2024 IEEE International Performance, Computing, and Communications Conference (IPCCC) | 979-8-3503-6794-2/24/$31.00 ©2024 IEEE | DOI: 10.1109/IPCCC59868.2024.10850023

Experimental Analysis of LoRaWAN for Optimizing Water Quality Monitoring with Reinforcement Learning-Driven Scheduling

Jui Mhatre[1], Minh Nguyen[2], Ahyoung Lee[1*], and Hoseon Lee[2]
[1]Department of Computer Science, Kennesaw State University, Marietta, GA 30060, USA.
[2]Department of Electrical and Computer Engineering, Kennesaw State University, Marietta, GA 30060, USA
Email: {[1]jmhatre1, [2]Hnguy130}@students.kennesaw.edu, {[1*]ahyoung.lee, [2]hoseon.lee}@kennesaw.edu.
([*]Corresponding Author)

***Abstract*—Long Range Wide Area Network (LoRaWAN) is a promising communication technology for environmental monitoring due to its low power consumption and long-range capabilities. Despite its advantages, several challenges are associated with LoRaWAN due to technical limitations, environmental factors, and operational complexities. Continued advancements in adaptive algorithms and AI-based optimization are essential for overcoming these challenges and fully realizing the potential of LoRaWAN in diverse IoT applications. Transmission parameter allocation is one of the most studied aspects of LoRaWAN, typically required to reduce energy consumption and improve the signal quality in dense LoRaWAN. Evaluations of the optimization algorithms for parameter allocation are usually done using simulators. However, they do not imitate the dynamic nature of the network environment and other signal characteristics. Thus, it becomes difficult to understand the performance of these algorithms when deployed on real devices. This paper introduces transmission parameter allocation strategies using a State–Action–Reward–State–Action (SARSA) and Deep Q-Learning Network (DQN) based Reinforcement Learning (RL)-based scheduling algorithm for allocating transmission parameters in LoRaWAN communication. We experimentally evaluate this algorithm in a water quality monitoring system using actual LoRaWAN devices to assess Signal to Noise ratio (SNR), Received Signal Strength Indicator (RSSI), Time on Air (ToA), and power consumption of our RL-based algorithms with the default and Adaptive Data Rate (ADR) in LoRaWAN communication.**

***Index Terms*—LoRaWAN, Energy Consumption, State–Action–Reward–State–Action (SARSA), Deep Q-Learning Network (DQN), Optimal Scheduling, Experimental Analysis.**

I. Introduction

The Internet of Things (IoT) has revolutionized various industries by enabling ubiquitous connectivity and intelligent automation. One of the prominent communication protocols in the IoT ecosystem is Long Range Wide Area Network (LoRaWAN), known for its low power consumption and wide coverage area. The efficiency and reliability of LoRaWAN heavily depend on the optimal allocation of transmission parameters such as spreading factor, transmission power, and channel frequency. Traditionally, static algorithms have been employed to manage these parameters. However, these static approaches often fall short in dynamic environments where network conditions and requirements vary significantly. The challenges associated with static algorithms in LoRaWAN transmission parameter allocation are multifaceted. Static algorithms are inherently inflexible and unable to adapt to changing conditions such as varying node densities, interference levels, and traffic patterns. This inflexibility can lead to suboptimal network performance, increased packet loss, reduced throughput, and higher energy consumption. Consequently, there is a growing need for more adaptive and intelligent methods to enhance the efficiency and reliability of LoRaWAN networks.

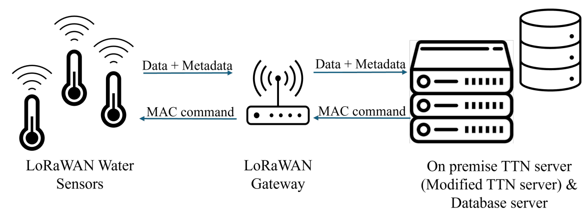


Fig. 1: LoRaWAN Architure

Artificial Intelligence (AI) is a promising solution to address these challenges. Studies are done to learn the dynamic network due to changes in traffic and interference leading to packet losses [1], [2]. AI-driven algorithms can dynamically adjust transmission parameters in real-time based on the current network state, thus optimizing performance [3] [4]. Machine learning techniques, in particular, can learn from historical data and predict optimal settings, providing a significant advantage over static methods. Reinforcement learning (RL) algorithms can perform online learning and adapt to dynamic. Despite the theoretical benefits of AI in this context, there are considerable hurdles in validating these approaches through simulation alone.

Addressing these issues is crucial to fully harness LoRaWAN's potential for sustainable and efficient environmental monitoring in real-world dynamic landscapes. While useful for preliminary testing, simulation environments often fail to capture the full complexity and variability of real-world deployments. Factors such as physical obstructions, varying environmental conditions, and real-time interference can significantly impact the performance of LoRaWAN networks and are challenging to replicate accurately in a simulated setting. Real-world data provides insights into these network phenomena. Therefore, to comprehensively evaluate the efficacy of AI-driven transmission parameter allocation, there is a pressing need for experimental analysis in real-world scenarios. This paper aims to address these gaps by performing an experimental study to evaluate and compare State–action–reward–state–action (SARSA) and Deep Q-learning network (DQN), RL—based scheduling algorithms with default ALOHA-based scheduling and adaptive data rate (ADR) feature of LoRWAN. Fig. 1 shows our network under consideration, where water quality monitoring sensors send data packets to LoRaWAN Gateway. The gateway forwards the packets to the on-premise TTN server for scheduling the transmission parameters using our algorithms. These predicted transmission parameters are sent back to the gateway which notifies them to the end devices through MAC commands.

In the remainder of this paper, we provide a detailed literature survey about existing papers performing experimental analysis of LoRaWAN in section II. Further, we present a system model, discuss our problem, and formulate it in section III. We provide solutions to solve this problem using our proposed scheduling algorithms in section IV followed by performance evaluation using comparative analysis in section VI. We conclude our paper with a short discussion of our observations and our results in section VII.

II. Related work

We performed a detailed literature survey of various studies that attempted to allocate transmission parameters to end devices. Based on our study, we can categorize them as,

A. Simulation-Based Transmission Parameter Allocation

Q-learning-based transmission parameter scheduling was evaluated using MULANE simulator in [5]. The paper, [6], proposes using RL for adaptive LoRaWAN transmission in industrial settings, enhancing reliability by about 10% without deviating from standard specifications. The reference [7] uses LoRaSim to evaluate their RL-based algorithm for allocation SF to the devices using contextual bandit problems, while transmission power is assigned centrally by treating it as a supervised ML problem. To ensure fewer collisions and a better PDR, [8] designs a deep RL algorithm for transmission parameter allocation in the physical layer. They designed a simulator for RL-based algorithm evaluation in LoRaWAN called LoRa-DRL. LP-MAB is another simulator designed in [9] with Mulitarm bandit problem (MAB) to reduce energy consumption and improve packet delivery ratio and coverage.

1) Limitations of the existing simulators: It is essential to compare various LoRaWAN simulation tools and compare them to see what features are missing in them and how close they go to a real network. Various LoRawAN simulation tools are studied in [10]. It enlists the features of each of the tools. The FREE simulator has duty cycle features but does not support AI-based algorithms. In the dynamic and ever-changing environment, this support is essential. NS3-based simulator lacks support for imperfect spread factors, energy consumption evaluation, and downlink traffic. LoRaWAN is known for its low power and energy consumption, which is why IoT networks use LoRaWAN. Missing support for this critical feature is a significant downside for this simulator.C++ based simulator also does not feature energy consumption evaluation. The capture effect is a considerable feature in reviewing the accuracy of PDR. LoRaEnergySim does not handle the capture effect. In [10], recommendations for simulators are also provided. The optimum configuration settings and network-based changes should be included. It is also essential to study the impact of multiple gateways on the simulation. In mobile networks, the challenges multiply due to multipath fading and increased interference. The effect of multiple gateways on a network is another essential simulator. Also, new features should be included, such as channel activity detection (CAD) as designed in our previous works [11].

B. Experiment-Based Transmission Parameter Allocation

Some studies perform experiment-based evaluations of LoRaWAN scheduling. Reference [12] shows that for dense networks, the ADR scheduling is inefficient and proposes using the multiagent DRL method to allocate spread factor and transmission power to meet QoS requirements. It uses a chip stack network server. Another paper [13] predicts SNR using an ML algorithm to obtain accurate SNR for optimal transmission parameters selection. It is performed in The Things Network server. It attempts to perform the ML on a network server. Their previous work [14] is an experimental evaluation by analyzing real-life data to show that path loss and shadow fading are related to environmental variables. They evaluate using ML algorithms and find the empirical path loss and shadow fading, which is used to set the transmission power to save the end node's energy. Another experiment-based evaluation in [15] provides a sliding window-based dynamic by the heuristic algorithm to allocate transmission parameters in LoRaWAN. Another paper [16] performs emulation using a real network server, but end devices are emulated. It proposes a multi-agent approach to efficient resource allocation in multi-SF LoRaWAN networks. It provides a heuristic-based approach for the resource as a transmission parameter allocation. It is an extension of [17], which performs scheduling and clock synchronization on real devices. It uses chirpstack network server to schedule the transmission and uses the heuristic algorithm for the scheduling under high-traffic and large-scale deployments,

In this survey, we found there are some major drawbacks with existing studies that should be handled,

- Simulation-based evaluations provide detailed studies of AI-based algorithms, but the tools are insufficient to study aspects, such as the dynamicity of the real environment.The results presented by algorithms using such tools should not be considered valid for the real world.
- Most experiment-based evaluations study the existing ADR algorithm in various scenarios. Very little research has been done to optimize the transmission parameters using new, but heuristic algorithms.

- Due to the complexity of network server implementation, the AI-based algorithms are implemented on end nodes, which increases the power consumption of the battery-powered end devices.
- There is a need to develop online algorithms for transmission parameter allocation to learn the changing network conditions and present more optimized scheduling results.

In this paper, we propose to use State–action–reward–state–action (SARSA) and Deep Q-learning network (DQN) for scheduling transmission parameters and perform an analysis such that

- We perform an experimental study such that algorithms are compared in a real network.
- We design an RL-based solution that enables continuous learning while scheduling and compares with ALOHA and adaptive data rate (ADR) algorithms.
- Our algorithms are deployed on the Things Network (TTN) server, run to schedule transmission parameters and sent to end devices using MAC commands.

III. System Model and Problem Formulation

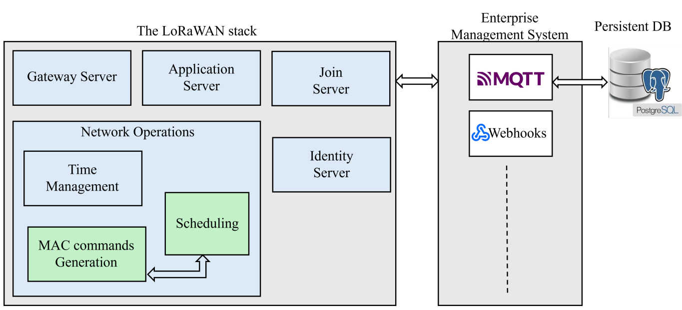


Fig. 2: On-Premise modified TTN Server Architecture

We used water quality monitoring systems as shown in Fig. 1 with LoRaWAN technology. The gateway acts as a forwarder to the network server. We adjusted the existing TTN server [18] and deployed it locally. We set up a PostgreSQL database to store uplink messages from the sensors. The network server consists of various modules, including the Gateway server, which handles communication with the LoRa gateway and traffic scheduling (Fig. 2). The application server manages the LoRaWAN application layer by decrypting and decoding uplink data, queuing downlink data, and encoding and encrypting downlink data. The Join Server deals with the LoRaWAN join flow, handling tasks such as authenticating the Network and Application Server and generating session keys. The Identity server stores registries of entities like applications with their end devices, gateways, users, organizations, OAuth clients, and authentication providers. We observe that evolving networks introduce more challenges. Existing simulators used to evaluate algorithms for improving the efficiency of LoRaWAN fail to include the aspects of the evolving network. LoRaWAN offers several advantages for environmental monitoring in urban areas but also faces specific challenges such as signal degradation and suboptimal transmission parameter allocation. Achieving coverage in urban environments is challenging due to buildings and obstacles, leading to signal loss and faster battery drainage for LoRaWAN devices deployed in hard-to-access locations [19]. In LoRaWAN deployments, optimizing transmission parameters is crucial for reliable communication, especially in urban areas. Existing algorithms may not be effective in urban environments. Simulation studies show that AI-based algorithms outperform traditional ones for parameter allocation. [7]. Our previous studies on transmission parameter allocation using RL algorithms are discussed in [20].

A. Problem Formulation

Due to dynamic and ever-changing environments, predicting obstacles and other environmental challenges is difficult. Transmission parameters used by end devices during uplink also decide the signal strength. The network server sends this parameter setting using MAC commands. To deal with the complex and changing environment, designing an AI-based solution for scheduling is necessary. One major issue in optimizing LoRWAN networks is the limited research performed on experimental analysis of AI algorithms for scheduling transmission parameters. Existing simulators do not simulate all the aspects of network traffic. Evaluating the scheduling solutions on simulators does not test them from end to end. Many faces of how the algorithms would perform go unverified. Moreover, existing experimental studies are inclined towards more static scheduling algorithms or the end device-hosted AI-powered scheduling algorithms. This increases the energy consumption of battery-powered devices. After preliminary analysis of algorithms on simulators, evaluating the scheduling algorithms using real devices is crucial. This helps us understand how the algorithm performs in a real network and reacts to unanticipated network changes.

IV. Proposed Solution

We propose using SARSA and DQN-based scheduling algorithms to predict each end device's spread factor and bandwidth. We formulate our problem as a Markov decision problem (MDP) and can be defined using components,

1) State (S): Metadata in the uplink message contains an Extended Unique Identifier (EUI) for each device. The state comprises Signal To Noise Ratio (SNR), data rate (SF+BW), demodulation floor, device margin, device locations, and channel steering information.
2) Action (A): we consider the data rate index [21] as an action to be predicted for each device.
3) Reward (R): The difference in battery levels between the previous and current messages acts as a reward. The lower this difference, the higher the reward obtained. Thus, the reward can be calculated as,
$$R = E_{m_i} - E_{m_{i-1}} \tag{1}$$
where E denotes the battery level and m denotes the uplink message identifier from device i
4) State transition probability matrix (P): Represents the probability of changing the signal quality and enhancing by selecting a particular data rate.

A. SARSA (State-Action-Reward-State-Action)

SARSA is an on-policy-based RL algorithm that updates its Q-values based on its action, following its current policy [22]. SARSA maintains a Q-table $Q(s,a)$ where each entry represents the expected cumulative reward for taking action a in state s. We assume that the s' is the received state from the uplink message, and based on previously stored s and previously predicted a, we obtained its rewards r in the current message. This is because, based on the last state and predicted action, the environment reacted and generated a reward in the current uplink message. We train the network using all this information and predicted action a' for the current uplink. Training is done using Bellman's equation denoted as,

$$Q(s',a') = Q(s',a') + \alpha * [r + \gamma * Q(s,a) - Q(s',a') \quad (2)$$

where γ is a discounting factor and α is learning rate for temporal difference learning. The learning rate α determines the behavior of the algorithm Sarsa. Too large values α will keep our algorithm from converging to optimal policy.

B. DQN (Deep Q-Learning Network)

DQN addresses the limitations of traditional Q-learning, which struggles with high-dimensional state spaces. DQN can handle large and complex state spaces by using deep neural networks as function approximators, enabling it to solve previously intractable problems for standard Q-learning methods [23]. DQN uses a neural network to approximate the Q-value function, Q$(s, a; \theta)$, where θ are the parameters (weights) of the neural network. Training in DQN is done using Bellman's equation to maximize Q-value as,

$$Q(s',a') = Q(s',a') + \alpha * [(r\_i + \gamma * maxQ'(s'\_i, a; \theta)) - Q(s',a')] \quad (3)$$

where γ is a discounting factor and α is learning rate for temporal difference learning. At any time step i, for state s'_i, at least one action a exists, whose estimated value $Q(s'\_i, a)$ is maximal. This action a is called greedy action. When we choose one of the greedy actions, we are using our current knowledge to our advantage. However, when we opt for one of the non-greedy actions, we are exploring, which helps us improve our estimate of the non-greedy action's value.

C. Overview of the RL Algorithms in The Things Network (TTN) Server

In the TTN server, the received uplink messages from the gateway are added to the worker pool. Each incoming request from each device is treated as a separate task, and worker routines process them simultaneously for quick and responsive server performance. It is submitted to the handle uplink message routine, which adapts the data rate by calling the scheduling algorithms we deployed on our network server. Fig. 3 shows details of the internal working of the network server. The predicted transmission parameters are sent to the uplink handler, who is responsible for generating the MAC command structure to send to the LoRaWAN gateway. The gateway forwards the MAC commands to the end devices for their future uplink message.

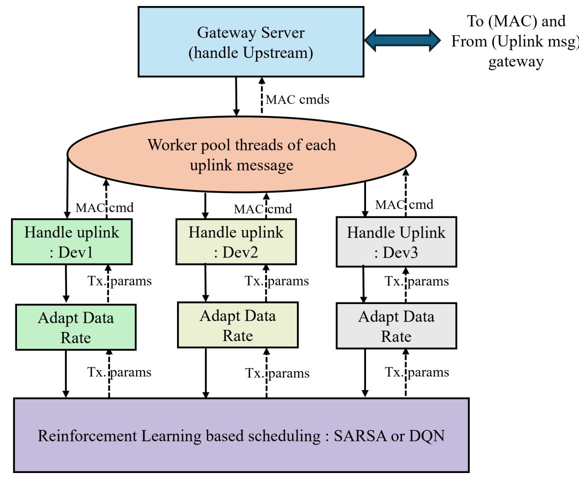


Fig. 3: Process in Network Server - Internals of TTN

V. Experiment Setup

We evaluate our algorithms using an experimental setup for LoRaWAN. Our network setup is spread over approximately 120 sq.m of experimental labs. We planted our gateway on one of the room's corners, and devices spread across the room. We use Lora shield with SX1272 (Fig. 4a) on Arduino UNO R3 and Dragino gateway LPS8v2 (Fig. 4b). For energy evaluation purposes, we used a 9V battery (Fig. 4c). We sent messages at a frequency of uplink per minute from each device. The signal characteristics such as SNR, RSSI, and Time on air for evaluation purposes are stored in PostgreSQL using MQTT broker as an enterprise management system as shown in Fig. 2. We evaluate the scheduling of each algorithm by using 3 sensors transmitting messages every 1 minute to the gateway. We capture the results by reading the uplink messages from the sensors. The metrics we store from the metadata of each uplink message are SNR, RSSI, and Consumed airtime for each message transmission. We also measure the voltage drop of each device during transmission. We transmit 385 messages per device; every packet is transmitted per minute. The uplink messages are stored in PostgreSQL and then evaluated.

VI. Performance Evaluation

The Network server is implemented in GoLang. We implemented DQN and SARSA algorithms for scheduling in the network server. Implementation details can be found in [24]. We can use any one algorithm at a time. Changing the algorithm for the network server requires a few changes, as shown in its documentation. We compare our proposed SARSA and DQN-based scheduling techniques with the ADR mechanism used in LoRaWAN and the default ALOHA-based mechanism. For selecting a scheduling algorithm among SARSA, DQN, and ADR, updates must be done to scrip in [24]. To enable ALOHA, select ADR-based scheduling in scrip and use the console to disable the ADR mechanism.

(a) LoRa shield (SX1272) for Arduino R3 (b) Dragino LPS8v2 gateway (c) Battery powered sensors

Fig. 4: Experimental Test Components of Water Quality Monitoring System

We use SNR, RSSI, and Latency as performance metrics. Fig. 5 compares our proposed scheduling techniques SARSA and DQN performances with existing ADR and ALOHA techniques. We performed transmission with each scheduling for 385 packets, one packet sent per minute, and recorded the metadata of each uplink message. The box plot for SNR values observed for all uplinks is shown in Fig. 5a. For DQN, about 75% of the uplink messages had SNR values above 12.35 dB. The median is more skewed towards the 75^{th} percentile, indicating that most values have higher SNR tending towards 13.5 dB. For SARSA-based scheduling, have about 75% of their uplinks with SNR more than 13 dB. On the contrary, the uplinks scheduled using ADR have a 25th percentile at 8 dB, much lower than our proposed algorithms. However, ALOHA-based scheduling has better SNR, with 3/4th uplinks having more than 12.5dB SNR. Here, we observe that DQN, SARSA, and ALOHA signals were stronger than DQN.

However, SNR is not sufficient for evaluation. RSSI gives an actual measure of how the signal reacted to the network and, after attenuation, the strength of the signal at the receiver. Fig. 5b compares RSSI of uplinks for scheduling strategies. We observe that both DQN and SARSA have higher RSSI values for their uplinks. The median values are higher than -80 dBm for both algorithms, and the minimum RSSI for any uplink is more than -90 dBm. Unlike our proposed algorithms, the ADR and ALOHA-based scheduling have much lower RSSI; their maximum achieved RSSI is almost equivalent to the minimum RSSI or DQN and SARSA. This shows that even with a higher SNR for ALOHA, it still has a lower RSSI. One of the reasons for this signal attenuation is the suboptimal transmission parameter selection of end devices in ALOHA.

Fig. 5c shows the distribution of latencies observed in uplink messages by each scheduling algorithm. In all the scheduling algorithms, the median latency is about 0.087s and is the same for all the algorithms. DQN observes variation in latencies where 25% of messages have lower latency than 0.038s

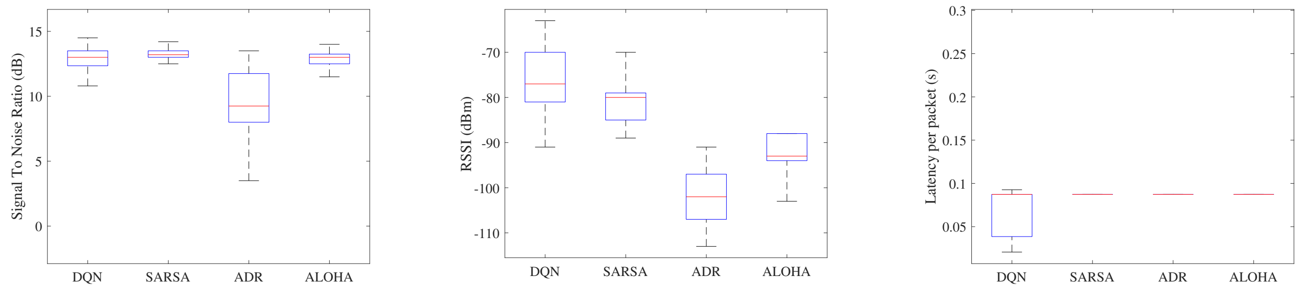
Energy consumption can be evaluated as the voltage level change rate in the battery. We connected the battery to the end device and transmitted it to the gateway. Fig. 6 shows the changing voltage levels. Readings are taken every 10 minutes as the voltage level of the battery is connected. We observe a steep drop in voltage for ALOHA, indicating higher energy consumption due to increased collisions and retransmissions. Our proposed algorithms perform better when compared to ADR and ALOHA. The DQN algorithm has the lowest energy consumption, about 13% lower than ADR and 77% lower than ALOHA. SARSA also has higher performance in terms of energy consumption than ADR and ALOHA. Based on the performance evaluation, we observe that signal quality, latency, and energy consumption can be improved using intelligent scheduling of transmission parameters. Our algorithm allocated unique spread factors and bandwidth to end devices based on signal characteristics and device locations. Improved performance is achieved due to lowering collisions and retransmissions, thus retaining the batteries.

VII. Conclusion

This paper focuses on the experimental analysis of reinforcement learning-based algorithms for scheduling and allocating transmission parameters for end devices in LoRaWAN. Dynamic changes in the network introduce signal attenuation, multipath fading, and mobility-induced issues. Existing simulators do not support these network scenarios and, thus, are unsuitable for evaluating scheduling algorithms. We operate on an on-premise TTN server using State–Action–Reward–State–Action (SARSA) and Deep Q-Learning Network (DQN) for scheduling. Experimental evaluation is done using real devices and real Gateway. Comparative analysis with improved adaptive data rate (ADR) shows improvements in signal strength and energy consumption. The energy consumption of DQN was reduced by 13% and 77% compared to ADR and ALOHA, respectively. For SARSA, the reduction was 7% and 66% compared to ADR and ALOHA, respectively. Additionally, improvements in RSSI, a measure of signal strength, were observed: DQN improved by 24% and 17%, and SARSA improved by 21% and 13% compared to ADR and ALOHA, respectively. As part of future work, we plan to implement mobility, transmission power and frequency allocation. We also plan to extend using solar-powered batteries and evaluate the performance of the outdoor gateway.

VIII. Acknowledgement

This research supported by US NSF grant from Research Experience for Undergraduate (REU) Site, CNS-2244450.



(a) SNR Comparison (b) RSSI Comparison (c) Latency Comparison

Fig. 5: Comparison of packet characteristics for the packets sent to LoRaWAN gateway.

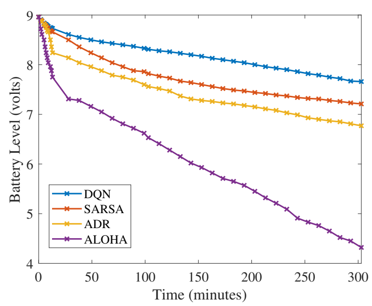


Fig. 6: Energy consumption evaluation based on voltage levels of 9v batteries for transmission in LoRaWAN

References

[1] A. E. Ferreira, F. M. Ortiz, L. H. M. Costa, B. Foubert, I. Amadou, and N. Mitton, "A study of the lora signal propagation in forest, urban, and suburban environments," *Annals of Telecommunications*, vol. 75, pp. 333–351, 2020.

[2] M. Rademacher, H. Linka, T. Horstmann, and M. Henze, "Path loss in urban lora networks: A large-scale measurement study," in *2021 IEEE 94th Vehicular Technology Conference (VTC2021-Fall)*. IEEE, 2021, pp. 1–6.

[3] D. Magrin, M. Capuzzo, and A. Zanella, "A thorough study of lorawan performance under different parameter settings," *IEEE Internet of Things Journal*, vol. 7, no. 1, pp. 116–127, 2019.

[4] O. A. Agbolade, "Lorawan: The dilemma of optimal transmission parameter," *World Journal of Advanced Research and Reviews*, vol. 22, no. 2, pp. 962–971, 2024.

[5] M. Chen, L. Mokdad, J. Ben-Othman, and J.-M. Fourneau, "Dynamic parameter allocation with reinforcement learning for lorawan," *IEEE Internet of Things Journal*, 2023.

[6] T. Fedullo, A. Morato, F. Tramarin, P. Bellagente, P. Ferrari, and E. Sisinni, "Adaptive lorawan transmission exploiting reinforcement learning: The industrial case," in *2021 IEEE International Workshop on Metrology for Industry 4.0 & IoT (MetroInd4. 0&IoT)*. IEEE, 2021, pp. 671–676.

[7] S. U. Minhaj, A. Mahmood, S. F. Abedin, S. A. Hassan, M. T. Bhatti, S. H. Ali, and M. Gidlund, "Intelligent resource allocation in lorawan using machine learning techniques," *IEEE Access*, vol. 11, pp. 10 092–10 106, 2023.

[8] I. Ilahi, M. Usama, M. O. Farooq, M. Umar Janjua, and J. Qadir, "Loradrl: Deep reinforcement learning based adaptive phy layer transmission parameters selection for lorawan," in *2020 IEEE 45th Conference on Local Computer Networks (LCN)*, 2020, pp. 457–460.

[9] B. Teymuri, R. Serati, N. A. Anagnostopoulos, and M. Rasti, "Lp-mab: Improving the energy efficiency of lorawan using a reinforcement-learning-based adaptive configuration algorithm," *Sensors*, vol. 23, no. 4, p. 2363, 2023.

[10] J. M. Marais, A. M. Abu-Mahfouz, and G. P. Hancke, "A review of lorawan simulators: Design requirements and limitations," in *2019 international multidisciplinary information technology and engineering conference (IMITEC)*. IEEE, 2019, pp. 1–6.

[11] J. Mhatre, A. Lee, H. Lee, and R. Aygun, "Design and implementation of multithreaded cad-enabled lora simulator for massive iot," in *2024 18th International Conference on Ubiquitous Information Management and Communication (IMCOM)*, 2024, pp. 1–7.

[12] A. Tellache, A. Mekrache, A. Bradai, R. Boussaha, and Y. Pousset, "Deep reinforcement learning based resource allocation in dense sliced lorawan networks," in *2022 IEEE International Conference on Consumer Electronics (ICCE)*. IEEE, 2022, pp. 1–6.

[13] M. González-Palacio, D. Tobón-Vallejo, L. M. Sepúlveda-Cano, M. Luna-delRisco, C. Roehrig, and L. B. Le, "Machine-learning-assisted transmission power control for lorawan considering environments with high signal-to-noise variation," *IEEE Access*, 2024.

[14] M. González-Palacio, D. Tobón-Vallejo, L. M. Sepúlveda-Cano, S. Rúa, and L. B. Le, "Machine-learning-based combined path loss and shadowing model in lorawan for energy efficiency enhancement," *IEEE Internet of Things Journal*, 2023.

[15] E. F. Silva, L. M. Figueiredo, L. A. de Oliveira, L. J. Chaves, A. L. de Oliveira, D. Rosário, and E. Cerqueira, "Adaptive parameters for lora-based networks physical-layer," *Sensors*, vol. 23, no. 10, p. 4597, 2023.

[16] C. Garrido-Hidalgo, L. Roda-Sanchez, F. J. Ramírez, A. Fernández-Caballero, and T. Olivares, "Efficient online resource allocation in large-scale lorawan networks: A multi-agent approach," *Computer Networks*, vol. 221, p. 109525, 2023.

[17] C. Garrido-Hidalgo, J. Haxhibeqiri, B. Moons, J. Hoebeke, T. Olivares, F. J. Ramirez, and A. Fernández-Caballero, "Lorawan scheduling: From concept to implementation," *IEEE Internet of Things Journal*, vol. 8, no. 16, pp. 12 919–12 933, 2021.

[18] "GitHub - TheThingsNetwork/lorawan-stack at v3.30 — github.com," https://github.com/TheThingsNetwork/lorawan-stack/tree/v3.30, [Accessed 17-06-2024].

[19] S. M. M. S. Zakaria, A. Zakaria, L. M. Kamarudin, R. Visvanathan, Y. A. Rahim, N. Azmi, and M. F. Elham, "Case study of lorawan-based smart elements in urban environment," in *AIP Conference Proceedings*, vol. 2898, no. 1. AIP Publishing, 2024.

[20] J. Mhatre and A. Lee, "Dynamic reinforcement learning based scheduling for energy-efficient edge-enabled lorawan," in *2022 IEEE International Performance, Computing, and Communications Conference (IPCCC)*, 2022, pp. 412–413.

[21] L. A. T. C. R. P. Workgroup, "LoRaWAN 1.0.3 Regional Parameters," https://lora-alliance.org/wp-content/uploads/2020/11/lorawan_regional_parameters_v1.0.3reva_0.pdf, [Accessed 25-06-2024].

[22] G. A. Rummery and M. Niranjan, *On-line Q-learning using connectionist systems*. University of Cambridge, Department of Engineering Cambridge, UK, 1994, vol. 37.

[23] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[24] J. Mhatre, "Ttn based local network server with rl scheduling," https://github.com/JuiMhatre/ttn-lorawan-local-server, [Accessed 01-07-2024].