IEEE Open Journal of the
Industrial Electronics Society

Received 11 May 2022; revised 21 June 2022 and 18 August 2022; accepted 15 September 2022. Date of publication 20 September 2022; date of current version 29 September 2022. The review of this paper was arranged by Associate Editor Daswin de Silva.

Digital Object Identifier 10.1109/OJIES.2022.3208140

Detecting Partial Shadowing and Mismatching Phenomena in Photovoltaic Arrays by Machine Learning Techniques

MICHEL PILIOUGINE, RUDY ALEXIS GUEJIA-BURBANO, AND GIOVANNI SPAGNUOLO (Fellow, IEEE)

Universitá degli Studi di Salerno, 84084 Fisciano, Italy

CORRESPONDING AUTHOR: MICHEL PILIOUGINE (e-mail: mpiliouginerocha@unisa.it)

This work was supported in part by the Ministero dell'Istruzione, dell'Università e della Ricerca (Italy) under Grant PRIN2020–HOTSPHOT 2020LB9TBC and Grant PRIN2017–HEROGRIDS 2017WA5ZT3_003, in part by the Università degli Studi di Salerno [FARB funds], and in part by the Ministerio de Ciencia, Innovación y Universidades (Spain) under Grant RTI2018–095097–B-I0.

ABSTRACT A photovoltaic array including several modules in series may show mismatching due to discrepancies among the module conditions, mainly due to partial shadowing. Therefore, the shape of the current–voltage curve deeply changes with respect to the one corresponding to uniform operation. This article shows that a small set of points around the maximum power allows us to detect the occurrence of the mismatching. This approach exploits such a limited information to detect if the module is subjected to mismatching, so that the adoption of a GMPPT algorithm can be avoided. The curvature change is identified by using different machine learning techniques: decision trees, multilayer perceptrons, radial basis functions, and support vector machines. To reduce the classification error, before the fitting of the models, we implement a novel process of selection of the training samples based on a self-organizing map. This procedure makes easier the optimization of the number of hidden neurons. The support vector classifier and the multilayer perceptron with one hidden layer outperform the other approaches, being the former better than the last for extreme mismatching. However, the prediction time of this multilayer perceptron is significantly smaller than the required by the support vector machine.

INDEX TERMS Decision tree, multilayer perceptron (MLP), photovoltaic mismatch diagnosis, radial basis function (RBF), self-organizing map (SOM), support vector machine (SVM).

NOMENCLATURE

ANN	Artificial neural network.
$\mathcal{C}$	Kernel constraint parameter of an SVM.
C_i	Coefficient of the interpolating polynomial.
CNN	Convolutional neural network.
γ	Scaling value of the kernel function on an SVM.
η	Diode ideality/quality factor (−).
$\vec{\phi}$	Kernel transformation on an SVM.
$G_{i,j}$	Each group created by the Kohonen SOM.
GMPPT	Global maximum power point tracking.
HMLP	Hybrid multilayer perceptron.
HMLP1	HMLP using *radbas* in the first hidden layer and *tansig* in the second hidden layer.
HMLP2	HMLP using *tansig* in the first hidden layer and *radbas* in the second hidden layer.
I	Current (A).
I_{ph}	Photo-generated current (A).
I_s	Diode dark-saturation current (A).
I_{sb}	Dark-saturation current of the bypass diode (A).
I–V	Current–voltage.
k	Boltzmann constant 1.381e–23 J/K.
KNN	k-nearest neighbor.
MLP	Multilayer perceptron.
MLP1	Multilayer perceptron with one hidden layer.
MLP2	Multilayer perceptron with two hidden layers.
MPP	Maximum power point of the I–V curve.
MPPT	Classical MPP tracking.
N_s	Number of cells in series of a PV module.
P_{max}	Maximum power of the PV device (W).
PV	Photovoltaic.

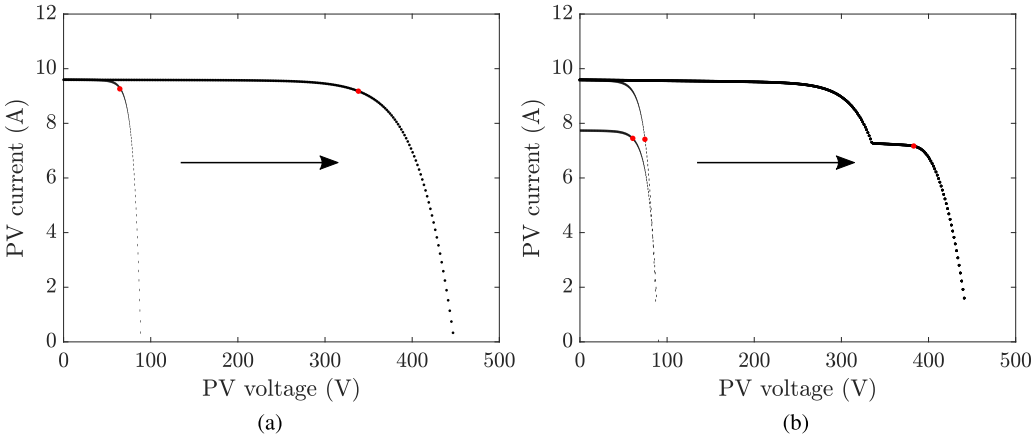

FIGURE 1. ***I–V*** **curves of a PV array composed of several modules in different scenarios. (a) under uniform condition. (b) under mismatched condition.**

q	Elementary charge 1.602e–19 C.
RBF	Radial basis function.
radbas	Gaussian transfer function.
R_h	Shunt/parallel resistance (Ω).
R_s	Series resistance (Ω).
SOM	Self-organizing map.
SVM	Support vector machine.
V	Voltage (V).
V_T	Thermal voltage of a PV device (V).
T	PV device temperature in Kelvin (K).
T_c	PV device temperature in Celsius degree (°C).
tansig	Hyperbolic-tangent transfer function.

I. INTRODUCTION

Photovoltaic (PV) modules and arrays are composed of solar cells connected in series. Failures, degradation, dirt, or shadows can make the irradiance not to be uniform among all the elements, leading to energy drops. Under some conditions, this fact results in the dissipation of power by the shaded cells, even leading to damage and risk of fire. Commercial PV modules are equipped with bypass diodes in parallel to every string of cells in series, so that the current difference between the shaded and the unshaded strings passes through the bypass diode of the shaded one [1]. Therefore, the shaded parts of the string are protected from damage, but the power drop is unavoidable. The detection of these anomalous conditions has to be prompt in order to take the best control action in order to maximize the power production for the current operating conditions.

The activation of one or more bypass diodes affects the current–voltage (*I–V*) curve that should exhibit several maximum power points (MPPs) [2]. During partial shadowing, the MPP tracking system drives the PV array toward one of the MPPs. If the string is not operating at the MPP occurring at the highest voltage, i.e., the rightmost one in the current–voltage plane, it is quite simple to recognize the occurrence of a shading condition. Indeed, in this case, the operating voltage is abnormally low with respect to the one corresponding to the unique MPP in uniform operating conditions; thus, it is very far from the open-circuit voltage. Unfortunately, this is not true if the MPP tracking system leads the PV array to work at the rightmost MPP, having an operating voltage a bit lower than the expected open-circuit voltage. It is true that the current at this case is abnormally low, but it is hard to distinguish this case from the one due to a uniform drop in the irradiance level over the whole array. Therefore, a mechanism allowing to distinguish between normal and mismatched or fault conditions would be welcome in this case.

A uniform decrease of the irradiance affecting all the modules of the array leads the inverter to drive all the cells in series to work in a similar operating point; it is almost the MPP of each cell, where the shape of the *I–V* curve is regularly rounded. Thus, the resulting total *I–V* curve is also quite rounded [see Fig. 1(a)]. Instead, if a partial shadow decreases the current output of one substring connected in series and the tracking system leads the array to the rightmost MPP, only the shaded substring works close to its MPP, whereas the operating point of all the nonshaded substrings belongs to a low-current region of a high irradiance curve; hence, it is a point very close to the open-circuit voltage and where the *I–V* curve slope is very high. As there are many substrings of the array operating at that point, this significantly affects the shape of the total *I–V* curve, thus making it not mildly rounded as in the uniform case, but rather squared [see Fig. 1(b)].

The objective of this article is to show the performance of some algorithms allowing to detect the absence or the occurrence of a mismatching event by using only a little portion of the *I–V* curve, specifically those points around the MPP. The goal is the detection of the change of the curve shape by using only those few *I–V* samples around the MPP at the highest voltage. Each tool should be able to classify an input *I–V* curve into two categories, i.e., unshaded (*NORMAL*) or mismatched (*FAULTY*). The supervised learning is performed through a training process involving a huge set of previously classified curves. Once the proposed method detects a possible mismatching condition for the module, the global MPP tracking (GMPPT) algorithm should be activated. In another case, the inverter will remain using the classical MPPT. This

GMPPT algorithm is beyond the scope of this work. Indeed, in this article, no new global MPPT method is proposed, but *an analysis that precedes the application of a GMPPT algorithm is introduced*.

Next, Section II is aimed at giving an overview of the approaches presented in the recent literature, with an emphasis to those employing machine learning tools, dedicated to the detection and classification of partial shadowing and, more in general, faults affecting PV systems through the analysis of the *I–V* curves. Afterward, a number of typical paradigms in the field of machine learning are reviewed and applied to the same problem.

In order to test and compare the performance of the machine learning techniques under study, several experimental test sets have been built. On the one hand, TEST#0 has been performed by using only one polycrystalline silicon module. On the other hand, TEST#1, TEST#2, and TEST#3 are based on measurements from two modules in series, by applying a different shadowing level in each case.

The first contribution of this work is the application of a novel procedure for selecting the best training samples, which is based on a Kohonen self-organizing map, as described in Section III-D. This procedure replaces a random selection and it is aimed at reducing the classification error. This neural network performs an unsupervised classification of the available samples in different groups, so that all the samples classified in the same group are very similar. Hence, a nonfrequent sample could not have similar cases, probably requiring its own group. Therefore, if only one sample is selected from each group, the special cases are ensured to be selected, whereas from a group with many similar cases, only one should represent all of them. As a consequence, the importance of the nonfrequent cases in the training process is increased significantly.

A further contribution of the work presented in this article consists in the way to extract the most relevant information of each *I–V* curve, described in Section III-C. Instead of feeding the models by using the raw dataset of measured *I–V* pairs, a third-degree polynomial has been used to fit the measured points, so that the obtained coefficients of the polynomial have been used as input of the approaches presented in the article. Therefore, each curve is represented by the four coefficients of the polynomial, so that all the methods compared in this work receive an input data structure of a fixed length.

As a first model to analyze, the behavior of the decision trees to deal with the classification of the *I–V* curves is studied (see Section III-E). Then, a support vector classifier is proposed to solve the same problem (Section III-F). Finally, several types of artificial neural networks (ANNs) are tested: a multilayer perceptron (MLP) in Section III-G, equipped with one or two hidden layers, and a radial basis function (RBF) in Section III-H, and some hybrid approaches in Section III-I. In Section IV, the results achieved through the proposed methods are discussed. The main conclusions of the work can be found in Section V.

Once the best machine learning approach is known, it can be merged with the MPPT algorithm and tested on-field in a number of cases, in order to show that the proposed detection of the partial shadowing occurrence allows us to reduce the power losses related with a repeated scan of the *I–V* curve to track the absolute MPP.

II. STATE OF THE ART

In the recent literature, a number of techniques aimed at detecting faults and mismatching conditions in PV arrays have been proposed. Many of them rely on the feature extraction from PV array photographs [3], [4], [5], [6], [7], [8]. In a few cases, e.g., in [9], image processing allows us to perform a classification of different possible faults through convolutional neural networks (CNNs). Such image-based approaches require expensive sensors and means to take images in the infrared, visible, or ultraviolet range.

Other approaches are based on the continuous measure of the electrical output of the PV array, which allows us to perform some inferences [10], [11], [12]. In some cases, the main electrical parameters are used to perform the identification, whereas in other cases, the input of the classification algorithm is a set of suitably identified model parameters. For instance, the single diode model is adopted and its series and shunt resistance values are identified. Unfortunately, such approaches require the measurement of the whole *I–V* curve, from the open circuit to the short-circuit condition, with a consequent power loss due to the temporary suspension of the MPP tracking operation.

In [13], partial shading and mismatching conditions are duly detected and not confused with a dynamic variation of the actual irradiance level due to the meteorological conditions. The proposed approach is based on the evaluation of a set of assets defined by different thresholds, so that the measured values are compared with the expected ones. Contrary to other approaches, the algorithm runs in parallel with the MPP tracking algorithm, so that the PV power production is preserved.

A high accuracy in fault detection and classification based on electrical data acquired at the PV array terminals is achieved by using some artificial intelligence approaches: In [14], a comprehensive review is proposed. The ANNs emerge as effective tools to diagnose and classify faults and degradation of PV sources.

Different neural network architectures have been proposed in the recent literature. The MLP is widely adopted: In [15], it allows us to make a categorization of the possible fault in PV arrays from different technologies. Once the *I–V* curves have been measured, the main electrical parameters are extracted and provided, together with the current irradiance and the cell temperature, to the neural network input. The measurement of the operating conditions is thus required again, as well as the need of the whole *I–V* curve to be provided at the neural network input. A similar approach is presented in [16], and [17], where MLPs are adopted to achieve a similar goal. The same objective is pursued in [18] and in [19]: In the former study, the estimation of the indicators required as MLP

input is performed by a wavelet transform. In the latter paper, instead, a CNN is adopted. An example of applications to the same problem of a probabilistic neural network is given in [20]. In [21], the k-nearest neighbor (KNN) is applied to PV modules diagnosis.

A comparative study of different machine learning tools is carried out in [22]. Faults are identified and classified into several categories by using decision trees, KNN, and SVMs. The proposed approach requires irradiance and temperature measurements and also a preliminary effort of characterization of the system in normal conditions. SVMs have been used for fault classification in PV arrays also in [23] and [24]. In the latter one, the principal component analysis is used to estimate the required indicators. Two SVMs have been combined in [25] to obtain a more accurate classifier.

Some recent approaches are based on a hybridization of different techniques. In [26], a recurrent gated unit is combined with a CNN to identify different types of faults, without extracting any indicator, by working directly with the information captured from the *I–V* curves. The system does not need any meteorological data.

In [27], it is shown that RBFs are a further powerful method to develop a PV fault classifier. A complementary stage based on a wavelet tool allows us to extract the indicators feeding the neural network. An extensive tree to classify the different types of faults is also proposed.

In conclusion, some recent approaches require a limited portion of the *I–V* curve, restricted around the MPP, which allows us to preserve the PV power production. Unfortunately, many of them are based on a preliminary characterization of the PV array operating in normal conditions, which represents the reference for the values of the indicators used by the technique. This initial step is mandatory to fix the boundaries between the normal and the faulty condition for the specific PV plant under study. It is worth to say that the reference conditions depend on the season of the year and on the level of degradation, so that reference values should be updated periodically. Furthermore, all the approaches require at least the values of the incident irradiance and the module temperature, not always available, especially for small installations.

The problem GMPPT is widely treated in the literature. The trapping of classical MPPT algorithms, which are local optimizers, in local MPPs is avoided by using global optimizers, which are often based on heuristic algorithms. In almost all the papers in the literature, the problem is afforded by starting from a power versus voltage curve showing multiple MPPs and the smartness in detecting the global maximum through the proposed algorithm is duly documented. Unfortunately, in all the papers that are focused on presenting new GMPPT algorithms, the analysis of the event that triggers the GMPPT algorithm to start the search is always neglected. It is sometimes said that the GMPPT starts when an abnormal PV power variation is detected. Unfortunately, this is not always due to the occurrence of a mismatching, e.g., partial shadowing, event, but it might be due to a uniform variation of the irradiance level the PV array is receiving. In such a condition, the GMPPT procedure is needlessly run, with a consequent power loss and a useless computational burden. The algorithm proposed in this article is just aimed at detecting the event that really requires a GMPPT approach or evidencing a fault condition. A number of different numerical procedures are compared to assess what is the best method to solve the problem thereof.

Consequently, the approach presented in this article is aimed at filling the research gap with respect to the current literature thanks to the following features.

1) The proposed approach does not require any initial evaluation of the PV array operating under standard conditions to settle suitable threshold values to use for fault detection and classification. Thus, it does not need any periodic update of the same thresholds.
2) The proposed approach does not need any irradiance and temperature measurement, which is a great advantage for small PV installations.
3) The proposed approach adopts an identification process running in the current–voltage plane, without working in any further space dedicated to fault indicators. This feature allows the proposed method to be independent of any PV array model and, thus, of its accuracy. This strategy has been also recently addressed in [26].
4) The training set is smartly chosen by exploiting an unsupervised neural network known as the self-organizing map.

III. METHODOLOGY

A. GENERATION OF SIMULATED I–V CURVES

In order to design a universal tool that is independent from the specific length of the PV string, from the specific electrical parameters of the PV modules and of the number of bypass diodes, the training set has been generated by means of a simulation model, which describes the PV string at a suitable level of granularity. This makes the study more general than the one that should be done by using measurements concerning a specific PV string.

The *I–V* curve of a PV string working under normal conditions has been simulated by (1), where V and I are voltage and current at the whole PV string terminals, I_{ph} is the photo-induced current, I_s is the saturation current, η is the ideality factor, N_s is the total number of cells of the string, R_s and R_h are the series and shunt resistances of the string, respectively, $V_T = kT/q$ is the thermal voltage (with $k = 1.381\text{e-}23$ J/K, $q = 1.602\text{e-}19$ C, and T in Kelvin is the cell temperature)

$$I = I_{\text{ph}} - I_s\left[\exp\left(\frac{V + IR_s}{N_s\eta V_T}\right) - 1\right] - \frac{V + IR_s}{R_h}. \quad (1)$$

As for the partially shadowed PV string, each string of cells protected by a bypass diode is simulated through a nonlinear equation; the equations of all these elements in the PV array are combined according to the Kirchhoff laws to consider the series connection of all the strings. The synthetic curves generated in this way use two levels of irradiance, since the

interest is in the MPP at the highest voltage only. Thus, it is irrelevant to the problem treated in this article if the mismatched *I–V* curve includes two MPPs or more. The two levels of irradiance considered are the normal and shaded ones, which refer to $I_{\text{ph},1}$ and $I_{\text{ph},2}$ as photo-induced currents, respectively. The most interesting case occurs when few groups of cells in the string are shadowed. In conclusion, each group with its bypass diode is modeled through (2) using $I_{\text{ph}} = I_{\text{ph},1}$ if it is working at a high irradiance level or $I_{\text{ph}} = I_{\text{ph},2}$ if it is working at a low irradiance level

$$I_i = I_{\text{ph}} - I_s\left[\exp\left(\frac{V_i + I_i R_{s,\text{str}}}{N_{s,\text{str}}\eta V_t}\right) - 1\right] - \frac{V_i + I_i R_{s,\text{str}}}{R_{h,\text{str}}} + I_{sb}\left[\exp\left(\frac{V_i}{V_t}\right) - 1\right] \tag{2}$$

where I_{sb} is dark saturation current of the bypass diode, and the subscript *str* (in $R_{s,\text{str}}$, $R_{h,\text{str}}$, and $N_{s,\text{str}}$) refers to each parameters scaled up to the string from the value of the same parameter referring to a single cell. As all the groups have been connected in series, all the unknown currents of all the groups must be equal and the sum of all the unknown voltages must be equal to the total voltage of the PV string.

The values of the five required parameters of the single-diode model, of the module temperature, and of the dark saturation current of the bypass diode have been randomly chosen within a feasible range

$I_{\text{ph}} \in [1, 12]$ A
$I_s \in [1e-12, 1e-5]$ A
$\eta \in [1, 2]$
$R_{s,\text{cell}} \in [0.001, 0.01]\ \Omega$
$R_{h,\text{cell}} \in [1, 50]\ \Omega$
$I_{sb} \in [1e-9, 1e-6]$ A
$T_c \in [0, 70]$ °C.

The model has been simulated by taking into account that the number of cells protected by a bypass diode can vary between 1 and 30 and the total number of cells also varies between 6 and 900. These wide ranges have been set in order to ensure the applicability of the achieved methods to a full range of possible photovoltaic arrays.

By following the procedure described above, two subsets of 5000 curves per class, thus a total of 10 000 curves, have been simulated: 1) a first subset following (1) and referring to PV strings operating under uniform conditions; 2) a second subset following (2) representing mismatched PV strings. When partial shadowing occurs, although the *I–V* curve has more than one MPP, the attention is focused on the rightmost one because an MPP tracking any other MPP should drive the operating point at a voltage value that is lower than the open-circuit voltage.

B. GENERATION OF THE DIFFERENT TEST SETS

Except for the decision tree, all the proposed approaches require a parameter tuning stage, consisting in repeating the training process by trimming the approach's specific parameters. For instance, MLP and RBF with a different number of neurons in the hidden layer are trained, so that their optimal architecture is determined. Thus, the training set of simulated curves is used as input each time the model is fitted; another independent set of curves is used as test set in order to determine the best architecture, e.g., the number of hidden neurons, namely the one resulting in the minimum error over that test set. This first test is named TEST#0.

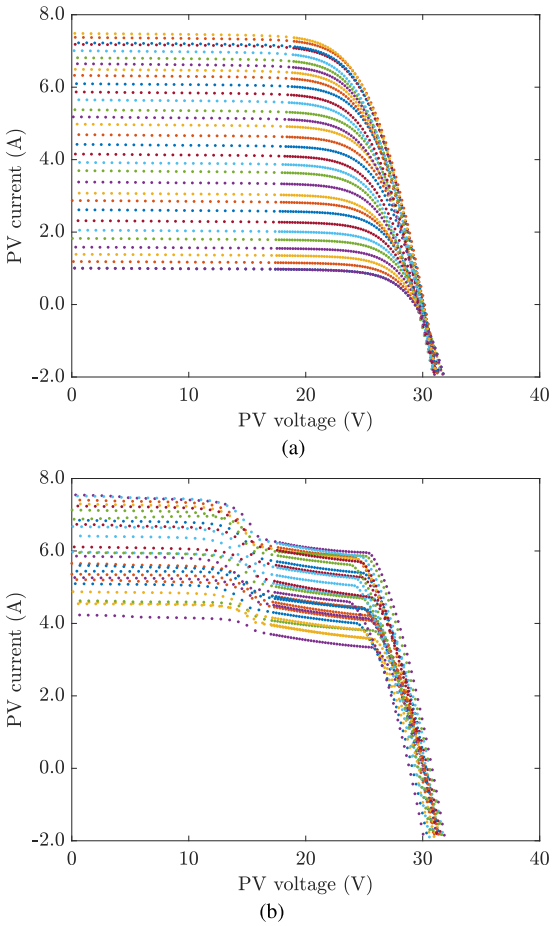

FIGURE 2. Experimental *I–V* curves measured under different conditions. (a) Under uniform conditions. (b) Under mismatched conditions.

The test set used in this article to tune each approach includes 2000 experimental *I–V* curves not related to the training set: 1000 of them corresponding to normal operating conditions and other 1000 to mismatched conditions. In order to obtain this test set, an experimental campaign of measurements has been performed, referring to a real commercial PV module composed corresponding to model PCB-195-A15 from the manufacturer YOCASOL [28]. Some examples of experimental curves corresponding to normal conditions are shown in Fig. 2(a), whereas Fig. 2(b) includes other cases of curves measured under mismatching conditions. The measurement system, in both its hardware and software components, used to acquire these *I–V* curves is described in [29], [30], and [31]. At the beginning of its lifetime, the module shows normal *I–V* curves; a few months after its first use, its

FIGURE 3. PV modules in series using a screen onto one of them to simulate partial shadowing conditions.

behavior progressively worsens resulting in stepped curves, up to a moment when it went out of order due to an electrical internal fault.

Once the best structure of each method has been identified by means of the TEST#0, the performance and generalization capabilities of the methods have been analyzed by using two additional test sets, which refer to an array of two PV modules FLEX SP50-L from SOLBIAN [32] connected in series. The following additional test sets are defined.

1) TEST#1: 500 *I–V* curves corresponding to normal operating conditions and other 500 curves obtained by shadowing one of the two modules by a screen reducing the incident irradiance until 30% of the actual irradiance (see Fig. 3).
2) TEST#2: 500 *I–V* curves under normal operating conditions and other 500 curves obtained by shadowing one of the two modules with a layer screen reducing the incident irradiance by 55%.
3) TEST#3: 500 *I–V* curves under normal operating conditions and other 500 curves obtained by shadowing the second module to reduce its irradiance by 88%.

C. EXTRACTING THE RELEVANT INFORMATION OF EACH I–V CURVE

In this work, not all the samples of the *I–V* curve are used in order to decide if the PV string or module is working under

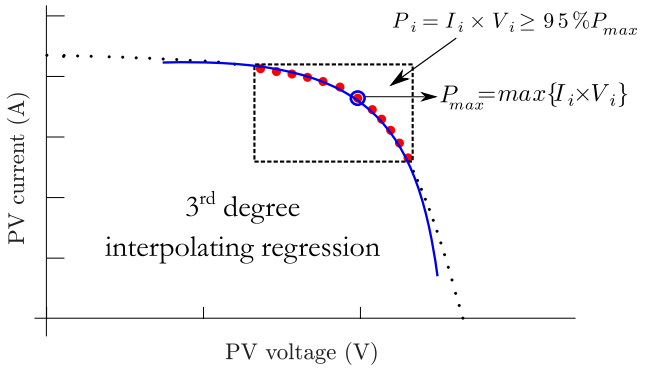

FIGURE 4. Selection of points around the MPP and estimation of the polynomial.

normal or fault operating conditions. Only the set of points close to the MPP is considered to detect a fault condition because they are almost the ones available without losing the tracking of the MPP during the usual operation of the PV array.

After simulating a set of full *I–V* curves, a procedure to select the points around the MPP has been executed to extract the useful part of the curve. Assuming that $P_{\max} = \max\{I_i \times V_i\}$ is the power value of the discrete point with the highest power, all the points having a power value greater than 95% of $P_{\max}$, have been chosen. In Fig. 4, these selected points are the highlighted ones enclosed inside the dotted box.

The number of samples selected in this way might be different from curve to curve. Unfortunately, all the machine learning tools used in this article require input samples of a fixed length. Therefore, each set of *I–V* points must be converted into a vector with fixed length. According to the current literature [33], [34], [35], a polynomial allows us to fit the set of selected points and calculating an accurate value of $P_{\max}$. Inspired by that idea, in this article, a third-degree interpolating polynomial (3) interpolating the samples around the MPP, as it is exemplified in Fig. 4 by a continuous blue line, has been used. A regression procedure has allowed to compute the coefficients $\{C_0, C_1, C_2, C_3\}$

$$C_3 V^3 + C_2 V^2 + C_1 V + C_0 \Longrightarrow (C_0, C_1, C_2, C_3)\,. \qquad (3)$$

The degree of the interpolating polynomial has been chosen according to the results that are described in the sequel, at the end of Section IV-C.

D. SELECTION OF THE TRAINING SAMPLES USING A SELF-ORGANIZING MAP

The initial set of simulated *I–V* curves has a total number of 10 000 samples: 5000 curves simulating normal conditions and other 5000 simulating mismatched conditions. Whereas most of the shapes of the normal curves appear to be similar, there is more discrepancy among the shapes of the faulty curves. In other words, in the faulty group, a lot of differently shaped curves fall. Some of those shapes are more frequent than other shapes in the initial training set. In general, the

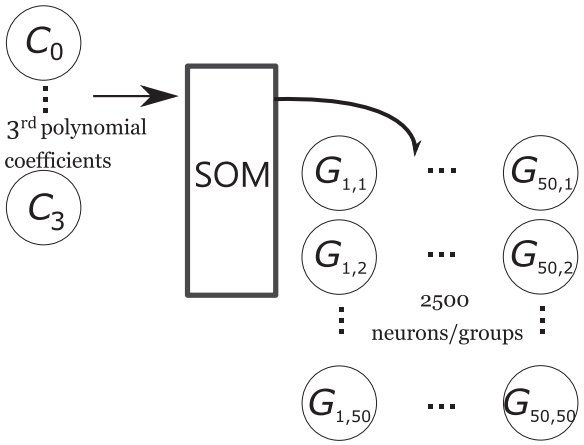

FIGURE 5. Self-organizing map to select the most representative curves.

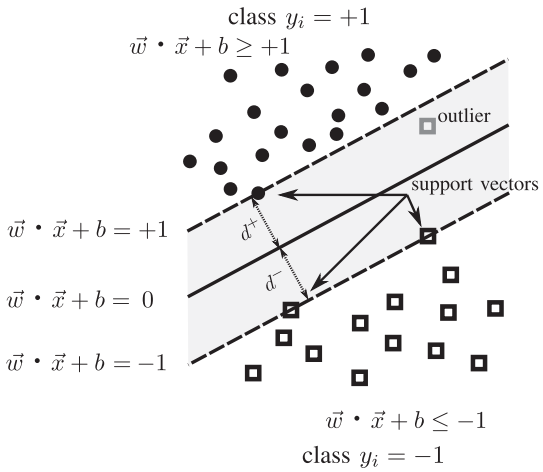

FIGURE 6. SVM maximal margin hyperplane to separate two categories.

machine learning models try to minimize a global function, in such a way the nonfrequent samples use to have very small influence on the final model. Therefore, the results when the model is applied to those cases have very high error. A way to solve this problem is to increase the weight of those nonfrequent samples in the training process.

Instead of using all the samples of the simulated set of *I–V* curves, a suitable selection of the most representative samples from this initial training set could be very useful to improve the accuracy of neural network models. We propose to perform this selection by means of a SOM, self-organizing map [36], a type of ANN discovers similarities among the input samples and it classifies the samples into a number of groups defined by the user. If there are many very similar samples, all of them are assigned to the same group. A special nonfrequent case is assigned to a new specific group. Finally, as from each group only one sample is selected, more importance to the nonfrequent cases in the selected set is given. From the initial set of 10 000 simulated *I–V* curves, the SOM classifies them into 2500 groups (see Fig. 5), by taking into account the features of each curve that are summarized in the vector (C_0, C_1, C_2, C_3). Then, the *I–V* curve closest to the mass-center of each group has been selected as the representative of that group. Therefore, 2500 *I–V* curves have been used to train the models, which is more representative than the whole initial set of 10 000 curves.

The SOM distributes the groups on a map so that the closer two groups, the more similar the elements that belong to both groups. The user only has to define the type of map (1-D, 2-D, 3-D,...) and the dimensions (for example, a 2-D map of 3×4 implies 12 groups), and the SOM network determines to which group each of the input samples belongs. For the case under study, a 2-D map of 50×50 (this implies the 2500 groups specified before) has been used.

Finally, some implementations of the SOM routine, such as *newsom/train* in Matlab [37], are run into two phases: the first group of iterations, which are characterized for having a high learning rate allowing great changes, and the second group of iterations, with a low learning rate for a small tuning of the results.

One of our main objectives in this article is to show the improvement introduced in the training process of the different techniques by means of using a Kohonen self-organizing map to perform a smart selection of the training samples. Therefore, for all the studied methodologies, we will perform a comparative analysis between the results using the proposed approach versus using 2500 samples that have been randomly selected from the initial 10 000 samples.

E. DECISION TREES

The most widely used classifiers are the decision trees, which have been successfully applied to detect faults in PV arrays and PV systems [38], [39]. Basically, a decision tree is a recursive splitting of the sample space by the application of a sequence of questions about the values of the attributes (or components) of the samples to be classified. Each question depends on the answer provided for the previous one, in such a way, they can be considered as the nodes of a tree where the leaves (nodes without children) must be one of the possible categories (in our case normal operation or fault operation). Each node divides the sample space into several groups depending on a decision about the attributes of the sample to be classified). Therefore, by descending from the root to the leaves through the internal nodes (decision nodes), it is possible to assign one of the possible categories to each input sample.

The *Statistics and Machine Learning Toolbox* of Matlab [37] includes the function *fitctree* that is able to generate a classification tree from a set of labeled samples (each sample has been previously classified into one of the possible categories). The implementation of this function is based on the CART algorithm described in [40]. The main advantage of this approach is that the obtained model (the decision tree itself) is very easy to be interpreted and applied by a human. In addition, the generation of the decision tree consumes very low computational resources in terms of time and memory, in

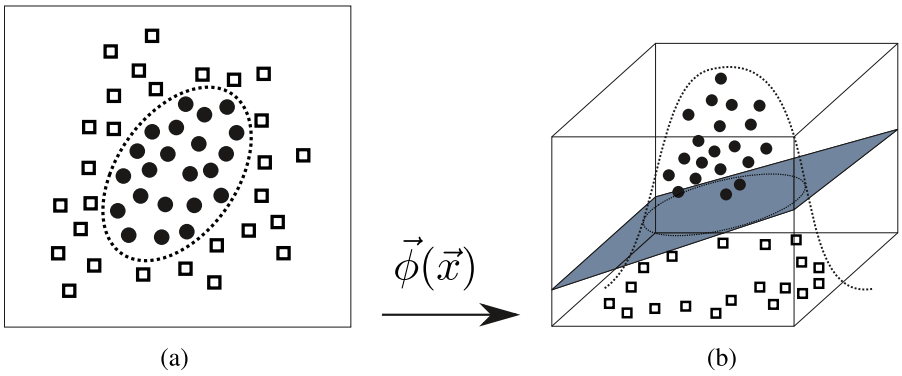

FIGURE 7. Kernel transformation to achieve a linearly separable set.

such a way, we can obtain an operational model in very few seconds.

F. SUPPORT VECTOR MACHINES

An alternative to the use of decision trees is the support vector machine (SVM), a supervised learning approach that is mathematically based on the statistical theory and that was initially proposed by [41]. The simplest SVM tries to define a hyperplane (an n-dimensional plane) to classify samples ($\vec{x_i} \in \mathbb{R}^p$) into two different categories ($y_i = +1$ or $y_i = -1$) that are linearly separable. For each hyperplane defined by $\vec{w} \bullet \vec{x} + b = 0$ (see Fig. 6), it is possible to calculate its margin as $\min\{d^+, d^-\}$, where d^+ is the distance to the hyperplane of the nearest sample (or samples) of the class $y_i = +1$ and d^- is the distance of the nearest sample of the class $y_i = -1$ (these samples are known as the support vectors). The hyperplane identified by an SVM is the one that maximizes its margin, and this fact only happens if $d^+ = d^-$.

This basic SVM cannot deal with the outliers, which are samples that actually belong to one class but that are very close to the samples of the other class (perhaps due to the presence of noise). In Fig. 6, the addition of the outlier sample implies a significant change in the estimated hyperplane. Instead, it is possible to let the previous hyperplane but allow some degree of misclassification among the samples of the training set. Each sample that is misclassified implies a penalization that is weighted by a user parameter $\mathcal{C}$ (also known a kernel constraint) that must be set before the training of the SVM. Depending on the value of $\mathcal{C}$, the outliers have more or less influence on the fitting of the hyperplane. In fact, the performance of the SVM is very dependent on this value.

SVMs may also deal with datasets for which it is impossible to find a way to separate them linearly without a high number of misclassifications, as it can be seen with a 2-D example in Fig. 7(a). By applying a function $\vec{\phi}(\cdot)$, the samples $\vec{x}$ from the input space are mapped into $\vec{\phi}(\vec{x})$ inside an alternative space with a high number of dimensions, where the transformed set of points can be linearly separated into two different categories by a hyperplane. Fig. 7(b) shows the transformations of the previous samples in a 3-D space where it is possible to find a plane that divides linearly the space and allows us to distinguish both categories. In any case, for applying the SVM, it is not necessary to use the function $\vec{\phi}(\cdot)$ itself. For a pair of samples $\vec{x_1}$ and $\vec{x_2}$, the scalar product of their transformations is needed. This is known as the kernel function, being the Gaussian kernel (provide by the following equation) the most popular for SVM and the one used in this article:

$$K(\vec{x_1}, \vec{x_2}) = \vec{\phi}(\vec{x_1}) \bullet \vec{\phi}(\vec{x_2}) = \exp\left(\gamma \cdot \|\vec{x_1} - \vec{x_2}\|^2\right) \quad (4)$$

where the parameter γ is a scaling value that should be set beforehand. Finally, it is known that SVMs work better if each component of the vector representing the input samples has been previously normalized taking into account the mean and the typical deviation of each component.

G. MULTILAYER PERCEPTRON

An ANN is a computing paradigm based on a biological model made up of several elementary units (known as neurons) organized in levels or layers, defining a relationship between their inputs and their outputs. Each individual neuron performs a very simple calculus getting a fixed number of input values, multiplying each one by a weight and then adding all the results. Finally, the transfer function of each neuron is applied to the obtained result. There are several available transfer functions, but the most typical is the hyperbolic tangent [42]. The main idea is to combine several of these simple units or neurons connecting several of them creating a network with a specific architecture able to capture the underlying function that rules the relationship between dependent and independent variables in a dataset where very complicated dynamic phenomena are involved.

In this article, the MLP ANN is used: It is a feedforward network (the information goes only in one direction without loops) formed by several layers of neurons between the input and the output. In a training phase, the inputs and their desired outputs are presented to the network and the weights (connection between neurons) are adjusted until reaching a mean-square error between the actual output and the desired output is low enough to consider that the MLP has been trained. Then, once the network has been trained, it is possible to calculate the estimated output just by entering the

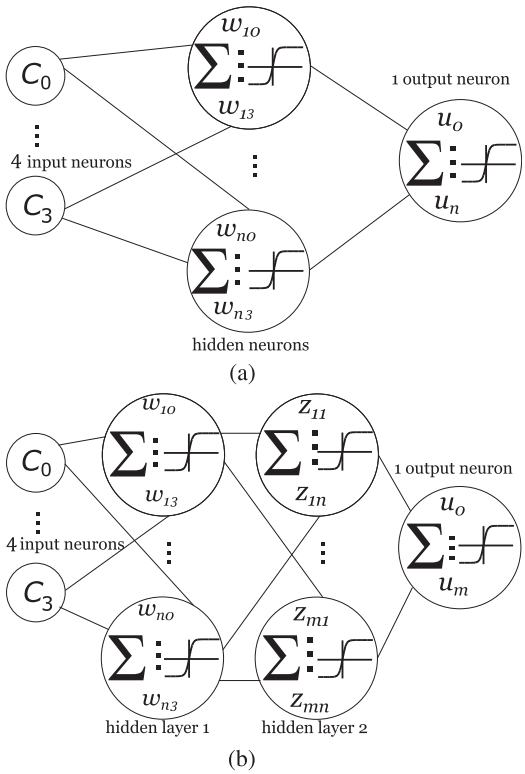

FIGURE 8. Different architectures for a multilayer perceptron. (a) With a unique hidden layer. (b) With two hidden layers.

inputs. During the training phase, the weights between neurons should be adjusted, taking into account the classification error of the training samples at the output layer. This error can be propagated from the last layer to the initial layer, modifying all the weights and bias; this process is repeated until the classification error reaches a minimum threshold. Instead of using the classical error back-propagation algorithm [43], based on the steepest descent optimization and with a very slow convergence, in this work, a variation based on the Levenberg–Marquardt method [44] is preferred.

Initially, to classify the samples into normal or fault, an MLP with only a single layer of hidden neurons is proposed [it can be seen in Fig. 8(a)]. Therefore, this MLP will be composed of an input layer, one hidden layer, and an output layer. There is one neuron for each input quantity; therefore, we have four neurons in the input layer. As we want to classify the samples into two categories, it is enough with only one neuron in the output layer that can be -1 or $+1$. The number i of hidden neurons must be set before the training phase. The determination of this number is not an easy task and there is not a generic rule widely accepted [45].

The optimal architecture of the MLP models (with reference to the number of neurons in each layer) has been found by implementing a grid search approach. It is based on a repeated run of the training routine with the number of hidden units changed between a minimum and a maximum, so that the outperforming configuration is found as the best one. In this work, we have tested the different options from $i = 2$ until $i = 20$). Another problem to take into account is that even using the same architecture and the same training set, the obtained result could be different because the initial values of the weight between neurons are randomly set at the beginning of the training algorithm. Hence, the same configuration should be tested several times (10 times in this work) obtaining different results.

With respect to the number of layers, it is accepted by the community only to test one hidden layer or two hidden layers, but the latter approach is reserved only for special cases with a very complicated underlying function [46], [47], [48]. Therefore, in this work, the MLPs with only one and two hidden layers have been tested. In fact, in the literature, there are cases where an MLP with two hidden layers [see Fig. 8(b)] is required to achieve better results than with only one layer [49]. This requires to determine the optimal combination of units for the first hidden layer ($i = 2, \ldots, 15$) and units for the second hidden layer ($j = 2, \ldots, 15$), repeating also the training 10 times for each combination. An approach very similar to this one has been described in a previous work [50].

H. RADIAL BASIS FUNCTIONS

There is another type of classifier known as RBF, in which the activation function of the neurons of the hidden layer is Gaussian [51]. The weights of each of the hidden neurons represent a point in an n-dimensional space, where n is the number of neurons in the input layer. For each input sample, each hidden neuron estimates the distance between said sample and the point that it represents, and to this distance, the Gaussian transfer function is applied. Finally, all the outputs of the hidden neurons are multiplied by some weights and summed up in an output layer neuron that determines the class to which the input sample belongs.

The underlying idea behind RBFs is that each neuron in the hidden layer represents a set of input samples that are similar, and therefore must be classified in the same way. The measure of such similarity is precisely the distance between the point represented by the hidden neuron and each input sample. The goal is to be able to classify a large number of input samples using a small set of hidden neurons. In this work, the number of units in the hidden layer is tested repeating the training procedure for values of i from 2 to 20. As in the case of the MLP, the initialization of the weights is made randomly, so the same value of i is again tested with the experimental curves (not seen during the training), a significant number of times (10 repetitions in our case), and the best RBF is considered for each value of i.

I. HYBRID APPROACHES WITH TWO HIDDEN LAYERS

To improve the performance, some authors have proposed the hybridization of MLP and RBF networks [52], [53]. Following this line, it would be possible to train an MLP with two layers of hidden neurons, but having in the first one all neurons with Gaussian transfer function *radbas*, and in the second layer, all neurons with hyperbolic tangent *tansig*

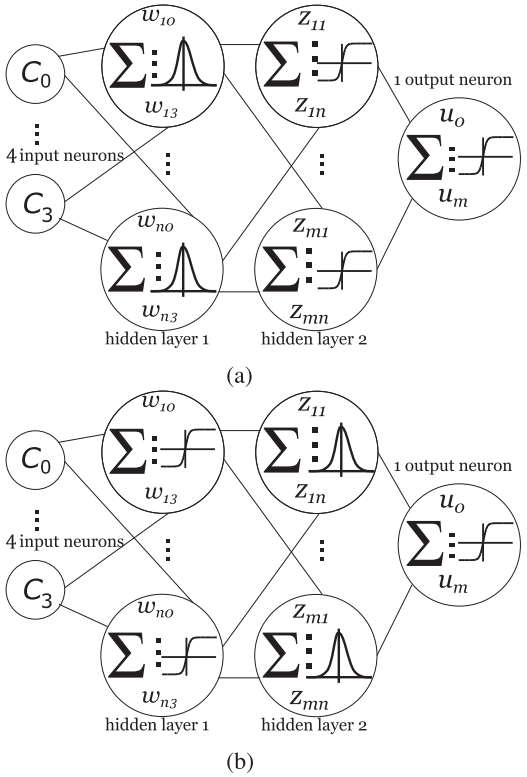

FIGURE 9. Different architectures for a hybrid multilayer perceptron. (a) First layer with *radbas* and second one with *tansig*. (b) First layer with *tansig* and second one with *radbas*.

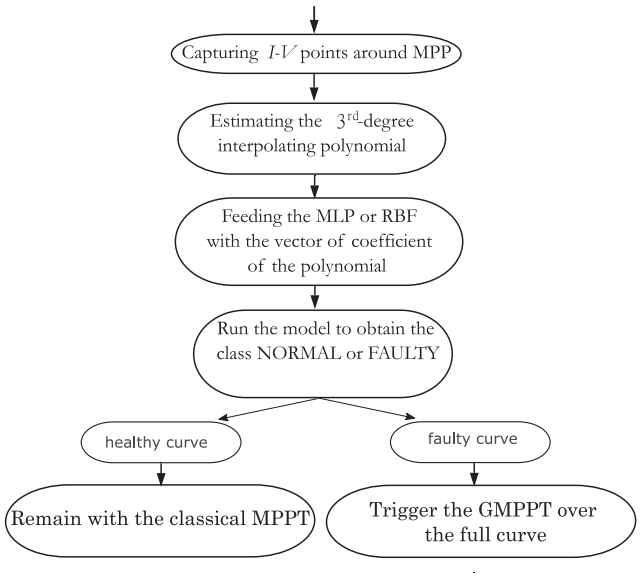

FIGURE 10. Flowchart summarizing the identification of mismatched conditions.

TABLE 1. Classification Errors (%), Total Training Time, and Predicting Time for Each Technique Using Kohonen SOM to Select the Training Set

	TREE	*SVM*	*MLP1*	*MLP2*	*HMLP1*	*HMLP2*	*RBF*
Total training (h)	$\simeq$0.0	5.74	6.22	39.16	40.34	50.13	3.94
Prediction time (ns)	152	2603	117	230	146	139	128
TEST #0 error (%)	7.1	0.3	0.3	0.25	0.25	0.25	0.3
TEST #1 error (%)	2.3	0.1	0.1	0.1	0.1	0.2	0.1
TEST #2 error (%)	22.4	5.8	5.8	11.4	5.8	7.1	11.3
TEST #3 error (%)	49.2	2.4	7.8	37.4	21.0	24.1	37.7

[referred in this article as HMLP1 and depicted in Fig. 9(a)]. Another option is the use of *tansig* for the first hidden layer and with *radbas* in the second hidden layer [noted as HMLP2 and shown in Fig. 9(b)]. In both cases, and as with the classic MLP of two hidden layers, in order to determine the optimal number of neurons for each layer it will be necessary to study the different possible combinations: for the first layer from $i = 2, \ldots, 15$, and for the second layer from $j = 2, \ldots, 15$. Again, each combination will be tested 10 times with a different random initialization.

IV. RESULTS

Once the best model has been identified and trained, it can be used to classify an *I–V* curve: The flowchart shown in Fig. 10 exemplifies the steps of the procedure proposed in this article.

The computer used for fitting the models is a laptop MSI Creator 17 A10SE with a CPU Intel i7-10875H with 8 cores and 32 GB RAM. As a first step, before fitting the models described in Section III, the SOM network is used to perform a selection of the most representative *I–V* curves from the initial set of 10 000 simulated ones. As it is explained in Section III-D, a 2-D self-organizing map with 50 × 50 neurons has been used. For each of the 2500 groups, the curve closest to its mass center is chosen as its representative, so that a total of 2500 *I–V* curves are selected to train the models. For the first phase of the routine *newsom/train* [37], 5000 iterations with a learning rate of 0.1 have been settled, whereas for the second phase, other 5000 iterations have been set with a learning rate of 0.01. The time required for this previous step has been a bit more than of 218 min, exactly 13 083 s.

By running *newsom/train*, over the total of 2500 *I–V* curves, 670 curves correspond to normal operating conditions and the remaining 1830 curves are associated with mismatched conditions. This confirms the greater variability of the curves representing the mismatched conditions.

The first row of Table 1 shows, for every proposed deep learning technique, the total time to execute the training routine for all the repetitions varying the parameters to be optimized. For example, in the SVM, the same training routine should be executed 240 × 90 times to cover a feasible range for $\mathcal{C}$ and γ. The same fact can be stated for the other paradigms, except for the decision tree whose fitting process is executed only once. More important than the training time is the prediction time for the best performing model of each proposed method (selecting the training model that optimizes the error over TEST#0). These times, in nanosecond, have been determined by running every technique on the same personal computer mentioned above (disabling previously the multicore feature because the models are intended to be run in a single-core embedded system). As these prediction times are

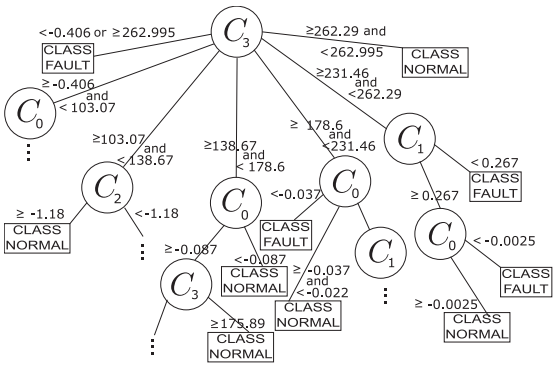

FIGURE 11. Decision tree obtained for the training set.

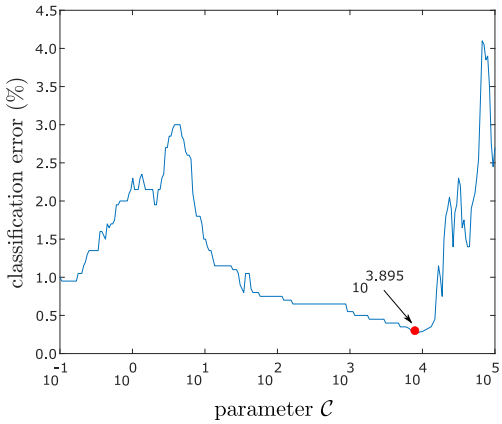

FIGURE 12. Evolution of the classification error in the SVM as a function of C.

very small to be measured accurately, each best trained model has been executed over a massive set of curves, and the mean execution times have been reported. The MLP with a single hidden layer of neurons results in the shorter computation time. On the other hand, the prediction time required by the best trained SVM is of one order of magnitude greater than the time required by all the other approaches.

At a first glance, all the proposed approaches obtain very low classification errors when they are applied to TEST#1, but for the other test sets, especially TEST#3, the results are very deceptive, except for the SVM and for the MLP with a unique hidden layer. The main difference is the level of shadow applied in each case. It seems that when the level of irradiance of the shadowed module in the series is below a threshold, it will not operate around the rounded part of the curve but also operates in a different point not learned during the training process. In fact, as some authors claim [54], the most relevant feature of the SVMs is their generalization power, i.e., their capacity to manage cases not seen during the training of the model. Now let us see the behavior of each model with more detail.

A. DECISION TREE RESULTS

As first, the classification of the *I–V* curves into *NORMAL* or *FAULTY* classes has been performed by using *decision trees*. The performance is summarized in the column headed by the label TREE of Table 1 that collects the classification errors over the different sets of experimental curves. The obtained result when fitting the tree, shown in Fig. 11, can be easily interpreted by a human. As it can be seen, for TEST#0 and TEST#1, the errors are not too high (7.1% and 2.3%, respectively). However, if the other test sets are used, very disappointing results are achieved. This is due to the tendency of this technique to overfit the training set [55]: It is able to capture with high level the irregularities of the samples used for training, but it sometimes cannot generalize for another cases that have not been analyzed during the fitting procedure.

B. SVM RESULTS

The SVM approach has been tested over the same training set of curves. It is worth to note that the implementation of the support vector classificator in Matlab, by *fitcsvm* [37], allows us to use different types of kernels, but the only one giving moderately acceptable results is the Gaussian kernel. Different values of γ and $\mathcal{C}$ within a range, from a minimum to a maximum repeating the training algorithm for each possible combination, have been tested. Finally, the best SVM is that one that achieves the smallest classification error over a test set, which is different from the training set. All the combinations for γ and $\mathcal{C}$ have been tested, considering 90 different values of $\gamma \in [0.1, 1.0]$ (linearly spaced) and 240 values of $\mathcal{C} \in [10^{-1}, 10^{5}]$ (logarithmically spaced). The total execution time for the training of all the combinations was 5.74 h. This time is required to optimize the two parameters γ and $\mathcal{C}$ simultaneously because all the possible combinations of them have to be tested, repeating the training procedure for every case (a lower total execution time could be possible reducing the range and interval step for γ and $\mathcal{C}$ but a worse solution could have been achieved). The best result over the experimental set of curves has been achieved with $\gamma = 0.69$ and $\mathcal{C} = 10^{3.895}$. The resulting classification error over TEST#0 is only 0.3%. The results over the other test sets have been summarized in Table 1 under the column headed by the label SVM. As it can be seen, this best trained SVM ensures the optimal results over TEST#1, TEST#2, and TEST#3, outperforming all other methods. This means that obtained SVM has a great power of generalization.

In Section III-F, it was stated that the performance of an SVM classifier depends on the value of the parameter $\mathcal{C}$. This dependence is shown in Fig. 12, which gives the classification error of different trained SVMs, by fixing the value of $\gamma = 0.69$ and by varying $\mathcal{C}$ from 10^{-1} to 10^{5}. Initially, as $\mathcal{C}$ increases, the classification error also increases. However, once a local maximum error around 3% is achieved, there is a decrement until the global minimum of 0.3% is reached.

C. ONE HIDDEN LAYER PERCEPTRON RESULTS

The MLP-based approach with only 1 hidden layer has been also tested, by increasing the number of neurons i from 2 until 20. For each number of hidden neurons, the training process

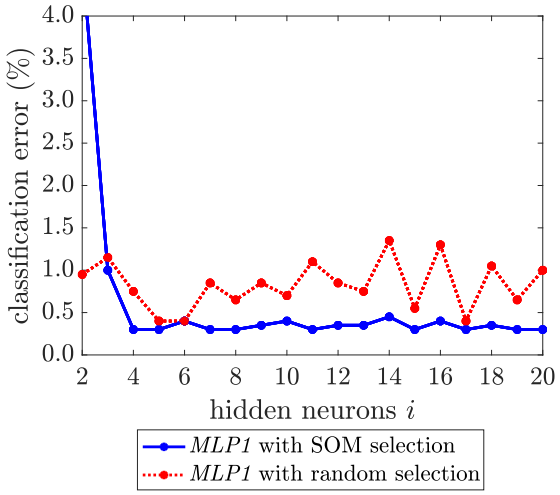

FIGURE 13. Classification error as a function of the number of hidden neurons for MLP1.

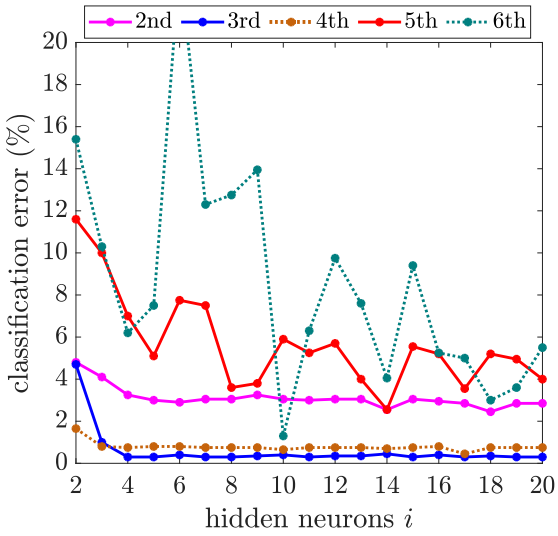

FIGURE 14. Classification error as a function of the number of hidden neurons for different interpolating polynomials.

has been executed 10 times and the results have been compared. For a fixed number of neurons in the hidden layer, the best trained network is the MLP with the minimum classification error over the experimental set of curves TEST#0, which are not used for the training. Finally, the optimal number of neurons has been fixed by taking into account the evolution of the error from 2 until 20. Fig. 13 shows the error as a function of the number i of neurons in the hidden layer (solid blue line). It is only necessary to use four neurons in the hidden layer to reach a minimum classification error of 0.3%, in such a way, since this point, increasing the number of hidden units does not improve the obtained results.

As it can be seen in the column titled MLP1 of Table 1, the results over TEST#1 and TEST#2 are identical to the ones obtained in the SVM case, highlighting also the generalization capabilities of the MLP. However, for TEST#3, which seems to be more challenging, the MLP1 is significantly worse than the SVM. In Fig. 13, it is shown the evolution of the error but in case of training the MLP not using the proposed smart selection of the samples, but using a random selection. This subplot is analyzed in Section IV-F.

The total training time is 6.22 h, required to perform the search to optimize the number of hidden neurons. Again, this time is required because the training procedure provided by Matlab has been repeated 10 times for every different value of i from 2 till 20, i.e., it has been run 190 times over the same training set. With the smallest prediction time of all the models (117 ns), the MLP with a single hidden layer achieves good results in terms of error for all the testing sets, only outperformed by the SVM model in the TEST#3 case.

The adoption of a third degree polynomial for interpolating the curve samples across the MPP is justified on the basis of different repetitions of the complete training process over the same training set, only varying the degree of that interpolating polynomial, from the second to the sixth. For the sake of brevity, in Fig. 14, only the results referring to the MLP with one hidden layer are shown. On the one hand, when using a low degree interpolating polynomial (second, third, and fourth), the classification error decreases as the number of neurons in the hidden layer increases, until it reaches a minimum, and then remains stable. On the other hand, for the fifth and sixth cases, the classification errors do not follow a clearly trend as a function of the number of hidden neurons. This comparative reveals that the third degree ensure the best classification errors in the considered range of hidden neurons.

D. RBF RESULTS

As it is explained in Section III-H, the training procedure of the RBF is also performed by changing the number of neurons in the hidden layer from $i = 2$ to $i = 20$. In addition, fixing the value of i, the training is also repeated 10 times, giving different trained RBFs due to the random initialization of the weights. In order to determine the optimal number of hidden neurons, we will take into account the best RBF among the 10 repetitions for the same value of i, i.e., the RBF with the smallest error over the experimental test set.

As it can be seen in Fig. 15 (solid blue line), the error of the best RBF over TEST#0 does not show a very clear tendency as a function of the number of hidden neurons, but it seems to reach a minimum value of 0.3% when $i = 8$ (the red line is referred RBFs without using the smart selection of the samples to be trained). In addition, the classification error over TEST#1 equals the results obtained by the SVM and MLP1. However, for TEST#2 and TEST#3, the achieved results are significantly worse.

E. TWO HIDDEN LAYER PERCEPTRON RESULTS

A further test has been done by adding to the MLP a second layer of hidden neurons (referred as MLP2). In order to determine the optimal architecture, all the possibilities between

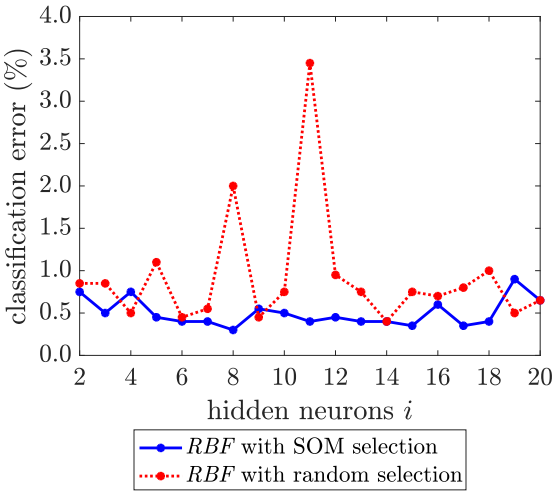

FIGURE 15. Classification error as a function of the number of hidden neurons for RBF.

2 and 15 neurons for the first and the second hidden layers ($14 \times 14 = 196$ different architectures) have been analyzed, by executing the training algorithm 10 times for each architecture. By using an architecture with two hidden layers, the best trained neural network (taking into account TEST#0) is achieved with 10 neurons in the first layer and 14 neurons in the second layer, with a classification error over of 0.25%. Using as test set TEST#1, as in the SVM and MLP1 cases, the error is again 0.1%. Unfortunately, the results obtained for TEST#2 and TEST#3 are quite bad.

The hybrid approach referred in Section III-I as HMLP1 achieves a minimum error over TEST#0 of 0.25% with 6 neurons in the first hidden layer (*radbas*) and 2 units in the second layer (*tansig*). This hybrid architecture, with a first hidden layer using a Gaussian transfer function, allows us to achieve good results for all the test sets except for TEST#3.

The neural network HMLP2, configured with 4 *tansig* units in the first layer and 4 *radbas* units in the second layer, is able also to reach an error of 0.25% over TEST#0. However, the results for the other test sets are worse than the obtained ones with HMLP1. In general, having more layers implies that the training set is overfitted. Moreover, as there are two parameters to optimize (the number of neurons of both layers), the total execution time is very high for the three cases with two hidden layers, as it can be seen in Table 1.

F. IMPROVEMENT DUE TO SOM

In order to quantify the improvement achieved by the smart selection of the samples based on the Kohonen SOM network, the obtained results when using a random selection of the samples are summarized in Table 2, and they can be compared with those ones shown in Table 1.

In general, the error values are smaller when using the approach proposed in this article, being the difference being very significant for the SVM and the MLP1. In both cases, the selection of the training samples using SOM appears to be crucial in order to have an operative model for both neural networks. Therefore, the implementation of the approach proposed in this work to select the training samples has a significant beneficial effect in terms of classification error, since the generalization power is improved significantly, except in the case of the decision trees.

TABLE 2. Classification Errors (%), Total Training Time, and Predicting Time for Each Technique Using a Random Selection of the Training Set

	TREE	*SVM*	*MLP1*	*MLP2*	*HMLP1*	*HMLP2*	*RBF*
Total training (h)	$\simeq 0.0$	18.02	7.84	64.93	47.02	54.72	2.90
Prediction time (ns)	168	3999	195	221	164	159	141
TEST #0 error (%)	4.1	0.3	0.55	0.3	0.3	0.35	0.45
TEST #1 error (%)	5.6	19.0	37.7	0.6	0.2	0.7	0.1
TEST #2 error (%)	43.9	14.8	43.2	12.4	13.3	19.5	11.5
TEST #3 error (%)	51.4	2.8	46.6	38.5	45.5	49.0	37.9

In addition to the aforementioned, as it can be seen in Figs. 13 and 15, when using the Kohonen SOM to select the training samples, the tendency of the classification error as a function of the number i of hidden neurons is very clear, being very easy to determine the optimal value of i. However, with the random selection of the training samples, it is very complicated to optimize the number of hidden neurons because the classification error seems to affect by a random oscillation.

V. CONCLUSION

A comparison of several paradigms belonging to the data mining field has been presented. They have been used to detect the occurrence of mismatched conditions in a photovoltaic array by using only a reduced set of points very close to the MPP of its characteristic curve. It has been demonstrated that mismatched conditions are detected without measuring the whole curve, thus without stopping the MPP tracking operation. The need of a fixed-length array of I–V samples as input for all the approaches compared in this article has suggested the adoption of a polynomial of a fixed degree that interpolates the points around the MPP. The four coefficients of this polynomial are collected in the input array for all the data mining approaches used in this article. The models have been fitted by using simulated curves covering a wide range of the array operating conditions. This guarantees that the proposed models can be applied to very different situations.

It is shown that the performance of the approaches is improved by selecting the most representative curves by using a Kohonen self-organizing map. This procedure allows us to give more importance in the training process to those curves that are less frequent, by minimizing the error for those cases. In addition, this smart selection of the training samples makes easier the optimization of the number of hidden neurons.

It has been shown that the decision tree is the only approach needing a short computation time to generate a result that can be easily understood by a human. Unfortunately, it gives the worst classification results.

The SVM and MLP with one hidden layer achieve the best results, being the former even better than the second one for

some extreme cases of mismatching. However, the main disadvantage of the support vector classifier is the high prediction time required for its execution. It has been also shown that adding a second layer of hidden neurons to the multilayer perceptron does not lead to improved results, due to overfitting problems and a very high training time.

ACKNOWLEDGMENT

The authors would like to thank the availability of the measurements provided by the research group of the *Laboratory of Photovoltaic Systems* of the *University of Málaga* lead by Prof. Mariano Sidrach-de-Cardona.

REFERENCES

[1] I. Tobías, C. del Cañizo, and J. Alonso, *Crystalline Silicon Solar Cells and Modules*, 2nd ed., ch. 7. Chichester, U.K.: Wiley, 2011, pp. 265–313, doi: 10.1002/9780470974704.ch7.

[2] L. Castañer, S. Bermejo, T. Markvart, and K. Fragaki, *Energy Production by a PV Array*, 2nd ed., ch. IIA-2. Waltham, MA, USA: Academic, 2012, pp. 645–658, doi: 10.1016/B978-0-12-385934-1.00018-0.

[3] R. Cavieres, R. Barraza, D. Estay, J. Bilbao, and P. Valdivia-Lefort, "Automatic soiling and partial shading assessment on PV modules through RGB images analysis," *Appl. Energy*, vol. 306, 2022, Art. no. 117964, doi: 10.1016/j.apenergy.2021.117964.

[4] S. Fan, Y. Wang, S. Cao, B. Zhao, T. Sun, and P. Liu, "A deep residual neural network identification method for uneven dust accumulation on photovoltaic (PV) panels," *Energy*, vol. 239, 2022, Art. no. 122302, doi: 10.1016/j.energy.2021.122302.

[5] H. P.-C. Hwang, C. C.-Y. Ku, and J. C.-C. Chan, "Detection of malfunctioning photovoltaic modules based on machine learning algorithms," *IEEE Access*, vol. 9, pp. 37210–37219, 2021, doi: 10.1109/ACCESS.2021.3063461.

[6] M. Le, V. S. Luong, D. K. Nguyen, V.-D. Dao, N. H. Vu, and H. H. T. Vu, "Remote anomaly detection and classification of solar photovoltaic modules based on deep neural network," *Sustain. Energy Technol. Assessments*, vol. 48, 2021, Art. no. 101545, doi: 10.1016/j.seta.2021.101545.

[7] A. M. Sizkouhi, M. Aghaei, and S. M. Esmailifar, "A deep convolutional encoder-decoder architecture for autonomous fault detection of PV plants using multi-copters," *Sol. Energy*, vol. 223, pp. 217–228, 2021, doi: 10.1016/j.solener.2021.05.029.

[8] W. Tang, Q. Yang, K. Xiong, and W. Yan, "Deep learning based automatic defect identification of photovoltaic module using electroluminescence images," *Sol. Energy*, vol. 201, pp. 453–460, 2020, doi: 10.1016/j.solener.2020.03.049.

[9] N.V. Sridharan and V. Sugumaran, "Convolutional neural network based automatic detection of visible faults in a photovoltaic module," *Energy Source Part A*, pp. 1–16, 2021, doi: 10.1080/15567036.2021.1905753.

[10] 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, 2017, doi: 10.1016/j.enconman.2017.09.019.

[11] T. W. David, G. A. Soares, N. Bristow, D. Bagnis, and J. Kettle, "Predicting diurnal outdoor performance and degradation of organic photovoltaics via machine learning; relating degradation to outdoor stress conditions," *Prog. Photovolt.*, vol. 29, no. 12, pp. 1274–1284, 2021, doi: 10.1002/pip.3453.

[12] J. Pan, W. He, Y. Shi, R. Hou, and H. Zhu, "Uncertainty analysis based on non-parametric statistical modelling method for photovoltaic array output and its application in fault diagnosis," *Sol. Energy*, vol. 225, pp. 831–841, 2021, doi: 10.1016/j.solener.2021.07.064.

[13] N. Rakesh, S. Banerjee, S. Subramaniam, and N. Babu, "A simplified method for fault detection and identification of mismatch modules and strings in a grid-tied solar photovoltaic system," *Int J. Emerg. Elect. Power Syst.*, vol. 21, no. 4, 2020, Art. no. 20200001, doi: 10.1515/ijeeps-2020-0001.

[14] M. Mansouri, M. Trabelsi, H. Nounou, and M. Nounou, "Deep learning-based fault diagnosis of photovoltaic systems: A comprehensive review and enhancement prospects," *IEEE Access*, vol. 9, pp. 126286–126306, 2021, doi: 10.1109/ACCESS.2021.3110947.

[15] A. Ul-Haq, H. F. Sindi, S. Gul, and M. Jalal, "Modeling and fault categorization in thin-film and crystalline PV arrays through multilayer neural network algorithm," *IEEE Access*, vol. 8, pp. 102235–102255, 2020, doi: 10.1109/ACCESS.2020.2996969.

[16] B. Basnet, H. Chun, and J. Bang, "An intelligent fault detection model for fault detection in photovoltaic systems," *J. Sensors*, vol. 2020, 2020, Art. no. 6960328, doi: 10.1155/2020/6960328.

[17] J. Wang, L. Wang, J. Qu, and Z. Qian, "Novel application of heterogeneous ensemble learning in fault diagnosis of photovoltaic modules," in *Proc. Int. Conf. Smart-Green Technol. Elect. Inf. Syst.*, 2021, pp. 118–124, doi: 10.1109/ICSGTEIS53426.2021.9650390.

[18] A. Haque, K. V. S. Bharath, M. A. Khan, I. Khan, and Z. A. Jaffery, "Fault diagnosis of photovoltaic modules," *Energy Sci. Eng.*, vol. 7, no. 3, pp. 622–644, 2019, doi: 10.1002/ese3.255.

[19] 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," *IEEE Access*, vol. 8, pp. 41889–41904, 2020, doi: 10.1109/ACCESS.2020.2977116.

[20] H. Zhu, L. Lu, J. Yao, S. Dai, and Y. Hu, "Fault diagnosis approach for photovoltaic arrays based on unsupervised sample clustering and probabilistic neural network model," *Sol. Energy*, vol. 176, pp. 395–405, 2018, doi: 10.1016/j.solener.2018.10.054.

[21] S. R. Madeti and S. Singh, "Modeling of PV system based on experimental data for fault detection using KNN method," *Sol. Energy*, vol. 173, pp. 139–151, 2018, doi: 10.1016/j.solener.2018.07.038.

[22] M. M. Badr, M. S. Hamad, A. S. Abdel-Khalik, R. A. Hamdy, S. Ahmed, and E. Hamdan, "Fault identification of photovoltaic array based on machine learning classifiers," *IEEE Access*, vol. 9, pp. 159113–159132, 2021, doi: 10.1109/ACCESS.2021.3130889.

[23] A. Eskandari, J. Milimonfared, M. Aghaei, and A. H. Reinders, "Autonomous monitoring of line-to-line faults in photovoltaic systems by feature selection and parameter optimization of support vector machine using genetic algorithms," *Appl. Sci.*, vol. 10, no. 16, 2020, Art. no. 5527, doi: 10.3390/app10165527.

[24] L. C. Chen et al., "Fault diagnosis and classification for photovoltaic arrays based on principal component analysis and support vector machine," *IOP Conf. Ser. Earth Environ.*, vol. 188, Oct. 2018, Art. no. 12089, doi: 10.1088/1755-1315/188/1/012089.

[25] 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, doi: 10.1109/TIE.2017.2703681.

[26] 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, doi: 10.1109/ACCESS.2020.3020296.

[27] 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, 2020, Art. no. 308, doi: 10.3390/en13020308.

[28] Yocasol, "54-cell polycrystalline module family," 2009. [Online]. Available: https://www.dropbox.com/s/zlrjzgeerbtmmzr/PCB_GBDefinitivos.pdf

[29] M. Piliougine, J. Carretero, L. Mora-López, and M. Sidrach-de-Cardona, "Experimental system for current–voltage curve measurement of photovoltaic modules under outdoor conditions," *Prog. Photovolt.*, vol. 19, no. 5, pp. 591–602, 2011, doi: 10.1002/pip.1073.

[30] M. Piliougine, J. Carretero, L. Mora-López, and M. Sidrach-de-Cardona, "New software tool to characterize photovoltaic modules from commercial equipment," in *Proc. WEENTECH Energy*, 2018, pp. 211–220, doi: 10.5281/zenodo.6380294.

[31] M. Piliougine et al., "Analysis of the degradation of single-crystalline silicon modules after 21 years of operation," *Prog. Photovolt.*, vol. 29, no. 8, pp. 907–919, 2021, doi: 10.1002/pip.3409.

[32] Solbian, "Flex SP50–L Flexible Photovoltaic Panels," [Online]. Available: https://www.dropbox.com/s/xtdp3axmxqkbg2a/SOLBIAN_50W_sp50l_eng.pdf

[33] D. Dirnberger and U. Kräling, "Uncertainty in PV module measurement - Part I: Calibration of crystalline and thin-film modules," *IEEE J. Photovolt.*, vol. 3, no. 3, pp. 1016–1026, Jul. 2013, doi: 10.1109/JPHOTOV.2013.2260595.

[34] K. Emery, Photovoltaic calibrations at the national renewable energy laboratory and uncertainty analysis following the ISO 17025 guidelines, no. NREL/TP-5J00-66873. Golden (CO, USA): Nat. Renewable Energy Lab., NREL, Sep. 2016. [Online]. Available: https://www.nrel.gov/docs/fy17osti/66873.pdf

[35] T. Rodziewicz, M. Rajfur, and M. Wacławek, "The use of two-diode substitute model in predicting the efficiency of PV conversion in low solar conditions," *Ecol. Chem. Eng. S.*, vol. 24, no. 2, pp. 177–202, 2017, doi: 10.1515/eces-2017-0012.

[36] T. Kohonen, "The self-organizing map," *Proc. IEEE*, vol. 78, no. 9, pp. 1464–1480, 1990, doi: 10.1109/5.58325.

[37] M. H. Beale, M. T. Hagan, and H. B. Demuth, *Matlab Deep Learning Toolbox. User's Guide. R2021b*. Natick, MA, USA: MathWorks, 2021. [Online]. Available: https://www.dropbox.com/s/vcle2ig7zh5h49l/nnet_ug.pdf

[38] Y. Zhao, L. Yang, B. Lehman, J. de Palma, J. Mosesian, and R. Lyons, "Decision tree-based fault detection and classification in solar photovoltaic arrays," in *Proc. 27th IEEE Appl. Power Electron. Conf. Expo.*, 2012, pp. 93–99, doi: 10.1109/APEC.2012.6165803.

[39] R. Benkercha and S. Moulahoum, "Fault detection and diagnosis based on C4.5 decision tree algorithm for grid connected PV system," *Sol. Energy*, vol. 173, pp. 610–634, 2018, doi: 10.1016/j.solener.2018.07.089.

[40] L. Breiman, J. Friedman, C. Stone, and R. Olshen, *Classification and Regression Trees*. New York, NY, USA: Taylor & Francis, 1984, doi: 10.1201/9781315139470.

[41] V. N. Vapnik, *Statistical Learning Theory*. Hoboken, NJ, USA: Wiley, 1998.

[42] S. O. Haykin, "Multilayer perceptrons," in *Neural Networks and Learning Machines*, 3rd ed., ch. 4. London, U.K.: Pearson, 2009, pp. 122–229.

[43] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986, doi: 10.1038/323533a0.

[44] H. Yu and B. M. Wilamowski, "Levenberg–Marquardt training," in *Intelligent Systems* (ser. The Industrial Electronics Handbook), B. M. Wilamowski and J. D. Irwin, Eds., ch. 12. Boca Raton, FL, USA: CRC Press, 2011, doi: 10.1201/9781315218427.

[45] J. F. Mas and J. J. Flores, "The application of artificial neural networks to the analysis of remotely sensed data," *Int J. Remote Sens.*, vol. 29, no. 3, pp. 617–663, 2008, doi: 10.1080/01431160701352154.

[46] R. D. Reed and R. J. Marks, *Neural Smithing. Supervised Learning in Feedforward Artificial Neural Networks*. Cambridge, MA, USA: MIT Press, 1999.

[47] B. Macukow, "Neural networks - State of art, brief history, basic models and architecture," in *Computer Information Systems and Industrial Management*, K. Saeed and W. Homenda, Eds. Cham, Switzerland: Springer, 2016, pp. 3–14, doi: 10.1007/978-3-319-45378-1_1.

[48] T. T. K. Tran, T. Lee, and J.-S. Kim, "Increasing neurons or deepening layers in forecasting maximum temperature time series?," *Atmosphere*, vol. 11, no. 10, 2020, Art. no. 1072, doi: 10.3390/atmos11101072.

[49] B. García-Domingo, M. Piliougine, D. Elizondo, and J. Aguilera, "CPV module electric characterisation by artificial neural networks," *Renew. Energy*, vol. 78, pp. 173–181, 2015, doi: 10.1016/j.renene.2014.12.050.

[50] M. Piliougine and G. Spagnuolo, "Mismatching and partial shading identification in photovoltaic arrays by an artificial neural network ensemble," *Sol. Energy*, vol. 236, pp. 712–723, 2022, doi: 10.1016/j.solener.2022.03.026.

[51] S. O. Haykin, "Kernel methods and radial-basis function networks," in *Neural Networks and Learning Machines*, 3rd ed., ch. 5. London, U.K.: Pearson, 2009, pp. 230–267.

[52] V. Ghorbani, M. Vadood, and M. S. Johari, "Prediction of polyester/cotton blended rotor-spun yarns hairiness based on the machine parameters," *Indian J. Fibre Text*, vol. 41, pp. 16–25, 2016. [Online]. Available: http://nopr.niscair.res.in/handle/123456789/33863

[53] S. N. Hidayat and K. Triyana, "Optimized back-propagation combined with radial basic neural network for improving performance of the electronic nose: Case study on the fermentation process of tempeh," *AIP Conf. Proc.*, vol. 1755, no. 1, 2016, Art. no. 20001, doi: 10.1063/1.4958466.

[54] J. Wang, D. Gao, S. Zhu, S. Wang, and H. Liu, "Fault diagnosis method of photovoltaic array based on support vector machine," *Energy Source Part A*, pp. 1–16, 2019, doi: 10.1080/15567036.2019.1671557.

[55] A. Amro, M. Al-Akhras, K. El Hindi, M. Habib, and B. A. Shawar, "Instance reduction for avoiding overfitting in decision trees," *J. Intell. Syst.*, vol. 30, pp. 438–459, 2021, doi: 10.1515/jisys-2020-0061.