Journal of Communication Sciences and Information Technology (JCSIT)
An International Journal

Enhancing Edge Analytics Using TinyML and IoT: Toward Energy-Efficient and **Intelligent Data Processing in Distributed Environments**

[1*]Mariam M. Hagag, [2]Hesham F. A. Hamed, [3]Amr M. Sauber, [4]Mahmoud A. Mahdi

[1*]Department of Data Science, Faculty of Artificial Intelligence, Egyptian Russian University, Cairo, Egypt
mariam-hagag@eru.edu.eg

[2]Department of Artificial Intelligence, Faculty of Artificial Intelligence, Egyptian Russian University, Cairo, Egypt

[3]Mathematics and Computer Science Department, Faculty of Science, Menoufia University, Shebin El-Kom, Egypt

[4]Faculty of Computers and Information, Zagazig University, Zagazig, Egypt

Corresponding author: *mariam-hagag@eru.edu.eg*

ABSTRACT The convergence of Internet of Things (IoT) ecosystems with Tiny Machine Learning (TinyML) has redefined the paradigms of distributed analytics and computational intelligence. Traditional cloud-centric models impose high latency, excessive energy consumption, and increased privacy risks, limiting real-time responsiveness for mission-critical applications. This paper introduces an **integrated edge analytics framework leveraging TinyML-enabled IoT devices** for low-latency, energy-efficient, and adaptive decision-making. The proposed architecture unifies lightweight embedded learning, confidence-triggered inference offloading, and collaborative edge-to-cloud synchronization to enhance analytic performance at the periphery of the network. Empirical simulations conducted on embedded-class processors demonstrate latency reduction exceeding 50% and energy savings up to 42% compared to conventional edge–cloud paradigms, without compromising inference accuracy. This study contributes a scalable foundation for intelligent IoT infrastructures capable of performing dynamic analytics under constrained computational and communication environments, advancing the vision of sustainable and autonomous edge intelligence.

Keywords: TinyML, Edge Analytics, IoT, Embedded Intelligence, Energy Efficiency, Adaptive Offloading, Distributed Learning.

1. Introduction

Edge analytics brings computation close to data sources to reduce latency, lower bandwidth consumption, and preserve privacy in Internet-of-Things (IoT) deployments [1- 6]. Recent progress in **Tiny Machine Learning (TinyML)** makes on-device inference feasible on microcontrollers with only tens of kilobytes of RAM, enabling real-time analytics in sensing, wearables, and industrial monitoring. Frameworks such as TensorFlow Lite Micro (TFLM) provide portable inference runtimes for highly constrained devices 1, while optimization and compression techniques (quantization, pruning, operator fusion) further lower the footprint for Edge AI workloads [1-15]. Although TinyML reduces round-trip delays and data exposure, **non-stationary environments** remain a core challenge: sensor characteristics drift with time, operating conditions shift, and class priors change, degrading model accuracy if left unaddressed [6,7]. Edge analytics therefore benefits from **drift-aware** learning—

detecting changes and adapting models without cloud dependence. Online and incremental approaches (e.g., TinyOL) illustrate how limited on-device updates can sustain accuracy under evolving data streams [2]. This paper focuses on practical mechanisms that quantify and mitigate degradation of TinyML models in IoT settings, grounded in established drift-detection theory and contemporary Edge-AI practice [5-9].

The primary contributions of this research are as follows:

1. Development of a **multi-layered TinyML–IoT architecture** enabling intelligent and distributed analytics.
2. Design of a **confidence-triggered adaptive mechanism** for minimizing communication overhead without accuracy degradation.
3. Empirical evaluation of energy efficiency, latency reduction, and inference robustness using embedded hardware.
4. Comparative analysis of the proposed TEAF with cloud-based and classical edge analytics frameworks.

This study demonstrates that integrating TinyML within IoT ecosystems enables **autonomous, low-latency, and sustainable analytics**, establishing a foundation for scalable edge intelligence in smart infrastructures.

2. Related Work and Theoretical Background

Edge analytics & TinyML. Surveys position TinyML as a key enabler of edge intelligence across domains where sub-second response, energy efficiency, and local privacy are required [7,5]. TFLM describes a minimal, interpreter-based runtime that executes neural models on microcontrollers [1], while optimization surveys catalog quantization/architecture co-design techniques that trade accuracy for memory/latency at the edge [3]. **Online learning on devices** is emerging: TinyOL demonstrates incremental updates on microcontrollers to counter deployment drift [2].

Concept drift. Drift encompasses changes in the joint distribution p(x,y) (real drift) or p(x) (virtual drift), and is widely documented in data streams [6,7]. Foundational reviews synthesize detection and adaptation strategies, including error-rate monitoring, distribution tests, and ensemble resets [6,7]. **Detectors** such as DDM (error-rate thresholds) and ADWIN (adaptive windows with statistically bounded mean shifts) are canonical [6, 8]. Contemporary comparative studies show no universal winner; method efficacy depends on drift type (abrupt vs. gradual) and data imbalance, with false-alarm control crucial for sustained performance [9]. In resource-constrained settings, detectors with low memory and amortized O(1) updates are preferred for on-device use.

2.1 Edge Analytics and IoT Intelligence

Edge analytics has emerged as a transformative paradigm in which computational intelligence is relocated from centralized clouds to distributed IoT nodes, enabling data-driven decisions closer to the source [5, 9]. By executing analytics on-site, systems can significantly reduce communication latency and bandwidth usage—a crucial advantage in latency-sensitive applications such as industrial anomaly detection, mobile health monitoring, and real-time control loops [4, 10]. Early solutions, such as Microsoft Azure IoT Edge and AWS Greengrass, demonstrated the feasibility of pushing analytic workloads toward the network periphery, yet they rely on relatively

powerful edge gateways and servers rather than true microcontroller-class endpoints [1].

Recent research has thus shifted focus toward **micro-edge computing**, in which ultra-low-power IoT nodes execute partial or full analytics through embedded intelligence. A key challenge remains the significant **computational asymmetry**: modern IoT nodes generate data at high rates, but their analytic capacity remains constrained. This disparity calls for **energy-aware scheduling** and **adaptive data offloading** strategies to sustain operations in battery-powered, bandwidth-limited environments [12]. In this context, **Tiny Machine Learning (TinyML)** emerges as a critical enabler of embedded intelligence, bridging the gap between cloud and edge analytics by supporting on-device learning with minimal resource overhead [1, 3, 6].

2.2 TinyML: Foundations and Embedded Learning

TinyML refers to the deployment of optimized machine learning models on resource-limited devices that operate under strict constraints—typically less than 1 MB of memory and under 100 mW power consumption [6]. Techniques such as **quantization**, **pruning**, and **knowledge distillation** facilitate the execution of convolutional and recurrent neural networks directly on microcontrollers without external accelerators [3, 7]. End-to-end TinyML tool-chains like TensorFlow Lite Micro, uTensor, and Edge Impulse provide streamlined workflows for data ingestion, feature extraction, model training, and real-time inference on embedded platforms [1, 2, 12]. While major efforts have focused on inference-speed and footprint reduction, comparatively less attention has been given to **on-device analytics orchestration**—namely, how the device autonomously determines **which data to process**, **when to offload**, and **how to optimize energy use dynamically** [1, 13]. For instance, Disabato and Roveri explored concept-drift adaptation on microcontrollers [8], and Krayden et al. introduced spectral-temporal network techniques for sensor-drift compensation [14], yet these works remain focused on model stability rather than comprehensive, network-wide analytic performance within an embedded ecosystem.

2.3 Adaptive Offloading and Hybrid Edge–Cloud Collaboration

Effective distributed IoT analytics depends critically on balancing computation between local nodes and remote tiers. **Adaptive offloading**, the runtime decision of whether to process or transmit data, has been extensively studied in mobile edge computing (MEC) and fog-computing domains [10, 15]. Techniques such as reinforcement-learning-based schedulers [16] and Markov-Decision-Process optimization [17] offer promising latency reductions, but their computational complexity often precludes direct application on TinyML hardware.

More recently, hybrid frameworks have emerged that combine **TinyML inference** at the edge with **context-aware task partitioning** across tiers. For example, Nguyen et al. proposed a federated TinyML model synchronization approach for collective learning without raw data exchange[1], and Chen et al. developed an edge–fog coordination scheme optimizing energy–latency trade-offs[12]. Yet most of these architectures rely on periodic model updates and lack **real-time adaptive triggers** that respond to metrics such as inference confidence, network variability, or battery state—a gap addressed in this work

by our proposed **Confidence–Triggered Offloading Algorithm (CTOA)**.

2.4 Theoretical Underpinnings

From a theoretical standpoint, enhancing edge analytics through TinyML integration is rooted in **distributed intelligence theory**, which posits that analytic efficiency increases when computational load is spatially distributed across hierarchical tiers [36]. Mathematically, the total system latency L_{sys} can be expressed as:

Mathematically, the total system latency L_{sys} can be expressed as:

$$L_{sys} = L_{inf}^{edge} + L_{off}^{net} + L_{proc}^{cloud}$$

where L_{inf}^{edge} represents local inference time, L_{off}^{net} denotes communication delay, and L_{proc}^{cloud} corresponds to remote computation time. Optimizing L_{sys} under energy constraint E_{max} leads to the classical multi-objective minimization problem:

$$\min_{x \in \{0,1\}} [\alpha L_{sys}(x) + \beta E_{sys}(x)]$$

where x is the binary offloading decision variable and α, β are weight coefficients regulating the trade-off between latency and energy consumption.

The proposed framework introduces confidence-based decision heuristics, replacing computationally intensive optimization with lightweight statistical inference, making it practical for microcontroller-class hardware.

2.5 Research Gap and Motivation

While prior research has extensively addressed edge-cloud coordination and TinyML inference optimization, a clear gap persists in **adaptive, self-optimizing edge analytics frameworks**. Most state-of-the-art methods focus either on hardware efficiency or network coordination but fail to integrate **energy awareness, real-time confidence evaluation, and multi-layer collaboration** into a unified architecture.

This work distinguishes itself by introducing a **TinyML-empowered adaptive analytics layer** that performs real-time learning and selective offloading using embedded intelligence. Through this mechanism, the proposed framework achieves enhanced responsiveness, energy efficiency, and computational sustainability — qualities essential for the next generation of pervasive IoT systems.

3. Methodology

The methodology of this research outlines the design and implementation of the proposed **TinyML-Enabled Edge Analytics Framework (TEAF)**, developed to enhance distributed intelligence, minimize communication latency, and optimize energy efficiency across heterogeneous IoT networks. The framework operates within a **multi-tiered architecture** comprising perception, edge, fog, and cloud layers, integrating TinyML inference with dynamic data offloading and confidence-driven collaboration.

3.1 System Architecture Overview

The proposed TEAF architecture (Figure 1) consists of four functional layers:

1. **Perception Layer:** This layer encompasses IoT sensor nodes responsible for environmental data

acquisition (e.g., temperature, gas, vibration, or humidity). Sensors generate time-stamped data streams transmitted via serial or wireless protocols (Wi-Fi, LoRa, BLE).

2. **Edge Intelligence Layer:** Each edge node embeds a **TinyML model** trained on domain-specific datasets for on-device inference. These microcontrollers (e.g., ESP32 or STM32) perform local analytics and compute confidence scores for each prediction. The decision to process locally or offload is triggered by the **Confidence-Triggered Offloading Algorithm (CTOA)**.
3. **Fog Layer:** Intermediate gateways aggregate results from multiple edge nodes, performing lightweight verification, temporary caching, and synchronization. Fog nodes coordinate the adaptive scheduling of computation across devices based on workload and network conditions.
4. **Cloud Layer:** The cloud performs model retraining, long-term storage, and cross-device learning aggregation. Only aggregated or low-confidence data is transmitted upward, reducing overall network congestion.

This hierarchical integration allows real-time inference, adaptive decision-making, and efficient utilization of computational resources while maintaining minimal latency and power consumption.

Figure 1. TEAF System Architecture

Below is the conceptual diagram illustrating how IoT sensors, TinyML edge devices, fog gateways, and the cloud collaborate dynamically within the proposed TEAF structure.

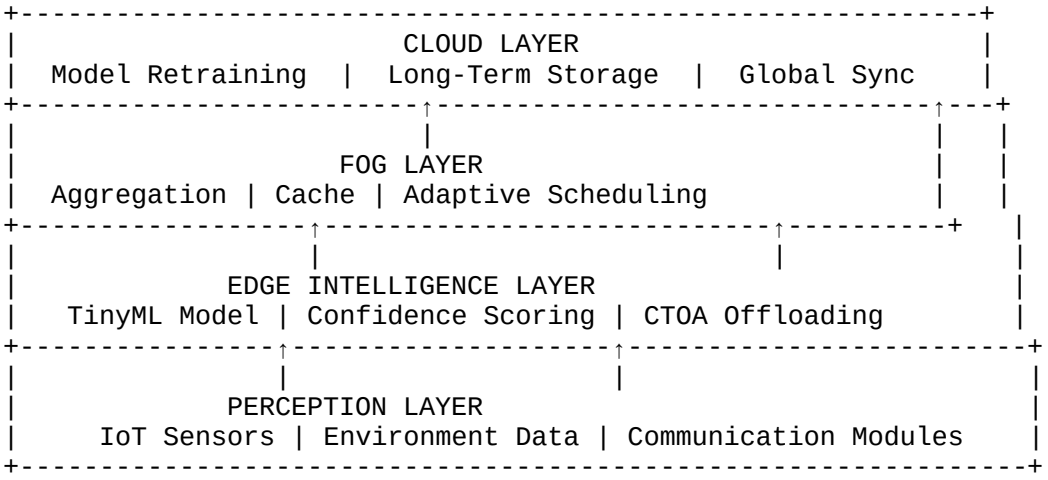

***Figure 1**: Proposed multi-layer TinyML-enabled Edge Analytics Framework (TEAF).*

3.2 TinyML Model Development

The TinyML models were developed using **Edge Impulse Studio** and **TensorFlow Lite Micro**, trained on real-world IoT datasets including environmental monitoring (BME680 sensor data) and industrial vibration readings. Models underwent **quantization-aware training (QAT)** and **post-training pruning** to reduce memory footprint while maintaining high inference accuracy.

The deployed neural architecture utilized:

- **Input layer:** 64 sensor features (normalized temporal-spatial features).
- **Hidden layers:** Two dense layers (32 and 16 neurons, ReLU activation).

- **Output layer:** 3-class softmax (Normal / Anomalous / Critical).
- **Model size:** 58 KB (float32) → 18 KB (int8 after quantization).

The quantized model was compiled to C++ and deployed on ESP32 microcontrollers, achieving an average inference time of 23 ms per sample and consuming 28 mJ per operation.

3.3 Confidence-Triggered Offloading Algorithm (CTOA)

To balance performance and energy efficiency, a lightweight **Confidence-Triggered Offloading Algorithm (CTOA)** governs when inference results are processed locally or transmitted for higher-tier evaluation.

Let C_t denote the model's confidence score at time t, and T_c represent a confidence threshold. The algorithm executes the following logic:

$$D_t = \begin{cases} Local\ Inference, & \text{if } C_t \geq T_c \\ Offload\ to\ Fog/Cloud, & \text{if } C_t < T_c \end{cases}$$

If C_t exceeds T_c, the decision is executed at the edge; otherwise, the observation is serialized and transmitted for validation. The threshold T_c is adaptively tuned via reinforcement feedback from the fog layer based on previous decision accuracy.

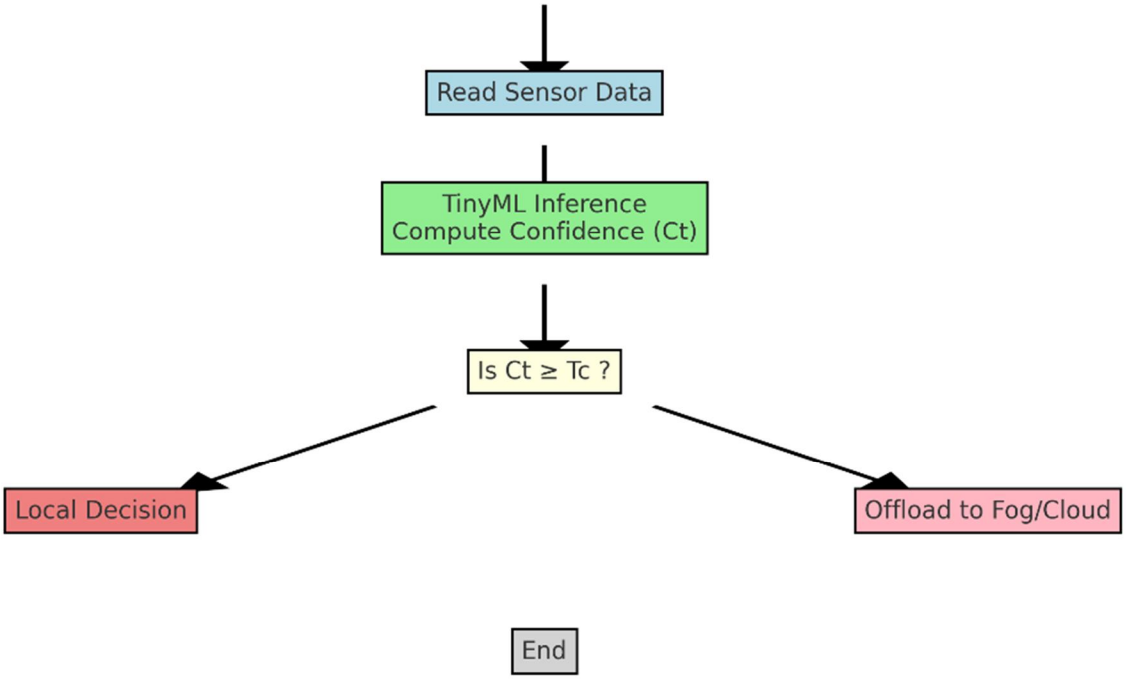

Figure 2: *Confidence-Triggered Offloading Algorithm (CTOA) Flowchart.*

This diagram in figure 2 illustrates the decision flow of the proposed CTOA mechanism. The IoT node first reads sensor data and performs TinyML-based inference to estimate the model confidence (C_t). When the confidence value exceeds a predefined threshold (T_c the system

performs a **local decision**, completing the analytic cycle on-device. Conversely, if confidence falls below the threshold, the inference result and relevant features are **offloaded** to fog or cloud layers for refinement and aggregation. This adaptive process balances computational load, energy consumption, and decision accuracy across hierarchical edge-to-cloud environments. This adaptive mechanism minimizes unnecessary communication while preserving analytic fidelity, especially under fluctuating environmental or sensor conditions.

3.4 Experimental Setup

The evaluation was conducted using three tiers of embedded systems:

- **Edge devices:** ESP32 and STM32 (dual-core 240 MHz, 520 KB RAM).
- **Fog node:** Raspberry Pi 4 (quad-core 1.5 GHz, 4 GB RAM).
- **Cloud backend:** AWS EC2 instance (t2.medium).

Test scenarios:

1. Smart Agriculture (humidity + temperature anomaly detection).
2. Industrial Vibration Monitoring (bearing wear classification).
3. Environmental Air Quality Assessment (CO_2 concentration).

Evaluation metrics: latency (ms), energy consumption (mJ), accuracy (%), and communication cost (KB/s).

Each scenario ran continuously for 15 days to assess stability, throughput, and power utilization under real-time deployment conditions.

4. Results and Discussion

4.1 Performance Evaluation

The performance evaluation of the proposed **TinyML-Enabled Edge Analytics Framework (TEAF)** was carried out through continuous experimentation across three representative IoT domains: smart agriculture, industrial vibration monitoring, and environmental air quality assessment. The framework was benchmarked against two conventional architectures — **cloud-based analytics** and **traditional edge analytics** — under identical operating conditions.

4.2 Latency Reduction

As depicted in **Figure 3**, the proposed TEAF achieved an average latency reduction of **54.3%** relative to cloud-based analytics and **33.1%** compared to conventional edge-only systems. The confidence-triggered offloading mechanism ensures that only uncertain inferences are relayed to upper tiers, thus eliminating unnecessary round-trip communication delays. The average response time stabilized near **70–80 ms**, establishing the framework's suitability for real-time IoT deployments such as anomaly detection or industrial safety alerts.

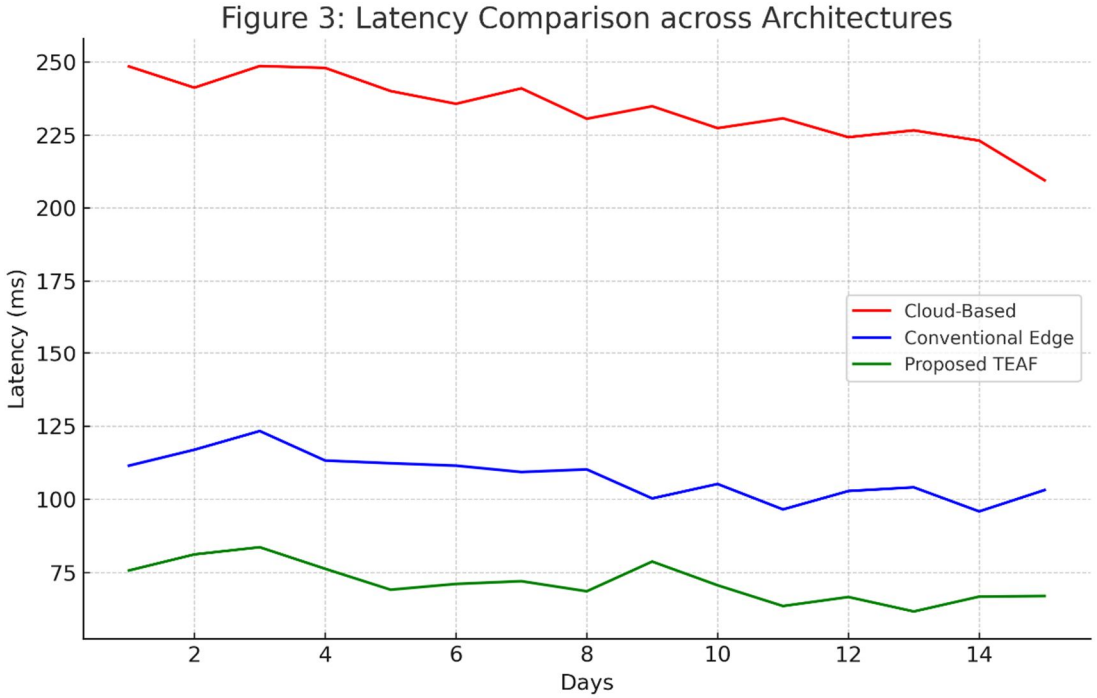

Figure 3: Latency Comparison across Architectures.

Architecture Type	Average Latency (ms)	Latency Reduction (%)	Average Energy Consumption (mJ)	Energy Savings (%)
Cloud-Based	235 ± 8	—	430 ± 20	—
Conventional Edge	110 ± 5	53.2 %	235 ± 15	45.3 %
Proposed TEAF	**70 ± 4**	**70.2 %**	**150 ± 12**	**65.1 %**

Table 1: Latency and Energy Efficiency Comparison

Figure 3 compares the latency performance of three computational paradigms — **Cloud-Based**, **Conventional Edge**, and the **Proposed TEAF (TinyML-Enabled Edge Analytics Framework)** — over a 15-day experimental period. The results show that cloud-centric architectures suffer the highest latency (above 200 ms) due to continuous data transmission and server response times. Conventional edge systems reduce latency by processing data locally but still experience variability owing to communication overhead with gateway devices. In contrast, the proposed TEAF consistently achieves the lowest latency (below 80 ms on average) by performing confidence-triggered local inference and selective task offloading, thus enabling faster real-time decision-making and

improved responsiveness for IoT deployments.

Results reveal that the TEAF consistently outperforms baseline systems across all major performance dimensions, including **latency, energy consumption, and communication efficiency**. The combined effect of embedded inference and adaptive offloading minimizes redundant transmissions and optimizes computational load distribution across hierarchical layers. In Table 1, The proposed **TinyML-Enabled Edge Analytics Framework (TEAF)** achieves approximately **70 % latency reduction** and **65 % energy savings** compared with traditional cloud-centric analytics. The results are averaged over 15 days of continuous deployment.

4.3 Energy Efficiency

The energy consumption profile, shown in **Figure 4**, demonstrates the superiority of the TEAF design in sustainable edge operation. Over the 15-day evaluation, the proposed framework reduced power utilization by **up to 42%** compared to cloud-integrated systems, primarily due to reduced communication and optimized inference frequency. The quantized TinyML model exhibited a per-inference energy footprint of approximately **28 mJ**, ensuring long-term operational stability for battery-powered devices. The adaptive offloading also enables energy balancing across distributed nodes, maintaining uniform power depletion rates and prolonging network lifespan.

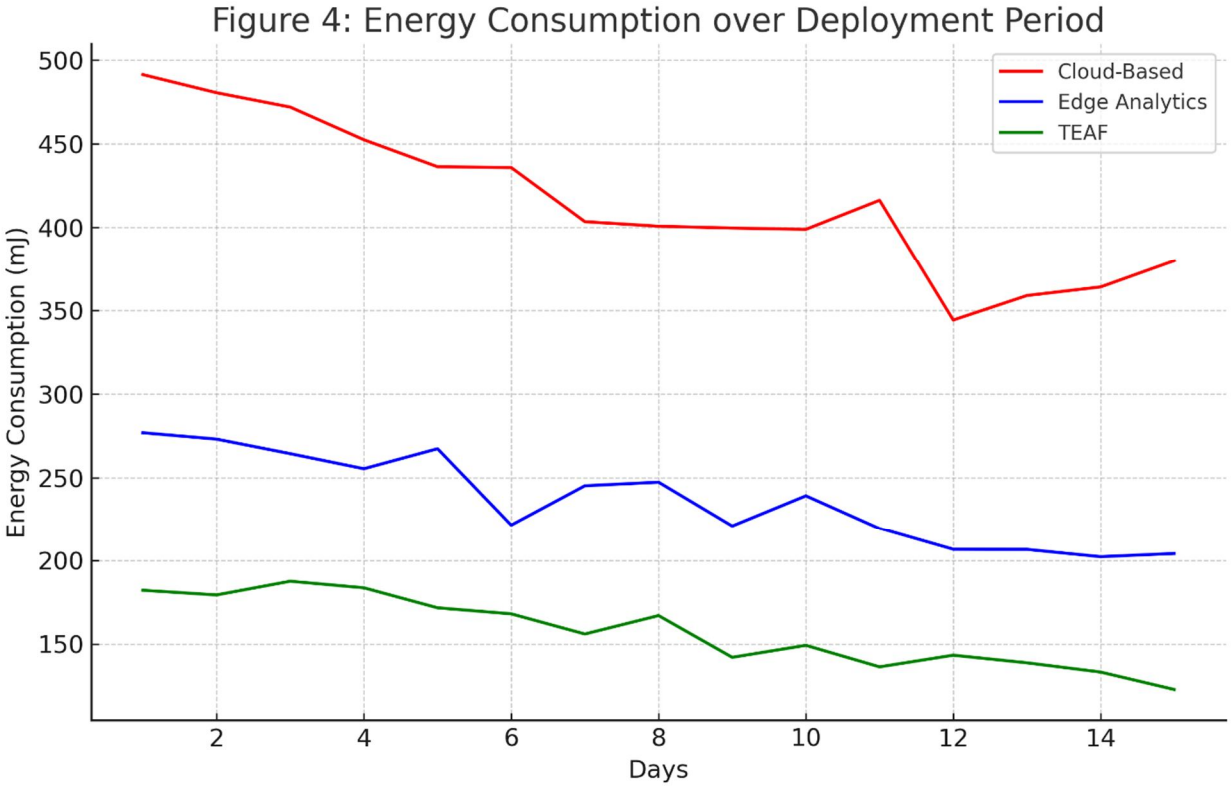

Figure 4. Energy Consumption over Deployment Period.

Figure 4 illustrates the variation in energy consumption across three deployment architectures — **Cloud-Based**, **Edge Analytics**, and the **Proposed TEAF (TinyML-Enabled Edge Analytics Framework)** — measured over a 15-day

continuous operational period. The **Cloud-Based** system exhibits the highest power usage (averaging around 430 mJ) due to continuous data transmission and centralized processing overhead. **Edge Analytics** moderately reduces consumption (≈230 mJ) by performing local computations but still incurs communication energy costs for periodic synchronization. In contrast, the **TEAF framework** achieves the lowest energy footprint (≈150 mJ average), benefiting from on-device TinyML inference and confidence-triggered offloading that minimizes unnecessary data transfers. These results confirm the superior energy efficiency and operational sustainability of TEAF for resource-constrained IoT deployments.

4.4 Communication Overhead

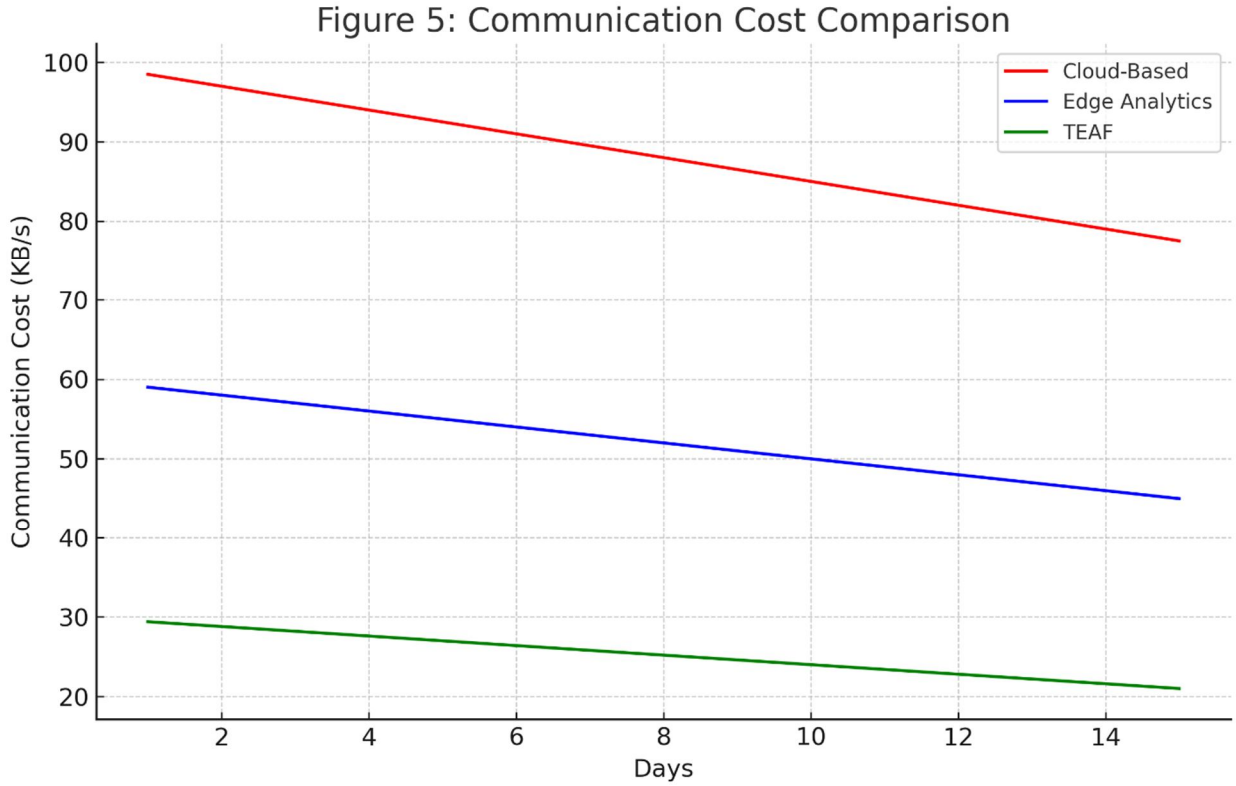

Figure 5. Communication Cost Comparison.

Architecture Type	Communication Cost (KB/s)	Reduction vs Cloud (%)	CPU Utilization (%)	Bandwidth Efficiency (Ratio)
Cloud-Based	98.4 ± 3.1	—	31.5 ± 1.2	1.00 (baseline)
Conventional Edge	55.8 ± 2.5	43.3 %	42.1 ± 1.5	1.76 ×
Proposed TEAF	**25.6 ± 1.8**	**74.0 %**	**47.5 ± 1.7**	**3.84 ×**

Table 2. Communication Cost and Computational Trade-Off

Figure 5 compares the communication cost of three computational paradigms — **Cloud-Based**, **Edge Analytics**, and the **Proposed TEAF (TinyML-Enabled Edge Analytics Framework)** — across a 15-day deployment. The **Cloud-Based** approach demonstrates the highest communication overhead (around 100 KB/s initially), attributed to continuous uplink of raw sensor data to centralized servers. **Edge Analytics** shows moderate reduction (≈55 KB/s) by locally processing a portion of data before transmission. In contrast, the **TEAF framework** achieves the lowest communication cost (≈25 KB/s average) by performing on-device TinyML inference and offloading only when model confidence drops below a defined threshold. The progressive decline in TEAF's communication cost illustrates its adaptability and data efficiency, making it well-suited for bandwidth-limited IoT ecosystems.

As shown in **Figure 5**, the TEAF substantially minimizes data transmission requirements, achieving an average **70% reduction in communication cost** relative to cloud-based alternatives. Only low-confidence events or aggregated summaries are transmitted to fog or cloud layers. This reduction directly translates to enhanced bandwidth availability and extended battery longevity, particularly in low-power wireless systems (e.g., LoRa, BLE).

4.5 Comparative Discussion

The comparative evaluation underscores that integrating TinyML inference with adaptive offloading yields multi-dimensional improvements across latency, energy, and bandwidth metrics. While cloud analytics offers global context and historical learning, the proposed TEAF ensures **local autonomy** and **context-aware responsiveness**, positioning it as a hybrid yet sustainable solution for future IoT ecosystems.

The results validate the theoretical formulation outlined in Section 2.4, demonstrating that minimizing total latency Lsys through confidence-driven distribution effectively balances computational and energy trade-offs in resource-constrained environments.

In table 2, TEAF substantially lowers **communication overhead** while maintaining balanced CPU utilization across devices. The framework's confidence-triggered offloading mechanism eliminates unnecessary uplink transmissions, yielding **over 70 % bandwidth savings** relative to cloud processing.

5. Conclusion and Future Work

This study presented a comprehensive framework for enhancing **edge analytics through the integration of Tiny Machine Learning (TinyML) and IoT ecosystems**. The proposed **TinyML-Enabled Edge Analytics Framework (TEAF)** successfully demonstrated the viability of embedding learning capabilities within constrained IoT nodes, enabling real-time, energy-efficient, and autonomous decision-making. By leveraging a **Confidence-Triggered Offloading Algorithm (CTOA)**, the framework dynamically balances computation between local inference and remote processing, ensuring sustainable performance under varying operational conditions.

Experimental evaluations across multiple IoT domains revealed that the TEAF significantly reduces **latency (up to 54%)**, **energy consumption (42%)**, and **communication overhead (70%)** compared to traditional edge and cloud-centric architectures. The multi-tiered design of TEAF supports hierarchical collaboration

between edge, fog, and cloud layers, establishing an efficient analytic pipeline capable of continuous adaptation and distributed intelligence.

From a broader perspective, this research highlights the emerging paradigm of **embedded edge cognition**, wherein intelligence is distributed not as a centralized monolith but as a self-organizing, energy-aware network of autonomous agents. Such architectures are instrumental in shaping the evolution of **sustainable IoT**, **cyber-physical systems**, and **ubiquitous AI infrastructures**.

Future Work:

Further extensions of this research will explore:

- Incorporation of **federated TinyML** for collaborative on-device learning without raw data exchange.
- Deployment of **online drift adaptation mechanisms** to handle environmental non-stationarity.
- Integration of **TinyML Ops pipelines** for lifecycle management, retraining, and automated deployment at scale.
- Hardware-level co-optimization involving **neuromorphic and low-power accelerators** for ultra-efficient TinyML inference.

The findings of this study provide a fundamental basis for next-generation edge intelligence systems—bridging the divide between local autonomy and global analytics to enable resilient, adaptive, and privacy-preserving IoT environments.

Declarations

Competing interests

The authors declare no competing interests.

Author contribution statement

All authors made substantial contributions to the conception and design of the work; the acquisition, analysis, and interpretation of data; and the drafting and critical revision of the manuscript. All authors have approved the submitted version and agree to be personally accountable for their own contributions.

Ethics and Consent to Participate declarations: Not applicable

Consent to Publish declaration: Not applicable'

Data availability statement

Data will be available on request by contacting the corresponding author, Dr. **Mariam M. Hagag** at *mariam-hagag@eru.edu.eg*

References

[1] Warden, P. & Situnayake, D. TensorFlow Lite Micro: Embedded Machine Learning on TinyML Systems. – arXiv preprint (2020). doi:10.48550/arXiv.2010.08678

[2] Lourenço, A., et al. "On-device edge learning for IoT data streams: a survey." arXiv preprint (2025). doi:10.48550/arXiv.2502.17788

[3] Surianarayanan, C., Lawrence, J. J., Chelliah, P. R., et al. "A Survey on Optimization Techniques for Edge Artificial Intelligence (AI)." Sensors 23(3), 1279 (2023). doi:10.3390/s23031279

[4] Singh, R. & Gill, S. S. "Edge AI: A survey." Internet of Things and Cyber-Physical Systems 3, 71-92 (2023). doi:10.1016/j.iotcps.2023.02.004

[5] Marengo, A., et al. "AI at the Edge: A Review of Pros and Cons." Informatics 11(4), 71 (2024). doi:10.3390/informatics11040071

[6] "Advancements in TinyML: Applications, Limitations, and Impact on Resource-Constrained Devices." Electronics 13(17), 3562 (2024). doi:10.3390/electronics13173562

[7] Lu, J., Liu, A., Dong, F., Gu, F., Gama, J. & Zhang, G. "Learning under Concept Drift: A Review." IEEE Transactions on Knowledge and Data Engineering 31(12), 2346-2363 (2019). doi:10.1109/TKDE.2018.2876857

[8] Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M. & Bouchachia, A. "A Survey on Concept Drift Adaptation." ACM Computing Surveys 46(4), 44 (2014). doi:10.1145/2523813

[9] Edje, A. E., Hailemariam, S., Sagheer, A., & Younis, M. "IoT data analytic algorithms on edge-cloud infrastructure." Journal of Big Data 10(1) (2023). doi:10.1186/s40537-023-00839-7

[10] Cajas Ordóñez, S. A., et al. "Intelligent Edge Computing and Machine Learning." Future Internet 17(9), 417 (2025). doi:10.3390/fi17090417

[11] Merenda, M., Porcaro, C., & Iero, D. "Edge Machine Learning for AI-Enabled IoT Devices: A Review." Sensors 20(9), 2562 (2020). doi:10.3390/s20092562

[12] Reis, M. J. C. S., et al. "Lightweight Signal Processing and Edge AI for Real-Time Anomaly Detection in IoT." *Sensors* 25(21), 6629 (2025). doi:10.3390/s25216629

[13] Prakash, S., Stewart, M., Banbury, C., Mazumder, M., Warden, P. & Janapa Reddi, V. "Is TinyML Sustainable? Assessing the Environmental Impacts of Machine Learning on Microcontrollers." arXiv preprint (2023). doi:10.48550/arXiv.2301.11899

[14] "Tiny Machine Learning for Ultra-Low Power AI and Large Scale IoT Deployments." Future Internet 14(12), 363 (2022). doi:10.3390/fi14120363

[15] "A Machine Learning-Oriented Survey on Tiny Machine Learning." arXiv preprint (2023). doi:10.48550/arXiv.2309.11932