sustainability

MDPI

Article

Real-Time Nuisance Fault Detection in Photovoltaic Generation Systems Using a Fine Tree Classifier

Collin Barker, Sam Cipkar, Tyler Lavigne, Cameron Watson and Maher Azzouz *

Electrical and Computer Engineering, University of Windsor, Windsor, ON N9B 3P4, Canada; barkerc@uwindsor.ca (C.B.); cipka112@uwindsor.ca (S.C.); lavignet@uwindsor.ca (T.L.); watso11r@uwindsor.ca (C.W.)

* Correspondence: mazzouz@uwindsor.ca

Abstract: Nuisance faults are caused by weather events, which result in solar farms being disconnected from the electricity grid. This results in long stretches of downtime for troubleshooting as data are mined manually for possible fault causes, and consequently, cost thousands of dollars in lost revenue and maintenance. This paper proposes a novel fault detection technique to identify nuisance faults in solar farms. To initialize the design process, a weather model and solar farm model are designed to generate both training and testing data. Through an iterative design process, a fine tree model with a classification accuracy of 96.7% is developed. The proposed model is successfully implemented and tested in real-time through a server and web interface. The testbed is capable of streaming in data from a separate source, which emulates a supervisory control and data acquisition (SCADA) or weather station, then classifies the data in real-time and displays the output on another computer (which imitates an operator control room).

Keywords: nuisance faults; photovoltaic farms; machine learning; fine tree classifier; supervisory control and data acquisition (SCADA)

Citation: Barker, C.; Cipkar, S.; Lavigne, T.; Watson, C.; Azzouz, M. Real-Time Nuisance Fault Detection in Photovoltaic Generation Systems Using a Fine Tree Classifier. *Sustainability* **2021**, *13*, 2235. https://doi.org/10.3390/su13042235

Academic Editor: J. C. Hernandez

Received: 30 January 2021
Accepted: 14 February 2021
Published: 19 February 2021

1. Introduction

The energy consumed in the form of electricity in Ontario is about 135 TWh per year [1]. This energy is currently derived using nuclear and fossil fuels and renewable sources (such as wind and photovoltaic). The growing proliferation of renewable sources could limit the dependence on fossil fuels, thus, reducing greenhouse gas emissions, which is a large contributor to global warming and climate change [2]. Most of the photovoltaic (PV) farms have a supervisory control and data acquisition (SCADA) system that is implemented to collect voltage and current data. There is also a weather station on-site that collects various weather information. All of these data can be analyzed during troubleshooting. Extreme weather conditions such as high ambient temperatures, non-uniform shading, heavy snow, and rain, and humidity can severely impact the energy harnessing of PV arrays [3]. Several hours of generation may be lost by the time an issue is identified, resolved and a formal report is submitted to the power system operator to which a PV farm is connected.

A PV fault is detected when there is an untypical output power reduction from the PV array [4]. PV farms are prone to various fault types, such as partial shading [5], short-circuit faults [6], open-circuit faults [7], and snow-rated faults [8]. A significant number of fault alarms that go off are the result of some weather-rated scenarios, such as a hot breeze that temporarily trips the ambient air temperature sensor found on the on-site weather station of a PV farm. These kinds of faults are known as nuisance faults and cause the entire system to be shut down even though there is no need to actually take the grid offline. Many times, these nuisance faults will clear themselves before anyone arrives on-site, making root cause analysis even more difficult, and thus, a nuisance. For the temporary faults, only data mining will help find the cause of the problem. Although they are not critical errors, a PV farm of size 10 MW sells about CAD 30,000 of electricity a day, so even an

hour of downtime can reasonably exceed CAD 5000 of lost revenue at noon [9]. The CAD 5000/h loss is the combined loss of both revenues from not operating along with the cost of sending three trucks with crew in them to the site to determine what went wrong. As solar intensity is not constant across the entire day, noon is the most profitable time as it has the most direct sunlight. While the farm makes nothing at night, a solar farm can make that much revenue in an hour, given summer noon sunlight. This number was not an average, but a worst-case scenario directly quoted from [9]. In Ontario, where solar subsidies are high, rates of up to USD 0.50/kwh are not unheard of [10].

Machine learning can play a significant role in fault detection, classification, and generation forecasting in PV farms [11,12]. In [13], a deep residual network model is developed for photovoltaic arrays to detect partial shading, short-circuit, open circuit, and degradation faults. An algorithm is developed based on artificial neural networks to detect open and short-circuit circuit faults in PV farms [14]. Line-to-line faults are detected based on support vector machine classifiers and multiresolution signal decomposition [15]. A low-cost web server is implemented in [16] using ESP32 for manual monitoring PV panels. However, the proposed system is not self-automated and does not feature intelligent monitoring or notification. In [17], a system for data collection from PV farms is implemented to identify possible extreme meteorological phenomena, relying on data collected from weather stations and other data sources that have access to the Internet. Using the mean voltage of PV panels and voltage dips, a fault detection method is developed in [18] based on power line communication.

An online detection algorithm that identifies faulty PV models is developed in [19] based on temperature scanning. However, this technique requires an infrared camera and is limited to small-size PV arrays. In [20], a probabilistic neural network is proposed to identify PV faults with 85% accuracy. However, the accuracy of fault classification can be improved when employing artificial neural networks (ANNs) [21]. Utilizing an ANN, a real-time monitoring system for PV panels is proposed in [22]. The NN model is developed to predict the output of PV arrays during normal environmental conditions. Based on that model, performance degradation in PV arrays can be detected during faulty conditions. In [23], a multilayer NN (MNN) is developed to classify open-circuit, short-circuit, mismatch, and multiple faults under partial shading. Most of the reported literature focuses on short-circuit, open-circuit, partial shading faults with less emphasis on weather-related (nuisance) faults. The paper contribution can be summarized as follows: (i) based on real-time data from a PV farm and PV-data generation model, a fine tree machine learning algorithm is trained to detect and classify nuisance faults; (ii) a SCADA server is implemented to test and validate the accuracy in real-time.

2. Problem Statement

Based on an in-person interview with the Director of Operations at Northwind Solutions, a Spark Power company (Northwind) in Ontario, Canada, it was determined that whenever a fault occurs, it normally takes a couple of hours to send a crew out and determine the fault that caused the issue [9]. All of the solar farms have a SCADA system that is implemented to collect voltage and current data, and there is a weather station on site that collects various weather information. All of these data can be analyzed during troubleshooting. By the time a nuisance fault is identified, resolved, and a formal report submitted to the power system operator, several hours of generation may be lost.

The primary goal of this paper is to detect faults in photovoltaic power plants using artificial intelligence (AI). Generation systems are of particular interest because revenue is directly based on power generated by electric companies. It is in the best interest of power system operators to minimize and mitigate any potential sources of downtime. One of the most frequent problems that plague power system operators are intermittent weather-related events that cause unpredictable situations resulting in the SCADA system responding with alarms. The Independent Energy System Operators (IESO)—which oversees the operation of the Ontario electricity grid—automatically disconnects solar

farms for protection immediately following alarms. Once a disconnect is initiated, an investigation must be conducted, and a formal report submitted to IESO outlining the cause of the fault and resolution measures in order to be reconnected to the grid [24]. These intermittent faults are the hardest to troubleshoot because the cause of the alarm is not obvious. The conditions may clear themselves before the root cause is found, making the reporting process even more difficult. All of this work results in serious revenue losses due to lost generation time, and the associated cost of technician crews also increases losses. According to Wayne Eyraud, Director of Operations of a renewable energy company, during peak generation times, a typical 10 MW solar farm can lose up to USD 5000 of lost revenue in just one hour of downtime [9].

The intermittent faults mentioned above are known in the industry as nuisance faults. These faults are weather-related events whose impacts on a solar farm induce the system to show faults, when the events that cause these faults have not happened. They are known as such due to their tendency to appear randomly. For example, a very hot and humid day may cause difficulty for the cooling system to keep the internal inverter hut temperature below the heat alarm threshold. This alarm would register in the operating room as an overheating inverter and appear as an electrical fault even though there is nothing electrically wrong. A trained technician crew would be sent out to respond to an overheating inverter when the only response required would be to have one person prop open the door to increase airflow. Many times, operators are left guessing when nuisance faults occur, and there has not been any research found that solves this problem through some kind of detection scheme.

3. Proposed Machine Learning Solution

The proposed solution is to utilize machine learning (ML) to determine which common nuisance fault caused the reported SCADA system alarm. This is achieved by analyzing the data that are already being collected by most typical solar farms. These data contain, at a minimum, basic voltage, current, and power measurements along with current weather data obtained from an on-site weather station. The weather data include the solar irradiance values along with the temperature and relative humidity of the ambient air. With available real-time measurements, fault conditions are classified and then used to train an ML algorithm. This algorithm is then implemented into a system to classify new data in real-time and detect potential nuisance fault conditions. The inherent nature of the algorithm allows it even to forecast upcoming potential faults. The proposed implementation of the ML algorithm is delivered as a Software as a Service (SaaS), which allows for centralized classification and storage of data points that can be easily recalled for further algorithm training.

In order to achieve the proposed solution, a supervisory classification training technique was chosen due to the problem being well defined. The nuisance faults can all be classified based on specific weather conditions. These include temperature, precipitation, cloud cover, and humidity. For the purposes of this research, five fault conditions were analyzed, as shown in Table 1.

Table 1. List of nuisance faults and their common causes.

Class	Condition	Causes
1	Snowfall	Temperature below freezing, precipitation conditions
2	Humidity	Humidex >90% (20% chance of setting alarm)
3	Cloudiness	Irradiance drops by 30%
4	Dust cover	Gradual degradation of panel performance caused by dust or dirt
5	High temperature	Temperature exceeds 31 °C (40% chance of setting alarm)

Since the fault scenarios are the direct result of known weather phenomena, as shown in Table 1, going with a supervised ML algorithm is the obvious choice of action. The data utilized in this paper were created by a weather generator (as detailed in Section 3.1) based

on climate records for the city of Windsor, Ontario, Canada. Each of these weather events can lead to a nuisance fault appearing based on the impacts weather has on solar farms. For example, a threshold of 31 °C represents the cutoff of the ambient air temperature. When the ambient air temperature rises above 31 °C, there will be a chance that an error would occur, representing a nuisance fault from overheating. Likewise, a humidity level of 90% represents the point at which a humidity error could occur. These nuisance faults have a higher chance of occurrence when their respective thresholds are violated.

The ambient temperature threshold (31 °C) is the outside air temperature at ground level when high-temperature alarms would start tripping on inverters. It does not always result in high-temperature alarms tripping because weather effects (such as wind, rain) would play a role in the inverter temperatures. This threshold is regarded to likely trigger a heatwave nuisance fault, which is to be predicted using the proposed model such that less costly measures can be taken to prevent more costly problems. This temperature may be subject to change depending on the installation and general conditions of the site of implementation, but it was the point chosen on available sample data and prevailing weather conditions of the area being used for the proposed model. The 90% humidity threshold is determined with the same rationale for the ambient temperature. It also belongs to the high humidity range specified by IET 60721-3-1 (i.e., above 75%).

It is worth mentioning that nuisance faults, defined in this paper, are weather-related and cause the SCADA system to raise alarms but are not actually real electrical or mechanical faults. This makes these events annoying to troubleshoot and resolve. For instance, dust builds up on solar arrays over time, and thus, deteriorates the performance of the solar farm. Similarly, cloudiness, which denotes the cloud impacts on a solar farm, and snowfalls are not physical faults. However, the SCADA system may interpret these events as undervoltage, or may not report it at all, but still shows as a negative statement. In the case of cloudiness, a sudden drop in light levels to less than 30% of the nominal sunlight would result in the chance of a cloud fault being thrown. The sensitivity of cloudiness detection depends on the 30% threshold. The higher the threshold, the lower the sensitivity, and vice versa. It is up to the SCADA operator to choose the threshold value based on his/her site experience. Over time, the site continued data collection can be used to fine-tune the nuisance fault thresholds and retrain the algorithm for improved accuracy.

This paper aims at devising an ML algorithm that can be added to the existing SCADA system to detect nuisance faults and show their actual causes, rather than the current method of manual data tracing and discussion. Neural networks are used for artificial intelligence when it is better for the machine to find its own patterns in the data being fed and arrive at its own conclusions. Given the data that were provided from a real PV farm that showed alarm codes that were attributed to thresholds in the data set, the outcomes were already defined. Through an iterative process of training various model types, it is determined that a Fine Tree Classifier is the most efficient and accurate approach to solving the problem.

3.1. Data Generation

Training an ML algorithm requires a huge set of data. Initially, this was provided by a solar farm from Southern Ontario. The data contained electrical and thermal readings from the inverters every 10 s. Data from the weather stations were reported every few minutes. The weather reporting is not as regular as electrical reporting. The faults are manually classified based on the alarm codes provided by the SCADA system. The electrical readings that are attributed to the fault code are then compared with the weather readings at the time. The results were promising and showed definite correlations between conditions and specific outcomes. However, since the data obtained were incomplete and did not span an entire year of weather cycles, they were not feasible to be used in the final model development. It is a great proof of concept that shows a clear correlation to weather conditions triggering nuisance faults. A more robust model is created using a generated data set that models local weather conditions and solar farm power characteristics. This

data set includes an entire year of weather and power data for a hypothetical solar farm located in the city of Windsor, Ontario. Seeds (a number used to initialize pseudorandom algorithms) are utilized to randomly generate each year's precipitation and temperature fluctuations within set parameters of a region's climate patterns. These random variations are then used to calculate the temperature, humidity, solar irradiance, dust levels, and snow depth at fifteen-second intervals for an entire year. Nuisance faults are generated as a result of the weather conditions with randomness to the actual occurrence to incorporate their unpredictable nature. This generated data set allows for a much more complete training data set to be created, resulting in increased robustness of the ML algorithm. The data generation process is broken down into two major sections: weather and PV data generations.

3.1.1. Weather Data Generation

Weather data generation is separated into defining initial conditions of the system and determining the impact of independent weather phenomena, which includes precipitation, cloud cover, extreme temperature events, and the accumulation of dust and snow, as demonstrated in Figure 1.

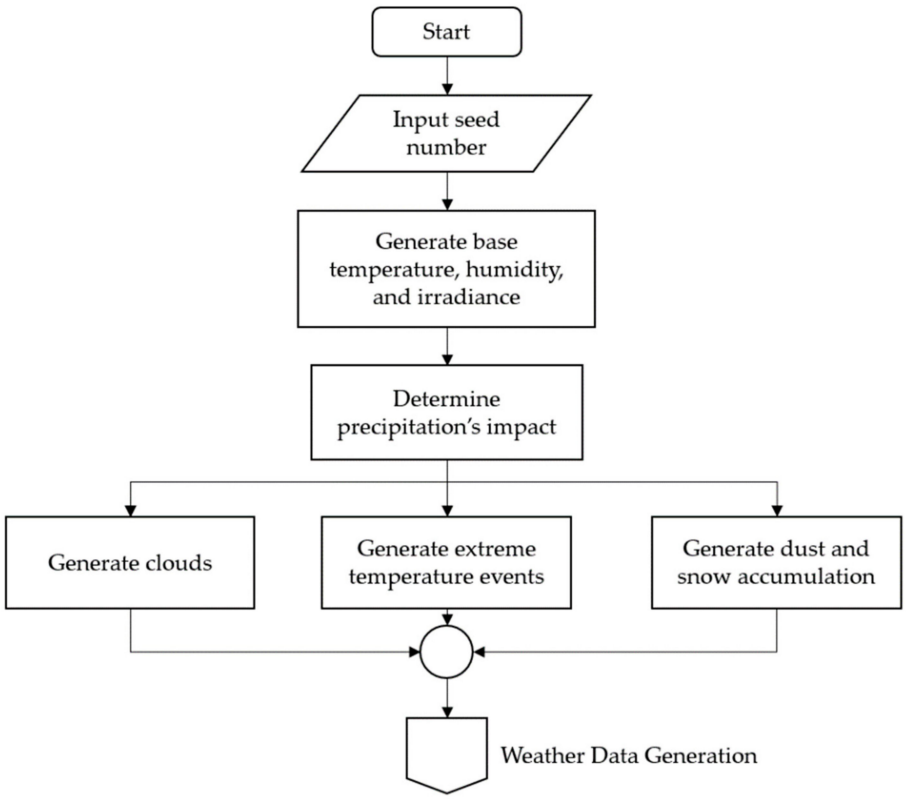

Figure 1. Weather data generation.

Defining these initial conditions involves determining the standard cycles of meteorology so that a second-by-second model can be inferred. This involves finding averages in annual and daily changes in temperature, humidity, precipitation, and solar irradiance. These data were aggregated from these five sources [25–29]. These data were then used in combination with a given random seed to create the initial conditions for a year, which would then have weather events added. The following weather model, i.e., described in (1)–(7), is developed using the aggregated data:

$$T_{year} = -13.5\cos\left(2\pi F_{c,year} t\right) + 9.1\ ^{\circ}\mathrm{C} \quad (1)$$

$$T_{day} = -8.92\cos\left(2\pi F_{c,day}t\right)\ ^{\circ}\mathrm{C} \quad (2)$$

$$T_{base} = T_{year} + T_{day}\ ^{\circ}\mathrm{C} \quad (3)$$

$$RH_{year} = -4.16\sin\left(2\pi F_{c,year}t\right) + 81.16\% \quad (4)$$

$$IR_{year} = -2.33\cos\left(2\pi F_{c,year}t\right) + 3.61\ \mathrm{kWh/m^2/day} \quad (5)$$

$$IR_{day} = 0.5\cos\left(4\pi F_{c,day}t\right) + 0.5\ \ \mathrm{kWh/m^2/day} \quad (6)$$

$$CC_{day} = 0.2\cos\left(2\pi F_{c,year}t - \frac{35}{365} * 2\pi\right) + 0.5\% \quad (7)$$

The weather model is developed by creating curves that would fit the tabulated data found from the meteorological references used. T_{year} is the general average temperature throughout the year. T_{day} is the typical daily temperature swing to be superimposed onto the yearly average temperature to create the temperature curve for the whole year known as T_{base}. RH_{year} is the relative humidity curve, while IR_{year} and IR_{day} are average annual and typical daily solar irradiance levels, respectively. CC_{day} is the cloud cover curve as measured in intensity as a result of typical precipitation patterns for the given region. $F_{c,\mathrm{year}}$ is the frequency for one a year, in seconds, and $F_{c,\mathrm{day}}$ is the frequency for a day in seconds.

Precipitation is considered to be the most impactful event on weather, as it modifies all of the parameters of weather over its duration. After precipitation is generated, its duration and its impact on the parameters are calculated, modifying the base year weather. For solar panels, cloudiness has a major impact on the performance, so randomized clouds are also added to mimic these conditions. For this, the ten recognized types of clouds are modeled, and their impact on solar irradiance is also calculated [30]. The ten recognized cloud types, in increasing opacity, are shown in Table 2.

Table 2. Cloud code and relative cloud type visible in sky cloud.

Code	Cloud Type
1	Cloudless
2	Cirrus
3	Altocumulus Cirrostratus Cirrocumulus
4	Altocumulus
5	Altostratus
6	Nimbostratus
7	Stratocumulus
8	Stratus
9	Cumulous
10	Cumulonimbus cumulous

The corresponding ground-level solar intensity factor from [30] for each cloud is then plotted, and a curve of best fit is obtained to continuously transition between each type as

$$\text{Solar Intensity Factor}(\%) = 0.4655\mathrm{C}^3 - 7.8953\mathrm{C}^2 + 26.596\mathrm{C} + 78.139 \quad (8)$$

where C is the cloud type number from Table 2. The relationship in (8) allows for the transition periods between different types of clouds through changing weather patterns.

Heatwaves and cold snaps also occur, both of which can negatively impact solar panel performance. These are modeled after the heat waves and cold snaps of Windsor, ON, where there are two to three of each per year, with each instance lasting 2–3 days. Their modified temperatures are incorporated by applying a scaling ramp function to the temperature for a random set of days. These phenomena would occur within seasonal parameters.

Dust and snow coverage of panels also greatly affect solar panel performance. Dust slowly accumulates over time, but in light rain, the dust cakes onto the panels, causing the panels to perform increasingly worse. The accumulation and rinsing of dust are modeled

based on the work carried out by [31]. Snow depth and its impact on solar intensity have also been studied, and the derived model from [32] is used as given by

$$\psi = \psi_o e^{-1.107061d} \quad (9)$$

where ψ_o represents the input irradiance flux density, ψ is the output irradiance flux density, and d is the depth of snow. Time to melt was also simulated based on an hour above 0 °C to warm the panels being required for the snow to melt and slip off.

3.1.2. PV Data Generation

Any available solar panel model includes transient characteristics. However, due to the low sampling time of the system being monitored [9], these models are not required as the transient behavior observed settles sooner than the sampling time. For increased efficiency, PV modeling equations are developed. The equations model the power output of the panel, along with the voltage and current for any given solar irradiance and panel temperature. The model assumes maximum power point tracking (MPPT) is integrated with the inverter. These are the basic parameters required for any successful modeling of solar panel conditions and faults [33]. Figure 2 displays the proposed method for generating PV data for fault classification.

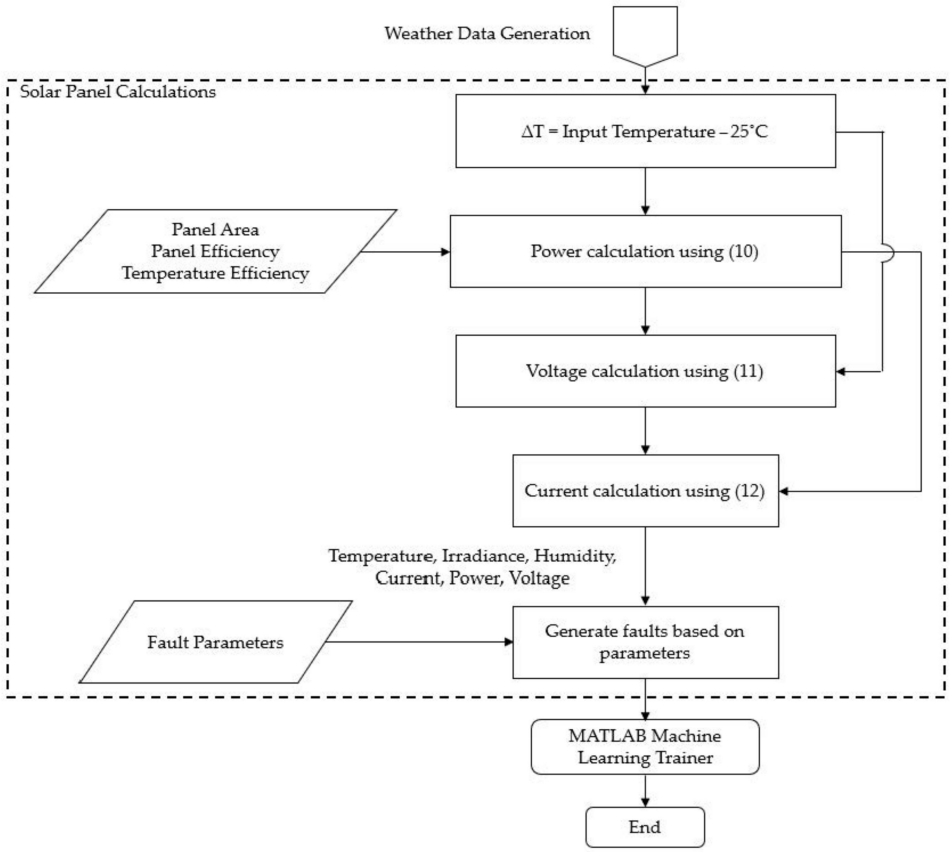

Figure 2. PV data generation.

The power characteristics of a PV panel are modeled using the panel area of the system and the efficiency characteristics available in the datasheet for a SunPower SPR-305E-WHT-D solar panel [34], as follows:

$$P = \psi A \eta \left(1 + \Delta T \tau_p\right) \quad (10)$$

where P is the PV output power, and A is the area of the panel, η denotes the panel efficiency, and ΔT is the difference in temperature from the performance testing parameter of 25°.

Usually, a sensor attached to the back of a module is used to measure its temperature [35]. This difference is then multiplied by the power temperature coefficient τ_p (which equals $-0.38\%/°C$). Solar panel performance is temperature-dependent, and all specifications are based upon one testing scenario, usually conducted at 25 °C. Therefore, this inherent property, known as the temperature coefficient of efficiency of the solar panel, must be incorporated into any model where the solar panel will be in operation outside of lab conditions.

To obtain a function of PV panel voltage and a function of its output power and temperature, a series of power–voltage (P/V) and current–voltage (I/V) curves are plotted as shown in Figures 3 and 4. These curves are inherent properties of the solar panel and are given by the manufacturer's specification. The MPPT data points are taken from the curves put into Excel. The coefficients for the modeling equation are then derived using curve fitting techniques, which yield equations of best fit, as shown in Figure 5, and given in (11).

$$V = 1.7537ln(P) + 45.339 \quad (11)$$

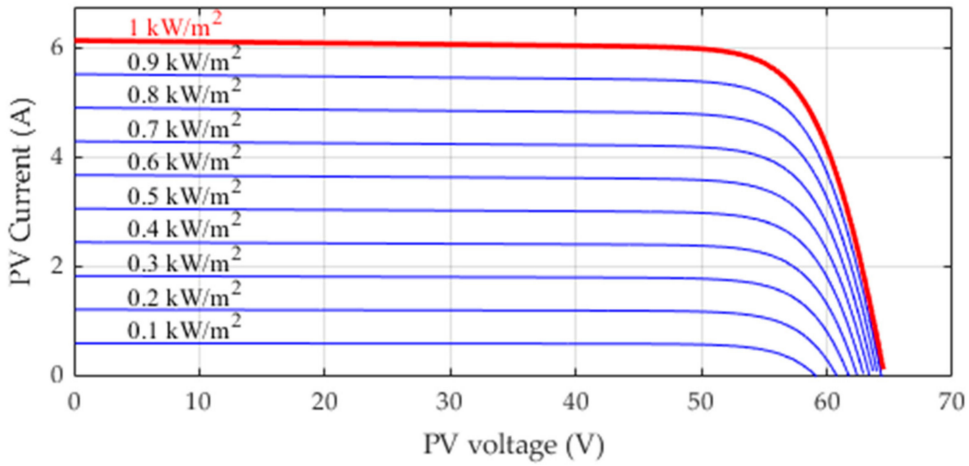

Figure 3. Photovoltaic current–voltage (I/V) characteristics at various irradiances.

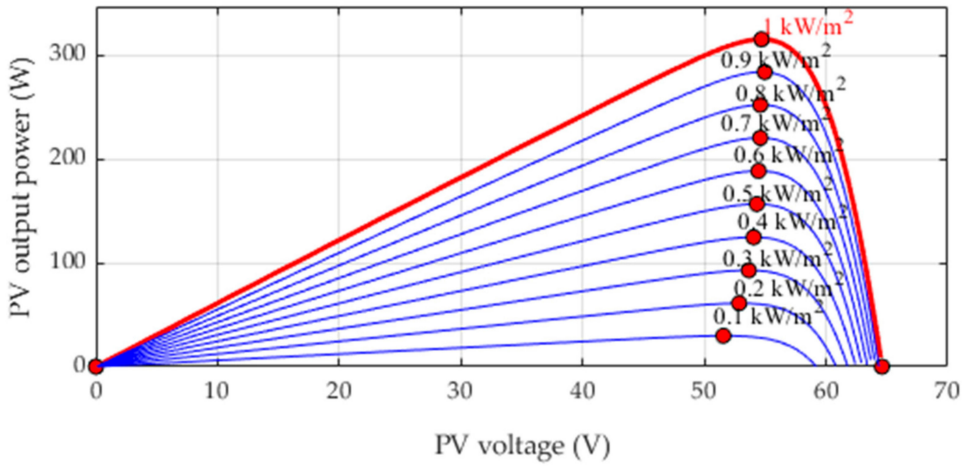

Figure 4. Photovoltaic power–voltage (P/V) characteristics at various irradiances.

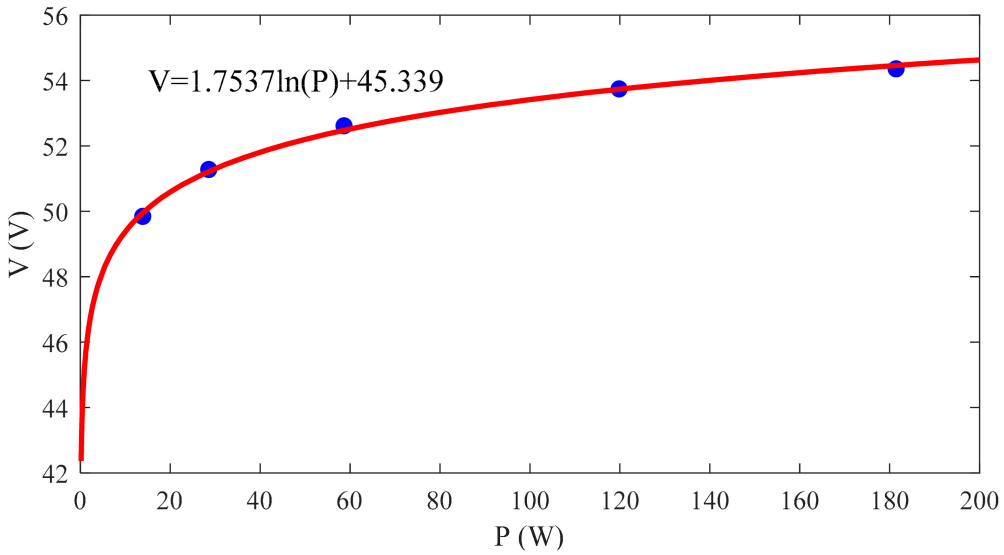

Figure 5. Extracted maximum peak power tracking voltage–power (V/P) curve.

Using (10) and (11), the PV output current can be determined by

$$I = \frac{\psi \mathrm{A} \eta \left(1 + \Delta \mathrm{T} \tau_p\right)}{1.7537 \ln(\mathrm{P}) + 45.339} \tag{12}$$

The PV voltage and current are calculated using (11) and (12) after adjusting the output power using τ_p in (10). Therefore, (11) and (12) take into account the temperature variations. It should be noted that the voltage and current in (11) and (12) typically represent readings from an inverter that applies an MPPT algorithm, which also responds to temperature variations.

3.2. Algorithm Training

The faults in Table 1 are implemented based on the weather conditions and built-in pseudorandomness, simulating the nature of nuisance faults. For example, when temperature and humidity levels reached certain thresholds, there would be a 33% chance that a heat fault event would occur. These fault events are coded to also create the outcome flag so that the manual classification could be automated. As described earlier, a supervised ML algorithm is trained that compares inputs to pre-classified outputs to determine faults rather than manually going and classifying the faults after the data were generated from the PV farm model, the correct classifications are immediately generated with the rest of the data.

The weather and PV data-generation processes are run multiple times to generate many years' worth of PV farm data. MATLAB's machine learning toolbox is used to construct and train the machine learning algorithm [36]. Many algorithms are tested to determine which model performs the best based on accuracy and efficiency. The classifiers had issues with the extended periods of dust and mainly confused it with normal operation. This was addressed by increasing the training set with seeds that had a more even number of faults overall. It was clear very early into the training process that decision trees were the best models for the kind of data being used in training due to the model training quickly and having the highest accuracy of the many that were tested. Training data sets are generated and chosen based on their distribution of fault conditions.

Since nuisance fault events are quite rare compared to the amount of normal operation that should be expected from a functional solar farm, the original data sets needed to be manipulated for training accuracy. A typical year-long data set provided by the model has a very disproportionate number of normal operation classifications than faulted conditions. This proportion of a single case to the rest of the case affects how sensitive the model is to

that particular data type. In the case of the model being discussed, the algorithm favors the "normal" fault case over the rest because it is the most prominent, but the exact opposite response is desired. This problem is known as overfitting. The final training data set is manipulated by under-sampling the data to filter out approximately 99.5% of the normal cases from multiple years of data. This shows how uncommon fault events are in the grand scheme of things, yet when they do happen, the economic impacts are significant.

The under-sampling technique filters through the data set only keeping 1 in 200 "normal" cases. When the filter script hits faulted scenarios, it stops filtering before and after the condition so that the AI can learn what kind of data lead up to a fault. This results in a single data set for training that has a more blended mix of various fault conditions and prevents the training AI from becoming too partial to the normal case and ignoring faulted scenarios in the interest of better raw accuracy.

The system's performance is evaluated based on its accuracy and how fast it could make the classifications. The error rate of the system is evaluated using two different methods. The first is the raw error rate: how many faults are incorrectly classified versus how many data points are classified in total. This is used to determine the bulk accuracy of the system and how well it classified the set.

In fine-tuning the system, a second method of determining accuracy is used. The sensitivity is the number of data points of a specific classification from a set that the system can correctly identify. The error rate is taken as how many faults are incorrectly classified minus the misclassified normal cases as a fraction of the total number of data points. The sensitivity of normal cases is ignored because the system would sometimes classify the circumstances of a normal case as an error, but if nothing is wrong on the operator's end, then no action would need to be taken. In the case of monitoring for nuisance faults, it is more desirable to misclassify normal operations as a potential fault than to misclassify faulted conditions as a normal operation as this creates a forecasting ability of the system. The confusion matrix in Figure 6 illustrates these concepts visually. Classes 1 through 5 are outlined in Table 1, and Class 0 is the normal case, where no fault is occurring.

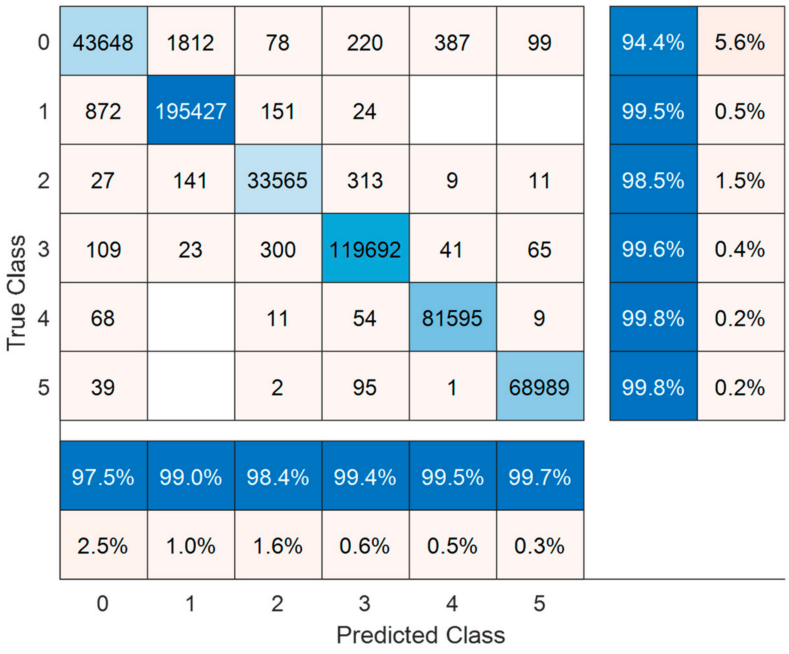

Figure 6. Confusion matrix showcasing training performance.

The confusion matrix in Figure 6 shows the correct classifications along the diagonal in blue and incorrect classifications anywhere else in red. The actual fault in the data, or the true class, is shown along the vertical axis. The predicted value returned by the system, or the predicted class, is shown on the horizontal axis, and the data are mapped accordingly.

The sensitivity of normal cases is calculated from here as the true positives of the class 0 row over the total cases in the class 0 row. This shows that the system classifies 81.68% of the normal cases correctly. The other 18.32% of cases in this class are ignored in calculating the accuracy because they would be the preferred misclassification of a normal operation as being a fault.

An example of this playing out, in reality, would be the monitoring AI notifying the operator of an overheating fault condition. However, the actual temperature sensors on the SCADA system are not signaling an alarm condition. This situation would be an incorrect classification by the AI but is desirable nevertheless because the operator can begin preparing for a fault event. The AI is anticipating this result based on all of the combined data it is analyzing. Hence, the operator may be able to begin an immediate heat mitigation response that will ultimately result in the alarms never going off and the PV system staying online the entire time. Ultimately this apparent fault with the AI from a numbers standpoint yields the predictive nature of the AI that was originally desired.

Through the iterative process of training various ML algorithms using various settings, a fine tree model was selected. The fine-tree classifier shines out from the rest of the options for a couple of different reasons. Firstly, it is computationally simple. A tree model classifier simply created a large decision tree to classify the data. At each node, a true/false decision must be made based on the data values so that the AI can successfully arrive at the conclusion it is given for that point. The other options, such as Gaussian mixtures and K-Nearest Neighbor algorithms, use complicated curve fitting techniques that are much slower than the fine tree. For example, on the same set of data, a fine tree model can be trained in minutes, while a KNN algorithm takes several days. Secondly, the accuracy of the fine tree was higher than all of the other options. The nature of the data being classified is such that the classifications are made based mainly on threshold values and correlations which are easily determined using decision trees.

The final fine-tree model is further tested for accuracy on more data generated from the weather/PV system model. Table 3 shows the total number of misclassifications minus the ignored normal cases. The mean accuracy was calculated to be 96.7%. Two separate AI models were trained using the same data and compared for accuracy. Each training session ultimately yields different results, and the best one can be hand-picked at the end.

Table 3. List of seeds and the recorded errors of two final candidates.

Seed	Misclassification Count	
	Classification Model A	Classification Model B
1234	168,383	171,419
826453	51,451	48,908
745123	55,588	53,531
785412	52,551	49,876
745698	199,216	193,852
132542	24,162	22,739
8712312	34,110	31,449
1325412	55,739	51,336
132123	48,594	44,225
462513	61,870	59,239
128465	56,774	52,254
432156	63,933	59,955

The seeds were chosen based on the wide variety of fault conditions that were generated in them. Most of the seeds generated a normal spread of faults, while other seeds had little to no dust accumulation issues. This was because the weather that year had regularly occurring rain to keep the panels clean. On the other end of the spectrum, 1234 and 745,698 had long periods of no rain that resulted in a severe buildup of dirt upon the panels. This resulted in a disproportionately large number of faults compared to other nominal scenario seeds.

4. Server Implementation

Once the development and training of an effective ML algorithm in the MATLAB Machine Learning Toolbox environment are complete, the trained ML algorithm is exported to a code script. This MATLAB language code script is then translated into C code using the C-coder tool for use outside the MATLAB ecosystem. This was to allow for a client-facing system that would collect and classify all the data as they were received and display the result to the operators. There were two choices for the implementation of this system: an embedded black box or a web application. Both of these approaches had their advantages, but a web application was picked for a few key reasons. First, in a web application, the data would be sent to a server to be processed rather than processed on site. This means that the algorithm can be retrained on the new data that were received to even further improve the performance. The black box approach would have made this impractical. Second, the low upfront cost to adopters supported by a web application means that more operators would be inclined to implement this for their system. The black-box approach, in contrast, would require special hardware hookups, leading to a higher upfront cost. Third, a web application allows for cheaper maintenance and updating as clients would be unaffected so long as there is no downtime, which is common for many web technologies and services.

The system was implemented with three distinct points where data exist. The diagram in Figure 7 displays the data pathways of the system.

Figure 7. Data flow diagram showing the connectivity between the data agent, server, and local operator.

Data in this system are created by the "data agent", the system that logs the data and sends them to the server. The data agent of most solar farms would be SCADA, or some system attached to SCADA designed to read all the sensor data and send them to the server. The SCADA system in question would require an external network interface to allow it to communicate with the server. The server is a representational state transfer (REST) service created using the Ruby on Rails web framework and the PostgreSQL relational database [37,38]. In a REST environment, requests are divided into categories [39]. The data agent makes only POST requests to the server as it only commits data autonomously. Operators make GET requests to view the data. For convenience, the service was deployed using Capistrano to a virtual dedicated server running Ubuntu Server 18.04 [40,41]. The study was served using the Puma web server through Nginx as a reverse proxy to easily manage and regulate requests [42,43]. The operator is the system where information of the solar farm is viewed. This system, being a web application, is viewed through a web browser from anywhere. The advantage of the system herein is data management and reclassification. Holding all of the data and sorting it according to the location of the collection means that the data can be reclassified to correctly reflect what fault occurred in erroneous cases. This, in turn, will allow for retraining of the algorithm or, with more advanced techniques, adaptive classification.

The speed of the algorithm is important as the server system would require this to run over and over on as much data as can be given to it. MATLAB offers a metric for training speed within their environment, but when the AI exported to C code, it ran considerably faster. When classifying within MATLAB, the sets in Table 2 took seconds to classify, whereas when given to the C code, the data were classified in only a few hundred milliseconds, including network latency. This proposed solution is effective at solving the problem as it is both cost-effective and requires minimal added infrastructure to install. This is due to the cloud infrastructure inherent in the SaaS delivery model. Data analysis is streamlined by actively monitoring SCADA and weather station data to determine a

possible outcome. As a result, the proposed solution will mitigate economic losses by minimizing generation downtime.

5. Conclusions

Information is key in any industry; it is essential for every decision made on a daily basis. However, the information needs to be analyzed in order to be used effectively in decision making. Solar generation is a burgeoning industry that creates and collects vast quantities of data, which are currently processed manually. When a fault occurs, a technician is dispatched to fix the issue. A large amount of the time, these issues will disappear while the technician is on the route, so sending out a technician would be a waste of time and money. In this paper, a fine tree model-based machine learning classifier is developed in the hopes that it could help industries save money from resolving this issue, commonly known as nuisance faults. This classifier is able to learn from the previous data to give the user a solution on what caused the fault condition. The initial data originally came from a real solar farm and were used to give a basis for this machine learning algorithm training. Due to incompleteness in the data obtained, a model was created using past Windsor weather history to create training and testing data. Using MATLAB, weather and PV data generators were created that were capable of generating infinite amounts of weather data based on a provided seed and the weather trends of the past five years.

The parameters that the classifier was trained on were temperature, humidity, irradiance, voltage, current, and power. Other parameters may be used in addition to training, but these six are the minimum most solar farms produce, which shows that any system is capable of detecting these faults without added sensing equipment. Once the parameters were chosen, the AI was trained using various models. Through an iterative design process, it was determined that the fastest model is a fine resolution decision tree algorithm that ultimately has an effective accuracy of over ninety-six percent. The initial AI model was trained using the MATLAB Machine Learning toolbox. The trained algorithm was then converted to C code using MATLAB's C Coder to allow for implementation on external systems and hardware independent of the MATLAB environment.

An interface was created to allow operators access to the classifier's output. This interface was given a variety of functions to allow the operators to understand what the algorithm returns, along with facilitating the ability to manually reclassify previous data if they are found to be wrong. From the aforementioned process, the initial hypothesis was deemed to be correct. The AI is capable of helping organize data faster, and it can give the user a clearer representation of the current situation. This solution could help companies save thousands of dollars per year, provided sufficient data are available. By implementing this machine learning system, a higher level of efficiency and reliability for solar generation may be obtained.

Author Contributions: Conceptualization, S.C. and T.L.; methodology, C.B. and C.W.; software, C.W.; validation, C.B., S.C., T.L. and C.W.; formal analysis, S.C.; investigation, C.B., S.C., T.L. and C.W.; resources, T.L.; data curation, C.B. and S.C.; writing—original draft preparation, C.B., S.C., T.L. and C.W.; writing—review and editing, C.B., S.C., T.L., C.W. and M.A.; visualization, C.B. and S.C.; supervision, M.A.; project administration, T.L. and M.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) under Grant RGPIN-2017-04990.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

References

1. CER—Provincial and Territorial Energy Profiles—Ontario. Available online: https://www.cer-rec.gc.ca/nrg/ntgrtd/mrkt/nrgsstmprfls/on-eng.html (accessed on 10 July 2020).
2. Lima, M.; Mendes, L.; Mothé, G.; Linhares, F.; de Castro, M.; da Silva, M.; Sthel, M. Renewable energy in reducing greenhouse gas emissions: Reaching the goals of the Paris agreement in Brazil. *Environ. Dev.* **2020**, *33*, 100504. [CrossRef]
3. Alam, M.K.; Khan, F.; Johnson, J.; Flicker, J. A Comprehensive Review of Catastrophic Faults in PV Arrays: Types, Detection, and Mitigation Techniques. *IEEE J. Photovoltaics* **2015**, *5*, 982–997. [CrossRef]
4. Abdulmawjood, K.; Refaat, S.S.; Morsi, W.G. Detection and prediction of faults in photovoltaic arrays: A review. In Proceedings of the 2018 IEEE 12th International Conference on Compatibility, Power Electronics and Power Engineering (CPE-POWERENG 2018), Doha, Qatar, 10–12 April 2018; IEEE: New York, NY, USA, 2018; pp. 1–8.
5. Silvestre, S.; Kichou, S.; Chouder, A.; Nofuentes, G.; Karatepe, E. Analysis of current and voltage indicators in grid connected PV (photovoltaic) systems working in faulty and partial shading conditions. *Energy* **2015**, *86*, 42–50. [CrossRef]
6. Pillai, D.S.; Ram, J.P.; Rajasekar, N.; Mahmud, A.; Yang, Y.; Blaabjerg, F. Extended analysis on Line-Line and Line-Ground faults in PV arrays and a compatibility study on latest NEC protection standards. *Energy Convers. Manag.* **2019**, *196*, 988–1001. [CrossRef]
7. Akram, M.N.; Lotfifard, S. Modeling and Health Monitoring of DC Side of Photovoltaic Array. *IEEE Trans. Sustain. Energy* **2015**, *6*, 1245–1253. [CrossRef]
8. Hashemi, B.; Cretu, A.-M.; Taheri, S. Snow Loss Prediction for Photovoltaic Farms Using Computational Intelligence Techniques. *IEEE J. Photovoltaics* **2020**, *10*, 1044–1052. [CrossRef]
9. Eyraud, W. (Director of Operations of NorthWind Solutions, A Spark Power Company). Interview with the authors in Windsor, ON, Canada. June 2019.
10. Canada Feed In Tariff | Ontario FIT, Greenstream Publishing Ltd. Available online: http://www.solarelectricityhandbook.com/canada-feed-in-tariff.html (accessed on 29 October 2020).
11. Youssef, A.; El-Telbany, M.; Zekry, A. The role of artificial intelligence in photo-voltaic systems design and control: A review. *Renew. Sustain. Energy Rev.* **2017**, *78*, 72–79. [CrossRef]
12. Viscondi, G.D.F.; Alves-Souza, S.N. A Systematic Literature Review on big data for solar photovoltaic electricity generation forecasting. *Sustain. Energy Technol. Assess.* **2019**, *31*, 54–63. [CrossRef]
13. Chen, Z.; Chen, Y.; Wu, L.; Cheng, S.; Lin, P. Deep residual network based fault detection and diagnosis of photovoltaic arrays using current-voltage curves and ambient conditions. *Energy Convers. Manag.* **2019**, *198*, 111793. [CrossRef]
14. Kumar, S.S.; Selvakumar, A.I. Detection of the faults in the photovoltaic array under normal and partial shading conditions. In Proceedings of the 2017 Innovations in Power and Advanced Computing Technologies (i-PACT) 2017, Vellore, Tamil Nadu, India, 21–22 April 2017; pp. 1–5. [CrossRef]
15. Yi, Z.; Etemadi, A.H. Line-to-Line Fault Detection for Photovoltaic Arrays Based on Multiresolution Signal Decomposition and Two-Stage Support Vector Machine. *IEEE Trans. Ind. Electron.* **2017**, *64*, 8546–8556. [CrossRef]
16. Allafi, I.; Iqbal, T. Design and implementation of a low cost web server using ESP32 for real-time photovoltaic system monitoring. In Proceedings of the 2017 IEEE Electrical Power and Energy Conference (EPEC), Saskatoon, SK, Canada, 22–25 October 2017; IEEE: New York, NY, USA, 2017; Volume 2017, pp. 1–5.
17. Haba, C.-G. Monitoring photovoltaic parks for damage prevention and optimal operation. In Proceedings of the 2017 International Conference on Electromechanical and Power Systems (SIELMEN), Iasi, Romania, 11–13 October 2017; pp. 321–326.
18. Han, J.; Jeong, J.-D.; Lee, I.; Kim, S.-H. Low-cost monitoring of photovoltaic systems at panel level in residential homes based on power line communication. *IEEE Trans. Consum. Electron.* **2017**, *63*, 435–441. [CrossRef]
19. Takashima, T.; Yamaguchi, J.; Otani, K.; Oozeki, T.; Kato, K.; Ishida, M. Experimental studies of fault location in PV module strings. *Sol. Energy Mater. Sol. Cells* **2009**, *93*, 1079–1082. [CrossRef]
20. Gul, S.; Haq, A.U.; Jalal, M.; Anjum, A.; Khalil, I.U. A Unified Approach for Analysis of Faults in Different Configurations of PV Arrays and Its Impact on Power Grid. *Energies* **2019**, *13*, 156. [CrossRef]
21. Chen, L.; Han, W.; Huang, Y.; Cao, X. Online Fault Diagnosis for Photovoltaic Modules Based on Probabilistic Neural Network. *Eur. J. Electr. Eng.* **2019**, *21*, 317–325. [CrossRef]
22. Samara, S.; Natsheh, E. Intelligent Real-Time Photovoltaic Panel Monitoring System Using Artificial Neural Networks. *IEEE Access* **2019**, *7*, 50287–50299. [CrossRef]
23. Ul-Haq, A.; Sindi, H.F.; Gul, S.; Jalal, M. Modeling and Fault Categorization in Thin-Film and Crystalline PV Arrays through Multilayer Neural Network Algorithm. *IEEE Access* **2020**, *8*, 102235–102255. [CrossRef]
24. IESO, Market Manual 7.3: Outage Management. Available online: http://www.ieso.ca/-/media/files/ieso/document-library/market-rules-and-manuals-library/market-manuals/system-operations/so-outagemanagement.pdf (accessed on 14 July 2020).
25. Windsor Climate. Available online: https://en.climate-data.org/north-america/canada/ontario/windsor-3064/#climate-table (accessed on 7 October 2020).
26. Windsor Temperatures: Averages by Month. Available online: https://www.currentresults.com/Weather/Canada/Ontario/Places/windsor-temperatures-by-month-average.php (accessed on 15 March 2019).
27. Farmzone:Windsor-EssexWest, Ontario. Available online: http://www.farmzone.com/statistics/wind/cl6139525/so001 (accessed on 20 March 2019).

28. Windsor Monthly Climate Averages. Available online: https://www.worldweatheronline.com/lang/en-ca/windsor-weather-averages/ontario/ca.aspx (accessed on 20 March 2019).
29. Solar Electricity Handbook 2019 Edition. Available online: http://solarelectricityhandbook.com/solar-irradiance.html (accessed on 20 March 2019).
30. Matuszko, D. Influence of the extent and genera of cloud cover on solar radiation intensity. *Int. J. Clim.* **2012**, *32*, 2403–2414. [CrossRef]
31. Karmouch, R.; Hor, H.E. Solar Cells Performance Reduction under the Effect of Dust in Jazan Region. *J. Fundam. Renew. Energy Appl.* **2017**, *7*, 1–4. [CrossRef]
32. O'Neill, A.D.J.; Gray, D.M. Solar Radiation Penetration through snow. In Proceedings of the International Symposia on the Role of Snow and Ice in Hydrology, Banff, AB, Canada, 6–20 September 1972; UNESCO: Paris, France, 1972; pp. 227–241.
33. Chao, K.-H.; Chen, P.-Y.; Wang, M.-H.; Chen, C.-T. An Intelligent Fault Detection Method of a Photovoltaic Module Array Using Wireless Sensor Networks. *Int. J. Distrib. Sens. Netw.* **2014**, *10*, 1–12. [CrossRef]
34. 305 SOLAR PANEL, Datasheet, SunPower Corporation. Available online: https://www.pocosolar.com/wp-content/themes/twentyfifteen/pdfs/Sunpower%20Solar%20Panels/sunpower_305wht_spec_sheet.pdf (accessed on 1 March 2019).
35. Dirnberger, D. Photovoltaic module measurement and characterization in the laboratory. In *The Performance of Photovoltaic (PV) Systems*; Elsevier BV: Amsterdam, The Netherlands, 2017; pp. 23–70.
36. MATLAB Statistics and Machine Learning Toolbox. Available online: https://www.mathworks.com/products/statistics.html (accessed on 15 February 2019).
37. Hansson, D.H. Ruby on Rails | a Web-Application Framework that Includes Everything Needed to Create Database-Backed Web Applications According to the Model-View-Controller (MVC) Pattern. Available online: https://rubyonrails.org/ (accessed on 20 July 2020).
38. PostgreSQL: The World's Most Advanced Open Source Database, the PostgreSQL Global Development Group. Available online: https://www.postgresql.org/ (accessed on 22 July 2020).
39. Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content. 2014. Available online: https://www.rfc-editor.org/info/rfc7231 (accessed on 22 July 2020).
40. Clements, T.; Hambley, L. A Remote Server Automation and Deployment Tool Written in Ruby. Available online: https://capistranorb.com (accessed on 22 July 2020).
41. The Leading Operating System for PCs, IoT Devices, Servers and the Cloud | Ubuntu, Canonical Ltd. Available online: https://ubuntu.com/ (accessed on 22 July 2020).
42. Puma: A Fast, Concurrent Web Server for Ruby & Rack, Evan Phoenix and Contributors. Available online: https://puma.io/ (accessed on 22 July 2020).
43. NGINX | High Performance Load Balancer, Web Server, & Reverse Proxy, F5 Inc. Available online: https://www.nginx.com/ (accessed on 22 July 2020).