Received: March 20, 2024. Revised: April 30, 2024.

INASS

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Detecting and Classifying Line-to-Line Fault in Photovoltaic Arrays Under Challenging Conditions Using Machine Learning Classifiers

Haider Abbas Hasan[1]* Mohanad Abd Shehab[2] Ammar Al-Gizi[2]

[1]*College of Engineering, University of Baghdad, Baghdad, Iraq*
[2]*Electrical Engineering Department, College of Engineering, Mustansiriyah University, Baghdad, Iraq*
*Corresponding author's Email: haider.el83@gmail.com

Abstract: Line-to-line fault (LLF) is considered one of the most common electrical faults occurring on the array side of the Photovoltaic System (PVS). This research proposes a diagnostic method to detect and classify LLFs under low mismatch levels, and higher fault impedances of up to 45 ohm when compared to other methods that did not exceed 25 ohm. The proposed method uses robust ML classifiers, namely Quadratic Discriminate Analysis (QDA) and Feed-Forward Neural Network (FNN), to design the required models. For this purpose, seven efficient features including array 2-voltages, array 2-currents, fill factor, maximum power to irradiance ratio, and voltage-temperature product, are extracted from the I-V characteristics curve of the employed Photovoltaic (PV) array. This is done under normal and faulty conditions, with a wide range of climate conditions. The faults are first detected by a detection module, then classified according to their mismatch levels by using classification module. Each module uses two Machine Learning (ML) models which represent the QAD and FNN classifiers. The results demonstrate that the proposed method can detect the LLFs, even under critical conditions of mismatch and impedance, with an average accuracy of 99.81% and 100% for QDA and FNN respectively. In addition, it classifies the severity of these faults with an average accuracy of 99.28% and 100% for both adopted models.

Keywords: Photovoltaic system, Line-to-Line fault, Fault detection and classification, Machine learning, Quadratic discriminate analysis, Feed-Forward neural network.

1. Introduction

PV is recognized as the best method of obtaining energy from the environment. Its market has the highest rate of growth on a global scale, due to its direct conversion of solar energy into electrical energy, pollution-free operation, long PV panel lifespan, lack of noise, installation in a variety of geographic locations, ease of maintenance, ability to be installed off-grid, and ability to connect to the power grid [1-3]. These benefits made the production of electricity using PV systems (PVSs) to increase rapidly. The PVS is divided into two main sections. The first is the DC side, which is primarily made of a solar PV array, which in turn consist of several PV modules, connected in series and parallel to produce the desired current and voltage for the inverter input. The second is the AC side, which is made of an inverter and the grid system [4, 5].

The PVS eventually experiences failures, as any other systems. These failures are typically caused by different PVS faults, which may occur in both the DC and AC sides [6, 7]. The majority of these faults take place in the DC side of the PVS, particularly in the PV array component [8]. Electrical faults are the most common in PV arrays. They include Short Circuit Fault (SCF), Open Circuit Fault (OCF), Arc Fault (AF), LLF, and Ground Fault (GF). The presence of these faults can lead to a reduction in the lifetime of the PV modules, a decrease in system power output creates significant safety hazards [9-11]. Hence, Fault Detection and Classification (FDC) in the PV array is a crucial task to mitigate potential risks and undesirable situations, that may occur during the operation of the PV array. It is imperative to promptly identify and rectify these

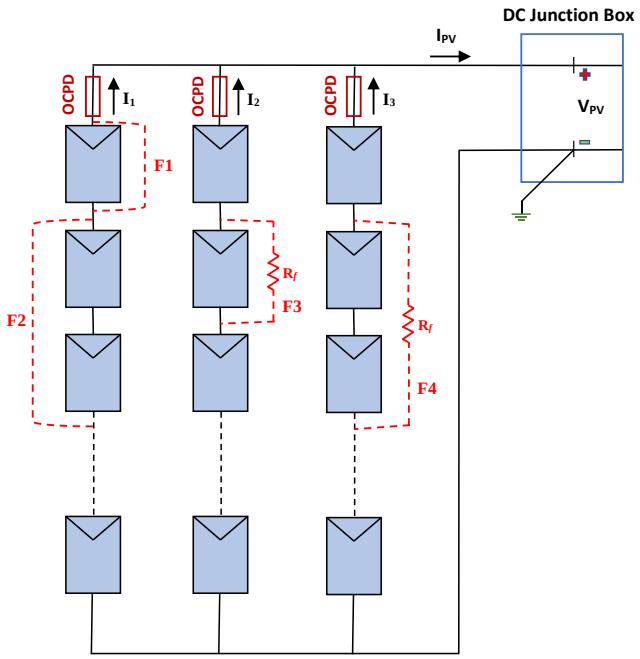


Figure. 1 10×3 PV array affected by four types of LLF

faults, to prevent the occurrence of unwanted scenarios [11, 12].

An LLF refers to an unintentional SCF between two points, which have different voltage potentials. This could happen due to corrosion, mechanical damage in the DC junction box, water intrusion, or wire chewing by rodents [4, 13]. The severity of the LLF is determined by the mismatch level (mismatch percentage), which is represented by the number of modules, which are impacted by an LLF in a string, to the number of hole modules in that string, or can be determined by the fault impedance value, which is defined by the impedance of the short-circuit path [14, 15].

Fig. 1 shows a 10x3 PV array, that is affected by four cases of LLF, two of them have 10% and 20% mismatch percentages, as in F1 and F2, respectively, and the others have the exact mismatch percentages but with fault impedance represented by R_f.

The presence of LLF causes a substantial voltage drop in the faulty string, causing a back-feeding current flow, to this string from the other non-affected ones. Higher mismatch levels in LLF result in higher fault current, due to the inclusion of more units in the fault loop. The same happens in the case of lower impedance values, where lower resistance values of the faulty path result in a larger reverse current passing through that path [10, 13].

To detect the LLF in the PV array, each string is fitted with a conventional protection device, as shown in Fig. 1, where an Over Current Protection Device (OCPD) is used, according to the National Electric Code (NEC), article 690 [16]. In practice, each OCPD is actually a fuse.

The fuse rating (I_f) of any OCPD is usually given by the following expression [14]:

$$I_f = 2.1\, I_{sc}^{STC} \tag{1}$$

Where I_{sc}^{STC} represents the string short circuit current at Standard Test Conditions (STC).

According to [15], the maximum faulty current (back-feeding current) through the string during an LLF is given by;

$$I_{\text{back}} = (n_s - 1) I_{SC}^{STC} \tag{2}$$

Where n_s is the number of strings in the PV array.

Although this conventional method has the benefits of being easy to implement with low cost, it is unable to detect the fault in the case of low back-feeding current values, that can be caused by a low mismatch level, and/or a high impedance of the shorted line. This is because the back-feeding current, which is produced from these cases, may not be able to melt the fuse [14, 15]. Other cases that make the fault undetectable, are the operation at low irradiance levels, and the presence of a blocking diode installed on each string of the PV array, which blocks the back-feeding current [4, 13, 17]. All these reasons make the detection of an LLF a big challenge.

Few studies attempt to detect and classify LLFs under conditions of low mismatch level and/or high fault impedance using ML methods. These methods include Decision Tree (DT) based [18], Graph-Based Semi-Supervised Learning (GBSSL) [19], Support Vector Machine (SVM) [10], Two-Stage SVM [20], Weighted Ensemble Learning-Based [21], multiple ML algorithms [15, 17], Deep Learning Method Based on Convolutional Neural Network (CNN) and/or Bidirectional Gated Recurrent Unit (Bi-GRU) [22], and Optimized SVM [23]. However, some of these studies don't involve critical values of mismatch and impedance [18, 19], where these studies did not include mismatch levels lower than 30% and fault impedances higher than 20 Ω. Other studies rely only on the detection approach, they showed low accuracy results, at low mismatch and high impedance conditions [10, 19, 20], e.g., [20] did not exceed 82.15% in the best case. Some studies could only classify the LLF as a part of other employed faults, where they are not concentrating on classifying this fault according to its mismatch levels [15, 21, 22]. Study [15] concentrates on the severity of fault impedances in LLF and ignores the

severity of mismatch levels, while [22] concentrates on the severity of mismatch levels and ignores the effect of impedance value. In addition, the [22] depended on deep learning that needs to a large dataset to train and learn the model. Other studies are able to get higher-accuracy results using Logistic Regression (LR), Navi bias (NB), and SVM classifiers, but with highest impedance value, which is at most 25 Ω [17, 21, 23]. Finally, almost all previous studies except [19] and [22] studies are common in using same ML techniques which are (DT, SVM, KNN, LR, and NB).

All the previously mentioned (except [19] and [22]) studies have in common that they used from the same algorithms group (DT, SVM, KNN, LR, and NB) in their methods and neglected the other algorithms.

This study proposes an intelligent method to detect and classify the LLF. The proposed method adopted challenging values of low mismatch levels, and high fault impedance values, under different climatic conditions, using efficient ML classifiers, FNN and QDA.

The main contributions of this study are:

- The primary features have been extracted under wide range of I-V characteristics of the PV array. MATLAB/Simulink-based model is utilized to differentiate between faulty and normal situations.
- New features denoted by γ_{mpp} and α were introduced to increase the diagnostic performance.
- Two modules have been joined, one for fault detection and the second for fault classification.
- Two different and robust ML classifiers were developed which are QDA and FNN.
- The suggested methods can detect and classify LLF with high accuracy and with low mismatch levels and high impedance challenging.

2. Fault detection and classification challenges

Since the I-V curve of any PVS is directly affected by any changes, that occur in the PV array, any fault condition may change the shape of that curve. However, detecting and classifying faults based on I-V curve changes is very difficult task, when compared to other methods. Distinguishing faulty and normal curves can be very challenging in some cases. For a better understanding, seven I-V curves of the PV array shown in Fig. 2 are recorded by simulating the array shown in Fig. 1 under a normal operation case and six different cases of LLF scenarios, with different values of mismatch and impedance, all these simulations are implemented under STC. These I-V curves are assigned from C0 to C6, as shown in Fig. 2.

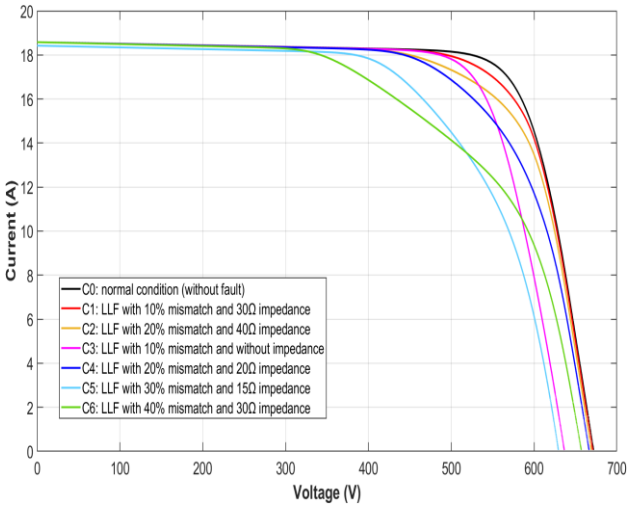


Figure. 2 I-V curves of normal condition and selective faulty cases

From Fig. 2, the following issues can be observed:

1) Curves represented by low mismatch levels and high impedances are too close to the normal curve C0 (especially C1 and C2), where it is a big challenge to distinguish between these faulty conditions and the normal condition.
2) Curves represented by low mismatch levels and low impedances, are very close to those represented by higher mismatch levels and higher impedances (C1 and C2, C3 and C4, C5 and C6), where it is considered another challenge to classify these faulty conditions from each other.

However, detecting and classifying these faults requires the extraction of efficient features from the I-V curve, collecting a sufficient dataset that covers many faulty cases under different conditions, and choosing a proper ML classifier for this dataset.

3. The proposed method

The proposed method is based on ML techniques to detect and classify the LLF, under challenging scenarios of mismatch level and fault impedance value. This method consists of two modules, the first has two classes, that represent the normal and faulty cases, where its primary objective is to detect the LLF case, through a binary classification process. The second module includes multiple classes, that represent the mismatch percentage severity, its job is to classify the detected LLF using a multiclass classification process, as shown in Fig. 3.

Each module involved two ML (QDA and FNN). The purpose of employing two classifiers is to compare their performance and select the best

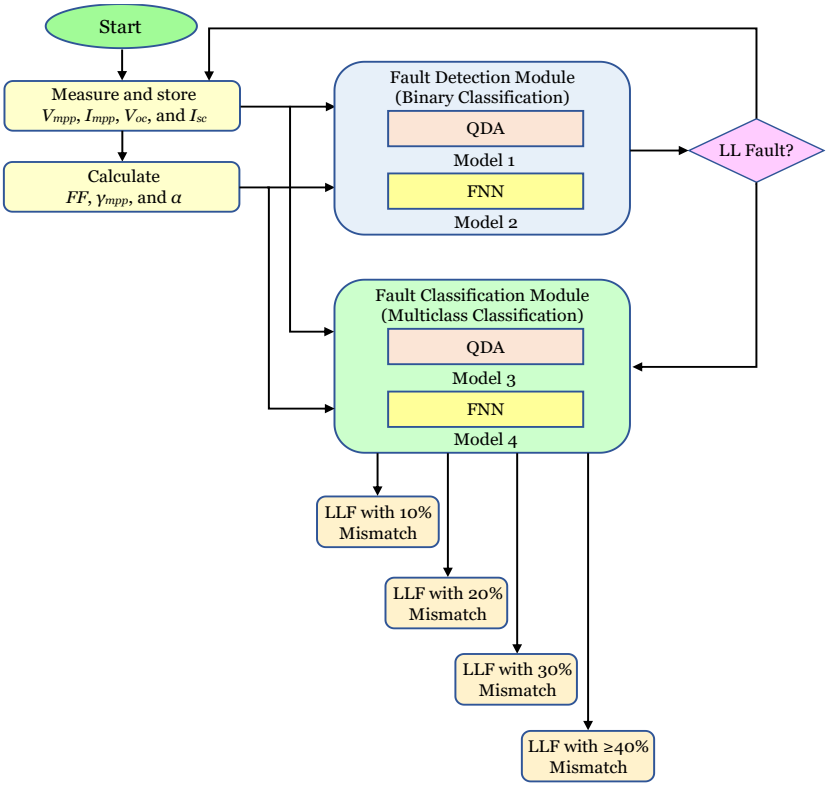


Figure. 3 The flowchart of the proposed method

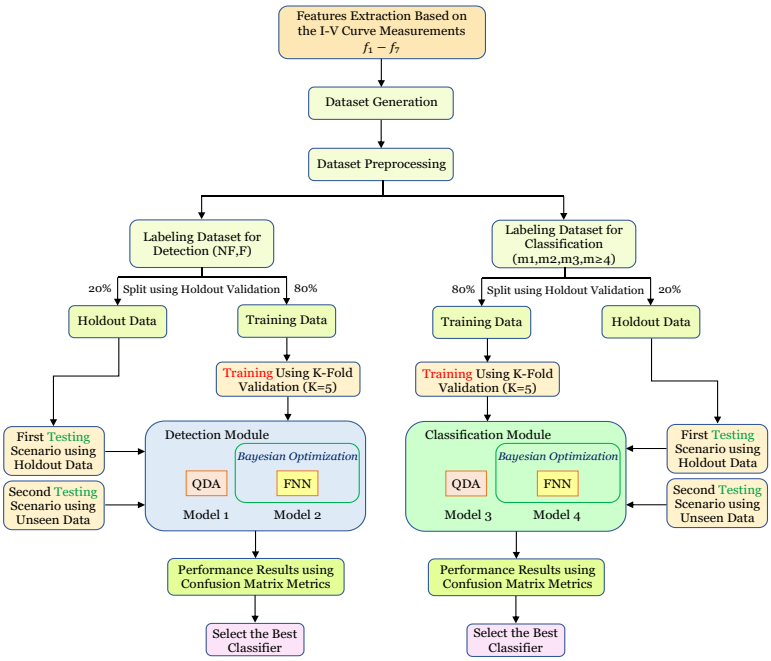


Figure. 4 The workflow structure of the two detection and classification modules

classifier in each module.

To implement these models, a set of features is extracted from the I-V curve of the PV array. These features are then used to collect the necessary training/testing dataset. The dataset is produced using MATLAB/Simulink, across several climate and fault scenarios. Subsequently, the dataset is labelled and subjected to preprocessing procedure, in order to facilitate its suitability for the subsequent training and testing phases.

The proposed models are trained using 80% of the dataset, while the remaining 20% are used for testing. In addition, an extra testing procedure is implemented, using another unseen dataset to assess the robustness, effectiveness, and generality of the proposed method. Finally, the performance of the

trained models is conducted using accuracy confusion matrix, and F1-score confusion matrix. The Bayesian Optimization (BO) method is adopted in this paper, to assign the optimal hyperparameters for the FNN, and hence achieve the best possible performance of the FNN model.

Fig. 4 shows the workflow structure of the proposed method. The subsequent subsections provide a more comprehensive elaboration of each individual phase.

3.1 Feature extraction

The initial step involves the extraction of appropriate features from the I-V curve. These features must possess certain characteristics, they must be capable of effectively describing a faulty scenario, and differentiating them from other complex scenarios, scalable for PVSs of different sizes, having smaller dimensions to enhance the training process, and conserve memory, and easily measurable. In this study seven distinct features are derived from the I-V curves of the employed PV array in the following manner:

- Features 1 to 4 ($f1$- $f4$): These features are regarded as the fundamental key elements of the PV array I-V curve, which are directly derived from it. These features are V_{mpp}, I_{mpp}, V_{oc}, and I_{sc}. Where V_{mpp} and I_{mpp} are the array voltage and current at the maximum power point of operation, respectively, while V_{oc} is the maximum voltage, that the array can produce with no load, and I_{sc} is the maximum current, that the array can deliver when the load is short-circuited.
- Feature 5 ($f5$): This feature is extracted from the previous four features. it is called the Fill Factor (FF), which determines the maximum power output from the array [10]. The FF is defined as,

$$FF = \frac{V_{mpp} \times I_{mpp}}{V_{oc} \times I_{sc}} \quad (3)$$

- Feature 6 ($f6$): This feature takes the irradiance (G) into account. It's called Gamma at MPP (γ_{mpp}), it's
- defined as the ratio of the output maximum PV power to the solar irradiance [15, 24], as given below.

$$\gamma_{mpp} = \frac{P_{mpp}}{G} = \frac{V_{mpp} \times I_{mpp}}{G} \quad (4)$$

- Feature 7 ($f7$): This feature (α) involves the surface temperature of the PV array (T). It is worth noting that the voltage of the PV array is directly influenced by T. Consequently, this feature is expressed by multiplying V_{mpp} and T as follows:

$$\alpha = V_{mpp} \times T \quad (5)$$

3.2 Dataset generation and labeling

Two different datasets are generated in this paper. The first is the original dataset (D), which is used for training and testing the proposed ML models. The second dataset is an unseen dataset (D_{unseen}), which is used to implement a second test process, to evaluate the generalization of these models. This is accomplished by modelling the PV array configuration as depicted in Fig. 1 using MATLAB/Simulink environment, to record and store the values of the seven features. The PV array consists of 3 parallel strings, each has 10 modules connected in series. The maximum power generated by this array is 9.5 kW under STC (Irradiance=1000 W/m^2 and Temperature=25^0 C). Each PV module is a type of SunPower SPR-315E-WHT-D with the following STC electrical specifications addressed in Table 1.

The D dataset is collected by simulating this model, under normal operation case which referred as F0. The other five cases of LLF are considered with different mismatch levels, namely F1 has 10% mismatch, F2 has 20%, F3 has 30%, F4 has 40%, and F5 50%. A total of 3960 samples are collected from this process. Note that each case of (F0-F5) is simulated under numerous scenarios of fault impedances, and climate conditions of irradiance and temperature. The D_{unseen} dataset has 324 samples, this dataset is collected by simulating the cases (F0-F5), and new two cases F6 of 60% and F7 of 70% mismatch, under different scenarios of fault impedances and climate conditions, when compared to the D dataset scenarios.

Each of the D and D_{unseen} datasets is divided into two sub-datasets. One is dedicated to the detection module, and the other is used for the classification module. In total, four sub-datasets are created, as detailed in Table 2. The sub-datasets associated with the detection module, contain all the acquired data

Table 1. Electrical specifications of the PV module

Maximum power (P_{max})	315.072 W
Open-circuit voltage (V_{oc})	64.6 V
Short-circuit current (I_{sc})	6.14 A
Voltage at maximum power point (V_{mp})	54.7 V
Current at maximum power point (I_{mp})	5.76 A

Table 2. The generated datasets with their labels for each module

Dataset type	Case of PV array	Irradiance (W/m^2)	Temp. (C)	Fault impedance (Ω)	No. of samples	Total no. of samples	Label
Detection module datasets							
D	Normal operation (F0)	100:100:1000	0:5:50	-	110	3960	NF
	LLFs (F1-F5)	100:100:1000	0:5:50	0:5:30	3850		F
D_{unseen}	Normal operation (F0)	150:200:550	2.5:10:22.5	-	9	324	NF
	LLFs (F1-F7)	150:200:550	2.5:10:22.5	3:10:33 & 45	315		F
Classification module datasets							
D	LLF (F1) (10% mismatch)	100:100:1000	0:5:50	0:5:30	770	3850	M10
	LLF (F2) (20% mismatch)	100:100:1000	0:5:50	0:5:30	770		M20
	LLF (F3) (30% mismatch)	100:100:1000	0:5:50	0:5:30	770		M30
	LLFs (F4 & F5) (40% & 50% mismatch)	100:100:1000	0:5:50	0:5:30	1450		M≥40
D_{unseen}	LLF (F1) (10% mismatch)	150:200:550	2.5:10:22.5	3:10:33 & 45	45	315	M10
	LLF (F2) (20% mismatch)	150:200:550	2.5:10:22.5	3:10:33 & 45	45		M20
	LLF (F3) (30% mismatch)	150:200:550	2.5:10:22.5	3:10:33 & 45	45		M30
	LLFs (F4-F7) (40%-70% mismatch)	150:200:550	2.5:10:22.5	3:10:33 & 45	180		M≥40

cases, while the sub-datasets employed in the classification module, include only the fault cases, i.e., excluding the normal case.

The detection module is designed to apply binary classification. As a result, the sub-datasets associated with this module, are labelled with only two classes: NF, which refers to the normal case F0, and F, which refers to the faulty cases (F1-F7). On the other hand, the classification module is developed as a multiclass classification system, in which its sub-datasets are assigned to four distinctive class labels. These labels are denoted as M10, which corresponds to the F1 case, M20 for F2, M30 for F3, and M≥40 for fault cases of mismatch values that equal or greater than 40% (F4-F7).

3.3 Data preprocessing

To increase the accuracy of the QDA and FNN classifiers and reducing their training cost, a data scaling method is needed, to reduce the difference in scale, among the feature values (attributes) in the dataset. Two main approaches are usually used for this purpose, namely normalization and standardization.

The standardization approach is applied in this research, due to its advantage in dealing with outliers, unlike the normalization approach [25]. Each attribute is standardized by the following equation [15]:

$$x' = \frac{x-\mu_x}{\sigma_x} \quad (6)$$

Where x' is the standardized attribute value, x is the original attribute value, μ_x and σ_x are the mean and standard deviation of x, respectively.

3.4 The employed classifiers

3.4.1. Quadratic discriminant analysis (QDA)

The Discriminant Analysis (DA) is a ML algorithm used for classification and dimensionality reduction. The primary goal of DA is to find the combination of features, that best separates the classes in a dataset. There are two types of DA, namely Linear Discriminant Analysis (LDA) and QDA. The QDA is an extension of LDA, that reduces the requirement of a shared covariance matrix across all classes. Conversely, every class is permitted to possess its individual covariance matrix. This provides greater flexibility for modelling complex relationships inside each class. The main purpose of QDA, is to probabilistically allocate a given observation(s) to the class with the highest likelihood. This is carried out using a quadratic discriminant function $g_k(x)$ as a classification rule, that estimates the parameters of a Gaussian distribution for each class [26, 27]. This function is obtained using the following algorithm [26]:

1) Construct a training dataset (X) consist of N samples (x) each consist of M variables (feature), and it is labeled with C classes (k).
2) Calculating the priori probability of each class (π_k) based on the training data:

$$\pi_k = \frac{n_k}{N} \quad , k = 1,2, \ldots\ldots, C \tag{7}$$

Where n_k is the number of samples in class k, and N is the total number of dataset samples.

3) Computing the mean vector for each class (μ_k) by the following equation:

$$\mu_k = \frac{1}{n_k} \sum_{i=1}^{n_k} x_i^k \tag{8}$$

Where x_i^k is the i-th sample in class k.

4) Calculating the covariance matrix $\sum_k$ for each class k as in Eq. (9).

$$\sum_k = \frac{1}{n_k - 1} \sum_{i=1}^{n_k} (x_i^k - \mu_k)(x_i^k - \mu_k)^T \tag{9}$$

5) Computing the quadratic discriminant function $g_k(x)$ for each class k:

$$g_k(x) = -\frac{1}{2}\log|\Sigma_k| - \frac{1}{2}(x - \mu_k)^T \Sigma_k^{-1}(x - \mu_k) + \log(\pi_k) \tag{10}$$

6) Finally, the classification process is accomplished by assigning the new sample x to the class with the highest discriminant function value:

$$\text{Class}(x) = \arg\max_k \, g_k(x) \tag{11}$$

As shown in Eq. (10), the decision boundary in QDA is quadratic, making it more flexible than the LDA, but it requires the estimation of more parameters. The algorithm is suitable when the assumption of equal covariance matrices, for all classes is not met, as in the datasets employed in this work. It is worth noting that the QDA does not possess any explicit user-configurable hyperparameters, that may be adjusted before the training. The algorithm computes the covariance matrices and mean vectors for each class, from the data through the training process.

3.4.2. Feed-forward neural network (FNN)

This network, also known as Multilayer Perceptron, is one of the main types of Artificial Neural Networks. It is widely employed in ML for classification tasks, due to its ability of modelling complex functions in data. The structure of the FNN is composed of neurons, also known as nodes, which are organized into layers. The initial layer is referred as the input layer, the last layer is known as the output layer, and the layers in between are the hidden layers. Every neuron in the hidden layers is fully connected with all the neurons in the preceding and succeeding layers. The connection between any two connected neurons is defined by a weight coefficient (w). Furthermore, each neuron in an individual layer (except the input layer) is characterized by a threshold coefficient, usually called the bias (b). The term "feed-forward" came from the fact, that the information inside this network flows only in one direction, moving only from the input nodes to the output nodes [28–30]. Fig. 5 shows the structure of the proposed FNN which is used for the classification task.

Note that the structure of the detection task FNN used in this paper only differs in the number of output classes, where it has only two classes (NF & F).

However, each neuron in the hidden and output layers has an activation function (*f*), which provides non-linearity function to the network, allowing it to model complex relation. The general output formula (z) of each hidden and output neuron is determined by Eq. (12) [29].

$$z = f\left(\sum_{i=1}^{N} w_i x_i + b\right) \tag{12}$$

Where, N is the number of input connections to the node, w_i is the weight associated with the i-th input connection, x_i is the input value from the i-th connection, b is the bias term associated with the node, and f is the activation function, which is applied as elementwise to the result.

The network is trained through these layers, to map input features to target class labels, through a process called supervised learning. The training of the FNN uses labelled data, to iteratively adjust the weights and biases, with a backpropagation process until the network can accurately predict the class labels for new unseen data [28, 31].

The FNN has many hyperparameters that must be set before the training process. The choice of hyperparameters significantly influences the performance and behaviour of the network [32]. These hyperparameters are represented by: (i) The number of hidden layers, (ii) The size of each hidden layer (number of neurons in each hidden layer), (iii) The choice of the activation function, (iv) The learning Rate, (v) The layer weights Initials, (vi) The layer biases Initials, (vii) The maximum number of training iterations, and (viii) Regularization strength.

In the proposed FNN model, some of these hyperparameters are manually set as follows:

- Learning Rate = 0.001

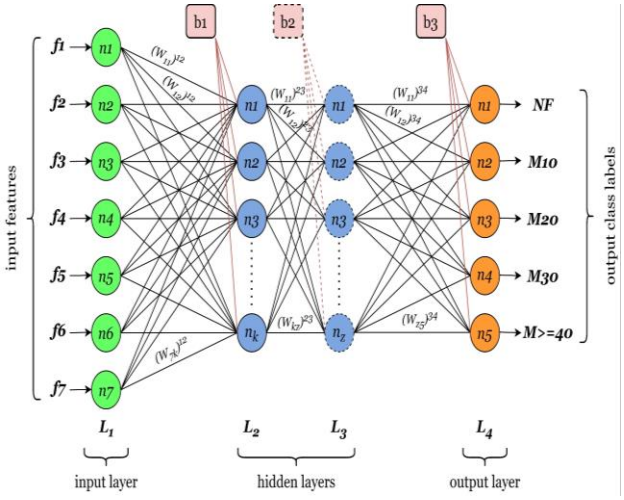


Figure. 5 FFN architecture used for LLF classification

- Layer weights Initializer = random values (0-1)
- Layer biases Initializer = zeroes
- Maximum number of training iterations = 1000
- The Activation function of the output layer = softmax

As shown in Fig. 5, the number of hidden layers and the number of neurons in each hidden layer are not initially set, because they are set using the BO method [33]. This technique is used to determine the ideal hyperparameter value for the FNN classifier and, which provides the best possible performance for the trained model. BO is acknowledged to be more effective than alternative techniques, at locating the ideal parameters in a reasonable length of time [34]. The activation function and the regularization strength are also set by using this optimizer. In this research the BO is used to select the optimal hyperparameter values from the following ranges and options:

- Number of hidden layers (1-2)
- Size of each hidden layer (1-100)
- Activation function of hidden layers (ReLU, tanh, and sigmoid).
- Regularization strength (nonnegative scalar between 0-1).

3.5 Training and testing

Before implementing the suggested detection and classification models, The D dataset as listed in Table 2, is divided into two sets of data using a Hold-Out Validation method [15], as illustrated in the workflow structure of the proposed method shown in Fig. 3. These sets are the training data (80% of D) and the holdout data (20% of D). The training data (D_{train}) is used to train the models, while the holdout data ($D_{holdout}$) is used for testing. Furthermore, another testing process is employed in this work by using the D_{unseen} as listed in Table 2.

3.5.1. Training

To avoid the overfitting problem in the training process and estimate its performance, a K-fold cross-validation (K-FCV) method [35] with k=5 is used in this paper. To train the QDA and FNN, in each module, a total of 3168 samples (D_{train}) (88 normal operation and 3080 LLF samples) are used, to train the detection module, whereas only 3080 samples of LLF cases are used to train the classification module.

The BO technique is used to determine the optimal values, of some hyperparameters of the FNN classifier through the training process.

3.5.2. Testing

Two testing scenarios are implemented in this paper. The first scenario is utilized to evaluate the performance of the trained models, using a portion of $D_{holdout}$ dataset. The second test scenario is used to validate the generalization performance of the proposed models, by examining their ability to detect faults by using the D_{unseen} dataset.

To test the trained QDA and FNN models in each module using the first scenario, a total of 792 samples $D_{holdout}$ (22 normal operation and 770 LLF samples) are used to test the detection module, whereas only 770 samples of LLF cases are used to test the classification module. On the other hand, to establish the second test scenario, a total of 324 samples of the D_{unseen} dataset (9 normal operation and 315 LLF samples) are used to test the detection module, whereas only 315 samples of LLF cases are used to test the classification module.

3.5.3. Performance metrics

A Confusion Matrix (CM) tool [36] is used to assess the performance of the ML models. The CM for binary classification problem is shown in Fig. 6.

Two confusion matrix (CM) metrics: classification accuracy and F1-score, have been used to assess the performance of the models.

- **Classification accuracy:** This metric represents the total number of correct predictions in each class divided by the total number of dataset samples. The accuracy can be calculated from the CM in Fig. 6 according to the following expression [37]:

$$Accuracy = \frac{TP}{TP+TN+FP+FN} \quad (13)$$

Where TP (True Positive) represents the numbers of data in class1 that are correctly predicted, TN (True Negative) represents the numbers of data

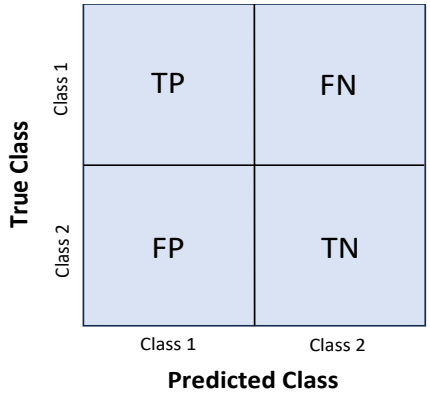


Figure. 6 The confusion matrix

in class2 which are correctly predicted, FP (False Positive) represents the numbers of data in class2 that are incorrectly predicted as class1, and FN (False Negative) represents the numbers of data in class1 that are incorrectly predicted as class2.

- **F1-score:** As seen in Table 2, the collected datasets are imbalanced, especially in the detection module. In this case, evaluating the performance using the classification accuracy metric alone is considered insufficient. F1-score is an efficient CM metric that is usually used to assess the performance of the ML classifiers in the case of an imbalanced dataset [38]. The F1-score is calculated from the CM as,

$$F1\ score = \frac{2\times Precision\times Recall}{Precision\times Recall} \quad (14)$$

Where *Precision* and *Recall* are another CM metrics that can be calculated from Eq. (15) and Eq. (16), respectively.

$$Precision = \frac{TP}{TP+FP} \quad (15)$$

$$Recall = \frac{TP}{TP+FN} \quad (16)$$

In this paper, a sample-weighted F1-score metric is used beside the accuracy metric for binary and multiclass classification problems to evaluate the performance of the four trained models. This metric can be computed as,

$$Weighted\ F1\ score = \sum_{i=1}^{N} w_i \times F1\ score_i \quad (17)$$

Where N is the number of classes in the dataset and w_i can be defined as,

$$w_i = \frac{number\ of\ data\ in\ class\ i}{Total\ number\ of\ dataset} \quad (18)$$

Table 3. The optimized hyperparameters of FNN models

Module type	Model no.	Optimized hyperparameters
Fault Det.	2	no. of hidden layers = 1 layer size =12 activation function =ReLU regularization strength=3.1989e-6
Fault Class.	4	no. of hidden layers = 1 layer size =20 activation function =Tanh regularization strength=1.004e-6

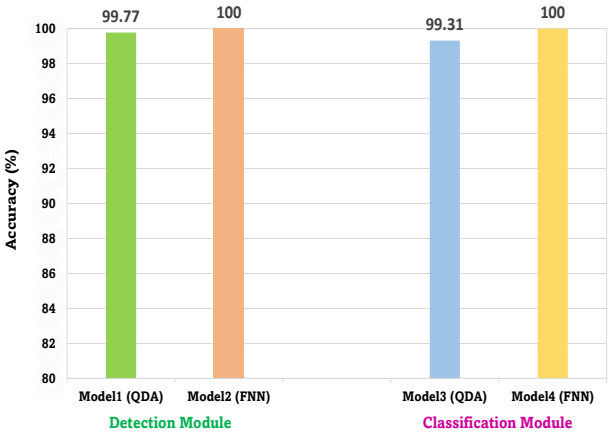


Figure. 7 The training accuracy of the four models

4. Results and discussion

4.1 Training results

The optimized parameters of the FNN models are listed in Table 3. The BO methods sets different parameters, to train each FNN classifier in both modules, except for the number of hidden layers, which is set with only one layer for both modules, giving the classifiers faster learning speed, than the two- or three-layers scenario.

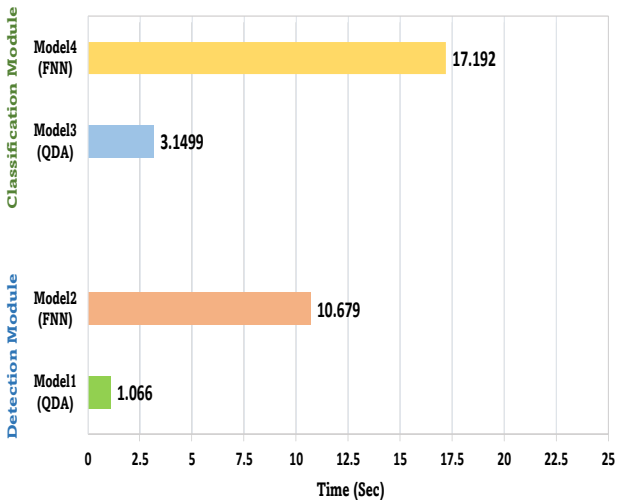


Figure. 8 The training time of each model

Table 4. Performance results of the proposed method using the two testing scenarios

Model type	Model no.	Classifier	F1-score		Average F1-score	Accuracy		Average accuracy
			1'st test scenario	2'st test scenario		1'st test scenario	2'st test scenario	
Fault detection	1	QDA	99.74	98.6	99.17%	99.62%	100%	99.81
	2	FNN	100%	100%	100%	100%	100%	100%
Fault classification	3	QDA	98.56%	100%	99.28%	98.57%	100%	99.28%
	4	FNN	100%	100%	100%	100%	100%	100%

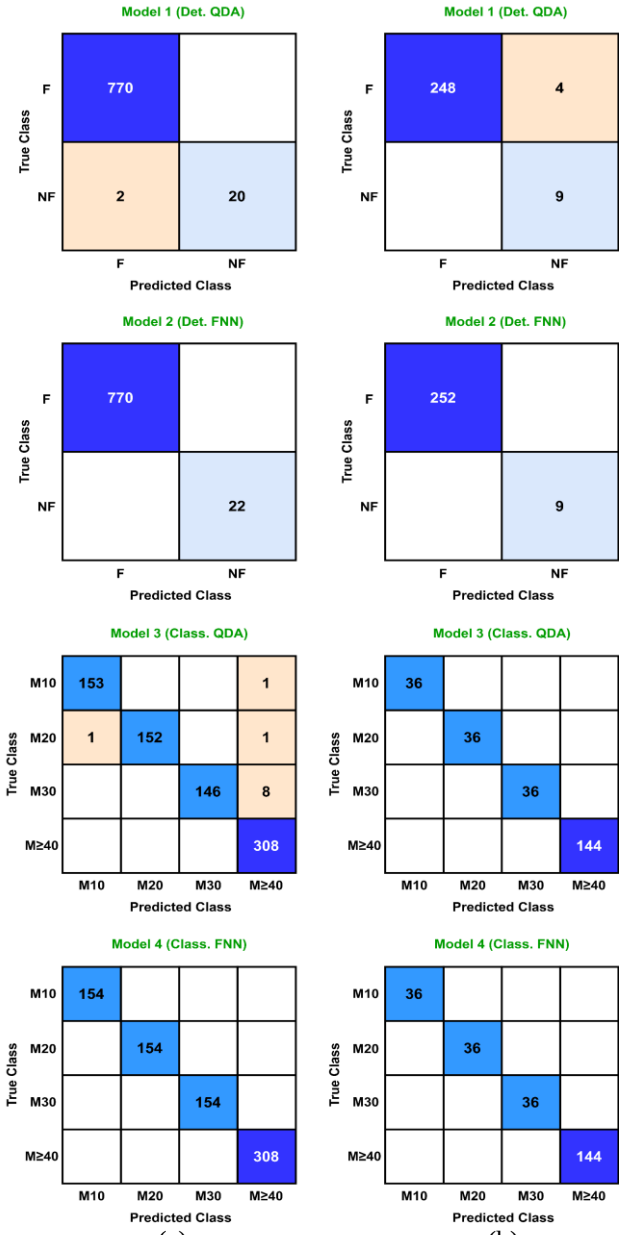
Fig. 7 shows the training accuracy (k-fold cross-validation accuracy) of the four models. The training accuracy reflects how well the model fits the training data. It gives an estimation of how well it generalizes to unseen data. From Fig. 7, it can be seen that the FNN classifier outperforms the QDA classifier, in terms of training accuracy for both detection and classification tasks with a training accuracy of 100%, making it more promising for detecting unseen faults.

Fig. 8 shows the amount of time, that the ML models take to learn from the training datasets. It is worth noting that the training time of the classification task is lower than the training time of the detection task. This can be attributed to the training datasets in classification modules, being larger than in the dataset in the detection modules, the multiclass classification learning process usually takes more time, than the binary classification process, and the FNN classifier in the classification module has more layers, than the detection module. In addition, The QDA classifier has a lower training time, than the FNN classifier in both modules.

4.2 Testing results

As mentioned before, the performance of the four trained models is evaluated based on two CM metrics: classification accuracy and F1-score. The accuracy and weighted F1-score of the four trained models for the two testing scenarios are summarized in Table 4, and the corresponding CMs are illustrated in Fig. 9.

As can be seen from the table, the proposed method was able to detect the faulty cases (F) with an average F1-score equal to 99.17% using the QDA classifier and 100% using the FNN classifier, while it could classify these faulty cases according to their mismatch severities (M=10, M=20, M=30, M≥40) with an average F1-score equal to 99.28% for the QDA classifier and 100% for the FNN classifier.



(a) (b)

Figure. 9 The testing CMs: (a) using hold-out data and (b) using unseen data

Table 5. Comparative results between the proposed method and other methods reported in the literature

Case study	Fault cases	Included severities of mismatch (mm) and impedance (R_f) in LLF	Used ML technique	Number of features	Accuracy
[15]	LLF OCF ARCF PSF MPPTF	mm = 25% R_f = 0-30 Ω	DT, KNN, SVM	4	Tested by holdout data Det = 100% SVM Class = 89.84% SVM
[22]	LLF OCF PSF	Mm = 25%, 50%, and 75% R_f = 0 Ω	CNN, Bi-GRU	3	Tested by holdout data Det = 99.46% Class = 100%
[21]	LLF OCF	mm = 16%, 33%, and 50% R_f = 0-25 Ω	weighted ensemble learning (LR,SVM, KNN)	16	Tested by holdout data Class = 98.61%
[17]	LLF GLF	mm = 10%, 20%, and >20% R_f = 0-25 Ω	(LR, NB, SVM) with Sequential Forward Search approach for feature selection	Det. LR=8, NB=3 SVM=2 Class. LR=4, NB=5 SVM=2	Tested by holdout data Det = 98 % NB Class = 96.4 % LR Tested by unseen data Det=97.67% NB Class=98.33% LR
[23]	LLF	mm = 10%, 20%, and ≥30% R_f = 0-25 Ω	(SVM) with a genetic algorithm for feature selection	Det. = 3 Class = 2	Tested by holdout data Det = 100% Class = 99.12% Tested by unseen data Det=100% Class=100%
Proposed method	LLF	mm = 10%, 20%, 30%, and ≥40% R_f = 0-45 Ω	QDA FNN	7	Tested by holdout data Det = 100% FNN Class = 100% FNN Tested by unseen data: Det = 100% FNN Class = 100% FNN

In contrast, the proposed method can detect the faulty cases F with an average accuracy of 99.81%, using the QDA model and 100% using the FNN model, while it can classify these faulty cases with an average accuracy of 99.28%, for the QDA model and 100%, for the FNN model.

From the testing result, the following conclusions can be addressed:

1) All the models exhibit high F1-score results despite being trained with imbalanced datasets.
2) The proposed method works efficiently, even with new fault cases, that are represented by the D_{unseen} dataset.
3) Both classifiers show a notable level of F1-score and accuracy in detecting and classifying the LLF, even under challenging conditions of low levels of mismatch, and high fault impedance up to 45 Ω. Nevertheless, the FNN classifier demonstrates a notable advantage over the QDA classifier, by achieving a superior accuracy of 100%.

4.3 Comparison with other methods

The effectiveness of the proposed method is compared to the outcomes of five different methods introduced in Section 1 [15, 22, 21, 17, 23], to provide additional validation for the proposed method. The comparison is based on various qualitative factors summarized in Table 5. Note that in studies that use more than one classifier (like [15, 17], and the proposed method), only the result of the classifier of the highest accuracy is listed in this table.

In addition to the implementation simplicity, Table 5 demonstrates that the proposed approach can handle the following challenges points:

1) The mismatch level classes included four different mismatches cases (M=10, M=20, M=30, M≥40) compared to other methods that did not exceed three classes; hence, it introduced a higher challenge in the classification task.

2) The values of fault impedance can reach up to 45 Ω compared to other mentioned methods, which did not exceed 30 Ω.
3) The performance accuracy is 100% in both testing scenarios using the FNN classifier compared to other methods.

In short, the proposed method was effectively able to detect and classify the LLF under several situations involving low mismatch levels and high fault impedances compared to the other reported methods listed in Table 5.

5. Conclusion

Detecting and classifying the LLF in the PVS under cases of low mismatch level and high fault impedance represent a big challenge. Little attention has been paid to the diagnosis of the LLF, that considers this scenario. This paper presents a robust ML method to detect and classify the LLF, under challenging conditions of mismatch and impedance. Two ML classifiers QDA and FNN are used in this work. These classifiers are employed to design four ML models: two for the detection module, and two for the classification module. The faulty cases are first distinguished from the normal case by the detection module, then the faults are classified by the classification module, according to their mismatch severity. The models are trained using a training dataset, extracted from seven key features of the PV array I-V curves, under normal operation and challenging LLF conditions, with different climates. The efficiency of this method is evaluated by testing the trained models with two testing datasets: a portion of the original collected dataset and a new unseen dataset. The proposed method can detect and classify the LLF under challenging scenarios of mismatch level, and fault impedance with a high average accuracy of 100% using the FNN model. The method outperforms the other methods reported in the literature, as it achieves better level of accuracy in diagnosing the LLF under higher fault impedance of 45 Ω.

Conflicts of Interest

The authors declare no conflict of interest.

Author Contributions

The first author has contributed substantially in this work in the methodology, software, validation, formal analysis, writing original draft preparation, writing review and editing. In contrast, the contributions of second and third authors are represented by visualization, supervision, work administration, and funding acquisition.

Acknowledgments

The authors would like to thank Mustansiriyah University (www.uomustansiriyah.edu.iq), Baghdad-Iraq, for its support in the present work.

References

[1] M. S. Arani and M. A. Hejazi, “The Comprehensive Study of Electrical Faults in PV Arrays”, *Journal of Electrical and Computer Engineering*, Vol. 2016, pp. 8712960, 2016.

[2] A. Al-Gizi, S. Al-Chlaihawi, M. Louzazni, and A. Craciunescu, “Genetically Optimization of an Asymmetrical Fuzzy Logic Based Photovoltaic Maximum Power Point Tracking Controller”, *Advances in Electrical and Computer Engineering*, Vol. 17, No. 4, pp. 69-76, 2017.

[3] H. Al-Zubaidi, M. A. Shehab, and A. Al-Gizi, “Electrical Fault Diagnosis of Solar PV Array Using Machine Learning Techniques”, In: *Proc. of 2023 13th International Symposium on Advanced Topics in Electrical Engineering (ATEE)*, Bucharest, Romania, pp. 1-4, 2023.

[4] S. Chandrasekharan, S. K. Subramaniam, and B. Natarajan, “Current Indicator Based Fault Detection Algorithm for Identification of Faulty String in Solar PV System”, *IET Renewable Power Generation*, Vol. 15, No. 7, pp. 1596-1611, 2021.

[5] A. Al-Gizi, A. Craciunescu, M. A. Fadel, and M. Louzazni, “A New Hybrid Algorithm for Photovoltaic Maximum Power Point Tracking under Partial Shading Conditions”, *Revue Roumaine des Sciences Techniques Serie Electrotechnique et Energetique*, Vol. 63, No. 1, pp. 52-57, 2018.

[6] Y. Zhao, J. F. De Palma, J. Mosesian, R. Lyons, and B. Lehman, “Line-Line Fault Analysis and Protection Challenges in Solar Photovoltaic Arrays”, *IEEE Transactions on Industrial Electronics*, Vol. 60, No. 9, pp. 3784-3795, 2013.

[7] Y. Y. Hong and R. A. Pula, “Methods of Photovoltaic Fault Detection and Classification: A Review”, *Energy Reports*, Vol. 8, pp. 5898-5929, 2022.

[8] S. R. Madeti and S. N. Singh, “A Comprehensive Study on Different Types of Faults and Detection Techniques for Solar

Photovoltaic System", *Solar Energy*, Vol. 158, pp. 161-185, 2017.

[9] S. T. Kebir, N. Cheggaga, A. Ilinca, and S. Boulouma, "An Efficient Neural Network-Based Method for Diagnosing Faults of PV Array", *Sustainability*, Vol. 13, No. 11, pp. 6194, 2021.

[10] Z. Yi and A. H. Etemadi, "A Novel Detection Algorithm for Line-to-Line Faults in Photovoltaic (PV) Arrays Based on Support Vector Machine (SVM)", In: *Proc. of 2016 IEEE Power and Energy Society General Meeting (PESGM)*, Boston, MA, USA, pp. 1-4, 2016.

[11] T. Pei and X. Hao, "A Fault Detection Method for Photovoltaic Systems Based on Voltage and Current Observation and Evaluation", *Energies*, Vol. 12, No. 9, pp. 1712, 2019.

[12] S. Gong, X. Wu, and Z. Zhang, "Fault Diagnosis Method of Photovoltaic Array Based on Random Forest Algorithm", In: *Proc. of 39th Chinese Control Conference (CCC)*, Shenyang, China, pp. 4249-4254, 2020.

[13] A. Y. Appiah, X. Zhang, B. B. K. Ayawli, and F. Kyeremeh, "Review and Performance Evaluation of Photovoltaic Array Fault Detection and Diagnosis Techniques", *International Journal of Photoenergy*, Vol. 2019, pp. 6953530, 2019.

[14] D. S. Pillai and N. Rajasekar, "A Comprehensive Review on Protection Challenges and Fault Diagnosis in PV Systems", *Renewable and Sustainable Energy Reviews*, Vol. 91, pp. 18-40, 2018.

[15] M. M. Badr, M. S. Hamad, A. S. Abdel-Khalik, R. A. Hamdy, S. Ahmed, and E. Hamdan, "Fault Identification of Photovoltaic Array Based on Machine Learning Classifiers", *IEEE Access*, Vol. 9, pp. 159113-159132, 2021.

[16] F. P. Hartwell, J. F. McPartland, and B. J. McPartland, *McGraw-Hill's National Electrical Code 2017 Handbook*, Ed. 29th, McGraw-Hill Education, New York, 2017.

[17] A. Eskandari, J. Milimonfared, and M. Aghaei, "Fault Detection and Classification for Photovoltaic Systems Based on Hierarchical Classification and Machine Learning Technique", *IEEE Transactions on Industrial Electronics*, Vol. 68, No. 12, pp. 12750-12759, 2021.

[18] Y. Zhao, L. Yang, B. Lehman, J. F. de Palma, J. Mosesian, and R. Lyons, "Decision Tree-Based Fault Detection and Classification in Solar Photovoltaic Arrays", In: *Proc. of 2012 Twenty-Seventh Annual IEEE Applied Power Electronics Conference and Exposition (APEC)*, Orlando, FL, pp. 93-99, 2012.

[19] Y. Zhao, R. Ball, J. Mosesian, J. F. de Palma, and B. Lehman, "Graph-Based Semi-Supervised Learning for Fault Detection and Classification in Solar Photovoltaic Arrays", *IEEE Transactions on Power Electronics*, Vol. 30, No. 5, pp. 2848-2858, 2015.

[20] Z. Yi and A. H. Etemadi, "Line-to-Line Fault Detection for Photovoltaic Arrays Based on Multi-Resolution Signal Decomposition and Two-Stage Support Vector Machine", *IEEE Transactions on Industrial Electronics*, Vol. 64, No. 11, pp. 8546-8556, 2017.

[21] A. Eskandari, M. Aghaei, J. Milimonfared, and A. Nedaei, "A Weighted Ensemble Learning-Based Autonomous Fault Diagnosis Method for Photovoltaic Systems Using Genetic Algorithm", *International Journal of Electrical Power & Energy Systems*, Vol. 144, pp. 108591, 2023.

[22] A. F. Amiri, S. Kichou, H. Oudira, A. Chouder, and S. Silvestre, "Fault Detection and Diagnosis of a Photovoltaic System Based on Deep Learning Using the Combination of a Convolutional Neural Network (CNN) and Bidirectional Gated Recurrent Unit (Bi-GRU)", *Sustainability*, Vol. 16, No. 3, pp. 1012, 2024.

[23] A. Eskandari, J. Milimonfared, M. Aghaei, and A. H. Reinders, "Autonomous Monitoring of Line-to-Line Faults in Photovoltaic Systems by Feature Selection and Parameter Optimization of Support Vector Machine Using Genetic Algorithms", *Applied Sciences*. Vol. 10, No. 16, pp. 5527, 2020.

[24] S. Rao, A. Spanias, and C. Tepedelenlioglu, "Solar Array Fault Detection Using Neural Networks", In: *Proc. of 2019 IEEE International Conference on Industrial Cyber Physical Systems (ICPS)*, Taipei, Taiwan, pp. 196-200, 2019.

[25] S. U. Jan and I. Koo, "A Novel Feature Selection Scheme and a Diversified-Input SVM-Based Classifier for Sensor Fault Classification", *Journal of Sensors*, Vol. 2018, pp. 7467418, 2018.

[26] A. Tharwat, "Linear vs. Quadratic Discriminant Analysis Classifier: A Tutorial", *International Journal of Applied Pattern Recognition*, Vol. 3, No. 2, pp. 145-180, 2016.

[27] W. Wu, Y. Mallet, B. Walczak, W. Penninckx, D. L. Massart, S. Heuerding, and F. Erni, "Comparison of Regularized Discriminant Analysis, Linear Discriminant Analysis and Quadratic Discriminant Analysis, Applied to

NIR Data", *Analytica Chimica Acta*, Vol. 329, No. 3, pp. 257-265, 1996.
[28] D. Svozil, V. Kvasnicka, and J. Pospichal, "Introduction to Multi-Layer Feed-Forward Neural Networks", *Chemometrics and Intelligent Laboratory Systems*, Vol. 39, No. 1, pp. 43-62, 1997.
[29] S. Razavi and B. A. Tolson, "A New Formulation for Feedforward Neural Networks", *IEEE Transactions on neural networks*, Vol. 22, No. 10, pp. 1588-1598, 2011.
[30] M. A. Shehab and K. Nihan, "Optimum, Projected, and Regularized Extreme Learning Machine Methods with Singular Value Decomposition and L-2-Tikhonov Regularization", *Turkish Journal of Electrical Engineering and Computer Sciences*, Vol. 26, No. 4, pp. 1685-1697, 2018.
[31] M. A. Shehab and N. Kahraman, "A Weighted Voting Ensemble of Efficient Regularized Extreme Learning Machine", *Computers & Electrical Engineering*, Vol. 85, pp. 106639, 2020.
[32] E. Elakkiya and S. Selvakumar, "Stratified Hyperparameters Optimization of Feed-Forward Neural Network for Social Network Spam Detection (SON2S)", *Soft Computing*, Vol. 26, No. 21, pp. 11915-11934, 2022.
[33] J. Wu, X. Y. Chen, H. Zhang, L. D. Xiong, H. Lei, and S. H. Deng, "Hyperparameter Optimization for Machine Learning Models Based on Bayesian Optimization", *Journal of Electronic Science and Technology*, Vol. 17, No. 1, pp. 26-40, 2019.
[34] H. Cho, Y. Kim, E. Lee, D. Choi, Y. Lee, and W. Rhee, "Basic Enhancement Strategies When Using Bayesian Optimization for Hyperparameter Tuning of Deep Neural Networks", *IEEE Access*, Vol. 8, pp. 52588-52608, 2020.
[35] S. Yadav and S. Shukla, "Analysis of K-Fold Cross-Validation over Hold-Out Validation on Colossal Datasets for Quality Classification", In: *Proc. of 2016 IEEE 6th International Conference on Advanced Computing (IACC)*, Bhimavaram, India, pp. 78-83, 2016.
[36] S. Visa, B. Ramsay, A. L. Ralescu, and E. V. D. Knaap, "Confusion Matrix-Based Feature Selection", In: *Proc. of 22nd Midwest Artificial Intelligence and Cognitive Science Conference*, Cincinnati, Ohio, USA, pp. 120-127, 2011.
[37] Ž. Vujović, "Classification Model Evaluation Metrics", *International Journal of Advanced Computer Science and Applications*, Vol. 12, No. 6, pp. 599-606, 2021.
[38] H. He and Y. Ma, *Imbalanced Learning: Foundations, Algorithms, and Applications*, Wiley-IEEE Press, New Jersey, 2013.