energies

MDPI

Article

Ensemble LVQ Model for Photovoltaic Line-to-Line Fault Diagnosis Using K-Means Clustering and AdaGrad

Peyman Ghaedi [1], Aref Eskandari [2,*], Amir Nedaei [3], Morteza Habibi [1], Parviz Parvin [1] and Mohammadreza Aghaei [4,5,*]

[1] Department of Physics and Energy Engineering, Amirkabir University of Technology (AUT), Tehran 15916-34311, Iran; peymanghaedi@aut.ac.ir (P.G.); mortezahabibi@aut.ac.ir (M.H.); parvin@aut.ac.ir (P.P.)
[2] Department of Electrical Engineering, Iran University of Science and Technology (IUST), Tehran 16846-13114, Iran
[3] Department of Electrical Engineering, Amirkabir University of Technology (AUT), Tehran 15916-34311, Iran; amirnedaei@aut.ac.ir
[4] Department of Ocean Operations and Civil Engineering, Norwegian University of Science and Technology (NTNU), 6009 Ålesund, Norway
[5] Department of Sustainable Systems Engineering (INATECH), University of Freiburg, 79110 Freiburg, Germany
* Correspondence: aeskandari@iust.ac.ir (A.E.); mohammadreza.aghaei@ntnu.no or mohammadreza.aghaei@imtek.uni-freiburg.de (M.A.)

Abstract: Line-to-line (LL) faults are one of the most frequent short-circuit conditions in photovoltaic (PV) arrays which are conventionally detected and cleared by overcurrent protection devices (OCPDs). However, OCPDs are shown to face challenges when detecting LL faults under critical detection conditions, i.e., low mismatch levels and/or high fault impedance values. This occurs due to insufficient fault current, thus leaving the LL faults undetected and leading to power losses and even catastrophic fire hazards. To compensate for OCPD deficiencies, recent studies have proposed modern artificial intelligence (AI)-based methods. However, various limitations can still be witnessed even in AI-based methods, such as (i) most of the models requiring a massive training dataset, (ii) critical fault detection conditions not being taken into consideration, (iii) models not being accurate enough when dealing with critical conditions, etc. To this end, the present paper proposes a learning vector quantization (LVQ)-based ensemble learning model in which three LVQs are individually trained to detect and classify LL faults in PV arrays. The initial LVQ vectors are determined using the k-means clustering method, and the learning rate is optimized by the adaptive gradient (AdaGrad) optimizer. The training and testing datasets are collected according to the PV array's current–voltage (I–V) characteristic curve, and several features are extracted based on the Canberra and chi-squared distance techniques. The model utilizes a small training dataset, considers various critical detection conditions for LL faults—such as different mismatch levels and fault impedance values—and the final experimental results show that the model achieves an impressive average accuracy of 99.26%.

Keywords: photovoltaic (PV); fault detection; line-to-Line (LL) faults; learning vector quantization (LVQ); K-means clustering; adaptive gradient (AdaGrad) optimizer; ensemble learning

check for updates

Citation: Ghaedi, P.; Eskandari, A.; Nedaei, A.; Habibi, M.; Parvin, P.; Aghaei, M. Ensemble LVQ Model for Photovoltaic Line-to-Line Fault Diagnosis Using K-Means Clustering and AdaGrad. *Energies* **2024**, *17*, 5269. https://doi.org/10.3390/en17215269

Academic Editor: Jiaqiang E

Received: 13 September 2024
Revised: 29 September 2024
Accepted: 20 October 2024
Published: 23 October 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

1.1. Motivation

There has been a notable focus on decreasing carbon footprints and shifting to sustainable energy sources in energy policies worldwide in recent years. This can be observed through the increased adoption of renewable energy sources (RES) to ensure a more environmentally friendly future. Among RES, solar photovoltaics (PV) have been recognized as a crucial energy source that effectively addresses environmental concerns [1].

However, it is crucial to reiterate that intense sunlight, persistent shading, humidity, and other factors can threaten PV components. These threats can lead to various electrical faults and breakdowns, impacting the system's normal conditions and, in some instances, potentially leading to fire hazards [2]. This underscores the undeniable importance of monitoring PV systems, a measure that should always be considered and addressed [3].

Among electrical failures in PV systems, line-to-line (LL) faults are frequently encountered and are particularly difficult to detect compared to other faults and failures. Two primary factors that determine the impact of LL faults are (i) the modules affected by the fault and (ii) the fault impedance, making LL fault detection quite challenging for protective devices [4].

LL fault detection and clearing in PV arrays is complex, and traditional devices like overcurrent protection devices (OCPDs) seem inadequate [5]. This inadequacy is due to various factors, including high fault impedance, a small number of modules involved in the LL fault (referred to as a low-mismatch level), low levels of irradiance, blocking diodes, and the in-service maximum power point tracker (MPPT). Therefore, a comprehensive examination is essential in accurately detecting, classifying, and also assessing the severity of LL faults in PV arrays [6].

1.2. Review of Related Work

To address the limitations and deficiencies of OCPDs in detecting the LL faults in PV arrays, recent research has shifted towards employing artificial intelligence (AI)-based methods, and particularly machine learning (ML) as a branch of AI [7].

In the beginning, single ML classifiers, such as support vector machine (SVM), k-nearest neighbors (kNN), naïve Bayes (NB), decision tree (DT), logistic regression (LogR), etc., were fine-tuned and employed to perform the fault detection and classification tasks in PV arrays. A single DT classifier was presented by R. Benkercha and S. Moulahoum [8], and it showed an accurate performance in LL fault detection and diagnosis in a grid-connected PV system (GCPVS). Furthermore, the authors in [9] developed a fault detection and diagnosis (FDD) model based on kNN in which data samples are collected using a detailed model of the PV system based on experimental data using standard test conditions (STC: irradiation = 1000 W/m^2, temperature = 25 °C, air-mass = 1.5) and nominal operating cell temperature (NOCT), and remarkable points from the manufacturer's datasheet.

Later, algorithm selection techniques appeared in various comparative studies, in which they started to test several single ML classifiers under different conditions to finally obtain the most accurate classifier. In [10], SVM, kNN, DT, NB, discriminant analysis (DA), etc. are compared along with different groups of features. Finally, kNN was nominated with a specific group of features. In another comparative algorithm selection approach [11], SVM was selected in comparison with kNN, DA, and DT when features were selected using the Salp Swarm Algorithm (SSA). Authors in [12] compared DT, kNN, and SVM to provide an optimal ML classifier for automatic PV faults detection and classification. Authors in [13] presented a hierarchical classification model which included several levels. Each level was in charge of comparing and selecting the most accurate classifier among LogR, NB, and SVM.

However, the use of single algorithms gradually showed its deficiencies. The main shortcoming was that to provide a comprehensive fault diagnosis scheme, new datasets started to become more challenging to classify, as they contained data samples under LL fault critical detection conditions. These conditions include low mismatch levels (small number of modules involved in the fault) and/or high fault impedance values. Hence, single classifiers were not powerful enough to deal with high levels of dataset non-linearity and computational complexity.

To overcome the challenge, several solutions were suggested in literature. One significant solution was to group single and weak ML classifiers and create an ensemble learning (EL) model [14]. EL models are based on obtaining the output predictions from each single classifier and combining the outputs using various techniques, such as majority

voting, weighted voting, etc., to produce the final output prediction. In [15], L. Wang et al. adopted three multi-class SVM (MSVM) classifiers and predicted the final result according to the majority voting technique. In addition, Z. Chen et al. presented an EL model in [16] which was a random forest (RF) algorithm (a multitude of decision trees) in which the final prediction was determined by the majority voting of DT classifiers output predictions. The tendency to ensemble learning continued—J. M. Huang et al. [17] used the adaptive boosting (AdaBoost) classifier. The model was the stage-wise additive modeling using a multi-class exponential loss function (SAMME). Exclusive LL fault detection and classification using a multi-classifier-based approach through a combination of several ML algorithms was pioneered in [18], in which the authors employed NB, SVM, and kNN and formed an EL model. The final prediction was based on a probabilistic majority voting approach. In [19], Emamian et al. introduced a novel IoT-based monitoring system to provide smart and autonomous monitoring. It used an EL model including NB, SVM, and kNN for the detection and classification of several types of LL faults. In [20], a genetic algorithm (GA) is also utilized to create a weighted EL model. They compared their model with the conventional majority-voting-based ensemble algorithm and proved to have comparatively higher accuracy in fault diagnosis and classification in PV arrays. More advanced versions of ensemble models can include the stacking ensemble model containing histogram-based gradient boosting, AdaBoost, extra trees (EXT), and gradient booting presented in Ref. [21]. In addition, an accurate ensemble model containing LR, DT, and SVM was proposed in Ref. [22]. Popular DT and SVM were also proposed along with kNN in an ensemble model by the authors of Ref. [23]. Stacking models have also been popular—another stacking ensemble model containing RF, extreme gradient boosting (XGBoost), light gradient boosting machine (LightGBM), EXT, and DT was recently presented in Ref. [24].

However, various gaps can still be witnessed in existing EL models such as:

(i) A number of previous EL models in literature which produce high accuracies have not taken LL fault critical detection conditions into consideration, casting doubt on their reliability in real-world usages.
(ii) Most of previous EL models available in literature do not produce satisfying accuracies when dealing with LL fault critical detection conditions.
(iii) Producing sufficient accuracies under LL fault critical detection conditions forced the EL models to utilize massive training datasets. However, real data sample acquisition from PV arrays is extremely challenging since PV components mostly operate outdoors and under harsh weather conditions.
(iv) Regarding the concept of PV protection, the accuracy of EL models can still be further improved to develop more reliable FDD schemes to be implemented in real-world contexts.

1.3. Paper Contributions

To address the abovementioned deficiencies, the present study provides the following novelties and contributions for detecting and classifying different types of LL faults in PV arrays:

- Learning vector quantization (LVQ) algorithms have been utilized for the first time as an ensemble learning model resulting in a significant performance enhancement and accuracy improvement;
- K-means clustering is employed to start with LVQ algorithms since it ensures an even spread of the initial codebook vectors (weight vectors) across the data and prevents poor initializations, less-than-optimal solutions, and slow convergence;
- An adaptive gradient (AdaGrad) optimizer is used to adaptively optimize the learning rates of LVQ algorithms for a quicker and more dependable convergence than a constant learning rate;
- Canberra distance and chi-squared distance methods are utilized in this study to derive different novel features from the PV array's I–V curve. The distances are measured based on five specific pre-defined points on the curve;

- The final model experiments have shown its exceptional ability to detect and classify LL faults, particularly under the most difficult detection conditions, i.e., high impedance values and low mismatch levels.

1.4. Paper Organization

The remaining content is structured as follows: Section 2 provides a comprehensive analysis of LL faults in PV arrays. A detailed explanation of the proposed approach can be found in Section 3, while Section 4 will cover the experimental implementation results and a thorough comparison and discussion. Lastly, Section 5 outlines the main ideas of the paper.

2. The Typical PV Array Configuration

The configuration depicted in Figure 1 showcases an efficient standalone PV array setup. This proposed system comprises PV modules, blocking diodes, bypass diodes, combiner boxes, a DC–DC converter, ground fault detector interrupters (GFDIs), and OCPDs, all working harmoniously. In Figure 1, each PV module is equipped with a bypass diode, which helps maintain PV performance. It comprises three parallel strings, each containing six modules connected in series. The efficient arrangement of the PV arrays through a combination of series and parallel connections ensures the achievement of the desired voltage and current levels (and, subsequently, the desired output power).

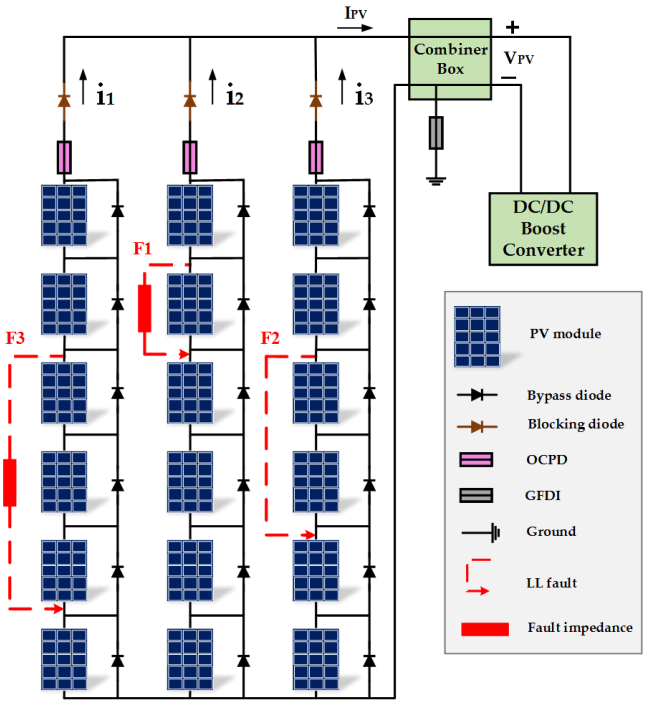


Figure 1. Structural and functional insights into a 3 × 6 PV array.

Comprehensive Analysis of Line-to-Line (LL) Faults in PV

An LL fault occurs when two points of different potential in an electric network or system are (usually accidentally) connected. In PV systems, an LL fault typically represents a short-circuit fault among PV modules or array cables with different potentials. Three main reasons can result in LL faults in PV arrays: (i) cable insulation breakdown; (ii) unintended short circuits between live conductors—for instance, a nail piercing exposed wiring; and (iii) LL faults occurring in the DC junction box due to physical damage, water penetration, or corrosion [25].

LL faults potential for significant impact on PV components cannot be underestimated. The fault current generated by LL faults can reach levels capable of damaging PV modules

and conductors, thereby posing a serious threat of fire hazards and compromising the overall efficiency of the PV system. This highlights the crucial significance of comprehending and promptly executing the required protective measures.

According to the National Electrical Code (NEC), having a single OCPD at every PV string is essential for overcurrent protection in a PV system [26]. The OCPD ensures the system's safety by always being on the fault path. The NEC provides clear guidelines for the maximum current and OCPD ratings in PV arrays. According to these guidelines, the maximum current for a specific circuit in PV arrays should be the sum of parallel module-rated short-circuit current (I_{SC}) multiplied by 125 percent. Additionally, the NEC mandates that overcurrent devices be rated at no less than 125 percent of the maximum current (1.25 I_{SC}). Therefore, by multiplying the two factors together, the resulting rating of OCPD should be no less than 1.56I_{SC} of PV modules. This calculation method ensures that the OCPD rating is sufficient to handle potential faults and maintain the system's safety and efficiency.

Figure 1 depicts various LL faults (F1, F2, and F3) in a PV array. They can occur at different mismatch levels, which can be expressed mathematically as Equation (1).

$$\text{Mismatch level} = \frac{N_{m_{faulty}}}{N_{m_{total}}} \quad (1)$$

where $N_{m_{faulty}}$ shows the number of modules in a string affected by a short-circuit fault and $N_{m_{total}}$ denotes the overall number of modules in a string. Mismatch level is typically displayed as a percentage.

Therefore, in Figure 1, F1 shows a 16.66% mismatch due to the involvement of only one out of six modules. Similarly, the mismatch percentages of F2 and F3 are 33.33% and 50%, respectively. Note that F1 and F3 are simulated with fault impedance, while F2 does not have fault impedance.

This study investigates how short circuit faults affect different quantities of modules across five distinct levels. These levels consist of one, two, three, four, and five modules. Each level is designed to demonstrate a specific mismatch percentage, which are 16.67%, 33.33%, 50%, 66.67%, and 83.33%, respectively. This study delves into fault impedance ranging from 0 to 25 Ω, with increments of 5 Ω.

Additionally, Figure 2 demonstrates the PV array I–V curve in normal conditions and various LL fault scenarios. It emphasizes that LL faults have a similar impact on the I–V curve under low mismatch levels and high impedance values compared to normal conditions, making it challenging to distinguish faulty conditions.

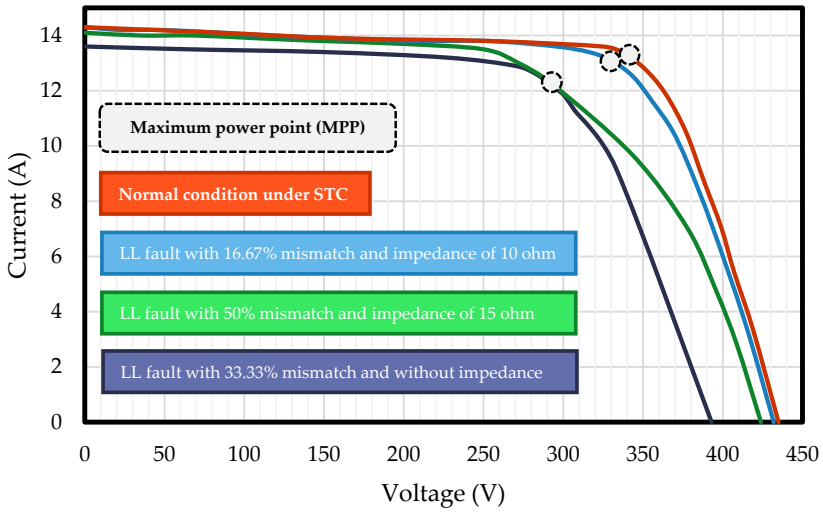


Figure 2. Investigating the role of impedance and mismatch percentages in determining LL fault severity in PV.

3. The Proposed Optimized Method of LVQs for Fault Detection

This study aims to propose a novel method for detecting and classifying faults in PV systems through an optimized ensemble learning model. The model uses different learning algorithms to evaluate system conditions and integrates PV array I–V curve measurements. The proposed model includes three different algorithms: learning vector quantization LVQ1, LVQ2, and LVQ3. The model uses the predictions of individual learning algorithms to make accurate decisions in the fault detection and classification process. The initial dataset uses five specific PV array I–V curve points under different normal and faulty scenarios. Subsequently, as illustrated in Figure 3, the feature extraction process is carried out using Canberra distance and chi-squared distance methods between each pair of the abovementioned points. After preprocessing (normalization), it is divided into two subsets: training and validation. Moving forward, a comprehensive evaluation is performed on three algorithms, and their accuracies are recorded. Then, to initialize the initial vectors of LVQ algorithms, a k-means clustering algorithm is utilized, and their optimal values are selected to start. Following this, the Adaptive Gradient (AdaGrad) optimizer is used to optimize the learning rate in LVQ algorithms. The best learning rate is calculated to obtain the highest accuracy in these algorithms. Subsequently, the predictions obtained from three LVQ algorithms enter an ensemble learning stage, and the outputs produced by the algorithms are merged using the majority voting method. Lastly, the final model is evaluated using an unseen dataset.

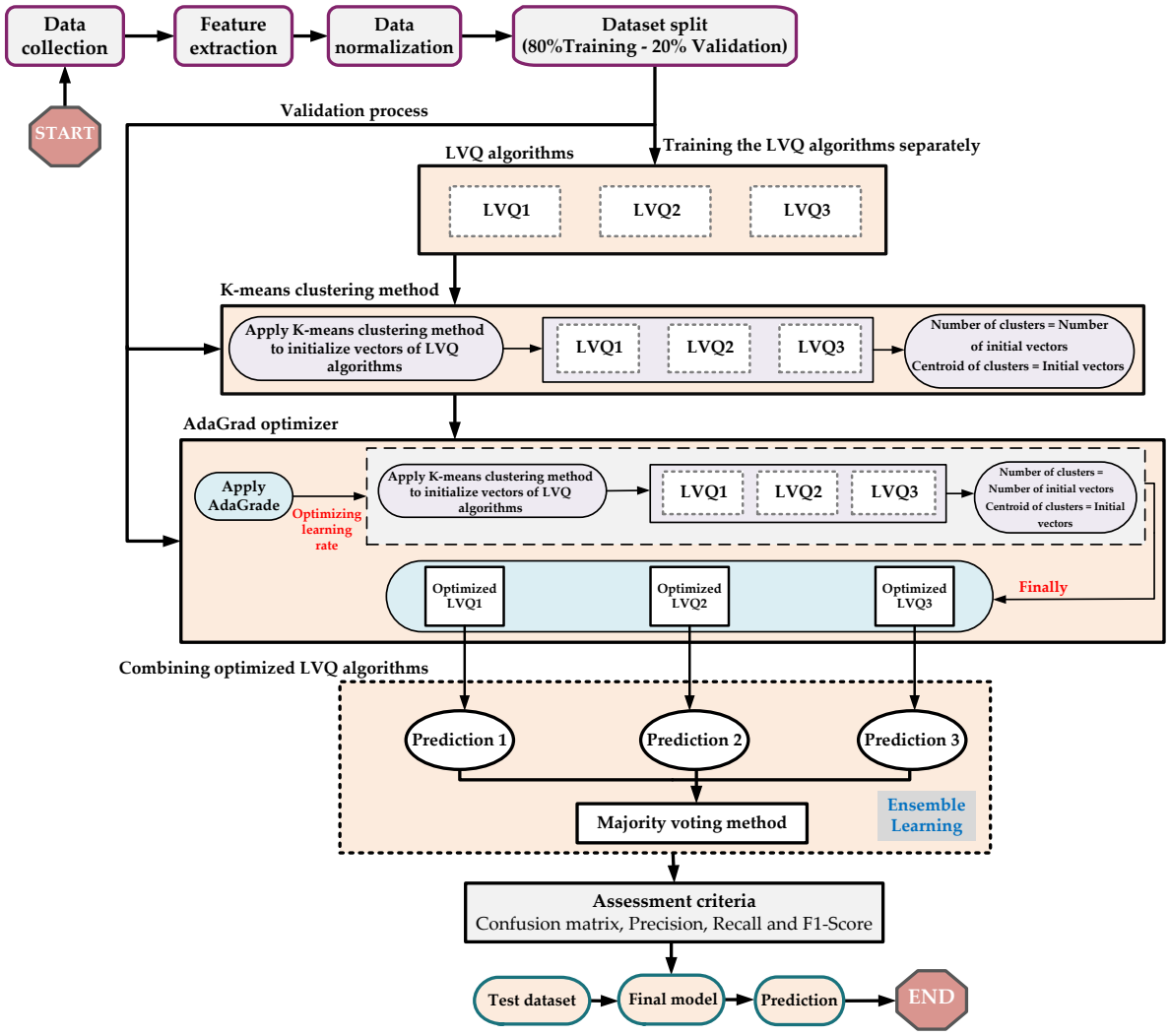


Figure 3. Graphical illustration of the proposed optimized method of LVQs for fault detection.

3.1. Initial Dataset Construction

The process of constructing a dataset for fault detection in PV arrays begins with PV array condition monitoring for fault characteristic identification. For this purpose, in this study, the PV array I–V curves are selected to be carefully examined under normal and faulty conditions. Then, five specific points are defined on the curves, including short-circuit current ($I_{PV} = I_{SC}$, $V_{PV} = 0$), open-circuit voltage ($I_{PV} = 0$, $V_{PV} = V_{OC}$), maximum power point ($I_{PV} = I_{MPP}$, $V_{PV} = V_{MPP}$), half of short-circuit current ($I_{PV} = 0.5I_{SC}$, $V_{PV} = V_{PV}(0.5I_{SC})$), and half of open-circuit voltage ($I_{PV} = I_{PV}(0.5V_{OC})$, $V_{PV} = 0.5V_{OC}$). Finally, the initial training and testing (unseen) datasets are separately created based on the five predefined points shown in Figure 4.

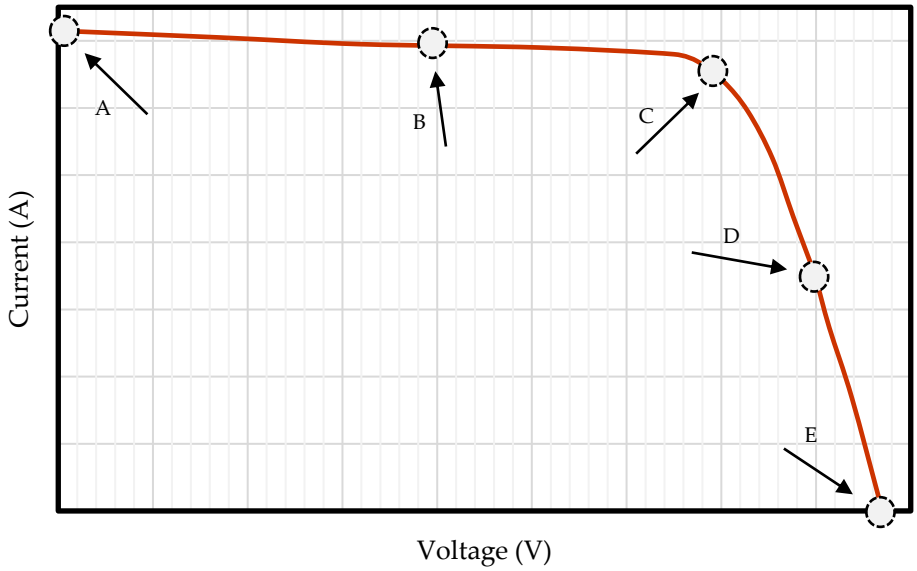


Figure 4. Specific points along the I–V curve of a PV array to construct the initial dataset.

3.2. Feature Extraction Phase

The methods used in this study for feature extraction are Canberra distance and chi-squared distance. The Canberra distance [27] is calculated by adding together the absolute differences between the corresponding elements in each vector, as illustrated in Equation (2).

$$\text{Canberra Distance} = \frac{|x_1 - x_2|}{|x_1| + |x_2|} + \frac{|y_1 - y_2|}{|y_1| + |y_2|} \tag{2}$$

On the other side, the chi-squared distance [28] can be used as a means of measuring the distance or similarity between two points, which is shown in Equation (3) for a two-dimensional space.

$$\text{Chi-squaredDistance} = \frac{(x_1 - x_2)^2}{x_1 + x_2} + \frac{(y_1 - y_2)^2}{y_1 + y_2} \tag{3}$$

In Equations (2) and (3), the pairs (x_1, y_1) and (x_2, y_2) show the respective coordinates of two points, as shown in Table 1.

Table 1. Extracted features from the I–V characteristic curves of PV arrays.

Extracted Features		
$f_1 = 1 + \frac{\lvert Isc - Ivoc/2 \rvert}{\lvert Isc \rvert + \lvert Ivoc/2 \rvert}$	$f_2 = 1 + \frac{\lvert Isc - Impp \rvert}{\lvert Isc \rvert + \lvert Impp \rvert}$	$f_3 = 1 + \frac{\lvert Isc - Isc/2 \rvert}{\lvert Isc \rvert + \lvert Isc/2 \rvert}$
$f_4 = \frac{\lvert -Voc \rvert}{\lvert Voc \rvert} + 1$	$f_5 = \frac{\lvert \frac{Voc}{2} - Vmpp \rvert}{\lvert \frac{Voc}{2} \rvert + \lvert Vmpp \rvert} + \frac{\lvert \frac{Ivoc}{2} - Impp \rvert}{\lvert \frac{Ivoc}{2} \rvert + \lvert Impp \rvert}$	$f_6 = \frac{\lvert \frac{Voc}{2} - Visc/2 \rvert}{\lvert \frac{Voc}{2} \rvert + \lvert Visc/2 \rvert} + \frac{\lvert \frac{Ivoc}{2} - Isc/2 \rvert}{\lvert \frac{Ivoc}{2} \rvert + \lvert Isc/2 \rvert}$
$f_7 = \frac{\lvert \frac{Voc}{2} - Voc \rvert}{\lvert \frac{Voc}{2} \rvert + \lvert Voc \rvert} + 1$	$f_8 = \frac{\lvert Vmpp - Visc/2 \rvert}{\lvert Vmpp \rvert + \lvert Visc/2 \rvert} + \frac{\lvert Impp - Isc/2 \rvert}{\lvert Impp \rvert + \lvert Isc/2 \rvert}$	$f_9 = \frac{\lvert Vmpp - Voc \rvert}{\lvert Vmpp \rvert + \lvert Voc \rvert} + 1$
$f_{10} = \frac{\lvert \frac{Visc}{2} - Voc \rvert}{\lvert \frac{Visc}{2} \rvert + \lvert Voc \rvert} + 1$	$f_{11} = \frac{Voc}{2} + \frac{(Isc - Ivoc/2)^2}{Isc + Ivoc/2}$	$f_{12} = Vmpp + \frac{(Isc - Impp)^2}{Isc + Impp}$
$f_{13} = Visc/2 + \frac{(Isc - Isc/2)^2}{Isc + Isc/2}$	$f_{14} = Voc + Isc$	$f_{15} = \frac{\left(\frac{Voc}{2} - Vmpp\right)^2}{\frac{Voc}{2} + Vmpp} + \frac{\left(\frac{Ivoc}{2} - Impp\right)^2}{\frac{Ivoc}{2} + Impp}$
$f_{16} = \frac{\left(\frac{Voc}{2} - Visc/2\right)^2}{\frac{Voc}{2} + Visc/2} + \frac{\left(\frac{Ivoc}{2} - Isc/2\right)^2}{\frac{Ivoc}{2} + Isc/2}$	$f_{17} = \frac{\left(\frac{Voc}{2} - Voc\right)^2}{\frac{Voc}{2} + Voc} + \frac{Ivoc}{2}$	$f_{18} = \frac{(Vmpp - Visc/2)^2}{Vmpp + Visc/2} + \frac{(Impp - Isc/2)^2}{Impp + Isc/2}$
$f_{19} = \frac{(Vmpp - Voc)^2}{Vmpp + Voc} + Impp$	$f_{20} = \frac{\left(\frac{Visc}{2} - Voc\right)^2}{\frac{Visc}{2} + Voc} + Isc/2$	

3.3. Z-Score Normalization

Normalizing an algorithm's inputs is essential for speeding up the network's learning process by standardizing input scales to ensure equalized scales within the same range of values for training and testing data. The models introduced in this study will employ either statistical or Z-score normalization, which entails normalizing each input vector X using the mean (μ) and standard deviation (σ) of the data. The distribution of the normalization process is expressed mathematically in Equations (4)–(6).

$$\mu_i = \frac{1}{n}\left(\sum_{i=1}^{n} X_i\right) \tag{4}$$

$$\sigma_i = \sqrt{\frac{\sum_{i=1}^{n} |X - \mu_i|^2}{n}} \tag{5}$$

$$X_i' = \frac{(X_i - \mu_i)}{\sigma_i} \tag{6}$$

The data dimension is denoted by n. It is essential to note that Z-score normalization transforms the data so that each feature has a mean of zero and a standard deviation of one.

3.4. Concept and Structure of LVQ Algorithms

LVQ, also known as learn vector quantization, is a type of neural network. It utilizes a prototype algorithm for supervised learning and classification. Similar to a self-organizing map, LVQ constructs its network using a competitive learning algorithm [29].

3.4.1. LVQ Algorithm Training Phase

Step #1: The LVQ training begins by initializing the weight vectors as shown in Figure 5. Initially, a subset of training vectors is chosen as weight vectors. The quantity of weight vectors is denoted as "m", and they are typically selected from the initial set of training vectors. The remaining vectors are set aside for the actual training process. Moreover, initial weights and classes are randomly assigned to the weight vectors.

***Step #2*:** The square of the Euclidean distance is calculated for every weight vector ($j = 1, \ldots, m$) and input vector element ($i = 1, \ldots, n$). As shown in Equation (7), calculating the Euclidean distance involves comparing the input vector components (x_i) with the corresponding weight vector components (w_{ij}). The algorithm's decision-making process is centered around the winning unit in j, which is the weight vector with the minimum

distance D(j). This winning unit is significant, as it represents the weight vector that is the closest match to the input vector, thereby influencing the subsequent steps of the algorithm.

$$D(j) = \sum_{i=1}^{n} \sum_{j=1}^{m} (x_i - w_{ij})^2 \quad (7)$$

***Step #3*:** The new weight of the winning unit is determined based on the relationship between the class of the input vector (T) and the class associated with the winning unit (Cj). When T matches with Cj, the weight vector gets updated by adding a portion (α) of the disparity between the input vector and the previous weight vector as defined in Equation (8). Conversely, if T does not equal Cj, the weight vector is updated by subtracting the same fraction (α) from the difference defined in Equation (9).

$$w_j(t+1) = w_j(t) + \alpha \times [x - w_j(t)] \quad (8)$$

$$w_j(t+1) = w_j(t) - \alpha \times [x - w_j(t)] \quad (9)$$

***Step #4*:** The LVQ training algorithm demonstrates its adaptability by adjusting the learning rate (α) after updating the weights. This learning rate, which determines the extent of weight adjustment, gradually decreases during the training process. This adaptive feature is crucial, as it allows the algorithm to effectively handle different data sets and training scenarios, enhancing its overall performance, as defined in Equation (10).

$$\alpha(t+1) = 0.5 \times \alpha(t) \quad (10)$$

***Step #5*:** The stopping condition is checked to see if it has been met. This condition can be defined as reaching the maximum number of iterations or reducing the learning rate to a negligible amount. If the stop condition is met, the training process concludes. If not, it returns to Step 2 to continue in another iteration.

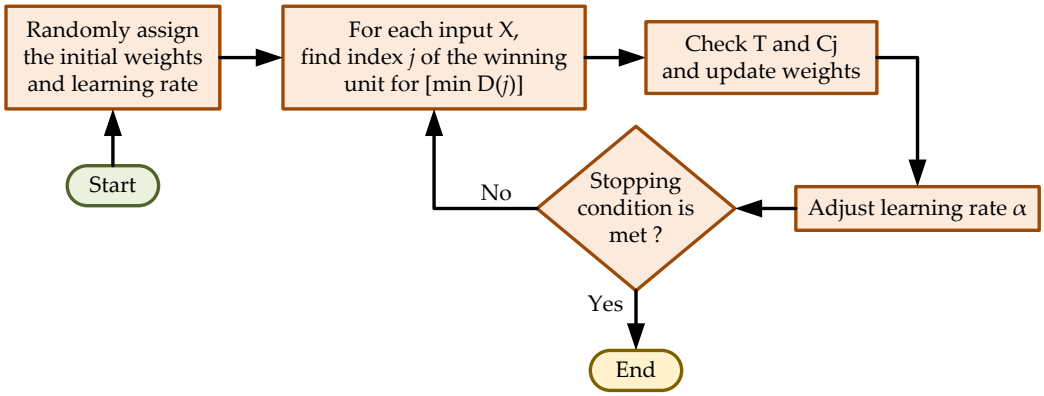


Figure 5. LVQ neural network.

The LVQ training algorithm follows a meticulous and iterative process. It begins by initializing the weight vectors and then proceeds to calculate the Euclidean distances, update the weight vectors based on the winner unit, adjust the learning rate, and examine the stop condition. This iterative nature is crucial, as it ensures that the algorithm is fine-tuned and optimized with each iteration, leading to a fully trained LVQ model that is ready for classification tasks.

3.4.2. Concept of LVQ2 Algorithm

Building upon the LVQ training process we have previously explored, the LVQ2 variant introduces a novel window condition. This condition is defined by a set of parameters, including the current input vector (x), the closest weight vector to x (w_c), the next closest

weight vector to x (w_r), the distance from x to w_c (S_c), and the distance from x to w_r (S_r). Equation (11) is then used to determine if the input vector x fits within this window, marking a significant advancement in the LVQ algorithm.

$$\frac{S_c}{S_r} > 1 - \theta \;\&\; \frac{S_r}{S_c} > 1 + \theta \tag{11}$$

Here, θ represents a threshold value, usually determined based on the number of training samples. The update of weight vectors in LVQ2 can be done using Equation (12) and Equation (13).

If the input vector x is in a class different from w_c:

$$w_c(t+1) = w_c(t) + \alpha \times [x - w_c(t)] \tag{12}$$

If the input vector x is in the same class as w_r:

$$w_r(t+1) = w_r(t) + \alpha \times [x - w_r(t)] \tag{13}$$

In Equations (12) and (13), α denotes the learning rate. The learning rate may change during the training phase and typically decreases gradually. The LVQ2 variant includes window conditions to adjust the update of the weight vectors based on the relative distances between the input and weight vectors. By taking into account the intervals and window conditions, the LVQ2 algorithm aims to enhance the accuracy of the training phase and refine the model's recognition capability.

3.4.3. Concept of LVQ3 Algorithm

LVQ3 is another variant of LVQ family that introduces a different approach to updating weight vectors. In LVQ3, two weight vectors, w_c and w_r, are considered for the window condition. which is defined by Equation (14).

$$\text{Min}[\frac{S_c}{S_r}, \frac{S_r}{S_c}] > (1 - \theta)(1 + \theta) \tag{14}$$

Here, θ is a constant and is usually set to 0.2. To update the weight vectors in LVQ3, Equations (14) and (15) are used:

If the input vector x is from a class different than w_c:

$$w_c(t+1) = w_c(t) + \beta \times [x - w_c(t)] \tag{15}$$

If vector x is in the same class as w_r:

$$w_r(t+1) = w_r(t) + \beta \times [x - w_r(t)] \tag{16}$$

In Equations (15) and (16), β is defined as a multiple of the learning rate (α), represented as $\beta = m \times \alpha$, where $0.1 < m < 0.5$. The learning rate α determines the extent of weight adjustment and can be modified during training.

The LVQ3 variation utilizes the two closest weight vectors to establish a window condition that aids in guiding the update of the weight vectors. By taking into account the relative distances between the input vector and two adjacent weight vectors, LVQ3 seeks to enhance the model's discriminatory capability and classification performance.

3.5. Vectors Initialization Using K-Means Clustering

The k-means clustering algorithm does not assign values directly to the weight vectors of the LVQ algorithms. Nevertheless, the result of the K-means clustering algorithm can be utilized as a starting point for initializing the weight vectors in the LVQ algorithms [30]. Applying K-means clustering to initializing weight vectors of the LVQ algorithms begins with randomly initializing k centroids C_1, C_2, C_3, …, C_k. Next, each data point x_i is

assigned to the nearest centroid based on the Euclidean distance defined in Equation (17). Then, each centroid C_j is updated to be the mean of all points assigned to cluster j based on Equation (18), where S_j is the set of points assigned to cluster j. These steps are repeated until the centroids no longer change or change very little. Following this, the majority label of the data points assigned to cluster j is determined by C_j based on Equation (19), where y_i represents the label of data point x_i. The process then moves on to initializing the LVQ weight vectors with the centroids C_j and their corresponding labels, marking the beginning of the learning process. During the learning process of LVQ, the weight vector and their labels are utilized. For each data point x_i with label y_i, the closest weight vector is first identified based on Equation (7). Then, the weight vector is updated according to whether the label matches, as defined in Equations (8) and (9). These steps are repeated for a predefined number of epochs or until the weight vectors converge. The cluster centroids acquired from the K-means clustering algorithm are employed as initial vectors for the LVQ algorithms. Each cluster centroid can be viewed as a representative vector for the data samples in that specific cluster.

$$\text{Cluster } (x_i) = \arg\min_j \left\| x_i - C_j \right\| \tag{17}$$

$$C_j = \frac{1}{|S_j|} \sum_{x_i \in S_j} x_i \tag{18}$$

$$\text{Label } (C_j) = \text{Mode}\left(\left\{ y_i : x_i \in S_j \right\}\right) \tag{19}$$

It is important to mention that the optimal number of clusters is determined using the elbow method, calculating the within-cluster-sum-square (WCSS) metric as defined in Equation (20).

$$\text{WCSS} = \sum_{i \in n} (X_i - C_i)^2 \tag{20}$$

For each value of the cluster, WCSS is calculated. WCSS is the sum of the squared distance between each point (X_i) and the centroid in a cluster (C_i).

3.6. Adaptive Gradient Optimizer (AdaGrad)

AdaGrad, also known as Adaptive Gradient, represents a variation of the gradient descent algorithm designed to adjust the learning rate for each parameter separately. Its fundamental concept involves reducing the learning rate for parameters with a high gradient and increasing it for those with a low gradient. This approach enables the algorithm to achieve quicker convergence when dealing with sparse data [31].

It is crucial to establish an optimal learning rate (α) in Equations (8), (9), (12), (13), (15) and (16) during the training process of LVQs after initializing the weight vectors. The implementation steps of AdaGrad can be added as follows: In Equation (21), G(t) is a matrix with diagonal elements where each element on the diagonal is the sum of the squares of the preceding gradients. At the same time, G(t + 1) is a diagonal matrix in which each element on the diagonal is the sum of the squares of the gradients after they have been updated. The gradient of the parameter is denoted as the gradient for the winning vector. Equation (22) adjusts the learning rate based on the history of the gradients. As the gradient accumulates, the learning rate decreases, allowing for more refined updates to the weights.

$$G(t+1) = G(t) + \text{gradient}^2 \tag{21}$$

$$\alpha(t+1) = \frac{\alpha(t)}{\sqrt{G(t+1)}} \tag{22}$$

$$w_j(t+1) = w_j(t) - \alpha(t+1) \times \text{gradient} \tag{23}$$

In Equation (23) the updated weight value in LVQ algorithms is denoted as $w_j(t+1)$, while the previous weight in LVQ algorithms is denoted as $w_j(t)$. The α refers to the learning rate of the AdaGrad optimizer, which is the learning rate of LVQ as well. These equations collectively describe how AdaGrad adapts the learning rate based on the accumulated gradients and updates the weight vectors accordingly. By adjusting the learning rate adaptively, AdaGrad aims to improve the efficiency and effectiveness of the optimization process.

3.7. Ensemble Learning Model

Utilizing the ensemble learning (EL) technique in fault classification provides numerous benefits, such as reducing the requirement for an extensive training dataset and reducing the probability of overfitting problem, often observed in methods that use only a single classifier. In this study, first, the three algorithms, namely LVQ1, LVQ2, and LVQ3 are optimized. Then, in the EL mode, every individual algorithm produces its prediction. The ultimate prediction is then decided by integrating the outcomes of each classifier.

A commonly used method to combine the outcomes generated by individual learning algorithms is through a voting-based combination approach. This involves employing a majority voting scheme, where the highest number of votes determines the predicted classes. Equation (24) determines the collection model's prediction output, which is based on votes.

$$\hat{y} = \underset{j}{\operatorname{argmax}} \sum_{i=1}^{n} p_{ji}(x) \tag{24}$$

The predicted output of the ensemble learning model is represented by $\hat{y}$. The number of class labels is denoted by j. In this paper, the number of learning algorithms is equal to three, which is represented by n. P_{ji} represents the output prediction of the ith learning algorithm.

3.8. Performance Evaluation

The study assessed the performance of the proposed model by considering various criteria and measures.

3.8.1. Confusion Matrix

The structured tabulation of predicted and actual classifications is known as a confusion matrix [32]. The dimensions of a confusion matrix are contingent upon the size of the readout layer—precisely, the number of labels, denoted by m rows and m columns, where m represents the unique label values. An example of a 2×2 confusion matrix is depicted in Figure 6.

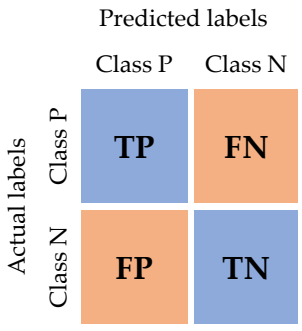


Figure 6. Example of a typical confusion matrix.

Within this matrix, the diagonal values signify accurately predicted classes, while the off-diagonal values denote inaccuracies in predictions. The initiations in Figure 6 are defined as:

- TP: Correctly predicted samples were labeled "Positive" and were actually "Positive";
- TN: Correctly predicted samples were labeled "Negative" and were actually "Negative";
- FP: Incorrectly predicted samples were labeled "Positive" but were actually "Negative";

- FN: Incorrectly predicted samples were labeled "Negative" but were actually "Positive"

The principal metrics derived from the confusion matrix include (1) accuracy, (2) precision, (3) recall, and (4) F1-score, as defined in Equations (25)–(28).

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{25}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{26}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{27}$$

$$\text{F1-score} = \frac{2 \times (\text{Precision} \times \text{Recall})}{\text{Precision} + \text{Recall}} \tag{28}$$

3.8.2. Loss Function

This specific loss function calculates the mean of the squared variances between the predicted and actual values. As mentioned in Equation (29), the variable $(y_i - y_{pred,i})$ denotes the difference for each data point, where y_i is the actual value and $y_{pred,i}$ shows the predicted value. By squaring this difference, the function ensures that it penalizes more considerable differences while treating positive and negative deviations equally. The normalization term $\frac{1}{n}$ ensures that the loss represents the average difference per data point (n shows the number of data samples), making it independent of the dataset size.

$$\text{Loss function} = \frac{1}{n} \sum_{i=1}^{n} \left(y_i - y_{pred,i} \right)^2 \tag{29}$$

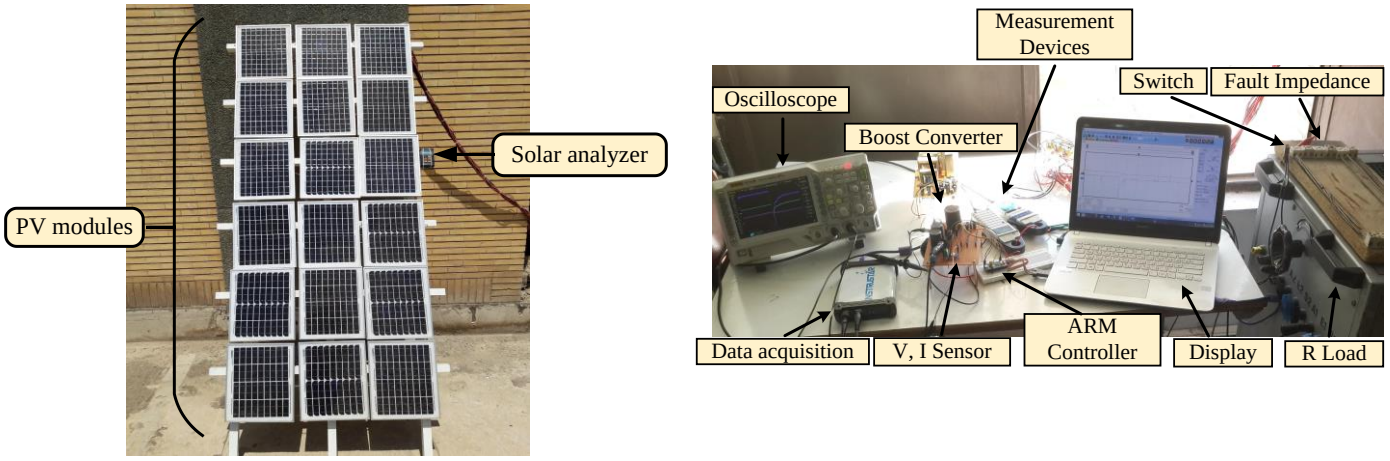
4. Comprehensive Analysis, Results and Discussion

4.1. Experimental System Description

An independent experimental PV system has been created to assess the suggested model. This system comprises a 3 × 6 PV array with three parallel strings, each containing six series of Yingli YL010D-18b PV modules, manufactured by PURE Solar, located in Huishan District, Wuxi City, Jiangsu, China. The specifications of the PV modules and the DC–DC boost converter, outlined in Table 2, are based on the design of the DC–DC boost converter shown in Figure 7.

Table 2. Technical specifications of the PV system.

	Parameter	Value
Yingli YL010D-18b PV modules	I_{SC}	0.61 A
	V_{OC}	22.5 V
	I_{MPP}	0.56 A
	V_{MPP}	18 V
	P_{MPP}	10 W
DC–DC boost converter	Inductor	2 mH
	Capacitor	390 µF, 400 V
	Switching frequency	40 kHz



(**a**) (**b**)

Figure 7. PV experimental setup: (**a**) PV array; (**b**) integral components such as the DC–DC boost converter and supplementary equipment.

4.2. The Process of Acquiring Data

The data collection for this study involved capturing samples from the I–V curves of a PV array at five specific points. These samples were taken under different environmental conditions (such as varying irradiance and temperature) and under both normal and faulty operating conditions. An I–V curve tracer was created to collect the data based on the design depicted in Figure 8. This setup included current and voltage sensors, a DC–DC converter, an ARM microcontroller responsible for controlling the switch, and a gate driver circuit used to obtain the I–V curves of the PV array. As shown in Figure 8, the ARM microcontroller adjusts the duty cycle to extract the features from the I–V curve. The testing algorithm for the I–V curve utilizes the STM32f103C8T6 ARM, manufactured by STMicroelectronics, Geneva, Switzerland. When conducting the testing, the controller reads voltage and current values. The tracking method is simple: first, the PV array applies a load with a low resistance to produce a high current with a low-duty cycle (around 10%). This causes the PV array to generate a low current, nearly the same as the PV array's short circuit current. This process is repeated with gradual increases in the duty cycle until it reaches a specific range (over 80%). The OC voltage, also known as the open-circuit voltage, is established by gauging the PV array's output voltage when it is in an open-circuit condition.

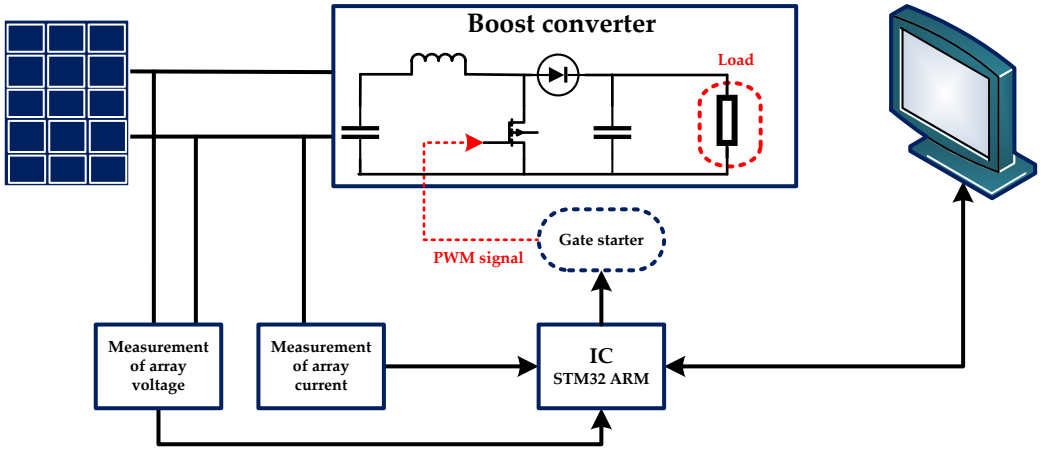


Figure 8. Data acquisition setup for generating the I–V curves of the PV array.

In the process of collecting experimental data, 1183 samples were gathered under various conditions, including no-fault and LL faults. The original dataset comprises

802 samples for training, 201 for validation, and 180 for testing the final model, as elaborated in Table 3.

Table 3. The data acquired from the experimental procedure.

Class Label	Training	Validation	Test
Normal condition	346	87	60
LL (one-module)	91	23	24
LL (two-module)	91	23	24
LL (more-than-two-module)	274	68	72
Total sample	802	201	180

Upon acquisition of the original dataset, a series of distinct features are derived in accordance with the specifications outlined in Table 1. Subsequently, the data points undergo standardization utilizing the Z-score normalization technique defined by Equation (5).

4.3. Experimental Results

During the initial step, the individual algorithms (LVQ1, LVQ2, and LVQ3) are fully trained and further validated using the training and validation datasets, respectively. After applying the validation dataset to three models, they reached the same validation accuracies of 43.28%.

After the first step, LVQs are optimized using the K-means clustering technique, which sets the initial vectors for LVQ algorithms. The results of LVQs at this step can be found in Table 4. It should be noted that for LVQ1, LVQ2, and LVQ3 algorithms, three clusters were selected respectively by the elbow method as the best number of clusters to obtain optimal results, as shown in Figure 9. The elbow method (see Equation (20)) aims to identify a point where the reduction rate shifts significantly and becomes less steep. This point typically resembles an elbow in the graph that determines the best number of clusters.

Table 4. Results of applying K-means on LVQs in training and validation processes.

Algorithm	Training Accuracy (%)	Validation Accuracy (%)
LVQ1	80	77.61
LVQ2	75.56	75.62
LVQ3	81.29	78.1

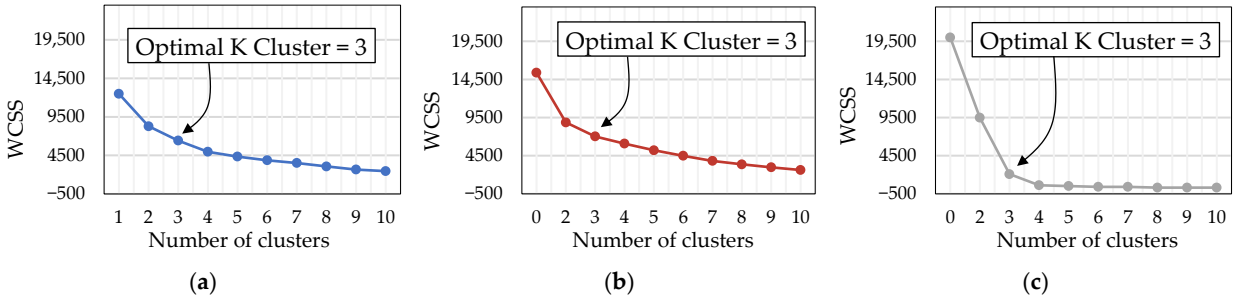


Figure 9. Determining optimal cluster number through WCSS analysis: (**a**) LVQ1; (**b**) LVQ2; (**c**) LVQ3.

Next, the AdaGrad method is utilized to determine the most optimal learning rate. Figure 10 illustrates the training loss of LVQs within the range from 0 to 0.02 of the learning rate, with the lowest learning rates being LVQ1 0.01, LVQ2 0.02, and LVQ3 0.02, respectively. The AdaGrad optimizer has identified the most optimal learning rates.

Based on the optimal learning rates determined by the AdaGrad optimizer, Figure 11 displays the training and validation learning curves as well as the training and validation

loss curves for the LVQ algorithm and LVQ2 algorithm across 400 epochs and for the LVQ3 algorithm across 100 epochs. As shown in Figure 11, the AdaGrad optimizer has exhibited excellent performance in the final epoch of each algorithm, resulting in the LVQ1 algorithm achieving 87% learning accuracy and 85.57% validation accuracy. Similarly, the LVQ2 and LVQ3 algorithms achieved 86.65% and 83.91% training accuracy, respectively, and 85.57% and 83.07% validation accuracy, respectively, representing significant improvements over the values presented in Table 4. This indicates an enhancement in the efficiency of the algorithms.

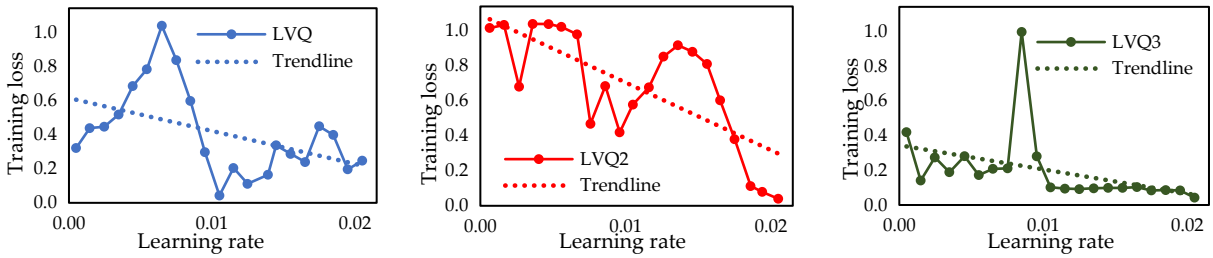


Figure 10. The trend of changes in learning rate to select the most optimum learning rate using AdaGrad: (**a**) LVQ1; (**b**) LVQ2; (**c**) LVQ3.

Figure 12 illustrates the trend of the gradient norm and its evolution over epochs during the training process, demonstrating the convergence of the training. A decreasing gradient norm signifies the model's progression towards the optimal solution. With continued training, diminishing gradients indicate that the model parameters are finely tuned to reduce loss.

In the final step, to improve the accuracies, the combination of three individual algorithms (LVQ1, LVQ2, and LVQ3) has been used in an ensemble learning model. It should be noted that the combination technique for integrating the result of LVQs was majority voting technique (see Equation (24)). Comparing the training and validation accuracies of the proposed ensemble learning model with three other ensemble-based algorithms, Table 5 demonstrates that the LVQ-based ensemble learning model outperforms the others, achieving a maximum accuracy of 100% in both the training and validation processes.

Table 5. Outcome of model development in training and validation processes.

Learning Method	Training Accuracy	Validation Accuracy
Boosting AdaBoost	84.41%	82%
Bagging random forest	99.87%	98.5%
Gradient boosting	100%	98.5%
The proposed method	100%	100%

Furthermore, Figure 13 shows the validation confusion matrices, providing a comprehensive summary of the validation results. It demonstrates the accuracy of the final model during validation by detailing the number of correct and incorrect classifications using different learning methods. Furthermore, Table 6 provides information on three additional evaluation metrics: precision, recall, and F1 score, which can be calculated from each confusion matrix using Equations (26)–(28).

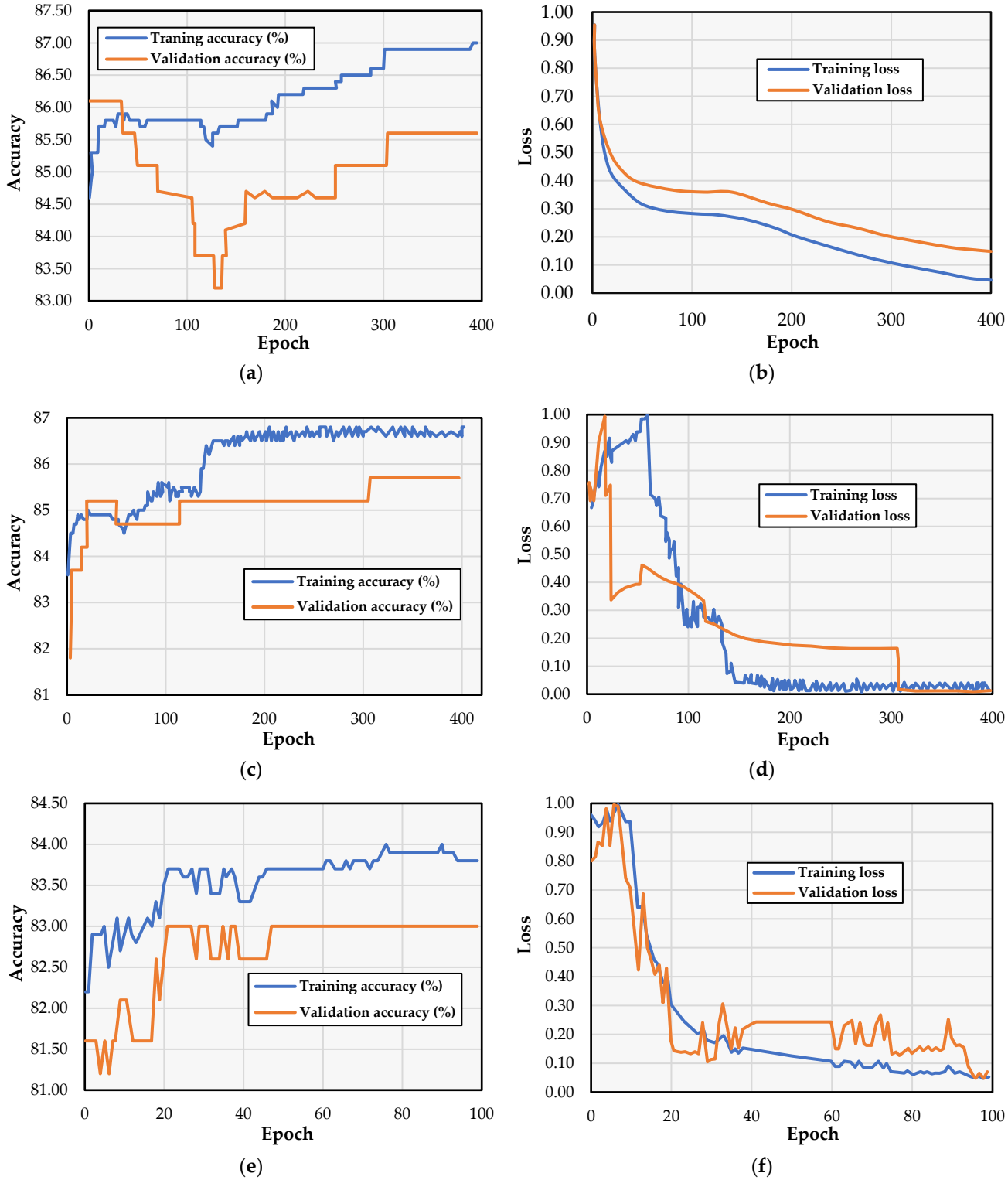


Figure 11. The training and validation performances for LVQs using optimal learning rates: (**a**) LVQ1 accuracy; (**b**) LVQ1 loss; (**c**) LVQ2 accuracy; (**d**) LVQ2 loss; (**e**) LVQ3 accuracy; (**f**) LVQ3 loss.

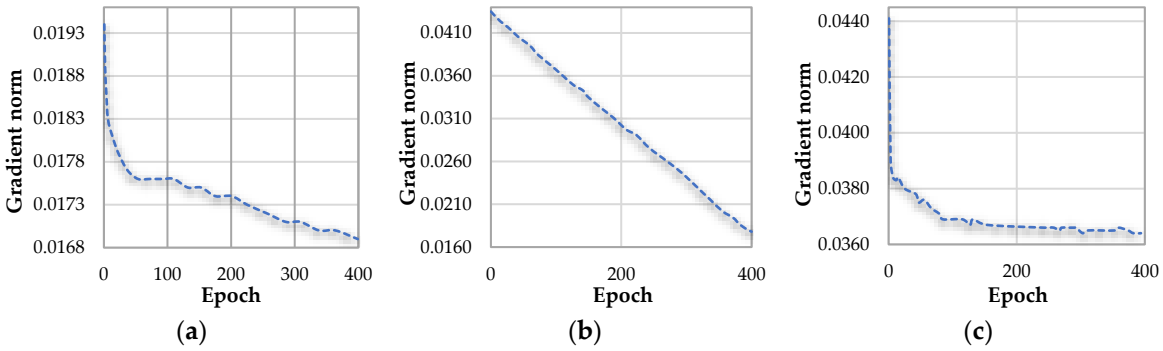


Figure 12. Evolution of gradient norm trends across epochs during training: (**a**) LVQ1; (**b**) LVQ2; (**c**) LVQ3.

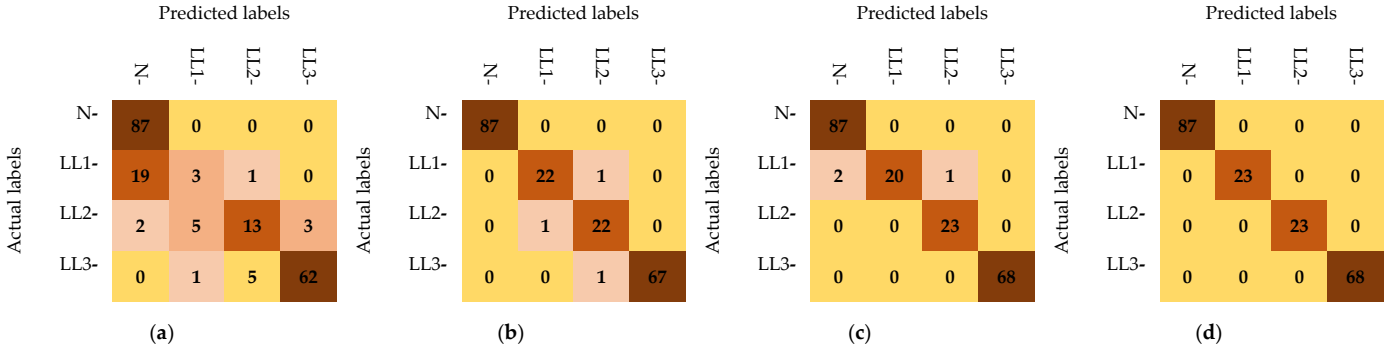


Figure 13. The confusion matrix of learning algorithms during validation process: (**a**) boosting AdaBoost; (**b**) bagging random forest; (**c**) gradient boosting; (**d**) the proposed method. Note: Confusion matrices use a heatmap color scheme where lighter colors represent low values, and darker colors represent high values. This makes it easy to visually assess the matrix at a glance.

Table 6. Various classification metrics in validation process.

Learning Method	Precision	Recall	F1-Score
Boosting AdaBoost	P_1 = 80.55% P_2 = 33.33% P_3 = 68.42% P_4 = 95.38%	R_1 = 100% R_2 = 13.07% R_3 = 56.62% R_4 = 91.17%	F_1 = 89.23% F_2 = 18.75% F_3 = 61.9% F_4 = 93.23%
Bagging random forest	P_1 = 100% P_2 = 95.65% P_3 = 91.66% P_4 = 100%	R_1 = 100% R_2 = 95.65% R_3 = 95.65% R_4 = 98.52%	F_1 = 100% F_2 = 95.65% F_3 = 93.61% F_4 = 99.25%
Gradient boosting	P_1 = 97.75% P_2 = 100% P_3 = 95.83% P_4 = 100%	R_1 = 100% R_2 = 86.95% R_3 = 100% R_4 = 100%	F_1 = 98.86% F_2 = 93% F_3 = 97.87% F_4 = 100%
The proposed method	P_1 = 100% P_2 = 100% P_3 = 100% P_4 = 100%	R_1 = 100% R_2 = 100% R_3 = 100% R_4 = 100%	F_1 = 100% F_2 = 100% F_3 = 100% F_4 = 100%

After validating the final model, the next step is to use test data samples that have not been seen before to assess the model's performance further. As shown in Figure 14, which contains the comprehensive confusion matrices of each learning method, the ultimate model (including the proposed method) demonstrates the highest performance with an accuracy of 97.77% during testing. Additional assessment metrics can be found in Table 7

to understand the outcome better. It should be noted that the average testing time for fault detection and classification in the proposed method is approximately 20 ms.

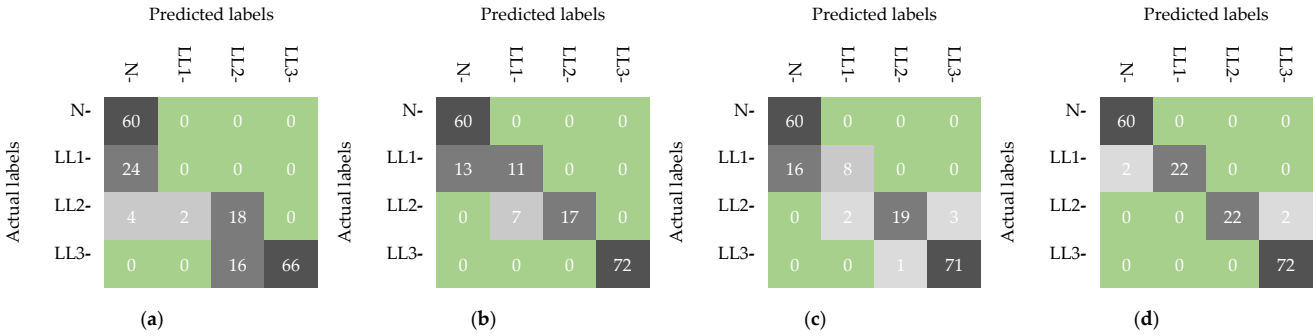


Figure 14. The confusion matrix of learning algorithms during testing process: (**a**) boosting AdaBoost; (**b**) bagging random forest; (**c**) gradient boosting; (**d**) the proposed method. Note: Confusion matrices use a heatmap color scheme where lighter colors represent low values, and darker colors represent high values. This makes it easy to visually assess the matrix at a glance.

Table 7. Various classification metrics in testing process.

Learning Algorithm	Testing Accuracy	Precision	Recall	F1-Score
Boosting AdaBoost	80%	P_1 = 68.18% P_2 = 0% P_3 = 75% P_4 = 100%	R_1 = 100% R_2 = 0% R_3 = 75% R_4 = 91.66%	F_1 = 81.08% F_2 = 0% F_3 = 75% F_4 = 95.65%
Bagging random forest	88.88%	P_1 = 82.19% P_2 = 61.11% P_3 = 100% P_4 = 100%	R_1 = 100% R_2 = 45.83% R_3 = 70.83% R_4 = 100%	F_1 = 90.22% F_2 = 52.38% F_3 = 82.92% F_4 = 100%
Gradient boosting	87.77%	P_1 = 78.94% P_2 = 80% P_3 = 95% P_4 = 95.94%	R_1 = 100% R_2 = 33.33% R_3 = 79.16% R_4 = 98.61%	F_1 = 88.23% F_2 = 47.05% F_3 = 86.36% F_4 = 97.27%
The proposed method	97.77%	P_1 = 96.77% P_2 = 100% P_3 = 100% P_4 = 97.29%	R_1 = 100% R_2 = 91.66% R_3 = 91.66% R_4 = 100%	F_1 = 98.36% F_2 = 95.65% F_3 = 95.65% F_4 = 98.63%

4.4. Discussion and Comparison

The efficiency of the proposed LVQ model is established by comparing it with several recent research studies in the area of PV fault detection, as shown in Table 8. To ensure a fair comparison, various criteria—including publication dates (recency), use of contemporary methods (novelty), and different comparison dimensions, such as the ability to assess severity (extra capability), data sample requirements (simplicity), consideration of critical fault detection conditions (comprehensiveness), and performance on LL fault data samples detection (accuracy)—are considered. In Table 8, LM = low mismatch levels, and HI = high fault impedance values. The analysis based on Table 8 yields the following insights:

- The model developed in [21] requires a very small number of training data samples. However, it is not able (i) to detect LL faults under critical detection conditions, (ii) to classify different LL faults, and (iii) to assess the severity of LL faults. In addition, the accuracy of the model in LL faults detection process is not reliable.
- The model in [22] can positively produce the maximum test accuracy of 100% in the test stage using a small number of training data samples. However, similarly to [21], it

is also not able (i) to detect LL faults under critical fault detection conditions, (ii) to classify various kinds of LL faults, or (iii) to provide a severity assessment of LL faults.

- In [23], the model positively considers LL faults critical detection conditions, i.e., low mismatch levels and high fault impedance values. It also requires a relatively small number of data samples. However, as only a single-module LL fault is presented in the study, the model is not able (i) to classify different LL faults or (ii) to assess the severity of LL faults. Moreover, the implied accuracy is not sufficient to provide a reliable model in real-world conditions.
- The model developed in [24] solved the problem of accuracy and yields an outstanding accuracy in LL fault detection using a small number of data samples. Another advantage is that it is also able to classify two types of LL faults. However, the model (i) does not consider high fault impedance values, which crucially affects the difficulty of LL fault detection, and (ii) as the classification is carried out based on inter- and intra-string LL faults, it is not able to provide a severity assessment. Moreover, the implied accuracy is not sufficient to ensure a reliable model in real-world conditions.
- Finally, the proposed LVQ method has attempted to address all the issues identified in previously mentioned references by:
    1. Considering high values of fault impedance and low mismatch levels to provide various critical LL fault detection conditions and provide a comprehensive model;
    2. Being able to classify different types of LL faults in PV arrays based on the number of modules involved in the fault occurrence (the mismatch level);
    3. Being able to assess the severity of LL faults in PV arrays by providing a severity-based classification process;
    4. Requiring a small dataset to develop a fully trained model able to detect and classify various LL faults;
    5. Producing an outstanding accuracy in LL fault detection and classification which proves the model reliability.

Table 8. Comparison of the proposed model to various ensemble models in existing literature.

Ref.	Method	LL Faults Critical Detection Conditions	LL Fault Classification	LL Fault Severity Assessment	LL Fault Training Data Samples	LL Fault Detection Accuracy	Further Explanation
[21]	A stacking ensemble model containing histogram-based gradient boosting, AdaBoost, extra trees, and gradient booting	-	-	-	N/A	83.33%	- The number of LL fault samples in the training stage is not reported by the paper. However, based on the total number of training samples (460), it seems to be very small. - The accuracy of 83.33% is the best "precision" of LL faults detection in the test stage in an ensemble model reported by the paper.
[22]	An ensemble model containing LR, DT, and SVM	-	-	-	900	100%	- It is reported that 2700 data samples are collected and uniformly distributed among three classes. Therefore, the number of LL fault samples is estimated to be approximately 900.

Table 8. *Cont.*

Ref.	Method	LL Faults Critical Detection Conditions	LL Fault Classification	LL Fault Severity Assessment	LL Fault Training Data Samples	LL Fault Detection Accuracy	Further Explanation
[23]	An ensemble model containing DT, kNN, and SVM	LM + HI	-	-	1500 + 1115	90.9%	- The exact number of LL faults training data samples is not reported by the paper. However, a total number of 9165 labeled training data samples are employed which can provide an approximately 1500 LL faults samples with a uniform distribution assumption. Additionally, 1115 unlabeled LL fault samples were used by the model. - The experimental test accuracy of LL fault detection is not reported by the paper. The 90.9% accuracy refers to how accurately the model can label the unlabeled LL fault training data samples in a semi-supervised approach.
[24]	A stacking ensemble model containing RF, extreme gradient boosting (XGBoost), light gradient boosting machine (LightGBM), extra trees (EXT), and DT	LM	✓	-	2100	99.8%	- The LL faults are only classified into inter- and intra-string LL faults. - The number of LL faults samples in the training dataset is not reported by the paper. However, based on a distribution figure in the paper, this seems to be approximately 2100 samples. - The mentioned accuracy refers to the LL fault average "precision" in the test stage.
This study	A hybrid k-means clustering and ensemble model containing three LVQ neural networks	LM + HI	✓	✓	456	99.1%	- The number of 456 data samples is only related to LL faults in the training stage. - The 99.1% accuracy is the average precision of LL fault detection in the test stage.

Note: The "✓" indicates the presence of comparative parameters in the relevant reference and the term "N/A" stands for "Not Applicable", meaning there is no information about it.

5. Conclusions

PV installations are increasingly growing throughout the world since solar energy is clean and easy to access. However, PV components are always prone to various faults and failures as they operate outdoors. LL faults are one of the most important and frequent anomalies PV systems may encounter which pose a significant risk and can have severe consequences, such as fire hazards, if not detected and cleared in time. This study introduced an ensemble learning model based on three optimized LVQ neural networks to detect and classify several kinds of LL faults in PV systems.

The proposed method considers high values of fault impedance and low mismatch levels to provide a wide variety of critical conditions in LL faults detection and classification. It is able to classify different types of LL faults in PV arrays based on different mismatch levels and to assess the severity of LL faults by providing a severity-based classification process. The proposed model requires only a small training dataset yet produces outstanding accuracy for LL fault detection and classification, which shows the model's reliability.

In brief, the model analyzes PV array I–V curves during normal and faulty events and uses Canberra and chi-squared distance techniques to extract multiple valuable features. To improve the model accuracy, a K-means clustering algorithm is used to initialize the LVQs

vectors where finally three clusters were selected for each LVQ by the elbow method as the best number of clusters to obtain optimal results. Additionally, the AdaGrad optimizer is applied to optimize the learning rate of LVQs and the lowest learning rates were 0.01, 0.02, and 0.02 for LVQ1, LVQ2, and LVQ3, respectively. The evolution of the gradient norm over epochs showed the convergence during the training process, where diminishing gradients indicate that the model parameters are finely tuned to reduce the loss. Finally, the optimized LVQ algorithms are merged into an ensemble learning algorithm using the majority voting technique. The model experimental implementation showed maximum training and validation accuracies of 100% and a testing accuracy of 97.77%.

Author Contributions: Conceptualization, P.G., A.E. and M.A.; methodology, P.G., A.E., A.N. and M.A.; software, P.G., A.E. and A.N.; validation, A.E. and M.A.; formal analysis, P.G., A.E., A.N. and M.A.; investigation, P.G., A.E., A.N. and M.A.; resources, A.E., P.P., M.H. and M.A.; data curation, P.G., A.E., A.N. and M.A.; writing—original draft preparation, P.G. and A.N.; writing—review and editing, A.E., P.P., M.H. and M.A.; visualization, P.G. and A.N.; supervision, A.E., P.P., M.H. and M.A.; project administration, A.E., M.H., P.P. and M.A.; funding acquisition, M.A. and P.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The datasets presented in this article are not available due to confidentiality.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Aghaei, M.; Nedaei, A.; Eskandari, A.; Milimonfared, J. Introductory Chapter: Solar Photovoltaic Energy. In *Solar Radiation—Measurements, Modeling and Forecasting for Photovoltaic Solar Energy Applications*; IntechOpen: London, UK, 2022. [CrossRef]
2. Hajji, M.; Yahyaoui, Z.; Mansouri, M.; Nounou, H.; Nounou, M. Fault detection and diagnosis in grid-connected PV systems under irradiance variations. *Energy Rep.* **2023**, *9*, 4005–4017. [CrossRef]
3. Al Mahdi, H.; Leahy, P.G.; Alghoul, M.; Morrison, A.P. A Review of Photovoltaic Module Failure and Degradation Mechanisms: Causes and Detection Techniques. *Solar* **2024**, *4*, 43–82. [CrossRef]
4. Nedaei, A.; Eskandari, A.; Milimonfared, J.; Aghaei, M. Fault resistance estimation for line-line fault in photovoltaic arrays using regression-based dense neural network. *Eng. Appl. Artif. Intell.* **2024**, *133*, 108067. [CrossRef]
5. Flicker, J.; Johnson, J. Photovoltaic ground fault detection recommendations for array safety and operation. *Sol. Energy* **2016**, *140*, 34–50. [CrossRef]
6. Eskandari, A.; Nedaei, A.; Milimonfared, J.; Aghaei, M. A multilayer integrative approach for diagnosis, classification and severity detection of electrical faults in photovoltaic arrays. *Expert Syst. Appl.* **2024**, *252*, 124111. [CrossRef]
7. Pillai, D.S.; Rajasekar, N. A comprehensive review on protection challenges and fault diagnosis in PV systems. *Renew. Sustain. Energy Rev.* **2018**, *91*, 18–40. [CrossRef]
8. Benkercha, R.; Moulahoum, S. Fault detection and diagnosis based on C4.5 decision tree algorithm for grid connected PV system. *Sol. Energy* **2018**, *173*, 610–634. [CrossRef]
9. Madeti, S.R.; Singh, S.N. Modeling of PV system based on experimental data for fault detection using kNN method. *Sol. Energy* **2018**, *173*, 139–151. [CrossRef]
10. Yahyaoui, Z.; Hajji, M.; Mansouri, M.; Bouzrara, K. One-Class Machine Learning Classifiers-Based Multivariate Feature Extraction for Grid-Connected PV Systems Monitoring under Irradiance Variations. *Sustainability* **2023**, *15*, 13758. [CrossRef]
11. Hichri, A.; Hajji, M.; Mansouri, M.; Nounou, H.; Bouzrara, K. Supervised machine learning-based salp swarm algorithm for fault diagnosis of photovoltaic systems. *J. Eng. Appl. Sci.* **2024**, *71*, 12. [CrossRef]
12. Badr, M.M.; Hamad, M.S.; Abdel-Khalik, A.S.; Hamdy, R.A.; Ahmed, S.; Hamdan, E. Fault Identification of Photovoltaic Array Based on Machine Learning Classifiers. *IEEE Access* **2021**, *9*, 159113–159132. [CrossRef]
13. Eskandari, A.; Milimonfared, J.; Aghaei, M. Fault Detection and Classification for Photovoltaic Systems Based on Hierarchical Classification and Machine Learning Technique. *IEEE Trans. Ind. Electron.* **2021**, *68*, 12750–12759. [CrossRef]
14. Lee, J.; Wang, W.; Harrou, F.; Sun, Y. Reliable solar irradiance prediction using ensemble learning-based models: A comparative study. *Energy Convers. Manag.* **2020**, *208*, 112582. [CrossRef]
15. Wang, L.; Liu, J.; Guo, X.; Yang, Q.; Yan, W. Online fault diagnosis of photovoltaic modules based on multi-class support vector machine. In Proceedings of the Proceedings—2017 Chinese Automation Congress, CAC, Jinan, China, 20–22 October 2017; pp. 4569–4574. [CrossRef]
16. Chen, Z.; Han, F.; Wu, L.; Yu, J.; Cheng, S.; Lin, P.; Chen, H. Random forest based intelligent fault diagnosis for PV arrays using array voltage and string currents. *Energy Convers. Manag.* **2018**, *178*, 250–264. [CrossRef]
17. Huang, J.-M.; Wai, R.-J.; Gao, W. Newly-Designed Fault Diagnostic Method for Solar Photovoltaic Generation System Based on IV-Curve Measurement. *IEEE Access* **2019**, *7*, 70919–70932. [CrossRef]

18. Eskandari, A.; Milimonfared, J.; Aghaei, M. Line-line fault detection and classification for photovoltaic systems using ensemble learning model based on I-V characteristics. *Sol. Energy* **2020**, *211*, 354–365. [CrossRef]
19. Emamian, M.; Eskandari, A.; Aghaei, M.; Nedaei, A.; Sizkouhi, A.M.; Milimonfared, J. Cloud Computing and IoT Based Intelligent Monitoring System for Photovoltaic Plants Using Machine Learning Techniques. *Energies* **2022**, *15*, 3014. [CrossRef]
20. Eskandari, A.; Aghaei, M.; Milimonfared, J.; Nedaei, A. A Weighted Ensemble Learning-Based Autonomous Fault Diagnosis Method for Photovoltaic Systems Using Genetic Algorithm. *Int. J. Electr. Power Energy Syst.* **2022**, *144*, 108591. [CrossRef]
21. Mellit, A.; Kalogirou, S. Assessment of machine learning and ensemble methods for fault diagnosis of photovoltaic systems. *Renew. Energy* **2022**, *184*, 1074–1090. [CrossRef]
22. Yang, N.C.; Ismail, H. Voting-Based Ensemble Learning Algorithm for Fault Detection in Photovoltaic Systems under Different Weather Conditions. *Mathematics* **2022**, *10*, 285. [CrossRef]
23. Badr, M.M.; Abdel-Khalik, A.S.; Hamad, M.S.; Hamdy, R.A.; Hamdan, E.; Ahmed, S.; Elmalhy, N.A. Intelligent fault identification strategy of photovoltaic array based on ensemble self-training learning. *Sol. Energy* **2023**, *249*, 122–138. [CrossRef]
24. Adhya, D.; Chatterjee, S.; Chakraborty, A.K. Stacking ensemble based fault diagnosis approach for improved operation of photovoltaic arrays. *Energy Sources Part A Recovery Util. Environ. Eff.* **2022**, *44*, 5421–5439. [CrossRef]
25. Zhao, Y.; De Palma, J.F.; Mosesian, J.; Lyons, R.; Lehman, B. Line–Line Fault Analysis and Protection Challenges in Solar Photovoltaic Arrays. *IEEE Trans. Ind. Electron.* **2013**, *60*, 3784–3795. Available online: https://www.academia.edu/26989165/Line_Line_Fault_Analysis_and_Protection_Challenges_in_Solar_Photovoltaic_Arrays (accessed on 20 June 2012). [CrossRef]
26. International Code Council. *2018 ISEP: International Solar Energy Provisions*; International Code Council: Washington, DC, USA, 2018.
27. Wang, W.; Wang, J.; Sun, Y. Fault Diagnosis of Rolling Bearing Based on Variational Mode Decomposition and Improved Canberra Distance in Symmetrized Dot Pattern Image. *J. Artif. Intell. Mach. Learn Data Sci.* **2023**, *1*, 39–48. Available online: https://urfpublishers.com/journal/artificial-intelligence (accessed on 9 March 2023). [CrossRef]
28. Chi-Square Distance. In *The Concise Encyclopedia of Statistics*; Springer: New York, NY, USA, 2008; pp. 68–70. [CrossRef]
29. Margin Analysis of the LVQ Algorithm. Available online: https://proceedings.neurips.cc/paper_files/paper/2002/hash/bbaa9d6a1445eac881750bea6053f564-Abstract.html (accessed on 4 June 2024).
30. Im, S.K.; Chan, K.H. Vector quantization using k-means clustering neural network. *Electron. Lett.* **2023**, *59*, e12758. [CrossRef]
31. Optimization for Machine Learning—Jason Brownlee—Google Books. Available online: https://books.google.com/books/about/Optimization_for_Machine_Learning.html?id=tW1HEAAAQBAJ (accessed on 11 March 2024).
32. Deng, X.; Liu, Q.; Deng, Y.; Mahadevan, S. An improved method to construct basic probability assignment based on the confusion matrix for classification problem. *Inf. Sci.* **2016**, *340–341*, 250–261. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.