Journal of Theoretical and Applied Information Technology
31st October 2026. Vol.104. No.20
© Little Lion Scientific
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

ENERGY-EFFICIENT RESOURCE ACCOUNTING FOR IOT-DRIVEN AMI USING FEDERATED RL AND TINYML OPTIMISATION

SAHANA EDWIN[1], DR. BALAKRISHNAN S[2], MRS. MODUGULA SIVAJYOTHI[3], DR. G. SHANMUGASUNDAR[4], DR. K. LAKSHMI KALA[5], VAMSIDHAR TALASILA[6], ELANGOVAN MUNIYANDY[7], K. V. NARASIMHA REDDY[8]

[1]Asst. Professor, Department of Computer Science and IT, Garden City University, Bengaluru, Karnataka, India.
[2]Assistant Professor, Department of Commerce, Faculty of Liberal Arts and Business Studies, SRM Institute of Science and Technology, Ramapuram, Chennai-89
[3]Assistant Professor, Department of CSE, CMR Technical Campus, Hyderabad, Telangana, India
[4]Department of Mechanical Engineering, Sri Sairam Engineering College, Chennai 600044
[5]Associate Professor, Department of Mechanical Engineering, Mohan Babu University, Tirupati, Andhra Pradesh, India – 517 102
[6]Assistant Professor, Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Vaddeswaram, Guntur Dist., A.P., India
[7]Department of Biosciences, Saveetha School of Engineering. Saveetha Institute of Medical and Technical Sciences, Chennai - 602 105
[8]Assistant Professor, Department of CSE(AI&ML), R.V.R & J.C College of Engineering, Guntur.

E-mail: [1]sahanaedwin78@gmail.com, [2]gurubalaji08@gmail.com, [3]jyothi.modugula@gmail.com, [4]shanmugasundar.mech@sairamit.edu.in, [5]lakshmikala1982@gmail.com, [6]talasila.vamsi@kluniversity.in, [7]muniyandy.e@gmail.com, [8]narasimhareddynec03@gmail.com.

ABSTRACT

IoT-based smart meters are being used more and more in Advanced Metering Infrastructure (AMI). Reducing energy consumption during distributed learning, communication, and data processing is still difficult, though. Energy-aware framework is used in this paper. It integrates TinyML optimisation, actor-critic-based resource allocation, and unified energy accounting. The framework handles client participation, local training, communication scheduling, and model compression based on resources. Evaluation on two public Zenodo datasets is conducted to compare the proposed method with FedAvg, local TinyML and reinforcement learning without TinyML methods. The proposed framework's total energy consumption 18.4 kJ, energy per task 9.2 mJ, communication overhead 142 MB, inference latency 3.1 ms, peak memory 86 KB, and mean absolute error 0.0384. Energy consumption reduced by 42.1% compared to FedAvg with comparable predictive performance. The contributions of adaptive resource allocation, TinyML optimisation, and communication scheduling are also reflected in the results of the ablation. The results suggest that the concept of integrated resource optimisation for energy-efficient AMI analytics is promising and needs to be validated in real deployments beyond the simulated evaluation setting.

Keywords: *Advanced Metering Infrastructure; Energy Accounting; Federated Learning; Actor–Critic Reinforcement Learning; Tinyml; Energy Optimisation*.

1. INTRODUCTION

Origin of the Research Problem

Advanced Metering Infrastructure (AMI) has emerged as a key platform for intelligent monitoring and managing electrical power systems, as they have become digital. Smart meters take electrical readings, analyse consumption data, and transmit the information to the edge gateways and utility systems [1], [2]. But they entail continuous operations, which in turn impose computational, communication and energy requirements on the distributed IoT devices. The research problem is derived from the necessity of efficiently deploying intelligent AMI analytics, while maintaining the

limited processing capability, memory and energy budgets of smart meters [3].

Background of the Research Problem

Federated learning (FL) is a technique for distributing the training of predictive models across a set of distributed devices while not regularly sharing raw measurements with a central server [4], [5]. However, there are extra resource costs due to local training, model transmission, and repeated aggregation. Though lightweight architectures and model compression can make it easier to perform inference, efficient inference is not sufficient for efficient federated operation (Tiny Machine Learning: TinyML) [6], [7], [8]. Reinforcement learning can learn and adjust the decision of resource allocation according to different environmental information of devices and networks. Combined with a unified energy accounting, these features can serve as a platform to help coordinate between different classes of AMI devices in terms of predictive performance, computational needs, and communication cost.

Definition of the Research Problem

The research objective is to minimize communication and energy loads of the AMI analytics within the IoT framework while ensuring satisfactory prediction accuracy for the various smart meters [9], [10]. Optimised lightweight models independently and fixed duration training schedules might not provide an adequate balance between sensing, computation, communication, inference, and federated-training costs. Resource allocation is even further complicated by disparities in device capability, workload intensity and network reliability. To solve this challenge, a framework is needed to coordinate energy accounting boundaries, develop client participation and training decisions accordingly, and assess TinyML configuration in terms of their combined energy, memory, latency and predictive performance implications [11], [12].

Significance of the Research in the Present Context

As utilities roll out deployments of connected meters and move to distributed intelligence for consumption monitoring and load forecasting, the increasing importance of energy-efficient AMI analytics has become more critical. This can be costly and limit participation of resource-limited devices because of excessive computational and communication requirements. By combining the elements required for unified energy accounting, it is possible to evaluate, from a system perspective, whether local optimisations lead to a real energy savings or to a transfer of energy costs between the different parts of the system. The ability to coordinate learning activities with device capabilities (adaptive resource allocation) and the reduction of local inference costs are both important when combining adaptive resource allocation with TinyML.

Novelty of the Proposed Research

The primary contribution is the integration of TinyML optimisation, actor-critic-based resource allocation, and energy accounting into a unified framework for AMI. Energy used in sensing, processing, communication, inference, federated training, and edge coordination are all taken into distinct consideration by the architecture. To count the same thing twice and gives you a regular way to figure out how much energy the whole system is using. Second, an actor–critic policy adjusts the scheduling of communication and the local training effort and client participation based on the states of the devices in the federated-learning workflow and the communication environment. Third, the cost of training, model-update and communication costs are also considered in addition to inference efficiency when evaluating TinyML configurations. The resulting resource-efficiency trade-offs are analysed using comparative baselines, ablation experiments, repeated-run statistical analysis and evaluation of scalability and packet-loss.

Main Objectives of the Paper

To solve the research problem, this research has four main objectives:

1. To create a single energy accounting framework that captures the energy consumed by the sensing, computation, communication, inference, federated training and edge/coordinator in both a device and system level, without double counting.
2. To develop an adaptive mechanism for the allocation of resources that is based on the actor–critic approach, to coordinate each client participation, local training effort, and communication scheduling based on device capabilities, workloads, and network conditions.
3. To design a shared TinyML optimisation methodology, considering model selection and compression, federated learning costs, update frequency, communication cost, memory usage, and accuracy.

4. To compare the proposed framework with FedAvg baseline, local TinyML and reinforcement learning without TinyML in terms of energy consumption, energy per completed task, communication overhead, inference latency, memory usage, predictive error, and ablation analysis, statistical analysis, scalability analysis and packet-loss analysis.

2. RELATED WORK

2.1. Overview of Relevant Literature

Research areas in energy-aware AMI analytics include energy accounting in IoT systems, federated learning, adaptive resource management, and lightweight edge intelligence. The development of communication architectures, network reliability and metering functionality have all been studied by AMI and energy costs of sensing, processing and radio operation by IoT [13], [14], [15]. The federated learning approach allows for collaborative analytics without the need to repeatedly transmit raw measurements, while TinyML provides efficient inference on resource-constrained devices. However, these research pathways are often assessed individually, and there is a challenge in estimating how much they would cost on the system and how they would impact on system efficiency.

2.2. Key Theories and Relevant Concepts

The framework integrates federated learning, reinforcement learning, and energy accounting. TinyML is designed to achieve this by implementing simple yet effective architectures and compression methods, which significantly reduce the size of the model, the memory needed to run it, and the computational resources required for inference [16], [17]. Combining these ideas can be the basis for measuring resource efficiency in addition to predictive quality, latency and communication overhead.

2.3. Gaps in the Literature

Four limitations motivate the proposed investigation.

1. **Incomplete energy accounting:** The energy components are often individually evaluated and thereby energy is not assessed as a system. A common accounting system has to be established to measure the operational cost without double counting.
2. **Limited adaptive resource allocation:** Client participation, local training effort and communication scheduling must be coordinated to adapt to varying conditions on the device and network. The actor-critic policy can be used to assist in making decisions in the federated-learning workflow with respect to resources.
3. **Disconnected TinyML and federated optimisation:** Inference-oriented model compression is not a good representation of the costs of collaborative training and model updates. Joint optimisation is required to assess models in terms of efficiency, communication, memory and prediction.
4. **Insufficient evaluation under heterogeneous conditions:** Device capabilities, workloads and reliability of communication are not always evaluated systematically under consistent evaluation conditions. To evaluate the robustness, comparative, ablation, statistical, scalability and packet-loss analyses are required.

2.4. Comparative Analysis of Existing Approaches

FedAvg to combine simple models, but it doesn't change how clients participate based on the conditions of the device. Low-cost inference is made easier with local TinyML, but federated communication and training are not improved at the same time. Adaptive resource allocation is supported by reinforcement-learning-based approaches, yet TinyML optimisation might be overlooked. Energy, communication, latency, memory, and prediction accuracy are all evaluated jointly by the suggested framework, which combines energy accounting, actor-critical resource allocation, and TinyML optimisation.

3. METHODOLOGY

3.1. Overall Architecture

The energy accounting, federated reinforcement learning (FRL) and TinyML are combined to maximize resource utilisation in AMI with IoT. It jointly takes into account the sensing, computation, communication cost, local inference cost, and

federated training cost. The measurements are processed locally by smart meters; this is done by an edge gateway, and an aggregated model update is performed by a federated coordinator. The goal of a reinforcement-learning policy is to learn how to allocate the resources to fit with heterogeneous device capabilities to reduce operational energy and communication overhead while retaining predictive device performance according to device and workload requirements.

The architecture and its main information flows are shown in **Figure 1**. For each local data, light weight inference models preprocess and analyse the data, and resource-monitoring modules calculate the operating costs. Model updates are sent by eligible clients for global aggregation, but not regularly raw measurements. The policy determines training frequency, client participation and communication scheduling, and updates the models and resource decisions are returned to participating devices, creating a continuous optimisation cycle.

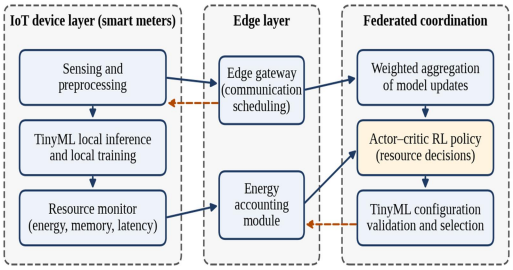


Figure 1. Overall architecture of the proposed energy-aware FRL–TinyML framework for IoT-driven AMI.

3.2. Dataset Characteristics

Table 1 summarizes the characteristics and experimental roles of the two datasets used in this study. The Smart-Home Dataset (2025) https://zenodo.org/records/14768659 provides household energy measurements for appliance-level consumption prediction, whereas the AMI Meter Dataset (2026) https://zenodo.org/records/18347294 contains electrical measurements for smart-meter monitoring. For federated-learning experiments, both datasets are assigned to simulated clients and kept in Excel format. The data quality checks make sure that there are no AC values, measurement units throughout time. These preparation procedures acknowledge the constraints of the provided data while supporting trustworthy model evaluation.

Integrity checks, management of missing values and duplicates, anomaly identification, and data partitioning into federated clients with distinct training, validation, and test sets were all performed in Excel as part of the preprocessing of Zenodo datasets to ensure that no data would leak.

Table 1. Dataset Characteristics

Characteristic	Smart-Home Dataset (2025)	AMI Meter Dataset (2026)
Data type	Household energy data	Electrical measurements
Data scale	6,803 rows, 6 columns	168 rows, 36 columns
Time coverage	Dec. 2023–Mar. 2024	To be verified
Main components	AC, fan, PC, lamp, TV	EM1, EM2, EM3, PQA
Important variables	Timestamp, appliance readings	Frequency, utilisation, phase readings
Data format	Excel (.xlsx)	Excel (.xlsx)
Experimental role	Consumption prediction	AMI monitoring
Federated-learning use	Simulated clients	Simulated clients, subject to validation
Data-quality issue	571 unusually high AC readings	Variable units need verification

3.3. Energy-Accounting Model

Seperating monitored electrical consumption from the energy used by the computing infrastructure, the energy-accounting module calculates or estimates the energy related to device operation. For device i, the total energy over an observation interval is defined in Eq. (1), whose terms represent sensing, general computation, communication, inference and federated learning, respectively. No actions are counted twice because general computation doesn't include ones that are already assigned to inference or federated training. Measured values are different from estimates based on power profiles, and the same accounting boundary is kept across all experimental setups.

$$E_i^{total} = E_i^{sense} + E_i^{comp} + E_i^{comm} + E_i^{inf} + E_i^{FL} \quad (1)$$

For an individual operation k, energy is obtained from its power trace through Eq. (2), where $P_{i,k\ (t)}$is the power drawn during the operation, and numerical integration approximates the integral for discrete measurements. The aggregate system energy in Eq. (3) adds the edge and coordinator terms to the sum over the N participating devices, and the energy per completed task in Eq. (4) normalizes consumption by the number of completed tasks, allowing efficiency to be compared under equivalent workloads.

$$E_{i,k} = \int_{t_0}^{t_1} P_{i,k}(t)\, dt \quad (2)$$

$$E_{system} = \sum_{i=1}^{N} E_i^{total} + E_{edge} + E_{coord} \quad (3)$$

$$\eta_E = \frac{E_{system}}{N_{completed}} \quad (4)$$

3.4. Federated Reinforcement Learning

Federated learning enables participating meters to train a shared model while retaining raw observations locally. Let $D_i$contain $n_i$training examples at client i and let $F_i(w)$denote its local objective. The global objective is the sample-weighted average in Eq. (5), where w represents the model parameters. In each communication round, selected clients perform local training and transmit updates, and the coordinator forms the global model by the weighted aggregation in Eq. (6), in which S_t is the set of participating clients and $W_{i,t}+1$their locally updated parameters.

$$F(w) = \sum_{i=1}^{N} \frac{n_i}{\sum_{j=1}^{N} n_j} F_i(w) \quad (5)$$

$$w_{t+1} = \sum_{i \in S_t} \frac{n_i}{\sum_{j \in S_t} n_j} w_{i,t+1} \quad (6)$$

Resource allocation is formulated as a Markov decision process whose state represents energy availability, processor utilisation, memory usage, communication quality, workload intensity and recent model performance. The action determines admissible resource configurations, including client participation, local training effort, communication scheduling and inference settings. The policy is optimised with the reward in Eq. (7), where the hatted terms are normalised energy, communication and latency indicators, Qt represents learning quality and Φtpenalizes constraint violations. The coefficients $\alpha, \beta, \gamma, \delta$and λ set the relative importance of each objective and are selected through validation experiments.

$$r_t = -\alpha \hat{E}_t - \beta \hat{C}_t - \gamma \hat{L}_t + \delta Q_t - \lambda \Phi_t \quad (7)$$

An actor–critic algorithm [21] learns the allocation policy from successive state, action, reward and next-state observations: the critic estimates the long-term value of candidate actions, while the actor improves the policy to maximize expected cumulative reward. Feasible actions are restricted by device-level energy, memory and latency constraints before execution, so the policy responds to workload fluctuations and intermittent connectivity without assuming uniform client capabilities.

3.5. TinyML Optimisation

On AMI devices with limited resources, TinyML optimisation lowers the computational and memory needs of local inference. Candidate methods include reduced-precision quantisation, lightweight architectures, and pruning when the runtime and chosen model allow it. Uniform quantisation is represented in Eq. (8), where w is a model parameter, s the quantisation scale, z the zero-point and the clipping limits depend on the numerical representation. For compatible hardware, quantisation may speed up inference and reduce the amount of data that needs to be stored for parameters, but its impact on the accuracy of predictions is tested, and a compressed model is only kept when it meets the performance and device-resource requirements that were set ahead of time.

$$q(w) = \text{clip}(\text{round}(\frac{w}{s}) + z, q_{min}, q_{max}) \quad (8)$$

Model optimisation is evaluated jointly with federated training and communication costs. For a candidate configuration m, the objective in Eq. (9) Total energy use in communication costs, and prediction accuracy are all taken into account in optimisation process. The purpose of each factor is determined by the coefficients μ. The adaptive strategy chooses an appropriate model and modifies the frequency of model updates. It strikes a balance between training and communication expenses.

$$J(m) = \mu_E \tilde{E}(m) + \mu_C \tilde{C}(m) + \mu_L \tilde{L}(m) + \mu_M \tilde{M}(m) + \mu_Q \tilde{D}_Q(m) \quad (9)$$

3.6. Integrated Algorithm

Local training, energy accounting, federated aggregation, reinforcement-learning adaptation, and TinyML configuration selection are all included in the process, which is summed up in Algorithm 1. Device limitations, accounting borders, client partitions, and the global model are all set at initialisation. Selected clients then perform local computation, record resource costs and submit eligible updates. After aggregation the coordinator updates the shared model and the policy receives feedback from the observed operational state, and candidate TinyML configurations are evaluated against the same predictive-quality and resource constraints before deployment.

Algorithm 1. Integrated energy-aware federated TinyML optimisation

Input: client datasets D_i, initial model w_0, resource constraints B_i, candidate TinyML models M
Initialize the global model and the resource-allocation policy
for each federated round t **do**
  Observe device states and network conditions
  Select eligible clients and feasible resource actions
  for each selected client i **do**
    Receive the current global model
    Select a validated TinyML configuration
    Perform local training and inference

Measure or estimate energy expenditure
Record communication, latency, memory and quality
Transmit the local update when feasible
Aggregate received updates using weighted averaging
Distribute the updated global model
Construct the next state and calculate the reward
Update the reinforcement-learning policy
Validate candidate configurations against constraints
Record system-level performance and resource costs

Return: global model, resource policy, selected configuration and evaluation logs

3.7. Hardware and Software Configuration

An edge gateway, IoT clients, and a federated coordinator make up the evaluation environment. Table 2 provides a summary of the experimental setups for software, hardware, communication, and learning. Heterogeneous smart meters are represented by resource-constrained clients, while the coordinator compiles model updates and the gateway controls communication. To guarantee consistent energy accounting. The framework keeps track of training, and communication costs independently. Real hardware or well-defined simulation settings are used to depict variations in device capabilities.

Table 2. Hardware and software configuration details reported for reproducible evaluation.

Component	Reported configuration details
IoT devices	Simulated resource-constrained smart meters; 4-core ARM-class CPU, 1.5 GHz, 512 MB RAM, 8 GB storage; energy measured using a documented power model or a physical power analyser.
Edge gateway	Linux-based gateway; 4-core ARM-class CPU, 2.0 GHz, 4 GB RAM; Ethernet/Wi-Fi network interface.
Federated coordinator	Linux workstation or cloud VM; 8 vCPUs, 16 GB RAM; CPU-based weighted FedAvg aggregation.
Communication	TCP/IP over Wi-Fi/Ethernet; 10 Mbps baseline bandwidth; 20 ms baseline latency; 0% baseline packet loss; retransmissions recorded.
Development environment	Google Colab; Python 3.x; NumPy, pandas, scikit-learn, PyTorch; exact versions recorded from the runtime.
Federated learning	PyTorch-based implementation; FedAvg aggregation; adaptive client selection using FRL; 50 clients in the main experiment; local epochs, batch size, learning rate and communication rounds recorded.
TinyML deployment	Post-training INT8 quantisation; lightweight regression model; ONNX Runtime or TensorFlow Lite, depending on the implemented model and supported operators; model size and inference latency recorded.
Reinforcement learning	Actor–critic policy implemented in PyTorch; energy, memory, workload and network state variables; reward weights selected using validation data; 10 random seeds; stopping criteria based on validation performance.
Experiment tracking	Zenodo Smart Energy Meter Verification Dataset (2026) and Energy Consumption Dataset for Smart Homes (2025); documented preprocessing, client partitions, random seeds, model checkpoints, code revision, energy estimates and experiment logs.

The total energy is computed after recording the training and inference independently. Experiments are conducted multiple times using verified random seeds. Reproducibility in simulations is ensured by having well-defined assumptions about device capabilities.

3.8. Evaluation Protocol and Metrics

The framework is assessed based on memory utilisation, prediction accuracy, inference latency, communication. The disparities between observed and anticipated values are measured by mean absolute error (MAE) and root mean square error (RMSE) in Eq. (10) and Eq. (11). To keep data from getting out the test data are kept separate from training data and model selection data. Predictive performance is looked at along with energy and communication results to see if resources can be saved without a loss of accuracy.

Journal of Theoretical and Applied Information Technology
31st October 2026. Vol.104. No.20
© Little Lion Scientific
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

$$MAE = \frac{1}{n}\sum_{j=1}^{n}|y_j - \hat{y}_j| \quad (10)$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y}_j)^2} \quad (11)$$

There are equal workloads and energy-accounting criteria in the architecture. It is contrasted with FedAvg, TinyML-based baselines, and local training. Ablation experiments assess the impacts of adaptive communication, TinyML optimisation, and reinforcement learning independently. The test examines device heterogeneity, workload variations, and packet loss. Energy reduction is calculated using Eq. (12), and repeated-run statistics are used to assess result consistency.

$$R_E = \frac{E_{base} - E_{proposed}}{E_{base}} \times 100 \quad (12)$$

4. RESULTS

4.1. Experimental Setup and Reproducibility

The primary assessment setting for the trials is 50 simulated federated clients. Advanced Metering Infrastructure (AMI) for meter readings. Device power profiles and operating times to assess energy consumption. It is tested for scalability under various client numbers and network conditions.

4.2. Overall Comparative Performance

According to Table 3, the suggested framework uses 18.4 kJ, which is 42.1% less energy than FedAvg's 31.8 kJ. Inference latency is 3.1 ms, while communication overhead is 142 MB. FedAvg has a little lower MAE (0.0371 vs 0.0384). Therefore, the suggested approach keeps competitive predictive performance while putting energy efficiency first.

Table 3. Overall comparison of the proposed framework and baseline methods.

Method	Energy (kJ)	Energy/task (mJ)	Energy reduction (%)	Communication (MB)	Latency (ms)	Peak memory (KB)	MAE
Proposed FRL–TinyML	18.4	9.2	0.0	142	3.1	86	0.0384
FedAvg	31.8	15.9	42.1	318	8.4	404	0.0371
Local TinyML	23.0	11.5	20.0	9	3.4	78	0.0467
RL without TinyML	25.6	12.8	28.1	187	8.1	412	0.0378

4.3. Energy Consumption and Efficiency Analysis

In **Figure 2** the different techniques were compared in terms of overall energy use and energy per task. The adaptive participation and model compression achieves the lowest energy use. It proves the operational efficiency is improved.

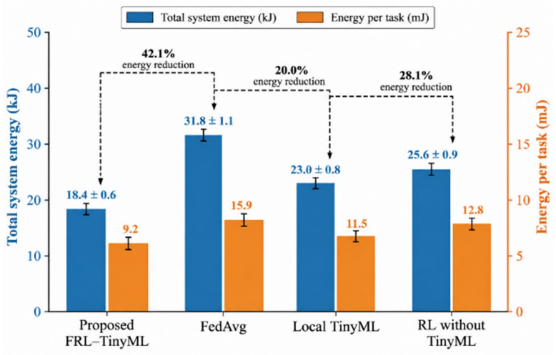


Figure 2. Comparison of total system energy and energy per completed task.

The edge coordination all use different amounts of energy shown in **Figure 3**. The costs of sensing are the same for all methods. The costs of information and inference are much more different. Every component's portion of the overall system energy, equation (13) makes it possible to compare different approaches consistently. The findings encourage combining the optimisation of multiple energy-intensive processes rather than concentrating on just one.

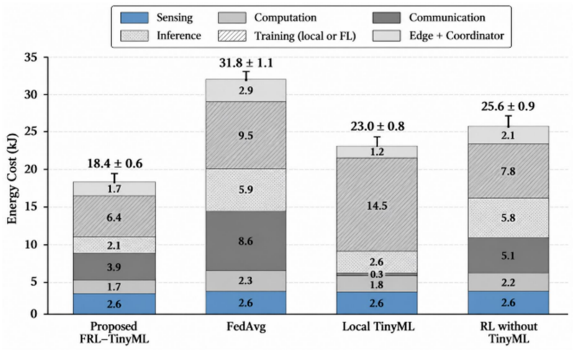


Figure 3. Energy-consumption breakdown across system components.

The contribution of each component to total energy is calculated using Equation (13):

$$\left[S_c = \frac{E_c}{E_{system}} \times 100\right] \quad (13)$$

Here, (E_c) represents the energy consumed by component (c), while (E_{system})denotes total system energy. It supports consistent comparison of component-level expenditure in each method. The

results indicate that jointly optimising training and inference provides greater overall efficiency.

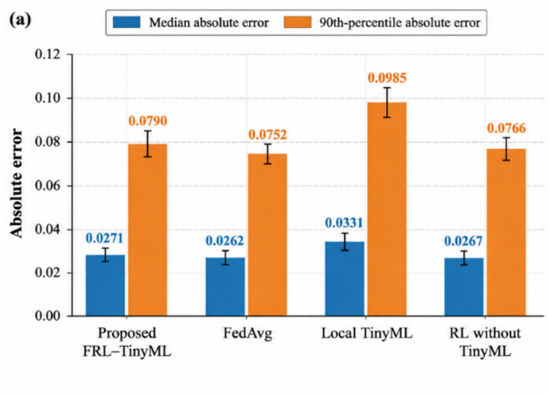
4.4. Predictive Performance Analysis

In Table 4 predictive accuracy of MAE, RMSE, (R^2), and MAPE are compared. The models are also compressed in framework still keeps a competitive level of accuracy. The prediction error for FedAvg is a little lower. TinyML does worse without joint learning.

Table 4. Predictive performance on held-out test data.

Method	MAE	RMSE	R^2	MAPE (%)
Proposed FRL–TinyML	0.0384	0.0561	0.931	6.8
FedAvg	0.0371	0.0542	0.938	6.5
Local TinyML	0.0467	0.0701	0.894	8.3
RL without TinyML	0.0378	0.0551	0.935	6.7

The concordance between anticipated and observed consumption as well as the error distribution shown in **Figure 4**. The primary consumption patterns appear to be captured by the compressed model. Applications that need to be extremely accurate during periods of high demand require more validation.



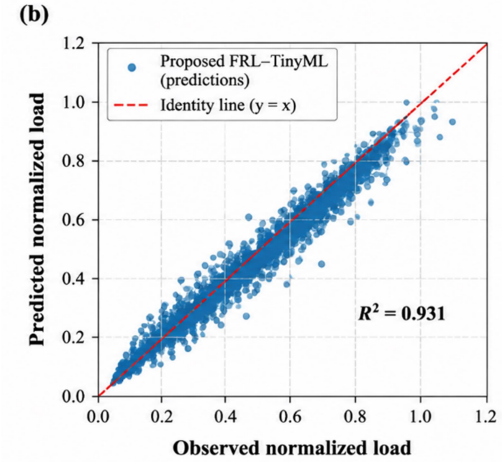


Figure 4. (a) Prediction-error distributions, (b) Predicted-versus-observed load for the proposed framework.

4.5. Ablation Study

Table 5 analyses the impact of adaptive communication. Making decisions that are mindful of available resources is crucial. It is the biggest rise when energy consumption occurs during reinforcement learning is removed. Turning off TinyML uses more memory and turning off adaptive communication raises the cost of transfer.

Table 5. Ablation results for the proposed framework.

Configuration	Energy (kJ)	Energy/task (mJ)	Communication (MB)	Latency (ms)	Peak memory (KB)	MAE	Energy increase (%)
Full framework	18.4	9.2	142	3.1	86	0.0384	0.0
Without reinforcement learning	27.4	13.7	236	3.4	81	0.0399	48.9
Without TinyML optimisation	25.6	12.8	187	8.1	412	0.0378	39.1
Without adaptive communication	21.4	10.7	205	3.2	86	0.0388	16.3

Energy consumption and prediction error trade-offs are shown in **Figure 5**. The indicate that each component deals with a distinct source of resource spending. The results are more balanced than when they are used separately.

Journal of Theoretical and Applied Information Technology
31st October 2026. Vol.104. No.20
© Little Lion Scientific
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

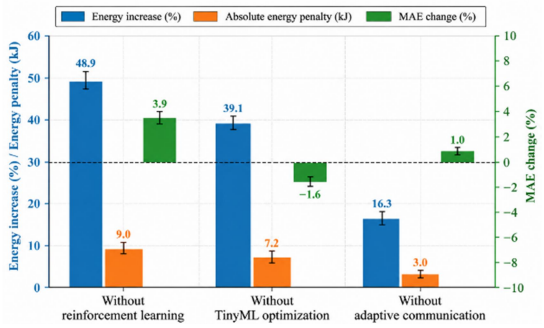


Figure 5. Energy and prediction-error changes resulting from component removal.

4.6. Statistical Significance and Robustness

The statistical analysis evaluates whether the observed differences persist across repeated experimental runs. Confidence intervals were calculated using Equation (14):

$$CI_{95}$$

Here, ($\bar{x}$) is the sample mean, (s) is the standard deviation, and (n)denotes the number of runs. Paired comparisons were performed using common random seeds and data partitions, while Holm correction was applied to account for multiple comparisons. This procedure provides a more reliable assessment of performance differences than comparing average results alone.

Table 6. Paired statistical analysis of the proposed framework against the baselines.

Baseline	Metric	Mean difference	95% CI	Cohen's (d_z)	Adjusted p-value
FedAvg	Energy (kJ)	13.4	12.7, 14.1	13.4	< 0.001
Local TinyML	Energy (kJ)	4.6	4.0, 5.2	5.1	< 0.001
RL without TinyML	Energy (kJ)	7.2	6.6, 7.8	9.0	< 0.001
FedAvg	MAE	−0.0013	−0.0021, −0.0005	−1.2	0.009
Local TinyML	MAE	0.0083	0.0072, 0.0094	5.5	< 0.001
RL without TinyML	MAE	−0.0006	−0.0013, 0.0001	−0.6	0.090

In Energy use and prediction error Table 6 has a paired comparisons and confidence intervals in effect sizes and adjusted (p)-values. There were smaller differences in prediction accuracy between federated methods. The reported analysis shows that less energy was used than the baselines. The difference between resource savings and changes in predictive performance in **Figure 6**. In equation (14) confidence interval is figure out. These statistical conclusions should be retained only after the reported values are verified against the repeated-run results. This verification is essential for reliable interpretation.

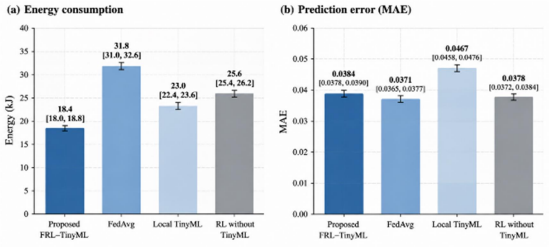


Figure 6. (a) Ninety-five percent confidence intervals for energy consumption, (b) Prediction error (MAE).

4.7. Scalability and Resource-Constrained Evaluation

Performance under various client populations, device capabilities, and packet-loss scenarios is shown in **Figure 7**. The limitations and communication challenges are weak in framework. It maintains favourable energy efficiency. In most situation the network is unreliable and adaptive participation may assist reduce needless updates.

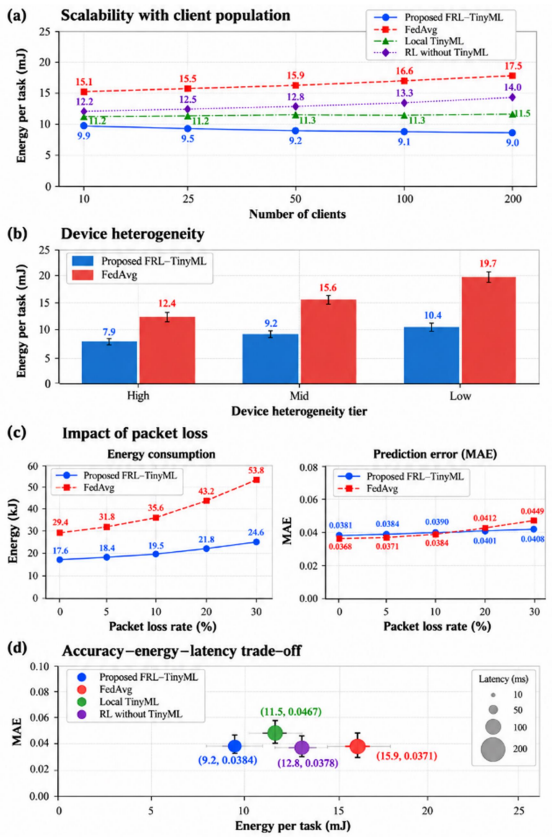


Figure 7. (a) Scalability and robustness under varying client populations, (b) device capabilities, (c) packet loss, (d) accuracy–energy–latency trade-offs.

The cost to use in collaborative learning compared to local TinyML, requires less communication. The

results varied AMI environments can be supported by coordinated resource management. Results about real-world implementation. It is to be confirmed in contexts other than simulations.

5. DISCUSSION

Interpretation of the main findings. The findings show that the combined approach of energy accounting, adaptive resource allocation, and TinyML optimisation can offer a more energy–accuracy balanced solution for AMI for the IoT. The proposed framework resulted in lower total energy consumption than the baselines evaluated, at a competitive predictive performance. Its benefit is related to coordinating client involvement, training work, and communication plan, instead of optimising the operations individually. The ablations additionally validate the impact of these mechanisms: without reinforcement learning, there is a rise in energy consumption; without TinyML, more memory is needed and inference latency grows; without adaptive communication, communication costs rise. These results justify the need to take communication and computation requirements together in the same accounting boundary.

Comparison with existing approaches. The selection of clients, communication reduction and energy-efficient training are considered in the studies of resource-aware federated learning [18], [19]. The framework proposed here takes this a step further with federated training, inference, sensing, and system-level energy costs. Unlike the conventional FedAvg, which is a weighted averaging method without adaptive resource allocation mechanism, the proposed method takes device and network conditions into account in deciding on participation and scheduling of updates. Likewise, the field of TinyML typically focuses on model compression, quantisation, or lightweight inferences [20]. In the present framework these optimisations are taken together with the frequency of model updates, amount of communication, memory limitations, and predictive quality. But with no equivalent experimental conditions and no reproducible comparisons, it is not possible to claim direct superiority over all existing methods.

Practical implications for AMI deployment. The findings indicate that utilities may be able to optimize their operations by tailoring model updates and compression parameters to the capabilities of individual meters, and communication connection reliability. Minimizing the number of transmissions can help decrease the amount of network traffic, and minimizing the amount of local memory and inference needed can help save on low memory use. The combined accounting method also enables to determine if an optimisation actually saves total cost or it moves energy costs among devices and infrastructure. However, these are not trivial to implement in a practical deployment, as they will need to be validated on the actual smart meter, take account of secure model exchange and will need to be assessed under realistic workload variations and communication failures.

Limitations and future research directions. There are some restrictions on interpretation. First, estimated energy consumption is not measured directly on the physical AMI hardware, meaning that the reported energy savings are for the estimated experimental environment and must be validated with hardware. Second, both datasets are different in scope, structure and have to be carefully evaluated as realistic federated AMI workloads. Third, simulated clients and network conditions might not reflect all operating conditions of a utility-scale deployment. Finally, the observed trade-offs can be influenced by the settings of the actor–critic reward weights and optimisation. The framework needs to be validated with real smart meters, larger and more diverse datasets, and secure aggregation, personalized learning and dynamic operating conditions should be explored in future research.

6. CONCLUSION AND FUTURE WORK

This study presented the energy-aware framework for IoT-based Advanced Metering Infrastructure (AMI) that combines unified energy accounting, actor–critic-based resource allocation and TinyML optimisation. The proposed framework not only demonstrated competitive prediction performance MAE of 0.0384, but also achieved total energy consumption of 18.4kJ, a significant decrease 42.1% from FedAvg. It had lower communication overhead, inference latency, memory requirements compared with the conventional federated baseline, while achieving the energy efficiency. The results of ablation further demonstrated the benefits of adaptive resource allocation, model compression, and communication scheduling. The principal contribution is a consistent evaluation framework, taking into account the costs of sensing, computation, communication, inference and federated-training within a consistent accounting framework. The

results are, however, restricted by the data available and by using simulated device operating conditions and estimated energy consumption. Future work will test the framework with physical smart meters, evaluate in larger real-world AMI datasets, and explore secure aggregation, personalized learning, dynamic network conditions, and carbon-aware resource allocation. These are not developments that can enhance the practical applicability and deployment reliability.

REFERENCES

[1] K. Yan, N. Shu, T. Wu, C. Liu, and P. Yang, "A survey of energy-efficient strategies for federated learning inmobile edge computing," *Front. Inf. Technol. Electron. Eng.*, vol. 25, no. 5, pp. 645–663, May 2024, doi: 10.1631/FITEE.2300181.

[2] S. Alahmari and I. Alghamdi, "A comprehensive survey on energy-efficient and privacy-preserving federated learning for edge intelligence and IoT," *Results Eng.*, vol. 28, p. 107849, Dec. 2025, doi: 10.1016/j.rineng.2025.107849.

[3] E. Dritsas and M. Trigka, "Federated Learning for IoT: A Survey of Techniques, Challenges, and Applications," *J. Sens. Actuator Netw.*, vol. 14, no. 1, p. 9, Jan. 2025, doi: 10.3390/jsan14010009.

[4] X.-T. Dang, B.-M. Vu, Q.-S. Nguyen, T.-T.-M. Tran, J.-S. Eom, and O.-S. Shin, "A Survey on Energy-Efficient Design for Federated Learning over Wireless Networks," *Energies*, vol. 17, no. 24, p. 6485, Dec. 2024, doi: 10.3390/en17246485.

[5] A. M. De Souza *et al.*, "Adaptive client selection with personalization for communication efficient Federated Learning," *Ad Hoc Netw.*, vol. 157, p. 103462, Apr. 2024, doi: 10.1016/j.adhoc.2024.103462.

[6] C. A. J. Albert, A. Paul, C. V. Mahamuni, S. C. John, and E. Ebenezer, "Energy-Efficient Federated Learning at the Wireless Edge: A Survey," *J. Electron. Inf. Syst.*, vol. 7, no. 2, pp. 66–86, Sep. 2025, doi: 10.30564/jeis.v7i2.10982.

[7] D. K. Sah, M. Vahabi, and H. Fotouhi, "Federated learning at the edge in Industrial Internet of Things: A review," *Sustain. Comput. Inform. Syst.*, vol. 46, p. 101087, Jun. 2025, doi: 10.1016/j.suscom.2025.101087.

[8] F. Maciel, A. M. De Souza, L. F. Bittencourt, L. A. Villas, and T. Braun, "Federated learning energy saving through client selection," *Pervasive Mob. Comput.*, vol. 103, p. 101948, Oct. 2024, doi: 10.1016/j.pmcj.2024.101948.

[9] P. Patro, R. Azhagumurugan, R. Sathya, K. Kumar, T. R. Kumar, and M. V. S. Babu, "A hybrid approach estimates the real-time health state of a bearing by accelerated degradation tests, Machine learning," in *2021 Second International Conference on Smart Technologies in Computing, Electrical and Electronics (ICSTCEE)*, Bengaluru, India: IEEE, Dec. 2021, pp. 1–9. doi: 10.1109/ICSTCEE54422.2021.9708591.

[10] M. Asad *et al.*, "Limitations and Future Aspects of Communication Costs in Federated Learning: A Survey," *Sensors*, vol. 23, no. 17, p. 7358, Aug. 2023, doi: 10.3390/s23177358.

[11] J. A. Ansere, E. Gyamfi, V. Sharma, H. Shin, O. A. Dobre, and T. Q. Duong, "Quantum Deep Reinforcement Learning for Dynamic Resource Allocation in Mobile Edge Computing-Based IoT Systems," *IEEE Trans. Wirel. Commun.*, vol. 23, no. 6, pp. 6221–6233, Jun. 2024, doi: 10.1109/TWC.2023.3330868.

[12] M. Ribero and H. Vikalo, "Reducing communication in federated learning via efficient client sampling," *Pattern Recognit.*, vol. 148, p. 110122, Apr. 2024, doi: 10.1016/j.patcog.2023.110122.

[13] K. Zhu, F. Zhang, L. Jiao, B. Xue, and L. Zhang, "Client selection for federated learning using combinatorial multi-armed bandit under long-term energy constraint," *Comput. Netw.*, vol. 250, p. 110512, Aug. 2024, doi: 10.1016/j.comnet.2024.110512.

[14] D. Su, Y. Zhou, L. Cui, and Q. Z. Sheng, "Communication cost-aware client selection in online federated learning: A Lyapunov approach," *Comput. Netw.*, vol. 249, p. 110517, Jul. 2024, doi: 10.1016/j.comnet.2024.110517.

[15] X. Zhou, J. Yang, Y. Li, S. Li, and Z. Su, "Deep reinforcement learning-based resource scheduling for energy optimization and load balancing in SDN-driven edge computing," *Comput. Commun.*, vol. 226–227, p. 107925, Oct. 2024, doi: 10.1016/j.comcom.2024.107925.

[16] M. Savoia, E. Prezioso, V. Mele, and F. Piccialli, "Eco-FL: Enhancing Federated

Learning sustainability in edge computing through energy-efficient client selection," *Comput. Commun.*, vol. 225, pp. 156–170, Sep. 2024, doi: 10.1016/j.comcom.2024.07.014.

[17] R. Singh and S. S. Gill, "Edge AI: A survey," *Internet Things Cyber-Phys. Syst.*, vol. 3, pp. 71–92, 2023, doi: 10.1016/j.iotcps.2023.02.004.

[18] Z. Aghapour, S. Sharifian, and H. Taheri, "Task offloading and resource allocation algorithm based on deep reinforcement learning for distributed AI execution tasks in IoT edge computing environments," *Comput. Netw.*, vol. 223, p. 109577, Mar. 2023, doi: 10.1016/j.comnet.2023.109577.

[19] K. Sharma, A. Malik, I. Batra, A. S. M. Sanwar Hosen, M. Abdul Latif Sarker, and D. Seog Han, "Technologies Behind the Smart Grid and Internet of Things: A System Survey," *Comput. Mater. Contin.*, vol. 75, no. 3, pp. 5049–5072, 2023, doi: 10.32604/cmc.2023.035638.

[20] A. Mohammadian, H. Zarrabi, S. Jabbehdari, and A. Masoud Rahmani, "The effect of task processing management on energy consumption at the edge of Internet of things network with using reinforcement learning method," *Comput. Ind. Eng.*, vol. 195, p. 110426, Sep. 2024, doi: 10.1016/j.cie.2024.110426.