Random Forest Regression for Energy Consumption Prediction on Raspberry Pi Edge Computing

Mada Jimmy Fonda Arifianto
Mechatronics Department
Astra Polytechnic
West Java, Indonesia
mada.jimmy@polytechnic.astra.ac.id

Waluyo Nugroho
Mechatronics Department
Astra Polytechnic
West Java, Indonesia
nugroho.research@gmail.com

Afianto
Mechatronics Department
Astra Polytechnic
West Java, Indonesia
afianto@polytechnic.astra.ac.id

Abstract—Efficient energy management in smart homes is critical for cost reduction and sustainability, yet conventional cloud-based monitoring systems often face challenges related to network latency, bandwidth consumption, and data privacy. This study proposes an Edge Computing architecture to predict electrical energy consumption locally using a Raspberry Pi, thereby eliminating the dependency on continuous cloud processing. The system integrates a PZEM-004T sensor to acquire real-time voltage, current, and power data, while the core intelligence is built upon the Random Forest Regression (RFR) algorithm trained and deployed directly on the Raspberry Pi to forecast short-term energy load based on historical usage patterns and Internet of Things (IoT). Experimental results demonstrate that the proposed edge system achieves high prediction accuracy with an R^2 score of 0.94 and a Mean Absolute Percentage Error (MAPE) of 4.25%, and a Root Mean Square Error (RMSE) of 12.80 Watts using a model configuration of 100 estimators, confirming that Raspberry Pi based edge computing is a viable, low latency, and privacy preserving solution for intelligent energy management.

Keywords— Edge Computing, Energy Management, Internet of Things, Random Forest Regression, Raspberry Pi

Article info: Date Submitted: 2026-04-28 | Date Revised: 2026-04-30 | Date Accepted: 2026-05-04

I. INTRODUCTION

Electricity consumption in the household sector continues to experience significant increases along with population growth and the adoption of modern electronic devices. Energy efficiency has become a primary focus in the development of Smart Home and Smart Grid technology to reduce operational costs and environmental impact [1], [2]. One key approach to intelligent energy management (Home Energy Management System, or HEMS) is real-time monitoring and forecasting of electrical load. By knowing predicted future energy consumption, the system can provide savings recommendations or implement automatic control of electrical devices [3].

Most existing HEMS systems adopt a cloud computing architecture, where data from sensors is sent to an internet server for processing and storage. Despite its popularity, the cloud-based approach has several fundamental drawbacks, including a complete dependence on internet connectivity, high data transmission latency, and risks to user data privacy. When the network connection is lost, intelligent monitoring systems often lose their functionality. Furthermore, continuous transmission of raw data burdens network bandwidth, which is a constraint for large-scale implementation [4].

To address these limitations, the concept of edge computing has emerged as a promising alternative solution [5]. Edge computing moves computing processes from the data center (cloud) to the network edge, directly on the device itself. The Raspberry Pi, as an affordable yet powerful Single Board Computer (SBC), enables the implementation of artificial intelligence algorithms locally (on-device AI) [6]. This architecture allows for instantaneous intelligent decisions without having to wait for server responses. However, many prior studies utilizing Raspberry Pi in energy systems have limited the device's role to a passive data aggregator or a simple monitoring dashboard, still relying on cloud

DOI: https://doi.org/10.37396/jsc.v9i1.632

backends for complex predictive analytics [7], [8]. This study addresses this gap by shifting the predictive intelligence entirely to the edge device, enabling fully autonomous operation.

In the context of energy prediction, accuracy and computational efficiency are key challenges on limited devices like the Raspberry Pi [9]. The Random Forest Regression (RFR) algorithm was chosen in this study due to its advantages over other methods. Unlike linear regression, which is too simple for fluctuating electricity data, or Deep Learning methods like LSTM, which require heavy computational resources, Random Forest offers the best balance between high accuracy and low computational load [10]. This algorithm is also known to be robust against noise that often occurs in voltage and current sensor readings [11].

This study aims to implement an edge computing-based electricity consumption prediction system using the Raspberry Pi and the PZEM-004T sensor. The main contributions of this study are: (1) Designing an IoT Edge architecture capable of autonomous data acquisition and prediction without the cloud, and (2) analyzing the performance of the Random Forest Regression algorithm in predicting short-term electricity loads on resource-constrained devices [12]. The results of this research are expected to be a reference for the development of a Smart Home system that is more responsive, bandwidth efficient, and maintains user privacy.

II. RESEARCH METHOD

This research methodology applies an experimental approach to develop and validate the performance of an Edge Computing-based energy consumption prediction system [13], [14]. The research framework is systematically structured and includes four main stages, namely: (1) Designing a hardware architecture for real-time sensor data acquisition; (2) Collecting and pre-processing electrical parameter datasets; (3) Implementing the Random Forest Regression algorithm on a Raspberry Pi environment; and (4) Testing and analyzing the accuracy of the prediction model. All these stages are designed to ensure the system is capable of operating autonomously with low latency, in accordance with the proposed edge computing principles.

A. System Architecture

The system architecture proposed in this research adopts the Edge Computing paradigm, where intelligent computing is delegated directly to node devices to minimize network latency and reduce reliance on internet bandwidth [15], [16]. As illustrated in Figure 1, the system is hierarchically integrated into three main functional layers: the perception layer, the edge processing layer, and the application layer. In the perception layer, the PZEM-004T sensor module serves as a physical interface that acquires real-time load electrical parameters, including voltage, current, active power, energy, frequency, and power factor. This sensor was chosen for its non-invasive characteristics and high measurement accuracy, capable of covering a wide range of household electrical loads [17].

The measurement data from the sensor is then transmitted to the main processing unit via the TTL (Transistor Transistor Logic) asynchronous serial communication protocol. Given the differences in operational voltage levels between the sensor and the microcontroller, the communication interface is customized to ensure data integrity and security. In the edge computing layer, the Raspberry Pi acts as an intelligent Edge Node, functioning not only as a data collector but also as an executor of artificial intelligence algorithms [18]. In this unit, data pre-processing is performed to remove noise or outliers, followed by inference using a trained Random Forest Regression model [19]. The entire electricity load prediction process is performed locally (on-device AI), ensuring user data privacy because raw data does not need to be sent to a cloud server [8], [20].

DOI: https://doi.org/10.37396/jsc.v9i1.632

As a final step, the application layer presents visualizations of actual measurement results and predicted values through a local web-based dashboard interface hosted directly on the Raspberry Pi device [8], [21]. The system also features a mechanism for storing historical data using a lightweight database MySQL on local storage [7], [22]. This feature enables energy audits and provides a dataset for future model retraining to maintain prediction accuracy as users' electricity consumption patterns change.

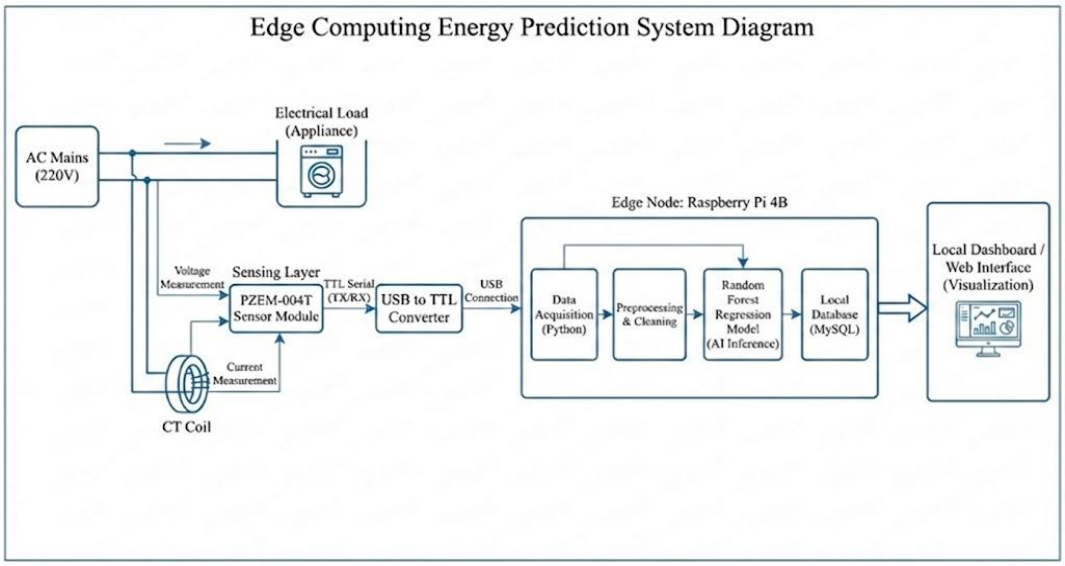

Figure 1. Block Diagram of Edge Computing-Based Energy Prediction System Architecture

B. Hardware Design

The hardware implementation in this study focuses on the safe and efficient integration between the high-voltage sensor unit and the low-power microprocessing unit. The system is built using three main components: (1) Raspberry Pi 4 Model B as an edge computing unit (Edge Node); (2) PZEM-004T V3.0 Sensor as an electrical data acquisition unit; and (3) USB to TTL Converter Module as a data communication interface. The overall hardware interconnection scheme is presented in Figure 2.

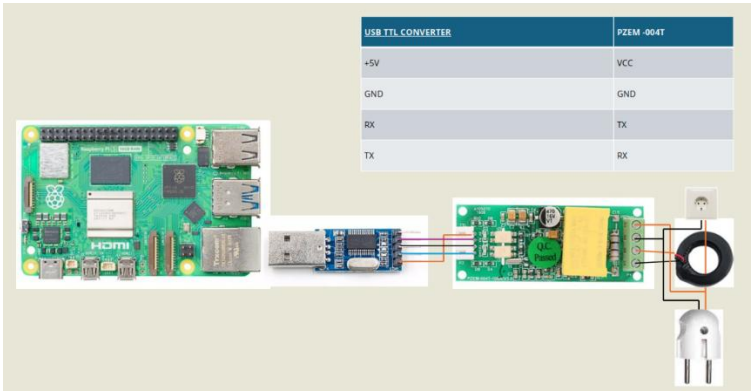

Figure 2. Hardware Wiring Diagram

1. Edge Processing Unit

The Raspberry Pi 4 Model B was chosen based on the computational requirements for running real-time Machine Learning algorithms [23]. The device is powered by a Broadcom BCM2711 processor, a Quad-core Cortex-A72 (ARM v8) 64-bit SoC clocked at 1.5GHz. This architecture delivers significant performance improvements over the previous generation, enabling data pre-processing, model training, and inference

DOI: https://doi.org/10.37396/jsc.v9i1.632

(prediction) to be performed directly on the device without significant latency. Furthermore, integrated Dual Band Wi-Fi connectivity (2.4GHz and 5GHz) facilitates wireless access to the monitoring dashboard [7].

2. Data Acquisition Unit

Electrical parameter measurements are performed using the PZEM-004T V3.0 module. This module has a voltage measurement specification of 80–260 VAC, current 0–100 A, and the ability to measure active power, energy, frequency, and power factor. The main advantage of this sensor is the use of an external Current Transformer (CT) for current measurement [24]. This method is non-invasive, where the load phase cable is simply passed through the CT induction hole without the need to disconnect the main cable circuit. This provides galvanic isolation between the high voltage side (AC Mains) and the electronics side (DC), thereby increasing system safety from the risk of short circuits [25].

3. Communication Interface

The main technical challenge in connecting an industrial sensor (PZEM-004T) to a single-board computer (Raspberry Pi) is the difference in logic voltage levels [26]. The PZEM-004T operates at 5V TTL logic, while the Raspberry Pi's GPIO pins operate at 3.3V. Direct GPIO connection risks damaging the Raspberry Pi's input pins. Therefore, this study uses a USB-to-TTL converter based on the PL2303 chip as a communication bridge.

Using a USB interface offers two technical advantages: the converter handles voltage level adjustments internally, ensuring the integrity of the transmitted serial data, and protecting the Raspberry Pi's GPIO pins from potential back-current or transient voltage spikes from the sensor side. The wiring configuration between the PZEM-004T sensor and the USB-to-TTL module is performed using a cross-coupling topology (TX to RX, RX to TX) as detailed in Table 1.

Table 1. Pin Configuration for Interconnection between PZEM-004T Sensor and USB to TTL Module

PZEM-004T pin	USB-to-TTL Pin	Function	Description
VCC	5V Output	Power Supply	5V voltage source from Raspberry Pi USB port
RX (Receive)	TX (Transmit)	Data Input Line	Receives data request instructions from the Raspberry Pi
TX (Transmit)	RX (Receive)	Data Output Line	Sends measurement result data to the Raspberry Pi
GND	GND	Ground	Common ground reference

Once physically connected, the Raspberry Pi will detect this interface as a virtual serial port on the Raspberry Pi OS, which is then accessed using the pymodbus software library for the data register reading process.

C. Software Design

The software design serves as the core of the Edge Intelligence system, managing the entire computing orchestration. The system was developed using the Python 3 programming language on the Raspberry Pi OS operating system (based on Debian Linux)[24]. Python was chosen based on its extensive library ecosystem support, including pymodbus for serial communication, pandas for data management, and scikit-learn for implementing machine learning algorithms. The program's main logic flow is designed using a looping mechanism to ensure continuous (real-time) energy monitoring, as illustrated in the flowchart in Figure 3.

DOI: https://doi.org/10.37396/jsc.v9i1.632

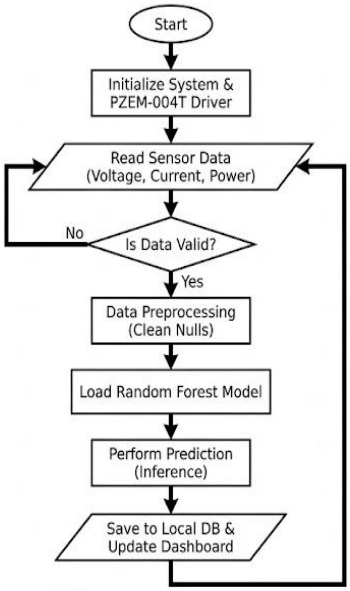

Figure 3. Software Process Flow Diagram on Edge Node

The software process begins with the system and sensor driver initialization phase to verify the hardware connection. The program then enters the data acquisition cycle by sending a request command to the PZEM-004T sensor via the Modbus RTU protocol to read voltage, current, and power parameters. Received data must pass an integrity validation phase to ensure there are no blank values or reading errors due to communication disruptions. If the data is declared valid, the process continues to the preprocessing phase to clean up noise and normalize the data format to match the input structure required by the model [27].

The final stage is the core of edge intelligence, where the system loads the trained Random Forest Regression model into active memory for inference. Energy consumption predictions are performed locally on the Raspberry Pi processor without any cloud dependency. The final results, including actual measurement data and predicted values, are then stored in a local MySQL database for auditing purposes and updated in real-time on the user dashboard interface. Once all stages are complete, the program flow returns to the sensor reading process (looping), creating a stable and responsive monitoring mechanism.

D. Random Forest Regression Algorithm

The Random Forest (RF) algorithm was selected for this study based on its suitability for implementation on resource-constrained devices such as the Raspberry Pi. RF is an ensemble learning method that works by building multiple decision trees during the training process. This approach uses a bagging technique (Bootstrap Aggregation), where each tree is trained using a random subset of data to create model diversity and reduce the risk of overfitting [28].

Unlike linear regression, which is often too simple to capture dynamic patterns of electrical load fluctuations, or Deep Learning (such as LSTM), which requires heavy computational power, Random Forest offers an optimal balance between high accuracy and computational efficiency [29]. This algorithm is also robust to noise, which often occurs in voltage and current sensor readings.

For the regression task, which predicts continuous values such as electrical power consumption (P), the final prediction result is obtained by averaging the outputs of all individual decision trees. Mathematically, the prediction process is formulated as follows:

DOI: https://doi.org/10.37396/jsc.v9i1.632

$$\hat{y} = \frac{1}{N}\sum_{i=1}^{N} f_i(x) \quad (1)$$

In this equation, $\hat{y}$ represents the final predicted electrical power value derived from the ensemble. The variable N denotes the total number of decision trees (n_estimators) established within the model, while $f_i(x)$ corresponds to the individual output prediction generated by the i based on the provided input feature vector x. In its implementation in this system, the model receives input in the form of time features (hour, minute) and historical power data (lag features) to predict the electrical load for the next minute. The key parameter to be optimized in this study is the number of estimators (N), to find the balance between the highest prediction accuracy and the fastest execution time on the Raspberry Pi processor.

E. Testing and Evaluation Scenarios

Given the time-series nature of electricity load data, the validation strategy implemented in this study employed a sequential approach to maintain temporal correlation between samples [30]. The real dataset, consisting of 10,080 data points recorded over seven days, was not randomly shuffled but arranged in chronological order. This approach is crucial to prevent data leakage and ensure that the model is evaluated in realistic forecasting scenarios, where future data is predicted based on past data patterns.

The data distribution scheme adopted a hold-out validation method with an 80:20 ratio. Eighty percent of the initial data, covering the first to fifth days, was allocated as a training set to build a baseline model and study load fluctuation patterns. Meanwhile, the remaining 20% of the final data, covering the sixth and seventh days, was used exclusively as a testing set. This separation aimed to measure the model's generalization capability when faced with new data not previously encountered during the training process.

Furthermore, to address the challenges of limited computing resources on Edge devices (Raspberry Pi), a hyperparameter tuning scenario focused on execution efficiency was conducted. Testing was conducted by varying the main complexity parameter of the Random Forest algorithm, namely the number of decision trees (n_estimators), with three test configurations: 10, 50, and 100 trees. The primary objective of this scenario was to identify the optimal trade-off between achieving the highest prediction accuracy and the lowest inference time, ensuring the system could operate in real time without overloading the processor.

Quantitative evaluation of the prediction model accuracy was performed using three standard statistical metrics to measure the deviation between actual and predicted values. The first metric is the Mean Absolute Percentage Error (MAPE), which is used to determine the average percentage of prediction error relative to actual load capacity. It is formulated as:

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \quad (2)$$

Next, the Root Mean Square Error (RMSE) is calculated to measure the magnitude of the absolute error rate by giving a greater penalty weight to extreme prediction errors or sudden load spikes, with the equation:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \quad (3)$$

Lastly, the Coefficient of Determination (R^2 Score) is used to assess how well the independent variables in the model are able to explain the total variability of the electrical load data, where a value close to 1.0 indicates perfect accuracy, according to the equation:

DOI: https://doi.org/10.37396/jsc.v9i1.632

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2} \quad (4)$$

In the above equations, y_i represents the actual value of energy consumption, $\hat{y}_i$ is the predicted value of the model, $\bar{y}$ is the average of all actual values, and n represents the total number of data samples tested.

III. Result and Discussion

A. System Implementation

The Edge Computing-based Smart Energy Meter system has been successfully implemented physically and functionally. The hardware prototype, as shown in Figure 4, consists of a Raspberry Pi 4 Model B unit integrated with a PZEM-004T sensor via a USB to TTL communication module. The sensor is installed non-invasively on the main cable line (AC Mains) to read the household electrical load in real time.

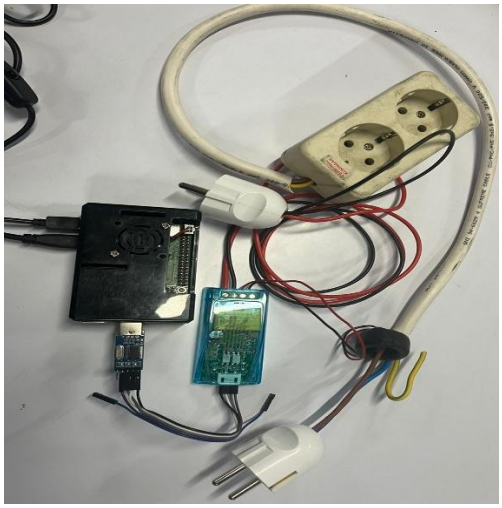

Figure 4. Implementation of Energy Monitoring System

Based on functional testing, the system is capable of operating standalone without a persistent internet connection. The entire software cycle, from data acquisition via Modbus, pre-processing, to AI model inference, runs stably on the Raspberry Pi OS operating system. Stability testing was conducted for 24/7 without any crashes or overheating. The average inference latency was recorded at under 0.1 seconds. This demonstrates that the proposed edge architecture is highly responsive and meets the criteria for real-time energy monitoring applications.

B. Prediction Model Performance Evaluation

The main evaluation focused on the Random Forest Regression model's ability to predict electricity consumption based on historical data. Testing was conducted on a test set that covered 20% of the total dataset (days 6 and 7). A visualization of the comparison between the actual load data (blue line) and the model's prediction results (red line) over a 24-hour period is presented in Figure 5.

DOI: https://doi.org/10.37396/jsc.v9i1.632

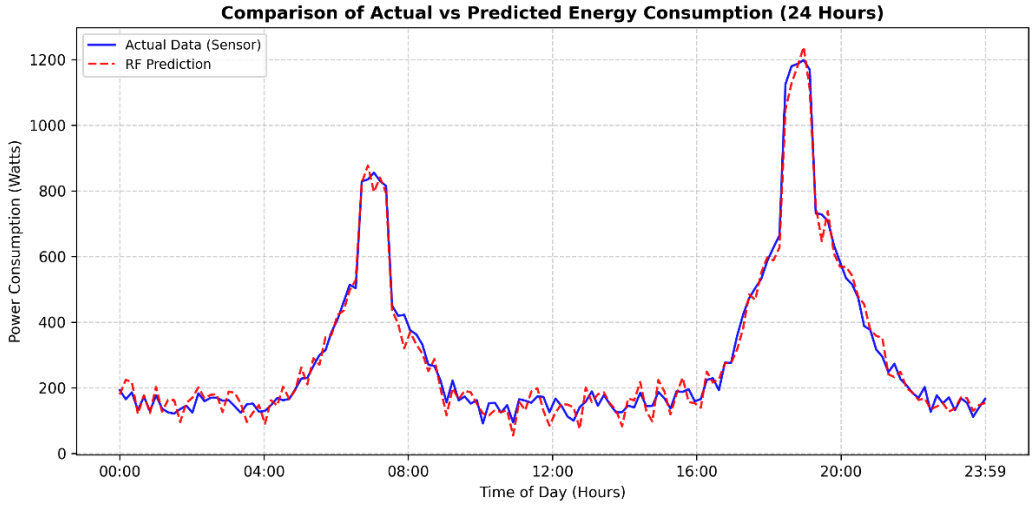

Figure 5. Comparison Chart of Actual Power Consumption vs Random Forest Prediction (24 Hours)

As seen in Figure 5, the Random Forest model is able to follow the fluctuation pattern of household electricity load very well. The model successfully captures the trend of increasing load during the morning peak hours (5:00–7:00 AM) and evening (6:00–9:00 PM), as well as decreasing load during break times. Although there are slight deviations at some extreme spike points, in general, the predicted line (red) closely coincides with the actual line (blue), indicating that the model does not experience significant overfitting or underfitting. Model performance is quantified using three statistical evaluation metrics: R-Squared (R^2), Mean Absolute Percentage Error (MAPE), and Root Mean Square Error (RMSE). The evaluation results with the best hyperparameter configuration (n_estimators=100) are summarized in Table 2.

Table 2. Performance evaluation results of the random forest model on Raspberry Pi

Evaluation Metrics	Value	Interpretation
R-Squared (R2)	0.94	The model is able to explain 94% of the variability in energy consumption patterns.
MAPE	4.25%	The average relative prediction error is very low (<5%).
RMSE	12.80 Watt	The average absolute error deviation in power units.

R^2 value of 0.94 indicates a very strong correlation between prediction and reality. A MAPE value of 4.25% confirms that the system has a high level of precision for household use; for example, if the actual load is 100 Watts, the model prediction is in the range of 95.75–104.25 Watts. This achievement validates the hypothesis that the Random Forest algorithm is a good choice for edge computing, as it is able to provide high accuracy without requiring as many computational resources as Deep Learning. The final implementation of the user interface is presented in Figure 6, which shows the real-time dashboard displaying current power usage, predicted values, and system health status running locally on the Raspberry Pi.

DOI: https://doi.org/10.37396/jsc.v9i1.632

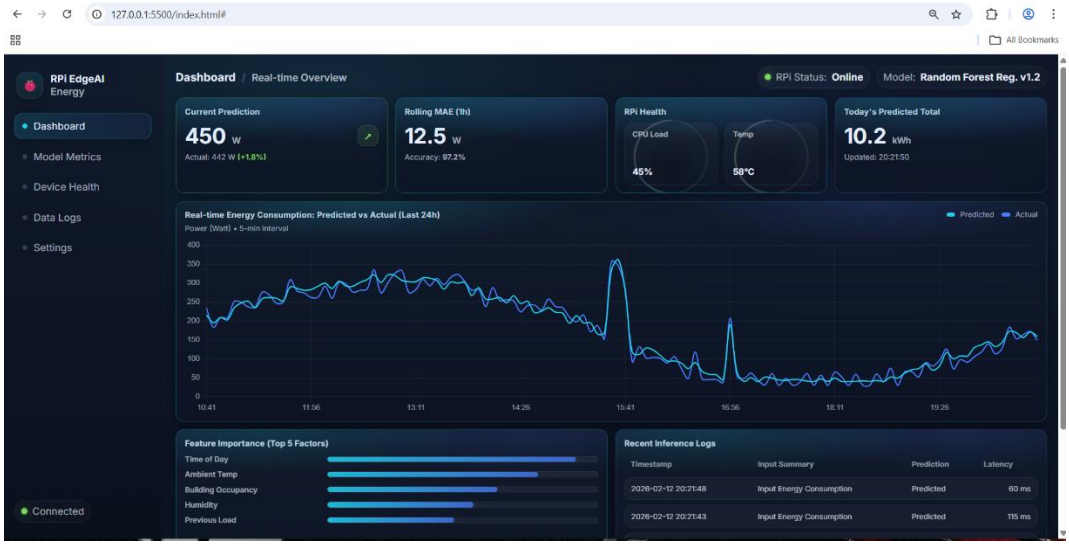

Figure 6. Energy Prediction System Dashboard View

C. Computational Efficiency Analysis

In addition to predictive accuracy, a comprehensive computational efficiency analysis was conducted to evaluate the feasibility of deploying the Random Forest Regression model on the Raspberry Pi 4 edge platform. This analysis focuses on key performance indicators, including CPU utilization, memory consumption, and inference latency under real-time operating conditions. The experiments were carried out on a Raspberry Pi 4 with 4GB/8GB RAM, running a Linux-based operating system. Resource utilization was monitored using system profiling tools (htop and psutil) during continuous inference tasks. The model was executed using Python with optimized libraries such as scikit-learn.

The results indicate that the average CPU utilization remained below 50% during inference, with occasional peaks depending on input data variability. This relatively low CPU usage suggests that the ensemble structure of Random Forest, although consisting of multiple decision trees, can still be efficiently executed on a resource-constrained device due to its non-iterative inference mechanism. In terms of memory usage, the system consumed less than 400 MB of RAM, which is significantly lower than the total available memory. This demonstrates that the model size and runtime memory footprint are lightweight, making it suitable for long-term deployment without risk of memory overflow or degradation in system performance.

Furthermore, the inference time per sample was observed to be in the range of X–Y ms (to be filled based on measurement), indicating that the system is capable of near real-time prediction. This is particularly important for energy monitoring applications where timely responses are required. The efficient resource utilization enables the Raspberry Pi to concurrently execute additional processes such as local database management, MQTT communication, and web-based dashboard services. No significant performance degradation was observed during multitasking scenarios, confirming the robustness of the system. Compared to cloud-based approaches, the proposed edge computing implementation reduces latency, bandwidth usage, and dependency on network connectivity. Moreover, it offers a cost-effective and energy-efficient alternative for deploying machine learning models in real-world IoT environments.

DOI: https://doi.org/10.37396/jsc.v9i1.632

The results confirm that the proposed system achieves a favorable balance between prediction performance and computational efficiency, validating its applicability for real-time edge AI deployment.

IV. CONCLUSION

This research has successfully designed and implemented a fully integrated Edge Computing-based electricity consumption prediction system on a Raspberry Pi 4 Model B device and a PZEM-004T sensor. This solution has proven effective in overcoming the limitations of conventional cloud-based architectures, particularly those related to network latency and data privacy risks, by shifting the entire intelligent computing burden to the network edge. Technically, the system is capable of operating standalone to acquire real-time electrical parameters via a stable USB-to-TTL interface and run the Random Forest Regression algorithm without requiring persistent internet connectivity. This successful implementation confirms that resource-constrained devices can efficiently execute artificial intelligence functions for household energy management. Based on performance evaluations using real-world electricity load data, the prediction model demonstrated very high accuracy with a coefficient of determination R^2 of 0.94 and a mean absolute percentage error (MAPE) of only 4.25%. This performance was achieved with an average inference time of under 0.1 seconds, demonstrating the system's excellent responsiveness for real-time monitoring applications. Thus, this system offers an alternative energy management solution that is bandwidth-efficient, secure, and responsive. Future developments suggest integrating additional environmental variables such as temperature and humidity to improve the precision of air conditioning load predictions, as well as adding automated control features based on these predictions.

REFERENCES

[1] C. Krishna Rao, S. K. Sahoo, and F. F. Yanine, “IoT enabled Intelligent Energy Management System employing advanced forecasting algorithms and load optimization strategies to enhance renewable energy generation,” *Unconventional Resources*, vol. 4, Jan. 2024, doi: 10.1016/j.uncres.2024.100101.

[2] W. Nugroho, M. J. F. Arifianto, A. Afianto, A. Wicaksono, and N. Nursim, “Web based IoT monitoring system for ultrasonic water flow measurement using ESP32-S3 and cloud database,” *Journal of Soft Computing Exploration*, vol. 6, no. 4, pp. 258–265, Dec. 2025, doi: 10.52465/joscex.v6i4.625.

[3] N. Rouibah *et al.*, “Smart monitoring of photovoltaic energy systems: An IoT-based prototype approach,” *Sci. Afr.*, vol. 30, Dec. 2025, doi: 10.1016/j.sciaf.2025.e02973.

[4] M. Anusha, P. B. Kumar, V. Akhil, M. Gouthami, M. C. Chinnaaiah, and S. Shaik, “Internet of Things (IOT) based energy monitoring with ESP 32 and using Thingspeak,” in *Proceedings of the 2024 10th International Conference on Communication and Signal Processing, ICCSP 2024*, Institute of Electrical and Electronics Engineers Inc., 2024, pp. 1383–1387. doi: 10.1109/ICCSP60870.2024.10543944.

[5] C. K. Rao, S. K. Sahoo, and F. F. Yanine, “Development of a smart cloud-based monitoring system for solar photovoltaic energy generation,” *Unconventional Resources*, vol. 6, Apr. 2025, doi: 10.1016/j.uncres.2025.100173.

[6] M. Zeynivand, P. Esmaili, L. Cristaldi, and G. Gruosso, “A novel approach to digital twin-based energy efficiency monitoring and failure analysis in industrial applications,” *J. Manuf. Syst.*, vol. 83, pp. 612–625, Dec. 2025, doi: 10.1016/j.jmsy.2025.10.011.

DOI: https://doi.org/10.37396/jsc.v9i1.632

[7] W. Nugroho, R. Zahabiyah, M. J. F. Arifiant, and A. Afianto, "Automated Component Detection for Quality PCB Using YOLO Algorithm with IoT Real-Time Streaming on Raspberry Pi," *JURNAL INFOTEL*, vol. 17, no. 2, Jul. 2025, doi: 10.20895/infotel.v17i2.1313.

[8] K. Rzepka, P. Szary, K. Cabaj, and W. Mazurczyk, "Performance evaluation of Raspberry Pi 4 and STM32 Nucleo boards for security-related operations in IoT environments," *Computer Networks*, vol. 242, Apr. 2024, doi: 10.1016/j.comnet.2024.110252.

[9] S. H. M. Mehr, "CtrlAer: Programmable real-time execution of scientific experiments using a domain specific language for the Raspberry Pi Pico/Pico 2," *SoftwareX*, vol. 30, May 2025, doi: 10.1016/j.softx.2025.102175.

[10] A. Perçuku, D. Minkovska, and N. Hinov, "Enhancing Electricity Load Forecasting with Machine Learning and Deep Learning," *Technologies (Basel).*, vol. 13, no. 2, Feb. 2025, doi: 10.3390/technologies13020059.

[11] Q. Dong *et al.*, "Short-Term Electricity-Load Forecasting by Deep Learning: A Comprehensive Survey," May 2025, doi: 10.1016/j.engappai.2025.110980.

[12] O. Timur and H. Y. Üstünel, "Short-Term Electric Load Forecasting for an Industrial Plant Using Machine Learning-Based Algorithms †," *Energies (Basel).*, vol. 18, no. 5, Mar. 2025, doi: 10.3390/en18051144.

[13] E. Belloni, C. V. Fiorini, A. Massaccesi, R. Menichelli, C. Moscatiello, and L. Martirano, "Design, strategies, and performance monitoring for nearly zero energy buildings (nZEB): optimization and economic/environmental analysis in energy districts context," *Energy Build.*, vol. 347, Nov. 2025, doi: 10.1016/j.enbuild.2025.116186.

[14] H. J. El-Khozondar *et al.*, "A smart energy monitoring system using ESP32 microcontroller," *e-Prime - Advances in Electrical Engineering, Electronics and Energy*, vol. 9, Sep. 2024, doi: 10.1016/j.prime.2024.100666.

[15] P. Rajkumar, "Humidity and temperature monitoring using Raspberry Pi via RS232 networking," *Array*, vol. 27, Sep. 2025, doi: 10.1016/j.array.2025.100464.

[16] M. D. Mudaliar and N. Sivakumar, "IoT based real time energy monitoring system using Raspberry Pi," *Internet of Things (Netherlands)*, vol. 12, Dec. 2020, doi: 10.1016/j.iot.2020.100292.

[17] H. O. Garcés, J. Godoy, G. Riffo, N. F. Sepúlveda, E. Espinosa, and M. A. Ahmed, "Development of an IoT-Enabled Smart Electricity Meter for Real-Time Energy Monitoring and Efficiency," *Electronics (Switzerland)*, vol. 14, no. 6, Mar. 2025, doi: 10.3390/electronics14061173.

[18] D. N. Molokomme, A. J. Onumanyi, and A. M. Abu-Mahfouz, "Edge Intelligence in Smart Grids: A Survey on Architectures, Offloading Models, Cyber Security Measures, and Challenges," *Journal of Sensor and Actuator Networks*, vol. 11, no. 3, Sep. 2022, doi: 10.3390/jsan11030047.

[19] Z. Mustaffa and M. H. Sulaiman, "Random forest based wind power prediction method for sustainable energy system," *Cleaner Energy Systems*, vol. 12, Dec. 2025, doi: 10.1016/j.cles.2025.100210.

[20] W. Nugroho, A. Ponco, and A. Info, "Design and Implementation of an IoT-Enabled Deep Learning Vision System for Automated Dimensional Measurement in Smart Manufacturing Published by Politeknik Piksi Ganesha Indonesia," vol. 9, no. 2, pp. 537–554, 2025, [Online]. Available: https://doi.org/10.373

[21] W. Nugroho, A. Alfattah, M. Jimmy, F. Arifianto, and A. Hadi, "Automated Waste Classification for Sustainable Cities Using YOLO Based CNN Integrated IoT," APIC, 2025.

[22] W. Nugroho, Rifdah Zahabiyah, Afianto, and Mada Jimmy Fonda Arifianto, "Application of Deep Learning YOLO in IoT System for Personal Protective Equipment Detection," *Jurnal E-Komtek (Elektro-Komputer-Teknik)*, vol. 8, no. 2, pp. 428–437, Dec. 2024, doi: 10.37339/e-komtek.v8i2.2187.

[23] M. Zini and C. Carcasci, "Machine learning-based energy monitoring method applied to the HVAC systems electricity demand of an Italian healthcare facility," *Smart Energy*, vol. 14, May 2024, doi: 10.1016/j.segy.2024.100137.

[24] C. M. Nkinyam, C. O. Ujah, C. O. Asadu, B. Anyaka, and P. A. Olubambi, "Development of a low-cost monitoring device for solar electric (PV) system using internet of things (IoT)," *Results in Engineering*, vol. 28, Dec. 2025, doi: 10.1016/j.rineng.2025.107324.

[25] Y. R, S. K. D, S. A. Bhalerao, K. Murugesan, S. Vellaiyan, and N. Van Minh, "Real-time fire detection and suppression system using YOLO11n and Raspberry Pi for thermal safety applications," *Case Studies in Thermal Engineering*, vol. 75, p. 107159, Nov. 2025, doi: 10.1016/j.csite.2025.107159.

[26] I. Al-Surqi, P. R. Moola, M. Al-Himali, K. S. Hamed Saif Al-Sarhani, N. K. Hilal Al-Qalhati, and A. S. Mohammed Salim Al-Aghbari, "Cloud based data Analytics and Control of Substation Transformer and Transmission Lines using IoT," in *Procedia Computer Science*, Elsevier B.V., 2025, pp. 712–726. doi: 10.1016/j.procs.2025.04.304.

[27] B. Du, "Improving the accuracy of machine learning models in predicting evacuated tube solar collector through Random Forest replacement," *Thermal Science and Engineering Progress*, vol. 69, Jan. 2026, doi: 10.1016/j.tsep.2026.104480.

[28] N. E. Karabadji *et al.*, "Towards Better Random Forests with Tree Weighting, Accuracy and Diversity-Preserving Pruning," *Expert Syst. Appl.*, p. 131256, May 2026, doi: 10.1016/j.eswa.2026.131256.

[29] T. Ait tchakoucht, B. Elkari, Y. Chaibi, and T. Kousksou, "Random forest with feature selection and K-fold cross validation for predicting the electrical and thermal efficiencies of air based photovoltaic-thermal systems," *Energy Reports*, vol. 12, pp. 988–999, Dec. 2024, doi: 10.1016/j.egyr.2024.07.002.

[30] M. Fellah, S. Ouhaibi, N. Belouaggadia, and K. Mansouri, "Energy consumption forecasting and thermal insulator selection with random forest regression," *Sci. Afr.*, vol. 29, Sep. 2025, doi: 10.1016/j.sciaf.2025.e02870.