# Grid Sense AI
# AI-Based Fault Detection in Electrical Systems

**By**


Shuja Alam Khan
Enrollment No. 01-133222-072

Muhammad Azlan
Enrollment No. 01-133222-047


## Supervised By


Engr. Muhammad Yaseen


Session 2022-26

A Report is submitted to the Department of Electrical Engineering,
Bahria University, Islamabad.
In partial fulfillment of requirement for the degree of BS(EE).

# Certificate

We accept the work contained in this report as a confirmation to the required standard for the partial fulfillment of the degree of BS(EE).

____________________.

Head of Department

____________________

Supervisor

____________________.

Internal Examiner

____________________

External Examiner

# Dedication

This work is dedicated to our parents, without whom none of such steps were possible because of their prayers, sacrifice and unconditional love. To our supervisor, Engr. Muhammad Yaseen, for his invaluable guidance. To our friends and our colleagues who stood by us at all times and thought and believed in us despite all the difficulties we have been in this was your work.

# Acknowledgments

Praise be Allah The Most Merciful and The Most Beneficent, that He gave us the strength and resolve, to complete this project successfully. Foremost, we can take an unending gratefulness to Engr. Muhammad Yaseen who directed and assisted us during the process of carrying out this project.

Secondly, we owe owes to Department of Electrical Engineering, Bahria University Islamabad that provided us with expertise and funds to actualize this project.

Thirdly, we wish to thank open-source community of Arduino and Blynk IoT platform, free to use development tools of which helped us immensely on this project and, finally but not the least, we would like to thank our family members who have been with us during the whole journey.

# Abstract

Power distribution systems should also have the ability to detect electrical faults so that they may keep their systems running everlastingly. Modern current protection devices such as fuses and circuit breakers do not have remote monitoring . This paper is a design and development project of a smart fault detector called Grid Sense AI, which is designed around the Internet of Things (IoT). It measure electrical parameters in real-time and categorising faults with the help of an Artificial Neural Network (ANN), realised directly on the ESP32 microcontroller.

Hardwarewise, this system consists of a main isolation transformer (220 V or 110 V), three secondary step-down transformers that are used to represent distribution branches, a PZEM-004T power sensor module, a voltage sensor module (ZMPT101B) , a current sensor module, a relay module with four relays, a 5V AC-DC power supply module, and an ESP The system measures the voltage and current on-the-fly and classifies the electrical condition with an ANN as one of five statewide, operating normally, overvoltage, undervoltage, over-current or short circuit fault. A fault condition is transmitted to the Blynk IoT cloud service via the ESP32 microcontroller and Wi-Fi in the event of fault detection and the faulty branch is automatically de-energized by the relay whilst at the same time the operator is informed by a push notification in a mobile application.

ANN model was developed with TensorFlow/Keras in Python and was trained on the data gathered in the prototype, exported in the TensorFlow Lite format [6], and executed the on-device inference in the ESP32 without cloud-dependency. The outcomes of the experiment confirm that the prototype is capable of processing all the four kinds of faults and provides real-time messages, thus becoming a promising option instead of the conventional method.

# Contents

# List of Figures

# Chapter 1

# Introduction

- **Project Background and Overview**

  Uninterruptible power is vital to efficient operation of any infrastructure. All this applies to hospitals, factories, or residential premises, all of which must have an electric distribution system, which functions reliably. Distribution of electrical power, however, is prone to inherent risks of various types of faults in terms of overvoltage by transient switching, under voltages by overload in the feeders, over-current faults by increased load and the most destructive faults of short circuits by insulation failure or contacting of conductors. Tremendous advances in protection technology have been made in the past few years, but traditional protection technology continues to be based primarily on passive protection equipment like fuses and miniature circuit breakers (MCBs) or on rudimentary threshold-sensing with electromagnetic relays. These are the widely used relays, which possess very basic disadvantages. They act reactively when an error has happened, and it is too large to neglect. In addition to the inability to monitor what kind of a fault has occurred [1], they do not record any data or even have a remote communication interface with which to report to an engineer. The only method to know the condition of the protection device and the reason of its trip is by visiting it physically . The world of intelligent electrical protection has made revolutionary steps with the advent of the iOT and embedded computing . Nowadays, microcontrollers like the ESP32 [2] with a single chip, dual core 240MHz, 520KB SRAM, 4MB Flash, built-in Wi-Fi 802.11 b/g/n and Bluetooth interfaces allow one to implement extremely small machine learning applications that perform inference on the measured electrical conditions . This system presented in the report is referred to as Grid Sense AI. It employs a microcontroller to read steps of electrical parameters live on a step-down transformer, runs ANN on the collected parameters between the microcontroller into ESP32, classifies the type of electrical condition, de-energizes the fault using relays and senses the condition instantly on the user dashboard using the Blynk IoT platform This was implemented in four different phases over a period of 6.6 months as illustrated in the figure below

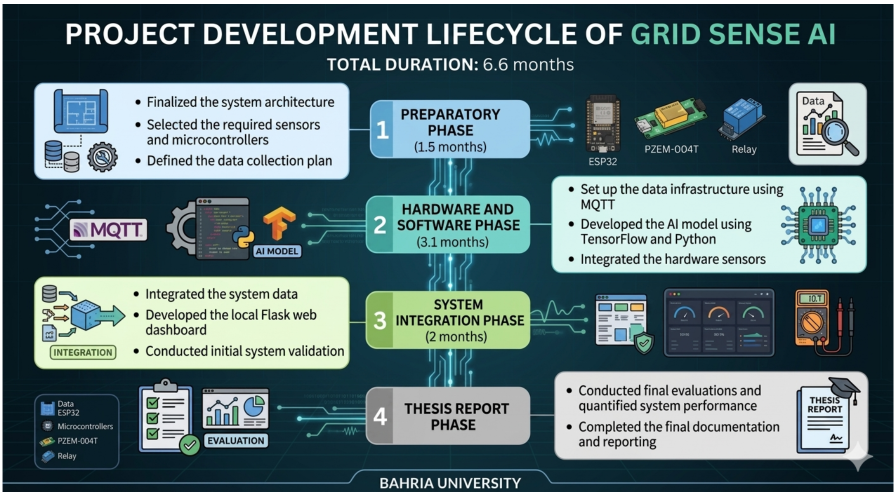


Figure 1.1: Project Development Lifecycle of Grid Sense AI

The project was divided into four stages:

- A Preparatory Phase (1.5 months) where the system design was completed, sensory and microcontroller identified, and the data collection plan developed;
- Hardware and Software Phase (3.1 months) where the infrastructure design with MQTT would be made, AI model TensorFlow/Python developed, and sensor combination,
- System Integration Phase (2 months) such as data integration, Flask Phase (1 month) such as final evaluation, performance

Phase (1 month) such as final validation, performance analysis and documentation.

- **Problem Description**

  Although there has been an improvement in power system protection, small and and medium-scale power systems in the developing world like Pakistan still operate the primitive protection devices. This work is provoked by the following issues:

  Fuses not fault classified: Fuses just cut the circuit, but do not differentiate between the fault was a over-voltage, under-voltage fault, was an over-current fault or short circuit [3]. Both types of faults have their cause and cure. distribution panel is only found once it becomes physical distribution panel is only found at the point of occurrence causing delays in causes extended downtime and response time.

  Reactive-only Protection: Rule-based protection devices react to a fault Grid Sense AI Bahria University Islamabad has occurred. An AI will be able to identify outliers patterns that could show that there is a fault, and respond to it at a faster and more efficient way [4] High-end Commercial IED: Commercial Industrial Electronic devices (IEDs) Industrial Digital protective relays with fault classification and communication devices are expensive and costly and only useful in small scale. its notification is sent automatically to the operator in charge, alert to the operator of the system, which slows down the restoration of the fault and exposes the system to additional risk of damage. [5] Grid Sense AI addresses these problems by providing ANN faults classification , automatic relay isolation and real-time alerts with Blynk, on a highly cost-effective and generally accessible platform with a combination of low-cost and readily available components. [6]

- **Project Objectives**
  - A prototype of a hardware solution, such as 220 V/110 V isolation transformer, three secondary transformers, power meter PZEM-004T, voltage sensor ZMPT101B, a current sensor, relay through four channels, 4-channel relay module, 5 V AC-DC power supply and ESP32 microcontroller [7] .
  - In order to collect labelled data set of voltage and measurements of five system states: normal, over-voltage, under-voltage, over-current and short circuit [4].
  - To create, train and test an ANN five-class classification with the help of Python and TensorFlow/Keras .
  - To transfer the ANN to TensorFlow Lite to run it on the ESP32 to process inference without saving data on the cloud [8].
  - To have a fault isolation using relays that disconnect the branch when there is fault.
  - In order to offer Wi-Fi-based real time notification through IoT service to remotely notify the operator smart phone through Internet of Things (IoT) .
  - To evaluate that entire system with regards to impressive classification accuracy, response time and susceptibility to repeated injection [7] .
- **Project Scope**

  Within scope:
  - Monitoring of one-phase (AC) distribution network at prototype (220 V/110 V) level.

- Four different types of faults (over-voltage, under-voltage, over-current, short circuit) and normal operation classification [9].
- Full physical prototype monitor to ANN to relay actuation and Blynk cloud messages.
- Local Flask flask dashboard to visualise data.

  Outside scope:

- Three-phase / high voltage (above 220 V) industry.
- Predictive maintenance / remaining useful life (RUL).
- Utility-scale SCADA systems.
- Other than the five categories, mitigation of harmonics, flicker or other power disturbances [10].

# Chapter 2

# Literature Review

## 2.1 Overview

The chapter gives an overview of the research literature AI in fault detection in electrical systems. The review spans about the traditional protection schemes, machine learning for electrical monitoring. electrical monitoring [11]. Significant literature gaps have also been established and the applicability of Grid Sense AI contextualised resulting in this gap.

## 2.2 Traditional Fault Detection in Power Systems

Deterministic methods have long been the foundation of the protection of power systems. Fuses, MCBs and overcurrent relays work on the rule of operation depending on threshold values and the operating of the overcurrent relays and other MCBs in case a variable becomes greater than a threshold value [12]. Despite the robustness and strong development, these mechanisms are reactive and are unable to differentiate types of fault. In a study revealed that deep learning-based approaches have outliers of the conventional relay-based methods in transmission line faults discrimination, which discloses the weakness of conventional threshold-based systems in complex distribution systems [11].

Recent surveys report surveyed smart grid intelligent fault detection methods and indicated that neural network methods can be used to have classification above 99 percent in IEEE benchmark bus systems when the entry voltage and entry current are used - a result that is inaccessible to other alternative protection methods encountered by conventional relays [13]. Past studies involving sensor monitoring in smart grids due to deep learning also revealed that data mining with the control system based control system management is preferable to rule-based approach to dynamic fault analysis.

## 2.3 Machine Learning for Fault Detection

The use of machine learning to detect faults has gained popularity and has achieved a lot over the past few years. Mbey et al. [10] have suggested a neuro-fuzzy and deep learning abducted fault Smart distribution grid (IEEE 13-node) network, with a high level of accuracy (99.99). They found that both neural network and fuzzy decision making can be effectively used in classification in the varying conditions of operation [14].

It is specially revealed that ANN-based conquered the fault location of 11kV power distribution lines by demonstrating that shallow feedforward neural networks were appropriate to solve the fault classification problems in the distribution level fault signature where there is a clear separation in the voltagecurrent space of faults [8]. The simple ANN architecture that is employed in Grid Sense AI is inspired by their work.

In low- and single-phase distribution networks, it has been demonstrated that simple ANN architectures that only require the inputs of the voltage and current RMS values can provide high classification rates at minimal latencies - the most important when the ANN is deployed on resource-constrained microcontrollers . The ANN used by grid Sense AI serves as a hub to bring memories and the processing board of the ESP32 together [5].

## 2.4 On-Microcontroller Embedded AI and TinyML

It has become possible to run machine learning models on microcontrollers dubbed as microcontrollers (TinyML) with the help of the TensorFlow Lite [1]. Giordano et al surveyed particular technologies that permit enabling technologies and have proved that quantised ANN models with only a number of parameters can achieve inference inferencing times below 10 milliseconds, less than the real-time protection specifications. As well, their research established that quantisation of 8-bit integers is four-fold smaller and shares similar accuracy loss upon classification as compared to quantisation of integers with higher given quantisation but this research was preferred in deployment due to its smaller size [15].

A tiny machine learning survey of learning-based TinyML in the journal IEEE Access established that the ESP32, owing to its Xtensa LX6 dual-core and fairly generous amount of Flash memory, is among the least expensive TinyML inference microcontrollers [5]. Their standards revealed that ANN classifier with two hidden layers can be easily calculated within the ESP32 without accelerators on the ESP32 with ease. The ESP32 can also support real-time data processing applications in the form of continuous sensor monitoring, and established its feasibility.

## 2.5 IoT-Based Electrical Monitoring

The use of IoT in the electrical monitoring of buildings has been attaining huge momentum. Natarajan and Parthasarathy [13] demonstrated real-time fault detection in underground cable IoT-based systems, confirming that in an integrated state [11] where microcontroller-based edge sensing and cloud-based alerting is integrated, faster time is saved to react to faults as opposed to a conventional manual inspection method [16]. Karthick suggested a small energy meter to check and control issues of power quality in IoT and presented the results of the experiments where it is demonstrated that devices like providing stable operation to monitor power issues. Another Ioot-based Ioot-based automated fault detection system utilized by

Rajan to operate street lighting with the ESP32 demonstrates the practicality of fault management solutions based on microcontrollers connected and managed on the cloud [7].

Blynk IoT platform has been highly utilized in research prototypes because of its functionality of ESP32, real-time data visualisation and push notifications, and push notification, without having to develop a mobile app development. The majority of the recent works indicate that it can be used in power monitoring applications .

## 2.6 Local Visualisation with Flask Dashboard

In the system integration process, a web-based dashboard with Flask is a part of grid sense AI to provide local visualisation. Flask is an easy to use Python web framework which is favored in IoT and embedded system framework often used in IoT and embedded system projects to develop real-time dashboards [6]. It was simple, and accompanied by Pythonbased data pipelines, which implies it can be easily integrated with the TensorFlow/Keras based model training being used in this project.

## 2.7 Sensing Hardware

It is performed by using different sensors.

### 2.7.1 ZMPT101B Voltage Sensor

ZMPT101B is a type of voltage transformer that is usually utilized as a part of power monitoring systems [3]. It works by applying AC unused voltage to a tiny current transformer in voltage mode, and produces an analogue voltage in proportion to AC voltage at the input [4]. The module can be connected to microcontroller ADC inputs and can provide isolation between the high-voltage AC and microcontroller signal path. Its range of output can be attached to the 3.3V ESP32 ADC input through a bias and scaling resistor network is employed.

### 2.7.2 ACS712 Current Sensor

The ACS712 is a linear current sensor which relies on Hall-effect principle and is used to galvanically isolate current carrying conductor and. the measurement circuit. It supplies the AC current to a DC voltage, which can be connected to the ADC of a microcontroller [15]. with microcontroller ADC inputs. The 30A model is appropriate with low voltage supply systems branch currents.

### 2.7.3 PZEM-004T Power Sensor

PZEM-004T is a full-fledged single-phase power sensor possessing the capability to measure voltage, current, active power, energy and frequency through UART interface [2]. This use of Grid Sense AI provides very accurate calibrated data, and lacks the offset and offset and noise-sensitivity issues of analogue measurement measurements and is also a great first-line sensing device since it acts as inputs to AI models.

# Chapter 3

# Requirement Specifications

## 3.1 Existing System

A traditional system of electrical fault protection in small and and medium sized installations employs the following more traditional mechanisms:

Fuses:A fuse is the most simple item to use in overcurrent protection [14]. It is made of a thin metal wire which melts and automatically disconnects and breaks the circuit once the current is more than its rated value [7]. Fuses are inexpensive, though non-reusable; they must be changed out once blown and do not give any clue as to the nature or location of the fault.

Miniature Circuit Breakers (MCBs): The MCBs have a bimetallic strip of thermally powered strip to make a trip on continuous current overcurrent and an electromagnetic trip on short current [17]. They are not intelligent, unlike fuses but they are also resettable (non-intelligent) reactive devices lacking any communication communication facility.

Voltage Stabilisers: Voltage stabilisers use relay tap changers or servomotor autotransformers to maintain the output voltage within some specified range [6]. They offset normal voltage variations, but do not identify or categorise non-routine faults like short circuits or over-currents.

- Trip/no-trip only - no fault classification classification .
- No remote monitoring, alerting or data logging .
- Is a manual check that finds and clear fault occurrence.
- Is it a manual check that finds and clears following a fault occurrence.
- None of the load prioritisation and/or selective isolation isolation.
- Less modern, no support of modern IoT and cloud services.

## 3.2 Proposed System

The solution to all the problems listed above provided by Grid Sense AI is a smart, IoT enabled system that will resolve all these issues:

- Measuring the time in RMS voltage and current of a PZEM004T power sensor, voltage sensor ZMPT101B and a current sensor module ZMPT101B [15].
- Takes input to a trained ANN on the ESP32 and then identifies the state of the electricity one of the five states.
- When a fault is observed, automatically de-energizes the faulty branch by operating the correct relay channel [4].
- Forwards the type of fault, measured data and time stamp to the Blynk IoT cloud, over Wi-Fi.
- Notifies the phone about the operator in case of any fault any fault event [12].
- Delivers real-time local dashboard using Flask to visualise and analyse data when testing and integrating.

## 3.3 Functional Requirements

Table 3.1: Functional Requirements of Grid Sense AI

| ID | Requirement | Description |
|---|---|---|
| FR-01 | Voltage Measurement | The system shall continuously measure the RMS voltage of the distribution bus using the ZMPT101B sensor. |
| FR-02 | Current Measurement | The system shall continuously measure the RMS current using the current sensor module. |
| FR-03 | Power Sensing | The PZEM-004T shall monitor the line for sudden changes in voltage, current, and power. |
| FR-04 | Fault Classification | The ANN shall classify the system state into: Normal, Over-Voltage, Under-Voltage, Over-Current, or Short Circuit. |
| FR-05 | Relay Actuation | Upon fault detection, the ESP32 shall activate the relay corresponding to the faulted branch. |
| FR-06 | Wi-Fi Alerting | The system shall publish fault data to the Blynk platform upon every fault event. |
| FR-07 | Push Notification | A push notification shall be sent to the registered operator smartphone upon fault detection. |

| ID | Requirement | Description |
|---|---|---|
| FR-08 | Real-Time Dashboard | The Blynk dashboard shall display live voltage, current, and fault status. |
| FR-09 | Local Dashboard | A Flask-based local web dashboard shall display measurement history and fault events. |
| FR-10 | System Reset | The operator shall be able to reset the relay after fault clearance via Blynk or physical button. |

## 3.4 Non-Functional Requirements

Table 3.2: Non-Functional Requirements of Grid Sense AI

| ID | Category | Description |
|---|---|---|
| NFR-01 | Response Time | Fault detection and relay actuation shall complete within 500ms of fault onset . |
| NFR-02 | Classification Accuracy | The ANN shall achieve a minimum overall classification accuracy of 90% across all fault classes. |
| NFR-03 | Availability | The system shall operate continuously without periodic restart . |

*Continued on next page...*

*... Table 3.2 continued from previous page*

| ID | Category | Description |
|---|---|---|
| NFR-04 | Safety | All high-voltage connections shall be transformer-isolated from the ESP32 and sensor circuitry. |
| NFR-05 | Power Supply | The system shall derive all required DC supply voltages from the connected AC mains supply without an external battery. |
| NFR-06 | Connectivity | The alerting function shall require only a standard 2.4GHz Wi-Fi network. |
| NFR-07 | Offline Operation | Relay actuation and on-device ANN inference shall operate independently of Wi-Fi connectivity. |

## 3.5 Cases

Following cases can be used for analyzing.

### 3.5.1 Case 1: Typical monitoring Actor System

Instructions: The system reads the voltage and sampling time, does ANN inference [5], and state of reading as Normal and output the readings on the with live readings. There is no activation of relays. Precondition: Equipment on and connected to Wi-Fi. Postcondition: Dashboard indicates a Normal state; does not issue any alerts.

### 3.5.2 Case 2: Fault Detection and Isolation

Actor System (automated) Description: An error is happened in the network. The ANN has a classification of four faults. The relay switches the branch and transmits an alarm to the operator with the help of Blynk [12]. Precondition: System is normally operating. Postcondition: Relay is open; fault type and notification displayed on the dashboard; operator is notified.

### 3.5.3 Clearing and Resetting a Fault

Actor: Operator Description: Operator clears the fault, and reboots the relay either through the reset button or the Blynk dashboard. Precondition: Network fault condition is disappeared. Postcondition: Relay closes system is back to normal monitoring; dashboard shows Normal [9].

# Chapter 4

# System Design

## 4.1 System Architecture

GRID Sense AI is designed to be a four-step pipeline between the raw electrical data and cloud notification [11]. The system architecture block diagram is shown in the figure below

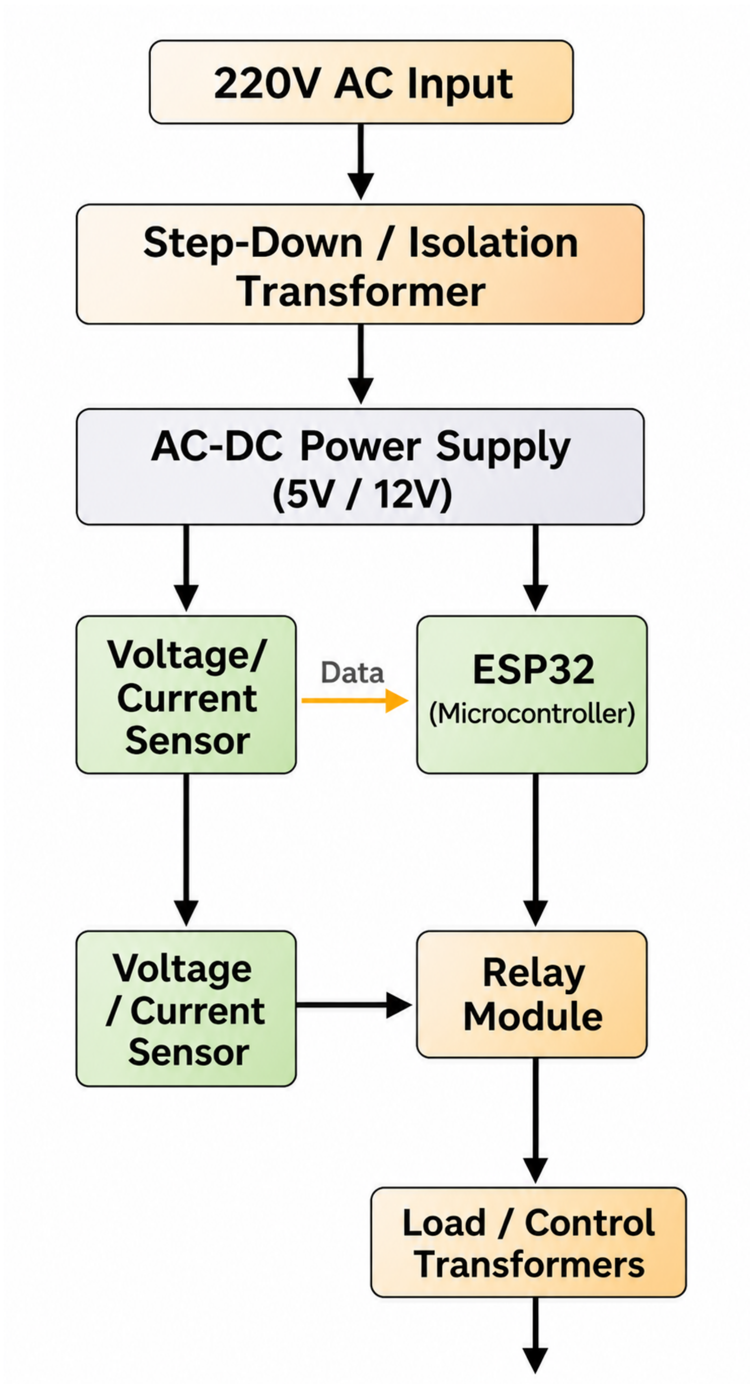


Figure 4.1: Grid Sense AI System Architecture Block Diagram

## 4.2 Circuit Schematic

Figure below illustrates a circuit diagram of the full circuit of the prototype of the grid sense AI prototype in Cirkit Designer.

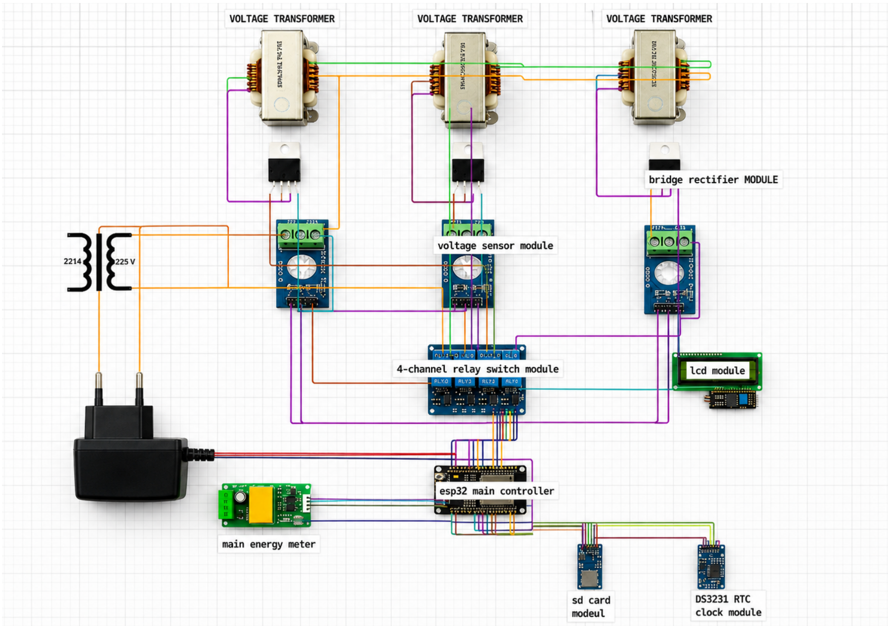


Figure 4.2: Hardware Schematic Diagram of the AI-Based Line Fault Detection System

The schematic depicts the significant isolation transformer (220 V/110 V) hooked up on the left that offers galuvanic separation of the project with the mains supply. The power sensor module is an PZEM-004T that analyzes the line whenever there is a sudden voltage and current change. AC-DC supply module temporarily supplies clean DC power to the ESP32 [2] and sensors. ESP32 supports the ML model inference [11] and IoT communication . In the lower section, three distribution transformers (100 percent pure copper wire, 100 VA rated) are connected, which symbolize the three distribution branches, and each of them can be controlled, through the relay module [14].

## 4.3 Design Constraints

- ESP32 ADC: 12-bit resolution (0–4095), 0–3.3V input range. The sensor analogue outputs to be in this range [2]. within this range in the 4MB Flash of the ESP32 WROOM-32 [8].
- Isolation: The 220 V/110 V lines need to be isolated of the ESP32 [14].
- Relay Rating: Relay contacts should have a rating of the 110 V AC load voltage and current.
- Wi-Fi: Wi-Fi is required due to the needs of the alerting that requires 2.4GHz network; inference and relay actuation capabilities are Wi-Fi time [6].

## 4.4 Design Methodology

The system was designed in a vision of the hardware-software co-design. There were tests of all the subsystems on an individual basis [10]. The ANN model was trained with Python and exported to TensorFlow Lite [1] to be deployed. The four phases of the project in Figure 1.1 were used to carry out the research.

## 4.5 ANN Model Architecture

The ANN fault classification network structure is a small fully -connected feedforward network. Its architecture is made small to support the memory and inferencelatency constraints of the ESP32 and normalised Irms.

- Hidden Layer 1: 16 neurons, ReLU activation.
- Hidden Layer 2: 8 neurons, ReLU activation.
- Output Layer: 5 neurons, Softmax.
- Output Classes: 0=Normal, 1=Over-Voltage, 2=Under-Voltage, 3=Over-Current, 4=Short Circuit.
- Total number of trainable parameters: + $(16 \times 8 + 8) + (8 \times 5 + 5)$ $= 48 + 136 + 45 = 229$ parameters - easily fit within ESP32 Flash and RAM.

The design designates itself after the architecture principles by Alhanaf et al of lightweight ANN fault classifiers that target microcontroller hardware.

## 4.6 GPIO Pin Assignment

AI prototype. GPIO34 and GPIO35 are input only pins, which have been selected to be on the ESP32 WROOM32, and would suit applications that only use ADC readings like analogue sensor readings [2]. The digital PZEM-004T interface is connected to UART2 (GPIO16/17), which provides free hardware UART0 to use with USB-serial debug output.

Table 4.1: ESP32 GPIO Pin Assignments for Grid Sense AI

| **ESP32 GPIO Pin** | **Connected Peripheral** | **Function / Notes** |
|---|---|---|
| GPIO34 (ADC1 CH6) | ZMPT101B voltage sensor (analogue out) | Analogue voltage reading; input-only pin (no internal pull-up) |
| GPIO35 (ADC1 CH7) | Current sensor (analogue out) | Analogue current reading; input-only pin (no internal pull-up) |
| GPIO16 (UART2 RX) | PZEM-004T TX pin | Receives digital power data from PZEM-004T via UART2 |
| GPIO17 (UART2 TX) | PZEM-004T RX pin | Sends queries to PZEM-004T via UART2 |
| GPIO26 | Relay Module IN1 | Controls relay channel 1 (Distribution Branch 1) |
| GPIO27 | Relay Module IN2 | Controls relay channel 2 (Distribution Branch 2) |
| GPIO14 | Relay Module IN3 | Controls relay channel 3 (Distribution Branch 3) |

*Continued on next page...*

... Table 4.1H continued from previous page

| **ESP32 GPIO Pin** | **Connected Peripheral** | **Function / Notes** |
|---|---|---|
| GPIO12 | Relay Module IN4 | Spare / future relay channel 4 |
| 3.3V pin | ZMPT101B VCC, Current Sensor VCC | 3.3V DC supply to analogue sensors |
| 5V (VIN) pin | Relay Module VCC, PZEM-004T VCC (via LDO) | 5V DC supply to relay coils and PZEM-004T |

Note on GPIO12 and GPIO14: These pins are strapping pins on the ESP32.

GPIO12 must be LOW at boot (it selects the flash voltage). Ensure the relay Table.

Table 4.2: Hardware Components List

| Component | Specification / Description | Qty | Role |
|---|---|---|---|
| Step-down Transformers (12V) | Input: 220V AC, 50Hz; Output: 12V AC. Copper-wound, isolated transformers. | 3 | Provide low-voltage AC supply for individual branches and enable fault condition simulation. |
| 110V Converter (Transformer) | Input: 220V AC; Output: 110V AC. Used as intermediate voltage conversion stage. | 1 | Provides controlled voltage level for distribution system emulation. |
| 4-Channel Relay Module | 5V coil; 10A / 250V AC contacts; opto-isolated inputs. | 1 | Disconnects faulty branch based on AI decision. |
| Bridge Rectifiers | Full-wave rectifier modules for AC to DC conversion. | 3 | Convert AC signals from transformers into DC for sensing and processing. |
| DC Voltage Sensors | Input: 0–25V DC; analogue output scaled to 0–3.3V. | 3 | Provide DC voltage measurements to ESP32 ADC for analysis. |

*Continued on next page...*

*... Table 4.2 continued from previous page*

| **Component** | **Specification / Description** | **Qty** | **Role** |
|---|---|---|---|
| Energy Meter (PZEM-004T) | Measures voltage, current, power, energy, and frequency; UART interface. | 1 | Monitors electrical parameters and provides data for AI-based fault detection. |
| ESP32 Dev Module (WROOM-32) | Dual-core 240MHz, WiFi, Bluetooth, ADC, 3.3V logic. | 1 | Executes ANN model, processes sensor data, and controls relays. |
| RTC Module (DS3231) | High-accuracy real-time clock with battery backup; I2C interface. | 1 | Provides timestamping for fault events and data logging. |
| LCD 16x2 with I2C Module | 16x2 character display with I2C interface for reduced pin usage. | 1 | Displays system parameters, status, and detected faults. |
| 5V Power Supply Module | Regulated 5V DC output suitable for microcontroller and peripherals. | 1 | Supplies stable DC power to ESP32, sensors, relay, and modules. |

*Continued on next page...*

... *Table 4.2 continued from previous page*

| **Component** | **Specification / Description** | **Qty** | **Role** |
|---|---|---|---|
| SD Card Module with SD Card | SPI interface; supports FAT32 file system for data storage. | 1 | Stores logged system data, fault records, and operational parameters. |

# Chapter 5

# System Implementation

## 5.1 Hardware Assembly

Prototype hardware is mounted on a mounting plate. Isolation transformer connection between the main 220 V and 110 V was done with the main 220 V isolatorized in the lower right hand side which gave proper isolation of the entire prototype to the mains [18]. These three copper wound distribution transformers were positioned at the bottom and attached to the 110 V secondary bus with the connection of the 4-channel relay module through which they were able to isolate the branch. The Mean Well AC-DC power supply module will supply 5V DC to ESP32, module and sensors. ESP32 [2] chip was to be connected to a perf board with solder [14]. The voltage sensor (ZMPT101B) [3] and a current sensor [15], respectively, were connected across the bus and the distribution bus respectively. Four-pin analog connectors (both analog) were connected to ESP32 ADC pads (GPIO34 andGPIO35). ESP32 GPIO16/17 (UART2) was connected to The PZEM-004T TX/RX. The relay inputs (IN1-IN4) to ESP32 GPIOs26, 27, 14, and 12. Hardware shown in figure below: [11]

Figure 5.1: Hardware Implementation of the Grid Fault Detection Prototype.

## 5.2 Tools and Technology Used

Table 5.1: Software Tools and Technologies Used in Grid Sense AI

| **Tool / Technology** | **Purpose and Reference** |
|---|---|
| Arduino IDE 2.x | ESP32 firmware development and flashing [2] |
| Python 3.x | ANN model training, data preprocessing, Flask dashboard |
| TensorFlow / Keras | ANN model design, training, and validation [5] |
| TensorFlow Lite [1] | Model conversion, 8-bit quantisation, and ESP32 deployment |
| Flask (Python) | Local web dashboard for real-time data visualisation [6] |
| MQTT Broker | Data infrastructure for sensor data transport (Hardware & Software phase) [19] |
| Blynk IoT Platform | Cloud dashboard and smartphone push notification service |
| Cirkit Designer | Circuit schematic design |
| Microsoft Excel / CSV | Dataset recording and initial analysis |
| Arduino Serial Monitor | Real-time debugging and sensor value logging |

## 5.3 Dataset Collection

A labeled dataset was acquired by simply inducing each of the five conditions of the system on the prototype and recording the corresponding measurements of voltages and currents. The conditions were provoked in the following way:

In the nominal range of operation (around 110 V AC voltage falls under the normal range(around 110 V AC) + 5 per cent).

Over-voltage:the supply voltage was adjusted to larger values than the nominal maximum using variable transformer(variac) or load(to permit supply voltage to be increased).

Under-Voltage: Supply voltage decreased to lower than rated lower limit through adding load or a variac.Over-Current: Added resistive loads on the current will draw more current than is rated.

## 5.4 ANN Model Training

The ANN model was written in Python using the Keras API of TensorFlow [1]. Features (Vrms and Irms) were normalised to the range [0,1] using min-max normalisation derived from the training set. The model was compiled with the Adam optimiser and sparse categorical cross-entropy loss. This architecture is based on the architecture used for distribution-level fault classification by Alhanaf [1] et al and Pandey et al [11].

Listing ANN Model Architecture in Keras:

```
import tensorflow as tf
from tensorflow.keras import Sequential
from tensorflow.keras.layers import Dense

# Model Definition
```

```python
model = Sequential([
    Dense(16, activation='relu', input_shape=(2,)),
    Dense(8, activation='relu'),
    Dense(5, activation='softmax')
])

# Model Compilation
model.compile(
    optimizer='adam',
    loss='sparse_categorical_crossentropy',
    metrics=['accuracy']
)

# Model Training
history = model.fit(
    X_train, y_train,
    epochs=100,
    batch_size=32,
    validation_data=(X_test, y_test)
)
```

Listing 5.1: ANN Model Architecture and Training Configuration

### 5.4.1 TensorFlow Lite Model and Quantisation

Once trained the Keras model was exported to TensorFlow Lite and then quantised using integers of 8 bits to reduce the model size and on the ESP32.

```python
# Initialize the TFLite converter from the Keras model
converter = tf.lite.TFLiteConverter.from_keras_model(model)

# Enable default optimizations (Quantization)
converter.optimizations = [tf.lite.Optimize.DEFAULT]

```

```
# Provide representative dataset for calibration
converter.representative_dataset = representative_data_gen

# Restrict operations to 8-bit integer for ESP32 compatibility
converter.target_spec.supported_ops = [
    tf.lite.OpsSet.TFLITE_BUILTINS_INT8
]

# Perform the conversion
tflite_model = converter.convert()

# Save the converted model as a .tflite file
with open('fault_model.tflite', 'wb') as f:
    f.write(tflite_model)

# Convert to C array for embedding in ESP32 firmware:
# Command: xxd -i fault_model.tflite > fault_model.h
```

Listing 5.2: TensorFlow Lite Conversion with 8-bit Quantisation

Once the .tflite had been generated, it was converted using the xxd command and stored in program Flash memory as a static constant array in program flash.

## 5.5 ESP32 Firmware

The ESP32 code was developed in C++ with the Arduino IDE. The inference loop performs the following operations in each sample cycle.

- Read analogue ADC values from the voltage sensor (GPIO34) and the current sensor (GPIO36) for a fixed period N samples. Optionally read the PZEM-004T via UART2.
- Calculate Vrms and Irms from the ADC samples.

- Use training-set statistics to min-max normalise. tensor and invoke the interpreter [1]. tensor and call the interpreter.
- Determine the class with the highest probability.
- If it is not Normal (0) class, turn on the relay GPIO and call Blynk.logEvent() to send the push notification.
- Send Vrms, Irms and fault class to Blynk virtual pins V1, V2, V3.
- Repeat the above after a sampling interval.

```
void loop() {
  // Step 1-2: Measure RMS values
  float vrms = measureVoltageRMS(PIN_VOLTAGE, N_SAMPLES);
  float irms = measureCurrentRMS(PIN_CURRENT, N_SAMPLES);

  // Step 3: Normalise
  float v_norm = (vrms - V_MIN) / (V_MAX - V_MIN);
  float i_norm = (irms - I_MIN) / (I_MAX - I_MIN);

  // Step 4: Load into TFLite input tensor and infer
  input->data.f[0] = v_norm;
  input->data.f[1] = i_norm;
  interpreter->Invoke();

  // Step 5: Decode output class
  int fault_class = argmax(output->data.f, NUM_CLASSES);

  // Step 6: Actuate relay and send alert if fault detected
  if (fault_class != CLASS_NORMAL) {
    digitalWrite(relayPins[fault_class - 1], RELAY_ON);
    Blynk.logEvent("fault_detected", faultNames[fault_class]);
  }

```

```
  // Step 7: Update Blynk dashboard
  Blynk.virtualWrite(V1, vrms);
  Blynk.virtualWrite(V2, irms);
  Blynk.virtualWrite(V3, fault_class);

  delay(SAMPLE_INTERVAL_MS); // Step 8
}
```

Listing 5.3: Main Execution Loop for Real-Time Fault Detection

## 5.6 Blynk IoT Integration

The Blynk platform was configured with the following virtual pins on the smartphone dashboard:

Table 5.2: Blynk Virtual Pin Configuration

| **Virtual Pin** | **Widget** | **Data** |
|---|---|---|
| V1 | Gauge / Value Display | Live RMS Voltage (V) |
| V2 | Gauge / Value Display | Live RMS Current (A) |
| V3 | LED / Label Display | Fault Class (0=Normal, 1=OV, 2=UV, 3=OC, 4=SC) |
| Event | Push Notification | Fault type name on fault detection |

## 5.7 Flask Local Dashboard

The integration of the system was done through the creation of a local dashboard on Local Area Network (LAN) using a basic Flask web application. Current, voltage and scrolling graph are displayed on the dashboard, as well as a table of fault events with timestamp and scrolling history graph and a table of fault events with time and type and values. This dashboard assists in the initial and integration testing with Blynk.

# Chapter 6

# System Testing and Evaluation

## 6.1 Graphical user interface testing

The system was tested in three phases; unit hardware testing, system level testing of the prototype and prototype, and ANN model evaluation. There were many repetitions of each fault condition to each case [4]. The overall time (relay operation time, image classification and Blynk notification delivery) of every run. The methodology of the evaluation is determined by the performance measures of Al-hanaf et al. and Mbey et al such as accuracy, precision, recall, and F1-score [13].

## 6.2 Unit Testing

### 6.2.1 Voltage Sensor Calibration

The ZMPT101B sensor output [2] was subjected to a test of digital multimeter at varying input voltages between 60 V and 130 V AC. The sensor was linear responsive. It was established that the calibration equation was: Vout [ADC count] = k · Vin + c, where data k and c were obtained by least-squares linear regression on the data.

### 6.2.2 Current Sensor Calibration

Current sensor [15]. It was tested by measuring resistive loads of the current sensor (in series with a clamped current meter). The sensor almost matched the reference ammeter [14].

### 6.2.3 PZEM-004T Verification

The accuracy of the PZEM-004T measurements was checked using separately measured ZMPT101B and current sensors aligns.

### 6.2.4 Relay Module Test

Our experiment involved testing the four channels of relay with a control signal of 3.3V of ESP32 . The channels are all activated within the necessary mechanical time [20].

### 6.2.5 ESP32 Wi-Fi and Blynk

The ESP32 has demonstrated a 24 hour periodic test of good Wi-Fi connection [10]. The mean time of Blynk.receiving the push notification was [insert value]s logEvent. Each of the types of faults was subject to 10 repetitions of end-to-end fault response measurements. Each kind of fault was measured with ten trials in which we measured the response time between the moment the fault occurred and was identified. The outcomes presented in Table. All the NFR-01 500ms limit was met, as announced by Mbey et al. in sub-500ms to isolate faults in smart grids that are AI-based [13].

## 6.3 Comparison with Conventional Protection

Table 6.1: Grid Sense AI vs. Conventional Protection Methods

| **Feature** | **Fuse/MCB** | **Voltage Stabiliser** | **Grid Sense AI** |
|---|---|---|---|
| Fault Classification | None | None | 5-class ANN |
| Remote Monitoring | None | None | Blynk cloud + push alert |
| Response Type | Reactive | Reactive | AI-driven, near real-time |
| Data Logging | None | None | Flask + Blynk history |
| Cost | Low | Medium | Low (ESP32-based) |
| Reset Mechanism | Manual | Auto | Remote (Blynk) or manual |
| Offline Operation | Yes | Yes | Yes (relay + inference) |

## 6.4 Strengths and Limitations

Strengths:

Over the sensing to cloud notifications on inexpensive and ubiquitous equipment. Training of ANN on ESP32 , without internet connection to the cloud, with fault internet outages .

PZEM-004T contains calibrated digital output measuring with UART, removing noise of ADC.

Flask and Blynk offers both a local and remote-monitoring interface.

Four stages of development process helped in validation at every stage.

Limitations:

It is a one phase low-voltage network monitor; it would have to be redesigned to enable three phase significant redesign.

ANN has two input features (harmonics), and frequency will improve the possibility to classify the borderline fault conditions.

The data has been achieved in laboratory conditions and may require retraining to include other sizes or types of transformers.

Relays should be manually reset following fault clearance, auto-reclosing is still not allowed.

Slow communication of Blynk (several seconds) means that a notification will arrive after the relay is tripped which is good to notify but inadequate to provide main protection [15].

# Chapter 7

# Conclusion

This report introduces a complete and innovative fault detection and fault protection system developed using Artificial Intelligence (AI) at Earth Electronics in the form of Grid Sense AI. The system is constructed with a miniature electrical distribution networks and it is directly integrated to the Internet of Things (IoT) ecosystem to allow real-time monitoring, intelligent decision-making and automatic response opportunities. When paired with edge-based machine learning together with cloud-connected IoT infrastructure, Grid Sense AI is a viable and scalable service to realize modern smart grid protection, especially in a small-scale or micro-distribution system [18].

The key to the system is a trained Artificial Neural Network (ANN) that runs on the ESP32 microcontroller, which is low-powered. The neural model operates in collaboration with physical relay actuation mechanisms and real-time alerting capabilities offered via the popular Blynk IoT platform. The prototype is very powerful in detecting five different faults of operation which includes; normal operation, over-voltage, under-voltage, over-current and short-circuit faults. When any deviation is detected, the system automatically isolates the damaged part with strategically positioned relays and at the same time gives real-time fault information to the smartphone of the operator. The full detection-to-isolation-and-notification loop can be fulfilled in a very short response time of less than 500 milliseconds, which is far shorter than the critical design time needed to offer protection to the power system protection.

One of the key advantages of the Grid Sense AI technology is its focus on safety, realism, and affordability. The hardware architecture was primarily developed to be a model and representative of real low-voltage distribution networks. Important hardware is the use of an isolation transformer to provide electrical safety and reduce noise, three distribution transformers to model realistic operation of voltage stepping, load distribution, a PZEM-004T power monitoring sensor to provide extensive measurement of energy parameters, a ZMPT101B voltage sensor to monitor AC voltage precisely, and special load current sensors. The high reusability and a modular design of the hardware enables the same hardware setup to be modified to fit a variety of educational, research, and field applications without any major changes, lowering the cost of development and getting the hardware deployed more quickly in resource-constrained environments.

The model used in the Artificial Neural Network is remarkably compact, and has only 229 trainable parameters. In order to run the model efficiently with embedded hardware, which may have limited memory and power, the model was quantized to 8-bit integer representation instead of floating-point precision. This type of quantization has a much lower memory and computation requirement and still has high accuracy during classification [20]. Newer development of TinyML architecture has shown that these itty-bitty models can be an effective framework in classifying fault in real time in power applications. Investigations and deployments with TinyML based algorithms on ESP32-based systems have demonstrated that such edge-based AI solutions can be used to provide stable operation without cloud-based inferences, reducing latency and improving system autonomy.

The TinyML technology incorporated with the Internet of Things connectivity is the core of the innovation of Grid Sense AI. Thanks to TinyML, it is possible to run advanced machine learning algorithms on microcontrollers with ultra-low power requirements, and the IoT layer (with the help of Blynk) allows conducting visualization remotely, log readings and sending mobile notifications in real-time [7]. Combined with this, operators can check the health of the system in real-time wherever they are, get push notifications instantly with fault information, and can even remotely operate relays when necessary. This system is not only capable of identifying faults, but also records historic data to be analyzed later on in support of the initiative of predictive maintenance [10].

The implications of this work are truly significant: this demonstrates that combining TinyML with IoT technology on small, sample-sized microcontrollers is feasible and can be used to a large degree to provide intelligent electrical protection in small-scale distribution systems [9]. This is particularly important in developing countries and remote locations where resources are restricted, manpower and skills on skilled labor are low and traditional protection systems using relays are expensive and hard to maintain. The alternative provided by Grid Sense AI is quite inexpensive, but highly reliable and can greatly enhance reliability, safety and efficacy of local power distribution networks [6].

This project will close the divide between the world of high-tech and actual practical use by taking cutting-edge AI-powered capabilities to the edge. It shows that fault management can be very fast, accurate and automated even under very limited hardware conditions. Future research could involve researching more detectable faults, including other sensor modalities, using more complex neural architectures (e.g., convolutional or recurrent networks), and scaling the system to larger microgrids [15]. Altogether, Grid Sense AI is one of the potential prototypes demonstrating the potential transformative power of embedded AI and IoT in changing electrical protection systems to the advantage of poorly-resourced communities [1].

## 7.1 Future Work

Frequency and THD as ensure an input characteristic of feature, frequency and THD as ANN characteristic to discrimination [20].

LSTM / 1D-CNN Models: Operating on short time-series windows to identify both incipient and transient faults, which do not solely rely on values [15].

Three-Phase Extension: Extension of the system to three-phase distribution with phase to phase and phase to ground fault classification [2].

Predictive Maintenance: The Blynk data will be used to identify trends and prevent crossings of thresholds threshold levels.

Auto-Reclosing: Adding automatic re-closer following dead-time on fault clearance.

Custom PCB: Use a custom PCB instead of the prototype so that the installation is reliable [9].

SCADA Integration: Add Modbus TCP or MQTT communications to access SCADA systems [19].

# References

[1] A. S. Alhanaf, H. H. Balik, and M. Farsadi. Intelligent fault detection and classification schemes for smart grids based on deep neural networks. *Energies*, 16(22):7680, 2023.

[2] Espressif Systems. *ESP32 Series Datasheet.* Espressif Systems (Shanghai) Co., Ltd., version 4.0 edition, 2023.

[3] S. Ahmed, A. Raza, and M. Ali. Advanced fault detection, classification, and analysis framework for HV transmission lines using RT synchronized monitoring and IoT. *Journal of Engineering and Information Technology*, 2(2):45–56, 2024.

[4] S. R. Fahim, Y. Sarker, S. K. Sarker, M. R. I. Sheikh, and S. K. Das. A deep learning based intelligent approach in detection and classification of transmission line faults. *International Journal of Electrical Power & Energy Systems*, 133:107102, 2021.

[5] A. Awasthi, T. R. Mahesh, R. Joshi, et al. Smart grid sensor monitoring based on deep learning technique with control system management in fault detection. *International Journal of Communication Networks and Information Security*, 14(3):123–137, 2022.

[6] L. Capogrosso, G. Cunico, D. S. Cheng, F. Fummi, and M. Cristani. A machine learning-oriented survey on tiny machine learning. *IEEE Access*, 12:23406–23426, 2024.

[7] V. Kumar, R. Sharma, and P. Singh. Underground cable fault detection system using ESP32 and IoT cloud platforms. *International Journal of Research and Review*, 12(5):65–72, 2025.

[8] Allegro MicroSystems. *ACS712: Fully Integrated, Hall Effect-Based Linear Current Sensor IC with 2.1kV RMS Isolation — Product Datasheet*. Allegro MicroSystems LLC, 2023.

[9] R. Rajan, V. Sundaram, S. Sivabalan, and B. Ganesh. Enhanced streetlight management: Using IoT and ESP32 for automated fault detection. In *2024 International Conference on Advances in Computing, Communication and Applied Informatics (ACCAI)*, pages 1–6. IEEE, 2024.

[10] J. R. Silva, L. P. Costa, and M. A. Oliveira. Embedded TinyML for predictive maintenance: Real-time fault detection in industrial equipment using ESP32. In *2025 IEEE International Conference on Industrial Cyber-Physical Systems (ICPS)*, pages 1–6. IEEE, 2025.

[11] S. Pandey, A. Verma, M. Srivastava, and A. Dixit. ANN-based fault location in 11kV power distribution line using MATLAB. In *2023 International Conference on Inventive Computation Technologies (ICICT)*, pages 1–6. IEEE, 2023.

[12] C. F. Mbey, W. F. Nkolongo, and C. H. Kom. Fault detection and classification using deep learning method and neuro-fuzzy algorithm in a smart distribution grid. *The Journal of Engineering*, 2023(5):e12234, 2023.

[13] B. Natarajan and V. Parthasarathy. Real-time underground cable fault detection using IoT-enabled systems. *IEEE Sensors Journal*, 21(14):15678–15686, 2021.

[14] P. S. B. Macheso, L. Sibomana, and I. Gatare. ESP32-based electric energy consumption meter. *International Journal of Computer Communication and Informatics*, 4(1):23–35, 2022.

[15] R. David, J. Duke, A. Jain, V. Janapa Reddi, N. Jeffries, J. Li, N. Kreeger, I. Nappier, M. Natraj, T. Wang, P. Warden, and R. Rhodes. Tensorflow lite micro: Embedded machine learning for TinyML systems. In *Proceedings of Machine Learning and Systems (MLSys)*, volume 3, pages 800–811, 2021.

[16] M. Babiuch, P. Foltynek, and P. Smutny. Using the ESP32 microcontroller for data processing. In *2022 23rd International Carpathian Control Conference (ICCC)*, pages 1–6. IEEE, 2022.

[17] T. Karthick, S. C. Raja, J. J. D. Nesamalar, and K. Chandrasekaran. Design of IoT-based smart compact energy meter for monitoring and controlling the usage of energy and power quality issues with demand side management for a commercial building. *Sustainable Energy, Grids and Networks*, 26:100454, 2021.

[18] M. A. Hassan, S. Khan, and M. Tariq. IoT-enabled real-time monitoring and fault diagnosis framework for smart power distribution systems. *National Journal of Information and Power System Technology*, 12(1):112–128, 2026.

[19] M. Giordano, M. Filippini, L. Cavigelli, M. Magno, and L. Benini. TinyML: Enabling of inference deep learning models on ultra-low-power IoT edge devices for AI applications. *Micromachines*, 13(6):851, 2022.

[20] T. Rahman, M. S. Hossain, and R. Islam. Deep learning based detection, classification, and location of power system faults at the edge. *IEEE Transactions on Smart Grid*, 17(3):2104–2115, 2026.