Journal of
Sensor and Actuator Networks

MDPI

Article

Enhanced Reinforcement Learning Algorithm Based-Transmission Parameter Selection for Optimization of Energy Consumption and Packet Delivery Ratio in LoRa Wireless Networks

Batyrbek Zholamanov, Askhat Bolatbek, Ahmet Saymbetov *, Madiyar Nurgaliyev, Evan Yershov, Kymbat Kopbay, Sayat Orynbassar, Gulbakhar Dosymbetova *, Ainur Kapparova, Nurzhigit Kuttybay and Nursultan Koshkarbay

Faculty of Physics and Technology, Al-Farabi Kazakh National University, 71 Al-Farabi, Almaty 050040, Kazakhstan; zholamanov.batyrbek@kaznu.kz (B.Z.); askhat.bolatbek@kaznu.edu.kz (A.B.); madiyar.nurgaliyev@kaznu.edu.kz (M.N.); yershov_evan@kaznu.edu.kz (E.Y.); kopbay_kymbat2@kaznu.edu.kz (K.K.); sayat.orynbassar@kaznu.edu.kz (S.O.); kapparova_ainur3@kaznu.edu.kz (A.K.); nurzhigit.kuttybyy@kaznu.edu.kz (N.K.); koshkarbay_n@kaznu.edu.kz (N.K.)

* Correspondence: asaymbetov@kaznu.kz (A.S.); gulbakhar.dossymbetova@kaznu.edu.kz (G.D.)

Abstract: Wireless communication technologies (WSN) are pivotal for the successful deployment of the Internet of Things (IoT). Among them, long-range (LoRa) and long-range wide-area network (LoRaWAN) technologies have been widely adopted due to their ability to provide long-distance communication, low energy consumption (EC), and cost-effectiveness. One of the critical issues in the implementation of wireless networks is the selection of optimal transmission parameters to minimize EC while maximizing the packet delivery ratio (PDR). This study introduces a reinforcement learning (RL) algorithm, Double Deep Q-Network with Prioritized Experience Replay (DDQN-PER), designed to optimize network transmission parameter selection, particularly the spreading factor (SF) and transmission power (TP). This research explores a variety of network scenarios, characterized by different device numbers and simulation times. The proposed approach demonstrates the best performance, achieving a 17.2% increase in the packet delivery ratio compared to the traditional Adaptive Data Rate (ADR) algorithm. The proposed DDQN-PER algorithm showed PDR improvement in the range of 6.2–8.11% compared to other existing RL and machine-learning-based works.

Keywords: LoRaWAN; wireless sensor networks; packet delivery ratio; reinforcement learning; Double Deep Q-Network with Prioritized Experience Replay (DDQN-PER); transmission parameter selection; Adaptive Data Rate; energy consumption

Citation: Zholamanov, B.; Bolatbek, A.; Saymbetov, A.; Nurgaliyev, M.; Yershov, E.; Kopbay, K.; Orynbassar, S.; Dosymbetova, G.; Kapparova, A.; Kuttybay, N.; et al. Enhanced Reinforcement Learning Algorithm Based-Transmission Parameter Selection for Optimization of Energy Consumption and Packet Delivery Ratio in LoRa Wireless Networks. *J. Sens. Actuator Netw.* **2024**, *13*, 89. https://doi.org/10.3390/jsan13060089

Academic Editor: Lei Shu

Received: 11 November 2024
Revised: 10 December 2024
Accepted: 17 December 2024
Published: 20 December 2024

1. Introduction

In the current landscape of rapidly evolving IoT applications, there is a growing demand for long-range low-power wide-area network (LPWAN) wireless transmission technologies that minimize energy consumption (EC), while ensuring cost-effectiveness. LPWAN technologies are specifically designed for interaction between machine-to-machine (M2M) systems and Internet of Things (IoT) devices. The main advantages of LPWAN technology over other wireless solutions include its extensive range of radio signal transmission, low power consumption of end devices, utilization of unlicensed frequency bands, and high network scalability. These benefits make LPWAN suitable for a wide range of applications, providing efficient data collection from various devices such as sensors, utility meters, and fire alarm devices.

There are several popular LPWAN technologies today, such as SigFox, long-range wide-area network (LoRaWAN), Narrow Band Internet of Things (NB-IoT), and Long-Term Evolution for Machines (LTE-M) [1]. SigFox and LoRaWAN operate in unlicensed frequency bands, which helps reduce operating costs. In contrast, NB-IoT and LTE-M use licensed cellular bands, which increases costs. SigFox is well suited for simple solutions involving small amounts of data, but has limitations in terms of data transmission capacity and flexibility. Consequently, LoRaWAN has emerged as the preferred choice for establishing networks that provide long-range communication with low power consumption. Nevertheless, selecting optimal transmission parameters within LoRa networks remains a significant challenge, as it is crucial for further reducing node EC and enhancing overall network efficiency.

LoRaWAN was developed and is maintained by the LoRa Alliance [2]. In order to minimize EC of network nodes and simultaneously maximize throughput, LoRaWAN employs the Adaptive Data Rate (ADR) mechanism, which automatically adjusts transmission parameters such as spreading factor (SF), bandwidth (BW), coding rate (CR), transmission power (TP), and carrier frequency (CF) [3]. This algorithm plays an essential role in managing the data transfer rate and transmission power within LoRaWAN networks. The selection of transmission parameters for LoRaWAN wireless networks is to achieve a trade-off between various parameters, such as EC and PDR, PRR, goodput. Reducing EC usually means reducing TP, which in turn reduces the probability of successful message delivery. Conversely, ensuring a high probability of message delivery requires increasing TP. Also, for long distances, a high SF value is required, since with a small SF value, the signal may not reach the node. Therefore, the optimization of transmission parameters of the LoRa network is very critical and requires careful research. Figure 1 illustrates the process of optimizing the data transmission parameters in the LoRa network environment using the optimal transmission parameter selection algorithm. The task of the algorithm is to select parameters such as SF, BW, TP and CR to achieve the best transmission characteristics. The simulation environment created in NS-3 allows evaluating the performance of the LoRa network taking into account metrics such as EC, PDR, PRR and goodput.

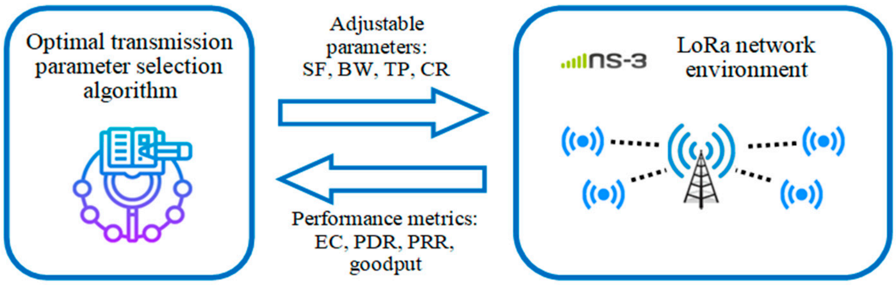

Figure 1. Optimal transmission parameter selection algorithm framework.

Many researchers worldwide have devoted considerable attention to the study of the ADR mechanism and its impact on network performance with static or mobile nodes under various conditions. The study in [4] discusses how adaptive parameter configuration can improve network performance in dense IoT deployments. In contrast, the authors in [5] investigate the flexibility of the ADR algorithm and its impact on network performance under various operating conditions. However, the ADR algorithm exhibits several limitations, which led to further research in this area [6,7]. Firstly, ADR demonstrates the best results with stationary devices; for mobile nodes that move from one point to another, a static ADR configuration will be ineffective, as the algorithm may not have sufficient time to adjust the network transmission parameters. Secondly, in conditions characterized by frequent changes in external factors, such as temporary interference, variations in device density, and object movement, ADR struggles to adapt the transmission parameters quickly, leading to a decrease in quality of service (QoS). Thirdly, ADR faces scalability issues; in large networks with uneven device distribution, the ADR mechanism does not account for these disparities, potentially resulting in network congestion due to inadequate optimization of transmission parameters. Based on the above limitations of the ADR mechanism, there is a need to develop new solutions for selecting optimal transmission parameters in wireless sensor networks, especially for large scale. To address these issues, a promising direction is the use of machine learning and deep learning methods that can learn from incoming data and predict optimal transmission parameters for specific network conditions. Such approaches will provide greater adaptability, performance, and scalability compared to traditional ADR.

The paper is organized as follows: after the introduction, Section 2 reviews related works and highlights the novelty of the proposed approach. Section 3 provides the LoRaWAN background. Section 4 describes the system model, including simulation parameters in the NS3-LoRaWAN environment, and the RL and ADR algorithms for selecting optimal transmission parameters. Section 5 presents the results, comparing the outcomes of the RL-based algorithm with those of the ADR algorithm, identifying node parameters that strike a balance between energy consumption (EC) and packet delivery ratio (PDR). Section 6 offers a comparative analysis of the proposed algorithm with the works of other researchers. Section 7 summarizes the findings and outlines future research directions.

2. Related Works

Modern researchers propose improved versions of ADR, such as SSFIR-ADR, LR + ADR, K-ADR and EARN to address the limitations of traditional ADR algorithm. These advanced algorithms consider the average SNR (Signal-to-Noise Ratio) value to update data transmission parameters, resulting in an improved PDR and reduced EC [8–11]. In the paper [8], the proposed SSFIR-ADR algorithm improves PDR and reduces energy consumption by over four times compared to the standard ADR, leveraging randomized spreading factor allocation to optimize LoRaWAN network performance. Jiang et al. presented the K-ADR algorithm, which uses the ordinary kriging function to dynamically adjust the transmission parameters, which can improve the packet delivery ratio by 21.46% compared with ADR and enhance the reliability under harsh environments [9]. Park J. et al. developed EARN, an improved greedy ADR mechanism that uses coding rate adaptation to optimize the trade-off between delivery ratio and energy consumption. In addition, large-scale simulation results show that this method outperforms traditional schemes in efficiency [10]. In paper [11], the authors presented a novel LR + ADR mechanism that significantly enhances the PDR while keeping the EC per delivered packet low. In real-world scenarios, LR + ADR demonstrates up to a 520% improvement in PDR compared to the traditional ADR and up to a 38% advantage over the best competitor, G-ADR. Recent studies suggest that the ADR mechanism and its modified versions do not always select the most efficient mode of network operation [12]. Inefficient energy usage, particularly in wireless sensor networks with autonomous wireless nodes, leads to accelerated node discharge, resulting in additional operational expenses. One of the solutions to this problem, along with an effective routing algorithm [13], security [14] and optimization of the indoor nodes localization [15,16], is the application of machine learning (ML) techniques to determine the most optimal operating mode of the entire network.

For the task of selecting transmission parameters for LoRa wireless network nodes, three ML methods are employed: supervised learning (SL), unsupervised learning (USL), and reinforcement learning (RL). In SL, network parameters, such as SF or TP, are typically chosen based on known data, utilizing regression or classification techniques for prediction [17]. In addition, these methods are also used to predict collisions. The paper [18] introduces a SL approach to configure two PHY-layer parameters aimed at reducing EC in LoRa networks. Similarly, the authors of [19] focus on enhancing the energy efficiency of end nodes by comparing classification algorithms, in particular k-NN, Naïve Bayes and Support Vector Machines (SVM), for assigning the SF in LoRa networks. Despite their effectiveness in certain scenarios, SL methods have significant limitations. One of the primary challenges is their reliance on labeled data, which is often expensive and time-consuming to collect. Additionally, SL methods require substantial computational resources for training, especially when dealing with large datasets. As a result, these models may be inefficient in real-world conditions where quick and cost-effective data processing is essential, limiting their applicability in LoRa networks.

Unsupervised learning is used to determine the most efficient operating modes of network nodes [20–22]. In the paper [20], a multi-hop clustering approach using the Mini batch K-means clustering (MBKMC) algorithm is proposed to address load imbalance and computational complexity in LoRaWAN networks, reducing collision rates and improving resource allocation efficiency. In the paper [21], the authors introduce a dynamic priority scheduling technology (PST) that utilizes a USL clustering algorithm to minimize packet collisions while enhancing transmission delay and energy consumption within the network. In the paper [22], the authors propose a K-means clustering-based algorithm to solve the LoRa SF distribution problems. Using USL methods has its own drawbacks, including lower accuracy compared to SL for specific tasks, as well as difficulty in determining the number of clusters and other hyperparameters. Moreover, the training process can be slow, especially for algorithms that require a large number of iterations. USL methods can find local minima or solutions that are far from optimal.

Reinforcement learning in the context of selecting transmission parameters offers an intuitive approach. This method allows a wireless network node (agent) to interact with its environment, receiving feedback in the form of "reinforcement" or "punishment", thereby finding the optimal transmission parameters [23–27]. In the paper [23], the authors proposed a new algorithm with a two-expert EXP4 algorithm to distribute the SF and TP to devices using a combination of decentralized and centralized approaches. The paper [24] presents a distributed Markov decision process (MDP) model for uplink transmission in Class A LoRaWAN devices, which improves the packet transmission performance through dynamic SF allocation strategies. The paper [25] presents the Low-Power Multi-Armed Bandit (LP-MAB) algorithm, which centrally configures transmission parameters on the network server to optimize power consumption while maintaining high packet delivery rate (PDR). Fedullo T. et al. propose a new RL-based adaptation strategy for LoRaWAN in industrial sensing systems, demonstrating improved packet reception compared to the standard ADR strategy while maintaining similar power consumption [26]. The paper [27] discusses the use of ML techniques, including RL, and proposes a novel proactive approach—"artificial intelligence-empowered resource allocation" (AI-ERA) to optimize resource allocation in LoRa-based IoT applications. Table 1 presents a comprehensive comparison of the proposed DDQN-PER algorithm with existing methods, including ADR-based approaches, SL, USL, and RL techniques. The Table 1 highlights the limitations of traditional algorithms and underscores the contributions of the proposed method.

As a result of the comparison of these three methods, supervised learning is effective in the presence of labeled data but is limited in its ability to adapt to new conditions. Unsupervised learning is used to optimize operating modes but is constrained by the accuracy and complexity of hyperparameter tuning. Reinforcement learning stands out for its ability to find optimal solutions in real time, making it the most promising approach. However, existing RL algorithms for the task of selecting transmission parameters in LoRa networks require significant training time and are both labor- and resource-intensive. Moreover, many reinforcement learning algorithms use the LoRa network gateway as the main agent. This can lead to overload and deterioration of the LoRa gateway performance due to the constant changing environment, especially for large-scale LoRa nodes. Therefore, we propose a new algorithm designed to effectively handle challenging conditions and focus on critical transitions, enabling faster identification of optimal transmission parameters. The developed DDQN-PER algorithm is a new solution for optimizing transmission parameters in static LoRaWAN networks. The algorithm combines the advantages of deep learning and Prioritized Experience Replay, which provides high adaptability, scalability and efficiency in complex network conditions, outperforming existing ADR methods and ML algorithms.

The main contributions of this study are summarized as follows:

1. A Novel Double Deep Q-Network with Prioritized Experience Replay (DDQN-PER) algorithm was proposed for optimizing LoRaWAN transmission parameters (SF, TP). The algorithm effectively addresses Q-value overestimation, enhances learning stability, and ensures efficient parameter selection in large-scale network environments.

2. To evaluate the performance of DDQN-PER, extensive simulations were conducted and compared with various ADR mechanisms, including ADR-MAX, ADR-AVG, and ADR-MIN, as well as other reinforcement learning methods such as Q-learning and DQN. The results demonstrate that DDQN-PER achieves optimal resource allocation within 24 h, maintaining low energy consumption and high scalability for networks with up to 1000 devices, while ensuring high PDR across diverse scenarios.
3. The simulation study considers challenging conditions such as high node density (up to 1000 nodes), varying simulation durations (up to seven days), and environments with obstacles. Results indicate that DDQN-PER significantly outperforms existing approaches in terms of energy efficiency, and robustness, making it highly adaptable to complex LoRaWAN deployments.

Table 1. Comprehensive comparison of the proposed DDQN-PER algorithm with existing methods.

Category	Method/ Reference	Key Features	Limitations	Proposed Solution (DDQN-PER)
ADR methods	Standard ADR [4]	Configures SF and TP based on historical SNR for static nodes.	Poor performance in networks with high density or varying conditions; limited scalability.	Achieves optimal resource allocation for static environments with high node density.
	SSFIR-ADR [8]	Uses randomized SF allocation to improve PDR and reduce energy consumption.	May result in suboptimal SF selection; lacks adaptability for large-scale static deployments.	Ensures accurate and scalable SF/TP selection while maximizing PDR.
	K-ADR [9]	Dynamically adjusts SF/TP using kriging functions to improve reliability.	Computationally intensive; limited validation in large-scale static networks.	Faster convergence and reliable performance in static large-scale networks.
	EARN [10]	Greedy ADR mechanism with coding rate adaptation to balance PDR and EC.	Greedy methods may converge to local optima; scalability issues with dense static networks.	Stable and globally optimal SF/TP selection in dense environments.
	LR + ADR [11]	Combines regression-based ADR with dynamic adaptation, improving PDR.	Requires significant computational resources; suboptimal for large-scale static node networks.	Efficient learning for optimal transmission parameters with minimal overhead.
Supervised Learning	SL for PHY-layer [18]	Configures PHY-layer parameters to reduce energy consumption.	Requires labeled data, costly to collect; computationally expensive.	Reduces resource requirements and achieves optimal transmission selection.
	k-NN, SVM, NB [19]	Compares classification algorithms to improve energy efficiency in LoRa nodes.	Limited scalability; higher complexity for static networks.	Optimized for static networks with efficient learning and lower overhead.

Table 1. *Cont.*

Category	Method/ Reference	Key Features	Limitations	Proposed Solution (DDQN-PER)
Unsupervised Learning	K-means [22]	Clusters nodes to optimize SF allocation and reduce collisions.	Requires careful hyperparameter tuning; less precise for static networks.	Faster optimization for static environments with high precise.
	MBKMC [20]	Mini-batch K-means for resource allocation.	Slow convergence for static, large-scale networks.	Accelerates convergence and optimizes resource allocation for static setups.
	PST with clustering [21]	Enhances energy consumption using clustering algorithms.	Accuracy limitations in static, large-scale networks.	Ensures stable, optimal transmission parameter selection.
Reinforcement learning	Two-expert EXP4 [23]	SF/TP optimization using RL with central-ized/decentralized learning.	Computationally heavy and unsuitable for static environments.	Efficient Q-learning structure optimized for static networks.
	MDP for Lo-RaWAN [24]	Distributed RL improves SF allocation for uplink.	Gateway overload issues in large-scale static setups.	Reduces gateway overload and improves scalability.
	LP-MAB [25]	Optimizes power consumption while maintaining PDR using MAB.	Requires high computational power and extended training time for large datasets.	Reduces training time while maintaining robustness and adaptability.
	RL-based adaptation strategy [26]	RL-based SF/TP optimization improves packet reception in LoRaWAN.	Does not address Q-value overestimation; limited evaluation for scalability.	Addresses Q-value overestimation for stable learning in large-scale networks.
	AI-ERA [27]	Proactive RL approach for resource allocation in LoRa IoT applications.	Requires high computational power and extended training time for large datasets.	Reduces training time while maintaining robustness and adaptability.

3. LoRaWAN Background

This section provides an overview of LoRaWAN technology, including its basic principles, network architecture, and key transmission parameters. This section also discusses the path loss model used for communication in LoRa networks and the performance metrics used to evaluate the proposed reinforcement learning algorithm.

3.1. LoRaWAN Overview

In 2014, the LoRa Alliance developed a standard called LoRaWAN by Semtech, which is a physical layer modulation method based on Chirp Spread Spectrum (CSS) technology [28,29]. LoRa uses CSS modulation to increase receiver sensitivity and reduce the risk of interference. The standard LoRaWAN architecture is a star topology, where end devices (nodes) transmit and receive signals at one or more access points (gateways). The received packets are then sent to network servers, which in turn are connected to standard Internet Protocol (IP) networks.

In LoRaWAN, the theoretical bit rate at SF_k, $k = 7, 8, 9 \ldots 12$, is given by

$$R^{SF_k} = \frac{BW \times SF_k \times CR}{2^{SF_k}} \tag{1}$$

where BW is the bandwidth in [Hz], CR is the coding rate, and SF indicates the spreading factor. Table 2 presents a summary of the bit rate and SNR for the LoRaWAN configuration, specifically at a bandwidth (BW) of 125 kHz.

Table 2. LoRaWan Configuration table [30].

Configuration	Bit Rate, b/s	Required SNR, dB
SF12/125 kHz	293	−20.0
SF11/125 kHz	537	−17.5
SF10/125 kHz	976	−15.0
SF9/125 kHz	1757	−12.5
SF8/125 kHz	3125	−10.0
SF7/125 kHz	5469	−7.5

3.2. The Path Loss Model

In wireless networks, the path loss is usually modeled using a logarithmic distance power law with a random term, and we use the logarithmic path loss model to analyze the LoRa networks, as follows [10]:

$$PL(d) = PL(d_0) + 10n\log_{10}\left(\frac{d}{d_0}\right) + X_\sigma \tag{2}$$

where

$PL(d)$—path losses over distances d;
$PL(d_0)$—losses at reference distance d_0;
n—attenuation coefficient;
X_σ—random value, normally distributed with zero mean and standard deviation σ.

3.3. LoRa Network Transmission Parameters

Below are the key transmission parameters that need to be considered in LoRa networks to minimize the power consumption of network nodes while maximizing PDR.

1. Bandwidth

Bandwidth affects the data transfer rate and range. In LoRa networks, the bandwidth values are: 125 kHz, 250 kHz and 500 kHz. For long distances, it is necessary to set the BW value to 125 kHz and vice versa for fast transmission over short distances, it is necessary to set the value to 500 kHz [31]. Therefore, for our work for a large-scale LoRa network, we chose a fixed setting and a value of 125 kHz.

2. Coding Rate

Coding rate is a parameter that determines the error correction coefficient in the transmitted data. In LoRa networks, 4 CR parameters are available—4/5, 4/6, 4/7 and 4/8. A higher CR provides greater protection against interference bursts, but increases ToA and energy consumption [31]. Therefore, in our work, the CR value of 4/5 was chosen.

3. Spreading Factor

Spreading factor is the degree to which data are broken down into longer symbols. LoRa has SF values between 7 and 12. In our work, the values of sf also vary from 7 to 12. A higher spreading factor increases the transmission range, but reduces the data rate and increases the time required for transmission. For example:

- SF7: Data transfer is fast, but range is limited.
- SF12: Range is increased, but data transfer is significantly slower.

4. Transmission Power

Transmission power is the signal strength with which a transmitting device in a LoRa network sends data. TP is measured in dBm and has a direct impact on the communication range, energy consumption, and overall network performance. In our simulation, we took the TP value from 2 dBm to 14 dBm with a step of 2 dBm.

3.4. Performance Metrics

To compare the performance of the proposed RL algorithm with other ADR and RL algorithms used for transmission parameter selection in LoRa communication, the following performance metrics were selected.

1. Packet Delivery Ratio

PDR is calculated with the ratio of successfully delivered packets to the total number of transmitted packets.

$$PDR = \frac{Number\ of\ successfully\ delivered\ packets}{Total\ number\ of\ packets\ sent} \tag{3}$$

PDR is used to assess the reliability of the network: the higher the PDR value, the better the network copes with data transmission.

2. EC per received packet

To calculate the EC for each packet received, you can use the following formula:

$$EC = \frac{Total\ energy}{Number\ of\ received\ packets} \tag{4}$$

where

Total energy is the total amount of energy (in joules) used to transmit packets.
Number of received packets is the total number of packets successfully received.

4. The System Model

This section provides an overview of the LoRaWAN network architecture, the simulation setup employed for evaluating the proposed model, and the ADR and RL algorithms. The network architecture outlines the essential components of the LoRaWAN protocol, while the simulation setup describes the experimental environment and the parameters utilized. The ADR and RL algorithms are discussed in terms of their role in optimizing network performance within a LoRaWAN system. Furthermore, the proposed DDQN-PER algorithm is introduced, alongside the ADR techniques, and the DQN algorithm, all of which are compared to assess their performance.

4.1. LoRaWAN Network Architecture

- In this paper, we consider a LoRa network consisting of a network server, one gateway (GW) with half-duplex operation mode and end devices (EDs) in a star topology as shown in Figure 2. All EDs belong to class A which have very low power consumption and are distributed evenly around the gateway. LoRa uses CSS modulation, which allows devices to operate at low power and withstand significant interference.
- End devices, also known as nodes, are sensors or IoT devices deployed in the area. They are responsible for collecting data and transmitting it to the network using LoRa modulation.
- Gateways act as intermediaries between end devices and the network server. Positioned within the communication range of the end devices, gateways receive uplink transmissions and forward them to the network server using high-speed backhaul communication, such as Ethernet or cellular networks.
- The network server is the central component responsible for managing the network. The network server processes data from the gateways, ensures reliable delivery to application servers, and applies error correction mechanisms. Additionally, the network server manages device authentication and communication integrity.

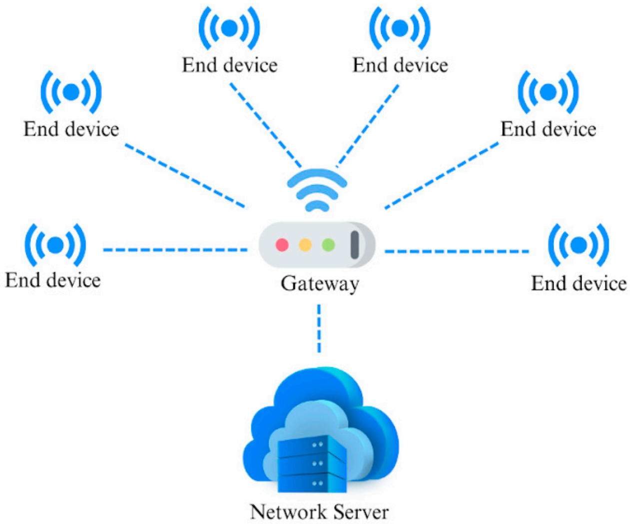

Figure 2. LoRaWAN network topology.

Figures 3 and 4 present the spatial arrangement of end nodes (blue dots) surrounding a gateway (red dot) with 1000 and 100 nodes, respectively. In both scenarios, the gateway is positioned at the center of the area, while the end nodes are continuously uniformly distributed around it.

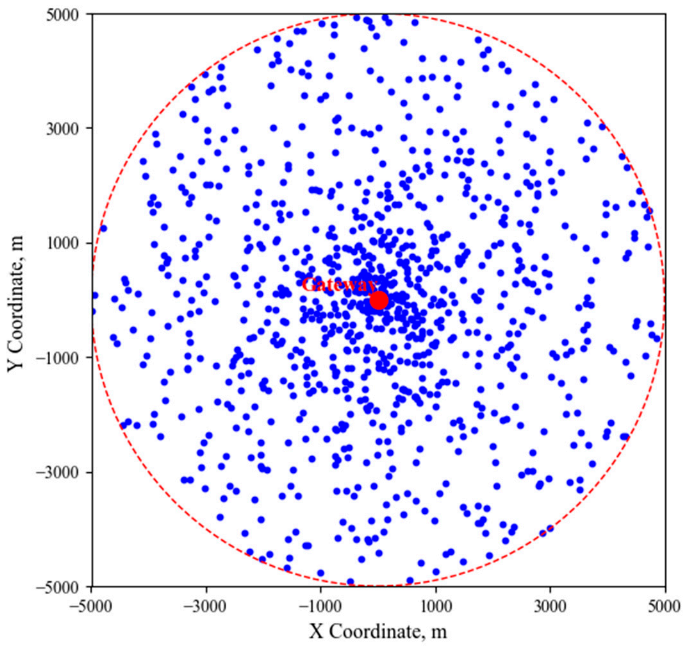

Figure 3. Continuous uniform distribution of 1000 nodes.

4.2. Simulation Setup

NS-3 with the LoRaWAN module was chosen as the simulation tool for establishing experiments in accordance with the System model from Section 4. NS-3 is a widely used network simulator that supports multiple network protocols, including LoRaWAN as shown in Figure 2. In NS-3, the LoRaWAN module allows you to simulate LoRaWAN networks, providing a framework for simulating communications between end devices, gateways, and the network server. A set of simulations in different scenarios were performed using the NS-3 LoRaWAN tool. To compare the results of the proposed algorithm with other ADR mechanisms and RL algorithms, four different scenarios changing the simulation time and the number of nodes were considered.

Scenario 1: The number of nodes varies from 10 to 100 with a step of 10. The simulation time was fixed and amounted to 1 day.
Scenario 2: The number of nodes varies from 100 to 1000 with a step of 100. The simulation time was fixed and amounted to 1 day.
Scenario 3: The number of nodes was fixed at 100. The simulation time was increased from 1 day to 7 days in 1-day increments.
Scenario 4: The number of nodes was fixed at 1000. The simulation time was increased from 1 day to 7 days in 1-day increments.

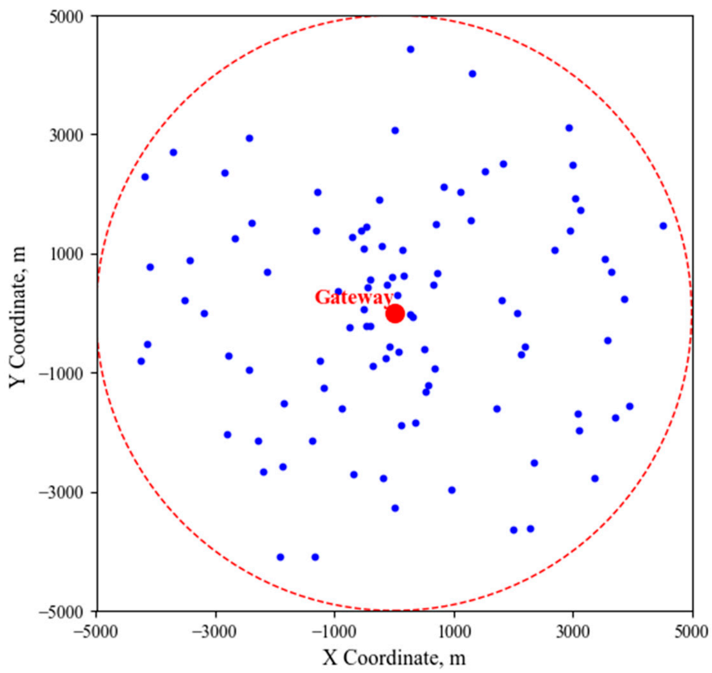

Figure 4. Continuous uniform distribution of 100 nodes.

In Table 3, we present the key simulation parameters, which were carefully selected to reflect real-world LoRaWAN deployment scenarios. The simulation time varied depending on the specific scenario, ranging from 1 to 7 days, to analyze the efficiency of the algorithms in both short-term and long-term perspectives. The radius of the simulation area was 5 km, which was chosen to approximate the typical conditions of large-scale LoRaWAN networks operating in the 868 MHz frequency band. For each scenario, NS3-LoRaWAN simulations were run twice: first in an open space environment and then with the inclusion of building obstacles. This dual approach allowed us to evaluate the adaptability of the algorithm to different environments. The number of nodes involved in the simulation also varied depending on the scenario, as mentioned earlier, to evaluate the performance of the algorithms in networks of different scales. A message inter-arrival time of 600 s was chosen to simulate the typical operating mode of low-power IoT devices, while a message size of 20 bytes reflects standard payloads. The SF range from 7 to 12 corresponds to standard LoRaWAN values, enabling the simulation of both short high-speed and long low-speed communication links. TP varied from 2 to 14 dBm to optimize the balance between energy consumption and communication range. The step size for the spreading factor is 1 (i.e., values progress as 7, 8, 9, 10, 11, 12). For the transmission power, the step size is 2 dBm (i.e., values progress as 2, 4, 6, 8, 10, 12, 14). The frequency of 868 MHz was chosen as the standard for LoRaWAN worldwide, while the bandwidth of 125 kHz ensures maximum transmission range with minimal energy consumption. Path loss model and receiver sensitivity of −137 dBm reflects real-world hardware characteristics, making the simulation more realistic. The CR was set to 4/5, which is a standard value in LoRaWAN, providing a good balance between data transmission reliability and efficient channel bandwidth utilization. In addition to the main simulations, we performed a comparative analysis of our proposed DDQN-PER algorithm with several existing ADR mechanisms, including ADR-AVG, ADR-MIN, and ADR-MAX. We also compared our proposed approach with traditional reinforcement learning algorithms such as Q-learning and Deep Q-Network. The simulation algorithms were selected to ensure an objective comparison between traditional approaches and modern reinforcement learning methods. This comprehensive comparison allowed us to evaluate the efficiency and effectiveness of our solution for selecting the optimal transmission parameter of the LoRaWAN network.

Table 3. Simulation parameters in NS-3.

Parameter	Value
Simulation algorithms	ADR-MIN, ADR-AVG, ADR-MAX, Q-learning, Deep Q-Network, DDQN-PER (proposed algorithm)
Simulation time	1 day–7 days
Simulation area	5 km radius, open area and area with obstacles
Number of nodes	[10–100 with step 10], [100–1000 with step 100]
Message inter-arrival time	600 s
Message size	20 bytes
Spreading factor (SF)	[7–12 with step size 1]
Transmission power (TP)	[2–14 with step size 2] dBm
Path loss	PL(d0) = 127.41, d0 = 40, n = 2.08, σ = 3.57
Receiver sensitivity	−137 dBm
Carrier frequency (CF)	868 MHz
Bandwidth (BW)	125 kHz
Coding rate (CR)	4/5

4.3. The ADR and RL Algorithms

1. ADR-MIN

In the paper [32], the authors present an improved ADR-MIN algorithm for selecting transmission parameters for LoRaWAN in noisy channel conditions. The ADR-MIN algorithm uses the minimum SNR value from the last 20 received packets to estimate the optimal SF and TP values. This approach focuses on the weakest signal conditions to ensure transmission stability even in high-noise conditions. The algorithm is suitable for noisy channels, but may result in excessive power consumption in conditions with good communication quality.

Workflow:

- Collect SNR for each of the last 20 packets.
- Select the minimum SNR value.
- Adjust transmission parameters: increase SF to enhance range and, if necessary, increase TP for reliability.

2. ADR-MAX

The paper by Peruzzo and Vangelista discusses an improved ADR-MAX algorithm designed to enhance power efficiency in LoRaWAN networks [33]. The ADR-MAX algorithm uses the maximum SNR value from the last 20 received packets to estimate the optimal SF and TP values. This method focuses on energy conservation, as it is based on the best signal conditions. The algorithm may be less effective in dynamic networks where signal quality degrades quickly.

Workflow:

- Collect SNR values from the last 20 packets.

- Select the maximum SNR value.
- Decrease TP or SF to minimize energy consumption while maintaining sufficient signal quality.

3. ADR-AVG

In the paper [34], Slabicki et al. propose an adaptive ADR-AVG mechanism for configuring transmission parameters in LoRaWAN networks, which improves performance and scalability in high-density environments by reducing collisions and interference. The ADR-AVG algorithm uses the average SNR value from the last 20 received packets to estimate the optimal SF and TP values. ADR-AVG is more complex to implement compared to ADR-MIN and ADR-MAX due to the need to dynamically account for network density.

Workflow:

- Collect SNR values from the last 20 packets.
- Calculate the average SNR value.
- Adjust SF and TP based on the average SNR to achieve a balance between energy consumption and reliability.

4. Q-learning

Q-learning is a reinforcement learning algorithm used to find the optimal action policy for an agent in a given environment and was first invented by Watkins, Christopher JCH and Peter Dayan [35]. In Q-learning, an agent interacts with the environment by performing actions and observing emerging states and rewards. The environment is defined by a set of states S and a set of possible actions A that the agent can take. The basic idea of Q-learning is to learn a function Q(s,a), which represents the expected utility value of performing action a in state s and then following the optimal policy. Q-learning updates the Q-values using the Bellman equation:

$$New\ Q(s,a) = Q(s,a) + \alpha\left(r + \gamma \max Q\left(s', a'\right) - Q(s,a)\right) \tag{5}$$

where

a is the learning rate;
r is the immediate reward received after taking action a;
γ is the discount factor;
s' is the new state after taking action a.

Q-learning iteratively updates the Q-values by exploring actions and observing the rewards and transitions. The agent aims to maximize its cumulative reward over time. Q-learning struggles with environments with large state-action spaces, as it requires a table to store all $Q(s,a)$ values.

5. Deep Q-Network

Deep Q-Network (DQN) is a deep learning algorithm derived from Q-learning, designed to tackle control problems in environments with high-dimensional states [36]. By integrating traditional Q-learning with neural networks, DQN enables agents to develop optimal decision-making strategies. The algorithm aims to estimate the value of actions in a given state, called the Q-value ($Q(s,a)$), using the Bellman equation to update values based on the rewards received and expected future rewards. Instead of storing a table of Q-values, DQN uses a deep neural network to approximate the Q-value function. To increase the stability of learning, DQN uses a replay buffer mechanism that preserves the agent's experience (state, action, reward, next state) and uses random samples from it for training. DQN uses two networks: the main network and the target network. The target network is updated less frequently than the main network, which contributes to the stability of Q-value updates. The agent interacts with the environment, collects experience, and periodically uses samples from the repeated buffer to update the Q-values, minimizing the RMS error between the predicted and target Q-values. DQN also uses an e-greedy strategy, which helps to find a balance between exploring new actions and using already known optimal actions.

By combining Q-learning with neural networks, DQN copes with large state spaces. The experience buffer and the target network contribute to the stability and efficiency of training. Training DQN requires significant computational resources and careful tuning of hyperparameters (learning rate, discount factor).

6. The DDQN-PER algorithm

In selecting the transmission parameters of the LoRa network, two conflicting objectives of energy minimization and PDR maximization make it difficult to choose the appropriate parameter during reinforcement learning. To address this, the DDQN-PER algorithm employs a multi-objective optimization strategy, incorporating both objectives into the reward function. This allows the algorithm to balance the trade-off between energy efficiency and reliable data transmission, ensuring optimal performance under varying conditions. Moreover, for large-scale LoRa nodes with various obstacles between nodes and the gateway will lead to congestion and deterioration of the LoRa gateway performance, and will be time-consuming and computationally expensive. To address these challenges, we propose the DDQN-PER algorithm, which effectively identifies optimal transmission parameters in diverse scenarios.

Double Deep Q-Network (DDQN) is an improvement over the traditional reinforcement learning algorithm Deep Q-Network that aims to address sequential decision-making issues [37]. DDQN addresses this problem by separating action selection and action evaluation into two stages using two separate networks. In traditional DQN, both functions—selecting the best action and evaluating the value of that action—are performed by the same network, which leads to the problem of Q-value inflation.

A target network is used to evaluate the Q-value, which computes the value Q for the selected action a in the next state s. The target value is calculated as:

$$Y_t = r + \gamma Q\left(s_{t+1}, argmax_a Q(s_{t+1}, a_{t+1}; \theta_t); \theta'_t\right) \tag{6}$$

where

- r is the reward obtained for performing action at in state st;
- γ is the discount factor reducing the weight of future rewards ($0 \leq \gamma \leq 1$);
- s_{t+1} is the next state;
- Q is Q-value of the target network
- θ_t, θ'_t are the parameters of the main and target networks, respectively.

The difference between the target and predicted Q-values is used to calculate the mean squared error loss function:

$$L(\theta_t) = E\left[\left(Y_t - Q(s_t, a_t; \theta_t)\right)^2\right] \tag{7}$$

where

- $L(\theta_t)$ is the loss function;
- Y_t is the target Q-value;
- $Q(s_t, a_t; \theta_t)$ is the predicted Q-value of the main network for current state st and action at;
- E is the expectation operator for averaging over samples.

One of the key problems with traditional DQN and other reinforcement learning methods is the overestimation of Q-values. This happens because the same Q-value is used for both action selection and evaluation. DDQN solves this problem by separating the action selection and evaluation processes using two different networks. This reduces bias and makes the learning process more stable and robust.

Thus, DDQN reduces bias and makes action estimation more accurate, which leads to more stable learning.

In standard Experience Replay, training examples are selected randomly from the agent's memory, which can be inefficient, especially in rare or critical situations. PER improves this process by giving higher priority to examples with high TD-error [38]. This means that the agent repeats important or difficult to predict transitions more often, which speeds up learning.

The TD error can be represented as follows [39]:

$$\delta = r + \gamma Q_{target}(s_{t+1}, argmax_a Q(s_t, a)) - Q(s_t, a_t) \tag{8}$$

where

- δ is the TD error, representing the discrepancy between expected and predicted Q-values;
- Q_{target} is the Q-value from the target network for the next state and the optimal action;
- r is the reward obtained for performing action at in state s_t;
- γ is the discount factor.

Prioritized Experience Replay (PER) allows the agent to pay more attention to those transitions that have a higher TD error (temporal difference error). This speeds up the learning process on critical episodes and helps to optimize the LoRa network parameters faster.

By reducing bias in action evaluation, DDQN allows the agent to better balance exploration of the environment and the use of already accumulated knowledge. In our case, this helps the agent more accurately select parameters, such as transmit power and spreading factors that improve network transmission performance. In a large-scale LoRa environment, this enables the agent to quickly and successfully find optimal network transmission parameters despite the challenges of multiple end-devices, network scale, and obstacles. Below is the algorithm of the proposed approach for clarity (Algorithm 1).

Figure 5 shows the DDQN-PER algorithm in interaction with the LoRa environment for our situation. The current network state and transmission parameters of the LoRa network are input to the DQN network. The measured SNR value is utilized to determine the current channel conditions. If the SNR is low, the algorithm may select a higher SF to increase transmission range or increase TP to strengthen the signal. If the SNR is high, the algorithm may reduce SF or TP to decrease energy consumption. The agent selects an action based on the current state. The action is a change in the transmission parameters SF and TP. The environment (LoRa network) returns a reward to the agent, which can be based on the success of the transmission.

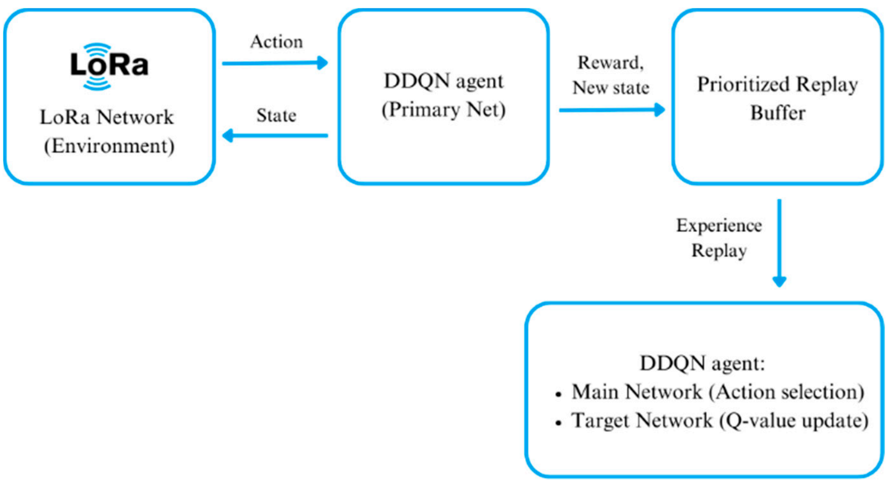

Figure 5. DDQN-PER algorithm.

All experiences (state, action, reward, next state) are stored in a prioritized replay buffer, and training is performed based on these data. The main network is trained on the selected experiences from the buffer, using the priorities of the prediction errors. The target network is updated with a fixed periodicity for the stability of the training. This process is repeated until the agent finds the optimal parameters for data transmission in the LoRa network. At the output, we obtain the optimal transmission parameters (SF, TP) for LoRa. This approach allows the agent to effectively manage the LoRa transmission parameters, adapting to changing network conditions.

Algorithm 1. Pseudocode of the DDQN-PER algorithm

Input: -Range of SF: [7 to 12 with step size 1]
-Range of TP: [2 to 14 with step size 2] dBm
-SNR: observed
-Simulation Environment Parameters: number of nodes, simulation time, simulation area radius, obstacle presence, message inter-arrival time, message size, path loss model, receiver sensitivity, CF, BW, CR
Initialization:
(a) Initialize *Q_network*, *Target_network*, and *PER_buffer* ← empty, *SNR* ← observed
(b) Set learning parameters: α, γ, ε
(c) Initialize LoRaWAN environment and start communication
repeat
if nodes generate a packet **then**
(a) Choose action *a_t* using ε-greedy policy in main network based on state *s_t = [s_SNR_t, s_TP_t, s_SF_t]*
(b) Send the packet with selected SF_i and TP_j
(c) Observe next state *s_{t+1} = [s_SNR_{t+1}, s_TP_{t+1}, s_SF_{t+1}]* and reward *R*
(d) Store *(s_t, a_t, R, s_{t+1})* in *PER_buffer*
Q-value update:
(a) Sample experiences from *PER_buffer* based on *TD error*
(b) Perform Q-value update using Formula (4).
(c) Compute the loss between the predicted and target Q-values using Formula (5)
(d) Backpropagate the loss to update main network
(e) Periodically update target network
end if
until the LoRaWAN network stops
Output: Optimal transmission parameters (SF, TP)

5. Results

This section presents the simulation results obtained under varying environmental conditions. The simulations were carried out using the NS3-LoraWAN module, focusing on analyzing key performance metrics. One of the primary observations across all scenarios was that the EC per node remained relatively stable, fluctuating within the narrow range of 0.18 to 0.19 mJ. Given the minimal variation in energy consumption, we turned our attention to comparing the PDR values across different algorithms. Below, we provide a detailed breakdown of the results for each scenario individually, offering insights into the performance of each approach under specific conditions. Also, from the figures below, it can be determined that the values of the PDR are very strongly dependent on the environmental parameters. The values of the PDR in open area are always higher by 0.4–0.5 value than in areas with obstacles.

First scenario. In the first scenario, the number of nodes was increased from 10 to 100 nodes with a step of 10. The simulation time was 1 day and was not changed throughout the first scenario. The simulation results for the first scenario can be seen in Figures 6 and 7 without obstacles and with obstacles, respectively. From the figures, it is clear that our proposed algorithm showed good results and were higher than other algorithms.

Second scenario. The second scenario is very similar to the first scenario. The main difference between the second and the first is the number of nodes. The second scenario was as close as possible to a large-scale network, the number of nodes varied from 100 to 1000 with a step of 100 nodes. Figures 8 and 9 show the simulation results for the second simulation without obstacles and with obstacles, respectively. As you can see in the figures, here too, our proposed algorithm DDQN-PER was the best among other algorithms in all numbers of nodes for both open area and area with obstacles.

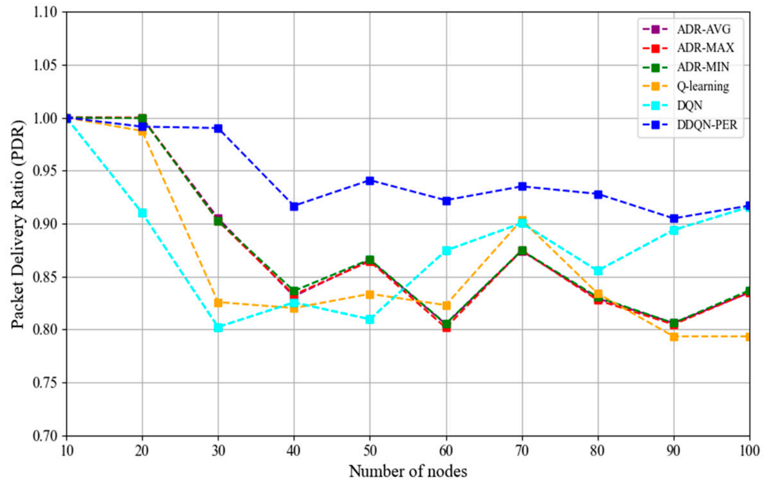

Figure 6. Simulation results in open area for 10–100 nodes.

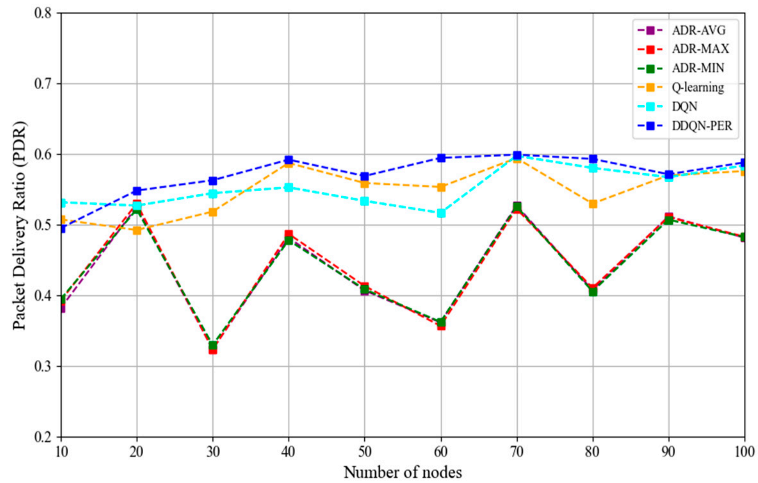

Figure 7. Simulation results in area with obstacles for 10–100 nodes.

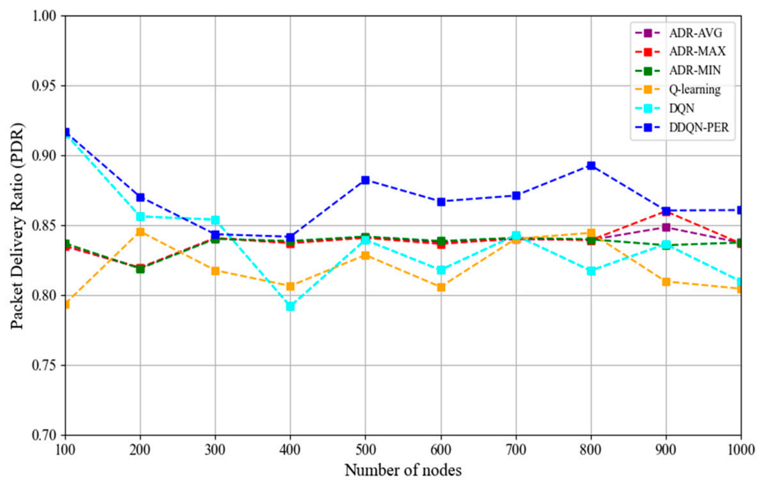

Figure 8. Simulation results in open area for 100–1000 nodes.

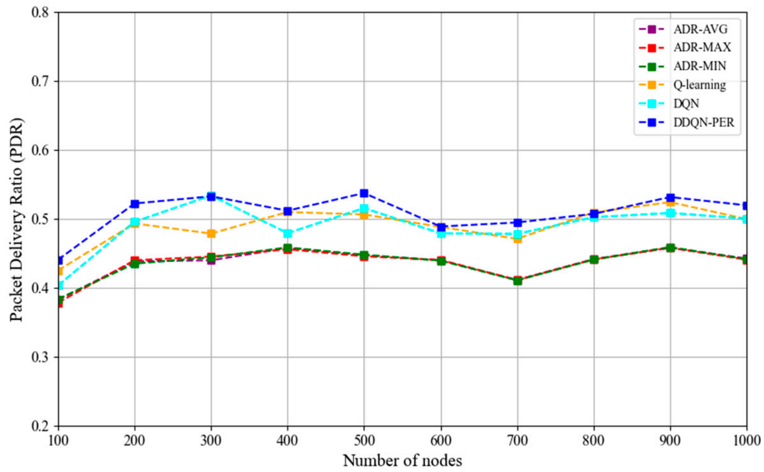

Figure 9. Simulation results in area with obstacles for 100–1000 nodes.

Third scenario. As shown in Figures 10 and 11, the PDR values for the ADR-MIN [32], ADR-MAX [33] and ADR-AVG [34] algorithms gradually increase and achieve better results than the RL algorithms with each day of simulation. The trend is typical for both open space and obstacle-ridden networks. This is explained by the fact that the ADR mechanisms adapt well with increasing time to select the optimal network transmission parameter. On one hand, this poses a challenge, as the ADR mechanism requires considerable time to adapt. While the ADR mechanism performed best after 7 days, it is important to consider the results from days 1 and 2, where our proposed DDQN-PER algorithm outperformed the others. By reducing the simulation time, we can minimize the duration required to select the optimal network transmission parameters.

Fourth scenario. In the last scenario, 1000 nodes were used for training and one to seven days of simulation time were used to select the optimal network transmission parameters. As shown in Figures 12 and 13, the ADR mechanism struggles to perform effectively in large-scale networks. In each case, our proposed algorithm demonstrated superior performance compared to the alternatives.

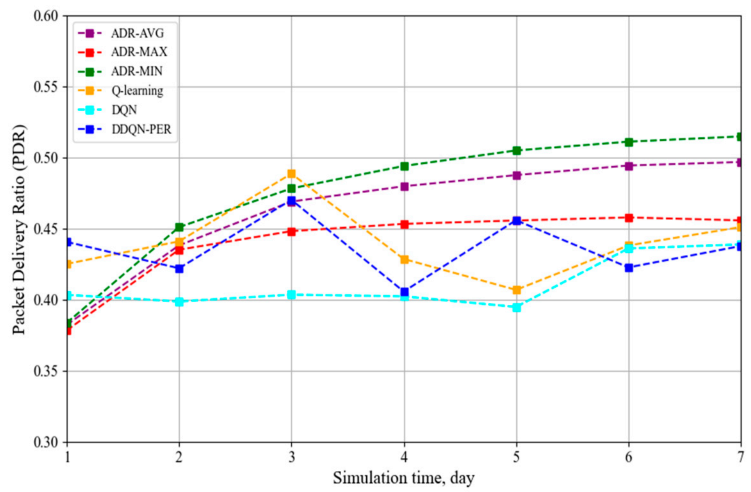

Figure 10. Simulation results for 100 nodes in open area for simulation time 1–7 days.

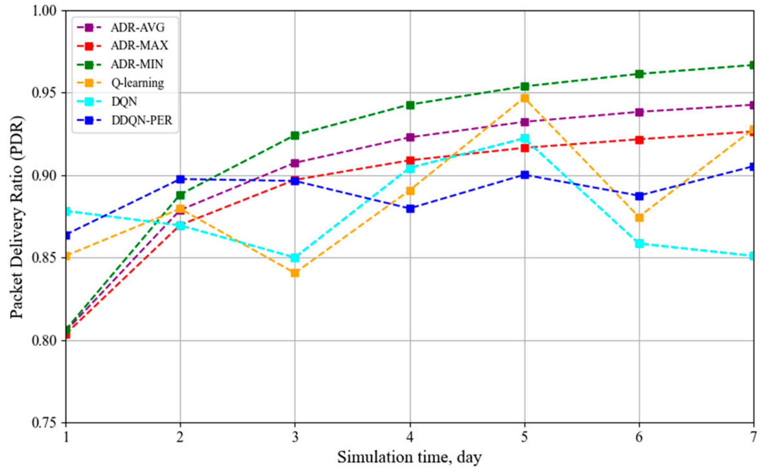

Figure 11. Simulation results for 100 nodes in an area with obstacles for simulation time 1–7 days.

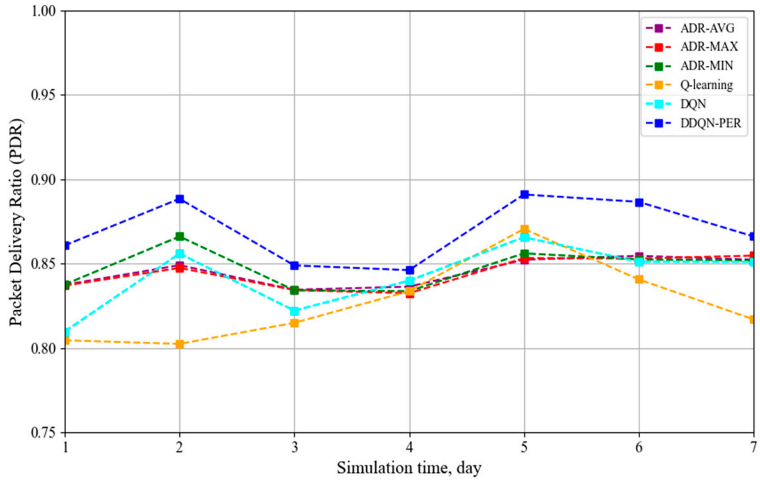

Figure 12. Simulation results for 1000 nodes in open area for simulation time 1–7 days.

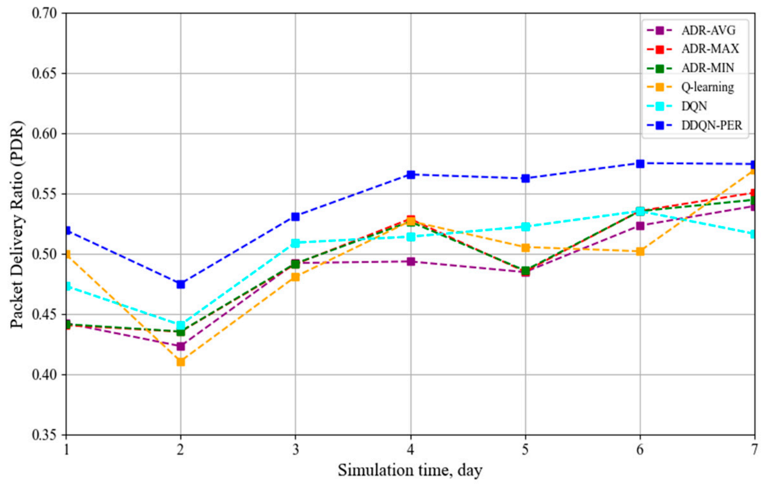

Figure 13. Simulation results for 1000 nodes in an area with obstacles for simulation time 1–7 days.

6. Discussion

Other studies demonstrate that ML, artificial intelligence, deep learning, and RL techniques can effectively optimize EC and enhance performance in LoRa networks. Network simulators are crucial in these studies, as they model complex network scenarios and evaluate the performance of proposed algorithms. For instance, the OMNeT++ simulator with the FLoRa library was employed in [19], while studies [40,41] utilized the NS-3 simulator, and [22] relied on LoraSim for simulating and testing the proposed methods. Additionally, the study in [18] implemented Lora-MAB in Python.

The proposed DDQN-PER algorithm has been extensively evaluated across various scenarios, including obstacle-prone environments, extended simulation durations, and networks with varying node densities.

Impact of obstacles: In simulations with obstacles, PDR values were consistently lower (by 0.4–0.5) compared to open spaces. This is due to increased signal loss and interference, which reduces the likelihood of successful packet delivery.

Adaptation of algorithms over time: Algorithms such as ADR-MIN, ADR-MAX, and ADR-AVG showed improvement in PDR as the simulation time increased (e.g., up to 7 days) because they require more time to adapt and select optimal transmission parameters. During the early days of the simulation, our DDQN-PER algorithm outperformed others as it adapts more quickly to changing environmental conditions by selecting optimal transmission parameters (SF, TP). However, longer simulations allowed other algorithms to reduce the performance gap.

Performance of the proposed algorithm with varying node densities: in simulations with a smaller number of nodes, the DDQN-PER algorithm consistently demonstrated better performance compared to other algorithms, maintaining a higher packet delivery ratio (PDR). As the network scaled up to 1000 nodes, the proposed algorithm continued to outperform other approaches in both open areas and obstacle-prone environments. This highlights its robustness and scalability when handling increased network densities.

For comparison, the table for our algorithm was taken with data for 1000 nodes in a space with obstacles, and for other algorithms, the maximum number of nodes that the authors considered in their simulation were taken. Table 4 summarizes the research methods, adjustable parameters, and ML accuracy from the reviewed studies. As illustrated in Table 4, our proposed algorithm demonstrated an improvement of 17.2% over the ADR. Significant progress was reached in [23], where parameters such as SF, TP, CR, and CF were analyzed using a hybrid approach of SL and RL. The study [19] utilized GRU with high training accuracy, demonstrating an 11% enhancement in PDR performance compared to ADR, although specific EC data were not provided. The main disadvantage of the method was its high resource intensity and long training time of up to 24 h, which is unacceptable for dynamic networks. For comparison, in addition to the standard ADR, we also considered specific cases where RL was utilized [40]. We compared the results with the ADR-MIN, ADR-AVG, and ADR-MAX algorithms, which use minimum, average, and maximum SNR values, respectively. The best results compared to ADRavg were achieved at high attenuation, improving EC by 7.05% and PDR by 9.09%. In the paper [41], the authors used the Multi-armed Bandits RL method, achieving a 40% improvement in EC compared to ADR. However, when experimenting with a single gateway, the PDR decreased by 22.7%, and with multiple gateways—by 6.7%. An additional disadvantage is slow convergence, especially in scenarios with a large number of nodes. The study [42] introduces a GRU-based deep learning approach for LoRaWAN resource allocation, achieving an 11% improvement in packet success ratio by predicting and assigning optimal spreading factors in real-time. The findings in [43] indicate an approximate improvement of 18% and 20% in PDR for SSFIR-ADR1 and SSFIR-ADR2, respectively, compared to the standard ADR algorithm. However, while the SSFIR-ADR algorithm achieves better PDR performance than ours, it is important to highlight that our simulation involved 1000 nodes, whereas theirs was limited to only 200 nodes, making the comparison less directly comparable. The proposed DDQN-PER algorithm showed PDR improvement in the range of 6.2–8.11% compared to other existing RL and machine-learning-based works.

The key distinction of our approach lies in its ability to optimize the balance between EC and PDR by effectively focusing on critical transitions through Prioritized Experience Replication. This significantly enhances the overall efficiency of the LoRaWAN network. We chose RL methods due to their ability to efficiently process large amounts of data and accurately predict parameters, making them ideal for the task of choosing the optimal network transmission parameters in large-scale and resource-intensive networks.

Table 4. Comparison of results with other studies.

Reference	Year	Method	Adjustable Parameters	Comparison with ADR	Simulation
Proposed algorithm	2024	RL (DDQN-PER)	SF, TP	EC-equal PDR—17.2%	NS-3 LoRaWAN
[23]	2022	Supervised ML and RL (EXP4)	SF, TP, CR, CF	no data	LoraSim
[19]	2021	Supervised ML and RL (EXP4)	SF	PDR—11%	NS-3 LoRaWAN
[40]	2022	RL (Q-Learning)	SF, TP	EC—7.05% PDR—9.09%	NS-3 LoRaWAN
[41]	2021	Multi-armed Bandits (RL)	SF	EC—40%	NS-3 LoRaWAN
[42]	2022	Gated Recurrent Unit	SF	PDR—11%	NS-3 LoRaWAN
[43]	2024	SSFIR-ADR	SF, TP	PDR-20%	NS-3 LoRaWAN

The novelty of our algorithm lies in the integration of DDQN-PER, which significantly accelerates the learning process and enhances the stability of parameter optimization. This approach provides a clear advantage in high-density networks and environments with significant interference, where traditional methods face challenges. Furthermore, our algorithm has been specifically tested in challenging scenarios, such as networks with up to 1000 nodes and the presence of obstacles, demonstrating superiority over existing methods as well as faster adaptation times, highlighting its robustness and adaptability.

Our proposed algorithm has several limitations, including scalability issues in multi-gateway networks, which require additional computational resources. The algorithm is also optimized for static nodes, and adapting it for mobile nodes will require further modifications. Additionally, the algorithm's performance in real-world conditions may be limited by differences in hardware and channel interference.

7. Conclusions

In conclusion, our work demonstrates that the Double Deep Q-Network with Prioritized Experience Replay is an effective solution for achieving a balance between EC and PDR in LoRa networks. By concentrating on critical transitions, our model accelerates the identification of optimal transmission parameters, significantly enhancing the performance of the LoRa gateway.

The proposed DDQN-PER algorithm outperformed other reinforcement learning and ADR algorithms over a 24-h period, particularly in scenarios with 1000 devices in both obstacle-laden and open environments. While ADR mechanisms showed improvements with extended simulation times for smaller networks, they struggled to replicate this success at scale; in contrast, our algorithm consistently delivered superior performance.

In the future, the algorithm will be implemented in real LoRaWAN networks to validate simulation results and assess its performance under real conditions. Additionally, the algorithm's adaptation for mobile nodes and scalability in large multi-gateway networks will be explored. Furthermore, there are plans to integrate the algorithm with other communication protocols to expand its applicability and improve efficiency.

Author Contributions: Conceptualization, B.Z. and A.B.; methodology, B.Z. and A.B.; software, E.Y. and M.N.; validation, K.K., S.O. and G.D.; formal analysis, E.Y. and A.K.; investigation, N.K.

(Nurzhigit Kuttybay) and N.K. (Nursultan Koshkarbay); writing—original draft preparation, B.Z. and A.B.; writing—review and editing, A.S. and M.N.; visualization, S.O.; supervision, A.S.; project administration, M.N. All authors have read and agreed to the published version of the manuscript.

Funding: This research has been funded by the Science Committee of the Ministry of Science and Higher Education of the Republic of Kazakhstan (Grant AP19678552).

Data Availability Statement: The original contributions presented in this study are included in this article. Further inquiries can be directed to the corresponding authors.

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

References

1. Mroue, M.; Ramadan, A.; Nasser, A.; Zaki, C. LPWAN Technologies in Smart Cities: A Comparative Analysis of LoRa, Sigfox, and LTE-M. In Proceedings of the Eighth International Conference on Information System Design and Intelligent Applications, Dubai, United Arab Emirates, 6–7 January 2024; pp. 219–231. [CrossRef]
2. Porobic, V.; Todorovic, I.; Isakov, I.; Kyslan, K.; Jerkan, D. Integrated Framework for Development, Emulation, and Testing of High-Level Converter Control Functions for Distributed Generation Sources. *IEEE Access* **2021**, *9*, 145852–145865. [CrossRef]
3. Lodhi, M.A.; Wang, L.; Farhad, A. ND-ADR. Nondestructive Adaptive Data Rate for LoRaWAN Internet of Things. *Int. J. Commun. Syst.* **2022**, *35*, e5966. [CrossRef]
4. Kufakunesu, R.; Hancke, G.; Abu-Mahfouz, A. A Fuzzy-Logic Based Adaptive Data Rate Scheme for Energy-Efficient LoRaWAN Communication. *J. Sens. Actuator Netw.* **2022**, *11*, 65. [CrossRef]
5. Ksiazek, K.; Grochla, K. Flexibility Analysis of Adaptive Data Rate Algorithm in LoRa Networks. In Proceedings of the 2021 International Wireless Communications and Mobile Computing (IWCMC), Harbin City, China, 28 June–2 July 2021; pp. 1393–1398. [CrossRef]
6. Soto-Vergel, A.; Arismendy, L.; Bornacelli-Duran, R.; Mendoza Cárdenas, C.H.; Montero-Arevalo, B.; Rivera, E.; Calle, M.; Candelo-Becerra, J.E. LoRa Performance in Industrial Environments: Analysis of Different ADR Algorithms. *IEEE Trans. Ind. Inform.* **2023**, *19*, 10501–10511. [CrossRef]
7. Ali, Z.; Qureshi, K.N.; Al-Shamayleh, A.S.; Akhunzada, A.; Raza, A.; Butt, M.F.U. Delay Optimization in LoRaWAN by Employing Adaptive Scheduling Algorithm with Unsupervised Learning. *IEEE Access* **2023**, *11*, 2545–2556. [CrossRef]
8. Farhad, A.; Pyun, J.-Y. LoRaWAN Meets ML. A Survey on Enhancing Performance with Machine Learning. *Sensors* **2023**, *23*, 6851. [CrossRef] [PubMed]
9. Jiang, Y.; Wang, M.; Wang, X. A Efficient Adaptive Data Rate Algorithm in LoRaWAN Networks: K-ADR. In Proceedings of the 24st Asia-Pacific Network Operations and Management Symposium (APNOMS), Sejong, Republic of Korea, 6–8 September 2023; pp. 183–188.
10. Park, J.; Park, K.; Bae, H.; Kim, C.K. EARN: Enhanced ADR with coding rate adaptation in LoRaWAN. *IEEE Internet Things J.* **2020**, *7*, 11873–11883. [CrossRef]
11. Moysiadis, V.; Lagkas, T.; Argyriou, V.; Sarigiannidis, A.; Moscholios, I.D.; Sarigiannidis, P. Extending ADR mechanism for LoRa enabled mobile end-devices. *Simul. Model. Pract. Theory* **2021**, *113*, 102388. [CrossRef]
12. Jouhari, M.; Saeed, N.; Alouini, M.S.; Amhoud, E.M. A survey on scalable LoRaWAN for massive IoT: Recent advances, potentials, and challenges. *IEEE Commun. Surv. Tutor.* **2023**, *25*, 1841–1876. [CrossRef]
13. Hakim, G.; Braun, R.; Lipman, J. Adapted Diffusion for Energy-Efficient Routing in Wireless Sensor Networks. *Electronics* **2024**, *13*, 2072. [CrossRef]
14. Ressi, D.; Romanello, R.; Piazza, C.; Rossi, S. AI-enhanced blockchain technology: A review of advancements and opportunities. *J. Netw. Comput. Appl.* **2024**, *225*, 103858. [CrossRef]
15. Perković, T.; Dujić Rodić, L.; Šabić, J.; Šolić, P. Machine Learning Approach towards LoRaWAN Indoor Localization. *Electronics* **2023**, *12*, 457. [CrossRef]
16. Nurgaliyev, M.; Bolatbek, A.; Zholamanov, B.; Saymbetov, A.; Kopbay, K.; Yershov, E.; Orynbassar, S.; Dosymbetova, G.; Kapparova, A.; Kuttybay, N.; et al. Machine Learning Based Localization of LoRa Mobile Wireless Nodes Using a Novel Sectorization Method. *Future Internet* **2024**, *16*, 450. [CrossRef]
17. Cuomo, F.; Garlisi, D.; Martino, A.; Martino, A. Predicting LoRaWAN Behavior: How Machine Learning Can Help. *Computers* **2020**, *9*, 60. [CrossRef]
18. Carvalho Filho, M.N.; Campista, M.E.M. Intelligent Configuration of PHY-Layer Parameters to Reduce Energy Consumption in LoRa. In Proceedings of the IEEE Latin-American Conference on Communications (LATINCOM), Rio de Janeiro, Brazil, 30 November–2 December 2022; pp. 1–6. [CrossRef]
19. Bouras, C.J.; Gkamas, A.; Salgado, S.A.K.; Papachristos, N. A comparative study of machine learning models for spreading factor selection in LoRa networks. *Int. J. Wirel. Netw. Broadband Technol. (IJWNBT)* **2021**, *10*, 100–121. [CrossRef]
20. Parween, S.; Hussain, S.Z. Efficient collision control and auction-based resource allocation mechanism in dense LoRaWAN network via TCP using DRL technique. *Int. J. Inf. Technol.* **2024**, *16*, 4039–4057. [CrossRef]

21. Alenezi, M.; Chai, K.K.; Alam, A.S.; Chen, Y.; Jimaa, S. Unsupervised learning clustering and dynamic transmission scheduling for efficient dense LoRaWAN networks. *IEEE Access* **2020**, *8*, 191495–191509. [CrossRef]
22. Asad Ullah, M.; Iqbal, J.; Hoeller, A.; Souza, R.D.; Alves, H. K-Means Spreading Factor Allocation for Large-Scale LoRa Networks. *Sensors* **2019**, *19*, 4723. [CrossRef]
23. Minhaj, S.U.; Mahmood, A.; Abedin, S.F.; Hassan, S.A.; Bhatti, M.T.; Ali, S.H.; Gidlund, M. Intelligent resource allocation in LoRaWAN using machine learning techniques. *IEEE Access* **2023**, *11*, 10092–10106. [CrossRef]
24. Chen, M.; Mokdad, L.; Charmois, C.; Ben-Othman, J.; Fourneau, J.M. An mdp model-based initial strategy prediction method for lorawan. In Proceedings of the ICC 2022-IEEE International Conference on Communications, Seoul, Republic of Korea, 16–20 May 2022; pp. 4836–4841. [CrossRef]
25. Teymuri, B.; Serati, R.; Anagnostopoulos, N.A.; Rasti, M. LP-MAB: Improving the Energy Efficiency of LoRaWAN Using a Reinforcement-Learning-Based Adaptive Configuration Algorithm. *Sensors* **2023**, *23*, 2363. [CrossRef] [PubMed]
26. Fedullo, T.; Morato, A.; Tramarin, F.; Ferrari, P.; Sisinni, E. Smart measurement systems exploiting adaptive LoRaWAN under power consumption constraints: A RL approach. In Proceedings of the 2022 IEEE International Workshop on Metrology for Industry 4.0 & IoT (MetroInd4.0&IoT), Trento, Italy, 7–9 June 2022; pp. 354–359. [CrossRef]
27. Farhad, A.; Pyun, J.Y. AI-ERA: Artificial intelligence-empowered resource allocation for LoRa-enabled IoT applications. *IEEE Trans. Ind. Inform.* **2023**, *19*, 11640–11652. [CrossRef]
28. Sornin, N.; Luis, M.; Eirich, T.; Kramp, T.; Hersent, O. *LoRaWAN 1.0 Specification*; LoRa Alliance, Inc.: Fremont, CA, USA, 2015.
29. Seller, O.; Sornin, N. Low Power Long Range Transmitter. U.S. Patent 20140219329, 2 February 2016.
30. *LoRaWAN 1.0.3 Regional Parameters*; LoRa Alliance: Fremont, CA, USA, 2018.
31. Libelium Comunicaciones Distribuidas S.L. *Waspmote LoRa 868 MHz 915 MHz SX1272 Networking Guide*; V4(2); Libelium Comunicaciones Distribuidas S.L.: Zaragoza, Spain, 2015.
32. Babaki, J.; Rasti, M.; Taskou, S.K. Improved configuration of transmission variables for LoRaWAN in high-noise channels. In Proceedings of the 2020 28th Iranian Conference on Electrical Engineering (ICEE), Tabriz, Iran, 4–6 August 2020; pp. 1–6. [CrossRef]
33. Peruzzo, A.; Vangelista, L. A power efficient adaptive data rate algorithm for LoRaWAN networks. In Proceedings of the 2018 21st International Symposium on Wireless Personal Multimedia Communications (WPMC), Chiang Rai, Thailand, 25–28 November 2018; pp. 90–94. [CrossRef]
34. Slabicki, M.; Premsankar, G.; Di Francesco, M. Adaptive configuration of LoRa networks for dense IoT deployments. In Proceedings of the NOMS 2018-2018 IEEE/IFIP Network Operations and Management Symposium, Taipei, Taiwan, 23–27 April 2018; pp. 1–9. [CrossRef]
35. Watkins, C.J.C.H.; Dayan, P. Q-learning. *Mach. Learn.* **1992**, *8*, 279–292. [CrossRef]
36. Fan, J.; Wang, Z.; Xie, Y.; Yang, Z. A theoretical analysis of deep Q-learning. *arXiv* **2020**, arXiv:1901.00137. [CrossRef]
37. Van Hasselt, H.; Guez, A.; Silver, D. Deep reinforcement learning with double q-learning. *Proc. AAAI Conf. Artif. Intell.* **2016**, *30*, 1. [CrossRef]
38. Schaul, T.; Quan, J.; Antonoglou, I.; Silver, D. Prioritized experience replay. *arXiv* **2016**, arXiv:1511.05952. [CrossRef]
39. Sharma, A.; Thangaraj, V. Intelligent service placement algorithm based on DDQN and prioritized experience replay in IoT-Fog computing environment. *Internet Things* **2024**, *25*, 101112. [CrossRef]
40. Carvalho, R.; Al-Tam, F.; Correia, N. Q-learning adr agent for lorawan optimization. In Proceedings of the 2021 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT), Bandung, Indonesia, 27–28 July 2021; pp. 104–108. [CrossRef]
41. Navas, R.E.; Dandachi, G.; Hadjadj-Aoul, Y.; Maillé, P. Energy-Aware Spreading Factor Selection in LoRaWAN Using Delayed-Feedback Bandits. In Proceedings of the 2023 IFIP Networking Conference (IFIP Networking), Barcelona, Spain, 12–15 June 2023; pp. 1–9. [CrossRef]
42. Farhad, A.; Kim, D.H.; Yoon, J.S.; Pyun, J.Y. Deep learning-based channel adaptive resource allocation in LoRaWAN. In Proceedings of the 2022 International Conference on Electronics, Information, and Communication (ICEIC), Jeju, Republic of Korea, 6–9 February 2022; pp. 1–5. [CrossRef]
43. Kufakunesu, R.; Hancke, G.P.; Abu-Mahfouz, A.M. Collision Avoidance Adaptive Data Rate Algorithm for LoRaWAN. *Future Internet* **2024**, *16*, 380. [CrossRef]