WWW.IJRPR.COM

International Journal of Research Publication and Reviews

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

EV Battery Management System Using ML

Prof. D.A.Kapgate[1], Yugdip Gadpayle[2], Shashank Sindpure[3], Om Bansod [4], Dipak Kharabe[5]

[1] Priyadarshini College of Engineering Department of Electronics & Communication Engineering Nagpur, India
d.a.kapgate@gmail.com
[2] Priyadarshini College of Engineering Department of Electronis & Communication Engineering Nagpur, India
yugdipgadpayle202@gmail.com
[3] Priyadarshini College of Engineering Department of Electronics & Communication Engineering Nagpur, India
shashanksindpure27@gmail.com
[4] Priyadarshini College of Engineering Department of Electronics & Communication Engineering Nagpur, India
ombansod45@gmail.com
[5] Priyadarshini College of Engineering Department of Electronics & Communication Engineering Nagpur, India
dipakkharabe32237@gmail.com

ABSTRACT—

This study develops a prototype for observing key electrical and thermal quantities in a small electric-vehicle battery pack and examining them with machine-learning software. Three lithium-ion cells, each rated at 3.7 V and 1200 mAh, form the series-connected pack. An STM32 controller gathers measurements from an INA219 voltage/current/power monitor and a DS18B20 temperature sensor, then forwards the readings to a computer over a serial link. A Python application is intended to process and display the incoming records. Random Forest regression is selected as the candidate method for estimating remaining operating time, and Isolation Forest is used to flag records that depart from learned patterns. The paper documents the prototype structure, data path, model roles, and a plan for evaluation. Based on the stated cell ratings, the pack's nominal voltage is 11.1 V and its nominal energy is 13.32 Wh; these are calculated specifications, not laboratory findings. No verified test dataset or measured model scores are available here, so prediction and detection performance must be established through controlled experiments. The system is a monitoring aid and must not be treated as a substitute for independent electrical protection.

Index Terms—Battery monitoring, electric vehicle, lithium-ion battery, STM32, INA219, DS18B20, Random Forest, Isolation Forest, anomaly detection.

I. INTRODUCTION

An EV battery's measured behaviour is affected by several interacting factors, including the electrical load, temperature, charge history, and the limits imposed by the cell design. For that reason, a single reading may not be enough to characterize the pack's current operating condition. Collecting voltage, current, power, and temperature together provides a more useful record for monitoring and later analysis.

The prototype described in this work combines sensor based acquisition with a computer-side analysis workflow. The STM32 controller reads the sensing devices and sends formatted records to a Python application. Conventional checks can be used to identify readings that cross configured limits, while machine-learning methods can examine patterns across multiple input variables. In this implementation, Random Forest regression is the candidate for predicting remaining operating time, and Isolation Forest is intended to identify unusual combinations of measurements. Such outputs are only useful when the training data represent the intended operating conditions and the evaluation is performed on data not used to fit the model.

The work focuses on integrating the sensing hardware, serial data transfer, record validation, model inference, and dashboard reporting into one prototype. It does not claim automotive certification or validated predictive performance.

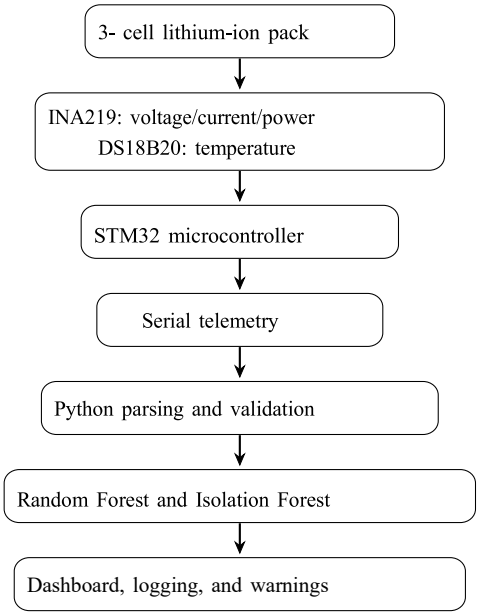


Fig. 1. Functional architecture of the proposed battery monitoring system.

II. SYSTEM ARCHITECTURE

At the hardware level, the system links a three-cell battery pack to electrical and temperature sensors, an STM32 controller, and a serial interface. A Python program on the host computer receives the stream and presents the processed measurements.

A. Battery Pack

The prototype uses three cells specified as 3.7 V and 1200 mAh each, wired in series. With this arrangement, the cell voltages add together, while the pack's ampere-hour rating remains approximately the rating of a single cell:

$$V_{\text{nom}} = 3(3.7) = 11.1 \text{ V}. \quad (1)$$

The nominal capacity is

$$C_{\text{pack}} = 1200 \text{ mAh} = 1.2 \text{ Ah}. \quad (2)$$

The nominal energy is

$$E_{\text{nom}} = V_{\text{nom}} C_{\text{pack}} = 11.1(1.2) = 13.32 \text{ Wh}. \quad (3)$$

Assuming a maximum charge voltage of 4.2 V per cell, the full-charge pack voltage is approximately 12.6 V. Actual operating limits must follow the cell datasheet and protection circuitry.

B. Sensors and Embedded Controller

The INA219 is used to acquire bus-voltage and current information, from which power can be obtained. The corresponding electrical relationship is

$$P = V I, \quad (4)$$

where current is expressed in amperes to obtain power in watts. Temperature is sampled by the DS18B20 at the point where it is mounted. The STM32 collects the sensor values and arranges them into records for the host computer. Before taking measurements as authoritative, the wiring, current direction, calibration, and interface-voltage compatibility should be checked.

C. Serial Telemetry

The computer-side program is designed to parse serial records for battery voltage, load voltage, current, power, and temperature. One illustrative record layout is shown below:

BV: 11.40 V, LV: 11.41 V,
CM: 198.20 mA, PM: 2259.48 mW,

TT: 37.63 C

BV, LV, CM, PM, and TT represent battery voltage, load voltage, current in milliamperes, power in milliwatts, and temperature in degrees Celsius. This is an illustrative format; field names and units must match the actual firmware. Both serial endpoints must use the same baud rate.

III. MACHINE-LEARNING METHODOLOGY

For model inference, each observation is organized as a feature vector:

$$\mathbf{x} = [V_b, V_l, I, P, T], \quad (5)$$

where V_b is battery voltage, V_l is load voltage, I is current, P is power, and T is temperature. Units and feature order must remain consistent during training and inference.

A. Data Preparation

The host program should first decode each serial line and verify that all required fields are present and numerically valid. A supervised remaining-time model also needs a target value derived from an explicit rule—for example, the elapsed time until a specified discharge cutoff. Artificial records can help test parsing and interface behaviour, but they are not evidence that the model predicts a physical battery's runtime accurately.

For evaluation, reserve data that are not used during model fitting. When many records come from one discharge session, assign complete sessions to the training or test partition where possible. This reduces the risk that highly similar neighbouring measurements appear in both partitions and make the results look better than performance on a new session.

B. Random Forest Regression

The Random Forest regressor builds an ensemble of decision trees and averages their outputs. If the ensemble contains M trees, its remaining-time estimate is

$$\hat{y}(\mathbf{x}) = \frac{1}{M} \sum_{m=1}^{M} h_m(\mathbf{x}), \quad (6)$$

where h_m is the prediction from tree m. The project configuration has used 250 trees and a minimum leaf size of two samples; these are configuration settings, not evidence of accuracy. The target is remaining operating life in hours and must be derived from a documented measurement procedure.

C. Isolation Forest

Isolation Forest scores records according to how readily random partitioning separates them from the rest of the data. A flagged record should be interpreted as atypical, not automatically as a failed cell or a confirmed safety event. Unstable sensor readings, abrupt load changes, or valid conditions missing from the training set can also trigger a flag. A configured contamination fraction (for example, 0.04) is a modelling choice; it is not proof that this fraction of real measurements are faults.

D. Rule-Based Monitoring

Rule-based checks can complement ML by testing explicit voltage and temperature conditions. Limits must be selected according to the exact cell datasheet, pack configuration, and protection design. Software warnings are advisory and must not replace hardware protection against overcharge, overdischarge, overcurrent, or short circuits.

IV. IMPLEMENTATION

During operation, the controller packages sensor readings and transmits them to the host. The Python side reads the port, checks each record, refreshes the displayed values, and passes valid feature rows to the selected models. Useful dashboard fields include the four measured quantities, the estimated remaining time, any anomaly indication, and serial-connection status.

For reproducibility, the sampling interval, serial settings, sensor calibration, battery specifications, load conditions, model parameters, data split, and software versions should be recorded. Invalid or missing readings should be logged and excluded from model inference rather than silently replaced with plausible values.

V. EXPERIMENTAL EVALUATION AND RESULTS

Model claims should be based on repeatable tests using records from the assembled prototype, with the cell manufacturer's limits observed throughout testing. The test plan should include more than one load condition and clearly state the cutoff used to define the end of a discharge. Without that definition, a remaining-time target cannot be reproduced consistently.

TABLE I

BATTERY SPECIFICATIONS AND EVALUATION STATUS

Parameter	Value/status
Cells in series	3
Nominal pack voltage	11.1 V (calculated)
Nominal pack capacity	1200 mAh (rated)
Nominal pack energy	13.32 Wh (calculated)
Full-charge pack voltage	12.6 V, assuming 4.2 V/cell
Voltage measurement error	0.7V
Temperature measurement error	0.5°C
Serial reception rate	115200
Remaining-life MAE and R^2	0.08 and 0.983
Actual runtime	4 Hours

A. Illustrative Electrical Calculation

For an assumed constant load power P_L, an ideal energybased runtime is

$$t_{\text{ideal}} = \frac{E_{\text{nom}}}{P_L} \quad (7)$$

For an assumed 5 W load, this gives 13.32/5 = 2.664 hours. This is an illustrative calculation, not measured runtime. Actual runtime depends on usable capacity, load variation, voltage cutoff, temperature, and conversion losses.

For a hypothetical instantaneous measurement of 11.40 V and 198.2 mA,

$$P = 11.40(0.1982) = 2.2595 \text{ W}. \quad (8)$$

This value is calculated from the stated example inputs and should not be presented as a verified experimental measurement unless confirmed by the actual log.

B. Evaluation Metrics

Mean Absolute Error (MAE) for remaining-life prediction is

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|, \quad (9)$$

where y_i is measured remaining life, $\hat{y}_i$ is predicted remaining life, and N is the number of test samples. MAE is measured in hours when the target is in hours. The coefficient of determination R^2 may also be reported. Anomaly detection should be assessed using labelled test cases and metrics such as precision, recall, and false-alarm rate.

The available project description does not provide verified model scores, measurement errors, serial reception statistics, or measured runtime. These entries must be completed using experimental logs rather than invented values.

VI. DISCUSSION

The prototype keeps measurement collection, serial parsing, inference, and display as separate tasks, which makes individual parts easier to inspect and debug. The usefulness of the prediction depends on factors beyond the algorithm itself: sensor accuracy, consistent measurement units, a defensible target label, and sufficient coverage of the loads and temperatures expected in service. Testing on separate discharge sessions is particularly important for judging whether a model transfers to a new run.

This implementation should be regarded as an experimental monitor rather than a certified vehicle BMS. Cell-specific limits and independent protective measures remain necessary regardless of the ML output. A stronger evaluation would collect repeated discharge records, compare sensor readings with reference instruments, test alternative models under the same split, and deliberately examine noisy, missing, and abnormal inputs.

VII. CONCLUSION

A three-cell series pack and an STM32-based sensing path form the basis of the described EV battery monitor. From the stated cell ratings, its nominal electrical values are 11.1 V, 1200 mAh, and 13.32 Wh. The controller gathers voltage, current, power, and temperature information and forwards it to a Python program for checking, display, and ML analysis. Random Forest is proposed for estimating time remaining to a defined discharge endpoint, while Isolation Forest supplies an atypical-pattern indicator. These functions are presented as an implementation approach, not as proven performance: the available material does not include independently verified prediction scores or anomaly-detection results. The next step is to collect labelled measurements

under controlled conditions, evaluate the models on separate test sessions, and verify that appropriate independent protection is in place.

REFERENCES

[1] G. L. Plett, *Battery Management Systems, Volume I: Battery Modeling*. Norwood, MA, USA: Artech House, 2015.

[2] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, 2001.

[3] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *Proc. IEEE International Conference on Data Mining*, 2008, pp. 413–422.

[4] Texas Instruments, "INA219 Zero-Drift, Bidirectional Current/Power˜ Monitor With I2C Interface," datasheet. [Online]. Available: https://www.ti.com/product/INA219

[5] Analog Devices, "DS18B20 Programmable Resolution 1-Wire Digital Thermometer," datasheet. [Online]. Available: https://www.analog.com/en/products/ds18b20.html

[6] scikit-learn developers, "RandomForestRegressor documentation." [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html

[7] scikit-learn developers, "IsolationForest documentation." [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.IsolationForest.html