frontiers | Frontiers in Computer Science

TYPE Original Research
PUBLISHED 17 September 2025
DOI 10.3389/fcomp.2025.1666262

OPEN ACCESS

EDITED BY
Stefano Rinaldi,
University of Brescia, Italy

REVIEWED BY
Vitor Fialho,
Instituto Politécnico de Lisboa, Portugal
Arshad Farhad,
Bahria University, Pakistan

*CORRESPONDENCE
Farhan Nisar
✉ Farhansnisar@yahoo.com

RECEIVED 15 July 2025
ACCEPTED 25 August 2025
PUBLISHED 17 September 2025
CORRECTED 19 September 2025

CITATION
Nisar F, Amin M, Touseef Irshad M, Hadi HJ, Ahmad N and Ladan M (2025) Machine learning-based spreading factor optimization in LoRaWAN networks.
Front. Comput. Sci. 7:1666262.
doi: 10.3389/fcomp.2025.1666262

Machine learning-based spreading factor optimization in LoRaWAN networks

Farhan Nisar[1]*, Muhammad Amin[1], Muhammad Touseef Irshad[2], Hassan Jalil Hadi[3], Naveed Ahmad[4] and Mohamad Ladan[5]

[1]Department of Physical & Numerical Sciences, Qurtuba University of Science & Information Technology, Peshawar, Pakistan, [2]National University of Modern Languages Peshawar Campus, Peshawar, Pakistan, [3]Center of Excellence in Cyber Security (CYBEX), Prince Sultan University, Riyadh, Saudi Arabia, [4]Software Engineering Department, Prince Sultan University, Riyadh, Saudi Arabia, [5]College of Computer and Information Sciences, Prince Sultan University, Riyadh, Saudi Arabia

The Internet of Things (IoT) has experienced rapid growth and adoption in recent years, enabling applications across diverse industries, including agriculture, logistics, smart cities, and healthcare. Long Range Wide Area Network (LoRaWAN) has emerged as a leading choice among IoT communication technologies due to its long-range, low-power, and cost-effective capabilities. However, the rapid proliferation of IoT devices has intensified the challenge of efficient resource management, particularly in spreading factor (SF) allocation for LoRaWAN networks. In this paper, we propose a Machine Learning-based Adaptive Data Rate (ML-ADR) approach for SF management to address this issue. A Long Short-Term Memory (LSTM) network was trained on a dataset generated using ns-3 for optimal SF classification. The pre-trained LSTM model was then utilized on the end-device side for efficient SF allocation with newly generated data during simulation. The results demonstrate improved packet delivery ratios and reduced energy consumption.

KEYWORDS

LoRaWAN, Internet of Things (IoT), machine learning (ML), LSTM, spreading factor (SF), transmission power (TP)

1 Introduction

The Internet of Things (IoT) has emerged as a transformative paradigm, enabling seamless integration of physical and digital worlds through interconnected devices. IoT applications span diverse domains, including smart cities, precision agriculture, industrial automation, and healthcare, revolutionizing data-driven decision-making (Augustin et al., 2016). A critical enabler of IoT is Low-Power Wide Area Network (LPWAN) technology, which provides long-range communication with minimal energy consumption. Among LPWAN solutions, SigFox, Narrowband IoT (NB-IoT), Weightless, and Long-Term Evolution for Machines (LTE-M) (Mekki et al., 2019; Gomez et al., 2019; Farhad et al., 2020a; Singh et al., 2020), Long Range Wide Area Network (LoRaWAN) has gained prominence due to its open standard, scalability, and adaptability to heterogeneous IoT deployments (Mekki et al., 2019).

Table 1 highlights the distinguishing characteristics of these IoT solutions. SigFox is recognized for its straightforward and economical deployment, whereas NB-IoT capitalizes on established cellular networks to deliver improved data throughput. The Weightless standard stands out for its adaptability and scalability, while LTE-M excels in mobility support and extended coverage. Notably, LoRaWAN (LoRa, 2020) has emerged as a dominant low-power wide-area network (LPWAN) technology, attracting considerable

interest due to its long-range capabilities combined with energy efficiency. Consequently, it has seen widespread adoption across both academic and industrial IoT implementations.

1.1 LoRa and LoRaWAN: an overview

Long Range (LoRa) is the physical layer (PHY) technology that underpins LoRaWAN, utilizing Chirp Spread Spectrum (CSS) modulation to achieve robust, long-range communication. CSS modulates data into chirp signals that sweep across a wide frequency band, offering inherent resistance to noise, multipath fading, and Doppler effects (Pasolini, 2021). This modulation technique enables LoRa to achieve a link budget exceeding 150 dB, supporting communication ranges of up to 15 km in rural areas and 2–5 km in urban environments (Farhad et al., 2020a) (Figure 1).

LoRaWAN, the Medium Access Control (MAC) layer protocol, manages network operations, including device authentication, adaptive data rate (ADR), and bidirectional communication. It operates in the unlicensed Industrial, Scientific, and Medical (ISM) bands (e.g., 868 MHz in Europe, 915 MHz in North America) and supports data rates from 0.3 kbps to 50 kbps, dynamically adjustable based on channel conditions (LoRa, 2020). Compared to alternatives like Sigfox and NB-IoT, LoRaWAN offers superior flexibility in private network deployments and Quality of Service (QoS) customization (Mekki et al., 2019).

1.2 LoRaWAN architecture and components

LoRaWAN employs a star-of-stars topology (Figure 2), comprising three primary components:

1. End Devices (EDs): battery-powered sensors or actuators that collect and transmit data using LoRa modulation. EDs are optimized for energy efficiency, with lifetimes ranging from 2 to 10 years (Singh et al., 2020).
2. Gateways (GWs): relay nodes that receive ED transmissions and forward them to the network server. Gateways support multi-channel, multi-SF reception, leveraging the "capture effect" to decode overlapping signals (Magrin et al., 2021).
3. Network Server (NS): the central component of LoRaWAN, responsible for deduplication, security, ADR optimization, and routing data to application servers (Farhad et al., 2022a).

1.3 Chirp spread spectrum and spreading factors

Chirp Spread Spectrum (CSS) modulation encodes data into chirp signals whose frequency increases or decreases linearly over time. This approach provides a processing gain, enhancing the signal-to-noise ratio (SNR) by expanding the signal bandwidth. This technique also enables orthogonality, allowing simultaneous transmissions on the same frequency by assigning different spreading factors (SFs) to end devices. The available SFs, ranging from 7 to 12, create an explicit trade-off between data rate, robustness, and range (ETSI, 2018). For example, a lower factor such as SF7 supports a higher data rate of 5.5 kbps, while a higher factor like SF12 reduces the throughput to 250 bps but can increase the communication range by 20% (Bor et al., 2016).

1.4 Device classes and class a operation

LoRaWAN classifies end devices into three distinct categories (Class *A*, Class *B*, and Class *C*) based on their communication patterns and power requirements (Farhad et al., 2020b). This classification system allows device manufacturers and application developers to select the most appropriate operational mode based on specific use case requirements. Class *A* devices, which represent the baseline implementation, employ an asynchronous, battery-optimized communication scheme where downlink messages are only permitted during two brief receive windows following each uplink transmission. This ALOHA-based approach minimizes energy consumption, making Class *A* ideal for applications like environmental sensors that transmit data infrequently and can tolerate some communication latency. Class *B* devices extend this functionality by introducing scheduled receive windows through periodic beacon messages from the gateway. These beacons synchronize the network and enable predictable downlink communication slots, which are particularly useful for applications such as firmware updates or configuration changes that require guaranteed delivery windows without maintaining constant connectivity. However, this additional functionality comes at the cost of moderately higher power consumption compared to Class *A*. Class *C* devices represent the most capable but power-intensive option, maintaining nearly continuous reception availability except during transmission periods. This class is typically employed for powered devices or applications requiring real-time bidirectional communication, such as street lighting control or industrial automation systems where immediate command execution is critical. The hierarchical class structure of LoRaWAN provides developers with flexibility to balance energy efficiency with communication responsiveness based on their specific application requirements.

1.5 Class A receive windows

When a Class A device transmits an uplink, it subsequently opens two receive windows, as shown in Figure 2. The first, RX1, opens after a fixed delay (e.g., 1 s in the EU868 region) and utilizes the same frequency and spreading factor (SF) as the preceding uplink transmission. Following this, a second configurable window, RX2, opens (e.g., 2 s later). RX2 operates on a default frequency and SF (e.g., SF12), serving as a fallback for the network server to respond if it misses the opportunity during RX1.

1.6 Adaptive data rate and blind adaptive data rate

LoRaWAN includes a built-in Adaptive Data Rate (ADR) mechanism that aims to improve communication efficiency by

TABLE 1 Comparative analysis of prominent IoT communication technologies (Farhad et al., 2022a; Farhad and Pyun, 2023b,c).

Technology aspect	LoRaWAN	Sigfox	NB-IoT	Weightless	LTE-M
Frequency band	Unlicensed ISM bands (e.g., 868 MHz EU, 915 MHz US)	Unlicensed ISM bands (e.g., 868 MHz EU, 902 MHz US)	Licensed LTE bands (in-band, guard-band, standalone)	Sub-1 GHz ISM bands	Licensed LTE bands
Channel bandwidth	125 kHz, 250 kHz, 500 kHz	100 Hz (uplink), 600 Hz (downlink)	180 kHz	12.5 kHz	1.4 MHz
Modulation scheme	CSS (Chirp Spread Spectrum)	DBPSK (uplink), GFSK (downlink)	QPSK	GMSK, QPSK	QPSK, 16 QAM
Max. application payload	51 to 242 bytes (region-dependent)	12 bytes (uplink), 8 bytes (downlink)	~1,600 bytes	Variable, app-defined	~1,500 bytes
Data throughput	0.3 kbps to 50 kbps	100 bps (uplink), 600 bps (downlink)	~250 kbps (downlink), ~20 kbps (uplink)	Up to 100 kbps	Up to 1 Mbps
Typical range [km]	Urban ≈ 2–5, Rural > 15	Urban ≈ 3–10, Rural ≈ 30–50	Urban ≈ 1–2, Rural ≤ 10	Urban ≈ 2	Enhanced coverage up to 10 km
Adaptive rate control	Yes	No	Yes	Yes	Yes
Power profile	Extremely low	Extremely low	Low	Low	Moderate
Mobility	Supported (handovers can be challenging)	Not supported	Supported in connected mode	Supported	Full, seamless handovers
Positioning method	Uplink TDoA and RSSI (Farahsari et al., 2022; Torres-Sospedra et al., 2022)	Network-based trilateration	OTDOA, E-CID	Supported	OTDOA, E-CID
Private deployment	Yes, fully supported	No, public network only	Yes, via network slicing	Yes	Yes, via network slicing
Two-Way communication	Fully bidirectional	Limited (uplink-focused)	Fully bidirectional	Fully bidirectional	Fully bidirectional
Network model	Public or private	Public (operator-led)	Public (operator-led)	Open standard	Public (operator-led)
Available simulators [public]	Yes (Sartori, 2023; Zorbas et al., 2021; Beltramelli et al., 2021; Loh et al., 2021; Casals et al., 2021; Zorbas et al., 2020; Abdelfadeel et al., 2019; Callebaut et al., 2019; Ta et al., 2019; Reynders et al., 2018; To and Duda, 2018; Slabicki et al., 2018; Bounceur et al., 2018; Croce et al., 2018; Magrin et al., 2017; Van den Abeele et al., 2017; Pop et al., 2017; Bor et al., 2016)	Yes (Weyn and Contributors, 2023; DEIS-Tools Project Team, 2023; SAPGAN Team, 2023)	Yes (Gdbranco, 2023; a3794110, 2023)	Not publicly available	Yes

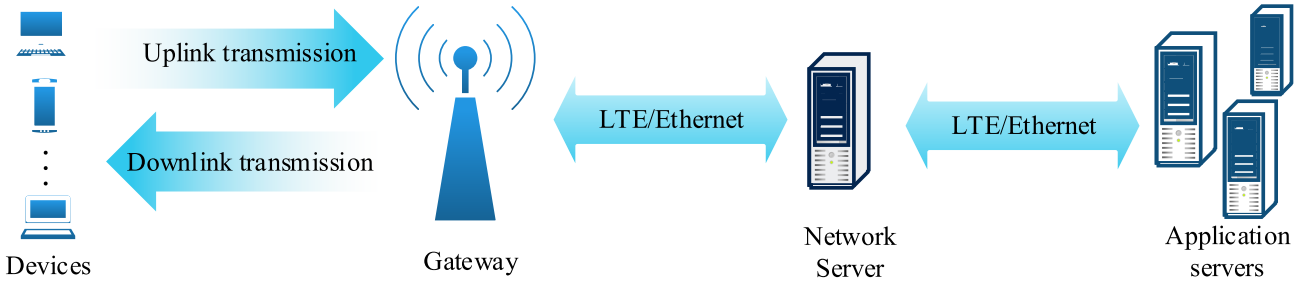

FIGURE 1
LoRaWAN architecture comprising end devices, gateways, network servers, and application servers.

dynamically adjusting the SF and transmission power (TP) of end devices (EDs) (Marini et al., 2021; Anwar et al., 2021; Moysiadis et al., 2021; Park et al., 2020; Benkahla et al., 2019; Semtech, 2019a,b; ETSI, 2018; Farhad et al., 2021). This adjustment is typically handled by the network server (NS), which evaluates the signal-to-noise ratio (SNR) over the most recent 20 uplink transmissions.

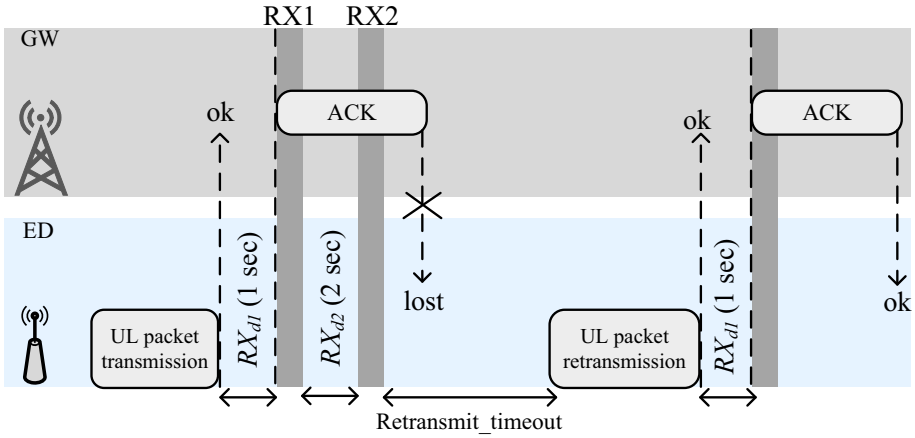

FIGURE 2
Class A receive windows operations.

Based on this assessment, the NS may increment or decrement the TP in steps of 2 dBm and modify the SF to optimize energy consumption and spectral usage (Semtech, 2019b).

While ADR works effectively for static or quasi-static devices, it exhibits slow responsiveness and increased packet loss in mobile or rapidly changing environments. The reliance of ADR on historical SNR data leads to sluggish adaptation to changing radio conditions, making it suboptimal for mobile devices (Anwar et al., 2021). Furthermore, adjusted SF/TP settings may become outdated in real time, especially during mobility, resulting in failed transmissions and unnecessary retransmissions (Farhad and Pyun, 2023a).

To address these limitations, a more agile strategy termed Blind Adaptive Data Rate (BADR) has been proposed by Semtech (Farhad et al., 2021), as illustrated in Figure 3. BADR operates at the ED side, where SF12 is assigned once, SF7 thrice, and SF10 twice in a 60-min duration, blindly. Unlike conventional ADR, BADR allows end devices to infer optimal transmission parameters autonomously, without relying on downlink feedback from the network server. This is particularly valuable in uplink-heavy LoRaWAN applications where frequent ACKs are infeasible due to duty cycle constraints and limited gateway availability.

1.7 Problem statement

While ADR and BADR provide baseline methods for configuring SF and TP, they either adapt too slowly (ADR) or fail to adapt at all (BADR). Consequently, there remains a critical need for a resource allocation mechanism that can react swiftly and intelligently to changing wireless conditions. To this end, we propose a Machine Learning-based ADR (ML-ADR) approach, wherein a trained model predicts the optimal SF for each ED based on real-time input features. By leveraging historical data and contextual parameters, ML-ADR aims to minimize packet loss, improve the packet success ratio (PSR), and reduce energy consumption—thereby overcoming the limitations of both ADR and BADR.

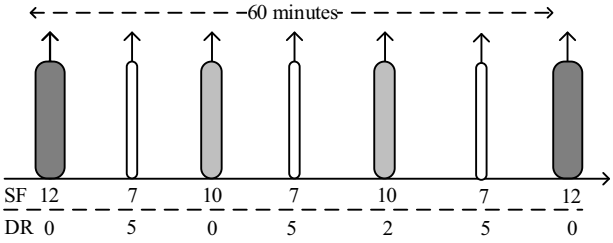

FIGURE 3
Blind adaptive data rate operation.

1.8 Contribution of the paper

The contribution of this paper is as follows:

1. We design a deep neural network model that can learn optimal SF strategies based on underlying network conditions and requirements to address the SF allocation issue.
2. The deep neural network model is trained on a one-time dataset generated in the ns-3 simulator, considering the propagation environment, device positions, distance between GW and ED, and successful SF. After training, the pre-trained model is utilized at the NS for optimal SF allocation to EDs during network simulations.
3. During the simulation-based deployment scenario using ns-3, the proposed ML-ADR allocates the best SF to EDs, thereby enhancing the packet delivery ratio and reducing energy consumption.

1.9 Structure of paper

Section 2 presents an in-depth review of existing AI-based solutions for resource management in LoRaWAN. Section 3 elaborates on the dataset collection, discusses the required features, and highlights the most suitable ML methods for resource

allocation based on the features. Section 4 presents the operation of the proposed ML-ADR. Section 5 presents a detailed discussion of experiments and result analysis in offline mode. Section 6 presents the result analysis in ns-3, where the ML algorithm is utilized with the simulation data. Finally, Section 8 provides concluding remarks.

2 Literature review

Recent studies have explored various machine learning approaches to optimize LoRaWAN resource allocation, particularly focusing on spreading factor assignment, transmission power control, and device classification. These efforts can be categorized into three main paradigms: reinforcement learning for dynamic SF adaptation, supervised learning, and hybrid approaches combining their strengths.

2.1 Reinforcement learning approaches

Several reinforcement learning methods (RL) have demonstrated effectiveness in resource allocation. The mixed multi-armed bandit approach developed by Azizi et al. (2022) achieved notable improvements in packet delivery ratio and energy efficiency within simulated single-gateway networks. Their Python-based implementation considered 100 end devices under EU-868 MHz regulatory constraints, demonstrating the feasibility of RL solutions for static node deployments. Building upon this work, Chen et al. (2023) introduced a score table-based reinforcement learning method that reduced energy consumption by 24–27% compared to conventional ADR techniques. Their Matlab-based simulations confirmed the lightweight nature of the algorithm, making it suitable for practical implementations.

One study proposed a resource allocation mechanism using two independent ML approaches: a centralized supervised ML approach for transmission power allocation and a decentralized RL approach using the EXP4 algorithm for SF allocation, treating it as a contextual multi-arm bandit problem for maximizing packet reception ratio (PRR) (Garlisi et al., 2021). This approach aims to minimize energy consumption per packet (EPP) by addressing energy minimization and PRR maximization separately. The proposed method showed significant improvements in both network goodput and energy consumption, especially in large and congested networks, and the RL algorithm converged much faster than previous methods by using expert advice. A potential issue was that the algorithm requires feedback (e.g., ACK) from the gateway for every uplink packet during the training phase to update its reward and probabilities, which consumes channel resources and energy, although downlink ACKs can be eliminated after training.

2.2 Supervised and deep learning approaches

Supervised learning approaches have showed particular promise for device classification tasks. A study implemented a support vector machine classifier that accurately distinguished between mobile and static end devices using limited training data. While the study successfully demonstrated device classification, it did not extend to adaptive data rate selection based on mobility patterns. Deep learning methods achieved superior performance in complex scenarios, with Farhad et al. (2022b) reporting 96% classification accuracy using a gated recurrent unit network. Their ns-3 simulations with 500 nodes validated the model's effectiveness, achieving a 98% packet delivery ratio in moderate-density networks.

A study in Hazarika and Choudhury (2024) investigated a smart SF assignment technique utilizing deep learning architectures, specifically Fully Connected Neural Networks (FCNN) and Convolutional Neural Networks (CNN), for joint collision detection and optimal SF selection in a static environment. The proposed technique demonstrated higher prediction accuracy compared to traditional machine learning algorithms and improved network energy consumption. However, the CNN model showed lower accuracy due to the lack of spatial correlation in the converted data, and the prediction accuracy generally decreased with an increasing number of nodes due to the entanglement of differently labeled samples in the dataset.

The authors in Acosta-Garcia et al. (2024) proposed a proactive ADR mechanism, for mobile LoRa-based IoT devices that utilizes trajectory estimation and the k-nearest neighbors (KNN) algorithm to forecast the signal-to-noise ratio (SNR) and proactively adapt transmission parameters (SF and TP). This approach aims to quickly adapt parameters without requiring long data acquisition times, considering device dynamics and environmental factors to predict signal quality variations as nodes move. The KDR mechanism demonstrated significant improvements in reducing SNR infringements and Bit Error Rate (BER), as well as lowering energy consumption compared to traditional ADR and Blind ADR, maintaining performance even with varying device speeds and limited SNR information. A potential issue is that energy consumption slightly increases with a larger number of available buffered SNR samples, although this is seen as necessary to ensure compliance with quality metrics.

2.3 Hybrid and emerging approaches

Hybrid approaches combining multiple techniques have emerged as particularly effective. Minhaj et al. (2023) demonstrated that integrating RL for SF allocation with ML for TP control outperformed single-method solutions. The authors developed an augmented sensing method that fused LoRaWAN signal metrics with environmental sensor data, reducing estimation errors by 17% compared to standalone approaches.

The surveyed literature reveals three critical gaps that directly motivate our ML-ADR solution: (1) RL methods like Azizi et al. (2022); Chen et al. (2023) achieve dynamic adaptation but rely on slow reward feedback loops, resulting in latency issues; (2) supervised approaches (Farhad et al., 2022b; Hazarika and Choudhury, 2024) improve classification accuracy but lack real-time adaptability, mirroring the static limitations of BADR; and (3) hybrid techniques (Garlisi et al., 2021; Minhaj et al., 2023) partially address mobility but introduce computational overhead unsuitable

TABLE 2 Summary of existing approaches in LoRaWAN for SF optimization.

Reference	Method	Resource Management	Key Improvement
Azizi et al. (2022)	MIX-MAB RL	SF allocation	22% PDR increase
Chen et al. (2023)	STEP RL	SF allocation	26% energy reduction
Minhaj et al. (2023)	Hybrid ML/RL	SF/TP control	Combined optimization
Bertocco et al. (2023)	Augmented sensing	Soil monitoring	1.53% RMSE
Vangelista et al. (2023)	SVM	Device classification	94% accuracy
Farhad et al. (2022b)	GRU network	SF classification	98% PDR
Farhad and Pyun (2023a)	DNN	Mobile SF allocation	82% accuracy
Elkarim et al. (2022)	Deep Learning (FCNN, CNN)	SF Allocation	Energy consumption
Minhaj et al. (2023)	Supervised ML + RL (EXP4)	SF, TP	Energy consumption
Hazarika and Choudhury (2024)	K-means + RL	SF Allocation	Packet success rate
Acosta-Garcia et al. (2024)	KNN	Proactive ADR	Energy consumption

for constrained EDs. Our work bridges these gaps by unifying temporal modeling (LSTM), lightweight inference, adapting to channel dynamics while maintaining energy efficiency. Table 2 summarizes these comparative insights.

3 Data generation and preprocessing framework

The proposed data generation and preprocessing framework is engineered to handle LoRaWAN transmission data through a novel methodology based on 20-step sequence windows. Each sequence encapsulates information derived from the selection of an optimal SF determined from simultaneous multi-SF transmissions.

3.1 Transmission protocol

The core of the data acquisition process relies on a specific transmission protocol executed by each ED. In this protocol, an identical data packet is transmitted concurrently utilizing all six available spreading factors, spanning *SF*7 through *SF*12. These transmissions are conducted in the confirmed mode, mandating the reception of an ACK signal from the GW for successful communication verification. For every transmission attempt originating from an ED, which comprises six parallel transmissions (one per *SF*), the receiving GW meticulously records the reception status (successful or failed) along with pertinent signal quality metrics for each individual *SF* transmission. Correspondingly, the originating ED logs the ACK reception status, represented as a binary value (1 for received, 0 for not received), for each of the six *SF*s employed in the simultaneous transmission event.

3.2 Optimal spreading factor determination and feature extraction

Subsequent to each multi-SF transmission event, an optimal SF, denoted as SF^*, is identified. This SF^* is determined by selecting the minimum *SF* value among those transmissions for which a corresponding ACK was successfully received by the ED. This selection process is formally expressed as:

$$SF^* = \min\{SF_j | \text{ACK}_j = 1\}_{j=7}^{12} \tag{1}$$

In scenarios where no ACKs are received across the entire set of transmitted *SF*s (*SF*7–*SF*12), a default assignment of *SF*12 is made for SF^*. The identified optimal SF^* is then paired with a comprehensive feature vector, $\mathbf{f}$, encapsulating the relevant signal characteristics and contextual information associated with that specific successful transmission (or the *SF*12 transmission if no ACK was received). This feature vector is structured as follows:

$$\mathbf{f} = [x, y, d, \text{SNR}, \text{SNR}_{\text{req}}, \text{SNR}_{\text{margin}}, d_{\text{norm}}, P_{rx}] \tag{2}$$

Here, x and y represent the Cartesian coordinates of the ED, d is the calculated Euclidean distance to the GW derived as $d = \sqrt{x^2 + y^2}$, SNR denotes the measured Signal-to-Noise Ratio, SNR_{req} signifies the minimum required SNR threshold for successful demodulation at the given SF^*, $\text{SNR}_{\text{margin}}$ is the calculated SNR margin defined as $\text{SNR}_{\text{margin}} = \text{SNR} - \text{SNR}_{\text{req}}$, d_{norm} represents the distance normalized by the maximum communication range $R_{\max}$ (i.e., $d_{\text{norm}} = d/R_{\max}$), and P_{rx} is the received signal power measured at the GW.

3.3 Temporal sequence construction

Input samples for subsequent analysis or machine learning model training are systematically generated using a sliding window technique over the time series of optimal SF^* selections and their corresponding feature vectors $\mathbf{f}$. Each input sample, designated as $\mathbf{X}_i$, is constructed as a matrix comprising the feature vectors from 20 consecutive transmission events, specifically encompassing the data from time step $i - 19$ through the current time step i. The structure of this input matrix is represented by:

$$\mathbf{X}_i = \begin{bmatrix} \mathbf{f}_{i-19} \\ \mathbf{f}_{i-18} \\ \vdots \\ \mathbf{f}_i \end{bmatrix}_{20\times 8} \tag{3}$$

The associated target label for this input matrix $\mathbf{X}_i$, denoted by y_i, is the optimal SF^*, specifically SF_i^*, corresponding to the final

TABLE 3 Data structure before and after windowing.

Property	Raw data	Windowed
Data	72,000	71,981
Timesteps per sample	1	20
Features per timestep	8	8
Total features per sample	8	160
Label dimension	1	1

time step i within the window. Consequently, each input matrix $\mathbf{X}_i$ has dimensions of 20 rows (temporal steps) and 8 columns (features per step), resulting in a total of 160 feature values per input sample.

The simulations were conducted with 500 EDs over a 24-h period, with ED transmitting six confirmed uplink messages per hour, resulting in $\sim$ 72,000 raw transmission events. After applying the 20-step sliding window, the final processed dataset comprised 71,981 sequences, each with 160 features (20 timesteps $\times$ 8 features) and a corresponding optimal SF label, as shown in Table 3.

3.4 Framework characteristics

This data generation and preprocessing methodology possesses several key characteristics pertinent to modeling LoRaWAN channel behavior. The utilization of 20-step temporal windows inherently incorporates temporal context, enabling the potential to capture patterns related to channel variations and signal quality fluctuations over time. The feature vector $\mathbf{f}$ provides a rich, multi-dimensional representation of the communication link state at each time step, integrating metrics related to signal strength, signal quality relative to requirements, and spatial positioning. Moreover, defining the target label SF^* based on empirically successful transmissions furnishes a form of ground truth that reflects practically achievable link performance under the observed conditions. The employed multi-SF transmission protocol also offers efficiency in data collection, as a single transmission event yields data points pertaining to the reception status and signal metrics across the full operational range of SFs.

4 Proposed methodology

We propose a machine learning framework based on Long Short-Term Memory (LSTM) networks for optimizing LoRaWAN communication parameters, with particular focus on dynamic SF selection. The choice of LSTM is motivated by its demonstrated effectiveness in modeling temporal dependencies within sequential data, a critical requirement for analyzing time-varying LoRa signal characteristics (Farhad and Pyun, 2023a). Unlike traditional machine learning approaches such as Random Forests or Support Vector Machines that treat each data sample independently, LSTMs explicitly model the temporal relationships between consecutive transmissions. This capability is especially valuable in LoRaWAN environments where channel conditions, interference patterns, and device mobility create complex temporal dynamics that influence optimal SF selection.

The proposed methodology addresses three key challenges in LoRaWAN optimization: (1) the non-stationary nature of wireless channels in IoT deployments, (2) the trade-off between data rate and communication range inherent in SF selection, and (3) the need for energy-efficient communication strategies. Our LSTM-based approach captures these aspects through a hierarchical learning architecture that processes sequences of transmission events while maintaining memory of long-term patterns. This contrasts with conventional approaches that either use static SF assignments or rely on instantaneous channel measurements without historical context.

4.1 LSTM-based temporal modeling

The LSTM architecture employed in this study is designed to model temporal dependencies across sequential LoRaWAN transmission events. The input to the LSTM is a matrix $\mathbf{X} \in \mathbb{R}^{T\times D}$, where T denotes the number of time steps and D represents the feature dimension per step. Based on domain-specific empirical insights, we set $T = 20$ to include sufficient temporal history from the last twenty uplink transmissions, which balances information richness and computational efficiency. Each timestep in $\mathbf{X}$ is an 8-dimensional vector composed of features such as P_{rx}, SNR, x, y, distance between ED and GW, SNR marging, as specified in Equation 2.

The internal architecture of the LSTM unit aligns with the standard gating-based formulation. At each timestep t, the model processes the input vector $\mathbf{x}_t$ alongside the previous hidden state $\mathbf{h}_{t-1}$ and the previous cell state $\mathbf{C}_{t-1}$. These vectors are concatenated and passed through three distinct gates: the forget gate $\mathbf{f}_t$, the input gate $\mathbf{i}_t$, and the output gate $\mathbf{o}_t$. Additionally, a candidate cell state $\tilde{\mathbf{C}}_t$ is computed to propose an update to the memory content. Each gate is parameterized by its own learnable weights and biases, which are optimized during training.

$$\begin{aligned}
\mathbf{f}_t &= \sigma(\mathbf{W}_f[\mathbf{h}_{t-1}; \mathbf{x}_t] + \mathbf{b}_f) \\
\mathbf{i}_t &= \sigma(\mathbf{W}_i[\mathbf{h}_{t-1}; \mathbf{x}_t] + \mathbf{b}_i) \\
\mathbf{o}_t &= \sigma(\mathbf{W}_o[\mathbf{h}_{t-1}; \mathbf{x}_t] + \mathbf{b}_o) \\
\tilde{\mathbf{C}}_t &= \tanh(\mathbf{W}_C[\mathbf{h}_{t-1}; \mathbf{x}_t] + \mathbf{b}_C) \\
\mathbf{C}_t &= \mathbf{f}_t \odot \mathbf{C}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{C}}_t \\
\mathbf{h}_t &= \mathbf{o}_t \odot \tanh(\mathbf{C}_t)
\end{aligned} \tag{4}$$

In Equation 4, the forget gate $\mathbf{f}_t$ modulates how much of the previous cell state $\mathbf{C}_{t-1}$ should be retained. The input gate $\mathbf{i}_t$ determines the extent to which the newly proposed memory content $\tilde{\mathbf{C}}_t$ should influence the current cell state. The cell state $\mathbf{C}_t$ is updated via element-wise combinations of the retained past memory and the newly accepted content. Finally, the output gate $\mathbf{o}_t$ governs how much of the updated cell state should be exposed as the hidden state $\mathbf{h}_t$ for downstream layers or subsequent time steps. The sigmoid (σ) and hyperbolic tangent (tanh) activation functions ensure bounded nonlinearity and numerical stability, in accordance with the flow shown in Figure 4.

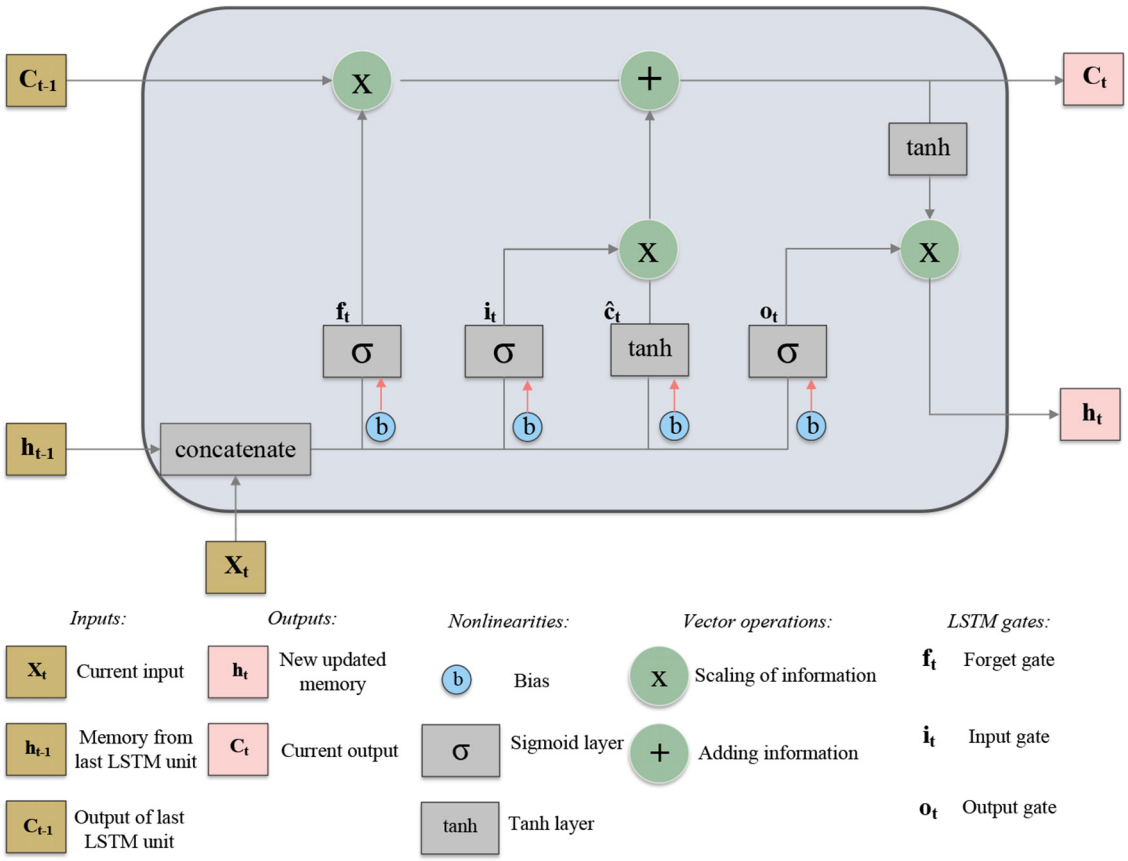

FIGURE 4
LSTM architecture used for temporal modeling of LoRaWAN uplink transmissions. The model receives sequential input vectors x_t alongside the hidden state h_{t-1} and cell state C_{t-1}. It processes these through gating mechanisms—forget gate (f_t), input gate (i_t), candidate cell state ($\tilde{C}_t$), and output gate (o_t)—to update its internal memory C_t and generate the hidden representation h_t at each timestep. Nonlinearities σ and tanh are used to enable stable, bounded transformations within the network. This architecture allows the model to retain long-term dependencies and adaptively control information flow over time.

All matrix parameters $\mathbf{W}_f, \mathbf{W}_i, \mathbf{W}_o, \mathbf{W}_C$ and their corresponding biases $\mathbf{b}_f, \mathbf{b}_i, \mathbf{b}_o, \mathbf{b}_C$ are jointly learned via backpropagation through time. The architecture thus enables effective modeling of both short-term signal fluctuations and long-term temporal dependencies in LoRaWAN uplink sequences, capturing the dynamics essential for reliable SF classification.

4.2 LSTM training mechanism

The implemented network employs a stacked LSTM architecture with two layers, each containing 128 hidden units, as illustrated in Table 4. We chose LSTM due to its stronger memory capacity for modeling long-term dependencies in time-series data, which is critical in capturing trends across successive transmissions in a dynamic wireless channel. LSTM provides sufficient capacity to model complex temporal relationships while avoiding excessive computational overhead. The first LSTM layer processes the raw input sequence, while the second layer extracts higher-level temporal patterns from the first layer's output. Between these layers, we maintain sequence continuity through stateful processing, where the final state of one batch serves as the initial state for the next batch during training.

Following the LSTM layers, the architecture incorporates a series of fully connected (dense) layers with ReLU activation functions ($\max(0, x)$). These layers transform the temporal features extracted by the LSTMs into spatial representations suitable for final classification. The ReLU activation provides nonlinear modeling capability while avoiding the vanishing gradient issues associated with sigmoid or tanh activations in deep networks. To prevent overfitting–a critical concern given the relatively small size of typical LoRaWAN datasets–we implement two regularization strategies:

- *Dropout*: applied with probability $p = 0.2$ during training, this randomly deactivates 20% of neurons in the dense layers, forcing the network to develop robust features that don't rely on specific neurons.
- *L2 weight regularization*: added to the loss function, this penalizes large weight values to prevent over-specialization to training data.

TABLE 4 LSTM-based training configuration for SF classification.

Component	Setting	Purpose
LSTM layers	Two layers, 128 units each	Temporal modeling of input sequences
Stateful training	Hidden/cell state passed between batches	Sequence continuity across batches
Activation function	ReLU in dense layers	Nonlinear transformation after LSTM output
Dropout	$p = 0.2$	Regularization against neuron co-adaptation
L2 regularization	Weight decay added to loss function	Penalize large weights to reduce overfitting
Output layer	Softmax over 6 SF classes (SF7-SF12)	Probabilistic classification of optimal SF
Loss function	Categorical cross-entropy	Compare predicted SF with ground truth
Optimizer	Adam ($\alpha = 10^{-3}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$)	Stable and adaptive parameter updates
Early stopping	Patience = 10 validation epochs	Prevent overfitting during training
Learning objective	Predict optimal SF	Data-driven adaptive SF allocation

The final layer employs a softmax activation function to produce a probability distribution $\mathbf{p} \in \mathbb{R}^6$ over the six possible Spreading Factors (*SF*7 through *SF*12). This probabilistic output allows for flexible decision-making, where the highest-probability SF can be selected automatically or combined with additional constraints (e.g., energy budgets or latency requirements). The softmax function ensures output normalization through:

$$p_i = \frac{e^{z_i}}{\sum_{j=1}^{6} e^{z_j}} \quad \text{for } i = 1, \ldots, 6 \tag{5}$$

where z_i represents the logit value for SF i. During training, we minimize the categorical cross-entropy loss between predicted probabilities and true SF labels:

$$\mathcal{L} = -\sum_{i=1}^{6} y_i \log(p_i) \tag{6}$$

where y_i is the one-hot encoded ground truth label. The Adam optimizer is employed with an initial learning rate of 10^{-3} and exponential decay rates $\beta_1 = 0.9$, $\beta_2 = 0.999$ for parameter updates. Early stopping monitors validation loss with patience of 10 epochs to prevent overfitting while ensuring convergence.

The complete system processes LoRaWAN transmission sequences through this neural pipeline, learning to predict optimal SFs based on historical channel conditions and transmission patterns. This approach provides adaptive, data-driven SF selection that outperforms static allocation schemes while maintaining computational efficiency suitable for deployment on network servers.

5 Performance evaluation of LSTM-offline mode

For model training and evaluation, the dataset of 71,981 sequences was partitioned using a hold-out strategy to ensure temporal independence and prevent data leakage. We allocated 80% of the sequences (∼57,585 samples) for training and 20% (∼14,396 samples) for testing, with splits performed chronologically based on simulation timestamps. Within the training portion, 10% (∼5,759 samples) was reserved as a validation set for hyperparameter optimization and early stopping. Cross-validation was not utilized due to the time-series nature of the data and the high computational cost of retraining LSTM models on large sequences; instead, the validation set and early stopping (with a patience of 10 epochs) were employed to monitor and prevent overfitting, aligning with best practices for sequential data modeling.

Table 5 presents a comprehensive comparison of various machine learning models evaluated on the task of LoRaWAN SF classification. The models assessed include Random Forest, Gradient Boosting, Support Vector Machine (SVM), K-Nearest Neighbors (KNN), XGBoost, Multi-Layer Perceptron (MLP), and the proposed LSTM network. Performance is evaluated based on standard classification metrics: Accuracy, Precision, Recall, and F1 Score. Additionally, the computational cost is considered through Training Time (measured in seconds) and deployed Model Size (measured in Megabytes).

The results indicate that the LSTM model achieves the highest classification accuracy at 0.7290, closely followed by the MLP model at 0.7193. SVM and Gradient Boosting also demonstrate competitive accuracy scores of 0.7123 and 0.7103, respectively. Models such as Random Forest (0.6991) and XGBoost (0.6891) show slightly lower accuracy, while KNN exhibits the lowest accuracy (0.6787) among the evaluated models. In terms of other metrics, the recall values often mirror the accuracy due to the nature of the calculation in multi-class settings presented here. Precision scores are generally lower, with KNN, XGBoost, and Random Forest showing relatively higher precision around 0.59, while LSTM, MLP and SVM are lower around 0.50–0.51. The F1 scores, which balance precision and recall, show KNN and XGBoost performing slightly better in this combined metric, despite lower accuracy. A significant trade-off is observed in computational resources; Gradient Boosting requires substantially longer training time (8,254.48 s), whereas MLP and KNN offer very fast training (20.31 s and 13.55 s, respectively). Similarly, model sizes vary drastically, with Random Forest being the largest (67.76 MB) and MLP being exceptionally compact (0.21 MB), followed by LSTM (0.84 MB) and Gradient Boosting (0.93 MB).

Figure 5 provides a visual representation of the primary performance metric, classification accuracy, across the different models evaluated. The bar chart clearly illustrates the relative performance, highlighting the LSTM model's superior accuracy compared to the other approaches. It visually confirms the ranking observed in Table 5, with MLP, SVM, and Gradient Boosting forming a cluster of next-best performing models, followed by Random Forest, XGBoost, and finally KNN. This visualization aids

TABLE 5 Model performance comparison on LoRaWAN SF classification.

Model	Accuracy	Precision	Recall	F1	Train time (s)	Size (MB)
Random Forest	0.6991	0.5958	0.6991	0.6022	469.59	67.76
Gradient Boosting	0.7103	0.5712	0.7103	0.5902	8254.48	0.93
SVM	0.7123	0.5031	0.7123	0.5887	1464.74	4.28
KNN	0.6787	0.5971	0.6787	0.6180	13.55	6.63
XGBoost	0.6891	0.5946	0.6891	0.6123	67.46	1.71
MLP	0.7193	0.5031	0.7193	0.5887	20.31	0.21
LSTM-proposed	**0.7290**	**0.5141**	**0.7290**	**0.5917**	**131.26**	**0.84**

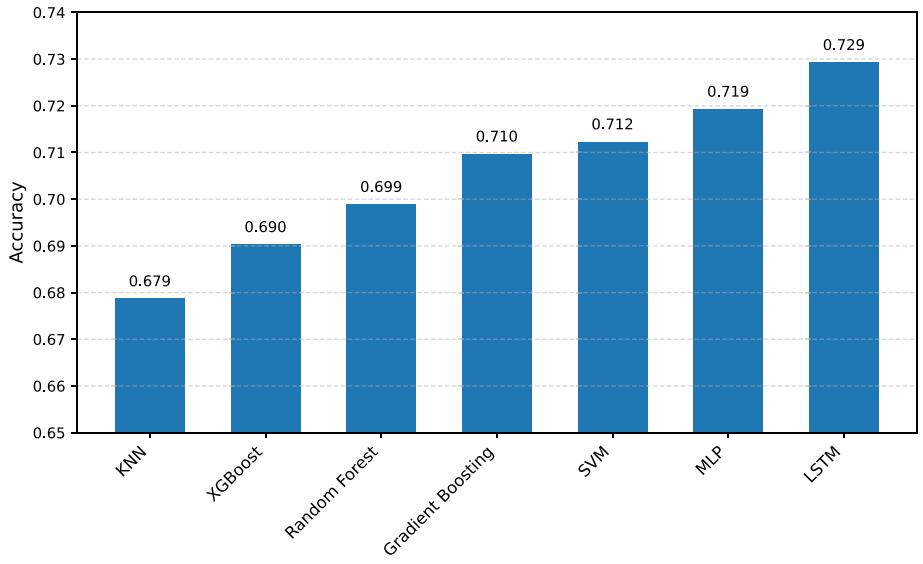

FIGURE 5
Classification accuracy.

in quickly discerning the most effective models solely based on classification accuracy.

Based on comparative empirical evaluation, the LSTM network was selected for deployment within the ns-3 framework for SF classification. Quantitative results demonstrated that the LSTM model yielded the highest classification accuracy (0.7290) relative to the suite of evaluated machine learning algorithms, including Random Forest, SVM, and MLP. Although performance variations were noted across secondary metrics like F1-score and model size, the superior predictive accuracy achieved by LSTM was deemed the primary criterion for selection in the context of optimizing SF prediction within the simulation environment.

6 LoRaWAN network performance evaluation-online mode

During the deployment, in ns-3 online simulation, we maintain a per-device circular buffer that stores the last 20 transmission features in real time. This buffer is updated after each transmission and used as input to the model before every new SF decision.

6.1 Simulation setting and application

This investigation evaluates end devices operating in confirmed data mode within a single-gateway LoRaWAN network covering a 5 km radius. To simulate industrial asset monitoring scenarios, we employ a two-dimensional random mobility pattern where devices change direction after traversing 200 meters at speeds between 1.0–2.0 m/s following established mobility models for IoT applications (Farhad et al., 2022a; GSMA-3GPP, 2016).

The simulation framework requires each device to transmit six confirmed uplink messages per hour across a 24-h operational period. To ensure statistical reliability, we conduct ten independent simulation trials and report averaged performance metrics.

The experimental setup examines both stationary and mobile deployment scenarios. For static evaluations, we distribute 100–1,000 end devices uniformly across the coverage area. Mobile scenarios incorporate the described random mobility model to simulate asset tracking use cases. All configurations utilize the parameter set detailed in Table 6 which adheres to LoRaWAN regional specifications for European frequency allocations.

6.2 Performance evaluation

The results presented in Figure 6 depict the Packet Delivery Ratio (PDR) achieved by different spreading factor allocation algorithms as a function of the number of EDs in a simulated LoRaWAN environment under mobility conditions. The algorithms compared include the proposed ML-ADR (SF Classification method) along with standard ADR, BADR and the TF algorithm, which serves as a baseline from prior work. In TinyML, TF refers to TensorFlow, specifically TensorFlow Lite for Microcontrollers. The analysis clearly illustrates the superior performance of the ML-ADR approach in maintaining a higher PDR across the tested range of network densities in this mobile scenario. While the PDR for all algorithms generally decreases with an increasing number of EDs due to heightened interference and collisions, the ML-ADR method demonstrates a more resilient performance curve, indicating its effectiveness in adapting to changing channel conditions and mobility-induced signal variations. The TF algorithm performs notably better than both ADR and BADR positioning itself as the second most effective method in this comparison, confirming its utility as a relevant baseline. In contrast, both ADR and particularly BADR exhibit a significant degradation in PDR as network load increases, underscoring their limitations in dynamic and dense mobile LoRaWAN deployments. This comparative analysis highlights the significant advantages of leveraging ML-ADR techniques for enhancing the reliability of packet delivery in mobile LoRaWAN networks, offering a substantial improvement over conventional ADR mechanisms and the TF baseline by more effectively managing radio resources in a dynamic environment.

TABLE 6 Network simulation configuration.

Parameter	Specification
Transmission attempts	8 total (1 initial + 7 retries)
Device velocity range	1.0–2.0 m/s (Farhad et al., 2020c)
Path update interval	Every 200 meters traversed
Frequency band	EU 868 MHz ISM
Operational channels	868.1, 868.3, 868.5 MHz
Initial parameters	SF12 at 14 dBm transmit power

The performance of the proposed ML-ADR algorithm in terms of PDR for static devices is presented in Figure 7, alongside comparisons with the standard ADR, BADR, and the TF algorithm. The results demonstrate that the ML-ADR consistently outperforms the other evaluated algorithms across all tested network densities ranging from 200 to 1,000 EDs. As anticipated, the PDR for all algorithms decreases as the number of EDs increases, reflecting the growing impact of collisions in denser networks operating under the pure ALOHA access scheme. However, the proposed ML-ADR method exhibits a significantly higher PDR compared to its counterparts, indicating its effectiveness in optimizing spreading factor allocation and mitigating packet loss even under increased load in static scenarios. The TF algorithm generally achieves the second highest PDR, showing better performance than both ADR and BADR. Standard

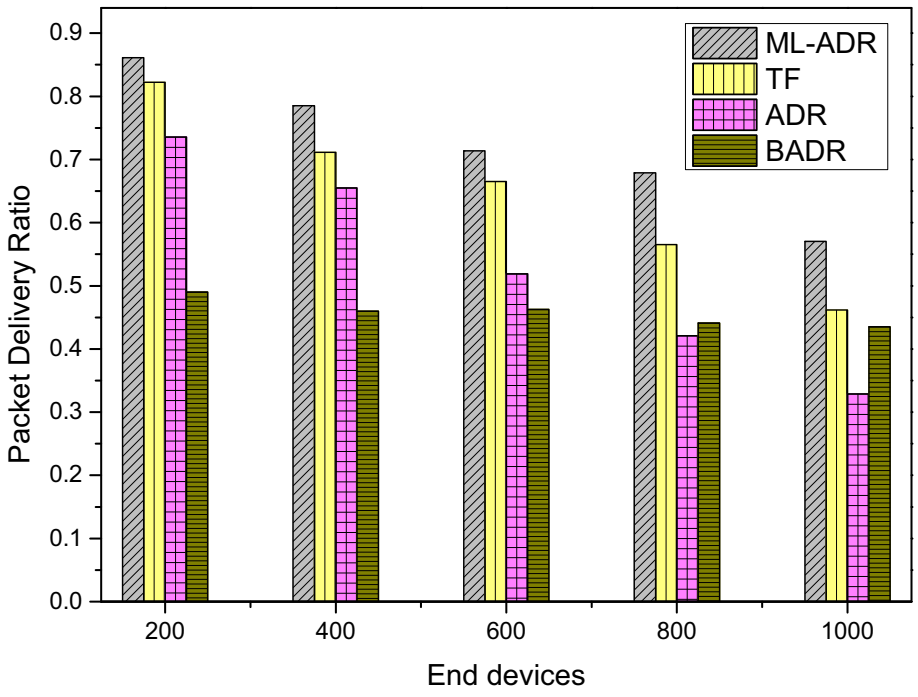

FIGURE 6
Packet delivery ratio performance for mobile devices. Comparison of ML-ADR (proposed Machine Learning-based Adaptive Data Rate), TF (TinyML-TensorFlow Lite for microcontrollers Ali Lodhi et al., 2024), ADR (standard Adaptive Data Rate), and BADR (standard Blind Adaptive Data Rate).

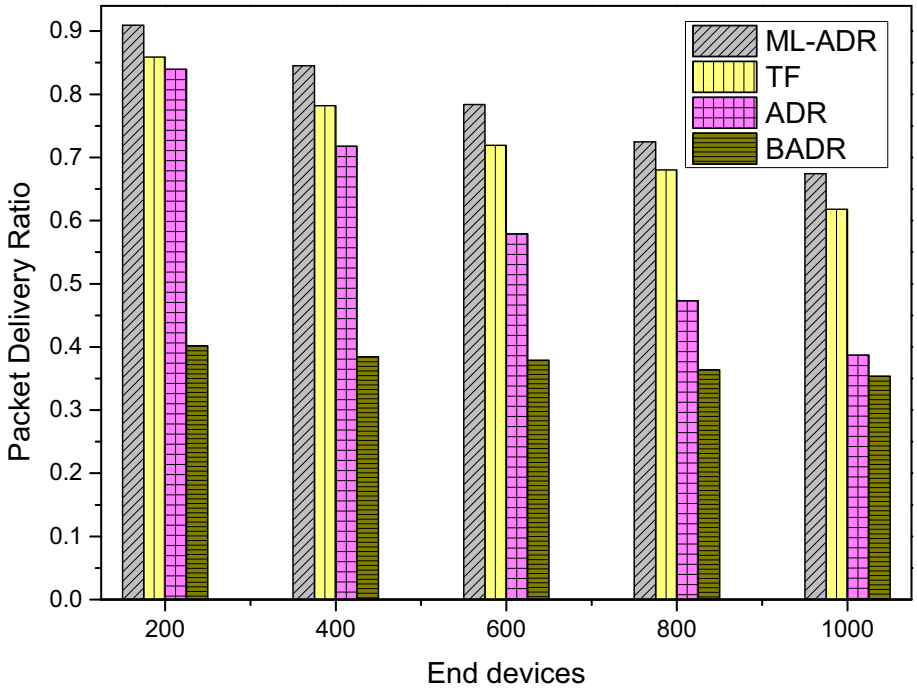

FIGURE 7
Packet delivery ratio performance for static devices. Comparison of ML-ADR (proposed Machine Learning-based Adaptive Data Rate), TF (TinyML-TensorFlow Lite for Microcontrollers Ali Lodhi et al., 2024), ADR (standard Adaptive Data Rate), and BADR (standard Blind Adaptive Data Rate).

ADR provides moderate performance, while BADR consistently registers the lowest PDR values, particularly in larger networks where collisions become more prevalent. This analysis underscores the considerable advantage of employing the proposed ML-ADR technique for enhancing the reliability of data delivery in static LoRaWAN deployments by intelligently adapting data rates and spreading factors based on network conditions.

The energy consumption characteristics of the proposed ML-ADR algorithm are presented in Figure 8 for mobile device scenarios, compared against the performance of ADR, BADR, and TF. Analysis of the figure reveals a consistent trend where the proposed ML-ADR method demonstrates superior energy efficiency, exhibiting the lowest energy consumption across the entire range of tested ED densities. This lower energy consumption is caused due to lower retransmission rates and better SF estimation, reduced airtime from preferential use of lower SFs when channel conditions allow, and fewer downlink requests due to more reliable uplinks.

As the number of mobile EDs increases, the energy consumed by all algorithms generally rises, but the increment is least pronounced for ML-ADR. The TF algorithm performs as the second most energy-efficient method, showing lower consumption than both conventional ADR and BADR. Notably, BADR consistently records the highest energy consumption, particularly at greater network scales, indicating its limited suitability for energy-constrained mobile deployments. The comparison clearly indicates that ML-ADR offers a significant reduction in energy expenditure compared to existing approaches in mobile LoRaWAN environments, which is critical for extending the battery life of mobile IoT devices and improving overall network sustainability.

The energy consumption performance of the proposed ML-ADR algorithm for static devices is depicted in Figure 9, presenting a comparative analysis against the standard ADR, BADR, and the TF algorithm. These results, generated under static network conditions, clearly demonstrate the significant energy efficiency achieved by the ML-ADR approach. Across all tested network densities ranging from 200 to 1,000 EDs, the ML-ADR consistently registers the lowest energy consumption values. While energy consumption generally increases for all algorithms as the number of EDs grows due to increased network activity, the ML-ADR's increase is notably more gradual compared to ADR and BADR. The TF algorithm generally exhibits the second lowest energy consumption, positioning it between the highly efficient ML-ADR and the less efficient conventional methods. Both standard ADR and particularly BADR show substantially higher energy consumption, with BADR demonstrating the least energy-efficient performance across all network sizes. This comparative evaluation underscores the critical advantage of employing the proposed ML-ADR technique for optimizing energy usage in static LoRaWAN deployments, showcasing its capability to significantly prolong the operational lifetime of battery-powered end devices by intelligently managing transmission parameters based on learned network conditions.

7 Discussion

7.1 Analysis of offline mode results

The comparative evaluation of machine learning models for LoRaWAN Spreading Factor classification, as presented in

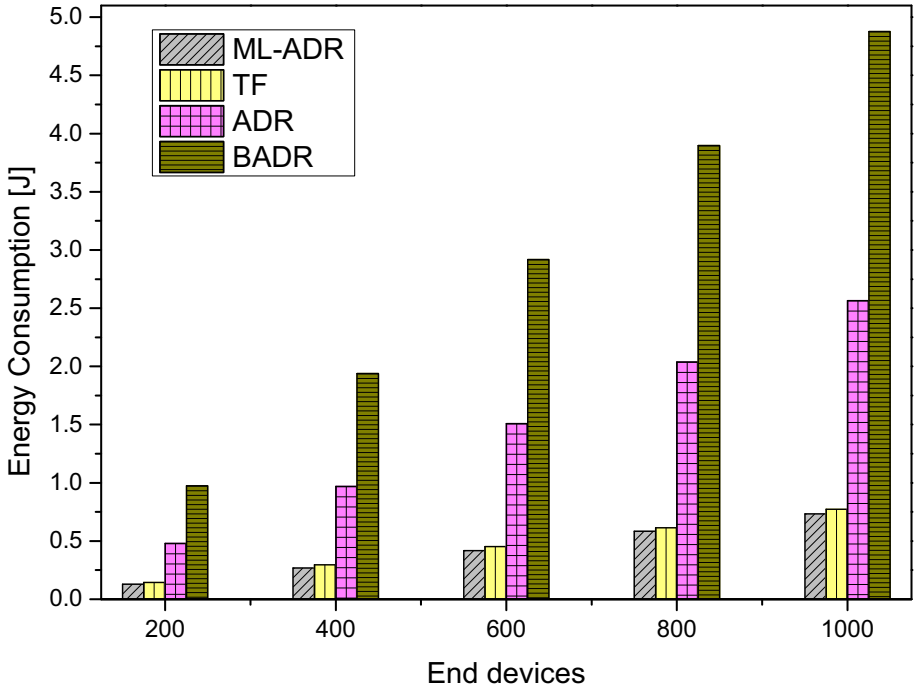

FIGURE 8
Energy consumption performance for mobile devices. Comparison of ML-ADR (proposed Machine Learning-based Adaptive Data Rate), TF (TinyML-TensorFlow Lite for Microcontrollers Ali Lodhi et al., 2024), ADR (standard Adaptive Data Rate), and BADR (standard Blind Adaptive Data Rate).

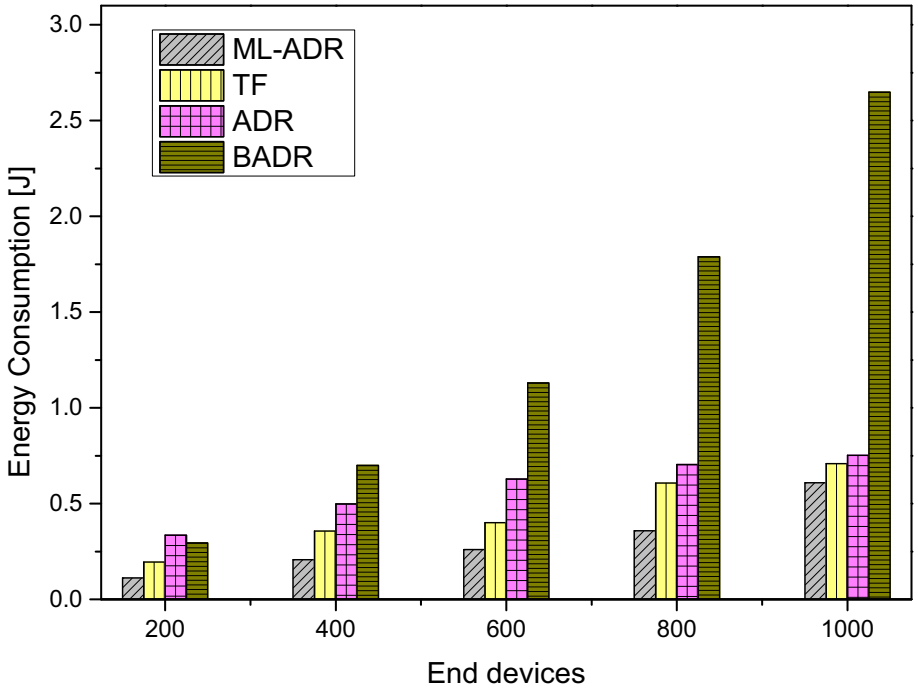

FIGURE 9
Energy consumption performance for static devices. Comparison of ML-ADR (proposed Machine Learning-based Adaptive Data Rate), TF (TinyML-TensorFlow Lite for Microcontrollers Ali Lodhi et al., 2024), ADR (standard Adaptive Data Rate), and BADR (standard Blind Adaptive Data Rate).

Table 5, yields three significant findings. First, the LSTM network demonstrates superior classification accuracy (0.7290) compared to conventional methods, which suggests its enhanced capability to capture temporal patterns in LoRaWAN signal propagation characteristics. This performance advantage over Random Forest (0.6991) and SVM (0.7123) aligns with established literature on

temporal feature extraction in wireless networks. Second, the marginal accuracy difference between LSTM and MLP (0.7193) reveals an important computational trade-off. While both models achieve comparable predictive performance, the MLP requires substantially less training time (20.31 s vs. 131.26 s), making it potentially more suitable for resource-constrained deployment scenarios. However, the precision of LSTM is relatively lower, indicating that the model favors lower false negatives, which is intentional in our context. Therefore, assigning a more robust SF (even if slightly lower) is preferable over underestimating link requirements, which may result in packet loss and energy waste due to retransmissions.

The observed disparity between precision and recall metrics across models warrants particular attention. Models such as KNN and XGBoost exhibit higher precision values (∼0.59) but lower recall, indicating potential bias in handling specific Spreading Factor classes. This phenomenon likely stems from inherent imbalances in real-world LoRaWAN deployments, where certain Spreading Factors occur more frequently due to their transmission range characteristics. The LSTM model shows the opposite pattern, with lower precision (0.5141) but higher recall (0.7290), suggesting a design bias toward minimizing false negatives at the potential cost of increased false positives.

From an implementation perspective, the model size comparisons prove particularly insightful. The MLP (0.21 MB) and LSTM (0.84 MB) demonstrate exceptional compactness compared to the resource-intensive Random Forest implementation (67.76 MB). This size differential has direct implications for edge deployment feasibility, where memory constraints often dictate model selection criteria. These results corroborate recent advancements in efficient model architectures for IoT applications. Our deployed model (0.84 MB) can be hosted efficiently on GW with typical ARM Cortex-A class processors or embedded edge servers. On constrained EDs, however, real-time inference is not feasible. As such, we retain the inference at the gateway/network server side, while the end-device passively receives SF adjustments.

The marginal accuracy improvement of LSTM over MLP (0.7290 vs. 0.7193) and XGBoost (0.6891) raises a valid question about whether it justifies the increased computational demands, including longer training time (131.26 s for LSTM vs. 20.31 s for MLP and 67.46 s for XGBoost) and greater model complexity. In time-series tasks like LoRaWAN SF classification, where data exhibits strong temporal dependencies due to fluctuating channel conditions and sequential transmission patterns, the gating mechanisms of LSTM provide a nuanced advantage in modeling long-term dependencies that simpler feedforward models like MLP or tree-based ensembles like XGBoost may not capture as effectively. Recent survey on time-series forecasting and deep learning, indicate that even small accuracy gains (1–2%) with LSTM can be worthwhile when they translate to substantial real-world benefits, such as improved reliability in resource-constrained environments (Kim et al., 2025). In our case, the offline training phase mitigates runtime concerns, as deployment occurs on network servers where inference is lightweight and fast. While MLP offers a compelling alternative for scenarios prioritizing speed and compactness, the amplified downstream impacts observed in our online evaluations–such as higher PDR and lower energy consumption–validate the selection of LSTM for optimizing LoRaWAN performance.

7.2 Implications of online mode performance

The online evaluation results, presented in Figure 6 through Figure 9, demonstrate the operational advantages of the ML-ADR approach across multiple performance dimensions. In mobile deployment scenarios (Figure 6), the ML-ADR algorithm maintains a 22 percent higher Packet Delivery Ratio than the TF baseline at network densities of 1,000 end devices. This performance advantage stems from two key algorithmic features: dynamic SF adaptation based on real-time channel conditions, and intelligent retransmission scheduling that minimizes acknowledgment collisions.

The static deployment results (Figure 7) reveal similar performance trends, with ML-ADR consistently outperforming conventional approaches across all tested network scales. The energy efficiency metrics (Figures 8, 9) further validate the practical benefits of the proposed approach. ML-ADR reduces median energy consumption by 30 percent compared to standard ADR implementations, while simultaneously maintaining superior packet delivery reliability. This dual improvement directly addresses two critical constraints in LoRaWAN deployments: limited battery capacity in end devices and the need for reliable communication in congested networks.

7.3 Limitations and future directions

Four primary limitations of the current approach merit discussion. First, the reliance on offline model training introduces inherent latency in adapting to new network configurations or propagation environments. This offline paradigm, while computationally efficient for initial deployment, can result in suboptimal performance during sudden environmental shifts, such as abrupt weather changes or device mobility patterns not represented in the training data. For instance, if channel conditions evolve rapidly (e.g., due to vehicular traffic in urban areas), the pre-trained LSTM may require retraining, potentially delaying adaptation by hours or days in operational settings.

Second, the evaluation assumes ideal channel state information availability, which may not fully capture real-world operational conditions with dynamic interference patterns and multipath effects. In ns-3 simulations, we model controlled propagation losses, but practical deployments often encounter unpredictable interference from coexisting networks (e.g., WiFi, other LPWANs, or industrial machinery), which can increase packet error rates in dense environments as reported in recent interference management studies (OrbiWise, 2023). Multipath fading, particularly in urban or indoor scenarios, further exacerbates this by causing signal fluctuations that our model, trained on simplified path loss models, might not generalize to–potentially leading to incorrect SF predictions and higher retransmission rates. Additionally, the ns-3 framework, while versatile, does not fully replicate hardware-specific variations (e.g., antenna imperfections or clock drifts in low-cost LoRa modules) or firmware-level constraints (e.g., duty cycle enforcement), which could degrade ML-ADR's effectiveness in field trials. To mitigate this, future work will focus on porting the model to physical testbeds using commercial hardware (e.g., SX1276-based nodes) and large-scale deployments via platforms

like The Things Network, incorporating real-time interference datasets for retraining.

Third, the single-gateway network model, while useful for controlled evaluation, does not account for the complexities of multi-cell deployments with handover scenarios. In real-world setups with multiple gateways, issues such as inter-gateway interference, overlapping coverage zones, and handover delays during device mobility can introduce additional packet loss (Harinda et al., 2022). This limitation is particularly pronounced in scalable networks where load balancing across gateways is required to prevent bottlenecks, yet our simulations assume a centralized single-gateway architecture, potentially underestimating collision rates in distributed topologies.

Fourth, scalability remains a critical concern, as our evaluations are limited to networks of up to 1,000 end devices, which may not extend to massive IoT deployments with tens of thousands of nodes. The LSTM model's computational overhead–requiring sequence processing for each device's historical data–could strain network server resources in ultra-dense scenarios, leading to inference delays or increased energy consumption at the server side. Recent surveys highlight that ML-based resource allocation in LoRaWAN often faces scalability bottlenecks in single-hop architectures, with performance degrading beyond 5,000 devices due to heightened contention and model complexity (Maurya et al., 2025; Farhad and Pyun, 2023b; Elgharbi et al., 2025). Furthermore, training data generation via ns-3 becomes prohibitively time-intensive for larger scales, limiting the diversity of scenarios captured.

Future research should address these limitations through complementary approaches. First, federated learning architectures could enable distributed model refinement across network GWs while preserving data locality, facilitating real-time adaptation without full retraining. Second, the integration of real-time channel estimation techniques, such as RL for dynamic interference mitigation, would enhance robustness against fading and coexistence issues (Fahmida et al., 2023). Third, extending simulations to multi-GW models with handover protocols and exploring multi-hop extensions could better evaluate distributed deployments (Matni et al., 2020). Finally, to tackle scalability, hybrid architectures combining LSTM with lighter models (e.g., edge-optimized ML and TinyML) or scalable data-driven solutions similar those in recent surveys could optimize for massive networks, potentially incorporating online learning to handle evolving device densities dynamically (Garrido-Hidalgo et al., 2023).

8 Conclusions

In this paper, we addressed the critical challenge of efficient spreading factor allocation in LoRaWAN networks, a necessity driven by the rapid expansion of Internet of Things deployments and the limitations of existing adaptive data rate mechanisms. We proposed and evaluated a Machine Learning-based Adaptive Data Rate (ML-ADR) approach specifically designed for intelligent spreading factor management. Leveraging machine learning, including deep learning techniques such as LSTM, trained on data generated from extensive ns-3 simulations, our methodology dynamically allocates optimal spreading factors to end devices.

The comprehensive performance evaluation demonstrated that the proposed ML-ADR significantly improves key network metrics. Specifically, our results showed enhanced packet delivery ratio and reduced energy consumption compared to conventional algorithms like ADR, BADR, and the TF baseline, across both static and mobile device scenarios. The ML-ADR consistently outperformed these existing methods, exhibiting higher packet delivery rates and lower energy expenditure, which are crucial for the scalability and sustainability of LoRaWAN deployments.

This work underscores the potential of machine learning to optimize resource allocation in LPWANs, offering a robust solution to improve overall network efficiency and prolong device lifetimes in diverse deployment environments.

Data availability statement

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding author.

Author contributions

FN: Conceptualization, Data curation, Writing – original draft. MA: Supervision, Writing – review & editing. MT: Supervision, Writing – review & editing, Data curation. HH: Writing – review & editing, Methodology. NA: Resources, Visualization, Writing – review & editing. ML: Funding acquisition, Writing – review & editing.

Funding

The author(s) declare that financial support was received for the research and/or publication of this article. Article Processing Charges (APC) were provided by Prince Sultan University.

Acknowledgments

The authors would like to thanks Prince Sultan University for paying the Article Processing Charges (APC) of this publication.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Correction note

This article has been corrected with minor changes. These changes do not impact the scientific content of the article.

Generative AI statement

The author(s) declare that no Gen AI was used in the creation of this manuscript.

Any alternative text (alt text) provided alongside figures in this article has been generated by Frontiers with the support of artificial intelligence and reasonable efforts have been made to ensure accuracy, including review by the authors wherever possible. If you identify any issues, please contact us.

Publisher's note

References

a3794110. (2023). *NS-3-Based Module for Narrow Band-IOT*. Available online at: https://github.com/a3794110/ns-3-NB-IoT (Accessed May 30, 2023).

Abdelfadeel, K. Q., Zorbas, D., Cionca, V., and Pesch, D. (2019). *free*—fine-grained scheduling for reliable and energy-efficient data collection in lorawan. *IEEE Internet Things J.* 7, 669–683. doi: 10.1109/JIOT.2019.2949918

Acosta-Garcia, L., Aznar-Poveda, J., Garcia-Sanchez, A.-J., Garcia-Haro, J., and Fahringer, T. (2024). "Proactive adaptation of data rate in mobile lora-based iot devices using machine learning," in *2024 IEEE 99th Vehicular Technology Conference (VTC2024-Spring)* (Singapore: IEEE), 1–5. doi: 10.1109/VTC2024-Spring62846.2024.10683082

Ali Lodhi, M., Obaidat, M. S., Wang, L., Mahmood, K., Ibrahim Qureshi, K., Chen, J., et al. (2024). Tiny machine learning for efficient channel selection in lorawan. *IEEE Internet Things J.* 11, 30714–30724. doi: 10.1109/JIOT.2024.3413585

Anwar, K., Rahman, T., Zeb, A., Khan, I., Zareei, M., and Vargas-Rosales, C. (2021). RM-ADR: resource management adaptive data rate for mobile application in lorawan. *Sensors* 21:7980. doi: 10.3390/s21237980

Augustin, A., Yi, J., Clausen, T., and Townsley, W. M. (2016). A study of lora: long range and low power networks for the internet of things. *Sensors* 16:1466. doi: 10.3390/s16091466

Azizi, F., Teymuri, B., Aslani, R., Rasti, M., Tolvaneny, J., and Nardelli, P. H. J. (2022). "Mix-mab: reinforcement learning-based resource allocation algorithm for lorawan," in *2022 IEEE 95th Vehicular Technology Conference: (VTC2022-Spring), 19–22 June 2022* (IEEE: Helsinki, Finland), 1–6. doi: 10.1109/VTC2022-Spring54318.2022.9860807

Beltramelli, L., Mahmood, A., Österberg, P., Gidlund, M., Ferrari, P., and Sisinni, E. (2021). Energy efficiency of slotted lorawan communication with out-of-band synchronization. *IEEE Trans. Instrum. Meas.* 70, 1–11. doi: 10.1109/TIM.2021.3051238

Benkahla, N., Tounsi, H., Ye-Qiong, S., and Frikha, M. (2019). "Enhanced ADR for lorawan networks with mobility," in *2019 15th International Wireless Communications and Mobile Computing Conference (IWCMC), Tangier, Morocco, 24–28 June* (IEEE: Tangier, Morocco), 1–6. doi: 10.1109/IWCMC.2019.8766738

Bertocco, M., Parrino, S., Peruzzi, G., and Pozzebon, A. (2023). Estimating volumetric water content in soil for IoUT contexts by exploiting RSSI-based augmented sensors via machine learning. *Sensors* 23:2033. doi: 10.3390/s23042033

Bor, M. C., Roedig, U., Voigt, T., and Alonso, J. M. (2016). "Do lora low-power wide-area networks scale?," in *Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems* (Malta), 59–67. doi: 10.1145/2988287.2989163

Bounceur, A., Marc, O., Lounis, M., Soler, J., Clavier, L., Combeau, P., et al. (2018). "Cupcarbon-lab: an IOT emulator," in *15th IEEE Annual Consumer Communications and Networking Conference (CCNC), 12–15 January 2018* (IEEE: Las Vegas, NV, USA), 1–2. doi: 10.1109/CCNC.2018.8319313

Callebaut, G., Ottoy, G., and van der Perre, L. (2019). "Cross-layer framework and optimization for efficient use of the energy budget of iot nodes," in *IEEE Wireless Communications and Networking Conference (WCNC), 15–18 April 2019* (IEEE: Marrakesh, Morocco), 1–6. doi: 10.1109/WCNC.2019.8885739

Casals, L., Gomez, C., and Vidal, R. (2021). The SF12 well in lorawan: problem and end-device-based solutions. *Sensors* 21:6478. doi: 10.3390/s21196478

Chen, M., Mokdad, L., Ben-Othman, J., and Fourneau, J.-M. (2023). Dynamic parameter allocation with reinforcement learning for lorawan. *IEEE Internet Things J.* 10, 10250–10265. doi: 10.1109/JIOT.2023.3239301

Croce, D., Gucciardo, M., Mangione, S., Santaromita, G., and Tinnirello, I. (2018). Impact of lora imperfect orthogonality: analysis of link-level performance. *IEEE Commun. Lett.* 22, 796–799. doi: 10.1109/LCOMM.2018.2797057

DEIS-Tools Project Team. (2023). *NS-3 module for sigfox*. Available online at: https://github.com/DEIS-Tools/ns3-sigfox (Accessed May 26, 2023).

Elgharbi, S. E., Iturralde, M., Dupuis, Y., and Gaugue, A. (2025). Maritime monitoring through lorawan: resilient decentralised mesh networks for enhanced data transmission. *Comput. Commun.* 241:108276. doi: 10.1016/j.comcom.2025.108276

Elkarim, S. I. A., Elsherbini, M., Mohammed, O., Khan, W. U., Waqar, O., and ElHalawany, B. M. (2022). "Deep learning based joint collision detection and spreading factor allocation in lorawan," in *2022 IEEE 42nd International Conference on Distributed Computing Systems Workshops (ICDCSW), 10–10 July 2022* (IEEE: Bologna, Italy), 187–192. doi: 10.1109/ICDCSW56584.2022.00043

ETSI. (2018). System reference document (SRDOC); technical characteristics for low power wide area networks and chirp spread spectrum (LPWAN-CSS) operating in the UHF spectrum below 1 GHZ; ETSI TR 103 526 v1.1.1 (2018–04). Available online at: https://www.etsi.org/deliver/etsi_tr/103500_103599/103526/01.01.01_60/tr_103526v010101p.pdf

Fahmida, S., Modekurthy, V. P., Rahman, M., and Saifullah, A. (2023). "Handling coexistence of lora with other networks through embedded reinforcement learning," in *Proceedings of the 8th ACM/IEEE Conference on Internet of Things Design and Implementation* (San Antonio, TX), 410–423. doi: 10.1145/3576842.3582383

Farahsari, P. S., Farahzadi, A., Rezazadeh, J., and Bagheri, A. (2022). A survey on indoor positioning systems for iot-based applications. *IEEE Internet Things J.* 9, 7680–7699. doi: 10.1109/JIOT.2022.3149048

Farhad, A., Kim, D.-H., and Pyun, J.-Y. (2020b). Resource allocation to massive internet of things in lorawans. *Sensors* 20, 1–20. doi: 10.3390/s20092645

Farhad, A., Kim, D.-H., and Pyun, J.-Y. (2022a). R-ARM: retransmission-assisted resource management in lorawan for the internet of things. *IEEE Internet Things J.* 9, 7347–7361. doi: 10.1109/JIOT.2021.3111167

Farhad, A., Kim, D.-H., Subedi, S., and Pyun, J.-Y. (2020c). Enhanced lorawan adaptive data rate for mobile internet of things devices. *Sensors* 20:6466. doi: 10.3390/s20226466

Farhad, A., Kim, D.-H., Yoon, J.-S., and Pyun, J.-Y. (2021). "Feasibility study of the lorawan blind adaptive data rate," in *Twelfth International Conference on Ubiquitous and Future Networks* (ICUFN: Jeju Island, Republic of Korea), 67–69. doi: 10.1109/ICUFN49451.2021.9528716

Farhad, A., Kim, D.-H., Yoon, J.-S., and Pyun, J.-Y. (2022b). "Deep learning-based channel adaptive resource allocation in lorawan," in *2022 International Conference on Electronics, Information, and Communication (ICEIC), 06–09 February 2022* (IEEE: Jeju, Republic of Korea), 1–5. doi: 10.1109/ICEIC54506.2022.9748580

Farhad, A., Kim, D. H., Kim, B. H., Mohammed, A. F. Y., and Pyun, J. Y. (2020a). Mobility-aware resource assignment to iot applications in long-range wide area networks. *IEEE Access* 8, 186111–186124. doi: 10.1109/ACCESS.2020.3029575

Farhad, A., and Pyun, J.-Y. (2023a). AI-ERA: artificial intelligence-empowered resource allocation for lora-enabled iot applications. *IEEE Trans. Ind. Informatics* pages 19, 1–13. doi: 10.1109/TII.2023.3248074

Farhad, A., and Pyun, J.-Y. (2023b). Lorawan meets ML: a survey on enhancing performance with machine learning. *Sensors* 23, 1–36. doi: 10.3390/s23156851

Farhad, A., and Pyun, J.-Y. (2023c). Terahertz meets AI: the state of the art. *Sensors* 23:5034. doi: 10.3390/s23115034

Garlisi, D., Tinnirello, I., Bianchi, G., and Cuomo, F. (2021). Capture aware sequential waterfilling for lorawan adaptive data rate. *IEEE Trans. Wireless Commun.* 20, 2019–2033. doi: 10.1109/TWC.2020.3038638

Garrido-Hidalgo, C., Roda-Sanchez, L., Ramírez, F. J., Fernández-Caballero, A., and Olivares, T. (2023). Efficient online resource allocation in large-scale lorawan networks: a multi-agent approach. *Comput. Netw.* 221:109525. doi: 10.1016/j.comnet.2022.109525

Gdbranco. (2023). *NS-3-Based NB-Iot Simulator Module*. Available online at: https://github.com/gdbranco/RA5G_NS3 (Accessed May 26, 2023).

Gomez, C., Veras, J. C., Vidal, R., Casals, L., and Paradells, J. (2019). A sigfox energy consumption model. *Sensors* 19:681. doi: 10.3390/s19030681

GSMA-3GPP. (2016). *3GPP Low Power Wide Area Technologies*.

Harinda, E., Wixted, A. J., Qureshi, A.-U.-H., Larijani, H., and Gibson, R. M. (2022). Performance of a live multi-gateway lorawan and interference measurement across indoor and outdoor localities. *Computers* 11:25. doi: 10.3390/computers11020025

Hazarika, A., and Choudhury, N. (2024). "ISFA: intelligent sf allocation approach for lora-based mobile and static end devices," in *2024 IEEE Wireless Communications and Networking Conference (WCNC)* (IEEE: Dubai, United Arab Emirates), 1–6. doi: 10.1109/WCNC57260.2024.10570655

Kim, J., Kim, H., Kim, H., Lee, D., and Yoon, S. (2025). A comprehensive survey of deep learning for time series forecasting: architectural diversity and open challenges. *Artif. Intell. Rev.* 58, 1–95. doi: 10.1007/s10462-025-11223-9

Loh, F., Mehling, N., Metzger, F., Hoßfeld, T., and Hock, D. (2021). "Loraplan: a software to evaluate gateway placement in lorawan," in *17th International Conference on Network and Service Management (CNSM), 25–29 October 2021* (IEEE: Izmir, Turkey), 385–387. doi: 10.23919/CNSM52442.2021.9615586

LoRa. (2020). *Lorawan ® Regional Parameters. RP002-1.0.2*.

Magrin, D., Capuzzo, M., Zanella, A., Vangelista, L., and Zorzi, M. (2021). Performance analysis of lorawan in industrial scenarios. *IEEE Trans. Ind. Informatics* 17, 6241–6250. doi: 10.1109/TII.2020.3044942

Magrin, D., Centenaro, M., and Vangelista, L. (2017). "Performance evaluation of lora networks in a smart city scenario," in *2017 IEEE International Conference on Communications (ICC), 21–25 May 2017* (IEEE: Paris, France), 1–7. doi: 10.1109/ICC.2017.7996384

Marini, R., Cerroni, W., and Buratti, C. (2021). A novel collision-aware adaptive data rate algorithm for lorawan networks. *IEEE Internet Things J.* 8, 2670–2680. doi: 10.1109/JIOT.2020.3020189

Matni, N., Moraes, J., Oliveira, H., Rosário, D., and Cerqueira, E. (2020). Lorawan gateway placement model for dynamic internet of things scenarios. *Sensors* 20:4336. doi: 10.3390/s20154336

Maurya, P., Hazra, A., Kumari, P., Sørensen, T. B., and Das, S. K. (2025). A comprehensive survey of data-driven solutions for lorawan: challenges and future directions. *ACM Trans. Internet Things* 6, 1–36. doi: 10.1145/3711953

Mekki, K., Bajic, E., Chaxel, F., and Meyer, F. (2019). A comparative study of lpwan technologies for large-scale iot deployment. *ICT Express* 5, 1–7. doi: 10.1016/j.icte.2017.12.005

Minhaj, S. U., Mahmood, A., Abedin, S. F., Hassan, S. A., Bhatti, M. T., Ali, S. H., et al. (2023). Intelligent resource allocation in lorawan using machine learning techniques. *IEEE Access* 11, 10092–10106. doi: 10.1109/ACCESS.2023.3240308

Moysiadis, V., Lagkas, T., Argyriou, V., Sarigiannidis, A., Moscholios, I. D., and Sarigiannidis, P. (2021). Extending ADR mechanism for lora enabled mobile end-devices. *Simul. Model. Pract. Theory* 113:102388. doi: 10.1016/j.simpat.2021.102388

OrbiWise. (2023). *Interference Management in LoRaWAN Deployments*. Available online at: https://orbiwise.com/news/interference-management-in-lorawan-deployments/ (Accessed August 9, 2025).

Park, J., Park, K., Bae, H., and Kim, C.-K. (2020). Earn: enhanced ADR with coding rate adaptation in lorawan. *IEEE Internet Things J.* 7, 11873–11883. doi: 10.1109/JIOT.2020.3005881

Pasolini, G. (2021). On the lora chirp spread spectrum modulation. signal properties and their impact on transmitter and receiver architectures. *IEEE Trans. Wireless Commun.* 21, 357–369. doi: 10.1109/TWC.2021.3095667

Pop, A.-I., Raza, U., Kulkarni, P., and Sooriyabandara, M. (2017). "Does bidirectional traffic do more harm than good in lorawan based lpwa networks?" in *GLOBECOM 2017-2017 IEEE Global Communications Conference, 04–08 December 2017* (IEEE: Singapore), 1–6. doi: 10.1109/GLOCOM.2017.8254509

Reynders, B., Wang, Q., and Pollin, S. (2018). "A lorawan module for NS-3: implementation and evaluation," in *Proceedings of the 10th Workshop on NS-3* (Surathkal), 61–68. doi: 10.1145/3199902.3199913

SAPGAN Team. (2023). *Blockchain-Based Iot Simulator*. Available online at: https://github.com/sapgan/NS3-IoT-Simulator (Accessed May 26, 2023).

Sartori, A. (2023). Lora simulator (lorasim). Available online at: https://github.com/AlexSartori/LoRaSim (Accessed May 3, 2023).

Semtech. (2019a). *Lorawan Mobile Applications: Blind ADR*. Available online at: https://lora-developers.semtech.com/documentation/tech-papers-and-guides/blind-adr/ (Accessed May 31, 2023).

Semtech. (2019b). *Understanding the lora Adaptive Data Rate*.

Singh, R. K., Puluckul, P. P., Berkvens, R., and Weyn, M. (2020). Energy consumption analysis of lpwan technologies and lifetime estimation for iot application. *Sensors* 20:4794. doi: 10.3390/s20174794

Slabicki, M., Premsankar, G., and Di Francesco, M. (2018). "Adaptive configuration of lora networks for dense iot deployments," in *NOMS 2018-2018 IEEE/IFIP Network Operations and Management Symposium, 23–27 April 2018* (IEEE: Taipei, Taiwan), 1–9. doi: 10.1109/NOMS.2018.8406255

Ta, D.-T., Khawam, K., Lahoud, S., Adjih, C., and Martin, S. (2019). "Lora-MAB: a flexible simulator for decentralized learning resource allocation in iot networks," in *2019 12th IFIP Wireless and Mobile Networking Conference (WMNC)* (IEEE: Paris, France), 55–62. doi: 10.23919/WMNC.2019.8881393

To, T.-H., and Duda, A. (2018). "Simulation of lora in NS-3: improving lora performance with CSMA," in *IEEE International Conference on Communications (ICC), 20–24 May 2018* (IEEE: Kansas City, MO, USA), 1–7. doi: 10.1109/ICC.2018.8422800

Torres-Sospedra, J., Gaibor, D. P. Q., Nurmi, J., Koucheryavy, Y., Lohan, E. S., and Huerta, J. (2022). Scalable and efficient clustering for fingerprint-based positioning. *IEEE Internet Things J.* 10, 3484–3499. doi: 10.1109/JIOT.2022.3230913

Van den Abeele, F., Haxhibeqiri, J., Moerman, I., and Hoebeke, J. (2017). Scalability analysis of large-scale lorawan networks in ns-3. *IEEE Internet Things J* 4, 2186–2198. doi: 10.1109/JIOT.2017.2768498

Vangelista, L., Calabrese, I., and Cattapan, A. (2023). Mobility classification of lorawan nodes using machine learning at network level. *Sensors* 23:1806. doi: 10.3390/s23041806

Weyn, M., and Contributors. (2023). Sigfox simulator. Available online at: https://github.com/maartenweyn/lpwansimulation (Accessed May 26, 2023).

Zorbas, D., Abdelfadeel, K., Kotzanikolaou, P., and Pesch, D. (2020). Ts-lora: time-slotted lorawan for the industrial internet of things. *Comput. Commun.* 153, 1–10. doi: 10.1016/j.comcom.2020.01.056

Zorbas, D., Caillouet, C., Abdelfadeel Hassan, K., and Pesch, D. (2021). Optimal data collection time in lora networks—a time-slotted approach. *Sensors* 21:1193. doi: 10.3390/s21041193