Final Year Project Proposal

**Machine Learning for IoT**

By:

**GHOUSSOUB Paul**

**HAGE Elia**

Supervisor:

**LAHOUD Samer**

Final Year Project Proposal

A report submitted in partial fulfillment of the requirements for the degree of

**COMPUTER AND COMMUNICATIONS ENGINEERING**

8/3/2019

école Supérieure d’Ingénieurs de Beyrouth

Université Saint-Joseph de Beyrouth

Contents

[List of figures 3](#_Toc2946448)

[1 Introduction 4](#_Toc2946449)

[2 Background Research 5](#_Toc2946450)

[3 Requirements 7](#_Toc2946451)

[3.1 Functional Requirements 7](#_Toc2946452)

[3.2 Secondary Goals 7](#_Toc2946453)

[3.3 Constraints 7](#_Toc2946454)

[4 Solution Design 8](#_Toc2946455)

[4.1 Design alternatives and comparison 8](#_Toc2946456)

[4.1.1 Exp3 8](#_Toc2946457)

[4.1.2 Exp4 8](#_Toc2946458)

[4.1.3 UCB1 10](#_Toc2946459)

[4.1.4 Epsilon-Greedy Algorithm 10](#_Toc2946460)

[4.1.5 Epsilon-Decreasing Algorithm 11](#_Toc2946461)

[4.1.6 Thompson Sampling 11](#_Toc2946462)

[4.1.7 Comparison and Conclusion 12](#_Toc2946463)

[4.2 Required Resources 12](#_Toc2946464)

[4.3 Testing Environment 12](#_Toc2946465)

[4.3.1 Fixed Spreading Factor – Varying Channels 13](#_Toc2946466)

[4.3.2 Varying Spreading Factor – Fixed Channels 13](#_Toc2946467)

[4.3.3 Varying Spreading Factor – Varying Channels 13](#_Toc2946468)

[5 Project Management 14](#_Toc2946469)

[6 References 15](#_Toc2946470)

# List of figures

[Figure 1 - Machine Learning Types 5](#_Toc2946471)

[Figure 2 - EXP3 Algorithm 8](file:///C:\Users\Paul%20Ghoussoub\Downloads\Proposal%20Report.docx#_Toc2946472)

[Figure 3 - EXP4 Algorithm 9](#_Toc2946473)

[Figure 4 - UCB1 Algorithm 10](#_Toc2946474)

[Figure 5 - Epsilon-Greedy Algorithm 10](#_Toc2946475)

[Figure 6 - β-distribution where α is higher than β 11](#_Toc2946476)

[Figure 7 - Algorithm Comparison Table 12](#_Toc2946477)

[Figure 8 - Project Management on Trello 14](#_Toc2946478)

# Introduction

Starting from September 2016, ESIB has been deploying the first academic Internet of Things (IoT) LoRa network in Lebanon. As defined by Semtech, LoRa is a wireless technology developed to create the low-power, wide-area networks (LPWANs) required for Machine-to-Machine (M2M) and Internet of Things (IoT) applications. The technology offers a very compelling mix of long range, low power consumption and secure data transmission and is gaining significant traction in IoT networks being deployed by wireless network operators.

Designing resource allocation schemes to support such stringent communication is a key challenge for IoT. In such a context, learning techniques are very adequate to devise self- organizing resource allocation solutions that match the unique IoT features.

It is obvious that the LoRa devices must be able to obtain their communication resources autonomously, because it is impractical to assume that they can communicate frequently with the network, given their stringent resource constraints. In this project, the network can guide the devices in selecting the appropriate spreading factor based on their channel quality and/or location. Further, the network can limit the access of devices to a restrained number of channels based on the density of traffic, in order to lower the collision rate.

Precisely, we will have two levels of learning with two approaches, Reinforcement Learning and Machine Learning:

•  The Device level: we will address the challenging learning of channel occupancy (the choice of both spreading factor and channel) through Reinforcement Learning (RL). The latter is adapted to the limited resources of the devices and the little interaction that they can have with other devices and with the network. The proposed RL algorithms will strive to maximize a tailored utility function that accounts for the successful transmissions (acknowledged by the network), the consumed energy (the spreading factor will be inversely proportional to the battery life), etc. A good candidate is the adversarial Multi-armed bandit (MAB) where each player is a given device and each arm is one of the available channels and/or spreading factors.

•  The Network level: on the network side, we will resort to Machine Learning (ML) to grasp the specificities of the network setting. This is done by learning for instance, the location of devices (fingerprinting-based localization), the density of devices, the type of devices (for QoS issues), etc. In particular, traces from transmissions introduced at the device level will serve to train an Artificial Neural Network. Hence, the network enables to operate dynamic spectrum allocation, where the number of necessary channels is allocated to the end-devices and the appropriate spreading factor suitable to any device (based on its location and the rate of successful transmissions).

The objective of the project is to adapt and assess Machine Learning algorithms for selecting radio resources in a LoRa network.

# Background Research

Before starting the work on the project, we needed to explore the Machine Learning world in order to understand the different approaches we can follow in order to implement a working solution to our project.

At first, we started by studying the different Machine Learning approaches: Supervised, Unsupervised and Reinforcement Learning.

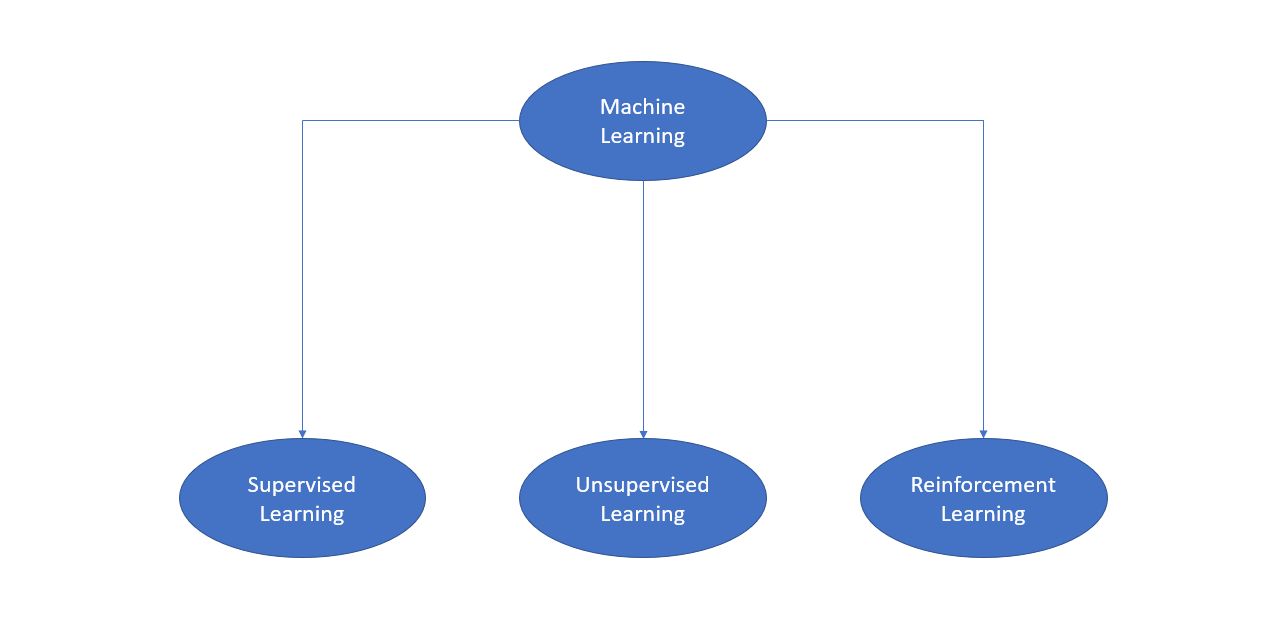


Figure 1 - Machine Learning Types

In supervised learning, all data is labeled, and the algorithms learn to predict the output from the input data.

In unsupervised learning, all data is unlabeled, and the algorithms learn to inherent structure from the input data.

Reinforcement learning is about taking suitable action to maximize reward in a particular situation. It is employed by various software and machines to find the best possible behavior or path it should take in a specific situation. [1]

We will be using reinforcement learning algorithms. In fact, we need an algorithm to search for a network configuration in which a successful packet transmission is possible while also being able to adapt to changes in the environment.

Reinforcement Learning helps to solve the Multiple-Armed Bandit (MAB) problem. In the multi-armed bandit problem, a gambler must decide which arm of K non-identical slot machines to play in a sequence of trials so as to maximize his reward. This classical problem has received much attention because of the simple model it provides of the trade-off between exploration (trying out each arm to find the best one) and exploitation (playing the arm believed to give the best payoff) [[2]](#_References).

Knowing that we will be using Reinforcement Learning, it is important to define the different types of Reinforcement [[3]](#_References):

* Positive Reinforcement:Defined as when an event occurs due to a particular behavior, it increases the strength and the frequency of the behavior. In other words, it has a positive effect on the behavior.
* Negative Reinforcement:Defined as strengthening of a behavior because a negative condition is stopped or avoided.

In the case of our project, positive reinforcement would be the case in which a packet is successfully transmitted.

LoRa is a low-power, wide area network through which IoT devices can communicate: Devices such as Arduinos can send and receive packets to each other. This communication requires an adequate network configuration. In fact, the distance between the devices or even the traffic on a specific radio channel could play a major role in the successful transmission of a packet. The parameters that are taken into consideration in this project are the spreading factor, the radio channel and the power.

In case of a successful transmission, the network configuration in which the transmission was performed will have a higher probability of appearing in future transmissions. On the other hand, negative reinforcement would be the case in which a packet could not be transmitted with a given network configuration. Such configuration will therefore have a lower probability of appearing in future transmissions.

There are many terms related to Reinforcement Learning that will be used in this report, including:

* Agent: An agent takes actions. Our agent, in this project, is the Arduino which is running the Reinforcement Learning algorithm.
* State: A state is a concrete and immediate situation in which the agent finds itself. An example of a state in our project is: (Spreading Factor = 12; Radio Channel 1).
* Reward: A reward is the feedback by which we measure the success or failure of an agent’s actions. The reward is usually 0 or 1 point. If the packet transmission is successful, the agent is rewarded 1 point. Else, no points are rewarded. [[4]](#_References)

# Requirements

## Functional Requirements

The main objective of the project is based on the device level which was described earlier. The requirements are the following:

* Design an experimental platform that reproduces realistic LoRa deployment conditions
* Use machine learning algorithms for selecting the optimal spreading factors and radio channels on LoRa devices

## Secondary Goals

A stretch goal for the project is to implement a solution for the problem at the network level, also described earlier.

* Generate traces for LoRa device transmissions
* Implement and train an artificial neural network using the generated traces
* Propose resource allocation decisions at the network level using the artificial neural network

## Constraints

* LoRa devices must obtain their communication resources autonomously
* Lower the scale of the problem: Implement a solution that requires a low number of devices that does not surpass 20 to 30 Arduinos.
* The devices have low memory and low computing power: 32KB flash memory and 16Mhz processor [[5]](#_References)
* We are legally not allowed to surpass a power of 14 dBm [6]

# Solution Design

## Design alternatives and comparison

After completing our research, we have found many ways in which we can apply Machine Learning to IoT devices. In fact, Reinforcement Learning presents many algorithms and choosing the correct algorithm is a vital objective.

### Exp3

The first algorithm we have found was Exp3. Exp3 stands for Exponential-weight algorithm for Exploration and Exploitation. This algorithm is notably easy to implement and does not require a lot of computing power or computing memory. [7]

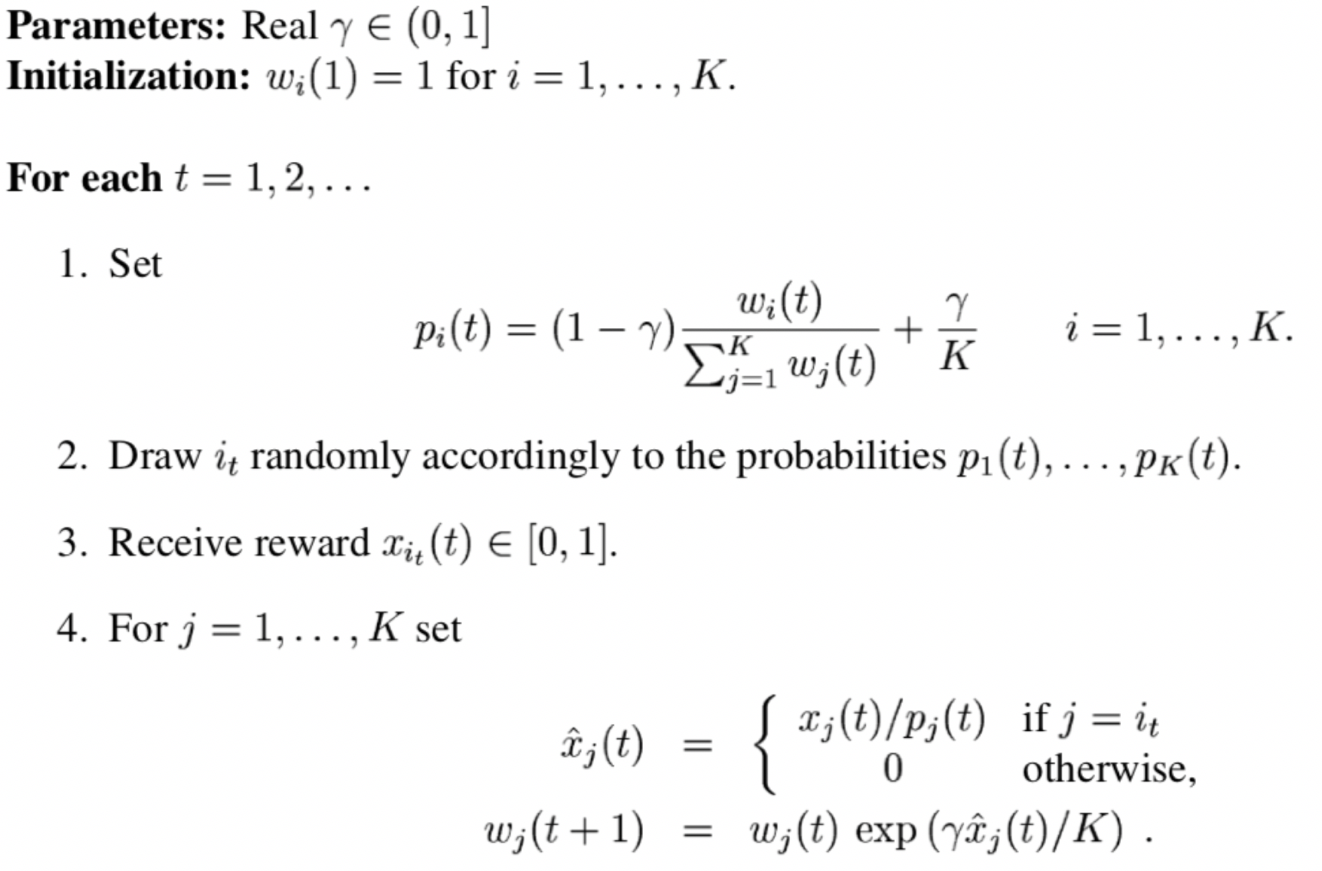


Figure - EXP3 Algorithm

### Exp4

Another algorithm that we have seen is Exp4: Exponential-weight Algorithm for Exploration and Exploitation using Expert advice. It is generally seen as an improvement of the Exp3 algorithm, but these improvements come at a cost. In fact, EXP3 ignores useful contextual information that could help in choosing the correct configuration for a certain communication. In our case, the server could send details regarding the bandwidth and distance between the client and the server in order to help the algorithm make the correct choice. [8]

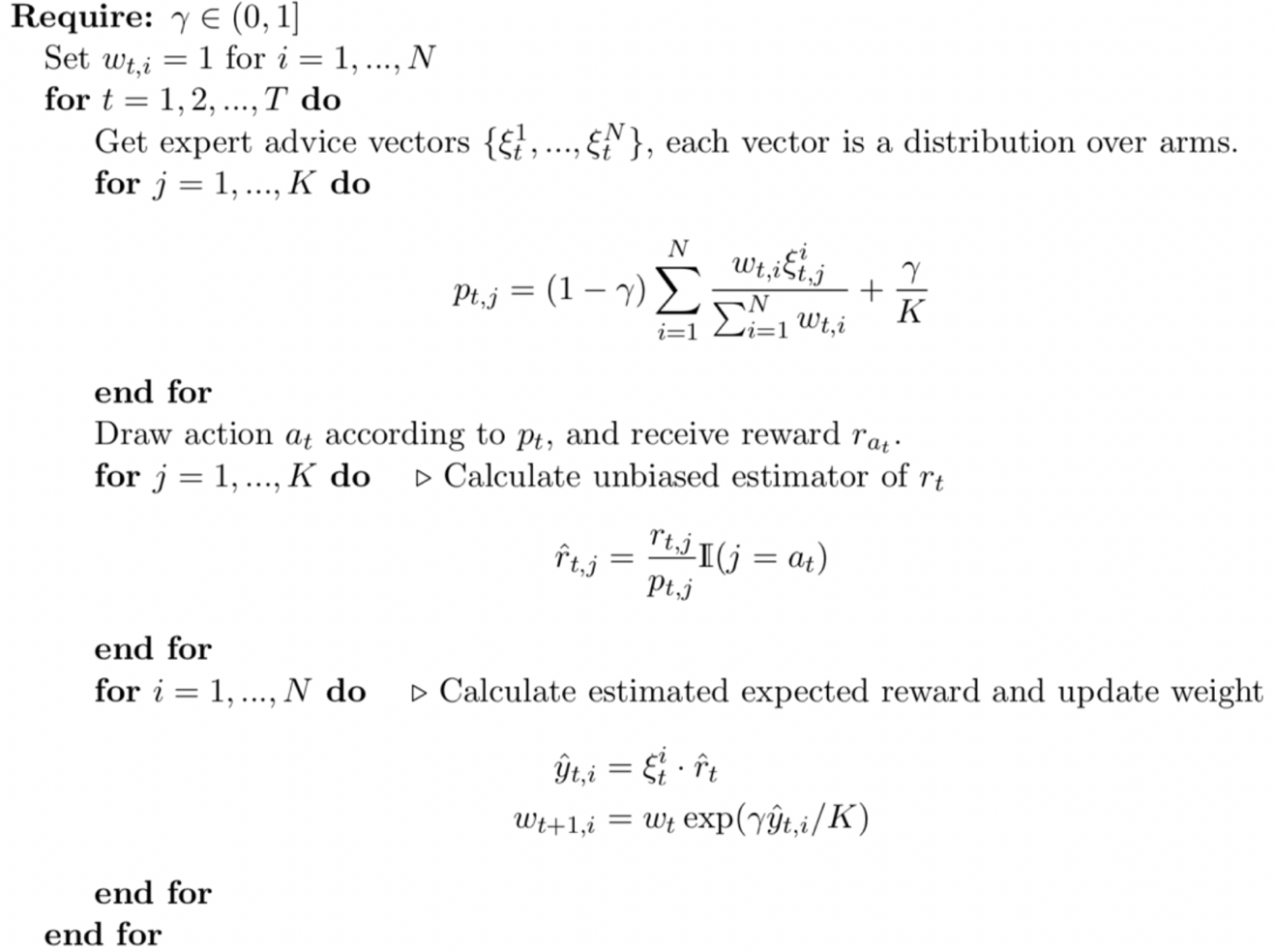


Figure 3 - EXP4 Algorithm

We can conclude that the algorithm presents many difficulties, especially regarding the expert advice and how the server could possibly send valuable information to the client.

### UCB1

UCB1, or Upper Confidence Bound, is also a Reinforcement Learning algorithm that we could implement in our project. It can be summed by the principle of optimism in the face of uncertainty: The algorithm simply picks the action that has the highest probability of succeeding. [9]

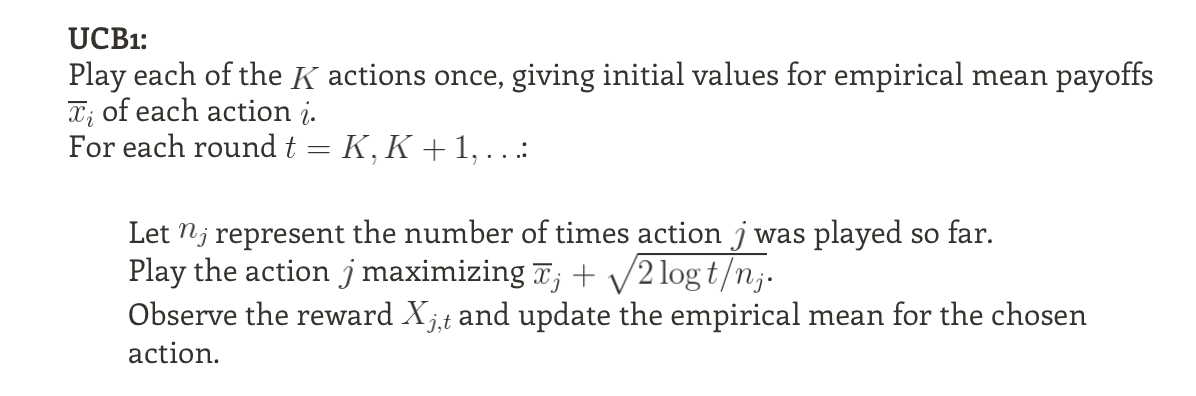


Figure 4 - UCB1 Algorithm

The algorithm is easy to implement and doesn’t introduce many difficulties.

### Epsilon-Greedy Algorithm

In this algorithm, we use a parameter ε that would be a fixed value between 0 and 1. Then we randomly choose a number between those values. If the value selected is smaller than ε, then we’re in the exploration phase and we choose and random channel/spreading factor to work on. If the value is higher than ε, then we choose the current best channel/spreading factor and work on it. [10]

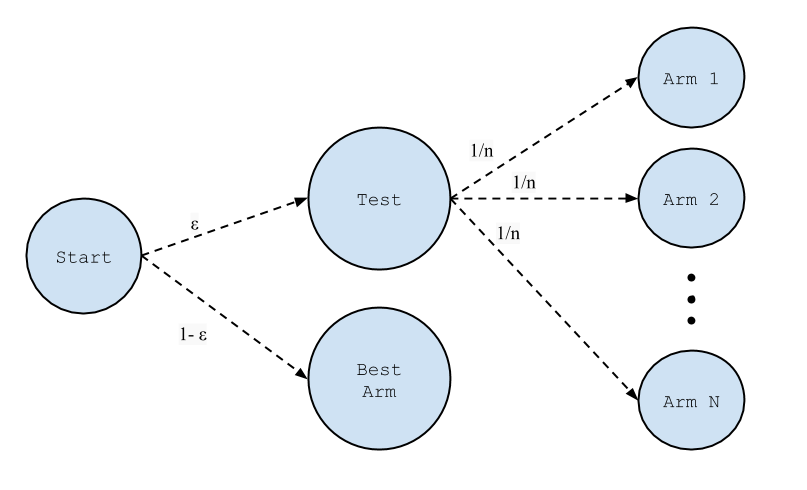


Figure 5 - Epsilon-Greedy Algorithm

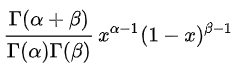
### Epsilon-Decreasing Algorithm

This algorithm resembles a lot the previous one, with the sole difference being that ε decreases over time, thus reducing the exploration phases and increasing the exploitation phases.

### Thompson Sampling

Also known as probability matching, it reflects the idea that the number of pulls for a given lever should match its actual probability of being the optimal lever.

It uses the β -distribution:

With 0 ≤ x ≤ 1 and α > 0 and β>0

Where :



α and β values will change with each iteration. When α is bigger than β, the probability distribution is more concentrated towards the higher values and vice-versa [11]

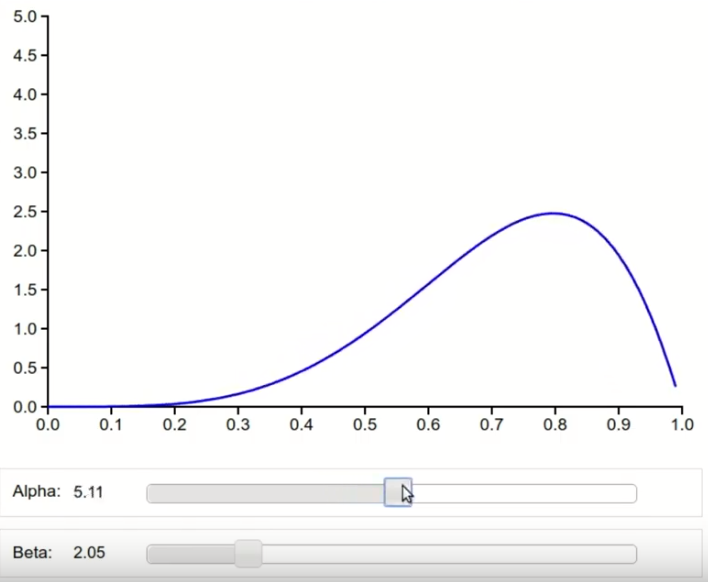


Figure 6 - β-distribution where α is higher than β

In the beginning, we have all channels/spreading factors with the same probabilities. We choose an x within the boundary of each β-distribution and the channel/spreading factor with the highest x is chosen. If the result of the experiment is a success, then the α parameter becomes α +1 for that arm and if it fails, then β = β +1.

With time, all the β-distributions will have realistic values, and the best channel/spreading factor will be chosen.

### Comparison and Conclusion

A simple way to compare the algorithms is by comparing their regret bounds. It's just the difference between optimal and actual reward. Every step that is taken by the algorithm in reinforcement learning has a return or reward associated with it. If you had taken the optimal steps at each trial, your total reward is called the optimal reward. In theory, all algorithms try to minimize this regret.

Another way to compare them is to conclude whether the algorithm is complex: This can be done by analyzing the memory allocation required for each algorithm and the difficulty of implementation.

|  |  |  |
| --- | --- | --- |
|  | Regret | Complex |
| EXP3 | O() | No |
| UCB1 | O() | No |
| EXP4 | O() | Yes |
| Epsilon-Greedy | O(T) | No |
| Thompson Sampling | O() | Yes |

Figure 7 - Algorithm Comparison Table

*K represents the number of states; T represents the number of iterations.*

In conclusion, we will move forward by implementing the EXP3 algorithm as it seems to be the perfect fit for our project.

Note that we might implement another algorithm, most probably UCB1, in order to compare its performance with EXP3.

## Required Resources

To complete the project, we will need:

* LoRa devices provided by ESIB
* Arduino

## Testing Environment

As previously mentioned, we will start by implementing the EXP3 algorithm. In fact, various scenarios can be put in place in our project: We know that LoRa uses 6 spreading factors ranging from 7 to 12 and that there also are 8 total radio channels. Putting together all of the combinations will result in a huge amount of states that require great number of Arduinos.

This is why we will lower our scale and we will initially work with 3 radio channels. We are therefore presented with two options:

1. Use 3 Arduinos as our servers. Each Arduino listens to one channel
2. Use the ESIB server. It can listen to all the channels simultaneously.

Using the ESIB server will lead to using LoRaWAN: This introduces many obstacles such as the security layer as well as using an MQTT server. It is deemed that it will burden the advancement of the project with no real gain when opposed to the first option, in which a LoRa communication is possible.

Therefore, we will be moving forward with the first option.

### Fixed Spreading Factor – Varying Channels

We will introduce a certain number of devices that will be sending packets to our servers through the available radio channels. All devices will be using a fixed spreading factor of 12.

We will have one device running the Reinforcement Learning algorithm. The goal is to see how this device changes the radio channel over time in order to accommodate for the traffic on the radio channels.

We will increase the number of devices to see which number lowers the chance of successful transmission and we aim to observe the device adjusting its radio frequency accordingly.

We will have 3 radio channels available:

* The first one will be used by many “Jamming” devices that will be sending random packets at random times
* The second one will have a lower number of jamming devices
* The third one will have the least amount of jamming devices

We will finally compare the results of this experiment with the result of a device that is choosing the radio channel randomly.

### Varying Spreading Factor – Fixed Channels

We will repeat the same experiment but with a fixed channel and varying spreading factors. We should note that two devices on the same channel but with different spreading factors will never have interfering packets.

We aim to see the device changing its spreading factor to accommodate for the increase in the number of devices using a specific spreading factor.

### Varying Spreading Factor – Varying Channels

Varying the two parameters will increase the number of decisions that the device can take. We aim to see the device adjusting its two parameters accordingly.

# Project Management

For the planning of our project, we will be using Trello. Trello is a task management application that gives you a visual overview of what is being worked on and who is working on it. It consists of boards, lists and cards that help keep track of the all the tasks of the project.

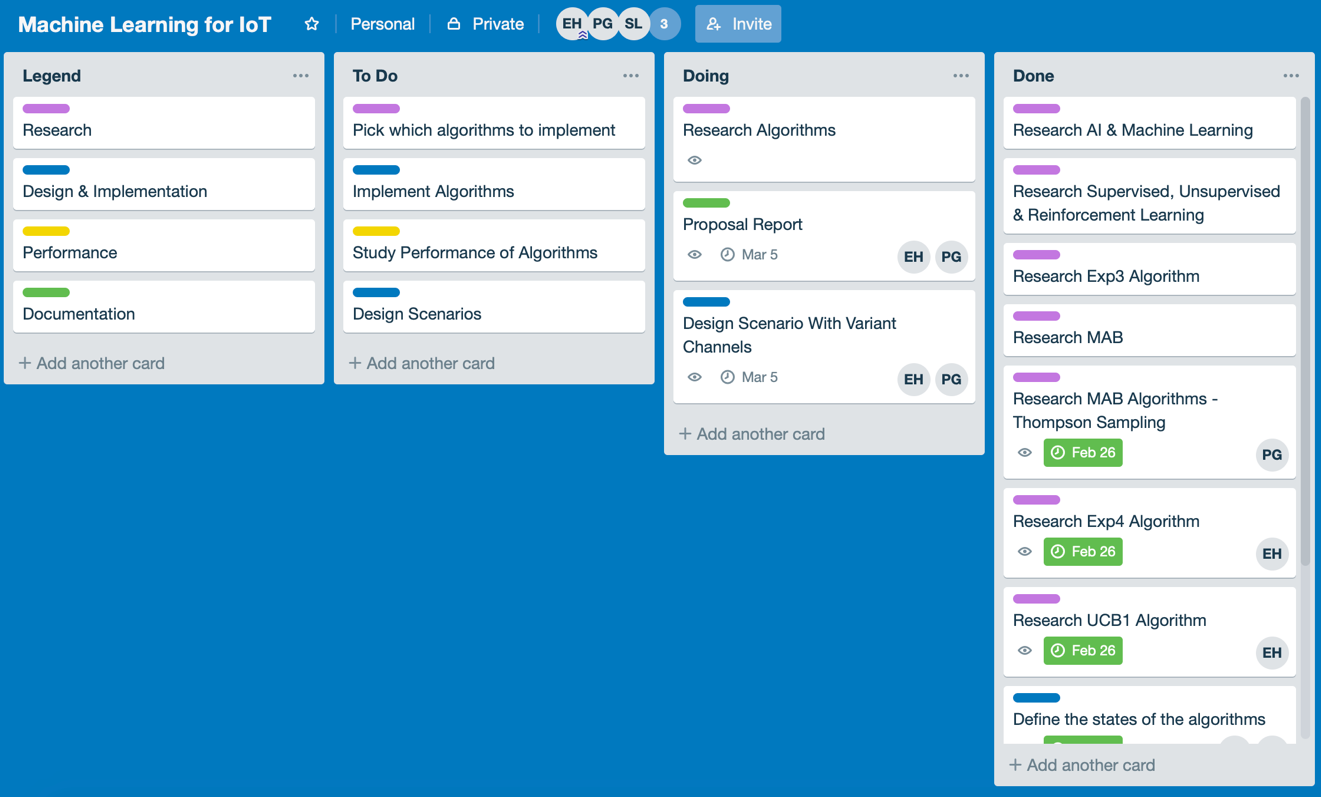


Figure 8 - Project Management on Trello

As can be seen in the picture above, each task can be associated to one or many members and generally has a description and a due date. This helps the team members stay updated and vastly improves flexibility in case a sudden and unexpected change in the plans are required.

# References

[1] <https://www.teradata.com/Blogs/The-Tree-of-Machine-Learning-Algorithms>

Title: The Tree of Machine Learning Algorithms

Updated: 10 Oct 2017

Author: Enrico Galimberti

[2] <https://ieeexplore.ieee.org/abstract/document/492488>

Title: Gambling in a rigged casino: The adversarial multi-armed bandit problem - IEEE Conference Publication

Updated: 6 Aug 2002

Authors: P. Auer, N. Cesa-Bianchi, Y. Freund, R.E. Schapire

[3] <https://www.geeksforgeeks.org/what-is-reinforcement-learning/>

Title: Reinforcement learning – GeeksforGeeks

Updated: n.d.

Author: Prateek Bajaj

[3] <https://skymind.ai/wiki/deep-reinforcement-learning>

Title: A Beginner's Guide to Deep Reinforcement Learning

Updated: ca. 2015

Author: Skymind Inc.

[5] <http://www.hobbytronics.co.uk/arduino-uno-r3>

Title: Arduino UNO R3 | A000066 | Arduino

Updated: ca. 2019

Author: Arduino

[6] <https://www.semtech.com/uploads/documents/etsi-compliance-sx1272-lora-modem.pdf>

Title: ETSI Compliance of the SX1272/3 LoRa Modem

Updated: 1 Jul 2013

Author: Semtech Corp.

[7] <https://jeremykun.com/2013/11/08/adversarial-bandits-and-the-exp3-algorithm/>

Title: Adversarial Bandits and the Exp3 Algorithm

Updated: 8 Nov 2013

Author: Jeremy Kun

[8] <http://banditalgs.com/2016/10/14/exp4/>

Title: Contextual Bandits and the Exp4 Algorithm

Updated: 14 Oct 2018

Author: Csaba Szepesvari

[9] <https://jeremykun.com/2013/10/28/optimism-in-the-face-of-uncertainty-the-ucb1-algorithm/>

Title: Optimism in the Face of Uncertainty: the UCB1 Algorithm

Updated: 28 Oct 2013

Author: Jeremy Kun

[10] <https://jamesmccaffrey.wordpress.com/2017/11/30/the-epsilon-greedy-algorithm/>

Title: The Epsilon-Greedy Algorithm

Updated: 30 Nov 2017

Author: James D. McCaffrey

[11] https://arxiv.org/pdf/1707.02038.pdf

Title: A Tutorial on Thompson Sampling

Updated:21 Nov 2017

Authors: Daniel J. Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband and Zhen Wen