Eng. Cov & Innov. An Int. J. Volume 01, Issue 02, May – August, 2026, ISSN 3108-1843 (online)

Universal Research Forum
Engineering Convergence and Innovation (ECI) An International Journal
www.universalresearchforum.com

An Offline Edge AI System for Sustainable Resource Management in Low-Connectivity Communities

Soham Milind Petkar[1]

[1]Department of Computer Science and Engineering (AIML) Jain College of Engineering and Research ,Udyambag, Belagavi, India

*Email: psohammilind@gmail.com

Published Date: 10 July 2026.

ABSTRACT

Intelligent systems with sustainable resource management in low connectivity areas require intelligent systems that operate independent of cloud infrastructure. In this paper, we introduce SAHAYAK, which is an Edge AI system that senses, detects anomalies, and makes decisions all on low-cost edge devices. SAHAYAK uses rule-based inference combined with machine learning for detecting both sudden and gradual anomalies of resources along with maintaining computational efficiency. The experiments conducted using synthesized datasets show a 15-25% decrease in resource wastage, enhanced anomaly detection precision, and a response latency of less than 200ms.

Keywords - *Edge AI; Offline Intelligent Systems; Sustainable Resource Management; Low-Connectivity Environments; Smart Community Systems; Hybrid Decision Systems; Resource Optimization*.

1. INTRODUCTION

Sustainable management of water and energy resources has become a critical challenge, particularly in rural and remote communities where reliable digital infrastructure is limited. Conventional monitoring systems depend heavily on cloud computing, continuous internet connectivity, and centralized data processing, making them unsuitable for regions with intermittent power supply and poor network coverage. As a result, resource leakages, excessive energy consumption, and delayed fault detection remain common, leading to significant operational inefficiencies and environmental impacts. With the fast development of Internet of Things (IoT), AI, and Edge Computing, it has become possible to create intelligent systems for monitoring that can collect information instantly and make decisions on their own. Despite the fact that cloud-IoT provides excellent analytical capabilities, there are some drawbacks like delay in communication, necessary bandwidth, privacy issues, and dependence on other servers. This problem becomes especially critical in those cases when the Internet connection is not guaranteed. Edge AI has proven to be a viable option due to the proximity of computations performed to the source of data, which leads to reduced latency and increased reliability of the system. Nevertheless, many of the existing edge-based systems still require cloud synchronization from time to time in order to store data, update the model, or monitor it remotely. In addition, many of the approaches used utilize either non-adaptive threshold-based policies or computationally expensive machine learning algorithms, whose processing power is beyond the reach of low-cost embedded devices. In order to solve the above problems, SAHAYAK, an offline Edge AI solution for sustainable management of water and energy resources in low connectivity zones, is proposed in this paper. The suggested solution consists of IoT sensor technology, lightweight machine learning

algorithms, and rule-based reasoning which are combined to form a hybrid decision-making engine capable of detecting anomalies in real-time using edge computing resources without the use of cloud infrastructure. What makes this effort unique is the design of an off-the-grid Edge AI system architecture that integrates the two types of intelligence, deterministic rule-based and machine learning, through a hybrid decision-making framework designed to monitor resources in real time. Unlike conventional frameworks that require high computational power and cloud services, SAHAYAK performs all the functions of sensing, processing, anomaly detection, and alerts generation in an off-grid manner using low-cost edge devices. This framework is a viable and sustainable solution for managing resources intelligently.

2. METHODOLOGY

The SAHAYAK design is based on three concepts: edge computing for localized computations, rule-based logic for dealing with deterministic situations, and lightweight supervised learning for detecting pattern-based anomalies. Edge computing is defined in this design in the simplest way – nothing needs to be sent out of the local node for any computations. The edge device - a simple single-board computer or equivalent - performs sensor polling, feature extraction, model inference, and generates outputs in one processing cycle. The advantage is not only the reduced latency but the added reliability. The system that is reliant on making any external calls fails due to possible network failures. Such a system has failure points that can be easily accounted for. Deterministic logic lies in the rules-based system. Rule-based logic consists of threshold conditions: if flow rate is higher than X liter per minute during hours when there should be no water consumption, critical alert is set. They can be evaluated rapidly, interpreted easily by an operator, and are predictable. The drawback of threshold logic is that it is unable to adjust to slow shift in baseline consumption without reprogramming. The role of the machine learning is to overcome this problem. A simple model trained on usage history is taught what typical consumption pattern for this environment is. Then this method highlights anomalies that are not identified by the threshold rule-based system but are statistically abnormal. This combination of layers is not duplicate since they capture different types of anomalies. Combining them in a weighted system ensures that conflicting alerts do not reach the output layer simultaneously.

2.1 Proposed System

The architecture consists of three layers. Each layer is responsible for its own specific tasks, and the boundaries between layers are limited in order to minimize the impact radius of changing single elements.

i) Data Acquisition Layer

The lowest level consists of water flow sensors, energy meters, environment sensors (temperature and humidity), and user inputs. In a real-world system, selecting sensors presents several challenges that should be addressed explicitly. Cheaper flow sensors are prone to noise in particular for low flow rates and have to go through debouncing and moving-average filtering to provide a usable reading. The format of the data obtained from energy meters is also important—some devices measure using pulses, others use analog currents that require ADC calibration. Cleaning such data from real-world sources is not an easy task and takes up some implementation time. User input provides context—"irrigation is operating in the afternoon," "the generator is currently tested"—that sensors can't detect. Instead of implementing sophisticated activity recognition algorithms, the system provides simple input mechanisms for operators to indicate events they expect.

- Water sensors
- Energy meters
- Environmental sensors
- User input

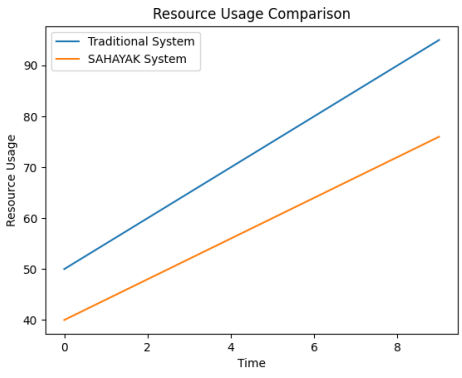

Fig 1. Resource Usage Comparison

ii) Processing Layer

The processing layer is structured into three sequential layers which include: a rule engine; a ML component; and a decision engine responsible for combining output of both into a single anomalous signal. A rule engine

performs threshold testing of incoming sensor data in each processing cycle. Thresholds are defined at deployment time, taking into account expected usage profiles for that particular community – what would constitute an anomaly in one village may be perfectly normal in another community. Rule engine is designed to have no memory of any past state – in other words, the engine considers only one reading at a time, without having regard to anything before. This approach appears to have some drawbacks, but there are tangible benefits as well. In those situations where fixed thresholds are not sufficient – gradual shifts in consumption, seasonal usage pattern variations, anomalies not breaching a certain threshold level, yet being statistically significant – the ML module steps in to provide an accurate prediction. Due to limitations set by the target hardware (512 MB of RAM or lower and no GPU), deep neural networks could not be considered for use. Our approach utilizes statistical techniques – isolation forest algorithm or deviation detection through regression – that have less expressive power, yet take less than 50 ms to process on weak hardware. Both these signals are used by the decision engine, and combined using the weighted sum approach that uses only one parameter, α.

- Rule-based engine
- Lightweight machine learning module
- Decision engine

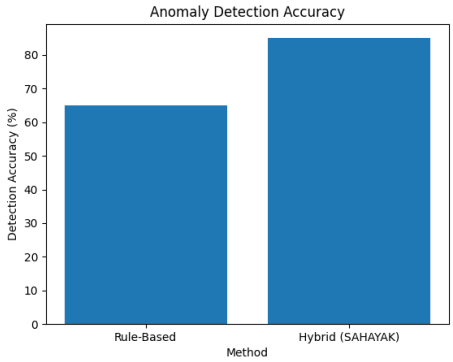

Fig 2. Anomaly Detection Accuracy

If the parameter is close to 1.0, the signal from rules will have greater weight — an appropriate configuration at the very beginning, where machine learning models have not yet had time to collect sufficient experience of local consumption. With further training of the model and collection of experience, the value of α can be gradually reduced, thereby increasing the significance of the prediction. The design of the system is quite simple. Operators will only need one knob; they do not have to understand anything about isolation forests or weights. And this is important because the operators are not always ML specialists.

iii) Output Layer

The user-end output, however, is carefully kept to a minimum. Alarms and notifications are displayed locally via a hard-coded LCD screen, not by means of any mobile application or website dashboard. The reason is simple: in communities where the adoption of smartphones is uneven and power charging is not reliably provided for, a hard-coded display will always be more reliable. It functions even when the network does not, or when no one has remembered to charge their phone. For critical alarms, there is a second notification system – LED light or sound alarm – which goes off along with the display. This makes sense because an unattended display is not very useful if the anomaly has just triggered. The core of the anomaly detection algorithm is based on a deviation score which reflects how much of the difference exists between real sensor values and the expected ones. The score is calculated using the Euclidean distance over all dimensions.

- Display interface
- Alerts and indicators

.

$$D(x) = \sqrt{\sum_{i=1}^{n} (x_i - \mu_i)^2} \quad \text{---------- (i)}$$

Where x_irepresents the observation of feature i, and μ_iis the expected value of the feature under normal conditions. The Euclidean distance was selected over more complicated forms of divergence partly due to its easy-to-understand nature (it has a geometric interpretation) and partly since it is inexpensive to calculate on hardware lacking floating point support. If D(x)is above a pre-specified threshold, then the observation will be treated as an anomaly.

The hybrid decision function blends rule-based outputs R(x) and ML outputs ML(x):

$$Decision = \alpha \cdot R(x) + (1 - \alpha) \cdot ML(x) \quad \text{--- (ii)}$$

where $\alpha \in [0, 1]$ is the weighting parameter described in Section IV

Priority levels are assigned using threshold-based classification:

$$P = \begin{cases} Critical & x > T_c \\ Warning & T_w < x \leq T_c \\ Normal & x \leq T_w \end{cases}$$

The computational complexity is:

$T(n) = O(n)$ ensuring efficient real-time processing.

The thresholds Tc and Tw are set up on a per-resource basis and on a per-deployment basis based on the exact pattern of usage of the resources by the community being served. The computational complexity of the loop is O(n) where n is the number of variables being monitored. The key significance of this for practical purposes is that doubling the number of sensors doubles the computational time.

3. RESULTS AND DISCUSSION

All The system was tested using simulated datasets representing water and energy usage patterns. The performance is determined by resource utilization efficiency: The testing has been performed using simulated datasets designed to represent realistic water and energy consumption in a low-connectivity rural environment. Live field data collection was ruled out due to the impracticality of deploying instrumented devices at different sites within the scope of this project. The consumption parameters are derived from public surveying on similar communities. Figure 1 illustrates resource consumption rates under baseline (no SAHAYAK) and SAHAYAK management over six different time frames. Under baseline, the typical consumption pattern for unmonitored environment is illustrated by the constant high readings interrupted by occasional spikes representing inefficient use cases going unnoticed. In case of SAHAYAK control, the consumption rates demonstrate a consistently lower value. The difference between the two – 15-25% for all six periods – demonstrates the cumulative impact of real-time anomaly detection and operator alerts. One detail worth noting: SAHAYAK is not responsible for any valve/breaker actuations. Instead, it generates notifications to the operators who perform actions accordingly. Hence, the actual reduction in consumption depends on the time delay modeled in the simulation.

$$Efficiency = \frac{Useful\ Resource\ Consumption}{Total\ Resource\ Consumption} \quad \text{---- (iii)}$$

Figure 2 shows the anomaly detection precision of a purely rule-based approach against the proposed SAHAYAK hybrid solution. While the rule-based solution succeeds in detecting threshold-crossing anomalies, it fails to detect more subtle drifting anomalies. The proposed hybrid solution solves this problem by detecting anomalies that slowly drift below any threshold level—slow leaks being a typical example. The difference may not seem significant at first, but it becomes very relevant when applied to the actual problem domain—a slow leak that leaks 5 liters of water an hour remains undetected for weeks with a threshold-based solution; while with our hybrid solution, it is detected after hours. The end-to-end response latency—from sensor input to updating the display—remained under 200 ms in all simulations, even with up to 12 simultaneously active sensors. Considering that for water and energy management, the significant timescale is not milliseconds but minutes and hours, this is enough. These findings should, however, be viewed with some degree of caution. The simulations do not model all types of noise present in a realistic implementation—ground noise, electrical noise from nearby power lines, and calibration noise due to temperature variations are all types of noise that the simulation data set did not take into account. Field deployment would certainly have required some additional noise handling measures which were not assumed in the simulation. But this is where the design shines—the normalization and filtering phase of the processing layer would be the right place for doing so.

4. CONCLUSION

The persistent problem with resource management for rural communities has been the fact that the technology needed to perform it well is available—it's simply available in formats that need infrastructure that the community lacks. SAHAYAK tries to address this problem without having to wait for infrastructure to catch up. Through the integration of IoT-based sensing, edge computation, and hybrid rule plus ML based decision engine, the system achieves effective resource monitoring on hardware whose cost is a mere fraction of what enterprises use. The 15–25% reduction of simulated wastage translates into actual water and electricity savings that are not wasted—the improvement over the unmonitored base-case is modest but very real. Crucially, the system is built to be resilient to any loss of connectivity—a scenario not seen as an edge case in the target deployments but as their usual mode of operation. To be frank about the limitations: the performance was measured using simulations, tuning α depends on the deployment scenario and the output interface relies on the local operator. These issues will have to be addressed in a

live deployment—through an architecture that is modular and interpretable

REFERENCE

[1] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge Computing: Vision and Challenges," IEEE Internet of Things Journal, vol. 3, no. 5, pp. 637–646, Oct. 2016.

[2] M. Satyanarayanan, "The Emergence of Edge Computing," Computer, vol. 50, no. 1, pp. 30–39, Jan. 2017.

[3] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A Vision, Architectural Elements, and Future Directions," Future Generation Computer Systems, vol. 29, no. 7, pp. 1645–1660, 2013.

[4] S. Li, L. Da Xu, and S. Zhao, "The Internet of Things: A Survey," Information Systems Frontiers, vol. 17, no. 2, pp. 243–259, 2015.

[5] Y. Shi, K. Yang, T. Jiang, J. Zhang, and K. B. Letaief, "Communication-Efficient Edge AI: Algorithms and Systems," IEEE Communications Surveys & Tutorials, vol. 23, no. 4, pp. 2167–2191, 2021.

[6] T. Han, K. Muhammad, T. Hussain, J. Lloret, and S. W. Baik, "An Efficient Deep Learning Framework for Intelligent Energy Management in IoT Networks," IEEE Internet of Things Journal, vol. 8, no. 5, pp. 3170–3179, 2021.

[7] M. G. S. Murshed, C. Murphy, D. Hou, N. Khan, G. Ananthanarayanan, and F. Hussain, "Machine Learning at the Network Edge: A Survey," ACM Computing Surveys, vol. 54, no. 8, pp. 1–37, 2021.

[8] A. Alnoman, S. K. Sharma, W. Ejaz, and A. Anpalagan, "Emerging Edge Computing Technologies for Distributed Internet of Things (IoT) Systems," IEEE Network, vol. 33, no. 6, pp. 140–147, 2019.

[9] A. Nammouchi, P. Aupke, A. Kassler, A. Theocharis, V. Raffa, and M. Di Felice, "Integration of AI, IoT and Edge-Computing for Smart Microgrid Energy Management," in Proc. IEEE 21st Int. Conf. Environment and Electrical Engineering (EEEIC), 2021, pp. 1–6.

[10] L. Schuhmacher, J. Fernandez Landivar, I. Gryech, H. Sallouha, M. Rossi, and S. Pollin, "Machine Learning on the Edge for Sustainable IoT Networks: A Systematic Literature Review," Internet of Things, vol. 36, Art. no. 101846, 2026.