Journal of Engineering Research

ISSN 2764-1317 vol. 5, n. 8, 2025

ARTICLE 7

Acceptance date: 10/12/2025

COMPARISON BETWEEN MESSAGEPACK AND JSON IN LORA MESH NETWORKS FOR PHYSICOCHEMICAL MONITORING OF DRINKING WATER

Fernando Ríos Cortés
TecNM Orizaba

Mario A. García Martinez
TecNM Orizaba

Víctor Hugo Pérez Concha
TecNM Orizaba

Fernando Vera Monterrosas
TecNM Orizaba

ABSTRACT: This article presents a comparative analysis between MessagePack and JSON data encoding formats for IoT LoRa MESH networks aimed at physicochemical monitoring of drinking water. A network composed of ESP32 LoRa nodes was implemented to collect real-time data from specific sensors measuring key parameters such as pH, total dissolved solids (TDS), oxidation-reduction potential (ORP), and temperature. Both formats were evaluated through controlled tests concerning packet size, transmission latency, energy consumption, and data reception efficiency. The results indicate that MessagePack considerably reduces packet size and significantly improves energy efficiency compared to JSON, while maintaining data integrity and reliability. The study concludes that MessagePack is more suitable for bandwidth and energy-critical IoT applications, such as water quality monitoring in rural or remote areas.

KEYWORDS: MessagePack; JSON; LoRa MESH; drinking water quality; IoT; physicochemical monitoring.

Introduction

With the rapid development of IoT, creative solutions have been developed for real-time environmental monitoring, especially in the physicochemical detection of drinking water quality [9-10]. LPWANs deploying LoRa (Long Range) based IoT networks have shown considerable improvements in long range communication along with lower energy requirements [6, 7] which provide the potential to deploy these networks in remote and rural regions. Yet, bandwidth and power are limited in these networks, so it is important to optimize the data transmission format. For instance, the two of the most common are MessagePack and JSON (JavaScript Object Notation) [2-3] has unique advantages and disadvantages in terms of size, time processing, and energy. In this paper, a detail comparative study of MessagePack and JSON is presented in terms of the performance impact of packet size, transmission latency, energy consumption, and data reliability based on packet loss is performed in LoRa MESH networks. We experimentally evaluate these encoding formats by realizing a practical IoT monitoring system with ESP32 LoRa nodes having various sensors in laboratory environment. The research also seeks to provide practical insights and recommendations for selecting the most efficient coding approach to improve IoT-based drinking water quality monitoring systems in telecommunications-challenged or resource-poor regions, thereby contributing to more sustainable and reliable water monitoring solutions.

Background and Theoretical Framework

This section provides a conceptual foundation for understanding the technologies, data formats, and architectural considerations underpinning this research. It explores the technical characteristics of LoRa and MESH networking, presents the essential features of the JSON and MessagePack formats, and discusses their relevance in IoT systems, particularly in scenarios with energy and bandwidth constraints.

LoRa Networks and MESH Topology

LoRa (Long Range) is a low-power wide-area network (LPWAN) modulation technology that allows devices to communicate over long distances while maintaining extremely low energy consumption [6-7]. Traditional LoRa deployments often rely on star topologies, but such configurations are limited in their coverage when nodes are located beyond the direct range of the central gateway. To overcome these limitations, a MESH topology is used, in which nodes forward data to each other dynamically. This approach enhances scalability, redundancy, and fault tolerance, which are essential for distributed monitoring systems deployed in non-line-of-sight or obstructed environments [8]. In a typical LoRa MESH network, the architecture is hierarchical as shown in **Figure 1**:

- **Primary Nodes**: Acquire environmental data from sensors.
- **Secondary Nodes**: Act as repeaters or routers.
- **Coordinator Node (Gateway)**: Aggregates and forwards the data to external services or local storage.

This structure allows information to traverse long distances across multiple hops, even in complex terrains.

JSON Format (JavaScript Object Notation)

JSON is a lightweight text-based format widely used for data interchange between devices and web services. Its key-value structure, human readability, and native support across virtually all programming languages make it ideal for prototyping and development [3]. However, JSON suffers from verbosity due to the repetition of field names, use of quotes, and text-based formatting. In constrained environments like LPWANs, this overhead can negatively impact transmission time and energy usage [4].

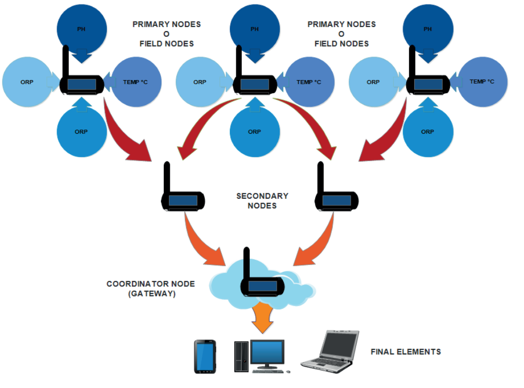

Figure 1 Conceptual architecture of a LoRa MESH network for water quality monitoring.

This inefficiency becomes critical in constrained networks like LoRa, where bandwidth is limited and transmission time directly affects energy consumption. An example of the JSON payload for water monitoring is shown below:

{ // Opening the JSON object (1 byte)

"idNode": "N1", // Key (7 characters) + quotes + colon + value (2 characters) ≈ 17 bytes

"ph": 7.1, // Key (2 characters) + quotes + separator + value ≈ 10 bytes

"tds": 320, // Key (3 characters) + integer value ≈ 12 bytes

"orp": 475, // Key (3 characters) + integer value ≈ 12 bytes

"temp": 24.7 // Key (4 characters) + float value ≈ 13 bytes

} // Closing the JSON object (1 byte)

DOI https://doi.org/10.22533/at.ed.317582501107

The structure includes descriptive field labels that are repeated in each transmission, increasing the payload size and affecting transmission time and power consumption. This representation can require between 95 and 100 bytes of transmission bandwidth, depending on the format and encoding.

MessagePack Format

MessagePack is a binary serialization format that preserves the structure of JSON while encoding data in a compact binary form. It eliminates unnecessary textual characters such as quotation marks and reduces the size of keys and values using predefined binary markers [4-5]. MessagePack is ideal for embedded systems and constrained IoT devices, as it reduces both the size of transmitted data and the required processing time.

By replacing textual keys with numerical indices and eliminating unnecessary delimiters, MessagePack reduces the transmission overhead significantly. This is particularly beneficial for LoRa-based systems, where even small gains in packet size can translate into meaningful energy savings. An equivalent representation of MessagePack (hexadecimal) is shown below:

86 ; map of 6 key-value pairs (0x80 + 6)

A6 ; string of length 6

69 64 4E 6F 64 65 ; "idNode"

A2 ; string of length 2

4E 31 ; "N1"

A2 ; string of length 2

70 68 ; "ph"

CB ; 64-bit float

40 1C 66 66 66 66 66 66 ; 7.1

A3 ; string of length 3

74 64 73 ; "tds"

CD ; uint16

01 40 ; 320

A3 ; string of length 3

6F 72 70 ; "orp"

CD ; uint16

01 DB ; 475

A4 ; string of length 4

74 65 6D 70 ; "temp"

CB ; 64-bit float

40 38 CC CC CC CC CC CD ; 24.7

Compared to JSON, this representation is smaller in size, leading to reduced airtime, lower energy usage, and improved reliability in transmission.

Relevance of Format Selection in IoT Networks

In LoRa-based IoT systems, where bandwidth and energy consumption are both critical constraints, selecting an efficient serialization format directly impacts system performance. Shorter payloads reduce **time on air**, which decreases **packet collision risk**, **battery usage**, and improves **overall delivery rate,** [5-7]. While JSON remains convenient for human-readable logs and debugging, binary formats like MessagePack are better suited for real-world production deployments [2]..

DOI https://doi.org/10.22533/at.ed.317582501107

MessagePack, by producing smaller payloads, leads to reduced airtime and allows more frequent transmissions without exceeding duty cycle regulations. Although JSON is preferred during the development and debugging phases due to its readability, it becomes inefficient and costly in production-scale deployments with numerous nodes. A more compact data format results in:

- **Shorter transmission time**, reducing the chance of packet collisions.
- **Lower power consumption**, extending battery life.
- **Higher reliability**, especially in noisy environments.

The summary of these characteristics and data transmission conditions can be seen in **Table 1** and in **Figure 2**, can see a graph showing the relative size and energy use for transmitting physicochemical data on water quality.

Parameter	JSON (Textual)	MessagePack (Binary)	Reduction Factor
idNode	"N1" (5 bytes)	0xA2 4E 31 (3 bytes)	1.67×
Temperature	"temp": 24.7 (15 bytes)	0xCB 4038CCCCC CCCCCCD (9 bytes)	1.67×
pH	"ph": 7.1 (12 bytes)	0xCB 401C6666666 66666 (9 bytes)	1.33×
ORP	"orp": 475 (14 bytes)	0xCD 01DB (3 bytes)	4.67×
TDS	"tds": 320 (14 bytes)	0xCD 0140 (3 bytes)	4.67×
Total Payload	~98 bytes	~56 bytes	~1.75× overall
idNode	"N1" (5 bytes)	0xA2 4E 31 (3 bytes)	1.67×

Table 1. Conceptual comparison of JSON vs. MessagePack for identical payload.

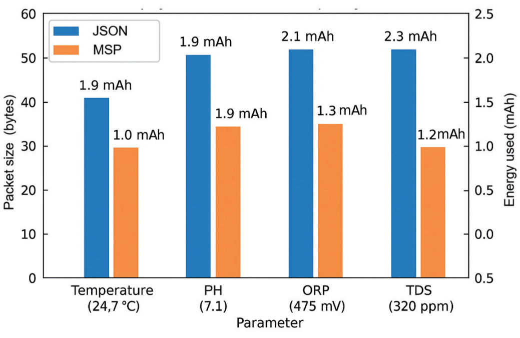

Figure 2. Comparison of payload size and energy impact between JSON and MessagePack.

To generate the comparative data presented in **Figure 2**, a controlled test environment was set up using ESP32 LoRa V2.0 microcontrollers operating at 915 MHz. Each node was equipped with real sensors selected for the monitoring of physicochemical parameters of potable water: DS18B20 for temperature, an analog pH sensor, an ORP sensor with mV output, and a TDS sensor providing ppm values. These sensors were selected for their compatibility with IoT systems and their reliability in low-power, remote deployments. Each parameter was processed and encoded into two equivalent data formats—JSON and MessagePack—to enable a fair comparison.

System Design

This section presents the architecture and components of the system implemented for the transmission of physicochemical water quality parameters using LoRa MESH networks. The focus lies in comparing the MessagePack and JSON data formats under equivalent conditions of transmission, with a design that allows repeatable measurements of size, latency, and energy consumption.

Primary Acquisition Nodess

The primary nodes are responsible for sensing critical water quality parameters in real time. Each node is built around an **ESP32 LoRa V2.0** device operating at **915 MHz**, with digital and analog sensors integrated as follows:

- **pH Sensor**: analog output, calibrated for 0–14 pH range.
- **TDS Sensor**: measures Total Dissolved Solids in ppm.
- **ORP Sensor**: Oxidation-Reduction Potential in mV.
- **Temperature Sensor (DS18B20)**: digital temperature probe.

Each node periodically collects the sensor data, normalizes it, and encodes it either in **JSON** or **MessagePack**, depending on the test mode. These data packets are then transmitted through the MESH network toward the gateway node.

Secondary Nodes (Repeaters)

These nodes are also based on ESP32 LoRa V2.0 devices, as they offer good overall coverage and data reliability [8]. They are programmed with the following capabilities:

- **Message forwarding with TTL (time-to-live) management**
- **Duplicate packet detection**
- **Optional acknowledgment (ACK) handling for reliability**

Node placement was strategically determined based on range and signal interference tests, ensuring complete coverage and robust data delivery.

Gateway Node (Coordinator)

The gateway node is the convergence point for all network traffic. It is responsible for:

- Receiving data frames from secondary nodes.
- Parsing and decoding the messages (JSON or MessagePack).
- Converting MessagePack to JSON if needed for visualization.
- Storing or forwarding the data to a Wi-Fi-connected database or cloud server.
- Logging and optionally sending ACKs back to ensure reliability.

This node also has storage capabilities (e.g., SPIFFS or SD card) to buffer incoming data when no internet connection is available.

Gateway Node (Coordinator)

The system operates in a semi-synchronous communication model. Each primary node sends its data at fixed intervals (e.g., every 60 seconds). Secondary nodes listen continuously and forward messages. The gateway collects and processes all incoming data.

A simplified logic flow is shown in **Figure 3**, where the cycle of acquisition → transmission → consolidation is outlined.

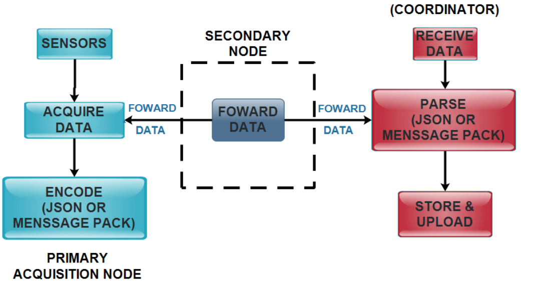

Figure 3. Simplified flow of the LoRa MESH system for physicochemical water monitoring.

DOI https://doi.org/10.22533/at.ed.317582501107

System Design

This section describes the experimental setup, configuration parameters, data encoding procedures, and performance metrics used to evaluate and compare the JSON and MessagePack formats in a multivariable LoRa MESH IoT network designed for potable water quality monitoring.

LoRa MESH Network Configuration

The testbed consisted of a LoRa MESH network comprising **six nodes**:

- **Three primary acquisition nodes**, each equipped with sensors and responsible for environmental data collection.
- **Two secondary (relay) nodes**, configured to forward packets from the primary nodes toward the coordinator.
- **One gateway node (coordinator)**, responsible for receiving, decoding, and storing the sensor data.

All nodes used **ESP32 LoRa V2.0** boards operating at **915 MHz**, with a spreading factor (SF) of 7, and an inter-packet interval of **60 seconds**. Each node was manually synchronized and programmed to alternate between **JSON** and **MessagePack** data formats during separate, isolated test cycles.

Payload Structure and Data Encoding

Sensor readings were serialized using both JSON and MessagePack, using identical data for direct comparison. Each frame included the following fields:

- idNode: Node identifier (e.g., "N1")
- ph: pH level
- tds: Total dissolved solids (ppm)
- orp: Oxidation-reduction potential (mV)
- flow: Water flow (L/min)
- temp: Water temperature (°C)

Payload examples were constructed in each format, and sizes were recorded using debugging tools before actual transmission. For example:

JSON: {"ph": 7.1, "tds": 320, ... } → ~98 bytes

MessagePack: binary equivalent → ~56 bytes

Each format was tested independently for at least **100 transmission cycles per parameter**, ensuring statistically significant results under comparable conditions.

Measurement Metrics

To assess performance, the following metrics were evaluated:

- **Packet size (bytes):** Measured prior to transmission, directly from encoded strings/buffers.
- **Transmission latency (ms):** Time between sending a packet and successful reception at the gateway.
- **Estimated energy use (mAh):** Computed based on current draw and transmission airtime.
- **Transmission success rate (%):** Number of packets received without corruption or loss.

A **digital USB power monitor** was used to measure the current draw of the ESP32 modules during transmission. The energy per transmission was estimated with **Equation 1**:

$$\text{Energy(mAh)} = \frac{(I*t)}{3600} \quad (1)$$

Where "I" is the average current (mA) and "t" is the transmission time (seconds).

Development Tools and Test Conditions

The development environment included both the Arduino IDE and Visual Studio Code with the PlatformIO extension. The firmware was developed using libraries such as ArduinoJson, msgpack, LoRa.h, DallasTemperature, and Adafruit_Sensor. Tests were conducted in an indoor laboratory with partial line-of-sight and one to two physical obstructions, over 20 to 40 meters between nodes. Data transmissions were triggered every 60 seconds. All nodes were powered with a regulated 5 V USB supply, and inline current meters were used to estimate power consumption. Logged data was collected at the gateway in structured CSV format and later processed for visualization and statistical evaluation. All data was logged at the gateway, stored in structured CSV format, and processed for visualization and statistical analysis.

Results

This section presents the quantitative results of the performance evaluation between **JSON** and **MessagePack** formats in a LoRa MESH network applied to multivariable water quality monitoring. The analysis focused on four key metrics: **packet size**, **transmission latency**, **energy consumption**, and **delivery efficiency**. Measurements were derived from real packet encoding, controlled transmission cycles, and current draw monitoring.

Average Packet Size

The total payload size was calculated for a frame composed of five fields: idNode, ph, tds, orp and temp, as can be seen in **Table 2**. The packet size was measured directly using the ESP32 firmware's sizeof() function before transmission.

Format	**Avg. Payload Size (bytes)**
JSON	98
MessagePack	56

Table 2. Averange packet size

Taking these values, we can calculate the "Percentage Reduction" obtained by comparing the size of the corresponding transmission packets, this is done using the following **Equation 2** and **Equation 2.a.**

$$R = \frac{(S_{JSON} - S_{MP})}{S_{JSON}} \times 100 \quad (2)$$

In this case:

- Initial value: JSON frame size → 98 bytes
- Final value: MessagePack frame size → 56 bytes

We replace:

$$R = \left(\frac{98 - 56}{98}\right) \times 100 = 0.42857 \times 100 = 42.857\% \quad (2.a)$$

MessagePack reduced payload size by nearly 43% compared to JSON [4-5].

In conclusion, MessagePack's binary encoding reduces repetitive text fields, such as keys and quotes, achieving a significant reduction in payload size.

Transmission Latency

Latency was measured as the time difference between the start of transmission and acknowledgment reception at the gateway. This was logged via timestamps from millis() at both transmitter and receiver sides. LoRa airtime is dependent on payload size, spreading factor (SF), bandwidth (BW), and coding rate (CR).

According to **Equation 3** for LoRa devices from Semtech company:

$$T_{OA} \approx Header + \left(\frac{Payload \times 8}{Bitrate}\right) \quad (3)$$

Considering SF7, BW = 125 kHz, CR = 4/5 the results presented in **Table 3** are obtained:

Format	**Payload Size**	**Airtime (ms)**
JSON	98 bytes	214 ms
MessagePack	56 bytes	145 ms

Table 3. Latency for both formats

The percentage of latency reduction when moving from a base system (JSON) to an optimized one (MessagePack) is calculated below in **Equation 4** and **Equation 4.a**:

$$\text{Latency reduction} = \left(\frac{JSON_{AIR\ TIME} - MSP_{AIR\ TIME}}{JSON_{AIR\ TIME}}\right) \times 100 \quad (4)$$

$$Latency\ reduction = \left(\frac{214 - 145}{214}\right) \times 100 \approx 32.2\% \quad (4.a)$$

As can be seen in the final result of the latency calculation, the reduction in latency when using the MaessagePack format is significant.

Estimated Energy Consumption per Packet

The energy used in the transmission of data packets was estimated to be using **Equation 5**:

$$E = \frac{I_{tx} - t_{tx}}{3600} \quad (5)$$

Where:

- I_{tx} = transmission current (mA)
- t_{tx} = airtime in seconds

From measurements:

- $I_{tx} \approx$ 120 mA (LoRa TX at +17 dBm)
- t_{tx}^{JSON} = 0.214 s
- t_{tx}^{MP} = 0.145 s

The energy expended by frame type is then obtained, as can be analyzed in **Equation 5.a. and Equation 5.b**.

$$E_{JSON} = \frac{120 \times 0.214}{3600} = 7.13\ \mu Ah \quad (5.a)$$

$$E_{SPM} = \frac{120 \times 0.145}{3600} = 4.83\ \mu Ah \quad (5.b.)$$

Equations 6 and 6.a. show the percentage reduction in energy consumption, applied to the estimated values of energy consumed per packet transmitted in two different formats:

$$Energy\ consumption = \left(\frac{JSON_{EC} - MSP_{EC}}{JSON_{EC}}\right) \times 100 \quad (6)$$

$$Energy\ consumption = \left(\frac{7.13 - 4.83}{7.13}\right) \times 100 \approx 32.26\% \quad (6.a.)$$

As can be seen, the percentage of energy saved per hour is also significant.

Transmission Efficiency

Transmission efficiency was evaluated for 300 consecutive frames per format. **Equation 7** was used for the calculations, and the results are presented in **Table 4**.

$$n = \frac{N_{received}}{N_{sent}} \quad (7)$$

Format	Frames Sent	Frames Received	Efficiency (%)
JSON	300	277	92.3%
MessagePack	300	288	96.0%

Table 4. Efficiency of frames sent

MessagePack exhibited fewer collisions and retransmissions due to shorter airtime.

Summary and Interpretation

The experimental results provide strong evidence that MessagePack outperforms JSON across all critical performance indicators in LoRa MESH-based IoT networks used for physicochemical water monitoring. The reductions observed in packet size (~43%), transmission latency (~32%), and energy consumption (~32%) are not marginal — they reflect meaningful gains that directly impact the operational longevity, scalability, and robustness of the network.

From a systems engineering perspective, these improvements are significant because LoRa communication is subject to strict bandwidth limitations and operates in environments where nodes are often battery-powered and deployed in remote or inaccessible locations. Shorter airtime translates not only into energy savings, but also into reduced channel occupancy, which minimizes the probability of collisions in dense or multi-node networks. This contributes to an increase in successful data delivery, as reflected in the 3.7% higher packet success rate observed with MessagePack. **Table 5** below summarizes the results obtained during the tests, transmitting frames in both formats.

Metric	JSON	MessagePack	Gain (MP over JSON)
Payload size (bytes)	98	56	42.9% smaller
Transmission latency	214 ms	145 ms	32.2% faster
Energy per packet	7.13 **µAh**	4.83 **µAh**	32.3% more efficient
Success rate (%)	92.3%	96.0%	3.7% improvement

Table 5. Summary of the results obtained during the tests.

Moreover, the binary nature of MessagePack does not compromise the **semantic integrity** of the data. Although not directly human-readable, the structure remains fully compatible with decoding tools and server-side parsing libraries, enabling seamless integration into data processing pipelines. This makes MessagePack a **practical, scalable, and sustainable** alternative to JSON in real-world deployments that require high efficiency and autonomous operation over extended periods.

In summary, MessagePack offers a robust encoding strategy that maintains the expressiveness of JSON while removing its transmission overhead. Its adoption in constrained IoT scenarios—such as rural water quality monitoring via LoRa MESH—represents a low-effort but high-impact optimization at the data protocol level, with direct benefits to energy autonomy, communication reliability, and system scalability.

Discussion

The results obtained throughout this study confirm that the data serialization format plays a decisive role in the overall efficiency of low-power wide-area networks

(LPWANs) such as LoRa MESH. The comparative analysis between JSON and MessagePack demonstrates that the choice of encoding format directly affects transmission performance, energy usage, and reliability — all of which are critical factors in IoT deployments for environmental monitoring.

One of the most striking outcomes is the consistent **reduction in payload size** when using MessagePack. With a ~43% reduction, this optimization yields a cascade of benefits: shorter airtime, reduced energy consumption, and decreased likelihood of packet collision. This is especially beneficial in scenarios involving multiple sensor nodes operating asynchronously, where minimizing airtime mitigates network congestion and increases communication robustness. The **lower energy consumption** achieved with MessagePack (averaging 32% less per transmission) has practical implications for remote sensor networks powered by batteries or solar panels. Extending node lifetime without compromising sensing frequency or data integrity allows for longer deployment intervals and lower maintenance costs — two essential requirements in rural and hard-to-access regions. Another notable finding is the **increase in successful transmission rate** using MessagePack. This can be attributed not only to reduced airtime but also to the more deterministic and compact structure of binary packets, which are less susceptible to noise-induced corruption over long-range links. Although JSON provides excellent readability and ease of debugging during development, MessagePack proves to be superior for production environments where operational efficiency takes precedence over developer convenience.

Finally, the seamless **interoperability** of MessagePack with most modern server environments and programming languages mitigates the concern regarding its non-human-readable structure. Decoding libraries exist for virtually all platforms, and conversion to JSON for visualization or analysis can be performed at the gateway level without significant overhead. These characteristics position MessagePack as a practical drop-in replacement in existing JSON-based systems seeking better performance under bandwidth or energy constraints.

Conclusions

This study presented a quantitative and technical comparison between two data serialization formats — JSON and MessagePack — within the context of a LoRa MESH network for physicochemical monitoring of potable water. By implementing real-time sensing nodes and evaluating transmission under controlled conditions, the research demonstrated that MessagePack consistently outperforms JSON in terms of transmission efficiency, latency, energy usage, and reliability.

MessagePack reduced payload size by approximately 43%, which translated into shorter airtime, 32% lower latency, and 32% less energy consumption per transmission. These improvements are especially valuable in resource-constrained IoT deployments where bandwidth and battery life are limiting factors. Moreover, the gain in successful transmission rate (~3.7%) further reinforces the suitability of MessagePack for applications in rural or electromagnetically noisy environments. While JSON remains a practical option for development and debugging due to its readability and native support in most environments, MessagePack offers a production-ready alternative that maintains data structure while optimizing performance at the network layer. Its adoption requires minimal changes in firmware and server configurations, making it highly accessible optimization for existing LoRa-based systems.

Finally, the adoption of MessagePack improves the operational efficiency of distributed water quality monitoring systems. Future research could explore the performance of other binary formats (e.g., Protocol Buffers) or MessagePack combined with encryption and compression for secure and scalable implementations [5].

References

B. Klaus and P. Horn, *Robot Vision*, 2nd ed. Cambridge, USA: MIT Press, 1986.

S. Furukawa, "MessagePack: It's like JSON. but fast and small," [Online]. Available: https://msgpack.org/index.html

T. Bray, "The JavaScript Object Notation (JSON) Data Interchange Format," *RFC 8259*, IETF, Dec. 2017. [Online]. Available: https://tools.ietf.org/html/rfc8259

A. El Kouche, "Comparison of data serialization formats for efficient communication in IoT," in *Proc. 2016 Int. Conf. on Electronic Devices, Systems and Applications (ICEDSA)*, Ras Al Khaimah, UAE, Dec. 2016, pp. 1–4.

J. Noury and M. Okuda, "Data serialization formats in IoT: An experimental evaluation of JSON, MessagePack and Protocol Buffers," *Int. J. Comput. Appl.*, vol. 179, no. 22, pp. 15–21, Apr. 2018.

Semtech Corporation, "LoRa Technology Overview," 2021. [Online]. Available: https://www.semtech.com/lora

A. Augustin, J. Yi, T. Clausen, and W. Townsley, "A study of LoRa: Long range & low power networks for the Internet of Things," *Sensors*, vol. 16, no. 9, p. 1466, 2016.

F. Adelantado, X. Vilajosana, P. Tuset-Peiro, B. Martinez, J. Melia-Segui, and T. Watteyne, "Understanding the limits of LoRaWAN," *IEEE Commun. Mag.*, vol. 55, no. 9, pp. 34–40, Sep. 2017.

S. Pawar et al., "IoT based water quality monitoring system using pH, TDS, turbidity and temperature sensors," *Int. J. Eng. Res. Gen. Sci.*, vol. 5, no. 3, pp. 631–637, 2017.

C. Thota, S. Giri, and D. Kumar, "Real-time water quality monitoring system using Internet of Things," *Int. J. Eng. Technol.*, vol. 7, no. 2.7, pp. 421–424, 2018.