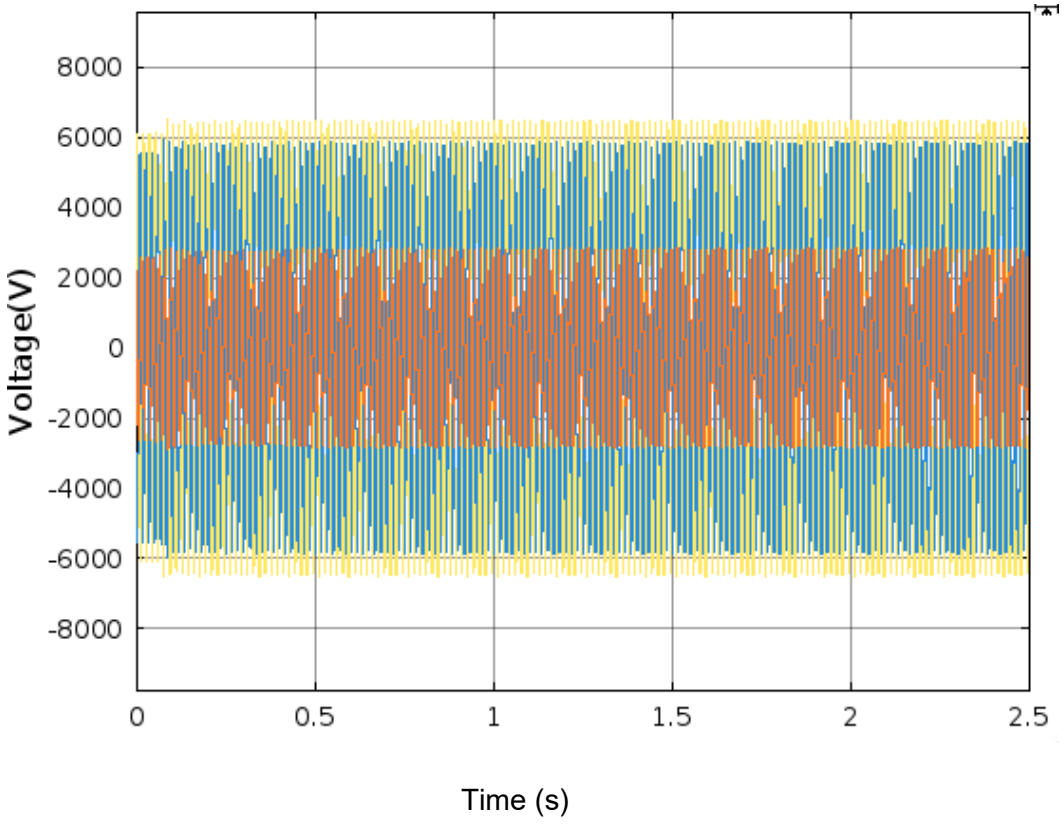

Figure 5.10: Voltage Output

5.4.2.1 Grid Voltage and Current

Figure 5.11 depicts the Voltage and grid current; it can be seen that the current lags behind the voltage, which means the grid phase voltage and grid phase current that must be injected into the grid in order for the grid-connected PV system to function at unity power factor.

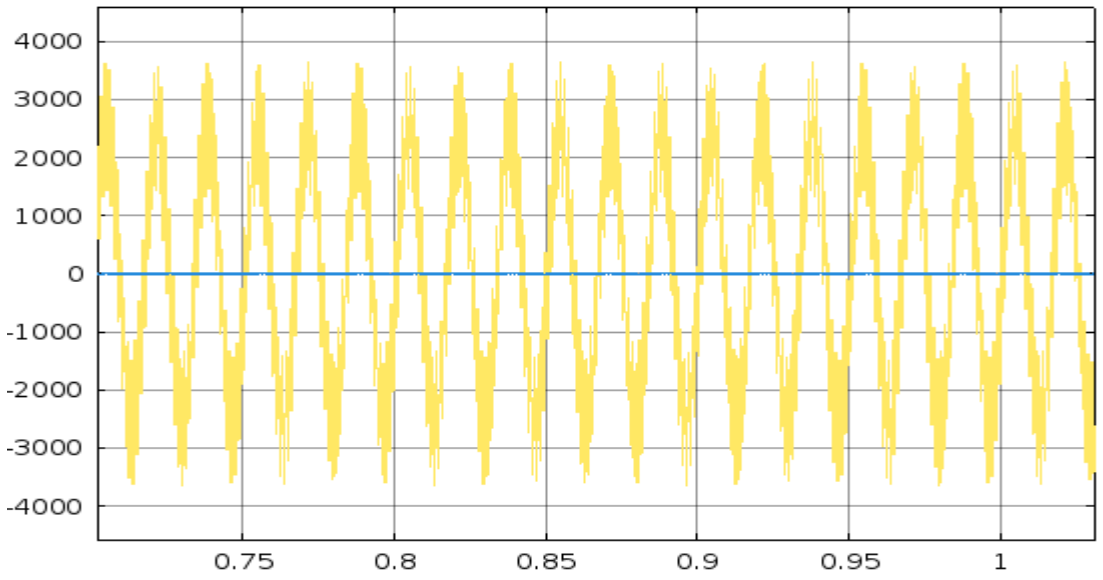

Figure 5.11: Voltage and Current phase

5.4.2.2 Control loop error

In Figures 5.12 and 5.13, I_d and I_q current axis control respectively, shows the tracking response of the controller; it shows that d and q control axis error oscillates between a value which is not that far from 0. This shows the tracking ability of the controller to provide the appropriate power to the grid.

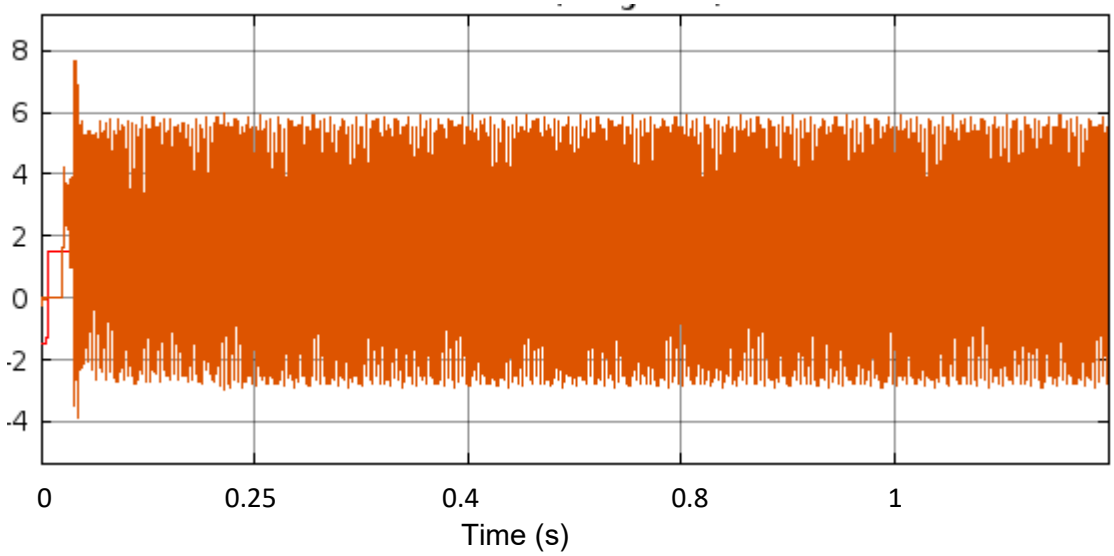

Figure 5.12: I_d current axis control

The reactive current components control the active power. Errors are processed through PI controllers to generate voltage and it adjusts the system accordingly. I_q also oscillate

around 0 with demonstrate an appreciable tracking response as seen in Figure 5.13 I_q current axis control.

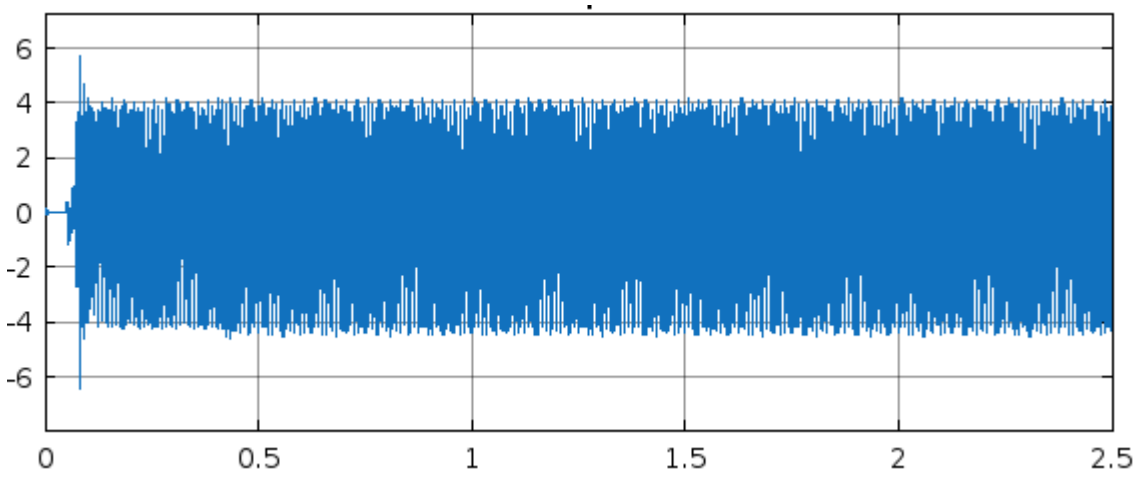

Figure 5.13: I_q current axis control

5.5 Proposed LSTM networks for a fault detection and classification strategy

This section provides the scope of the architecture of the neural network as well as the process from data acquisition to implementation of the LSTM model used to detect and classify the fault on the distribution side of the system.

5.5.1 Data acquisition from MATLAB to PYTHON

These steps are crucial for getting the data ready for model training and the following libraries were used to create features each and one of them with its own utility:

- ✓ **Pandas:** Pandas was capable of handling datasets comprising bus voltages, currents, power flows, and other pertinent information for the IEEE 13-bus system in an efficient manner.
- ✓ **Scikit-Learn:** For faults identification in such systems, the Python scikit-learn module is a vital resource since it offers strong tools for data analysis and machine learning. This library is used in the model to encode and normalize variable. These power flow equations may be used to generate features with scikit-learn, which we can then employ in machine learning models for fault detection.

- ✓ **Numpy:** These equations can be set up and solved iteratively with NumPy to find discrepancies that point to errors.

5.5.2 Data acquisition from MATLAB to PYTHON

At first, the IEEE 13 bus system is run in scenarios 1 and 2 with and without integration of the PV system at bus 633. Voltage and current measurements are the main data that the LSTM model that is going to be trained needs. Figures 5.14 and 5.15 present fault implementations.

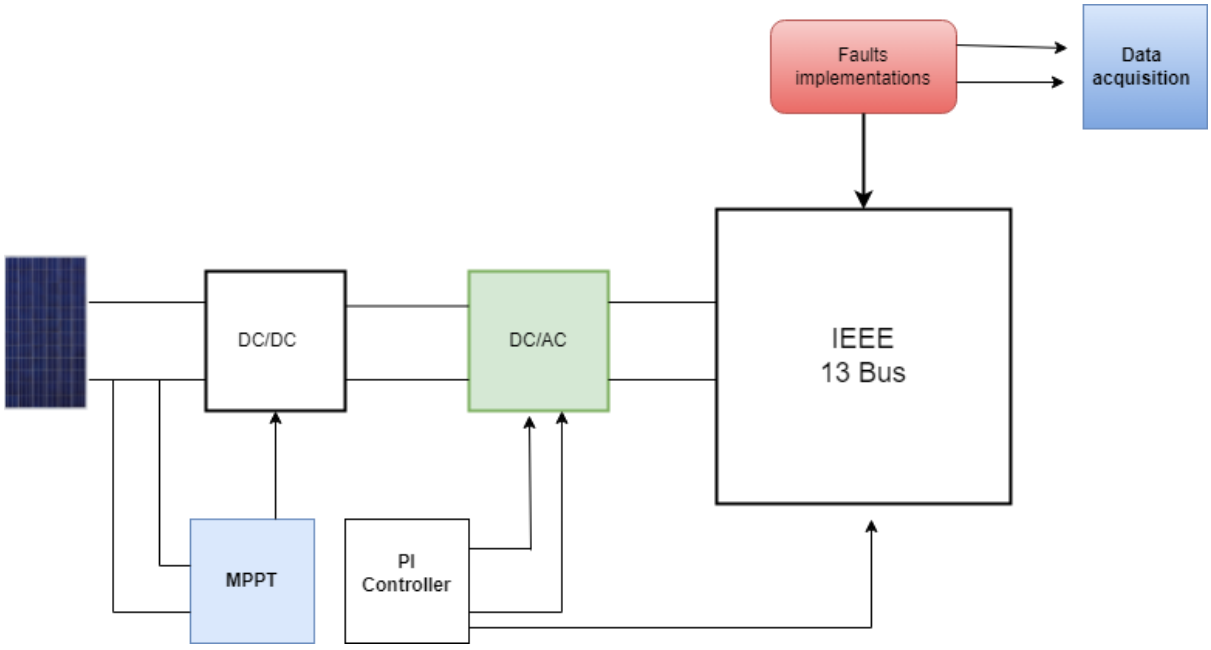

Figure 5.14: Faults Implementations

Data are acquired via voltage measurements then sent to a MATLAB sheet that is transformed to a variable. The same variable is used in Python tu run the faults detection algorithm. Data augmentation was used to have enough data for training. Figure 5.15 presented the data acquisition from MATLAB to Python.

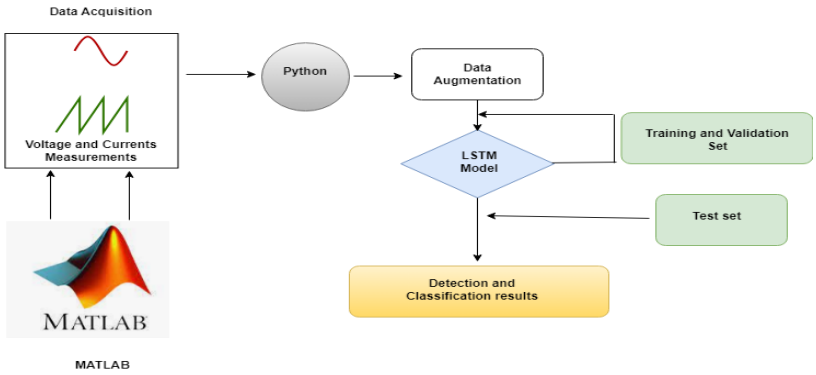

Figure 5.15: Data acquisition from MATLAB to PYTHON

Ideally once the data are measured with sensor they are sent to a Google Sheet API that was later on transform as a CSV file in Python, in order to facilitate the reading of the data. A Google sheet was created as presented in the chart flow depicted in Figure 5.2. It helps in order to save the data on the cloud and eventually for future studies to do real time simulations since MATLAB data will be sent in real time to the google sheet and those data will be used in the Python program designed for that purpose. In this study, the Data acquisition process was developed in such way that measurements can be accessed on the cloud without having to run extensive mat lab model although both are not connected in order to perform the simulations in real time. Faults are implementing in the IEEE 13 Bus system using the figure below. Once that is done, a fault is simulated based on Figure 5.16, which shows the fault implementation configuration. In the fault implementation, the model considers that when the voltage gets to zero and current exceed 10A there is a short circuit on the specific phase and between 380V to 400V for three-phase to ground faults.

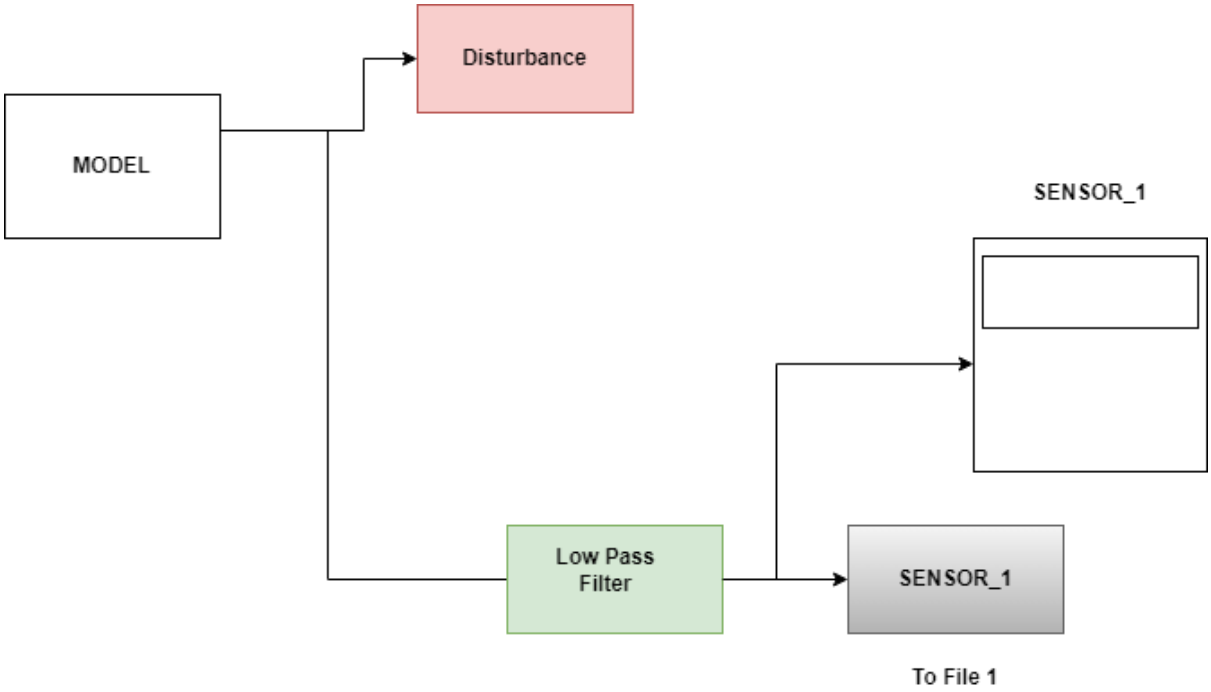

Figure 5.16: Faults Implementation configuration

5.6 Faults detection on the Distribution side (PV side)

In this section, the one, two and three phase faults are introduced into the grid tied PV system on the distribution side. Faults are runned under different epochs and neural network architectures.

5.6.1 Number of Epochs 500

Figure 5.17 depicts the best validation performance at epoch 155, it exceeded the goal of 0.01. With a gradient that kept up with the Mean square error with a value of $9.9.10e^{-8}$ as shown in Figure 5.18, the proposed model performs its tasks with fewer to no errors.

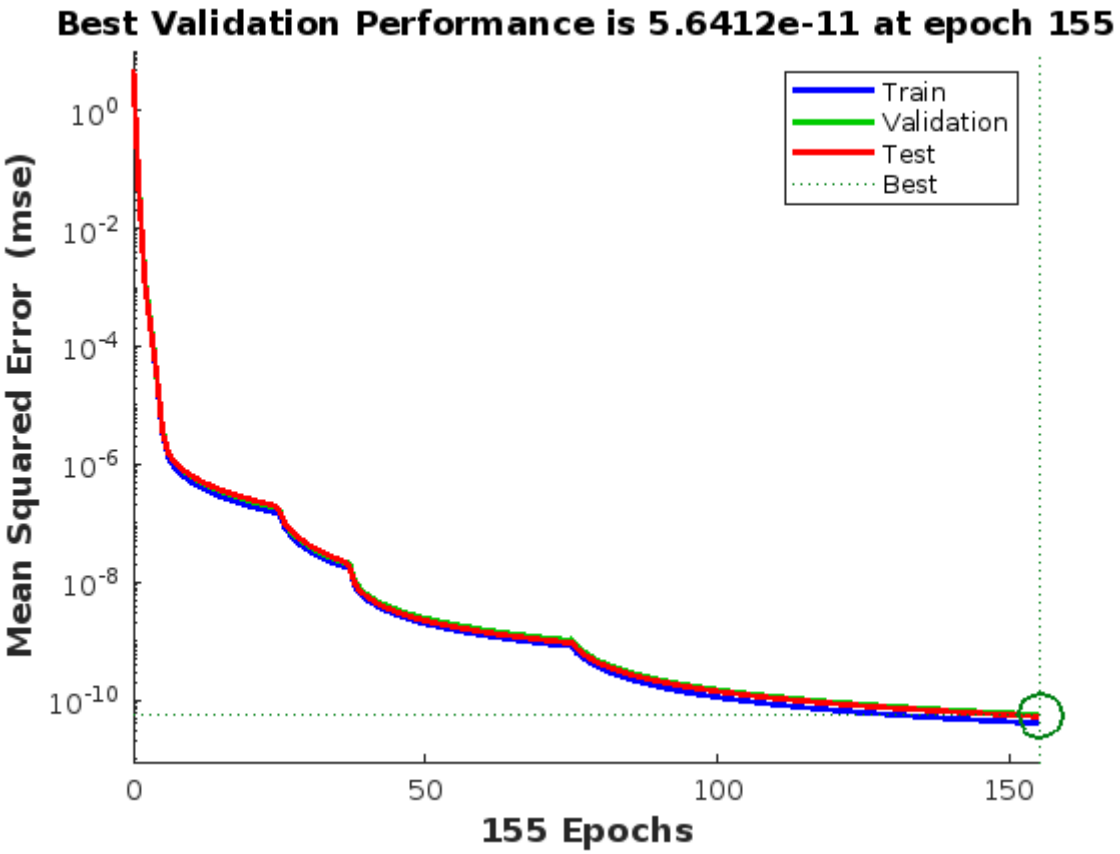

Figure 5.17: Best Validation Performance at epoch 155.

The validation failed to decrease at any time as it is constant to zero. Which shows no over fitting. The gradient is excellent since it does not exceed zero, the convergence is significant. The learning rate parameter keeps on decreasing; it presents ability to minimize the error. Figure 5.18 depicted the gradient performance.

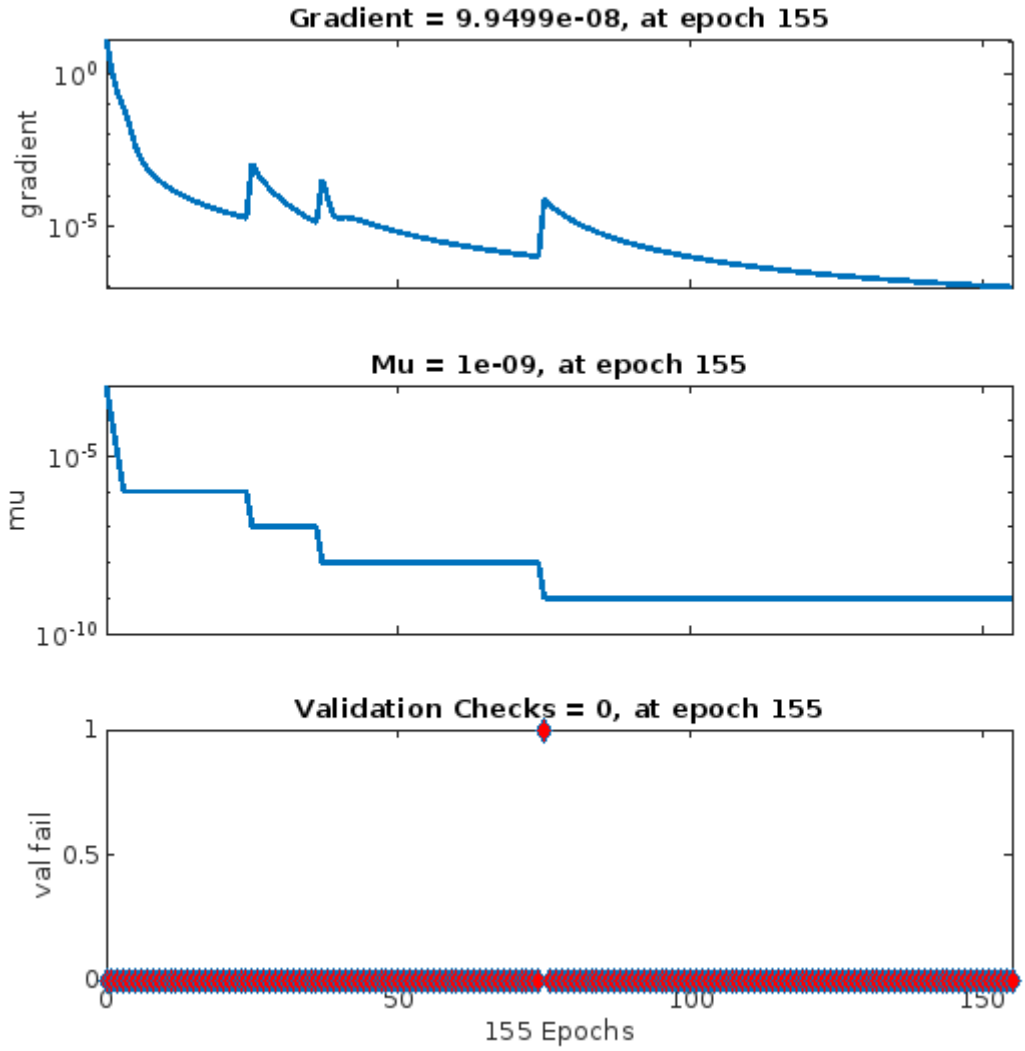

Figure 5.18: Gradient performance.

Figures 5.19 and 5.20, respectively, display the Regression ratio and the Confusion matrix 1. The training, validation and test data revealed to provide an exceptional regression outcome of 1 which means that all the true positive and false negative scenarios were processed correctly and the target was achieved.

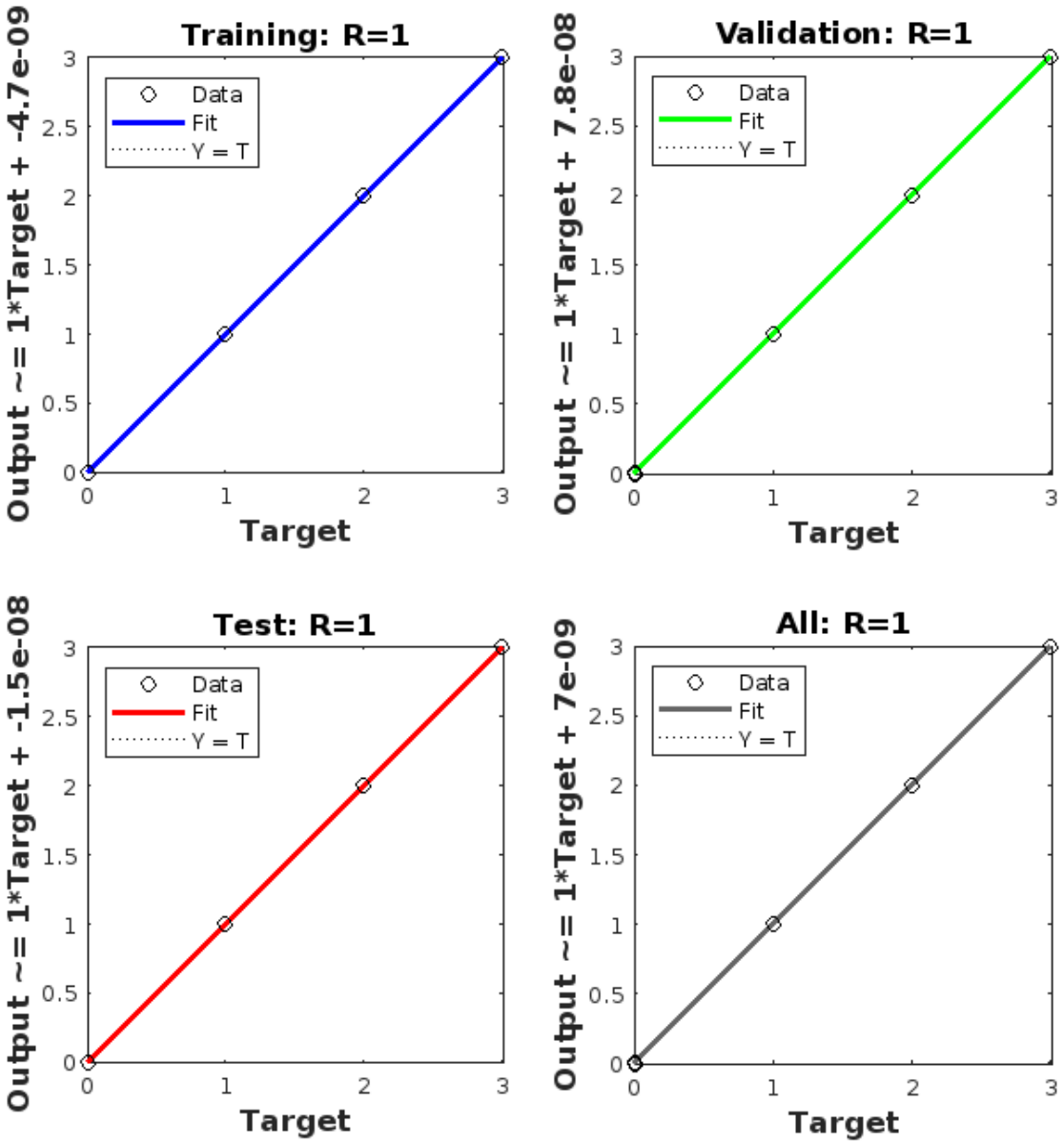

Figure 5.19: Regression ratio.

Confusion matrix allows a better readability of the data. A total of about a thousand faults have been simulated randomly. With 674 faults developed and runned that correspond to no faults case and have been detected perfectly. With the rest of non-faults cases been found as they are. Figure 5.20 depicts the Confusion Matrix 1.

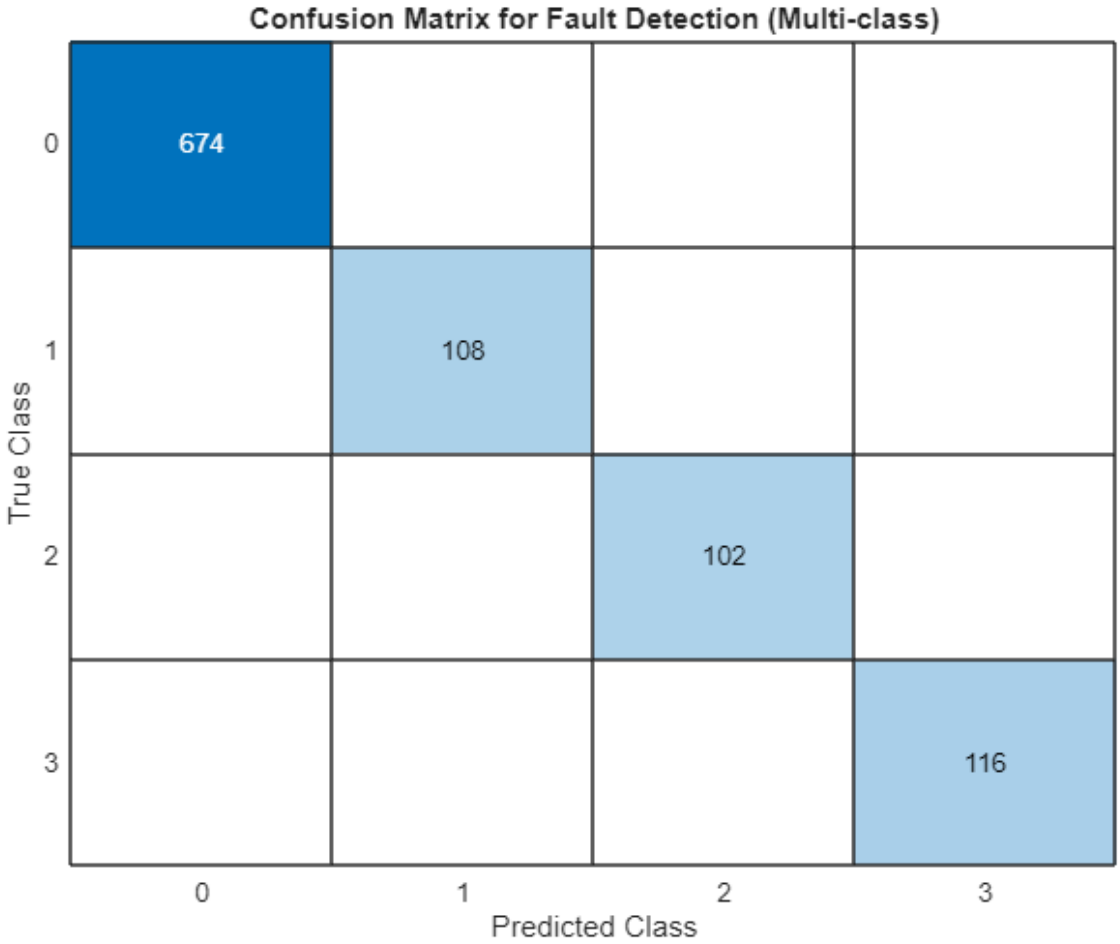

Figure 5.20: The Confusion matrix 1

5.6.2 Number of Samples 10000

The number of samples used in training a fault detection system's ANN significantly impacts its performance and output accuracy. A higher number of samples provides the ANN with a richer dataset, allowing it to learn more effectively from a variety of fault and non-fault scenarios. This diversity in data helps the network generalize better, reducing the chances of overfitting and improving its ability to accurately classify faults in new, unseen data. By the time the number of samples was set to 10000, the model was able to learn faster and more efficiently. A larger dataset enhances the ANN's training process by providing more examples to refine its weights and biases. This

results in a more reliable and robust fault detection system capable of making precise predictions in real-world applications. Figure 5.21 presents the Confusion matrix with 10000 samples and the performance with 10000 samples. The number of true positive scenarios has exponentially increased with 7023 scenarios of no fault detected classified correctly and to the performance of $6.58.11^{-8}$ at 161 epochs.

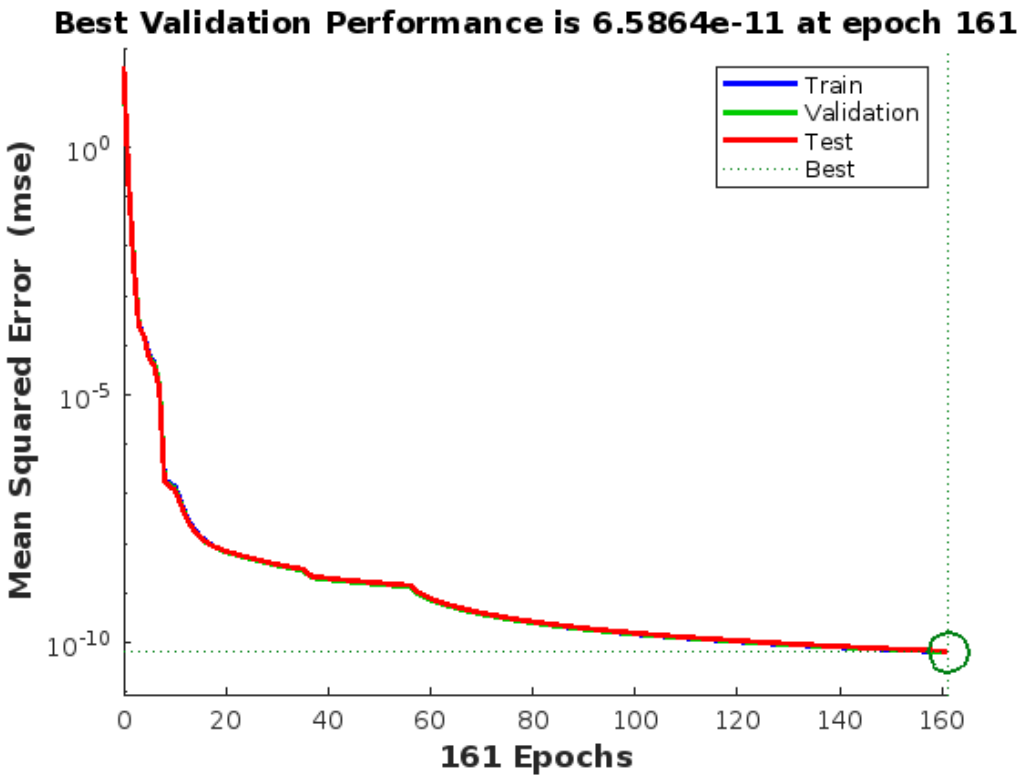

Figure 5.21: Best Validation Performance at epoch 161.

Figure 5.22 shows the Gradient performance 2, the gradient descent is about $9.91.10^{-8}$, and validation checks are at zero.. With error Histogram of performance that shows the accuracy of the proposed model.

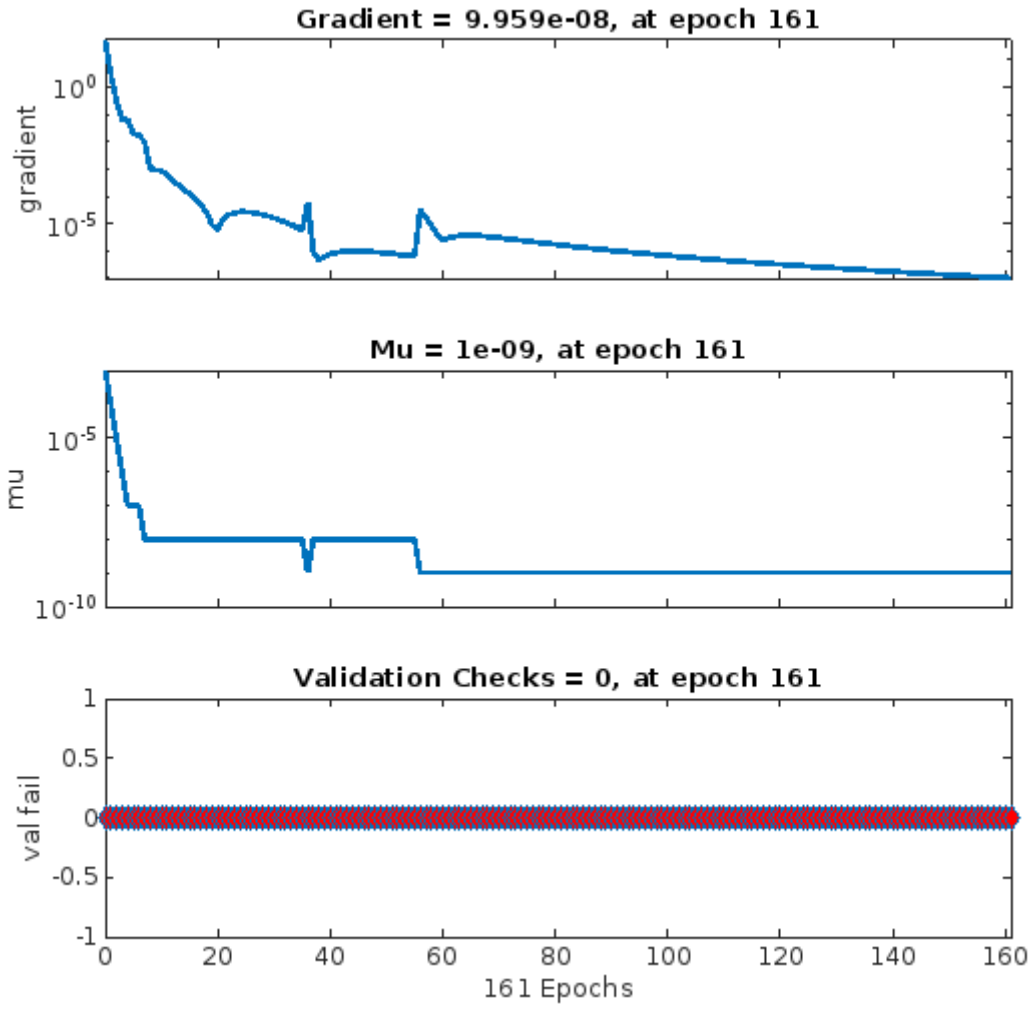

Figure 5.22: Gradient performance 2

Figure 5.23 relates to the Regression performance 2, it was seen that the regression was one which means that there is no error. The target has been achieved since the output of the testing, validation and training regression shows a 100% accuracy.

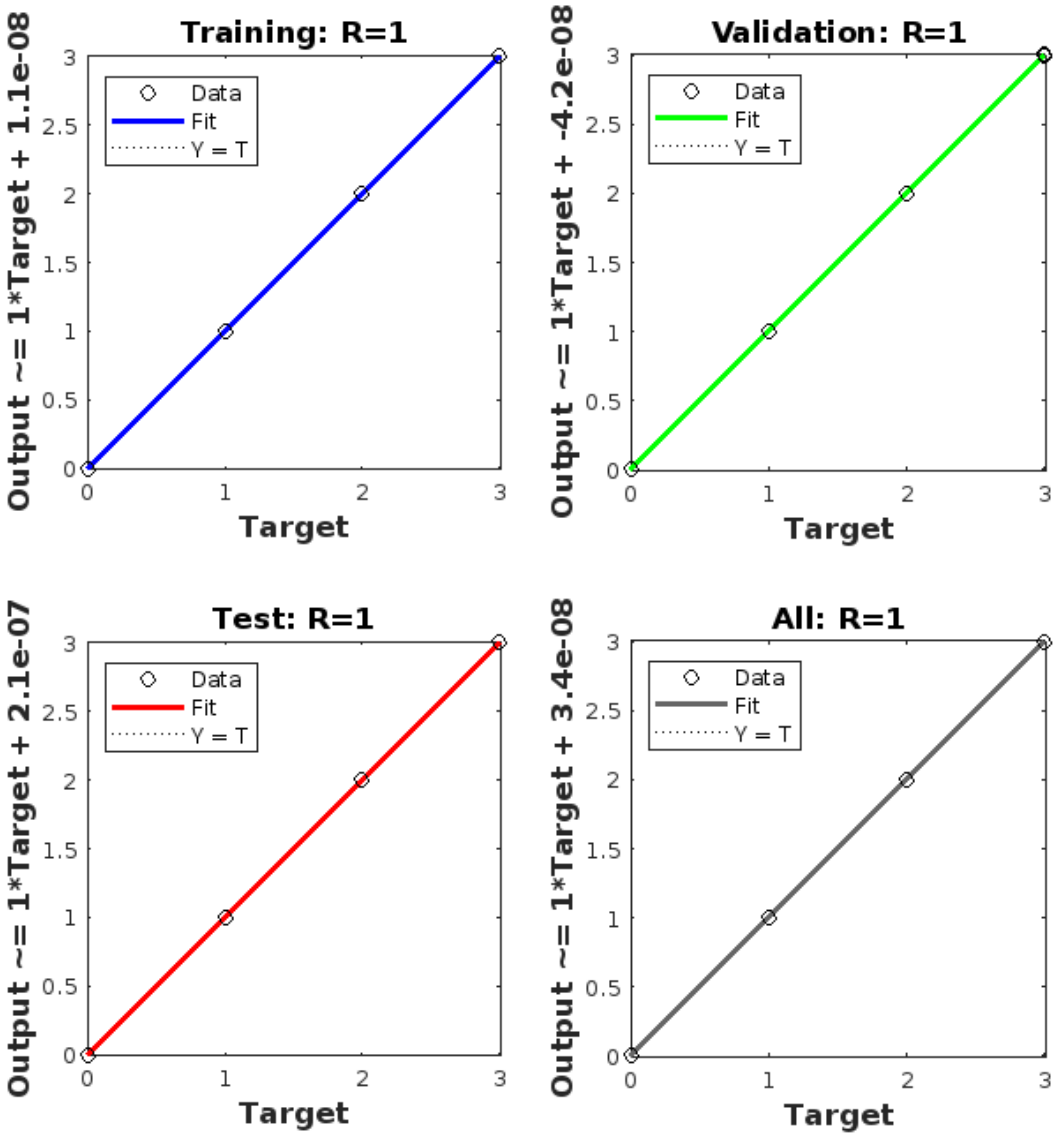

Figure 5.23: Regression Performance 2.

The Confusion matrix 2 shows some imbalance in the results as shown in Figure 5.24 since it simulated 2 times more true positive scenarios where there are no faults that the other three scenarios. Besides the true positive rates is highly satisfying with no faults predicted wrong

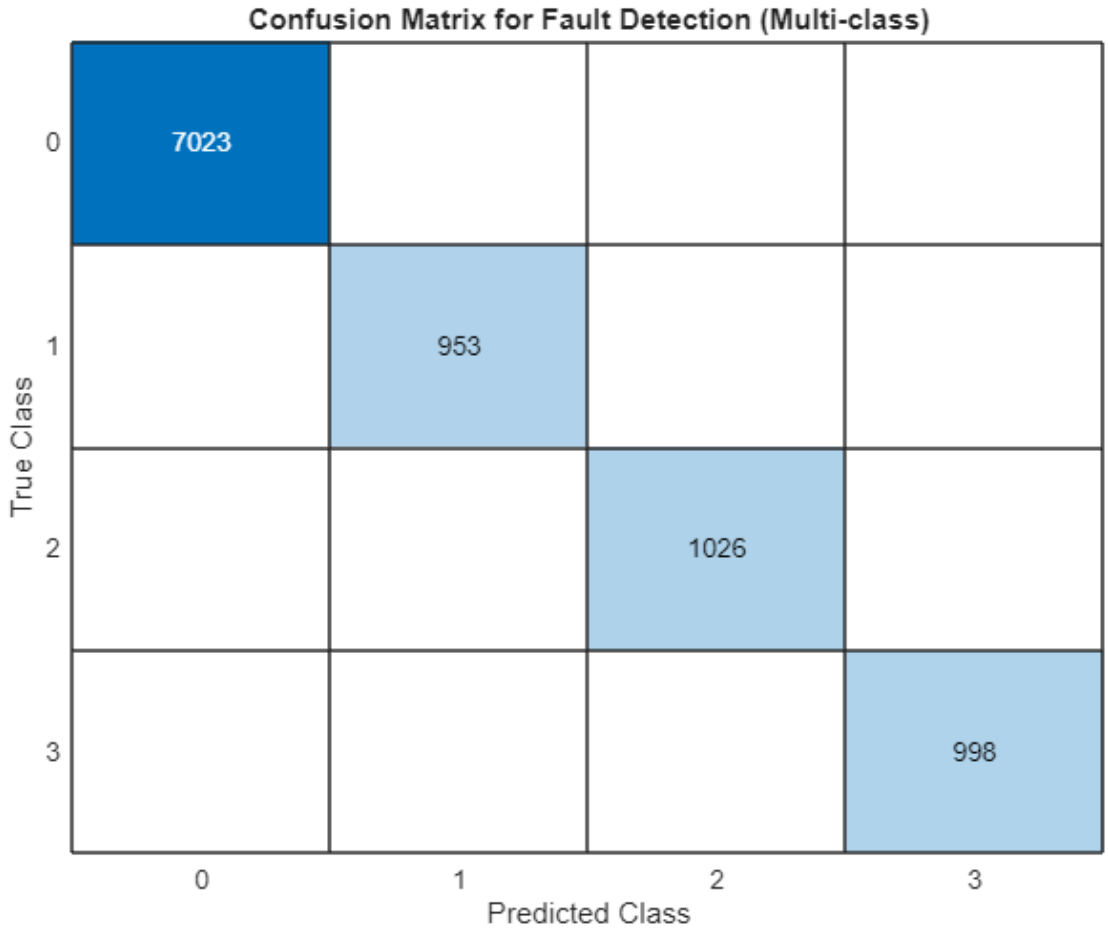

Figure 5.24: The Confusion matrix 2

5.7 Fault probability

Fault probability values played a significant role in determining the output when applied to fault detection systems. In such systems, the feed forward model is trained to classify different states, such as normal operation or various fault conditions. The fault probability values, which represent the likelihood of a particular fault occurring, serve as key input features during both the training and prediction phases. High fault probability values indicate a strong correlation between certain input signals (e.g., voltage or current deviations) and a specific fault, which helps the network adjust its weights and biases to improve fault classification accuracy. Reason why it has been set to 0.8.

As the feedforward processes these probability values, they influence the activation functions within the network layers, ultimately impacting the final decision or classification output. The higher the fault probability, the more likely the ANN is to predict the presence of a fault, thereby guiding the system toward taking appropriate corrective actions to maintain operational safety and efficiency. Figure 5.25 presents

the model performance with 500 samples; the best performance of the MSE was achieved at 284 epochs, exceeding the goal of 0.01.

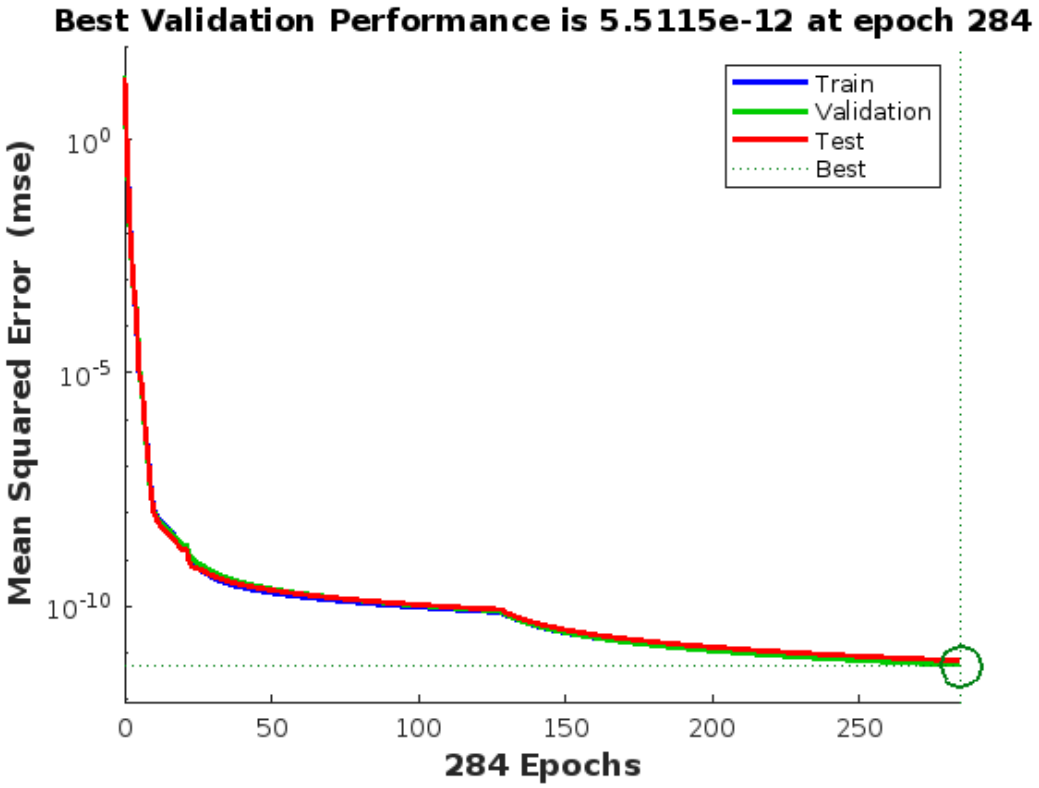

Figure 5.25: Best Validation Performance at epoch 284.

Figure 5.26 shows the Gradient Performance 3, the gradient descent is far less than 0, close $9.91.10^{-8}$, and validation checks are optimal with minimum failures. The learning rate parameters is below 0.001 which shows that the training has been properly executed.

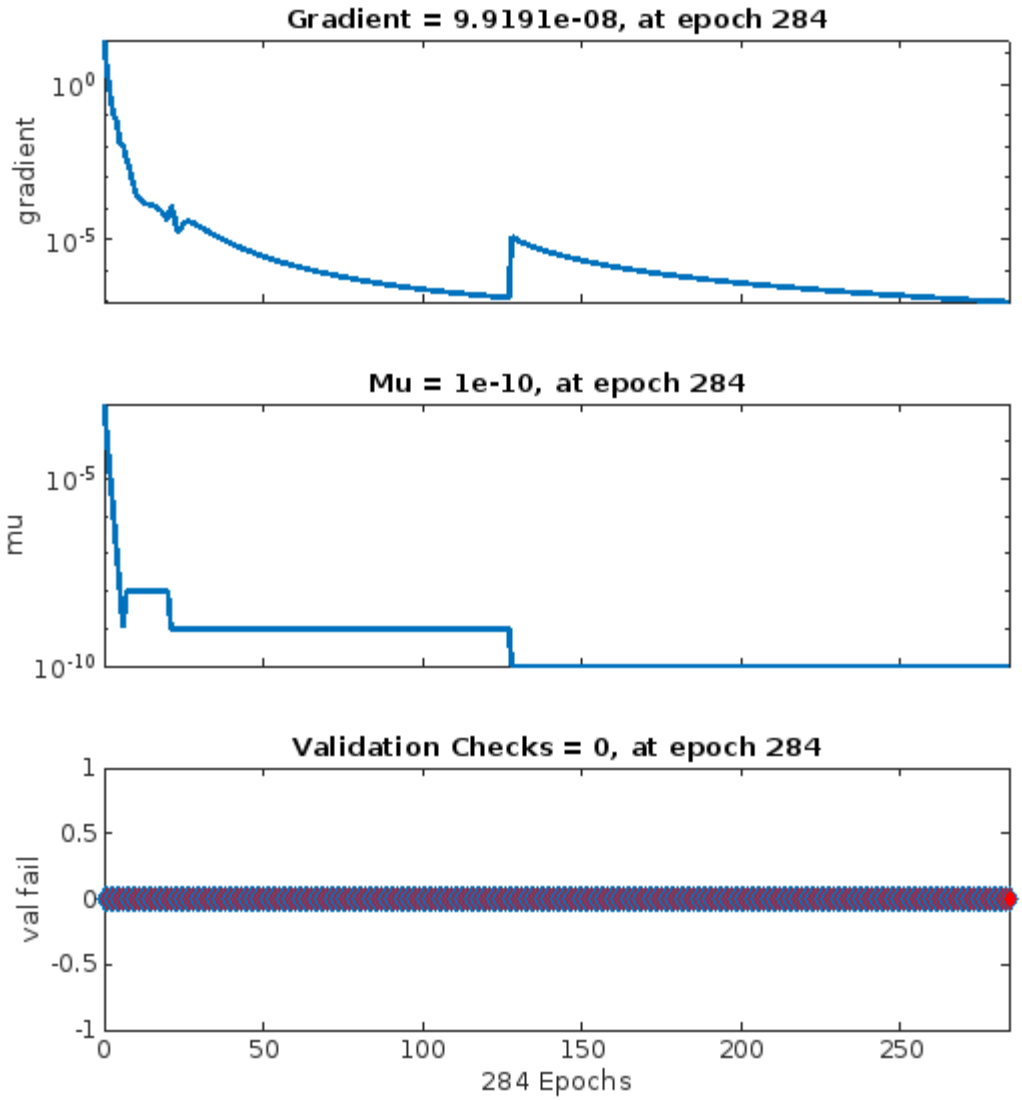

Figure 5.26: Gradient performance 3.

Figure 5.27 related to the Regression performance 3, it was seen that the regression was one which means that there is no error. The output of the validation and testing curve shows a 100% results between false negative and true positive.

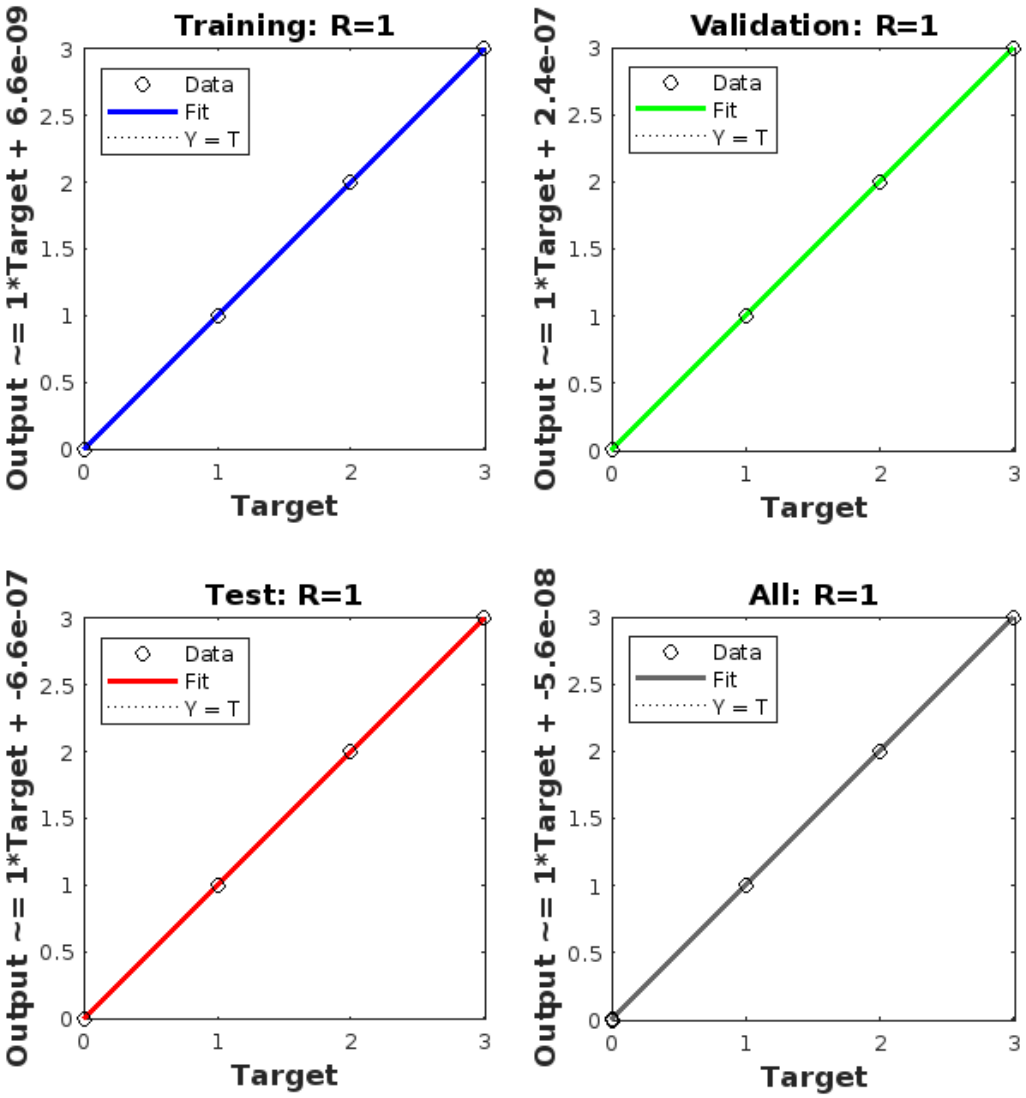

Figure 5.27: Regression Performance 3.

Figure 5.28 depicts the Confusion Matrix 3, it can be seen that the model brought great results with all faults and no faults scenarios detected and classified successfully. With three-phase to phase faults being the most faults detected and classified accurately

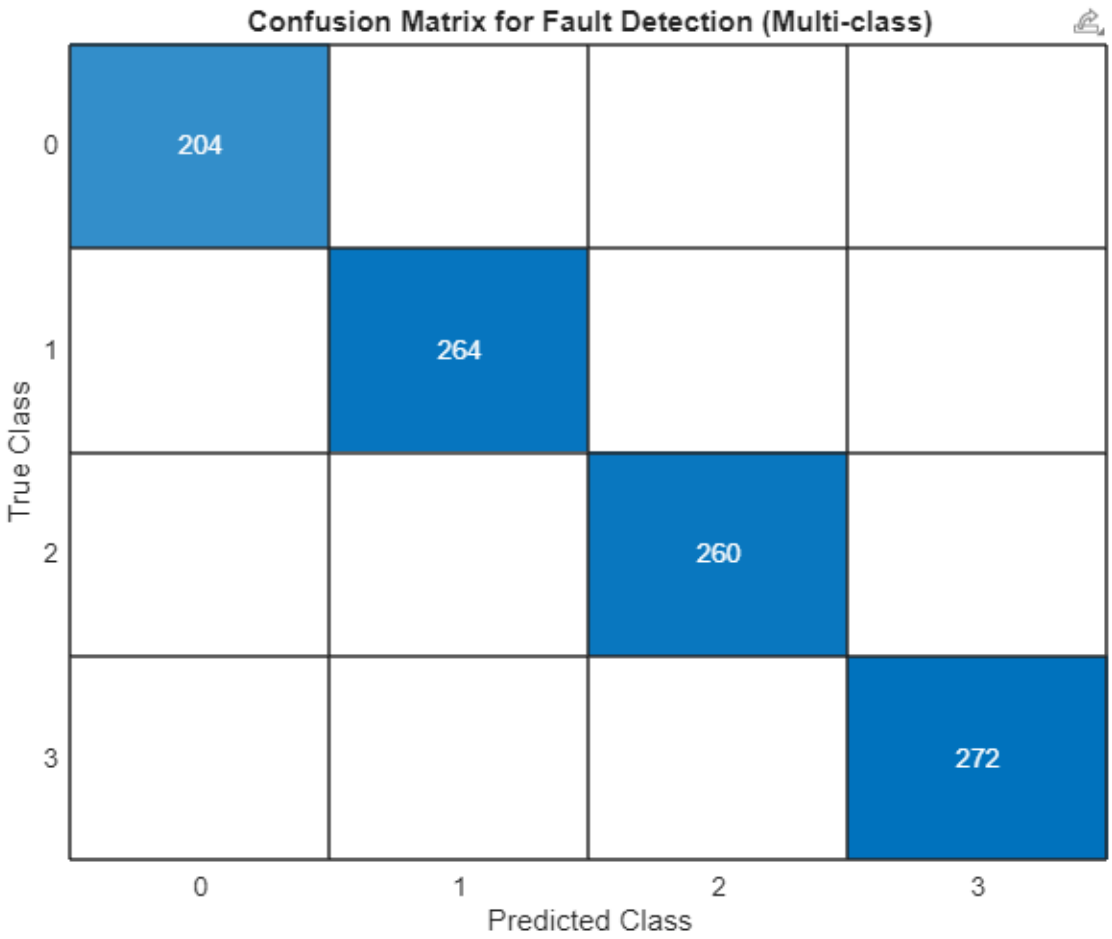

Figure 5.28: Confusion Matrix 3.

5.8 Algorithm changes

Table 5.1 presents the model performance with different algorithm functions, which are: Levenberg-Marquardt, Gradient Descent, Resilient propagation and BFGS Quasi Newton. Some performance metrics have been used to assess the proposed fault detection and classification approach for single, two and three-phase short circuits. The Mean square error actually increased along the training from Levenberg-Marquardt by about $5.5.10e^{-12}$ to 0.00017 with BFGS Quasi Newton. The objective was achieved with an error inferior to 0.01 and a regression ratio of 1. The Gradient kept up with the mean square error and increased as well from Levenberg-Marquardt to BFGS Quasi Newton. The Confusion Matrix has been split and presenting all the parameters which are true and false positive, false and true negative values. With a distribution of fault occurrence more or less equal for all the functions. The true positives values revealed to be high compared to the other scenarios which were equal to zero, proving that all the faults and no faults scenarios were classified and detected

successfully. Table 5.1 presented the Model performance with different algorithm functions.

Table 5.1: Model performance with different algorithm functions

Parameters		Levenberg-Marquardt	Gradient Descent	Resilient propagation	BFGS Quasi Newton
Mean Square Error		$5.5.10e^{-12}$	$1.20.10e^{-5}$	0.026	0.00017
Regression		1	1	0.99	0.99
Gradient		$9.9.10e^{-8}$	0.00008	$5.9.10e^{-6}$	0.0026
No Faults	True positive	204	197	199	193
	False negative	0	0	0	0
	False positive	0	0	0	0
	True Negative	0	0	0	0
Single-phase faults	True positive	264	259	290	256
	False negative	0	0	0	0
	False positive	0	0	0	0
	True Negative	0	0	0	0
Two-phase faults	True positive	260	264	240	281
	False negative	0	0	0	0
	False positive	0	0	0	0
	True Negative	0	0	0	0
Three-phase faults	True positive	272	280	270	270
	False negative	0	0	0	0
	False positive	0	0	0	0
	True Negative	0	0	0	0

5.9 Fault detection

Using the algorithm described in Appendix 1, the fault detection algorithm has been implemented. Figure 5.29 shows the Sequential Model Network 1-50-100 used in the fault detection model. The computation has been segmented with two different scenarios where the number of epochs and LSTM varied over time from 50, 100 epochs to 100 and 200 LSTM. To evaluate how the mean square error will change depending on the scenario.

```
Model: "sequential_8"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_12 (LSTM)              (None, 100)               40800

 dense_7 (Dense)             (None, 1)                 101

 repeat_vector_6 (RepeatVec  (None, 20, 1)             0
 tor)

 lstm_13 (LSTM)              (None, 20, 100)           40800

 time_distributed_5 (TimeDi  (None, 20, 1)             101
 stributed)

=================================================================
Total params: 81802 (319.54 KB)
Trainable params: 81802 (319.54 KB)
Non-trainable params: 0 (0.00 Byte)
```

Figure 5.29: Sequential Model Network 1-50-100

This network has a total number of 203 neurons. The target Mean Square error was less than 0.1. The training algorithm was Stochaistic Gradient descent. The activation function used were the Rectifier Linear Unit (Relu) and Sigmoid. The structure of the LSTM model given by figure. The figure shows the computation of the number of epochs. It can be seen that the best performance was validated around the from the first 10. The model was tested with different numbers of neurons and epochs to see where to get the best performance. A total of 81822 and 323622 parameters have been trained as presented in Figure 5.29. The computation was stopped at 100 and 200 epochs for some since the model presented a convergence earlier in the simulation so they were no point to add more epochs. The Network presented an

acceptable MSE which is less 0.01. The model managed to achieve an acceptable ROC which is above 50% as presented in Figure 5.30.

```
Epoch 1/50
1/1 - 5s - loss: 0.1401 - val_loss: 4.5594e-06 - 5s/epoch - 5s/step
Epoch 2/50
1/1 - 0s - loss: 0.1388 - val_loss: 1.8452e-05 - 44ms/epoch - 44ms/step
Epoch 3/50
1/1 - 0s - loss: 0.1376 - val_loss: 4.7592e-05 - 42ms/epoch - 42ms/step
Epoch 4/50
1/1 - 0s - loss: 0.1364 - val_loss: 8.9927e-05 - 43ms/epoch - 43ms/step
Epoch 5/50
1/1 - 0s - loss: 0.1352 - val_loss: 1.4747e-04 - 45ms/epoch - 45ms/step
Epoch 6/50
1/1 - 0s - loss: 0.1339 - val_loss: 2.2293e-04 - 44ms/epoch - 44ms/step
Epoch 7/50
1/1 - 0s - loss: 0.1326 - val_loss: 3.1843e-04 - 42ms/epoch - 42ms/step
Epoch 8/50
1/1 - 0s - loss: 0.1312 - val_loss: 4.3681e-04 - 47ms/epoch - 47ms/step
Epoch 9/50
1/1 - 0s - loss: 0.1299 - val_loss: 5.8154e-04 - 42ms/epoch - 42ms/step
Epoch 10/50
1/1 - 0s - loss: 0.1285 - val_loss: 7.5683e-04 - 41ms/epoch - 41ms/step
Epoch 11/50
1/1 - 0s - loss: 0.1270 - val_loss: 9.6805e-04 - 43ms/epoch - 43ms/step
Epoch 12/50
1/1 - 0s - loss: 0.1255 - val_loss: 0.0012 - 43ms/epoch - 43ms/step
Epoch 13/50
1/1 - 0s - loss: 0.1239 - val_loss: 0.0015 - 45ms/epoch - 45ms/step
Epoch 14/50
1/1 - 0s - loss: 0.1223 - val_loss: 0.0019 - 54ms/epoch - 54ms/step
Epoch 15/50
1/1 - 0s - loss: 0.1206 - val_loss: 0.0023 - 47ms/epoch - 47ms/step
Epoch 16/50
1/1 - 0s - loss: 0.1189 - val_loss: 0.0029 - 47ms/epoch - 47ms/step
Epoch 17/50
1/1 - 0s - loss: 0.1171 - val_loss: 0.0035 - 43ms/epoch - 43ms/step
Epoch 18/50
1/1 - 0s - loss: 0.1152 - val_loss: 0.0043 - 47ms/epoch - 47ms/step
Epoch 19/50
1/1 - 0s - loss: 0.1133 - val_loss: 0.0052 - 45ms/epoch - 45ms/step
Epoch 20/50
1/1 - 0s - loss: 0.1113 - val_loss: 0.0064 - 44ms/epoch - 44ms/step
Epoch 21/50
1/1 - 0s - loss: 0.1092 - val_loss: 0.0079 - 42ms/epoch - 42ms/step
Epoch 22/50
1/1 - 0s - loss: 0.1071 - val_loss: 0.0097 - 42ms/epoch - 42ms/step
```

Figure 5.30: Computation Network 1-50-100

Validation loss accuracy is increasing which means that there is no overfitting and gives comprehensive overview of the training set. The loss curve follows the validation , explaining a successful learning. Figure 5.31 shows the Loss and validation accuracy network 1-50-100.

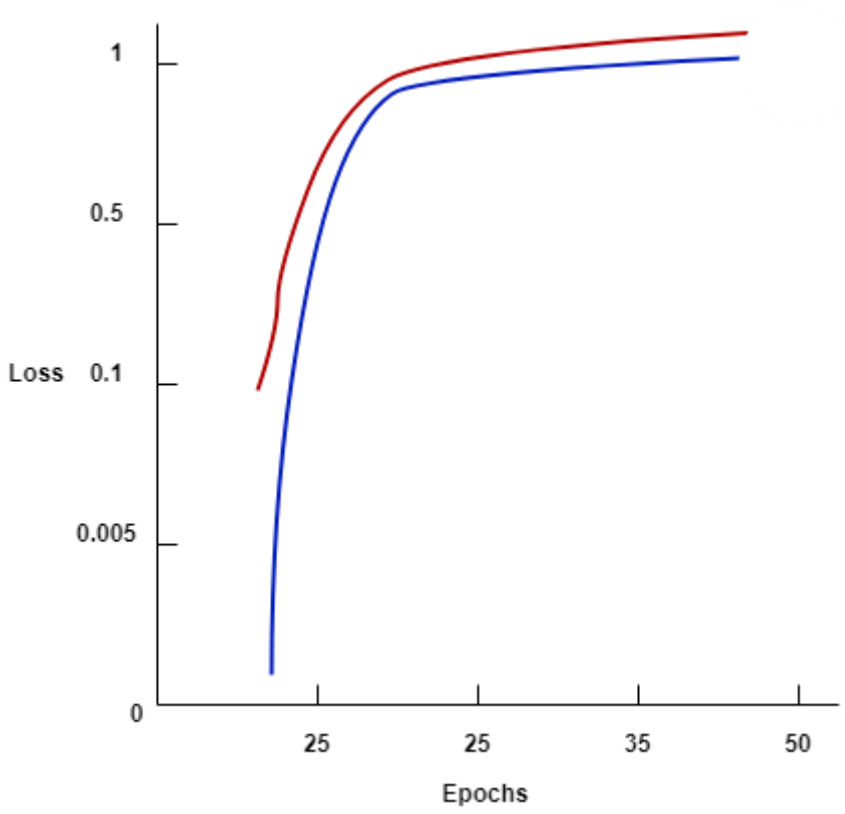

Figure 5.31: Loss vs Validation accuracy Network 1-50-100

5.9.1 50 epochs 200 layers MSE 2

Figure 5.32 presents the structure of Network 5 used in this case. This network is made of 274 neurons. The number of neurons in the LSTM has been improved to 200. It dramatically improved the model. Since the 20 first epochs were able to bring a validation loss of less than 0.1.

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_4 (LSTM)               (None, 200)               161600

 dense_6 (Dense)             (None, 1)                 201

 repeat_vector_2 (RepeatVec  (None, 20, 1)             0
 tor)

 lstm_5 (LSTM)               (None, 20, 200)           161600

 time_distributed_2 (TimeDi  (None, 20, 1)             201
 stributed)

 dense_8 (Dense)             (None, 20, 10)            20

=================================================================
Total params: 323622 (1.23 MB)
Trainable params: 323622 (1.23 MB)
Non-trainable params: 0 (0.00 Byte)
```

Figure 5.32: Sequential Model Network 1-50-200

The structure was basically the same with a Relu and sigmoid function at the output. A total number of 323622 parameters were trained in this scenario in order to get the best validation results. The Network 5 performed well since the accuracy and validation loss was very efficient as the validation accuracy increased over time as presented in Figure 5.33.

```
Epoch 1/50
1/1 - 3s - loss: 0.1400 - val_loss: 8.8624e-06 - 3s/epoch - 3s/step
Epoch 2/50
1/1 - 0s - loss: 0.1393 - val_loss: 3.6542e-05 - 55ms/epoch - 55ms/step
Epoch 3/50
1/1 - 0s - loss: 0.1387 - val_loss: 9.2530e-05 - 54ms/epoch - 54ms/step
Epoch 4/50
1/1 - 0s - loss: 0.1380 - val_loss: 1.7103e-04 - 55ms/epoch - 55ms/step
Epoch 5/50
1/1 - 0s - loss: 0.1374 - val_loss: 2.6824e-04 - 56ms/epoch - 56ms/step
Epoch 6/50
1/1 - 0s - loss: 0.1368 - val_loss: 3.6870e-04 - 56ms/epoch - 56ms/step
Epoch 7/50
1/1 - 0s - loss: 0.1362 - val_loss: 4.4829e-04 - 60ms/epoch - 60ms/step
Epoch 8/50
1/1 - 0s - loss: 0.1356 - val_loss: 4.9345e-04 - 52ms/epoch - 52ms/step
Epoch 9/50
1/1 - 0s - loss: 0.1350 - val_loss: 5.0724e-04 - 55ms/epoch - 55ms/step
Epoch 10/50
1/1 - 0s - loss: 0.1344 - val_loss: 4.9988e-04 - 58ms/epoch - 58ms/step
Epoch 11/50
1/1 - 0s - loss: 0.1338 - val_loss: 4.8335e-04 - 58ms/epoch - 58ms/step
Epoch 12/50
1/1 - 0s - loss: 0.1332 - val_loss: 4.6747e-04 - 53ms/epoch - 53ms/step
Epoch 13/50
1/1 - 0s - loss: 0.1326 - val_loss: 4.5926e-04 - 57ms/epoch - 57ms/step
Epoch 14/50
1/1 - 0s - loss: 0.1320 - val_loss: 4.6245e-04 - 59ms/epoch - 59ms/step
Epoch 15/50
1/1 - 0s - loss: 0.1314 - val_loss: 4.7941e-04 - 55ms/epoch - 55ms/step
Epoch 16/50
1/1 - 0s - loss: 0.1308 - val_loss: 5.0989e-04 - 56ms/epoch - 56ms/step
Epoch 17/50
1/1 - 0s - loss: 0.1302 - val_loss: 5.5417e-04 - 60ms/epoch - 60ms/step
Epoch 18/50
1/1 - 0s - loss: 0.1296 - val_loss: 6.1196e-04 - 71ms/epoch - 71ms/step
Epoch 19/50
1/1 - 0s - loss: 0.1290 - val_loss: 6.8282e-04 - 53ms/epoch - 53ms/step
Epoch 20/50
1/1 - 0s - loss: 0.1285 - val_loss: 7.6589e-04 - 55ms/epoch - 55ms/step
Epoch 21/50
1/1 - 0s - loss: 0.1279 - val_loss: 8.5781e-04 - 56ms/epoch - 56ms/step
Epoch 22/50
1/1 - 0s - loss: 0.1273 - val_loss: 9.5353e-04 - 59ms/epoch - 59ms/step
```

Figure 5.33: Computation Network 1-50-200

The best performance analysis obtained from Figure 5.33, carried out on the network, was the ROC. To evaluate the NN classifier's quality. Threshold values are applied to the outputs for each class over the range [0, 1]. Figure 5.34 depicts the ROC Network 1, where the curve tends toward the top and left margins of the plot but not towards the edge. It is above 50% which is satisfactory. This suggests that we haven't yet attained the highest precision.

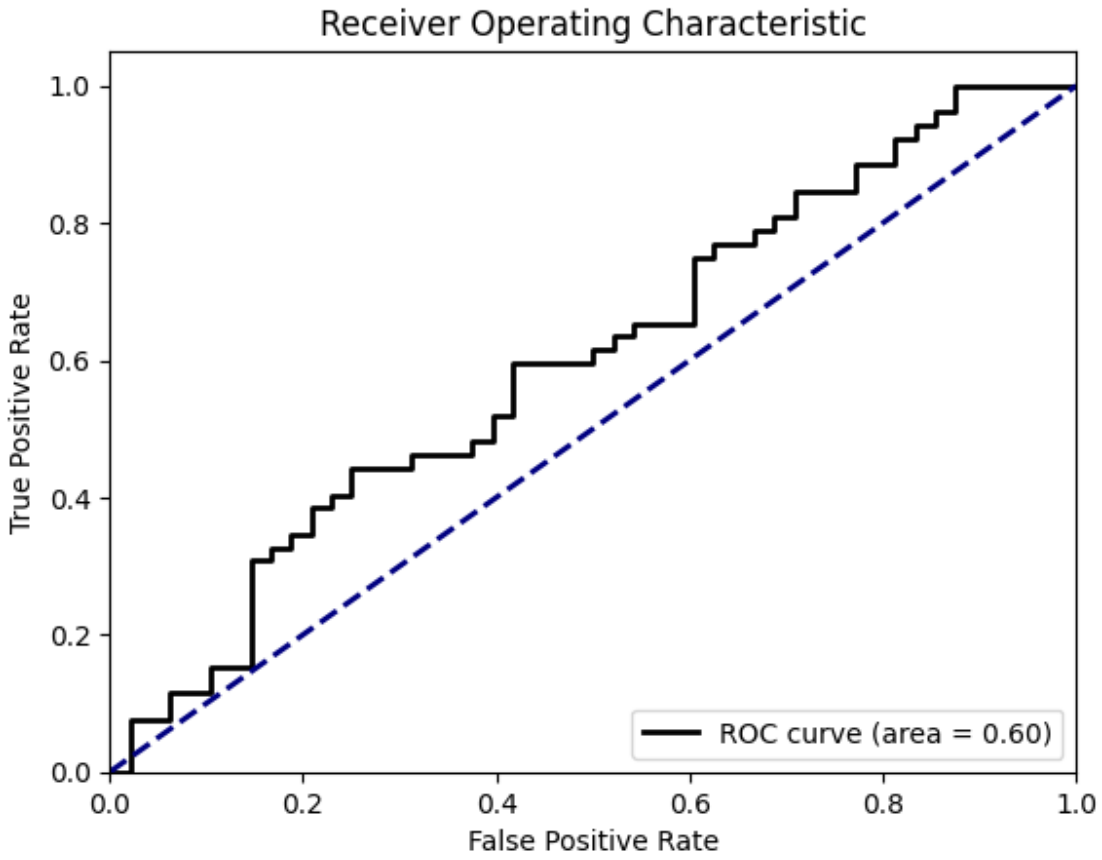

Figure 5.34: ROC Network 1.

5.9.2 Epochs number 100 with LSTM 100 layers

In this case, the number of epochs and the LSTM are equal to 100. From the ten first computations, the model obtained a satisfactory error less than 0 as predicted. It can be seen that after the 13th epoch, there is a decrease in performance, which is why the computation was stopped. The Network was able to provide an accuracy loss less than 0.0001 from the 1st to 12th epochs, then increased until it reached 0.1. Figure 5.35 presents the computation 100-100.

```
Epoch 1/100
1/1 - 4s - loss: 0.1400 - val_loss: 2.2483e-06 - 4s/epoch - 4s/step
Epoch 2/100
1/1 - 0s - loss: 0.1393 - val_loss: 1.2570e-05 - 63ms/epoch - 63ms/step
Epoch 3/100
1/1 - 0s - loss: 0.1386 - val_loss: 3.1428e-05 - 63ms/epoch - 63ms/step
Epoch 4/100
1/1 - 0s - loss: 0.1379 - val_loss: 6.1048e-05 - 63ms/epoch - 63ms/step
Epoch 5/100
1/1 - 0s - loss: 0.1372 - val_loss: 1.0313e-04 - 66ms/epoch - 66ms/step
Epoch 6/100
1/1 - 0s - loss: 0.1365 - val_loss: 1.5964e-04 - 63ms/epoch - 63ms/step
Epoch 7/100
1/1 - 0s - loss: 0.1357 - val_loss: 2.3295e-04 - 77ms/epoch - 77ms/step
Epoch 8/100
1/1 - 0s - loss: 0.1350 - val_loss: 3.2595e-04 - 61ms/epoch - 61ms/step
Epoch 9/100
1/1 - 0s - loss: 0.1343 - val_loss: 4.4230e-04 - 59ms/epoch - 59ms/step
Epoch 10/100
1/1 - 0s - loss: 0.1335 - val_loss: 5.8626e-04 - 58ms/epoch - 58ms/step
Epoch 11/100
1/1 - 0s - loss: 0.1328 - val_loss: 7.6289e-04 - 61ms/epoch - 61ms/step
Epoch 12/100
1/1 - 0s - loss: 0.1321 - val_loss: 9.7825e-04 - 73ms/epoch - 73ms/step
Epoch 13/100
1/1 - 0s - loss: 0.1314 - val_loss: 0.0012 - 68ms/epoch - 68ms/step
Epoch 14/100
1/1 - 0s - loss: 0.1307 - val_loss: 0.0016 - 66ms/epoch - 66ms/step
Epoch 15/100
1/1 - 0s - loss: 0.1300 - val_loss: 0.0019 - 75ms/epoch - 75ms/step
Epoch 16/100
1/1 - 0s - loss: 0.1293 - val_loss: 0.0023 - 73ms/epoch - 73ms/step
Epoch 17/100
1/1 - 0s - loss: 0.1287 - val_loss: 0.0028 - 87ms/epoch - 87ms/step
Epoch 18/100
1/1 - 0s - loss: 0.1281 - val_loss: 0.0032 - 73ms/epoch - 73ms/step
Epoch 19/100
1/1 - 0s - loss: 0.1275 - val_loss: 0.0035 - 84ms/epoch - 84ms/step
Epoch 20/100
1/1 - 0s - loss: 0.1269 - val_loss: 0.0036 - 75ms/epoch - 75ms/step
```

Figure 5.35: Computation Network 1-100-100

5.9.3 Epochs number 100 with LSTM 200 layers

In this scenario the number of LSTM layers has been increased to 200. Contrarily to the previous scenario, the results were less satisfactory. From the 7th epoch, the model started to increase and give accurate results, but the target was not achieved. Figure 5.36 depicts the Computation Network 1-100-200.

```
Epoch 1/100
1/1 - 4s - loss: 0.1400 - val_loss: 4.7473e-06 - 4s/epoch - 4s/step
Epoch 2/100
1/1 - 0s - loss: 0.1392 - val_loss: 3.0435e-05 - 54ms/epoch - 54ms/step
Epoch 3/100
1/1 - 0s - loss: 0.1382 - val_loss: 8.4438e-05 - 55ms/epoch - 55ms/step
Epoch 4/100
1/1 - 0s - loss: 0.1372 - val_loss: 1.7827e-04 - 56ms/epoch - 56ms/step
Epoch 5/100
1/1 - 0s - loss: 0.1361 - val_loss: 3.2713e-04 - 96ms/epoch - 96ms/step
Epoch 6/100
1/1 - 0s - loss: 0.1350 - val_loss: 5.5389e-04 - 80ms/epoch - 80ms/step
Epoch 7/100
1/1 - 0s - loss: 0.1339 - val_loss: 8.9259e-04 - 90ms/epoch - 90ms/step
Epoch 8/100
1/1 - 0s - loss: 0.1328 - val_loss: 0.0014 - 108ms/epoch - 108ms/step
Epoch 9/100
1/1 - 0s - loss: 0.1316 - val_loss: 0.0021 - 103ms/epoch - 103ms/step
Epoch 10/100
1/1 - 0s - loss: 0.1305 - val_loss: 0.0033 - 89ms/epoch - 89ms/step
Epoch 11/100
1/1 - 0s - loss: 0.1295 - val_loss: 0.0049 - 90ms/epoch - 90ms/step
Epoch 12/100
1/1 - 0s - loss: 0.1287 - val_loss: 0.0069 - 89ms/epoch - 89ms/step
Epoch 13/100
1/1 - 0s - loss: 0.1282 - val_loss: 0.0083 - 91ms/epoch - 91ms/step
Epoch 14/100
1/1 - 0s - loss: 0.1278 - val_loss: 0.0086 - 88ms/epoch - 88ms/step
Epoch 15/100
1/1 - 0s - loss: 0.1272 - val_loss: 0.0081 - 92ms/epoch - 92ms/step
Epoch 16/100
1/1 - 0s - loss: 0.1264 - val_loss: 0.0073 - 106ms/epoch - 106ms/step
Epoch 17/100
1/1 - 0s - loss: 0.1255 - val_loss: 0.0065 - 91ms/epoch - 91ms/step
Epoch 18/100
1/1 - 0s - loss: 0.1247 - val_loss: 0.0058 - 97ms/epoch - 97ms/step
Epoch 19/100
1/1 - 0s - loss: 0.1240 - val_loss: 0.0052 - 100ms/epoch - 100ms/step
Epoch 20/100
1/1 - 0s - loss: 0.1234 - val_loss: 0.0049 - 121ms/epoch - 121ms/step
```

Figure 5.36: Computation Network 1-100-200

5.10 Two-phase faults

Figure 5.37 shows the Sequential Model Network 2 used for this section, called Network 6. A total of 81802 parameters were trained in Python. With a total number of neurons of. The choice of the structure was kept since it has provided acceptable results.

```
Model: "sequential_8"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_12 (LSTM)              (None, 100)               40800

 dense_7 (Dense)             (None, 1)                 101

 repeat_vector_6 (RepeatVec  (None, 20, 1)             0
 tor)

 lstm_13 (LSTM)              (None, 20, 100)           40800

 time_distributed_5 (TimeDi  (None, 20, 1)             101
 stributed)

=================================================================
Total params: 81802 (319.54 KB)
Trainable params: 81802 (319.54 KB)
Non-trainable params: 0 (0.00 Byte)
```

Figure 5.37: Sequential Model Network 2

5.10.1 Epochs number 50 with 100 LSTM

As opposed to the previous scenarios, the computation was worth going to the end. Since the performance achieved was great. And the validation error accuracy kept on decreasing overtime which shows the efficiency of the model to predict the faults when much more iterations are computed with an MSE of less 0. Figures 5.38 and 5.39 present, respectively, the results of the Computation 50-100 and the Loss vs Validation Accuracy Network 2-50-100.

.

```
_________________________________________________________________
None
Epoch 1/50
1/1 - 4s - loss: 0.1989 - val_loss: 0.0070 - 4s/epoch - 4s/step
Epoch 2/50
1/1 - 0s - loss: 0.1017 - val_loss: 0.0579 - 51ms/epoch - 51ms/step
Epoch 3/50
1/1 - 0s - loss: 0.0606 - val_loss: 0.1408 - 52ms/epoch - 52ms/step
Epoch 4/50
1/1 - 0s - loss: 0.0649 - val_loss: 0.2151 - 51ms/epoch - 51ms/step
Epoch 5/50
1/1 - 0s - loss: 0.0875 - val_loss: 0.2481 - 46ms/epoch - 46ms/step
Epoch 6/50
1/1 - 0s - loss: 0.1003 - val_loss: 0.2394 - 56ms/epoch - 56ms/step
Epoch 7/50
1/1 - 0s - loss: 0.0965 - val_loss: 0.2045 - 52ms/epoch - 52ms/step
Epoch 8/50
1/1 - 0s - loss: 0.0829 - val_loss: 0.1591 - 51ms/epoch - 51ms/step
Epoch 9/50
1/1 - 0s - loss: 0.0684 - val_loss: 0.1148 - 49ms/epoch - 49ms/step
Epoch 10/50
1/1 - 0s - loss: 0.0589 - val_loss: 0.0785 - 63ms/epoch - 63ms/step
Epoch 11/50
1/1 - 0s - loss: 0.0566 - val_loss: 0.0528 - 72ms/epoch - 72ms/step
Epoch 12/50
1/1 - 0s - loss: 0.0601 - val_loss: 0.0372 - 50ms/epoch - 50ms/step
Epoch 13/50
1/1 - 0s - loss: 0.0659 - val_loss: 0.0297 - 52ms/epoch - 52ms/step
Epoch 14/50
1/1 - 0s - loss: 0.0701 - val_loss: 0.0284 - 69ms/epoch - 69ms/step
Epoch 15/50
1/1 - 0s - loss: 0.0709 - val_loss: 0.0322 - 55ms/epoch - 55ms/step
Epoch 16/50
1/1 - 0s - loss: 0.0681 - val_loss: 0.0408 - 54ms/epoch - 54ms/step
Epoch 17/50
1/1 - 0s - loss: 0.0633 - val_loss: 0.0540 - 58ms/epoch - 58ms/step
Epoch 18/50
1/1 - 0s - loss: 0.0586 - val_loss: 0.0710 - 71ms/epoch - 71ms/step
Epoch 19/50
1/1 - 0s - loss: 0.0555 - val_loss: 0.0901 - 55ms/epoch - 55ms/step
Epoch 20/50
```

```
Epoch 21/50
1/1 - 0s - loss: 0.0559 - val_loss: 0.1240 - 66ms/epoch - 66ms/step
Epoch 22/50
1/1 - 0s - loss: 0.0580 - val_loss: 0.1334 - 52ms/epoch - 52ms/step
Epoch 23/50
1/1 - 0s - loss: 0.0595 - val_loss: 0.1357 - 54ms/epoch - 54ms/step
Epoch 24/50
1/1 - 0s - loss: 0.0599 - val_loss: 0.1312 - 49ms/epoch - 49ms/step
Epoch 25/50
1/1 - 0s - loss: 0.0588 - val_loss: 0.1214 - 49ms/epoch - 49ms/step
Epoch 26/50
1/1 - 0s - loss: 0.0568 - val_loss: 0.1084 - 53ms/epoch - 53ms/step
Epoch 27/50
1/1 - 0s - loss: 0.0548 - val_loss: 0.0945 - 59ms/epoch - 59ms/step
Epoch 28/50
1/1 - 0s - loss: 0.0535 - val_loss: 0.0816 - 64ms/epoch - 64ms/step
Epoch 29/50
1/1 - 0s - loss: 0.0531 - val_loss: 0.0711 - 77ms/epoch - 77ms/step
Epoch 30/50
1/1 - 0s - loss: 0.0535 - val_loss: 0.0637 - 78ms/epoch - 78ms/step
Epoch 31/50
1/1 - 0s - loss: 0.0542 - val_loss: 0.0597 - 80ms/epoch - 80ms/step
Epoch 32/50
1/1 - 0s - loss: 0.0548 - val_loss: 0.0589 - 87ms/epoch - 87ms/step
Epoch 33/50
1/1 - 0s - loss: 0.0548 - val_loss: 0.0611 - 83ms/epoch - 83ms/step
Epoch 34/50
1/1 - 0s - loss: 0.0542 - val_loss: 0.0658 - 91ms/epoch - 91ms/step
Epoch 35/50
1/1 - 0s - loss: 0.0533 - val_loss: 0.0727 - 96ms/epoch - 96ms/step
Epoch 36/50
1/1 - 0s - loss: 0.0524 - val_loss: 0.0807 - 97ms/epoch - 97ms/step
Epoch 37/50
1/1 - 0s - loss: 0.0518 - val_loss: 0.0888 - 91ms/epoch - 91ms/step
Epoch 38/50
1/1 - 0s - loss: 0.0517 - val_loss: 0.0960 - 87ms/epoch - 87ms/step
Epoch 39/50
1/1 - 0s - loss: 0.0518 - val_loss: 0.1012 - 71ms/epoch - 71ms/step
Epoch 40/50
1/1 - 0s - loss: 0.0521 - val_loss: 0.1036 - 77ms/epoch - 77ms/step
Epoch 41/50
```

```
Epoch 41/50
1/1 - 0s - loss: 0.0521 - val_loss: 0.1032 - 95ms/epoch - 95ms/step
Epoch 42/50
1/1 - 0s - loss: 0.0519 - val_loss: 0.1001 - 93ms/epoch - 93ms/step
Epoch 43/50
1/1 - 0s - loss: 0.0515 - val_loss: 0.0952 - 81ms/epoch - 81ms/step
Epoch 44/50
1/1 - 0s - loss: 0.0510 - val_loss: 0.0893 - 80ms/epoch - 80ms/step
Epoch 45/50
1/1 - 0s - loss: 0.0506 - val_loss: 0.0834 - 90ms/epoch - 90ms/step
Epoch 46/50
1/1 - 0s - loss: 0.0504 - val_loss: 0.0783 - 88ms/epoch - 88ms/step
Epoch 47/50
1/1 - 0s - loss: 0.0503 - val_loss: 0.0746 - 94ms/epoch - 94ms/step
Epoch 48/50
1/1 - 0s - loss: 0.0503 - val_loss: 0.0725 - 87ms/epoch - 87ms/step
Epoch 49/50
1/1 - 0s - loss: 0.0503 - val_loss: 0.0722 - 85ms/epoch - 85ms/step
Epoch 50/50
1/1 - 0s - loss: 0.0502 - val_loss: 0.0735 - 86ms/epoch - 86ms/step
<keras.src.callbacks.History at 0x784a6e965c60>
```

Figure 5.38: Computation Network 2-50-100

The loss and training curve accuracy are increasing overtime until it reaches one meaning 1001% accuracy. The network were able to learn without any issues and the generalization has been properly done as well. The figure 5.39 presents the Loss and validation accuracy network 2-50-100.

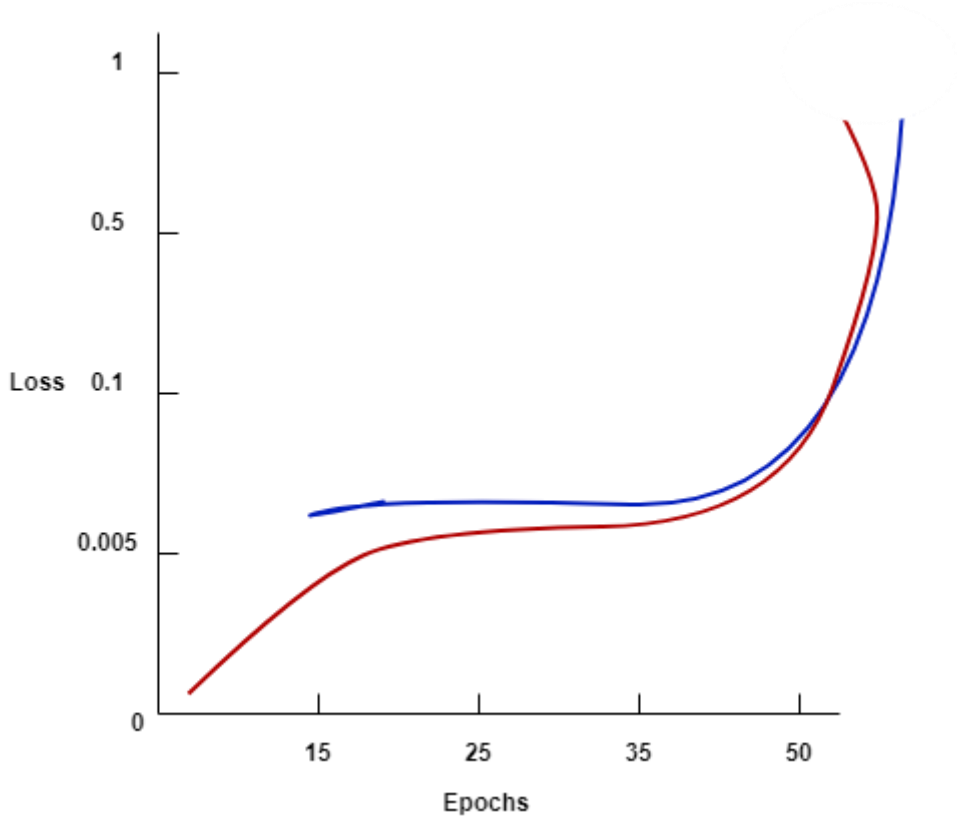

Figure 5.39: Loss vs Validation accuracy Network 2-50-100.

5.10.2 Epochs number 100 with LSTM 100 layers

In this scenario, the computation 100-100 reacted as the previous one, as the model ran, the performance was satisfying, and the target was achieved as well. Just that with an equal number of epochs and LSTM layers, it presented much better results in Figures 5.40 and 5.41, respectively. Computation Network 2-50-100 plotted the results from the computation 100-100 and Loss vs Validation Accuracy Network 2-50-100.

```
Epoch 1/100
1/1 - 4s - loss: 0.1727 - val_loss: 0.0365 - 4s/epoch - 4s/step
Epoch 2/100
1/1 - 0s - loss: 0.1505 - val_loss: 0.0165 - 43ms/epoch - 43ms/step
Epoch 3/100
1/1 - 0s - loss: 0.1385 - val_loss: 0.0060 - 40ms/epoch - 40ms/step
Epoch 4/100
1/1 - 0s - loss: 0.1351 - val_loss: 0.0025 - 41ms/epoch - 41ms/step
Epoch 5/100
1/1 - 0s - loss: 0.1373 - val_loss: 0.0026 - 41ms/epoch - 41ms/step
Epoch 6/100
1/1 - 0s - loss: 0.1409 - val_loss: 0.0032 - 42ms/epoch - 42ms/step
Epoch 7/100
1/1 - 0s - loss: 0.1425 - val_loss: 0.0030 - 44ms/epoch - 44ms/step
Epoch 8/100
1/1 - 0s - loss: 0.1415 - val_loss: 0.0025 - 41ms/epoch - 41ms/step
Epoch 9/100
1/1 - 0s - loss: 0.1388 - val_loss: 0.0023 - 41ms/epoch - 41ms/step
Epoch 10/100
1/1 - 0s - loss: 0.1355 - val_loss: 0.0030 - 45ms/epoch - 45ms/step
Epoch 11/100
1/1 - 0s - loss: 0.1325 - val_loss: 0.0050 - 41ms/epoch - 41ms/step
Epoch 12/100
1/1 - 0s - loss: 0.1306 - val_loss: 0.0079 - 42ms/epoch - 42ms/step
Epoch 13/100
1/1 - 0s - loss: 0.1298 - val_loss: 0.0113 - 52ms/epoch - 52ms/step
Epoch 14/100
1/1 - 0s - loss: 0.1298 - val_loss: 0.0145 - 43ms/epoch - 43ms/step
Epoch 15/100
1/1 - 0s - loss: 0.1302 - val_loss: 0.0168 - 45ms/epoch - 45ms/step
Epoch 16/100
1/1 - 0s - loss: 0.1305 - val_loss: 0.0178 - 42ms/epoch - 42ms/step
Epoch 17/100
1/1 - 0s - loss: 0.1303 - val_loss: 0.0174 - 42ms/epoch - 42ms/step
Epoch 18/100
1/1 - 0s - loss: 0.1295 - val_loss: 0.0160 - 46ms/epoch - 46ms/step
Epoch 19/100
1/1 - 0s - loss: 0.1283 - val_loss: 0.0138 - 45ms/epoch - 45ms/step
Epoch 20/100
1/1 - 0s - loss: 0.1268 - val_loss: 0.0114 - 43ms/epoch - 43ms/step
Epoch 21/100
1/1 - 0s - loss: 0.1255 - val_loss: 0.0092 - 41ms/epoch - 41ms/step
Epoch 22/100
1/1 - 0s - loss: 0.1245 - val_loss: 0.0072 - 42ms/epoch - 42ms/step
```

```
Epoch 80/100
1/1 - 0s - loss: 0.0952 - val_loss: 0.0156 - 47ms/epoch - 47ms/step
Epoch 81/100
1/1 - 0s - loss: 0.0948 - val_loss: 0.0159 - 42ms/epoch - 42ms/step
Epoch 82/100
1/1 - 0s - loss: 0.0944 - val_loss: 0.0162 - 44ms/epoch - 44ms/step
Epoch 83/100
1/1 - 0s - loss: 0.0939 - val_loss: 0.0166 - 45ms/epoch - 45ms/step
Epoch 84/100
1/1 - 0s - loss: 0.0935 - val_loss: 0.0169 - 43ms/epoch - 43ms/step
Epoch 85/100
1/1 - 0s - loss: 0.0931 - val_loss: 0.0171 - 46ms/epoch - 46ms/step
Epoch 86/100
1/1 - 0s - loss: 0.0927 - val_loss: 0.0173 - 45ms/epoch - 45ms/step
Epoch 87/100
1/1 - 0s - loss: 0.0923 - val_loss: 0.0174 - 43ms/epoch - 43ms/step
Epoch 88/100
1/1 - 0s - loss: 0.0918 - val_loss: 0.0175 - 43ms/epoch - 43ms/step
Epoch 89/100
1/1 - 0s - loss: 0.0914 - val_loss: 0.0175 - 43ms/epoch - 43ms/step
Epoch 90/100
1/1 - 0s - loss: 0.0910 - val_loss: 0.0176 - 43ms/epoch - 43ms/step
Epoch 91/100
1/1 - 0s - loss: 0.0906 - val_loss: 0.0176 - 42ms/epoch - 42ms/step
Epoch 92/100
1/1 - 0s - loss: 0.0902 - val_loss: 0.0176 - 41ms/epoch - 41ms/step
Epoch 93/100
1/1 - 0s - loss: 0.0898 - val_loss: 0.0177 - 42ms/epoch - 42ms/step
Epoch 94/100
1/1 - 0s - loss: 0.0894 - val_loss: 0.0179 - 50ms/epoch - 50ms/step
Epoch 95/100
1/1 - 0s - loss: 0.0890 - val_loss: 0.0181 - 42ms/epoch - 42ms/step
Epoch 96/100
1/1 - 0s - loss: 0.0886 - val_loss: 0.0183 - 42ms/epoch - 42ms/step
Epoch 97/100
1/1 - 0s - loss: 0.0881 - val_loss: 0.0185 - 44ms/epoch - 44ms/step
Epoch 98/100
1/1 - 0s - loss: 0.0877 - val_loss: 0.0187 - 55ms/epoch - 55ms/step
Epoch 99/100
1/1 - 0s - loss: 0.0873 - val_loss: 0.0188 - 42ms/epoch - 42ms/step
Epoch 100/100
1/1 - 0s - loss: 0.0869 - val_loss: 0.0190 - 42ms/epoch - 42ms/step
<keras.src.callbacks.History at 0x79a8ff2159c0>
```

Figure 5.40: Computation Network 2-50-100

The loss and validation accuracy moves simultaneously and do not oscillate so the signal is pretty stable. The generalization beyond training set is acceptable. The Figure 5.41 depicts the Loss and Validation accuracy network 2-50-100.

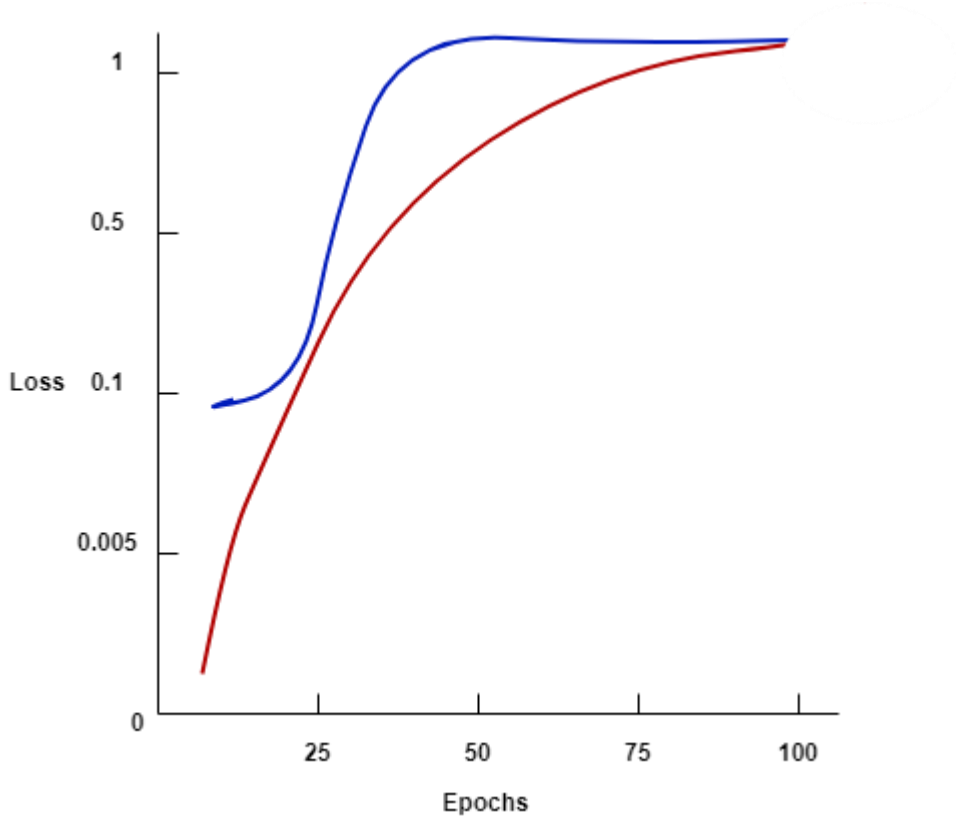

Figure 5.41: Loss vs Validation Accuracy Network 2-50-100.

5.10.3 Epochs number 100 with LSTM 200 layers

In this section, the number of LSTM layers was increased to 200. In order to see how the system will act with larger number of layers. It reveals that the model achieved the desire performance goal under 0 error, as the iterations were processed, the model even attained less than 0,001 before the 20th epochs. Which means that the number of layers slightly improved the model as depicted in Figure 5.42, Computation Network 2-100-100.

```
Epoch 1/100
1/1 - 4s - loss: 0.1556 - val_loss: 0.0056 - 4s/epoch - 4s/step
Epoch 2/100
1/1 - 0s - loss: 0.1463 - val_loss: 0.0105 - 53ms/epoch - 53ms/step
Epoch 3/100
1/1 - 0s - loss: 0.1513 - val_loss: 0.0024 - 52ms/epoch - 52ms/step
Epoch 4/100
1/1 - 0s - loss: 0.1414 - val_loss: 0.0018 - 55ms/epoch - 55ms/step
Epoch 5/100
1/1 - 0s - loss: 0.1387 - val_loss: 0.0075 - 54ms/epoch - 54ms/step
Epoch 6/100
1/1 - 0s - loss: 0.1428 - val_loss: 0.0082 - 52ms/epoch - 52ms/step
Epoch 7/100
1/1 - 0s - loss: 0.1428 - val_loss: 0.0039 - 52ms/epoch - 52ms/step
Epoch 8/100
1/1 - 0s - loss: 0.1386 - val_loss: 7.4113e-04 - 56ms/epoch - 56ms/step
Epoch 9/100
1/1 - 0s - loss: 0.1357 - val_loss: 0.0012 - 52ms/epoch - 52ms/step
Epoch 10/100
1/1 - 0s - loss: 0.1365 - val_loss: 0.0027 - 54ms/epoch - 54ms/step
Epoch 11/100
1/1 - 0s - loss: 0.1379 - val_loss: 0.0024 - 52ms/epoch - 52ms/step
Epoch 12/100
1/1 - 0s - loss: 0.1369 - val_loss: 9.7270e-04 - 73ms/epoch - 73ms/step
Epoch 13/100
1/1 - 0s - loss: 0.1344 - val_loss: 6.1724e-04 - 51ms/epoch - 51ms/step
Epoch 14/100
1/1 - 0s - loss: 0.1328 - val_loss: 0.0020 - 51ms/epoch - 51ms/step
Epoch 15/100
1/1 - 0s - loss: 0.1329 - val_loss: 0.0034 - 51ms/epoch - 51ms/step
Epoch 16/100
1/1 - 0s - loss: 0.1334 - val_loss: 0.0034 - 56ms/epoch - 56ms/step
Epoch 17/100
1/1 - 0s - loss: 0.1328 - val_loss: 0.0021 - 51ms/epoch - 51ms/step
Epoch 18/100
1/1 - 0s - loss: 0.1313 - val_loss: 8.8322e-04 - 51ms/epoch - 51ms/step
Epoch 19/100
1/1 - 0s - loss: 0.1299 - val_loss: 6.0409e-04 - 68ms/epoch - 68ms/step
Epoch 20/100
1/1 - 0s - loss: 0.1296 - val_loss: 9.4488e-04 - 51ms/epoch - 51ms/step
Epoch 21/100
1/1 - 0s - loss: 0.1297 - val_loss: 0.0011 - 56ms/epoch - 56ms/step
```

```
Epoch 80/100
1/1 - 0s - loss: 0.1005 - val_loss: 0.0062 - 52ms/epoch - 52ms/step
Epoch 81/100
1/1 - 0s - loss: 0.1001 - val_loss: 0.0064 - 54ms/epoch - 54ms/step
Epoch 82/100
1/1 - 0s - loss: 0.0997 - val_loss: 0.0065 - 55ms/epoch - 55ms/step
Epoch 83/100
1/1 - 0s - loss: 0.0993 - val_loss: 0.0066 - 53ms/epoch - 53ms/step
Epoch 84/100
1/1 - 0s - loss: 0.0989 - val_loss: 0.0067 - 50ms/epoch - 50ms/step
Epoch 85/100
1/1 - 0s - loss: 0.0985 - val_loss: 0.0068 - 70ms/epoch - 70ms/step
Epoch 86/100
1/1 - 0s - loss: 0.0981 - val_loss: 0.0070 - 51ms/epoch - 51ms/step
Epoch 87/100
1/1 - 0s - loss: 0.0977 - val_loss: 0.0072 - 60ms/epoch - 60ms/step
Epoch 88/100
1/1 - 0s - loss: 0.0973 - val_loss: 0.0073 - 52ms/epoch - 52ms/step
Epoch 89/100
1/1 - 0s - loss: 0.0970 - val_loss: 0.0075 - 52ms/epoch - 52ms/step
Epoch 90/100
1/1 - 0s - loss: 0.0966 - val_loss: 0.0076 - 54ms/epoch - 54ms/step
Epoch 91/100
1/1 - 0s - loss: 0.0962 - val_loss: 0.0078 - 55ms/epoch - 55ms/step
Epoch 92/100
1/1 - 0s - loss: 0.0958 - val_loss: 0.0079 - 55ms/epoch - 55ms/step
Epoch 93/100
1/1 - 0s - loss: 0.0954 - val_loss: 0.0080 - 53ms/epoch - 53ms/step
Epoch 94/100
1/1 - 0s - loss: 0.0950 - val_loss: 0.0082 - 55ms/epoch - 55ms/step
Epoch 95/100
1/1 - 0s - loss: 0.0946 - val_loss: 0.0083 - 67ms/epoch - 67ms/step
Epoch 96/100
1/1 - 0s - loss: 0.0943 - val_loss: 0.0085 - 52ms/epoch - 52ms/step
Epoch 97/100
1/1 - 0s - loss: 0.0939 - val_loss: 0.0086 - 57ms/epoch - 57ms/step
Epoch 98/100
1/1 - 0s - loss: 0.0935 - val_loss: 0.0088 - 54ms/epoch - 54ms/step
Epoch 99/100
1/1 - 0s - loss: 0.0931 - val_loss: 0.0089 - 54ms/epoch - 54ms/step
Epoch 100/100
1/1 - 0s - loss: 0.0928 - val_loss: 0.0090 - 53ms/epoch - 53ms/step
<keras.src.callbacks.History at 0x79a8fefe6620>
```

Figure 5.42: Computation Network 2-100-100

5.10.4 Layers 200 with 50 epochs

The ROC was used to assess the performance of the NN classifier obtained from the best computation Network 2-100-100 provided in Figure 5.43. The outputs for every class are subjected to threshold values within the interval [0, 1].

```
Epoch 1/50
1/1 - 4s - loss: 0.1433 - val_loss: 0.0222 - 4s/epoch - 4s/step
Epoch 2/50
1/1 - 0s - loss: 0.1551 - val_loss: 0.0061 - 56ms/epoch - 56ms/step
Epoch 3/50
1/1 - 0s - loss: 0.1415 - val_loss: 3.0493e-04 - 56ms/epoch - 56ms/step
Epoch 4/50
1/1 - 0s - loss: 0.1394 - val_loss: 0.0040 - 54ms/epoch - 54ms/step
Epoch 5/50
1/1 - 0s - loss: 0.1447 - val_loss: 0.0024 - 49ms/epoch - 49ms/step
Epoch 6/50
1/1 - 0s - loss: 0.1419 - val_loss: 7.8688e-05 - 49ms/epoch - 49ms/step
Epoch 7/50
1/1 - 0s - loss: 0.1368 - val_loss: 0.0024 - 52ms/epoch - 52ms/step
Epoch 8/50
1/1 - 0s - loss: 0.1362 - val_loss: 0.0067 - 55ms/epoch - 55ms/step
Epoch 9/50
1/1 - 0s - loss: 0.1383 - val_loss: 0.0074 - 52ms/epoch - 52ms/step
Epoch 10/50
1/1 - 0s - loss: 0.1382 - val_loss: 0.0044 - 50ms/epoch - 50ms/step
Epoch 11/50
1/1 - 0s - loss: 0.1355 - val_loss: 0.0012 - 66ms/epoch - 66ms/step
Epoch 12/50
1/1 - 0s - loss: 0.1333 - val_loss: 1.5708e-04 - 51ms/epoch - 51ms/step
Epoch 13/50
1/1 - 0s - loss: 0.1333 - val_loss: 5.8093e-04 - 53ms/epoch - 53ms/step
Epoch 14/50
1/1 - 0s - loss: 0.1341 - val_loss: 6.6127e-04 - 58ms/epoch - 58ms/step
Epoch 15/50
1/1 - 0s - loss: 0.1337 - val_loss: 2.6966e-04 - 51ms/epoch - 51ms/step
Epoch 16/50
1/1 - 0s - loss: 0.1319 - val_loss: 5.7080e-04 - 51ms/epoch - 51ms/step
Epoch 17/50
1/1 - 0s - loss: 0.1304 - val_loss: 0.0021 - 64ms/epoch - 64ms/step
```

```
Epoch 33/50
1/1 - 0s - loss: 0.1216 - val_loss: 0.0013 - 50ms/epoch - 50ms/step
Epoch 34/50
1/1 - 0s - loss: 0.1213 - val_loss: 0.0013 - 55ms/epoch - 55ms/step
Epoch 35/50
1/1 - 0s - loss: 0.1208 - val_loss: 0.0014 - 64ms/epoch - 64ms/step
Epoch 36/50
1/1 - 0s - loss: 0.1202 - val_loss: 0.0019 - 52ms/epoch - 52ms/step
Epoch 37/50
1/1 - 0s - loss: 0.1195 - val_loss: 0.0025 - 52ms/epoch - 52ms/step
Epoch 38/50
1/1 - 0s - loss: 0.1190 - val_loss: 0.0031 - 53ms/epoch - 53ms/step
Epoch 39/50
1/1 - 0s - loss: 0.1186 - val_loss: 0.0034 - 53ms/epoch - 53ms/step
Epoch 40/50
1/1 - 0s - loss: 0.1181 - val_loss: 0.0033 - 52ms/epoch - 52ms/step
Epoch 41/50
1/1 - 0s - loss: 0.1175 - val_loss: 0.0029 - 53ms/epoch - 53ms/step
Epoch 42/50
1/1 - 0s - loss: 0.1169 - val_loss: 0.0025 - 53ms/epoch - 53ms/step
Epoch 43/50
1/1 - 0s - loss: 0.1164 - val_loss: 0.0023 - 53ms/epoch - 53ms/step
Epoch 44/50
1/1 - 0s - loss: 0.1160 - val_loss: 0.0024 - 52ms/epoch - 52ms/step
Epoch 45/50
1/1 - 0s - loss: 0.1155 - val_loss: 0.0026 - 52ms/epoch - 52ms/step
Epoch 46/50
1/1 - 0s - loss: 0.1150 - val_loss: 0.0031 - 53ms/epoch - 53ms/step
Epoch 47/50
1/1 - 0s - loss: 0.1144 - val_loss: 0.0036 - 51ms/epoch - 51ms/step
Epoch 48/50
1/1 - 0s - loss: 0.1139 - val_loss: 0.0040 - 77ms/epoch - 77ms/step
Epoch 49/50
1/1 - 0s - loss: 0.1135 - val_loss: 0.0040 - 77ms/epoch - 77ms/step
Epoch 50/50
1/1 - 0s - loss: 0.1130 - val_loss: 0.0039 - 81ms/epoch - 81ms/step
<keras.src.callbacks.History at 0x79a90c6db4c0>
```

Figure 5.43: Computation Network 2-50-200

The ROC Network 2 plot is shown in Figure 5.44, presenting a result of 70%, which means that for the entire computation, the true positive results that actually demonstrate that the model detects a fault without error is 70%.

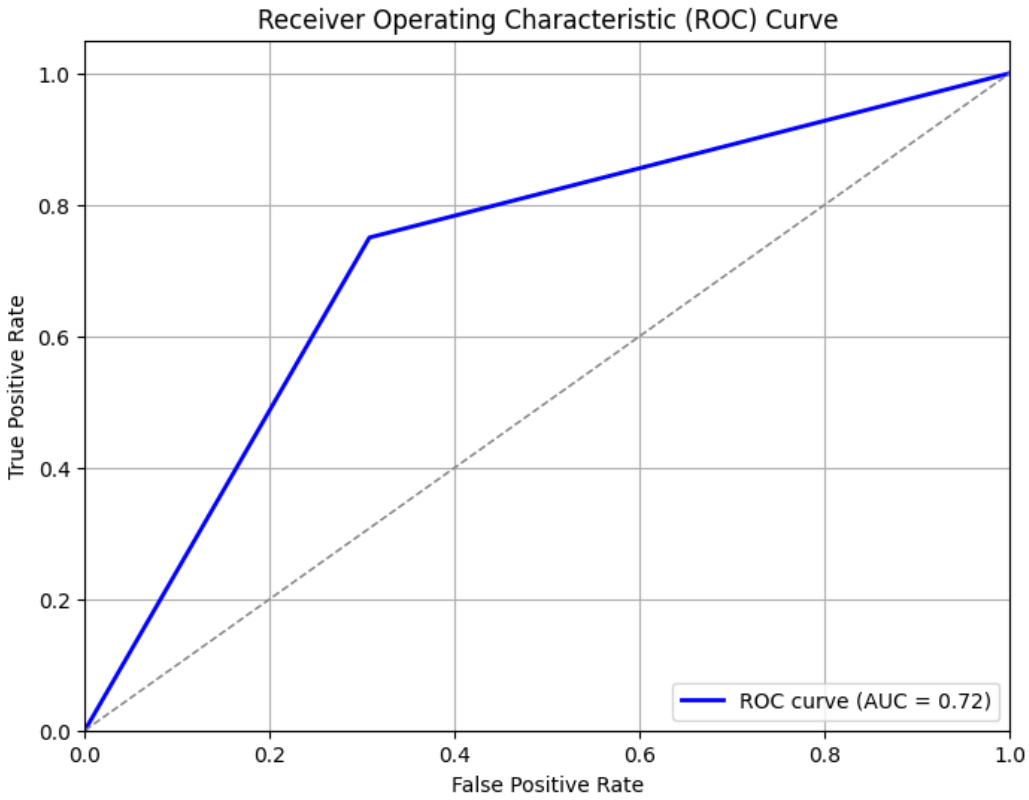

Figure 5.44: ROC Network 2.

5.11 Three-phase faults

This network has a total number of 274 neurons. The target Mean Square error was less than 0.1. The training algorithm was. The activation function used were the Rectifier Linear Unit (Relu) and Sigmoid. Sequential Model Network 3 with the computation 200-50 of the LSTM model was given by Figure 5.45.

```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_6 (LSTM)               (None, 100)               40800

 dense_9 (Dense)             (None, 1)                 101

 repeat_vector_3 (RepeatVec  (None, 20, 1)             0
 tor)

 lstm_7 (LSTM)               (None, 20, 100)           40800

 time_distributed_3 (TimeDi  (None, 20, 1)             101
 stributed)

 dense_11 (Dense)            (None, 20, 10)            20

=================================================================
Total params: 81822 (319.62 KB)
Trainable params: 81822 (319.62 KB)
Non-trainable params: 0 (0.00 Byte)
```

Figure 5.45: Sequential Model Network 3.

Figure 5.46 shows the computation 3-50-100 of the number of epochs. It can be seen that the convergence comes around the 17th epoch that we highlighted. The model was tested with different numbers of neurons and epochs to see where to get the best performance. In the figure, the different computations were displayed. For both cases, a total of 81822 and 323622 parameters have been trained. The computation was stopped at 100 and 200 epochs for some since the model presented a convergence earlier in the simulation, so they were no point in adding more epochs. The Network allowed to get an MSE of less than 0.001, then increased to 0.1 as presented in Figure 5.47, which depicts the loss vs the Validation Accuracy Network 3-50-100.

```
Epoch 1/50
1/1 - 4s - loss: 0.1400 - val_loss: 4.6526e-06 - 4s/epoch - 4s/step
Epoch 2/50
1/1 - 0s - loss: 0.1393 - val_loss: 2.1142e-05 - 66ms/epoch - 66ms/step
Epoch 3/50
1/1 - 0s - loss: 0.1386 - val_loss: 5.6025e-05 - 64ms/epoch - 64ms/step
Epoch 4/50
1/1 - 0s - loss: 0.1379 - val_loss: 1.1001e-04 - 75ms/epoch - 75ms/step
Epoch 5/50
1/1 - 0s - loss: 0.1372 - val_loss: 1.8726e-04 - 68ms/epoch - 68ms/step
Epoch 6/50
1/1 - 0s - loss: 0.1365 - val_loss: 2.9120e-04 - 62ms/epoch - 62ms/step
Epoch 7/50
1/1 - 0s - loss: 0.1358 - val_loss: 4.2693e-04 - 60ms/epoch - 60ms/step
Epoch 8/50
1/1 - 0s - loss: 0.1351 - val_loss: 6.0013e-04 - 64ms/epoch - 64ms/step
Epoch 9/50
1/1 - 0s - loss: 0.1344 - val_loss: 8.1643e-04 - 64ms/epoch - 64ms/step
Epoch 10/50
1/1 - 0s - loss: 0.1338 - val_loss: 0.0011 - 64ms/epoch - 64ms/step
Epoch 11/50
1/1 - 0s - loss: 0.1332 - val_loss: 0.0013 - 84ms/epoch - 84ms/step
Epoch 12/50
1/1 - 0s - loss: 0.1326 - val_loss: 0.0015 - 62ms/epoch - 62ms/step
Epoch 13/50
1/1 - 0s - loss: 0.1320 - val_loss: 0.0016 - 62ms/epoch - 62ms/step
Epoch 14/50
1/1 - 0s - loss: 0.1313 - val_loss: 0.0016 - 61ms/epoch - 61ms/step
Epoch 15/50
1/1 - 0s - loss: 0.1307 - val_loss: 0.0016 - 59ms/epoch - 59ms/step
Epoch 16/50
1/1 - 0s - loss: 0.1301 - val_loss: 0.0015 - 58ms/epoch - 58ms/step
Epoch 17/50
1/1 - 0s - loss: 0.1295 - val_loss: 0.0014 - 60ms/epoch - 60ms/step
Epoch 18/50
1/1 - 0s - loss: 0.1288 - val_loss: 0.0014 - 58ms/epoch - 58ms/step
Epoch 19/50
1/1 - 0s - loss: 0.1282 - val_loss: 0.0013 - 59ms/epoch - 59ms/step
Epoch 20/50
1/1 - 0s - loss: 0.1276 - val_loss: 0.0014 - 59ms/epoch - 59ms/step
Epoch 21/50
1/1 - 0s - loss: 0.1270 - val_loss: 0.0014 - 59ms/epoch - 59ms/step
```

Figure 5.46: Computation Network 3-50-100

From 0 to 50 epochs, both curves keeps on increasing until reaching 0.001.Which means that if training were consider on a longer timeframe like a thousands epoch it would have reached 1 or 100% accuracy. Figure 5.47 provides the Loss and Validation accuracy network 3-50-100.

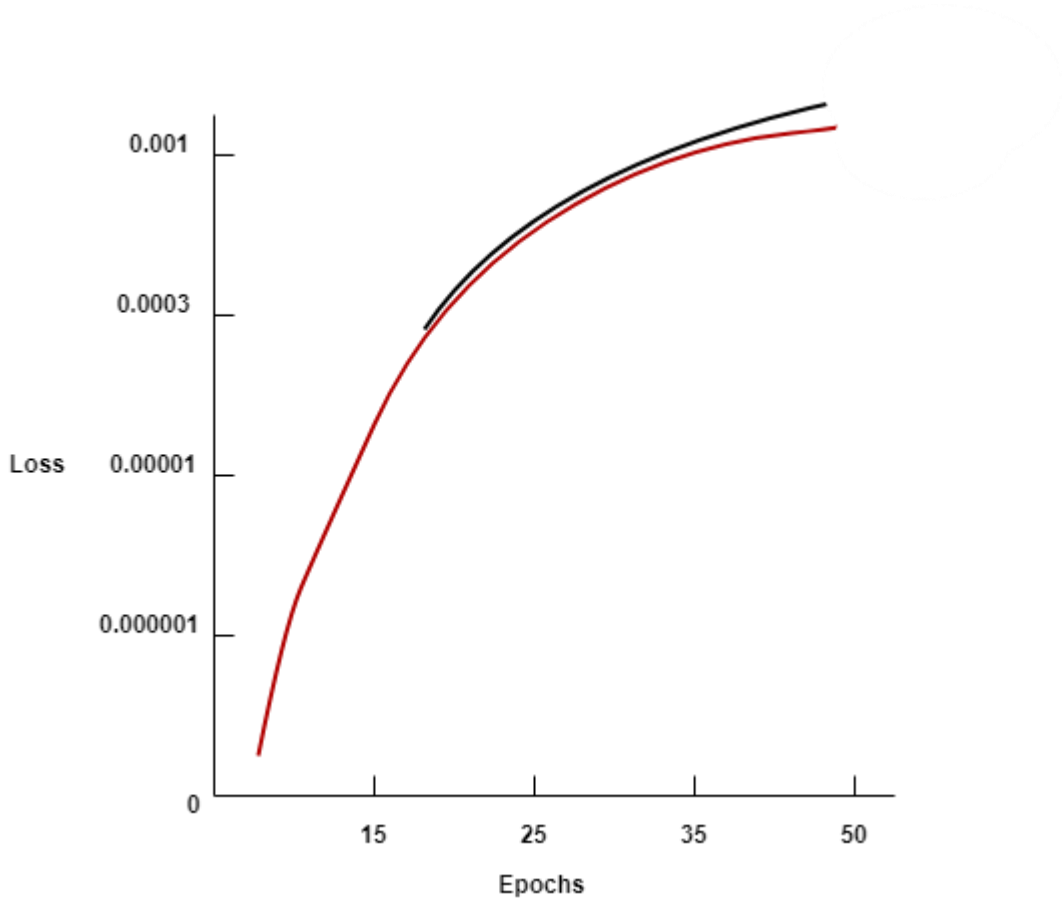

Figure 5.47: Loss vs Validation Accuracy Network 3-50-100

5.11.1 100 Epochs 100 LSTM MSE4

This configuration presents an equal number of epochs and LSTMS layers. The performance goal of less than 0 as error was successfully achieved. Due to the oversampling of the data, the error increased slightly over the iterations without affecting the results much. But the MSE remained less than 0 with an increasing validation loss over the computation 3-100-100 as depicted in Figure 5.48. The Validation vs Loss accuracy Network is plotted in Figure 5.49.

```
Epoch 1/100
1/1 - 5s - loss: 0.1400 - val_loss: 3.4012e-06 - 5s/epoch - 5s/step
Epoch 2/100
1/1 - 0s - loss: 0.1394 - val_loss: 1.8789e-05 - 47ms/epoch - 47ms/step
Epoch 3/100
1/1 - 0s - loss: 0.1387 - val_loss: 4.7920e-05 - 48ms/epoch - 48ms/step
Epoch 4/100
1/1 - 0s - loss: 0.1381 - val_loss: 9.1991e-05 - 46ms/epoch - 46ms/step
Epoch 5/100
1/1 - 0s - loss: 0.1375 - val_loss: 1.5006e-04 - 47ms/epoch - 47ms/step
Epoch 6/100
1/1 - 0s - loss: 0.1369 - val_loss: 2.1757e-04 - 58ms/epoch - 58ms/step
Epoch 7/100
1/1 - 0s - loss: 0.1363 - val_loss: 2.8489e-04 - 46ms/epoch - 46ms/step
Epoch 8/100
1/1 - 0s - loss: 0.1357 - val_loss: 3.4136e-04 - 46ms/epoch - 46ms/step
Epoch 9/100
1/1 - 0s - loss: 0.1351 - val_loss: 3.8232e-04 - 46ms/epoch - 46ms/step
Epoch 10/100
1/1 - 0s - loss: 0.1345 - val_loss: 4.0940e-04 - 46ms/epoch - 46ms/step
Epoch 11/100
1/1 - 0s - loss: 0.1339 - val_loss: 4.2735e-04 - 47ms/epoch - 47ms/step
Epoch 12/100
1/1 - 0s - loss: 0.1333 - val_loss: 4.4166e-04 - 47ms/epoch - 47ms/step
Epoch 13/100
1/1 - 0s - loss: 0.1327 - val_loss: 4.5754e-04 - 47ms/epoch - 47ms/step
Epoch 14/100
1/1 - 0s - loss: 0.1321 - val_loss: 4.7908e-04 - 46ms/epoch - 46ms/step
Epoch 15/100
1/1 - 0s - loss: 0.1315 - val_loss: 5.0905e-04 - 53ms/epoch - 53ms/step
Epoch 16/100
1/1 - 0s - loss: 0.1309 - val_loss: 5.4912e-04 - 50ms/epoch - 50ms/step
Epoch 17/100
1/1 - 0s - loss: 0.1303 - val_loss: 6.0029e-04 - 50ms/epoch - 50ms/step
Epoch 18/100
1/1 - 0s - loss: 0.1297 - val_loss: 6.6305e-04 - 49ms/epoch - 49ms/step
Epoch 19/100
1/1 - 0s - loss: 0.1291 - val_loss: 7.3734e-04 - 48ms/epoch - 48ms/step
Epoch 20/100
1/1 - 0s - loss: 0.1286 - val_loss: 8.2273e-04 - 48ms/epoch - 48ms/step
Epoch 21/100
1/1 - 0s - loss: 0.1280 - val_loss: 9.1799e-04 - 54ms/epoch - 54ms/step
```

Figure 5.48: Computation Network 3-100-100

From 0 to 80 epochs the training and validation accuracy curve are steady until it moved forward from 90 epochs with a loss of 0.1. The fact that the training and validation accuracy data are going up means that the learning process was efficient. Figure 5.49 presents the Loss and Validation accuracy network 3-100-100.

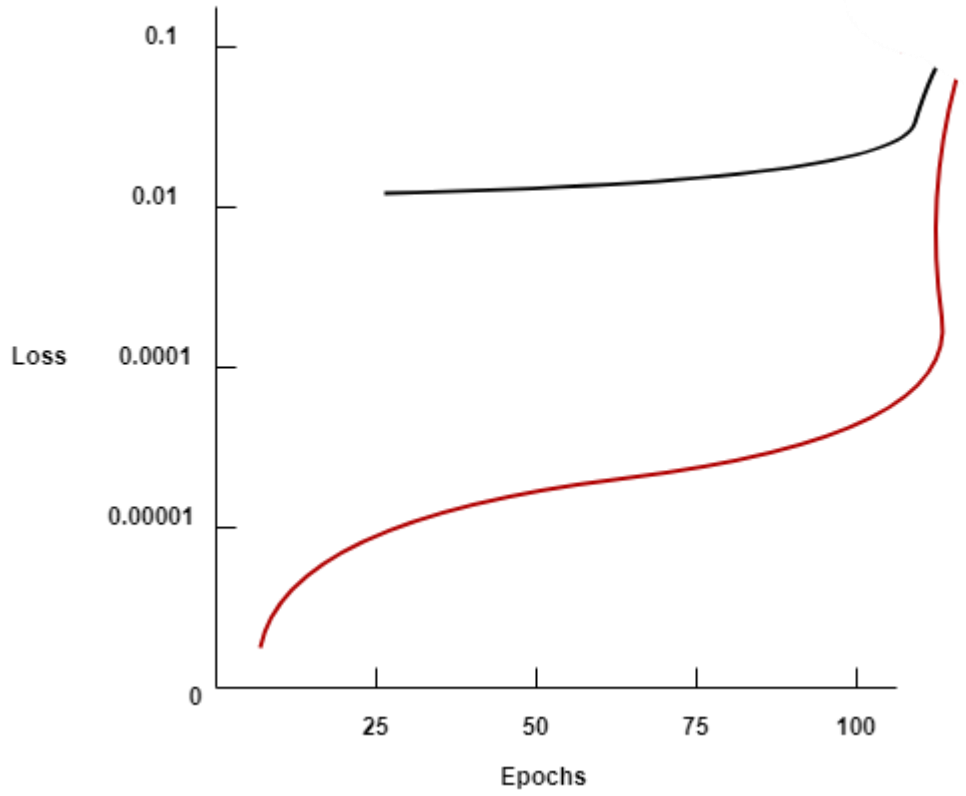

Figure 5.49: Loss vs Validation accuracy Network 3-100-100.

5.11.2 50 epochs with 200 LSTM

There are different numbers of epochs and LSTMS layers in the arrangement that follows. It was successful to meet the performance target of fewer than 0 as error. The error decreased slightly over the iterations, reaching less than 0.001 around the 10th epoch, but the results were not significantly affected. Figures 5.50 and 5.51 present the Computation Network 3-50-100 and the Loss vs Validation accuracy of the model, respectively.

```
Epoch 1/50
1/1 - 4s - loss: 0.1400 - val_loss: 4.6526e-06 - 4s/epoch - 4s/step
Epoch 2/50
1/1 - 0s - loss: 0.1393 - val_loss: 2.1142e-05 - 66ms/epoch - 66ms/step
Epoch 3/50
1/1 - 0s - loss: 0.1386 - val_loss: 5.6025e-05 - 64ms/epoch - 64ms/step
Epoch 4/50
1/1 - 0s - loss: 0.1379 - val_loss: 1.1001e-04 - 75ms/epoch - 75ms/step
Epoch 5/50
1/1 - 0s - loss: 0.1372 - val_loss: 1.8726e-04 - 68ms/epoch - 68ms/step
Epoch 6/50
1/1 - 0s - loss: 0.1365 - val_loss: 2.9120e-04 - 62ms/epoch - 62ms/step
Epoch 7/50
1/1 - 0s - loss: 0.1358 - val_loss: 4.2693e-04 - 60ms/epoch - 60ms/step
Epoch 8/50
1/1 - 0s - loss: 0.1351 - val_loss: 6.0013e-04 - 64ms/epoch - 64ms/step
Epoch 9/50
1/1 - 0s - loss: 0.1344 - val_loss: 8.1643e-04 - 64ms/epoch - 64ms/step
Epoch 10/50
1/1 - 0s - loss: 0.1338 - val_loss: 0.0011 - 64ms/epoch - 64ms/step
Epoch 11/50
1/1 - 0s - loss: 0.1332 - val_loss: 0.0013 - 84ms/epoch - 84ms/step
Epoch 12/50
1/1 - 0s - loss: 0.1326 - val_loss: 0.0015 - 62ms/epoch - 62ms/step
Epoch 13/50
1/1 - 0s - loss: 0.1320 - val_loss: 0.0016 - 62ms/epoch - 62ms/step
Epoch 14/50
1/1 - 0s - loss: 0.1313 - val_loss: 0.0016 - 61ms/epoch - 61ms/step
Epoch 15/50
1/1 - 0s - loss: 0.1307 - val_loss: 0.0016 - 59ms/epoch - 59ms/step
Epoch 16/50
1/1 - 0s - loss: 0.1301 - val_loss: 0.0015 - 58ms/epoch - 58ms/step
Epoch 17/50
1/1 - 0s - loss: 0.1295 - val_loss: 0.0014 - 60ms/epoch - 60ms/step
Epoch 18/50
1/1 - 0s - loss: 0.1288 - val_loss: 0.0014 - 58ms/epoch - 58ms/step
Epoch 19/50
1/1 - 0s - loss: 0.1282 - val_loss: 0.0013 - 59ms/epoch - 59ms/step
Epoch 20/50
1/1 - 0s - loss: 0.1276 - val_loss: 0.0014 - 59ms/epoch - 59ms/step
Epoch 21/50
1/1 - 0s - loss: 0.1270 - val_loss: 0.0014 - 59ms/epoch - 59ms/step
```

Figure 5.50: Computation Network 3-50-100

From 15 to 30 epochs the training and validation curve is stable until 40th epochs were it went up with a loss of 1. The fact that training and validation data have been correctly partitioned means that the network is fitting. Figure 5.51 depicted the Loss and Validation accuracy network 3-50-100.

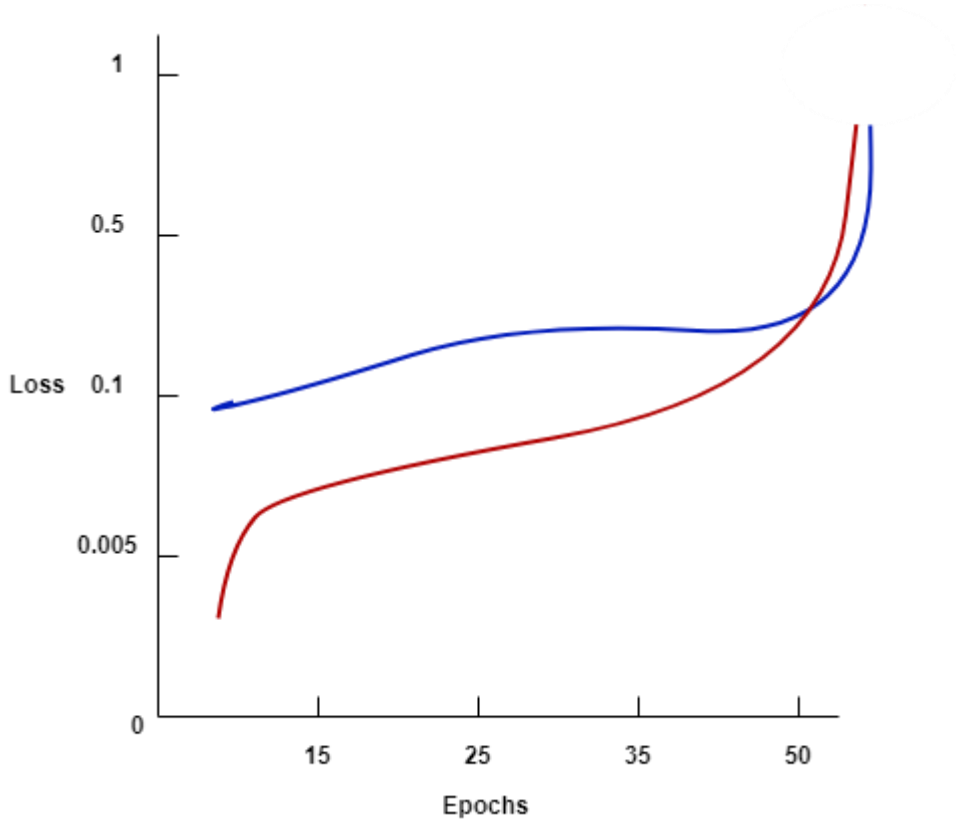

Figure 5.51: Loss vs Validation accuracy Network 3-50-100

5.11.3 100 EPOCHS TO 200 LSTM

The following layout has varying numbers of epochs and LSTMS layers, respectively 100 and 200. Meeting the performance goal of less than 0 errors was successful. The error did not dramatically alter the results, but it did slightly reduce during the iterations, reaching less than 0.001 around the 15th epoch. Figures 5.52 and 5.53 depict the Computation Network 3-100-100 and Loss vs Validation accuracy for the Network, respectively.

```
Epoch 6/100
1/1 - 0s - loss: 0.1373 - val_loss: 0.0061 - 58ms/epoch - 58ms/step
Epoch 7/100
1/1 - 0s - loss: 0.1388 - val_loss: 0.0101 - 56ms/epoch - 56ms/step
Epoch 8/100
1/1 - 0s - loss: 0.1411 - val_loss: 0.0099 - 82ms/epoch - 82ms/step
Epoch 9/100
1/1 - 0s - loss: 0.1403 - val_loss: 0.0065 - 57ms/epoch - 57ms/step
Epoch 10/100
1/1 - 0s - loss: 0.1373 - val_loss: 0.0029 - 54ms/epoch - 54ms/step
Epoch 11/100
1/1 - 0s - loss: 0.1346 - val_loss: 0.0011 - 63ms/epoch - 63ms/step
Epoch 12/100
1/1 - 0s - loss: 0.1338 - val_loss: 0.0010 - 56ms/epoch - 56ms/step
Epoch 13/100
1/1 - 0s - loss: 0.1345 - val_loss: 0.0014 - 64ms/epoch - 64ms/step
Epoch 14/100
1/1 - 0s - loss: 0.1350 - val_loss: 0.0014 - 60ms/epoch - 60ms/step
Epoch 15/100
1/1 - 0s - loss: 0.1343 - val_loss: 9.4508e-04 - 58ms/epoch - 58ms/step
Epoch 16/100
1/1 - 0s - loss: 0.1326 - val_loss: 9.8863e-04 - 55ms/epoch - 55ms/step
Epoch 17/100
1/1 - 0s - loss: 0.1310 - val_loss: 0.0020 - 60ms/epoch - 60ms/step
Epoch 18/100
1/1 - 0s - loss: 0.1301 - val_loss: 0.0035 - 60ms/epoch - 60ms/step
Epoch 19/100
1/1 - 0s - loss: 0.1301 - val_loss: 0.0048 - 56ms/epoch - 56ms/step
Epoch 20/100
1/1 - 0s - loss: 0.1301 - val_loss: 0.0051 - 56ms/epoch - 56ms/step
Epoch 21/100
1/1 - 0s - loss: 0.1297 - val_loss: 0.0044 - 59ms/epoch - 59ms/step
Epoch 22/100
1/1 - 0s - loss: 0.1287 - val_loss: 0.0031 - 70ms/epoch - 70ms/step
Epoch 23/100
1/1 - 0s - loss: 0.1275 - val_loss: 0.0019 - 61ms/epoch - 61ms/step
Epoch 24/100
1/1 - 0s - loss: 0.1266 - val_loss: 0.0013 - 73ms/epoch - 73ms/step
Epoch 25/100
1/1 - 0s - loss: 0.1262 - val_loss: 0.0011 - 57ms/epoch - 57ms/step
Epoch 26/100
1/1 - 0s - loss: 0.1260 - val_loss: 0.0011 - 57ms/epoch - 57ms/step
Epoch 27/100
1/1 - 0s - loss: 0.1257 - val_loss: 0.0012 - 59ms/epoch - 59ms/step
```

Figure 5.52: Computation Network 3-100-100

The training signal accuracy was at first slow from 0 to 25th epochs and validation was stable around 0.01 until both at 25th epochs rose up to 0.1 accuracy and probably much more if training was considered on longer timeframe. Figure 5.53 showed the Loss and Validation Accuracy Network 3-100-100.

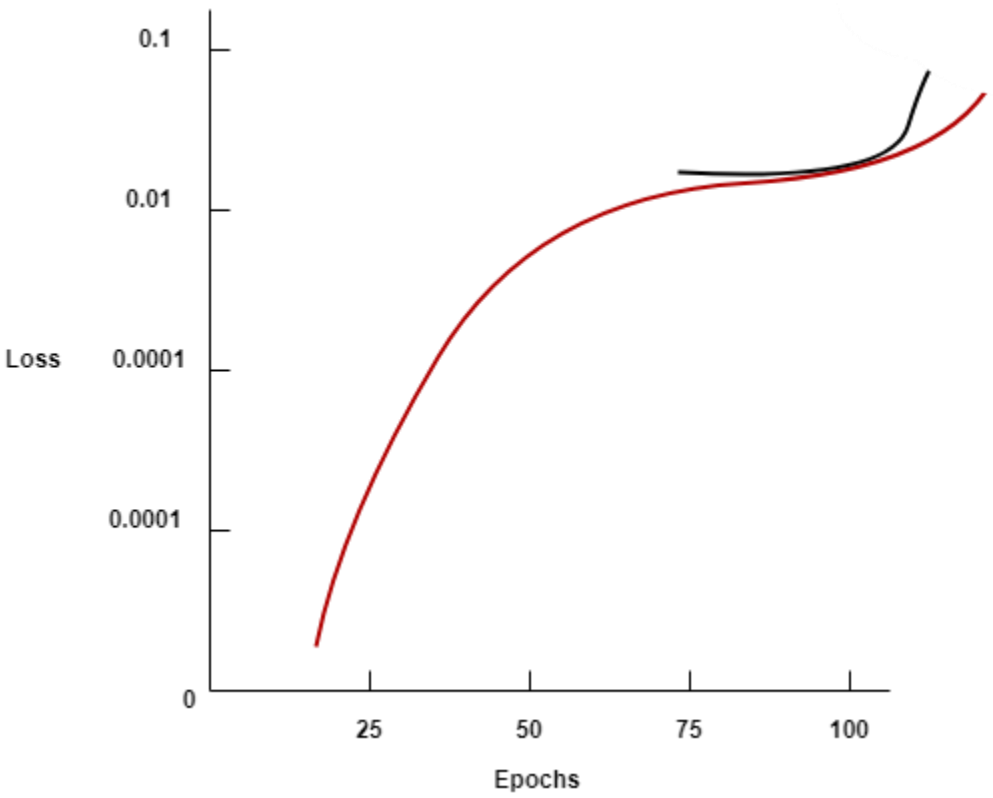

Figure 5.53: Loss vs Validation Accuracy Network 3-100-100.

It can be seen that there is a convergence at. The Best performance analysis carried out on the network was the ROC from the Network of Figure 5.53. To evaluate the NN classifier's quality, this plot was employed. Threshold values are applied to the outputs for each class over the range [0, 1]. The number of outputs more than or equal to the threshold, divided by the number of true (1s) targets, and the number of outputs less than the threshold, divided by the number of untrue (0s) targets, are the two values that are calculated for each threshold. Figure 5.54 depicts the ROC Network 3, where the curve tends toward the top and left margins of the plot but not towards the edge. This suggests that we haven't yet attained the highest precision, but it is a satisfactory result over 50%.

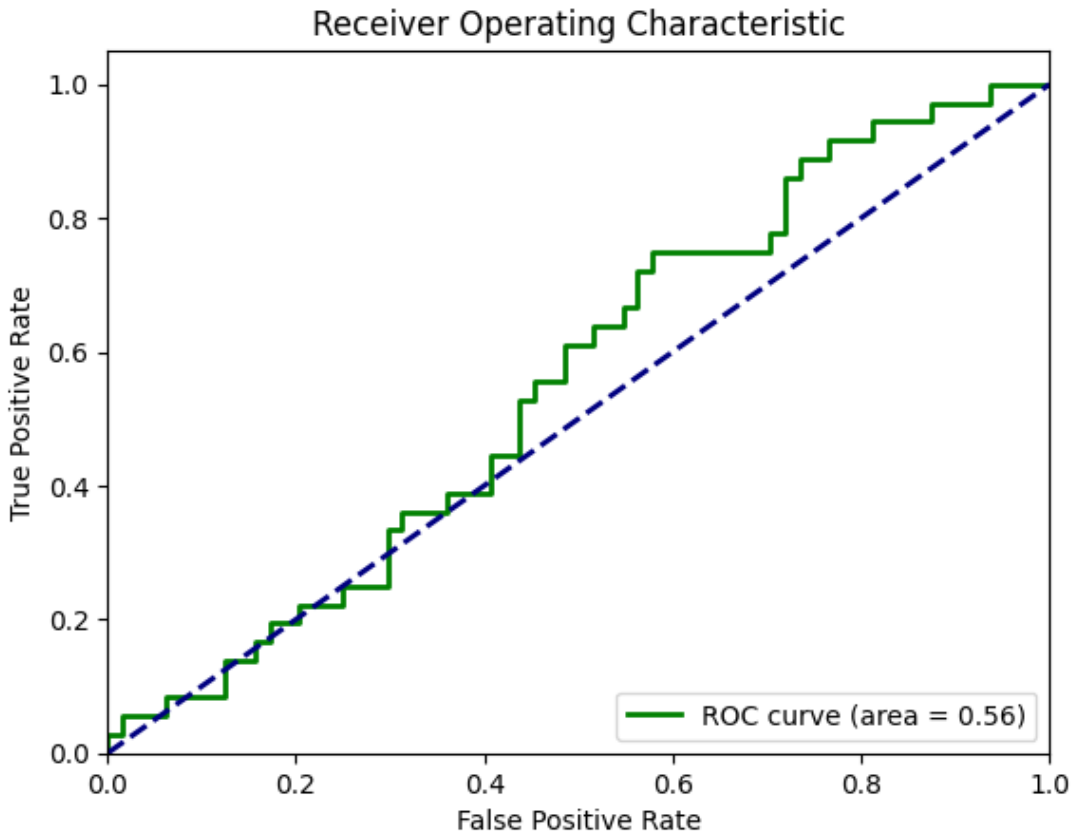

Figure 5.54: ROC Network 3

5.11.4 Three-phase to ground

Similar to the three previous scenarios, the number of epochs and layers was altered over the iterations in order to see when the model performed the best. Table 5.2 shows the results obtained for three-phase to ground faults.

Table 5.2: Results obtained from three-phase to ground faults

Best Validation results	Epochs number	LSTM layers
0.00082	50	100
0.00005	100	100
0.00001	50	200
0.00006	100	200

Figure 5.55 was used to assess the quality of the NN classifier obtained from the Network with 200 LSTM and fewer epochs than usual. The outputs for every class are subjected to threshold values within the interval [0, 1]. For each threshold, two values are calculated: the number of outputs less than the threshold, divided by the number

of untrue (0s) targets, and the number of outputs more than or equal to the threshold, divided by the number of true (1s) targets. The ROC plot is shown in Figure 5.55, with the curve of an overall percentage of 60%. Which is satisfactory as it is above the 50% average.

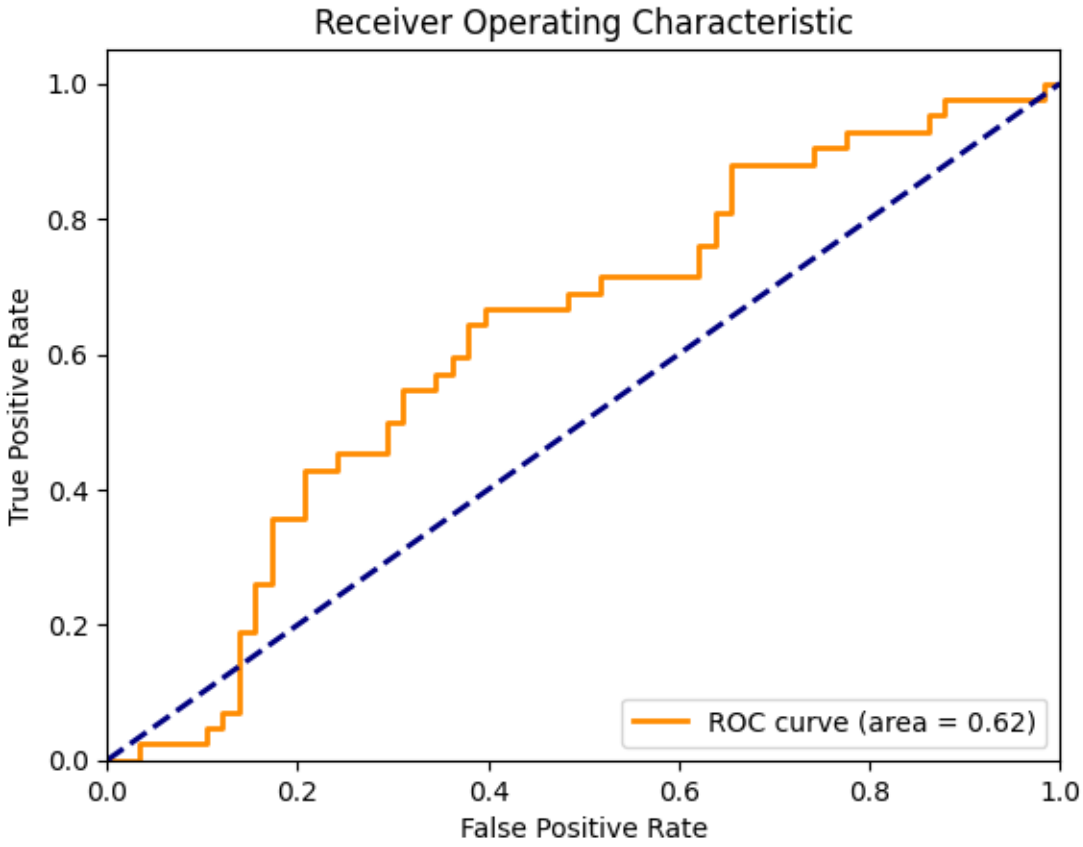

Figure 5.55: ROC Network 4

5.12 Classification of faults

In this section, the classification algorithm is introduced for the base case by following the procedures presented in the Appendix 2. Classification of faults lie on differentiating the type of faults whether it is a single or double phase fault and so forth, while detection means identifying the disturbance that matches the condition that considered as faults. The Single phase, two phase and, three phase faults, as mentioned above, has been applied at Bus 632. Table 5.3 explains the classification algorithm

Table 5.3: Fault Classification Algorithm

Step	Description
1. Data Definition	The three vectors resulting from the three voltage measurements from Bus 632 are extracted and converted into an array. (Clarified "measurement" to "measurements" for subject-verb agreement)
2. Creation of a DataFrame	The data frame is used with (df). Consider clarifying what the DataFrame contains, e.g., voltage readings or time series data. Also, "daft" corrected to "data".
3. Classification Function	The classify_fault() function actually looks at those three voltage vectors and considers that if, at some point in time, the value of any voltage is equal to 0, it means that there is a short circuit fault. Depending on the fault, the function checks if the fault is repeating itself in the other vectors simultaneously. Based on the length of faulty phases, it returns a string indicating the type of fault and specifies the faulty phases. (Grammar corrected and flow improved slightly without changing your intent. Consider breaking into two sentences for readability.)
4. Apply Classification	Then it is applied to each sequence of the arrays. (Minor grammar fix: “in” → “to”)
5. Display the Results	The results are displayed and are ready to be trained and tested using the proposed LSTM model. (Grammar: "and ready" changed to "and are ready")

The computation of IEEE 13 bus system was done in Python, and introduce different types of faults at 5 seconds in bus 632. Three phase short circuit faults example are respectively presented in Figure show the computation of the bus 632 from API. Table 5.4 presents the binary classifier used in the study to classify the faults. A slight change is implemented into the LSTM with the introduction of a Softmax function instead of a Sigmoid.

Table 5.4: Binary classifier

Type of faults	Network output			
	A	B	C	G
A-G faults	1	0	0	1
B-G faults	0	1	0	1
C-G faults	0	0	1	1
A-B faults	1	1	0	0
B-C faults	0	1	1	0
A-C faults	1	0	1	0
ABC faults	1	1	1	0
ABC-G faults	1	1	1	1
AB-G faults	1	1	0	1
BC-G faults	0	1	1	0

AC-G faults	1	0	1	1
A faults	1	0	0	0
B faults	0	1	0	0
C faults	0	0	1	0

5.13 Discussion of the results

Table 5.5 presents the results obtained from the fault Classification. The focus was on ROC and Mean Squared Error as evaluation criteria of the whole system.

Table 5.5: Faults Classification Output

Fault Types	Fault resistance (Ω)	Mean Square Error	Fault class	ROC
A-G faults	2.5	0.04830487	1ph	65%
B-G faults	2.5	0.02718176	1ph	68%
C-G faults	2.5	0.04522349	1ph	70%
A-B faults	2.5	0.04503655	1ph	60%
B-C faults	2.5	0.06157168	1ph	58%
A-C faults	2.5	0.02689977	1ph	60%
ABC faults	0.5	0.02987322	3ph	63%
ABC-G faults	0.5	0.05373997	3ph	61%
AB-G faults	0.5	0.01642987	2ph	60%
BC-G faults	0.5	0.03961619	2ph	59%
AC-G faults	0.5	0.02005851	2ph	70%
A faults	2.5	0.01879908	1ph	69%
B faults	2.5	0.03794814	1ph	63%
C faults	2.5	0.02359094	1ph	68%

5.13.1 Impacts of the number of epochs

The model evaluated the effect of increasing the number of epochs for different short circuit faults. The performance target was fixed, and the number of neurons in the hidden layer of the worst-performing network was kept constant while examining how increasing the number of epochs (iterations) impacted the results. As the number of iterations increased, so did the training time. Upon reviewing the performance plots, it was observed that the mean squared error (MSE) shifted within a certain range as the number of epochs increased from a lower value to 1000 epochs. A crucial consideration is balancing the improvement in the MSE with the potential decrease in the ANN's generalization ability as the number of epochs grows.

5.13.2 Impacts of the number of LSTM

The impact of adding more hidden layers on the performance of the ANN was analyzed for all fault types. Specifically, the effect of using 2 hidden layers was studied with the number of neurons (H) ranging from 5 to 25, where 25 is the maximum allowed by the network equation and weight rule. Increasing the number of hidden layers enhanced the ANN's performance for single-phase-to-ground faults, resulting in a lower mean squared error (MSE) compared to a single-layer network. Excellent regression values were achieved, though the performance difference between the final architecture and the two-hidden-layer networks was minimal. Similar results were observed for two-phase, two-phase-to-ground, and three-phase faults as well.

5.13.3 Activation function

In an LSTM neural network designed to detect faults in a power system, activation functions play a crucial role in determining the network's ability to model and learn complex temporal patterns. LSTM networks utilize various gates (input, forget, and output gates) controlled by sigmoid activation functions to regulate the flow of information through the cell states. The sigmoid activation function ensures that the gate values are within a range of 0 to 1, enabling the network to decide which information to retain or forget at each time step. Additionally, the tanh activation function is used within the LSTM cells to maintain the cell state values within a range of -1 to 1, which helps in normalizing the network's output and maintaining stability during training. By effectively managing these dynamics, the activation functions in LSTM networks enable the model to accurately capture and predict the intricate patterns of power system behaviour, thereby improving the detection and classification of faults with high precision and reliability.

5.13.4 Performance of the classification and detection algorithm

A separate test dataset, different from the one used to train the neural networks, was utilized. This involved simulating faults at locations distinct from those used during training, with different fault resistances and fault inception angles. Some of the results are summarized. The fault detection achieved 100% accuracy, and the results for fault classification, section identification, and location were highly promising. The highest errors were observed for high-resistance faults. For example, a 1.53% error was recorded for a B-G fault with 2.5Ω resistance and a 60° fault inception angle at location 854. A 2.60% error was recorded for the same fault at the same location and fault angle, but with 5Ω resistance. In general, the percentage errors for the test dataset were all under 3.0%. The performance test u demonstrates the effectiveness of the proposed method, provided that the fault type is accurately identified to trigger the correct algorithms for fault section identification and location.

5.14 Proposed modified SMOTE technique for performance improvement

Problems like learning enough models or class imbalance are attempted to be resolved in deep learning and other neural networks by distinct approaches. Data augmentation, in particular, is a widely used editing technique that makes the model more resilient to small changes in input data. It is also used to increase the size of the training data set. Data augmentation called augmentation enables the artificial creation of fresh training data from pre-existing training data. Within the device. When performing data augmentation for learning applications like image recognition or video processing, minor adjustments or noise are typically applied to pre-existing samples. Furthermore, data augmentation is frequently used in natural language processing. Given that altering the word's order also modifies the meaning of natural language processing, the term "augmented data" refers to something completely different from "image data." It is frequently used in text and visual data, according to scientific studies. Generally speaking, the primary goal of data augmentation is to identify potential differences between the samples that the model uses for training. In this study, measurements of voltage and current are made on various bus of the system with the use of sensors. Machine learning and statistical data analysis methods are used during data augmentation.

The contribution brought to this area is as follows:

i. Application of data augmentation to enhance accuracy of LSTM based Detection model is proposed, with the usage of SMOTE as an augmentation method.
ii. Experimental data gathered on PYTHON are later checked with evaluation metrics such as mean square error, confusion matrix and F1 score.

5.15 Description of the Deep Learning Method

This method incorporates SMOTE into the deep learning model's training pipeline. During each epoch, the training data is injected with synthetic minority class examples produced by SMOTE, guaranteeing that the model obtains a balanced distribution of class instances. Then, in order to minimize the loss function and enhance classification performance, the deep learning model is trained using conventional optimization techniques as Adam or stochastic gradient descent (SGD).

5.16 Problem Formulation

The proposed SMOTE algorithm is performed for generating additional data during the training stage of the algorithm in order to improve the general performance of the model and upgrade the ROC as well as the Confusion matrix.

5.17 Class Imbalance Processing

Let p_X(x) represent the original probability density of the minority class, and let x_0 represent a candidate minority class sample for SMOTE interpolation. Z is the name of the artificially produced sample produced by SMOTE. Let d be the class's dimension design. The definition of r is the Euclidean distance between the minority class sample x_0 and its selected k^{th} neighbour.

We present a mathematical study of the SMOTE oversampling technique created by Chawla et al. (2002). In particular, we calculate the general minority class density p_X(x) and the probability density function of the SMOTE-generated patterns, p_Z(x).

Theorem 1: For each random variable x, let x be a random sample. Let Z be a random variable, defined by Algorithm 1 as a random linear interpolation between x_k K-nearest neighbors. Then, the probability density of Z is given by:

$$p_Z(z) = (N-K)\binom{N-1}{K}\int_x p_X(x)\int_{r=\|z-x\|}^{\infty} p_X\left(x+\frac{(z-x)r}{\|z-x\|}\right)\left(\frac{r^{d-2}}{\|z-x\|^{d-1}}\right).B\left(1-I_{B(x,r)};N-K-1\right)drdx \quad (5.1)$$

Where $B\left(1-I_{B(x,r)};N-K-1\right)drdx$ is the incomplete Beta Function

Lemma 1: Given a specific minority class sample x_0 , the conditional probability density function of the SMOTE generated patterns x_0, p_Z(x), is assessed as follows:

$$p_Z(z) = (N-K)\binom{N-1}{K}\int_{r=\|z-x\|}^{\infty} p_X\left(x+\frac{(z-x)r}{\|z-x\|}\right)\left(\frac{r^{d-2}}{\|z-x\|^{d-1}}\right)\times B\left(1-I_{B(x,r)};N-K-1\right)dr \quad (5.2)$$

In essence, this theorem provides a complete analytical solution for the probability density of the samples that SMOTE created. The formulas are precise and don't use any approximations or assumptions.

5.18 Explanation of the proposed SMOTE algorithm

The SMOTE actually helped to oversample the data that was available and create relatively similar data to that one in order to improve the performance of the model and normalise the data. Figure 5.56 shows the SMOTE Algorithm that was used.

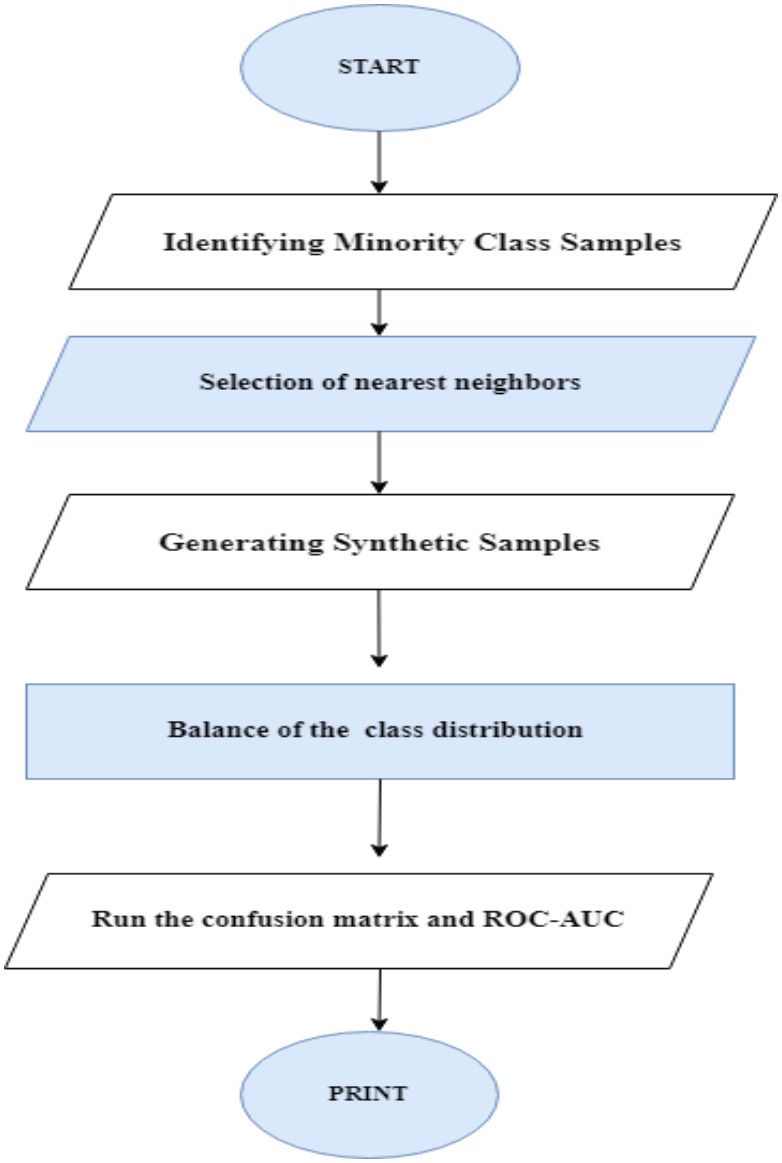

Figure 5.56: SMOTE Algorithm

5.18.1 Single phase short circuit at bus 632

Based on the datasets of the y_test and y_pred from the detection model, SMOTE was used in this case to duplicate the number of data in minority class. So that we get an acceptable number of true positive in the confusion matrix as well as an improvement of the Receiver Operating Characteristic. The algorithm described in Appendix. 3 was used. The make classification function was used to generate synthetic datasets. The sample consists of 10000 datasets with 2 features and 5 k neighbours that are used for duplication and creating synthetic values. The Algorithm was run in Python and provides the following results. With Number of samples = 10000; k neighbours = 5, random state = 42. Figure 5.57 shows the ROC of the model, which is more efficient

when we introduced the SMOTE to oversample the data, as we get an AUC equal to 0.96. Which means that the model has a high efficiency.

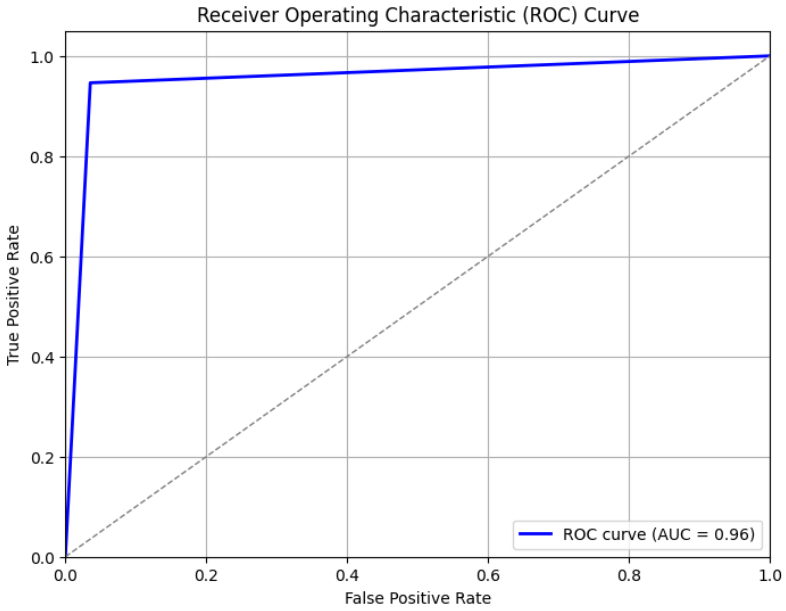

Figure 5.55: ROC of the proposed model

Table 5.6 presents the confusion matrix; it can be seen that it presented exceptional results with more than 80% of positive results. More than 10000 faults have been simulated with several parameters to test the network. Three phase faults were the most simulated with up to 841 faults found right.

Table 5.6: Confusion Matrix

A-G	99	0	0	0	0	0	0	0	0	0	0	0	0	0
B-G	0	656	0	0	0	0	0	0	0	0	0	0	0	0
C-G	0	0	192	0	0	0	0	0	0	0	0	0	0	0
A-B	0	0	0	315	0	0	0	0	0	0	0	0	0	0
B-C	0	0	0	0	208	0	0	0	0	0	0	0	0	0
A-C	0	0	0	0	0	296	0	0	0	0	0	0	0	0
ABC	0	0	0	0	0	0	58	0	0	0	0	0	0	0
ABCG	0	0	0	0	0	0	0	10	0	0	0	0	0	0
ABG	0	0	0	0	0	0	0	0	184	0	0	0	0	0
BCG	0	0	0	0	0	0	0	0	0	841	0	0	0	0
AC-G	0	0	0	0	0	0	0	0	0	0	39	0	0	0
A	0	0	0	0	0	0	0	0	0	0	0	48	0	0
B	0	0	0	0	0	0	0	0	0	0	0	0	189	
C	0	0	0	0	0	0	0	0	0	0	0	0	0	629
A-G	**C**	**B**	**A**	**ACG**	**BCG**	**ABG**	**ABCG**	**ABC**	**AC**	**BC**	**AB**	**CG**	**BG**	**AG**

5.19 Discussion of the results

The results from the LSTM model for fault detection indicate strong performance in identifying and classifying faults. The MSE is low, suggesting that the model's

predictions are close to the actual values. The True Positive Rate is high, demonstrating the model's accuracy in correctly identifying faults. The Receiver Operating Characteristic curve shows a high area under the curve (AUC), indicating that the model has a good balance between sensitivity and specificity as compared to the model before implementation of the SMOTE. Finally, the Confusion Matrix reveals a high number of correctly classified instances, with minimal false positives and false negatives, further confirming the model's effectiveness in fault detection. Table 5.7 shows the Results analysis after SMOTE implementation

Table 5.7: Results analysis after SMOTE implementation

Faults	Receiver Operating Characteristic	True Positive	False negative	Confusion Matrix
One phase	0.96	1936	1859	90%
Two phase	0.9	1961	1834	96%
Three phase	0.93	1950	1833	95%
Three phase to ground	0.96	1880	1900	98%

From the results that the model provided, it has been noticed that SMOTE has an effect on the model performance. It actually helped to distribute equally the data during the process. And obtaining a model that responding accordingly judging by the rate of true positives and false negative.

When working with imbalanced datasets, the SMOTE has shown to be a highly effective method for improving fault classification in deep learning models. Fault detection datasets are skewed in many real-world circumstances, having a disproportionately small number of examples of fault conditions relative to normal operating states. This disparity may result in skewed models that detect errors less precisely. In order to solve this problem and balance the dataset, SMOTE creates synthetic samples of the minority class. This allows the deep learning model to learn from both fault and non-fault occurrences more efficiently. As a result, the model exhibits increased fault classification accuracy and robustness, lowering the possibility of defects going undiscovered and raising system reliability. The ROC significantly improved in all the short circuits scenarios from an average of 60% to 90%. The Appendix show the confusion matrix obtained for the two-phase, three-phase and ground faults, with more or less the same results and greater efficiency before the implementation of the SMOTE.

5.20 Comparative analysis

Two well-known programming languages, each having specific advantages and uses in the fields of computational engineering and data science, are Python and MATLAB. The need for reliable and powerful computational tools is increasing, so it's critical to evaluate and contrast these languages using performance metrics.

5.21 Metrics for comparison

Performance metrics offer a numerical framework for assessing these technologies' many features, including scalability, memory use, execution speed, and ease of hardware and software integration. Users can decide which language best fits their unique demands and limits by methodically examining these indicators. The Following metrics will be used for deep analysis of the two softwares: precision, accuracy, recall, training time, prediction time, model complexity and Resource utilization.

5.21.1 Data for preparation

Python is a top language for artificial intelligence and machine learning, providing sophisticated tools and frameworks to improve error detection and prediction algorithms. Table 5.8 presents the Features of MATLAB & PYTHON.

Table 5.8: Features of MATLAB & PYTHON

Parameters	MATLAB	PYTHON
Library used	Deep learning toolbox	Tensorflow, Keras
Advantages	- For power system analysis and machine learning applications, MATLAB provides specialized toolboxes including Simscape Electrical, Power System Toolbox, and the Deep Learning Toolbox. Pre-built models and functionality found in these toolboxes can help save time and effort. - Complex power system model development and analysis benefit from MATLAB's integrated environment, which integrates computation, visualization, and programming. - PYTHON is a flexible tool for creating and evaluating fault analysis algorithms since it can be integrated with different	- PYTHON is a flexible tool for creating and evaluating fault analysis algorithms since it can be integrated with different pieces of hardware and software - For creating sophisticated fault detection and mitigation techniques, PYTHON offers strong optimization and control design capabilities. - Python can manage large-scale data processing and boost performance with libraries like Dask and tools like Cython, which makes it appropriate for demanding fault analysis jobs. - Python's syntax is simple to understand and intuitive, making it suitable for novices

	pieces of hardware and software.	while also offering sufficient capability for more experienced users. - Because of its sizable and vibrant community, Python offers a plethora of courses, documentation, and forums for problem-solving and knowledge-sharing. - Due to its open-source nature and lack of cost, Python is a desirable choice for individuals, startups, and enterprises with tight budgets.
Disadvantages	- For large-scale simulations and computations, MATLAB can be slower than other compiled languages like C/C++; this could be a drawback for particularly big or sophisticated power system models. - Because of their close ties to the MATLAB environment, MATLAB models and scripts may not be as portable to other systems. - Although MATLAB toolboxes come with a lot of built-in functions, users may need to develop their own functions or scripts because the toolboxes may not always offer the flexibility needed for highly specialized studies.	- Although Python has a wealth of robust libraries, some of them might not be as developed or tailored as MATLAB toolboxes for specific specialized uses in power systems. - In Python, it's frequently necessary to piece together various libraries to obtain the same capability, which can be more complex and time-consuming than in MATLAB, which offers toolboxes specifically developed for power system research.

5.22 Analysis and Discussion

Performance, efficiency, and usability differences between MATLAB and Python are demonstrated in the training assessment of fault analysis models. Model building and training are made easier by MATLAB, which has built-in support for a number of

machine learning methods and makes use of its Deep Learning Toolbox. Because of its intuitive interface and integrated tools tailored for engineering applications, MATLAB's training process may be set up more quickly. Python, on the other hand, is more flexible and has a larger selection of machine learning libraries, such as PyTorch, Tensor Flow, and Keras.

But this flexibility necessitates more labour-intensive setup and configuration by hand, which could make the training procedure more difficult. Furthermore, even while Python performs similarly, it could need more tweaking to be optimized. Overall, the analysis shows that although Python has a steeper learning curve, it offers more sophisticated customization choices and gives a deeper understanding of the training experience. Table 5.9 shows the best performance analysis obtained in Python against the MATLAB training.

Table 5.9: the best performance analysis

Software	Epochs	Learning rate	Training loss
MATLAB	50	0.001	0.2863358
	100	0.001	0.0086258
	200	0.001	0.0050513
PYTHON	50	0.001	0.0198723
	100	0.001	0.0046311
	200	0.001	0.0050513

Table 5.10 presents a comparison between the model in Python with an equivalent model built in MATLAB and Model complexity. MATLAB is effective for electrical implementation and behaviour analysis but lacks flexibility and tools for proper algorithm development with neural network. While PYTHON is easy for program customization and huge database analysis or processing.

Table 5.10: Comparison between the model in Python with an equivalent model built in MATLAB

Parameters	MATLAB	PYTHON
Integration of deep learning	Measurements made were sent to a MATLAB script, then made as a variable. Then deep learning toolbox was used to build an LSTM with a Relu and Softmax function for classification.	Data were acquired via an API that was built using Google Sheets. The Python model has the potential to simulate in real time.
Data manipulation	It was quite easy since blocks were already provided in the	It was more complex but gave a better understanding of what

	library, but it made the customisation difficult. Because each block script is specifically designed for a purpose.	was aimed as an overall goal. Since the Python library allowed users to use many tools for data manipulation, Smote was very useful for this purpose.

5.23 Summary

This research focused on fault detection and classification in grid-tied PV systems using LSTM neural networks. Traditional methods struggles to achieve high accuracy due to the dynamic nature of grid-connected PV systems and varying fault scenarios. The research fills a methodological gap by leveraging LSTM's ability to process time-series data, enabling precise identification of short-circuit faults. A comprehensive synthetic dataset is generated, and data augmentation techniques like SMOTE is applied to address class imbalance. The LSTM model demonstrates superior accuracy, faster response times, and higher detection rates compared to traditional methods. The study highlights the advantages of an open framework for fault analysis, contributing to advancing deep learning applications for enhancing PV system safety and reliability.

CHAPTER SIX

CONCLUSION AND RECOMMENDATIONS

6.1 Conclusions

Deep learning in particular, a branch of artificial intelligence (AI), is revolutionizing fault analysis in the distribution system by providing previously unheard-of capacities for problem identification, prediction, and mitigation. The inadequacy of conventional fault detection techniques increases with the complexity and pressure placed on contemporary power distribution networks. These techniques frequently rely on pre-established rules or models, which are ill-equipped to handle the massive volumes of data and complex patterns connected to electrical failures. This is where artificial intelligence , and deep learning in particular, comes in to offer a revolutionary solution. Deep learning models LSTM networks are especially well-suited. Because of their superior time-series data analysis capabilities, these models are perfect for identifying and categorizing defects in electrical systems where the order of events is critical. Deep learning models can identify tiny trends that can point to an imminent defect by learning from past data, which enables early diagnosis and intervention. With less downtime and maintenance expenses, this predictive capability greatly improves the distribution system's efficiency and reliability.
Furthermore, deep learning can adapt to the dynamically shifting nature of power systems, increasing its accuracy over time, thanks to its capacity to handle and learn from enormous datasets. When fresh data becomes available, deep learning models can automatically update their parameters, in contrast to previous methods that might need a lot of manual tuning and recalibration. This flexibility is essential in an industry where working conditions are ever-changing.

The contribution of this model was that the developed LSTM model was able to be customize along the simulation and validate both the primary objective which was to detect four specific types of faults: Single, two, three phase faults to grounds faults and prove that SMOTE had the ability to customize the data in such way that it could improve the overall performance of a given LSTM model. First, the three shorts circuits detection and classification scenarios were simulated with two different case one which only test the IEEE 13 bus then another one includes the implementation of a 2.5MW PV system. It was noticed that the ROC was acceptable around 60% which means that the model prediction was average. Secondly, the SMOTE algorithm was implemented in order to improve the performance of the ROC, it managed to reach 90% of accuracy in both detection and classification. Thirdly, a brief comparison was provided using a modified LSTM model provided in MATLAB. Under similar conditions, training and validation sets were analysed. That comparison was provided in order to show the limitation in terms of customization of the data when it comes to writing MATLAB scripts as compared to PYTHON.

6.2 Deliverables

The following section summarizes the work performed and the key findings in each chapter to accomplish the research aim and objectives.

In Chapter Two, there was a discussion of an extensive literature review. The chapter began with the introduction of distribution system in Power System. Then the study went on to depict the different constituents of a distribution system such as: sectionalizer, potential and normal transformer, recloser, fuses, lines, switch, voltage regulator and Supervisory Control and Data Acquisition (SCADA). Various among the most common converters used in the distributed generation were also discussed. This work had for main interest to develop a fault detection and classification model which integrated a PV system with a IEEE 13 Bus System. Distribution automation system were also introduced in order to get a broad overview of the evolution of the distribution system used in power system. Fuse, circuit breakers, and protective relays are examples of devices used in conventional distribution network protection that are used to identify and isolate problems in electrical distribution networks. Measurements of current and voltage are used by these systems to detect anomalous conditions like as overloads, ground faults, and short circuits. The main objectives are to protect the network, reduce service outages, and guarantee the safety of both people and equipment. However, the integration of distributed generation and renewable energy sources makes distribution networks more complex, making it difficult for traditional protection techniques to maintain selectivity, sensitivity, and reliability. This has prompted research into increasingly sophisticated defense strategies, like adaptive relaying and the use of clever algorithms to boost distribution network security. Then a brief presentation of various types of distribution architectures such as: ring, loop and radial distribution.

In Chapter Three, an essential component of guaranteeing the dependability and security of electrical power systems is fault investigation. It entails the detection and diagnosis of malfunctions that can impair a power system's regular operation and result in equipment damage, blackouts, and safety risks. Conventional procedures including differential protection, overcurrent protection, and distance protection are the mainstay of traditional fault detection systems. These techniques look for changes in the levels of current, voltage, or impedance by using preset thresholds and parameters. Conventional approaches are useful in many situations, but they frequently encounter difficulties when integrating distributed energy resources and complicated network architectures. This can lead to issues like lower sensitivity and ambiguity in fault location. Hybrid fault detection techniques have been developed to overcome the shortcomings of traditional methods. To improve the precision and dependability of

defect detection, these approaches include several traditional techniques or incorporate extra technology like signal processing or wavelet transforms. In systems with a significant concentration of renewable energy sources, where generation intermittency and variability can make fault detection more difficult, hybrid approaches are especially helpful.

The next step forward in fault analysis is represented by intelligent fault detection techniques. These methods use sophisticated computational techniques including machine learning algorithms, fuzzy logic, and ANNs to more accurately identify and diagnose problems. Over time, intelligent techniques can become more accurate by learning from past data and adjusting to changing network conditions.

They are particularly useful in contemporary power systems that use smart grid technology and have significant automation levels. In addition to being better at identifying faults, intelligent fault detection systems can also reveal the underlying causes of faults, allowing for more targeted and preventive maintenance approaches. In conclusion, hybrid and intelligent approaches have replaced traditional methods in fault analysis, each of which offers advances in reliability, adaptability, and accuracy of detection in ever-more complex power systems.

Chapter Four presents the Data analysis and system modelling. In order to analyse and model the data of a grid-connected photovoltaic system connected to an IEEE 13-Bus test feeder in MATLAB, a comprehensive simulation environment must be built in order to examine the relationships between distributed generation and the electrical grid. The IEEE 13-Bus system, which incorporates a variety of components such transformers, voltage regulators, loads, and line segments, is commonly used to test and validate power system analysis methodologies because of its modest complexity. A PV system's integration into this network entails modelling the PV array, inverter, and control systems before attaching them to a particular bus in the 13-Bus network. This configuration makes it possible to investigate the effects of solar power generation on the grid's voltage stability, power quality, and load flow.

Short circuit detection and classification become crucial jobs once the system is modelled in MATLAB. This is because faults in a grid-connected photovoltaic system can cause serious interruptions and raise safety issues. To address this, a recurrent neural network (RNN) model known as a LSTM can be developed and trained in Python using data obtained from the MATLAB simulation. RNNs are known for their efficacy in sequence prediction tasks. To understand the intricate temporal connections inside the system, the LSTM model is trained on time-series data reflecting a range of operating circumstances, including faulty and normal states.

Because the LSTM model can capture the temporal patterns of voltage and current during fault events, it is especially well-suited for identifying and classifying short circuits. The IEEE 13-Bus PV system's time-series data may be analysed by the LSTM, which can then be used to identify fault types (such as single-phase-to-ground and line-to-line) and even locate the problem with high accuracy. This combination of Python-based LSTM for sophisticated fault analysis and MATLAB-based simulation for data production creates a potent instrument for guaranteeing the dependability and security of grid-connected PV systems in contemporary power networks.

Chapter Five, the process of acquiring data from a MATLAB simulation to the Google Sheets API facilitates easy data access and storage, offering real-time availability and flexibility for subsequent Python analysis. To create pertinent data, such as voltage, current, or power output from a system like a grid-connected PV system, the procedure first runs a simulation in MATLAB. When the simulation data is prepared, it is stored as a CSV file after being transformed into an appropriate format, frequently as a table or array. Next, this data can be easily uploaded to a Google Sheets document via the Google Sheets API utilizing MATLAB's web options. To ensure secure access to the sheets, the API requires Google OAuth 2.0 authentication. The simulation data is imported and sorted into designated cells or ranges within Google Sheets by connecting MATLAB and Sheets.

With this configuration, Python users can access the data from any location at any time. The strong tools available in Python, including gspread and pandas, make it easier to retrieve data from Google Sheets for further processing, analysis, and visualization. This method is the best choice for situations when continuous access to the most recent simulation results is required because it guarantees that the data is not only safely kept but also easily accessible for dynamic applications.

Then it came to building a fault detection and classification system for short-circuit faults in a grid-connected photovoltaic system with an IEEE 13 Bus design is a highly effective use of an LSTM (Long Short-Term Memory) model. Using a software program such as MATLAB, first the IEEE 13 Bus was first tested with the implementations of the various short circuits faults at random bus, after that the grid-connected PV system is then simulated. This produces time-series data on voltage, current, and power under both normal operating conditions and different fault scenarios, such as short circuits. A PV system of 2.5MW is connected to the IEEE 13 Bus at Bus 633. This time-series data is then used to train the LSTM model, which is renowned for its capacity to capture dependencies over time in sequential data. Sequences of pre-processed and normalized data are sent into the LSTM network, and via this process, the model is trained to identify patterns that differentiate between fault kinds and normal operations.

The allocations of data during training was done such that 70% for training sets and 30% for validation. In order to reduce the classification error, the model iteratively modifies its internal parameters throughout the training phase. Performance is assessed using ROC curves and confusion matrices, as well as metrics like Mean Squared Error (MSE).

The range of MSE expected was less than 0.1 for all the different scenarios with an acceptable ROC of 60% for all the four short circuit cases. After being trained, the LSTM model can identify and categorize short-circuit faults in real-time, accurately predicting the kind of fault based on the data that is received. This approach enhances grid reliability by enabling timely fault detection and classification, leading to quicker responses and mitigating potential damage to the PV system and connected infrastructure. That was done intentionally, in order to justify and demonstrate the implementation of the SMOTE to show how Python is excellent for oversampling the data in order to improve the performance.

The model's performance was much enhanced by the use of SMOTE which was especially evident in the improvement of the ROC (Receiver Operating Characteristic) curve. The model was first trained on an unbalanced dataset that had an underrepresentation of certain error categories. With a ROC AUC (Area Under the Curve) score of roughly 0.6, this imbalance caused the model to be biased towards the majority class, which in turn led to a lower true positive rate and a less successful overall classification performance.

SMOTE was used to create synthetic samples for the minority class, balancing the dataset without just using copies of already-existing samples. The model was able to better understand the traits of each class thanks to this balanced dataset, which enhanced its capacity to recognize errors in all areas. The ROC AUC score rose to 0.93 after the model was retrained using the SMOTE-augmented data, demonstrating a notable improvement in the model's discriminatory capacity. Higher sensitivity (true positive rate) and specificity were shown in the revised ROC curve, which contributed to more accurate and dependable fault detection—a critical component in preserving the stability and safety of grid-connected PV systems.

There are a number of significant differences between the LSTM model created in Python and a comparable one created in MATLAB, especially in terms of flexibility and computational efficiency. The Python model is perfect for more complicated or experimental applications since it can be implemented with libraries such as Tensor Flow or PyTorch and typically provides more customization choices and access to state-of-the-art deep learning algorithms. Furthermore, Python's larger data science and machine learning ecosystem makes it simpler to integrate with a variety of tools

and APIs, like Google Sheets for data collection. Conversely, the MATLAB-integrated LSTM model gains from MATLAB's powerful numerical computation powers and Simulink integration, which is particularly useful for control system or embedded application engineers. The LSTM implementation in MATLAB might be easier to use for users accustomed to the platform, but it might not offer as much flexibility or support for the most recent developments in deep learning as Python does.

6.3 Future works

In the future it would be beneficial to analyse how that same work can be done in real time simulation with a software like Opal RT real time simulator. In order to see how the LSTM model would react at such incident. Besides the fact of handling sequential data it has several limitations such as noise, LSTM may overfit to noise, leading to poor generalization performance and struggles to handle outliers in the data. And also implementation of Computer vision might help since, the operator on site will be able to have images and real appreciation of where and how severe the short circuit fault was.

Then the study can extend the fault types to analyse and develop appropriate detection and classification model. It might be cyber-attacks, PV side faults…

6.4 Publications related to the work

- T.L.MAKOSSO, A. Almaktoof, K. Aboalez, "Integration of Artificial Neural Network in a IEEE 5 BUS System" published in Journal of Electrical Engineering and Applied Sciences (IJEEAS), 2024
- T.L. MAKOSSO, A. Almaktoof, K. Aboalez, REVIEW OF DIFFERENT TYPES OF NEURAL NETWORK ARCHITECTURES, published in Journal of Electrical Engineering and Applied Sciences (IJEEAS), 2024.
- T.L. MAKOSSO, A. Almaktoof, K. Aboalez, A systematic Literature Review on Faults detection techniques, under review in Mansoura Engineering Journal,2024.
- T.L. MAKOSSO, A. Almaktoof, K. Aboalez, PV Power Output Prediction Using Deep Learning, published in International Journal of Applied Engineering & Technology (IJAET), 2024.

- T.L. MAKOSSO, A. Almaktoof, K. Aboalez, Fault Detection and classification in Grid-Connected Photovoltaic Systems Using deep learning, under review in Mansoura Engineering Journal,2024.

T.L. MAKOSSO, A. Almaktoof, K. Aboalez, AI-Driven Fault Detection in Electrical Grids: Comparative analysis between Gradient Descent with Momentum and Adaptive Learning and resilient propagation algorithms, Accepted by International Conference on AI and Generative AI (FICAILY2025).

References

Abdali, Ali, Kazem Mazlumi, and Reza Noroozian. 2019. “High-Speed Fault Detection and Location in DC Microgrids Systems Using Multi-Criterion System and Neural Network.” *Applied Soft Computing Journal* 79:341–53. https://doi.org/10.1016/j.asoc.2019.03.051.

Abdullah, Beren Sardar, and Siddeeq Y. Ameen. 2022. “Investigation the Impact of Using Integrated PV System at Avro City in Duhok-Iraq.” *2022 International Conference on Decision Aid Sciences and Applications, DASA 2022*, 1334–38. https://doi.org/10.1109/DASA54658.2022.9765160.

Ahanch, Mojtaba, Roy McCann, and Alan Mantooth. 2021. “Hybrid AC Transmission System with Back-To-Back Converter Configuration and MTDC Operation Considering PV Energy Integration.” *2021 6th IEEE Workshop on the Electronic Grid, EGRID 2021*, 1–6. https://doi.org/10.1109/eGRID52793.2021.9662161.

Ahmad, Sanaullah, and Azzam Ul Asar. 2021. “Reliability Enhancement of Electric Distribution Network Using Optimal Placement of Distributed Generation.” *Sustainability (Switzerland)* 13 (20). https://doi.org/10.3390/su132011407.

Al-Hudaib, Rashed Hudaib, and Saad Alghuwainem. 2022. “Influence of Bulk PV Penetration on Power System Transient Stability.” *Proceeding - 6th International Conference on Information Technology, Information Systems and Electrical Engineering: Applying Data Sciences and Artificial Intelligence Technologies for Environmental Sustainability, ICITISEE 2022*, 717–22. https://doi.org/10.1109/ICITISEE57756.2022.10057781.

Alboali, Jaafar, Abdullah Alsuwaid, Mustafa Aljafar, and Muhammad Khalid. 2021. “Incentive-Based Assessment of Residential Solar PV Systems: A Case Study.” *Proceedings of 2021 31st Australasian Universities Power Engineering Conference, AUPEC 2021*, 1–5. https://doi.org/10.1109/AUPEC52110.2021.9597710.

Alharthi, Yahya Z., Nayeff Najjar, Ahmed Mechraoui, Mohammed Moalla, and Souaad Mohammed. 2021. “Design and Simulation of PV Grid-Connected System Considering Senstivity Analysis.” *Conference Record of the IEEE Photovoltaic Specialists Conference*, 2363–67. https://doi.org/10.1109/PVSC43889.2021.9518476.

Ali, Zulfiqar, Yacine Terriche, Le Quang Nhat Hoang, Syed Zagam Abbas, Mustafa Alrayah Hassan, Muhammad Sadiq, Chun Lien Su, and Josep M. Guerrero. 2021. “Fault Management in DC Microgrids: A Review of Challenges, Countermeasures, and Future Research Trends.” *IEEE Access* 9:128032–54.

https://doi.org/10.1109/ACCESS.2021.3112383.

Angelin, M. J., and R. Geetha. 2023. “Increasing the Power in Photovoltaic Systems Using a Floating PV System Compared with a Rooftop PV System by Limiting the Temperature Loss.” *Proceedings of International Conference on Contemporary Computing and Informatics, IC3I 2023* 6 (Deambi 2016): 1506–10. https://doi.org/10.1109/IC3I59117.2023.10398137.

Ashoornezhad, Ali, Qasem Asadi, Reza Saberi, and Hamid Falaghi. 2023. “Considering the Impact of PV-Battery Systems on Long-Term Distribution Network Planning.” *2023 27th International Electrical Power Distribution Conference, EPDC 2023*, 161–66. https://doi.org/10.1109/EPDC59105.2023.10218884.

Asmar, Joseph Al, Mohammad Shraif, Hicham El Khoury, and Jack Monhem. 2022. “Economic Performance of a 250kW Solar PV System: Application to Lebanon.” *2022 4th IEEE Middle East and North Africa COMMunications Conference, MENACOMM 2022*, 100–105. https://doi.org/10.1109/MENACOMM57252.2022.9998192.

Ayiad, Motaz M., Helder Leite, and Hugo Martins. 2021. “State Estimation for Hybrid VSC Based HVDC/AC: Unified Bad Data Detection Integrated with Gaussian Mixture Model.” *IEEE Access* 9:91730–40. https://doi.org/10.1109/ACCESS.2021.3092308.

Aziz, Farkhanda, Azhar Ul Haq, Shahzor Ahmad, Yousef Mahmoud, Marium Jalal, and Usman Ali. 2020. “A Novel Convolutional Neural Network-Based Approach for Fault Classification in Photovoltaic Arrays.” *IEEE Access* 8:41889–904. https://doi.org/10.1109/ACCESS.2020.2977116.

Babu, Allen Solly P., and P. R. Subadhra. 2022. “Power Oscillation Damping By Utilizing PV-STATCOM.” *2022 IEEE Delhi Section Conference, DELCON 2022*, 1–6. https://doi.org/10.1109/DELCON54057.2022.9753472.

Batmani, Yazdan, Mehran Takhtabnus, and Rahmatollah Mirzaei. 2022. “DC Microgrid Fault-Tolerant Control Using State-Dependent Riccati Equation Techniques.” *Optimal Control Applications and Methods* 43 (1): 123–37. https://doi.org/10.1002/oca.2776.

Biju, Hannah, K. P. Jaheer Mukthar, Norma Ramírez-Asís, Jorge Castillo-Picon, Guillermo Pelaez-Diaz, and Liset Silva-Gonzales. 2024. “A Systematic Literature Review on Mercantilism.” *Studies in Systems, Decision and Control* 487:739–49. https://doi.org/10.1007/978-3-031-35828-9_61.

Biswal, Ananya Pritilagna, and Krishna Roy. 2022. “Modelling and Performance Analysis of a Hybrid Wind and PV Generation System.” *2nd Odisha International Conference on*

Electrical Power Engineering, Communication and Computing Technology, ODICON 2022, 1–6. https://doi.org/10.1109/ODICON54453.2022.10009994.

Cai, Jun, Fang Liu, Zhen Fan, Denis Sidorov, and Mukesh Kumar Pathak. 2022. "EID-Based Robust Controller Design of PV Grid-Connected System with Parameter Uncertainty." *Chinese Control Conference, CCC* 2022-July:5289–93. https://doi.org/10.23919/CCC55666.2022.9902129.

Cao, Haiou, Chenbin Zhou, Yihua Meng, Jiaoxiao Shen, and Xiayin Xie. 2024. "Advancement in Transformer Fault Diagnosis Technology." *Frontiers in Energy Research* 12 (July): 1–16. https://doi.org/10.3389/fenrg.2024.1437614.

Chatterjee, Soumesh, and Biman Kumar Saha Roy. 2020. "Fast Identification of Symmetrical or Asymmetrical Faults during Power Swings with Dual Use Line Relays." *CSEE Journal of Power and Energy Systems* 6 (1): 184–92. https://doi.org/10.17775/CSEEJPES.2019.01440.

Chen, Han Yun, and Ching Hung Lee. 2020. "Vibration Signals Analysis by Explainable Artificial Intelligence (XAI) Approach: Application on Bearing Faults Diagnosis." *IEEE Access* 8:134246–56. https://doi.org/10.1109/ACCESS.2020.3006491.

Chen, Xia, Mengxuan Shi, Jianyu Zhou, Yin Chen, Wenping Zuo, Jinyu Wen, and Haibo He. 2020. "Distributed Cooperative Control of Multiple Hybrid Energy Storage Systems in a DC Microgrid Using Consensus Protocol." *IEEE Transactions on Industrial Electronics* 67 (3): 1968–79. https://doi.org/10.1109/TIE.2019.2898606.

Cortés-Caicedo, Brandon, Federico Molina-Martin, Luis Fernando Grisales-Noreña, Oscar Danilo Montoya, and Jesus C. Hernández. 2022. "Optimal Design of PV Systems in Electrical Distribution Networks by Minimizing the Annual Equivalent Operative Costs through the Discrete-Continuous Vortex Search Algorithm." *Sensors* 22 (3). https://doi.org/10.3390/s22030851.

Dai, Zhihui, Ningning Liu, Cheng Zhang, and Siqi Yan. 2019. "State-Space Analysis Based Pole-to-Ground Line Fault Isolation Strategy for LCC-HVDC Systems." *IET Generation, Transmission and Distribution* 13 (10): 1933–41. https://doi.org/10.1049/iet-gtd.2018.6806.

De-Jesús-Grullón, Ramón E., Rafael Omar Batista Jorge, Abraham Espinal Serrata, Justin Eladio Bueno Díaz, Juan José Pichardo Estévez, and Nestor Francisco Guerrero-Rodríguez. 2024. "Modeling and Simulation of Distribution Networks with High Renewable Penetration in Open-Source Software: QGIS and OpenDSS." *Energies* 17

(12). https://doi.org/10.3390/en17122925.

Deori, Pooja, Ritupan Bhuyan, and Anish Ahmad. 2022. “Improved Control Scheme for Grid Connected Solar PV System with Fuzzy MPPT.” *10th IEEE International Conference on Power Electronics, Drives and Energy Systems, PEDES 2022*, 1–6. https://doi.org/10.1109/PEDES56012.2022.10080434.

Deshmukh, Shruti, Shruti Limkar, Rushikesh Nagthane, V. N. Pande, and Arti V. Tare. 2023. “Design of Grid-Connected Solar PV System Integrated with Battery Energy Storage System.” *2023 3rd Asian Conference on Innovation in Technology, ASIANCON 2023*, 1–6. https://doi.org/10.1109/ASIANCON58793.2023.10269854.

Dommaraju, Jyothsna, and T. Sreejith Kailas. 2022. “Harmonic Reduction through Neutral Point Clamped Grid-Connected PV System Using IcosΦ Algorithm.” *2022 1st International Conference on Sustainable Technology for Power and Energy Systems, STPES 2022*, no. Mmc, 1–6. https://doi.org/10.1109/STPES54845.2022.10006515.

Elsanabary, Ahmed, Saad Mekhilef, Mehdi Seyedmahmoudian, and Alex Stojcevski. 2021. “A Novel Circuit Configuration for the Integration of Modular Multilevel Converter with Large-Scale Grid-Connected PV Systems.” *Proceedings of the Energy Conversion Congress and Exposition - Asia, ECCE Asia 2021*, 232–37. https://doi.org/10.1109/ECCE-Asia49820.2021.9478992.

Etukudor, Christie, Valentin Robu, Wolf Gerrit Fruh, David Flynn, and Hope Orovwode. 2021. “Yield Assessment of Off-Grid PV Systems in Nigeria.” *2021 IEEE PES/IAS PowerAfrica, PowerAfrica 2021*, 1–5. https://doi.org/10.1109/PowerAfrica52236.2021.9543317.

Forouzesh, Alireza, Mohammad S. Golsorkhi, Mehdi Savaghebi, and Mehdi Baharizadeh. 2021. “Support Vector Machine Based Fault Location Identification in Microgrids Using Interharmonic Injection.” *Energies* 14 (8): 1–14. https://doi.org/10.3390/en14082317.

Ge, Yuanyu, Jun Xie, Jianan Duan, Shanxi Xing, Mingtao Liu, and Qiuyan Zhang. 2022. “Intelligent Dispatch of Wind-Thermal Hybrid System Based on Deep Reinforcement Learning Considering Flexible Ramping Capacity Provided by Wind Power.” *2022 5th International Conference on Power and Energy Applications, ICPEA 2022*, 698–703. https://doi.org/10.1109/ICPEA56363.2022.10052417.

Haddadi, Aboutaleb, Ilhan Kocar, Jean Mahseredjian, Ulas Karaagac, and Evangelos Farantatos. 2020. “Performance of Phase Comparison Line Protection under Inverter-Based Resources and Impact of the German Grid Code.” *IEEE Power and Energy*

Society General Meeting 2020-Augus (June). https://doi.org/10.1109/PESGM41954.2020.9282027.

Han, Yang, Mengling Yang, Ping Yang, Lin Xu, Xu Fang, Ke Zhang, and Frede Blaabjerg. 2019. “Reduced-Order Model for Dynamic Stability Analysis of Single-Phase Islanded Microgrid with BPF-Based Droop Control Scheme.” *IEEE Access* 7:157859–72. https://doi.org/10.1109/ACCESS.2019.2950069.

Harrou, Fouzi, Bilal Taghezouit, and Ying Sun. 2019. “Improved KnN-Based Monitoring Schemes for Detecting Faults in PV Systems.” *IEEE Journal of Photovoltaics* 9 (3): 811–21. https://doi.org/10.1109/JPHOTOV.2019.2896652.

Hasan, Mahmudul, Zannatul Mifta, Nafisa Atia Salsabil, Sumaiya Janefar Papiya, Mehnaz Hossain, Paromita Roy, Nahid Ur Rahman Chowdhury, and Omar Farrok. 2023. “A Critical Review on Control Mechanisms, Supporting Measures, and Monitoring Systems of Microgrids Considering Large Scale Integration of Renewable Energy Sources.” *Energy Reports* 10 (November): 4582–4603. https://doi.org/10.1016/j.egyr.2023.11.025.

Hategekimana, Pascal, Adria Junyent Ferre, Joan Marc Rodriguez Bernuz, and Etienne Ntagwirumugara. 2022. “Fault Detecting and Isolating Schemes in a Low-Voltage DC Microgrid Network from a Remote Village.” *Energies* 15 (12). https://doi.org/10.3390/en15124460.

Hategekimana, Pascal, Adrià Junyent-Ferré, Etienne Ntagwirumugara, and Joan Marc Rodriguez Bernuz. 2024. “Improved Methods for Controlling Interconnected DC Microgrids in Rural Villages.” *AIMS Energy* 12 (1): 214–34. https://doi.org/10.3934/energy.2024010.

Hu, Binhui, and Jianhua Wang. 2022. “Research on Flexible Power Limiting Strategy of PV Power Generation System Based on Secant Method.” *Proceedings - 2022 4th International Conference on Electrical Engineering and Control Technologies, CEECT 2022*, 1137–42. https://doi.org/10.1109/CEECT55960.2022.10030477.

Hung, Truong Ngoc. 2022. “Methods for Fault Location in High Voltage Power Transmission Lines: A Comparative Analysis.” *International Journal of Renewable Energy Development* 11 (4): 1134–41. https://doi.org/10.14710/IJRED.2022.46501.

Jayamaha, D. K.J.S., N. W.A. Lidula, and Athula D. Rajapakse. 2019. “Wavelet-Multi Resolution Analysis Based ANN Architecture for Fault Detection and Localization in DC Microgrids.” *IEEE Access* 7:145371–84. https://doi.org/10.1109/ACCESS.2019.2945397.

Kalimuthukumar, S., K. Rajesh, B. Kannapiran, G. Manikandan, and R. Selvameena. 2021. "A SEPIC Converter with Adaptive Sliding Mode Control for Grid-Connected Solar PV Systems." *Proceedings - 2021 3rd International Conference on Advances in Computing, Communication Control and Networking, ICAC3N 2021*, 1023–26. https://doi.org/10.1109/ICAC3N53548.2021.9725608.

Kamga Sagoun, W. E., I. Chihi, L. Sidhom, M. Trabelsi, and G. E. Georghiou. 2021. "Performance Evaluation of Various Faults Classifiers for Grid-Connected PV Systems." *International Conference on Electrical, Computer, and Energy Technologies, ICECET 2021*, no. December, 1–5. https://doi.org/10.1109/ICECET52533.2021.9698671.

Kavitha, M., D. Godwin Immanuel, C. R.Edwin Selva Rex, V. Meenakshi, M. Pushpavalli, Supriya Singari, and Vinoba Baskaran. 2021. "Energy Forecasting of Grid Connected Roof Mounted Solar PV Using PVSOL." *Proceedings of the 2021 IEEE International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems, ICSES 2021*, 1–6. https://doi.org/10.1109/ICSES52305.2021.9633888.

Kharusi, Khalfan Al, Abdelsalam El Haffar, and Mostefa Mesbah. 2022. "Fault Detection and Classification in Transmission Lines Connected to Inverter-Based Generators Using Machine Learning." *Energies* 15 (15). https://doi.org/10.3390/en15155475.

Khotsirwong, Nonthawat, Terapong Boonraksa, Promphak Boonraksa, Thipwan Fangsuwannarak, Asada Boonsrirat, and Boonruang Marungsri. 2022. "Weekly Power Generation Forecasting Using Deep Learning Techniques: Case Study of a 1.5 MWp Floating PV Power Plant." *2022 International Conference on Power, Energy and Innovations, ICPEI 2022*, no. Icpei, 1–4. https://doi.org/10.1109/ICPEI55293.2022.9986628.

Kumar, Jaynendra, Anshul Agarwal, and Nitin Singh. 2020. "Design, Operation and Control of a Vast DC Microgrid for Integration of Renewable Energy Sources." *Renewable Energy Focus* 34 (September 2020): 17–36. https://doi.org/10.1016/j.ref.2020.05.001.

Kumar, Sachin, Kumari Sarita, Akanksha Singh S. Vardhan, Rajvikram Madurai Elavarasan, R. K. Saket, and Narottam Das. 2020. "Reliability Assessment of Wind-Solar Pv Integrated Distribution System Using Electrical Loss Minimization Technique." *Energies* 13 (21). https://doi.org/10.3390/en13215631.

Lacerda, Vinícius Albernaz, Renato M. Monaro, David Campos-Gaona, Denis V. Coury, and Olimpo Anaya-Lara. 2020. "Distance Protection Algorithm for Multiterminal HVDC Systems Using the Hilbert-Huang Transform." *IET Generation, Transmission and Distribution* 14 (15): 3022–32. https://doi.org/10.1049/iet-gtd.2019.1551.

Ladjici, Ahmed Amine, and Ahmed Tiguercha. 2023. “Deep Reinforcement Learning for Microgrid Power Management System.” *9th 2023 International Conference on Control, Decision and Information Technologies, CoDIT 2023*, 1144–49. https://doi.org/10.1109/CoDIT58514.2023.10284334.

Lavanya, S., S. Prabakaran, and N. Ashok Kumar. 2022. “A Deep Learning Technique for Detecting High Impedance Faults in Medium Voltage Distribution Networks.” *Engineering, Technology and Applied Science Research* 12 (6): 9477–82. https://doi.org/10.48084/etasr.5288.

Le, Thi Thu Huong, Yongsu Kim, and Howon Kim. 2019. “Network Intrusion Detection Based on Novel Feature Selection Model and Various Recurrent Neural Networks.” *Applied Sciences (Switzerland)* 9 (7). https://doi.org/10.3390/app9071392.

Li, Botong, Dingchuan Zheng, Bin Li, Xinru Jiao, Qiteng Hong, and Liang Ji. 2023. “Analysis of Low Voltage Ride-through Capability and Optimal Control Strategy of Doubly-Fed Wind Farms under Symmetrical Fault.” *Protection and Control of Modern Power Systems* 8 (1). https://doi.org/10.1186/s41601-023-00310-0.

Li, Cheng, and Bo Wang. 2023. “A Knowledge Graph Method towards Power System Fault Diagnosis and Classification.” *Electronics (Switzerland)* 12 (23): 1–14. https://doi.org/10.3390/electronics12234808.

Li, Guanbin, Qing Chen, and Jianlei Zhang. 2020. “Novel Faulted Section Location Method for Distribution Network Based on Status Information of Fault Indicating Equipment.” *Applied Sciences (Switzerland)* 10 (17). https://doi.org/10.3390/app10175910.

Li, Huihui, Linfeng Gou, Huacong Li, and Zhidan Liu. 2023. “Physics-Guided Neural Network Model for Aeroengine Control System Sensor Fault Diagnosis under Dynamic Conditions.” *Aerospace* 10 (7). https://doi.org/10.3390/aerospace10070644.

Li, Yujun, Jiapeng Li, Liansong Xiong, Xian Zhang, and Zhao Xu. 2020. “DC Fault Detection in Meshed MTdc Systems Based on Transient Average Value of Current.” *IEEE Transactions on Industrial Electronics* 67 (3): 1932–43. https://doi.org/10.1109/TIE.2019.2907499.

Li, Zhengxi, Na An, Kai Wang, Junxiong Ma, Jin Gao, Tingjian Liu, and Haoyu Liu. 2023. “Emergency Control for Sending-End Power System with Renewable Energy under HVDC Blocking Based on Deep Reinforcement Learning.” *Proceedings - 2023 5th International Conference on Electrical Engineering and Control Technologies, CEECT 2023*, 102–6. https://doi.org/10.1109/CEECT59667.2023.10420701.

Lin, Wei Chen, Wei Tzer Huang, Kai Chao Yao, Hong Ting Chen, and Chun Chiang Ma. 2021. “Fault Location and Restoration of Microgrids via Particle Swarm Optimization.” *Applied Sciences (Switzerland)* 11 (15). https://doi.org/10.3390/app11157036.

Liu, Siyuan, Ziao Yuan, Jinchao Chen, Yifan Chen, Mengze Yu, Zhiyuan Liu, and Yingsan Geng. 2024. “A Cost-Effective Current-Limiting Hybrid DC Circuit Breaker Based on Hybrid Semiconductors.” *Electronics (Switzerland)* 13 (10). https://doi.org/10.3390/electronics13101948.

Liu, Xiaochen, Xiaohua Liu, Yi Jiang, Tao Zhang, and Bin Hao. 2023. “Photovoltaics and Energy Storage Integrated Flexible Direct Current Distribution Systems of Buildings: Definition, Technology Review, and Application.” *CSEE Journal of Power and Energy Systems* 9 (3): 829–45. https://doi.org/10.17775/CSEEJPES.2022.04850.

Lozanov, Yavor, Svetlana Tzvetkova, and Angel Petleshkov. 2023. “Simulation Model For Evaluation Of Power Quality Indicators In Industrial Power Supply Systems With Grid-Tied PV System.” *2023 18th Conference on Electrical Machines, Drives and Power Systems, ELMA 2023 - Proceedings*, 1–4. https://doi.org/10.1109/ELMA58392.2023.10202315.

Madhubabu, Arava, and G. V.Siva Krishna Rao. 2021. “Review of Technology Involved in Floating Solar PV System.” *Proceedings of the 6th International Conference on Communication and Electronics Systems, ICCES 2021*, 376–78. https://doi.org/10.1109/ICCES51350.2021.9489048.

Maniraju, Usha, and Thangamuthu Senthil Kumaran. 2024. “An Auto-Encoder Bio Medical Signal Transmission through Custom Convolutional Neural Network.” *IAES International Journal of Artificial Intelligence* 13 (2): 1312–25. https://doi.org/10.11591/ijai.v13.i2.pp1312-1325.

Manmai, Sontaya, Sillawat Romphochai, Natin Janjamraj, Surin Ngaemngam, and Krischonme Bhumkittipich. 2021. “Load Frequency Control of Large Scale PV-BESS Generation Installation Using Power System Stabilizer.” *ICEMS 2021 - 2021 24th International Conference on Electrical Machines and Systems*, 2378–81. https://doi.org/10.23919/ICEMS52562.2021.9634254.

Mazibuko, Ntombenhle, Kayode T. Akindeji, and Katleho Moloi. 2024. “A Review on the Impact of Transmission Line Compensation and RES Integration on Protection Schemes.” *Energies* 17 (14): 3433. https://doi.org/10.3390/en17143433.

Mohammadi, Ebrahim, and Gerry Moschopoulos. 2021. “Optimization of PV Array-to-Inverter

Power Ratio in Grid-Connected Systems to Maximize System Profit." *Conference Proceedings - IEEE Applied Power Electronics Conference and Exposition - APEC*, 2084–89. https://doi.org/10.1109/APEC42165.2021.9487276.

Mohanty, Pratap Ranjan, Praneeth K. Guptha, N. Sowmya, S. S.Jaheer Hussain, and G. Siva Nanda Kishore. 2023. "Phase Locked Loop (PLL) MPPT Based Single Stage Grid-Connected PV System." *2023 1st International Conference on Circuits, Power, and Intelligent Systems, CCPIS 2023*, 1–5. https://doi.org/10.1109/CCPIS59145.2023.10291632.

Muhammad, Fazal, Haroon Rasheed, Daniel Westerman Spier, Eduardo Prieto-Araujo, and Oriol Gomis Bellmunt. 2022. "Control Design and Fault Handling Performance of MMC for MMC-Based DC Distribution System." *IEEE Access* 10 (December): 126695–711. https://doi.org/10.1109/ACCESS.2022.3224773.

Muralidhar, K., and N. Rajasekar. 2021. "Soiling and Shading Analysis of Hybrid PV System for Rural Application Using PVsyst Software." *3rd IEEE International Virtual Conference on Innovations in Power and Advanced Computing Technologies, i-PACT 2021*, 1–7. https://doi.org/10.1109/i-PACT52855.2021.9697016.

Naidu, O. D., Sinisa Zubic, A. V.S.S.R. Sai, A. N. Praveen, Patrick Cost, and Hakan Eriksson. 2022. "Economical Setting-Free Double-Ended Fault Locator for Transmission Lines: Experiences From Recent Pilot Installations." *IEEE Access* 10 (September): 96805–20. https://doi.org/10.1109/ACCESS.2022.3205020.

Nair, Unnikrishnan Raveendran, and Ramon Costa-Castello. 2020. "A Model Predictive Control-Based Energy Management Scheme for Hybrid Storage System in Islanded Microgrids." *IEEE Access* 8:97809–22. https://doi.org/10.1109/ACCESS.2020.2996434.

Nallamothu, Bala Krishna. 2024. "Design and Analysis of Grid Connected Solar PV System Using PVsyst Software." *2024 IEEE Students Conference on Engineering and Systems (SCES)*, 1–6. https://doi.org/10.1109/SCES61914.2024.10652298.

Nandini, K. K., N. S. Jayalakshmi, and Vinay Kumar Jadoun. 2022. "Energy Management System for PV Integrated Utility Grid with Electric Vehicle as Storage System." *2022 2nd International Conference on Power Electronics and IoT Applications in Renewable Energy and Its Control, PARC 2022*, 1–6. https://doi.org/10.1109/PARC52418.2022.9726655.

Nougain, Vaibhav, Sukumar Mishra, Soumya Shubhra Nag, and Aleksandra Lekic. 2023. "Fault Location Algorithm for Multi-Terminal Radial Medium Voltage DC Microgrid."

IEEE Transactions on Power Delivery 38 (6): 4476–88. https://doi.org/10.1109/TPWRD.2023.3318689.

Petitjean, Gérald. 2004. “Introduction Aux Réseaux de Neuronnes,” 1–17.

Polytechnique, Ecole Nationale. 2013. “D ’ Ingénieur d ’ Etat En Automatique Détection et Diagnostic Des Défauts Dans Les Systèmes Photovoltaïques Par Réseaux de Neurones Promotion 2013.”

Rabbi, Fazley, Sakera Rashid, Chitra Roy, Md Saidur Rahman, Manoj Sarker, and K. M. Rumman. 2023. “Performance Analysis of Various Sun-Tracking Systems in Terms of PV-Based Power Generation in the Hilly Region of Bangladesh.” *2023 International Conference on Information and Communication Technology for Sustainable Development, ICICT4SD 2023 - Proceedings*, 189–93. https://doi.org/10.1109/ICICT4SD59951.2023.10303394.

Radhiansyah, Tri Desmana Rachmilda, and Deny Hamdani. 2021. “Performance Analysis of Offshore Floating PV Systems in Isolated Area.” *2021 3rd International Conference on High Voltage Engineering and Power Systems, ICHVEPS 2021*, 651–55. https://doi.org/10.1109/ICHVEPS53178.2021.9600926.

Rahman, Md Wazedur, Karthikeyan Velmurugan, Md Sultan Mahmud, Abdullah Al Mamun, and Prasanth Ravindran. 2021. “Modeling of a Stand-Alone Wind-PV Hybrid Generation System Using (MATLAB/SIMULINK).” *Proceedings - IEEE 2021 International Conference on Computing, Communication, and Intelligent Systems, ICCCIS 2021*, 1000–1006. https://doi.org/10.1109/ICCCIS51004.2021.9397194.

Raj, Nithin, Saly George, and G. Jagadanand. 2015. “Open Transistor Fault Detection in Asymmetric Multilevel Inverter.” *2015 IEEE International Conference on Signal Processing, Informatics, Communication and Energy Systems, SPICES 2015*, 1–5. https://doi.org/10.1109/SPICES.2015.7091480.

Rajendran Pillai, Vipin Raj, Rohit Rajasekharan Nair Valsala, Veena Raj, Muhammed Iskandar Petra, Satheesh Kumar Krishnan Nair, and Sathyajith Mathew. 2023. “Exploring the Potential of Microgrids in the Effective Utilisation of Renewable Energy: A Comprehensive Analysis of Evolving Themes and Future Priorities Using Main Path Analysis.” *Designs* 7 (3). https://doi.org/10.3390/designs7030058.

Rezaei, Omid, Omid Mirzapour, Mohammad Panahazari, and Hassan Gholami. 2022. “Hybrid AC/DC Provisional Microgrid Planning Model Considering Converter Aging.” *Electricity* 3 (2): 236–50. https://doi.org/10.3390/electricity3020014.

Rezapour, Hamed, Sadegh Jamali, and Pierluigi Siano. 2024. “Wide-Area Protection System for Radial Smart Distribution Networks.” *Applied Sciences (Switzerland)* 14 (11). https://doi.org/10.3390/app14114862.

Rituraj, Rituraj, and Peter Kadar. 2021. “Performance Analysis of 50 KWp Rooftop Grid-Connected PV System.” *SISY 2021 - IEEE 19th International Symposium on Intelligent Systems and Informatics, Proceedings*, 177–82. https://doi.org/10.1109/SISY52375.2021.9582513.

Rosa De Jesus, Dan A., Paras Mandal, Yuan Kang Wu, and Tomonobu Senjyu. 2020. “Deep Learning Ensemble Based New Approach for Very Short-Term Wind Power Forecasting.” *IEEE Power and Energy Society General Meeting* 2020-Augus. https://doi.org/10.1109/PESGM41954.2020.9281473.

Sadeghkhani, Iman, Mohamad Esmail Hamedani Golshan, Ali Mehrizi-Sani, Josep M. Guerrero, and Abbas Ketabi. 2018. “Transient Monitoring Function-Based Fault Detection for Inverter-Interfaced Microgrids.” *IEEE Transactions on Smart Grid* 9 (3): 2097–2107. https://doi.org/10.1109/TSG.2016.2606519.

Safadi, Shorooq Ala’a-din, Zaid Sadiq Saifi, Jaffar Jallad, Samer Alsadi, Tareq Foqha, Ali Elrashidi, and Mohammad Kanan. 2024. “Integrated Electrical Installation and PV System for Qabatya School: Design and Simulation.” *2023 2nd International Engineering Conference on Electrical, Energy, and Artificial Intelligence (EICEEAI)*, no. 1, 1–4. https://doi.org/10.1109/eiceeai60672.2023.10590321.

Salehimehr, Sirus, Seyed Mahdi Miraftabzadeh, and Morris Brenna. 2024. “A Novel Machine Learning-Based Approach for Fault Detection and Location in Low-Voltage DC Microgrids.” *Sustainability (Switzerland)* 16 (7). https://doi.org/10.3390/su16072821.

Sarangi, Swetalina, Binod Kumar Sahu, and Pravat Kumar Rout. 2021. “Review of Distributed Generator Integrated AC Microgrid Protection: Issues, Strategies, and Future Trends.” *International Journal of Energy Research* 45 (10): 14117–44. https://doi.org/10.1002/er.6689.

Senthilkumar, S., K. Balachander, and V. M. Mohamed Mansoor. 2024. “A Hybrid Technique for Impact of Hybrid Renewable Energy Systems on Reliability of Distribution Power System.” *Energy* 306 (February 2023): 132383. https://doi.org/10.1016/j.energy.2024.132383.

Shravani, Chapala, Narasimham R L, and Tulasi Ram Das G. 2024. “UPQC-Based Power Quality Improvement in Grid-Linked PV, Battery & Wind Systems.” *E3S Web of*

Conferences 547:01007. https://doi.org/10.1051/e3sconf/202454701007.

Sistani, Alireza, Seyed Amir Hosseini, Vahideh Sadat Sadeghi, and Behrooz Taheri. 2023. "Fault Detection in a Single-Bus DC Microgrid Connected to EV/PV Systems and Hybrid Energy Storage Using the DMD-IF Method." *Sustainability (Switzerland)* 15 (23). https://doi.org/10.3390/su152316269.

Tan, Pei Seng, Tong Boon Tang, and Eric Tatt Wei Ho. 2022. "Explainable Artificial Intelligence Applied to Deep Reinforcement Learning Controllers for Photovoltaic Maximum Power Point Tracking." *2022 International Conference on Future Trends in Smart Communities, ICFTSC 2022*, 82–86. https://doi.org/10.1109/ICFTSC57269.2022.10040061.

Tang, Yiying, Ran Lv, Zhefei Xu, Cong Wang, Mingxing Guo, Ciwei Gao, and Shengbing Guan. 2023. "A Virtual Power Plant Economic Scheduling Strategy Based on Deep Reinforcement Learning." *2023 5th International Conference on Power and Energy Technology, ICPET 2023*, 1365–70. https://doi.org/10.1109/ICPET59380.2023.10367600.

Tariq, Rizwan, Ibrahim Alhamrouni, Ateeq Ur Rehman, Elsayed Tag Eldin, Muhammad Shafiq, Nivin A. Ghamry, and Habib Hamam. 2022. "An Optimized Solution for Fault Detection and Location in Underground Cables Based on Traveling Waves." *Energies* 15 (17). https://doi.org/10.3390/en15176468.

Tejesh, B., B. S.Ganesh Charan, Ch Murali Krishna, and P. V. Manitha. 2022. "Development of Hybrid PV Wind System for EVs with Battery Management System." *2022 2nd International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies, ICAECT 2022*, 1–4. https://doi.org/10.1109/ICAECT54875.2022.9808062.

Thomas, Polly, Ajin George Reji, Ashams Mathew, and D. Aswin. 2020. "Stand Alone Distribution Feeder Inter Area Fault Location Identification System for Indian Utility." *2020 IEEE 5th International Conference on Computing Communication and Automation, ICCCA 2020*, no. January, 258–62. https://doi.org/10.1109/ICCCA49541.2020.9250916.

Thoufeer, K. K., and Manas Kumar Jena. 2023. "A Deep Learning Scheme for Efficient Power System Waveform De-Noising and Reconstruction." *2023 10th IEEE International Conference on Power Systems, ICPS 2023*, no. December, 1–6. https://doi.org/10.1109/ICPS60393.2023.10428746.

Tran, Quang Thinh, and Sy Dzung Nguyen. 2022. “Bearing Fault Diagnosis Based on Measured Data Online Processing, Domain Fusion, and ANFIS.” *Computation* 10 (9). https://doi.org/10.3390/computation10090157.

Truong, Dinh Nhon, Van Phuong Ta, Mi Sa Nguyen Thi, Van Thuyen Ngo, Huu Vinh Nguyen, and Xuan Hoa Pham Thi. 2023. “Improving Transient Stability of a PV-Battery Based Microgrid System.” *Proceedings of 2023 International Conference on System Science and Engineering, ICSSE 2023*, 456–61. https://doi.org/10.1109/ICSSE58758.2023.10227148.

Truong, Dinh Nhon, Mi Sa Nguyen Thi, Van Thuyen Ngo, Van Phuong Ta, Van Tri Bui, and Huu Vinh Nguyen. 2021. “Applying Dynamic Voltage Restorer to Mitigate the Voltage Sag in the Grid Connected Solar PV System.” *Proceedings of 2021 International Conference on System Science and Engineering, ICSSE 2021*, 176–80. https://doi.org/10.1109/ICSSE52999.2021.9538424.

Uddin Khan, Md Asif, Qiteng Hong, Adam Dysko, and Campbell Booth. 2019. “Review and Evaluation of Protection Issues and Solutions for Future Distribution Networks.” *2019 54th International Universities Power Engineering Conference, UPEC 2019 - Proceedings*. https://doi.org/10.1109/UPEC.2019.8893528.

Vadivel, Srinivasan, C. S. Boopthi, Sridhar Ramasamy, Mominul Ahsan, Julfikar Haider, and Eduardo M.G. Rodrigues. 2021. “Performance Enhancement of a Partially Shaded Photovoltaic Array by Optimal Reconfiguration and Current Injection Schemes.” *Energies* 14 (19): 1–21. https://doi.org/10.3390/en14196332.

Vanam, Satyanarayana, and Sreedhar Gurijala. 2022. “Hybrid Energy Management System & Power Compensation Of PV & Wind Integrated Grid System With Fuzzy Based D-STATCOM.” *Proceedings of the 2022 International Conference on Electronic Systems and Intelligent Computing, ICESIC 2022*, 206–12. https://doi.org/10.1109/ICESIC53714.2022.9783520.

Wang, Guishuo, Xiaoli Wang, and Xiang Gao. 2021. “Improved Seamless Switching Control Strategy for AC/DC Hybrid Microgrid.” *IEEE Access* 9:55790–801. https://doi.org/10.1109/ACCESS.2021.3071821.

Wang, Haomiao, Hongliang Ma, Huan He, Zhiruo Meng, Xuesong Liu, and Siyuan Cai. 2020. “Application Resolution of Deep Learning in Electric Power System.” *Proceedings - 2020 7th International Forum on Electrical Engineering and Automation, IFEEA 2020*, 348–52. https://doi.org/10.1109/IFEEA51475.2020.00079.

Wang, Jiahao, Peng Li, Miao Wen, Lei Jiang, Youpeng Pan, and Linyuan Li. 2022. “Multi-Time Scale Coordinative Strategy of Reactive Power and Voltage Control for Frequent Power Flow Fluctuation in New Power System.” *EI2 2022 - 6th IEEE Conference on Energy Internet and Energy System Integration*, 3327–32. https://doi.org/10.1109/EI256261.2022.10117355.

Wang, Yongshuai, Zengqiang Chen, Mingwei Sun, and Qinglin Sun. 2021. “Load Frequency Active Disturbance Rejection Control for an Interconnected Power System via Deep Reinforcement Learning.” *Proceedings of 2021 IEEE 10th Data Driven Control and Learning Systems Conference, DDCLS 2021*, 94–99. https://doi.org/10.1109/DDCLS52934.2021.9455664.

Weng, Geping, Jiaorong Ren, Jianli Yang, Yichuan Lou, Han Jiang, and Bingyu Weng. 2023. “Key Technologies for User Side Load Storage Interactive Regulation of New Power Systems Based on Deep Learning Algorithms.” *Proceedings - 2023 2nd International Conference on Data Analytics, Computing and Artificial Intelligence, ICDACAI 2023*, 330–36. https://doi.org/10.1109/ICDACAI59742.2023.00069.

Wu, Tiezhou, Fanchao Ye, Yuehong Su, Yubo Wang, and Saffa Riffat. 2019. “Coordinated Control Strategy of DC Microgrid with Hybrid Energy Storage System to Smooth Power Output Fluctuation.” *International Journal of Low-Carbon Technologies* 15 (1): 46–54. https://doi.org/10.1093/ijlct/ctz056.

Wu, Yuan Kang, Pin Yue Wang, and Quoc Thang Phan. 2024. “Study on the Strategies to Mitigate the Effects of Partial Shading on PV Power Generation Systems.” *Conference Record - Industrial and Commercial Power Systems Technical Conference*, 1–7. https://doi.org/10.1109/ICPS60943.2024.10563766.

Xie, Zhongrun, Bin Li, Jiawei He, Qingquan Li, Wenbo Wang, Ye Li, and Botong Li. 2023. “Hybrid HVDC System Fault Transient Analysis Considering Traveling Wave Propagation and Converter Control Response.” *International Journal of Electrical Power and Energy Systems* 147 (November 2022). https://doi.org/10.1016/j.ijepes.2022.108794.

Xu, Yi, Liang Qin, Yi Zhang, Kaipei Liu, and Frede Blaabjerg. 2021. “DC Fault Current Estimation in a Multi-Terminal Hybrid MMC-HVDC System Considering Fault Ride Through Control.” *2021 IEEE Energy Conversion Congress and Exposition, ECCE 2021 - Proceedings*, no. October 2022, 3146–53. https://doi.org/10.1109/ECCE47101.2021.9595408.

Yan, Ziming, and Yan Xu. 2020. “Real-Time Optimal Power Flow: A Lagrangian Based Deep

Reinforcement Learning Approach." *IEEE Transactions on Power Systems* 35 (4): 3270–73. https://doi.org/10.1109/TPWRS.2020.2987292.

Yin, Y., A. Zamani, Z. Zhang, and Y. Fu. 2021. "Ground Fault Protection of Microgrid Interconnection Lines Using Distance Relay with Residual Voltage Compensation." *2021 74th Conference for Protective Relay Engineers, CPRE 2021*. https://doi.org/10.1109/CPRE48231.2021.9429847.

Yu, Guangxin, Wenjun Liang, Yonghu Wu, and Yuan Fang Li. 2023. "Explore the Correlation of Key Influence Factors on the Performance of Offshore Floating PV Power System Using Bifacial PV Modules and Their Optimization Method." *2023 IEEE 7th Conference on Energy Internet and Energy System Integration, EI2 2023*, 1679–85. https://doi.org/10.1109/EI259745.2023.10512383.

Zaben, Muiz M., Muhammed Y. Worku, Mohamed A. Hassan, and Mohammad A. Abido. 2024. "Machine Learning Methods for Fault Diagnosis in AC Microgrids: A Systematic Review." *IEEE Access* 12 (February): 20260–98. https://doi.org/10.1109/ACCESS.2024.3360330.

Zhang, Lin, Nengling Tai, Wentao Huang, Jian Liu, and Yanhong Wang. 2018. "A Review on Protection of DC Microgrids." *Journal of Modern Power Systems and Clean Energy* 6 (6): 1113–27. https://doi.org/10.1007/s40565-018-0381-9.

Zhang, Yichen, Jianzhe Liu, Feng Qiu, Tianqi Hong, and Rui Yao. 2022. "Deep Active Learning for Solvability Prediction in Power Systems." *Journal of Modern Power Systems and Clean Energy* 10 (6): 1773–77. https://doi.org/10.35833/MPCE.2021.000424.

Zhou, Dan, Shangren Chen, Hanyun Wang, Minyuan Guan, Lihua Zhou, Jian Wu, and Yaojie Hu. 2021. "Autonomous Cooperative Control for Hybrid AC/DC Microgrids Considering Multi-Energy Complementarity." *Frontiers in Energy Research* 9 (August): 1–10. https://doi.org/10.3389/fenrg.2021.692026.

Zhou, Shunyong, Yalan Zeng, Sicheng Li, Hao Zhu, Xue Liu, and Xin Zhang. 2022. "Surface Defect Detection of Rolled Steel Based on Lightweight Model." *Applied Sciences (Switzerland)* 12 (17). https://doi.org/10.3390/app12178905.

Appendix1: SMOTE implementation algorithm

```
from collections import Counter
from sklearn.datasets import make_classification
from imblearn.over_sampling import SMOTE
from matplotlib import pyplot
from numpy import where
from numpy import unique
y_test, y_predict = make_classification(n_samples=10000, n_features=2, n_redundant=0,
  n_clusters_per_class=1, weights=[0.99], flip_y=0, random_state=1)
# summarize class distribution
unique_values, counts = unique(y_predict, return_counts=True)
counter = Counter(dict(zip(unique_values, counts)))
counter = Counter(y_predict)
print(counter)
# transform the dataset
oversample = SMOTE()
y_test, y_predict = oversample.fit_resample(y_test, y_predict)
# summarize the new class distribution
counter = Counter(y_predict)
print(counter)
# scatter plot of examples by class label
for label, _ in counter.items():
  row_ix = where(y_predict == label)[0]
  pyplot.scatter(y_test[row_ix, 0], y_test[row_ix, 1], label=str(label))
pyplot.legend()
pyplot.show()
```

```
from imblearn.over_sampling import SMOTE
sm = SMOTE(random_state=42, k_neighbors=5)

```

```
X_res, y_res = sm.fit_resample(y_test, y_predict)
```

```
from random import randrange, uniform
from sklearn.neighbors import NearestNeighbors
import numpy as np
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, recall_score

X_train, X_test, y_train, y_test = train_test_split(X_res, y_res, test_size=0.2, random_state=42)
rf = RandomForestClassifier(random_state=42)
rf.fit(X_train, y_train)
y_pred = rf.predict(X_test)
```

```
confusion_matrix(y_test, y_pred)
```

```
array([[1932,   75],
       [  84, 1869]])
```

Appendix 2: Fault detection algorithm

```
import pandas as pd
```

```
import numpy as np
from keras.models import Sequential
from keras.layers import Dense
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
from sklearn.linear_model import LogisticRegression
```

```
df = pd.read_excel('/content/drive/MyDrive/Fin.xlsx')
print(df)
```

Show hidden output

```
df.iloc[5, 8] = 0
print(df)
```

Show hidden output

```
# Check if any row contains a value of 0
faulty_columns = (df == 0).any(axis=0)
if faulty_columns.any():
    print("Fault detected in the following columns:")
    print(df.columns[faulty_columns])
else:
    print("No faults detected.")
```

```
df.iloc[5, 8] = 0
print(df)
```

Show hidden output

```
# Check if any row contains a value of 0
faulty_columns = (df == 0).any(axis=0)
if faulty_columns.any():
    print("Fault detected in the following columns:")
    print(df.columns[faulty_columns])
else:
    print("No faults detected.")
```

Show hidden output

```
import pandas as pd
import numpy as np
import seaborn as sns
from sklearn.metrics import confusion_matrix
from numpy import array
# Import TimeseriesGenerator from tensorflow.keras
from tensorflow.keras.preprocessing.sequence import TimeseriesGenerator
from keras.layers import LSTM, Dense, Dropout
from sklearn.preprocessing import MinMaxScaler



from keras.layers import RepeatVector
from keras.layers import TimeDistributed
from keras.utils import plot_model

# Load the Excel file into a DataFrame
  # Replace with the path to your Excel file
df = pd.read_excel('/content/drive/MyDrive/Fin.xlsx')

# Convert the DataFrame to a numpy array
data = df.values

```

```python
import pandas as pd
import numpy as np
import seaborn as sns
from sklearn.metrics import confusion_matrix
from numpy import array
# Import TimeseriesGenerator from tensorflow.keras
from tensorflow.keras.preprocessing.sequence import TimeseriesGenerator
from keras.layers import LSTM, Dense, Dropout
from sklearn.preprocessing import MinMaxScaler



from keras.layers import RepeatVector
from keras.layers import TimeDistributed
from keras.utils import plot_model

# Load the Excel file into a DataFrame
  # Replace with the path to your Excel file
df = pd.read_excel('/content/drive/MyDrive/Fin.xlsx')

# Convert the DataFrame to a numpy array
data = df.values

# Assuming each row in the Excel file represents a sequence
# You may need to reshape your data accordingly if each row is not a sequence
# Reshape the data into sequences
sequence_length = 20  # Specify the length of each sequence
sequences = []
for i in range(len(data) - sequence_length + 1):
    sequence = data[i:i + sequence_length]
    sequences.append(sequence)

# Convert sequences to numpy array
sequences = np.array(sequences)

# Split sequences into features (X) and target (y)
X = sequences[:, :-1]  # Features (all columns except the last)
y = sequences[:, -1]   # Target (last column)
```

```
41
42 # prepare sequence
43 length = 5
44 seq = array([i/float(length) for i in range(length)])
45 X = seq.reshape(len(seq), 1, 1)
46 y = seq.reshape(len(seq), 1, 1)
47 # define LSTM configuration
48 n_neurons = length
49 n_in = len(sequence)
50 n_batch = length
51 n_epoch = 100
```

```
1 TRAIN_RATIO = 0.8
2 VAL_RATIO = 0.2
3 TEST_RATIO = 0.2
4 dataset_size = len(X)
```

```
1 X_train = X[:int(dataset_size*TRAIN_RATIO)]
2 y_train = y[:int(dataset_size*TRAIN_RATIO)]
```

```
1 X_validation = X[:int(dataset_size*TRAIN_RATIO):int(dataset_size*(TRAIN_RATIO+ VAL_RATIO))]
2 y_validation = y[:int(dataset_size*TRAIN_RATIO):int(dataset_size*(TRAIN_RATIO+ VAL_RATIO))]
```

```
1 X_test = X[:int(dataset_size*(TRAIN_RATIO+ VAL_RATIO)):]
2 y_test = y[:int(dataset_size*(TRAIN_RATIO+ VAL_RATIO)):]
```

```
1 # create LSTM
2 model = Sequential()
3 model.add(LSTM(200, activation='relu',  input_shape=(1, 1)))
4 model.add(Dense(1))
5 model.add(RepeatVector(n_in))
6 model.add(LSTM(200, activation='sigmoid', return_sequences=True))
7 model.add(TimeDistributed(Dense(1)))
8
9 num_classes = 10
10 model.add(Dense(num_classes))
11 model.compile(optimizer='adam', loss='mse', metrics=['accuracy'])
```

```
10 model.add(Dense(num_classes))
11 model.compile(optimizer='adam', loss='mse', metrics=['accuracy'])
```

```
/usr/local/lib/python3.10/dist-packages/keras/src/layers/rnn/rnn.py:204: UserWarning: Do not pass an `input_shape`/`input_dim` argument to a layer. When using Sequential models, prefer using an `Input(shape)` object as the first layer in the model instead.
  super().__init__(**kwargs)
```

```
[ ] 1 print(model.summary())
    2 # train LSTM
    3 model.fit(X_train, y_train, validation_data=(X_validation, y_validation), epochs=n_epoch, batch_size=n_batch, verbose=2)
    4
    5
```

Show hidden output

```
[ ] 1 print(y_test)
```

```
[ ] 1 y_predict = model.predict(X_test)
```

Show hidden output

```
[ ] 1 print(y_test)
    2 print(y_predict)
```

Appendix 3: MPPT algorithm

```
function D  = PandO(Param, Enabled, V, I)

% MPPT controller based on the Perturb & Observe algorithm.

% D output = Duty cycle of the boost converter (value between 0 and 1)
%
% Enabled input = 1 to enable the MPPT controller
% V input = PV array terminal voltage (V)
% I input = PV array current (A)
%
% Param input:
Dinit = Param(1);  %Initial value for D output
Dmax = Param(2);   %Maximum value for D
Dmin = Param(3);   %Minimum value for D
deltaD = Param(4); %Increment value used to increase/decrease the duty cycle D
% ( increasing D = decreasing Vref )
%

persistent Vold Pold Dold;

dataType = 'double';

if isempty(Vold)
    Vold=0;
    Pold=0;
    Dold=Dinit;
```

```
end
P= V*I;
dV= V - Vold;
dP= P - Pold;

if dP ~= 0 & Enabled ~=0
    if dP < 0
        if dV < 0
            D = Dold - deltaD;
        else
            D = Dold + deltaD;
        end
    else
        if dV < 0
            D = Dold + deltaD;
        else
            D = Dold - deltaD;
        end
```

Appendix 4: Overview of the IEEE 13 Bus in PYTHON

```
[ ]  1 df = pd.read_excel('/content/drive/MyDrive/Fin.xlsx')
     2 print(df)
```

	Bus611	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	\
0	Current	NaN	NaN	Voltage	NaN	NaN	
1	1	1.0	1.0	1	1.0	1.0	
2	1	1.0	1.0	1	1.0	1.0	
3	1	1.0	1.0	1	1.0	1.0	
4	1	1.0	1.0	1	1.0	1.0	
5	1	1.0	1.0	1	1.0	1.0	
6	1	1.0	1.0	1	1.0	1.0	
7	1	1.0	1.0	1	1.0	1.0	
8	1	1.0	1.0	1	1.0	1.0	
9	1	1.0	1.0	1	1.0	1.0	
10	1	1.0	1.0	1	1.0	1.0	
11	1	1.0	1.0	1	1.0	1.0	
12	1	1.0	1.0	1	1.0	1.0	
13	1	1.0	1.0	1	1.0	1.0	
14	1	1.0	1.0	1	1.0	1.0	
15	1	1.0	1.0	1	1.0	1.0	
16	1	1.0	1.0	1	1.0	1.0	
17	1	1.0	1.0	1	1.0	1.0	
18	1	1.0	1.0	1	1.0	1.0	
19	1	1.0	1.0	1	1.0	1.0	
20	1	1.0	1.0	1	1.0	1.0	

	Bus632	Unnamed: 7	Unnamed: 8	Unnamed: 9	...	Unnamed: 63	Unnamed: 64	\
0	Current	NaN	NaN	Voltage	...	Voltage	NaN	
1	800	50.0	40.0	1300	...	3	3.0	
2	-50	-50.0	-40.0	500	...	3	3.0	
3	50	50.0	40.0	-500	...	3	3.0	
4	-800	-50.0	-40.0	-1300	...	3	3.0	
5	800	50.0	40.0	1300	...	3	3.0	
6	-50	-50.0	-40.0	500	...	3	3.0	
7	50	50.0	40.0	-500	...	3	3.0	
8	-800	-50.0	-40.0	-1300	...	3	3.0	
9	800	50.0	40.0	1300	...	3	3.0	
10	-50	-50.0	-40.0	500		3	3.0	

1	3.0	6	4.0	3.0	2200	1500.0
2	3.0	1.5	1.5	1.5	600	600.0
3	3.0	-6	-4.0	-3.0	-2200	-1500.0
4	3.0	-1.5	-1.5	-1.5	-600	-600.0
5	3.0	6	4.0	3.0	2200	1500.0
6	3.0	1.5	1.5	1.5	600	600.0
7	3.0	-6	-4.0	-3.0	-2200	-1500.0
8	3.0	-1.5	-1.5	-1.5	-600	-600.0
9	3.0	6	4.0	3.0	2200	1500.0
10	3.0	1.5	1.5	1.5	600	600.0
11	3.0	-6	-4.0	-3.0	-2200	-1500.0
12	3.0	-1.5	-1.5	-1.5	-600	-600.0
13	3.0	6	4.0	3.0	2200	1500.0
14	3.0	1.5	1.5	1.5	600	600.0
15	3.0	-6	-4.0	-3.0	-2200	-1500.0
16	3.0	-1.5	-1.5	-1.5	-600	-600.0
17	3.0	6	4.0	3.0	2200	1500.0
18	3.0	1.5	1.5	1.5	600	600.0
19	3.0	-6	-4.0	-3.0	-2200	-1500.0
20	3.0	-1.5	-1.5	-1.5	-600	-600.0

	Unnamed: 71	Time
0	NaN	0
1	1200.0	1
2	600.0	2
3	-1200.0	3
4	-600.0	4
5	1200.0	5
6	600.0	6
7	-1200.0	7
8	-600.0	10
9	1200.0	11
10	600.0	12
11	-1200.0	13
12	-600.0	14
13	1200.0	15
14	600.0	16
15	-1200.0	17
16	-600.0	18
17	1200.0	19
18	600.0	20
19	-1200.0	21
20	-600.0	22

[21 rows x 73 columns]

Appendix 5: 13 Node IEEE Test Feeder Line Parameter Specifications

Table 1: Length parameters

Node A	Node B	Length(ft)	Configurations
632	645	500	603
632	633	0	602
633	634	500	XFM-1
645	646	300	603
650	632	2000	601
684	652	800	607
632	671	2000	601
671	684	300	604
671	680	1000	601
671	692	0	Switch
684	611	300	605
692	675	500	606

Table 2: Phase to phase parameters

Node	Load	Ph-1	Ph-1	Ph-2	Ph-2	Ph-3	Ph-3
	Model	kW	kVar	kW	kVar	kW	kVar
634	Y-PQ	160	110	120	90	120	90
645	Y-PQ	0	0	170	125	0	0
646	D-Z	0	0	230	132	0	0
652	Y-Z	128	86	0	0	0	0
671	D-PQ	385	220	385	220	385	220
675	Y-PQ	485	190	68	60	290	212
692	D-I	0	0	0	0	170	151
611	Y-I	0	0	0	0	170	80
	Total	1158	606	973	627	1135	753

Table 3: Grid neutral configuration

Configuration	Phasing	Phase	Neutral	Spacing
		ACSR	ACSR	ID
601	BACN	556,500 026/7	4/0 06/1	500
602	CABN	4/0 06/1	4/0 06/1	500
603	CBN	1/0	1/0	505
604	CAN	1/0	1/0	505
605	CN	1/0	1/0	510
Configuration	**Phasing**	**Phase**	**Neutral**	**Spacing**
606	ABCN	250,000 AA, CN	None	515
607	AN	1/0 AA, TS	1/0 Cu	520