The Egyptian International Journal of Engineering Sciences and Technology

ISSN: 1687-1006 p
ISSN: 2682-3640 e

https://eijest.journals.ekb.eg/

Vol. 41 (2023) 75–91

DOI: 10.21608/EIJEST.2022.144314.1161

IoT based Fourth Generation SCADA System for High Voltage Networks Fault Diagnosis based on BSDT-ANN

Amr A. Fayyad[a]*, Amal F. Abdel-Gawad[b], Saber M. Saleh[c], Ahmed Alenany[d]

[a] *Egyptian Black Sand Company, Baltim, Egypt*
[b] *Dean, Faculty of Computers and Informatics, Zagazig University, Zagazig, Egypt*
[c] *Department of Electrical Power and Machines, Faculty of Engineering, Fayoum University, Fayoum, Egypt*
[d] *Department of Computer and Systems, Faculty of Engineering, Zagazig University, Zagazig, Egypt*

ARTICLE INFO

Article history:

Received 13 June 2022
Received in revised form 15 August 2022
Accepted 03 September 2022
Available online 03 September 2022

Keywords:

Fault Classification
Location, Machine
Learning Ensemble
Classifier Artificial
Neural Network Boosted
Decision Tree
Microcontroller
Transmission Line
ThingSpeak
IoT
SCADA

ABSTRACT

Smart Grid substations rely on conventional Supervisory Control and Data Acquisition (SCADA) systems for remote supervision and control. However, these systems are limited in the geographical area they cover. Recently, the Internet of Things (IoT) has paved the way for connecting a vast number of devices to the Internet, which would be effective and beneficial for power system automation and data acquisition. In this paper, an intelligent low-cost SCADA system based on IoT for transmission line fault diagnosis is proposed. In the first step of the proposed scheme, voltage and current signals at the relaying point are preprocessed and analyzed using Discrete Cosine Transform (DCT). Next, signal energy components are extracted and sent to a Boosted Decision Tree (BSDT), a reliable and fast ensemble classifier, to identify fault type and, accordingly, a specific ANN is activated to estimate fault location. The diagnosis data is sent to a microcontroller to be displayed, trip the load circuit, and allow communication with Cloud ThingSpeak platform via the ESP8266 wi-fi communication module which makes data available anywhere all over the world. The approach is applied on a real-world HV transmission line, located between Samalut and Cairo, Egypt, where more than 16000 faults cases are well tested. The results show the reliability, validity, and effectiveness of the proposed approach.

1. Introduction

Transmission lines play a vital role in the distribution of electricity all over the world. Due to their direct exposure to theatmosphere for hundreds of kilometers, they suffer from high failure rates [1]. The process of repairing transmission lines faults may result in huge economic losses, especially if it takes long periods of time [2]. For this reason, determining the exact location of a transmission line fault is very important.

Nowadays, cost-effective microprocessor-based digital relays in power system protection enable real-time measurements of various signals which can be used to detect, classify, and locate faults. Relays became physically smaller, easier to set and test, and able to communicate [3]. Meanwhile, great progress was made in software for relay protection algorithms, digital signal processing (DSP), data mining, and machine learning, etc.

Machine learning techniques are widely used in fault classification, i.e., whether it is line-to-line, line-

* Corresponding author. Tel.: +20 01142784683.
E-mail address: amrfayyad13@gmail.com.

to-ground, line-to-line-to-ground, or three lines to ground. For example, decision tree is popular due to its interpretability and short inference time [4]. It builds a classification model based on iterative partitioning of the training data [5],[6]. In ensemble classifiers, predictions are performed using multiple classification techniques to achieve higher accuracy and avoid overfitting. Tree-based ensemble techniques are commonly used for classification and regression in many research fields [7],[8]. Two such tree-based ensemble techniques are the Boosted Decision Tree (BSDT) for classification [9] and bagged decision tree for regression problems [10].

Locating the fault is an additional challenge than determining the type of fault. Many fault location techniques are available, including impedance techniques, traveling wave techniques, wavelet analysis, and machine learning approaches [11]. Impedance relay is better suited for lines of medium length. However, it suffers from arc effects. Traveling wave method increases the speed of protective relaying. However, it cannot distinguish between waves reflected from the fault itself and the far end of the line and it uses high sampling rates. On the other hand, the wavelet dependent methodologies [12],[13] rely on human expertise in their decision making. Artificial neural networks (ANN), a universal function approximator, can be used to predict fault location with higher accuracy compared to other algorithms [14-16]. For example, Obi et al. [17] employed ANN in detecting and locating faults using GPS and GSM for remote information access. However, the system is tested on fault resistance up to only 50 Ohms. In addition, Ankar and Yadav [18] used ANN to accurately locate faults in a HVDC Transmission System. Also, Raj and Chandran [19] employed ANN for fault location. However, the system was not tested on big testing samples.

In addition to determine fault type and location, transmission line monitoring plays a key role in maintaining the required levels of grid performance, reliability, and security. For instance, Mirsaeidi et al. [3] developed an online micro-grid system based on phasor measurement units (PMUs) for adaptive protection, which updates the desired protective response in a timely manner in response to changes in system circumstances or requirements. However, PMUs call for intricate eigenvalue and eigenvector calculations. Transmission line monitoring of faults also makes it possible to perform predictive maintenance, particularly for outdated transmission line infrastructure, by warning probable faults in real-time or by making decisions offline using corporate strategic management procedures [20].

Recently, Power systems worldwide are moving towards smart grid systems which are an advanced technique of power transmission, distribution, measurement, energy control, and planning using digital technology and advanced communication systems [21],[22]. With the commercial availability of cloud computing, the smart grid can be Cloud-Based system [23]. As Cloud Computing has helped in bringing IoT to a reality [30], SCADA systems have increasingly adopted IoT technologies [24],[25] making use of cloud computing to significantly reduce infrastructure costs and increase ease of maintenance and smart grid integration. Cloud computing also allows data sharing with third parties as depicted by Talaat et al in [23]. Dhend and Chile [21] built a SCADA system using GSM. However, due to the use of the cellular network, this design is expensive and lacks a cloud or database to make fault predictions. Minal Karalkar et al. [26] used IoT in recording fault detection events but the system did not report any data about fault location. In [27], Monica et al. proposed a protection scheme against faults integrated with IoT mechanism, to inform the responsible person with location information along the overhead line. However, they did not discuss the results or any hardware implementation. Wang et al. [28] used Lora and IoT for fault monitoring in the distribution network. Dhanalakshmi, and Sunkari [29] employed fuzzy logic in fault classification and IoT for remote monitoring. However, the system is tested in simulation only. Tom and Sankaranarayanan [30] proposed a SCADA system integrated with fog, and it uses IoT for the distribution system automation. Don et al. [31] used IoT for remote fault data display and storage but did not reveal any results or curves. Mohammed [32] proposed smart systems for monitoring important parameters in electrical substations and transformers based on IoT, however, it did not discuss the faults. The use of IoT in power generation, smart energy applications, data transmission networks, and business is illustrated by Ahmad and Zhang in [33]. The use of IoT in fault automated recording in other fields is depicted in [34-36].

Based on this context, this paper proposes an intelligent system, based on an IoT device, which continuously monitors the ideal working status of transmission lines. This makes fault data of real-world power transmission grids, the transmission line between the city of Cairo and Samalut, available on the internet and so it can be accessed from any place in the world and provide location details to the authorized person. First, the measured field data of currents and voltages can be stored in a local or a remote cloud server, fed to a proposed robust BSDT tree, and then to the proposed ANNs for fault diagnosis. The system can monitor the outage of the transmission lines in real-time. The proposed approach used MATLAB Simulink to simulate, classify faults using the novel BSDT algorithm, and then locate them using the designed ANNs locator. Next, the results are sent, via serial port, to an Arduino microcontroller which compares the transferred data with a set of predefined values. The faults are then classified, and the microcontroller sends a signal to the relay that interfaces with the system to isolate the fault as well as a parallel signal to the LCD to display the kind and location of the fault as transferred from the module that simulated for fault location identification. The detected fault event is then immediately transmitted via the ESP8266 wi-fi communication module to the cloud where it will be saved in the ThingSpeak MySQL database.

The paper is organized as follows. In Section 2, the power system under study as well as the outline of the proposed scheme is presented. The classification and regression techniques, BSDT tree and ANN, are reviewed in Section 3 including preprocessing and training. In Section 4, the integrated IoT model including the hardware and ThingSpeak platform is proposed. Section 5 provides several experiments, on the Transmission line between the city of Cairo and Samalut, Egypt, to test the validity of the proposed scheme. Finally, conclusions and possible future extensions for the current work are drawn in Section 6.

2. Transmission line understudy and overview

In this paper, the power system model under study is a 500 kV, 50 Hz three-phase double end sources transmission line of 209 km length between Samalut and Cairo, depicted in Figure 1. The transmission line is modeled using the distributed parameters model whose values are given in Table 1.

The overall hardware and software framework of the proposed protection scheme is also depicted in Figure 2. Voltage and current signals, at bus-1 at the sending end of the transmission line in Figure 1, are first preprocessed and then fed to the BSDT_ANN modules for fault type and location estimation. This part is simulated in MATLAB/Simulink on the host PC. A low-cost hardware platform of ATmega328P microcontroller is used to display the fault type and location on LCD and trip the load circuit. The microcontroller is serially communicated with the Internet of Things ESP8266 Wi-Fi module which sends fault information to the cloud to be available on the internet.

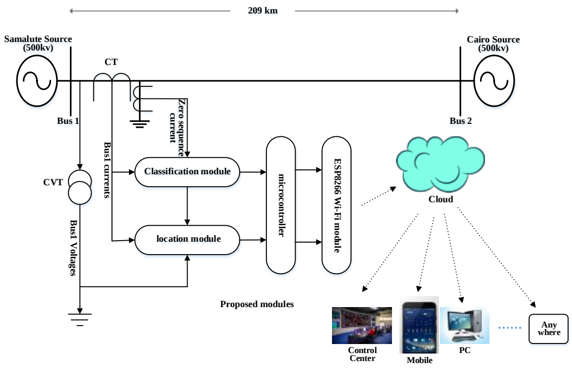


Fig. 1. The transmission line under study and the proposed fault diagnosis scheme.

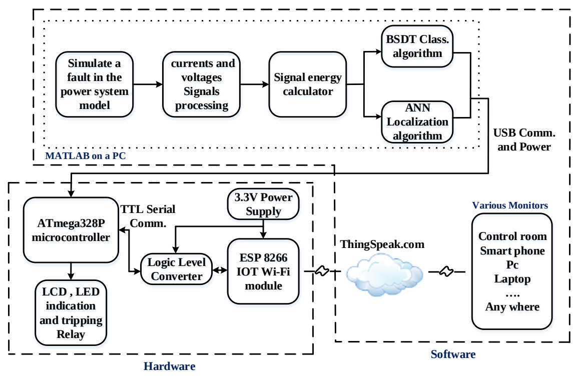


Fig. 2. Block diagram of the proposed relay hardware and software fault diagnosis scheme.

Table 1. Transmission line parameters.

Parameters	Unit	Value
Positive sequence resistance	**(Ω/km)**	0.0217
Zero sequence resistance	**(Ω/km)**	0.247
Positive sequence inductance	**(mh/km)**	0.96129
Zero sequence inductance	**(mh/km)**	2.4828
Positive sequence capacitance	**(nf/km)**	8.1618
Zero sequence capacitance	**(nf/km)**	8.1618
Rated voltage	**(kV)**	500
Load angle of Samalut source	**(Degree)**	0
Load angle of Cairo source	**(Degree)**	-10

3. The proposed BSDT_ANN protection scheme

In this section, the details of the design of the proposed fault diagnosis scheme are described. This includes the pre-processing of current and voltage signals, the preparation of training data, and the training of both BSDT and ANN for fault classification and location, respectively.

3.1. Pre-processing and Discrete Cosine Transform for feature extraction

To reduce the size and time for the boosted decision tree and the neural network training, pre-processing is needed to remove redundant information and noise from pre-or post-fault signals. First, samples for training are generated with MATLAB, in which voltage and current signals per unit values for different fault cases are generated in various stages along the transmission line. These instantaneous voltage and current values are sampled every 0.1 msec or equivalently at a rate of 10 kHz satisfying Nyquist sampling criteria. A second-order Butterworth filter with a 400 Hz cutoff frequency is used to process the signals to remove higher-order harmonics, beyond 7, resulting from noise and non-linear power electronic components. After sampling and filtering, the discrete cosine transform (DCT) is applied. DCT allows very accurate reconstruction of a sequence from only few DCT coefficients. This is useful for the transmission of reduced data and machine learning applications. Four standard variants of DCT are available [37]. In this paper, the more accurate DCT-4 is used for fault signals. The DCT transforms a discrete-time signal x of length N to another signal y of the same length N according to the formula:

$$y(k) = \sqrt{\frac{2}{N}} \sum_{n=1}^{N} x(n) \cos\left(\frac{\pi}{4N}(2n-1)(2k-1)\right) \qquad (1)$$

for $k = 1$ to N.

3.2. Preparing datasets for training and testing

Although the DCT coefficients have quite distinctive features for fault signals, they are not directly employed in decision-making. For fast data processing, the spectral energy for DCT signal is preferred. Signal energy is the sum of the square of the DCT coefficients calculated as follows [38].

$$E = \sum_k |y(k)|^2 \qquad (2)$$

Where E is the energy of DCT coefficients, and k is the sample number since the signal is discrete. A three-phase transmission lines consists of A, B, and C phases, in addition to the ground line. Therefore, in this paper, ten types of faults are simulated with different fault resistance and location as shown in Table 2. To generate the dataset to be used to train and test the BSDT tree and the ANN, these faults are induced, one at a time, and the DCT and its energy are calculated. A MATLAB program script is used to calculate the DCT coefficients and their spectral energy for the overall signals. Next, coefficients energy values are fed to the BSDT and the ANN to be trained and to predict the fault type and its location in testing. Based on the details in Table 2, a dataset of 6880 samples will be used for training and testing. For training and testing of BSDT, a dataset sample consists of the four currents (IA, IB, IC, and IG) taken at the relaying point at one side of the transmission line, whereas for the ANN, a dataset sample consists of the three currents (IA, IB, and IC) and the three voltages (VA, VB, and VC) at the relaying point, i.e., six inputs are fed to the ANN.

Table 2. Fault scenario parameters used in BSDT and ANN training and testing.

Parameters	***Value***
Fault Type	**AG, BG, CG, ABG, BCG, ACG, AB, BC, AC, ABC.**
Fault Location (km)	**0.8, 3, 5:5:205**
Fault Resistance(Ω)	**0.05, 0.15, 0.25, 0.5, 1, 2, 5, 10, 15, 25, 35, 45, 50, 60, 70, 80**

3.3. Proposed fault diagnosis algorithms

Correct fault classification allows for single-pole tripping and reclosing while identifying the exact fault location in a faster recovery of the supply after the fault [14]. In this section, the proposed methodology of BSDT-ANN has been adopted after a wide range of experiments performed to reliably determine fault type and locate it.

3.3.1. BSDT for fault classification

The decision tree is one of the most used techniques for classification. They are easy to interpret [39] and more suitable for real-time tasks because of their moderate training time and fast classification speed as they can be easily implemented in a digital platform using nested If…then…else statements [4], [40]. For this reason, decision trees are adopted in this paper for fault classification. As its name suggests, a decision tree takes the form of a tree where at each node, starting from the top root node, specific conditions are tested iteratively, and decisions are made until the output class corresponding to the given data is determined. Decision trees are trained using several algorithms, e.g., ID3 [41],[42] with the objective of maximizing the homogeneity or reducing the entropy of examples after a split occurs at each node. Once the tree is built, it can be pruned to reduce its size.

Another strategy in machine learning is to use ensemble classifiers [9]. These techniques use several learning algorithms so that they can produce more accurate models and avoid overfitting [43]. Therefore, in this paper, the robust ensemble method of boosting decision tree is employed. In Boosting, an equal weight, i.e., using a uniform probability distribution, is given to the sample training data, Dataset1, at the very starting round. The Dataset1 is then given to a base learner, DT Model1. The misclassified instances by DT Model1 are assigned a weight higher than the correctly classified instances but considering that the total probability sums to 1. This boosted data, Dataset2, is then fed to the second base learner, DT Model2, and so on. Finally, a voting is applied on the combined results of each learner as shown in Fig. 3. In this paper, ten learners are used for fault classification.

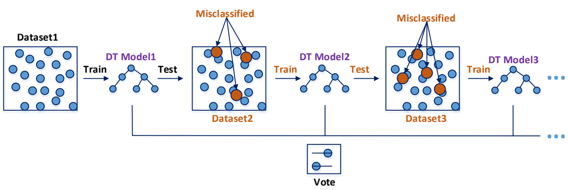


Fig. 3. Process flow diagram describing Ensemble Boosted decision tree training.

The challenge of a fault classification scheme is to satisfy the constraints of real-time operation. This is accomplished by using a reduced number of key features to reduce computational complexity while maintaining acceptable classification accuracy. Therefore, only calculated energy of the DCT of the three-phase currents and zero sequence current at relaying point are fed as inputs to the BSDT in the fault classification module.

A training data consisting of 6880 examples covering different faults are used to train the BSDT fault classifier. For each individual predictor case, there are four inputs representing features and one output for each response data encoding the corresponding fault class in decimal represented by a number from 1 to 10 for various phase-to-phase (e.g., AB, BC, etc.) and phase-to-ground (e.g., AG, BG, etc.) faults. This corresponds to (4×6880) predictor data (input data), and (1×6880) response data (target data).

The training data are shuffled and divided into two subsets via the option of Cross-Validation by 5-folds in the GUI classification learner in MATLAB to protect against overfitting by partitioning the dataset into 5 folds. First, the model is tested with the first fold, and it is trained with the remaining folds. The second fold serves as the testing set in the second iteration, the rest as the training set, and so on. In order to generate a single estimate of accuracy, the five results can then be averaged.

The BSDT approach is used to determine robust thresholds to classify the faults. Fig. 4 Shows the overall scheme included the trained BSDT-based fault classification designed module. Once a fault type is determined, the location module is activated, and the location of the fault is estimated by using the designed ANNs for that purpose.

3.3.2. ANN for fault location estimation

Artificial neural networks (ANN) are known to be a universal function approximator and so can be used in estimating fault location. The most well-known training algorithm for multi-layer neural networks is the Backpropagation algorithm. It searches for the best set of weights that minimizes the mean square error using a gradient descent optimization technique. Although decision tree has less inference time, ANN is more accurate for regression problems [15].

Two BSDT classifiers are employed in series. First, faults are classified into one of ten faults: AG, BG, CG, ABG, BCG, ACG, AB, BC, AC, or ABC (discussed in the previous section). Next, these faults are further classified into one of the four categories LG, LLG, LL or LLL by a second simple BSDT classifiers as depicted in Fig. 4. Then, a MATLAB script activates only the corresponding ANN fault locator. The location module consists of four ANNs, one for each fault category (LG, LLG, LL, LLL) as shown in Fig. 4. The inputs of each ANN are six calculated energies of the DCT coefficients of the three-phase currents and voltages at the relaying point. The module output is the location of the fault in the line from the relaying point. Hence, to train LG, LLG, and LL faults, each ANN needs 6 x 2004 input data and 1 x 2004 output data. On the other hand, LLL ANN needs only 6 x 668 input data and 1 x 668 output data as it contains one fault type, i.e., ABCG, see Table 2. As the design of ANN fault locator is more difficult than fault classifier due to the enormous number of possible outputs, several trials are made to arrive at the best possible ANN architecture, for instance, ANN4 is designed to contain three hidden layers with 8, 8, and

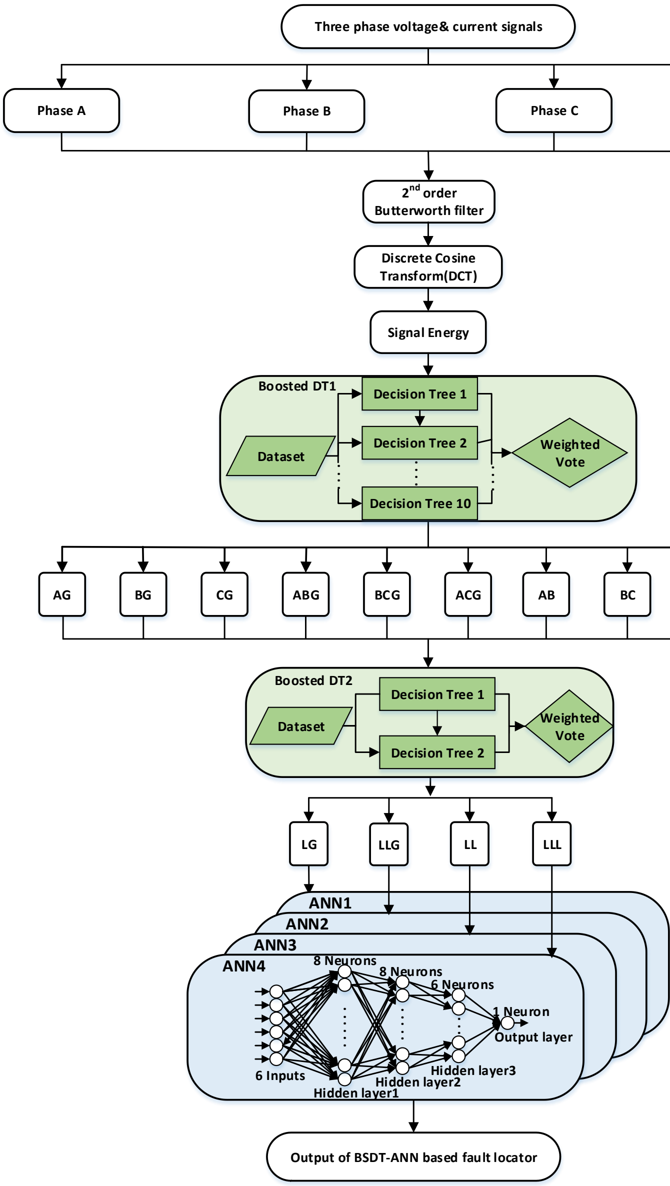


Fig. 4. The overall scheme for fault classification and location based on BSDT_ANN

6 neurons for layer1, layer2, and layer3 respectively. This design was found as the best design for accurately locating the faults within the same category of faults after many trials. Fig. 4 Shows the overall scheme included the trained ANNs-based fault location designed module.

4. The IoT model for the proposed system

The internet of things (IoT) is a system which can link various devices, analog and digital machines, animals, objects, or people, and can send data over a network without human intervention [44]. Each device in the IoT system is supplied with Unique Identifiers (UIDs). In mobile technology fifth generation (5G), enormous number of smartphones linked with IoT devices is expected to produce a huge amount of data transmission varying from a small number of bytes up to plentiful gigabytes [45]. IoT in this study is used for designing a cost-effective network for transmission of fault information to the cloud so that it becomes available anywhere to facilitate monitoring in real-time and troubleshooting. The details of the proposed IoT system for fault diagnosis is presented in this section.

4.1. ESP8266-01 for IoT

The ESP8266 is a low-cost and highly integrated Wi-Fi System On a Chip (SOC)module incorporating TCP/IP networking software, and a built-in microcontroller, produced by Espressif Systems in Shanghai, China for IoT utilization applications [46]. ESP8266 can connect a microcontroller to a wireless network. When the microcontroller sends the fault data to the ESP8266 unit, it can be stored on the server as well as being sent to a receiver. The faults can then be monitored anywhere simply by using the ThingSpeak platform on a PC or the android application of ThingViewer on a smartphone.

The chip was brought to the interest of Western fabricators for the first time in August 2014 with the ESP8266-01 module, made by Ai-Thinker third-party manufacturer. This small board enables microcontrollers to communicate to a Wi-Fi domain and uses Hayes-style commands (AT commands) for making easy TCP/IP connections. At first, there were difficulties because there was almost no documentation in English either on the slide or on the commands on it. The extremely low price and simplicity of the unit attracted many hackers to translate Chinese documents and explore the unit, chip, and software on it. In October 2014, the need for a separate microcontroller to program the chip was eliminated as Espressif Systems released a Software Development Kit (SDK) to program the chip directly. Another alternative of "Unofficial Development Kit" is introduced by Mikhail Grigorev [47]. Other mostly open-source SDKs, include Arduino, a C++-based firmware [48].

Some of the advantages of the Esp8266-01 module are represented in the small size, and being a serial Wi-Fi module, it is very easy to connect with the Arduino board via serial connection. The ESP8266-01 has a built-in microcontroller, so it can be used as a standalone microcontroller and Wi-Fi module in one amazing combo. The ESP8266-01 can operate continuously in industrial environments, due to its wide range of operating temperatures. With highly integrated on-chip features and a minimal number of separated external components, the chip provides compactness, reliability, and durability. ESP8266-01 built with an extra-low power consumption 32-bit Tensilica processor, which reaches a maximum clock speed of 160 MHz as well as a standard digital peripheral interfaces, RF balun, antenna switches, power amplifier, low noise receive amplifier, power management system, and filters. All of them are got together in one tiny package.

The Real-Time software and Wi-Fi stack saves about 80% of the power used in processing to be available. The power-saving architecture provides three modes of operation: active mode, sleep mode, and deep sleep mode. This enables designs depending on a battery to operate for much longer.

Table 3 depicts some specs of the used chip in this paper proposal. The active pins include the General-Purpose Input/Outputs (GPIOs) and Analogue to Digital Converter (ADC) pins with which external devices can connect to the ESP8266 Microcontroller Unit (MCU). The unit is packaged as 2 × 4 Dual In Line (DIL), as shown in Fig. 5. Many ESP-xx modules involve SMT LED which can be used to blink whenever.

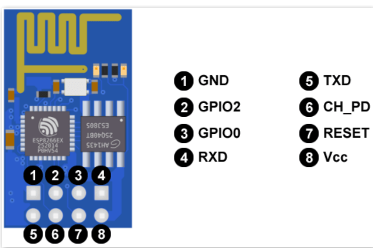


Fig. 5. Pinout of ESP-01

All ESP board types have one serial port (known as a UART or USART) or more used to communicate with the Arduino board or other devices. ESP module communicates via RX and TX pins and uses TTL logic levels of 3.3V. For serial connection with Arduino microcontroller, the ESP-01 module is connected to the microcontroller's Transmit (Tx) and Receive (Rx) pins via a level converting circuit.

4.2. Hardware integration with software

The proposed system consists of two modules based on BSDT and backpropagation ANN (BP-ANN) that classifies ten types of faults, as well as estimates the location of the faults. The two modules are simulated in MATLAB environment running on a host PC. Each module is integrated simultaneously with a cost-effective, accurate, and fast Microchip ATmega328P microcontroller-based Arduino unit to display fault type and location on LCD, trip the load circuit and sending these data to ESP8266 module for communication with ThingSpeak platform over the cloud. Arduino is a high-performance prototyping platform for open-source electronics built on easy-to-use modular hardware and software. This is what stimulated its use for the integration with MATLAB/Simulink proposed models and ESP8266 module. The Arduino board operates at 1.8 V to 5.5 V and has a 32 KB of flash memory for storing programs (with 0.5 KB used for the bootloader), 2 KB of SRAM, 1 KB of EEPROM for storing parameters, a 16 MHz crystal oscillator, a USB connector, 14 digital input/output pins of which 6 can be used as PWM outputs, 6 analogue I/O pins, ATmega16U2 USB-to-TTL Serial chip, and RX and TX LEDs (flashes in data transmission). Arduino is programmed with the Arduino software Integrated Development Environment (IDE) and interfaced with MATLAB by the Support Package software for Arduino and a type B USB cable.

The 8- bit ATmega328P microcontroller on the board is pre-programmed with a bootloader that enables uploading the proposed algorithm to it without the use of an external programmer. The block diagram of the proposed relay based on simulation, ESP8266-01 module, and Arduino Microchip ATmega328P microcontroller board is shown in Fig. 6. The protection algorithm fed into the microcontroller compares the transferred data (via serial port), from the fault BSDT classifier in MATLAB environment, with a set of predefined values. Next, the faults are classified, and the microcontroller sends a trip signal to the interfacing relay to disconnect a lamp showing the fault isolation and sends a parallel signal to the LCD to display the fault type and the fault position transferred from the designed ANN built in MATLAB for fault location estimation. The Microcontroller controls four LEDs that indicate which phase is faulted as shown in Fig. 7. The complete power system network, signal analysis, the BSDT, and ANN algorithms were implemented in MATLAB/Simulink environment.

The ESP8266-01Wi-Fi module contains an Analog RF transmitter, Analog RF receiver, RF Balun, and antenna switch to connect to the global ThingSpeak and send data through its analogue Wi-Fi circuit using HTTP requests to store the history of all fault events of the monitored transmission line in the ThingSpeak MySQL database. Fig. 7 depicts the proposed IoT-based experimentally co-simulated system.

The Arduino digital pins can be configured to be serially communicated with the ESP8266-01 Wi-Fi board. A logic level converter circuit, shown in Fig. 8, is needed to interface the 5 V TTL logic to the 3.3 V TTL for serial communication with ESP8266-01. When the ESP-01 board is bought, it comes with a preinstalled AT firmware which is compatible with the Arduino IDE so the Arduino IDE software built-in serial monitor and Bare Minimum software.

script can be used to configure the ESP board before starting work (e.g., to set operation mode and to set station IP address). Fig.8 depicts the proposed scheme showing the various components used in the hardware implementation. A typical ESP8266 module draws some 170mA and so, an external power supply is used. The typical setup hardware part is depicted in Fig. 9.

Table 3. specs for ESP-01.

Name	Active pins	LEDs	Antenna	Dimensions (mm)	Notes	ADC	Wi-Fi	voltage
ESP-01	6	Yes	PCB trace	14.3 × 24.8	1 MiB Flash, AI-Cloud, and black PCB from AI-Thinker.	10 bits	IEEE 802.11 b/g/n	+3.3 V; can handle up to 3.6 V

4.3. The ThingSpeak

One of the most important parts of this paper is the ThingSpeak, an open internet of things platform that will permit to collect, analyze, and act on collected data. To use ThingSpeak, first it required to create an account on ThingSpeak web site depicted in Fig.10. Next, a channel is created and two fields which are to be monitored are added for fault type and its location.

Fault diagnosis data are uploaded to the channel through the channel fields and ID. Also, configuring actuators, to control a trip signal for instance, is also possible with ThingSpeak. Furthermore, ThingSpeak allows data analysis, visualization, and storing the history of all fault events in the MySQL database. Thus, fault diagnostics can be monitored from anywhere and can be sent as alerts using email and Twitter with the help of React app in ThingSpeak [29].

The main task of this IoT proposal based on ThingSpeak is to predict the need for maintenance which is an important task especially for old-age equipment to decrease maintenance costs and crew labor hours in searching for the location of a fault in

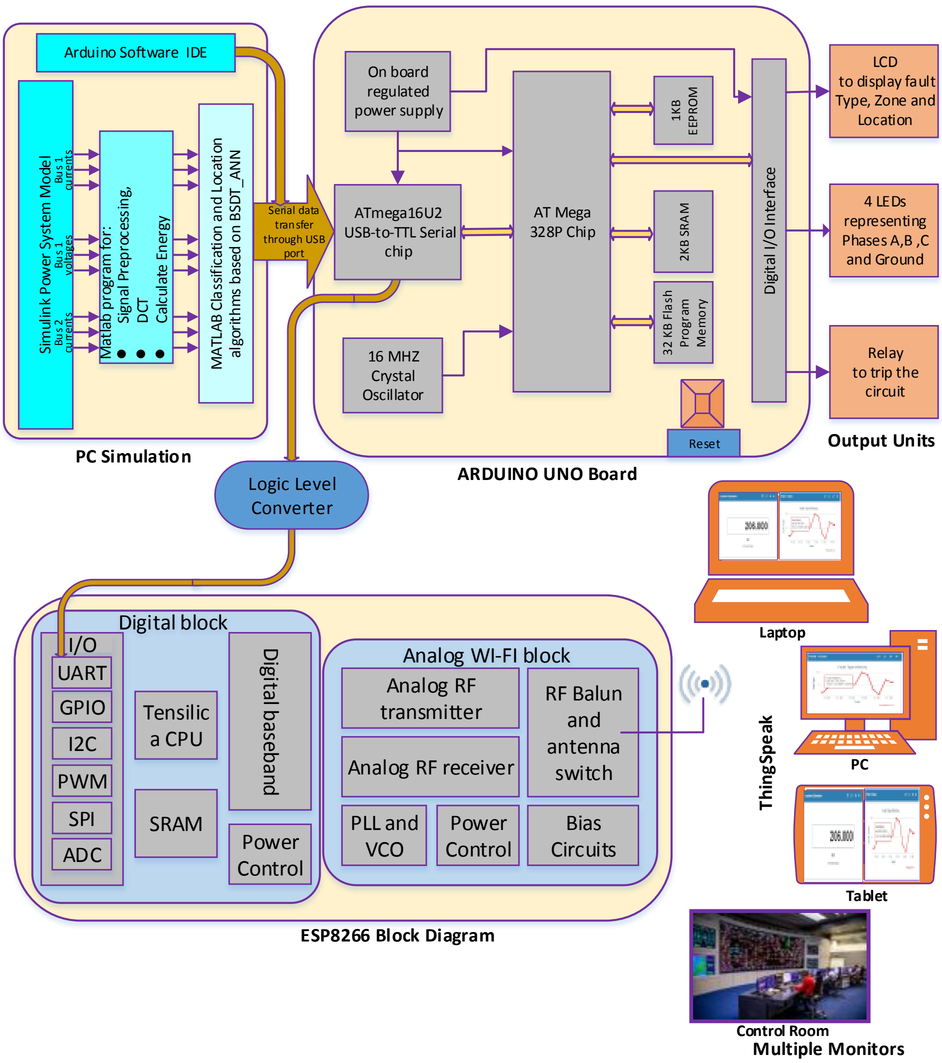


Fig. 6. Proposed software and hardware integrated fault diagnosis

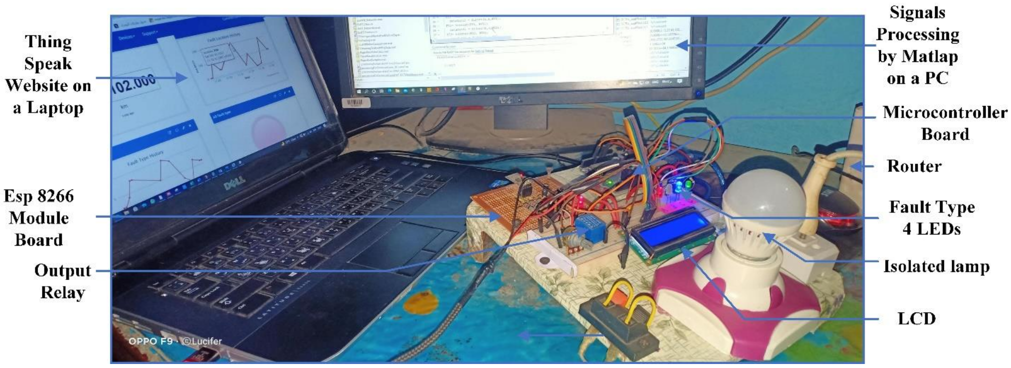


Fig. 7. Experimental co-simulation arrangement for the proposed interactive scheme employing Thing Speak platform.

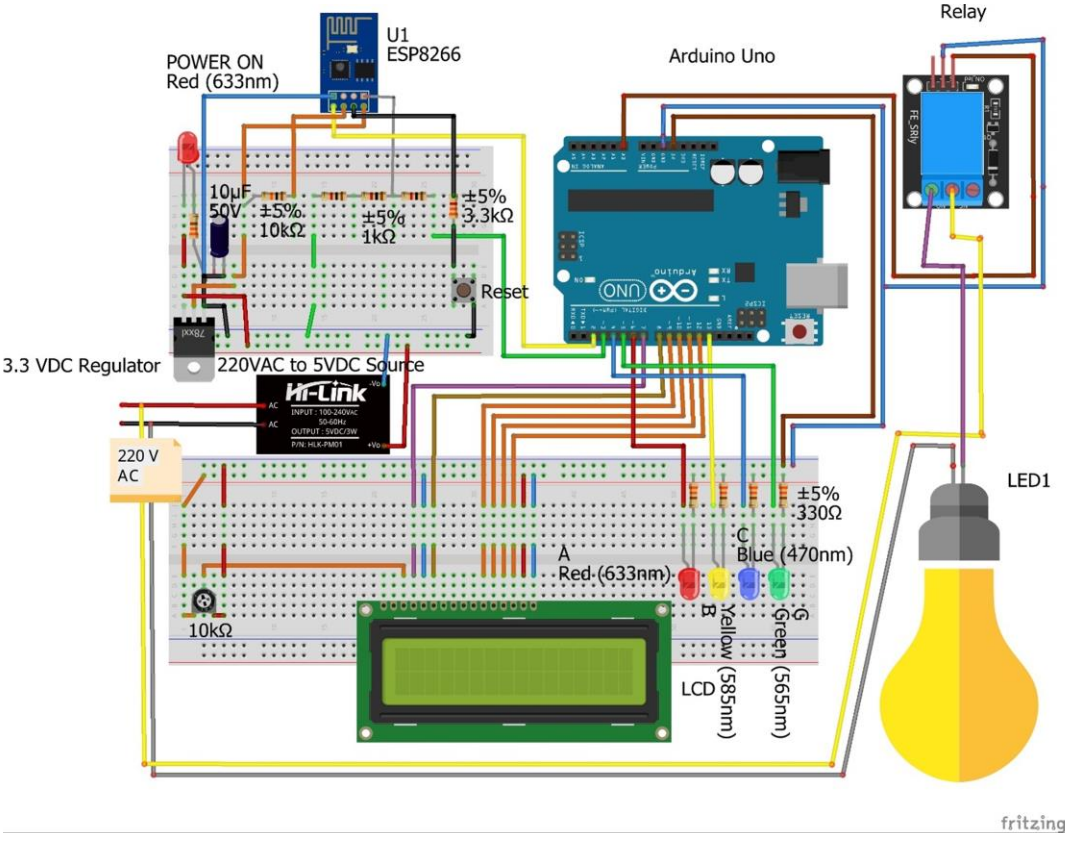


Fig. 8. The circuit diagram of fault IoT monitoring scheme in Fritzing.

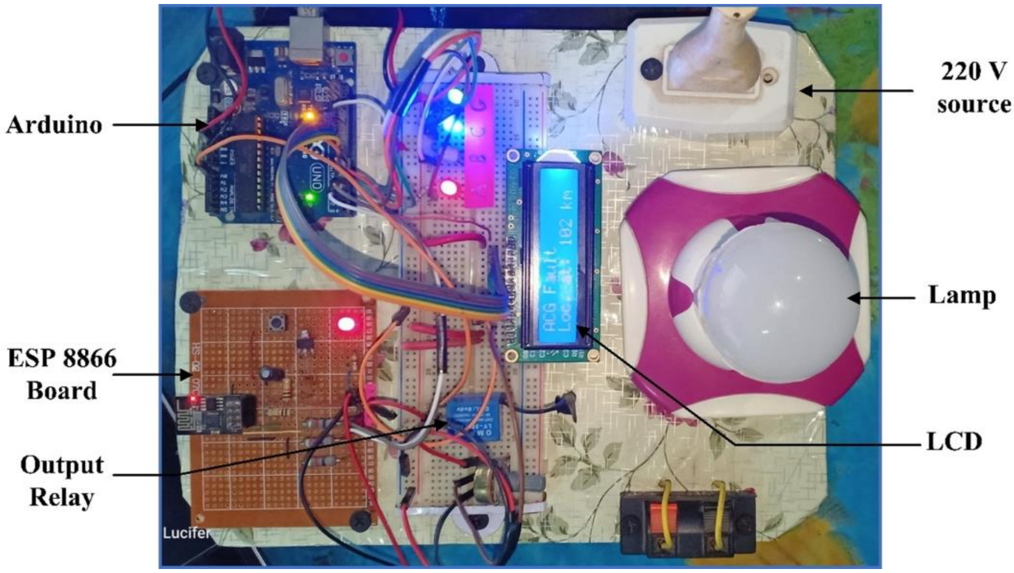


Fig. 9. Hardware setup

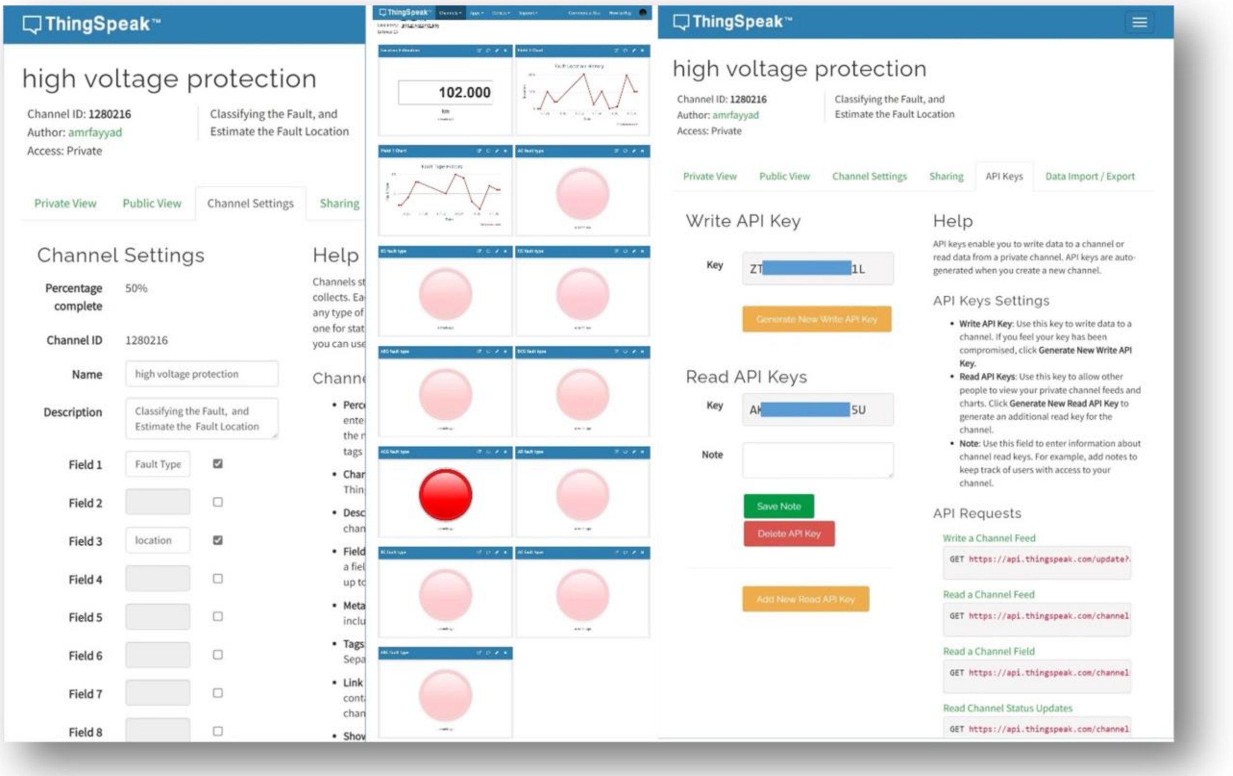


Fig. 10. ThingSpeak platform interface.

difficult terrain. For example, the maintenance technician may receive a pop-up message on his smartphone informing him that a certain fault type at a certain location is occurred, which accelerates solving the issue. In hardware, the ESP8266-01 board Wi-Fi module is used to send the BSDT output of fault classification and ANN output of fault location to the ThingSpeak channels. The values of the BSDT-ANN system output are uploaded to the ThingSpeak designed channels every 15 sec with the initially configured ThingSpeak channel.

The server containing the fault diagnosis system may be installed locally at the relay point or remotely on the cloud. IoT Android application can communicate with the server every 2 seconds and refresh its data fields according to server data sent by RTUs (Remote Terminal Units) [36]. There are many open-source android APKs used for IoT, among which, the ThingView software is used in the current work.

5. Results

This section discusses the results in testing the diagnosis algorithm and the related monitoring system.

5.1. Results of diagnosis algorithms

The proposed classifier and locator are extensively tested using a reasonably large data set consisting of 16640 samples with wide variability in fault location along with the proposed 209 km transmission line.

The overall performance of the classifier is evaluated using fault classification accuracy defined as:

$$\%Accuracy = \frac{N_c \, x \, 100}{N_t} \tag{3}$$

Where, N_c is the total number of correctly classified cases and N_t is the total number of fault classification cases. In addition, the performance of classifier for each individual class is evaluated using the *sensitivity* or *recall* defined as:

$$\%Recall = \frac{N_{c_i} \, x \, 100}{N_{t_i}} \tag{4}$$

Were, $\boldsymbol{N_{c_i}}$ is total number of correctly classified cases belonging to the i-th class and $\boldsymbol{N_{t_i}}$ is the total number of cases in the original training set belonging to the same class. As can be seen from Table 4, the overall classifier accuracy is 99.98% (corresponding to 3 incorrect cases out of 16640) while the recall is 100% for LLG, LL, and LLL faults, and 99.94% for the LG fault.

To evaluate the accuracy of fault location estimate, the percentage error defined as

$$\% \, location \; Error = \frac{[actual \; distance - estimated \; distance] \; x \; 100}{Length \; of \; line} \tag{5}$$

is used, where the mean value and standard deviation of the percentage error are given in Table 4. As can be

seen, the error is below 0.5%.

To test the robustness of the proposed scheme, simultaneous changes in resistance, location, and type are induced and 560 samples are generated and tested. The results are shown in Table 5. The recall is 100% for all classes and again the % error in fault location is below 0.5%.

Although BSDTs and ANNs are trained for fault resistance range up to 80 Ω, the test is performed for a resistance up to 100 Ω. Also, the proposed scheme has been tested for various faults at different location from the relaying point, ranging from 2 km to 207 km. The results of some selected examples for the robust test with wide changes in fault input parameters are given in Table 6 for different fault scenarios.

It can be noticed that the Boosted decision tree and neural network sensitivity were not affected by any changes in the fault parameters thus depicting the robustness of the proposed algorithm to any parameter variations. AlsoThe inference time recorded in Tables 4, 5, and 6 for the BSDT classifier, shows that the proposed fault scheme can be used to develop very high-speed protection systems where it ranges from 0.0075 sec to 0.0104 sec.

Table 4. General test results of the proposed scheme for varying in fault location of 16640 fault cases.

Type of fault	Total fault cases	Performance of fault classifier				Performance of fault locator	
		Right cases	Wrong cases	% Recall	average inference time (sec)	Error mean value	Standard Dev.
L–G	4992	4,989	3	99.94	0.0079	0.0134	0.0318
L–L–G	4992	4992	0	100.00	0.0093	0.0177	0.0958
L–L	4992	4992	0	100.00	0.0102	0.3900	0.5580
L–L–L	1664	1664	0	100.00	0.0104	0.0203	0.1143

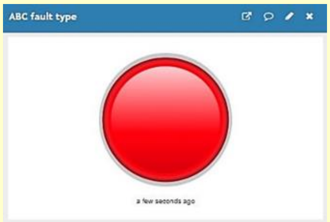
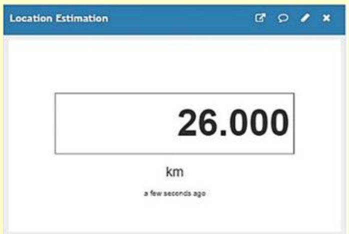
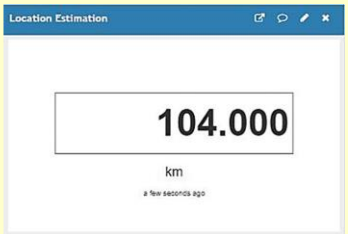
Table 5. Test results of the proposed scheme for varying resistance, type, and location of 560 fault cases.

Type of fault	Total fault cases	Performance of fault classifier				Performance of fault locator	
		Right cases	Wrong cases	% Recall	average inference time	Error mean value	Standard Dev.
L–G	168	168	0	100.00	0.0084	0.0592	0.1194
L–L–G	168	168	0	100.00	0.0080	0.0953	0.2128
L–L	168	168	0	100.00	0.0080	0.4805	0.8973
L–L–L	56	56	0	100.00	0.0080	0.1193	0.2882

Table 6. Test results of the proposed location and classification schemes for various fault types with varying fault resistance and distance from relaying point.

Scenario no.	Fault Type	Fault Resistance	Fault Distance (Km)	Classifier Desired O/P	Estimated Output	% Recall	Inference Time (sec.)	Estimated Location	% Error
1	BG	0.04	2	2	2	100	0.0077	1.99	0.0062
2	ABG	4	103	4	4	100	0.0075	102.99	0.0029
3	BC	19	43	8	8	100	0.0078	42.44	0.2682
4	BCG	27	207	5	5	100	0.0078	206.97	0.0139
5	ABC	51	27	10	10	100	0.0079	26.98	0.0074
6	AC	73	103	9	9	100	0.0079	104.03	-0.4908
7	CG	100	2	3	3	100	0.0078	2.62	-0.2947
8	AG	19	13	1	1	100	0.0079	13.02	-0.0095
9	AB	51	201	7	7	100	0.0077	198.08	1.3993
10	ACG	73	103	6	6	100	0.0077	102.99	0.0027

Table 7. Fault results of table 6 as obtained in Proteus and using ThingSpeak.

Fault scenario no.	*4*	*5*	*6*
Fault type on Think speak			
Fault Location on Think speak			
With Simulation by Proteus			

5.2. Results of IoT monitoring system

ThingSpeak can be used to monitor and display fault information in a user-friendly interface as shown in Table 7 which shows some fault diagnosis results (secenarios 4, 5, and 6) that were collected in Table 6.

The table shows the results of fault type and location for each fault secenario which coincide with the simulation results in Proteus. The fault type window for a specific occured fault its circle is turned into a dark red coulour as shown also in Fig. 10 . It is noted that, the numbers have been rounded up so that it saves the computations. Form Table 6, Fault scenario 4 is for a BCG fault made at 207 km from the relaying point, Secenario 5 is for an ABC fault occuring at 27 km from the relaying point, and Secenario 6 is for an AC fault at 103 km from the relaying point and The proposed classifier and locator's actual result values are displayed in the same table.

Acquisition of data for ten consecutive different faults in the ThingSpeak cloud as displayed on the ThingSpeak platform is shown in Fig.11. ThingSpeak handles the ten various faults shown in Table 6 throughout the ten-minute period between 11.28 and 11.38 in a good and trustworthy manner. It is clear that Thingspeak was successful in recording every incident of a fault. The authorized individual can use this historical data to identify the most frequent fault areas and follow them in predictive maintenance.

Fig. 12 presents the same curves using the android ThingView APK. Of course, the availability of a mobile application for fault diagnosis makes fault management easier and faster. Once the authorized person open the application he can know the fault and all information about it. The straight horizonal line in the curves means the fault not fixed yet.

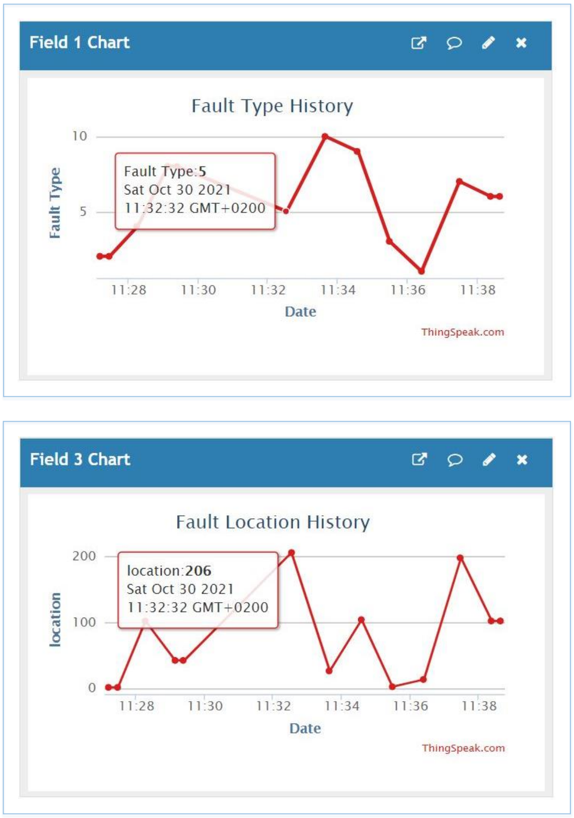


Fig 11. Historical diagnosis data in ThingSpeak website in the PC.

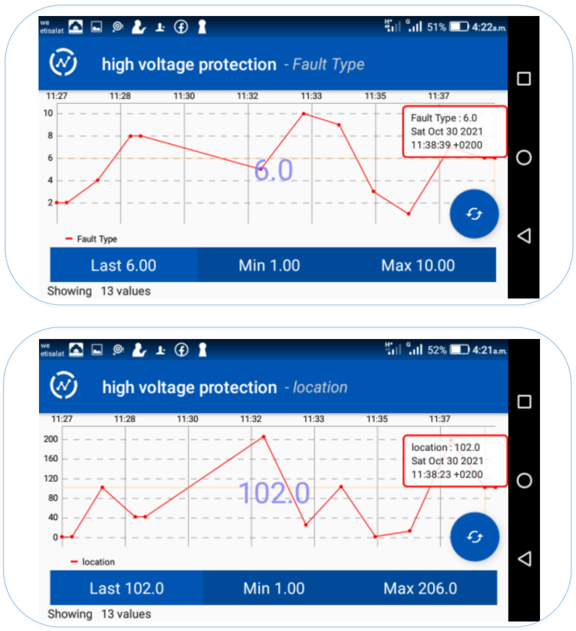


Fig. 12. Historical diagnosis data in the android application of ThingView.

6. Conclusions and future work

The upcoming revolution of the Internet of things (IoT) based on cloud computing presents a huge opportunity for enhancing power system quality and stability. In this perspective, a low-cost IoT based SCADA system integrated with BSDT_ANN machine learning algorithm, for fault diagnosis of transmission lines, is proposed in this paper. The diagnosis algorithm and power system model are simulated in MATLAB, implemented in hardware, and an Arduino board and ESP8266-01 module are used for power line fault monitoring. The system classifies, locates the faults, and alerts the operators by sending diagnosis information through the Thing speak cloud for resolving the faults in a short period of time and saving the data in MySQL database for future maintenance prediction. The results show the effectiveness, robustness of the proposed IoT based system, as well as its high speed in protective relaying systems of transmission lines. The proposed technique is straightforward to implement in a digital plateform. Compared to standard SCADA systems, the proposed IoT-based SCADA scales well with a large number of sensors where the system can handle a large amounts of data as it depends on cloud computing which have large storage resources, and allows data sharing with third party software. As a result, an authorised person can know quickly the fault data by checking a software program in his smartphone.

In the future, this IoT-based architecture may be enhanced towards faster communication speed and more reliability for power automation. Also, the system may be extended to include other power line smart sensors to achieve better quality in power delivery and reliability.

References

[1] Kumar A. W., Verma H. K., and Singh S., "Incremental Power Improves Relay Selectivity to Transmission Line Faults", J. Inst. Eng. Ser. B, vol. 100, no. 1, pp. 61–68, 2019.

[2] Järventausta P., Makinen A., Kivikko K., Verho P., Karenlampi M., Chrons T., Vehvilainen S., Trygg P. and Rintaopas A., "Using advanced AMR system in low voltage distribution network manage", 19th Int. Conf. electricity distribution, (CIRED 2007), Vienna, pp. 1-4, May 2007.

[3] Mirsaeidi S., Said D. M., Mustafa M. W., Habibuddin M. H., and Ghaffari K., "Modeling and simulation of a communication-assisted digital protection scheme for micro-grids", Renew. Sustain. Energy Rev., vol. 57, pp. 867–878, 2016, doi: 10.1016/j.rser.2015.12.218.

[4] Wang Y. and Yu S. Z., "Supervised learning real-time traffic classifiers", J. Networks, vol. 4, no. 7, pp. 622–629, 2009.

[5] Samantaray S.R., "Decision tree-based fault zone identification and fault classification in flexible AC transmissions-based transmission line", IET Generation, Transmission & Distribution, vol. 3, no. 5, pp. 425–436, 2009.

[6] Abdelgayed T. S., Morsi W. G., and Sidhu T. S., "A new harmony search approach for optimal wavelets applied to fault classification", IEEE Trans. Smart Grid, vol. 9, no. 2, pp. 521–529, 2018, doi: 10.1109/TSG.2016.2555141.

[7] Qu Y. et al., "Boosted Decision Tree Analysis of Surface-enhanced Laser Desorption / Ionization Mass Spectral Serum Profiles Discriminates Prostate Cancer from Noncancer Patients", Clin. Chem., vol. 1843, pp. 1835–1843, 2002.

[8] Li X., Wu Y., Tan Y., Cheng P., Wu J., and Wang Y., "Method based on floating car data and gradient-boosted decision tree classification for the detection of auxiliary through lanes at intersections", ISPRS Int. J. Geo-Information, vol. 7, no. 8, 2018, doi: 10.3390/ijgi7080317.

[9] Ağir T. T., "Using Machine Learning Algorithms For Classifying Transmission Line Faults", Dicle Univ. J. Eng., vol. 2, pp. 227–234, 2022, doi: 10.24012/dumf.1096691.

[10] Ashok V. and Yadav A., "A Novel Decision Tree Algorithm for Fault Location Assessment in Dual-Circuit Transmission Line Based on DCT-BDT Approach", no. c. Springer International Publishing, AISC 941, pp. 801–809, 2020, doi: 10.1007/978-3-030-16660-1_78.

[11] Subramani C., Jimoh A. A., and Davidson I. E., "Fault Investigation Methods on Power Transmission Line: A Comparative Study", IEEE PES Power Africa Conf., pp. 93–97, 2016.

[12] Kapoor G., "Protection Scheme For Double Circuit Transmission Lines Based On Wavelet Transform", ICTACT J. Microelectron., vol. 1680, no. October, pp. 656–664, 2018, doi: 10.21917/ijme.2018.0114.

[13] Guo M. F., Yang N. C., and You L. X., "Wavelet-transform based early detection method for short-circuit faults in power distribution networks", Int. J. Electr. Power Energy Syst., vol. 99, no. January, pp. 706–721, 2018, doi: 10.1016/j.ijepes.2018.01.013.

[14] Koley E., Kumar R., and Ghosh S., "Low-cost microcontroller-based fault detector, classifier, zone identifier and locator for transmission lines using wavelet transform and artificial neural network: A hardware co-simulation approach", Int. J. Electr. Power Energy Syst., vol. 81, pp. 346–360, 2016.

[15] Sakr F., Bellotti F., Berta R., and De Gloria A., "Machine learning on mainstream microcontrollers", Sensors (Switzerland), vol. 20, no. 9, 2020.

[16] Fayyad A. A., Abdel-Gawad A. F., Alenany A. M., and Saleh S. M., "Transmission Line Protection Using High-Speed Decision Tree and Artificial Neural Network : A Hardware Co-simulation Approach", Electr. Power Components Syst., vol. 49, no. 13-14, pp. 1181–1200, 2022, doi: 10.1080/15325008.2022.2050446.

[17] Obi O. E., Ezechukwu O. A., and Ezema C. N., "An Extended ANN-Based High Speed Accurate Transmission Line Fault Location for Double Phase to- Earth Fault on Non-Direct-Ground", Int. J. Eng. Sci. Technol., vol. 1, no. 1, pp. 31–47, 2017, doi: 10.29121/IJOEST.v1.i1.2017.04.

[18] Ankar S. and Yadav A., "Wavelet-ANN Based Fault Location Scheme for Bipolar CSC-Based HVDC Transmission System", First Int. Conf. Power, Control Comput. Technol., pp. 85–90, 2020.

[19] .Krishna GN., Raj J., and Chandran L. R., "Transmission Line Monitoring and Protection with ANN aided Fault Detection , Classification and Location", Proc. Second Int. Conf. Smart Electron. Commun., no. October, 2021, doi: 10.1109/ICOSEC51865.2021.9591911.

[20] De Santis E., Rizzi A., Sadeghian A., and Mascioli F. M. F., "A learning intelligent system for fault detection in smart grid by a one-class classification approach", Proc. Int. Joint Conf. Neural Netw. (IJCNN), pp. 1-8, Jul. 2015, doi: 10.1109/IJCNN.2015.7280756.

[21] Dhend M. H., and Chile R. H., "Innovative scheme for smart grid distribution SCADA system", 2015 IEEE 2nd International Future Energy Electronics Conference, IFEEC 2015. 2015. doi: 10.1109/IFEEC.2015.7361557.

[22] Bo Z. Q., Lin X. N., Wang Q. P., Yi Y. H., and Zhou F. Q., "Developments of power system protection and control", Prot. Control Mod. Power Syst., vol. 1, no. 1, pp. 1–8, 2016.

[23] Talaat M., Alsayyari A. S., Alblawi A., and Hatata A. Y., "Hybrid-Cloud-Based Data Processing for Power System Monitoring in Smart Grids", Sustain. Cities Soc., p. 102049, 2020, doi: 10.1016/j.scs.2020.102049.

[24] Industrial Control Systems: The Four Generations of SCADA Architectures (maderelectricinc.com). last accessed: April 18, 2022

[25] Church P., Mueller H., Ryan C., Gogouvitis S. V., Goscinski A., and Tari Z., "Migration of a SCADA system to IaaS clouds – a case study", 2017, doi: 10.1186/s13677-017-0080-5.

[26] Karalkar S. W. M., Adlok S., Ramteke A., Parchake S., Tawade R., and Manwar S., "Transmission Line Fault Detection using IoT", Int. J. Res. Eng. Sci. Manag., vol. 2, no. 3, pp. 1–3, 2019.

[27] Monica M., Sivakumar P., Sivani S., Sandhiya D., and Fathima J. S., "Overhead Line Fault Detection and Control using Internet of Things (IoT)", Bull. Sci. Res., pp. 17–21, 2020, doi: 10.34256/bsr2013.

[28] Wang J., De-Zhi, Liang, and Jun-Yan, "Design and Research of Overhead Line Fault Monitoring System for Medium Voltage Distribution Network Based on LoRa and NB-IoT", DEStech Trans. Comput. Sci. Eng., 2019, doi: 10.12783/dtcse/icaic2019/29401.

[29] Dhanalakshmi N., and Sunkari D. R., "Design of IoT Based Transmission Line Fault Monitoring System", Int. J. Electr. Eng. Technol., vol. 12, no. 6, pp. 336–345, 2021, doi: 10.34218/IJEET.12.6.2021.032.

[30] Tom R. J. and Sankaranarayanan S., "IoT based SCADA integrated with Fog for power distribution automation", Iber. Conf. Inf. Syst. Technol. Cist., pp. 1–4, 2017, doi: 10.23919/CISTI.2017.7975732.

[31] Don M., Rejoice T., Paul T., Vinod K., and Menon R., "Automatic Fault Detection and Location of Transmission Lines using IoT", no. May, pp. 295–299, 2019.

[32] Mohammed A. Y., "Low Cost Portable System for Converting Mosul Electrical Substations to Smart One' s", vol. 26, no. 2, pp. 323–339, 2021.

[33] Ahmad T. and Zhang D., "Using the internet of things in smart energy systems and networks", Sustain. Cities Soc., vol. 68, no. December 2020, p. 102783, 2021, doi: 10.1016/j.scs.2021.102783.

[34] Seabra J. C., Costa M. A., and Lucena M. M., "IoT based intelligent system for fault detection and diagnosis in domestic appliances", IEEE Int. Conf. Consum. Electron. - Berlin, ICCE-Berlin, vol. 2016-Octob, pp. 205–208, 2016, doi: 10.1109/ICCE-Berlin.2016.7684756.

[35] Reddy, Kishore R.D., Ramjee M., Sakpal, Amar P., Kiran, and Raghavendra M.V., "Automated Fault Recognition in Power Line Patrolling Using GSM and IoT Technology", Indian J. Public Heal. Res. Dev., vol. 9, no. 12, pp. 1548-1552, 2018, doi: 10.5958/0976-5506.2018.02076.4.

[36] Bandyopadhyay K., Ghosh D., Ghosh S., and Bhowmick S., "IoT Based Home Security System using Atmega328P, ESP01, and ThingSpeak Server", Int. Adv. Res. J. Sci. Eng. Technol., vol. 6, no. 5, 2019, doi: 10.17148/IARJSET.2019.6512.

[37] Oppenheim, Alan V., Ronald W. Schafer, and John R. Buck," Discrete-Time Signal Processing", 2nd Ed. Upper Saddle River, NJ: Prentice Hall, 1999.
[38] Perez R. S., Oviedo A. C., Peñaranda J. C., and Ramos G., "A novel fault classification method using Wavelet transform and Artificial Neural Networks", Proc. Int. Conf. Harmon. Qual. Power, ICHQP, vol. 2016-Decem, pp. 448–453, 2016.
[39] Decision Trees Compared to Regression and Neural Networks | DTREG
[40] Igual R. and Medrano C., "Research challenges in real-time classification of power quality disturbances applicable to microgrids: A systematic review", Renew. Sustain. Energy Rev., vol. 132, no. July, pp. 110050, 2020, doi.org/10.1016/j.rser.2020.110050
[41] Quinlan J. R., "Induction of decision trees", Mach. Learn., vol. 1, no. March, pp. 81–106, 1986, DOI: 10.1007/ BF00116251.
[42] Mohanty S. K., Karn A., and Banerjee S., "Decision Tree Supported Distance relay for Fault Detection and Classification in a series compensated line", 2020 IEEE Int. Conf. Power Electron. Smart Grid Renew. Energy, PESGRE 2020, pp. 1–6, 2020.
[43] Džeroski Ž. B. S., Panov P., "Machine Learning, Ensemble Methods in", Meyers R. Encycl. Complex. Syst. Sci. Springer, New York, NY, pp. 5317-5325, 2009, doi: 10.1007/978-0-387-30440-3_315.
[44] Sejdiu B., Ismaili F., and Ahmedi L., "Integration of semantics into sensor data for the IoT: A systematic literature review", International Journal on Semantic Web and Information Systems, vol. 16, no. 4, pp. 1–25, 2020.
[45] Hussain A. I., and Sayed A. Z., "Optimal User Association of LTE/Wi-Fi/Wi-Gig Bands in 5G Cellular Networks", International Journal on Semantic Web and Information Systems (IJSWIS), vol. 17, no. 2, pp. 22-40, 2021.
[46] "ESP8266 Overview". Espressif Systems. Retrieved 2017-10-02.
[47] Mikhail Grigorev. "Unofficial Development Kit for Espressif ESP8266 (GitHub Repository)".
[48] ESP8266 - Wikipedia.