HUMBERTO AKIRA UEHARA SASAKI
RODRIGO MORIMOTO SUGUIURA

INDOOR LOCATION SYSTEM: HUMAN TRACKING APPLICATION IN A SHIP MANEUVERING SIMULATOR

São Paulo
2018

HUMBERTO AKIRA UEHARA SASAKI
RODRIGO MORIMOTO SUGUIURA

INDOOR LOCATION SYSTEM: HUMAN TRACKING APPLICATION IN A SHIP MANEUVERING SIMULATOR

Trabalho apresentado à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Engenheiro Mecatrônico. Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos (PMR).

São Paulo
2018

HUMBERTO AKIRA UEHARA SASAKI
RODRIGO MORIMOTO SUGUIURA

INDOOR LOCATION SYSTEM: HUMAN TRACKING APPLICATION IN A SHIP MANEUVERING SIMULATOR

Trabalho apresentado à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Engenheiro Mecatrônico. Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos (PMR).

Orientador:

Ph.D. Prof. Eduardo Aoun Tannuri

São Paulo
2018

ACKNOWLEDGMENTS

Many thanks to the following persons that provided considerable help towards the conception of this work: Felipe Marino Moreno, Alex Huang Saratani, Jair Antunes, Leonardo Ramos, Renato Moura and Thiago Bockholt.

"At its heart, engineering is about using science to find creative, practical solutions. It is a noble profession."

-- Queen Elizabeth II

"Science is about knowing, engineering is about doing."

-- Henry Petroski

"Building the future and keeping the past alive are one in the same thing."

-- Iroquois Pliskin

"Change is the law of life. And those who look only to the past or present are certain to miss the future."

-- John F. Kennedy

"It's not about changing the world. It's about doing our best to leave the world... the way it is. It's about respecting the will of others, and believing in your own."

-- Vic Boss

ABSTRACT

One of the main challenges of this century is to how get information about anything, anywhere and use it somehow. One of the concepts created based on this premise is the Internet of Things (IoT), which is an idea that everything will be connected and will share information with the use of internet. One information which have a lot of value to multiple industries is the location. It can be the location of anything, from a machine, a vehicle, consumer goods and people. Everything needs to be tracked. Nowadays it can be achieved using GPS, but this technology is not reliable on closed spaces, with low satellite signal. To solve these problems Indoor location systems are researched and developed. The requirements to a good indoor location system is to build a solution with a good accuracy, low cost and modular, which can be replicated on other spaces. This research has the purpose of create and prototype an indoor location solution which can fulfill these requirements. To attain these objectives, this research study the most common Indoor Location Systems approaches, like Time of Arrival (ToA), Time Difference of Arrival (TDoA), Received Signal Strength Indicator (RSSI), Dead Recon, Motion Fingerprint and Signal Fingerprint model with Machine Learning algorithms, using of Wi-Fi and Bluetooth signals, and data from sensors, such as accelerometers, gyroscope and electronic compasses. This project evaluated and selected some of those solutions, such as Signal Fingerprint with Machine Learning, implemented it and tested on a ship maneuvering simulator, by tracking a pilot assistant inside the room, which was a challenge since this room is usually crowded with people and have a lot of electromagnetic noise from the simulators equipment.

Keywords – *Indoor location system*, *Machine Learning*, *Artificial Neuron Network*, *Fingerprint*.

LIST OF ABBREVIATIONS

Adam	Adaptive Moment Estimation
ANN	Artificial Neural Network
AoA	Angle of Arrival
AP	Access Point
BLE	Bluetooh Low Energy
CPU	Central Process Unity
DR	Dead Reckoning
ELU	Exponential Linear Unit
ESS	Estimated Signal Strength
GPS	Global Positioning System
ID	Identification
IoT	Internet of Things
LPS	Local Positioning System
MAC	Media Access Control
Nadam	Nesterov-accelerated Adaptive Moment Estimation
pINS	plain Inertial Navigation System
ReLU	Rectified Linear Unit
RF	Radio Frequency
RFID	Radio Frequency Identification
RMSProp	Root Mean Square Propagation
RSS	Received Signal Strength
RSSI	Received Signal Strength Indicator
SGD	Stochastic Gradient Descent
SHS	Step and Heading System
SSD	Standard Sample Deviation

TDoA	Time Difference of Arrival
ToA	Time of Arrival
TPN-USP	Numerical Offshore Tank from University of São Paulo
UDP	User Datagram Protocol
UWL	Ultra-Wide-Band
WLAN	Wireless Local Area Network

LIST OF FIGURES

LIST OF TABLES

CONTENTS

PART I

INTRODUCTION

1 INDUSTRY 4.0

"The Fourth Industrial Revolution is still in its nascent state. But with the swift pace of change and disruption to business and society, the time to join in is now."

-- Gary Coleman, Global Industry and Senior Client Advisor, Deloitte Consulting

The term "Industry 4.0" alludes to the crescent trend of creating more connected and automated environments. A major keystone of this movement is the concept of "smart factory", in which cyber-physical systems monitor the physical manufacture processes, make decentralized decisions and cooperate with each other alongside with humans in real-time [1]. As it can be seen in Fig.1, to some this shift can also be referred as the *Fourth Industrial Revolution* for some due to its impact on the industry mindset and reshaping of production means.

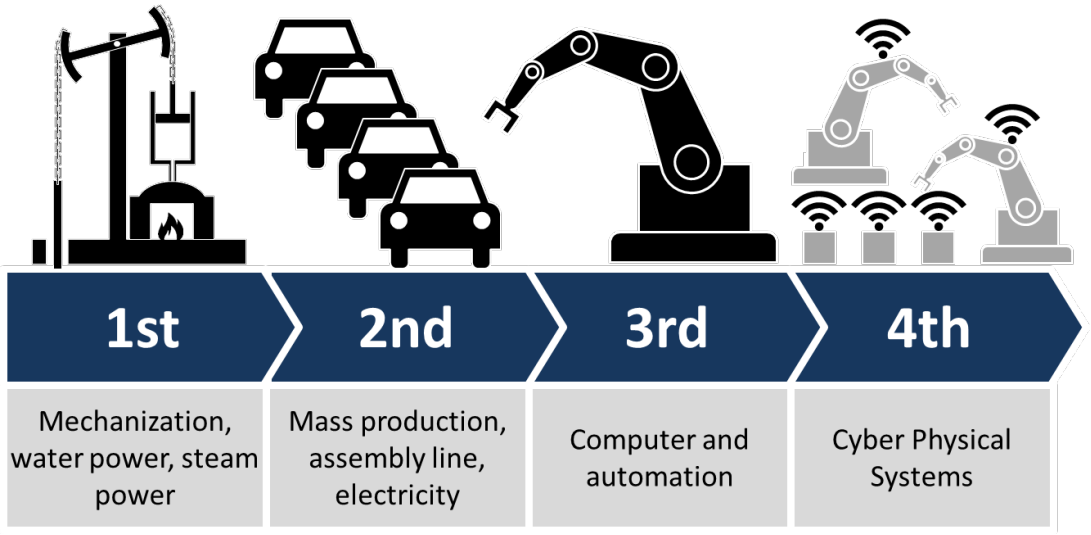


Figure 1: All four industrial revolutions on perspective.

As firstly introduced as "Industrie 4.0", it referred to one of the key initiatives from the German government for promoting the computerization of the manufacturing [2]. By aiming to its enterprises to reach higher technological levels of solutions, the country could boost its productivity and better shape its industries to the demand of the 21st century.

Although being a top priority for many companies, there is not yet a general closed consensus around its concept. However, Hermann et al. [3] identified, from quantitative text analysis and qualitative literature review, four design principles in Industry 4.0:

- Interconnection: the idea of interoperability between machines, devices, sensors and people.
- Information transparency: the creation of a virtual copy of the physical world through sensors.
- Decentralized decisions: by combining both of the previous principles, cyber physical systems can make decisions on their own, performing tasks as autonomously as possible. Highlighting the key role of embedded computers, sensors and actors.
- Technical assistance: the shifting human role from simple operators towards strategic decision-maker and the delegation of unpleasant and unsafe tasks to robots. Accentuating the importance of these complex systems to aggregate and display information as comprehensive as possible for the human.

In summary, as Heuchemer once stated [4] :

> After German Chancellor Angela Merkel (...) ordered a study about the manufacturing environment, the ACATECH (German Academy of Science and Engineering) (...) drafted the vision of Industrie 4.0, which was to be the coordinated initiative between the IT world, universities and various manufacturing associations to reshape industry. It would seek to combine the physical, virtual, IT and cybersystems and thereby create a new working environment between worker and machine.
> (...)
> Industry 4.0 represents a highpoint of dynamic achievement, where every company, whether a large OEM, major tier supplier or smaller job shop, can implement and benefit from the technologies and communications platforms available today.

Furthermore, Diammandis [5] highlights some other aspects such as device connectivity, that allows smart products to communicate seamlessly an automate cumbersome tasks, for instance the analysis of production quotas, programming of predictive maintenance or input designs remotely.

1.1 Internet of Things (IoT)

With the technology advancement in portable devices added to the widespread and decreasing cost of broadband Internet, the scenario shows up as ideal for the creation

of gadgets with built in Wi-Fi capabilities. This exactly phenomenon of connecting everything on internet is so relevant that is known as Internet of Things (IoT) [6] - which could be synthesized by the words of Jouret [7]: “anything that can be connected, will be connected”. Following this trend, it two main branches emerged: one for the industry and the other for services.

The focus on the first one is called ”Industry 4.0” as discussed before. The latter one is usually associated with the concept of “Anything as a Service” (XaaS) [8], which brings the trend of creating operations on Cloud Computing to various real applications, for instance, the monitoring of a baby by built-in cameras inside toys or refrigerators with Internet access, usually all connected at the bottom end with smartphones.

1.2 Indoor location applications

One of the problems appeared in the IoT scope, common to both industry and service purposes, is the localization and tracking of objects inside a building. If the Global Positioning System (GPS) presents as a well-rounded solution for outdoor applications, the indoor counterpart shows its limitations since the satellite signals, in the vastly majority of cases, cannot reach the target reliably.

This presents a big challenge for several applications that have in its core some form of localization logic, such as: tracking of people and objects for security reasons when using an autonomous forklift inside a warehouse or for time reduction reasons when searching a certain product; the creation of a pathfinder algorithm to help people inside big and complicated facilities like oil platforms; personal access control without the usage of tags such as badges; promotional vouchers distribution to clients when they are passing nearby a specific product; crowd control and more.

With indoor positioning systems it is possible to have a wide scope of real applications. Shopping centers, airports, museums and other big or crowded buildings could be easily cited as possible places where individuals that have difficulties in finding what they are looking for would be benefited with a reliable real-time Local Positioning System (LPS).

2 OBJECTIVES

The main objective of this work is to resolve an indoor localization problem by implementing in a test area a fixed number of solutions and finally comparing each one with the intend to find an optimized single solution.

In order to develop an Indoor Positioning System, this work proposes a first general study of well-known methodologies found on the literature, following by a prototyping phase accordingly with the criteria of precision, relatively low cost, modularization and adaptation to different scenarios.

2.1 Implementation area and scenario

A general solution will be implemented and tested on one maneuvering simulator of the Numerical Offshore Tank of the University of São Paulo (TPN-USP). More specifically, at the Full mission simulator, as shown in the Fig.2.

Defined the area, the implementation case proposed is the location of the maritime pilot during a simulation of maneuvering, in which he becomes very active and can walk over long distances.

2.2 Project Requirements

2.2.1 Functional requirements

A LPS prototype capable of determine the maritime pilot's position during a ship maneuvering simulation will be developed with an accuracy of 1.5 meters inside the room. As a tracking record, it should also be able to estimate the walked distance inside the test zone. The refresh rate need to be lower than 10 seconds to provide a smoother real-time experience. Furthermore, for economic reasons, the final cost of the solution should not surpass R$ 800,00.

Figure 2: Full mission simulator projector from TPN-USP.

2.2.2 Non-Functional requirements

2.2.2.1 Prototype requirements

Concerning the operation mode, the prototype's setup and execution process should not present any difficulty to non-specialized persons. In more practical terms, its operational interface should not have lots of setup steps, neither the gadgets should not present any discomfort to the tracking person.

Furthermore, as the simulator room will still be fully operational, all installed gadgets should be smoothly integrated, leaving no gaps that could eventually ending to interfere a maneuvering simulation procedure. Hence, they are forced to be placed, if necessarily, as discrete as possible.

2.2.2.2 Organizational requirements

One minor problem mentioned by various users is the relatively small size of the simulation room, that is restricted for numerous factors, such as economical, physical or computational. The full mission simulator at TPN-USP has around 10 meters from the left side to the right and the itself is a representation of a bridge vessel, however real ones can even reach four times the cited size.

Apart some aesthetic misrepresentations, a major problem detected was when a mar-

itime pilot performs a maneuvering procedure with great quantity of displacements. These movements mainly consist in reaching an extreme side of the bridge for better viewing the vessel's relative positioning. As the simulator room are not big enough, the pilot can sometimes walk a little distance that in reality would need more effort.

Currently, for mimicking this distance effect, a button on each side of the bridge was installed that, when pressed, it applies a zoom effect on the projector's screen. Still, breaking some immersion aspect from the simulation.

Therefore, this project should aim to leave the localization procedure and eventually the tracking information of a target person available and accessible for future integration with the maneuvering simulator itself, such as for automatic view perspective change.

2.3 Project planning

Firstly, a study of what existed on the state of the art topic of Indoor Location Systems. With the knowledge of existed developed solutions, from it, some possible methods were found and only the more interesting ones were tested. Next, a battery of tests was elaborated focusing in reducing even more the solution scope for implementing.

2.4 Text layout

This thesis is organized with a general view of the possible LPS found in the literature on the Chapter 3, interesting remarks from it were given in Chapter 4. The used methodology is described in Chapter 5 following by the used tools in Chapter 6. Some preliminary tests were conducted in Chapter 7, in which demonstrated the difficulties of working on indoor environments. Chapter 8 presented the study validation for an approach with Machine Learning. All previous considerations plus some further tests are evaluated in order to converge into a final solution in Chapter 9, with the results being shown in Chapter 10. Finally, some considerations are made in Chapter 11.

PART II

LITERATURE REVIEW

3 STATE OF THE ART

As well described in works like [9], the Global Positioning System (GPS) although is an excellent solution for most outdoor applications, generating errors of the order of few meters when treating with its indoor counterpart it shows a decrease in accuracy and availability due to the signals used for communicating with the satellites being easily blocked, attenuated or reflected [10]. Therefore, lots of other methods were developed.

In an attempt to organize and focus on some common characteristics found on the solutions, this work suggests the following classification of possible approaches to be adopted:

- In terms of hardware: Device-based and Device-free solutions (3.1);
- In terms of detection method: Range-based and Range-free methods (3.2).

Based on [11], even more sub-classifications can be made within these cited methods as will be discussed in the nexts sections (3.2.1 and 3.2.2).

3.1 Indoor localization: classification by hardware

3.1.1 Device-based solutions

In this case, the target person carries a communication capable device that is responsible to send some piece of information to fixed Access Points (APs) placed around the interested environment.

By adopting this type of solution, the final product may be restricted in more controlled scenarios, where, for example, it would be possible to give a device beforehand to a person. Although this would be a compromise since with a carrying device, the measure reliability, alongside with location accuracy, leans to be higher.

3.1.2 Device-free solutions

As the name suggests, the object or the person whose position it is desired does not carry any type of device. This approach presents as the ideals for scenarios where it is difficult or even impossible to attach a communication device on the target. There are some examples presented in [12] such as elderly monitoring, wildlife protection, emergency or intruder detection.

3.2 Indoor localization: classification by communication method

3.2.1 Range-based methods

Using physical distance as an estimation, optimization models that solve localization problems are generated. Its most used attributes are: Time of Arrival (ToA), Time Difference of Arrival (TDoA) or even propagation models generated from Received Signal Strength (RSS) [11].

3.2.1.1 Time of Arrival (ToA)

In this method, the idea is to have an unlocated device emitting a signal to the anchor nodes, calculate the travel time of the signal and then measure the distance between the device and the nodes. For trace a position, it is imperative to have at least three anchor nodes since the estimated location is the intersection of the circles which are centered on its.

Still, given the environment interference on the signal or the accuracy of the calculations, the circle intersection could be misled and not occur, therefore to use of filtering techniques are needed. Among them, there is a simple Linear Square (LS) estimator implemented in [13] and [14] or a Weighted Least Square (WLS) in [15]. Other more sophisticated solutions can be also found in [16] with the creation of the “Guoguo”, a better version of the authors previous work with the usage of the fine-grained adaptive ToA estimation approach; and in [17], where a Kalman Filter is implemented with an optimal fusion criterion method.

3.2.1.2 Time Difference of Arrival (TDoA)

By taking the exact time of the signal arriving at each receiver, whose position is previously known and since each one of these times will be different, it is possible to infer an estimated distance to the target. For this method, the essential is the relative measurements of time and a synchronization for the receivers.

The signal sender is on a hyperboloid of one receiver and by intersecting the hyperbolic curves of each one, is possible to estimate the sender's location. On surveillance field, this technique can be also referred as Multilateration.

However, problems cited on the previous section such as interference and obstacles are still present, and still filtering techniques are required. Solutions cited before can also be implemented here, but a more interesting one is presented in [18] where the authors implement a mobile beacon node.

3.2.1.3 Propagation model

The Received Signal Strength (RSS)[1] can also be used for measuring distance since the intensity decrease can be correlated with the distance in a path loss lognormal shadowing model. Using the propagation method and the RSS alongside with three base station at least, it is possible to determine an estimated position by Trilateration.

3.2.1.4 Angle of Arrival (AoA)

For this approach, the interested measure is the angle of the arrived signals on the anchor nodes. Using at least two anchor nodes, it is possible to infer the location of the target. Yet, the error is usually higher than other methods since the estimation depends on the inferred region, typically a line, of the received signal and the calculation errors.

3.2.2 Range-free methods

As described in [11], the Range-based methods are heavily accompanied with the method known as Fingerprint. Also called Scene Analysis, it is essentially a pattern recognition method which tries to match patterns by extracting characteristics of a scene from a particular viewpoint [19]. In [11], Du et al. subdivide them in three main cate-

[1] As mentioned in [41], RSSI is a measure scaled from 0 up to 255 with each manufacturer defining its maximum value freely. While RSS is the standardized measure in decibel-milliwatts (dBm).

gories:

- Visual fingerprint-based localization;
- Motion fingerprint-based localization;
- Signal fingerprint-based localization.

3.2.2.1 Visual fingerprint-based localization

By capturing the image of the surroundings, it is possible to track a position with the recognition of a specific landmark. However, alongside with the computing cost for processing the image shows as a great challenge when faced against more real time applications, there is the fact that a continuously operation of a camera would consume too much energy to conceive a practical device design or even an application via smartphones. Also, if there were not enough barriers to overcome, this method cannot reach a high precision [11]. Some other approach includes the detection of a steady image, the usage of Infrared lasers that shows a great challenge to overcome when trying to implement in real non-controlled environments since it is extremely dependable on the line-of-sight.

3.2.2.2 Motion fingerprint-based localization

Sensors such as accelerometers, gyroscope and electronic compasses, which are commonly already built-in motion detector devices on smartphones, are essential for this method. Schindhelm [20] demonstrated that is possible, by combining the information retrieved from the accelerometer and the compass, to calculate an estimation of the location of the device; and Vo and De [21] increase its precision by matching its reading with a map.

Still, these sensors alone could only give the information of displacement, requiring an additional source to situate at the zero-time mark. In navigation, this exact process of calculating the target position by taking a known initial one and advancing it with estimated speeds over the elapsed time is named Dead-Reckoning (DR) [22] - which has its roots on ancient navigators who used celestial observations to determine a reliable position and then correct it for effects like current and wind [23]. Techniques of DR can be classified in two types of groups according to Basiri [9]:

- Plain Inertial Navigation Systems (pINS);

- Step and Heading Systems (SHS).

Along the pINSs, the use of inertial sensors from smartphones is possible due to the increase of accuracy in the past few years, but they still cannot provide proper accuracy due to negative effects, such as heading drift caused by gyroscope bias [24]. An integration with a GPS module is a form to correct the accumulated error suggested by many [21], and a hybrid GPS-Accelerometer-Compass (GAC) system is proposed in [25], in which calibrates infrequently with the GPS, to reduce the drift error, and briefly, to not compromise the energy consumption of the system. Another more directly approach to solve the errors is with the use of Kalman filter and Magnetic Angular Rate Update to collect data with more accuracy, as shown Zampella in [26].

By the other hand, SHS uses estimation of step lengths and heading [9]. Peak-detection, zero crossing, template matching and spectral frequency analysis to detect steps. In [27], it is shown an interesting usage of the accelerometer for a SHS. Instead of simply taking the double-integrate acceleration from the sensor itself, method very susceptible to noises, fluctuations and miscalculations on the displacement; the presented CompAcc identifies a rhythmic acceleration-signature in a person's walking pattern, computes the number of steps and multiplies with and approximation of a step size based on the individual's height and weight.

This approach has some issues like the measurement of the next step position on slippery ground, shuffling and use of elevators, which make the detection zero velocity thresholds or zero angular velocity a big challenge [9].

Attempts based on Statistics and Machine learning were also proposed to solve these problems, examples can be found with the use of Artificial Neuron Network (ANN) [28] and Fuzzy Pedestrian Dead Reckoning (FPDR) system with fuzzy magnetic map matching algorithm [29].

3.2.2.3 Signal fingerprint-based localization

Commonly this method consists of a preliminary offline training phase - which forms a fingerprint database of correlated RSSI patterns from various Access Points (APs) and fix locations - and then followed by an online fingerprint matching phase - when the matching algorithm searches some related RSSI to infer a location. Although APs are commonly used as a specif term for Wi-Fi routers, the method can be implemented with theoretically with any kind of signals, not strictly with WLAN signals.

As presented by Du et al. in [11], there are two approaches to infer this location on the matching phase:

- Deterministic form: where the location is calculated by interpolating different patterns into Euclidean distance;
- Probabilistic form: which correlates directly the distribution signal strength to a fixed position.

Usually, the approach is accompanied by the Weighted k-Nearest Neighbours algorithm, in which considers the level of proximity between the tracking object and the fixed APs, since there are usually more of the latter and also some will not even receive the incoming necessary signal of the object. In the Fig.3, it is shown an overview of a signal fingerprint process approach used by Jekabsons [30].

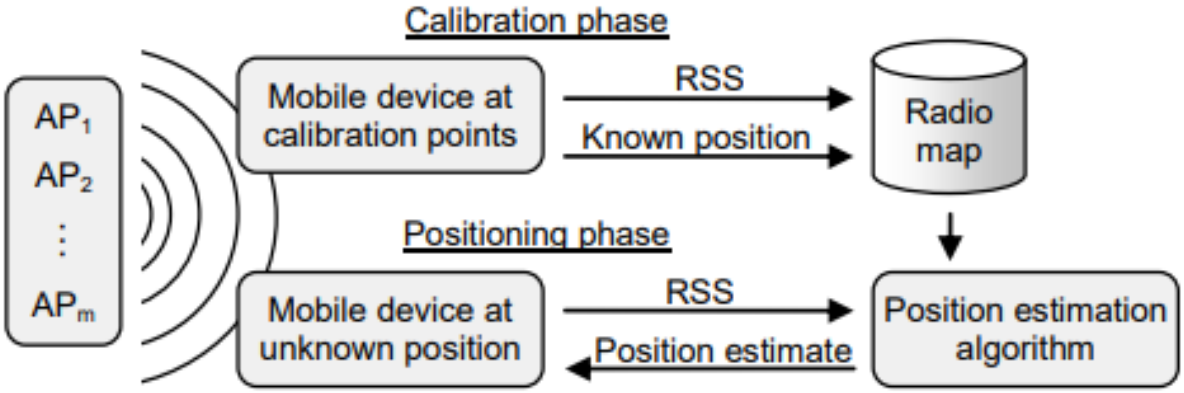


Figure 3: Schematic of the signal fingerprint configuration setup (extracted from [30]).

Jekabsons also find out that the orientation of the moving object offered some major problems with the RSS fluctuations. Altini et al. [31] used a BLE fingerprint-based indoor LPS with multiples ANNs, with its better results reaching 90% precision in 0.5 m of accuracy during specific locations such as corridors, benefiting from system adaptability to better chose the ANN model accordingly with the tracking user's orientation or failure of the base stations and the usage of predictive connection system.

Mehmood, Tripathi and Tipdecho [32] successfully implemented an ANN based solution with a 30% error within 1 meter and 60% within 1-2 meters. Pahlavani, Gholamia and Azimi [33] added the information of orientation and time as inputs in the ANN, achieving an average error of 2.20 meters.

3.3 Radio Frequency signals technologies in LPS

Whatever the method adopted for approaching a LPS, one major characteristic found in almost all of them is the usage of a some type of Radio Frequency (RF) signal to act as an intermediary to obtain some kind of information that will be used to calculate a distance. Some of the most used RF signals implemented for LPS are:

- Radio-Frequency Identification;
- Bluetooth Low Energy;
- Wireless Local Area Network;
- Ultrasound
- Ultra-wide-band

3.3.1 Radio-Frequency Identification (RFID)

It consists of a tag attached to objects of interest that are automatically identify by electromagnetic fields. The tags contain a unique digital signature information, allowing an individual identification of each tag. Typically, the ensemble of the system is composed by tags or transponders, a reader or transceiver, and a host processor unity. Bai et al. suggest that RFID systems can be classified into three main categories [34]:

- Passive systems: In this scenario, the reader emits a radio signal, if the tag enters on the signal field, it will be energized by the signal itself and sends back its identification (ID) and some other data stored. The reader then passes these pieces of information to the main unity, which processes the data, sending to the reader that finally transmits back the processed data to the tag.

Figure 4: Representation of a Passive RFID system (extracted from [34]).

- Semi-passive systems: Analogically to the Passive RFID systems, the principles remain unchanged. Except that here a battery is embedded in the tag, whose sole responsibility is to provide the energy for telemetry and sensor asset monitoring circuits, not for generating Radio Frequency (RF).
- Active systems: For the Active RFID systems, the tags are powered with a battery and transmits periodically data stored, such as location of a merchandise, price, color, date of purchase, etc. The reader catches the ID and other data and cross-reference within a database.

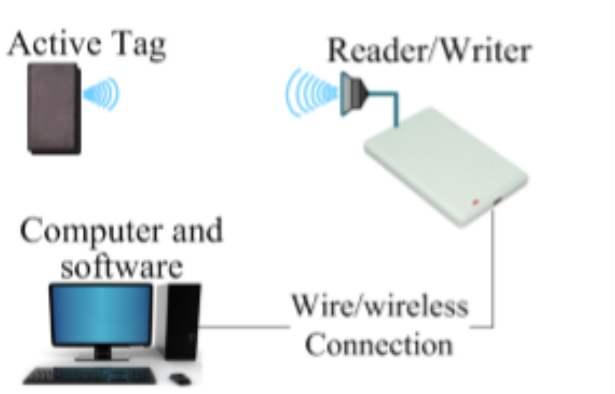


Figure 5: Representation of an Active RFID system (extracted from [34]).

Active RFID systems have a vantage over Passive ones in the operation range and power requirement, since the tags no longer need to be power. Yet, the usage of portable batteries can lead to more maintenance routines.

Although [34] presents as "a superior technique for tracking objects/people", Bai et al. highlight three major issues when using RFID for indoor positioning: one-way com-

munication links, multi-paths effects and unstable RSS. Consequently, hybrid approaches have emerged, as shown in [35] with GPS, in [36] with WLAN and in [37] with UWB. More unorthodox fusions such as with infrared or ultrasound are also cited still in [34], hence focusing the combination solution for increasing the precision and also reducing the final cost.

3.3.2 Bluetooth Low Energy (BLE)

Bluetooth Low Energy (BLE) technology is characterized by five states in the link layer standby, scanning, advertising, initiating and connection. Furthermore, BLE allows basic actions such as packet reception and request for basic information without establishing a connection [38].

Two states are involved in the process of devices search: advertising and scanning which can be used to send packages with 31 bytes of data embedded without the need to be paired. It allows the exchange of data to be used to the indoor location. In [38] it is used the packet delays, which is the use of the time spent to the receiver receive the package from de sender to estimate the distance between both devices; and in [39], the RSS and Transmitting power value to estimate the distance.

3.3.3 Wireless Local Area Network (WLAN)

It is considered by [9] as one of "the most popular positioning technologies provided based on the RF-based technologies". This affirmation could not be considered truer than ever since many buildings already dispose of an existing WLAN infrastructure, thus, creating a solution around this easier and cheaper and making a low-cost application viable as demonstrated in [40].

Despite the fact that most modern WLANs are based on the protocol IEEE 802.11 [41], relying in the method Orthogonal Frequency Division Multiplexing (OFDM) for transmitting the signals, which is present on our daily lives in each electronic device wireless connected, its latest operating commercial versions works in the 5 GHz band. This change reduced the interference with other common products, like wireless keyboards and phones or microwaves, that work in 2.4 GHz, consequently, WLAN now stands as an interesting opportunity for positioning in indoor environments. Usually, the most wanted measures are the signal strength and flight time.

Nevertheless, as remarked in [42], the accuracy of a solution based on WLAN is

around 5 to 15 meters, being less than BLE's counterpart. Furthermore, infsoft - an enterprise specialized on implementing indoor tracking solutions - stated the difficulties of integration of iOS devices.

Other works with this approach include the positioning by fingerprint [43], detaching the attempt to create an adaptive solution, although the necessity of a training phase, and by distance estimation [44], in which proves the reliability of a range-based method.

3.3.4 Ultrasound

An interesting advantage, cited by [45], of the ultrasound signal is the slow propagation speed, the low cost of the transducers and specially the negligible penetration in walls. The last one, considered by many as an inconvenience in terms of range, it is actually very useful when facing indoor environments with lots of walls and, hence, interference of other emitters. The approach given by [45] was based on ToA alongside with a mobile node.

Applying a similar methodology, but using the ultrasound emitters of the smartphones instead, Lazik and Rowe presents its work in [46]. This shift drives the application towards a more practical and widespread implementation. Moreover, according to Plack in [47], the maximum human audible frequency is around 19 to 20 kHz, gap that is exploited by Rowe since the smartphone microphones can work as high as 24 kHz.

Díaz et al. presents a more robust solution in [37], increasing the quality of the emitting signals, and by consequence reducing the appearance of noise. Instead of using the embedded microphone on the smartphone, an external module is plugged to it, overcoming the hardware restriction for maximum sampling rate, of 44.1 kHz, and frequencies - in which some cases can have 20 kHz low-pass filters.

Other more traditional approaches, by focusing essentially the reduction of the noise, such as with the Least Square Method in [48] or with Kalman Filters in [49], can also generate results with precision order of less than one centimeter.

3.3.5 Ultra-wide-band (UWB)

Ultra-wide-band is a wireless technology for transmitting huge amounts of digital data over a wide spectrum of frequency over 500 MHz [50]. Both transmitter and receiver are synchronized with an accuracy of trillionths of a second because the signal frequency is changed periodically, thus, being very useful in terms of security and for not inferring

with other RF devices [51].

UWB can pass over obstacles such as doors and walls that normally would be a problem with other RF technologies. Furthermore, the consumption of energy is low, economizing on the supply part and aiding even more for avoiding interference since with a lower power the pulses of UWB are short enough to not overlap a emitting pulse to most signal reflections [51].

Alarifi et al. [52] present strong arguments for implementing solutions based on UWB in indoor localization applications by comparing with other technologies. One practical example is shown in [14], Conti uses the UWB for allowing very time resolution, essential for a ToA approach.

4 STATE OF THE ART OVERVIEW

Taking into account these papers on the previous part, some approaches were more interesting than others. For facilitating the initial prospects, the solution was designed for a device-based concept. This choice allowed not only a communication establishment between the moving target and a stationary base control system, but also more flexibility in the design aspect since a device-free solutions would restrict the final prototype towards the usage of cameras - which are not so suitable in crowded environments.

Regarding the methods, excluding the visual-based ones, practically all of them were able, in this first general overview, to become viable candidates for this work's purpose. Therefore, some more specific tests were conducted.

PART III

MATERIALS AND METHODS

5 MACHINE LEARNING METHODOLOGY

The concept of Machine Learning goes back to the first study attributed to Arthur Samuel in 1959 [53], in which he explored what it would be later the central pillar of Machine Learning: the idea of giving computers the ability to learn without being explicitly programmed [54].

A more updated definition is given by Tom Mitchell [55], who precises the scope of a well-posed learning problem as:

> A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E.

Further in this monograph the functionality and the theory behind each of the tested algorithms is described, along with the reasons to chose each one.

5.1 Support Vector Machine

Support Vector Machine (SVM) is one type of machine learning algorithm based on the statistical learning theory. Developed by Vapnik [56] after initial studies found in [57]. The official documentation for the Open Source Computer Vision Library (OpenCV) presents SVM as follows [58]:

> A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. In other words, given labeled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorizes new examples.

Taking for instance a 2D scenario where two distinct groups are separated as shown in the Fig.6, the hyperplane is a simple linear function.

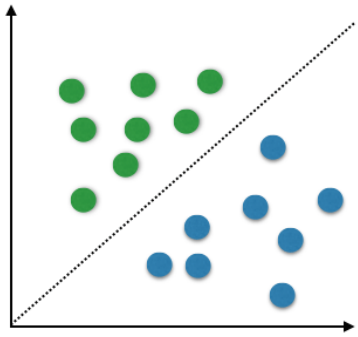


Figure 6: Representation of a linear SVM scenario.

The procedure to determine the optimal hyperplane is given by maximizing the distance between the closest data points of each class dataset. These used points are called the *support vectors* as shown in Fig.7.

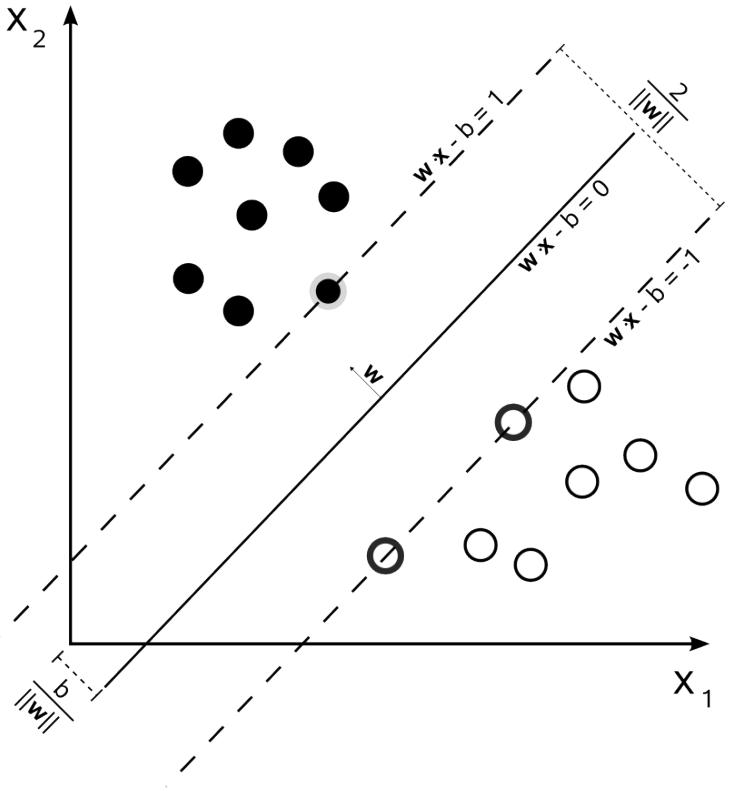


Figure 7: Representation of the margin calculation process.

Mathematically [59], being $\vec{w}$ a perpendicular vector to the hyperplane, a sample is classified accordingly with:

$$\begin{cases} \vec{w} \cdot \vec{x_a} + b \geq 1 & \text{for a classification group A} \\ \vec{w} \cdot \vec{x_b} + b \geq -1 & \text{for a classification group B} \end{cases} \tag{5.1}$$

Simplifying the decision rule equation 5.1 into a single equation:

$$y_i(\vec{x_i} \cdot \vec{w} + b) - 1 \geq 0 \begin{cases} y_i = 1 & \text{if } x_i \in A \\ y_i = -1 & \text{if } x_i \in B \end{cases} \tag{5.2}$$

Taking $\vec{x_+}$ and $\vec{x_-}$ as vectors to the extreme points of the segregating hyperplane (*support vectors*), the total margin's size will be given by:

$$\text{Margin's width} = (\vec{x_+} - \vec{x_-}) \cdot \frac{\vec{w}}{||w||} = ((1-b) - (-1-b))\frac{1}{||w||} = \frac{2}{||w||} \tag{5.3}$$

For the criteria adopted in equation 5.1 to be true, the margin's width needs to be maximum. Inverting it, the problem becomes a Quadratic Programming Problem, allowing efficiently solving.

$$\begin{cases} \min_{\{b,\vec{w}\}} \frac{||w||^2}{2} \\ \text{subject to } y_i(\vec{x_i} \cdot \vec{w} + b) - 1 \geq 0 \end{cases} \tag{5.4}$$

Applying the method of Lagrange Multipliers in equation 5.4:

$$\mathcal{L} = \max_{\{b,\vec{w}\}} \left\{ \frac{||w||^2}{2} - \sum_{i=1}^{N} \alpha_i [y_i(\vec{x_i} \cdot \vec{w} + b) - 1] \right\} \text{ with } \alpha_i \geq 0 \tag{5.5}$$

Calculating the partial derivatives of $\vec{w}$ and b:

$$\begin{cases} \frac{\partial L}{\partial \vec{w}} = \vec{w} - \sum_{i=1}^{N} \alpha_i y_i \vec{x_i} = 0 \Rightarrow \vec{w} = \sum_{i=1}^{N} \alpha_i y_i \vec{x_i} \\ \frac{\partial L}{\partial b} = -\sum_{i=1}^{N} \alpha_i y_i = 0 \Rightarrow \sum_{i=1}^{N} \alpha_i y_i = 0 \end{cases} \tag{5.6}$$

Returning equation 5.6 back to 5.5:

$$\begin{cases} \mathcal{L} = \max_{\{\alpha_1, \cdots, \alpha_N\}} \left\{ \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \boxed{x_i \cdot x_j} \right\} \\ \text{s.t. } \alpha_i \geq 0, \ \ \vec{w} = \sum_{i=1}^{N} \alpha_i y_i \vec{x_i}, \ \ \sum_{i=1}^{N} \alpha_i y_i = 0 \end{cases} \tag{5.7}$$

By finding $\alpha_1, \cdots, \alpha_N$ that maximize $\mathcal{L}$, it is easy to return to equation 5.6 to find the value of $\vec{w}$ and b, the latter one with the decision rule at equation 5.2. Furthermore, it is proven that this maximum is unique, differently than ANNs, detailed on the next section, in which various solutions are possible.

5.1.1 Kernel trick

However, although until this point SVM was already a well defined theory, there were still some scenarios where the separating hyperplane became too complex to a certain degree in which the speed and convergence of the classification algorithm became almost unfeasible.

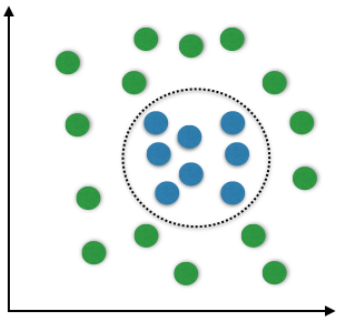


Figure 8: Scenario where a Kernel trick is used in SVM.

Taking Fig.8 for instance, the data could be better separated if mapped on a higher space dimension. Being ϕ this transformation $\phi : \mathcal{X} \rightarrow \mathcal{X}'$, the equation 5.7 becomes:

$$\begin{cases} \mathcal{L} = \max_{\{\alpha_1,\cdots,\alpha_N\}} \left\{ \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \boxed{\phi(x_i) \cdot \phi(x_j)} \right\} \\ \text{s.t. } \alpha_i \geq 0, \;\; \vec{w} = \sum_{i=1}^{N} \alpha_i y_i \phi(\vec{x_i}), \;\; \sum_{i=1}^{N} \alpha_i y_i = 0 \end{cases} \tag{5.8}$$

With equation 5.8, it is possible to better understand how computationally heavy the process would be, with each iteration needing to first transformed by $\phi(x_i)$. To solve this practical problem, in 1992, Boser, Guyon and Vapnik [60] introduced the usage of kernels into SVM by:

$$K(x_i, x_j) =< \phi(x_i), \phi(x_j) > \tag{5.9}$$

Now, knowing the spacial transformation ϕ was not more necessarily, hence, reducing the total computational process time during training and allowing more complex analysis.

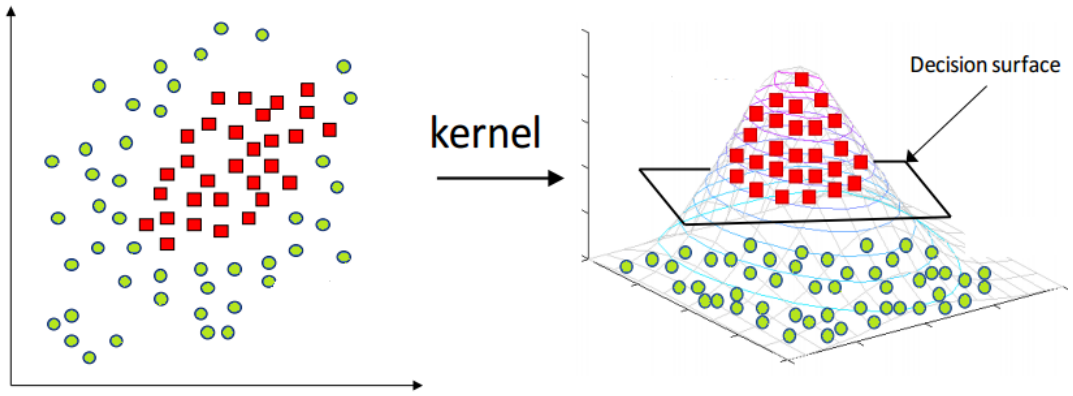


Figure 9: Kernel trick spatial representation (extracted from [61]).

Various possibilities for the kernel function exist, but the one used on this work was the *Radial Basis function kernel*. As described by Chang and Lin in [62], an advantage of this function is the smooth frontier presented between the labeled data – making it possibly useful when classifying normal distributed data.

$$K(x_i, x_j) = exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right) = exp(-\gamma||x_i - x_j||^2) \text{ with } \gamma > 0 \tag{5.10}$$

This γ hyperparameter multiplies the distance between the hyperplane and the support vectors. Low values of γ considers dataset even far away from the decision boundary, while high values end with lowering the influence of these distant data.

5.1.2 Soft margin

Alongside with the previous hyperparameter γ, Cortes and Vapnik [63] presented another hyperparameter C, in which played with the margin by ignoring some data and misclassifying them if too inseparable.

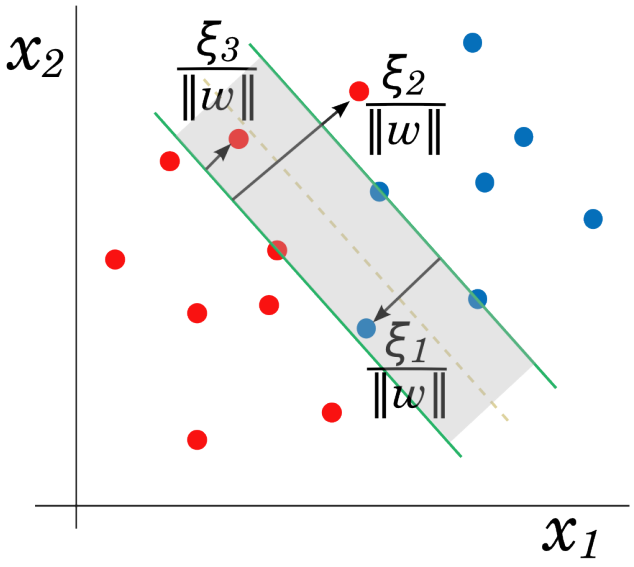


Figure 10: Scenario where some data is purposely misclassified.

Being ξ_i a slack variable, the idea is to minimize its influence, hence its Primal form equation 5.4 becomes:

$$\begin{cases} \min_{\{b,\vec{w}\}} \left\{ \frac{\|w\|^2}{2} + C \sum_{i=1}^{N} \xi_i \right\} \\ \text{s.t. } y_i(\vec{x_i} \cdot \vec{w} + b) - (1 - \xi_i) \geq 0, \quad \xi_i \geq 0 \end{cases} \tag{5.11}$$

Rewriting in the Dual form and adding the Kernel trick, the SVM problem becomes finally:

$$\begin{cases} \mathcal{L} = \max_{\{\alpha_1,\cdots,\alpha_N\}} \left\{ \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \right\} \\ \text{s.t. } 0 \leq \alpha_i \leq C, \;\; \vec{w} = \sum_{i=1}^{N} \alpha_i y_i \phi(\vec{x_i}), \;\; \sum_{i=1}^{N} \alpha_i y_i = 0 \end{cases} \tag{5.12}$$

Both γ and C hyperparameters can be estimated using grid search and evaluated with a cross-validation process [64].

5.1.3 Multi-class classification

When treating with more than two-class classification problem, there are some general strategies for these machine learning problems. A better suited for SVM due its own mathematical nature is the "one-against-one" strategy, in which consists in constructing

one SVM for each pair of classes, thus a problem with n classes will have $\frac{n(n-1)}{2}$ SVMs trained [65].

Again, Chang and Lin [62] resolve this multi-class classification problem with the usage of LIBSVM, a Library for SVM developed by themselves.

5.2 Artificial Neural Networks

Inspired by the biological neural networks found in animals' brains, Artificial Neural Networks (ANNs) are a general framework designed to process complex data inputs by eventually allowing different machine learning algorithms work at the same time. It is widely used in various fields, such as speech and image recognition, finance, medical diagnosis, etc.

Being one of many approaches used in Machine Learning algorithms, an interesting feature present here is the fact that ANN learns from processing many examples without specific rules into the inputs. For instance, in an image recognition problem, when training the ANN for identifying a specific object such a cat, the only exigence will be the correct label into the images of "cat" and "no cat". The ANN will generate identifying characteristics without the prior knowledge about cats, such as if they have fur or tails.

5.2.1 Network Architecture

As shown in the Fig.11, ANNs are composed by nodes called "Artificial Neurons", that will be discussed later, and connections between them, or "edges". Usually, ANNs are aggregated into layers:

- Input layer (first layer): where the system input enters. It can be subdivided into several other artificial neurons accordingly to the transformation required.
- Hidden layer (middle layer): in which composes the application of the analysis method adopted.
- Output layer (last layer): the answer that is expected from the system.

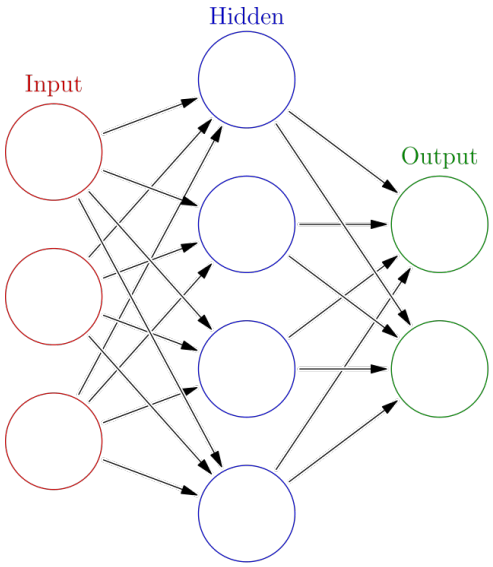


Figure 11: General representation of ANN.

Each artificial neuron and edge have a weight and the general idea of this approach is that it adjusts as the learning proceeds. The more example the ANN is exposed, alongside corrective feedbacks about its predictions, better will be the ending result [66].

Still, it is possible to freely increase the complexity of an ANN by adding countless number of layers and reshaping the input data. Most deep learning models are based on an ANN and some examples can be found in:

- Image recognition: with the usage of convolutional layer with *MaxPooling* [67];
- Timed series data: with the recurrent layers;
- Real time object detection: with the *Darknet* framework and applied in the YOLOv3 (You Only Look Once) algorithm [68].

5.2.2 Artificial Neurons

Conceived from the biological neurons counterpart, an artificial neuron is a mathematical function which process the sum of the multiplied by a weight, producing an output.

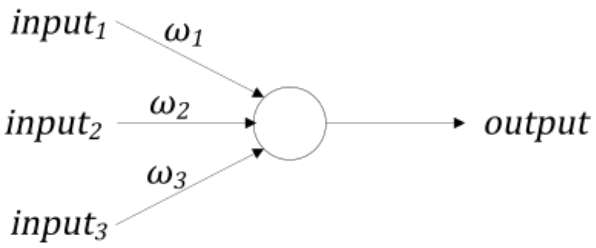


Figure 12: General representation of an artificial neuron.

5.2.3 Activation function

Also being called as transfer function, it generates the output signal in a neuron given a single or set of inputs. In this work, four of them were used: Rectified Linear Unit (ReLU), Exponential Linear Unit (ELU), Sigmoidal and Softmax.

5.2.3.1 Rectified Linear Unit (ReLU)

The ReLU function is defined as the positive part of its argument. Taking x as the neuron input, the output $f(x)$ is given by:

$$f(x) = x^+ = max\,(x, 0) \tag{5.13}$$

Although simple, the advantage presents with the constant gradient, resulting in a faster learning rate. The sparsity generated by negative inputs ($f(x) = 0, \forall x \leq 0$) and cheap computing power required are factors for being interesting on ANN as hidden layers.

5.2.3.2 Exponential Linear Unit (ELU)

Similar to ReLU on its positive inputs, ELU is defined as:

$$f(x) = \begin{cases} \alpha(e^x - 1) & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases} \tag{5.14}$$

With the additional parameter α being positive. Since ELUs can have negative values, it pushes the mean of the activations closer to zero, resulting in an even faster learning and convergence.

5.2.3.3 Sigmoidal function

Often referred as a special case of the logistic function, the sigmoidal function is given by the formula:

$$f(x) = \left(\frac{1}{1+e^{-x}}\right) \tag{5.15}$$

Due to the fact its smooth derivative [69] and and for appearing commonly in Statistics as a cumulative distribution function, its usage could be interesting for a normal distribution dataset.

5.2.3.4 Softmax function

The softmax function is a generalization of logistic regression that normalizes a K dimensional vector z of arbitrary real values into another same size vector $\sigma(z)$, whose components sums to 1, in other words, $\sigma(z)$ is a probability vector [70]. Mathematically, the equation is given by:

$$\sigma(z)_j = \left(\frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}\right) \tag{5.16}$$

Translating to this works variables, $\sigma(z)$ becomes the neuron output $f(x)$, while x is the input vector of size K. Usually, this function is used as the last layer with cross-entropy loss due to its output representing an accuracy ratter than a distance (such as the mean square error). Hence, being very useful on a multi-class classification [71] [72].

5.2.4 Multi-class regression network

A multi-class regression network is a ANN which create a regression model that receive some inputs which describe something like an incoming, pixel colour, age, gender, job and give a output describing a class, like a flower or a animal specie, social class. It is called regression network because the model is calculated by a regression, using labeled data associated to some inputs, by monitoring some parameter like minimizing a minimum quadratic error or maximizing the model accuracy, that are improved by the use of optimization algorithm.

To create a network capable to make a multi-class regression it is possible to follow some approaches, like the one describe by the article LSTM Fully Convolutional Networks

for Time Series Classification [73], but due to the kind of input and output data which will be describe on the chapter IV, a architecture using categorical cross-entropy Loss with relu, sigmoidal and softmax layers, as commonly described by the ANN community [67], [74], [75].

On the following section the key parameters used to build the multi class regression is briefed explained.

5.2.5 Loss

The loss function calculates the parameter used to evaluate network such as the Mean square error, or the model accuracy. Since the fingerprint solution a multi-class classificator is desired, the chosen loss function was the categorical cross-entropy. This function rewards/penalises probabilities of correct classes only, which is the what the model to resolve this works problems is about. The formulation of this loss function is represented by the equation 5.17, which N represent the number of train data, $\hat{Y}$ represent a vector contain the probability of a given input to be one of the output class and Y represent a one-hot encoded, or the desired vector contain the number one, representing 100% confidence that a given input is a output class. [76]

$$J = \left(-\frac{1}{N} \left(\sum_{i=1}^{N} y_i \log(\hat{y}_i) \right) \right) \tag{5.17}$$

5.2.6 Optimizer

Optimizer algorithms are responsible to shape the weights and bias from each neuron, by trying to minimize or maximize an Objective function defined by internal learnable parameters such as accuracy or quadratic error [77]. In this work, the following algorithms were used:

- Stochastic Gradient Descent (SGD);
- Root Mean Square Propagation (RMSProp);
- Adaptive Moment Estimation (Adam);
- Nesterov-accelerated Adaptive Moment Estimation (Nadam).

5.2.6.1 Stochastic Gradient Descent

For the SGD, the model's parameters θ are updated following the equation described:

$$\theta = \theta - \eta \cdot \nabla_\theta J(\theta, x^{(i)}, y^{(i)}) \tag{5.18}$$

Where η refers to the learning rate, which determines the size of each step; $J(\theta)$ is the objective function and $\nabla_\theta J(\theta)$ is the objective function's gradient with respect to the parameters. The difference towards the Batch Gradient Descent is that SGD takes into account each training example $x^{(i)}$ alongside its label $y^{(i)}$.

5.2.6.2 Root Mean Square Propagation

The RMSProp is an improved version that resolves Adagrad's radically diminishing learning rates by updating recursively as a decaying average of all past squared gradients $E[g^2]$.

$$\begin{cases} E[g^2]_t = 0.9E[g^2]_{t-1} + 0.1g_t \quad \text{with } g_{t,i} = \nabla_{\theta_t} J(\theta_{t,i}) \\ \theta_{t+1} = \theta_t + \frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} g_t \end{cases} \tag{5.19}$$

With ϵ being a smoothing term that avoids the division by zero. Ruder [78] indicates $\eta = 0.001$ as a good default value for the learning rate.

5.2.6.3 Adaptive Moment Estimation

Adam is an optimizer that computes adaptive learning rates for each parameter while storing an exponentially decaying average of past squared gradients v_t – such as Adadelta and RMSProp – and storing another exponentially decaying average of past gradient m_t – similar to momentum.

$$\begin{cases} m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \\ v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \end{cases} \tag{5.20}$$

And to counteract the bias generated from the initialization vectors being zeros at start and the provoked small decay rates on initial steps, equation 5.20 is then further incremented to:

$$\begin{cases} \hat{m}_t = \frac{m_t}{1-\beta_1^t} \\ \hat{v}_t = \frac{v_t}{1-\beta_2^t} \end{cases} \tag{5.21}$$

Finally, the parameters update is given by:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t \tag{5.22}$$

It is mentioned in [78] that good empirically values are $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$ and the Adam optimizer works well in practice and compares favorably to other adaptive learning-method algorithms.

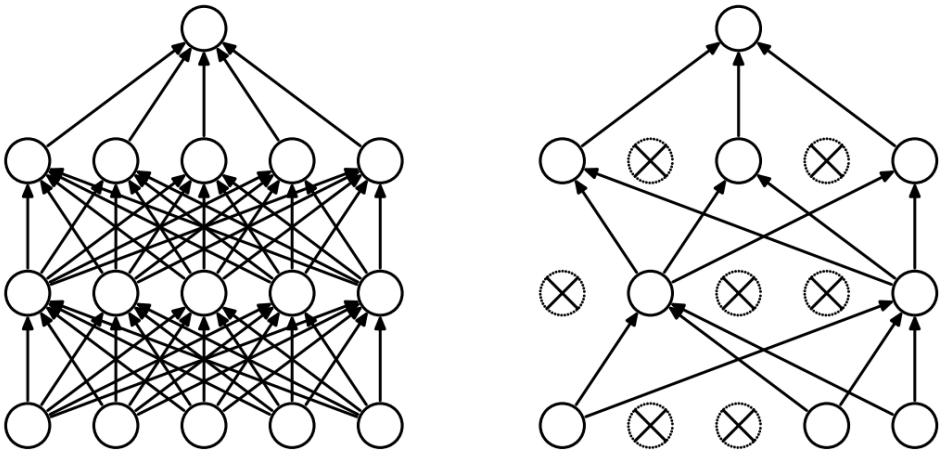
5.2.6.4 Nesterov-accelerated Adaptive Moment Estimation

Nadam is a combination of the Nesterov-accelerated gradient into Adam optimizer. In short, the parameter update expression is given by:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \left(\beta_1 \hat{m}_t + \frac{(1-\beta_1) g_t}{1-\beta_1^t} \right) \tag{5.23}$$

5.2.7 Dropout

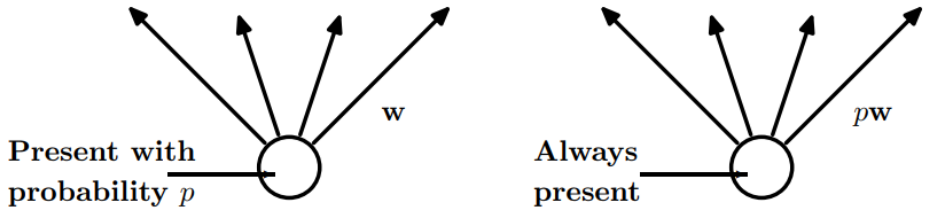
When a network is too closely fit to a limited set of input data, an error called *Over-fitting* occurs. One countermeasure against it is the usage of a regularization technique named *Dropout*. As first introduced in [79], the term "refers to dropping out units (hidden and visible) in a neural network", or more directly, temporary removing entirely a neuron's input and output from the network as shown in Fig.13 with crossed units representing dropped neurons.



(a) Standard Neural Net (b) After applying dropout.

Figure 13: Dropout visual representation extracted from [79].

A dropout happens randomly during uniquely the training phase and accordingly to a fixed probability p, a tunable hyperparameter (Fig.14). This rate is fixed to all neurons and will repeatedly occur at each training step. Ended this initial phase, the neuron's calculated weight w will be then multiply by p, ensuring that for any hidden unit the expected output will be the same as the actual output at test time.



(a) At training time (b) At test time

Figure 14: Dropout during training phase and at test scenario (extracted from [79]).

Srivastava et al. [80] reveled that the application of the dropout technique can improve ANN's performance on fields such as vision, speech recognition, document classification and computational Biology, yet with a drawback of increasing training time.

6 TOOLS UTILIZED

6.1 Hardware components

For implementing this work's solution, it was proposed the usage of five NodeMCUs, one ESP32 and one Raspberry Pi 3. All of them are easily interchangeable in the operation view point for any other same type of device, however, they were selected for its accessibility on the market and relative low price.

6.1.1 NodeMCU

NodeMCU is an open source IoT platform and development kit based on the chip ESP8266 from Espressif Systems, in which integrates General Purpose Input/Output (GPIO), reserved pins for communication Inter-Integrated Circuit (I2C), Serial Peripheral Interface (SPI), Analog-to-Digital Converter (ADC) and other more features. It is specially interesting for its low cost, its Application Programming Interface (API) being quite similar to the Arduino and for its WiFi connection capabilities [81].

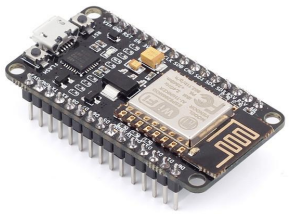


Figure 15: A NodeMCU board.

6.1.2 ESP32

Being the sucessor of the ESP8266, ESP32 is a series of low-cost, low-power system microcontrollers with integrated WiFi and dual-mode Bluetooth created and developed by Espressif Systems [82]. The development kit has 4 MB of flash memory and a CPU

Xtensa® Dual-Core 32-bit LX6, presenting as a more powerful solution if compared with its predecessor, yet more expensive.

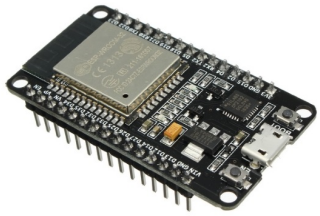


Figure 16: An ESP32-dev board.

6.1.3 Raspberry Pi 3

Developed by the Raspberry Pi Foundation, it was firstly introduced as a tool for teaching basic computer science, soon it became widely spread for its low price and its broad application purposes within IoT. It was used the 3^{rd} generation model B, one of the most recent available models. This model has a Quad Core processor operating in 1.2 GHz, Broadcom BCM2837 64-bit, with 1 GB of RAM and wireless and BLE integrated [83].

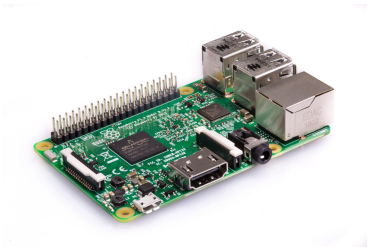


Figure 17: A Raspberry Pi 3 board.

6.2 Software

6.2.1 PlatformIO

PlatformIO is an open source integrated development environment for IoT, having on its repertory thousands of the most used libraries for multiple architectures and embedded devices. Besides some interesting features such as the intelligent code completion and smart code linter, it allows multi-projects workflow and a cross-platform build system that enables the usage of various boards and frameworks without any external dependencies

[84].

For this work, the version chosen one was the PlatformIO for Microsoft's Visual Studio Code.

6.2.2 Scikit Learn

Scikit-Learn is a Open-source library for python, which contain simple and efficient tools for data mining and data analysis like, classification, regression, clustering, dimension reduction, model selection and prepossessing [85]. In this work some of the algorithms like Gaussian mixture models, cross-validation and the support vector machine was used from this library.

6.2.3 TensorFlow

Originally developed by researches and engineers from the Google Brain team within Google's AI organization, TensorFlow is an open-source software library for dataflow programming across a range of tasks.

PART IV

DEVELOPMENT

7 PRELIMINARY TESTS

Once filtered the more interesting approaches, the remaining ones are still numerous and if considered until the development phase, it would generate a huge branch of practical solutions, with some of them being only little cosmetic different one from another. For instance, there is no such big impact on the scope of the procedure A itself by changing tool B for C if procedure A is used on both.

Translating for this work ambit, the selection of signal type is merely a tool for validating an improved general methodology for a LPS. Notwithstanding, the focus will be on the methods for calculating the distance itself.

7.1 Signal type chosen

The selection of the most reliable type of signal was considered the first and most important bottleneck, since it will determine which components are going to be used and the code's core shape.

Both BLE and WLAN were by far the top ones to be considered since its usage on small IoT applications is well known and widely spread, turning its devices easily buyable and accessible.

On the other hand, the RFID, although very commonly found on badges or goods, its short range functionality restricts the operations around the RFID sensors, ending by acting merely as pass detection devices. Hence, if looking for tracking applications, the sensors would have a considerable increase in cost for its higher robustness and specifications.

Also, the usage of UWB and Ultrasound fall on the same problem of elevated costs, although not as much as the RFID, both would present a scalability problem when treating bigger environments. Since the signal's strength is correlated to the distance between emitter and receiver, more and more sensors are required as the size of the control area

increases.

In the case of the UWB, the problem is even worse since its reach is lesser than the normal WLAN, leading to a necessity to have even more sensors, that are already more expensive. Its gain would be restricted to only more marginal signal's reliability since its *modus operandi* is analogous to the WLAN.

For the Ultrasound, besides its economic concerns, an additional challenge is presented when faced a crowded scenario. The property of not penetrating in walls [45] - although being ideal in terms of reliability and consistency inside buildings with lots walls or fixed objects - when facing moving obstacles that cannot be parameterized presents another difficulty. Since its signal is easily disrupted by physical obstacles, a reliable communication cannot be guaranteed these dynamic environments.

7.1.1 BLE vs WLAN signal

Despite being both capable for being implemented, one should be selected as the sole communication method for unifying the code development process. Thus, a reliability test was proposed to eliminate the less prominent one.

The test consisted in fixing a signal emitter device, getting the measurements of RSS from a receiver during a certain time window with known various instances of distance. Alongside the analysis for the type of signal, the Range-based method of Propagation Model was tested for evaluate its efficiency in indoor environments.

7.1.1.1 Propagation method study formulation

A distance measurement with WLAN signal by using the Propagation Model is studied on [88], in which Basai cited three ways for calculating it given a RSSI (or RSS): the first one is a simple curve fitting that is self-explanatory, while the other two are based on different models.

One of these is the Estimated Signal Strength (ESS) formula, which estimates the distance d from a parameterized one (d_0) with known RSSI ($RSSI_0$):

$$d_{ESS} = d_0 \times 10^{\left(\frac{RSSI - RSSI_0}{-10\gamma}\right)} \tag{7.1}$$

Where γ is the path-loss exponent, being $\gamma = 2$ for free space case and $\gamma = 4$ for mobile devices. Al Qathrady and Helmy [86] also used this same formula for the BLE

counterpart.

Lastly, there is the Friis transmission equation for free space propagation, given by:

$$d_{Friss} = \sqrt{\frac{P_t G_t G_r \lambda^2}{(4\pi)^2 P_r}} \tag{7.2}$$

Where G_t is the transmitter antenna gain, G_r, the receiver antenna gain, P_r, the receive signal power, P_t, the transmitted signal power and λ, the wave length in meter. In the study of Basai [88], the adopted were $G_t = 1$, $G_r = 1$, $\lambda = 0.125m$ (for 2.4GHz) and:

$$P_t = VI = 3.3V \times 1A = 3300mW \tag{7.3}$$

Converting into decibel-milliwatts:

$$P_t(dBm) = -10.log_{10}\left(\frac{3300mW}{1mW}\right) = -35.19 \tag{7.4}$$

Then simplifying the equation 7.2 and preparing for a received signal power in dBm:

$$d_{Friss} = \sqrt{\frac{35.19dBm \times (0.125m)^2}{(4\pi)^2 P_r}} \tag{7.5}$$

The propagation of uncertainty, or Standard Sample Deviation (SSD), (σ_d) into a variable $d = f(a_1, a_2, ..., a_n)$, taking into account all its a_1, a_2, ..., a_n parameters alongside with its respectively uncertainties σ_{a1}, σ_{a2}, ..., σ_{an} , is calculated as follows:

$$\sigma_d = \sqrt{\left(\frac{\partial d}{\partial a_1}\right)^2 \sigma_{a_1}^2 + \left(\frac{\partial d}{\partial a_2}\right)^2 \sigma_{a_2}^2 + \cdots + \left(\frac{\partial d}{\partial a_n}\right)^2 \sigma_{a_n}^2} \tag{7.6}$$

On the EES formula, the error propagation is from both the parameterized and the tested measure:

$$\frac{\partial d_{ESS}}{\partial RSSI} = d_0.10^{\frac{RSSI-RSSI_0}{-10\lambda}} log(10)\left(\frac{1}{-10\lambda}\right) \tag{7.7}$$

$$\frac{\partial d_{ESS}}{\partial RSSI_0} = d_0.10^{\frac{RSSI-RSSI_0}{-10\lambda}} log(10)\left(\frac{1}{10\lambda}\right) \tag{7.8}$$

$$\sigma_{d_{ESS}} = \frac{log(10)d_{ESS}}{10\gamma}\sqrt{\sigma_{RSSI}^2 + \sigma_{RSSI_0}^2} \tag{7.9}$$

And on the Friis transmission equation, the error is only from the measured RSSI itself:

$$\frac{\partial d_{Friis}}{\partial P_r} = d_{Friss}\frac{-1}{2.P_r} \tag{7.10}$$

$$\sigma_{Friis} = \frac{d_{Friis}}{2.P_r}\sigma_{Pr} \tag{7.11}$$

For the BLE signal, another regular parametric model used on [86], besides the equation 7.1 was tested on this work.

$$d = A \times \left(\frac{RSSI}{RSSI_0}\right)^B + C \tag{7.12}$$

Where A, B and C are coefficients that varies with the device used. For instance, Ma et al. [87] used the following values for a Nexus 5: $A = 0.4203$, $B = 6.9476$ and $C = 0.54992$. Other formulations were presented such as with the generated from a regular machine learning method:

$$d = D \times RSSI + E \tag{7.13}$$

In which D and E are found after a training section.

With the difficulty on determine the correct value for the device used on the test of this work, the equation 7.12 was abandoned. And any other form of evaluation by the usage of a previous training phase was also discarded, such as the equation 7.13, since it would infer a correction bias generated from the nature of the methods. Hence, remaining only the equation 7.1 also tested for the WLAN.

7.1.1.2 Comparison results

For the WLAN signal study, the microcontroller ESP32 was set as the receiver of an open WLAN created by another microcontroller, the NodeMCU - same as used in [88]. The data exchanged were restricted only to a minimum in order to maintain a simple WiFi connection without any other important service operating on the background. With an approximated scan test time of 3 minute test, the total sample number was 674 measurements and the environment was of course indoors.

Control distance	**mean RSS**	SSD	**ESS distance**	ESS error	**Actual Error**
0.2 m	-26.13 dBm	1.234 dBm	0.39 m	0.05 m	94.3%
0.4 m	-30.03 dBm	1.256 dBm	0.57 m	0.09 m	43.3%
0.5 m	-30.73 dBm	1.532 dBm	0.67 m	0.10 m	34.9%
0.7 m	-38.36 dBm	1.511 dBm	0.45 m	0.07 m	35.9%
1.0 m	-37.42 dBm	1.298 dBm	1.00 m	0.14 m	-
1.5 m	-52.99 dBm	1.506 dBm	0.74 m	0.11 m	50.8%
2.0 m	-52.99 dBm	3.156 dBm	0.66 m	0.08 m	66.9%

Table 1: WLAN signal study measurement with ESS method

Control distance	**mean RSS**	SSD	**Friss distance**	Friss error	**Actual Error**
0.2 m	-26.13 dBm	1.234 dBm	0.00831 m	0.00001 m	95.8%
0.4 m	-30.03 dBm	1.256 dBm	0.00780 m	0.00013 m	98.0%
0.5 m	-30.73 dBm	1.532 dBm	0.00762 m	0.00012 m	98.5%
0.7 m	-38.36 dBm	1.511 dBm	0.00811 m	0.00010 m	98.8%
1.0 m	-37.42 dBm	1.298 dBm	0.00722 m	0.00010 m	99.3%
1.5 m	-52.99 dBm	1.506 dBm	0.00752 m	0.00011 m	99.5%
2.0 m	-52.99 dBm	3.156 dBm	0.00764 m	0.00007 m	99.6%

Table 2: WLAN signal study measurement with Friis method

On the second signal study, for the BLE, the ESP32 became the emitter this time, while an android smartphone was put on the place of the receiver. This was due to the fact that NodeMCUs does not have an BLE module. Again, on the same test environment, with the approximated scan time, the total number of samples obtained was 359.

Control distance	**mean RSS**	SSD	**ESS distance**	ESS error	**Actual Error**
0.2 m	-64.3 dBm	1.339 dBm	0.29 m	0.05 m	42.8%
0.4 m	-68.7 dBm	1.074 dBm	0.47 m	0.07 m	18.0%
0.5 m	-72.3 dBm	1.167 dBm	0.72 m	0.12 m	43.5%
0.7 m	-72.5 dBm	1.089 dBm	0.73 m	0.12 m	3.86%
1.0 m	-75.2 dBm	0.796 dBm	1.00 m	0.13 m	-
1.5 m	-72.3 dBm	0.737 dBm	0.71 m	0.09 m	52.7%
2.0 m	-72.3 dBm	0.279 dBm	0.72 m	0.07 m	64.3%

Table 3: BLE signal study measurement

7.2 Conclusions on distance measurement methods

Within the Range-based methods already discussed on 3.2.1, the first one that can be discarded is the AoA for its higher inferred errors. Both ToA and TDoA presents some

problems around its core physical principle. As the light speed is extremely high, the whole system architecture would need to be exceptionally precise in order to capture such slight time variation. Some ideas for delaying the reach time, for instance the creation of a logic for looping n times the sending and receiving of a signal, are hugely affected by the own components' internal clock or the randomness of signal disruption of an indoor environment.

Lastly, the Propagation Model, which was tested on indoor location in 7.1.1, shown very poor results, which is explained by the data analysis made on the section 8.3, proving its usage being of exceedingly high level of difficulty, which reason will be enplaned on the section 8.3. Therefore, the Fingerprint method, both Motion and Signal, will be focused from now on.

7.3 WLAN advantages over BLE

Despite the inconclusive results in regards of distance calculation presented in 7.1.1.2, an interesting piece of information can be extracted: the WLAN can reach farther, since the mean WLAN RSS is lower than the mean BLE RSS from a same distance. This can incur in a data extraction with less noise for the subsequent test.

Moreover, looking at the data dispersion from the WLAN, as it can be seen on the Fig.18, it seems to appear a Gaussian distribution, with the orange bar representing the frequency histogram and the blue curve what it would be a real normal distribution given the mean and standard deviation of the sample.

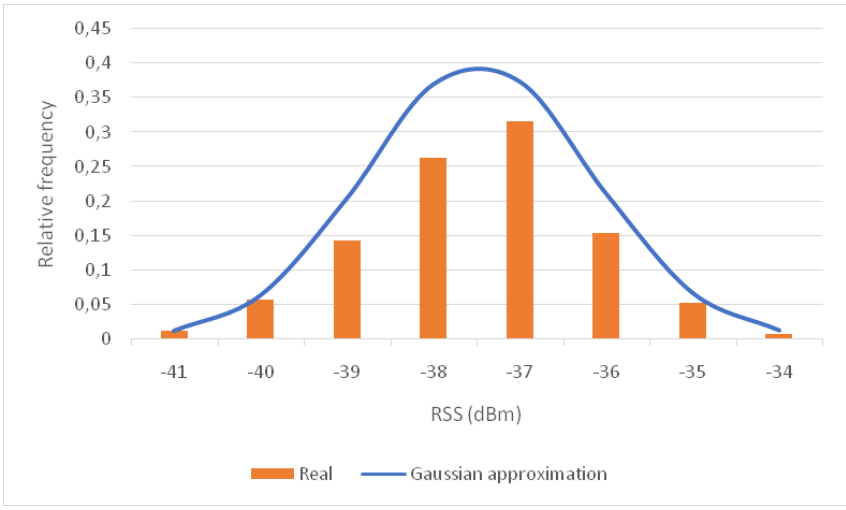


Figure 18: Sample data from a WLAN measurement.

Also, concerning the user usability for the final prototype, the usage of WLAN can be more convenient. An usage of BLE implies necessarily an active connection from the user to a central processing unity, and with WLAN, there is a possibility of using the probe mode - in which a device searches for available WiFi - or even the data exchange to the internet of a device connected carried with the user. Therefore, acting as a passive solution, in which the user would not need to do anything for the tracking system to work.

8 FINGERPRINT METHOD ANALYSIS

Notwithstanding the poor results given with the Propagation Model in 7.1.1, the next test had the Fingerprint method, described in 3.2.2.3, as the target to be evaluated.

For obtaining a better analysis, the learning problem should be well-posed. In this regard, accordingly to the definition presented in 5, the cited variables are related as follows:

- Task T: classify certain space regions;
- Experience E: manually label the association between regions and RSSI;
- Measure P: accuracy of correct regions found by the machine learning process;

Regardless of all the next measurements being obtained in RSS, since the receptors are development kit based as described in 6.1.1, for simplifying the data analysis and transmission and also knowing that the real decibel-milliwatts unity will not serve for any application purpose, it will be defined from now on that RSSI is the modulus of RSS.

8.1 Prospection of patterns in a position

An implementation of the Fingerprint method is necessarily preceded of a pattern identification. Analogously with the method's origin name - in which a "fingerprint" can be associated exclusively to a sole "person" - here in this work, something needs to represent the role of the fingerprint, since the role of the "person", or desired searched object, in this case is the localization.

From the study in 7.1.1, an excellent replace for the fingerprint role is the WLAN signal, since it presented a steady variation of its measures and higher values of RSS as discussed in 7.3. Hence, translating the problem into more practical terms, the goal here is to find a RSSI pattern that could represent a physical position.

8.1.1 System architecture

For this test, the setup consisted of five NodeMCUs acting as stationary receptors, reading the RSSI from a fixed position ESP32, the signal emitter. The five NodeMCUs then send the collected information over an User Datagram Protocol (UDP) communication to the Central Unity responsible to process the data (CPU), organize it and store it conveniently in a structured file. Here, a Raspberry Pi did this role for hardware installation simplicity only.

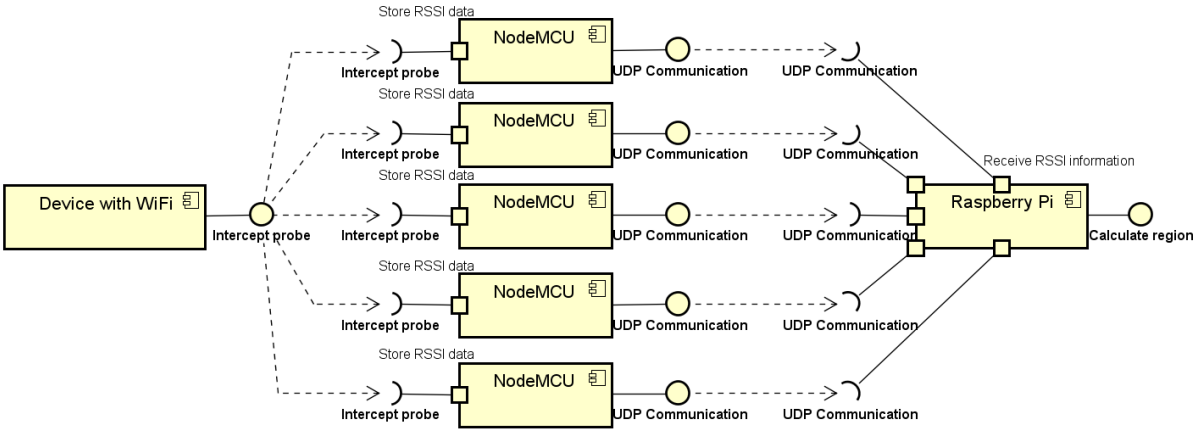


Figure 19: System UML component diagram.

All the five NodeMCUs and the CPU were connected on the same wireless network, while the emitter device remained outside. Furthermore, all the NodeMCUs were set in *promiscuous mode*, making them invisible for other devices inside the network, potentially increasing the security level of the system.

8.1.2 Communication message structure

For the proposed system architecture, there are only two types of communication needed to be specified: the ESP32 - NodeMCUs (emitter - receptors) and the NodeMCUs - Raspberry Pi (receptors - central unity).

8.1.2.1 Probe message structure

The ESP32 was set to send continuously a probe request, without connecting to any specific AP, this way, producing results that would not be intrinsically related to this emitter, generalizing the test since every device able to connect to an AP will start with a probe request at the beginning when establishing a communication [89].

In addition, other interesting aspect of the usage of probe request as the message to be "intercepted" is that it contains the emitter's MAC header, followed by the Frame body, in which contains all necessary information to connect the device into the network, and an Frame Check Sequence (FCS) as shown in the Fig.20.

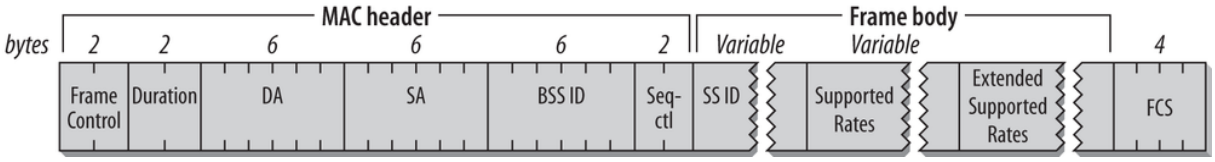


Figure 20: Probe request message structure (extracted from [89]).

The most relevant part of the message is the fact that it carries the emitter's MAC address. With this, once known the device MAC address number carried by the person wanted to be tracked, it is possible to launch the localization procedure without any active action from the user - therefore, leaning towards a more user-friendly final interface.

8.1.2.2 Data collected message structure

After collected n measures of RSSI from the emitter in a fixed scan time t, the NodeMCUs send this stored data towards the central unity. Since the first step consisted of a probe interception procedure, there was not a two way communication established. However, here, for internal data exchange in order to provide better system integrity, reliability and for possible future improvements, a communication protocol needed to be adopted.

The UDP was chosen as the communication protocol for this task for its library being easily accessible, since it is widely implemented in various other common wireless applications, and specially because it is faster if compared to the other equally famous protocol, the Transmission Control Protocol (TCP). This difference is due to the lack of control of the message integrity, where, in the latter, when some data packet is lost, the receptor sends a request for resending the original message [90].

In an attempt to reach into a more agile final solution, a possible measure information loss was preferred because the ability to process information closer to real-time was prioritized over some marginal gain in message integrity of a noisy, or even absent, measure.

For better comprehension of the whole message packet, the message itself was separated into four parts, in which each one represented the bits address of the message as squares on the Fig.21, Fig.22, Fig.23, Fig.24 and Fig.25.

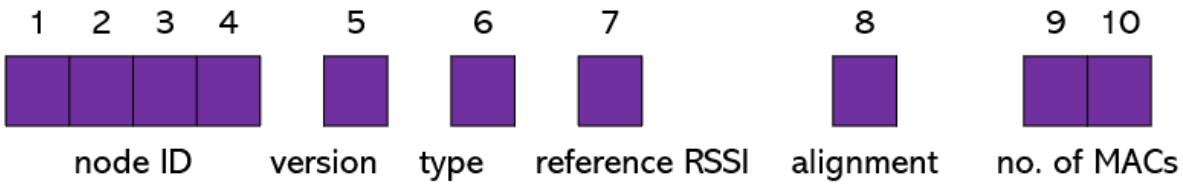


Figure 21: NodeMCU sent message (first part).

The message first part, Fig.21, consisted in:

- Bits 1 to 4 (node ID): identification of the message sender;
- Bits 5 and 6 (version and type): information of the which code and which type of measure procedure currently in use;
- Bit 7 (reference ID): value of RSSI for a specific distance;
- Bit 8 (alignment): alignment operation for better performance;
- Bits 9 and 10 (no. of MACs): indicates the total number of MAC addresses found on the collected measures.

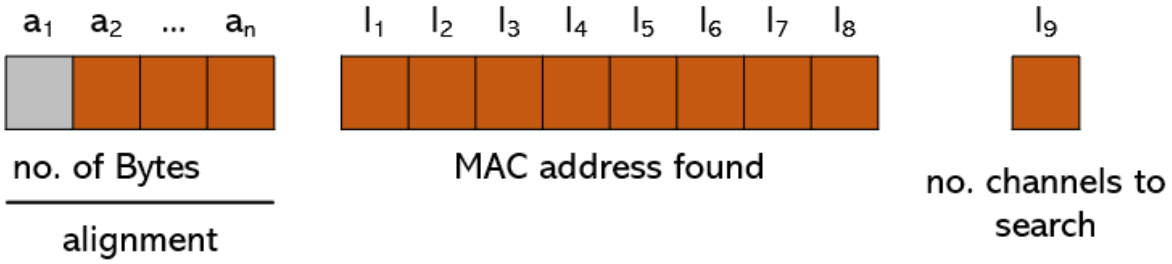


Figure 22: NodeMCU sent message (second part).

From the number of MACs, the first loop was opened. At the start of each loop, the structure presented in Fig.22 repeated each time until all the MACs found were described.

- Bits a_1-a_n: alignment procedure;
- Bits l_1 to l_8: MAC address number found on the measure;
- Bit l_9: number of channels in which the measure was found.

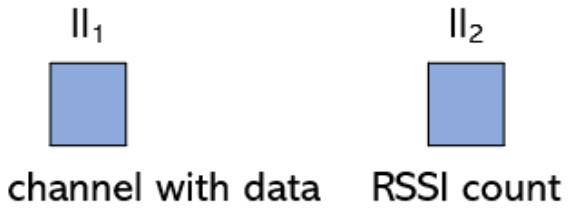


Figure 23: NodeMCU sent message (third part).

The second loop repeats around the number of channels in which a RSSI measure was found. For each iteration, the bits arrangement on Fig.23 was present:

- Bit ll_1: a channel with RSSI measure found;
- Bit ll_2: number of different RSSI measures found.

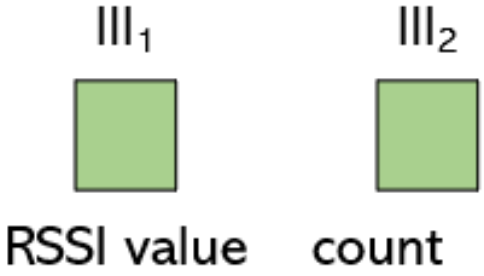


Figure 24: NodeMCU sent message (fourth part).

The last loop consisted in decomposing the RSSI values found, as shown in Fig.24:

- Bit lll_1: RSSI value found;
- Bit lll_2: number of repetitions of this RSSI.

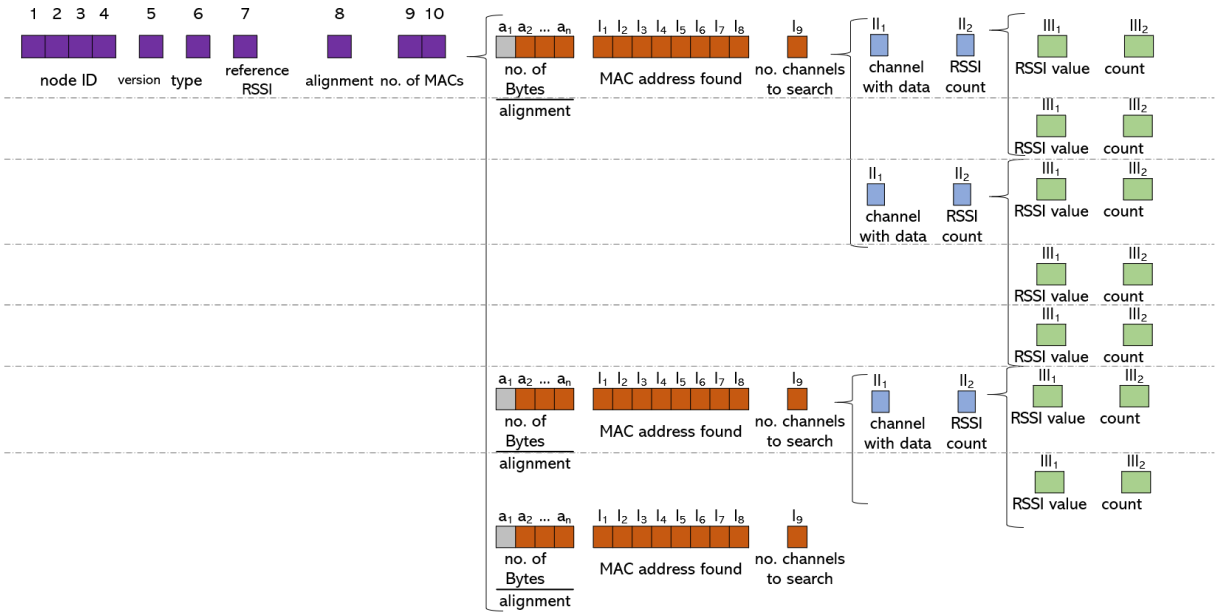


Figure 25: NodeMCU sent message (complete representation).

Finally, Fig.25 shows the complete message sent from each NodeMCU to the CPU. Note that theoretically with this approach, it is possible to infer a new device MAC address once all the already established devices are set, since all the stationary MAC addresses would be already mapped. Therefore, increasing even more the degree of generalization of the solution.

8.1.3 Results

All five NodeMCUs captured the RSSI measurement of the ESP32 emitter placed in a fixed position. For validating the correlation between signal fingerprint and position in space, the general shape of the collected data should remained relatively constant regardless of some time variation among measurements.

By organizing the collected RSSI into small intervals, a frequency histogram could be elaborated. Furthermore, since test could be extremely time consuming, an evaluation of an approximated minimum quantity of samples needed to generate this same general shape was also conducted.

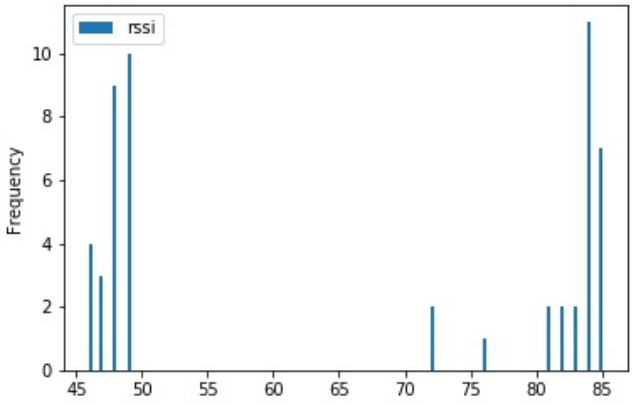


Figure 26: Collected RSSI information for 50 samples.

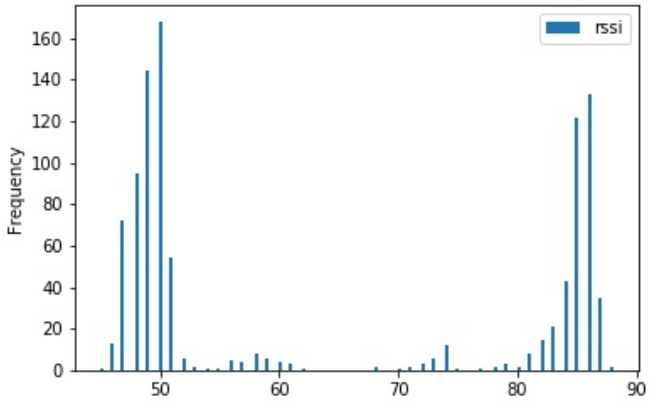


Figure 27: Collected RSSI information for 1000 samples.

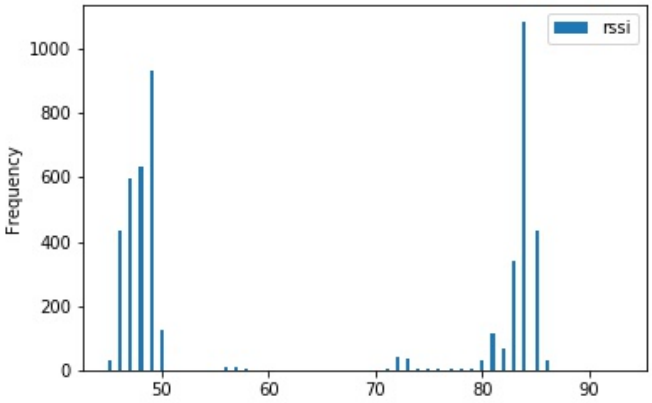


Figure 28: Collected RSSI information for 5000 samples.

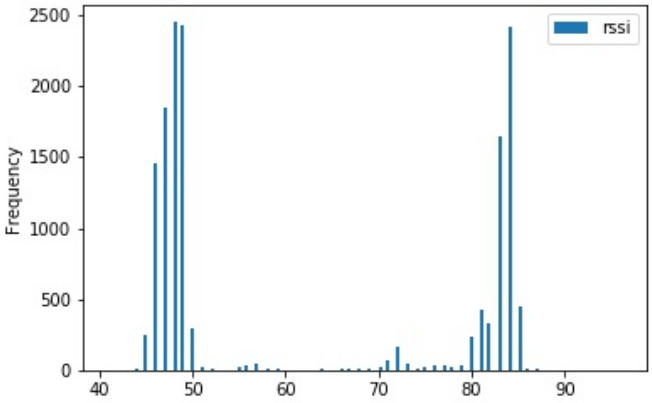


Figure 29: Collected RSSI information for 15000 samples.

The results presented here only reinforces the already established conclusion arrived in 7.1.1, highlighting the difficulty in working with the Propagation model for indoor environments. Still, comparing the different data collected, a general shape of an aggregation of two to three normal distributions could be observed in all five NodeMCUs.

Also, the data pattern from Fig.28 and Fig.29 are almost the same, thus pointing up the marginal data gain after 5000 samples. Empirically, it was found that a number between 500 and 2000 was already enough to achieve a good data distribution. Fewer values compromised the data resolution to the point the bars' size variation was not sufficiently smooth, possibly leading towards rough truncation, negatively affecting the training phase.

8.2 Prospection of patterns in regions

Expanding the scope of the previous analysis, now the turning point is finding a more general relationship, not only with a single position, but with a region on the space. For this step a room was taken as test place, where it was divided into three "easily" determinable regions.

The Fig.30 shows an out of scale representation of the floor plan, whose dimensions are around 470cm x 490cm. The red dots, following its ID number, represents the positions of the NodeMCUs, in which acts the exact same way from the previous experiment in 8.1 catching the RSSI from an emitter by intercepting its continuous probe requests in promiscuous mode.

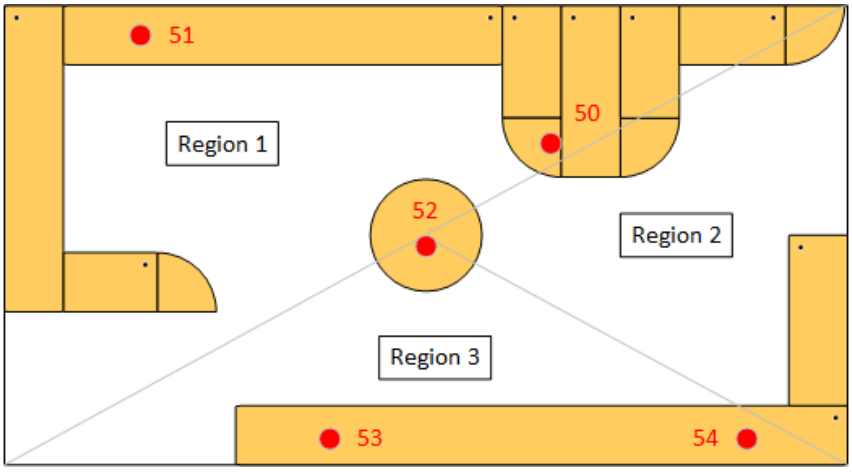


Figure 30: Test area floor plan representation.

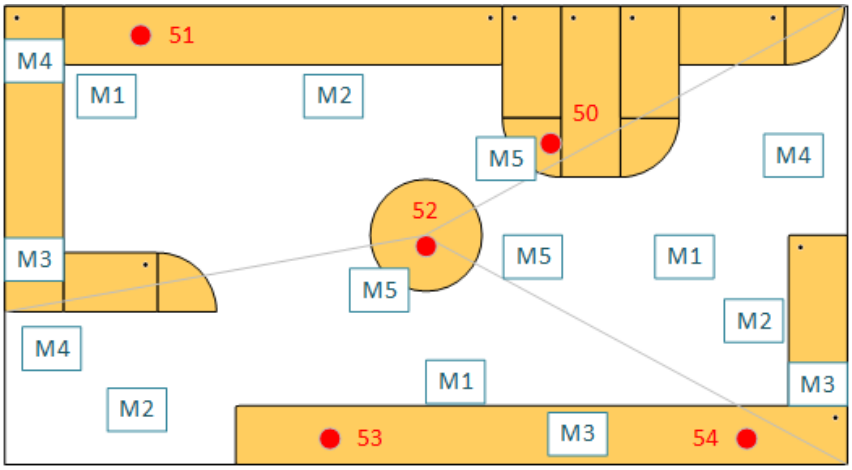


Figure 31: Measurement positions on the first test area.

For each region, five measures of 500 RSSI values of each node were taken. An external battery was attached to the ESP32, which stayed unmovable during the extraction of the data. The actual physical position are shown in the Fig.31, in which it can be seen that areas near the frontier were taken into account for better characterize the regions.

8.2.1 SVM approach

The support vector machine model trained with a RBF kernel, as described on the chapter 3, by searching hyper-parameters with a grid contain thirteen C values in a range of 10^{-2} to 10^{10} and thirteen Gamma values in range of 10^{-9} to 10^{3} distributed evenly in a log space, resunting on a heatmap represented by the figure 32

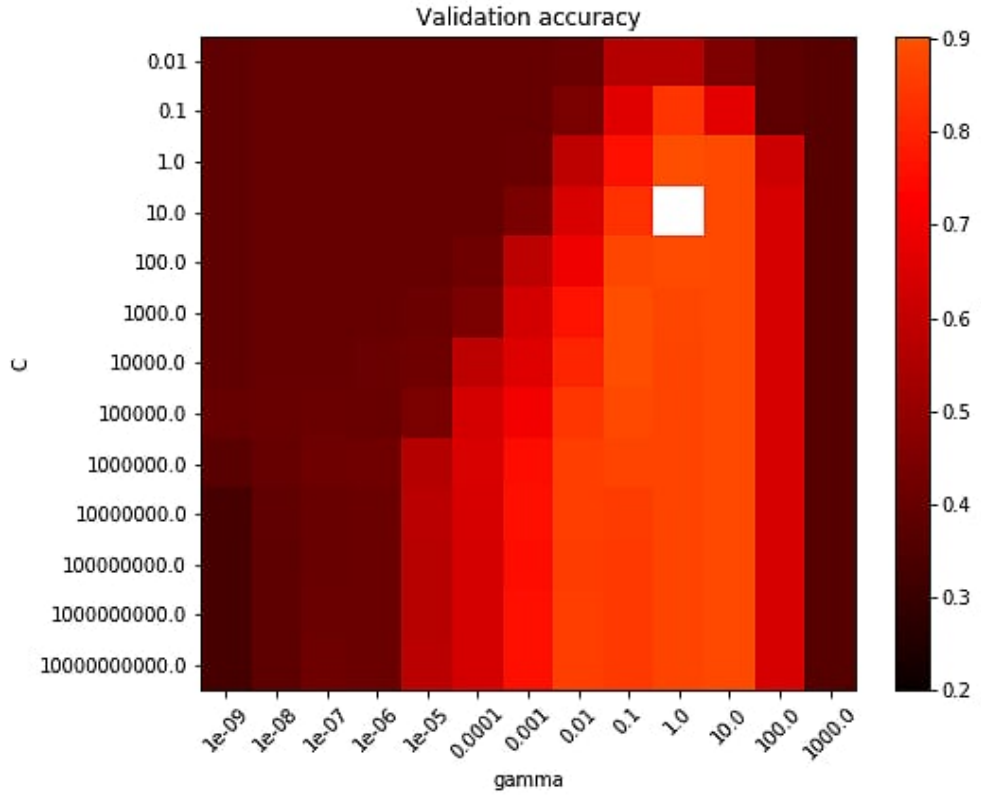


Figure 32: SVM hyper-parameters grid

To evaluate each network architecture a cross-validation method was used to divide the train data in ten folds which nine were used to train a model and one to validate it. Doing so, was possible to train ten different models using the same dataset, which is useful, since it is possible to take a mean accuracy to mitigate the bias caused by some model outlier.

The accuracy from the best model,as described by the figure 32 was arround 91%. To evaluate a possible model overfitig, the model was used to predict a second dataset ,made by collection data from other positions than the ones used to create the training datase, giving a acuraccy of 71%.

8.2.2 ANN approach

Several ANN architectures were tested with the collected RSSI data measurements. They essentially differentiate themselves by changing some hyper-parameters such as the type and quantity of neuron layers, optimization method, layer dropout, activation function on each layer to find which architecture would provide the best accuracy and spend less time to train.

8.2.2.1 Common network configuration

Some elements were common to to all architectures:

- The input layer consisted by the five RSSI data of each NodeMCU;
- Each proposed hidden layer were composed by fifteen neurons;
- For the output layer, a softmax function acted as activation on each neuron;
- The number of neurons on the output layer was defined by the quantity of regions, in this test, the number was three;
- The cross-entropy was used as loss function;
- The batch size was defined as fifty;
- A stop callback was set if on the last 2000 steps the accuracy did not improve.

By using the same Cross-validation method in the SVM approach in 8.2.1, a general comparison could be established between the SVM model and all other ANN models.

For this analysis the SGD, RMSProp and Adam optimizers were used with some variation on the activation functions used. The compared parameters were Internal accuracy – resulted only by the training values – and Test accuracy – from another battery of measurements which were not used on the ANN training.

8.2.2.2 SGD optimizer

Configuration type	**1**	**2**	**3**	**4**	**5**	**6**
no. ReLU layers	0	1	2	2	2	1
no. Sigmoidal layers	1	1	1	2	1	2
	Sigmoidal in first				ReLU in first	

Table 4: Configuration types for SGD optimizer.

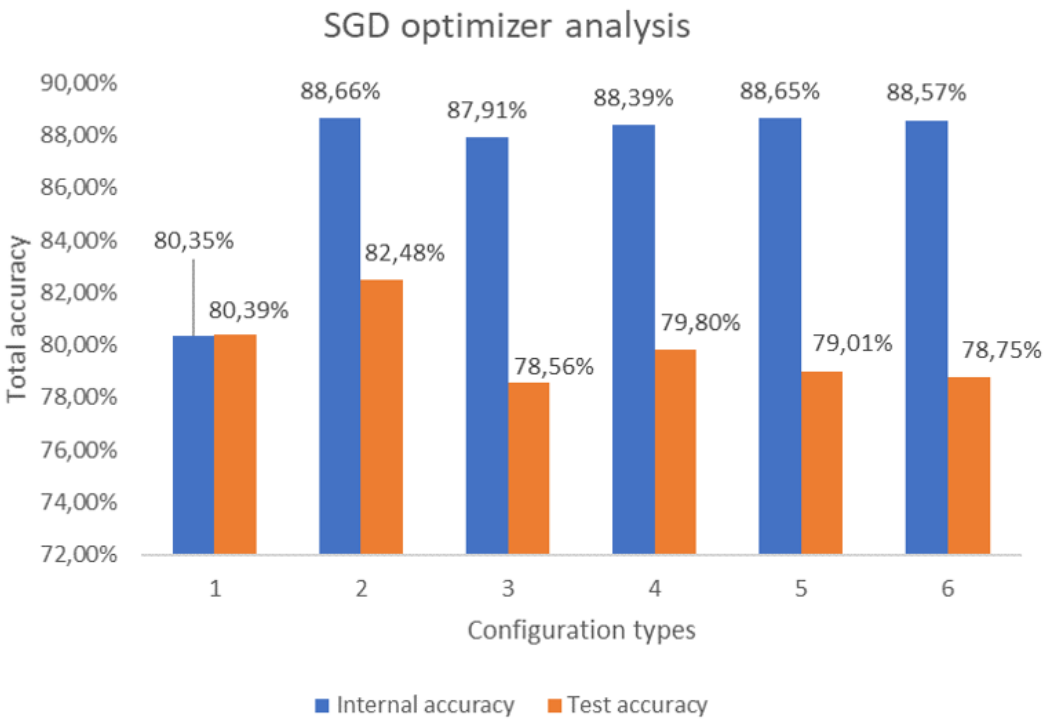


Figure 33: Accuracy results with SGD optimizer.

8.2.2.3 Adam optimizer

Configuration type	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
no. ReLU layers	0	1	1	2	2	2	3	1	2	2
no. Sigmoidal layers	1	0	1	0	1	2	1	1	1	2
	Sigmoidal in first							ReLU in first		

Table 5: Configuration types for Adam optimizer.

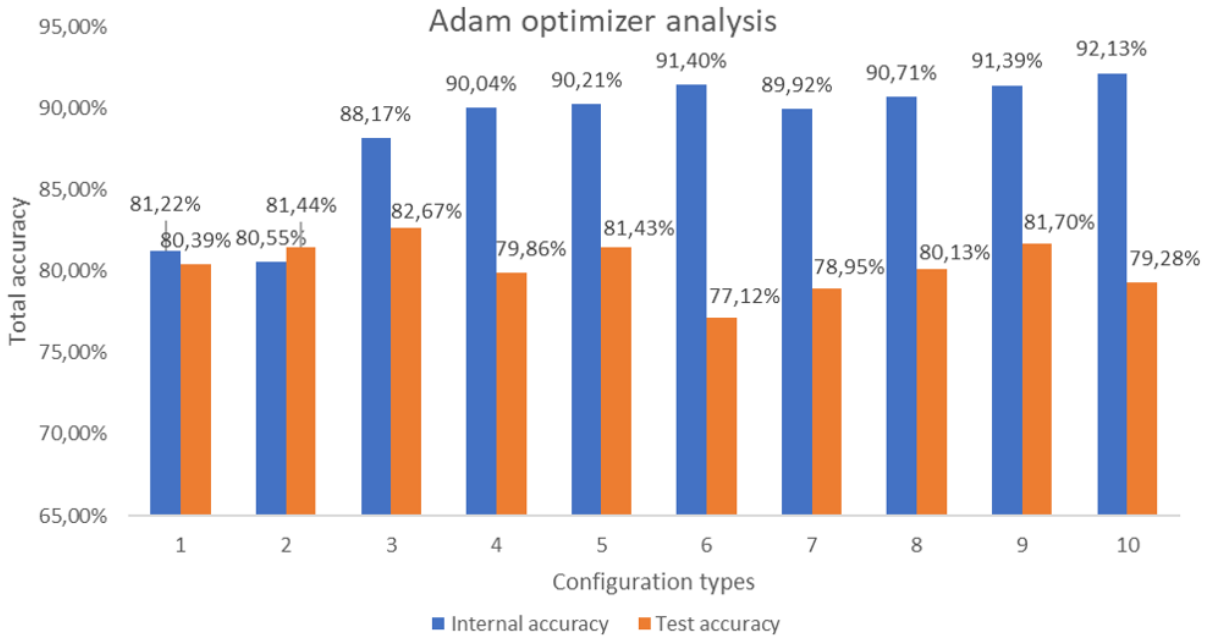


Figure 34: Accuracy results with Adam optimizer.

8.2.2.4 RMSProp optimizer

Varying the order of the ReLU and Sigmoidal layers did not impact significantly as shown in Fig.33 and Fig.34, therefore, for the next test this information was suppressed from the table 6.

Configuration type	**1**	**2**	**3**	**4**	**5**
no. ReLU layers	1	1	2	2	2
no. Sigmoidal layers	0	1	0	1	2

Table 6: Configuration types for RMSProp optimizer.

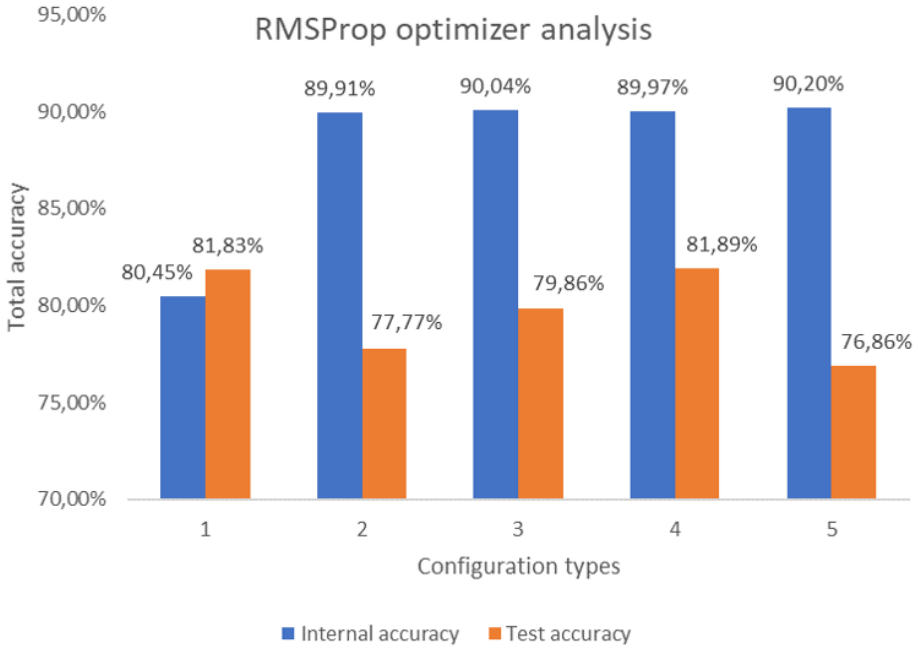


Figure 35: Accuracy results with RMSProp optimizer.

8.2.2.5 Optimizers comparison

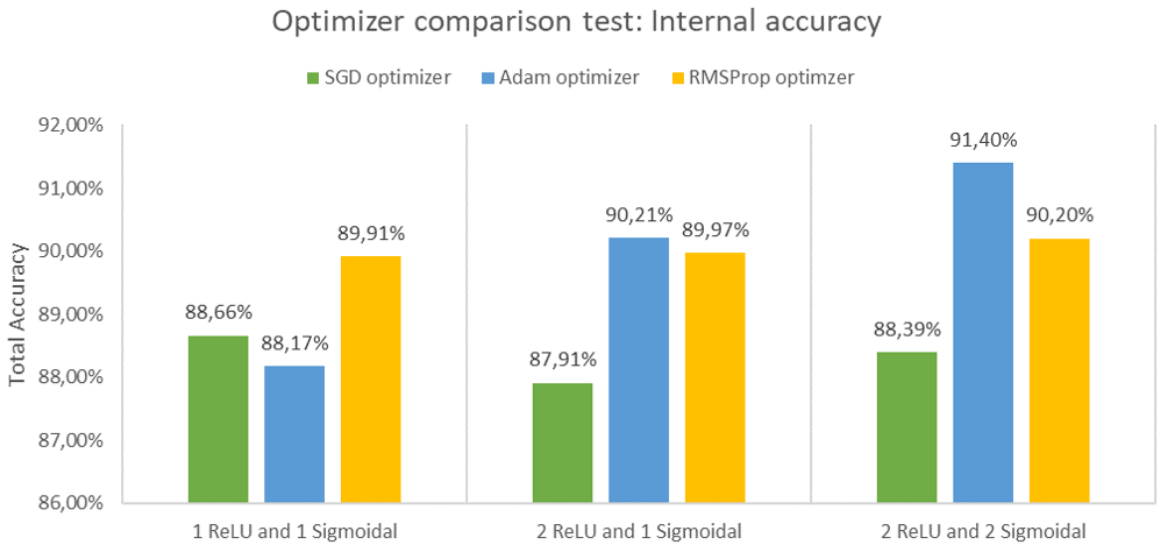


Figure 36: Optimizers' internal accuracies analysis.

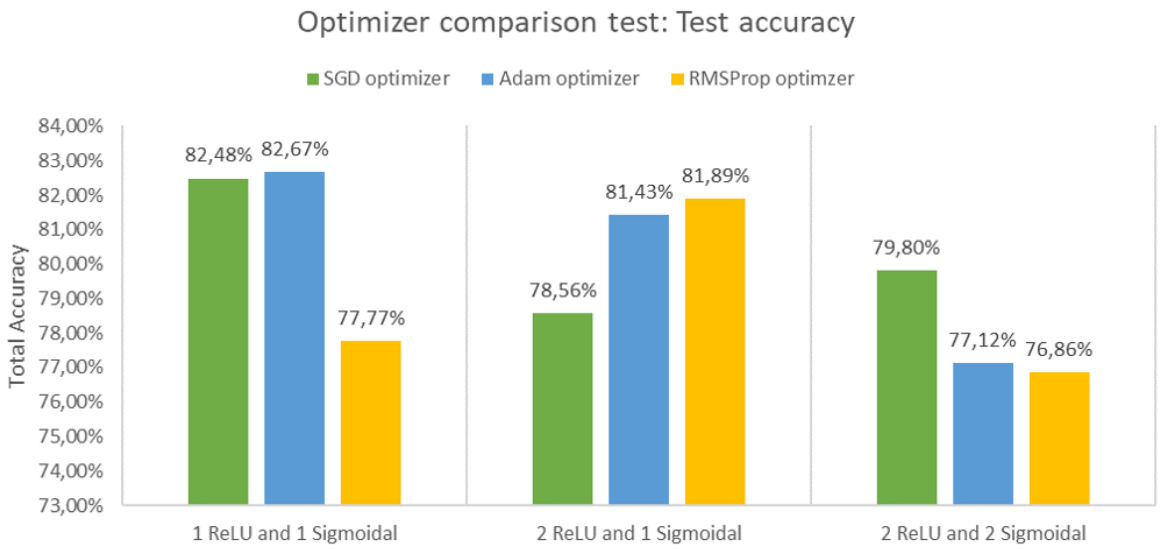


Figure 37: Optimizers' test accuracies analysis.

8.2.2.6 Dropout analysis

As mentioned in 5.2.7, some configurations were taken to be tested if a dropout usage could better avoid overfitting. A variation up to 50% of dropout was suggested, varying 10% in each step. The resulted accuracy can be found in Fig.38 to Fig.41.

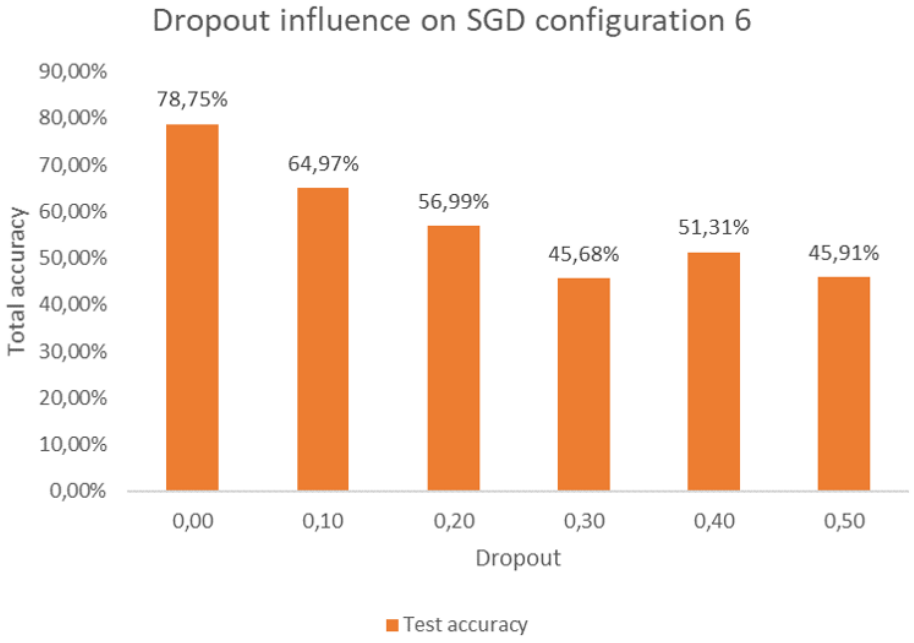


Figure 38: Dropout influence for configuration 6 with SGD optimizer.

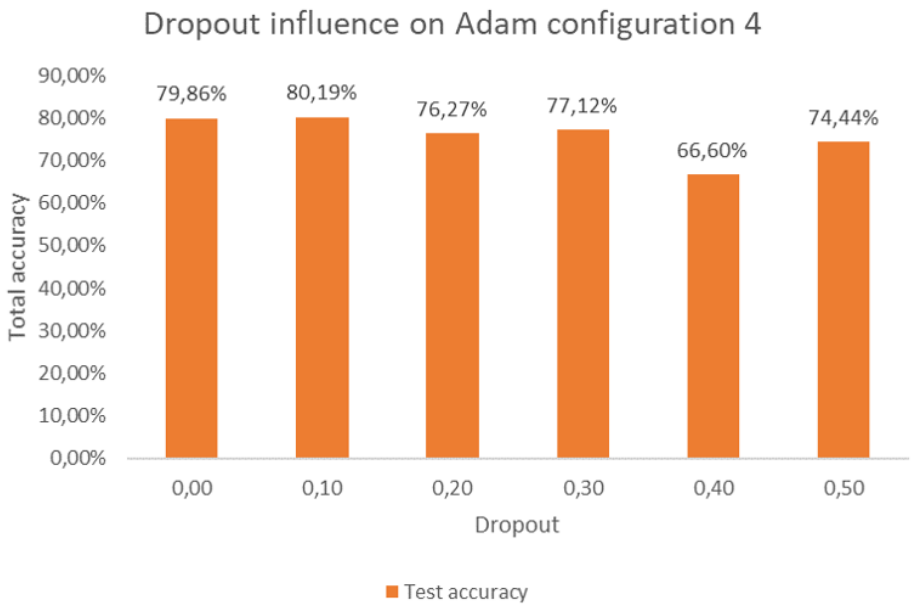


Figure 39: Dropout influence for configuration 4 with Adam optimizer.

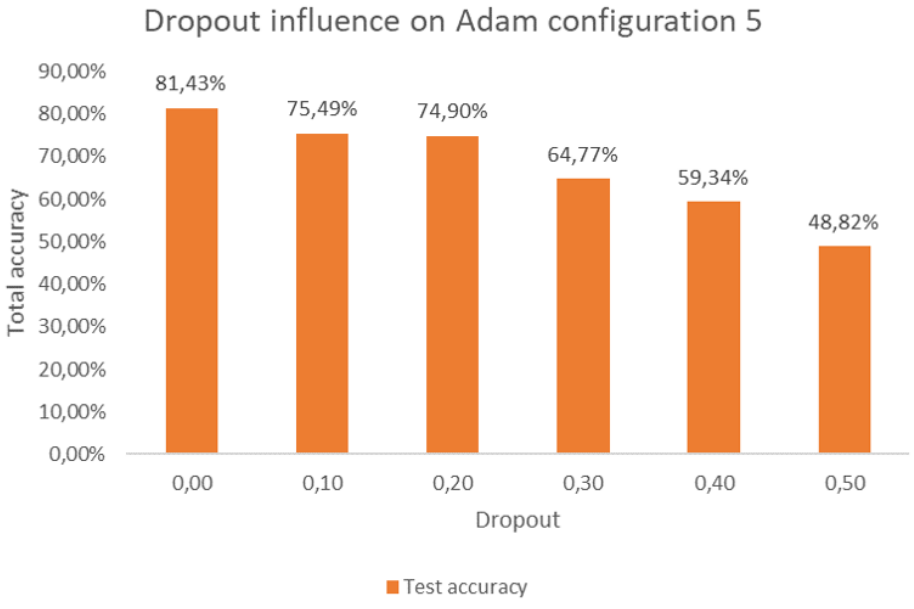


Figure 40: Dropout influence for configuration 5 with Adam optimizer.

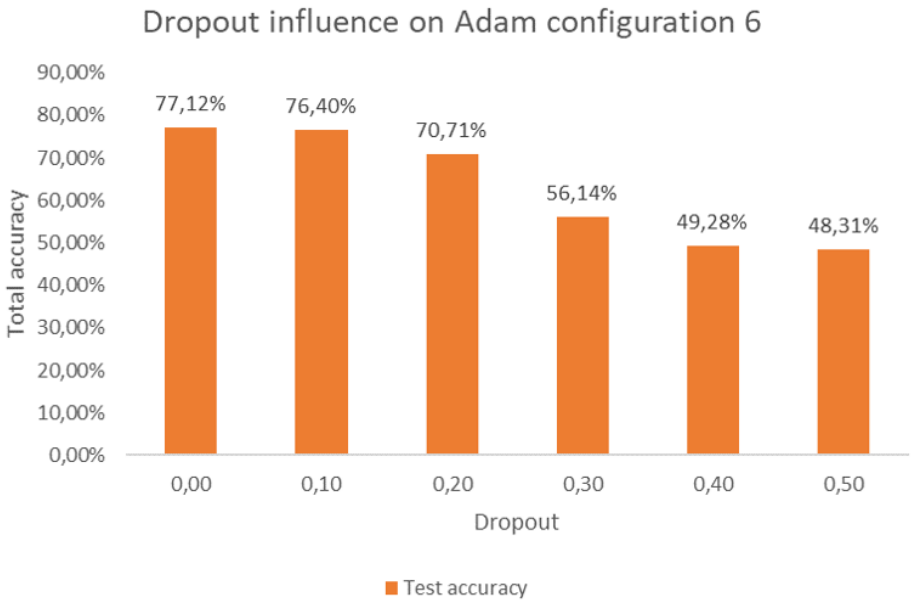


Figure 41: Dropout influence for configuration 6 with Adam optimizer.

8.2.2.7 Accuracy analysis

From Fig.33, Fig.34 and Fig.35, it can be noted that an increase in the internal accuracy did not necessarily implied in a higher test accuracy. Some models with a mediocre internal accuracy presented better test results than the ones with greater internal accuracy. This can be explained as the models being Overfitted.

Comparing the type of optimizer used, Fig.36 and Fig.37 showed that all of them have similar final accuracies, varying very little as can be directly seen in the scale of the graphics axis. Thus, only one was selected of the implementation phase: the Adam optimizer for being faster in the training process.

The dropout reduced in general the test accuracy on all four tested ANN configurations except for the type number 4 with Adam optimizer (Fig.39). Yet, none substantial improvement could be noted.

PART V

IMPLEMENTATION

9 PROTOTYPE'S FINAL TUNING

For the last phase of the project, the same study applied in 8.2 was now tried on the real test area, the Full mission simulator projector room of TPN-USP.

9.1 Physical environment configuration setup

9.1.1 Regions subdivision

Firstly, as described on 2.2.1, due to the measures of the simulator room not being too big, lesser than 10 square meters with around 8 meters from one side to another, the approach of dividing the room itself into minor regions did not compromise the proposed objective's accuracy. The Fig.42 shows an out-of-scale floor plan of the already presented Full mission simulator room in 2.1 and Fig.2.

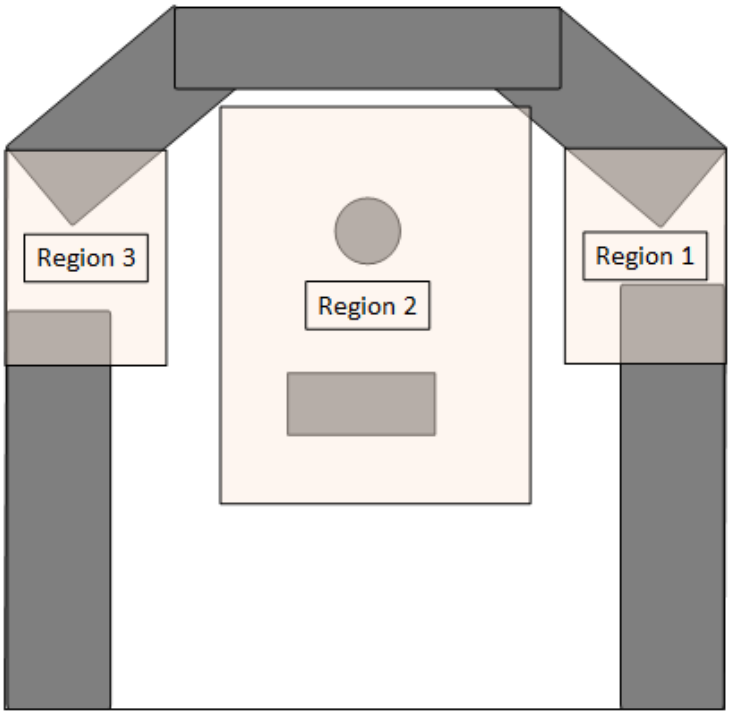


Figure 42: Regions subdivision on the test room.

It were considered three main regions taking into account the most common and critical places where a maritime pilot stands on the bridge during a maneuvering procedure. Both regions 1 and 3 represent areas where in real life are far away from the region 2, therefore were picked as critical and of most interest. Furthermore, additional measurements over RSSI values that are not contained in the previous regions were considered for training purposes, therefore, implicitly describing a "fourth region" that are physically from the extreme bottom of region 2 until further bottom on the simulator room.

9.1.2 Nodes' placement

In the study in 8.2, one aspect that was not taken much care was the receptors' placement on the room. Although theoretically it is not relevant since the signals could also establish just another pattern description, this same new pattern could also infer some other signal pattern shape that does not present a good compatibility to the adopted ANN formulation. Hence, with the intent of smoothing the pattern received on the NodeMCUs, a better configuration was thought as shown in Fig.43, in which the location of the receptors are depicted by red dots.

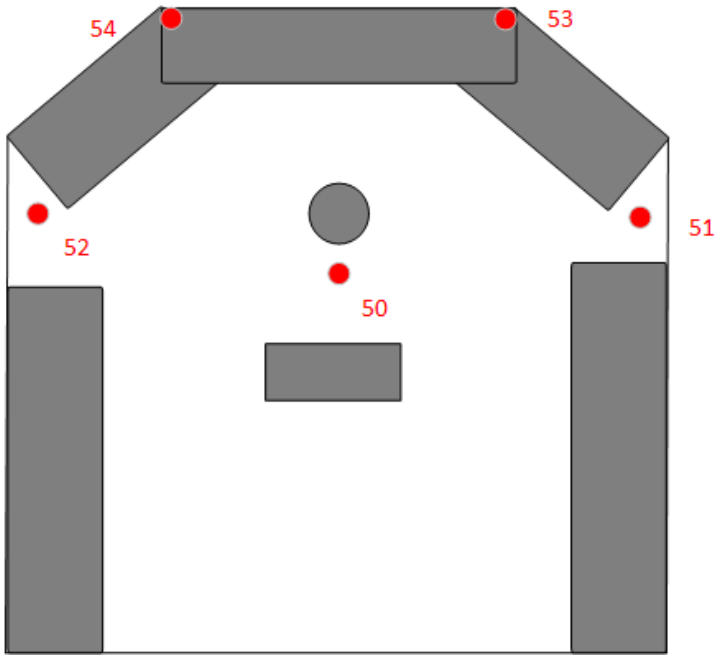


Figure 43: NodeMCUs' placement on the test room.

All the signal receptors were placed on the ceiling for a better general signal reading. Also, if compared with the Fig.42, it can be seen that at least one NodeMCU is placed on the approximate center of the interested region, while the other two are relatively close

and between regions with a good sight of a emitter inside all three regions.

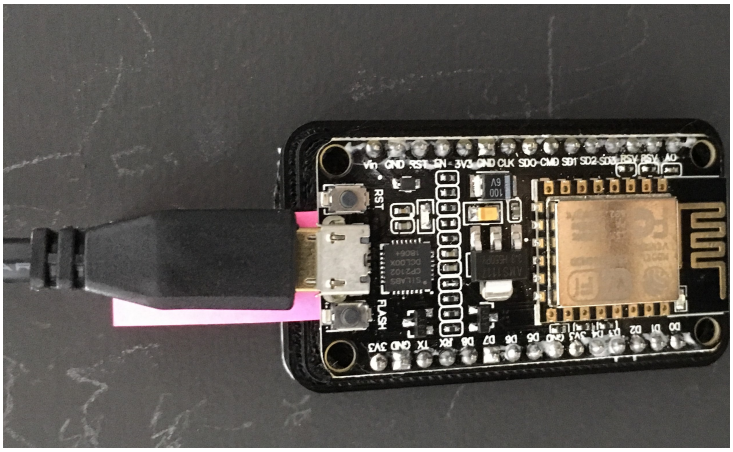


Figure 44: NodeMCU fixed on the wall with a printed support base.

9.1.3 Training phase

Up until this point, the training phase consisted only in gathering the measurements of a certain static position that was then later labeled as inside a region. directing towards the creation of a smoother model, a moving data capturing procedure was tested and compared with the previous approach. Although less practical for a recurrent real application, the idea is to only be done once, thus, zero impacting the final user usability view point.

For the static evaluation, it were taken around 2000 measurements for each NodeMCU in a fixed position inside a certain region. Five static positions were chosen for each of the three regions.

Now for the moving case, the measurement procedure was thought with the time duration in mind, with a 50-minute collecting RSSI process for each region. Furthermore, aiming a final real-time implementation, a new requirement of retrieving all five RSSI from a same time sample was inserted for this latter collect method.

9.2 ANN configuration prospection

From the data collected, ANNs different configurations were created and then evaluated in order to find a more suitable one for this application context. The analysis procedure adopted was the same as mentioned in 8.2.2.1. Hence, all the accuracies results for the next tests are the Test accuries, from inputs not used for training (test accuracies).

9.2.1 Static versus moving collected data

A standard network configuration was adopted for the next three tests as shown in the table 7.

ANN hidden layers	
no. ELU layers	2
no. Sigmoidal layers	1

Table 7: Standard adopted hidden layer configuration for tests in 9.2.1, 9.2.2 and 9.2.3.

From preliminary verification during the models' creation it could be already verify that for the static collected data the test accuracy was lesser than the 33 % (from a random guess in one of three regions), varying just a little around 26 %. Thus, further tests with the static data were aborted. For the moving collected data, the next tests, which were based on, proved to have undoubtedly better accuracies.

9.2.2 Interested 3-region versus 3-plus-1-region model

Using the same ANN configuration presented in table 7, a comparison test was conducted in order to verify the proposition suggested in 9.1.1 of characterizing a fourth region could improve the general accuracy rate of the system. The idea behind this was that the noisy measurements could be related to this fourth region (or "not in any of the three regions"), therefore conditioning the ANN to better recognize if some junky measurement can be filtered.

By adding this extra region, in practical terms a new output neuron was added to the network, resulting in four neurons in total. This new neuron had the same exact parameters as the other ones in the output layer as mentioned in 8.2.2.1.

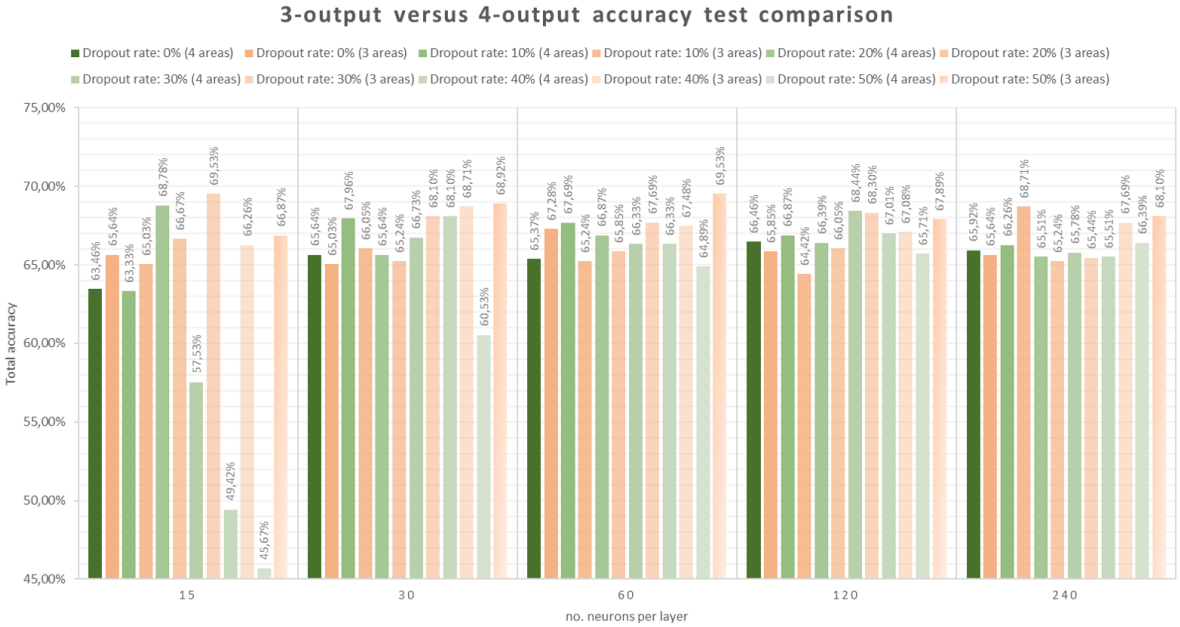


Figure 45: Comparison test between three and four output neurons.

As Fig.45 showed, in average the test accuracy with four regions (green bars) were undoubtedly worse than the ones modeled with only the three original regions (beige bars). Only in some cases the four regions were better, still, only marginally. Thus, the fourth region model was abandoned.

9.2.3 Adam versus Nadam optimizer

As shown in 8.2.2.7, SGD, RMSProp and Adam optimizers presented similar results but with different training time required, reason why the latter was chosen. Dozat, in [91], suggests that the addition of Nesterov momentum into Adam could improves the convergence speed and even the quality of learned models.

This next test in Fig.46 is a comparison between these two optimizers – Adam (red colored bars) and Nadam (blue ones) – with the intend to verify the mentioned general assumptions in this particular problem presented in this work. The network configuration was the same of the previous test as shown in table 7.

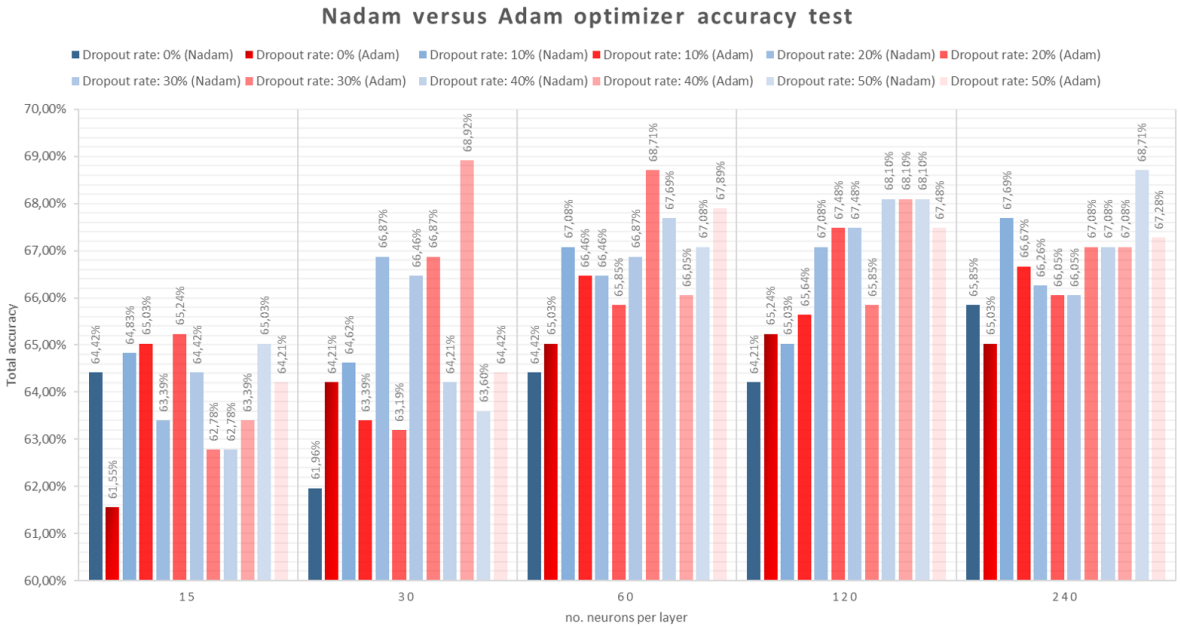


Figure 46: Comparison test between Nadam and Adam optimizers.

Besides some discrepancies with Adam and Nadam such as seen for 15 neurons per layer with 0% of dropout and some with 30 neurons per layer, the final accuracy difference did not surpass 4.8% and in average the difference was indeed inferior than 3%. Although an improvement in quality could not be noticed, Nadam took 3 minutes lesser than Adam to finish its training. Thus, for the same time reasons, Nadam was selected over Adam from now on.

9.2.4 Influence of ELU neurons

An uniform test with only layers composed of ELU neurons was proposed in an attempt to evaluate if its usage could be advantageous for modeling this problem. The table 8 shows the studied configurations of ELU layers.

Configuration type	**1**	**2**	**3**
no. ELU layers	4	6	11

Table 8: Configuration types for evaluating ELU neurons.

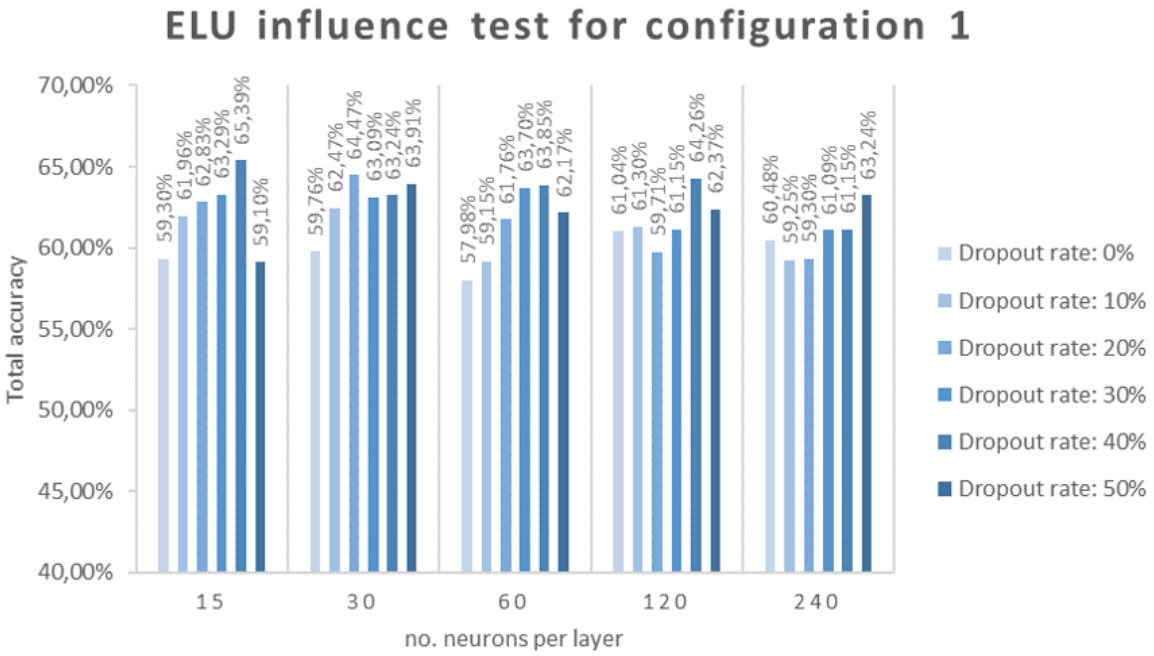


Figure 47: ELU configuration 1 accuracy test analysis.

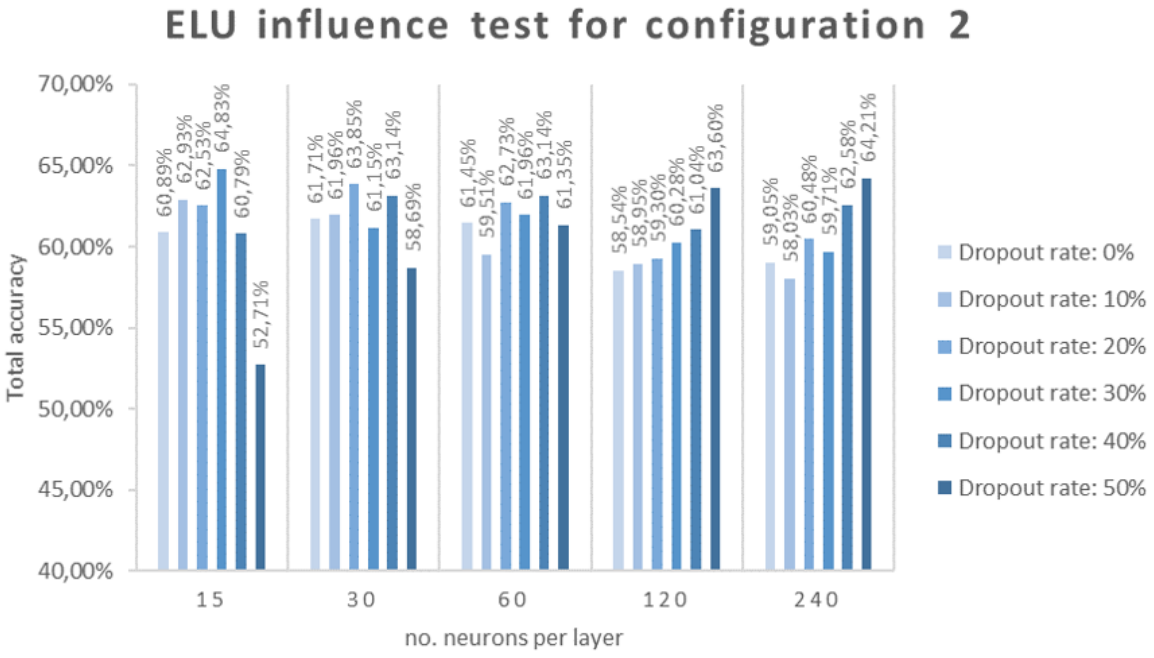


Figure 48: ELU configuration 2 accuracy test analysis.

Comparing the results obtained in Fig.47 and Fig.48, it can be inferred that an increase in the number of ELU layers tends towards a general better test accuracy until 120 neurons per layer.

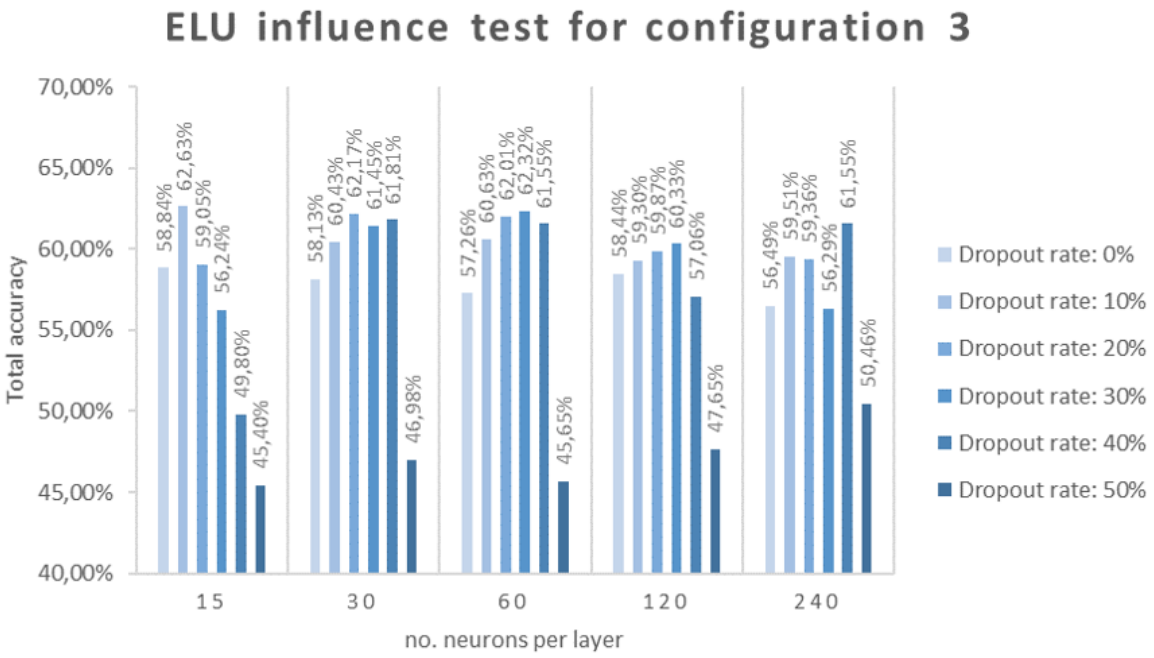


Figure 49: ELU configuration 3 accuracy test analysis.

Nonetheless, a major increase in the number of ELU layers such as in the configuration 3 can hurt the model as shown in the Fig.49. Therefore, this model seems to have a good representation with some layers of ELU neurons.

9.2.5 Influence of Sigmoidal neurons

Configuration type	**1**	**2**	**3**	**4**
no. ELU layers	0	0	2	2
no. Sigmoidal layers	2	2	0	2
no. ReLU layers	0	2	0	0

Table 9: Configuration types for evaluating Sigmoidal neurons' influence.

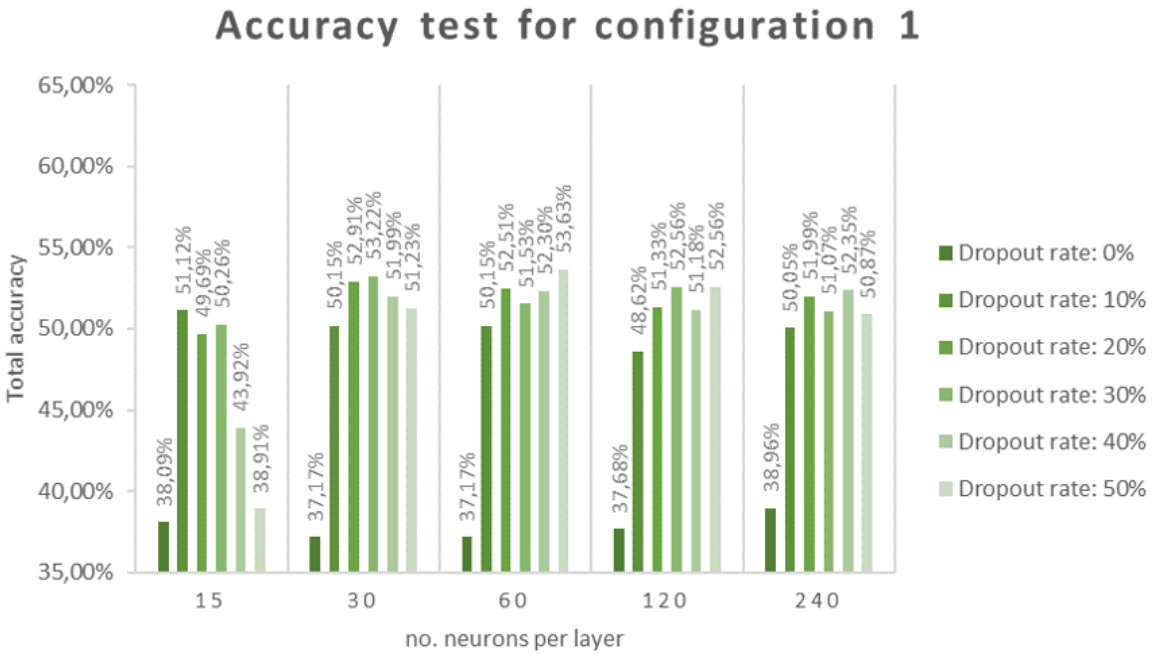


Figure 50: Sigmoidal influence test configuration 1 analysis.

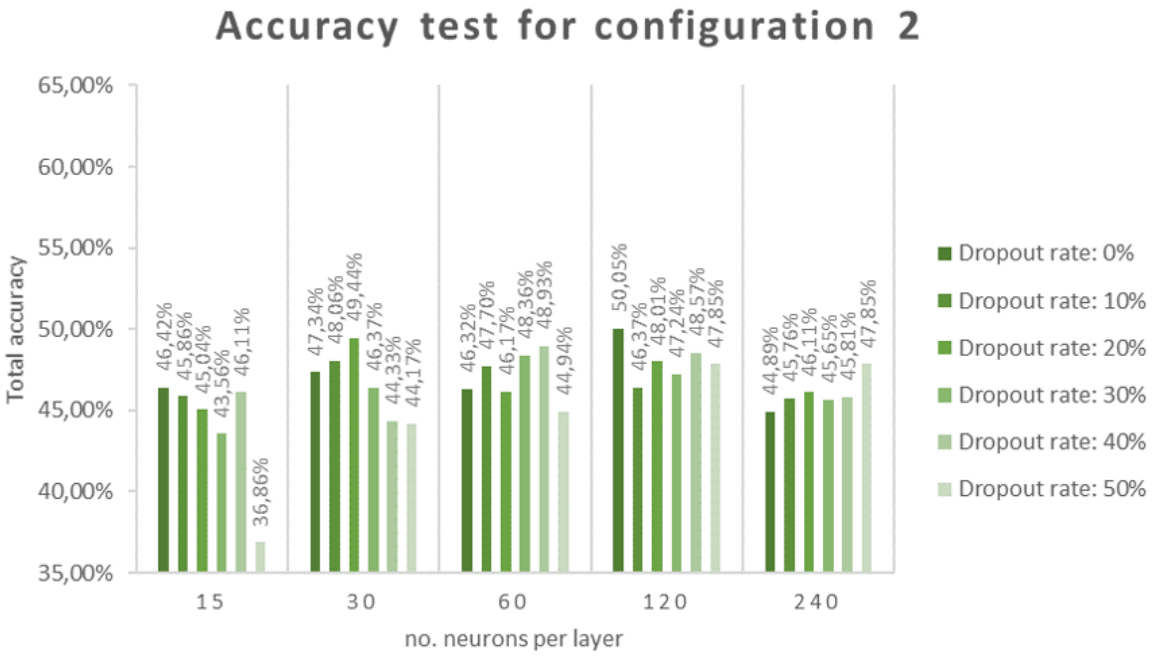


Figure 51: Sigmoidal influence test configuration 2 analysis.

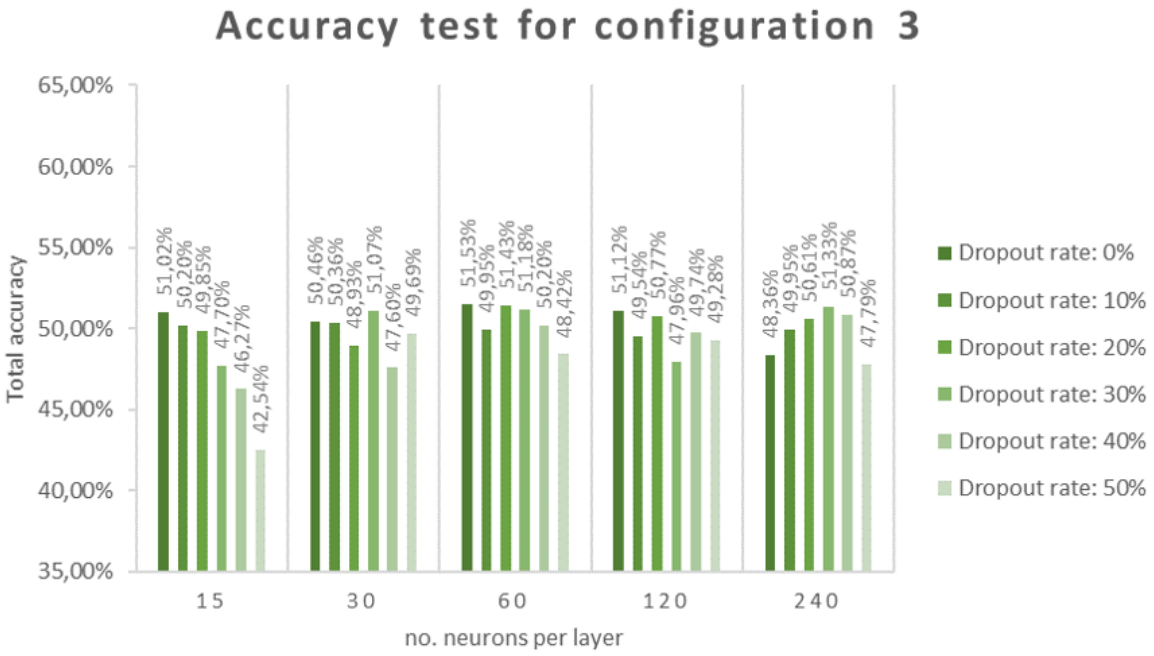


Figure 52: Sigmoidal influence test configuration 3 analysis.

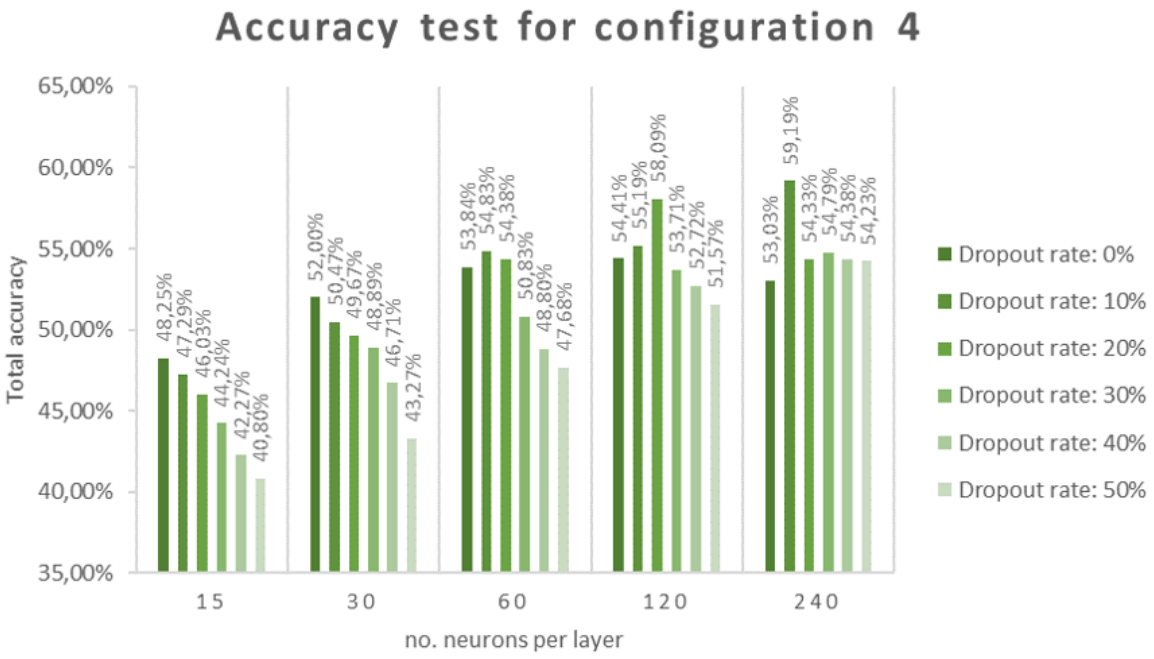


Figure 53: Sigmoidal influence test configuration 4 analysis.

A Sigmoidal neuron layer by itself did not bring any beneficial influence on the test accuracy as demonstrated in Fig.50, however, when attached with a ELU neuron layer, the results drastically improved as seen in the Fig.53. Furthermore, the usage of ReLU

layers did not considerably improve the final accuracy if compared with the usage of ELU, thus, being discarded for the final ANN configuration.

9.2.6 Number of inputs evaluation

This work decided to implement five NodeMCUs instead of three, the minimum necessarily mathematically to determine three different regions. These additional inputs were inferred as a system redundancy aiming to achieve a better reliability, yet, the extra noise was also implicitly considered. The next tests focused in verifying the influence of more inputs for the model's accuracy.

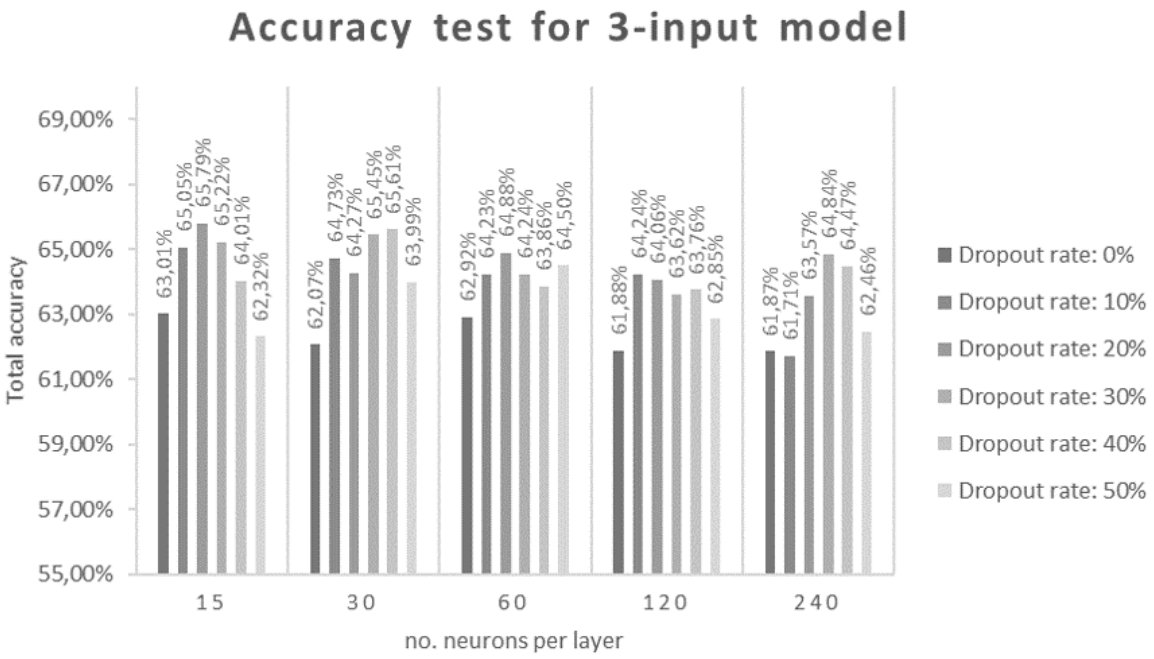


Figure 54: Accuracy evaluation considering only three NodeMCUs.

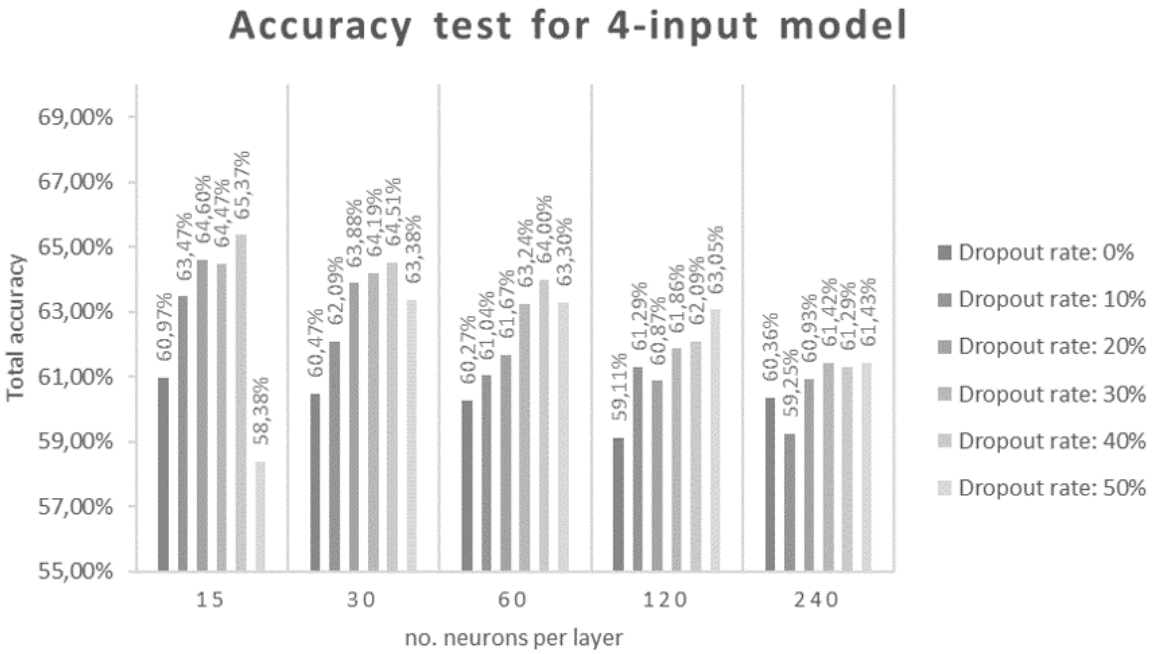


Figure 55: Accuracy evaluation considering only four NodeMCUs.

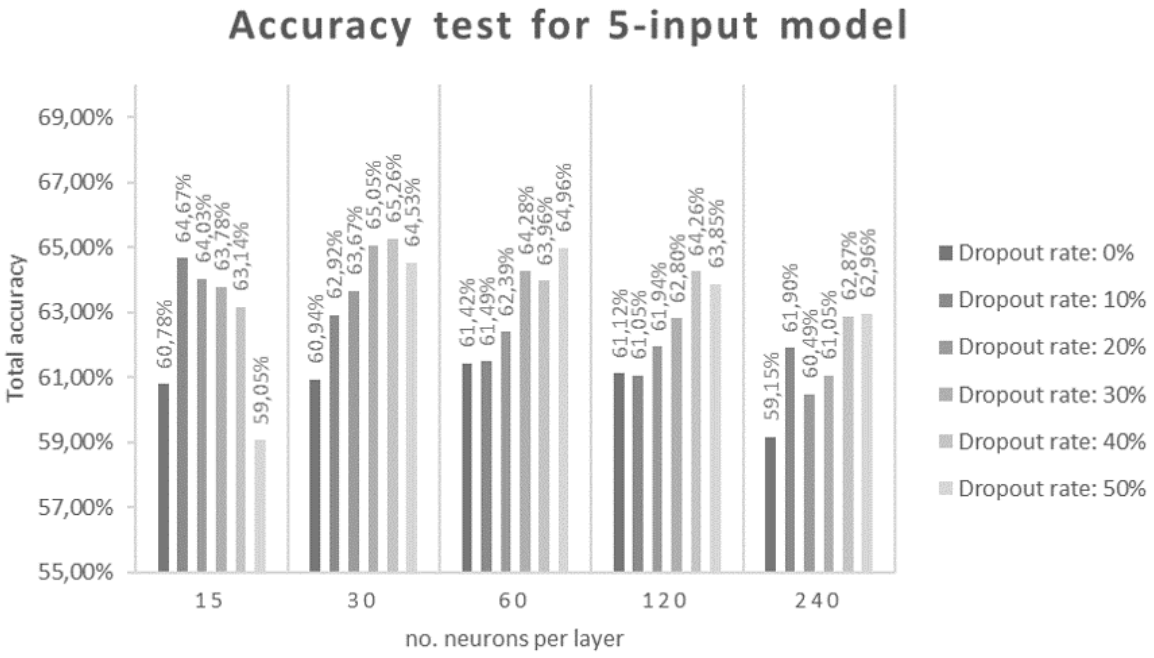


Figure 56: Accuracy evaluation considering all five NodeMCUs.

All results showed some little accuracy variation, a maximum of 4 % in average for all cases tested in Fig.54, Fig.55 and Fig.56. Hence, adding more purely inputs in the ANN without any prior manipulations did not improve nor reduced the general accuracy performance of the model.

9.2.7 ANN final configuration proposal

Considering all the previous tests, empirically, the best ANN configuration is presented in table 10:

ANN configuration		Resulted from
Type of collected data	Moving collected	subsection 9.2.1
no. of inputs	5 RSSI reads	subsection 9.2.6
no. of outputs	3 regions	subsection 9.2.2
Optimizer used	Nadam	subsection 9.2.3
Learning rate	0.02	Nadam default value
Batch size	30	Nadam default value
no. ELU layers	2	subsection 9.2.4
no. Sigmoidal layers	1	subsection 9.2.5
no. ReLU layers	0	subsection 9.2.5
no. of neurons per layer	120	figure 57
Dropout rate	40%	figure 57

Table 10: Final ANN configuration adopted.

With the ANN's parameters cited in table 10, more models were trained since even with the same parameters a ANN can stabilize its weights differently. Hence, the final best model presented test accuracies as shown in Fig.57.

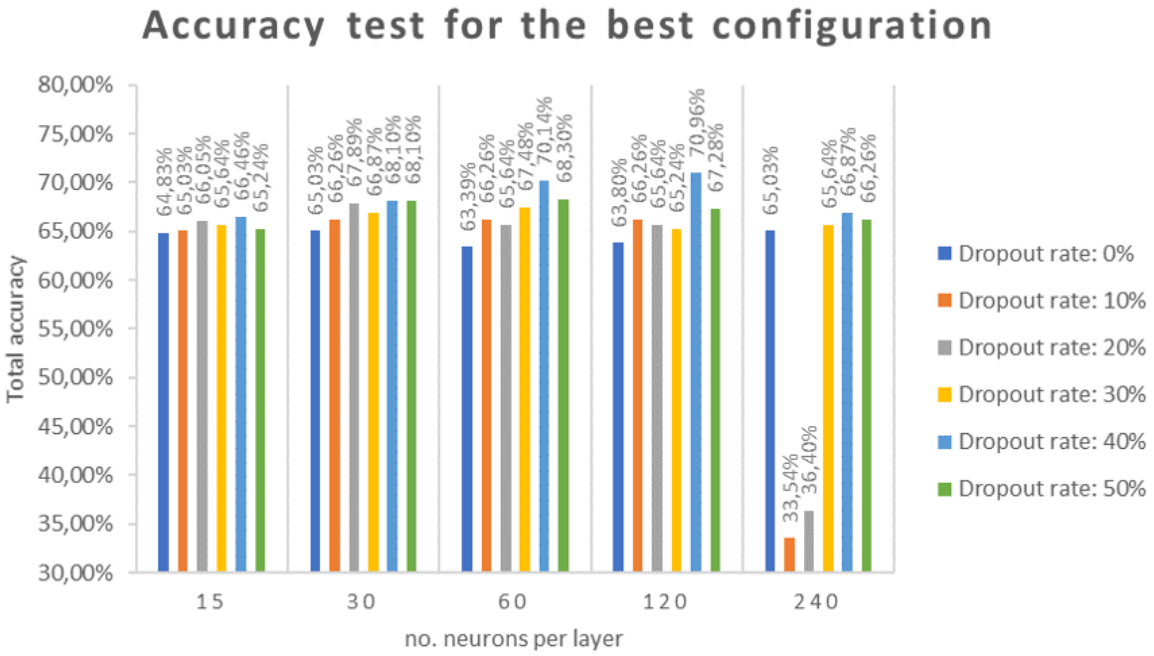


Figure 57: Accuracy evaluation with the best configuration found empirically.

9.3 Real time estimator algorithm

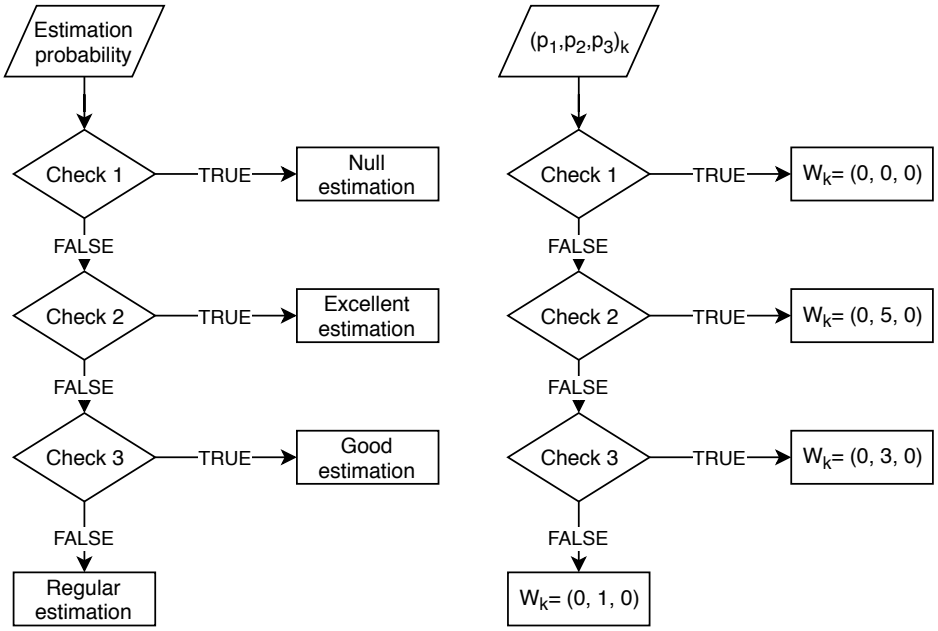
As introduced in 9.1.3, for a real time usage, it is mandatory for the system to obtain all five RSSI measurements from the same time window. With these five inputs, the ANN estimates the probability of the detecting signal being in each of the three regions.

Six multithreads in total were created in the CPU program. Five of them were used to listen to the stored RSSI measurements received from each of the NodeMCUs; while the sixth one were incurred to runs the region estimation procedure. A time window t were fixed in such a way that while the CPU is listening incoming data from a discrete time step k, concurrently it generates the probability of the tracking person being in each of the desired three regions in $k - 1$.

Innumerous complex factors contribute to generate a variation in the number of received RSSI quintuples, still, empirically, $t = 10\,seconds$ ensured that at least three RSSI quintuples were found. The final region verdict is given after an evaluation of all the calculated estimations during t, theoretically, achieving better accuracy than the previous tests in 9.2.

9.3.1 Decision tree weighting

A decision tree logic was implemented to classify each estimation given by the ANN model accordingly with the quality of the generated results. The fluxogram shown in Fig.58 describes a general view of the procedure.



(a) Qualitative grading. (b) Quantitative grading.

Figure 58: Fluxogram of the decision tree for weighting probabilities estimations.

These presented "checks" consisted in:

- Check 1: the estimation is worthless if gives an equally distributed probability of being in all three regions;
- Check 2: a high discrepant probability is considered of major relevance;
- Check 3: if there is a considerable gap between the highest and middle probabilities, then the estimation relatively good, otherwise, of minor relevance.

More specifically, *Check 1* verifies if all three given probabilities (p_1, p_2, p_3) are higher than 30%; *Check 2* if there is a p_i higher than 75%; and *Check 3* if the difference between p_{higher} and p_{middle} is higher than 10%.

The final answer to the region is given by the vector W in which is recursively summed by the decision tree output. Some weights were defined to each estimation 1, 3 and 5, respectively representing excellent, good and regular estimation quality. A neglected estimation simply skips the vector W sum, virtually, summing $(0, 0, 0)$.

For instance, the table 11 shows that at the end of all evaluations of probability, the decision vector is $(6, 4, 0)$. If it were considered only the average of all five probabilities, the final answer would be that the target is on the Region 2. However, due to the extremely high probability discrepancy found at $k = 1$, the final answer was driven towards the Region 1.

k^{th} measure	**Probabilities received**	**Qualitative grade**	**Quantitative grade**	**Decision vector sum**
1	(0.841, 0.009, 0.150)	Excellent	$W_1 = (5, 0, 0)$	(5,0,0)
2	(0.311, 0.350, 0.339)	Null	$W_2 = (0, 0, 0)$	(5,0,0)
3	(0.130, 0.457, 0.413)	Regular	$W_3 = (0, 1, 0)$	(5,1,0)
4	(0.315, 0.515, 0.170)	Good	$W_4 = (0, 3, 0)$	(5,4,0)
5	(0.431, 0.311, 0.258)	Regular	$W_5 = (1, 0, 0)$	**(6,4,0)**

Table 11: Decision vector manipulation logic step by step example.

High probabilities were taken as better estimations because it was supposed that this is due to the incoming data being similar to the most common one received during the training phase, therefore characterizing this region correctly.

10 FINAL RESULTS

A final test was conducted with a person carrying the tracking device attached with a external battery in his pocket. The test consisted in the person continuously walking inside of the delimited three regions during approximately 14 min. until the sample number reached 50 – totaling 150 collected samples.

The estimations given at the end of each time window cycle of 10 seconds are classified into three types: *correct estimations*, if the prediction correctly outputs the real physical region and otherwise, *false positives*; a third one denoted as *critical errors* referred as the errors that can be considered unacceptable in a real application, such an estimation showing region 1 when in reality the person is in the 3 and vice versa. The table 12 summarizes the obtained results.

Actual region	**Correct estimations**	**False positives**	**Critical errors**	**Total accuracy**
Region 1	45/50	5/50	1/5	90.00 %
Region 2	33/50	17/50	-	66.00 %
Region 3	43/50	7/50	1/7	86.00 %
Whole area	121/150	29/150	2/29	**80.67 %**

Table 12: Collected prototype's results for accuracy test.

PART VI

CONCLUSION

11 EVALUATION OF THE CREATED PROTOTYPE

11.1 Prototype usability design

From the concerns cited in 2.2.1, the NodeMCUs' placement did not disrupted the maneuvering simulation procedure, neither the room's aesthetics – important point for better immersing the user. For the carried device, it was used a radio belt clip. A 3D printed support was made, allowing the ESP32 fixation with a double sided tape. The whole carried device can be seen in Fig.59.

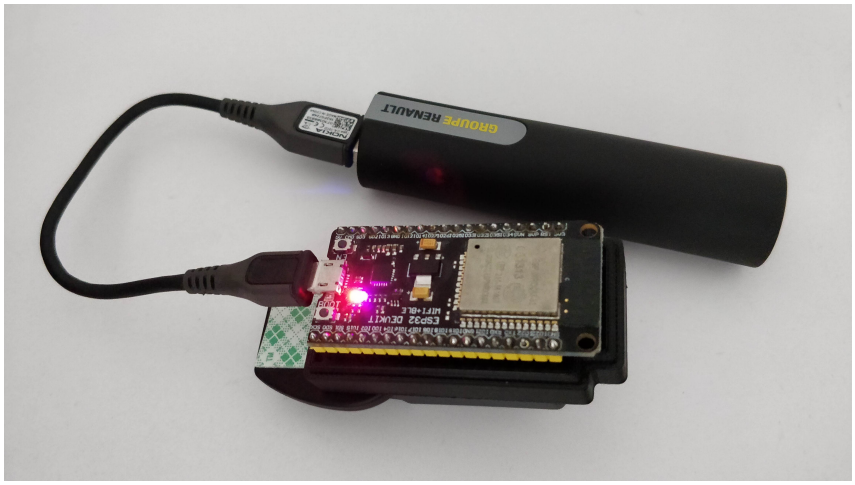


Figure 59: Device carried by the tracked person.

Since it did not occupy much space, the power bank needed for supplying energy to the ESP32 was left apart, forcing it to be carried inside the pocket as shown in Fig.60.

Figure 60: Device attached on pocket of the tracked person.

From the operational view point, the tracking can be initialized some little seconds after connecting the ESP32 to the power bank, offering no difficulty to the tracked user. However, the setup process on the CPU are still somehow complicated, requiring the manual opening of the application code from the prompt, in which will also be presented the tracking in real-time. For the future, a simplification with a simple application will be extremely helpful for more accessible usage.

11.2 Region estimator algorithm

The implemented system operates with a cycle time of 10 seconds, a little higher than the 8 seconds envisioned on 2.2.1. However, it was necessarily in order to give a better accuracy rate of 80.67% as shown in 10. Furthermore, translating into more physical estimations, as regions 1 and 3 are lesser than 1.5 square meters, it is possible to affirm that the prototype has an even higher average accuracy of 88.00%, therefore respecting the cited requirement. A higher discretization of the region 1 could eventually generate better general accuracy results and improve the system final accuracy.

11.3 Solution total final cost

The table 13 shows the prototype final total costs in Reais (Brazilian currency). The power bank described is the energy supply for the ESP32 that the tracking person carries. One important piece not considered was the cost of the Wi-Fi router simply because this

solution can use any already established local router with just a simple binding some IP addresses for more conveniently usage.

Component	Price per unit	Quantity	End price
ESP32	R$ 75.00	1 un.	R$ 75.00
NodeMCU	R$ 49.00	5 un.	R$ 245.00
Raspbery Pi 3	R$ 269.90	1 un.	R$ 269.90
Power bank	R$ 25.00	1 un.	R$ 25.00
Radio belt clip	R$ 40.00	1 un.	R$ 40.00
		Total cost	R$ 654.90

Table 13: Prototype's components costs.

Some other major costs that could not be obtained were the energy consumption of the training process neither the computers', since they were not exclusively bought for the creation of this prototype; and the costs of the 3D printed supports for the NodeMCUs and ESP32, although not being so expensive.

11.4 Solution benchmark

Comparing with similar setups found in the literature, the results were in average better than the ones found in [31] [32] and [33], as mentioned in 3.2.2.3. An explanation for this is the usage of discrete outputs (Regions) that represents various single space coordinate points (x, y), therefore simplifying the correlation burden of ANNs.

However, this prototype did not feature many of interesting system adaptable situations presented in [31] nor its comparatively low update cycle. Adding the limited scalability of this prototype, since it requires an information quintuple of RSSI constantly to obtain the average 80.67 %.

Although theoretically the application in its current state could be implemented for any number of receptors – even fewer, but not lesser than the desired regions to distinguish – as the size of the implemented environment increases or it becomes more complex with physical obstacles, walls or barriers, some receptor readings could be lost in certain positions. For treating this scenario, various submodels with fewer receptor entries could be implemented.

REFERENCES

[1] Marr, B. "**What Everyone Must Know About Industry 4.0**", Forbes, June 20th 2016. Available online at: https://www.forbes.com/sites/bernardmarr/2016/06/20/what-everyone-must-know-about-industry-4-0.

[2] Kagermann, H.; Wahlster, W.; Helbig, J. "**Recommendations for implementing the strategic initiative Industrie 4.0: Final report of the Industrie 4.0 Working Group**", *National Academy of Science and Engineering* (ACATECH), Munich: Herbert Utz Verlag, April 2013.

[3] Hermann, M.; Pentek, T.; Otto, B. "**Design Principles for Industrie 4.0 Scenarios**", in *Proceedings of the IEEE 49th Hawaii International Conference on System Sciences* (HICSS), pp. 3928-3937, Koloa, Hawaii, USA, January 2016. doi: 10.1109/HICSS.2016.488

[4] Heuchemer, B. "**Let's Set The Record Straight On Industry 4.0**", Manufacturing.net, July 06th 2016. Available online at: https://www.manufacturing.net/article/2016/06/lets-set-record-straight-industry-40.

[5] Diamandis, P. H. "**3 Major Shifts Are About to Transform Manufacturing as We Know It**", Singularity Hub, May 01st 2018. Available online at: https://singularityhub.com/2018/05/01/3-major-shifts-are-about-to-transform-manufacturing-as-we-know-it/. Accessed in May 08th 2018.

[6] Wenger, E. "**The First Law of IoT: Things that Can Be Connected, Will Be Connected**" Cisco blogs, February 25th 2016. Available online at: https://blogs.cisco.com/security/the-first-law-of-iot. Accessed in May 06th 2018.

[7] "**Anything that can be connected, will be connected**", ABB, October 18th 2017. Available online at: http://www.abb.com/cawp/seitp202/d082cbc99d347238c12581bd002fe015.aspx. Accessed in May 06th 2018.

[8] Rouse, M. "**XaaS (Anything as a Service)**", SearchCloudComputing, TechTarget, November 2017. Available online at: https://searchcloudcomputing.techtarget.com/definition/XaaS-anything-as-a-service. Accessed in May 06th 2018.

[9] Basiri, A.; Lohan, E. S.; Moore, T.; Winstanley, A.; Peltola, P.; Hill, C.; Amirian, P.; e Silva, P. F. "**Indoor location based services challenges, requirements and usability of current solutions**", *Computer Science Review*, vol. 24, pp. 1-12, May 2017. ISSN: 1574-0137. doi: 10.1016/j.cosrev.2017.03.002

[10] Kjærgaard, M. B.; Blunck, H.; Godsk, T.; Toftkjær, T.; Christensen, D. L.; Grønbæk, K. "**Indoor Positioning Using GPS Revisited**", in *Proceedings of the 8th International Conference on Pervasive Computing*, Helsinki, Finland, May 2010. Part of: Floréen, P.; Krüger, A.; Spasojevic, M. (Eds.). "**Lecture Notes in Computer Science**", vol. 6030, Berlin, Heidelberg: Springer, 2010. ISBN: 978-3-642-12654-3. doi: 10.1007/978-3-642-12654-3_3

[11] Du, H.; Zhang, C.; Ye, Q.; Xu, W.; Kibenge, P. L.; Yao, K. "**A hybrid outdoor localization scheme with high-position accuracy and low-power consumption**", *EURASIP Journal on Wireless Communications and Networking*, no. 1, December 2018. doi: 10.1186/s13638-017-1010-4

[12] Chang, L.; Xiong, J.; Wang, Y.; Chen, X.; Hu, J.; Fang, D. "**iUpdater: Low Cost RSS Fingerprints Updating for Device-Free Localization**", in *Proceedings of the 37th International Conference on Distributed Computing Systems* (IEEE ICDCS), pp. 900-910, Atlanta, GA, USA, 2017. ISBN: 978-1-5386-1792-2. doi: 10.1109/ICDCS.2017.216

[13] Caffery, J. J. "**A new approach to the geometry of TOA location**", in *Proceedings of the 52nd Vehicular Technology Conference* (IEEE VTS), vol. 4, pp. 1943-1949, Boston, MA, USA, 2000. doi: 10.1109/VETECF.2000.886153

[14] Conti, A.; Dardari, D.; Win, M. Z. "**Experimental results on cooperative UWB based positioning systems**", in *Proceedings of the IEEE International Conference on Ultra-Wideband* (IEEE ICUWB), vol. 1, pp. 191-195, Hannover, Germany, 2008. doi: 10.1109/ICUWB.2008.4653316

[15] Dardari, D.; Conti, A.; Lien, J.; Win, M. Z. "**The Effect of Cooperation on Localization Systems Using UWB Experimental Data**", *EURASIP Journal on Advances in Signal Processing*, December, 2008. doi: 10.1155/2008/513873

[16] Liu, K.; Liu, X.; Li, X. "**Guoguo: Enabling Fine-Grained Smartphone Localization via Acoustic Anchors**", *IEEE Transactions on Mobile Computing*, vol. 15, no. 5, pp. 1144-1156, May 2016. doi: 10.1109/TMC.2015.2451628

[17] Xiong, J.; Shu, L.; Wang, Q.; Xu, W.; Zhu, C. "**A Scheme on Indoor Tracking of Ship Dynamic Positioning Based on Distributed Multi-Sensor Data Fusion**", *IEEE Access*, vol. 5, pp. 379-392, 2017. doi: 10.1109/ACCESS.2016.2607232

[18] Luo, J.; Shukla, H. V.; Hubaux, J.-P. "**Non-Interactive Location Surveying for Sensor Networks with Mobility-Differentiated ToA**", in *Proceedings of the 25th IEEE International Conference on Computer Communications* (IEEE INFOCOM), pp. 1-12, Barcelona, Spain, 2006. doi: 10.1109/INFOCOM.2006.190

[19] Cook, B.; Buckberry, Graham; Scowcroft, I.; Mitchell, J.; Allen, T. "**Location by Scene Analysis of Wi-Fi Characteristics**", in *London Communications Symposium*, London, England, 2006.

[20] Schindhelm, C. K.; Gschwandtner, F.; Banholzer, M. "**Usability of apple iPhones for inertial navigation systems**", in *Proceedings of the 22nd International Symposium on Personal, Indoor and Mobile Radio Communications* (IEEE PIMRC), pp. 1254-1258, Toronto, Ontario, Canada, 2011. doi: 10.1109/PIMRC.2011.6139701

[21] Vo, Q. D.; De, P. "**A Survey of Fingerprint-Based Outdoor Localization**", *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 491-506, 2016. doi: 10.1109/COMST.2015.2448632

[22] Beauregard, S.; Haas, H. "**Pedestrian dead reckoning: A basis for personal positioning**", in *Proceedings of the 3rd Workshop on Positioning, Navigation and Communication* (WPNC' 06), pp 27-36, Hannover, German, 2006.

[23] The Editors of Encyclopaedia Britannica. "**Dead reckoning**", *Encyclopædia Britannica*. Retrieved from: https://www.britannica.com/technology/dead-reckoning-navigation. Accessed on March 26th 2018.

[24] Racko, Jan; Brida, Peter; Perttula, Arto; Parviainen, Jussi; Collin, Jussi. "**Pedestrian dead reckoning with particle filter for handheld smartphone**", in *Proceedings of IEEE International Conference on Indoor Positioning and Indoor Navigation* (IPIN), pp. 1-7, Alcalá de Henares, Madrid, Spain, 2016. doi: 10.1109/IPIN.2016.7743608

[25] Youssef, M.; Yosef, M. A.; El-Derini, M. "**GAC: Energy-Efficient Hybrid GPS-Accelerometer-Compass GSM Localization**", in *Proceedings of IEEE Global Telecommunications Conference* (GLOBECOM), pp. 1-5, Miami, Florida, USA, 2010. doi: 10.1109/GLOCOM.2010.5684304

[26] Zampella, F.; Khider, M.; Robertson, P.; Jiménez, A. "**Unscented Kalman filter and Magnetic Angular Rate Update (MARU) for an improved Pedestrian Dead-Reckoning**", in *Proceedings of IEEE/ION Position, Location and Navigation Symposium*, pp. 129-139, Myrtle Beach, South Carolina, USA, 2012. doi: 10.1109/PLANS.2012.6236874

[27] Constandache, I.; Choudhury, R. R.; Rhee, I. "**Towards Mobile Phone Localization without War-Driving**", in *Proceedings of IEEE International Conference on Computer Communications* (INFOCOM), pp. 1-9, San Diego, California, USA, 2010. doi: 10.1109/INFCOM.2010.5462058

[28] Shin, B.; Lee, J. H.; Lee, H.; Kim, E.; Kim, J.; Lee, S.; Cho, Y.-S.; Park, S.; Lee, T. "**Indoor 3d pedestrian tracking algorithm based on PDR using smarthphones**", in *Proceedings of the 12th International Conference on Control, Automation and Systems* (ICCAS), pp. 1442-1445, JeJu Island, South Korea, 2012.

[29] Li, C.; Zheng, J.; Jiang, Z.; Liu, X.; Yang, Y.; Zhang, B. "**A Novel Fuzzy Pedestrian Dead Reckoning System for Indoor Positioning Using Smartphone**", in *Proceedings of the IEEE 82nd Vehicular Technology Conference* (VTC), pp. 1-5, Boston, Massachusetts, USA, 2015. doi: 10.1109/VTCFall.2015.7390800

[30] Jekabsons, G.; Kairish, V.; Zuravlyov, V. "**An Analysis of Wi-Fi Based Indoor Positioning Accuracy**", *The Journal of Riga Technical University*, vol. 44, no. 1, pp. 131-137, 2011. doi: 10.2478/v10143-011-0031-4

[31] Altini, M.; Brunelli, D.; Farella, E.; Benini, L. "**Bluetooth indoor localization with multiple neural networks**," in *Proceedings of IEEE 5th International Symposium on Wireless Pervasive Computing* (ISWPC), pp. 295-300, Modena, Italy, 2010. doi: 10.1109/ISWPC.2010.5483748

[32] Mehmood, H.; Tripathi, N.; Tipdecho, T. "**Indoor Positioning System Using Artificial Neural Network**", *Journal of Computer Science*, vol. 6, no. 10, pp. 1219-1225, 2010. doi: 10.3844/jcssp.2010.1219.1225

[33] Pahlavani, P.; Gholami, A.; Azimi, S. "**An Indoor Positioning Technique Based on a Feed-Forward Artificial Neural Network Using Levenberg-Marquardt Learning Method**", in *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* (ISPRS), vol. XLII-4/W4, pp.435-440, 2017. doi: 10.5194/isprs-archives-XLII-4-W4-435-2017

[34] Bai, Y. B.; Wu, S.; Wu, H.; Zhang, K. "**Overview of RFID-Based Indoor Positioning Technology**", in *Proceedings of Geospatial Science Research_2 Symposium*, pp. 1-10, Melbourne, Australia, 2012. ISBN 9780987252715.

[35] Sardroud, J. M. "**Influence of RFID technology on automated management of construction materials and components**", *Scientia Iranica*, vol. 19, no. 3, pp. 381-392, June 2012. doi: 10.1016/j.scient.2012.02.023

[36] Schulcz, R.; Varga, G.; Tóth, L. "**Indoor location services and context-sensitive applications in wireless networks**", in *Proceedings of IEEE International Conference on Indoor Positioning and Indoor Navigation* (IPIN), Zurich, Switzerland, 2010. doi: 10.1109/IPIN.2010.5646764

[37] Álvarez, C. N.; Cintas, C. C. "**Accuracy evaluation of probabilistic location methods in UWB-RFID systems**", *Aalborg University: Department of Electronic Systems* (Master Thesis 10th Semester), 2010.

[38] Contreras, D.; Castro, M.; de la Torre, D. S. "**Performance evaluation of bluetooth low energy in indoor positioning systems**", *Transactions on Emerging Telecommunications Technologies*, vol. 28, no. 1, August 2014. doi: https://doi.org/10.1002/ett.2864

[39] Kalbandhe, A. A.; Patil, S. C. "**Indoor Positioning System using Bluetooth Low Energy**, in *Proceedings of IEEE International Conference on Computing, Analytics and Security Trends* (CAST), pp. 451-455, Pune, India, 2016. doi: 10.1109/CAST.2016.7915011

[40] Kul, G.; Özyer, T.; Tavli, B. "**IEEE 802.11 WLAN based Real Time Indoor Positioning: Literature Survey and Experimental Investigations**, *Procedia Computer Science*, vol. 34, pp. 157-164, 2014. doi: 10.1016/j.procs.2014.07.078

[41] **"IEEE Standard for Information technology–Telecommunications and information exchange between systems Local and metropolitan area networks–Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications**", in *IEEE Std 802.11-2016* (Revision of IEEE Std 802.11-2012), pp.1-3534, 14 Dec. 2016. doi: 10.1109/IEEESTD.2016.7786995

[42] Infsoft. "**Indoor Positioning, Tracking and Indoor Navigation with Wi-Fi**". Available in: https://www.infsoft.com/technology/sensors/wifi. Accessed on April 1st 2018.

[43] Alshami, I. H.; Ahmad, N. A.; Sahibuddin, S.; Firdaus, F. "**Adaptive Indoor Positioning Model Based on WLAN-Fingerprinting for Dynamic and Multi-Floor Environments**, *Sensors*, Eds. MDPI, Basel, Switzerland, vol. 17, no. 8, August 2017. doi: 10.3390/s17081789

[44] Mazuelas, S.; Bahillo, A.; Lorenzo, R. M.; Fernandez, P.; Lago, F. A.; Garcia, E.; Blas, J.; Abril, E. J. "**Robust Indoor Positioning Provided by Real-Time RSSI Values in Unmodified WLAN Networks**", *IEEE Journal of Selected Topics in Signal Processing*, vol. 3, no. 5, pp. 821-831, October 2009. doi: 10.1109/JSTSP.2009.2029191

[45] Medina, C.; Segura, J. C.; de la Torre, A. "**Ultrasound Indoor Positioning System Based on a Low-Power Wireless Sensor Network Providing Sub-Centimeter Accuracy**", *Sensors*, Eds. MDPI, vol. 13, no. 3, pp. 3501-3526, 2013. doi: 10.3390/s130303501

[46] Lazik, P.; Rowe, A. "**Indoor pseudo-ranging of mobile devices using ultrasonic chirps**", in *Proceedings of the 10th Conference on Embedded Network Sensor Systems*, Toronto, Ontario, Canada, 2012. Eds. ACM, pp.88-112. doi: 10.1145/2426656.2426667

[47] Plack, C. J. "**The Sense of Hearing**". 2nd edition, Eds. Psychology Press, United Kingdom, 2013.

[48] Qi, J.; Liu, G.-P. "**A Robust High-Accuracy Ultrasound Indoor Positioning System Based on a Wireless Sensor Network**", *Sensors*, Basel, Switzerland, 2017. Eds. MDPI, vol. 17, no. 11. doi: 10.3390/s17112554

[49] Xiong, J.; Shu, L.; Wang, Q.; Xu, W.; Zhu, C. "**A Scheme on Indoor Tracking of Ship Dynamic Positioning Based on Distributed Multi-Sensor Data Fusion**", *IEEE Access*, vol. 5, pp. 379-392, 2017. doi: 10.1109/ACCESS.2016.2607232

[50] Mitchell, B. "**What Does UWB Mean? An Explanation of Ultra-Wideband (UWB Definition)**", February 1st 2018. Available at: https://www.lifewire.com/ultra-wide-band-817953. Accessed on march 25th 2018.

[51] Rouse, M. "**Definition: ultra wideband**, June 2008. Available at: https://whatis.techtarget.com/definition/ultra-wideband. Accessed on March 25th 2018.

[52] Alarifi, A.; Al-Salman, A.; Alsaleh, M.; Alnafessah, A.; Al-Hadhrami, S.; Al-Ammar, M. A.; Al-Khalifa, H. S. "**Ultra Wideband Indoor Positioning Technologies: Analysis and Recent Advances**", *Sensors*, Basel, Switzerland, vol. 16, no. 5, 2016. doi: 10.3390/s16050707

[53] A. L. Samuel, "**Some Studies in Machine Learning Using the Game of Checkers**," in *IBM Journal of Research and Development*, vol. 3, no. 3, pp. 210-229, July 1959. doi: 10.1147/rd.33.0210

[54] Gero, J. S.; Sudweeks, F. "**Artificial Intelligence in Design '96**". 1st edition, Eds. Kluwer Academic Publishers, Dordrecht, Netherlands, 1996. doi: 10.1007/978-94-009-0279-4

[55] Mitchell, T. M. "**Machine Learning**", 1st edition, Eds McGraw Hill, 1997. ISBN: 0070428077

[56] Vapnik, V. N. "**Estimation of Dependences Based on Empirical Data**", *Nauka*, Moskow, USSR, 1979. English translation: Springer Verlag, New York, USA, 1982.

[57] Vapnik, V. N.; Chervonenkis, A. Y. "**On the uniform convergence of relative frequencies of events to their probabilities**", *Theory of Probability and its Applications*, vol. 16, no. 2, pp. 283–305, 1971. doi: 10.1137/1116025

[58] OpenCV dev team. "**Introduction to Support Vector Machines**", last update in November 10th, 2014. Available online at: `https://docs.opencv.org/3.0-beta/doc/tutorials/ml/introduction_to_svm/introduction_to_svm.html`. Accessed in: September, 2018.

[59] Winston, P. "**MIT 6.034: Artificial Intelligence**": online course. Massachusetts Institute of Technology: MIT, Fall 2010. Available online at: `https://www.youtube.com/watch?v=_PwhiWxHK8o`

[60] Boser, B. E.; Guyon, I. M.; Vapnik, V. N. "**A training algorithm for optimal margin classifiers**", in *Proceeding of the 5th Annual Workshop of Computational Learning Theory*, vol. 5, pp. 144-152, Pittsburgh, USA, 1992.

[61] Jain, R. "**Simple Tutorial on SVM and Parameter Tuning in Python and R**", *Harckearth blog*, February 21st, 2017. Available online at: `https://www.hackerearth.com/blog/machine-learning/simple-tutorial-svm-parameter-tuning-python-r/`. Accessed in: September, 2018.

[62] Chang, C.-C.; Lin, C.-J. "**LIBSVM: A Library for Support Vector Machines**", in *ACM Transactions on Intelligent Systems and Technology* (TIST), vol. 2, no. 3, pp. 27:1-27:27. New York, USA: ACM, 2011. doi: 10.1145/1961189.1961199

[63] Cortes, C.; Vapnik, V. N. "**Support-vector networks**, in *Machine Learnin*, vol. 20, no. 3, pp. 273-297, 1995. doi: 10.1007/BF00994018

[64] Kohavi, R. "**A study of cross-validation and bootstrap for accuracy estimation and model selection**", in *Proceedings of International Joint Conferences on Artificial Intelligence Organization* (IJCAI), pp. 1137-1143, 1995.

[65] Milgram, J.; Cheriet, M.; Sabourin, R. "**"One Against One" or "One Against All": Which One is Better for Handwriting Recognition with SVMs?**", in *10th International Workshop on Frontiers in Handwriting Recognition*, La Baule, France, October 2006.

[66] DeepAI. "*Neural Network*". Available at: `https://deepai.org/machine-learning-glossary-and-terms/neural-network`. Accessed on October 23rd 2018.

[67] Chollet, F. "**Keras: The Python Deep Learning library**", 2015. Available online at: `https://keras.io`. Accessed in: October, 2018.

[68] Redmon, J.; Farhadi, A. "**YOLOv3: An Incremental Improvement**", *ArXiv e-prints*, e-print: 1804.02767, April 2018.

[69] Kang, N. "**Multi-Layer Neural Networks with Sigmoid Function**", *Towards Data Science*, June 27th, 2017. Available online at: https://towardsdatascience.com/multi-layer-neural-networks-with-sigmoid-function-deep-learning-for-rookies-2- Accessed in: October 2018.

[70] Bishop, C. M. "**Pattern Recognition and Machine Learning**". New York, USA: Springer-Verlag, 2006. ISSN: 1613-9011

[71] Rosebrok, A. "**Softmax Classifiers Explained**", *Pyimagesearch*, September 12th, 2016. Available online at: https://www.pyimagesearch.com/2016/09/12/softmax-classifiers-explained/. Accessed in September, 2018.

[72] Lan, H. "**The Softmax Function, Neural Net Outputs as Probabilities, and Ensemble Classifiers**", *Towards Data Science*, November 13th, 2017. Available online at: https://towardsdatascience.com/the-softmax-function-neural-net-outputs-as-probabilities-and-ensemble-classifi Accessed in: October, 2018.

[73] Karim, F.; Majumdar, S.; Darabi, H.; Chen, S. "**LSTM Fully Convolutional Networks for Time Series Classification** " *IEEE Access*, vol. 6, pp. 1662-1669, 2018. doi: 10.1109/ACCESS.2017.2779939

[74] Rosebrok, A. "**Multi-label classification with Keras**", *Pyimagesearch*, May 7th, 2018. Available online at: https://www.pyimagesearch.com/2018/05/07/multi-label-classification-with-keras/. Accessed in September, 2018.

[75] Kieffer, P. "**Multiclass Iris prediction with tensorflow keras**", *Kanggle*. Open code available at: https://www.kaggle.com/pierrek20/multiclass-iris-prediction-with-tensorflow-keras. Accessed in: October, 2018.

[76] DiPietro, R. "**A Friendly Introduction to Cross-Entropy Loss**", May 2nd, 2016. Available online: https://rdipietro.github.io/friendly-intro-to-cross-entropy-loss/. Accessed in: October, 2018.

[77] Bushaev, V. "**How do we 'train' neural networks?**, *Towards Data Science*, November 27th, 2017. Available online at: https://towardsdatascience.com/how-do-we-train-neural-networks-edd985562b73. Accessed in: October, 2018.

[78] Ruder, S. "**An overview of gradient descent optimization algorithms**", *ArXiv e-prints*, e-print: 1609.04747, September, 2016. Available online at: https://arxiv.org/pdf/1609.04747.pdf.

[79] Hinton, G. E.; Srivastava, N.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. R. "**Improving neural networks by preventing co-adaptation of feature detectors**", *ArXiv e-prints*, e-print: 1207.0580, July, 2012. Available online at: https://arxiv.org/pdf/1207.0580.pdf.

[80] Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. "**Dropout: A Simple Way to Prevent Neural Networks from Overfitting**", *Journal of Machine Learning Research*, vol. 15, pp. 1929-1958, 2014.

[81] NodeMCU team. "Home page". Available online at: `http://nodemcu.com/index_en.html`. Accessed in: October 2018.

[82] Espressif. "ESP32 overview". Available online at: `https://www.espressif.com/en/products/hardware/esp32/overview`. Accessed in: Ocotber 2018.

[83] Raspberry Pi Foundation. "Raspberry Pi 3 Model B". Available online at: `https://www.raspberrypi.org/products/raspberry-pi-3-model-b/`. Accessed in: October 2018.

[84] PlatformIO dev team. "Home page". Available online at: `https://platformio.org/`. Accessed in: October 2018.

[85] Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Müller, A.; Nothman, J.; Louppe, G.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; Duchesnay, É. "**Scikit-learn: Machine Learning in Python**", *Journal of Machine Learning Research* (JMLR), vol. 12, pp. 2825-2830, January, 2012. ISSN: 1532-4435

[86] Al Qathrady, M.; Helmy, A. "**Improving BLE Distance Estimation and Classification Using TX Power and Machine Learning: A Comparative Analysis**", in *Proceedings of the 20th ACM International Conference on Modelling, Analysis and Simulation of Wireless and Mobile Systems* (MSWIM'17), pp.79-83, Miami, Florida, USA, 2017. doi: 10.1145/3127540.3127577

[87] Ma, Z.; Poslad, S.; Bigham, J.; Zhang, X.; Men, L. "**A BLE RSSI Ranking based Indoor Positioning System for Generic Smartphones**", *Wireless Telecommunications Symposium* (WTS), pp. 1-8, Chicago, Illinois, 2017. doi: 10.1109/WTS.2017.7943542

[88] Barai, S.; Biswas, D.; Sau, B. "**Estimate distance measurement using NodeMCU ESP8266 based on RSSI technique**", in *Proceedings of IEEE Conference on Antenna Measurements & Applications* (CAMA), pp. 170-173, Tsukuba, Japan, 2017. doi: 10.1109/CAMA.2017.8273392

[89] Gast, M. S. "**802.11 Wireless Networks: The Definitive Guide**", 2nd edition. Sebastopol: O'Reilly Media, Inc, 2011. ISBN: 0596100523.

[90] Parziale, L.; Britt, D. T.; Davis, C.; Forrester, J.; Liu, W.; Matthews, C.; Rosselot, N. "**TCP/IP Tutorial and Technical Overview**", 8th edition. IBM Redbook. Available online at: `https://www.redbooks.ibm.com/pubs/pdfs/redbooks/gg243376.pdf`

[91] Dozat, T. "**Incorporating Nesterov Momentum into Adam**", in *International Conference on Machine Learning Workshop* (ICLR), 2016.