



International Journal of Research Publication and Reviews

Journal Homepage: www.ijrpr.com ISSN 2582-7421

AI Driven Solar Microgrid With Predictive Load Management

Prof Atul Malode, Prathamesh Bhagwat, Harsh Meshram, Bhavik Pipare, Aniket Bhalkar

Department of Industrial Internet of Things, Priyadarshini College of Engineering, Nagpur, Maharashtra, India

ABSTRACT

Such small systems, which are solar-assisted and battery-backed, must account for limited stored energy, variable availability of renewable energy, and loads of varying priority. This paper introduces a prototype of a powerfully simple, AI-assisted solar microgrid for predictive load management, derived from the technical implementation TinyML-Based Weather-Aware Smart Load Management System. The solar microgrid is constituted of an ESP32 embedded controller, a 3S2P battery pack, voltage-divider circuits, a current and power monitor INA226, a temperature sensor DS18B20, Wi-Fi weather-data acquisition, an OLED display, an IP-based web dashboard, a 100W DC-to-AC inverter, and three sets of relays for load control. The AI-assisted load management approach employs a TinyML classifier on ten electrical, thermal, weather, and time features to determine the energy condition of the load as being either HIGH, MEDIUM, LOW, or CRITICAL. Load states can range from Normal, Emergency, and Critical loads, while voltage and temperature rules independent of the model can take precedence. The project report proposes a synthetic dataset of approximately 5,000 samples, demonstrating a validation accuracy of 95.9% and a test accuracy of 96.08%. Due to the synthetic nature of the dataset, the accuracy figure is an approximation of the model's success on the prepared dataset and does not predict field performance. The paper offers an overview of the architecture, technologies, equations, test conditions, and limitations of the prototype.

Index Terms— Battery management, embedded artificial intelligence, energy management, ESP32, Internet of Things, microgrid, predictive load management, TinyML.

I. INTRODUCTION

Solar assisted microgrids and power systems with battery backup have variable loads and variable supplies of energy. Cloud, rain, and other weather conditions will affect the amount of solar energy and the battery has a limited amount of energy available. In a bad energy period, if everything is running constantly, there will only be a limited time before the battery runs out of energy. The fixed set point voltage may be adequate to protect the battery from damage but it doesn't necessarily account for the effects of current load, battery temperature, weather, and duration.

The internal technical implementation on which this paper is based is referred to as "TinyML-Based Weather-Aware Smart Load Management System" for which the paper title is "AI Driven Solar Microgrid With Predictive Load Management" and the internal technical title remains as described in the technical implementation reference. An example of combining sensor data with weather data, local ML inference, and load switching based on load priority on an ESP32.

The general objective is to monitor the present energy condition and to sequential load shed loads when needed. Typically, normal loads are sequentially shed as the energy condition decreases, while critical loads are shed once the energy condition reaches a backup setting. A integrated prototype architecture, a four state classifier, small light weight device and deterministic safety overrides, and monitoring networked and local are described.

II. RELATED WORK AND RESEARCH GAP

Intelligent energy management: priority load shedding, PV prediction, battery-state estimation (data-driven), IoT-based monitoring, edge ML, Rathinam and Dayalan on priority load shedding with and without renewable-energy generation [1], Maitanova et al. on solar-power prediction based on free-to-access weather reports [2], Ng et al. on data-driven prediction of battery state of charge and state of health [3], and Sui et al. on machine-learning-driven lithium-ion battery state-of-health prediction [4].

There had been similar previous work on processing the machine-learning inference on light-weighted microcontrollers instead of each decision on distributed cloud [5]. In the home energy management and IoT-scheduling work, attempts were made to explore the power of distributed optimisation and control of distributed energy resources [6]-[10]. These works also pave the way for real-time sensing, forecasting, and load prioritising.

The missing piece of the implementation puzzle that most battery-inverter systems haven't got their heads around yet - that state-of-charge estimates don't all require load shedding, and that a cloud-aware system needs to be persistent. So we create a local classifier based on some combination of battery, load, weather, and time, then connect it to relay commands. Weather is a proxy for solar availability - but not solar output.

III. PROPOSED SYSTEM ARCHITECTURE

All has gained. The ESP32 controls and monitors it, takes data, makes features and runs TinyML inference at home. The power is fed from a 3S2P lithium-ion pack 11.1 V nominal and 4 Ah rated battery pack, converted via a 100 W inverter (10.75 V input) into 230 V for the loads, with the battery pack's voltage measured using a 100 k Ω /10 k Ω divider (solar voltage is measured via a similar divider). Measures of current and power are using the INA226, with battery temperature being measured with a DS18B20.

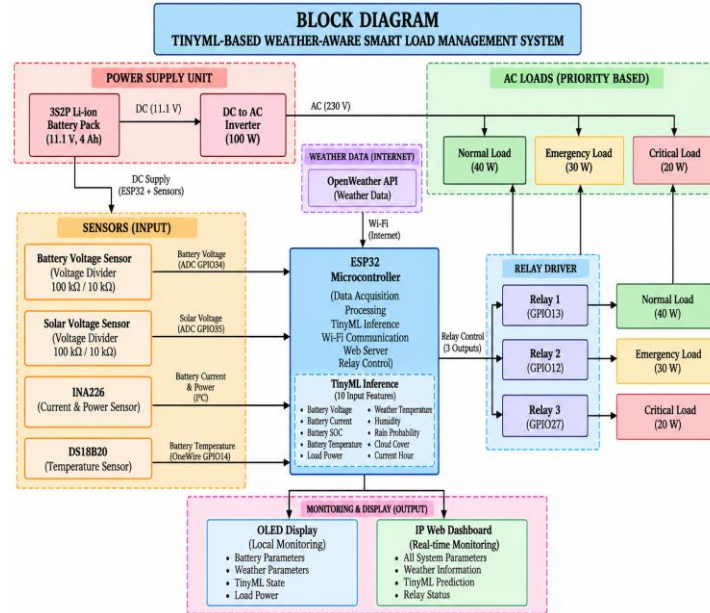


Fig. 1. Overall system block diagram: battery/inverter, sensors, ESP32/TinyML, weather API, relays, loads, OLED and dashboard.

Weather temperature, humidity, chance of rain and cloud cover is collected by the ESP32 via Wi-Fi and the Open Weather service. The classifier has 10 ranked inputs: Battery voltage, battery current, battery SOC, battery temperature, load power, weather temperature, humidity, chance of rain, cloud cover and hour. The predicted class is displayed on the local device and on the IP dashboard. There are 3 relay outputs to switch the Normal, Emergency and Critical load groups.

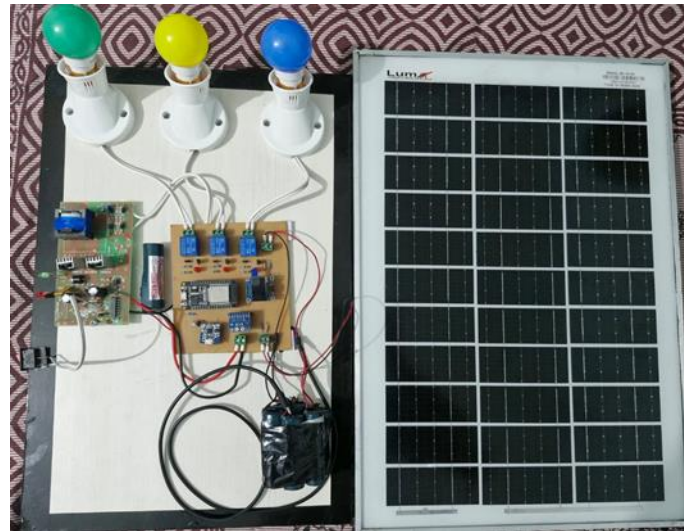


Fig. 2. Photograph of the assembled hardware prototype showing ESP32, battery, sensors, inverter, relays and wiring

IV. HARDWARE, TECHNOLOGIES AND TOOLS

The hardware and software components listed for the prototype are Table I. These components are arranged as the sensing, processing, communicating, actuating, and user-interface levels. The ratings and resistor values are taken from the project report and are left to be checked against the final physical product if this were to be published.

Component / technology	Purpose in the prototype
ESP32 microcontroller	Sensor acquisition, Wi-Fi, local TinyML inference, relay control and dashboard hosting
3S2P Li-ion battery, 11.1 V nominal, 4 Ah	Stores energy for the prototype
100 k Ω and 10 k Ω voltage-divider resistors	Battery and solar voltage sensing at ESP32 ADC inputs
INA226 current/power monitor	Measures battery-side current and power
DS18B20 digital temperature sensor	Measures battery temperature
100 W DC-to-AC inverter	Converts battery DC to AC for loads
Three relay channels	Switch Normal, Emergency and Critical load groups
OLED display	Local status and measurement display
Wi-Fi and OpenWeather API	Acquires weather data for model features
TinyML / Edge Impulse workflow	Dataset preparation, model training and embedded inference
IP-based web dashboard	Displays electrical, weather, prediction and relay information

The system uses embedded C / C++ style microcontroller firmware, sensor libraries, Wi-Fi, HTTP weather fetch, the trained TinyML model, and a web interface. The report mentions Edge Impulse as the model-development workflow and Open Weather as the weather source. An exact version of the firmware, set of libraries, revision of the board, the API endpoint used, and compilation flags should all be copied from the final build.

Software/tool	Use
Arduino/ESP32 development environment (verify exact IDE/version)	Firmware development and deployment
Edge Impulse	Dataset handling, model training and deployment workflow
OpenWeather API	Weather data retrieval
Web browser on local network	Access to the ESP32 IP dashboard
Serial monitor / logs	Debugging sensor values, network responses and inference output

V. MATHEMATICAL MODEL AND FEATURE PROCESSING

As the name, the battery controller is assigned with such many electrical and weather parameters and the output they can hold under actual and nominal situations. Electrical Parameters Vb- battery voltage Ib- battery current SOC- estimated state of charge Tb- battery temperature PL- load power Weather Parameters Tw- weather temperature H- humidity R- rain probability C- cloud cover H- hour of day Input feature vector to the classifier - Vb, Ib, SOC, Tb, PL, Tw, H, R, C, h (Note:- order of features and unit should be same with the unit used while training the model)

SOC calculated from BVolt by project report (pick your references): The linear estimate is:

$$1 = \text{clamp} [(V_b - V_{\text{empty}}) / (V_{\text{full}} - V_{\text{empty}}) 100, 0, 100]$$

Where Vfull=11.60V, Vempty=9.60V in our implementation. This was only in terms of the cell voltage and was only an estimate- we would not use this as a fuel gauge if it were outside of the same temperature range or with a different load.

Relation by following the voltage-divider is:

$$V_{\text{ADC}} = V_{\text{in}} \frac{R_2}{R_1 + R_2} \quad (2)$$

For example, if R1=100k and R2=10k, then the ideal divider output VADC=Vin/11 – as before – can be used before component tolerances and ADC calibration. Do not overload the ADC input.

Electrical power is related to current and voltage as:

$$P = V \cdot I \quad (3)$$

The power loss will be not as the AC side, there is some power in the conversion of the inverter efficiency. If the power is from the current and the voltage, the calibration and the sign convention of the sensor shall be described explicitly.

The model then computes a score or a probability for each class and predicts the class of a sample as:

(4) $y_{\text{atte}} = \arg \max_k p(y=k | x)$ where; $k \in \{\text{HIGH, MEDIUM, LOW, CRITICAL}\}$

The controller then processes the state estimate for the individual relay outputs; each are safety checked individually. These equations assume the following; the project report has not included a measured solar-generation forecast or a calibrated electrochemical -SOC- model of the cell.

VI. TINYML TRAINING AND LOAD CONTROL

The report describes approximately 5,000 synthetically generated samples, balanced across four energy-state classes with about 1,250 samples per class. The reported training/testing partition contains approximately 4,006 training samples and 994 test samples. A compact neural network with hidden layers of approximately 20 and 10 neurons and a four-class output layer was trained for about 30 cycles with a learning rate of 0.0005. The model is intended for inference on the ESP32.

Model setting	Reported value
Dataset size	Approximately 5,000 synthetic samples
Classes	HIGH, MEDIUM, LOW, CRITICAL
Samples per class	Approximately 1,250
Train/test split	Approximately 4,006 / 994
Network structure	Two hidden layers: approximately 20 and 10 neurons; 4 output classes
Training cycles	Approximately 30
Learning rate	0.0005
Inference platform	ESP32

At runtime, the ESP32 gathers sensor measurements, retrieves the current weather features, performs SOC estimation, and constructs the input vector and invokes the model. The class with the highest score is assumed to be the predicted energy state. A technique for external data collection from the weather service and local computation of the core classifier is also shown.

Predicted energy state	Normal load	Emergency load	Critical load	Control intent
HIGH	ON	ON	ON	Normal operation
MEDIUM	ON	ON	ON	Continue operation while monitoring
LOW	OFF	ON	ON	Shed lower-priority demand
CRITICAL	OFF	OFF	ON	Retain only the critical group

State to relay mapping is performed after individual safetychecks. If battery voltage or temperature condition is serious, then normal model based control is skipped to the default protection logic. This is an important distinction because classifier only do not protect battery. The system also reports to OLED and IP dashboard the value of battery, weather, class, confidence, load power and relay.

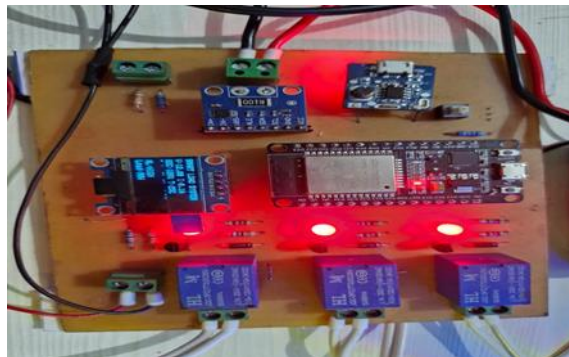


Fig. 3. Photograph of the relay driver and the three connected load groups, with Normal/Emergency/Critical labels.

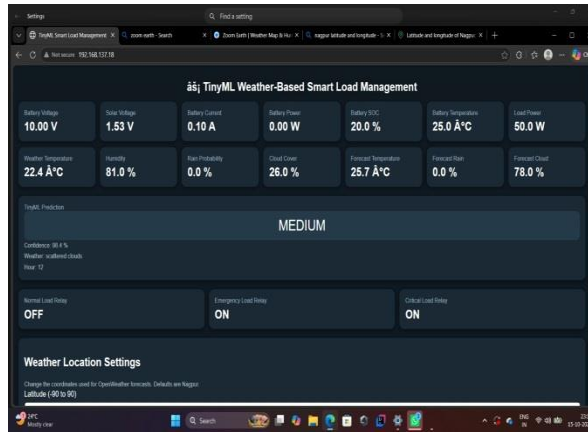


Fig. 4 Screenshot of the IP dashboard showing battery, weather, TinyML class, confidence and relay status.

VII. TEST CONDITIONS AND EVALUATION

As noted in the project report, software/model evaluation and prototype functional-level checks are documented. Unlike the conditions in the project report, which are listed in Table II, the fields in the table below are designed to be completed by recording actual test logs. Synthetic model metrics cannot be displayed as field validation.

Test / condition	Procedure or condition	Expected/recorded outcome
Model validation	Validation split from synthetic dataset	Reported validation accuracy: approximately 95.9%
Independent model test	Approximately 994 test samples	Reported test accuracy: 96.08%
HIGH/MEDIUM state	Model predicts HIGH or MEDIUM and safety limits are normal	All three load groups may remain ON
LOW state	Model predicts LOW and safety limits are normal	Normal OFF; Emergency and Critical ON
CRITICAL state	Model predicts CRITICAL and safety limits are normal	Normal and Emergency OFF; Critical ON
Low-battery protection	Apply the configured critical-voltage condition	Safety logic should override normal model decision; record measured threshold
High-temperature protection	Apply/observe configured critical-temperature condition	Safety logic should override normal model decision; record threshold
Weather/network availability	Compare operation with valid and unavailable weather data	Record API success/failure and fallback behaviour
Dashboard and OLED	Observe updates during sensor and relay changes	Record update latency and any missing values

The exact trip settings, sample interval, refresh interval, calibration, relay switching delay and test duration for each test must be shown in the final experimental version for reproducibility. If not measured, they should be shown as not recorded and not as recorded.

Metric	Reported result	Interpretation
Validation accuracy	≈95.9%	On the prepared synthetic validation set
Independent test accuracy	96.08%	On the prepared synthetic test set
Class-wise correct rate: HIGH	≈98%	Reported class-wise value
Class-wise correct rate: MEDIUM	≈95.6%	Reported class-wise value

Metric	Reported result	Interpretation
Class-wise correct rate: LOW	$\approx 95.2\%$	Reported class-wise value
Class-wise correct rate: CRITICAL	$\approx 95.6\%$	Reported class-wise value
AUC	≈ 1.00 reported	Dataset-specific; include the averaging/method details if available

An example inference (shown on the right) where MEDIUM was selected with a probability of about 0.89 is provided for illustration of the model interface, however this does not constitute evidence that such a high confidence level is general to all operating conditions. Synthetic labels and synthetic feature distributions can encode oversimplified assumptions so a high test score can still correspond to poor generalization to batteries in the field, sensor noise, fluctuating loads, and weather.

VIII. LOAD POWER ANALYSIS

For example, the project report states the loads as follows: Normal load- 40 W, Emergency load- 30 W, Critical load- 20 W. That means there is a total of 90 W at the inverter at all times, which is less than the nominal rating of the inverter (100W without considering startup surges, inverter inefficiencies, and power factor). The calculations of the nominal load are given in Table III.

Operating state	Loads connected	Nominal connected load	Reduction vs. all ON
HIGH	Normal + Emergency + Critical	$40 + 30 + 20 = 90 \text{ W}$	0 W
MEDIUM	Normal + Emergency + Critical	90 W	0 W
LOW	Emergency + Critical	$30 + 20 = 50 \text{ W}$	40 W
CRITICAL	Critical only	20 W	70 W

The load shedding as installed then drops connected nominal demand by 40 W in LOW, and 70 W in CRITICAL mode from 90 W all-load demand. That is an arithmetical number derived on given load ratings, and does not imply energy conservation, it does not translate into backup time, which will be shorter with the variable battery charge state, power factor, load particularities and inverter efficiency.

Estimate ideal battery energy: $E_{\text{nominal}} = V_{\text{nominal}} \text{ Capacity} = 11.1 \text{ V} \times 4 \text{ Ah} = 44.4 \text{ Wh}$. (5)

The 44.4 Wh figure is a nominal energy figure. Usable energy is lower, taking account of conversion losses, discharge limits, battery characteristics, nominal and actual voltage. This should not be taken as a field run time claim, there has been no measured discharge testing.

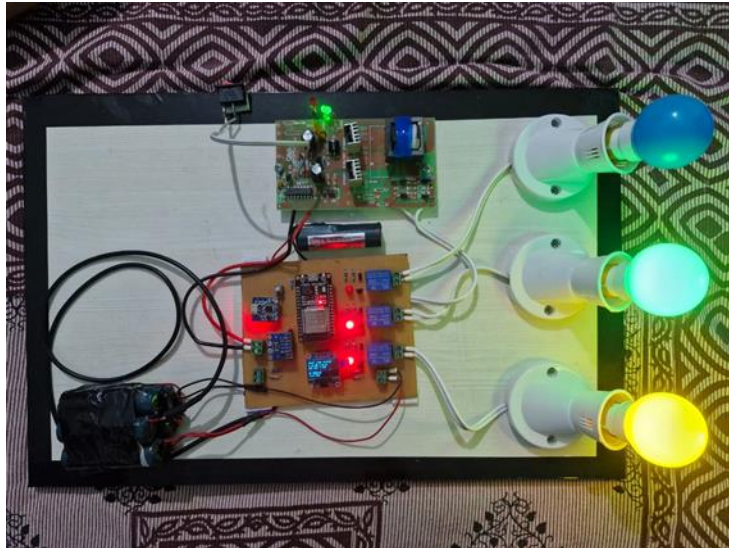


Fig. 6. Photograph of the assembled load test setup, with load ratings and safe wiring visible.

IX. DISCUSSION, LIMITATIONS AND SAFETY

This architecture enables 4 of 5 benefits of small microgrid: multi-parameter sensing, weather aware input parameters, local inference, load-priority control. Local inference is the ability to infer a classification for each individual measurement without always having to send the measurement to a remote server to be classified, and dashboard displays the state of the system to human user. Priority scheme produces a simple, intuitive control response under energy constrained operation.

Simulation data for training (some trained level may not be able to be used in real hardware and the field for this application) State-of-charge based on voltage is load, temperature, chemistry and ageing sensitive - Coulomb count or battery model calibration preferable Weather forecast data is not a measurement of PV power and / or solar irradiance Weather data may become out-dated before network outage Relay and inverter ratings, isolation, fusing, battery-management circuitry, enclosure design.

This inverter 100w and battery 3s2p now is a prototype system representative only and the full microgrid controller (as follows) is not yet run for the microgrid system with industrial size, why use this set. When operate on AC side, note all relay, insulation, fuse and other requirements are appropriate rated and be cautious for the safety. No replacing actual battery protection property of hardware as model output.

X. CONCLUSION AND FUTURE WORK

The paper has presented: AI Driven Solar Microgrid With Predictive Load Management on: TinyML-Based Weather-Aware Smart Load Management System The prototype consists of: ESP32 with battery and solar voltage sensing, INA226 current / power measurement, DS18B20 temperature sensing, weather data, four class TinyML model, relay based priority management, oled display, IP based dashboard. The control policy implemented on the prototype does not de-schedule any load to a lower priority state by de-escalating the Normal load to LOW state and keeping the Critical load only to CRITICAL state.

Validation of model accuracy - 95.9% validation accuracy and 96.08% test accuracy was done on synthetic data and has to be validated on real data. Also validation of the advantages of prescriptive choices by the prototype has to be done - real-time data (weather/load labels), real backup time/energy consumption measurement, real-measurement of SOC estimation gain, sensor/APIs failures testing, inference time/memory footprint reporting, protection and switching hardware testing.

ACKNOWLEDGMENT

We express our deep sense of gratitude to our guide Prof. Atul Malode of Priyadarshini College of Engineering for his help in providing reviews, ideas and suggestions that improved our project work. Because of inspiring and guidance provided by him, the project work is reached up to success level. It has been privilege and pleasure working under his guidance.

REFERENCES

- [1] R. Rathinam and S. Dayalan, "Demand Side Load Management by Using Priority Based Load Shedding Algorithm with and without Renewable Energy Generation," 2019.
- [2] N. Maitanova, J.-S. Telle, B. Hanke, M. Grotke, T. Schmidt, K. von Maydell, and C. Agert, "A Machine Learning Approach to Low-Cost Photovoltaic Power Prediction Based on Publicly Available Weather Reports," 2020.
- [3] M.-F. Ng, J. Zhao, Q. Yan, G. J. Conduit, and Z. W. Seh, "Predicting the State of Charge and Health of Batteries Using Data-Driven Machine Learning," 2020.
- [4] X. Sui et al., "A Review of Non-Probabilistic Machine Learning-Based State of Health Estimation Techniques for Lithium-Ion Battery," 2021.
- [5] A. Moin, M. Challenger, A. Badii, and S. Günnemann, "Supporting AI Engineering on the IoT Edge Through Model-Driven TinyML," 2022.
- [6] "Home Energy Management System Based on Genetic Algorithm for Load Scheduling: A Case Study Based on Real Life Consumption Data," 2023.
- [7] "Towards Energy-Aware TinyML on Battery-Less IoT Devices," 2023.
- [8] A. Ajitha et al., "Design and Implementation of Smart Home Energy Management System for Indian Residential Sector," 2024.
- [9] S. Panda et al., "Priority-Based Scheduling in Residential Energy Management Systems Integrated with Renewable Sources Using Adaptive Salp Swarm Algorithm," 2024.
- [10] B. K. Jha, A. Tiwari, R. B. Kuhada, and N. M. Pindoriya, "IoT-Enabled Smart Energy Management Device for Optimal Scheduling of Distributed Energy Resources," 2024.
- [11] Espressif Systems, "ESP32 Series Datasheet and Technical Documentation."
- [12] Analog Devices, "INA226 High-Side or Low-Side Measurement, Bidirectional Current/Power Monitor: Data Sheet."
- [13] Maxim Integrated, "DS18B20 Programmable Resolution 1-Wire Digital Thermometer: Data Sheet."
- [14] Edge Impulse, "Embedded Machine Learning Documentation."
- [15] OpenWeather, "Weather API Documentation."