IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 3, 2021, accepted November 21, 2021, date of publication November 25, 2021,
date of current version December 7, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3130889

Fault Identification of Photovoltaic Array Based on Machine Learning Classifiers

MOHAMED M. BADR[1], MOSTAFA S. HAMAD[2], (Senior Member, IEEE), AYMAN S. ABDEL-KHALIK[1], (Senior Member, IEEE), RAGI A. HAMDY[1], (Senior Member, IEEE), SHEHAB AHMED[3], (Senior Member, IEEE), AND EMAN HAMDAN[4]

[1]Department of Electrical Engineering, Alexandria University, Alexandria 21544, Egypt
[2]Department of Electrical and Control Engineering, Arab Academy for Science, Technology and Maritime Transport, Alexandria 1029, Egypt
[3]CEMSE Division, King Abdullah University of Science and Technology, Thuwal 23955, Saudi Arabia
[4]Department of Marine Engineering Technology, Arab Academy for Science, Technology and Maritime Transport, Alexandria 1029, Egypt

Corresponding author: Mohamed M. Badr (mmustafabadr@gmail.com)

This work was supported by the Information Technology Industry Development Agency's (ITIDA) Information Technology Academia Collaboration (ITAC) Collaborative Funded Project through Advanced Research Projects (ARP) under Grant ARP2020.R28.18.

ABSTRACT Fault identification in Photovoltaic (PV) array is a contemporary research topic motivated by the higher penetration levels of PV systems in recent electrical grids. Therefore, this work aims to define an optimal Machine learning (ML) structure of automatic detection and diagnosis algorithm for common PV array faults, namely, permanent (Arc Fault, Line-to-Line, Maximum Power Point Tracking unit failure, and Open-Circuit faults), and temporary (Shading) under a wide range of climate datasets, fault impedances, and shading scenarios. To achieve the best-fit ML structure, three distinct ML classifiers are compared, namely, Decision Tree (DT) based on different splitting criteria, K-Nearest Neighbors (KNN) based on the different metrics of distance and weighting functions, and Support Vector Machine (SVM) based on different Kernel functions and multi-classification approaches. Also, Bayesian Optimization is adopted to assign the optimal hyperparameters to the fault classifiers. To investigate the performance of classifiers reported, both simulation and experimental case studies are carried out and presented.

INDEX TERMS Photovoltaic array faults, machine learning, decision tree, nearest neighbors, support vector machine, Bayesian optimization.

I. INTRODUCTION

THE SOLAR PHOTOVOLTAIC (PV) industry has experienced significant growth over the past years due to the technology's clear economic and environmental benefits. The world's net electricity generation from grid-connected PV systems is expected to rise from 34 billion kilowatt-hours in 2010 to 452 billion kilowatt-hours in 2040 [1]. Although PV systems don't incorporate moving parts and usually require low maintenance, they are still subjected to diverse faults across the various system components (e.g., PV generator (i.e., module, string, or array), power-processing stage(s), batteries, and/or the utility grid) [2]–[4]. While the scalability of PV technology is an advantage, it also poses additional challenges. Once PV modules are electrically connected, any fault among the modules can affect the entire system's performance. Proper fault detection and/or identification may is thus necessary to avoid significant energy generation loss and large capital expenditures. Large solar PV plants composed of series-parallel PV modules configurations (i.e., PV array) also exhibit higher voltage and current ratings, leading to a higher risk of large fault currents or DC arcs. Thus, undetected PV array faults may not only cause power losses, but also may lead to safety issues, PV array/system degradation, and/or fire hazards.

The PV array is commonly subjected to a variety of faults including, Partial Shading (PS) conditions, Maximum Power Point Tracking (MPPT) unit failure, PV module hot spots/micro-cracks and electrical faults (Open-Circuit (OC), Line-to-Line (LL), Line-to-Ground (LG), and Arc Fault (AF)) [5]. These faults contribute to energy losses and/or system degradation, which, in turn, increase maintenance

The associate editor coordinating the review of this manuscript and approving it for publication was Guangya Yang.

costs or fire hazards. For instance, a survey conducted in the U.K. [6] showed that PV system faults caused an estimated energy loss of 18.9%. This supports the need for an effective monitoring system to alert system operators of such faults. Another example is that of a large PV power plant in California, U.S.A, where an electrical fire occurred on April 30, 1987, due to a LL fault, which completely destroyed the power converter [7]. The downtime energy loss was estimated to be 180 k$. The replacement of the power converter including mandatory improvements to the protection system costed approximately 300 k$. In a physical solar PV array, identification of the aforesaid faults incorporates conventional protection systems such as Ground Fault Protection Devices (GFPDs), Overcurrent Protection Devices (OCPDs), and Arc-Fault Circuit Interrupters (AFCIs), as stated in the U.S.A. National Electrical Code® (NEC®), in article 690 [8]. While such protective devices are necessary, they provide limited diagnostic capability, hence, further analysis is needed to discriminate amongst the different types of faults [3], [5], [9]. For instance, the OCPD may fail to detect fault currents due to several reasons including [3], [5]: nonlinear output characteristics of PV array, the effect of MPPT controller on the fault current magnitude, solar irradiance conditions, the challenge to detect a LL fault in the case of PS or when utilizing blocking diodes, and the difficulty to clear a fault current in case of a high fault impedance and small mismatched fault locations. Thus, fault localization/discrimination using traditional troubleshooting requires monumental time, which necessitates the employment of an efficient monitoring system that is capable of determining fault type and monitoring PV system architecture, which is accelerating system recovery after fault clearance.

In general speaking, the monitoring system architecture can be divided into three stages: 1) data acquisition, which is an essential stage for obtaining an accurate and reliable database, 2) the pre-processing stage of the measured quantities, and 3) the Fault Detection and Diagnosis (FDD) technique, which aims to **detect** (i.e., discriminate between healthy and fault conditions) and **diagnose** (i.e., distinguish one fault type from another) faults. The PV array/system size, operation, and maintenance cost determine the most appropriate FDD technique to be recruited. Based on the available literature [10]–[29], the techniques applied in PV array faults identification could be categorized as Signal Processing Techniques (SPTs), Artificial Intelligence (AI)-based techniques, and Hybrid Techniques (HTs).

The SPTs utilize real-time sensed data such as solar irradiance and temperature with the sensed data from the PV system. The SPTs may employ a predefined threshold(s) to compare measured quantities with expected quantities from simulation, or to analyze measured quantities in order to generate the fault signal(s) [10]–[16]. The SPTs commonly suffer from: 1) defining a predefined threshold(s), 2) the lack of model updates has a bad influence on a predefined threshold(s), as system parameters change with seasonal variations, 3) inaccurate simulation models can affect the role of the detection and/or diagnosis module(s), 4) false tripping signal under low irradiance or shading scenarios, and/or 5) complex implementation and cost-effectiveness.

Similar to SPTs, the AI-based techniques employ real-time sensed data, and then adopt one of AI approaches such as Fuzzy Logic Control (FLC), Neural Network (NN), Decision Tree (DT), K-Nearest Neighbors (KNN), or Support Vector Machine (SVM) to identify the type of fault(s) [17]–[22].

HTs use a combination of SPTs and/or AI-based techniques or employ some modifications to existing techniques to enhance the role of the monitoring system [23]–[29].

The AI-based techniques, specifically, Machine Learning (ML) classifiers and HTs have shown high detection and diagnostic accuracy for PV array faults under different scenarios compared to SPTs. Although the HTs can achieve high classification accuracy, they suffer from implementation complexity and a lack of intuitive visualization. Thus, AI-based techniques are adopted in this research. Specifically, in the PV array fault detection and diagnosis, the DT, KNN, and SVM classifiers have proven their effectiveness in most of the available literature. However, most of the works done have recruited these classifiers to detect and diagnose only a subset of the faulty cases, while overlooking some other severe faults under different scenarios such as hybrid faults (permanent and temporary faults), low location mismatch, and high impedance faults. On the other hand, the influence of different setups on the classifier behavior has been overlooked, such as splitting criteria in DT, distance metrics as in KNN, and Kernel functions as in SVM. Moreover, the available studies have not shown how different hyperparameters for the mentioned classifiers are tuned, which has a high influence on the classifier(s) performance. In this paper, these gaps have been now taken into consideration in order to select the optimal ML models for the proposed FDD algorithm framework.

The main contributions of this paper can be summarized in the following bullets:

- A detailed review on various faults, namely, permanent and temporary faults in the PV array is presented.
- Introduce an effective framework for the PV array faults identification based on supervised ML models.
- The proposed framework takes into consideration the minimum number of input variables.
- Advanced setups for DT, KNN, SVM classifiers are investigated.
- The key importance for DT, KNN, SVM classifiers hyperparameters tuning is elucidated.
- The study has been validated by experimental results.

This paper is organized as follows. Section II presents the employed PV system. Section III discusses the behavior of the PV array during permanent and temporary faults, as well as explains the shortcomings of the common protective devices. Section IV presents the methodology employed for the proposed FDD algorithm. Section V presents a comprehensive quantitative evaluation of the candidate classifiers to locate PV array faults, as, the simulation results are

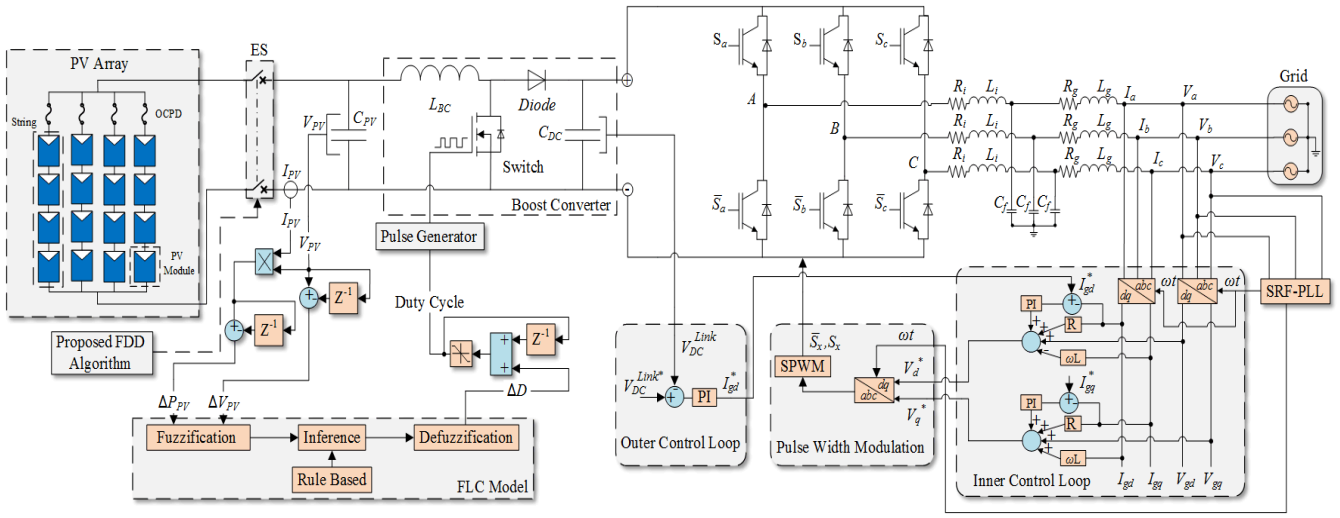

FIGURE 1. The power and control circuits employed for the grid-connected PV system.

experimentally supported using a small-scale PV system. The conclusions are present in Section VI.

II. EMPLOYED GRID-CONNECTED PV SYSTEM

The grid-connected PV system employed in this study is illustrated in Fig. 1. It was designed using the steps presented in [2]. It generally comprises the power and control circuits. A brief description of each circuit is given in the following subsections.

A. POWER CIRCUIT

A 4 kW PV array is employed in this comparative study, which comprises four parallel PV strings ($n_p = 4$) having four series PV modules ($m_s = 4$) each. An OCPD is installed with each string to isolate the faulty one.

The maximum power generated from the utilized PV array is 4 kW under healthy and Standard Test Conditions (STC—irradiance of 1000 W/m^2 and temperature of 25°C). The main electrical specifications of the polycrystalline PV module used in the PV array at STC are given in Table 1.

TABLE 1. The employed PV module electrical specifications at STC.

Parameters	Symbol	Value	Unit
Maximum Power	P_{mp}^{STC}	250	W
Open-Circuit Voltage	V_{ocm}^{STC}	37.6	V
Short-Circuit Current	I_{scm}^{STC}	8.81	A
Maximum Power Voltage	V_{mp}^{STC}	30.4	V
Maximum Power Current	I_{mp}^{STC}	8.23	A

B. CONTROL CIRCUIT

The former control circuit (FLC-based MPPT) is used to regulate the boost converter to extract the maximum power available from the PV array and increase its terminal voltage. Whereas, 49 of the fuzzy rules were formulated to cover all possible scenarios for increasing or decreasing the PV array voltage and/or power. The second one (controller of the Three-Phase, Two-Level Voltage Source Inverter), which has two control loops: 1) the voltage for the DC-Link is regulated by a Proportional-Integral (PI) controller and 2) the grid current is controlled by the current-controlled sinusoidal pulse width modulation in a direct quadrature (dq) synchronous frame.

III. TYPES OF PV ARRAY FAULTS

This section gives a brief summary of the common PV array faults and the main challenges to properly detect these faults using traditional protection systems. Some illustrative examples are also given to highlight these challenges.

The PV array faults could broadly be categorized into permanent and temporary faults, as detailed in Fig. 2 and explained in the following subsections.

A. PERMANENT FAULTS

The permanent faults include the following fault types [2], [5], [12], [30]:

1) ARC FAULT

Under normal conditions, the value of impedance is very small between PV modules interconnections. The discontinuity of any Current-Carrying Conductors (CCCs) may create a current path through the air (F1), which may initiate an electrical fire, as illustrated in Fig. 2. AFs are classified into series or parallel arc faults [9], [27], [30]–[32]. The loss of interconnection between PV modules or at the junction box may establish a series AF. On the other hand, when two adjacent conductors with different potential are placed close to each other, a parallel AF may occur [9], [27], [30].

The NEC®-2014 standards recommend employing an AFCI in PV systems with system voltage equals to or higher

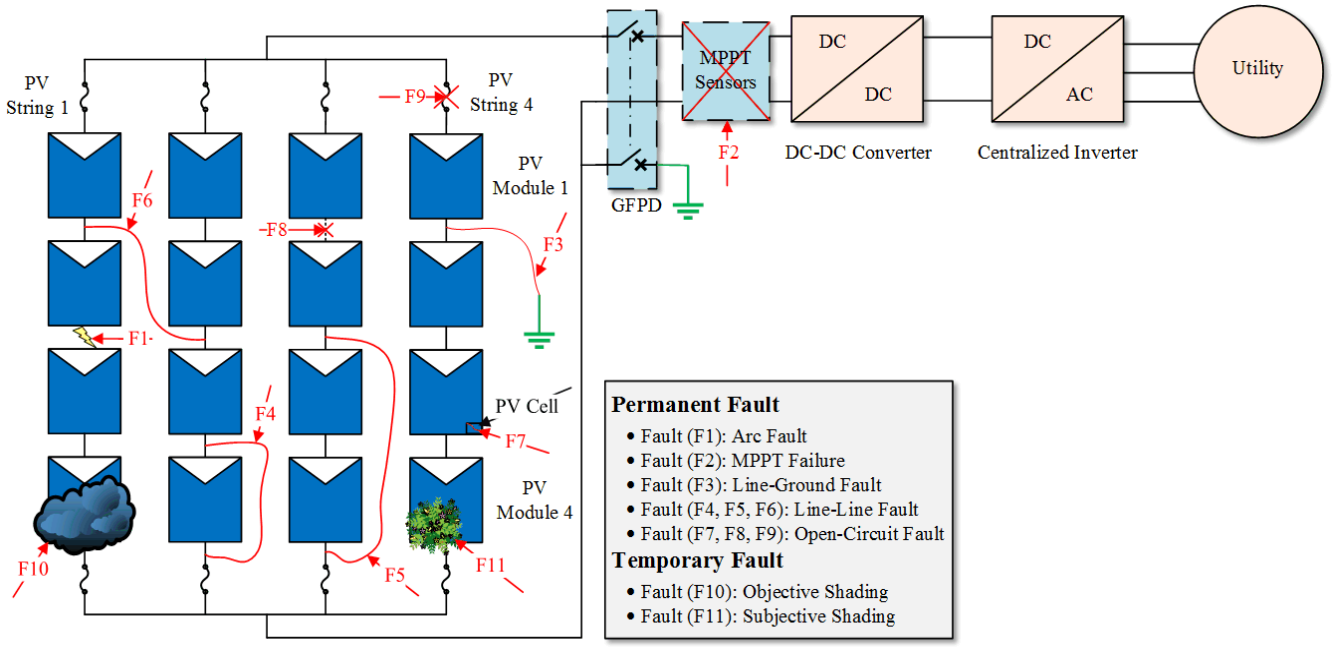

FIGURE 2. A typical grid-connected PV system consists of a PV array (4 x 4) with various types of PV array faults (permanent and temporary) followed by a two-stage converter to the utility grid.

than 80 V [8]. Nevertheless, the AFCI may fail to detect this fault type due to the following reasons [32]: 1) PV array components/connections may attenuate the propagated arc signal from the arc location to the detector, 2) false tripping of the AFCI due to capture signals from other sources, and 3) its installation. Literature has demonstrated different techniques to detect different types of AFs [5], [9], [31], [32].

In this study, a series AF is studied under different climate datasets and shading scenarios using the procedures given in [27], [31] by emulating the AF with a high impedance (R_{AF}) in the simulation study.

2) MPPT UNIT FAILURE

This fault type happens due to failure in the MPPT unit, which leads to random operating points [2], [24].

This case can be simulated by multiplying the measured signals applied to the MPPT controller by random gains.

3) LINE-TO-GROUND FAULT

This fault occurs when one or more CCCs directly, or through a fault impedance, establish an unintentional path to the ground, which is one of the most common faults in grounded PV systems. The ground faults are out of scope in this study, since they can be deemed as a special case of a LL fault involving a grounded point. This type of fault can easily be detected by the GFPD [12]. The sources of ground faults are explained in [30] and [32].

4) LINE-TO-LINE FAULT

A LL fault occurs due to a short-circuit between two different potential levels at any location in the PV array [3], [30], [32]. This type of fault may occur within the same string (F4 or F5) or across different strings (F6), as shown in Fig. 2.

A LL fault may cause a reverse current flow (I_{back}) to the faulty PV string, which is commonly avoided by installing an OCPD (i.e., fuse) in series with each PV string [8].

The amplitude of the I_{back} depends on climate datasets, potential difference between the fault points (i.e., the number of PV modules between the fault points), and fault impedance (R_{fault}) [3], [32]. The maximum expected value of the I_{back} through the faulty string can be obtained from (1) [5]:

$$I_{back} = (n_p - 1)I_{scs}^{STC} \tag{1}$$

The current rating of OCPD (I_N) in (2) shall be at least 156% of the string short-circuit current (I_{scs}^{STC}) at STC [8].

$$I_N \geq 1.56I_{scs}^{STC} \tag{2}$$

In grounded PV systems, the negative point is grounded, as shown in Fig. 3. Thus, a single OCPD in every parallel PV string is enough to protect the array against overcurrent conditions, since an OCPD will always be in the fault path. On the other hand, in ungrounded PV systems, the positive and negative conductors are not grounded. Therefore, two OCPDs should be installed in the upper and bottom conductors [8], as illustrated in Fig. 2. Hence, in case of fault, at least one OCPD will be in the fault path.

The OCPD can easily clear a LL fault when the magnitude of I_{back} is higher than I_N. Small R_{fault} values and high levels of solar irradiance and potential difference lead to higher reverse current magnitude. Nevertheless, several cases challenge the detection of this fault using OCPD, which are illustrated below:

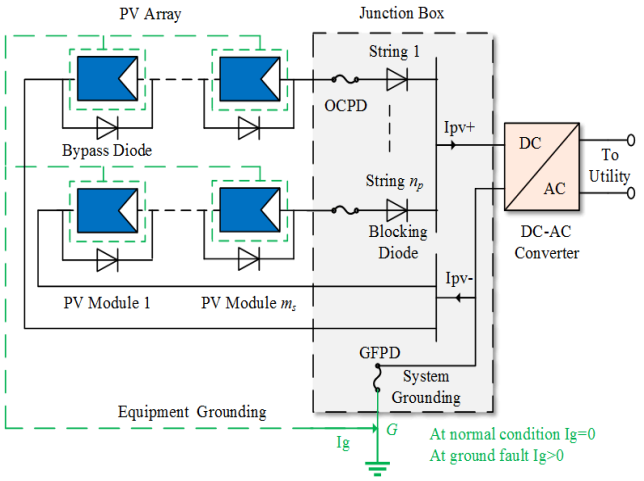

FIGURE 3. The conventional protection systems in the PV array scheme.

Case 1: Low solar irradiance levels, a small number of PV modules between fault points, and/or high R_{fault} values yield a small I_{back}, which is insufficient to melt the fuse. Fig. 4 shows the relation between two LL fault examples with different R_{fault} values: F4 (short-circuit across one module, 25% location mismatch) and F5 (short-circuit across two series modules, 50% location mismatch). It is clear from Fig. 4 that the PV array current for the F4 fault (i.e., low location mismatch) remains unaffected under high values of fault impedance, which is opposite of the F5 fault case.

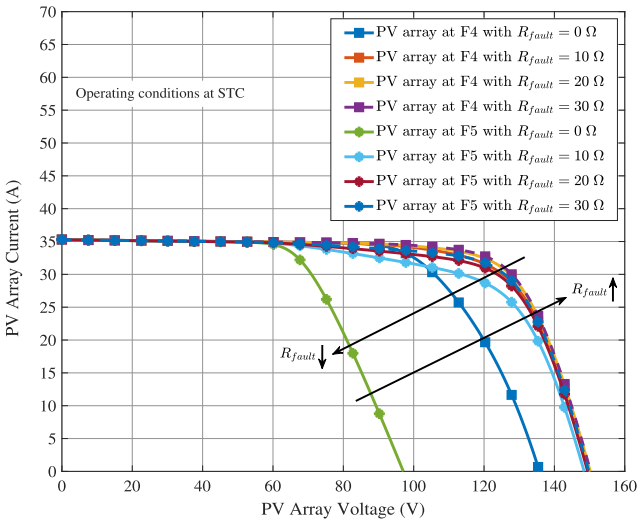

FIGURE 4. I-V characteristics of PV array under LL fault with different fault impedance values at STC.

Case 2: Blocking diode may optionally be installed in series with each OCPD, as shown in Fig. 3. Although, the blocking diode is able to block small and large reverse currents properly, it also raises some other challenges, such as [3], [12]: 1) OCPD will be unable to detect the reverse current, I_{back}, since this current is blocked, 2) they add extra power losses, and 3) since I_{back} is blocked, the PV array power may increase to the same power level corresponding to an OC fault, as shown in Fig. 5, which, in turn, challenges the discrimination between LL and OC fault types.

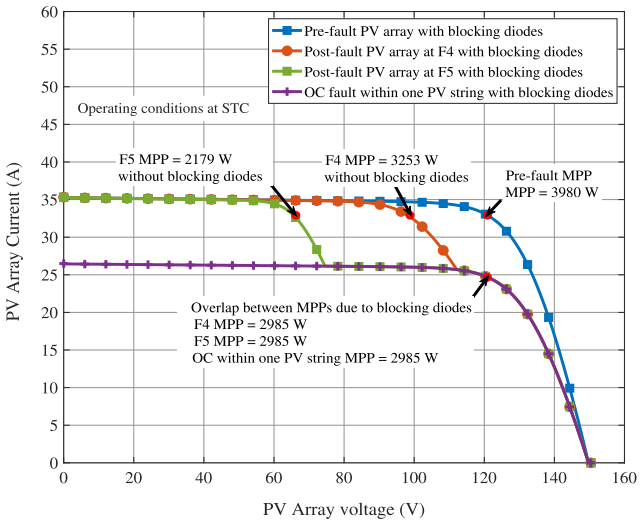

FIGURE 5. I-V characteristics of PV array under LL and OC faults using blocking diodes at STC.

Case 3: Generally, every MPPT method has its own dynamic response [33]. Moreover, the MPPT controller may quickly converge to a new Maximum Power Point (MPP), which, in turn, reduces the current I_{back} before the OCPD is able to clear it, since the OCPD clears fault currents according to its (current-versus-time) characteristics [3]. If the MPPT controller converges faster to the new MPP, the magnitude of $|I_{back}|$ will be below the tripping threshold given by (2), hence, it becomes undetectable. This gap is known as the "blind spot" [3], [26], [32]. If the I_{back} lasts longer than the fuse melting time, it can be cleared by the OCPD. To depict this case, two LL fault cases (F4 and F5) are investigated under STC and assuming $R_{fault} = 0\ \Omega$.

For the F4 or F5 cases shown in Figs. 6 and 7, respectively, the MPPT controller will detect the sudden drop (point **B**) of output power (current) and start to maximize the array power by decreasing the voltage. As a result, the operating point, for both F4 and F5, cases is moving from point **B** to **D** gradually. As shown in Fig. 6, for the F4 case, the peaks of $|I_{back}|$ under both operating points are insufficient to melt the fuse. To sum up, it will be challenging to properly identify a LL fault with a small voltage difference. On the other hand, as illustrated in Fig. 7, although the MPPT controller will reduce the I_{back}, the OCPD can properly detect and clear the F5 case depending on the response speed for the employed MPPT method, since the peak of $|I_{back}|$ corresponding to point **B** is larger than I_N $(= 2.9I_{scs}^{STC})$ of the PV string fuse. Based on the previous discussion, the blocking diodes will clearly affect the proper operation of the monitoring system. Hence, they are removed from the considered system. Furthermore, the most challenging location for the OCPDs for this case (F4) will be studied under different scenarios.

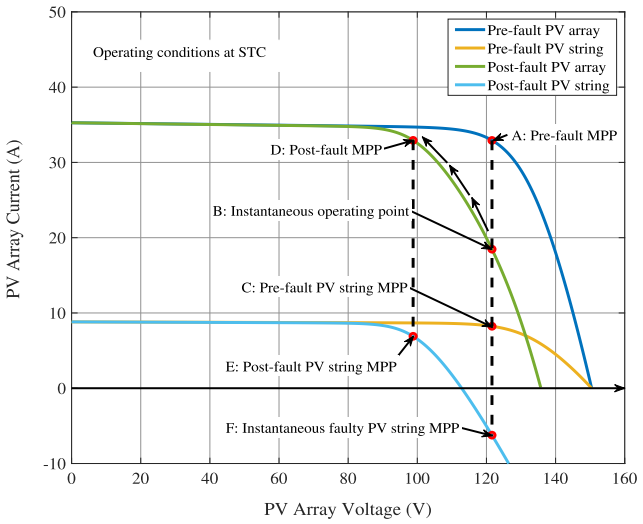

FIGURE 6. I-V characteristics of PV array under LL fault across one PV module (F4) at STC.

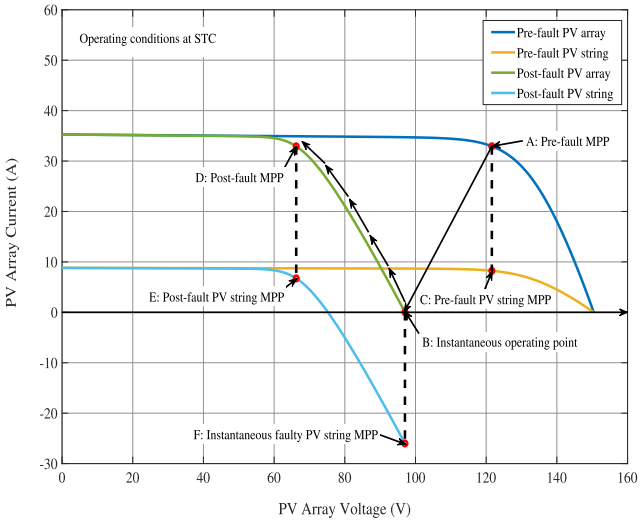

FIGURE 7. I-V characteristics of PV array under LL fault across two PV modules (F5) at STC.

5) OPEN-CIRCUIT FAULT

There are multiple reasons that lead to this type of fault [9], [27], such as cracks in cells/module (F7), disconnection in string (F8), or blowing a string fuse (F9), as shown in Fig. 2.

B. TEMPORARY FAULTS

This category of faults happens mainly due to objective or subjective shading on the PV generator, which reduces the system energy yield [2], [12], [34]. The latter is classified into dynamic and static shading [34]. Some malfunctioning cases due to shading and the associated protection device are depicted as follows.

1) OBJECTIVE SHADING

This type of shading is temporary by nature (unavoidable) and depends on the weather (e.g. heavy clouds) or environmental pollution (e.g. haze or smog) [2], which consequently reduces the effective solar irradiance, F10 as shown in Fig. 2.

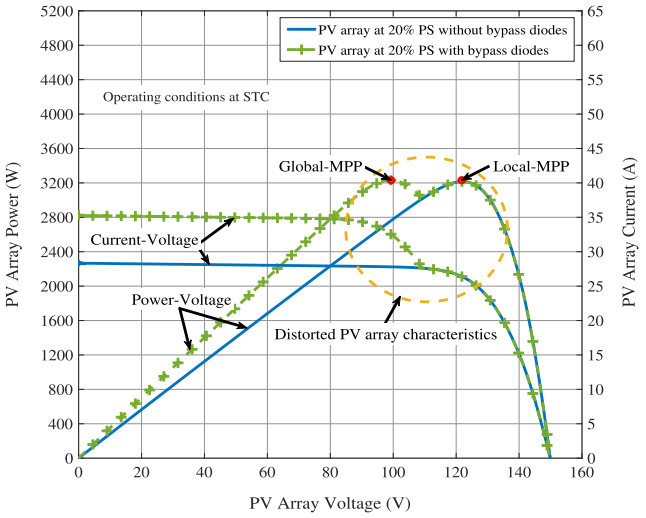

FIGURE 8. PV array characteristics under 20% PS with/without bypass diodes at STC.

2) SUBJECTIVE SHADING

Subjective shading could be categorized into static and dynamic shading [34]. The obstructions that cling to the PV modules could be defined as ''static shading''. While the change of the shaded area of the PV generator with respect to the daily sun trajectory could be defined as ''dynamic shading''. The methods utilized to avoid subjective shading are discussed in [34] and [35]. In addition to power losses, the shading may also, cause a hot-spot(s) with temperatures higher than 150°C, which damage the PV generator [36]. These hot-spots can be avoided by employing a parallel bypass diode, as shown in Fig. 3, or by sub-grouping of PV cells. Also, the PV array power can be enhanced in the case of F7 fault using bypass diodes. However, under shading or F7 fault case, bypass diodes distort the PV array characteristics, as shown in Fig. 8, which yields a local MPP (or, multiple local MPPs) for significant periods, hence, the PV array voltage collapses below the minimum allowable inverter voltage, which affects the inverter lifetime.

IV. PROPOSED METHODOLOGY FOR FDD ALGORITHM

As clarified in the introduction section, the proposed FDD algorithm is based on ML models.

The ML is a subset of artificial intelligence and is used to build models based on sample data in order to make predictions without human intervention or relying on a predetermined equation [37], [38]. The most commonly used ML algorithms are categorized into supervised and unsupervised ML [37]. Since the inputs and outputs of the ML algorithm are known in this study, supervised ML algorithms will be employed. The goal of the proposed FDD algorithm is to classify the faults into categories. Therefore, supervised ML algorithms based on this classification task are utilized. The

TABLE 2. The considered faults scenarios and their parameter values adopted for datasets collection.

Status of the PV array	Solar Irradiance	Temperature	Shading	Impedance	Number of Samples	Case Number
Healthy Condition	50:25:1200 W/m^2	0:5:55°C	-	-	564	0
Fault Condition	50:25:1200 W/m^2	0:5:55°C	0%:20%:80%	R_{AF} & R_{fault}	13536	1
Arc Fault (F1)	50:25:1200 W/m^2	0:5:55°C	0%:20%:80%	20:20:100 Ω & 200 Ω	2820	2
MPPT Unit Failure (F2)	50:25:1200 W/m^2	0:5:55°C	0%:20%:80%	-	2820	3
LL Fault (F4)	50:25:1200 W/m^2	0:5:55°C	0%:20%:80%	0:10:30 Ω	2820	4
OC Fault (F7, F8, and F9)	50:25:1200 W/m^2	0:5:55°C	0%:20%:80%	-	2820	5
Shading (F10 and F11)	50:25:1200 W/m^2	0:5:55°C	20%:20%:80%	-	2256	6

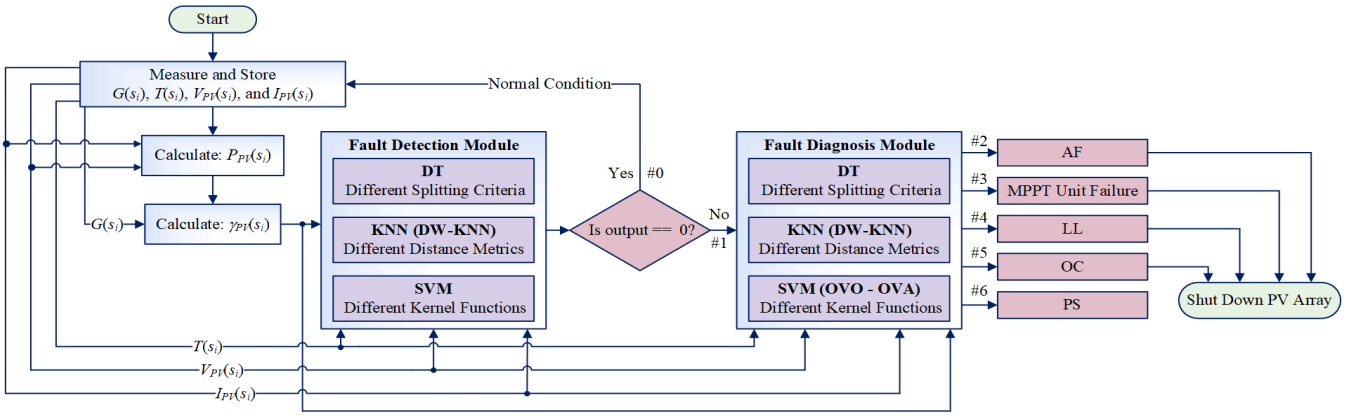

FIGURE 9. The flow diagram of the proposed FDD algorithm framework for the PV array faults.

proposed FDD algorithm framework has two main modules, as illustrated in Fig. 9. The former, **fault detection**, is used to discriminate between healthy and fault conditions. Whereas, the role of the latter, **fault diagnosis**, is used to distinguish between the permanent faults, namely, series AF (F1), MPPT unit failure (F2), LL fault across one PV module (F4), and OC fault (F7, F8, and F9), and temporary faults, namely, objective (F10) and subjective (F11) shading. The latter module is triggered by the detection module output. Moreover, the diagnosis module is triggering the Emergency Switch (ES) to disconnect the PV array under the permanent faults scenarios.

In each module, three ML classifiers with different setups will be recruited, as illustrated in Fig. 9, namely, DT based on three splitting criteria (Gini Index, Towing Rule, and Deviance), KNN based on four distance metrics (Euclidean, City Block, Mahalanobis, and Cosine), and SVM based on four Kernel functions (Linear, Quadratic, Cubic, and GRB).

The hyperparameters tuning method highly affects the behaviors of ML models. So, Bayesian Optimization is adopted to assign the optimal hyperparameters for each setup of the employed classifiers. Also, this tuning method will assign optimal distance weighting functions (uniform, inverse, or squared inverse) in KNN and optimal multi-classification approach (OVO or OVA) in SVM at diagnosis module only. The three ML classifiers and their setups lead to different ML models for each module (11 models for detection and also for diagnosis), as shown in Fig. 10, which offers the possibility to choose the optimal model for each module based on the comparative case study introduced in subsequent sections.

A. STAGES OF FDD FRAMEWORK

The FDD algorithm illustrated in Fig. 9 is applied to the PV system given in section II, while applying the possible PV array faults shown in Fig. 2, and summarized in Table 2.

The process of adapting the fault classifiers with the proposed framework is shown in the flowchart given in Fig. 10.

1) STAGE 1 - DATA ACQUISITION

The first block is collecting the relevant datasets. The datasets are extracted by using the employed PV system model represented in MATLAB®/Simulink, section II.

The simulations are carried out using Matlab 2020a and running on Lenovo$_{TM}$, Core i5-5200U CPU @ 2.20 GHz processor with 8 GB RAM.

A total of 14100 samples and their corresponding labels are collected under different scenarios, as given in Table 2. The proposed FDD algorithm has four input quantities at every instance s_i, namely, solar irradiance $G(s_i)$, temperature $T(s_i)$, PV array voltage $V_{PV}(s_i)$, and PV array current $I_{PV}(s_i)$. The $G(s_i)$ and $T(s_i)$ are obtained from the reference PV module, which is located at the unshaded portion of the PV array site. While the $V_{PV}(s_i)$ and $I_{PV}(s_i)$ are already measured by the FLC-based MPPT, as illustrated in Fig. 1.

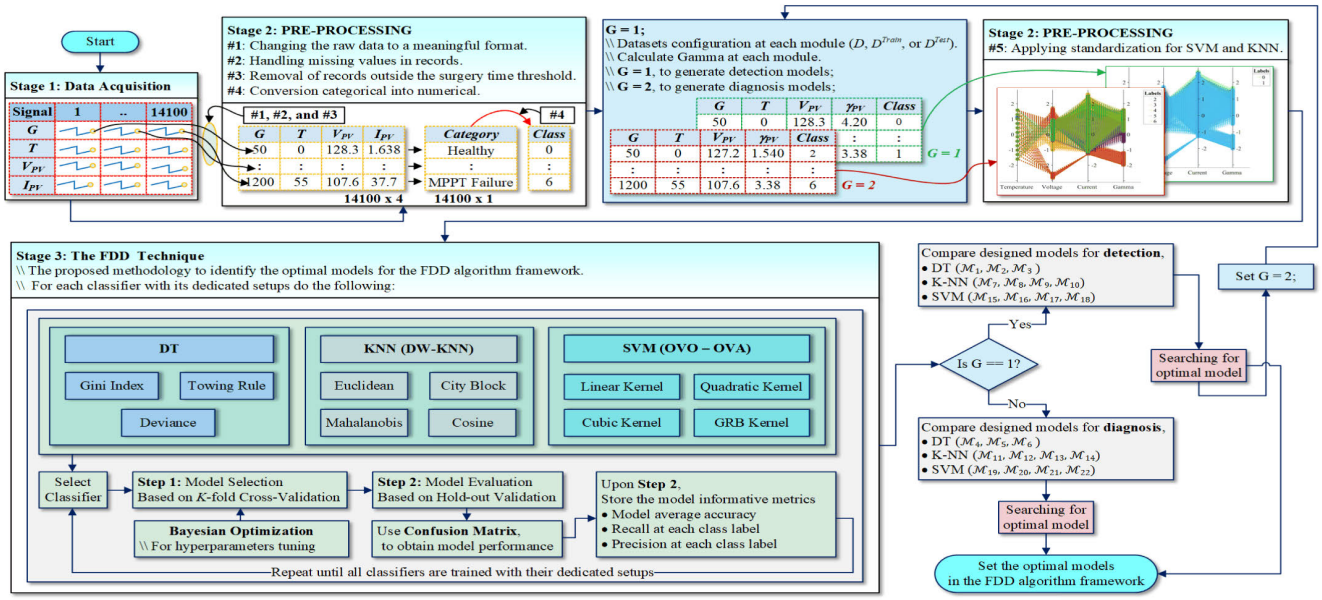

FIGURE 10. The workflow to obtain optimal supervised ML models for detection and diagnosis modules.

The PV array voltage and current at s_i are used to calculate the output power $P_{PV}(s_i)$ and gamma $\gamma_{PV}(s_i)$. The latter is defined as the ratio between the PV array output power and the solar irradiance, as given by (3) [12], [39].

$$\gamma_{PV}(s_i) = \frac{P_{PV}(s_i)}{G(s_i)} = \frac{V_{PV}(s_i) * I_{PV}(s_i)}{G(s_i)} \tag{3}$$

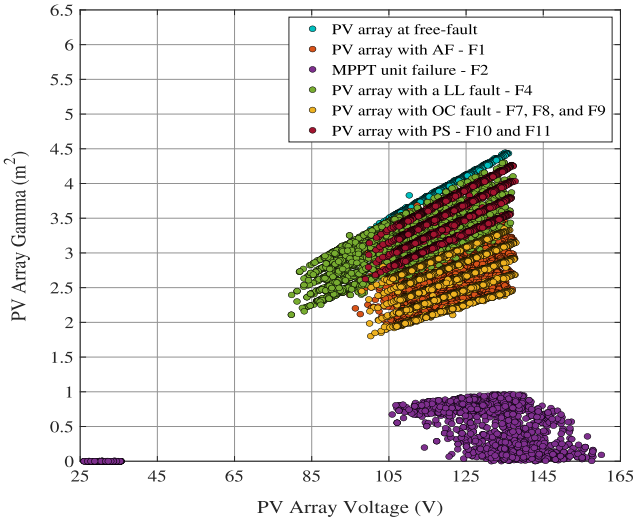

FIGURE 11. Overlapping the MPPs of a PV array at different faults cases.

2) STAGE 2 - PRE-PROCESSING

The next stage is to pre-process the collected datasets, as illustrated in Fig. 10. This is carried out by: 1) changing the raw data to a meaningful format, 2) handling missing values in records, 3) removal of records outside the surgery time threshold, 4) conversion categorical into numerical (i.e., case number), as illustrated in Table 2, and 5) then either normalization or standardization approaches are applied in order to change the input quantities "attributes" with dynamic range (i.e., inter-attribute differences in scale) into specific range (i.e., attribute scaling, attributes are on a similar scale) in order to achieve better results. As an example to date pre-processing, the possible MPPs for the employed PV array under both healthy and fault conditions are plotted in two dimensions (V_{PV}-versus-γ_{PV}), as shown in Fig. 11. Although the PV array under healthy/fault case(s) has different output behavior, there is a notable overlap between the operating points (MPPs) for these cases. Hence, the output voltage and current under a fault case may be similar/nearest to another healthy/fault case, which challenges the successful discrimination between these cases. This problem can be mitigated by either normalization or standardization approaches [40].

The normalization is carried out by rescaling each attribute in the datasets into a range from zero to one [23], [40].

In [23], normalization of V_{PV} and I_{PV} have been performed according to a reference PV module. However, this approach has limitations include: any shading or mismatch on the reference PV modules or over the PV array may cause incorrect normalization of V_{PV} and I_{PV} data leading to inexact discrimination between cases.

Normalization can be also done by applying (4), which is denoted as $Min - Max$ normalization [40].

$$Normalization_{Min}^{Max} = \frac{(x_{i|\varphi}) - Min(\varphi)}{Max(\varphi) - Min(\varphi)} \tag{4}$$

where $x_{i|\varphi}$ is the i^{th} data point at attribute φ. While, Min and Max represent the minimum and maximum values of attribute φ, respectively.

As indicated in (4), a single outlier or even a very small value in the attribute φ may force the remaining values of the attribute φ toward zero.

In standardization approach, each attribute is rescaled such that the mean value is zero while the standard deviation equals one [40]. The standardized values of an attribute φ is called a $Z-score$ and is obtained from (5).

$$Z-score|_{\varphi}=\frac{\varphi-\mu_{\varphi}}{\sigma_{\varphi}} \tag{5}$$

where $Z-score|_{\varphi}$ is standardized values of attribute φ, φ is an attribute that is being standardized, μ_{φ} and σ_{φ} are the mean and standard deviation of an attribute φ, respectively. From (5), the standardization approach is robust compared to the $Min-Max$ normalization since no limits are imposed on the range of standardized data. This enables the standardization approach to deal with outliers in the datasets. In this research, the standardization approach will be applied to KNN and SVM classifiers. Whilst, the DT classifier does not require rescaling the attributes because it just compares each attribute with a certain threshold value. Hence, it does not matter whether these attributes are in one range or not.

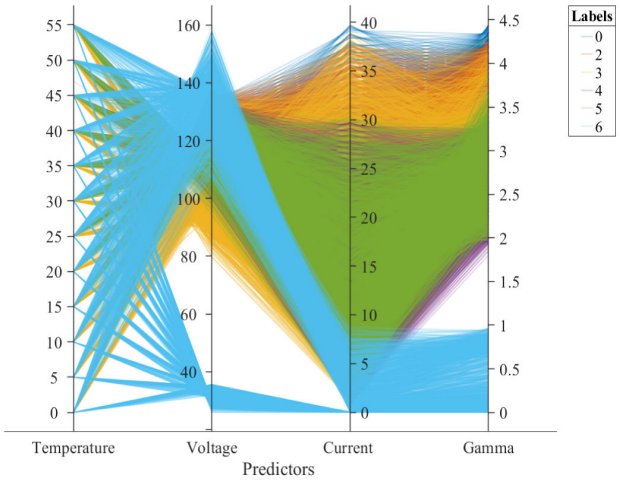

FIGURE 12. The utilized input quantities before applying standardization.

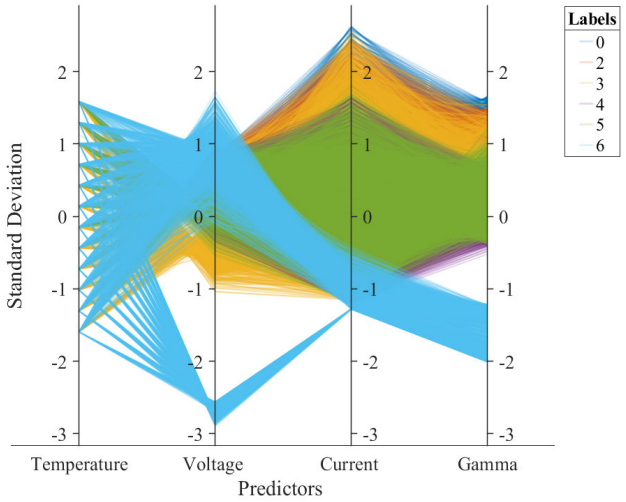

FIGURE 13. The utilized input quantities after applying standardization.

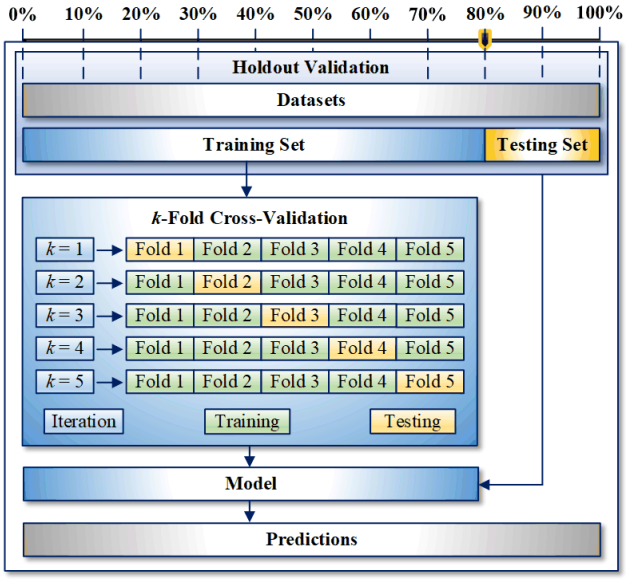

FIGURE 14. Workflow for hold-out and k-fold cross-validation method.

The parallel coordinates plot for the utilized four input quantities to the ML classifier before applying standardization is shown in Fig. 12, which depicts a clear overlapping between the healthy case (0) and/or different fault scenarios (1), namely, AF (2), MPPT unit failure (3), LL (4), OC (5), and PS (6). By applying standardization, as shown in Fig. 13, this overlapped is highly relieved. Besides, a specific range for each attribute is achieved to decrease the non-linearity intensity due to the dynamic range, as shown in Fig. 13.

3) STAGE 3 - FDD TECHNIQUE

As discussed previously, the FDD algorithm is based on ML models. Selecting a suitable ML algorithm is vital to ensure proper classification. This is usually done by comparing the performance of different ML models, as shown by the flowchart given in Fig. 10. In order to evaluate any model, there are two common methods, namely, the hold-out validation and k-fold cross-validation, as shown in Fig. 14.

The hold-out validation is carried out by dividing the collected datasets (D), given by (6), into two sets, as shown in Fig. 14, namely, training datasets (typically, $\sim$ 80%) and the testing datasets (typically, $\sim$ 20%). The training datasets (D^{Train}) are used to train and build the model. Then, the model is evaluated using testing datasets (D^{Test}) [41], [42].

On the other hand, in k-fold cross-validation, the training datasets (also, can be the original datasets (i.e., D)), are randomly partitioned into k equally sized datasets (i.e., subsets or folds) as shown in Fig. 14. The algorithm is trained according to (7) for k iterations and tested according to (8) in each iteration. This process is repeated until all folds are used as

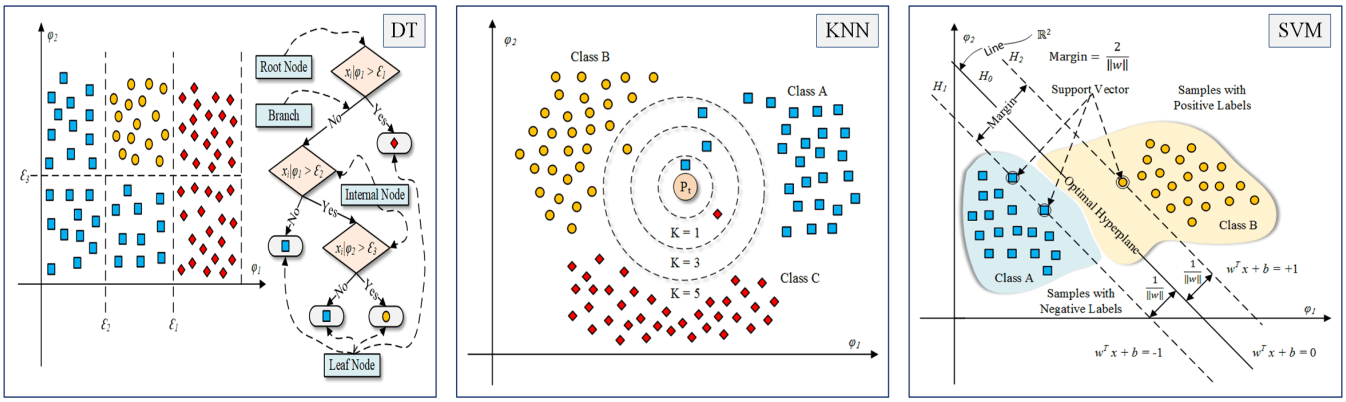

FIGURE 15. The ML classifiers recruited for the FDD algorithm framework, namely, DT, KNN, and SVM.

test data for once. The error for each test fold is calculated to obtain the average classifier error, as indicated in (9).

$$D = \{(x_i, y_i), 1 \leqslant i \leqslant n, x_i \in \mathbb{R}^d, y_i \in \{\theta_1, .., \theta_c\}\} \quad (6)$$

$$Training\ Samples = \frac{(k-1)D^{Train}}{k} \quad (7)$$

$$Testing\ Samples = \frac{D^{Train}}{k} \quad (8)$$

$$A_{CL}^{E} = \frac{1}{k}\sum_{1}^{k} E_k \quad (9)$$

where $D = (D^{Train} + D^{Test})$ is the dataset of n samples in a d-dimensional space belonging to one of y_i labels, θ is an integer number, and k is a positive integer number. E_K is the error of each k iteration and A_{CL}^{E} is the average CL error. In this study, the following steps are followed to generate the final ML model for either detection or diagnosis modules at different setups associated with each classifier.

Step 1: The k-fold cross-validation is used for model selection ($\mathcal{M}$) from multiple arising models based on the optimal performance during the hyperparameters tuning phase.

Step 2: After the tuning (training, **Step 1**) phase, the generalization performance for the final model ($\mathcal{M}$) is evaluated again using the hold-out validation method. A percentage of 65% of the datasets (D^{Train}, for detection = 9165 and diagnosis = 8799 samples) is used for the tuning phase based on Bayesian Optimization. While, the remaining 35% of the datasets (D^{Test}, for detection = 4935 and diagnosis = 4737 samples) will be used for the testing phase, upon which the optimal model is selected, at each module.

Step 3: Optionally, all datasets (D, for detection = 14100 and diagnosis = 13536 samples) and the hyperparameters, which are used to generate the final model (**Step 1**), can be used for training again. This step has been considered herein.

Step 4: As illustrated in Fig. 10, if the model performance is unsatisfactory, there are a number of issues that must be examined such as the attributes which are not properly identified (**Stage 1**), data not being cleaned appropriately (**Stage 2**), the unsuitable ML algorithm utilized, or model parameters are not properly tuned (**Stage 3**). The k-fold cross-validation method is preferably used for model selection (**Step 1**) because it provides a better estimate of how well the model will perform with random train and test datasets. Besides, it helps avoid the overfitting problem [40], [41], [43]. The recommended values of k are given [41], which is set to 5 in this study.

B. EMPLOYED MACHINE LEARNING CLASSIFIERS

This section introduces a brief summary of the main idea behind the ML algorithms, namely, DT, KNN, and SVM employed in this comparative study, as illustrated in Fig. 10.

1) DECISION TREE (DT) CLASSIFIER

The DT classifier structure consists of nodes, branches, and leaves, as shown in Fig. 15, which can easily be visualized and interpreted. To classify an observation, the attribute test condition at the root node initially decides the appropriate branch to be followed. Based on the obtained decision, the algorithm continues to another interior node with a new test condition, or to a leaf node associated with the class label to be assigned to the observation [38], [42].

Several induction algorithms can be recruited for DT classifier such as ID3 (Iterative Dichotomiser 3), C4.5 (Successor of ID3), and CART (Classification And Regression Tree) [44]. Each induction algorithm has its own splitting criterion such as Shannon entropy (used with ID3) and normalized Shannon entropy (employed with C4.5) [45]. While, Gini Index, Towing Rule, or Deviance are commonly used with the CART algorithm [46]. Based on the employed induction algorithm and the splitting criterion, the data structure and its behavior will be different. The CART induction algorithm will be employed in this study since it can easily handle outliers, missing values, and noisy data. The workflow for the CART algorithm has been illustrated in [44] and [46]. The mentioned splitting criteria employed with the CART are investigated in each fault module.

The CART is growing by binary splits, such that the root or any interior node has exactly two outgoing branches. Hence,

a deep or shallow tree could be produced. The DT depth can be controlled by imposing a certain stopping criterion, such as the maximum number of splits or minimum size of observations associated with each leaf or parent node. The maximum number of splits has been chosen to be the stopping criterion and it is optimized using the tuning method.

2) K-NEAREST NEIGHBOR (KNN) CLASSIFIER

The basic idea behind the KNN is to find a group of K points in the training datasets that are nearest to an unknown test point (P_t), based on a particular distance function [47], [48]. The test point is assigned to a class label according to the majority of the K neighbors points nearest to the given test point. As shown in Fig. 15, the number of neighbors nearest to the test point (i.e., K) is a key tuning parameter that affects the performance of the KNN model.

The generalization performance can be sensitive to noisy data if K is a too small integer number, especially one. On the other hand, if K is a large integer number, the points in the K nearest neighbors can involve instances from various classes, which enhances the performance of generalization at the cost of prediction speed.

The K can be assigned according to the tuning methods or by taking the square root of the total number of observations in the training datasets ($\mathrm{K} = \sqrt{n}, n \in D^{Train}$) [47]. In this study, this issue is left to the adopted tuning method. The distance metric can also affect the KNN model performance. There are diverse families of distance measures such as Minkowski, Inner Product, Squared Chord, and Vicissitude [47], [49]. A comparative study between distance metrics enlisted in Table 3 has been performed either in the detection or diagnosis module.

TABLE 3. The considered distance metrics with the KNN classifier.

Distance Metric	Equation
Euclidean	$\sqrt{\sum_{\varphi=1}^{d}\left\lvert x_{i\lvert\varphi} - P_{t\lvert\varphi}\right\rvert^2}$
City Block	$\sum_{\varphi=1}^{d}\left\lvert x_{i\lvert\varphi} - P_{t\lvert\varphi}\right\rvert$
Mahalanobis	$\sqrt{(x_{i\lvert d} - P_{t\lvert d})(\frac{Covariance}{Matrix})^{-1}(x_{i\lvert d} - P_{t\lvert d})^T}$
Cosine	$1 - \frac{\sum_{\varphi=1}^{d}(x_{i\lvert\varphi})(P_{t\lvert\varphi})}{\sqrt{\sum_{\varphi=1}^{d}(x_{i\lvert\varphi})^2}\sqrt{\sum_{\varphi=1}^{d}(P_{t\lvert\varphi})^2}}$

Assigning the class label θ_{P_t} for the test point P_t can be deduced by (10), which is called Majority Voting approach.

This approach has difficulty dealing with imbalanced datasets and cost-sensitive learning [48].

$$\theta_{P_t} = \underset{v}{\operatorname{argmax}} \sum_{(x_i, y_i)\in D_z} \delta(v, y_i) \tag{10}$$

The Distance-Weighted Voting approach represents another approach that adopts weighting the neighbors' votes according to their distances [50]. Hence, (10) can be modified to (11) by adding the weighting factor (w_i), where w_i indicates the weight of the i^{th} nearest neighbor, and δ is an indicator function that returns one in case its argument is true and zero otherwise. Other approaches to compute the weighting factors are given in [49] and [50].

$$\theta_{P_t} = \underset{v}{\operatorname{argmax}} \sum_{(x_i, y_i)\in D_z} w_i \delta(v, y_i) \tag{11}$$

This research adopts two methods to compute the weight: 1) the weight is computed as the reciprocal of the distance (12) and 2) the weight is represented by the reciprocal of the squared distance (13) [50], [51]. The predicted class θ_{P_t} can also be obtained by minimizing the expected classification cost (14) [52], this method applied herein.

$$w(x_{i|\varphi}, P_{t|\varphi}) = \left(Distance(x_{i|\varphi}, P_{t|\varphi})\right)^{-1} \tag{12}$$

$$w(x_{i|\varphi}, P_{t|\varphi}) = \left(Distance(x_{i|\varphi}, P_{t|\varphi})\right)^{-2} \tag{13}$$

$$\theta_{P_t} = \underset{\theta_{P_t}=\theta_1,\ldots,\theta_c}{\operatorname{argmin}} \sum_{y_i=\theta_1}^{\theta_c} \hat{p}(y_i|P_t) C(\theta_{P_t}|y_i) \tag{14}$$

where D_z contains K training samples that are nearest to the P_t, $\hat{p}(y_i|P_t)$ is the posterior probability of class y_i for observation P_t, and $C(y_i|\theta_{P_t})$ indicates the true misclassification cost to classify an observation as θ_{P_t} when its true is y_i.

The influence of Distance Weighted KNN (DW-KNN) has been studied by adopting (12) and (13) for the distance metrics which are enlisted in Table 3, besides non-weighted KNN (uniform, all $y_i \in D_z$ are weighted equally). The tuning method's role herein is to assign the optimal weighting function (uniform, inverse, or squared inverse) for the reported distance metrics and the optimal number of neighbors.

3) SUPPORT VECTOR MACHINE (SVM) CLASSIFIER

The SVM is essentially a binary classifier, thus, the number of classes (N_c), is exactly two [2]. However, the SVM can be adapted to handle the multi-classification problems using one of the two most common approaches, namely, One-Versus-One (OVO) or One-Versus-All (OVA) approach [53], [54].

The principle of operation of SVM depends on the type of sample data. In the case of samples that are linearly separable, as illustrated in Fig. 15, there are many possible separating hyperplanes (or, separators), which can separate two classes. However, concerning the optimal choice, the most interesting choice corresponds to one with the largest possible margin [2]. The margin is completely defined by finding the Support Vectors where the data points are located on the boundary of the slab (or, line) [26], [53].

In order to maximize the margin, it needs to minimize the $\|w\| = \sqrt{w^T w}$ with the condition that no samples are existing between the two boundaries (**Case 1**, hard margin condition).

The SVM is also able to handle the samples that are not fully linearly separable (**Case 2**, soft margin condition). This is carried out by introducing a positive slack variable $\xi_i \geq 0$ for $\forall_i$. The constrained optimization problem for the separable case (**Case 1**) will be converted into a new form by introducing the slack variable using (15) [37], [54].

The penalty parameter (C), in (15), is a tradeoff between the margin and the training errors. This, in turn, controls the under/over-fitting problem.

$$\begin{cases} Minimize: \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i \\ Subject\ to: y_i(w^T x_i + b) \geq 1 - \xi_i \quad \forall_i \end{cases} \tag{15}$$

where x_i and y_i are the data point and the class label, respectively. The w, b, and $\|w\|$ are the vector normal to the hyperplane, bias, and the Euclidean norm of w, respectively.

For the data clouds shown in Fig. 15 that are non-linearly separable input space (**Case 3**, non-linear classification problem), instead of utilizing Linear Kernel, given by (16), the other Kernel functions such as Polynomial or Gaussian Radial Basis (GRB) which given in (17) and (18), respectively, can be employed to map the original training instances (x_i) to a higher-dimensional space, where data clouds are more likely to be linearly separable [37], [40], [54].

$$Linear\ Kernel = x_i^T . x_j \tag{16}$$

$$Polynomial\ Kernel = (1 + x_i^T . x_j)^{\rho} \tag{17}$$

$$GRB\ Kernel = exp(\frac{-\|x_i - x_j\|^2}{2\sigma^2}) \tag{18}$$

where ρ denotes the degree for the Polynomial Kernel and the σ indicates the width of the GRB Kernel.

In each module, the linear and non-linear Kernel functions are investigated. The non-linear Kernel employed are Polynomial (Quadratic and Cubic) and the GRB Kernel. In the diagnosis module, N_c>2. Hence, the SVM is built based on one of the multi-classification SVM approaches.

The tuning method has been adopted to assign the optimal approach (OVO or OVA) at each recruited Kernel function. Also, the tuning method is adopted in each module to assign the optimal values, namely, penalty parameter for all Kernel and the width of the GRB Kernel.

C. HYPERPARAMETERS TUNING

The training performance of ML algorithms is directly influenced by their initial hyperparameters tuning phase, which is a mandatory step in order to achieve an optimally trained model in a sensible amount of time [55]. There are several tuning methods that could be used including Manual Search [55], Grid Search [56], Random Search [57], and Bayesian Optimization [58].

Among these methods, Bayesian Optimization (BO) has been adopted in this study to assign the optimal hyperparameters to the studied classifiers. This method is recognized to be better than other methods in finding the optimal parameters in a sensible amount of time, and is efficient in optimizing black-box functions which are difficult to evaluate [58].

The hyperparameters that will be optimized for the three studied ML classifiers are given in Table 4. In the BO method, a probabilistic model of a true objective function $f(\lambda)$ will be built and used to select the most promising hyperparameters to evaluate the function $f(\lambda)$. In this study, the optimization goal is to find the global optimum (λ^*) for a black-box function f, (19), in a minimum number of steps. The searching range for hyperparameters is specified by the Lower (LB) and Upper (UB) Bounds, as given by Table 4. The dedicated range in Table 4 is logarithmically rescaled. The number of samples $n \in D^{Train}$.

TABLE 4. Hyperparameters configurations.

Classifier Setup	Hyperparameters Settings	
	Optimize	Bounds
DT		
Gini Index		LB = 1
Towing Rule	Maximum Number of Splits	
Deviance		UB = $n-1$
KNN		
Euclidean		LB = 1
City Block	Distance Weighting Functions	
Mahalanobis	Number of Neighbors	
Cosine		UB = $\frac{n}{2}$
SVM		
Linear Kernel		LB = 10^{-3}
Quadratic Kernel	Multi-Class Approach - all Kernel	
Cubic Kernel	Penalty Parameter - all Kernel	
GRB Kernel	Kernel Width - GRB Kernel	UB = 10^{+3}

$$\lambda^* = \underset{\lambda \in \Lambda}{\text{argmax}}\, f(\lambda) \tag{19}$$

where Λ represents the hyperparameters space, which can include discrete, continuous, and/or categorical (e.g., distance weighting functions as in KNN or multi-classification approach as in SVM) variables. For any arbitrary $\lambda \in \Lambda$, the $f(\lambda)$ can, then, be obtained.

Algorithm 1 Pseudocode for the Bayesian Optimization

Input: Acquisition Function, Stopping Criterion, $t, f, \hat{f}$, and Λ
Output: Optimal Hyperparameter(s)
for $t = 1, 2, 3, 4, \ldots\ldots, 50$ *do*
Find: $\lambda_t = \underset{\lambda \in \Lambda}{\text{argmax}}\, \hat{f}(\lambda | \mathcal{H}_{1:t-1})$
Evaluate: $\mathcal{F}_t = f(\lambda_t)$
Update: $\mathcal{H}_{1:t} = \{\mathcal{H}_{1:t-1}, (\lambda_t, \mathcal{F}_t)\}$
end for

The BO is an iterative process until a predefined stopping criterion is reached [55]. The stopping criterion is chosen as the number of iterations (t) and is set to 50. The pseudocode for the Bayesian Optimization algorithm is given in Algorithm 1. The Gaussian Process is used to build the response surface function ($\hat{f}$). Once the $\hat{f}$ is estimated, the acquisition function is computed to find the point where f is maximized. Among the available acquisition functions [55], [58], the Expected Improvement function is used in this study. The $f(\lambda)$ could be the model prediction accuracy or the minimum Cross-Validation (CV) error of the ML model. The latter will be used herein to be the true objective function.

The results for the **22** optimized ML models based on the BO method (i.e., $\mathcal{M}_1$ to $\mathcal{M}_{22}$) are given in Fig. 16, where

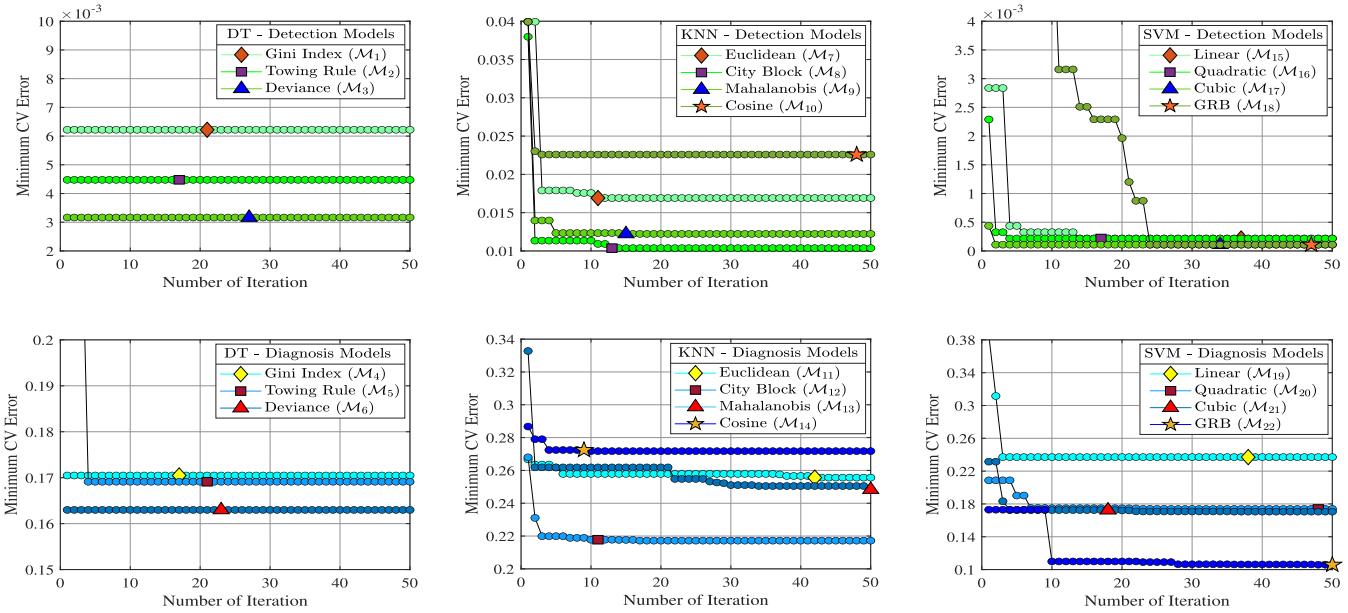

FIGURE 16. Minimum Cross-Validation (CV) Error of the DT, KNN, and SVM models based on different splitting criteria, distance metrics, and Kernel functions, respectively during the Bayesian Optimization tuning process.

the vertical axis represents the minimum *CV* error and the horizontal axis represents the number of iterations of the BO. The $\mathcal{M}_1$ to $\mathcal{M}_6$, $\mathcal{M}_7$ to $\mathcal{M}_{14}$, and $\mathcal{M}_{15}$ to $\mathcal{M}_{22}$ represents the optimal DT, KNN, and SVM models, respectively.

TABLE 5. Confusion Matrix for the multi-classification problem.

Class	Predicted			
	$\boldsymbol{\theta_1}$	$\boldsymbol{\theta_2}$	.....	$\boldsymbol{\theta_M}$
True $\boldsymbol{\theta_1}$	$\mathrm{TP}_{\theta_1}(1,1)$	$\mathrm{E}_{\theta_1\theta_2}(1,2)$	.....	$\mathrm{E}_{\theta_1\theta_M}(1,M)$
True $\boldsymbol{\theta_2}$	$\mathrm{E}_{\theta_2\theta_1}(2,1)$	$\mathrm{TP}_{\theta_2}(2,2)$	.....	$\mathrm{E}_{\theta_2\theta_M}(2,M)$
True ⋮	⋮	⋮	.....	⋮
True $\boldsymbol{\theta_N}$	$\mathrm{E}_{\theta_N\theta_1}(N,1)$	$\mathrm{E}_{\theta_N\theta_2}(N,2)$	.....	$\mathrm{TP}_{\theta_N}(N,M)$

D. CLASSIFICATION METRICS

Among the available evaluation methods [59], the Confusion Matrix (*CM*) is used. Whereas, by using *CM*, many classification metrics can be obtained [43], [59].

The *CM* is a tool for analyzing the performance of binary and multi-classification problems. It results in a single matrix (N_c x N_c) which thereby helps the designer to understand the type of the error and observe the relation between the classifier outputs (predicated) and true (actual) ones. The *CM* employed for measuring the multi-classification performance is illustrated in Table 5. The True Positive (TP) represents correct predictions at a specific class label and Errors (E) indicates the misclassified cases. After obtaining the *CM* for multi-classification problems, **Recall** and **Precision** can be deduced for a particular class label ($\boldsymbol{\theta}$) and the overall classifier **Accuracy** from (20) to (22), respectively. These equations can be applied for a binary-classification problem, for this case, where the dimension of the *CM* is (2 x 2).

1) RECALL

Indicates the effectiveness of the classifier to classify θ_N correctly to the total number of samples represents in θ_N as indicated in (20).

$$Recall_{(\theta_N)} = \frac{CM(N,N)}{\sum_{M=1}^{N_C} CM(N,M)} \tag{20}$$

2) PRECISION

Represents the proportion of samples in θ_M that are correctly classified by the classifier to the total number of predictive samples in θ_M as given in (21).

$$Precision_{(\theta_M)} = \frac{CM(M,M)}{\sum_{N=1}^{N_C} CM(N,M)} \tag{21}$$

3) CLASSIFICATION ACCURACY

Represents the overall effectiveness of the classifier. It can be deduced from (22).

$$Accuracy = \frac{\sum_{N=1}^{N_C} CM(N,N)}{\sum_{N=1}^{N_C}(\sum_{M=1}^{N_C} CM(N,M))} \tag{22}$$

where N and M refer to the index of a row and a column, respectively for *CM*. The $CM(N,M)$ stands for the number of samples at class N that are assigned to class M by the adopted classification method. The diagonal ($N = M$) of the *CM* captures the correct classification decisions.

TABLE 6. The prediction time given by the designed models.

Detection Models	Prediction Time (ms)	Diagnosis Models	Prediction Time (ms)
DT Models		**DT Models**	
Gini Index ($\mathcal{M}_1$)	3.537	Gini Index ($\mathcal{M}_4$)	4.429
Towing Rule ($\mathcal{M}_2$)	3.093	Towing Rule ($\mathcal{M}_5$)	6.794
Deviance ($\mathcal{M}_3$)	3.689	Deviance ($\mathcal{M}_6$)	7.231
KNN Models		**KNN Models**	
Euclidean ($\mathcal{M}_7$)	59.678	Euclidean ($\mathcal{M}_{11}$)	376.681
City Block ($\mathcal{M}_8$)	17.09	City Block ($\mathcal{M}_{12}$)	116.838
Mahalanobis ($\mathcal{M}_9$)	2976.898	Mahalanobis ($\mathcal{M}_{13}$)	2820.551
Cosine ($\mathcal{M}_{10}$)	3381.214	Cosine ($\mathcal{M}_{14}$)	429.739
SVM Models		**SVM Models**	
Linear ($\mathcal{M}_{15}$)	4.376	Linear ($\mathcal{M}_{19}$)	40.502
Quadratic ($\mathcal{M}_{16}$)	4.46	Quadratic ($\mathcal{M}_{20}$)	103.053
Cubic ($\mathcal{M}_{17}$)	6.401	Cubic ($\mathcal{M}_{21}$)	230.522
GRB ($\mathcal{M}_{18}$)	8.043	GRB ($\mathcal{M}_{22}$)	183.389

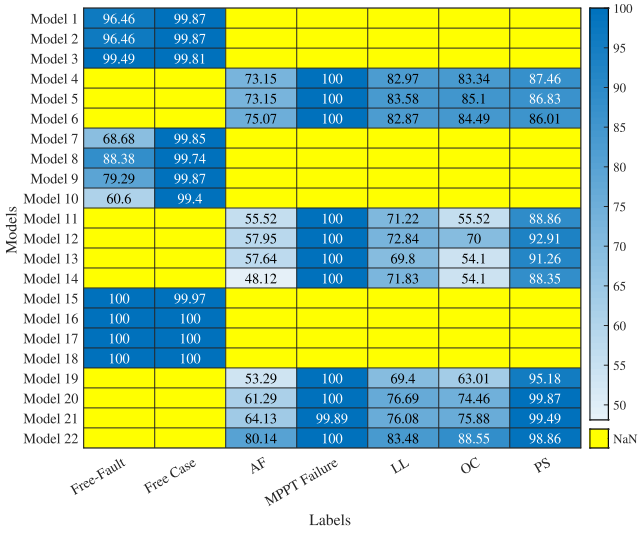

FIGURE 17. The labels recall accuracy at independent test datasets.

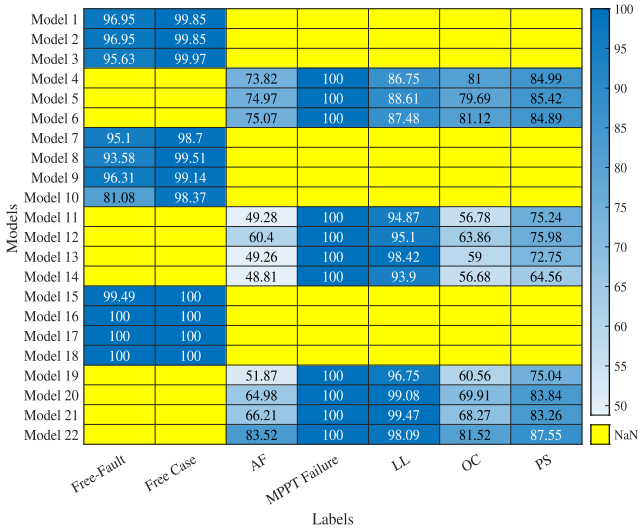

FIGURE 18. The labels precision accuracy at independent test datasets.

V. QUANTITATIVE EVALUATION OF FAULT CLASSIFIERS

In this section, the performance of detection and diagnosis modules of the ML models-based are evaluated by using simulation and experimental case studies.

A. SIMULATION VERIFICATION

Simulation tests are developed with the MATLAB®/ Simulink, as discussed previously, section IV.

The optimal models ($\mathcal{M}_1$ to $\mathcal{M}_{22}$) that give minimum cross-validation error during the BO tuning process as shown in Fig. 16, they will be tested using the independent testing datasets (D^{Test}) to investigate their generalization performance and then define the optimal ML model at each module. Besides, the prediction time of these models is also considered, as enlisted in Table 6.

The obtained models are compared in terms of different aspects as given in Tables 7 to 9. After obtaining the optimal hyperparameters for the mentioned models, the considered classifiers are retrained again using these parameters with all collected datasets (D) for real-world use. The FDD framework has two modules as discussed previously. Based on the given input variables, the detection module output can either indicate healthy (Free-Fault) or fault conditions. While, the diagnosis module output can be AF, MPPT unit failure, LL, OC, or PS cases.

The most informative metrics to investigate the model performance in classifying the labels are the **recall** and **precision** indices. The recall and precision for each label of the obtained models are shown in Figs. 17 and 18, respectively.

1) PREDICTION TIME INDEX

The prediction time/speed is an important informative metric in monitoring systems generally, which is indicated in Table 6 for the designed models. It indicates the timing consumed by the model ($\mathcal{M}$) to predict the given instances (D^{Test}).

As given in Table 6, the DT models whether designed for detection or diagnosis give the highest prediction speed compared to other models in KNN or SVM.

Regarding KNN models, these models showed the lowest prediction speed. As discussed previously, increasing the number of neighbors leads to decreasing prediction speed by KNN models as observed in $\mathcal{M}_{10}$ based on the Cosine distance metric. Whereas, the number of neighbors of this model is high, it is set by 2698 according to the adopted tuning method. On the other hand, $\mathcal{M}_9$ and $\mathcal{M}_{13}$ based on the Mahalanobis distance metric show the worst prediction speed regardless of their number of neighbors. To sum up, the number of neighbors besides the employed distance metric highly affects the prediction speed.

TABLE 7. The obtained DT models based on different splitting criteria.

The DT Models Based on the CART algorithm	Fault Detection Module Based on DT Models			Fault Diagnosis Module Based on DT Models		
	Gini Index	Towing Rule	Deviance	Gini Index	Towing Rule	Deviance
Model Index	$\mathcal{M}_1$	$\mathcal{M}_2$	$\mathcal{M}_3$	$\mathcal{M}_4$	$\mathcal{M}_5$	$\mathcal{M}_6$
Consumed Time (s)	44.469	47.204	43.899	53.677	61.182	54.827
Optimized Number of Splits	30	30	25	590	578	624
Total Number of Nodes	61	61	51	1181	1157	1249
Number of Leaves	31	31	26	591	579	625
Tree Depth (root $\mapsto$ fariest leaf node)	9	9	11	28	31	32
CV Accuracy (%)	99.37	99.55	99.68	82.95	83.08	83.70
CV Misclassified Cost	57	41	29	1500	1488	1434
Test Accuracy (%)	99.73	99.73	99.79	85.30	85.68	85.68
Test Misclassified Cost	13	13	10	696	678	678
CV-Accuracy on all Datasets (%)	99.51	99.67	99.68	85.89	86.41	86.80
CV-Misclassified Cost on all Datasets	68	46	44	1909	1839	1786

TABLE 8. The obtained KNN models based on different distance metrics.

The KNN (DW-KNN) Models	Fault Detection Module Based on The KNN Models				Fault Diagnosis Module Based on The KNN Models			
	Euclidean	City Block	Mahalanobis	Cosine	Euclidean	City Block	Mahalanobis	Cosine
Model Index	$\mathcal{M}_7$	$\mathcal{M}_8$	$\mathcal{M}_9$	$\mathcal{M}_{10}$	$\mathcal{M}_{11}$	$\mathcal{M}_{12}$	$\mathcal{M}_{13}$	$\mathcal{M}_{14}$
Consumed Time (s)	96.402	84.058	667.16	701.39	118.33	118.46	659.51	191.3
Optimized Distance Weighting Function	Squared Inverse	Squared Inverse	Squared Inverse	Squared Inverse	Squared Inverse	Squared Inverse	Uniform	Inverse
Optimized Number of Neighbors	10	4	11	2698	352	65	18	80
CV Accuracy (%)	98.30	98.96	98.77	97.74	74.44	78.21	75.16	72.75
CV Misclassified Cost	155	95	112	207	2249	1917	2185	2397
Test Accuracy (%)	98.60	99.29	99.04	97.85	73.61	78.15	73.86	71.81
Test Misclassified Cost	69	35	47	106	1250	1035	1238	1335
CV-Accuracy on all Datasets (%)	98.9	99.5	99.3	98.0	74.91	80.12	73.12	73.47
CV-Misclassified Cost on all Datasets	159	74	104	276	3396	2690	3625	3590

2) FAULT DETECTION MODULE

The detection models attain high detection accuracy. This can be notable from the recall and precision indices, as shown in Figs. 17 and 18, respectively.

In the **Detection based DT**, among three splitting criteria, as given in Table 7. The $\boldsymbol{\mathcal{M}_3}$ based on Deviance achieves a high detection accuracy of 99.79% using the test data.

In the **Detection based KNN**, four models attained based on different distance metrics, as illustrated in Table 8. Based on Table 8, the City Block metric $\boldsymbol{\mathcal{M}_8}$ achieves high detection accuracy of 99.29% using the test data.

In the **Detection based SVM**, three models based different Kernel functions out of four functions as illustrated in Table 9, achieve the same high detection accuracy of 100% using the test data, namely, the Quadratic $\mathcal{M}_{16}$, Cubic $\mathcal{M}_{17}$, and GRB $\mathcal{M}_{18}$ Kernel. According to Figs. 17 and 18, the $\mathcal{M}_{16}$, $\mathcal{M}_{17}$, and $\mathcal{M}_{18}$ have high detection accuracies in comparison to all detection models either in the DT, KNN, or SVM. Therefore, the **final model** in the **detection module** can be one of them. Nevertheless, $\boldsymbol{\mathcal{M}_{17}}$ is preferably employed as the module since its prediction speed is higher than $\mathcal{M}_{18}$, as illustrated in Table 6, and it has a high cross-validation accuracy compared to $\mathcal{M}_{16}$, as given in Table 9.

3) FAULT DIAGNOSIS MODULE

The setups of the classifiers applied in detection will be investigated also in the diagnosis module.

In the **Diagnosis based DT**, the Gini Index $\mathcal{M}_4$ gives less diagnostic accuracy than Towing Rule $\mathcal{M}_5$ and Deviance $\mathcal{M}_6$. The $\mathcal{M}_5$ and $\mathcal{M}_6$ have similar diagnostic accuracy of 85.68% on the test data, however, $\mathcal{M}_5$ is preferable than $\mathcal{M}_6$. Although the former model has a low cross-validation accuracy compared to $\mathcal{M}_6$, the number of its nodes is less, as given in Table 7, which simplifies its practical application.

TABLE 9. The obtained SVM models based on different Kernel functions.

The SVM Models	Fault Detection Module Based on The SVM Models				Fault Diagnosis Module Based on The SVM Models			
	Linear	Quadratic	Cubic	GRB	Linear	Quadratic	Cubic	GRB
Model Index	$\mathcal{M}_{15}$	$\mathcal{M}_{16}$	$\mathcal{M}_{17}$	$\mathcal{M}_{18}$	$\mathcal{M}_{19}$	$\mathcal{M}_{20}$	$\mathcal{M}_{21}$	$\mathcal{M}_{22}$
Consumed Time (s)	6395.4	7174.1	2993.5	418.78	16564	24700	13640000	13472
Optimized Penalty Parameter	510.2906	25.757	.15396	655.7566	8.5597	.62949	.0055898	978.587
Optimized Kernel Width	Not Applicable	Not Applicable	Not Applicable	1.9844	Not Applicable	Not Applicable	Not Applicable	1.1865
Optimized Multi-Class Approach	Not Applicable	Not Applicable	Not Applicable	Not Applicable	OVO	OVO	OVA	OVO
Number of Binary SVM CLs	1	1	1	1	10	10	5	10
CV Accuracy (%)	99.97	99.97	99.98	99.98	76.24	82.64	82.49	89.37
CV Misclassified Cost	2	2	1	1	2090	1527	1540	935
Test Accuracy (%)	99.97	100	100	100	75.38	81.73	82.41	89.84
Test Misclassified Cost	1	-	-	-	1166	865	833	481
CV-Accuracy on all Datasets (%)	99.95	99.97	99.99	99.99	76.04	81.81	82.80	90.59
CV-Misclassified Cost on all Datasets	6	3	1	1	3243	2461	2327	1273

In the **Diagnosis based KNN**, as illustrated in Table 8, the City Block metric $\mathcal{M}_{12}$ achieves high diagnostic accuracy of 78.15% using the test data compared to all KNN models for diagnostic. The data structure, number of neighbors, and distance metric impose the most appropriate weighting function to be applied. Hence, the squared inverse has been set as the most appropriate weighting function to all detection KNN models according to the BO. However, the weighting function applied with distance metrics in diagnosis models KNN based has differed. As, the Mahalanobis $\mathcal{M}_{13}$ and the Cosine $\mathcal{M}_{14}$ metric, which are based on uniform and inverse functions, respectively. This indicates the importance of tuning and advanced setup as well. As shown in Figs. 17 and 18, the KNN models, either designed for detection or diagnosis modules, are not suitable to the FDD framework. Whereas, their recall and precision indices show a high misclassification cost compared to DT and SVM models.

In the **Diagnosis based SVM**, the GRB Kernel $\mathcal{M}_{22}$ achieves the highest diagnostic accuracy of 89.84% using the test data compared to all diagnostic models whether in DT, KNN, or SVM, as shown in Figs. 17 and 18. Hence, it is promoted to be the **final model** in the **diagnosis module**. The most effective multi-classification approach during the tuning phase is OVO for all Kernel functions except for $\mathcal{M}_{21}$ with Cubic Kernel which is based on the OVA approach.

4) FAULT CLASSES

Increasing the number of classes in the classifier(s) may lead to an increasing the possibility of a misclassification cost. Hence, in this study, two separate modules have been introduced to enhance the integrity of the fault modules, which can be notable from the obtained results. It may be noted that with two-class classification, the attained accuracies for the detection models are more than 90%. However, the performance reduces as the number of classes increases as illustrated by the diagnostic models, where the number of labels is more than 2.

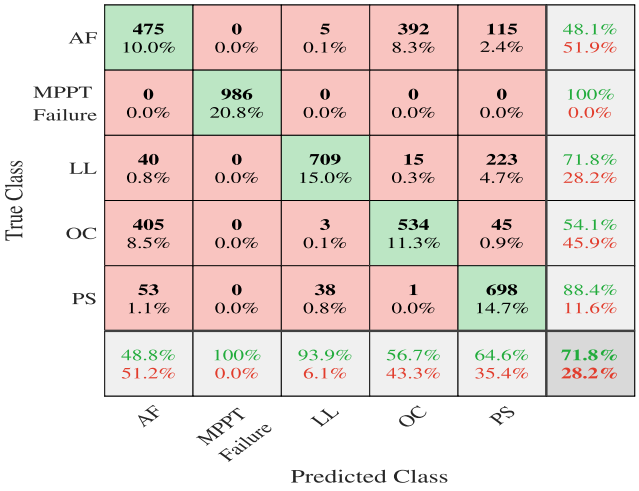

FIGURE 19. The *CM* for the worst diagnostic model, KNN ($\mathcal{M}_{14}$).

The misclassification rates of the diagnostic models can be interpreted by inspecting the recall and precision indices given in Figs. 17 and 18, respectively. It is observed that, the AF and OC fault cases show the lowest recall and precision percentages among all fault cases. This is simply because there are many similar scenarios between these two fault cases, which can also be observed by the *CM* associated with the worst $\mathcal{M}_{14}$ and the optimal $\mathcal{M}_{22}$ diagnostic model, as shown in Figs. 19 and 20, respectively. Similarly, there is also a large similarity between LL fault and PS case. In contrast to all fault cases, the MPPT unit failure case shows the highest true classification rate. This can be interpreted by Fig. 11, since, there is no overlapping between this fault case and other faulty cases in voltage and/or gamma values.

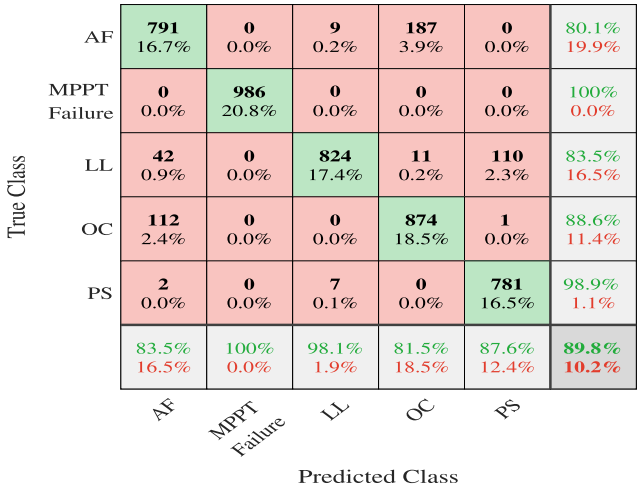

FIGURE 20. The CM for the optimal diagnostic model, SVM ($\mathcal{M}_{22}$).

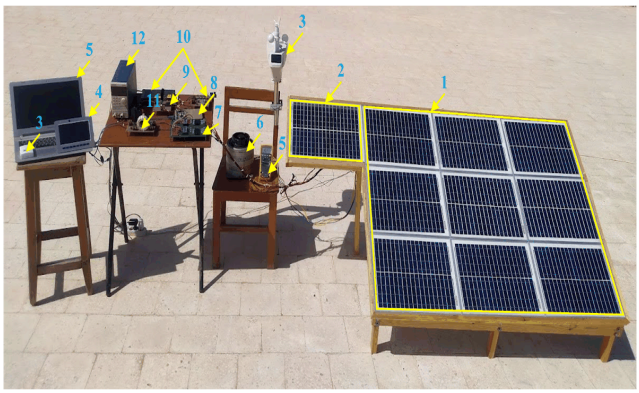

FIGURE 21. Experimental setup: 1) PV array, 2) reference PV module, 3) and 4) weather station transmitter and receiver appliances, respectively (for reference only), 5) data station, 6) fault impedance, 7) PV combiner and sensing circuit, 8) microcontroller board, 9) drive circuit, 10) power circuit, 11) boost converter, 12) DC-electronic load.

B. EXPERIMENTAL VERIFICATION

Experimental tests are developed to investigate the performance of the DT, KNN, and SVM practically.

Fig. 21 illustrates the prototype PV system implemented in this study. It consists of a PV array ($m_s = 3$ x $n_p = 3$), a DC-DC boost converter with an MPPT programmed with Perturb and Observe algorithm, and a DC-electronic load. The main experimental setup parameters are given in Table 10. A 100 V DC-DC boost converter is designed according to the operating characteristics of the PV array. The experimental system is not grounded. So, two OCPDs at each string have been installed, as discussed previously. However, it is noted that none of the tested faults have succeeded to melt any fuse.

The required attributes, namely, PV array voltage and current, solar irradiance, temperature are extracted and analyzed via the data station. These variables are extracted using a microcontroller board based on the ATmega328P (Arduino UNO) and the MATLAB®/Simulink platform. As shown in Fig. 21, the reference module used to obtain the incident solar irradiance and temperature has been placed in the same location and position as other modules in the array. Also, it has identical electrical specifications as the working modules.

Total 1362 faults were collected under both clear and cloudy days. Each class (Free-Fault, AF, LL, OC, PS, MPPT unit failure) has 227 instances. The considered faults are recorded at a 5 kHz sampling frequency. The hybrid faults also have been considered in practice for all permanent faults.

TABLE 10. Experimental setup.

Device	Parameter
PV Module	Polycrystalline Silicon 20 W, V_{ocm}^{STC} = 22.32 V, V_{mp}^{STC} = 18 V, I_{scm}^{STC} = 1.22 A, I_{mp}^{STC} = 1.11 A
Fuse	Threshold $\geq 1.56 I_{scs}^{STC}$ = 2 A
DC-Electronic Load	Model: KUNKIN - KP184

TABLE 11. Experimental results for the detection and diagnosis models using the considered classifiers.

Detection Models	Test Accuracy (%)	Diagnosis Models	Test Accuracy (%)
DT Models		**DT Models**	
Gini Index	98.31	Gini Index	83.37
Towing Rule	98.31	Towing Rule	85.89
Deviance	99.15	Deviance	86.14
KNN Models		**KNN Models**	
Euclidean	95.58	Euclidean	76.57
City Block	96.00	City Block	80.10
Mahalanobis	96.00	Mahalanobis	76.57
Cosine	92.43	Cosine	75.31
SVM Models		**SVM Models**	
Linear	99.36	Linear	85.64
Quadratic	99.57	Quadratic	87.65
Cubic	99.78	Cubic	87.90
GRB	99.57	GRB	88.16

TABLE 12. The recall and precision indices for SVM-based on Cubic Kernel for detection and GRB Kernel for diagnosis module.

Index (%)	Class						
	Normal	Fault	AF	MPPT	LL	OC	PS
Recall	78/79	397/397	64/80	70/79	68/80	72/79	76/79
Precision	78/78	397/398	64/83	70/70	68/78	72/85	76/81

TABLE 13. Analyzing the similarities and differences between the final FDD algorithm framework and methods in [18], [21], [23], [26], and [27].

Case Study	Framework	Method	Objective	Hybrid Faults Consideration	Low-Irradiation Consideration	Input Data
Proposed Algorithm	Detection Diagnosis	SVM-Cubic SVM-GRB	Free-Fault, AF, LL, OC, PS, and MPPT Failure	All permanent faults considered with PS	✓	G, T, V_{PV}, and I_{PV}
[18]	Detection Diagnosis	DT-Entropy DT-Entropy	Free-Fault, LL, OC, and PS	Not Focused	Not Focused	V_{PV}, I_{PV}, time, V^*_{ocm}, I^*_{scm}, and (T: module & ambient)
[21]	Detection Diagnosis	PNN	Free-Fault and LL	Not Focused	Not Focused	G, T, V_{PV}, and I_{PV}
[23]	Detection Diagnosis	GBSSL	Free-Fault, LL, and OC	Not Focused	✓	V_{PV}, I_{PV}, V^*_{ocm}, and I^*_{scm}
[26]	Diagnosis	MSD with SVM	Free-Fault and LL	Considering a subset of scenarios	✓	V_{PV} and I_{PV}
[27]	Diagnosis	CNN through 2D scalograms	Free-Fault, AF, LL, OC, and PS	Considering only LL and OC faults	✓	G, T, V_{PV}, I_{PV}, V_{oc}, I_{sc}, and boost output voltage and current

Different values from R_{AF} and R_{fault} enlisted in Table 2 are tested to generate instances for LL and AF, respectively. The setups used to attain the models ($\mathcal{M}_1$ to $\mathcal{M}_{22}$) have been adopted to obtain these models in practice as well. Table 11 gives the experimental results using all considered classifiers and their setups. These models have been trained and tested with 65% and 35% of the collected data, respectively.

As given in Table 11, the detection and diagnosis models based on DT and SVM show high detection and diagnostic accuracy compared to all KNN models, which fully complies with the simulation study. It is observed that the SVM based on Cubic and GRB Kernel testified an excellent detection and diagnostic accuracy of 99.78% and 88.16%, respectively among all attained models using experimental test data. Also, these setups for the SVM classifier gave the best performance in a simulation study as well. The recall and precision indices in each class label (fault case) of the optimal SVM models are given in Table 12.

C. COMPARISON WITH OTHER METHODS

Based on the previous simulation and experimental verification, the SVM based on Cubic and GRB Kernel testified the most suitable models for discriminating and distinguishing between the considered faults in both carried out studies.

The final FDD framework is compared with five methods that have been introduced in the available literature, namely, DT based on entropy splitting criterion [18], Probabilistic Neural Network (PNN) [21], Graph-Based Semi-Supervised Learning (GBSSL) [23], Multi-Resolution Signal Decomposition (MSD) with SVM [26], Convolutional Neural Network (CNN) through two-dimensional (2D) scalograms [27]. The comparison is carried out based on multiple qualitative aspects, as depicted in Table 13. This comparison highlights the main merits of the proposed FDD algorithm in reducing the number of required sensors, while focusing on different fault cases and scenarios. Upon screening Table 13, the [27] has focused on the same fault cases considered in this study except MPPT failure case and AF with shading scenarios. Hence, for a fair quantitative comparison, the final framework is compared with [27]. In the final framework, the SVM based on Cubic Kernel shows a high detection accuracy of 100% (simulation) and 99.78% (experimental), while a diagnostic accuracy of 89.84% (simulation) and 88.16% (experimental) has been accomplished by the SVM based on GRB Kernel on test datasets. While in [27], the diagnostic accuracy is 73.53% (simulation). It is observed that the final framework outperforms [27] in classifying similar fault cases. Furthermore, it has a reasonable application cost and can be adapted to be employed in large-scale PV systems.

VI. CONCLUSION

This paper introduced a fault detection and diagnosis algorithm for PV array, which is able to discriminate different fault types, namely, AF, LL, OC, MPPT unit failure, and PS under a wide range of unexpected scenarios. A comparative case study was carried out between three supervised ML classifiers, namely, DT, KNN, and SVM under different setups each. Based on this study, 11 ML models have been designed for the detection and another 11 models for the diagnosis module. These models have been obtained based on both simulation and experimental results. Multiple performance indices, namely, prediction speed, test and cross-validation accuracy, and recall and precision accuracy have been involved to define the most suitable model in each fault module. It was noted that by defining the suitable attributes given to the FDD framework and by adopting the proper hyperparameters when tuning the fault module-based ML models, the fault modules yielded the best performance for the detection and diagnosis of the considered PV array faults. Among these models, the SVM model based on Cubic

and GRB Kernel show an excellent detection and diagnosis performance, respectively, which is verified based on both simulations and experiments.

REFERENCES

[1] *The International Energy Outlook 2013*, U.S. Energy Information Administration (EIA), Washington, DC, USA, 2013.

[2] M. M. Badr, M. S. Hamad, A. S. Abdel-Khalik, and R. A. Hamdy, "Fault detection and diagnosis for photovoltaic array under grid connected using support vector machine," in *Proc. IEEE Conf. Power Electron. Renew. Energy (CPERE)*, Aswan City, Egypt, Oct. 2019, pp. 546–553.

[3] 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 Trans. Ind. Electron.*, vol. 60, no. 9, pp. 3784–3795, Sep. 2013.

[4] A. Triki-Lahiani, A. B. Abdelghani, and I. Slama-Belkhodja, "Fault detection and monitoring systems for photovoltaic installations: A review," *Renew. Sustain. Energy Rev.*, vol. 82, pp. 2680–2692, Feb. 2018.

[5] D. S. Pillai and N. Rajasekar, "A comprehensive review on protection challenges and fault diagnosis in PV systems," *Renew. Sustain. Energy Rev.*, vol. 91, pp. 18–40, Aug. 2018.

[6] S. K. Firth, K. J. Lomas, and S. J. Rees, "A simple model of PV system performance and its use in fault detection," *Sol. Energy*, vol. 84, pp. 624–635, Apr. 2010.

[7] D. E. Collier and T. S. Key, "Electrical fault protection for a large photovoltaic power plant inverter," in *Proc. Conf. Rec. 20th IEEE Photovoltaic Spec. Conf.*, Las Vegas, NV, USA, Sep. 1988, pp. 1035–1042.

[8] *Article 690-Solar Photovoltaic Systems*, U.S. National Electrical Code, Washington, DC, USA, pp. 623–638, 2014, pp. 623–638.

[9] M. K. Alam, F. H. Khan, J. Johnson, and J. Flicker, "PV faults: Overview, modeling, prevention and detection techniques," in *Proc. IEEE 14th Workshop Control Modeling Power Electron. (COMPEL)*, Jun. 2013, pp. 1–7.

[10] A. Chouder and S. Slivestre, "Automatic supervision and fault detection of PV systems based on power losses analysis," *Energy Convers. Manage.*, vol. 51, no. 10, pp. 1929–1937, 2010.

[11] M. Miwa, S. Yamanaka, H. Kawamura, H. Ohno, and H. Kawamura, "Diagnosis of a power output lowering of PV array with a (-dI/dV)-V characteristic," in *Proc. IEEE 4th World Conf. Photovoltaic Energy Conf.*, May 2006, pp. 2442–2445.

[12] R. Hariharan, M. Chakkarapani, G. S. Ilango, and C. Nagamani, "A method to detect photovoltaic array faults and partial shading in PV systems," *IEEE J. Photovolt.*, vol. 6, no. 5, pp. 1278–1285, Sep. 2016.

[13] S. Silvestre, M. A. da Silva, A. Chouder, D. Guasch, and E. Karatepe, "New procedure for fault detection in grid connected PV systems based on the evaluation of current and voltage indicators," *Energy Convers. Manage.*, vol. 86, no. 10, pp. 241–249, 2014.

[14] N. Gokmen, E. Karatepe, B. Celik, and S. Silvestre, "Simple diagnostic approach for determining of faulted PV modules in string based PV arrays," *Sol. Energy*, vol. 86, no. 11, pp. 3364–3377, Nov. 2012.

[15] A. Drews, A. C. de Keizer, H. G. Beyer, E. Lorenz, J. Betcke, W. G. J. H. M. van Sark, W. Heydenreich, E. Wiemken, S. Stettler, P. Toggweiler, S. Bofinger, M. Schneider, G. Heilscher, and D. Heinemann, "Monitoring and remote failure detection of grid-connected PV systems based on satellite observations," *Sol. Energy*, vol. 81, no. 4, pp. 548–564, Apr. 2007.

[16] N. Gokmen, E. Karatepe, S. Silvestre, B. Celik, and P. Ortega, "An efficient fault diagnosis method for PV systems based on operating voltage-window," *Energy Conver. Manage.*, vol. 73, pp. 350–360, Sep. 2013.

[17] P. Ducange, M. Fazzolari, B. Lazzerini, and F. Marcelloni, "An intelligent system for detecting faults in photovoltaic fields," in *Proc. 11th Int. Conf. Intell. Syst. Design Appl.*, Nov. 2011, pp. 1341–1346.

[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. 27th Annu. IEEE Appl. Power Electron. Conf. Expo. (APEC)*, Feb. 2012, pp. 93–99.

[19] M. S. Ramakrishna and S. N. Singh, "Modeling of PV system based on experimental data for fault detection using kNN method," *Sol. Energy*, vol. 173, pp. 139–151, Oct. 2018.

[20] H. Mekki, A. Mellit, and H. Salhi, "Artificial neural network-based modelling and fault detection of partial shaded photovoltaic modules," *Simul. Model. Pract. Theory*, vol. 67, pp. 1–13, Sep. 2016.

[21] E. Garoudja, A. Chouder, K. Kara, and S. Silvestre, "An enhanced machine learning based approach for failures detection and diagnosis of PV systems," *Energy Convers. Manage.*, vol. 151, pp. 496–513, Nov. 2017.

[22] M. N. Akram and S. Lotfifard, "Modeling and health monitoring of DC side of photovoltaic array," *IEEE Trans. Sustain. Energy*, vol. 6, no. 4, pp. 1245–1253, Oct. 2015.

[23] Y. Zhao, R. Ball, J. Mosesian, J. F. D. Palma, and B. Lehman, "Graph-based semi-supervised learning for fault detection and classification in solar photovoltaic arrays," *IEEE Trans. Power Electron.*, vol. 30, no. 5, pp. 2848–2858, May 2015.

[24] L. L. Jiang and D. L. Maskell, "Automatic fault detection and diagnosis for photovoltaic systems using combined artificial neural network and analytical based methods," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2015, pp. 1–8.

[25] L. C. Chen, P. J. Lin, J. Zhang, Z. C. Chen, Y. H. Lin, L. J. Wu, and S. Y. Cheng, "Fault diagnosis and classification for photovoltaic arrays based on principal component analysis and support vector machine," *IOP Conf. Ser., Earth Environ. Sci.*, vol. 188, no. 1, pp. 1–8, 2018.

[26] Z. Yi and A. H. Etemadi, "Line-to-line fault detection for photovoltaic arrays based on multiresolution signal decomposition and two-stage support vector machine," *IEEE Trans. Ind. Electron.*, vol. 64, no. 11, pp. 8546–8556, Nov. 2017.

[27] F. Aziz, A. U. Haq, S. Ahmad, Y. Mahmoud, M. Jalal, and U. Ali, "A novel convolutional neural network-based approach for fault classification in photovoltaic arrays," *IEEE Access*, vol. 8, pp. 41889–41904, 2020.

[28] W. Gao and R.-J. Wai, "A novel fault identification method for photovoltaic array via convolutional neural network and residual gated recurrent unit," *IEEE Access*, vol. 8, pp. 159493–159510, 2020.

[29] V. S. B. Kurukuru, F. Blaabjerg, M. A. Khan, and A. Haque, "A novel fault classification approach for photovoltaic systems," *Energies*, vol. 13, no. 2, p. 308, Jan. 2020.

[30] A. Y. Appiah, X. Zhang, B. B. K. Ayawli, and F. Kyeremeh, "Review and performance evaluation of photovoltaic array fault detection and diagnosis techniques," *Int. J. Photoenergy*, vol. 2019, pp. 1–19, Feb. 2019.

[31] J. Flicker and J. Johnson, "Electrical simulations of series and parallel PV arc-faults," in *Proc. IEEE 39th Photovoltaic Spec. Conf. (PVSC)*, Jun. 2013, pp. 3165–3172.

[32] M. K. Alam, F. Khan, J. Johnson, and J. Flicker, "A comprehensive review of catastrophic faults in PV arrays: Types, detection, and mitigation techniques," *IEEE J. Photovolt.*, vol. 5, no. 3, pp. 982–997, May 2015.

[33] A. Gupta, Y. K. Chauhan, and R. K. Pachauri, "A comparative investigation of maximum power point tracking methods for solar PV system," *Sol. Energy*, vol. 136, pp. 236–253, Oct. 2016.

[34] P. Bulanyi and R. Zhang, "Shading analysis & improvement for distributed residential grid-connected photovoltaics systems," in *Proc. 52nd Annu. Conf. Austral. Sol. Council*, May 2014, pp. 1–12.

[35] M. R. Maghami, H. Hizam, C. Gomes, M. A. Radzi, M. I. Rezadad, and S. Hajighorbani, "Power loss due to soiling on solar panel: A review," *Renew. Sustain. Energy Rev.*, vol. 59, pp. 1307–1316, Jun. 2016.

[36] K.-H. Chao, S.-H. Ho, and M.-H. Wang, "Modeling and fault diagnosis of a photovoltaic system," *Electric Power Syst. Res.*, vol. 78, no. 1, pp. 97–105, Jan. 2008.

[37] K. E. Pilario, M. Shafiee, Y. Cao, L. Lao, and S.-H. Yang, "A review of kernel methods for feature extraction in nonlinear process monitoring," *Processes*, vol. 8, no. 1, p. 24, Dec. 2019.

[38] S. J. Al'Aref, K. Anchouche, G. Singh, P. J. Slomka, K. K. Kolli, A. Kumar, M. Pandey, G. Maliakal, A. R. Van Rosendael, A. N. Beecy, and D. S. Berman, "Clinical applications of machine learning in cardiovascular disease and its relevance to cardiac imaging," *Eur. Heart J.*, vol. 40, no. 24, pp. 1975–1986, Jun. 2019.

[39] *Photovoltaic System Performance—Part 1: Monitoring*, document IEC 61724-1, International Electrotechnical Commission, 2017.

[40] S. U. Jan and I. Koo, "A novel feature selection scheme and a diversified-input SVM-based classifier for sensor fault classification," *J. Sensors*, vol. 2018, pp. 1–21, Sep. 2018.

[41] S. Yadav and S. Shukla, "Analysis of k-fold cross-validation over hold-out validation on colossal datasets for quality classification," in *Proc. IEEE 6th Int. Conf. Adv. Comput. (IACC)*, Feb. 2016, pp. 78–83.

[42] A. Maheshwari, N. Davendralingam, and D. A. DeLaurentis, "A comparative study of machine learning techniques for aviation applications," in *Proc. Aviation Technol., Integr., Oper. Conf.*, Jun. 2018, p. 3980.

[43] M. Ali, D.-H. Son, S.-H. Kang, and S.-R. Nam, "An accurate CT saturation classification using a deep learning approach based on unsupervised feature extraction and supervised fine-tuning strategy," *Energies*, vol. 10, no. 11, p. 1830, Nov. 2017.
[44] M. Suknovic, B. Delibasic, M. Jovanovic, M. Vukicevic, D. Becejski-Vujaklija, and Z. Obradovic, "Reusable components in decision tree induction algorithms," *Comput. Statist.*, vol. 27, no. 1, pp. 127–148, Mar. 2012.
[45] Y. Wang and S.-T. Xia, "Unifying attribute splitting criteria of decision trees by tsallis entropy," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Mar. 2017, pp. 2507–2511.
[46] M. E. H. Daho, N. Settouti, M. E. A. Lazouni, and M. E. A. Chikh, "Weighted vote for trees aggregation in random forest," in *Proc. Int. Conf. Multimedia Comput. Syst. (ICMCS)*, Apr. 2014, pp. 438–443.
[47] H. A. A. Alfeilat, A. B. A. Hassanat, O. Lasassmeh, A. S. Tarawneh, M. B. Alhasanat, H. S. E. Salman, and V. B. S. Prasath, "Effects of distance measure choice on K-nearest neighbor classifier performance: A review," *Big Data*, vol. 7, no. 4, pp. 221–248, Dec. 2019.
[48] D. M. Atallah, M. Badawy, A. El-Sayed, and M. A. Ghoneim, "Predicting kidney transplantation outcome based on hybrid feature selection and KNN classifier," *Multimedia Tools Appl.*, vol. 78, no. 14, pp. 20383–20407, Jul. 2019.
[49] Q. Cao, "Mixed weighted KNN for imbalanced datasets," *Int. J. Performability Eng.*, vol. 14, no. 7, pp. 1391–1400, 2018.
[50] S. Zhang, "Cost-sensitive KNN classification," *Neurocomputing*, vol. 391, pp. 234–242, May 2020.
[51] Z. Geler, V. Kurbalija, M. Radovanović, and M. Ivanović, "Comparison of different weighting schemes for the kNN classifier on time-series data," *Knowl. Inf. Syst.*, vol. 48, no. 2, pp. 331–378, Aug. 2016.
[52] C. Castaldello, A. Gubert, F. Galvanin, A. Casonato, R. Padrini, M. Barolo, and F. Bezzo, "A model-based support for diagnosing von Willebrand disease," *Comput. Aided Chem. Eng.*, vol. 40, pp. 2779–2784, Jan. 2017.
[53] G. Madzarov, D. Gjorgjevikj, and I. Chorbev, "A multi-class SVM classifier utilizing binary decision tree," *Informatica*, vol. 33, no. 2, pp. 233–242, 2009.
[54] Y. Zhang and L. Wu, "Classification of fruits using computer vision and a multiclass support vector machine," *Sensors*, vol. 12, no. 9, pp. 12489–12505, Sep. 2012.
[55] W. Jia, C. Xiu-Yun, Z. Hao, X. Li-Dong, L. Hang, and D. Si-Hao, "Hyper-parameter optimization for machine learning models based on Bayesian optimization," *J. Electron. Sci. Technol.*, vol. 17, no. 1, pp. 26–40, 2019.
[56] I. Syarif, A. Prugel-Bennett, and G. Wills, "SVM parameter optimization using grid search and genetic algorithm to improve classification performance," *Telkomnika*, vol. 14, no. 4, pp. 1502–1509, 2016.
[57] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *J. Mach. Learn. Res.*, vol. 13, pp. 281–305, Feb. 2012.
[58] 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.
[59] A. Tharwat, "Classification assessment methods," *Appl. Comput. Informat.*, vol. 17, no. 1, pp. 168–192, Jan. 2021.

MOHAMED M. BADR received the M.Sc. degree in electrical engineering from Alexandria University, Alexandria, Egypt, in 2020, where he is currently pursuing the Ph.D. degree in electrical engineering. His research interests include mechatronics, renewable energy, and power electronics.

MOSTAFA S. HAMAD (Senior Member, IEEE) received the B.Sc. and M.Sc. degrees in electrical engineering from Alexandria University, Alexandria, Egypt, in 1999 and 2003, respectively, and the Ph.D. degree in electrical engineering from the University of Strathclyde, Glasgow, U.K., in 2009. From 2010 to 2014, he was an Assistant Professor with the Department of Electrical and Control Engineering, College of Engineering and Technology, Arab Academy for Science, Technology and Maritime Transport (AASTMT), Alexandria, where he is currently a Professor. His research interests include power electronics applications in power quality, electric drives, distributed generation, HVDC transmission systems, and renewable energy.

AYMAN S. ABDEL-KHALIK (Senior Member, IEEE) received the B.Sc. and M.Sc. degrees in electrical engineering from Alexandria University, Alexandria, Egypt, in 2001 and 2004, respectively, and the Ph.D. degree in electrical engineering from Alexandria University, Alexandria, Egypt, and Strathclyde University, Glasgow, U.K., in 2009, under a dual channel program. He is currently a Professor with the Electrical Engineering Department, Faculty of Engineering, Alexandria University. His current research interests include electrical machine design and modeling, electric drives, energy conversion, and renewable energy. He also serves as an Associate Editor for IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS and *IET Electric Power Applications* journal, and the Editor-in-Chief for *Alexandria Engineering Journal*.

RAGI A. HAMDY (Senior Member, IEEE) received the B.Sc. and M.Sc. degrees from Alexandria University, Alexandria, Egypt, in 1991 and 1994, respectively, and the Ph.D. degree from Heriot-Watt University, U.K., in 1999. He is currently a Professor with the Electrical Engineering Department, Faculty of Engineering, Alexandria University. His current research interests include electric machines, electric drives, and power electronics.

SHEHAB AHMED (Senior Member, IEEE) received the B.Sc. degree in electrical engineering from Alexandria University, Alexandria, Egypt, in 1999, and the M.Sc. and Ph.D. degrees from the Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX, USA, in 2000 and 2007, respectively. He was with Schlumberger Technology Corporation, Houston, TX, USA, from 2001 to 2007, developing downhole mechatronic systems for oilfield service products. He was with Texas A&M University at Qatar, from 2007 to 2018. He is currently a Professor of electrical engineering with the CEMSE Division, King Abdullah University of Science and Technology (KAUST), Saudi Arabia. His research interests include mechatronics, solid-state power conversion, and electric machines.

EMAN HAMDAN received the B.Sc., M.Sc., and Ph.D. degrees in electrical engineering from Alexandria University, Egypt, in 2000, 2004, and 2012, respectively. She is currently an Assistant Professor with the Department of Marine Engineering Technology, College of Maritime Transport and Technology, Arab Academy for Science, Technology and Maritime Transport (AASTMT), Alexandria, Egypt. Her research interests include automatic control applications, coding techniques in MIMO networks, multi-sensors networks, and optimization techniques.