Fault Identification of Photovoltaic Array Based on Machine Learning Classifiers

MOHAMED M. BADR1,*, MOSTAFA S. HAMAD2, (SENIOR, IEEE), AYMAN S. ABDEL-KHALIK3, (SENIOR, IEEE), RAGI A. HAMDY1, (SENIOR, IEEE), SHEHAB AHMED2, (SENIOR, IEEE), AND EMAN HAMDAN4

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

*Corresponding author: Mohamed M. Badr (e-mail: mmustafabadr@gmail.com).

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 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]: non-linear 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 faulty conditions) and diagnose (i.e., distinguish one fault type from another) faults in the PV array. The PV array/system size, operation, and maintenance cost determine the most appropriate FDD technique to be recruited. Based on the available literature, FDD techniques in PV array could be categorized as Signal Processing Techniques (SPTs) [10]–[16], Artificial Intelligence (AI)-based techniques [17]–[22], and Hybrid Techniques (HTs) [23]–[29].

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, 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 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 FDD algorithm. Section V presents a comprehensive quantitative evaluation of the candidate classifiers to locate PV array faults. The simulation results are 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.
Mohamed M. Badr et al.: Fault Identification of PV Array Based on ML Classifiers

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

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\(^\circ\)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         | \(P_{STC}^{mp}\) | 250   | W    |
| Open-Circuit Voltage  | \(V_{STC}^{ocm}\) | 37.6  | V    |
| Short-Circuit Current | \(I_{STC}^{scm}\) | 8.81  | A    |
| Maximum Power Voltage | \(V_{STC}^{mp}\) | 30.4  | V    |
| Maximum Power Current | \(I_{STC}^{mp}\) | 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 module 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 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.
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.

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

$$I_N \geq 1.56 I_{stc}^{scs} \quad (2)$$

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.

FIGURE 3. The conventional protection systems in the PV array scheme.
The OCPD can easily clear a LL fault when the magnitude of $I_{\text{back}}$ is higher than $I_N$. Small $R_{\text{fault}}$ 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.

**Case 1:** Low solar irradiance levels, a small number of PV modules between fault points, and/or high $R_{\text{fault}}$ values yield a small $I_{\text{back}}$, which is insufficient to melt the fuse. Fig. 4 shows the relation between two LL fault examples with different $R_{\text{fault}}$ 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 resistance, which is opposite of the F5 fault case.

**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, $I_{\text{back}}$, since this current is blocked, 2) they add extra power losses, and 3) since $I_{\text{back}}$ is blocked, the PV array power may increase to the same power level corresponding to point $B$ without blocking diodes. As shown in Fig. 6 for the F4 case, the peaks of $|I_{\text{back}}|$ 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 $I_{\text{back}}$, the OCPD can properly detect and clear the F5 case depending on the response speed for the employed MPPT method, since the peak of $|I_{\text{back}}|$ corresponding to point $B$ is larger than $I_{\text{cl}}$ ($= 2.9I_{\text{s}}$) 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.

**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 $I_{\text{back}}$ 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 $|I_{\text{back}}|$ 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 $I_{\text{back}}$ 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 $R_{\text{fault}} = 0$ Ω.

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 $|I_{\text{back}}|$ 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 $I_{\text{back}}$, the OCPD can properly detect and clear the F5 case depending on the response speed for the employed MPPT method, since the peak of $|I_{\text{back}}|$ corresponding to point $B$ is larger than $I_{\text{cl}}$ ($= 2.9I_{\text{sc}}$) 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.

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.
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 or 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 proposed FDD algorithm framework has two main modules, as illustrated in Fig. 9, namely, DT based fault diagnosis, which will be recruited, as illustrated in Fig. 9, namely, DT based diagnosis module is triggering the 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

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.

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], [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].
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 (OVO or OVA) in SVM at diagnosis module only.

The three ML classifiers and their setups lead to different ML behaviors of ML models. So, Bayesian Optimization is adopted to choose the optimal model for each module based on the performance metrics (i.e., accuracy, precision, recall, F1-score, etc.).

The simulations are carried out using Matlab 2020a and running on LenovoTM 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.

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].

$$\gamma_{PV}(s_i) = \frac{P_{PV}(s_i)}{G(s_i)} = \frac{V_{PV}(s_i) \cdot I_{PV}(s_i)}{G(s_i)}$$  \hspace{1cm} (3)

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

### 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/m²      | 0:5:55°C    | -       | -         | 564              | 0           |
| Fault Condition        | 50:25:1200 W/m²      | 0:5:55°C    | 0%-20%:80% | $R_{AF}$ & $R_{fault}$ | 13536         | 1           |
| Arc Fault (F1)         | 50:25:1200 W/m²      | 0:5:55°C    | 0%-20%:80% | 20:20:100 $\Omega$ & 200 $\Omega$ | 2820         | 2           |
| MPPT Unit Failure (F2) | 50:25:1200 W/m²      | 0:5:55°C    | 0%-20%:80% | -         | 2820              | 3           |
| LL Fault (F4)          | 50:25:1200 W/m²      | 0:5:55°C    | 0%-20%:80% | 0:10:30 $\Omega$ | 2820         | 4           |
| OC Fault (F7, F8, and F9) | 50:25:1200 W/m²  | 0:5:55°C    | 0%-20%:80% | -         | 2820              | 5           |
| Shading (F10 and F11)  | 50:25:1200 W/m²      | 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.
Mohamed M. Badr et al.: Fault Identification of PV Array Based on ML Classifiers

FIGURE 10. The workflow for obtaining optimal supervised ML models for detection or diagnosis module.

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

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-$\gamma_{PV}$), 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].

$$\begin{align*}
\text{Normalization} & = \frac{x_{i|\varphi} - \text{Min} (\varphi)}{\text{Max} (\varphi) - \text{Min} (\varphi)} \\
\end{align*}$$

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

As indicated in (4), a single outlier or even a very small value in the attribute $\varphi$ may force the remaining values of the attribute $\varphi$ 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 $\varphi$ is called a $Z$-score and is obtained from (5).

$$Z - \text{score}_{\varphi} = \frac{\varphi - \mu_{\varphi}}{\sigma_{\varphi}}$$

where $Z - \text{score}_{\varphi}$ is standardized values of attribute $\varphi$, $\varphi$ is an attribute that is being standardized, $\mu_{\varphi}$ and $\sigma_{\varphi}$ are the mean and standard deviation of an attribute $\varphi$, 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.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/
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.](image1)

![FIGURE 13. The utilized input quantities after applying standardization.](image2)

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

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.

![FIGURE 14. Workflow for hold-out and k-fold cross-validation method.](image3)

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 \( D^{\text{Train}} \) are used to train and build the model. Then, the model is evaluated using testing datasets \( D^{\text{Test}} \) [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 test data for once. The error for each test fold is calculated to obtain the average classifier error, as indicated in (9).

\[
D = \{ (x_i, y_i), 1 \leq i \leq n, x_i \in \mathbb{R}^d, y_i \in \{ \theta_1, \ldots, \theta_c \} \} \tag{6}
\]

\[
\text{Training Samples} = \frac{(k-1)D^{\text{Train}}}{k} \tag{7}
\]

\[
\text{Testing Samples} = \frac{D^{\text{Train}}}{k} \tag{8}
\]

\[
A_{CL}^E = \frac{1}{k} \sum_{1}^{k} E_k \tag{9}
\]

where \( D = (D^{\text{Train}} + D^{\text{Test}}) \) is the dataset of \( n \) samples in a \( d \)-dimensional space belonging to one of \( y_i \) labels, \( \theta \) 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 ($D_{\text{Train}}$, for detection = 9165 and diagnosis = 8799 samples) is used for the tuning phase based on Bayesian Optimization. While, the remaining 35\% of the datasets ($D_{\text{Test}}$, 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], [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 ($P_x$), 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 (i.e., \( K = \sqrt{n} \)). 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 Viscissitude \([47], [49]\). A comparative study between distance metrics enlisted in Table 3 has been performed either in the detection or diagnosis modules.

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

| Distance Metrics       | Equation                                      |
|------------------------|-----------------------------------------------|
| Euclidean              | \( d_{ij} = \sqrt{\sum_{d=1}^{D} (x_{ij} - P_{t|v})^2} \) |
| City Block             | \( d_{ij} = \sum_{d=1}^{D} |x_{ij} - P_{t|v}| \) |
| Mahalanobis            | \( d_{ij} = \sqrt{(x_{ij} - P_{t|v})^T \Sigma^{-1} (x_{ij} - P_{t|v})} \) |
| Cosine                 | \( d_{ij} = 1 - \frac{\sum_{t=1}^{D} (x_{ij} - P_{t|v})^2}{\sum_{t=1}^{D} (P_{t|v})^2} \) |

Assigning the class label \( \theta_{P_t} \) 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 \([48]\).

\[
\theta_{P_t} = \text{argmax}_v \sum_{(x_i, y_i) \in D_z} \delta(v, y_i) \tag{10}
\]

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 \((w_{ij})\), where \( w_{ij} \) indicates the weight of the \( i^{th} \) nearest neighbor, and \( \delta \) 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 \([49], [50]\).

\[
\theta_{P_t} = \text{argmax}_v \sum_{(x_i, y_i) \in D_z} w_{ij} \delta(v, y_i) \tag{11}
\]

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 \( \theta_{P_t} \) can also be obtained by minimizing the expected classification cost \((14)\) \([52]\), this method applied herein.

\[
w(x_{ij|v}, P_{t|v}) = \frac{1}{\text{Distance}(x_{ij|v}, P_{t|v})} \tag{12}
\]

\[
w(x_{ij|v}, P_{t|v}) = \left(\frac{1}{\text{Distance}(x_{ij|v}, P_{t|v})}\right)^2 \tag{13}
\]

\[
\theta_{P_t} = \text{argmin}_{\theta_{P_t} = \theta_{1}, \ldots, \theta_{c}} \sum_{y_i = \theta_{1}}^{\theta_{c}} \hat{p}(y_i | P_t) C(\theta_{P_t} | y_i) \tag{14}
\]

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 \( C(\theta_{P_t} | y_i) \) indicates the true misclassification cost to classify an observation as \( \theta_{P_t} \) when its true is \( y_i \).

The influence of Distance Weighted KNN (DWM-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.

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

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].

\[
\begin{align*}
\text{Linear Kernel} &= x_i^T . x_j \\
\text{Polynomial Kernel} &= (1 + x_i^T . x_j)\rho \\
\text{GRB Kernel} &= \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)
\end{align*}
\]

where \(\rho\) denotes the degree for the Polynomial Kernel and \(\sigma\) 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].

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 \((\lambda^*)\) for a black-box function \(f_1\), 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 \(n \in D_{Train}\).

\[
\lambda^* = \arg\max_{\Lambda} f(\lambda)
\]

where \(\Lambda\) 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 \(\Lambda\)

**Output:** Optimal Hyperparameter(s)

for \(t = 1, 2, 3, 4, \ldots, 50\) do

Find: \(\lambda_t = \arg\max_{\Lambda} \hat{f}(\lambda|H_{1:t-1})\)

Evaluate: \(\mathcal{F}_t = f(\lambda_t)\)

Update: \(H_{1:t} = \{H_{1:t-1}, \lambda_t, \mathcal{F}_t\}\)

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., \(M_1 \to M_{22}\)) are given in Fig. 16, where the vertical axis represents the minimum CV error and the horizontal axis represents the number of iterations of the BO. The \(M_1 \to M_{6}, M_7 \to M_{14}, M_{15} \to M_{22}\) represents the optimal DT, KNN, and SVM models, respectively.

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 \((N_c \times N_c)\) which thereby helps the designer to understand the
TABLE 5. Confusion Matrix for the multi-classification problem.

| Class         | Predicted |          |          |          |
|---------------|-----------|----------|----------|----------|
|               | $\theta_1$ | $\theta_2$ | $\theta_3$ | $\theta_M$ |
| True $\theta_1$ | TP$_{\theta_1}$ (1, 1) | $E_{\theta_1\theta_2}$ (1, 2) | $E_{\theta_1\theta_M}$ (1, $M$) |          |
| True $\theta_2$ | $E_{\theta_2\theta_1}$ (2, 1) | TP$_{\theta_2}$ (2, 2) | $E_{\theta_2\theta_M}$ (2, $M$) |          |
| True $\theta_M$ | 0         | 0        | 0        | 0        |

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

$$Accuracy = \frac{\sum_{N=1}^{NC} CM(N, N)}{\sum_{N=1}^{NC} \sum_{M=1}^{CM} CM(N, M)}$$  \hspace{1cm} (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.

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 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 ($\theta$) 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).

Recall: Indicates the effectiveness of the classifier to classify $\theta_N$ correctly to the total number of samples represents in $\theta_N$ as indicated in (20).

$$Recall(\theta_N) = \frac{CM(N, N)}{\sum_{M=1}^{NC} CM(N, M)}$$ \hspace{1cm} (20)

Precision: Represents the proportion of samples in $\theta_M$ that are correctly classified by the classifier to the total number of predictive samples in $\theta_M$ as given in (21).

$$Precision(\theta_M) = \frac{CM(M, M)}{\sum_{N=1}^{NC} CM(N, M)}$$ \hspace{1cm} (21)
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 (\(M\)) 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 \(M_{10}\) 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, \(M_9\) and \(M_{13}\) 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.

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 \(M_3\) based on Deviance achieves a high detection accuracy of 99.79% using the test data.

### 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 (\(M_1\)) | 3.537                | Gini Index (\(M_4\)) | 4.429                |
| Towing Rule (\(M_2\)) | 3.093                | Towing Rule (\(M_5\)) | 6.794                |
| Deviance (\(M_3\))  | 3.689                | Deviance (\(M_6\))  | 7.231                |
| **KNN Models**    |                      | **KNN Models**    |                      |
| Euclidean (\(M_7\)) | 59.678               | Euclidean (\(M_{11}\)) | 376.681               |
| City Block (\(M_8\)) | 17.09                | City Block (\(M_{12}\)) | 116.838               |
| Mahalanobis (\(M_9\)) | 2976.898             | Mahalanobis (\(M_{13}\)) | 2820.551               |
| Cosine (\(M_{10}\))  | 3381.214             | Cosine (\(M_{14}\))  | 429.739               |
| **SVM Models**    |                      | **SVM Models**    |                      |
| Linear (\(M_{15}\)) | 4.376                | Linear (\(M_{19}\)) | 40.502               |
| Quadratic (\(M_{16}\)) | 4.46                | Quadratic (\(M_{20}\)) | 103.053               |
| Cubic (\(M_{17}\))  | 6.401                | Cubic (\(M_{21}\))  | 230.522               |
| GRB (\(M_{18}\))   | 8.043                | GRB (\(M_{22}\))   | 183.389               |

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 \(M_8\) 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 \(M_{17}\), Cubic \(M_{17}\), and GRB \(M_{17}\) Kernel. According to Figs. 17 and 18, the \(M_{16}\), \(M_{17}\), and \(M_{18}\) 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, \(M_{17}\) is preferably employed as the module since its prediction speed is higher than \(M_{18}\), as illustrated in Table 6, and it has a high cross-validation accuracy compared to \(M_{16}\), as given in Table 9.

![FIGURE 17. The labels recall accuracy at independent test datasets.](image)

![FIGURE 18. The labels precision accuracy at independent test datasets.](image)
### TABLE 7. The obtained DT models based on different splitting criteria.

| Model Index | Fault Detection Module Based on DT Models | Fault Diagnosis Module Based on DT Models |
|-------------|-----------------------------------------|------------------------------------------|
|             | Gini Index | Towing Rule | Deviance | Gini Index | Towing Rule | Deviance |
| M<sub>1</sub> | 44.469     | -           | 43.899   | M<sub>4</sub> | 53.677     | 61.182    | 54.827   |
| M<sub>2</sub> | 47.204     | -           | -        | M<sub>5</sub> | 590        | 578      | 624      |
| Optimized   | 30         | 30          | 25       | Total Number of Nodes | 61 | 61 | 51 |
| Number of Splits | 61 | 61 | 51 |
| Number of Leaves | 31 | 31 | 26 |
| Tree Depth (root → farthest leaf node) | 9 | 9 | 11 |
| CV Accuracy (%) | 99.37 | 99.55 | 99.68 |
| CV Misclassified Cost | 57 | 41 | 29 |
| Test Accuracy (%) | 99.73 | 99.73 | 99.79 |
| Test Misclassified Cost | 13 | 13 | 10 |
| CV-Accuracy on all Datasets (%) | 99.51 | 99.67 | 99.68 |
| CV-Misclassified Cost on all Datasets | 68 | 46 | 44 |

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

| Model Index | 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 |
| M<sub>7</sub> | 96.402    | 84.058     | 667.16     | 701.39 | M<sub>11</sub> | 118.33     | 118.46     | 659.51    | 191.3    |
| M<sub>8</sub> | 98.30     | 98.96      | 98.77      | 97.74  | M<sub>12</sub> | 74.44      | 78.21      | 75.16     | 72.75    |
| Optimized Distance | Squared Inverse | Squared Inverse | Squared Inverse | Squared Inverse | Optimized Distance |
| Weighting Function | Squared Inverse | Squared Inverse | Squared Inverse | Squared Inverse | Optimized Distance |
| Optimized Number of Neighbors | 10 | 4 | 11 | 2698 |
| CV Accuracy (%) | 98.80 | 99.29 | 99.04 | 98.85 |
| CV Misclassified Cost | 155 | 95 | 112 | 207 |
| Test Accuracy (%) | 98.60 | 99.29 | 99.04 | 98.85 |
| Test Misclassified Cost | 69 | 35 | 47 | 106 |
| CV-Accuracy on all Datasets (%) | 98.9 | 99.5 | 99.3 | 98.0 |
| CV-Misclassified Cost on all Datasets | 159 | 74 | 104 | 276 |

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

| Model Index | Fault Detection Module Based on The SVM Models | Fault Diagnosis Module Based on The SVM Models |
|-------------|-----------------------------------------------|-----------------------------------------------|
|             | Linear | Quadratic | Cubic | GRB | Linear | Quadratic | Cubic | GRB |
| M<sub>15</sub> | 6395.4 | 7174.1 | 2993.5 | 418.78 | M<sub>19</sub> | 16564 | 24700 | 13640000 | 13472 |
| M<sub>16</sub> | 510.2906 | 25.757 | .15396 | 655.7566 | M<sub>20</sub> | 8.5597 | .62949 | .0055898 | 978.587 |
| Optimized Penalty Parameter | Not Applicable | Not Applicable | Not Applicable | 1.9844 | Optimized Penalty Parameter | Not Applicable | Not Applicable | Not Applicable | 1.1865 |
| Optimized Kernel Width | Not Applicable | Not Applicable | Not Applicable | Not Applicable | Optimized Penalty Parameter | Not Applicable | Not Applicable | Not Applicable | Not Applicable |
| Number of Binary SVM CLs | 1 | 1 | 1 | 1 | Number of Binary SVM CLs | 1 | 1 | 1 | 1 |
| CV Accuracy (%) | 99.97 | 99.97 | 99.98 | 99.98 | CV Accuracy (%) | 99.97 | 99.97 | 99.99 | 99.99 |
| CV Misclassified Cost | 2 | 2 | 1 | 1 | CV Misclassified Cost | 1 | .1 | - | - |
| Test Accuracy (%) | 99.97 | 100 | 100 | 100 | Test Accuracy (%) | 99.97 | 100 | 100 | 100 |
| Test Misclassified Cost | 1 | - | - | - | Test Misclassified Cost | 1 | - | - | - |
| CV-Accuracy on all Datasets (%) | 99.95 | 99.97 | 99.99 | 99.99 | CV-Accuracy on all Datasets (%) | 99.95 | 99.97 | 99.99 | 99.99 |
| CV-Misclassified Cost on all Datasets | 6 | 3 | 1 | 1 | CV-Misclassified Cost on all Datasets | 6 | 3 | 1 | 1 |
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 $M_4$ gives less diagnostic accuracy than Towing Rule $M_5$ and Deviance $M_6$. The $M_5$ and $M_6$ have similar diagnostic accuracy of 85.68% on the test data, however, $M_5$ is preferable than $M_6$. Although the former model has a low cross-validation accuracy compared to $M_6$, the number of its nodes is less, as given in Table 7, which simplifies its practical application.

In the Diagnosis based KNN, as illustrated in Table 8, the City Block metric $M_{12}$ 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 is differed. As, the Mahalanobis $M_{13}$ and the Cosine $M_{14}$ 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 $M_{22}$ 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 $M_{21}$ 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.

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 $M_{14}$ and the optimal $M_{22}$ 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.

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 ($m_s = 3 \times n_p = 3$), 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.
TABLE 10. Experimental Setup.

| Device                  | Parameter                  | Parameter                  |
|-------------------------|----------------------------|----------------------------|
| PV Module               | $V_{scm}^{STC} = 22.32 \, V$ | $V_{mp}^{STC} = 18 \, V$   |
|                         | $I_{scm}^{STC} = 1.22 \, A$ | $I_{mp}^{STC} = 1.11 \, A$ |
| Fuse                   | Threshold $= 1.56I_{mp}^{STC} = 1.99 \, A$ |                         |
| DC-Electronic Load     | Model: KUNKIN - KP184      |                            |

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 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.

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

| Detection Models | Accuracy (%) | Test Models | Accuracy (%) | Test Models |
|------------------|--------------|-------------|--------------|-------------|
| DT Models        |              | SVM Models  |              |             |
| Gini Index       | 98.31        | Linear      | 99.36        |             |
| Towing Rule      | 98.31        | Quadratic   | 99.57        |             |
| Deviance         | 99.15        | Cubic       | 99.78        |             |
| Cosine           | 92.43        | GRB         | 99.57        |             |
| KNN Models       |              | SVM Models  |              |             |
| Euclidean        | 95.58        | Cubic       | 97.90        |             |
| City Block       | 96.00        | GRB         | 88.16        |             |
| Mahalanobis      | 96.00        | SVM Models  |              |             |
| Cosine           | 92.43        | SVM Models  |              |             |

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.
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 cases. Hence, 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.

ACKNOWLEDGMENT
ITIDA’s ITAC collaborative funded project Fund category: Advanced Research Projects (ARP) ARP2020.R28.18.

REFERENCES

[1] U.S. Energy Information Administration (EIA), “The International Energy Outlook 2013,” 2013.
[2] M. M. Badr, M. S. Hamad, A. S. Abdel-Khalik, and R. A. Hamey, “Fault Detection and Diagnosis for Photovoltaic Array Under Grid Connected Using Support Vector Machine,” 2019 IEEE Conference on Power Electronics and Renewable Energy (CPERE), Aswan, Egypt, 2019, pp. 546-553.
[3] Y. Zhao, J. de Palma, J. Mosesian, R. Lyons and B. Lehman, “Line-Line Fault Analysis and Protection Challenges in Solar Photovoltaic Arrays,” in IEEE Transactions on Industrial Electronics, vol. 60, no. 9, pp. 3784-3795, Sept. 2013.
[4] A. Triki-Lahiani, A. Bennani-Ben Abdelghani, and I. Slama-Belkhdjia, “Fault detection and monitoring systems for photovoltaic installations: A review,” Renewable and Sustainable Energy Reviews, vol. 82, pp. 2890–2910, 2018.
[5] D. S. Pillai and N. Rajasekar, “A comprehensive review on protection challenges and fault diagnosis in PV systems,” Renewable and Sustainable Energy Reviews, vol. 91, pp. 18–40, 2018.
[6] S. K. Firth, K. J. Lomas, and S. J. Rees, “A simple model of PV system performance and its use in fault detection,” Solar Energy, vol. 84, no. 4, pp. 624–635, 2010.
[7] D. E. Collier and T. S. Key, “Electrical fault protection for a large photovoltaic power plant inverter,” Conference Record of the Twentieth IEEE Photovoltaic Specialists Conference, Las Vegas, NV, USA, 1988, pp. 1035-1042 vol.2.
[8] U.S. National Electrical Code, “Article 690—Solar Photovoltaic Systems,” pp. 623–638, 2014.
[9] M. K. Alam, F. H. Khan, J. Johnson, and J. Flicker, “PV faults: Overview, modeling, prevention and detection techniques,” 2013 IEEE 14th Workshop on Control and Modeling for Power Electronics (COMPEL), Salt Lake City, UT, 2013, pp. 1–7.
[10] A. Choudre and S. Silvestre, “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 (d/dt)V Characteristic,” 2006 IEEE 4th World Conference on Photovoltaic Energy Conversion, Wakolao, HI, 2006, pp. 2442-2445.
[12] R. Haricharan, M. Chakkapaneni, G. Saravana Ilango, and C. Nagamani, “A Method to Detect Photovoltaic Array Faults and Partial Shading in PV Systems,” in IEEE Journal of Photovoltaics, vol. 6, no. 5, pp. 1278-1285, 2016.
VOLUME 1, 2021

19

V. S. Bharath Kurukuru, F. Blaabjerg, M. A. Khan, and A. Haque, “A Novel 

M. K. Alam, F. Khan, J. Johnson, and J. Flicker, “A Comprehensive 

A. Y . Appiah, X. Zhang, B. Beklisi, K. Ayawli, and F. Kyeremeh, “Review 

F. Aziz, A. Ul Haq, S. Ahmad, Y . Mahmoud, M. Jalal, and U. Ali, “A Novel 

A. Drews, A. C. de Keizer, H. G. Beyer, E. Lorenz, J. Betcke, W. G. J. H. 

Z. Yi and A. H. Etemadi, “Line-to-Line Fault Detection for Photovoltaic 

S. Ahmad, Y . Mahmoud, M. Jalal, and U. Ali, “A Novel 

Y. Zhao, Z. Chao, M. Mosessian, and J. Lehman, “Decision tree-based fault detection and classification in solar photovoltaic arrays,” 

M. N. Akram and S. Lottifard, “Modeling and Health Monitoring of DC Side of Photovoltaic Array,” in IEEE Transactions on Sustainable Energy, vol. 6, no. 4, pp. 1245-1253, Oct. 2015.

Y. Zhao, R. Ball, J. Mosessian, J. de Palma, and B. Lehman, “Graph-Based Semi-supervised Learning for Fault Detection and Classification in Solar Photovoltaic Arrays,” in IEEE Transactions on Power Electronics, vol. 30, no. 5, pp. 2848-2858, May 2015.

L. L. Jiang and D. L. Maskell, “Automatic fault detection and diagnosis for photovoltaic systems using combined artificial neural network and analytical based methods,” 2015 International Joint Conference on Neural Networks (IJCNN), Killarney, 2015, pp. 1-8.

C. Chen, P. J. Lin, J. Zhang, C. 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,” JEP Conference Series: Earth and Environmental Science, vol. 188, no. 1, pp. 1-8, 2018.

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,” in IEEE Transactions on Industrial Electronics, vol. 64, no. 11, pp. 8546-8556, Nov. 2017.

F. Aziz, A. Ul Haq, S. Ahmad, Y. Mahmoud, M. Jalal, and U. Ali, “A Novel Convolutional Neural Network-Based Approach for Fault Classification in Photovoltaic Arrays,” in IEEE Access, vol. 8, pp. 41889-41904, 2020.

W. Gao and R. J. Wai, “A Novel Fault Identification Method for Pho
tovoltaic Array via Convolutional Neural Network and Residual Gated Recurrent Unit,” IEEE Access, vol. 8, pp. 159493-159510, 2020.

V. S. Bharath Kurukuru, F. Blaabjerg, M. A. Khan, and A. Haque, “A Novel Fault Classification Approach for Photovoltaic Systems,” Energies, vol. 13, no. 2, 2020.

A. Y. Appiah, X. Zhang, B. Beklisi, K. Ayawli, and F. Kyeremeh, “Review and Performance Evaluation of Photovoltaic Array Fault Detection and Diagnosis Techniques,” International Journal of Photoenergy, 2019.

J. Flicker and J. Johnson, “Electrical simulations of series and parallel PV arc-faults,” 2013 IEEE 99th Photovoltaic Specialists Conference (PVSC), Tampa, FL, 2013, pp. 3165-3172.

M. K. Alam, F. Khan, J. Johnson, and J. Flicker, “A Comprehensive Review of Catastrophic Faults in PV Arrays: Types, Detection, and Mitigation Techniques,” in IEEE Journal of Photovoltaics, vol. 5, no. 3, pp. 982-997, May 2015.

A. Gupta, Y. K. Chauhan, and R. K. Pachauri, “A comparative investigation of maximum power point tracking methods for solar PV system,” Solar Energy, vol. 136, pp. 236-253, 2016.

P. Bulanyi and R. Zhang, “Shading Analysis & Improvement for Distributed Residential Grid-Connected Photovoltaic Systems,” The 52nd Annual Conference of the Australian Solar Council, 2014.

M. Reza, H. Hizam, C. Gomes, M. Amran, M. Ismael, and S. Hajighorbani, “Power loss due to soiling on solar panel: A review,” Renewable and Sustainable Energy Reviews, vol. 59, pp. 1307-1316, 2016.

K. H. Chao, S. H. Ho, and M. H. Wang, “Modeling and fault diagnosis of photovoltaic system,” Electric Power Systems Research, vol. 78, no. 1, pp. 97–105, 2008.

K. E. Piliaro, M. Shafiee, Y. Cao, L. Lao, and S. Yang, “A Review of Kernel Methods for Feature Extraction in Nonlinear Process Monitoring,” Processes, vol. 8, no. 24, 2020.

S. J. Al’Aref, K. Anchouche, G. Singh, P. J. Slomka, K. K. Kolli, A. Kumar, M. Pandey, G. Maleika, A. R. V. Rosendael, A. N. Bercy, D. S. Berman, J. Leipsic, K. Nieman, D. Andreini, G. Pontone, U. J. Schoepf, L. J. Shaw, H. J. Chang, J. Narula, J. J. Bax, Y. Guan, and J. K. Min, “Clinical applications of machine learning in cardiovascular disease and its relevance to cardiac imaging,” European heart journal, vol. 40, no. 24, pp. 1975-1986, 2019.

International Electrotechnical Commission, “IEC 61724-1: Photovoltaic system performance - Part 1: Monitoring,” 2017.

S. U. Jan and I. Koo, “A Novel Feature Selection Scheme and a Diversified-Input SVM-Based Classifier for Sensor Fault Classification,” Journal of Sensors, 2018.

S. Yadav and S. Shukla, “Analysis of k-Fold Cross-Validation over Hold-Out Validation on Colossal Datasets for Quality Classification,” IEEE 6th International Conference on Advanced Computing (IACC), Bhumavan, 2016, pp. 78-83.

A. Maheshwari, N. Davendralingam, and D. DeLaurentis, “A Comparative Study of Machine Learning Techniques for Aviation Applications,” 2018 Aviation Technology, Integration, and Operations Conference, 2018.

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, 2017.

M. Suknovic, B. Delibasic, M. Jovanovic, M. Vukicevic, D. Becjejski-Vujaklija, and Z. Obradovic, “Reusable components in decision tree induction algorithms,” Computational Statistics, vol. 27, no. 1, pp. 127-148, 2012.

Y. Wang and S. Xia, “Unifying attribute splitting criteria of decision trees by Tsallis entropy,” 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), New Orleans, LA, 2017, pp. 2507-2511.

M. El Habib Daho, N. Settouti, M. E. A. Lazouni, and M. E. A. Chikhi, “Weighted vote for trees aggregation in Random Forest,” 2014 International Conference on Multimedia Computing and Systems (ICMCS), Marrakech, 2014, pp. 438-443.

V. S. Prasatha, H. A. A. Alfeilate, A. B. Hassanate, O. Lasassmehe, A. S. Tarawnehf, M. B. Alhasanatag, and H. E. S. E. Salame, “Effects of Distance Measure Choice on KNN Classifier Performance: A Review,” Big Data, vol. 7, no. 4, pp. 221-248, 2019.

Dulai 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 and Applications, vol. 78, no. 14, pp. 20383–20407, 2019.

Q. Cao, L. La, H. Liu, and S. Han, “Mixed weighted KNN for imbalanced datasets,” International Journal of Performability Engineering, vol. 14, no. 7, pp. 1391–1400, 2018.

E. Zhang, “Cost-sensitive KNN classification,” Neurocomputing, vol. 391, pp. 234–242, 2020.

Z. Geler, V. Kurbalija, M. Radovanovic, and M. Ivanovic, “Comparison of different weighting schemes for the knn classifier on time-series data,” Knowledge and Information Systems, vol. 48, no. 2, pp. 331–378, 2016.

C. Castaldello, A. Gubert, F. Galvain, A. Casonato, R. Padini, M. Barolo, and F. Bezzo, “A Model-Based Support for Diagnosing Willebrand Disease,” vol. 40, pp. 2779-2784, Computer Aided Chemical Engineering, 2017.

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.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/
MOHAMED M. BADR received the M.Sc. degree in Electrical Engineering from Alexandria University, Alexandria, Egypt, in 2020. He is currently working toward the Ph.D. degree in Electrical Engineering at Alexandria University, Alexandria, Egypt. His research interests include mechatronics, renewable energy, and power electronics.

MOSTAFA S. HAMAD (SM’19) obtained 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 Strathclyde University, Glasgow, U.K., in 2009. From 2010 to 2014, he was an Assistant Professor in the Department of Electrical and Control Engineering, College of Engineering and Technology, Arab Academy for Science, Technology and Maritime Transport (AASTMT), Alexandria, Egypt, 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 (SM’12) 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, 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, Alexandria, Egypt. He serves as an Associate Editor of IEEE Transactions on Industrial Electronics and IET Electric Power Applications Journal. Also, he serves as the Executive Editor of Alexandria Engineering Journal. His current research interests include electrical machine design and modelling, electric drives, energy conversion, and renewable energy.

RAGI A. HAMDY (SM’19) received B.Sc. and M.Sc. on 91 and 94 from Alexandria University, Egypt, and Ph.D. on 99 from Heriot-Watt University, UK. He is currently a Professor with the Electrical Engineering Department, Faculty of Engineering, Alexandria University, Alexandria, Egypt. His current research interests include electric machines, Electric Drives, and power electronics.

SHEHAB AHMED (SM’12) 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 Professor of Electrical Engineering with the CEMSE Division at 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 Ph.D., M.Sc. and B.Sc. degrees in Electrical Engineering from Alexandria University, Egypt, in 2012, 2004 and 2000 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 network, and optimization techniques.