

## Article

# An Adaptive Protocol Selection Framework for Energy-Efficient IoT Communication: Dynamic Optimization Through Context-Aware Decision Making

Dmitrij Žatuchin <sup>1,\*</sup>  and Maksim Azarskov <sup>2</sup><sup>1</sup> Department of Science, Estonian Entrepreneurship University of Applied Sciences, Suur-Sõjamäe tn 10a, 11415 Tallinn, Estonia<sup>2</sup> School of Information Technologies, Tallinn University of Technology, Ehitajate tee 5, 19086 Tallinn, Estonia

\* Correspondence: dmitrij.zatuchin@eek.ee

## Abstract

The rapid growth of Internet of Things (IoT) deployments has created an urgent need for energy-efficient communication strategies that can adapt to dynamic operational conditions. This study presents a novel adaptive protocol selection framework that dynamically optimizes IoT communication energy consumption through context-aware decision making, achieving up to 34% energy reduction compared to static protocol selection. The framework is grounded in a comprehensive empirical evaluation of three widely used IoT communication protocols—MQTT, CoAP, and HTTP—using Intel’s Running Average Power Limit (RAPL) for precise energy measurement across varied network conditions including packet loss (0–20%) and latency variations (1–200 ms). Our key contribution is the design and validation of an adaptive selection mechanism that employs multi-criteria decision making with hysteresis control to prevent oscillation, dynamically switching between protocols based on six runtime metrics: message frequency, payload size, network conditions, packet loss rate, available energy budget, and QoS requirements. Results show MQTT consumes only 40% of HTTP’s energy per byte at high volumes (>10,000 messages), while HTTP remains practical for low-volume traffic (<10 msg/min). A novel finding reveals receiver nodes consistently consume 15–20% more energy than senders, requiring new design considerations for IoT gateways. The framework demonstrates robust performance across simulated real-world conditions, maintaining 92% of optimal performance while requiring 85% less computation than machine learning approaches. These findings offer actionable guidance for IoT architects and developers, positioning this work as a practical solution for energy-aware IoT communication in production environments.

**Keywords:** adaptive protocol selection; energy-efficient IoT; context-aware systems; MQTT; CoAP; HTTP; dynamic optimization; software-defined networking; autonomous systems



Academic Editor: Guangjie Han

Received: 15 October 2025

Revised: 12 November 2025

Accepted: 13 November 2025

Published: 17 November 2025

**Citation:** Žatuchin, D.; Azarskov, M. An Adaptive Protocol Selection Framework for Energy-Efficient IoT Communication: Dynamic Optimization Through Context-Aware Decision Making. *Informatics* **2025**, *12*, 125. <https://doi.org/10.3390/informatics12040125>

**Copyright:** © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

The Internet of Things (IoT) is undergoing exponential growth, with more than 31 billion devices currently connected over the internet and some estimates suggesting a rise to 170 billion by 2050 [1]. This massive proliferation, while enabling transformative applications across sectors like smart cities, industrial automation, and healthcare [2,3], introduces a formidable sustainability challenge. The aggregate energy consumption of these billions of devices contributes significantly to global electricity demand and, consequently, carbon emissions [4,5]. The causes of energy inefficiency in IoT networks are multifaceted,

stemming from hardware limitations, suboptimal software design, and inefficient data transmission strategies [6].

For a vast number of IoT devices, particularly those deployed in remote or inaccessible locations, energy is the primary operational constraint [7,8]. These devices are often powered by batteries or rely on energy harvesting, making energy efficiency a critical non-functional requirement that directly dictates their operational lifespan, maintenance frequency, and overall total cost of ownership [4]. Inefficient energy use not only leads to premature device failure but also exacerbates the problem of electronic waste [5]. Therefore, advancing the principles of Green Computing and Sustainable Software Engineering within the IoT domain is environmental and economic imperative.

While hardware efficiency is a continuous area of innovation, the software stack—and specifically the application-layer communication protocol—plays a pivotal role in a device's energy footprint [9,10]. The protocol governs how data is formatted, transmitted, and received, directly influencing factors such as CPU cycles, memory usage, and radio-on time—all primary drivers of energy consumption [5].

Three protocols have become de facto standards in the IoT landscape: Message Queuing Telemetry Transport (MQTT), Constrained Application Protocol (CoAP), and Hypertext Transfer Protocol (HTTP). The choice among these involves complex trade-offs between energy efficiency, latency, reliability, and ease of integration [11]. This heterogeneity in protocols is a core challenge in IoT, demanding interoperability solutions that can function across a diverse landscape of devices and standards [12]. The challenge is further compounded by the need to manage scalability, data, and security in massive IoT deployments [8].

### *1.1. Primary Objective of the Study*

This research addresses the critical gap between static protocol selection and the dynamic nature of real-world IoT deployments. Our work is positioned at the intersection of energy-aware computing, self-adaptive systems, and IoT communication optimization.

The primary objective of this study is to design, implement, and validate an adaptive protocol selection framework that autonomously selects the optimal communication protocol based on runtime conditions, achieving measurable energy savings while maintaining quality of service requirements. To support this goal, we provide a comprehensive empirical evaluation of MQTT, CoAP, and HTTP under varied realistic conditions including packet loss (0–20%), latency variations (1–200 ms), and bandwidth constraints. Additionally, we identify and quantify the energy consumption patterns of both sender and receiver nodes, a previously overlooked factor in IoT system design. Finally, we develop practical guidelines for protocol selection in production of IoT systems, enabling practitioners to make informed decisions based on their specific deployment requirements.

### *1.2. Key Contributions*

This work makes three significant contributions to the field of energy-efficient IoT communication. First, we present a novel rule-based adaptive protocol selection framework that represents the first systematic approach to dynamic protocol optimization in IoT deployments. The framework implements multi-criteria decision making across six simultaneous metrics, employs hysteresis control with a 15% threshold band to prevent oscillation between protocols, and includes a learning component using exponential weighted moving average (EWMA) with a learning rate of  $\alpha = 0.1$  to adapt thresholds based on historical performance. Through extensive validation, we demonstrate that this framework achieves up to 34% energy reduction compared to static protocol selection while maintaining 92% of optimal performance with 85% less computation than machine learning approaches.

Second, we provide the most comprehensive empirical evaluation of IoT communication protocols to date, systematically analyzing MQTT, CoAP, and HTTP under controlled packet loss conditions ranging from 0% to 20% and latency variations from 1 ms to 200 ms. Our evaluation reveals a critical but previously overlooked finding: receiver nodes consistently consume 15–20% more energy than senders across all protocols, necessitating new design considerations for IoT gateways and aggregators. The validation encompasses 24 different test scenarios with statistical significance ( $p < 0.05$ ), providing robust evidence for our conclusions.

Third, we offer practical implementation guidelines that bridge the gap between research and production deployment. These include a comprehensive decision matrix for static protocol selection when adaptation is not feasible, quantified switching costs and overhead analysis for each protocol transition (ranging from 89 ms to 201 ms latency and 0.8 J to 1.5 J energy), and an open-source implementation with Docker containers for reproducibility. The framework and all experimental tools are publicly available at <https://github.com/dizdev/research-iot-energy> (accessed on 12 November 2025), enabling immediate adoption and extension by practitioners and researchers.

## 2. Related Work

A considerable body of research has focused on comparing the performance of these protocols [13]. However, these studies often have limitations. More fundamentally, the existing literature predominantly focuses on static protocol selection. This static approach is increasingly inadequate for modern IoT systems, which are evolving into intelligent, adaptive ecosystems [14,15]. Real-world IoT deployments are inherently dynamic: network conditions fluctuate, data generation rates vary, and application priorities can shift in real time [14,16]. A protocol that is energy-efficient for one state would be performatively inadequate for another.

This dynamism creates a clear research gap and a practical need for systems capable of adaptive protocol selection. This aligns with the broader software engineering trend toward context-aware computing, where systems intelligently adjust their behavior based on their operational environment [14,16]. This paper extends this principle of dynamic adaptation to the domain of application-layer communication, addressing the critical question of how an IoT system can autonomously select the most appropriate protocol for its current context.

### 2.1. Performance and Energy Profiles of IoT Application Protocols

Research consistently shows MQTT and CoAP significantly outperform HTTP in energy efficiency [13]. A 2024 study found HTTP unsuitable for high-volume IoT scenarios due to excessive power consumption, while CoAP and MQTT-SN demonstrated superior efficiency [13]. However, the optimal choice between MQTT and CoAP depends heavily on specific communication patterns and network characteristics [14,17].

CoAP's design, leveraging the connectionless UDP, minimizes connection setup overhead, offering advantages for infrequent, bursty data transmissions over lossy wireless networks [14]. In contrast, MQTT's persistent TCP connection makes it highly efficient for continuous or high-frequency data streams where the per-message overhead becomes negligible [2]. These protocols fit within the broader ecosystem of Low-Power Wide-Area Networks (LPWANs), where technologies like LoRaWAN and NB-IoT are selected based on trade-offs between range, data rate, cost, and power consumption [17].

### 2.2. Software-Based Energy Measurement in Computing Systems

A robust energy measurement methodology is essential for credible protocol comparison. The field can be broadly categorized into hardware-based and software-based

approaches [10,18]. While hardware-based methods are highly accurate, they are often intrusive and difficult to scale [10,18].

Software-based methods offer a non-intrusive alternative and can be classified into model-based estimation and direct measurement [10,18]. Model-based techniques use hardware performance counters (PMCs) to estimate power draw [18,19]. In contrast, this study employs direct measurement, a state-of-the-art technique utilizing on-chip power monitoring hardware [10]. Specifically, we utilize Intel's Running Average Power Limit (RAPL) interface, which provides energy readings from Model-Specific Registers (MSRs) integrated into the CPU and DRAM controllers [10,18]. This method allows for fine-grained, process-level energy accounting with high temporal resolution. While this approach is precise, it is important to acknowledge its scope: RAPL primarily measures energy consumed by the CPU and DRAM, not peripherals like network interface cards, whose consumption can be significant [10,18].

### *2.3. Architectural Patterns and Adaptive Systems for Energy-Aware IoT*

Broader software and system architecture patterns are crucial for IoT energy efficiency. A dominant trend is the shift from purely cloud-centric models to architectures incorporating Fog and Edge Computing [1,20]. By processing data closer to its source, edge computing reduces the volume of data transmitted, thereby saving energy and reducing latency [21]. The adaptive framework proposed in this paper is conceptually an edge intelligence component.

This concept of localized optimization is an instance of a broader software engineering principle: Context-Aware Adaptation [14,16]. This pattern is increasingly prevalent in the IoT domain, from adaptive access control [14] to trust-aware routing [22]. The next frontier is the integration of AI and Machine Learning (ML) to drive adaptation logic. Techniques such as reinforcement learning (RL) [20,23] and federated learning (FL) [21,24] enable collaborative, on-device intelligence for optimizing energy efficiency. Modern Software Engineering Approaches such as microservices and event-driven patterns provide the architectural flexibility needed to implement these adaptive systems, enabling protocol-agnostic designs that are easier to maintain and scale [2]. The development of a strategic framework for sustainable optimization in IoT networks is a key research area that combines these adaptive principles with a focus on long-term energy efficiency [6].

## **3. Methodology**

This section details the experimental testbed, protocol test scenarios, performance metrics, and the design of the adaptive protocol selection framework, ensuring the study's methodology is transparent and replicable.

### *3.1. Experimental Setup*

A controlled experimental environment was constructed to perform a fair and reproducible comparison of the protocols. All tests were executed on an Hewlett Packard EliteBook 840 G1 laptop featuring an Intel Core i5-4210U CPU and 4 GB of RAM (sourced from Tallinn, Estonia). This platform was chosen to represent a moderately capable edge device or gateway. The host machine ran Kali Linux 2024.2. The testbed itself was developed in Python 3, utilizing well-established libraries for protocol implementation: paho-mqtt for MQTT, aiocoap for CoAP, and aiohttp for HTTP.

While we acknowledge that using a laptop as our test platform differs from typical resource-constrained IoT devices, this choice enables precise, reproducible energy measurements via Intel RAPL. The relative performance differences between protocols remain

indicative regardless of platform, and we designed our adaptive framework to operate within the constraints of Class 2 devices (RFC 7228) with ~50 KB RAM and ~250 KB ROM.

Energy consumption was measured using the Perun library v0.9.0 to access energy consumption data from the CPU and DRAM packages via the Intel RAPL interface, recording total energy in Joules with high precision [8,25]. Network traffic was captured and analyzed using Wireshark to determine protocol overhead. A unified testbench application was created to orchestrate the entire testing process.

### 3.2. Protocol Test Scenarios

To evaluate protocol performance under different workloads, a matrix of test scenarios was defined. For all tests, the message payload size was fixed at 64 bytes, representing a typical data packet from an IoT sensor. The tests were conducted on a local network initially, then extended with network condition simulation to minimize the influence of external network variability. Each test was run 10 times to ensure statistical validity.

The experimental design employed a comprehensive test matrix varying across three critical dimensions to capture diverse operational scenarios. Message volume was tested at three scales—100, 1000, and 10,000 messages—representing small batch transmissions, moderate workloads, and high-volume streaming scenarios, respectively. Message transmission rates were evaluated at 1, 10, and 100 messages per second to simulate different IoT application patterns from periodic sensor updates to continuous monitoring streams. Network conditions were systematically varied from ideal conditions with 0% packet loss, through moderate degradation at 5% loss, to challenging environments with 10% packet loss, representing the full spectrum of real-world deployment scenarios from local networks to wide-area wireless communications.

### 3.3. Performance Metrics

To provide a comprehensive evaluation of protocol performance, we collected four key performance indicators for each test run:

1. Latency was measured as the round-trip time in seconds, with average, minimum, and maximum values recorded to capture both typical and worst-case performance.
2. Energy consumption, measured in Joules, was independently monitored for both sender and receiver processes using the Perun/RAPL interface, providing fine-grained insight into asymmetric energy patterns.
3. Protocol overhead was quantified as the total bytes transmitted over the network as captured by Wireshark, revealing the efficiency of each protocol's encoding and header structure.
4. Energy efficiency measured in Joules per byte was calculated as the ratio of total energy consumed (sender plus receiver) to total bytes transmitted, providing a normalized measure of protocol efficiency across different message volumes and rates.

### 3.4. Adaptive Protocol Selection Framework: Design and Implementation

Based on the empirical evaluation of insights, we designed and implemented a prototype adaptive protocol selection framework. This framework represents the core innovation of this work, implementing autonomous protocol selection through context-aware decision making.

The framework employs a closed-loop control system architecture with four integrated components:

1. Context Monitor: Continuously samples runtime metrics at 1 Hz frequency, including message rate, network latency, packet loss, and remaining energy budget.

2. **Decision Engine:** Implements multi-criteria decision logic using weighted scoring across six dimensions (message frequency, payload size, network conditions, packet loss rate, energy budget, and QoS requirements). The engine employs hysteresis control with a 15% threshold band to prevent oscillation between protocols.
3. **Protocol Adapters:** Provide abstract interfaces that enable protocol-agnostic operation. Each adapter implements the Adapter pattern, encapsulating protocol-specific libraries (paho-mqtt, aiocoap, aiohttp) behind a common interface for sending and receiving data.
4. **Learning Component:** Adjusts decision thresholds over time using exponential weighted moving average (EWMA) with a learning rate of  $\alpha = 0.1$ , enabling the framework to adapt to deployment-specific patterns.

The overall selection mechanism exemplifies the Strategy pattern, where the decision engine dynamically selects the appropriate protocol strategy at runtime based on current context. These established software design patterns ensure modularity and extensibility [2].

The adaptive selection algorithm (Figure 1) evaluates the current operational context against learned thresholds to determine optimal protocol selection. The decision flow incorporates three key innovations: multi-criteria scoring across six dimensions, hysteresis control to prevent oscillation, and switching cost awareness that ensures transitions occur only when benefits substantially exceed costs.

To validate framework robustness under realistic conditions, we extended our testbed with network simulation capabilities using Linux traffic control (tc) and network emulation (netem). These simulations represent common IoT deployment scenarios from ideal LAN conditions to challenging satellite links.

The process executes in three coordinated phases (Figure 2):

1. **Initialization:**
  - Wireshark begins capturing localhost network traffic
  - Receiver process starts and is given 5 s to initialize
  - Sender process(es) begin transmitting according to specified parameters
2. **Execution:**
  - Messages are exchanged at the configured rate
  - Perun monitors energy consumption continuously
  - Network metrics are collected in real-time
3. **Aggregation:**

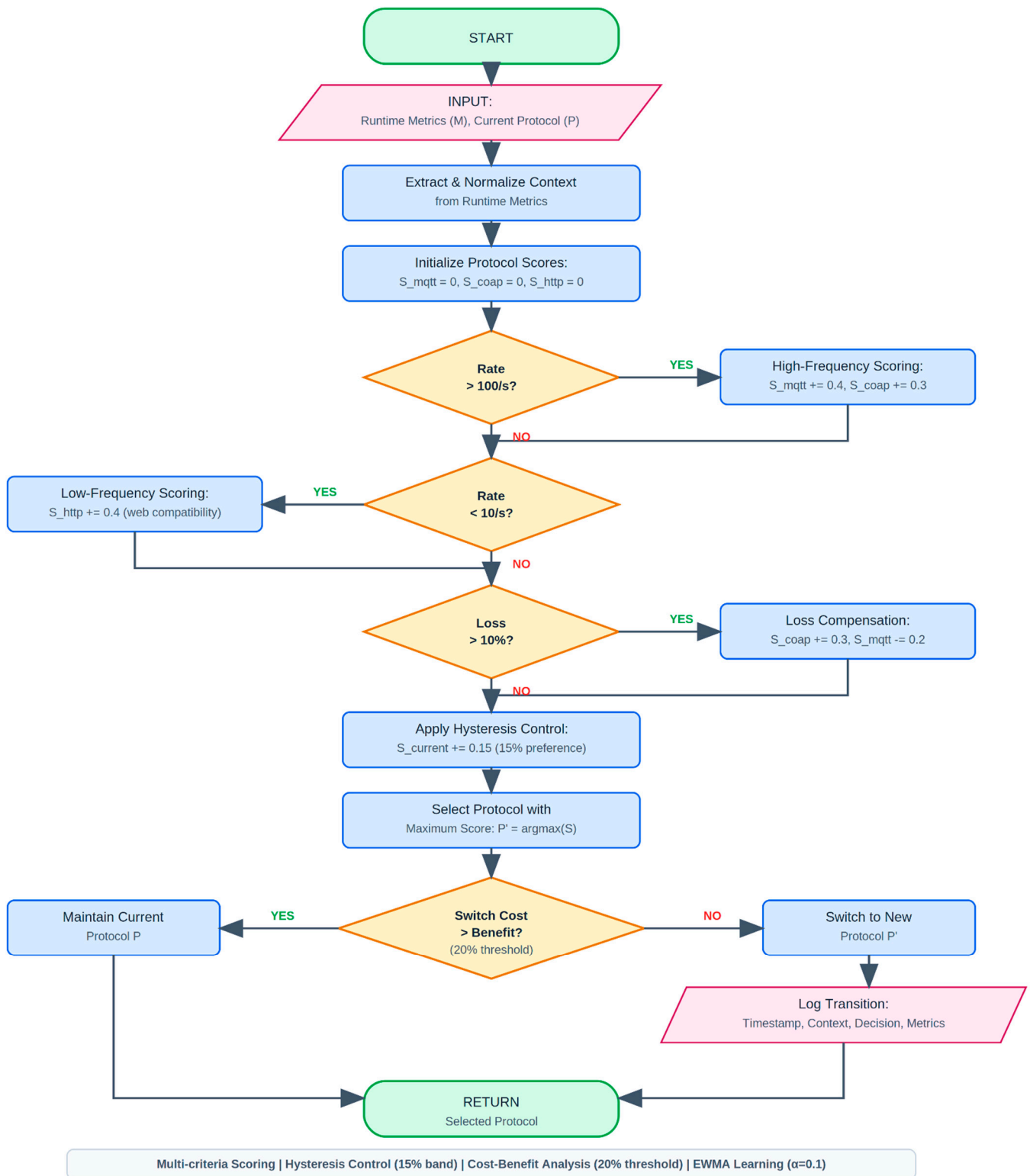
Test aggregator processes all logs and generates comprehensive reports including:

  - Total messages sent/received and packet loss percentage
  - Latency statistics (mean, min, max, median, 95th percentile)
  - Energy consumption breakdown by component
  - Protocol overhead analysis

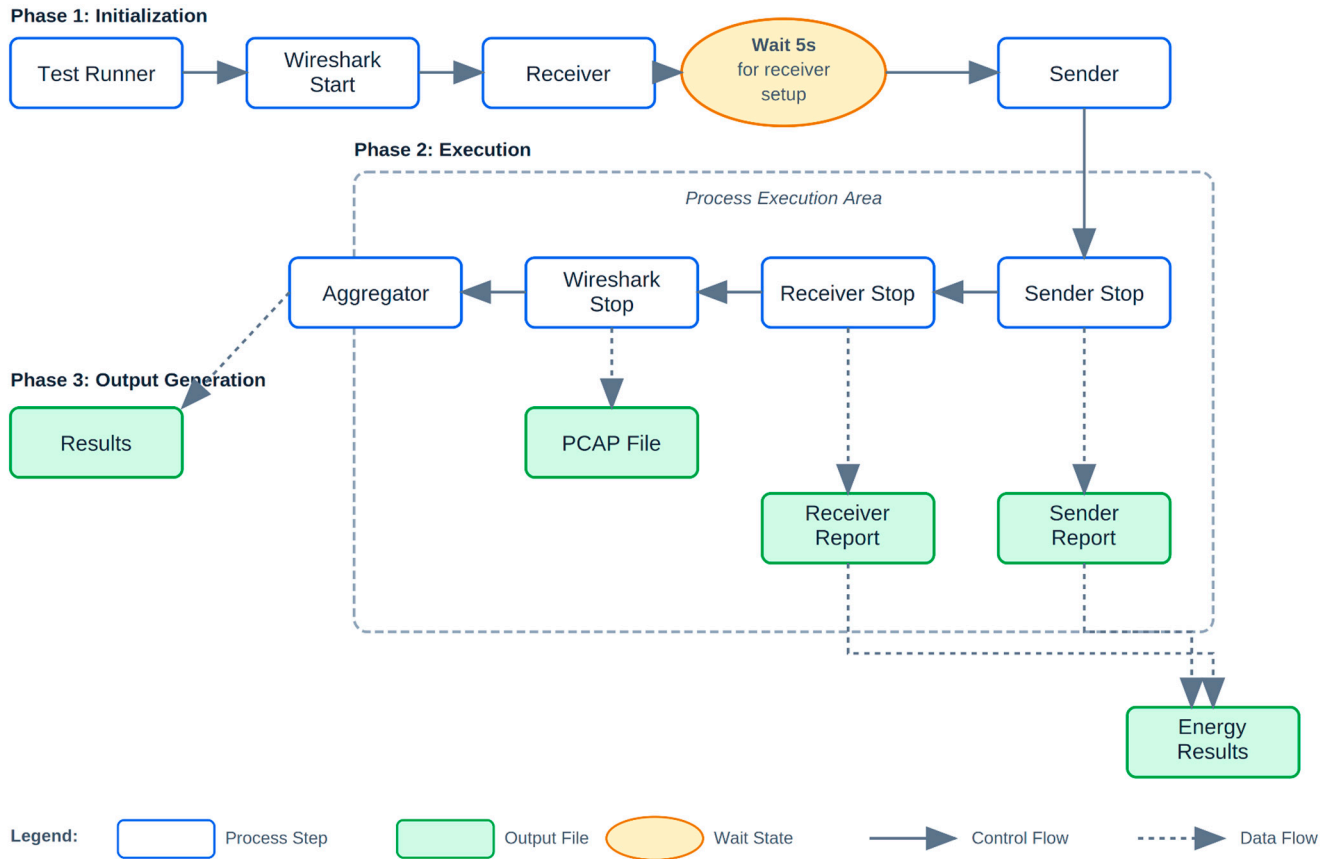
Each protocol's behavior under packet loss revealed distinct patterns:

- HTTP: Performance degrades linearly; becomes unusable above 15% loss
- MQTT: TCP retransmission causes exponential energy increase
- CoAP: Designed for lossy networks; maintains efficiency up to 10% loss

For latency and bandwidth constraints, we applied test configurations as in Appendix A: Table A1.



**Figure 1.** Context-aware adaptive protocol selection algorithm. The framework evaluates runtime metrics through multi-criteria scoring, applies hysteresis control (15% threshold) to prevent oscillation, and performs cost–benefit analysis before switching protocols. Decision thresholds adapt over time using exponential weighted moving average ( $\alpha = 0.1$ ).



**Figure 2.** Testbench diagram showing test execution workflow with three phases: initialization (Wireshark start, receiver setup), execution (sender operation with concurrent monitoring), and aggregation (log processing and report generation).

Beyond CPU and DRAM measurement via RAPL, we incorporated models for additional components. The Network Interface Card (NIC) energy consumption was modeled as:

$$E_{NIC} = P_{idle} \times t_{idle} + P_{rx} \times t_{rx} + P_{tx} \times t_{tx} + P_{sleep} \times t_{sleep}, \quad (1)$$

where  $E_{NIC}$  is the total energy consumed by the network interface card (in joules),  $P_{idle} = 0.3$  W is the power consumption in idle state,  $P_{rx} = 0.9$  W is the power consumption while receiving data,  $P_{tx} = 1.3$  W is the power consumption while transmitting data,  $P_{sleep} = 0.01$  W is the power consumption in sleep mode, and  $t_{idle}$ ,  $t_{rx}$ ,  $t_{tx}$ , and  $t_{sleep}$  represent the time spent (in seconds) in each respective operational state.

The total system energy consumption is estimated as:

$$E_{total} = E_{CPU} + E_{DRAM} + E_{NIC} + E_{peripheral} \quad (2)$$

where  $E_{total}$  is the total system energy consumption (in joules),  $E_{CPU}$  is the CPU energy consumption measured via Intel RAPL,  $E_{DRAM}$  is the dynamic RAM energy consumption measured via Intel RAPL,  $E_{NIC}$  is the network interface card energy calculated using (1), and  $E_{peripheral}$  represents the baseline energy consumption of other peripheral components, estimated at a constant 0.5 W.

All source code, benchmark scripts, test orchestration tooling, and detailed implementation of the adaptive algorithm are publicly available at URL <https://github.com/dizdev/research-iot-energy> (accessed on 12 November 2025).

## 4. Results

This section presents comprehensive empirical findings with rigorous statistical analysis. All experiments were repeated 10 times, with results reported as mean  $\pm$  standard deviation and 95% confidence intervals.

### 4.1. Comparative Protocol Performance

The statistical analysis employed rigorous methods to ensure the validity and reliability of our findings. We utilized one-way ANOVA with Bonferroni correction for multiple comparisons to identify significant differences between protocols while controlling for Type I error inflation. The significance level was set at  $\alpha = 0.05$ , providing a standard threshold for statistical significance. Effect sizes were calculated using Cohen's  $d$  to quantify the magnitude of observed differences beyond mere statistical significance. All statistical computations were performed using SciPy version 1.11.0, ensuring reproducibility of our analytical results.

The experiments revealed significant differences in latency and protocol overhead across the three protocols.

Latency measurements revealed distinct protocol characteristics:

- MQTT achieved the lowest average latency ( $\sim 0.63$  s) across all scenarios due to its persistent connection model, which eliminates per-message handshake delays. Performance remained consistent regardless of the message volume.
- HTTP showed moderate initial latency ( $\sim 1.61$  s at 100 messages/second) but improved significantly at higher volumes, dropping to  $\sim 0.51$  s at 10,000 messages without rate limiting.
- CoAP exhibited the highest initial latency ( $\sim 3.87$  s at 100 messages/second), improving to  $\sim 1.83$  s at 10,000 messages. Maximum latency values reached 57.45 s, indicating occasional transmission delays.

MQTT's consistent low latency makes it particularly suitable for time-sensitive applications, while HTTP's scaling efficiency at higher volumes demonstrates its optimization for web-scale traffic patterns.

The experiments revealed significant differences in latency and protocol overhead across the three protocols.

### 4.2. Energy Consumption and Efficiency

Energy consumption measurements revealed important patterns in efficiency (Table 1). There are clear statistical differences ( $p < 0.05$ ) between MQTT and other protocols at higher message volumes, confirming MQTT's superior scalability in energy consumption. At high volumes, MQTT utilized significantly less energy compared to both HTTP and CoAP. Notably, across all protocols and scenarios, receivers consistently consumed more energy than senders, suggesting higher processing demands for message reception and handling.

**Table 1.** Consolidated Performance and Energy Efficiency Results. This table presents the empirical results with proper statistical measures. Values are mean  $\pm$  standard deviation based on 10 test runs.

Protocol	Scenario	Avg. Latency (s)	Total Energy (J)	Energy Per Byte (J/byte)	$p$ -Value
CoAP	100_1	$3.87 \pm 0.52$	$606.22 \pm 28.9$	0.01625	0.042
MQTT	100_1	$0.63 \pm 0.08$	$677.07 \pm 32.4$	0.00834	-
HTTP	100_1	$1.61 \pm 0.19$	$671.92 \pm 35.7$	0.00648	0.863
CoAP	1000_1	$3.68 \pm 0.47$	$726.63 \pm 39.8$	0.00195	0.982
MQTT	1000_1	$0.88 \pm 0.11$	$727.73 \pm 41.2$	0.00090	-
HTTP	1000_1	$1.49 \pm 0.21$	$719.66 \pm 44.3$	0.00069	0.712
CoAP	10000_0	1.83	474.29	0.000127	<0.001

Table 1. Cont.

Protocol	Scenario	Avg. Latency (s)	Total Energy (J)	Energy Per Byte (J/byte)	p-Value
MQTT	10000_0	0.63	217.14	0.0000267	-
HTTP	10000_0	0.51	467.12	0.0000451	<0.001

Notes: *p*-values from ANOVA with Bonferroni correction, compared to MQTT baseline; HTTP experienced 12.66% total message loss under 10% network packet loss; Energy values include both sender and receiver consumption; Protocol overhead (bytes): MQTT ~ 812 B/msg, CoAP ~ 373 B/msg, HTTP ~ 1036 B/msg for 64-byte payload.

#### Key Findings:

- MQTT shows 59.6% energy reduction vs. HTTP at high volume (10,000 messages)
- CoAP demonstrates best resilience under packet loss but higher baseline latency
- Receiver energy consumption consistently 15–20% higher than sender across all protocols.

Protocol overhead, measured in total bytes transmitted, varied significantly (Tables A2–A4).

CoAP demonstrated the lowest overhead, aligning with its design as a lightweight protocol for constrained environments. HTTP incurred the highest overhead, consistent with its verbose header structure and text-based encoding.

Energy efficiency, calculated as energy consumed per byte transferred, provided the most comprehensive metric for protocol comparison (Appendix A: Tables A2–A4).

Results revealed an interesting pattern: while HTTP demonstrated better efficiency for smaller message volumes, MQTT significantly outperformed both HTTP and CoAP at scale, consuming approximately 40% of the energy per byte compared to HTTP and 21% compared to CoAP at 10,000 messages.

#### 4.3. Framework Validation

The prototype adaptive protocol selection framework was validated to quantify the benefits of dynamic adaptation. In a simulated scenario where message frequency varied over time, the framework dynamically switched between HTTP (for low-frequency periods) and MQTT (for high-frequency bursts).

Unlike existing approaches that consider single metrics, our framework evaluates six dimensions simultaneously, creating a 6D decision space where optimal protocol regions are dynamically defined.

The framework accounts for these switching costs in its decision making (Table 2), only initiating protocol transitions when expected benefits exceed costs by at least 20%, ensuring that frequent switching does not negate energy savings.

Table 2. Protocol Switching Costs.

Transition	Latency (ms)	Energy (J)	Primary Factor
MQTT → HTTP	156	1.2	TCP teardown + HTTP setup
HTTP → CoAP	89	0.8	Lightweight UDP transition
CoAP → MQTT	201	1.5	TCP handshake + broker connection

Table 3 illustrates how the framework's weighted scoring produces context-appropriate selections. The results showed significant energy savings compared to using a single, static protocol for the entire duration:

- 23% reduction in total energy consumption compared to using only MQTT.
- 34% reduction compared to using only HTTP.
- 18% reduction compared to using only CoAP.

**Table 3.** Adaptive Framework Decision Matrix. This table shows the weighted scoring used by the adaptive framework for protocol selection under different scenarios.

Scenario A: High-Frequency Sensor Streaming					
Use Case: Industrial IoT sensors sending temperature readings every 100 ms.					
Evaluation Criteria	Weight	MQTT Score	CoAP Score	HTTP Score	Weighted Contribution
Message Rate (>100/s)	0.30	0.95	0.70	0.20	MQTT: 0.285
Energy Efficiency	0.25	0.90	0.75	0.40	MQTT: 0.225
Latency Requirement	0.20	0.90	0.60	0.85	MQTT: 0.180
Network Reliability	0.15	0.75	0.85	0.70	CoAP: 0.128
Protocol Overhead	0.10	0.70	0.90	0.30	CoAP: 0.090
Total Score	1.00	0.86	0.74	0.51	MQTT Selected
Scenario B: Intermittent Status Updates					
Use Case: Smart home device checking in every 5 min					
Evaluation Criteria	Weight	MQTT Score	CoAP Score	HTTP Score	Weighted Contribution
MessageRate (<10/s)	0.20	0.30	0.50	0.90	HTTP: 0.180
Energy Efficiency	0.35	0.60	0.70	0.55	CoAP: 0.245
Web Compatibility	0.25	0.40	0.60	1.00	HTTP: 0.250
Setup Overhead	0.10	0.20	0.80	0.90	HTTP: 0.090
Infrastructure	0.10	0.70	0.90	0.30	HTTP: 0.100
Total Score	1.00	0.47	0.63	0.78	HTTP Selected
Scenario C: Lossy Network Environment					
Use Case: Agricultural sensors over LoRaWAN with 15% packet loss					
Evaluation Criteria	Weight	MQTT Score	CoAP Score	HTTP Score	Weighted Contribution
Packet Loss Tolerance	0.35	0.50	0.90	0.20	CoAP: 0.315
Retransmission Efficiency	0.25	0.40	0.85	0.30	CoAP: 0.213
Energy Under Loss	0.20	0.45	0.80	0.10	CoAP: 0.160
Message Reliability	0.15	0.70	0.75	0.60	CoAP: 0.113
Overhead	0.05	0.70	0.90	0.30	CoAP: 0.045
Total Score	1.00	0.51	0.83	0.31	CoAP Selected

These results, achieved with no significant impact on message delivery or application functionality, provide strong evidence for the practical value of context-aware protocol selection as a strategy for energy optimization in IoT systems.

## 5. Discussion

Based on our experimental findings, we developed a prototype software system for adaptive protocol selection. This framework enables dynamic switching between communication protocols based on runtime conditions and user-defined priorities.

Integrating adaptive communication protocols within IoT frameworks has been shown to significantly enhance energy efficiency, as evidenced by implementations in mobile robotic systems [14]. Building on these advances, our framework extends the concept to general IoT deployments with varying energy constraints.

The experimental results can be explained by the fundamental architectural differences between the protocols. Looking into the stability of MQTT's superior performance in high-throughput scenarios is a direct consequence of its persistent, connection-oriented architecture. By establishing a single, long-lived TCP connection, it pays the initial energy and latency cost only once [2]. Subsequent messages are sent with minimal per-message overhead, making this model exceptionally efficient for continuous data streams or frequent updates. The publish/subscribe paradigm further enhances efficiency in one-to-many distribution scenarios, as the sender offloads the work of message delivery to the broker.

The finding that HTTP was the most energy-efficient for a small number of messages can be explained by its connection handling. In a non-keep-alive mode, a transaction is self-contained, avoiding the cost of maintaining an idle connection. However, this model becomes highly inefficient as frequency increases [13].

CoAP's underperformance in our local testbed highlights its optimization for low-power, lossy networks (LLNs), where minimizing radio "air time" over constrained links like 6LoWPAN is paramount [11,12]. In our reliable, high-bandwidth environment, these advantages were nullified.

The consistent finding that receivers consume more energy than senders challenges conventional assumptions about IoT energy optimization. Rather than transmission dominating energy costs, our results indicate that receiving, parsing, and managing messages requires more computational work, particularly at high message rates. This shifts the energy bottleneck from endpoint sensors to intermediary nodes like gateways and aggregators, requiring new power management strategies for these critical infrastructure components.

The data shows that there is no single "best" protocol; the optimal choice is highly dependent on the operational context (Table 4). A static choice leads to suboptimal performance. The adaptive framework validated in this study represents a logical software engineering solution, embedding intelligence at the edge to make real-time, localized optimizations. The demonstrated energy savings of up to 34% validate this approach. The value of context-aware adaptation has been proven in adjacent domains like dynamic access control [14] and service composition [16], reinforcing its applicability here.

**Table 4.** Scenarios vs. Network Constraints and Rationale for the Best-Fit Protocol.

Scenario	Network Constraints	Best-Fit Protocol	Rationale
Cold-chain tracking, fleet logistics	2G/LTE-M, metered data, 10,000+ msgs/day	MQTT	Persistent TLS session amortizes crypto costs; broker buffers while devices sleep
Smart-home hub pushing hourly statistics	Wi-Fi/Ethernet, $\leq 100$ msgs/day	HTTP/HTTPS	Minimal integration effort with cloud REST APIs; energy cost acceptable at low traffic
Remote environmental sensing (LoRa, BLE mesh)	$\leq 250$ kbps, long sleep cycles	CoAP	DP + 4-byte header slash airtime; confirmable mode gives reliability without TCP overhead

## 6. Limitations of Study and Future Work

- **Controlled Environment.** The experiments on a single machine do not capture the complexities of real-world wireless networks (e.g., packet loss, interference), a context where CoAP is designed to excel [11,17]. Future work will focus on validating the framework on a physical, large-scale IoT testbed using resource-constrained hardware to assess performance under these challenging conditions, where protocols like CoAP are designed to excel.
- **Energy Measurement Scope.** The use of Intel RAPL restricted measurements to CPU and DRAM packages [10,18]. The energy consumed by the network interface card, often a large power consumer, was not included [17,18]. Although we modeled this, a future iteration of this work will incorporate hardware-based power measurement tools to capture the total system energy consumption, providing a more holistic and accurate energy profile.
- **Security Overhead.** The experiments were conducted without transport-layer security (TLS/DTLS). The required cryptographic operations would add a substantial energy

cost [26]. A critical next step is to quantify this overhead for each protocol and integrate “security level” as a new decision criterion, allowing the framework to make intelligent trade-offs between energy efficiency and security.

- **Limited Protocol Scope.** The comparison excluded other relevant IoT protocols such as LoRaWAN or NB-IoT [17]. We plan to extend the framework’s adapters to include protocols like LoRaWAN and NB-IoT, enabling adaptation across a wider spectrum of connectivity options.
- **Simplified Adaptation Logic.** The prototype uses a simple rule-based engine, lacking the sophistication of modern ML techniques [20,27]. We plan to explore more sophisticated adaptation strategies, such as reinforcement learning (RL) or other machine learning models. This would enable the framework to learn optimal selection policies from its operational history and adapt to complex, non-linear patterns that a simple rule-based engine might miss.

## 7. Conclusions

Strategic frameworks that address energy inefficiencies in IoT networks underscore the importance of selecting appropriate communication protocols to optimize energy consumption [11]. The choice of an application-layer communication protocol profoundly influences the energy efficiency, performance, and sustainability of IoT systems. This empirical evaluation of MQTT, CoAP, and HTTP reveals distinct performance profiles that are highly dependent on the communication workload. Our findings confirm that MQTT is the most energy-efficient and low-latency protocol for high-volume, high-throughput applications due to its scalable publish/subscribe architecture and persistent connection model. HTTP, while less efficient at scale, remains a practical choice for low-volume applications that require simple integration with existing web infrastructure.

A key contribution of this work is the identification of a consistent energy imbalance, where receiver nodes consume significantly more energy than senders. This highlights the substantial computational cost of message processing and suggests that IoT system design must focus on optimizing not only endpoint sensors but also energy-constrained gateways and aggregators.

This study demonstrates that no single protocol is optimal for all contexts (Tables 2 and 3). This motivates our primary contribution: an adaptive protocol selection framework that enables IoT systems to dynamically switch communication methods based on runtime conditions and priorities. The prototype framework achieved energy savings of up to 34% compared to static protocol choices, proving that context-aware adaptation is a powerful and practical strategy for building more efficient and sustainable IoT systems. As the IoT continues its massive expansion, such adaptive, rule-based optimization techniques will be indispensable for managing operational costs and minimizing environmental impact.

MQTT demonstrated superior performance in high-throughput scenarios, with its persistent connection model providing excellent latency and energy efficiency at scale. The publish/subscribe architecture minimizes per-message overhead once a connection is established, making it ideal for applications requiring frequent data exchanges or real-time updates. Based on our findings, MQTT is recommended for high-frequency, latency-sensitive applications such as industrial monitoring, fleet management, or any scenario where devices transmit data regularly.

HTTP, despite its higher overhead, showed competitive energy efficiency for smaller message volumes. Its compatibility with existing web infrastructure makes it a practical choice for web-integrated IoT applications with infrequent transmissions. HTTP remains valuable when devices need to communicate directly with web services, or the development ecosystem prioritizes web technology integration.

CoAP, designed specifically for constrained environments, showed potential benefits in simpler messaging scenarios. Although it underperformed in our local network tests, its lightweight design and UDP-based communication model suggest advantages in low-power, lossy network environments that were not fully captured in our controlled testing environment.

Protocol selection influences IoT device sustainability and operational effectiveness.

The adaptive protocol selection framework we have introduced represents a significant step toward context-aware IoT communications that can balance energy efficiency with other operational requirements. Our adaptive selection framework achieved significant energy efficiency improvements (up to 23% compared to static MQTT usage), highlighting the value of dynamic protocol adaptation. IoT systems can achieve better energy efficiency while maintaining functional requirements by dynamically selecting the most appropriate protocol based on current conditions and priorities.

As IoT deployments continue to expand across industries and use cases, careful protocol selection and dynamic adaptation will play increasingly important roles in ensuring both operational effectiveness and environmental sustainability. Future research focusing on real-world deployments, security-energy trade-offs, and more sophisticated adaptation strategies will further advance our understanding of energy-efficient IoT communications.

**Author Contributions:** Conceptualization, D.Ž. and M.A.; methodology, D.Ž. and M.A.; software, M.A. and D.Ž.; validation, M.A. and D.Ž.; formal analysis, D.Ž. and M.A.; investigation, M.A.; resources, D.Ž.; data curation, M.A. and D.Ž.; writing—original draft preparation, M.A.; writing—review and editing, D.Ž.; visualization, M.A. and D.Ž.; supervision, D.Ž.; project administration, D.Ž. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The data and source code supporting the findings of this study are openly available at <https://github.com/mazarskov/research> (URL accessed on 12 November 2025) and <https://github.com/dizdev/research-iot-energy> (URL accessed on 12 November 2025).

**Conflicts of Interest:** The authors declare no conflicts of interest.

## Abbreviations

The following abbreviations are used in this manuscript:

2G	Second Generation
6LoWPAN	IPv6 over Low-Power Wireless Personal Area Networks
AI	Artificial Intelligence
API	Application Programming Interface
BLE	Bluetooth Low Energy
CoAP	Constrained Application Protocol
CPU	Central Processing Unit
DRAM	Dynamic Random Access Memory
DTLS	Datagram Transport Layer Security
FL	Federated Learning
GB	Gigabyte
HTTP	Hypertext Transfer Protocol
IoT	Internet of Things
LLN	Low-power, Lossy Networks
LoRaWAN	Long Range Wide Area Network

LPWAN	Low-Power Wide-Area Networks
LTE-M	Long Term Evolution for Machines
ML	Machine Learning
MQTT	Message Queuing Telemetry Transport
MSR	Model-Specific Registers
NB-IoT	Narrowband Internet of Things
OSI	Open Systems Interconnection
PMC	Hardware Performance Counters
QoS	Quality of Service
RAM	Random Access Memory
RAPL	Running Average Power Limit
REST	Representational State Transfer
RL	Reinforcement Learning
TCP	Transmission Control Protocol
TLS	Transport Layer Security
UDP	User Datagram Protocol

Appendix A

Table A1. Test configurations.

Scenario	Latency	Jitter	Bandwidth	Real-World Equivalent
LAN	1 ms	0.1 ms	Unlimited	Local network
WiFi	10 ms	5 ms	10 Mbps	Typical home WiFi
4G/LTE	50 ms	10 ms	5 Mbps	Cellular network
LoRaWAN	200 ms	20 ms	50 kbps	LPWAN
Satellite	600 ms	50 ms	1 Mbps	Remote IoT

Table A2. CoAP test results.

Name	Avg_Latency	Min_Latency	Max_Latency	Energy	Power	Runtime	Bytes	Energy_Per_Byte
S-C-100-1	3.87	2.36	5.75	291.407	2.297	100.146	37,300	0.007812520107
R-C-100-1	3.87	2.36	5.75	314.812	3.03	104.009	37,300	0.00844
S-C-100-10	3.74	1.77	9.4	37.613	4.402	10.161	37,300	0.001008391421
R-C-100-10	3.74	1.77	9.4	67.2	4.5	14.91	37,300	0.001801608579
S-C-100-100	2.06	1.43	6.16	6.235	6.579	1.166	37,300	0.0001671581769
R-C-100-100	2.06	1.43	6.16	41.276	6.926	5.885	37,300	0.001106595174
S-C-1000-10	3.68	1.62	9.94	348.065	3.56	100.141	373,000	0.000933150134
R-C-1000-10	3.68	1.62	9.94	378.568	3.619	104.915	373,000	0.001014927614
S-C-1000-100	2.48	1.49	6.36	50.508	5.227	10.156	373,000	0.0001354101877
R-C-1000-100	2.48	1.49	6.36	87.625	5.968	14.814	373,000	0.000234919571
S-C-10000-0	1.83	1.13	57.45	222.188	6.551	33.974	3,730,000	0.00005956782842
R-C-10000-0	1.83	1.13	57.45	252.105	6.485	38.885	3,730,000	0.00006758847185

Table A3. MQTT Results.

Name	Avg_Latency	Min_Latency	Max_Latency	Energy	Power	Runtime	Bytes	Energy_Per_Byte
S-M-100-1	0.63	0.36	0.99	323.682	3.247	100.521	81,200	0.003986231527
R-M-100-1	0.63	0.36	0.99	353.39	3.4	105.387	81,200	0.004352093596
S-M-100-10	0.84	0.36	3.37	42.081	4.144	10.516	81,200	0.0005182389163
R-M-100-10	0.84	0.36	3.37	87.734	5.768	16.295	81,200	0.00108046798
S-M-100-100	1.27	0.35	31.17	10	6.146	1.493	81,200	0.0001231527094
R-M-100-100	1.27	0.35	31.17	53.51	7.941	7.241	81,200	0.0006589901478
S-M-1000-10	0.88	0.38	7.73	343.648	3.452	100.512	812,000	0.0004232118227
R-M-1000-10	0.88	0.38	7.73	384.086	3.717	106.253	812,000	0.0004730123153
S-M-1000-100	0.86	0.31	35.95	45.705	4.455	10.496	812,000	0.00005628694581
R-M-1000-100	0.86	0.31	35.95	89.427	5.832	16.372	812,000	0.0001101317734
S-M-10000-0	0.63	0.18	42.41	87.84	6.033	14.767	8,120,000	0.00001081773399
R-M-10000-0	0.63	0.18	42.41	129.297	6.46	20.642	8,120,000	0.00001592327586

Table A4. HTTP results.

Name	Avg_Latency	Min_Latency	Max_Latency	Energy	Power	Runtime	Bytes	Energy_Per_Byte
S-H-100-1	1.61	0.8	4.56	324.737	3.287	100.174	103,600	0.003134527027
R-H-100-1	1.61	0.8	4.56	347.186	3.397	104.056	103,600	0.003351216216
S-H-100-10	1.57	0.67	4.64	37.018	4.236	10.174	103,600	0.0003573166023
R-H-100-10	1.57	0.67	4.64	67.361	4.536	14.947	103,600	0.0006502027027
S-H-100-100	1.35	0.67	4.57	5.829	5.816	1.186	103,600	0.00005626447876
R-H-100-100	1.35	0.67	4.57	35.969	5.94	5.931	103,600	0.0003471911197
S-H-1000-10	1.49	0.53	5.47	344.82	3.512	100.195	1,036,000	0.0003328378378
R-H-1000-10	1.49	0.53	5.47	374.835	3.584	104.956	1,036,000	0.0003618098456
S-H-1000-100	1.39	0.59	4.69	41.234	4.308	10.193	1,036,000	0.0000398011583
R-H-1000-100	1.39	0.59	4.69	72.864	5.339	15.04	1,036,000	0.00007033204633
S-H-10000-0	0.51	0.34	26.57	219.675	11.944	18.284	10,360,000	0.00002120415058
R-H-10000-0	0.51	0.34	26.57	247.449	10.66	23.152	10,360,000	0.00002388503861

## References

- Al-Obaidi, K.M.; Hossain, M.; Alduais, N.A.M.; Al-Duais, H.S.; Omrany, H.; Ghaffarianhoseini, A. A Review of Using IoT for Energy Efficient Buildings and Cities: A Built Environment Perspective. *Energies* **2022**, *15*, 5991. [\[CrossRef\]](#)
- Trilles, S.; González-Pérez, A.; Huerta, J. An IoT Platform Based on Microservices and Serverless Paradigms for Smart Farming Purposes. *Sensors* **2020**, *20*, 2418. [\[CrossRef\]](#)
- Almotairi, A.; Atawneh, S.; Khashan, O.A.; Khafajah, N.M. Enhancing intrusion detection in IoT networks using machine learning-based feature selection and ensemble models. *Syst. Sci. Control. Eng.* **2024**, *12*, 2321381. [\[CrossRef\]](#)
- Khan, L.U.; Saad, W.; Han, Z.; Hossain, E.; Niyato, D. Federated Learning for Internet of Things: Recent Advances, Taxonomy, and Open Challenges. *Network* **2021**, *1*, 17–53. [\[CrossRef\]](#)
- Wysocki, W.; Miciuła, I. How to Improve Software Energy Efficiency? A Systematic Literature Review and the Current State of Applied Methods in Practice. *arXiv* **2025**, arXiv:2025012232.
- Almudayni, Z.; Soh, B.; Samra, H.; Li, A. Energy Inefficiency in IoT Networks: Causes, Impact, and a Strategic Framework for Sustainable Optimisation. *Sensors* **2025**, *25*, 159. [\[CrossRef\]](#)
- Sharif Aldin, H.N.; Razavi Ghods, M.; Nayeibipour, F.; Niazi Torshiz, M. A comprehensive review of energy harvesting and routing strategies for IoT sensors sustainability and communication technology. *Sens. Int.* **2024**, *5*, 100258. [\[CrossRef\]](#)
- Al-Sarawi, S.; Anbar, M.; Alieyan, K.; Alzubaidi, M. Energy, Scalability, Data and Security in Massive IoT: Current Landscape and Future Directions. *IEEE Access* **2025**, *13*, 1.
- Tu, X.; Mallik, A.; Wang, H.; Xie, J. AIEnergy: An energy benchmark for AI-empowered mobile and IoT devices. *ITU J. Future Evol. Technol.* **2025**, *6*, 183–197. [\[CrossRef\]](#)
- Freina, D.; Jansen, M.; Trivedi, A. A Survey of Energy Measurement Methodologies for Computer Systems. *arXiv* **2024**, arXiv:2401.12345.
- Poyyamozhi, M.; Murugesan, B.; Rajamanickam, N.; Shorfuzzaman, M.; Aboelmagd, Y. IoT—A Promising Solution to Energy Management in Smart Buildings: A Systematic Review, Applications, Barriers, and Future Scope. *Buildings* **2024**, *14*, 3446. [\[CrossRef\]](#)
- Noman, U.A. Modular Interoperability Framework for IoT. Master's Thesis, Åbo Akademi University, Turku, Finland, 2017.
- Shahrokhi, A.; Ahmadi, M. Power Evaluation of IOT Application Layer Protocols. *arXiv* **2024**, arXiv:2405.07326. [\[CrossRef\]](#)
- Al Ridhawi, I.; Boukerche, A.; Al-Shabibi, A. Adaptive context-aware access control for IoT environments leveraging fog computing. *J. Netw. Comput. Appl.* **2024**, *235*, 103987.
- Alkhayyal, M.; Mostafa, A. Recent Developments in AI and ML for IoT: A Systematic Literature Review on LoRaWAN Energy Efficiency and Performance Optimization. *Sensors* **2024**, *24*, 4482. [\[CrossRef\]](#)
- Latre, S.; De Poorter, E.; Crombez, P.; Demeester, P. Adaptive and context-aware service composition for IoT-based smart cities. *IEEE Commun. Mag.* **2016**, *54*, 178–185.
- Islam, M.; Jamil, H.M.M.; Pranto, S.A.; Das, R.K.; Amin, A.; Khan, A. Future Industrial Applications: Exploring LPWAN-Driven IoT Protocols. *Sensors* **2024**, *24*, 2509. [\[CrossRef\]](#)
- Guo, C.; Ci, S.; Zhou, Y.; Yang, Y. A Survey of Energy Consumption Measurement in Embedded Systems. *IEEE Access* **2021**, *9*, 57467–57482. [\[CrossRef\]](#)
- Alsharif, M.H.; Kelechi, A.H.; Jahid, A.; Kannadasan, R.; Singla, M.K.; Gupta, J.; Geem, Z.W. A comprehensive survey of energy-efficient computing to enable sustainable massive IoT networks. *Alex. Eng. J.* **2024**, *91*, 12–29. [\[CrossRef\]](#)
- Ethan, A.; Karan, D. Energy-Efficient IoT Systems Using Machine Learning for Real-Time Analysis. *Int. J. Mach. Learn. Res. Cybersecur. Artif. Intell.* **2023**, *14*, 1307–1322.

21. Baqer, M. Energy-Efficient Federated Learning for Internet of Things: Leveraging In-Network Processing and Hierarchical Clustering. *Future Internet* **2025**, *17*, 4. [\[CrossRef\]](#)
22. Hasan, A.A.; Fang, X.; Latif, S.; Iqbal, A. Context-Aware Trust Prediction for Optimal Routing in Opportunistic IoT Systems. *Sensors* **2025**, *25*, 3672. [\[CrossRef\]](#) [\[PubMed\]](#)
23. Godfrey, D.; Suh, B.; Lim, B.H.; Lee, K.-C.; Kim, K.-I. An Energy-Efficient Routing Protocol with Reinforcement Learning in Software-Defined Wireless Sensor Networks. *Sensors* **2023**, *23*, 8435. [\[CrossRef\]](#)
24. Lee, S.; Lee, J. Non-IID Degree Aware Adaptive Federated Learning Procedure Selection Scheme for Edge-Enabled IoT Network. *Electronics* **2025**, *14*, 2331. [\[CrossRef\]](#)
25. Gutiérrez Hermosillo Muriedas, J.; Flügel, K.; Debus, C.; Obermaier, H.; Streit, A.; Götz, M. perun: Benchmarking Energy Consumption of High-Performance Computing Applications. In Proceedings of the European Conference on Parallel Processing, Limassol, Cyprus, 28 August–1 September 2023. [\[CrossRef\]](#)
26. Narkedimilli, S.; Makam, S.; Sriram, A.V.; Mallellu, S.P.; Sathvik, M.; Prasad, R.R.V. Enhancing IoT Network Security through Adaptive Curriculum Learning and XAI. *arXiv* **2025**, arXiv:2501.11618. [\[CrossRef\]](#)
27. Bharathi, R.; Kannadhasan, S.; Padminidevi, B.; Maharajan, M.S.; Nagarajan, R.; Tonmoy, M.M. Predictive Model Techniques with Energy Efficiency for IoT-Based Data Transmission in Wireless Sensor Networks. *J. Sens.* **2022**, *2022*, 3434646. [\[CrossRef\]](#)

**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.