

AI-Based Fault Prediction for Small Solar PV Systems: An ESP32 Acquisition Unit and a Proposed Machine-Learning Framework

Dr. R. A. Keswani (Guide), Prof. K. G. Gaurkhede (Co-guide), Harsh Bhoyar,
Khushali Thote, Shatur Thakre, Yash Bobade and Ayush Surankar

Department of Electrical Engineering
Priyadarshini College of Engineering, Nagpur, India

Abstract: Artificial intelligence (AI) methods for photovoltaic (PV) fault diagnosis and prediction are usually developed on well-instrumented plants with reliable irradiance data and plentiful labelled examples, conditions that small, low-cost installations rarely meet. This paper presents a three-stage machine-learning framework for fault prediction in such installations, together with the ESP32-based acquisition unit that will supply its data. The unit is built and running. It measures PV voltage (GPIO34), current through an ACS712-20A sensor (GPIO35) and panel temperature through a DS18B20 sensor (GPIO4), computes power, and shows fixed-threshold status labels on a 16×2 I²C display. The AI framework is proposed and has not yet been implemented. Its first stage is an Isolation Forest trained only on normal-operation data, so that abnormal behaviour can be flagged before labelled fault data exist. Its second stage is a Random Forest classifier that assigns a fault class once staged-fault data have been collected. Its third stage predicts slowly developing faults: it fits trends to daily performance and thermal indices and estimates the number of days until each index crosses a fault threshold, giving an early warning before the fault becomes significant. Sudden faults such as open or short circuits cannot be predicted in this way and are only detected. All stages use features that combine raw measurements with physics-based residuals: the difference between measured power and the power expected from irradiance and panel temperature, and the difference between measured and expected panel temperature. A data-quality gate, a fault-class and labelling plan, a blocked train-test split and an evaluation protocol covering classification quality and prediction lead time are specified. No model has been trained, so no accuracy or prediction-error figures are reported; all pending results are marked.

Keywords: Photovoltaic fault prediction, machine learning, trend forecasting, Isolation Forest, Random Forest, ESP32

I. INTRODUCTION

PV modules face electrical faults, partial shading, soiling, hotspots and gradual degradation over their service life [1]. Not every fault is cleared by protective devices: line-to-line faults at low irradiance can produce currents too small to operate overcurrent protection [2]. Detecting such conditions therefore requires interpreting operating data, not just reacting to protection trips.

Machine learning has become a prominent way to do this. Reviews of artificial neural network (ANN) methods show that learned models can detect and classify a range of PV faults from electrical and environmental inputs [3], and specific ANN-based diagnosis techniques have been demonstrated on instrumented systems [4]. These results rest on two conditions: trustworthy measurements, including irradiance and temperature, and enough labelled examples of each



fault [3], [5]. A small rooftop or laboratory installation monitored by a low-cost microcontroller meets neither condition at the outset. Its sensors are uncalibrated, its environmental context is coarse, and it has no fault history.

This paper asks how AI-based fault prediction should be designed for that starting point. It describes the authors' ESP32 acquisition unit as it currently exists, then specifies a three-stage learning framework. The framework can begin operating on normal data alone, uses physical knowledge to make its features informative, and refuses to score invalid measurements. It becomes a fault classifier as labelled data accumulate, and it forecasts gradual faults from their trends so that they can be addressed before they cause significant loss. Implemented and proposed elements are separated throughout.

Section II reviews related work and defines terms. Section III states the research gap and contributions. Sections IV and V present the architecture and the implemented hardware. Section VI describes the data pipeline, and Section VII the AI fault-prediction method. Section VIII covers the dataset and labelling plan, and Section IX the evaluation protocol. Section X discusses limitations, and Section XI concludes.

II. RELATED WORK AND LITERATURE REVIEW

A. Model-Based and Statistical Detection

Before learned models, and still alongside them, detection has relied on comparing measured output with a physical or empirical expectation. Platon et al. [6] predicted PV output from environmental measurements and reported faults when the measured value deviated beyond a statistically derived bound. Silvestre et al. [7] used current and voltage indicators referenced to simulated values, which also allowed broad fault types to be distinguished. Harrou et al. [8] monitored one-diode-model residuals with statistical control charts. These methods are interpretable and need little training data, but each depends on an accurate model and on good irradiance and temperature measurements.

B. Machine-Learning-Based Fault Detection

Chine et al. [4] combined analysis of electrical attributes with ANNs to identify fault types in a PV system. In a broader survey, Li et al. [3] examined how ANN-based PV diagnosis studies select their inputs and architectures, and noted that reported performance hinges on how representative the training data are. Ensemble tree methods are a common alternative for tabular sensor data. A Random Forest [9] aggregates many decision trees, each trained on a bootstrap sample with random feature selection, and provides class probabilities and feature-importance measures. Where labelled faults are scarce, unsupervised anomaly detection is attractive. Isolation Forest [10] flags points that random partitioning isolates quickly, and it needs no fault labels. Chandola et al. [11] define anomalies as departures from expected behaviour and identify contextual anomalies, which are abnormal only relative to conditions such as time or environment. PV output, which depends on irradiance, is a clear example of contextual behaviour.

C. Low-Cost Data Acquisition

Low-cost platforms have been used to collect and transmit PV data. López-Vargas et al. [12] built an Arduino-based system with 3G connectivity for solar home systems, and Pereira et al. [13] used a Raspberry Pi to stream plant measurements to the cloud. Livera et al. [5] emphasise that performance-data diagnosis is only as reliable as the electrical and meteorological data behind it, a constraint that weighs heavily on such platforms.

D. Terminology

Table I fixes the terms used in this paper and states the project's position for each. In particular, fault prediction here means forecasting when a slowly developing condition will reach a fault threshold. Full predictive maintenance, which estimates the remaining life of components from long condition histories [14], is beyond the scope of this work.



TABLE I. TERMINOLOGY AND PROJECT STATUS

Term	Meaning in this paper	Data needed	Project status
Fault detection	Deciding that an abnormal condition is present	Measurements and a decision rule or model	Fixed thresholds implemented; AI detection proposed (Stage 1)
Fault classification	Assigning the condition to a fault type	Labelled examples of each type	Proposed (Stage 2)
Anomaly detection	Flagging departures from expected behaviour; not every anomaly is a fault	Normal-operation data and context	Proposed (Stage 1)
Fault prediction	Forecasting when a gradual condition will cross a fault threshold	Daily indices over weeks of operation	Proposed (Stage 3)
Predictive maintenance	Estimating component remaining life to schedule maintenance	Long histories including degradation and failures	Not addressed

III. RESEARCH GAP AND CONTRIBUTIONS

The learned methods reviewed above [3], [4] are trained and evaluated where good measurements and labelled faults are available. The model-based methods [6]–[8] need accurate references, and the low-cost platforms [12], [13] collect data without interpreting it. Less attention is given in these works to how an AI fault-prediction system should be staged when a very low-cost unit starts with no labelled faults, coarse irradiance context and measurement channels that can fail silently. This paper does not claim that using machine learning for PV fault diagnosis or prediction is new. Its contributions are specific to that setting:

- a staged design in which an unsupervised anomaly model trained on normal data provides detection from the start, and a supervised classifier is added once staged-fault data exist;
- a physics-informed feature set that adds expected-power and expected-temperature residuals to the raw measurements, so the models learn deviations from context rather than raw magnitudes;
- a data-quality gate whose checks are derived from failure modes identified in the actual firmware, so that measurement faults are reported as such instead of being misclassified as PV faults;
- a trend-based prediction stage that estimates the time remaining before gradual faults, such as soiling and thermal stress, reach a fault threshold;
- a fault-class definition and labelling protocol based on safely staged conditions; and
- an evaluation protocol with a time-blocked train–test split, per-class metrics, detection delay and prediction lead time.

IV. SYSTEM ARCHITECTURE

Fig. 1 shows the three layers. Layer 1 (edge acquisition) exists apart from the data-quality flags. Layer 2 (cloud telemetry) carries time-stamped records to a database and attaches weather context. Layer 3 (AI fault prediction) extracts features, runs the three model stages and assigns alert priorities. Only the solid-outlined blocks are present in the current firmware.



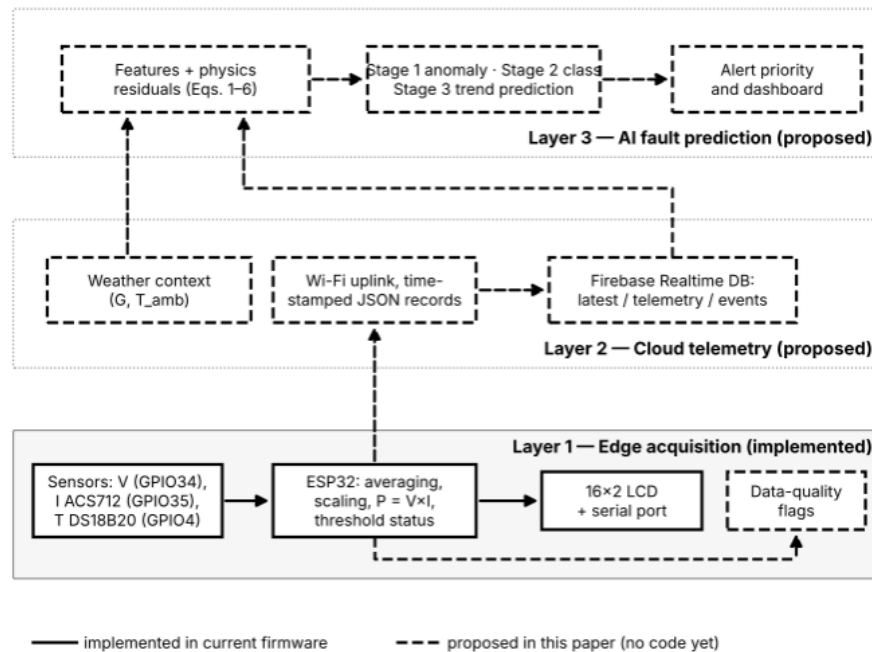


Fig. 1. System architecture. Solid outlines: implemented. Dashed outlines: proposed and not yet coded. Running the models in the cloud rather than on the ESP32 lets them use history and external context, and allows them to be retrained without changing the firmware. Inference at the edge could be considered later, once the model has stabilised.

V. HARDWARE: IMPLEMENTED ACQUISITION UNIT

The acquisition unit (Fig. 2) uses an ESP32 DevKit board with the sensors listed in Table II. Every entry in the table was taken from the firmware source. The firmware averages repeated 12-bit conversions, applies nominal scale factors, computes power as the product of voltage and current, and cycles four display screens of 2.5 s each.

TABLE II. ACQUISITION UNIT AS IMPLEMENTED IN FIRMWARE

Item	Implemented detail
Voltage input	GPIO34; mean of 20 samples at 2 ms spacing; sensor-module factor 5.0; 11 dB attenuation
Current input	GPIO35; ACS712-20A, 0.100 V/A, 2.50 V zero; mean of 50 samples; $\times 1.5$ divider compensation; values below 0.05 A set to zero
Temperature input	DS18B20 on 1-Wire bus at GPIO4; first device only
Display	16x2 LCD, I ² C address 0x27, SDA GPIO21, SCL GPIO22; serial output at 115200 baud
Status rules (priority order)	TEMP FAULT ($T \geq 80^\circ\text{C}$), TEMP WARNING ($T \geq 60^\circ\text{C}$), VOLT FAULT ($V > 25\text{ V}$), LOW VOLTAGE ($V < 5\text{ V}$), otherwise NORMAL
Not present	Wi-Fi, cloud database, weather data, irradiance or ambient sensors, any AI or machine-learning model, alerts



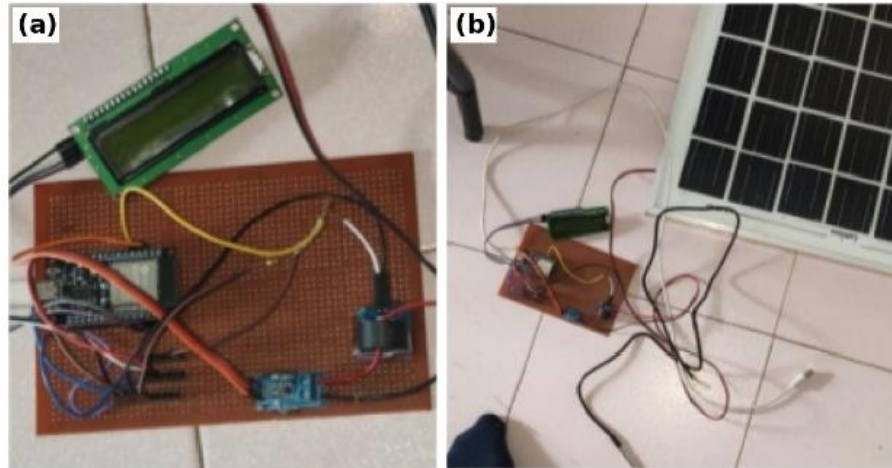


Fig. 2. Implemented acquisition unit: (a) perfboard with ESP32, sensors and LCD; (b) unit connected to the PV panel. Three properties of this unit matter directly for AI training. First, the computed voltage cannot exceed 16.5 V (a 3.3 V full scale multiplied by 5.0), so readings near that value are saturated. Second, the ESP32 ADC is most accurate at 11 dB attenuation for inputs of about 150–2450 mV [15], which corresponds to roughly 0.75–12.25 V of PV voltage here. Third, a missing temperature sensor is reported by the library as -127°C . Records affected by any of these would teach a model wrong patterns, so they are intercepted by the data-quality gate in Section VI.

VI. DATA PIPELINE (PROPOSED)

A. Telemetry

Training data require continuous, time-stamped records. The firmware will be changed in four ways: blocking display delays will be replaced by scheduled tasks, network time synchronisation will be added, a sequence-numbered local buffer will be kept, and Wi-Fi will reconnect automatically. Because GPIO34 and GPIO35 belong to ADC1, they remain usable while Wi-Fi is active, whereas ADC2 does not [15]. Records will be written to a Firebase Realtime Database, which stores data as a JSON tree and synchronises changes to connected clients in real time [16]. Each record will hold the sequence number, device timestamp, V, I, P, T_{panel} , the existing status label and a data-quality bit field. Irradiance G and ambient temperature T_{amb} will be attached in the cloud from a weather service, or from local sensors if they are added later.

B. Data-Quality Gate

Each record is checked before it can be used for training or inference (Table III). Flagged records are stored and reported, but they are never passed to the models.

TABLE III. DATA-QUALITY CHECKS APPLIED BEFORE AI PROCESSING

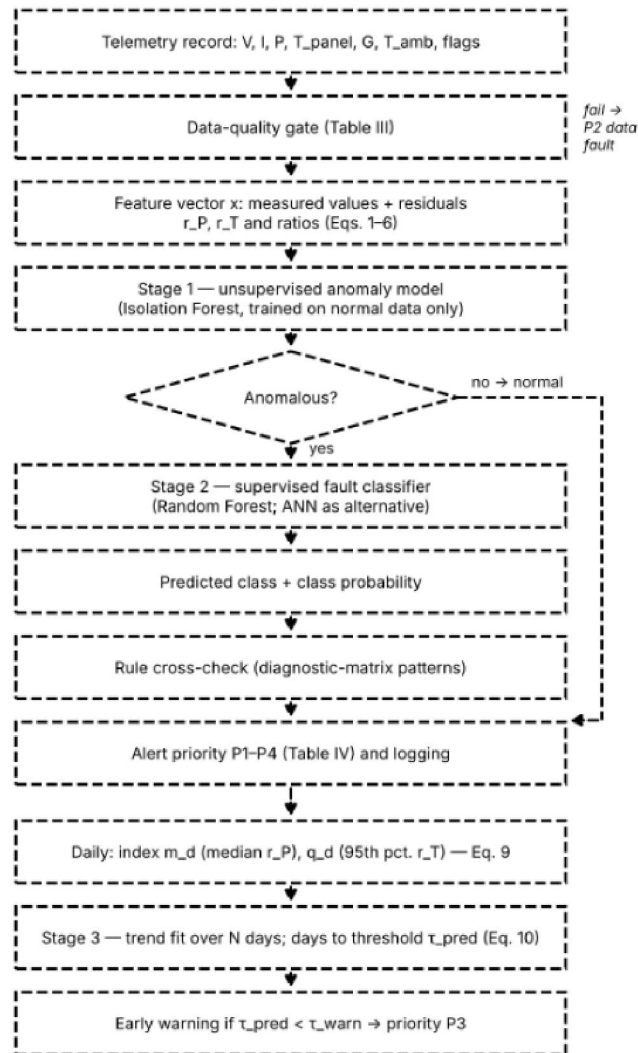
Flag	Condition	Reason
TEMP_SENSOR_LOST	$T_{\text{panel}} = -127^{\circ}\text{C}$	Library value for an absent DS18B20; currently unhandled
V_SATURATED	$V \geq 16.4\text{ V}$	Close to the 16.5 V computational ceiling
V_LOW_CONFIDENCE	$V > 12.25\text{ V}$	ADC pin above the recommended 2450 mV range [15]
I_CLAMPED	Raw current $< -0.05\text{ A}$ before clamping	Offset drift or reversed wiring hidden by the clamp



STALE / GAP	Delayed timestamp or missing sequence number	Late or lost records
NO_CONTEXT	G or T_amb unavailable for the timestamp	Residual features cannot be computed

VII. AI-BASED FAULT DETECTION METHOD (PROPOSED)

Fig. 3 shows the proposed pipeline. Stages 1 and 2 act on individual records; Stage 3 acts on daily summaries. None of it is implemented yet.



Entire pipeline is proposed; no AI model exists in the current firmware.

Fig. 3. Proposed three-stage AI fault-prediction pipeline.

A. Physics-Informed Features

Raw voltage, current and power vary strongly with irradiance, so a model trained on them alone would spend much of its capacity learning the weather. The feature set therefore adds quantities that express how far the panel is from what



its conditions predict. An expected operating temperature follows from a NOCT-based correlation surveyed by Skoplaki and Palyvos [17]:

$$T_{c,est} = T_{amb} + ((NOCT - 20) / 800) \cdot G \quad (1)$$

Expected power follows a simplified linear temperature dependence of PV output [18], evaluated at the measured panel temperature:

$$P_{exp} = P_{STC} \cdot (G / 1000) \cdot [1 + \gamma \cdot (T_{panel} - 25)] \quad (2)$$

Here P_{STC} is the rated power at standard test conditions, γ is the (negative) power temperature coefficient and NOCT is the nominal operating cell temperature, all taken from the module datasheet. **[REQUIRED: P_{STC} , γ , NOCT and $I_{SC,STC}$ of the actual panel]** The residual and ratio features are

$$r_P = (P - P_{exp}) / P_{exp} \quad (\text{only when } G \geq G_{min}) \quad (3)$$

$$r_T = T_{panel} - T_{c,est} \quad (4)$$

$$\rho_I = I / (I_{SC,STC} \cdot G / 1000) \quad (5)$$

$$x = [V, I, P, T_{panel}, G, T_{amb}, r_P, r_T, \rho_I, h] \quad (6)$$

where h is the hour of day and $I_{SC,STC}$ is the short-circuit current at standard test conditions. The irradiance gate G_{min} avoids unstable ratios at low light. A starting value of 150 W/m² is proposed, taken from the irradiance condition in the project's supplied diagnostic matrix; this value has not been independently verified. Because the unit drives a fixed load rather than operating at maximum power, r_P and ρ_I will have a non-zero baseline. The models learn that baseline from normal data, so (2) and (5) do not need to be exact.

B. Stage 1: Unsupervised Anomaly Detection

Stage 1 uses Isolation Forest [10], trained on records from verified normal operation only. The method builds random isolation trees and measures how many splits are needed to separate a point. Points that isolate quickly receive high scores. For a record x , with $E[h(x)]$ the average path length over the trees and n the training sample size, the anomaly score is [10]

$$s(x, n) = 2^{-E[h(x)] / c(n)} \quad (7)$$

where $c(n)$ is the average path length of an unsuccessful search in a binary search tree of n points, used for normalisation. Scores close to 1 indicate anomalies. A record is flagged when s exceeds a threshold s_{th} for k consecutive valid records. This persistence condition suppresses the brief mismatches between coarse weather-service irradiance and fast cloud transients at the panel. **[TO BE SET FROM NORMAL DATA: s_{th} , k , number of trees, sample size]** Stage 1 can run as soon as normal data exist, before any fault has been staged.

C. Stage 2: Supervised Fault Classification

Records flagged by Stage 1 are passed to a classifier that assigns a fault class. A Random Forest [9] is proposed as the primary model. It suits small tabular datasets, gives class probabilities and reports feature importances that can be checked against physical expectations. The predicted class is the majority vote of B trees, and the probability for class c is the fraction of trees voting for it:

$$\hat{p}_c(x) = (1/B) \cdot \sum_{b=1..B} I[h_b(x) = c], \quad \hat{y} = \arg \max_c \hat{p}_c(x) \quad (8)$$

A compact ANN, of the kind reviewed in [3], will be trained on the same features as a comparison. Predictions with low maximum probability will be reported as an "unclassified anomaly" rather than forced into a class. **[TO BE SET: probability cut-off]**

D. Stage 3: Trend-Based Fault Prediction

Stages 1 and 2 react to conditions that are already present. Some PV faults, however, develop gradually: dust accumulates over days, and thermal stress around a weakening cell or connection can grow over weeks. For these conditions the framework adds a prediction stage that forecasts when the condition will become a fault. For each day d , a daily performance index is formed as the median power residual over that day's valid, irradiance-gated records:



$$m_d = \text{med} \{ r_{p,i} : \text{record } i \text{ on day } d, G_i \geq G_{\min}, \text{ no quality flag} \} \quad (9)$$

A thermal index q_d is formed in the same way from the daily 95th percentile of r_T . Using the median makes m_d insensitive to short cloud transients, so it tracks slow change in the panel rather than the weather. Over a sliding window of the last N days, a straight line $m_d \approx a + b \cdot d$ is fitted by least squares. If the slope b is negative, the predicted number of days until the index reaches a fault threshold m_{th} is

$$\hat{\tau} = (m_{th} - \hat{m}_D) / \hat{b}, \quad \hat{b} < 0 \quad (10)$$

where $\hat{m}_D$ is the fitted index on the current day D . An early-warning alert is raised when $\hat{\tau}$ falls below a warning horizon τ_{warn} , together with the fitted slope and its confidence interval so the user can judge the forecast. The same procedure applied to q_d predicts thermal-stress faults. **[TO BE SET FROM DATA: $N, m_{th}, q_{th}, \tau_{\text{warn}}$]** A linear trend is deliberately simple. Once enough history exists, it can be compared with learned forecasters, such as a regression model on lagged daily indices. Cleaning events reset the soiling trend and are logged so that the window restarts after each one.

Stage 3 has clear limits. It can predict only faults that announce themselves through a gradual drift in the indices. Abrupt faults such as an open circuit, a short circuit or a sudden connector failure give no advance trend, so they are detected by Stages 1 and 2 and the rules, not predicted.

E. Rule Cross-Check and Alert Priority

Learned predictions are cross-checked against simple electrical patterns from the supplied diagnostic matrix: an open circuit at $V \geq 17.5$ V with $I < 0.1$ A, a short circuit at $V < 2.5$ V with $I \geq 4.0$ A, and standby at $V < 2.0$ V with $I < 0.1$ A. These values still need to be verified for the actual panel. The open-circuit pattern is also unreachable until the voltage scaling is corrected. Disagreement between a rule and the classifier is logged for review rather than silently resolved. Table IV assigns priorities.

TABLE IV. PROPOSED ALERT PRIORITY SCHEME

Level	Trigger	Action
P1 Critical	$T_{\text{panel}} \geq 80$ °C (existing rule), or short-circuit class or pattern	Immediate notification; event logged
P2 High	Any data-quality flag; $T_{\text{panel}} \geq 60$ °C; open-circuit class or pattern	Notification; AI scoring suspended for flagged records
P3 Medium	Stage 1 anomaly with a Stage 2 class (shading, soiling, overheating), an unclassified anomaly, or a Stage 3 prediction with $\hat{\tau} < \tau_{\text{warn}}$	Dashboard alert, daily summary and predicted days to threshold
P4 Info	Standby, $G < G_{\min}$, or a single-record anomaly without persistence	Logged only

VIII. DATASET AND LABELLING PLAN

No dataset exists yet. Table V defines the target classes and how examples of each will be obtained. Faults are staged only where this is safe for a small panel and the measurement hardware. Conditions that cannot be staged safely will be supplemented by simulation and clearly marked as simulated in any results.

TABLE V. TARGET CLASSES AND DATA COLLECTION PLAN

Class	How examples are obtained	Expected signature (to be confirmed)	Label source
Normal	Unobstructed operation across clear and cloudy days	r_P near its learned baseline	Operator log



Partial shading	Opaque cover over a recorded fraction of cells at $G \geq G_{\min}$	Fall in I , ρ_I and r_P	Staging log with start and end times
Soiling	Uniform translucent film or dust layer on the glass	Gradual, moderate fall in r_P	Staging log
Open circuit	Load disconnected	$I \approx 0$ with raised V	Staging log
Short circuit / low voltage	Simulation, or a controlled low-resistance load within sensor rating	Low V with high I	Simulation flag or staging log
Overheating	Not staged; recorded opportunistically, otherwise simulated	High r_T	Rule-confirmed events
Gradual soiling (Stage 3)	Panel left uncleaned for several weeks; cleaning dates logged	Steady downward trend in m_d	Cleaning log and measured trend
Sensor fault	DS18B20 unplugged	TEMP_SENSOR_LOST (handled by the gate, not the model)	Gate flag

Each staging event will be logged with its start and end times, and records inside the window will receive the event label. Records within one minute of the boundaries will be excluded as transitional. Adjacent records are strongly correlated, so a random split would place near-duplicates in both training and test sets and inflate the scores. The data will therefore be split by whole days, with complete days held out for testing. Class imbalance, with normal records expected to dominate, will be handled with class weights rather than synthetic oversampling. Stage 3 needs a different kind of data: several continuous weeks without cleaning, so that a natural soiling trend can be observed and the date on which m_d actually crosses m_{th} can be recorded. **[TO BE COLLECTED: number of days and records per class; length of the uncleaned period]**

IX. EVALUATION PROTOCOL

For each fault class k , precision P_k , recall R_k and $F1_k = 2P_kR_k/(P_k+R_k)$ will be computed from the confusion matrix on held-out days [19]. Because the classes are imbalanced, the summary figure will be the macro-averaged F1-score:

$$F1_{\text{macro}} = (1/K) \cdot \sum_{k=1..K} F1_k \quad (11)$$

Overall accuracy will be reported but not used as the main figure, since a model that always predicts the normal class would score highly on it. Stage 1 will be assessed separately as a binary detector by its recall on staged events and its false-alarm rate per day of normal operation. Timeliness will be measured as the detection delay

$$D = t_{\text{alert}} - t_{\text{onset}} \quad (12)$$

for each staged event. Stage 3 will be evaluated by its prediction error, the absolute difference in days between the predicted and actual threshold-crossing dates, and by its lead time, the number of days between the first early warning and the actual crossing. All models will be compared with the existing fixed-threshold rules on the same test days, so that any benefit from AI is measured rather than assumed. Table VI lists the evaluations; no results are available yet.

TABLE VI. EVALUATIONS AND PENDING RESULTS

Evaluation	Procedure	Result
Sensor calibration	Compare V , I and T_{panel} with reference instruments	[PENDING]
Stage 1 detection	Recall on staged events; false alarms per normal day	[PENDING]



Stage 2 classification	Per-class precision, recall and F1; macro-F1; confusion matrix	[PENDING]
ANN comparison	Same split and features as the Random Forest	[PENDING]
Baseline comparison	Fixed-threshold rules on the same test days	[PENDING]
Detection delay	D for each staged event (Eq. 12)	[PENDING: median and maximum]
Stage 3 prediction error	predicted – actual threshold-crossing date (days)	[PENDING]
Stage 3 lead time	Days between first early warning and actual crossing	[PENDING]
Feature importance	Random Forest importances checked against physical expectations	[PENDING]

X. DISCUSSION AND LIMITATIONS

The framework's dependence on environmental context is its largest weakness. Weather-service irradiance is spatially and temporally coarse, which will add noise to r_p and p_l . The persistence condition reduces false alarms caused by this mismatch but cannot recover short events. A small local irradiance reference would materially improve the features.

A single small panel limits both the variety and the realism of faults. Line-to-line and line-to-ground faults in multi-string arrays cannot be reproduced at all. Shading and soiling on one module may not represent array behaviour, and simulated classes may not match field signatures. Any classifier trained here should therefore be described as validated only for the conditions tested. Prediction is limited further: Stage 3 forecasts only gradual conditions, needs weeks of data before its first forecast, and its linear trend may not hold if soiling is interrupted by rain or wind.

The acquisition channels have not been calibrated, and the unit still has the scaling and robustness issues noted in Section V. The data-quality gate prevents the worst of these from reaching the models but does not make uncalibrated values accurate. Finally, this paper specifies a method; it does not demonstrate one. Its claims will stand or fall on the results in Table VI.

XI. CONCLUSION AND FUTURE SCOPE

This paper has presented an ESP32 acquisition unit for a small PV system and a proposed three-stage AI framework for predicting, detecting and classifying faults from its data. The unit, which is implemented, measures voltage, current and panel temperature and applies fixed local thresholds. The proposed framework adds a data-quality gate built from identified hardware failure modes, physics-informed residual features, an Isolation Forest that needs only normal data, a Random Forest classifier trained on staged faults, and a trend-based stage that predicts the days remaining before gradual faults such as soiling and thermal stress cross a fault threshold. It also specifies a labelling plan, a day-blocked evaluation with macro-F1, detection delay, prediction error and lead time, and a direct comparison with the existing threshold rules. Prediction is claimed only for gradual faults that show a measurable trend; abrupt faults are detected, not predicted.

The next steps are to calibrate the sensors and correct the voltage scaling, implement non-blocking firmware with cloud upload, collect normal-operation data to train Stage 1, stage the faults in Table V to build the Stage 2 dataset, record an extended uncleaned period for Stage 3, and then complete the evaluations in Table VI. Extending prediction towards remaining-life estimation for components could be considered once long-term operating histories exist.



ACKNOWLEDGMENT

The authors thank their project guide, Dr. R. A. Keswani, their co-guide, Prof. K. G. Gaurkhede, and the Department of Electrical Engineering, Priyadarshini College of Engineering, Nagpur, for their support.

REFERENCES

- [1] A. Mellit, G. M. Tina, and S. A. Kalogirou, "Fault detection and diagnosis methods for photovoltaic systems: A review," *Renew. Sustain. Energy Rev.*, vol. 91, pp. 1–17, 2018, doi: 10.1016/j.rser.2018.03.062.
- [2] Y. Zhao, J.-F. de Palma, J. Mosesian, R. Lyons, and B. Lehman, "Line–line fault analysis and protection challenges in solar photovoltaic arrays," *IEEE Trans. Ind. Electron.*, vol. 60, no. 9, pp. 3784–3795, 2013, doi: 10.1109/TIE.2012.2205355.
- [3] B. Li, C. Delpha, D. Diallo, and A. Migan-Dubois, "Application of artificial neural networks to photovoltaic fault detection and diagnosis: A review," *Renew. Sustain. Energy Rev.*, vol. 138, Art. no. 110512, 2021, doi: 10.1016/j.rser.2020.110512.
- [4] W. Chine, A. Mellit, V. Lughi, A. Malek, G. Sulligoi, and A. Massi Pavan, "A novel fault diagnosis technique for photovoltaic systems based on artificial neural networks," *Renew. Energy*, vol. 90, pp. 501–512, 2016, doi: 10.1016/j.renene.2016.01.036.
- [5] A. Livera, M. Theristis, G. Makrides, and G. E. Georghiou, "Recent advances in failure diagnosis techniques based on performance data analysis for grid-connected photovoltaic systems," *Renew. Energy*, vol. 133, pp. 126–143, 2019, doi: 10.1016/j.renene.2018.09.101.
- [6] R. Platon, J. Martel, N. Woodruff, and T. Y. Chau, "Online fault detection in PV systems," *IEEE Trans. Sustain. Energy*, vol. 6, no. 4, pp. 1200–1207, 2015, doi: 10.1109/TSTE.2015.2421447.
- [7] S. Silvestre, A. Chouder, and E. Karatepe, "Automatic fault detection in grid connected PV systems," *Solar Energy*, vol. 94, pp. 119–127, 2013, doi: 10.1016/j.solener.2013.05.001.
- [8] F. Harrou, Y. Sun, B. Taghezouit, A. Saidi, and M.-E. Hamlati, "Reliable fault detection and diagnosis of photovoltaic systems based on statistical monitoring approaches," *Renew. Energy*, vol. 116, pp. 22–37, 2018, doi: 10.1016/j.renene.2017.09.048.
- [9] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001, doi: 10.1023/A:1010933404324.
- [10] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *Proc. 8th IEEE Int. Conf. Data Mining (ICDM)*, 2008, pp. 413–422, doi: 10.1109/ICDM.2008.17.
- [11] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, pp. 1–58, 2009, doi: 10.1145/1541880.1541882.
- [12] A. López-Vargas, M. Fuentes, and M. Vivar, "IoT application for real-time monitoring of solar home systems based on Arduino™ with 3G connectivity," *IEEE Sensors J.*, vol. 19, no. 2, pp. 679–691, 2019, doi: 10.1109/JSEN.2018.2876635.
- [13] R. I. S. Pereira, I. M. Dupont, P. C. M. Carvalho, and S. C. S. Jucá, "IoT embedded Linux system based on Raspberry Pi applied to real-time cloud monitoring of a decentralized photovoltaic plant," *Measurement*, vol. 114, pp. 286–297, 2018, doi: 10.1016/j.measurement.2017.09.033.
- [14] T. P. Carvalho, F. A. A. M. N. Soares, R. Vita, R. da P. Francisco, J. P. Basto, and S. G. S. Alcalá, "A systematic literature review of machine learning methods applied to predictive maintenance," *Comput. Ind. Eng.*, vol. 137, Art. no. 106024, 2019, doi: 10.1016/j.cie.2019.106024.
- [15] Espressif Systems, "Analog to Digital Converter," ESP-IDF Programming Guide, v3.1.5. [Online]. Available: <https://docs.espressif.com/projects/esp-idf/en/v3.1.5/api-reference/peripherals/adc.html> (accessed Oct. 18, 2026).
- [16] Google, "Firebase Realtime Database," Firebase documentation. [Online]. Available: <https://firebase.google.com/docs/database> (accessed Oct. 18, 2026).
- [17] E. Skoplaki and J. A. Palyvos, "Operating temperature of photovoltaic modules: A survey of pertinent correlations," *Renew. Energy*, vol. 34, no. 1, pp. 23–29, 2009, doi: 10.1016/j.renene.2008.04.009.



[18] D. L. Evans, "Simplified method for predicting photovoltaic array output," Solar Energy, vol. 27, no. 6, pp. 555–560, 1981, doi: 10.1016/0038-092X(81)90051-7.

[19] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," Inf. Process. Manage., vol. 45, no. 4, pp. 427–437, 2009, doi: 10.1016/j.ipm.2009.03.002.

