Nazarbayev University

School of Engineering and Digital Sciences

Alleviate collisions in LoRa networks using Reinforcement Learning

Kamila Salimzhanova, Timur Ismailov, Sultan Kasenov

Supervisor: Dimitrios Zorbas

April 25, 2025

Abstract

As Low Power Wide Area Networks (LPWANs) continue to expand to support the increasing demands of Internet of Things (IoT) applications, they face major limitations in terms of scalability, collision management, and network reliability. These challenges are particularly pronounced in LoRaWAN, a widely adopted LPWAN protocol that relies on ALOHA-based medium access mechanisms. As network density increases, lack of coordination in ALOHA-based transmission leads to high collision rates and decreased packet delivery performance. In this work, we propose a novel reinforcement learning (RL)-driven framework that enhances LoRaWAN performance by introducing intelligence at the edge, without requiring changes to the existing protocol stack. Our solution leverages the SARSA algorithm to enable end-devices (EDs) to autonomously learn optimal transmission slots based on their local experience. A lightweight synchronization scheme ensures that slot selection remains consistent across devices, while preserving LoRaWAN compatibility. To optimize learning behavior, we perform comprehensive hyperparameter tuning and evaluate policy generalization through transfer learning experiments. The entire framework is deployed and tested in a real-world testbed built using MicroPython on ESP32-S3, and custom network server. Experimental results show that our RL-based approach achieves over 36% improvement in Packet Delivery Ratio (PDR) compared to traditional Pure ALOHA and Slotted ALOHA methods, with only minimal energy overhead. To promote reproducibility and support future innovation in this area, we provide open-source implementations of the testbed and protocol logic.

Chapter 1

Introduction

1.1 Background

IoT (Internet of Things) refers to the concept in which low-power, resource-limited devices embedded with sensors and wireless communication are applied to collect data from physical environments in a distributed manner Vázquez-Gallego et al. (2020). Traditional medium access control (MAC) protocols, such as Aloha and Time-Slotted Aloha (TS-Aloha), are often employed in such networks. However, these protocols may face challenges in environments with a high number of devices or frequent transmission collisions. Based on their channel sharing approaches, these protocols may be classified as schedule-based or content-based strategies. These protocols allow users to initiate the transmissions without prior coordination, which, in turn, may trigger collisions and lower overall network performance Acik et al. (2023). In addition, random access MAC protocols operate without centralized coordination that permits end devices to transmit based on contention. While this may alleviate system complexity, it also enhances the risk of packet collisions, which leads to congestion as well as reduced reliability and energy efficiency Vázquez-Gallego et al. (2020). Recent studies depict the application of reinforcement learning in ALOHA-based MAC protocols to enhance the packet transmission efficiency by allowing each node to learn independently and optimize its transmission slots using a "trial-and-error" approach. Thus, it mitigates the probability of collisions in a dense network. Some studies claim that this learning-based strategy demonstrated strong performance improvements across different network topologies Acik et al. (2023).

The project investigates the application of the State-Action-Reward-State-Action (SARSA) reinforcement learning algorithm within a deterministic mesh LoRa (Long-Range) network so that communication between end devices and a gateway would enhance its efficiency through improved packet delivery and reception rates, specifically packet delivery rate (PDR) and packet reception rate (PRR). LoRa, a low-power wide-area network (LPWAN) technology, is commonly applied in IoT environments, where energy efficiency and reliable data transmission are paramount.

While initial experiments were carried out using the ns-3 network simulator, this project extends the work into a real-world testbed environment to evaluate the practical viability of the SARSA-based approach. Key metrics include PDR, packet PRR, and energy consumption. One of the anticipated challenges is managing channel access effectively in the presence of potential data collisions. Additionally, because SARSA requires continuous Q-value updates after each transmission, the algorithm is expected to incur a higher energy cost compared to simpler, non-adaptive protocols. Ultimately, the project aims to demonstrate that SARSA can serve as a robust and intelligent MAC layer alternative capable of improving communication reliability in mesh LoRa networks.

In addition to implementing the SARSA algorithm, we applied transfer learning to enhance training efficiency and convergence speed by leveraging knowledge from prior simulations. Transfer learning has emerged as a valuable approach for new devices in IoT networks to adapt faster, especially when it is energy-constrained environments. In one of the studies, this approach demonstrated improved convergence speed, reducing training time, a critical advantage in large IoT environments where new devices continuously appear Yılmaz et al. (2021). A systematic evaluation of hyperparameters — including learning rate and epsilon — was conducted to optimize the performance of the reinforcement learning model.

1.2 Problem statement

The rapid growth of the Internet of Things (IoT) has increased the demand for efficient, scalable, and energy-optimized wireless communication protocols. LoRA, as a leading LPWAN technology, is widely used for its long-range and energy-efficient communication. However, the medium access control (MAC) layer of LoRa networks has a small bottleneck due to its vulnerability to collisions and ineffective channel access in dense environments. Since devices using this layer protocol, such as Aloha or time-slotted Aloha, transmit data without prior coordination, some packets may be lost due to an overlap. This leads to an adverse effect on the PDR. This issue is further exaggerated by variability in channel conditions, for instance, fading, and by the lack of acknowledgements in LoRaWAN's class A communications, devices are often unaware of transmission failures. In theory, ALOHA's maximum channel utilization is eighteen and four percent, making it inefficient for high-density networks, particularly without redundancy or error correction mechanisms Heusse et al. (2023).

Hence, traditional MAC protocols are simple and easy to implement, but suffer from frequent packet collisions and lack adaptability. These limitations hinder network scalability, reduce the above-mentioned metrics, PDR and PRR. Besides collisions, energy consumption is also increased due to retransmissions and idle listening.

The investigated problem centers on improving channel access and transmission efficiency in deterministic mesh LoRa networks by incorporating the SARSA reinforcement learning (RL) algorithms. Contrary to MAC approaches, SARSA enables dynamic decision-making based on the rewards and states, which allows end-devices to learn and select optimal transmission slots adaptively. However, this approach also introduces challenges, including computational complexity and energy overhead due to continuous Q-value updates.

Thus, the central research problem is: How can the SARSA-based reinforcement learning approach be implemented effectively in LoRa MAC protocols to enhance data transmission metrics and network performance, while maintaining energy efficiency and lowering as much computational complexity as possible?

Moreover, the paper investigates the impact of key SARSA hyperparameters - discount factor, learning rate, exploration rate, and reward - on the PDR and PRR within LoRa networks.

Additionally, we explore the integration of transfer learning to improve learning efficiency. By transferring knowledge from previously trained environments or scenarios (e.g., different node densities or SF configurations), we aim to accelerate convergence and reduce the training overhead in new but related LoRa deployment contexts. This is particularly important for real-world deployments where training from scratch in every new scenario is not feasible due to time and energy constraints.

1.3 Aims and objectives

Aims: The primary aim of this project is to design and implement a SARSA-based reinforcement learning mechanism within LoRa MAC protocols to improve packet transmission reliability and overall network performance, while maintaining energy efficiency. The project also aims to explore the adaptability of such learning models to real-world deployment scenarios by leveraging transfer learning strategies.

Objectives:

1. Implement a SARSA-based learning end-node for dynamic adaptation to varying network conditions due to improved decision-making in the LoRa MAC protocol.
2. Investigate the influence of SARSA hyperparameters (discount factor, learning rate, exploration rate) on network performance metrics such as PDR, PRR, and convergence time.
3. Develop a testbed environment that mimics realistic LoRa network scenarios with multiple end devices and varying transmission conditions.
4. Compare the SARSA-based approach with baseline MAC strategies (e.g., Aloha, Time-slotted Aloha) to validate its advantages and limitations in practical settings.
5. Implement transfer learning to improve the convergence rate of newly joined devices and document PDR growth.

1.4 Solution approach

This project adopts a reinforcement learning (RL) methodology, specifically the SARSA algorithm, to enhance dynamic decision-making within LoRa MAC protocols. The approach involves both simulation and practical testing to evaluate the algorithm's performance across different conditions. The proposed methodology aims to achieve improved packet delivery and reception reliability while maintaining energy efficiency and adaptability.

1.4.1 Reinforcement Learning Integration with LoRa MAC

A SARSA algorithm is integrated into the MAC layer of LoRa end-devices. Each end-device functions as an autonomous agent that learns optimal transmission behavior based on observed network feedback such as packet success/failure and channel conditions. The SARSA model is trained to maximize a reward signal, which is a function of successful transmissions and energy consumption.

Key hyperparameters such as the learning rate, discount factor, and exploration rate are tuned experimentally. These values are critical for convergence behavior and effectiveness, and their impact on performance metrics like PDR , PRR, and convergence time are thoroughly studied.

1.4.2 Simulation Environment and Comparisons with Other MAC protocols

A simulation environment is developed to emulate realistic LoRa network conditions using multiple end-devices with varied transmission parameters. Baseline MAC protocols, such as pure Aloha and TS-Aloha, are implemented to provide comparative metrics (PDR and PRR) for SARSA-based decision-making.

Performance evaluation focuses on:

- Packet delivery/reception ratio improvements
- Transmission success under congestion
- Energy consumption per node

These comparisons allow a quantifiable assessment of the benefits and trade-offs introduced by the SARSA integration.

Transfer Learning for Cold-Start Scenarios

To address the issue of slow learning for newly joined devices, transfer learning is employed. Pretrained models are reused and fine-tuned on new devices, significantly reducing the time required to reach optimal behavior. The effect of transfer learning on convergence speed and early-stage PDR is measured and documented.

1.5 Summary of contributions and achievements

This project contributes to the enhancement of LoRa MAC protocols by integrating reinforcement learning techniques, specifically the SARSA algorithm, to improve network performance in dynamic and constrained environments. The following key contributions and achievements have been made:

1. This project contributes to the enhancement of LoRa MAC protocols by integrating reinforcement learning techniques, specifically the SARSA algorithm, to improve network performance in dynamic and constrained environments. The following key contributions and achievements have been made:
    - Developed a customized MAC protocol using the SARSA algorithm that enables end-devices to learn and adapt their transmission behavior based on feedback from the environment.
    - The solution allows dynamic decision-making in response to changing network conditions, reducing collisions and improving packet success rates.
2. Conducted an extensive analysis of how key SARSA hyperparameters—discount factor, learning rate, and exploration rate — impact network performance metrics such as (PDR, PRR, and convergence time.
3. Benchmarked the SARSA-based approach against standard MAC strategies such as pure ALOHA and Slotted ALOHA, showing consistent improvements in PDR and PRR under medium to high traffic scenarios.
4. Successfully integrated transfer learning to accelerate the convergence of newly introduced end-devices.

1.6 Organization of the report

This report is organized into seven chapters, each detailing a critical aspect of the research process and findings.

Chapter 1 introduces the problem statement, research motivation, and objectives. It sets the context for the study by describing the need for intelligent MAC protocol design in LoRa networks using reinforcement learning.

Chapter 2 presents the literature review, examining prior works on LoRa MAC protocols, reinforcement learning applications, and SARSA algorithm fundamentals, and the usage of transfer learning in IoT environments.

Chapter 3 discusses the architecture, including the design of the SARSA-based MAC protocol, hyperparameter tuning strategies, and the simulation setup. It also describes the integration of transfer learning to enhance real-world scalability.

Chapter 4 analyzes the experimental results, comparing performance metrics such as PDR, PRR, convergence time, and energy efficiency across different approaches and hyperparameter settings.

Chapter 5 interprets the key findings, highlighting their implications, limitations, and the practicality of using reinforcement learning in constrained wireless networks.

Chapter 6 concludes the report with a summary of contributions and future directions, including the potential for expanding the model to support other learning algorithms or MAC strategies.

Chapter 2

Literature Review

This chapter presents a comprehensive review of existing literature relevant to the development of MAC protocols in LoRa networks, reinforcement learning applications in wireless systems, and the use of transfer learning in intelligent communication. Identifies the research gaps that motivate the current project and establishes the foundation upon which the proposed solution is built.

2.1 State-of-the-Art in LoRa MAC Protocols

LoRaWAN, a widely adopted low-power wide-area network (LPWAN) protocol, is used for long-range, energy-efficient communication in IoT. Its MAC layer facilitates support for a variety of device classes, where class A is the most relevant for energy-limited devices. While the simplicity of the Aloha protocol is suitable for low-traffic scenarios, increased device density may introduce challenges in terms of collisions. To address these issues, several enhancements and alternative MAC strategies have been proposed. Some studies suggest that as network traffic increases, the frequency of collision rates also rises, leading to packet loss and delays Tsakmakis et al. (2022).

Time-slotted ALOHA (TS-ALOHA), in particular, has been widely studied as a potential enhancement over the traditional pure ALOHA protocol for LoRaWAN Tsakmakis et al. (2022). Unlike Pure Aloha, this protocol organizes the communication into channels of one frame. End devices are allowed to transmit at the start of a chosen time slot, which significantly reduces the probability of packet losses due to overlapping transmissions. The chief operating principle behind TS-ALOHA lies in synchronization Polonelli et al. (2019). By aligning transitions with specific time intervals, the system ensures that two or more packets may collide if they are transmitted at the same time slot rather than overlapping at any time. This difference from Pure Aloha doubles the maximum throughput and reduces collision rates. While time-slotted ALOHA addresses packet collision through better temporal coordination, it does not eliminate contention entirely.

As it is mentioned above, in pure ALOHA, devices transmit data without sensing the channel, leading to a high probability of packet collisions, particularly as network density increases. This uncoordinated approach results in decreased Packet Delivery Ratio (PDR) and overall network throughput. Another drawback of the Aloha and TS-Aloha protocols is the scalability changes, due to network congestion and reduced efficiency in the presence of a large number of devices. These challenges underscore the need for more adaptive and intelligent MAC strategies, such as SARSA (State-Action-Reward-State-Action). This reinforcement learning algorithm offers the potential to dynamically adjust transmission parameters based on real-time network feedback.

2.2 Reinforcement Learning and SARSA algorithm

Reinforcement learning (RL) has appeared as a powerful tool to address the dynamic nature of wireless networks. Traditional MAC protocols often rely on static configurations, which may not adapt to changing environments. On contrary, RL enables devices to learn through interactions with their environment, making it suitable for dynamic decision-making in real time Prakash et al. (2023). Using reinforcement learning algorithms, for instance, SARSA for scheduling in LoRa networks, improves data transmission metrics, that is twenty percent better compared to Aloha and TS-Aloha Baimukhanov et al. (2024). In several studies, RL-based scheduling approaches have also shown considerable promise in optimizing spreading factor (SF) allocation, with one implementation yielding a 147 percent increase in throughput at the end nodes Sandoval et al. (2019). These findings, however, were primarily derived from simulation environments.

Additional research has explored the application of supervised machine learning techniques in high-density IoT deployments, revealing that such methods can significantly enhance energy efficiency and maintain packet reception ratios (PRR) of at least 95 percent, even under heavy network congestion Minhaj et al. (2023). Similarly, machine and deep learning approaches have proven effective in refining localization accuracy in energy-constrained IoT settings Ouameur et al. (2020).

Efficient SF assignment plays a crucial role in network performance: lower SF values enable faster data transmission, which can reduce latency and channel contention. Real-world experimental results suggest that RL-driven SF selection minimizes packet collisions and increases throughput Hong et al. (2023). Beyond performance metrics, RL-based approaches also offer advantages in terms of reduced complexity and enhanced reliability for gateway communication.

Overall, reinforcement learning has proven to be a promising approach for optimizing network performance in LoRa and IoT networks. By enabling dynamic decision-making in real time, RL algorithms, particularly SARSA, offer significant improvements in key metrics like throughput, energy efficiency, and packet reception ratios.

2.3 Transfer Learning in IoT

Transfer learning is a method that involves leveraging the knowledge gained from one task or domain and using it to improve performance in a different, but related, task or domain. Transfer learning helps to shorten the time required to learn a new task or domain, and it is anticipated to enhance both the initial and final performance of the model in the new task or domain when compared to learning without transfer Yılmaz et al. (2021). Transfer learning enhances the accuracy and scalability of model, specifically in dynamic environments like smart cities Ahmed et al. (2024). Transfer learning can be particularly useful in areas such as smart city infrastructure, healthcare, and security, where models need to adapt quickly to changing conditions or new devices. However, challenges such as data privacy must be addressed to maximize its potential in real-world IoT applications.

2.4 Relevance to This Project

This project focuses on improving the efficiency and performance of LoRa networks through advanced MAC protocols and a reinforcement learning algorithm. Traditional protocols such as Aloha and Time-Slotted Aloha are increasingly insufficient as the number of devices in the network grows, leading to issues such as packet collisions and reduced scalability. The

incorporation of reinforcement learning, particularly the SARSA algorithm, addresses these challenges by allowing the network to adapt dynamically to real-time conditions, improving metrics such as throughput and packet reception ratio. Furthermore, the potential use of transfer learning offers an exciting opportunity to enhance the adaptability and performance of models in diverse IoT environments, such as smart cities and healthcare applications. By integrating SARSA and transfer learning, this project aims to develop a more robust, adaptive, and scalable solution for LoRa networks, reducing the challenges associated with conventional MAC protocols and enabling the efficient management of high-density networks.

2.5 Summary

In conclusion, this chapter has provided an overview of key literature that is MAC protocols in LoRa networks, the application of reinforcement learning (RL) in wireless communication, and the use of transfer learning (TL) for enhancing network performance. Traditional MAC protocols, like Aloha and Time-Slotted Aloha, have limitations in handling network congestion and device scalability, which can lead to packet loss and reduced throughput. The SARSA algorithm, a reinforcement learning approach, has shown significant improvements in optimizing network performance by enabling real-time dynamic decision-making. Additionally, transfer learning has been highlighted as a promising technique for improving model performance in dynamic IoT environments, where a newly joined device learns from the existing devices. These advancements in machine learning, combined with the challenges identified in traditional protocols, lay the foundation for this project's exploration of intelligent and adaptive solutions for LoRa networks.

Chapter 3

Project Architecutre

This chapter presents the design and architecture of our experimental testbed, which serves as the foundation for evaluating the proposed reinforcement learning framework under real-world LoRaWAN conditions. We describe the hardware and software components used, including end-devices based on ESP32 and Raspberry Pi. Emphasis is placed on the modularity and scalability of the testbed, which allows for flexible configuration and deployment in various network scenarios. Additionally, we detail the implementation of the synchronization mechanism, the lightweight RL agent embedded on each device, and the communication protocols used to ensure low-latency, energy-efficient operation. This chapter also outlines the development of supporting tools for data logging, monitoring, and reproducible experimentation.

3.1 Testbed Physical Design

Figure 3.1 illustrates the architecture of our experimental testbed, developed to evaluate the proposed reinforcement learning framework in a controlled yet realistic LoRaWAN setting. The testbed comprises fifteen ESP32-S3 microcontrollers acting as end-devices (EDs), a single LoRa gateway, and a Raspberry Pi (RPi) serving as the central server. Communication between the server and EDs is conducted over Wi-Fi, while peer-to-peer communication among devices and with the gateway occurs via LoRa using the SX126X transceiver module.

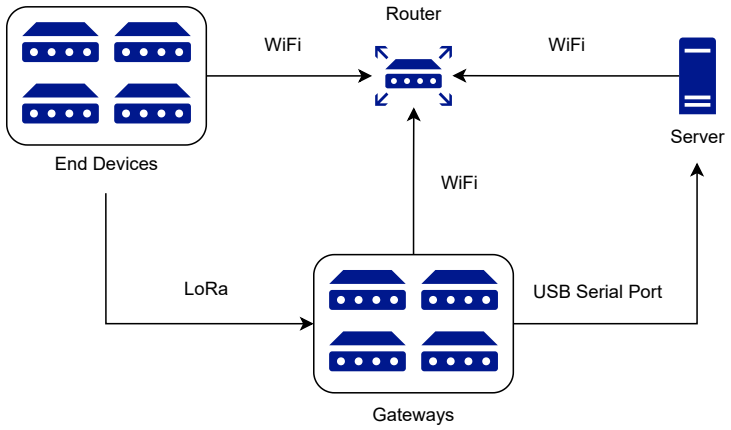
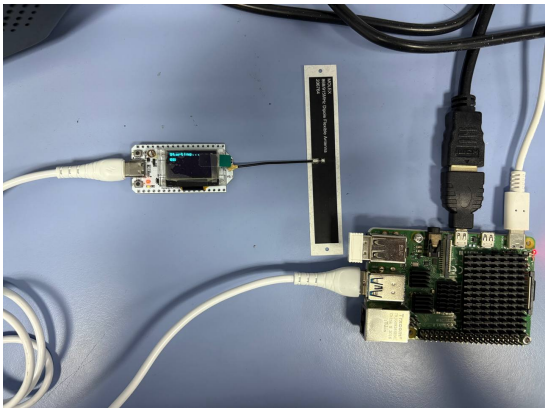
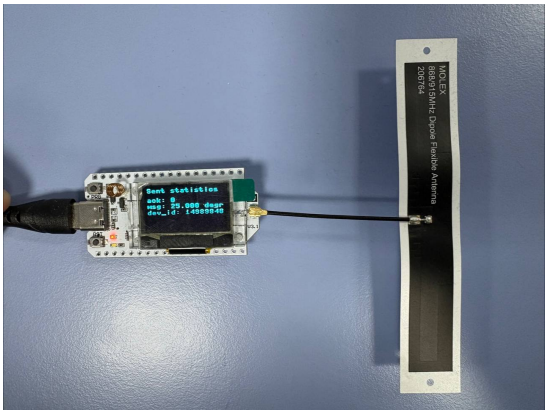


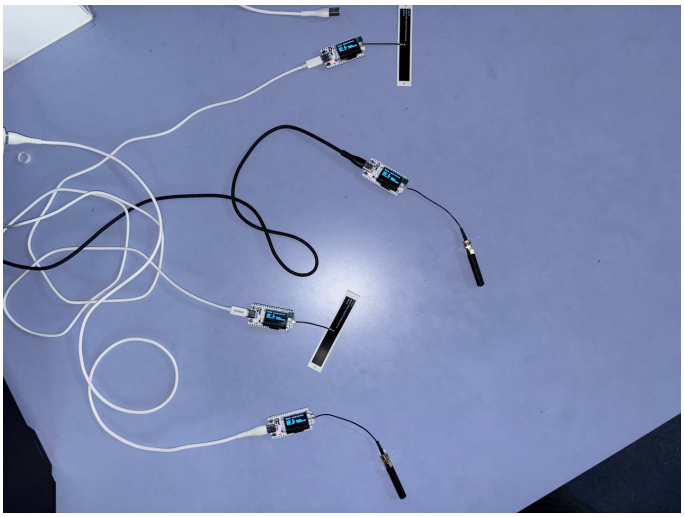
Figure 3.1: Testbed architecture



(a) Gateway device connected to a Raspberry PI server over serial port



(b) A single end device after sending its statistics to the server



(c) Multiple end devices

Figure 3.2: Photos of devices

3.2 Testbed Experiments Flow

The experimental workflow is structured into three distinct phases, as illustrated in Figure 3.3. The first phase, Start, involves configuring the environment by assigning static IP addresses, uploading the required firmware to the devices, and initializing all system processes. In the second phase, Setup, the network server (NS) broadcasts the experiment parameters to each end-device. Upon receiving these parameters, the devices begin transmitting data over the LoRa network to the gateway (GW). The third phase, Experiment, is triggered once data transmission is underway; the NS monitors for acknowledgment (ACK) messages from all devices, indicating that they have successfully completed their LoRa transmissions. After receiving ACKs from all devices, the server then computes key performance metrics, including PDR and PRR, and logs the results for further analysis. Then the system transitions back to the Start phase, completing one full experimental cycle. We used a Raspberry PI as a network server (Figure 3.2a) and 16 ESP32 microcontrollers as end devices and a gateway (Figure 3.2).

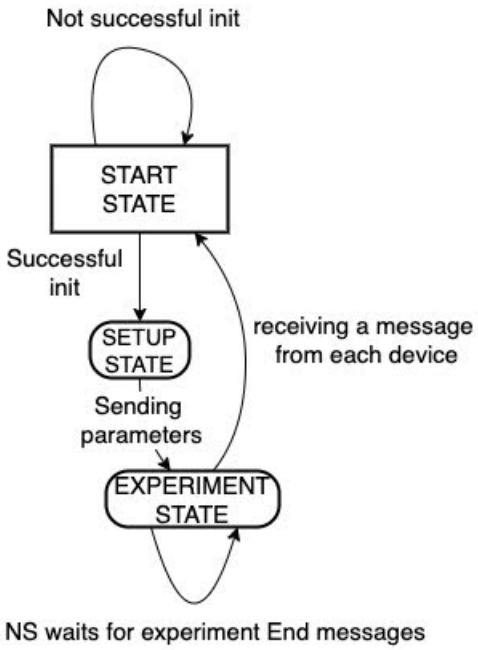


Figure 3.3: Abstarct experimental workflow

3.3 Reinforcement Learning Framework

3.3.1 Base Model

The reinforcement learning (RL) model deployed in this work is designed to enable LoRaWAN end-devices to autonomously learn optimal transmission slots in a dynamic, uncoordinated wireless environment. The RL agent interacts with its environment by observing transmission outcomes and selecting time slots in which to transmit, with the objective of maximizing the long-term PDR. We adopt the SARSA algorithm as the learning strategy, which incrementally updates Q-values for state-action pairs using feedback from each interaction.

At each decision step, the agent selects an action a in a given state s, receives a reward $r(s, a)$, and observes the next state s'. The Q-value update follows the SARSA rule:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[R_t + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t) \right], \tag{3.1}$$

where α is the learning rate, determining how much new information overrides previous estimates, and γ is the discount factor, which controls the importance of future rewards.

In our implementation, the state is represented by the history of recent transmission outcomes (success/failure) within a sliding time window. The action corresponds to a selection of a transmission slot within a fixed-length frame. The reward is defined as a binary value: 3 for a successful transmission (i.e., ACK received from the gateway), and -1 otherwise. This simple yet effective reward structure encourages the agent to find low-collision slots over time.

Next, Listing 3.1 presents the main implementation of the `update_q_value` method, a core component of the SARSA learning process responsible for updating the Q-value associated with a specific action.

```
1     def update_q_value(self, rl_state, slot, reward, next_slot):
2
3         # remove current slot
4         self.q_table[rl_state].remove(slot.num)
5         td_target = reward + self.gamma * next_slot.q_value
```

```
6           td_error = td_target - slot.q_value
7
8           # update the slot q-value
9           slot.q_value += self.learning_rate * td_error
10
11          # return slot to table
12          self.q_table[rl_state].insert(slot.num, slot.q_value)
```

Listing 3.1: Q-table Update Function

3.3.2 Heap and Hashmap Structures

One of the core operations in reinforcement learning is selecting the action with the highest Q-value for a given state. In naive implementations using arrays or dictionaries, this requires a linear scan through all possible actions to identify the maximum value, resulting in a time complexity of $O(n)$ per decision. While this is manageable for small state-action spaces, it becomes increasingly inefficient as the number of slots or actions grows.

To address this challenge, we employ a **max-heap** data structure, which allows for constant-time ($O(1)$) retrieval of the maximum Q-value and logarithmic-time ($O(\log n)$) insertions and updates. This is particularly beneficial in real-time systems such as embedded IoT devices, where computational efficiency is crucial. We tested the implementation with different Q-table sizes and the results demonstrated a logarithmic pattern (see Fig. 3.4) in performance.

However, heaps alone do not provide efficient access for arbitrary updates or removals of Q-values associated with specific state-action pairs—operations that are essential during the SARSA learning updates. To overcome this, we integrate a **hashmap** alongside the heap.

The hashmap maps each (s, a) pair to its corresponding index in the heap, enabling:

- **Constant-time ($O(1)$) lookups and deletions** of specific Q-values.
- **Efficient Q-table updates** without needing to search the heap linearly.
- **Fast policy evaluation** by allowing the agent to quickly retrieve and update its best-known actions.

This hybrid heap-hashmap approach combines the best of both worlds: the priority queue behavior of a heap for fast maximum Q-value access and the direct access efficiency of a hashmap for dynamic updates. Together, they ensure that the RL agent can make quick decisions and perform timely learning updates, even under the constraints of low-power microcontrollers.

3.4 Tuning SARSA hyperparameters

The main goal of this project is to create a package scheduling protocol that is more effective than the conventional approach. Therefore, it is important to maximize the performance of our algorithm. According to (3.1), there are 3 controlled parameters that influence the behavior of the algorithm, as well as the exploration rate parameter that we added for our implementation.

- **Learning rate (α)** - Determines how much the new information overwrites the previous one.
- **Discount factor (γ)** - Determines how much influence a new experience has.

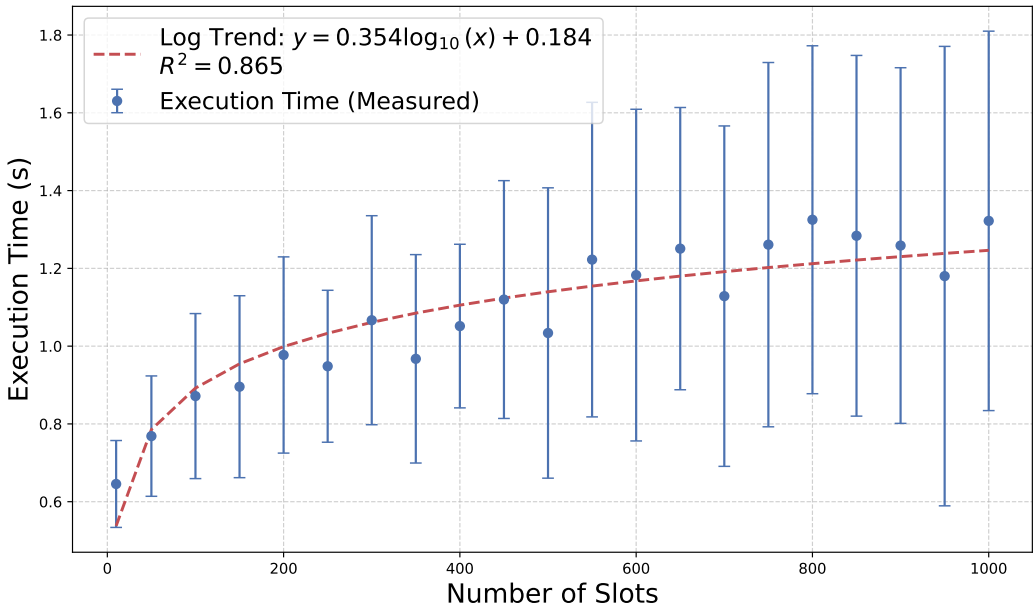


Figure 3.4: A

- **Reward (R_t)** - Reward for the resulting state after taking a certain action.
- **Exploration rate** - A coefficient that represents the chance that the end device ignores the SARSA algorithm and chooses a random slot. It is done to minimize the risk of being stuck in a local maxima/minima. This parameter is reduced to zero over time.

In order to find the most optimal values, we had to iterate through each parameter and see how they influence the performance of our system. We used PDR as a measure of performance and utilized the testbed's server capabilities to send the SARSA parameter through WiFi to iterate through each parameter.

```
    e_rate = 0
    while e_rate <= 1:
        for i in range(5):
            print(i, ". testing exploration rate at: ", e_rate)
            init_experiment(e_rate, 0.1, 0.9, -3)
            statistics_handler()
            init = random.randint(1, 65535)
        e_rate += 0.1
    l_rate = 0.1
    while l_rate <= 1:
        for i in range(5):
            print(i, ". testing learning rate at: ", l_rate)
            init_experiment(0.2, l_rate, 0.9, -3)
            statistics_handler()
            init = random.randint(1, 65535)
        l_rate += 0.1
    gamma = 0
    while gamma <= 1:
        for i in range(5):
            print(i, ". testing gamma at: ", gamma)
            init_experiment(0.2, 0.1, gamma, -3)
            statistics_handler()
            init = random.randint(1, 65535)
```

```
24          gamma += 0.1
25      no_ack_reward = -1
26      while no_ack_reward >= -15:
27          for i in range(5):
28              print(i, ". testing no acknowledgement reward at: ",
    now_ack_reward)
29              init_experiment(0.2, 0.1, 0.9, no_ack_reward)
30              statistics_handler()
31              init = random.randint(1, 65535)
32              no_ack_reward -= 2
```

Listing 3.2: Server iterates through each parameter

All the parameters except for reward are ranged from 0 to 1. For iterating through reward values we chose to only change the negative reward for failing to send a package. We did this because the actual values of positive and negative rewards do not matter as much as their proportion to one another. The behavior of SARSA is mainly influenced by the difference between how good the reward and how harsh the punishment is.

3.5 Transfer learning

At the start of each experiment each device starts with no knowledge about it's environment. They probe one slot every frame and, over time, learn which slots they can occupy (i.e. converge). However, in the real world, there would be situations where a new device could be introduced into a network of other devices that have already converged. It would be more efficient if older devices could share their knowledge of the network to the new device so that it could converge faster.

We created a rudimentary implementation of a transfer learning mechanism. We dedicated one device to be the one that receives knowledge from other devices. Other devices package their own q-tables and send them to the new device. All of the devices have already converged, so at the time of sending their q-tables they already have a preferred slot in which they send their package. Therefore, upon receiving a copy of a q-table, the new device sets the highest q-value in the table to zero so that it does not try to send a package to an occupied slot. After receiving all q-tables, the new device averages them and sets the result as its own q-table.

The server was set up to start an experiment on a set of devices. After 100 frames, the devices have already converged, and the server introduces a new device and sends a signal to old devices to share their knowledge with it. We then compare how fast the old devices converge in the first 100 frames with how fast the new device converges.

Chapter 4

Results

This section presents the experimental results obtained from evaluating our proposed reinforcement learning-based collision mitigation framework in a real-world LoRaWAN testbed. We compare the performance of our SARSA-based approach against baseline protocols—Pure ALOHA and Slotted ALOHA—using key metrics such as Packet Delivery Ratio (PDR) and Packet Reception Rate (PRR). The experiments were conducted under varying network conditions to assess the adaptability, scalability, and effectiveness of the learning algorithm. Additionally, we analyze the impact of hyperparameter tuning and transfer learning on overall performance.

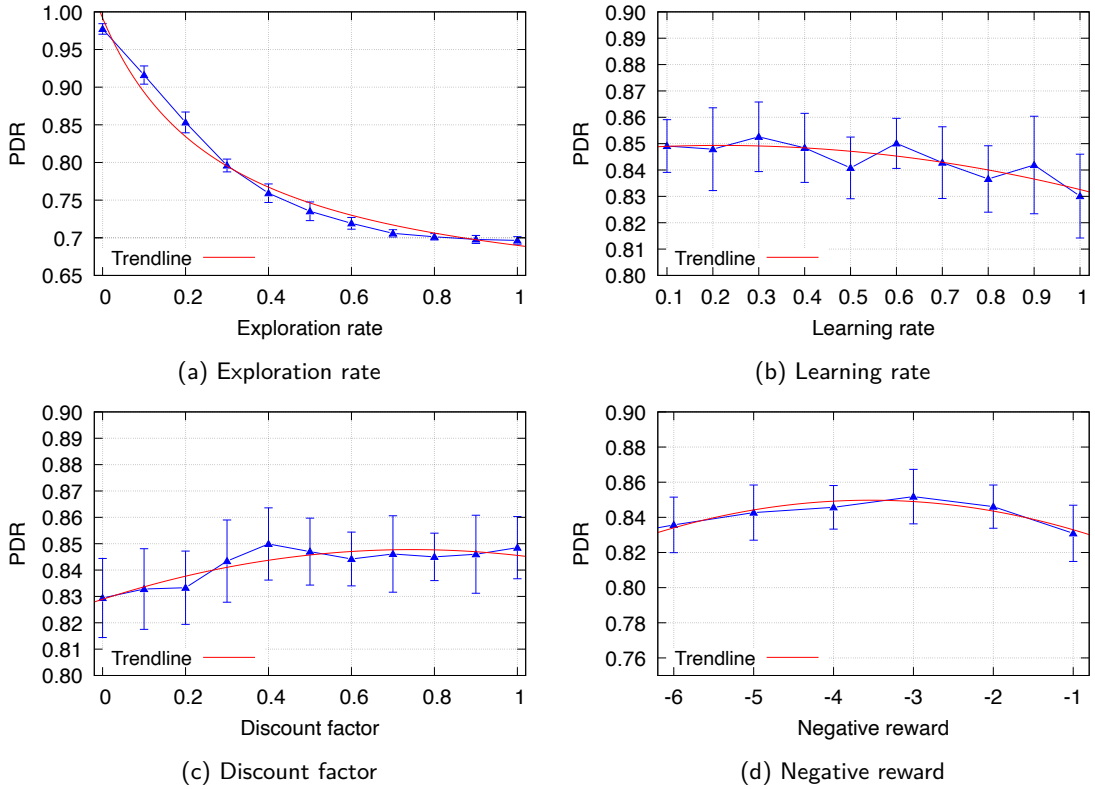
4.1 Parameter tuning

Figure 4.1 shows the effect of four hyperparameters on the average Packet Delivery Ratio (PDR). Each subplot includes error bars that indicate the standard deviation across multiple trials. In the case of ϵ, we observe a sharp decline in PDR as exploration increases, with the best result achieved at $\epsilon = 0.0$, suggesting that a fully greedy policy is optimal in this scenario. For γ, PDR improves steadily with higher discounting, peaking near $\gamma = 1.0$, which highlights the benefit of emphasizing long-term rewards. The PDR values for different learning rates α remain relatively stable, with a slight improvement around $\alpha = 0.3$ to $\alpha = 0.6$, indicating that moderate learning rates strike a good balance. Finally, the subplot for reward penalty demonstrates that reducing the severity of negative rewards (i.e., making them less negative) improves performance, with the optimal PDR occurring at a penalty of -3.0.

4.2 Varying network population and transmissions number

Figure 4.2a and figure 4.2b presents the results of experiments designed to evaluate the scalability of the tested protocols under different network populations. In particular, we study how the SARSA-based reinforcement learning approach performs as the number of EDs increases while keeping the frame size constant. This setup allows us to evaluate the agent's ability to identify and utilize idle time slots more efficiently than ALOHA and Slotted-ALOHA.

The results show that the proposed SARSA-based approach consistently outperforms the baseline protocols, even with up to 15 EDs active in the network. The most significant performance gains—22.5% and 36% improvement in PDR over Slotted-ALOHA and Pure ALOHA, respectively—are observed when 9 EDs are deployed. In this setup, the network remains sufficiently uncongested to allow the RL agent to fully exploit its learning capability. Importantly, these reliability improvements come with minimal energy overhead: the maximum



(a) Exploration rate

(b) Learning rate

(c) Discount factor

(d) Negative reward

Figure 4.1: SARSA parameter iteration results.

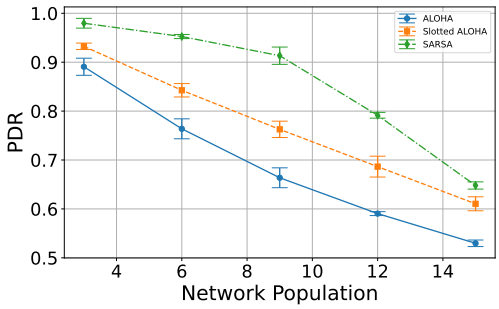
observed increase in energy consumption is just 2.7%, or approximately 1.3 mJ per device, demonstrating that the SARSA-based method maintains energy efficiency while enhancing communication performance.

In the final set of experiments, we evaluate the performance of the tested protocols when EDs transmit multiple uplinks per frame. Two scenarios were considered, each involving 6 EDs with different transmission loads. The corresponding results for PDR are shown in Figure 4.2c.

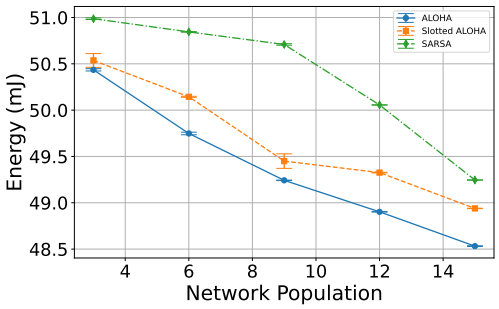
The first scenario is characterized by a balanced load: 2 EDs transmit 3 uplinks, 2 EDs transmit 2 uplinks, and the remaining 2 EDs transmit 1 uplink each. The second scenario, on the other hand, presents a heavier and more asymmetric load: 4 EDs transmit 3 uplinks, while the remaining 2 EDs transmit 2 and 1 uplink, respectively. In both cases, the SARSA-based RL approach achieves a PDR improvement of 14 percentage points over Pure ALOHA and 10 percentage points over Slotted-ALOHA. This improvement comes with only a marginal increase in energy consumption—up to 1.8 mJ per device, or just 2.37%—demonstrating that the learning-based method remains energy-efficient even under high traffic diversity. These results confirm the benefit of using lightweight on-device learning to manage slot contention in diverse and traffic-intensive LPWAN environments.

4.3 Transfer learning

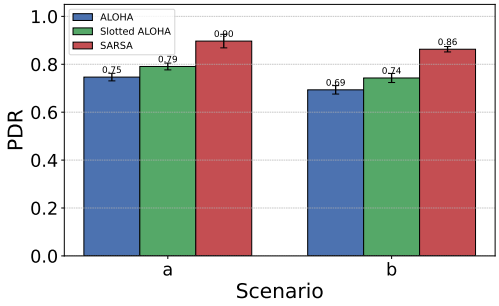
The environment for transfer learning experiments were not sufficient to give any meaningful results. Due to the small number of devices, the devices converged too fast to accurately compare the speed of convergence. Figure 4.2e shows that a device with transferred knowledge has a higher PDR when compared to a regular a regular device, although the error margin in the first 10 frames is too high to measure the convergence rate.



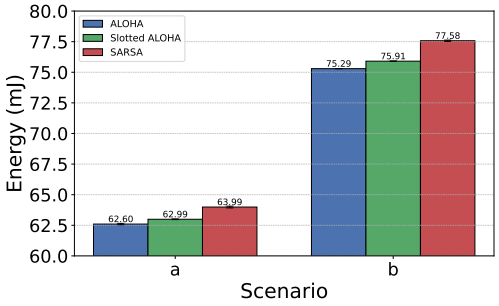
(a) PDR vs. number of EDs



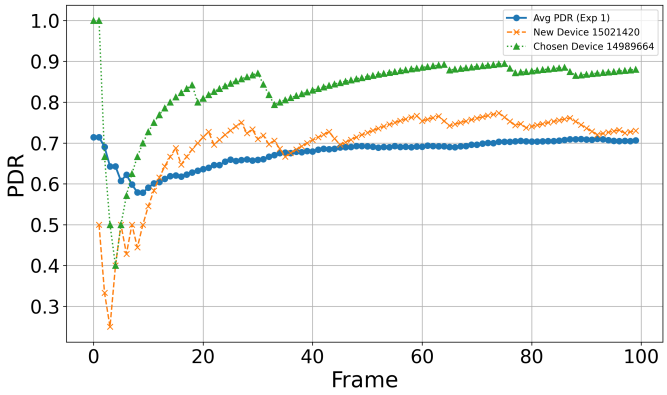
(b) Energy per Frame/ED vs. number of EDs



(c) PDR in multi-uplink scenarios



(d) Energy per Frame/ED in multiple scenarios



(e) Convergence of a device with transferred Q-table vs. regular devices

Figure 4.2: Performance comparison of SARSA, ALOHA, and Slotted-ALOHA under different conditions. (a)–(b): Effect of increasing the number of end-devices on PDR and energy consumption. (c)–(d): PDR and energy efficiency in multi-uplink scenarios with 6 EDs. (e): Faster convergence when Q-table knowledge is transferred across devices.

Chapter 5

Discussion and Analysis

The goals of this project have been achieved. Our team has developed a packet scheduling algorithm that uses reinforced learning to reduce collision rate. The SARSA algorithm has proven to be more efficient when compared to conventional packet scheduling algorithms. On average, SARSA successfully delivers more packets than Aloha or Slotted-Aloha on every network population density. We also fine-tuned its parameters to maximize performance. The transfer learning experiments were not conclusive and require further research.

5.1 SARSA performance

Figure 4.2a clearly shows that SARSA outperforms ALOHA and Slotted-ALOHA on any population density. Because ALOHA is essentially random, its collision rate is proportionate to the network population, whereas SARSA adapts to collisions and is able to utilize any available space in the time frame. However, as the network population approaches its full capacity, SARSA's PDR becomes more similar to that of ALOHA's, as shown in Figure 4.2a. When the network is full, there are no more slots to send packets to, so collisions become inevitable. In that scenario, it is expected for SARSA to not be significantly more efficient than random scheduling. Overall, SARSA is less likely to cause collisions between two devices than ALOHA or Slotted-ALOHA and the advantage is most evident at lower network population densities.

5.2 Hyperparameter tuning

Figure 4.1 revealed that some parameters are more influential than other. Exploration rate in particular has a significant impact on SARSA's PDR. As seen in Figure 4.1a, PDR peaks at 0 exploration rate and is at its minimum at 1.

With no exploration rate, SARSA would probe the same slot until its q-value is no longer the highest in the q-table. If the network population does not exceed the capacity, it is safe to assume that each device would eventually find an unoccupied slot. When each device has occupied its slot, it would never try another and risk causing a collision since there is no chance of random exploration, hence the near perfect performance.

When exploration rate is set to 1, devices always choose slots randomly. Notice that SARSA's performance at exploration rate 1 in Figure 4.1a is similar to Slotted-ALOHA's performance in Figure 4.2a. Indeed, when exploration rate is set to 1, SARSA behaves identical to Slotted-ALOHA.

Although exploration rate is reduced to 0 over time, it is clear that it has significant impact on SARSA's performance. The results show that it should be kept at 0 from the start on

smaller networks such as our testbed with only 15 devices. Further experimentation with more devices is needed.

5.3 Transfer learning

The testbed was not sufficient to conduct experiments on transfer learning. Devices in such a small network achieve convergence too quickly to accurately measure the speed. Therefore we cannot comprehensively compare the convergence speed of a newly introduced device.

Furthermore, Figure 4.2e shows that the device with transferred knowledge generally performs better than an average device. Because the new device gathers q-tables from every device in the testbed, it has knowledge of the entire network which gives it an advantage. In the field, however, it is expected that a new device would query only neighboring devices and would have limited knowledge of the network.

The experiment should be conducted on a bigger network or in a simulation. Lowering the learning rate may help with the convergence speed.

Chapter 6

Conclusions and Future Work

6.1 Conclusion

In this work, we proposed and implemented a reinforcement learning-based communication strategy using the SARSA algorithm to improve the efficiency and reliability of packet transmissions in a slotted communication environment. We specifically targeted performance optimization in the context of constrained wireless networks, where traditional methods like ALOHA and Slotted ALOHA tend to suffer from increased collision rates and reduced throughput as network load increases. Our SARSA agent learns to dynamically select transmission slots in a way that minimizes collisions and maximizes long-term packet delivery ratio (PDR), even under high network density.

A comprehensive hyperparameter optimization was carried out to determine the most effective configuration for SARSA. We evaluated the influence of four critical parameters: the exploration rate (ϵ), discount factor (γ), learning rate (α), and the magnitude of the negative reward assigned to unsuccessful transmissions. The results showed that a greedy policy ($\epsilon = 0.0 - 0.2$), a high discount factor ($\gamma \approx 0.9 - 1.0$), and moderate learning rates produced the highest PDR. Moreover, we found that the relative difference between positive and negative rewards had a significant effect on learning stability, with a penalty of around -3.0 yielding optimal results.

To ensure scalability and efficient decision-making during execution, we implemented a priority queue using a min-heap structure. This data structure allows the agent to quickly retrieve the action with the highest Q-value without traversing the entire action space, which is particularly advantageous for deployment on low-resource microcontrollers where both time and memory are constrained.

We also explored the feasibility of transfer learning by reusing trained Q-values across similar environments, significantly accelerating convergence and improving adaptability in new network scenarios. This capability makes the SARSA agent robust and reusable across deployments with varying slot configurations or network topologies.

Finally, we validated our approach through implementation on a real-world testbed comprising LoRa-enabled nodes. The system was evaluated under realistic communication conditions to assess its performance beyond simulations. The full source code, along with deployment instructions and sample configurations, has been made publicly available as open-source software to promote reproducibility and community collaboration.

In summary, our work demonstrates that reinforcement learning, when carefully optimized and efficiently implemented, can significantly outperform traditional MAC strategies in wireless networks. The integration of algorithmic efficiency, transfer learning, and real-world testing provides a strong foundation for further research and deployment in practical IoT and embed-

ded systems.

6.2 Future work

While reinforcement learning (RL)-based communication strategy using the SARSA algorithm to decrease packet collisions in LoRa has been successfully implemented and validated, several aspects remain unexplored, and additional improvements could further enhance its performance. The lessons learned during the implementation and testing phase provide clear directions for future research and development.

6.2.1 Scalability and Large-Scale Deployment

In the given testbed, the system was evaluated using a limited number of LoRa devices, which allowed us to observe the behavior of the algorithm in controlled conditions. Nevertheless, as the number of devices in the network increases, the dynamics of packet collisions will evolve. Future research should focus on extending the evaluation to larger networks, ideally in the range of hundreds or even thousands of devices, to examine the system's performance under heavy load. This could involve testing the RL framework in diverse environments, such as smart cities or industrial IoT settings, where the number of devices and traffic patterns may fluctuate dynamically.

6.2.2 Hybrid models

Hybrid models combining RL with traditional methods like TDMA (Time Division Multiple Access) or CSMA (Carrier Sense Multiple Access) could be explored. Such hybrid approaches might offer new strategies, where RL dynamically adapts the network under varying traffic conditions, while traditional methods provide a standard behavior for well-understood scenarios. Exploring the combination of RL and other scheduling protocols could also help mitigate some of the computational constraints faced by low-resource devices in the IoT domain.

6.2.3 Transfer Learning and Generalization

While our current approach demonstrates strong performance in dynamic and heterogeneous network conditions, it relies on training that aggregates data from all EDs to achieve optimal results. To enable broader applicability, future work should focus on exploring different strategies for Q-table aggregation and policy sharing across devices. Investigating distributed learning schemes—where each device learns independently or shares partial information—could reduce communication overhead and enhance scalability.

References

Acik, S., Kosunalp, S., Tabakcioglu, M. B. and Iliev, T. (2023), 'Improving the performance of aloha with internet of things using reinforcement learning', *Electronics* **12**(17), 3550.

Ahmed, A. I., Etiabi, Y., Azim, A. W. and Amhoud, E. M. (2024), A unified deep transfer learning model for accurate iot localization in diverse environments, *in* '2024 IEEE 35th International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)', pp. 1–6.

Baimukhanov, B., Gilazh, B. and Zorbas, D. (2024), Autonomous lightweight scheduling in lora-based networks using reinforcement learning, *in* '2024 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)', IEEE, pp. 268–271.

Heusse, M., Caillouet, C. and Duda, A. (2023), 'Performance of unslotted aloha with capture and multiple collisions in lorawan', *IEEE Internet of Things Journal* **10**(20), 17824–17838.

Hong, S., Yao, F., Zhang, F., Ding, Y. and Yang, S.-H. (2023), 'Reinforcement learning approach for sf allocation in lora network', *IEEE internet of things journal* **10**(20), 18259–18272.

Minhaj, S. U., Mahmood, A., Abedin, S. F., Hassan, S. A., Bhatti, M. T., Ali, S. H. and Gidlund, M. (2023), 'Intelligent resource allocation in lorawan using machine learning techniques', *IEEE Access* **11**, 10092–10106.

Ouameur, M. A., Caza-Szoka, M. and Massicotte, D. (2020), 'Machine learning enabled tools and methods for indoor localization using low power wireless network', *Internet of Things* **12**, 100300.

Polonelli, T., Brunelli, D., Marzocchi, A. and Benini, L. (2019), 'Slotted aloha on lorawan-design, analysis, and deployment', *Sensors* **19**(4), 838.
URL: *http://dx.doi.org/10.3390/s19040838*

Prakash, O., Pattanayak, P., Rai, A. and Cengiz, K. (2023), Machine learning and deep reinforcement learning in wireless networks and communication applications, *in* 'Paradigms of Smart and Intelligent Communication, 5G and Beyond', Springer, pp. 83–102.

Sandoval, R. M., Garcia-Sanchez, A.-J. and Garcia-Haro, J. (2019), 'Optimizing and updating lora communication parameters: A machine learning approach', *IEEE Transactions on Network and Service Management* **16**(3), 884–895.

Tsakmakis, A., Valkanis, A., Beletsioti, G., Kantelis, K., Nicopolitidis, P. and Papadimitriou, G. (2022), 'An adaptive lorawan mac protocol for event detection applications', *Sensors* **22**(9), 3538.

Vázquez-Gallego, F., Tuset-Peiró, P., Alonso, L. and Alonso-Zarate, J. (2020), 'Delay and energy consumption analysis of frame slotted aloha variants for massive data collection in internet-of-things scenarios', *Applied Sciences* **10**(1), 327.

Yılmaz, S., Aydogan, E. and Sen, S. (2021), 'A transfer learning approach for securing resource-constrained iot devices', *IEEE Transactions on Information Forensics and Security* **16**, 4405–4418.