A Project Report on

Efficient Fault Classification in Grid-Integrated Distributed Generators Using Ensemble CNN-LSTM Model on Raspberry Pi

Submitted to

Department of Electrical Engineering, National Institute of Technology, Raipur

Submitted By:

Mohit Dua (21117061)

Neeraj Yadav (21117064)

Tanishk Gupta (21117099)

Under the Guidance of:

Dr. Narendra. D. Londhe

Professor,

Department of Electrical Engineering,

CERTIFICATE

We hereby certify that the work which is presented in B.Tech. Major Project Report entitled **"Efficient Fault Classification in Grid-Integrated Distributed Generators Using Ensemble CNN-LSTM Model on Raspberry Pi"**. An authentic record of our own work completed between **July 2024 and November 2024** has been submitted to the Department of Electrical Engineering at the National Institute of Technology Raipur in partial fulfillment of the requirements for the award of the Bachelor of Technology in Electrical Engineering, under the supervision of **Dr. Narendra. D. Londhe**, Professor, Department of Electrical Engineering, NIT Raipur. The matter in this has not been submitted by us for the award of any other degree elsewhere.

Mohit Dua (21117061)

Neeraj Yadav (21117064)

Tanishk Gupta (21117099)

This is to certify that the above statement made by the candidates is correct to the best of our knowledge.

Signature of Supervisor:

Dr. Narendra D. Londhe

Professor

Electrical Engineering

National Institute of Technology, Raipur

<u>CANDIDATE DECLARATION</u>

We declare that this written submission entitled "**Efficient Fault Classification in Grid-Integrated Distributed Generators Using Ensemble CNN-LSTM Model on Raspberry Pi**" for the Bachelor of Technology in Electrical Engineering of **National Institute of Technology Raipur, Chhattisgarh** expresses our thoughts in our own words, and when we have borrowed words or ideas, we have properly cited and referenced the original sources. Furthermore, we affirm that we have followed all guide-lines for academic integrity and honesty and have not invented, distorted, or falsified any ideas, data, facts, or sources in our work.

Mohit Dua

Roll No: 21117061 **(Signature of student)**

Neeraj Yadav

Roll No: 21117064

(Signature of student)

Tanishk Gupta

Roll No: 21117099 **(Signature of student)**

ACKNOWLEDGMENT

First and foremost, we offer our sincerest gratitude to our supervisor
Dr. Narendra D. Londhe who has supported us throughout our major project with his utmost patience and enlightened us with his vast pool of knowledge whilst us the room to work in our way. We attribute the level of our success in completing the project to his encouragement and effort without which, this project would not have been a success.

We would also like to thank the technical staff members of the lab who availed us the required apparatus and provided help when we went astray with the knowledge of the working of the instruments.

In our daily schedule, we are blessed to have a friendly and cheerful group of fellow students and everyone else who had their minute contribution to the completion of this major project.

Mohit Dua (21117061)

Neeraj Yadav (21117064)

Tanishk Gupta (21117099)

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. ABSTRACT

Abstract

This project focuses on developing an intelligent fault classification system for microgrids integrated with distributed generators. Fault detection and classification are critical for maintaining system reliability and safety, especially in modern power grids. Our approach involves generating fault data using MATLAB simulations of a microgrid to create a comprehensive dataset representing various fault scenarios.
A deep learning model based on an Ensemble CNN-LSTM architecture was trained using this dataset. This model leverages advanced techniques to identify patterns and trends in spatial and temporal data, making it highly effective for fault classification. Once trained, the model was deployed on a Raspberry Pi, an edge computing device, for real-time fault detection and classification.

The system integrates a wireless TCP/IP communication link between MATLAB and the Raspberry Pi, enabling seamless data transmission whenever a fault occurs. The Raspberry Pi processes the received data autonomously, computes the probabilities of ten possible fault types, and predicts the fault with the highest probability as the most likely scenario.
In addition to fault prediction, this project demonstrates the potential of integrating machine learning with edge devices for scalable and decentralized energy systems. By ensuring that fault detection is automated and performed in real-time, the system significantly reduces downtime and enhances grid stability. The use of cost-effective and readily available hardware like Raspberry Pi ensures affordability and practical applicability in diverse scenarios, from small-scale grids to complex power networks.
By combining machine learning, edge computing, and wireless connectivity, this project provides a robust and practical solution for efficient fault detection in microgrids. This system minimizes response times, reduces the need for manual intervention, and enhances the reliability of distributed power systems, paving the way for future innovations in smart grid technology.

Keywords: Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), Transmission Control Protocol/Internet Protocol (TCP/IP), TensorFlow, Raspberry Pi, Distributed Generators (DG), Doubly Fed Induction Generator (DFIG)

2. INTRODUCTION

An electrical power system is made up of the interconnection of the generation, transmission, and distribution systems. A vital component of the power system, the transmission system acts as a link between a generating station and a distribution center. Its intricate structure makes a wide range of faults inevitable. Single-phase (LG) and even multi-phase (LL, LLG, LLL) faults are among the different types of transmission line malfunctions. For the power system to operate reliably, speedier fault clearing and power supply restoration is a desired quality for power engineers and power corporations. Owing to the complex system, it is difficult to maintain an uninterrupted power supply to end the consumer due to fault and also increasing the reliability of the users over the power supply [1]. Fault analysis, which has received increased attention recently, is therefore essential to analyzing a transmission line's fault. A secure relay functioning is guaranteed by a competent fault detection system. Therefore, the requirement for efficient fault classification ultimately results in the provision of high-speed protection and the reduction of power instability. Traditional methods, reliant on rule-based systems, struggle to cope with the complexities and variations inherent in a grid intertwined with distributed generators.

This project embarks on the exploration of fault classification within a grid integrated with distributed generators, leveraging the computational prowess of TensorFlow and the discriminative power of Ensemble CNN-LSTM. At the forefront of implementation is the Raspberry Pi, a cost-effective and versatile computing platform that aligns with the principles of edge computing within smart grids. The integration of deep learning models into this compact device promises to deliver a robust fault classification system capable of simulating and identifying diverse fault scenarios within the distributed generation landscape.

2.1. Motivation

As the global transition towards sustainable energy gains momentum, the integration of DGs becomes not just desirable but imperative. However, the increased complexity of grid systems, stemming from diverse sources of energy generation, necessitates a paradigm shift in fault detection methodologies. The motivation to harness the capabilities of TensorFlow and Ensemble CNN-LSTM lies in their potential to provide a scalable and adaptable solution to the evolving challenges of fault classification in distributed generation-integrated grids.

The urgent need for reliable fault detection and classification emanates from the potential consequences of grid failures. Swift identification of faults is critical to preventing cascading failures and ensuring the stable operation of the power grid. In this context, the application of deep learning techniques becomes particularly relevant, as they have demonstrated remarkable success in pattern recognition and complex data analysis.

2.2. Objective:

The overarching objectives of this project encompass a multi-faceted approach to address the intricacies of fault classification in a distributed generation-integrated grid:

Development of a Simulated Fault Dataset: A cornerstone of any machine learning project is the availability of a diverse and representative dataset. This project involves the creation of a comprehensive dataset that captures various fault scenarios within a grid enriched with distributed generators. This dataset will serve as the basis for training and evaluating the Ensemble CNN-LSTM model.

Implementation of an Ensemble CNN-LSTM Model using TensorFlow: TensorFlow, a dynamic deep learning framework, is chosen for its flexibility and ease of use. The Ensemble CNN-LSTM model designed for fault classification will be optimized to accommodate the unique characteristics of faults in grids integrated with distributed generators. The architecture will be tailored to extract meaningful features from the data, ensuring a high level of accuracy in fault identification.

Integration with Raspberry Pi: The implementation of the Ensemble CNN-LSTM model extends to the edge of the grid, represented by the Raspberry Pi. The adaptation of the model to operate in real-time on this edge computing device enhances the system's responsiveness. This integration aligns with the principles of edge computing, decentralizing the computational load and facilitating quick decision-making at the grid's periphery.

3. LITERATURE REVIEW

Transmission lines are critical components of power systems, and the timely and accurate identification of faults is crucial for ensuring the reliability and stability of the electrical grid. In recent years, researchers have explored innovative approaches to fault classification, with a growing interest in leveraging the capabilities of Raspberry Pi for real-time monitoring and analysis. The integration of smart grid technologies has paved the way for more sophisticated fault detection mechanisms. Traditional methods often rely on SCADA systems, but there is a growing need for decentralized and cost-effective solutions.

Raspberry Pi, with its low cost and versatility, emerges as a promising candidate for implementing intelligent fault detection systems on transmission lines. Several studies have highlighted the suitability of Raspberry Pi for power system applications. Its ability to interface with sensors and process data in real-time makes it an attractive platform for fault classification. The compact size and low power consumption of Raspberry Pi contribute to its feasibility for deployment in remote or unmanned locations on transmission lines. To enhance the accuracy of fault classification, machine learning algorithms have been widely employed.

Researchers have explored the implementation of algorithms such as Support Vector Machines (SVM) and Convolutional Neural Networks (Ensemble CNN-LSTM) on Raspberry Pi to process the acquired data and classify different types of faults. While the integration of Raspberry Pi in fault classification shows promise, challenges such as communication reliability, scalability, and algorithm optimization need further exploration. Future research should focus on refining the hardware-software integration, exploring advanced machine learning techniques, and addressing the practical challenges of field deployment.

4. METHODOLOGY

4.1. STUDIED SYSTEM

The system studied is the distribution network integrated with the DG and DFIG Wind Turbine. The system is modeled in MATLAB R2021a environment. The single diagram (SLD) of the distribution lines integrated with DFIG Wind Turbine and DG and PV solar panel is shown in Fig 1. The base frequency of system of the model is 60 Hz. The DG, DFIG Wind Turbine and PV Array with a capacity of 9MW with a DC to AC converter (Inverter) each are included in the system, are connected to Bus-2, Bus-3 and Bus-4 respectively. The rating of the utility grid is 1000 MVA working on 66KV voltage [2]. The system consists of four transformers: T - 1 steps down the voltage from 66 KV to 20 KV (66 KV/20 KV), T - 2 steps up the voltage from 400 V to 20 KV (400 V / 20 KV), while T - 3 steps up the voltage from 575 V to 20 KV (575 V / 20KV) and T-4 again steps up the voltage output of Inverter of value 400 V to 20 KV. There are three pi-section distribution lines DL-1, DL-2 and DL-3 having a length of 50 Kms each. There are total of four loads in the system Load-1 (L-1) connected to Bus-1, Load-2 (L-2) connected to Bus-2, Load-3 (L-3) and Load-4 (L-4) connected to Bus-4 respectively. L-1 is a reactive load of 6 MW and 2 MVAR rating while L-2 is resistive load of 3 MW and L-3 is resistive load of 3 MW and L-4 is also resistive load of 3MW.

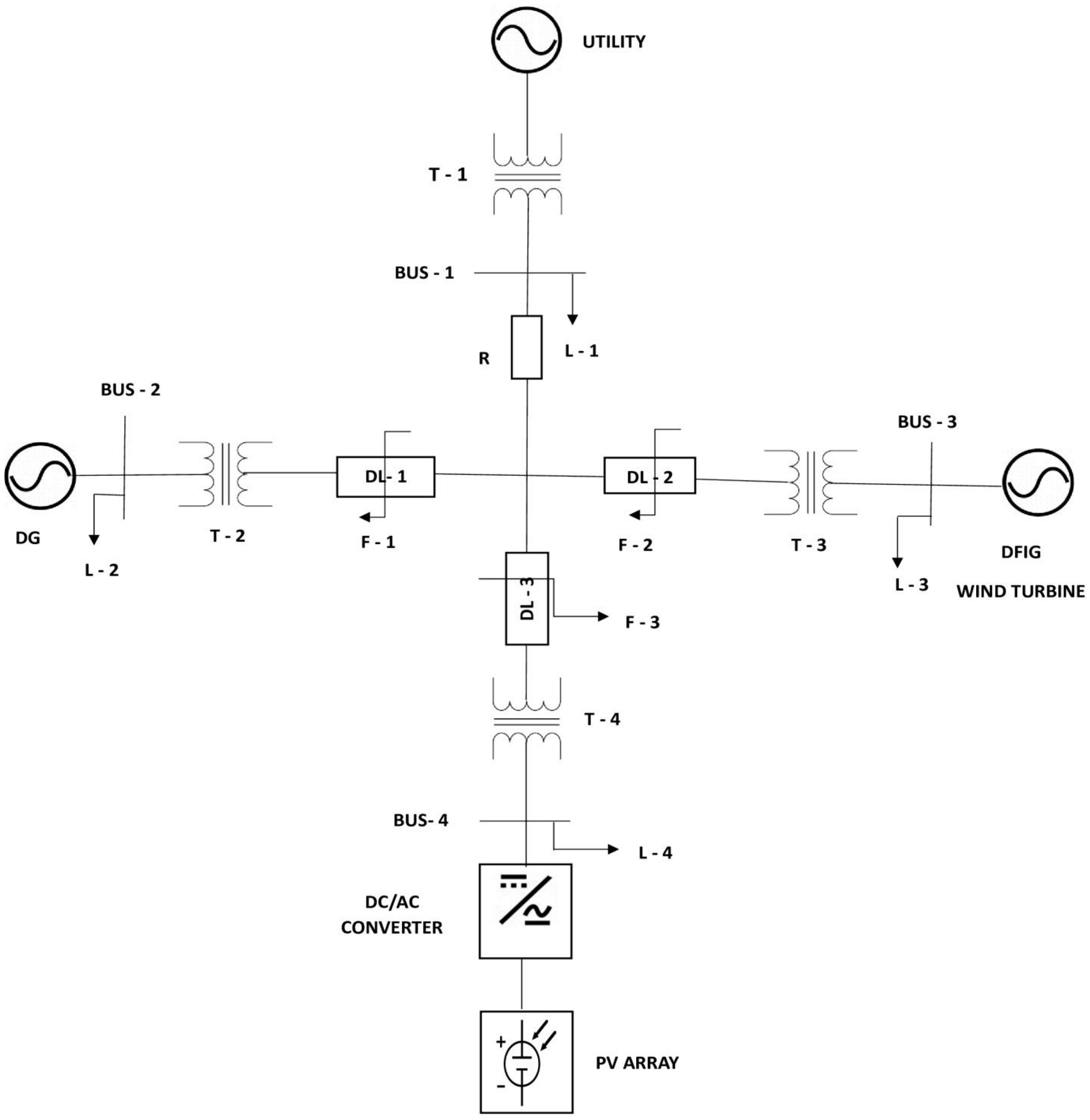


Fig. 1. Single line diagram of distribution network integrated with DG, DFIG Wind Turbine and PV Array.

- **Diesel Generator:**

A diesel generator is type of power generator that uses diesel engine to generate electrical energy. It consists of a diesel engine, an alternator (generator), a fuel system, and a control panel. Diesel generators are commonly used as backup power sources in situations where the main power supply is unreliable or during periods of peak electricity demand [3].

The diesel engine is the primary component of the generator and is responsible for converting fuel (diesel) into mechanical energy. This energy is then used to drive the generator. The alternator is connected to the diesel engine and converts the mechanical energy generated by the engine into electrical energy. This electrical energy is then used to power electrical devices and appliances. Diesel generators use diesel fuel, which is stored in a fuel tank connected to the generator. The fuel is injected into the engine where it combusts, producing the necessary mechanical energy. Diesel generators generate a significant amount of heat during operation. A cooling system is employed to dissipate this heat and prevent the generator from overheating. The exhaust system is responsible for expelling the by-products of the combustion process safely. It often includes a muffler to reduce noise. The control panel is the interface that allows users to start, stop, and monitor the generator. It typically includes controls for adjusting the output voltage, frequency, and other parameters.

Diesel generators serve as vital components supporting the main utility grid in diverse capacities. Primarily recognized for emergency backup power, these generators play a pivotal role during grid failures, ensuring uninterrupted electricity supply to different loads. Diesel generators find application in remote areas and islanded grids, functioning either as the primary power source or as part of hybrid systems, integrating renewable energy.

- **Doubly Fed Induction Generator (DFIG):**

A Doubly Fed Induction Generator (DFIG) is a type of wind turbine generator system [4] used in modern wind power plants. It is a popular choice for large-scale wind turbines due to its ability to provide better grid support and efficiency compared to fixed-speed wind turbine systems.

Here are some key features and components of a DFIG wind turbine:

- ✓ The main component is the doubly fed induction generator, which is an asynchronous (induction) generator.
- ✓ "Doubly fed" refers to the fact that both the stator and the rotor windings are connected to the power grid, allowing for variable speed operation.
- ✓ The stator is the stationary part of the generator, and the rotor is the rotating part. The stator is directly connected to the grid and provides constant- frequency electrical power.
- ✓ The rotor is connected to the grid through a power electronics converter system, allowing for variable rotor speed and control.
- ✓ DFIG wind turbines operate at variable speeds, meaning that the rotational speed of the rotor can be adjusted to optimize energy capture in varying wind conditions.
- ✓ Variable speed operation helps in maximizing energy capture and reducing mechanical stress on the turbine components.
- ✓ DFIG turbines are typically connected to the grid through the stator winding, providing a constant-frequency electrical output to the grid.
- ✓ The power electronics converter on the rotor side allows for control of the active and reactive power injected into the grid.

DFIG Wind Turbine provide enhanced grid support, improved efficiency, and the flexibility to operate at variable speeds.

- **Photovoltaic Array (PV Array):**

  A Photovoltaic (PV) Array is an assembly of interconnected solar panels designed to convert sunlight into electricity using photovoltaic cells. PV arrays are a cornerstone of modern renewable energy systems, widely used in residential, commercial, and industrial applications for sustainable energy generation.

Here are some key features and components of a PV Array:

- ✓ **Photovoltaic Cells:**
  - ▪ The basic building block of the PV array, made of semiconducting materials (commonly silicon).
  - ▪ Converts sunlight into direct current (DC) electricity through the photovoltaic effect.
- ✓ **Modules and Panels:**
  - ▪ PV cells are grouped into modules, and modules are combined to form panels.
  - ▪ Panels are arranged in series and parallel to meet voltage and current requirements.
- ✓ **Inverter Integration:**
  - ▪ Converts the DC electricity generated by the PV array into alternating current (AC) for compatibility with the power grid and household appliances.
- ✓ **Mounting Structures:**
  - ▪ Provides physical support and alignment of the PV array for optimal sunlight exposure.
  - ▪ Can include fixed-tilt or tracking systems that adjust the array's angle to follow the sun.
- ✓ **Bypass Diodes:**
  - ▪ Protect the system by reducing the effects of shading on specific cells or

modules, ensuring uninterrupted power generation.

- ✓ **Grid Connection or Standalone Systems:**
  - ▪ Grid-connected PV arrays feed surplus power into the grid and draw power when solar generation is insufficient.
  - ▪ Standalone systems often include batteries for energy storage to ensure continuous supply.
- ✓ **Efficiency and Durability:**
  - ▪ PV arrays are designed to maximize energy conversion efficiency and withstand environmental factors like wind, rain, and temperature fluctuations.

4.2. DATA GENERATION

Significant causes of fault in a power system include weather, short circuits between two conductors, flashovers, and insulation failure. These faults fall into two categories: unsymmetrical faults (LG, LL, LLG) and symmetrical faults (LLL). Fault analysis is essential to the research of fault conditions and their effects on the power system during the behavior of voltage and current signals. As a result, defects in the system can be effectively classified and identified using current and voltage waveforms.

Simulation of fault is done in MATLAB/SIMULINK environment. A three-phase fault block available in a Simulink is used to simulated a real time scenario. The simulation model of the system under study is used to create the three-phase voltage and current signal data. Relays are used to sample three-phase voltage and current readings and are capable of detecting faults in three distribution lines (DL-1, DL-2, DL-3). The frequency of the system is 60 Hz and sampling frequency is 4.8 kHz [5]. Therefore, the total number of samples in one cycle will be = 80 (i.e.4800÷60) samples. A vast number of datasets are generated by simulating various fault scenarios, which helps to test and train the suggested model's adaptability. This is achieved by setting different parameters and configurations of the modelled system. The simulated data is fetched through both the faulty buses in which fault has occurred for quick identification of a fault [2]. The different configurations for the fault cases are tabulated in Table 1. For the generation of all types of faults, 10 fault location is considered from 3 different fault lines with 7 different fault resistance and 12 different inception angles. The total no. of cases in each of the fault class is 840 = (10×7×12). The total number of cases in each of the fault class with their corresponding label is tabulated in Table 2. The potential combinations of fault resistances and fault inception angles at various lines and locations within the system under study are used to simulate the various sorts of faults. There are differences in the fault resistance between low and high value sets. The fault current level would change as a result of altering the fault resistances. The fault occurs with different characteristics at any angle between 0° and 360°.

Table 1:

Configuration of fault cases simulation.

Parameters	**Configuration**	**No. of Cases**
Fault type	AG, BG, CG, AB, BC, AC, ABG, BCG, ACG, ABC	10
Fault line	DL-1, DL-2, DL-3 (each with a 50 km distance)	3
Fault location (in km)	1, 6, 11, 16, 21, 26, 31, 36, 41, 46 (at every 5 Km in a line)	10
ult Inception gle (in degree)	0, 30, 60, 90, 120, 150, 180, 210, 240, 270, 300, 330	12
esistance (in ohm)	0.001,0.01, 0.1, 1, 10, 50, 100	7

Table 2:

Total cases in each fault class

Fault Class Label	**Fault Class**	**No. of cases**
1	AG	840
2	BG	840
3	CG	840
4	AB	840
5	BC	840
6	AC	840
7	ABG	840
8	BCG	840
9	ACG	840
10	ABC	840

4.3. Training of Model:

The project implements a neural network model using TensorFlow/Keras to classify or predict from a given dataset. The architecture integrates multiple layers with specific functionalities, as depicted in the model summary and output visualization.

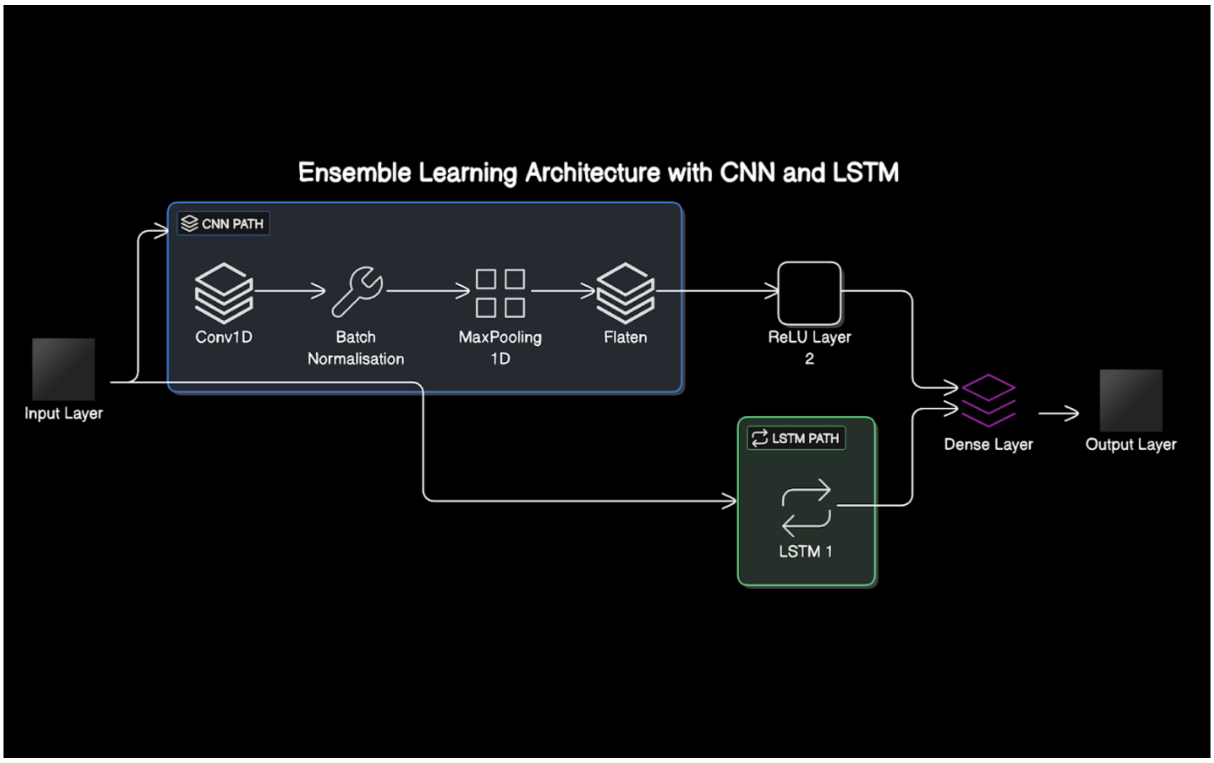


Fig. 2. Ensemble Learning Architecture with CNN and LSTM

The model begins with an input layer named input_layer_5, which accepts data in the shape (None, 80, 6). This indicates a dataset comprising samples with 80-time steps, each having six features. The subsequent layers employ functional APIs to define the architecture.

1. **Functional Layers**: The first two functional layers, functional_3 & functional_4, apply operations to the I/P data, resulting in outputs of shapes (None, 640) & (None, 100), respectively. These layers involve trainable parameters, with functional_3 having 25,216 & functional_4 containing 43,200 parameters. These transformations likely use dense or convolutional layers to extract features.

2. **Concatenation**: After feature extraction, the outputs from the functional layers are concatenated using the concatenate_1 layer. This concatenation merges the features into a

combined output of shape (None, 740) for downstream processing.

3. **Dense Layers**: -

- The dense_2 layer takes the concatenated output and applies a dense transformation, reducing it to a shape of (None, 128) with 94,848 trainable parameters. Dense layers in neural networks are used for learning complex feature relationships.
- A dropout layer (dropout_1) follows, introducing regularization to prevent overfitting by randomly deactivating neurons during training. The dropout layer retains the shape (None, 128) since it only modifies neuron activity.

4. **Output Layer:** The final dense layer, dense_3, outputs predictions in a shape of (None, 10), corresponding to ten possible classes or categories. This layer uses 1,290 trainable parameters. A SoftMax activation function is likely used here for multi-class classification.

The model contains a total of **328,462 parameters**, with **163,906 trainable parameters** contributing to learning, and **648 non-trainable parameters** used for pre-initialized features.

Model Evaluation and Results

The model is evaluated using a test dataset, and the results are stored in the variable **score**. The output provided shows the loss value and accuracy score:

- Loss: **0.0603454**
- Accuracy: **0.9825396** (approximately 98.25%)

These metrics indicate a well-performing model with minimal error and high predictive accuracy.

Training Process

The training process involves plotting training and validation accuracies using Matplotlib. The graph uses the following configurations:

- Training accuracy is plotted in red ('r') with a linewidth of 3.
- Validation accuracy is plotted in blue ('b') with the same linewidth. The plot includes labelled axes ("Epochs" and "Accuracy") and a title ("Accuracy Curves"). The legend differentiates between "Training Accuracy" and "Testing Accuracy," providing a clear visualization of model performance over epochs.

This detailed explanation covers the model structure, its parameters, evaluation metrics, and visualization for the report. Let me know if you need further assistance with formatting or additional sections.

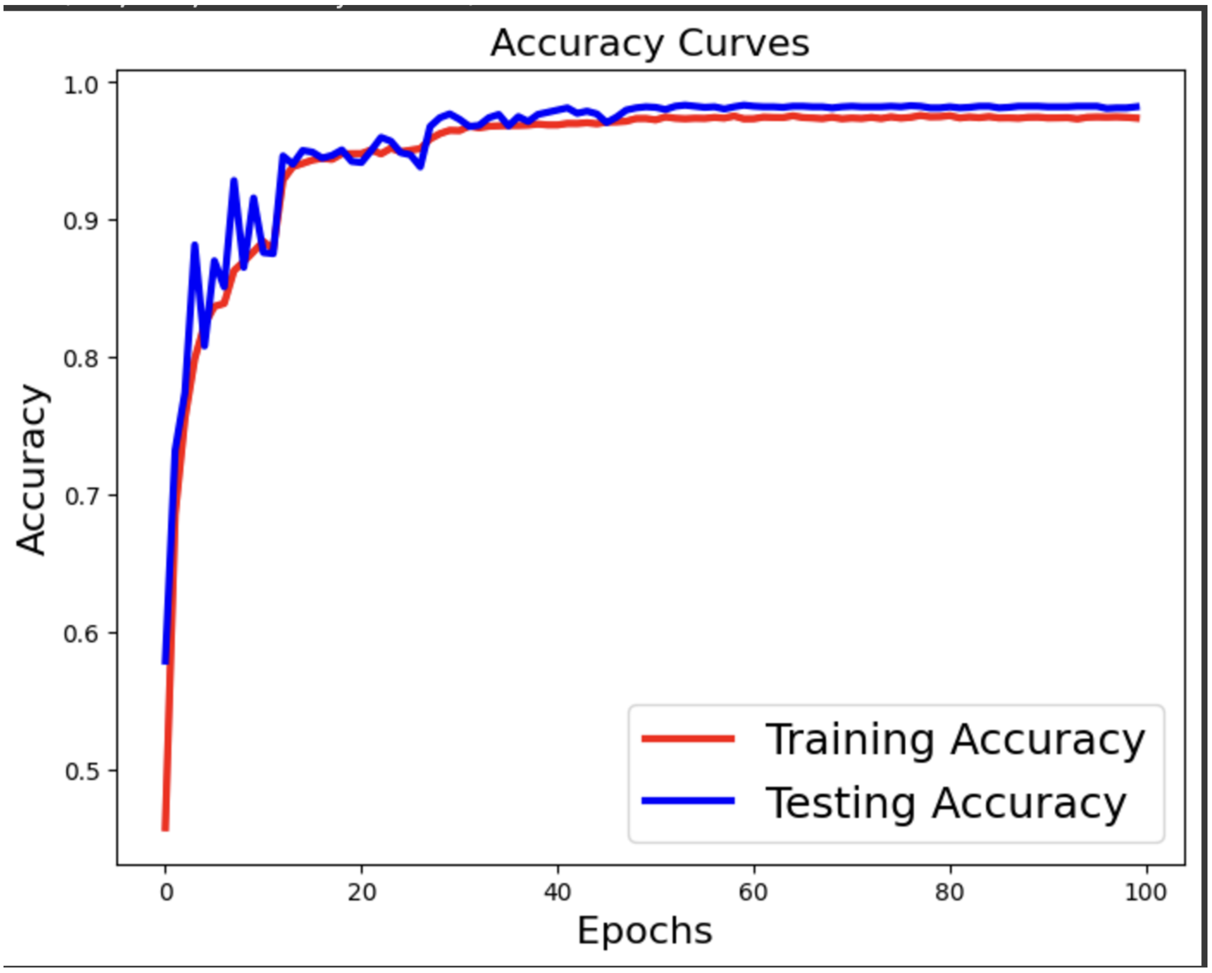


Fig. 3. Training and Testing Accuracy Curve of Model

The graph illustrates the accuracy curves for both training and testing datasets over 100 epochs. The red line represents the training accuracy, while the blue line denotes the testing accuracy. These curves provide insights into the model's learning behaviour and generalization capabilities throughout the training process.

Observations:

1. **Initial Phase (Epochs 0-10)**:

   - During the initial epochs, the training accuracy rapidly increases from approximately 50% to nearly 90%.

- The testing accuracy follows a similar trend, indicating that the model quickly learns meaningful patterns from the data.

2. **Intermediate Phase (Epochs 10-50)**:

   - Both training and testing accuracies steadily improve and converge towards the 95-98% range.
   - The close alignment of training and testing accuracies in this phase demonstrates that the model is not overfitting but effectively generalizing to unseen data.

3. **Final Phase (Epochs 50-100):**

   - The training accuracy plateaus at around 98-99%, while the testing accuracy stabilizes slightly lower, at approximately 97-98%.
   - This consistent performance indicates that the model has reached its maximum learning capacity without further improvements.

Key Insights:

- High Generalization: The minimal gap between training and testing accuracies suggests that the model is well-regularized and capable of generalizing beyond the training dataset.
- No Overfitting: Overfitting typically manifests as a widening gap between training and testing accuracies as training progresses. Here, the parallel nature of the two curves indicates that overfitting is not a concern.
- Effective Training: The steep increase in accuracy during the initial epochs highlights the model's efficiency in learning key features from the dataset.

The accuracy curves demonstrate an excellent balance between learning and generalization. The model achieves a high accuracy of **98.25%** on the testing dataset, as seen in the provided evaluation output, reinforcing its reliability for practical applications. This performance reflects a well-designed architecture and an appropriately tuned training process.

Optimizer Details

The optimizer plays a crucial role in training the neural network by adjusting the weights to minimize the loss function. While the optimizer details are not explicitly shown in the provided screenshots, it is inferred from the context of TensorFlow/Keras projects that a commonly used optimizer, such as **Adam**, may have been employed.

Adam Optimizer

If the Adam optimizer is used, it integrates the advantages of both the RMSprop and SGD optimizers, offering adaptive learning rates for each parameter. This is highly beneficial for models with complex architectures like the one in this project.

✓ Key Features of Adam:

- Combines momentum and adaptive learning rates for faster convergence.
- Automatically adjusts the learning rate during training, making it robust for various tasks.
- Requires less hyperparameter tuning compared to other optimizers.

Relevance to Model Performance

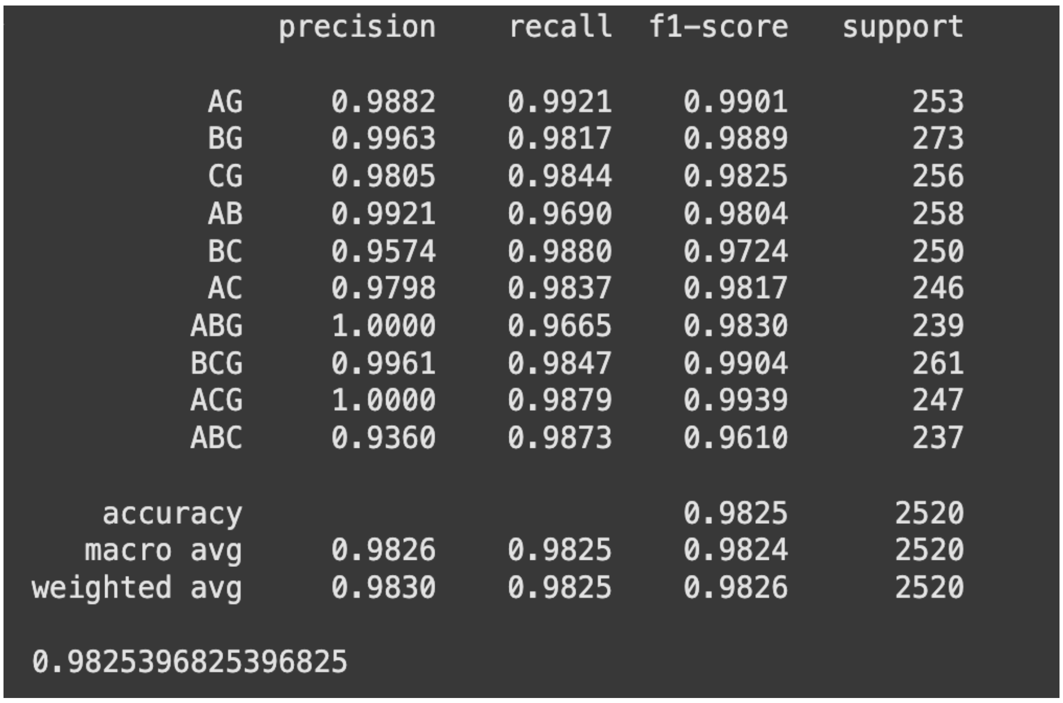
The choice of optimizer directly impacts the speed of convergence and the ability to reach a global minimum in the loss function. Given the model's rapid learning phase in the initial epochs (as evidenced by the accuracy curves), the optimizer effectively updates weights for meaningful learning. Additionally, the consistent performance across training and testing datasets suggests a stable optimization process without oscillations or divergence.

Learning Rate

The learning rate used in the optimizer is a critical hyperparameter influencing model training. If the learning rate is set too high, the model may overshoot the optimal solution, while a low learning rate could lead to slow convergence. The observed smooth increase in accuracy and the plateauing at high values imply that the learning rate was appropriately chosen, either statically or adaptively.

Table 3:

Classification Report



```
               precision    recall  f1-score   support

          AG     0.9882    0.9921    0.9901       253
          BG     0.9963    0.9817    0.9889       273
          CG     0.9805    0.9844    0.9825       256
          AB     0.9921    0.9690    0.9804       258
          BC     0.9574    0.9880    0.9724       250
          AC     0.9798    0.9837    0.9817       246
         ABG     1.0000    0.9665    0.9830       239
         BCG     0.9961    0.9847    0.9904       261
         ACG     1.0000    0.9879    0.9939       247
         ABC     0.9360    0.9873    0.9610       237

    accuracy                         0.9825      2520
   macro avg     0.9826    0.9825    0.9824      2520
weighted avg     0.9830    0.9825    0.9826      2520

0.9825396825396825
```

4.4. Significance of Raspberry Pi

Raspberry Pi, developed by the Raspberry Pi Foundation, stands as a paradigm of accessibility and versatility in the realm of single-board computers. At its core, the credit card-sized device packs a Broadcom BCM2711 Quad-core Cortex-A72 processor, coupled with 4GB LPDDR4-3200 SDRAM, offering a robust foundation for a myriad of applications. Boasting USB 3.0 and Gigabit Ethernet for connectivity, along with Wi-Fi and Bluetooth options, Raspberry Pi has become a potent tool for networking and IoT projects. Its compatibility with various operating systems, notably Raspberry Pi OS, facilitates programming in languages such as Python, Scratch, and Java, appealing to a diverse user base.

A distinguishing feature lies in its General-Purpose Input/Output (GPIO) pins, allowing users to interface with an array of external devices, amplifying its utility in physical computing and experimental projects. The active and engaged community surrounding Raspberry Pi plays a pivotal role, providing an extensive repository of resources, tutorials, and forums. The hardware's evolution across generations, exemplified by the Raspberry Pi 4, introduces advancements like 4Kp60 video decoding and dual micro-HDMI ports, enhancing its multimedia capabilities.

However, Raspberry Pi's significance transcends its technical prowess; it embodies a democratization of computing access. Affordability, combined with its multifaceted applications, positions it as an invaluable tool for enthusiasts, educators, and professionals alike. Its role in fostering innovation, particularly in IoT and educational initiatives, underscores its impact on the democratization of technology education globally. In essence, Raspberry Pi not only serves as a powerful computing device but also as a catalyst for empowerment, inspiring creativity and learning in the ever-expanding landscape of technology.

5. EXPERIMENTAL SETUP

To get started with the classification of transmission faults, the following components are required:

5.1. Required hardware:

- Raspberry Pi 4 MODEL B 4GB RAM
- A boot media i.e., a microSD card 32 GB
- A display
- A micro-HDMI to HDMI cable
- A keyboard
- A mouse
- 5V/3A 15W USB-C Power Supply (official Raspberry Pi power supply)
- Ethernet Connection / WIFI

Step 1: Flash microSD card using Raspberry Pi Imager

Visit the Raspberry Pi website and download the latest Raspberry Pi OS through the Raspberry Pi imager to install the OS on the microSD card, by following the onscreen instructions. Choose the device, the OS, and the storage, write the OS, and apply the OS customization. Now insert the microSD card into the respective slot provided in the Raspberry Pi.

Step 2: Connect the Raspberry Pi with the accessories

By using the USB ports, connect the keyboard and mouse to the Raspberry Pi. Plug the HDMI0 port into the primary monitor which serves as a display with the help of a micro-HDMI to HDMI cable. For networking and connectivity, plug the ethernet cable into the ethernet port. Finally, connect the power supply to the Raspberry Pi, now the boot screen will be displayed within minutes.

Specifications of Raspberry Pi

1. **Processor:** Broadcom BCM2711, Quad-core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz
2. **Memory (RAM):** 4GB LPDD R4-3200 SDRAM
3. **Connectivity:**
   a. 2 × USB 3.0 ports
   b. 2 × USB 2.0 ports
   c. Gigabit Ethernet (RJ45)
   d. 802.11b/g/n/a wireless LAN
   e. Bluetooth 5.0
4. **Video and Sound:**
   a. 2 × micro-HDMI ports (supports up to 4Kp60)
   b. MIPI DSI display port [8]
   c. MIPI CSI camera port
   d. 4-pole stereo audio and composite video port
5. **Multimedia:**
   a. H.265 (4Kp60 decode)
   b. H.264 (1080p60 decode, 1080p30 encode)
   c. OpenGL ES 3.0 graphics
6. **Storage:**
   a. MicroSD card slot for operating system and data storage
7. **GPIO:**
   a. 40-pin GPIO header (fully backward-compatible with previous Raspberry Pi boards)
8. **Power:**
   a. 5V/3A DC via USB-C connector
   b. 5V DC via GPIO header
9. **Operating Temperature:** 0°C to 50°C ambient
10. **Dimensions:** 85.6mm × 56.5mm × 17mm

<u>Step 3</u>: Raspberry Pi Configuration

We are setting up Bluetooth, locale settings, configuring the username and password checking for software updates, and clicking restart. Raspberry Pi desktop is ready.

<u>Step 4:</u> Installing TensorFlow and install dependencies on raspberry pi

We activated the virtual environment before running the pip commands, and the installation of NumPy and other supporting Python packages was done. Installed TensorFlow touch vision and touch audio from the official website of TensorFlow on the Raspberry Pi. By activating the virtual environment in the terminal of the Raspberry Pi, to check the version of the torch and other pip packages we created a Python script in the terminal.

<u>Step 5:</u> Finding the IP address of the Raspberry Pi

Connect to the WIFI or access the network through the ethernet/LAN and enter the command if-config in the terminal so that we can get the IP address.

<u>Step 6:</u> Connecting an LCD display to Raspberry Pi

Identify GPIO pins, understand the configuration, and connect the GPIO pins to the LCD display with the male-to-female jumper wires and install the library called **Adafruit_CharLCD** on the Raspberry Pi and install the library **RPi.GPIO** for controlling GPIO pins on the Raspberry Pi. Define LCD pin mapping set up the GPIO pins and test the LCD display using Python script

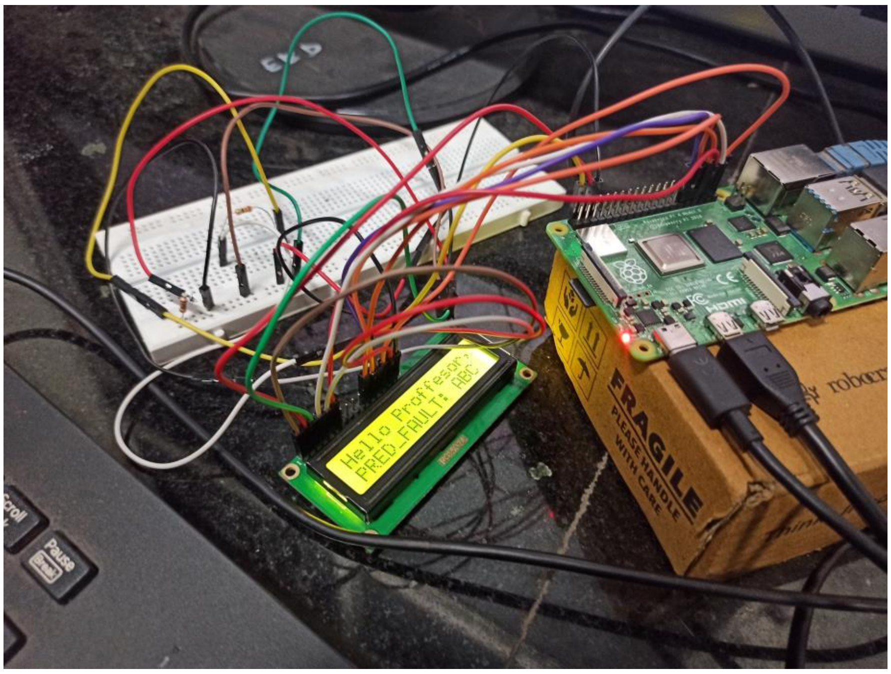


Fig.4: Experimental setup of hardware

5.2. Deployment of the TensorFlow model on Raspberry Pi

Deployment of TensorFlow Model

The deployment of a TensorFlow model involves making the trained model available for use in real-world applications or production environments. Our model was trained using TensorFlow and Keras on the simulated fault dataset, and we saved the model using the tensorflow.keras.models.save_model() function in the HDF5 format (.h5). The training was conducted on a high-performance computer due to the computationally intensive nature of deep learning, which requires substantial hardware resources.

We saved the trained model to the Raspberry also we created a Python script to load the TensorFlow model using the tensorflow.keras.models.load_model() function. Once the model was successfully loaded, we provided input data for inference and ran the script to test the model's performance on the Raspberry Pi. After verifying its functionality, we further optimized the deployment to ensure the model ran efficiently on the limited computational resources of the Raspberry Pi. After running the test scripts, we made TCP/IP connection to receive input data from remote device and replaced the test data path with the actual received data path.

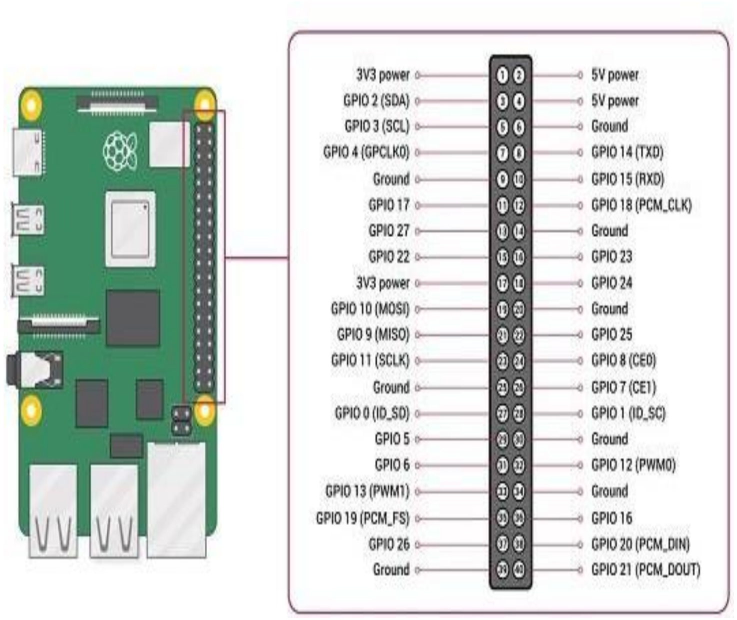
5.3. Interfacing of 16×2 LCD with Raspberry Pi

Interfacing a 16x2 LCD (Liquid Crystal Display) with a Raspberry Pi [9] involves connecting the LCD to the GPIO (General Purpose Input/Output) pins of the Raspberry Pi and programming it to display information.

Components Required: The components required for LCD display & Raspberry Pi interfacing are as follows:

- Raspberry Pi
- 16×2 LCD
- Resistor 330 Ω
- Jumper Cables
- Breadboard

Raspberry Pi and 16×2 LCD Pin Diagrams:



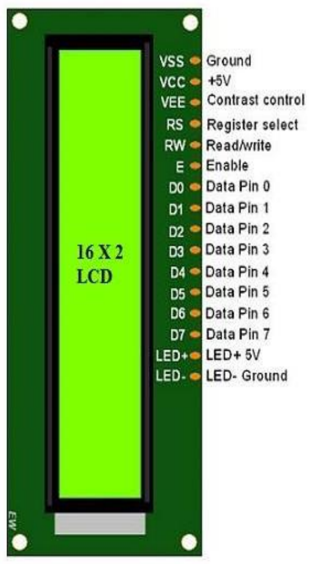


Fig.5: Raspberry Pi GPIO Pin diagram

Fig.6: 16×2 LCD Pin diagram

Table 4:

Connections of 16×2 LCD pins with Raspberry Pi GPIO pins are as shown in the table below:

16×2 LCD	**Raspberry Pi**
VSS	GND
VDD	5V
VE/VO	Middle Pin of 4.7KΩ POT
RS	GPIO26
RW	GND
E	GPIO19
D4	GPIO13
D5	GPIO6
D6	GPIO5
D7	GPIO21
A/LED+	To 5V through 330 ohms resistor
K/LED-	GND

The testing output can be obtained by running the Python code. This shows that the LCD and Raspberry Pi are correctly connected and prepared to display data.

5.4. Establishing wireless connection

We are facilitating the Raspberry Pi to receive data from remote system. The data contains voltage and current values of all three phases. Data will be received by Raspberry pi and converted to Numpy format. Our Deep Learning Algorithm will analyze the data

and predict the type of fault whose probability will be maximum.
To establish wireless connection, we have used TCP/IP data communication protocol. We have two systems communicating via TCP/IP networking: a **Sender** system running MATLAB and a **Receiver** system, the **Raspberry Pi**, running Python.

Sender (MATLAB System)
The MATLAB system acts as the **Sender** in this communication. The process starts by simulating fault data, which is saved in an Excel file (or CSV format). The system then initiates a TCP connection to the Raspberry Pi using its IP address (172.22.75.XXX) and a specified port number (98XX). The MATLAB script reads the fault data from the file in binary mode and sends the file size first to the Raspberry Pi, followed by the file's data in chunks. The data is sent using the **fwrite** function, where each chunk is sent sequentially until the entire file is transmitted

Receiver (Raspberry Pi – Python System)
The **Receiver** is the Raspberry Pi, which is programmed to accept incoming data over TCP/IP. The Python script running on the Raspberry Pi listens for incoming connections, accepts them, and retrieves the fault data sent by the MATLAB system. Upon receiving the data, the Raspberry Pi first stores the received file in a specified directory. Then, it processes the file, converting it into a format that can be fed into a deep learning model.

6. RESULT

We created a Python script in the Thonny IDE on Raspberry PI. We loaded and tested the deployed Tensorflow model on the Raspberry Pi we also made TCP/IP connection for receiving data. Also connected an LCD display to the Raspberry Pi through GPIO pins and wrote necessary python script. As soon as we sent data from the remote system Raspberry Pi Receives all the data, the data is converted to Numpy format and then our Tensorflow model takes that data analysed it and performed classification of faults and displayed on the LCD display as shown in the below figure.

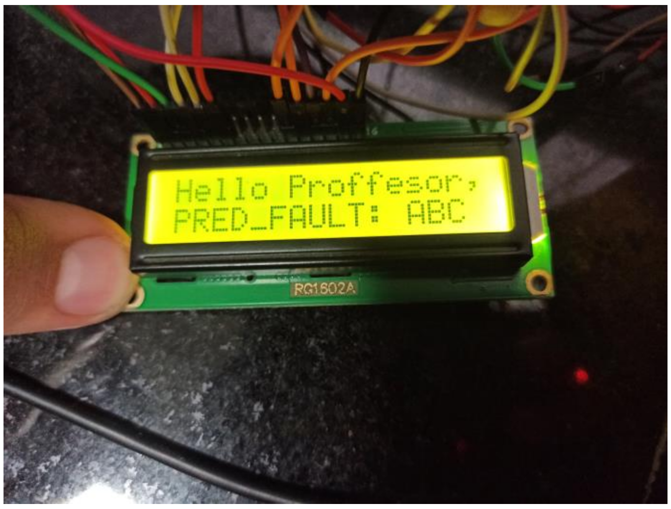


Fig.7: LCD display showing the predicted fault

7. CONCLUSION

In summary, our Convolutional Neural Network-Long Short-Term Memory (Ensemble CNN-LSTM) model, specially designed for fault classification in a grid with distributed generators and run on a Raspberry Pi, combines strong theory with practical usability. The convolutional layers, with different filter sizes and numbers, are great at recognizing complex spatial patterns in situations where faults might happen. Activation functions like ReLU add a non-linear touch, helping the model understand nuanced relationships. Batch Normalization keeps things stable and speeds up learning, crucial in real-world situations with dynamic and complex fault patterns.

The Flatten layer smoothly turns spatial data into a simpler form, making it easier for the fully connected layers to pick up on important features. These linear layers, with their adjustable transformations and Re LU activations, play a big role in creating abstract representations for the final classification. The Soft-max layer wraps up the architecture, turning the model's output into probabilities for different fault categories. What's note-worthy is that this model is adaptable to places with limited resources, like the Raspberry Pi, making it useful in real- world situations with distributed generators.

Throughout training, the model continuously learns and refines its understanding of fault patterns, showing a strong convergence. Our careful design choices, from layer selection to using a learning rate schedular, help the model looks promising, future work could explore methods like attention mechanisms to make it even better. In essence, this Ensemble CNN-LSTM model, with its smart design and practical implementation, is a significant step toward improving fault classification in grids with distributed generators, laying the groundwork for more resilient and adaptable power systems.

8. REFERENCES

[1] Pavan Kumar Bais, Narendra D. Londhe, Ritesh Raj, Hira Singh Sachdev. "Faulty Line Localization in IEEE 30 Bus System Using CNN-LSTM", 2022 Fourth International Conference on Emerging Research in Electronics, Computer Science and Technology (ICERECT), 2022

[2] Praveen Rai, Narendra D. Londhe, and Ritesh Raj. "Fault classification in power system distribution network integrated with distributed generators using CNN." Electric Power Systems Research 192 (2021): 10691

[3] "Overcoming Barriers To Scheduling Embedded Generation To Support Distribution Networks" PDF). BERR. Archived from the original (PDF) on 2010-03-04. Retrieved 2014-07-15.

[4] Wind Farm - Doubly-Fed Induction Generator (DFIG) https://in.mathworks.com/help/sps/ug/wind-farm-dfig-detailed-model.html

[5] Shi, Shenxing, et al. "Fault classification for transmission lines based on group sparse representation." IEEE Transactions on Smart Grid 10.4 (2018): 4673- 4682.

[6] Li, Dan, et al. "Classification of ECG signals based on 1D convolution neural network." 2017 IEEE 19th International Conference on e- Health Networking Applications and Services (Healthcom). IEEE, 2017.

[7] Scherer, M. P. D., Müller, A., Behnke, S. (2010). Evaluation of pooling operations in convolutional architectures for object recognition. International Conference of Artificial Neural Networks, pp. 92-101.10.1007/978-3-642- 15825-4_10

[8] https://www.raspberrypi.com/products/raspberry-pi-4-model- b/specifications/

[9] https://iotstarters.com/interfacing-16x2-lcd-with-raspberry- pi/? expand_article = 1

PROJECT_REPORT_FINAL[1].pdf

ORIGINALITY REPORT

16%	7%	10%	7%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Praveen Rai, Narendra D. Londhe, Ritesh Raj. "Fault classification in power system distribution network integrated with distributed generators using CNN", Electric Power Systems Research, 2020 Publication	3%
2	Pavan Kumar Bais, Narendra D. Londhe, Ritesh Raj, Hira Singh Sachdev. "Faulty Line Localization in IEEE 30 Bus System Using CNN-LSTM", 2022 Fourth International Conference on Emerging Research in Electronics, Computer Science and Technology (ICERECT), 2022 Publication	1%
3	Submitted to National Institute of Technology, Raipur Student Paper	1%
4	www.packtpub.com Internet Source	1%
5	www.iotstarters.com Internet Source	1%

6	idoc.pub Internet Source	1 %
7	Submitted to The Scientific & Technological Research Council of Turkey (TUBITAK) Student Paper	<1 %
8	Submitted to University of Bradford Student Paper	<1 %
9	www.electronics-lab.com Internet Source	<1 %
10	Vivek S. Sharma, Shubham Mahajan, Anand Nayyar, Amit Kant Pandit. "Deep Learning in Engineering, Energy and Finance - Principles and Applications", CRC Press, 2024 Publication	<1 %
11	espritrock.org Internet Source	<1 %
12	Submitted to University of Technology Bahrain Student Paper	<1 %
13	Submitted to Vaal University of Technology Student Paper	<1 %
14	Submitted to Heart of Yorkshire Education Group Student Paper	<1 %
15	www.coursehero.com Internet Source	<1 %

16	Submitted to University of North Carolina, Greensboro Student Paper	<1%
17	Submitted to ABV-Indian Institute of Information Technology and Management Gwalior Student Paper	<1%
18	Ashwin Pajankar. "Practical Linux with Raspberry Pi OS", Springer Science and Business Media LLC, 2021 Publication	<1%
19	Submitted to Liverpool John Moores University Student Paper	<1%
20	Submitted to Southern New Hampshire University - Continuing Education Student Paper	<1%
21	Submitted to De Montfort University Student Paper	<1%
22	Submitted to Heald College, LLC Student Paper	<1%
23	Submitted to University of Bedfordshire Student Paper	<1%
24	fastercapital.com Internet Source	<1%

www.electromaker.io

25	Internet Source	<1 %
26	Submitted to Heriot-Watt University Student Paper	<1 %
27	Submitted to Teaching and Learning with Technology Student Paper	<1 %
28	blog.actorsfit.com Internet Source	<1 %
29	econpapers.repec.org Internet Source	<1 %
30	electronicshobbyists.com Internet Source	<1 %
31	ethesis.nitrkl.ac.in Internet Source	<1 %
32	Lecture Notes in Electrical Engineering, 2014. Publication	<1 %
33	Submitted to University of Ibadan Student Paper	<1 %
34	igrid.net.au Internet Source	<1 %
35	www.ijraset.com Internet Source	<1 %
36	www.memoryexpress.com Internet Source	

		<1%
37	Anurag Tikariha, Narendra D. Londhe, Baidyanath Bag, Ritesh Raj. " Classification of faults in an 30 bus transmission system using fully convolutional network ", International Transactions on Electrical Energy Systems, 2021 Publication	<1%
38	J.L. Rodriguez-Amenedo, S. Arnalte, J.C. Burgos. "Automatic generation control of a wind farm with variable speed wind turbines", IEEE Transactions on Energy Conversion, 2002 Publication	<1%
39	Muyiwa Adaramola. "Wind Resources and Future Energy Security - Environmental, Social, and Economic Issues", Apple Academic Press, 2019 Publication	<1%
40	Submitted to University of Cape Town Student Paper	<1%
41	Zhenhua Cai, Canbing Li, Nengling Tai, Wentao Huang, Sheng Huang, Juan Wei. "Voltage suppression strategy for multi-stage frequency regulation of DC-side energy storage batteries in doubly-fed induction	<1%

generator", Computers and Electrical Engineering, 2024
Publication

42	espace.etsmtl.ca Internet Source	<1 %
43	Submitted to mzu Student Paper	<1 %
44	www.seeedstudio.com Internet Source	<1 %
45	Atefeh Pour Shafei, J. Fernando A. Silva, J Monteiro. "Convolutional Neural Network Approach for Fault Detection and Characterization in Medium Voltage Distribution Networks", e-Prime - Advances in Electrical Engineering, Electronics and Energy, 2024 Publication	<1 %
46	Bogdan M. Wilamowski, J. David Irwin. "The Industrial Electronics Handbook - Five Volume Set", CRC Press, 2019 Publication	<1 %
47	COMPEL: The International Journal for Computation and Mathematics in Electrical and Electronic Engineering, Volume 33, Issue 1-2 (2014-03-28) Publication	<1 %

48	Submitted to Indian Institute of Technology, Bombay Student Paper	<1%
49	S. Karthikeyan, M. Akila, D. Sumathi, T. Poongodi. "Quantum Machine Learning - A Modern Approach", CRC Press, 2024 Publication	<1%
50	Submitted to University of Reading Student Paper	<1%
51	archives.njit.edu Internet Source	<1%
52	dokumen.pub Internet Source	<1%
53	techcritix.com Internet Source	<1%
54	www.fastercapital.com Internet Source	<1%
55	Florin Iov, Frede Blaabjerg, Anca-Daniela Hansen. "Analysis of a variable-speed wind energy conversion scheme with doubly-fed induction generator", International Journal of Electronics, 2003 Publication	<1%
56	Ion Boldea. "The Electric Generators Handbook - 2 Volume Set", CRC Press, 2019 Publication	<1%

57	J.G. Slootweg, W.L. Kling. "Is the answer blowing in the wind?", IEEE Power and Energy Magazine, 2003 Publication	<1 %
58	D. Xiang, L. Ran, P.J. Tavner, S. Yang. "Control of a Doubly Fed Induction Generator in a Wind Turbine During Grid Fault Ride-Through", IEEE Transactions on Energy Conversion, 2006 Publication	<1 %

Exclude quotes	On	Exclude matches	Off
Exclude bibliography	On		