UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Carlos Raúl Morales Hernández

# Analysis of a Wireless Sensor Network behavior using Machine Learning Techniques

Florianópolis, Santa Catarina, Brazil
2021

Carlos Raúl Morales Hernández

# Analysis of a Wireless Sensor Network behavior using Machine Learning Techniques

Dissertação submetida ao Programa de Pós-Graduação
em Engenharia Elétrica da Universidade Federal de
Santa Catarina para a obtenção do título de mestre
em Engenharia Elétrica.
Orientador: Prof. Fernando Rangel de Sousa, Dr.

Florianópolis, Santa Catarina, Brazil
2021

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Hernández, Carlos Raúl Morales
   Analysis of a wireless sensor network behavior using machine learning techniques / Carlos Raúl Morales Hernández ; orientador, Fernando Rangel de Sousa, 2021.
   116 p.

   Dissertação (mestrado) - Universidade Federal de Santa Catarina, Centro Tecnológico, Programa de Pós-Graduação em Engenharia Elétrica, Florianópolis, 2021.

   Inclui referências.

   1. Engenharia Elétrica. 2. Wireless sensor networks. 3. Deep learning. 4. Time series. 5. Energy saving. I. Sousa, Fernando Rangel de. II. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Engenharia Elétrica. III. Título.

Carlos Raúl Morales Hernández

**Analysis of a Wireless Sensor Network behavior using Machine Learning Techniques**

O presente trabalho em nível de mestrado foi avaliado e aprovado por banca examinadora composta pelos seguintes membros:

Prof. Jefferson Luiz Brum Marques, Ph.D.
Universidade Federal de Santa Catarina

Prof. Maicon Deivid Pereira, Dr.
Universidade Federal da Bahia

Nestor Charles Fernandes, Eng.
Traceback Technologies

Certificamos que esta é a **versão original e final** do trabalho de conclusão que foi julgado adequado para obtenção do título de mestre em Engenharia Elétrica.

ICP-Edu
Documento assinado digitalmente
Telles Brunelli Lazzarin
Data: 08/11/2021 11:40:45-0300
CPF: 023.400.719-25
Verifique as assinaturas em https://v.ufsc.br

Prof. Telles Brunelli Lazzarin, Dr
Coordenador

ICP-Edu
Documento assinado digitalmente
Fernando Rangel de Sousa
Data: 05/11/2021 14:20:37-0300
CPF: 884.649.114-91
Verifique as assinaturas em https://v.ufsc.br

Prof. Fernando Rangel de Sousa, Dr.
Orientador

Florianópolis, Santa Catarina, Brazil, 2021.

This work is dedicated to my family and friends.

## ACKNOWLEDGEMENTS

- To my parents Dilys and Raúl for their guidance, their love, and caring all the time. For being the parents anyone can wish for.
- To Amanda, for being with me always, in good, but more important during bad times. For your support through this horrible pandemic. For being my backbone always.
- To my tutor, Fernando Rangel de Sousa, for his guidance, for giving a chance to this foreign student, for believing in me, his patience with my portunhol, the long walks, and all the shared knowledge.
- To Orlando and Niurka for taking care of my mom, to my aunts Idania and Alberto, and all my family for their love and unconditional support.
- To my in-laws, Susset and Alexander, for their support, their advice, and their company.
- To all the wonderful teachers I had during the master's degree.
- To CAPES for its funding, which made this work possible.
- To my friends from Cuba and our long talks about sacrifice and a brighter future.

*“Your time is limited
so don’t waste it living someone else’s
life. Don’t be trapped by dogma - which is
the result of other people’s thinking.”
(Steve Jobs, 1955-2011)*

## ABSTRACT

This work presents a review of the main concepts of Wireless Sensor Network (WSN)s and tackles one of their main problems, which is energy consumption. This is done using the data collected by a network deployed at the Hydropower power plant in Cachoeira Dourada. First, an exploratory data analysis of the WSN using statistical and machine learning methods was performed to discover insights about the current state of the network. The analysis provided information about which nodes are more stable, correlations between the data that can be exploited to optimize transmissions, and information about the stability of the links. The work also proposes the use of a Deep Learning model, in a dual prediction scheme, to reduce the transmissions between devices in the network, reduce congestion, and save energy. To do so, a review of data prediction strategies in WSNs is performed. Different neural network based models are introduced and compared using different error metrics in prediction. Finally, a measure of the reduction in data transmission is given, considering different error thresholds. Results show that the model can save a considerable amount of data in transmission, from 70% to 90%, and still maintain a good representation of the measured data.

**Keywords**: Wireless sensor networks. Deep learning. Time series. Energy saving.

## RESUMO

Este trabalho apresenta uma revisão dos principais conceitos das WSN e aborda um de seus principais problemas, que é o consumo de energia. Isso é feito a partir dos dados coletados por uma rede implantada na Usina Hidrelétrica de Cachoeira Dourada. Primeiro, uma análise exploratória de dados da WSN usando métodos estatísticos e de aprendizado de máquina foi realizada para descobrir padrões e entender o estado atual da rede. A análise forneceu informações sobre quais nós são mais estáveis, correlações entre os dados que podem ser explorados para otimizar as transmissões e informações sobre a estabilidade dos links. O trabalho também propõe a utilização de um modelo de Deep Learning, em esquema de predição dual, para reduzir as transmissões entre dispositivos da rede, diminuir o congestionamento e economizar energia na transmissão. Para fazer isso, uma revisão das estratégias de previsão de dados em WSNs é realizada. Diferentes modelos baseados em redes neurais são introduzidos e comparados usando diferentes métricas de erro na predição. Finalmente, uma medida da redução na transmissão é dada, considerando diferentes limiares de erro. Os resultados mostram que o modelo pode economizar uma quantidade considerável de dados na transmissão e ainda manter uma boa representação dos dados medidos.

**Palavras-chave**: Redes de sensores sem fio. Aprendizado profundo. Séries temporais. Conservação de energia.

## RESUMO EXPANDIDO

### Introdução

As Redes de sensores sem fio (WSNs - Wireless Sensor Networks) são uma área de pesquisa popular com muitas aplicações possíveis. Em comparação com redes cabeadas, uma WSN oferece mais benefícios, por exemplo, o tempo e o custo necessários para instalação são menores, podem ser acopladas a corpos móveis para monitorar seus parâmetros de perto, e o ambiente monitorado pode ser reconfigurado mais facilmente. Esta tecnologia é uma escolha atraente para soluções inteligentes, como automação industrial e residencial, monitoramento de integridade estrutural, monitoramento de eventos naturais extremos e aplicações militares. No entanto, essas redes de sensores são normalmente alimentadas por bateria e às vezes sua localização torna a substituição das baterias uma tarefa muito difícil, portanto, a conservação de energia é um dos desafios mais importantes nestas redes. Em uma WSN aproximadamente 80% da energia é consumida na transmissão e depois no sensoriamento e processamento, portanto, é muito importante estender sua vida útil o máximo possível. Muitos protocolos de roteamento e comunicação foram projetados para resolver este problema, além de circuitos de ultra baixa potência e sistemas de captação de energia. Outra solução possível para melhorar o tempo de vida da rede, e aquela explorada neste trabalho, é usar métodos de previsão de dados para reduzir o congestionamento e a transmissão de dados desnecessários.

### Objetivos

O objetivo principal deste trabalho é a proposta de um método de previsão para reduzir a transmissão de dados em uma WSN, localizada em uma usina hidrelétrica em Cachoeira Dourada, utilizando estratégias de Deep Learning em um Esquema de Predição Dupla (DPS - Dual Prediction Scheme), para reduzir o consumo de energia e melhorar a vida útil dos sensores.
Os objetivos específicos deste trabalho são:

a) Realizar uma revisão bibliográfica dos principais aspectos das WSN, como arquitetura de sensores, topologias e aplicações.
b) Estudo dos métodos exploratórios de análise de dados.
c) Aquisição dos dados do banco de dados da rede de sensores.
d) Compreender, reformatar e limpar os dados extraídos.
e) Explorar e experimentar os dados usando métodos estatísticos e visualizações para extrair ideias e possíveis problemas.
f) Detecção de anomalias nos dados.
g) Estudo de diferentes estratégias de redução de energia em WSN usando algoritmos de previsão de dados.
h) Estudo de algoritmos para previsão de séries temporais.
i) Propor e desenvolver um modelo de previsão de dados para economizar energia na transmissão.

## Metodologia

A metodologia para atingir os objetivos propostos iniciou por uma ampla revisão dos principais conceitos, arquiteturas, aplicações e desafios atuais das Redes de Sensores Sem Fio. Através desta extensa revisão, foi possível determinar as abordagens atuais para a tarefa de conservação de energia em Redes de Sensores Sem Fio e os algoritmos utilizados na literatura. Depois disso, outra revisão dos métodos de previsão de séries temporais foi realizada. Essa revisão permitiu entender os algoritmos e fazer uma suposição daqueles que teriam um melhor desempenho com os dados. Em seguida, foi realizada uma análise exploratória dos dados, em Python, utilizando análises estatísticas e técnicas de aprendizado de máquina, incluindo algoritmos de detecção de anomalias, que permitiram encontrar correlações, compreender a estrutura do conjunto de dados e o estado geral dos sensores. Por fim, foi realizado o desenvolvimento dos algoritmos de predição, também em Python com Tensorflow como framework para Redes Neurais Profundas. Várias previsões foram realizadas no conjunto de teste para comparar os algoritmos e definir qual deles apresentou a maior precisão. Além disso, uma comparação da taxa de transmissão reduzida foi feita em uma simulação do esquema de previsão dupla.

## Resultados e Discussão

Uma análise de detecção de anomalias foi realizada usando técnicas básicas e de aprendizado de máquina. A análise permitiu a detecção de comportamentos estranhos nos dados de temperatura. Um algoritmo baseado em redes neurais (Sequence to Sequence with Attention) foi desenvolvido e testado no conjunto de dados para reduzir a quantidade de dados transmitidos de nós sensores para os Cluster Heads ou Estações Base, em um esquema de predição dupla para otimizar a energia (Dual Prediction Scheme). O método foi capaz de fornecer previsões precisas e previsões estáveis de longo prazo. A porcentagem de pontos transmitidos e a precisão dos dados finais podem ser ajustados ajustando um valor limite. Os resultados mostraram que foi possível economizar até 90% na transmissão mantendo uma transmissão com precisão relativamente boa dos dados transmitidos, concluindo que o algoritmo, em tese, pode reduzir consideravelmente a energia consumida na transmissão.
A solução resultou em um artigo apresentado na Conferência Internacional de Tecnologia de Medição e Instrumentação de 2021 IEEE (I2MTC) e publicado pelo IEEE sob o nome de Predição de Dados Multivariados em uma Rede de Sensor Wireless baseada em Modelos de Sequência a Sequência.

## Considerações Finais

Os objetivos objetivos traçados para o projeto foram alcançados. Os algoritmos e a análise dos dados foram realizados em Python, usando Tensorflow como framework para Deep Learning. Os algoritmos de detecção de anomalias provaram ser eficazes em encontrar comportamentos estranhos nos dados de temperatura. O algoritmo de predição de dados mostrou-se muito eficaz, reduzindo consideravelmente a quantidade de transmissão, dependendo da precisão necessária à aplicação. O algoritmo não foi implementado em uma rede real devido aos impedimentos apresentados pela

pandemia Covid-19. A próxima etapa é implementar o algoritmo de redução de dados em uma rede real para verificar o desempenho real e a compensação entre a complexidade e a redução no número de transmissões.

**Palavras-chave**: Redes de sensores sem fio. Aprendizado profundo. Séries temporais. Conservação de energia.

# LIST OF FIGURES

## LIST OF TABLES

## LIST OF ABBREVIATIONS AND ACRONYMS

| | |
|---|---|
| 1D-CNN | One Dimensional Convolutional Neural Networks |
| ADC | Analog to Digital Converter |
| AI | Artificial Intelligence |
| ANN | Artificial Neural Network |
| ARIMA | Auto Regressive Integrated Moving Average |
| BCDCP | Base station Controlled Dynamic Clustering Protocol |
| BS | Base Station |
| CH | Cluster Head |
| CNN | Convolutional Neural Networks |
| DETS | Double Exponential Smoothing |
| DL | Deep Learning |
| DNN | Deep Neural Networks |
| DPS | Dual Prediction Scheme |
| ED | Event Detection |
| EMI | Electro Magnetic Interference |
| ETS | Exponential Smoothing |
| FCL | Fully Connected Layer |
| FFD | Fully Functional Device |
| GRNN | General Regression Neural Network |
| GRU | Gated Recurrent Unit Network |
| IoT | Internet of Things |
| IQR | Interquartile Range |
| LEACH | Low-Energy Adaptive Clustering Hierarchy |
| LSTM | Long Short-Term Memory Neural Network |
| MAE | Mean Absolute Error |
| ML | Machine Learning |
| MLP | Multilayer Perceptron |
| PCA | Principal component analysis |
| RFD | Reduced Functional Device |
| RMSE | Root Mean Squared Error |
| RNN | Recurrent Neural Networks |
| RSSI | Received Signal Strength Indicator |
| Seq2Seq | Sequence to Sequence |
| SGD | Stochastic Gradient Descent |
| SPE | Spatial Process Estimation |
| SPS | Single Prediction Scheme |
| SVM | Support Vector Machine |
| WSN | Wireless Sensor Network |

# CONTENTS

# 1 INTRODUCTION

Advances in microelectronics have impulsed the development of small, low-cost, and low-power electronic devices capable of sensing the environment and with considerable processing capacity. These developments have been, in part, fueled by the rapid advances of the Internet of Things (IoT) and our necessity of connectivity. IoT has made possible applications such as Smart Cars, Traffic Management Systems, Smart Houses, Digital Twins, and many others. All these applications need to collect data from the environment or devices, this has made Wireless Sensor Networks (WSN) be among the most rapidly evolving technological domains because of the many advantages they provide.

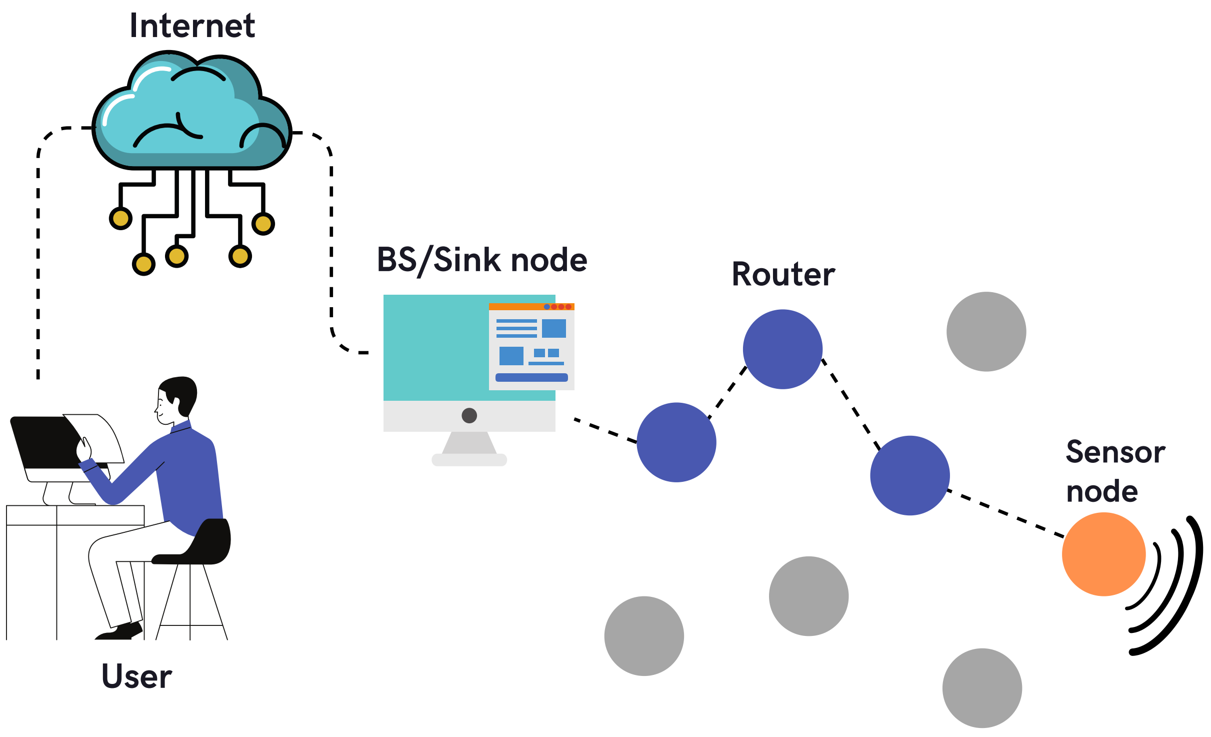
## 1.1 WIRELESS SENSOR NETWORK

WSNs are a popular research area with many possible applications. Compared with wired networks, a WSN offers more benefits, for example, the time and cost needed for installation are lower, they can be coupled with moving bodies to monitor their parameters from close, and the monitored environment can be reconfigured more easily (VAN DER BENT, 2010). This technology is an attractive choice for smart solutions, such as industry and home automation, machinery monitoring for early fault detection, predictive maintenance, environmental monitoring, smart agriculture, and other IoT applications.

A WSN consists of spatially distributed sensors, routers, and one or more Base Station (BS), also called Sink node, deployed in a region of interest, as depicted in Figure 1. Sensors can monitor physical conditions in real-time, such as temperature, vibration, motion, air quality, and many others, producing sensory data (SENOUCI; MELLOUK, 2016). Transducers can work as sensors and also as actuators, providing IoT applications with the capacity to interact with the environment. A sensor node can behave both as a sensor and as a data router. BS collect data from sensors and communicates with the end-user or WSN owner in a direct connection, also, there may be multiple BS and multiple users in the same WSN.

The concept of a network of wireless sensors is very old. It was first introduced by the United States military industry and the Defense Advanced Research Projects Agency (DARPA) (JINDAL, 2018). Some of the first uses can be traced back to the Vietnam War and were also used during the Cold War to detect soviet submarines with acoustic sensors in a still functioning network, monitoring ocean conditions (DILHAC; BOITIER, 2016; JINDAL, 2018). The use and popularity of these networks expanded from the military complex to the rest of the industries as the price and size of electronic devices reduced. Applications based in these networks can be scaled relatively easily by adding more devices when the network follows a distributed architecture. Increasing the

Figure 1 – Typical WSN elements.



Source – Author.

number of nodes can even improve the state of the network. They also require lower budgets for deployment since they do not require cabling or a predefined structure and sensors are very inexpensive. They can provide solutions for a wide range of applications due to their adaptability to different environments, for example, sensor nodes can be installed underwater, they can be dropped from airplanes into battlefields or in very harsh environments to collect real-time measurements.

While sensor networks provide many advantages, they also present some drawbacks. A problem with these networks is the security. It is possible that sensors suffer from physical attacks in harsh, non monitored environments, this makes that security of sensors cannot be assured (SHARMA et al., 2014). Also, deployment in vast areas creates many access points to the network. This is why security is one of the main problems to be tackled in WSNs. Another major issue is energy consumption since most devices use batteries as an energy source, therefore it is very important to extend their life as long as possible. Many routing and communication protocols have been designed to tackle this problem, besides ultra low power circuits and energy harvesting systems. Another possible solution to improve the lifetime of the network is to use data prediction methods to reduce congestion by reducing the amount of unnecessary data transmission.

## 1.2 OBJECTIVES

The general and specific objectives of this work are described below.

### 1.2.1 General Objective

The main goal of this work is the proposal of a forecasting method to reduce data transmission in a WSN, located in a hydropower plant at Cachoeira Dourada, using Deep Learning strategies in a Dual Prediction Scheme, to reduce energy consumption and improve the lifetime of the sensors.

### 1.2.2 Specific Objectives

a) To perform a bibliographic review of the main aspects of WSN, such as the architecture of sensors, topologies, and applications.

b) Study of the exploratory data analysis methods.

c) Acquisition of the data from the Wireless Sensor database.

d) Understanding, reformat, and clean the extracted data.

e) To explore and experiment with the data using statistical methods and visualizations to extract insights and possible problems.

f) Detection of abnormal behavior in the data.

g) Study of different strategies for energy reduction in WSNs using data prediction algorithms.

h) Study of algorithms for time series forecasting.

i) Propose and develop a model for data prediction to save energy in transmission.

# 2 STATE OF THE ART. A SURVEY ON WIRELESS SENSOR NETWORKS

## 2.1 INTRODUCTION

This chapter presents a review of the main concepts of Wireless Sensor Networks. The structure of the main components of the network is described together with the main topologies and organizations. Also the main fields of applications with examples of these.

## 2.2 SENSOR NODES ARCHITECTURE

Sensor nodes are the core element of WSNs, they sense, collect, process and transmit measures from the environment or a particular process. These devices are usually very inexpensive when compared to conventional sensors. Sensor nodes, also called motes, are electrical devices that consist of a processing unit with a storage unit, a transceiver, sensors with an Analog to Digital Converter (ADC), and a power source (MOSCHITTA; NERI, 2014), Figure 2 presents the components in a typical sensor node.

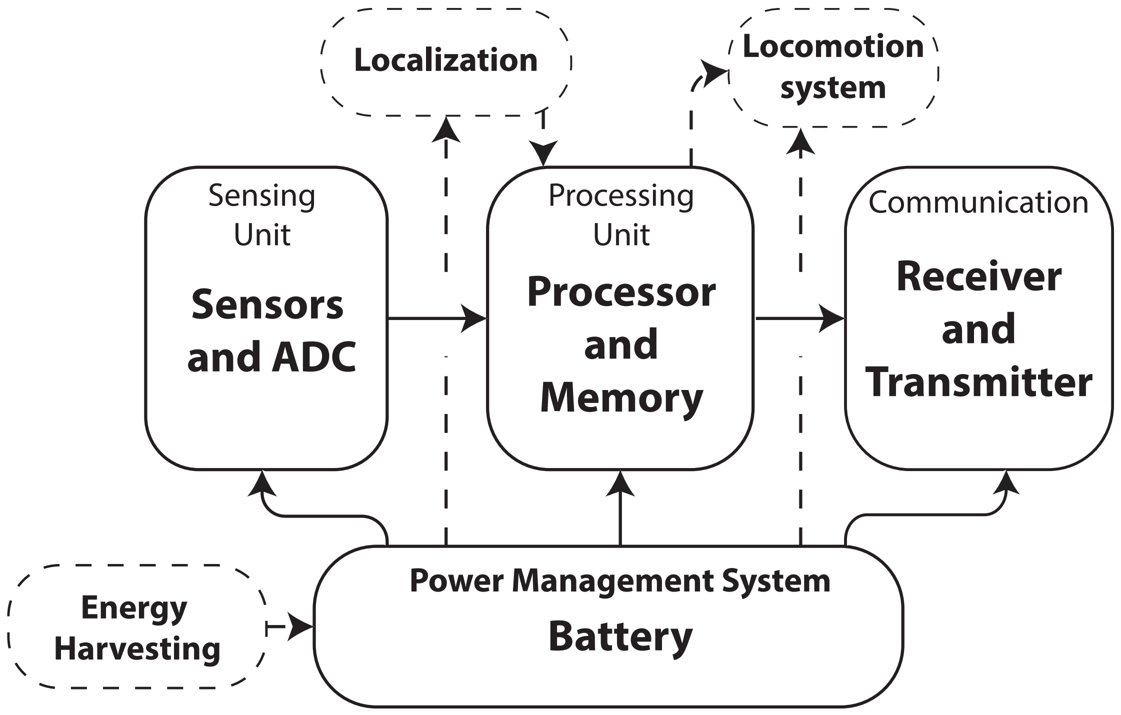
The power source is usually a battery that powers the entire sensor node. Sensors, depending on the application, are sometimes deployed in regions of hard access, this makes prolonging battery lifetime one of the key aspects of WSNs. The selection of the battery depends on the power consumption of the mote, the lifetime required by the application, deployment period, and cost. The Lead Acid battery is widely used in designing nodes where the deployment period of nodes is more than four months and the deployed WSN solution is needed to be cost-effective (KUMAR; REDDY, 2020).

The communication unit consists of a bidirectional wireless communication channel. The unit can be infrared, optical, RF short range radios, ultrasonic, and inductive fields, being the RF short range radios the most cost-effective and presenting lower power consumption. The majority of the power of the sensor node is consumed by this unit, therefore they can be set to sleep mode when is not required (KUMAR; REDDY, 2020).

The processing unit has an internal memory, and a microcontroller, which can store and process measures from the sensing unit. The sensing unit connects the sensor node to the physical world by measuring the environment and acting on it (if there are actuators). The sensing unit can also have a memory for keeping a log of the data (ABDULKAREM et al., 2020).

Depending on the application, sensors can also have an Energy Harvesting system, which provides the motes with the ability to gather energy from some external source, the sensor can have a power management system connected to solar panels, that way the is not limited to the autonomy given by the battery. When the WSN is large, mobile, or is deployed randomly in a region, sensors can include a localization system so the owners can know the location in case a reparation or replacement is needed. WSNs can be static, mobile, or hybrid, regarding to their physical distribution. When the network is mobile, sensors can include a locomotion system to navigate in a determined region, the node can also be embedded in some mobile device, in that case, is not part of the sensor node.

Figure 2 – Typical architecture of a sensor node.

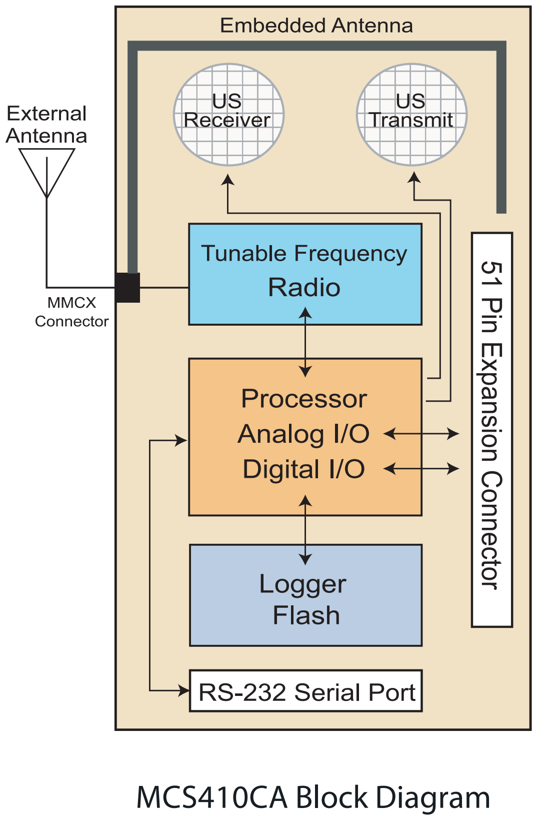


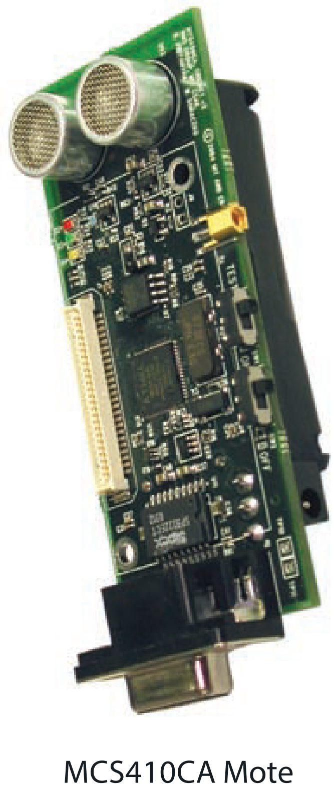
Source – Author.

A Base Station, also known as a Sink or Gateway, is a device with better communication capacities and more processing power than nodes of the network. The BS is the gateway between the user and the sensor nodes, which means that all data flows to the BS, therefore it must be capable of managing large amounts of traffic. The power consumption in these devices is not a problem, because it can be directly connected to the power supply. The location of the base station can significantly affect the lifetime performance and throughput of the network (ABDULKAREM et al., 2020).

Figure 3 presents an example of the commercial mote MCS410CA, Cricket Mote, which is a low-power, small, wireless smart sensor (INC., n.d.). The device has support for Light, Temperature, Barometric Pressure, Acceleration/Seismic, Acoustic, Magnetic, GPS, and other sensors, with up to one year of Battery Life on AA Batteries (Using Sleep Modes). The Cricket Mote includes an ultrasonic sensor that is used to measure location.

Figure 3 – Example of Commercial Mote MCS410CA.





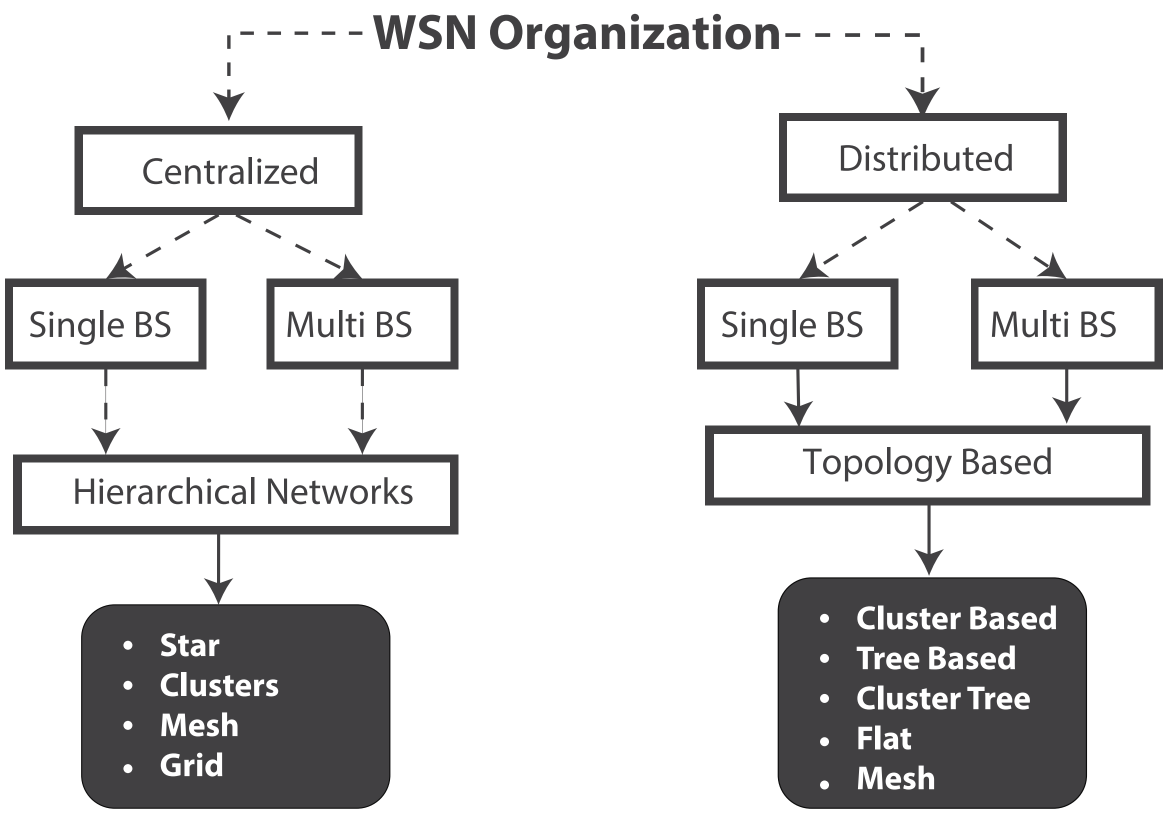
Source – Author.

## 2.3 WSN ORGANIZATION

The application determines the number of sensors, which determines the organization of the network, its coverage, connectivity, and design. For applications that demand wide areas to be covered, taking into account the battery duration of sensors and the possibility of damaged nodes during deployment, and later malfunctioning, large numbers of sensors are usually deployed. In such cases, the design of the network needs to be scalable, this is usually made by clustering sensor nodes based on their localization, and the environment or process they sense (ABBASI; YOUNIS, 2007). However, this may not be the case for all applications. We can classify WSNs by their organization in two categories, Centralized and Distributed, and subsequent categories, as presented in Figure 4.

In a centralized organization, a central node coordinates the flow of information to the network. Sensor nodes execute a schedule generated by the central node. In this scheme, the central node is in charge of organizing every task in the network, such as programming sleep times, enter in low-power modes, schedule transmissions, inform the rest of the nodes about changes or decisions made by the network owners and generate routing paths and distribute them to each node.

Figure 4 – WSN Organization Classification.



Source – Author.

In contrast, in a distributed organization, nodes are more autonomous, they schedule their own tasks and data processing. Each node builds its own routing information by talking with each other and decisions are locally taken (SONG, J. et al., 2007). Following the classification scheme presented in Figure 4, we can further divide both centralized and distributed networks into single-sink or multi-sink.

a) Single-sink: in this scheme, as the name suggests, there is a single BS to where all the information flows, reducing the forwarding time and simplifying the routing.

b) Multi-sink: in a multi-sink scheme, the information flows to different BS, where tasks are distributed. These schemes are used for applications where the region of interest is large, there is a high network density, redundancy is needed, etc. Compared with single-sink networks, multi-sink networks are more widely used due to their outstanding advantages in network lifetime and energy balance (FU; YANG, 2020).

### 2.3.1 Centralized Organization

Centralized organizations are better for networks where there is going to be a device with more energy and processing capacity. This is because the central device has to be responsible for processing a large amount of received data coming from the sensors and transmit it to the BS. The central node has to provide services such as node localization, event detection, and traffic routing (CARLOS-MANCILLA et al., 2016). In a centralized architecture, there can be also many sink nodes, in that case, each one is in charge of a specific part of the network.

Centralized WSNs are usually stationary, or part stationary with some mobile parts. In a stationary network, once sensors are deployed they do not change position, they cannot navigate in the monitored environment. Nodes are deployed in well defined strategic positions. The aim is to provide better data collection and processing performance. In these structures, it must be ensured that the optimal location for the sinks and the sensors are chosen before deployment to optimize the flow of information and energy consumption.

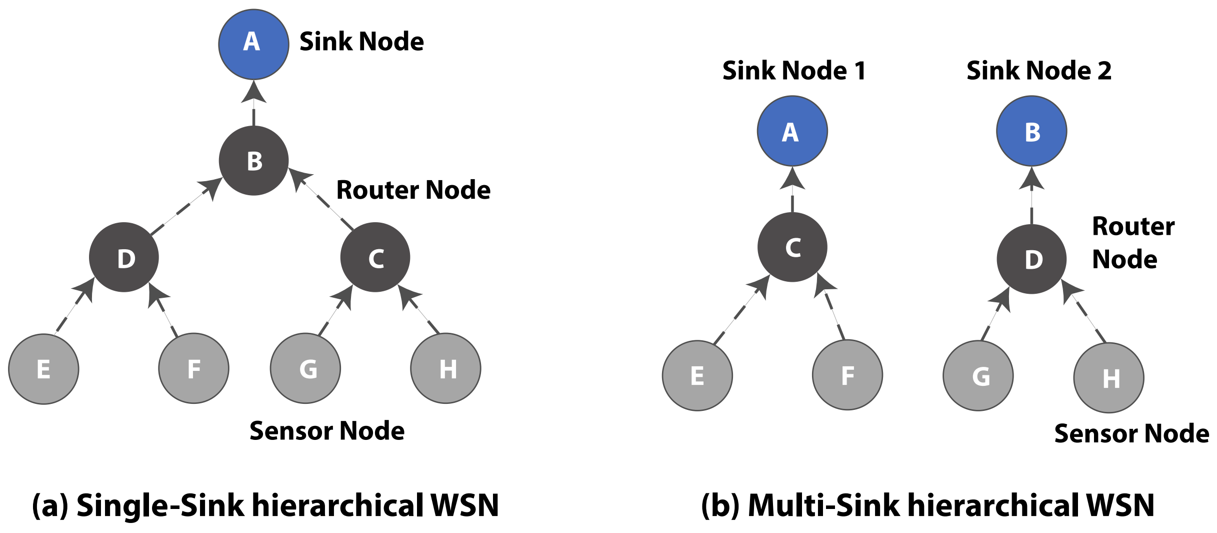
#### 2.3.1.1 Hierarchical Organized Network

In a Hierarchical Network, nodes have different roles, and priorities are defined according to these. There are usually two types of devices in these networks, Reduced Functional Device (RFD) which has limited capabilities (Sensor nodes), and Fully Functional Device (FFD). FFD can communicate with any other device in a network, but an RFD can talk only with an FFD device (Routers and Coordinators). Usually, routers and coordinators have more processing and energy resources than RFDs (FARAHANI, 2008), which makes the WSN heterogeneous.

Figure 5(a) shows an example of a Single-Sink hierarchical network. We can see how all the information flows to a single node in the top (node A), which is the BS. In this case, nodes C and D must route their own traffic, and the traffic from lower sensors to node B, and node B to A. We can expect node B to be the one receiving more traffic. Nodes B, C, and D are routers. The rest of the nodes in the bottom (E, F, G, H) are sensor nodes that only transmit information to the upper routers.

Figure 5(b) presents an example of a Multi-Sink hierarchical network. Nodes A and B are the sink nodes. Nodes C and D are routers, they transmit the data to the BS closer to them. The rest of the nodes are sensors that only transmit the information related to them.

Figure 5 – Single and Multi-Sink schemes in WSNs.



Source – Author.

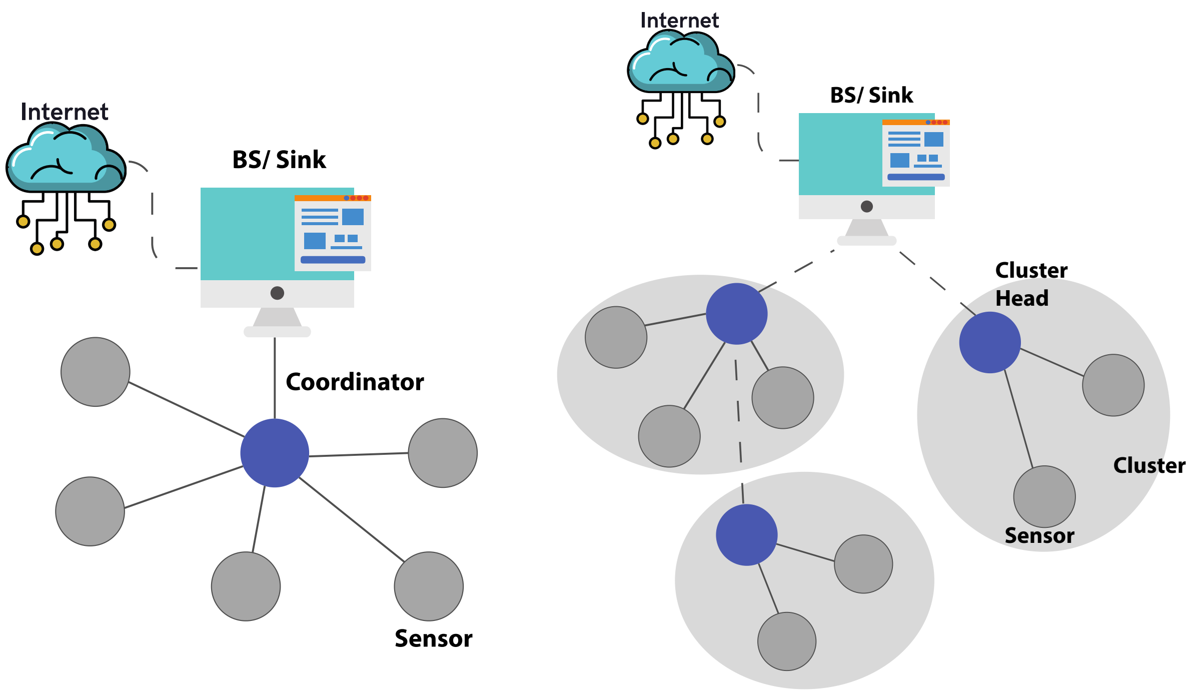
#### 2.3.1.2 Star Structure

In this structure, sensor nodes communicate with a central node or coordinator, which communicates with the BS, as presented in Figure 6(a). The coordinator can also be the BS, in this case, there is no coordinator, only sensors and a sink. All the communication must pass through the central node and sensors cannot communicate with each other. This organization present scalability problems because is not possible to stack more sensors without connecting them directly to the coordinator. An advantage of this structure is its easy installation and robustness.

#### 2.3.1.3 Cluster Based Structure

Hierarchical networks can be organized in clusters, with single or multiple BS. Clusters have a leader, also called Cluster Head (CH), see Figure 6(b). A CH can be automatically selected by the sensors or assigned when designing the network. A CH can be one of the sensor nodes or act only as a router, in the case of hierarchical networks, clusters are static and are usually deployed more carefully, in regions with more access and have more computational and energy resources. Clustering helps stabilize the network, avoid packet losses and reduce the number of packet collisions (ABBASI; YOUNIS, 2007). When the network is organized in clusters, the communication between the sensor node and the BS is responsibility of the CH. Sensors connect only with their CH, therefore they are not affected by changes in other parts of the network. CHs have the responsibility of informing the sensors about changes and decisions made by the owner and reporting the sensed data to the BS, they can also schedule tasks for the nodes.

Figure 6 – Star and Cluster organized WSN



**(a) WSN in a Star organization.** **(b) WSN in a Cluster Organization.**

Source – Author.

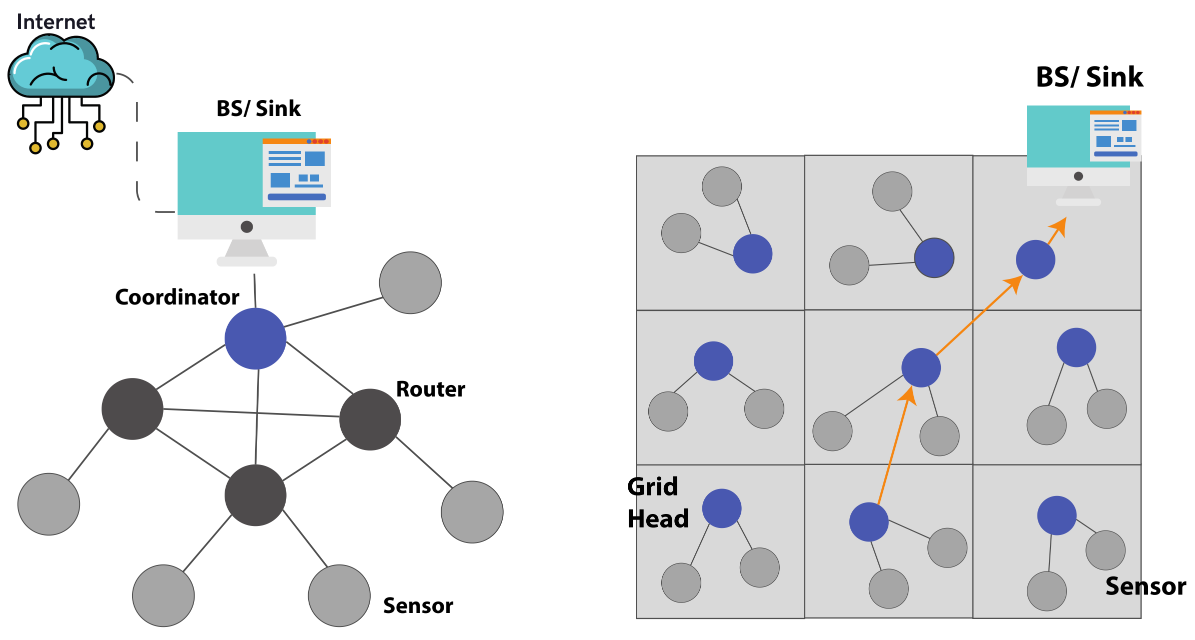
### 2.3.1.4 Mesh Structure

In a Mesh structure, data can flow through different paths from sensors to the BS. There are usually coordinators and routers involved. A mesh network, where all the devices are connected to each other is called a full mesh (SANDEEP VERMA, 2013). This topology is reliable and very robust because if a sensor node or even a router fails, communication between the rest and the BS is not affected. A Mesh topology is presented in Figure 7(a), all traffic flows to the coordinator through the routers, we can see how if a router fails, the rest of the routers and sensors are not affected. An example of a mesh network can be founded in (DENG et al., 2017). The authors designed a WSN in a Mesh structure, called DigiMesh, for collecting and monitoring the electric field under high voltage direct current (HVDC) transmission lines.

### 2.3.1.5 Grid Structure

In this topology, the region of interest is divided into square grids of the same size. There is a Grid Head in every grid (SANDEEP VERMA, 2013) (similar to CHs) responsible for receiving the data from sensors in the grid and transmit information to them. The routing is performed in a grid by grid form, grid heads communicate with other grid heads to reach the Sink, see Figure 7(b). An example of this topology can be founded in (LIU et al., 2010). The authors proposed A Low Power Grid-based Cluster Routing Algorithm of Wireless Sensor Networks (LPGCRA). The algorithm divides the network into grids according to the location of the sensors and then it reorganizes the grid into clusters which then communicates with the BS. In (XU et al., 2001) the authors used the Geographical Adaptive Fidelity protocol to reduce energy consumption, which divides the network into grids with a Grid Head per grid. Each node knows the network topology. The algorithm identifies nodes with similar routing characteristics for communication and keeps the rest of the nodes off.

Figure 7 – Mesh and Grid organized WSN



**(a) WSN in Mesh Structure.** **(b) WSN in Grid Structure.**

Source – Author.

## 2.3.2 Distributed Organization

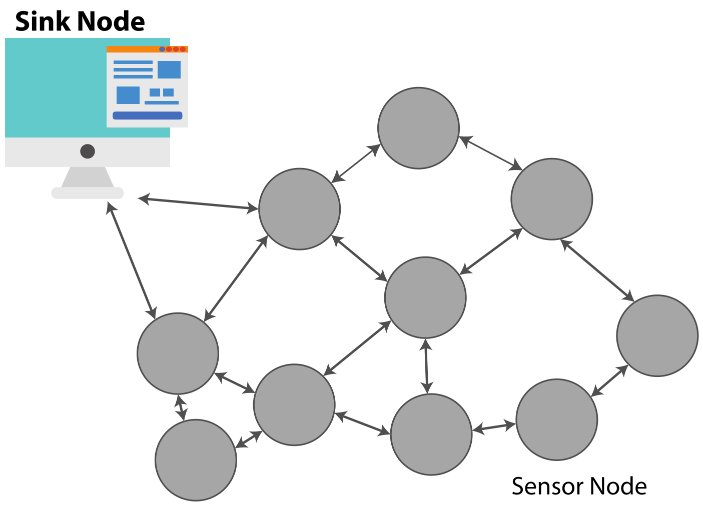
There are some applications where the location of the sensors is not known before deployment or they are randomly distributed in a region of interest. This is the case of Distributed WSNs. In these networks, nodes have not specific role assigned and there is no fixed infrastructure or previously known topology. Once sensors are deployed, they start scanning the radio coverage searching for the closest neighbors to establish a link (CAMTEPE; YENER, 2005).

As stated in (MAMUN, 2012), before analyzing different topologies in a distributed WSN, we can make some assumptions: there is a large number of nodes deployed randomly in a region of interest, with the BS outside the region and both sensors and BS stationary after deployment; the network is homogeneous, therefor all nodes have the same processing, communication, and energy capabilities and sensing is done at a fixed rate.

#### 2.3.2.1 Flat Organized Network

A flat or unstructured organization is the case of absence of topology, Figure 8 presents a distributed WSN in this setup. In this case, communication with the BS is made via multi-hop routes. A given sensor node broadcasts the data and some control information that is received or generated, if the information did not reach the destination, the operation is repeated, in a process called Flooding. This technique does not take into account energy constraints. When using the Flooding technique, duplicated packets may appear since this is a blind technique that broadcasts in many nodes, also sensors in the same region, will likely generate very similar or duplicated data (MAMUN, 2012). This is a very greedy technique when it comes to power consumption.

Figure 8 – Distributed WSN in a flat architecture.



Source – Author.

#### 2.3.2.2 Cluster Based

Clustering is particularly useful for applications that require scalability. In hierarchical networks, clusters are defined when the network is created, before deployment. This makes deployment and maintenance a more easy task but restricts the possible scenarios to predefined fields and stationary targets. The role of CH consumes more resources, therefore, in homogeneous networks, which is the case assumed for distributed WSNs, CH are rotated over multiple nodes to balance the load and energy consumption, the position is elected at different time intervals.

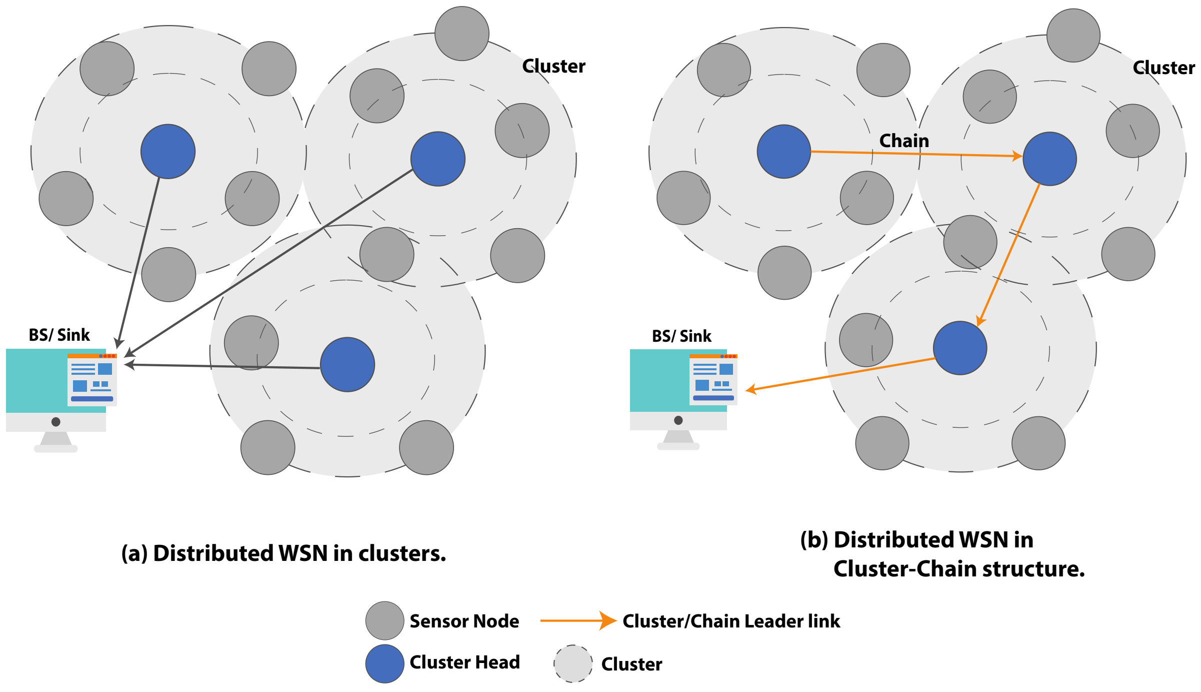
Figure 9 shows two cluster architectures and how sensors and CHs organize. Figure 9(a) presents a network in a normal cluster organization, where CHs communicate directly to the BS. In this approach, if the BS is far from the nodes that are acting as CHs at the moment, communication will require a high amount of energy, which will reduce network lifetime. An alternative is presented in Figure 9(b), which is a Cluster-Chain structure, where CHs form a Chain, communicating with each other in order to reach the BS. In distributed WSNs, there must be an algorithm or method to select the CHs dynamically, the authors in (MAMUN, 2012) define the following approaches for CH selection and cluster formation.

a) **Probabilistic Method:** Sensors are selected as CHs at a given time, based on a certain probability. Once Cluster Heads are elected, they broadcast their status to the rest of the network, that way, sensor nodes can select the cluster to which they must belong based on the energy required for communication (HEINZELMAN et al., 2000). From that moment forward, the CH acts as the coordinator of its corresponding cluster and is in charge of directing information to the BS. One of the most popular algorithms of this type is Low-Energy Adaptive Clustering Hierarchy (LEACH) proposed in (HEINZELMAN et al., 2000), which is a self-organizing, adaptive clustering protocol that uses randomization to distribute the energy load evenly among the sensors in the network.

b) **By Election Phase:** All sensors broadcast their information to the rest of the nodes in the network, based on this information they form a cluster and select a CH. Classic LEACH protocols ignore the residual energy of node, node density, and other factors. CH Election Phase introduces the node density factor and the residual energy factor, achieving a better energy balance (LI, H. et al., 2013).

c) **Assigned by the Base Station:** Clusters are assigned by the BS. After deployment, all sensor nodes communicate with the BS which applies residual energy and predetermined energy threshold as a criterion to determine the optimal cluster formations (GEETHA et al., 2012). The BS uses balanced clustering technique, with an approximately equal number of nodes for distribution of the load of the CHs, and iterative cluster splitting algorithm to get the ideal number of clusters, then creates several CH-to-CH paths for communication (PADMANABHAN; KAMALAKKANNAN, 2012). An example is the Base station Controlled Dynamic Clustering Protocol (BCDCP).

Figure 9 – Distributed WSN in a cluster organization.

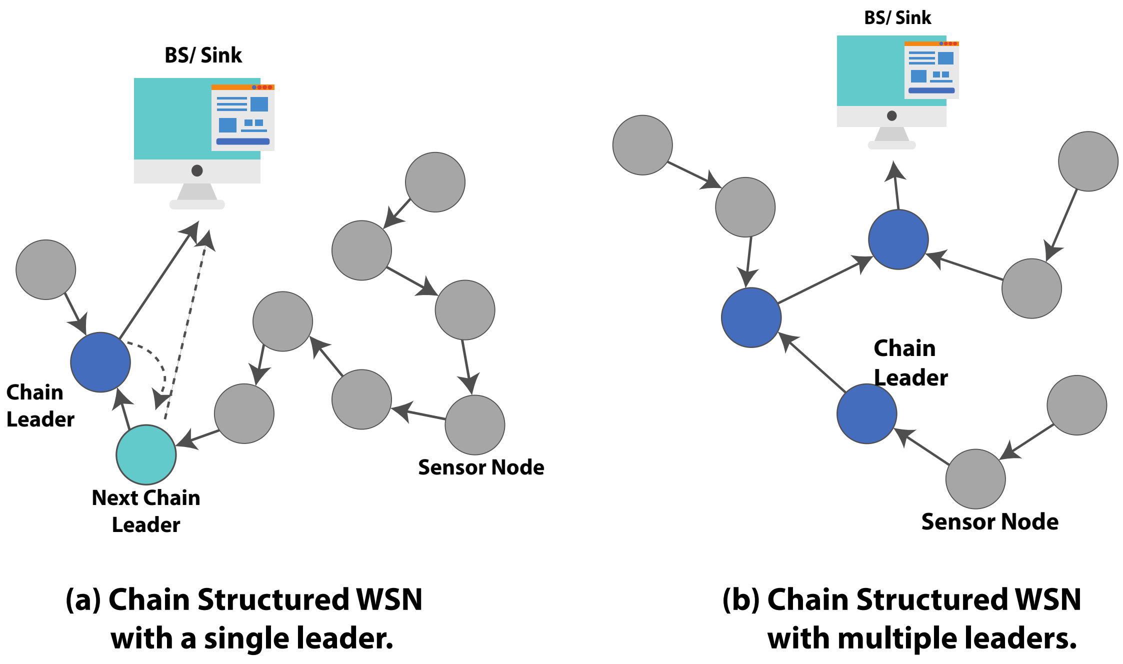

Cluster
BS/ Sink
(a) Distributed WSN in clusters.
Chain
Cluster
BS/ Sink
(b) Distributed WSN in Cluster-Chain structure.
Sensor Node
Cluster Head
Cluster/Chain Leader link
Cluster

Source – Author.

#### 2.3.2.3 Chain-Based

An alternative to clusters is transmission chains. In this setup, as in clusters, a node is selected in the chain to act as a leader. All nodes communicate through the chain, sending data to the next node, which is called the successor node, towards a leader node. The chains are usually formed considering minimal distances between nodes. The leader node is rotated through the chain also, this will distribute the energy load evenly among the sensor nodes in the network. All sensors send their data to the selected leader this way. This facilitates data aggregation(MAMUN, 2012). Power-efficient gathering in sensor information systems (PEGASIS) protocol, described in (LINDSEY; RAGHAVENDRA, 2002), is one of the firsts protocols that used chains. According to the authors, the PEGASIS protocol exceeded the lifetime of the network twice as much as LEACH. An improvement to this approach was proposed in (HADJILA et al., 2013). In addition to forming multiple chains they construct the main chain grouping the leader nodes, one node of the main chain collects, aggregate and transmit the data to the BS, improving energy consumption. Figure 10 presents two examples of chain-based WSNs, with single and multiple chain leaders, notice that the leader nodes rotate in the network to balance energy.

Figure 10 – Chain Based WSN with single and multiple chain leaders.


BS/ Sink
Chain Leader
Next Chain Leader
Sensor Node
(a) Chain Structured WSN with a single leader.
BS/ Sink
Chain Leader
Sensor Node
(b) Chain Structured WSN with multiple leaders.

Source – Author.

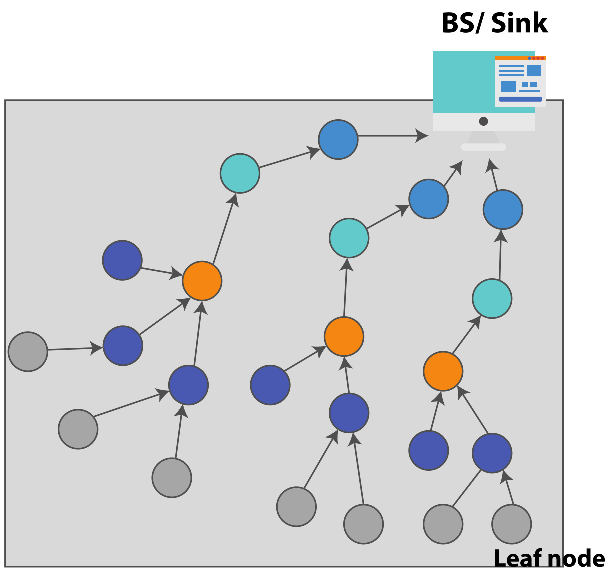
### 2.3.2.4 Tree-Based

In this architecture, the deployed network creates a logical tree. Data is passed from leaves to parent nodes. In turn, a receiver node receiving data from the child node sends data to the receiver's parent node after aggregating data with its own data, that way, data flows from lead nodes to the root, which is the BS, see Figure 11. Logical Trees avoids the use of Flooding and broadcast (MAMUN, 2012). An example of Tree-Based WSN is encountered in (GETACHEW; CHUNG, 2012), where authors proposed a modification of the Tree-Based routing, named Information Selection Branch Grow (ISBG) which aims to optimize the network in achieving a higher network lifetime.

## 2.4 APPLICATIONS

Almost any applications can be classify into Event Detection (ED) and Spatial Process Estimation (SPE) (BURATTI et al., 2009). In the first case, sensors are deployed to detect specific events or anomalies, such as fires, floods, earthquakes, etc. In this type of applications, must be ensured that the number of sensors is sufficient to detect the event and transmit it to the BS. In the second case, the goal is to estimate the behavior of a spatial process based on samples collected by the sensor nodes. As for the domain of the application, five fields can be identified, these are Military, Industrial,

Figure 11 – Tree-Based WSN.



Source – Author.

Healthcare, Environmental, Home applications, Agriculture and Infrastructure.

a) **Military:** The flexibility and the capacity of self-organization that WSNs provide makes them able to adapt to many requirements. For instance, in battlefield operations, large scale WSNs consisting of many thousands of nodes, which are non-manually deployed are often used (KANDRIS et al., 2020). An example of a military WSNs is described in (KRISHNAMURTHY et al., 2006), where authors present VigilNet, which is a military WSN to collect and verify information about the enemy capabilities and position of targets. Another example is in (LIM et al., 2010), where authors described a Body Sensor Networks (BSN) for real time monitoring of soldiers state.

b) **Healthcare:** In the healthcare system, WSNs are used to monitor patients in hospitals or in their homes, providing real time monitoring of their health parameters or as an ED system, in the form of wearable devices in most cases. An example of this is in (KIM et al., 2013), where authors proposed a wearable health monitoring system to collect data from an electrocardiogram, and in (KAKRIA et al., 2015), which presents a remote patient monitoring system in the form of a wearable belt.

c) **Industrial:** In Structural Health Monitoring applications to monitor the seismic and/or man-made acceleration in buildings (PENTARIS et al., 2014). In (DENG et al., 2017) for the monitoring of High Voltage Direct Current (HVDC) transmission lines. To monitor toxic and combustible gases in the Oil and Gas

Industry (ALIYU et al., 2015).

d) **Environmental:** They have been used widely for environmental monitoring of remote, hostile areas and ED of natural phenomenon. In (KHEDO et al., 2020) the authors used a WSN for monitoring seismic activity in the island of Mauritius, which can be extended for tsunami and volcanic activity monitoring. In (LOPES PEREIRA et al., 2014) is described the design of a WSN capable of collecting geophysical measurements on remote active volcanoes, and in (MALHOTRA; VIRMANI, 2017) a Tsunami detection system for a distributed WSN in water bodies.

e) **Home:** WSNs are very practical for the creation of smart environments because they allow the interconnection and data collection from devices in a very easy form. An example of a smart home application is (GHAYVAT et al., 2015), where authors develop a reliable, efficient, economical, real-time WSN for a smart home system.

## 2.5 CONCLUSIONS

Sensor nodes are simple, in most cases, battery-powered devices, provided with sensors to monitor the environment, a processing unit, the capacity to communicate wirelessly with other nodes, and other components depending on the application. WSNs can be organized in a centralized or a distributed architecture. Centralized architectures are more fixed and concentrate the decision-making process in a central node, contrary to distributed networks, therefore it is more common to see the first one in applications with less number of sensors and previous knowledge of the terrain. Also, WSNs are used in many applications, being the most common fields the Military, Healthcare, Industry, Home, and Environmental.

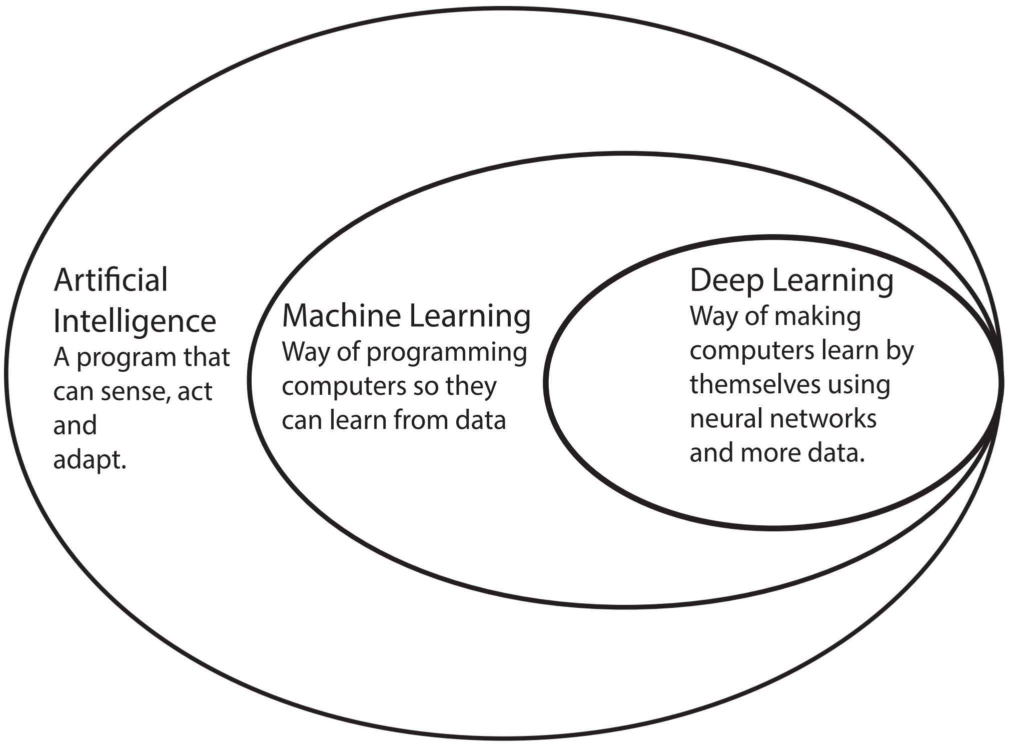
# 3 PREDICTION METHODS FOR ENERGY REDUCTION

This chapter introduces the problem of Multivariate Data Prediction for data reduction and energy saving in WSNs. It is also presented an introduction to Machine Learning for time series analysis and general time series forecasting methods, the problem of Data Prediction in WSNs with the different prediction schemes, together with a review of related works.

## 3.1 MACHINE LEARNING AND DEEP LEARNING CONCEPTS

The terms Artificial Intelligence (AI), Deep Learning and Machine Learning are indistinctly used, but they are not the same. Machine Learning is a subset of AI and Deep Learning is a subset of Machine Learning, as Figure 12, these two are the tools that made possible the current advances in AI. Artificial Intelligence is the branch of computer science that allows computers to perform tasks that normally require human intelligence.

Figure 12 – Subsets of Artificial Intelligence.


Artificial Intelligence
A program that can sense, act and adapt.
Machine Learning
Way of programming computers so they can learn from data
Deep Learning
Way of making computers learn by themselves using neural networks and more data.

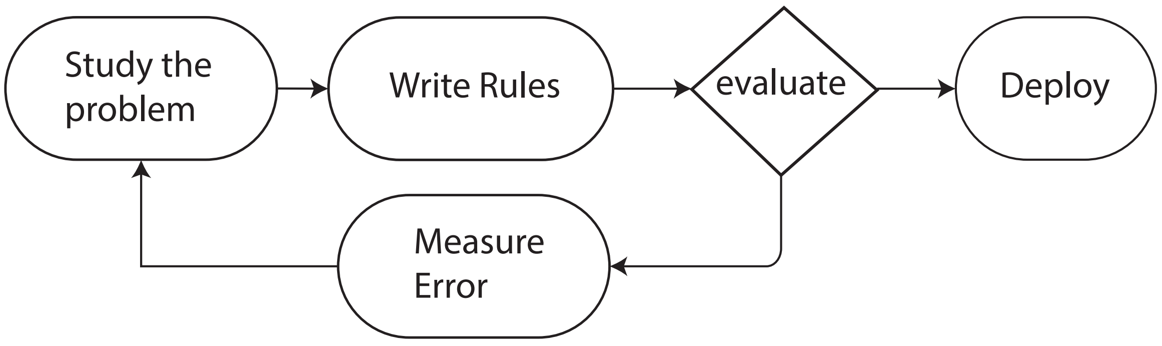
Source – Author

### 3.1.1 Machine Learning

Machine Learning is the science of programming computers to make them learn from experience (GÉRON, 2019). According to (MITCHELL, 1997), a computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E.

For example, let us say we want to predict if a person is jogging, walking, sitting, or running based on acceleration data coming from your smartphone. Using conventional programming you would have to define many rules to determine if a person is doing one of the activities, and even so, different persons can run faster or slower, the person can be smaller or higher or move differently when walking. There are many factors to consider, therefore we would probably end up with a long list of rules and a complicated code, this is the approach to problem solving in conventional programming (see Figure 13), which consist of taking the input data and define rules (if, then, else) to obtain answers.

Figure 13 – Traditional programming approach.

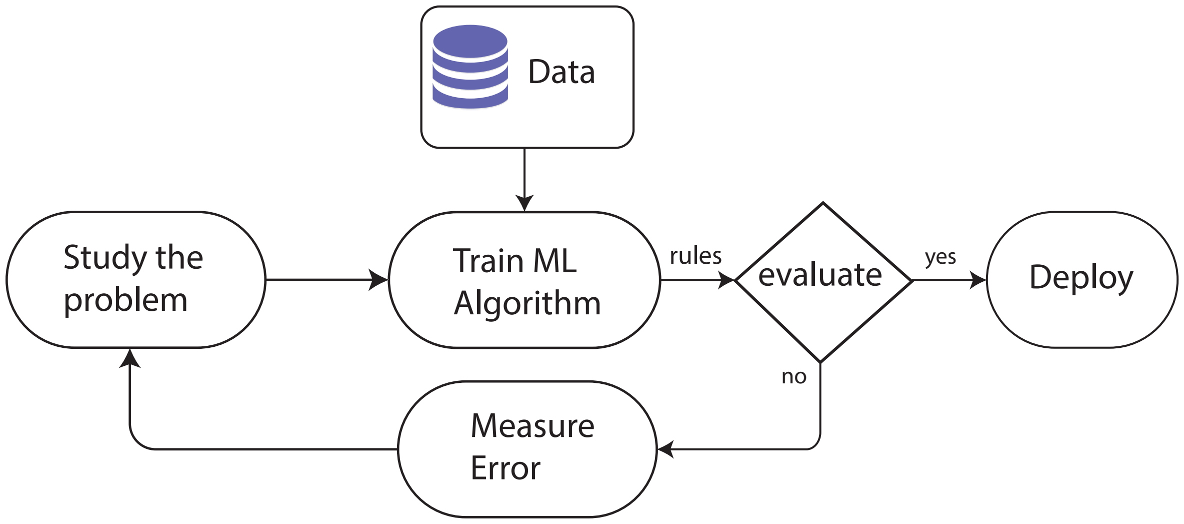


Source – Author

In Machine Learning we take data and define the desired answers for that input data to obtain rules (see Figure 14 ). To solve the previous situation using Machine Learning, we can extract a many examples of acceleration data from our phone and label this data according to the corresponding activity (running, jogging, walking, etc), then the Machine Learning algorithm figures out the specific patterns in the data that determines what makes one activity different than the other. The main difference is that the algorithm learns by itself how to identify the activities through continuous exposure to the data.

The process of continuous exposure to the data is called *training*; each input example extracted from the phone to form the dataset is called *training example*; the data used for training is called *training set*; the data used to validate the results of the

Figure 14 – Machine Learning programming approach.



Source – Author

algorithm is the *testing set* or *validation set*, and the amount of times the algorithm is exposed to the training data is called *epochs*.

This new paradigm creates situations that were not feasible to do using conventional programming. An example is the classification of images, it would be very difficult to create a program, using conventional programming, capable of identifying if there is a cat or a dog in an image, or what breed of dog and cat is present, but using Machine Learning this becomes a really easy task.

#### 3.1.1.1 Types of Machine Learning Systems

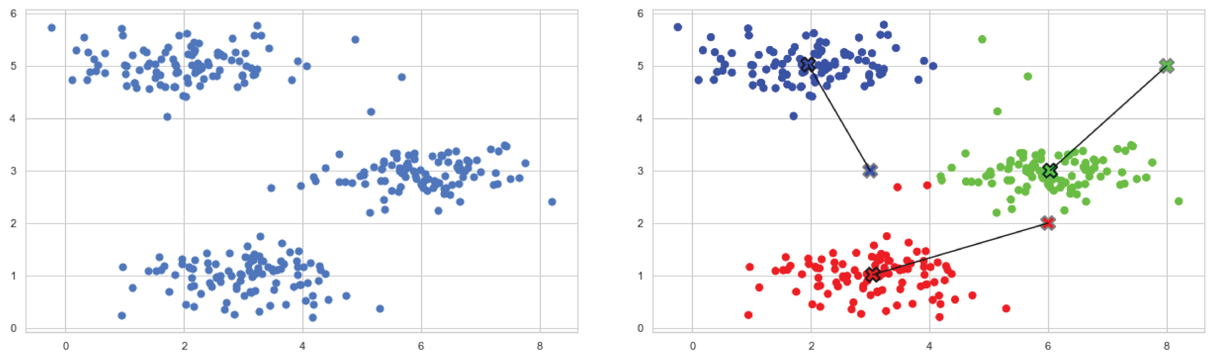
Machine Learning algorithms can be classified according to how they learn from the input data. They can be classified into supervised, unsupervised, semi-supervised and reinforcement learning techniques (GADEPALLY et al., 2019).

**Supervised Learning:** In supervised learning we provide the algorithm with the data and the correct answer for every example in the data, called *labels*. The previous situation (jogging, walking, in section 3.1.1) can be an example of a supervised learning problem if we provide to the algorithm a label for each data example, indicating how the data looks like for each different action, this is an example of a *classification* problem. Another typical problem is to predict a target numerical value, such as the price of a house the next year given a set of *features* (current price, square area, city, etc.) called *predictors*. This is a *regression problem*. To train this type of algorithms we need to provide many *training examples*, including the predictors and the corresponding label (prices). Some of the most common supervised machine learning algorithms are k-

Nearest Neighbors; Linear Regression; Logistic Regression; Support Vector Machines (SVM), Decision; Decision Trees and Random Forests, and Neural Networks (GÉRON, 2019).

**Unsupervised Learning:** In unsupervised learning we do not have or we do not know the labels, therefore the algorithm will try to learn by itself the patterns in the data. We can use this type of algorithms to separate classes in a group *(Clustering)* and to understand better our data. We can see a example of clustering in Figure 15. We started with unlabeled data and then used an unsupervised ML algorithm to cluster the data into the number of classes we suspect (GOODFELLOW et al., 2016).

Figure 15 – (Left) Unlabeled data. (Right) Clustered data in three classes using K-Means algorithm.



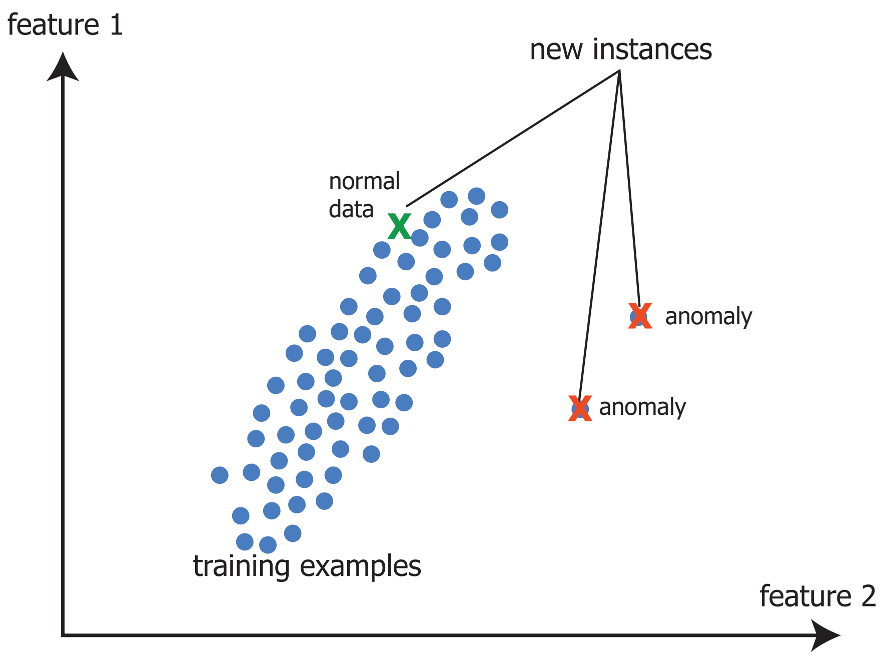
Source – Author

Another important use of unsupervised learning is anomaly detection. Sometimes we need to clean up our data before using it in another algorithm. Machine learning algorithms are very good at detecting patterns in data, they can do it by learning the normal behaviour of the data through continuously exposure to the training examples. After the training is performed, it is expected that the algorithm can remember the normal behaviours in the data, and have more difficulty when identifying points that are abnormal, this is because there will be very few examples of abnormal data, therefore it will be harder to the algorithm to reproduce or remember these points, an example of how an abnormal data would look like is in Figure 16.

**Semi-supervised learning:** These are algorithms that can handle partially labeled training data. Semi-supervised problems include some labeled data and mostly much more data with unknown labels, the goal is to classify some of the unlabeled data using the labeled information. Some photo-hosting services, such as Google Photos, are good examples of this (GÉRON, 2019).

**Reinforcement Learning:** In this type, the learning system, now called the agent, interacts with the environment and receives rewards or penalties. The system learns by itself the best strategy to obtain the greatest amount of rewards possible.

Figure 16 – Anomaly detection example.



Source – Author

These are common learning algorithms used by robots to learn to move in an environment, stand and walk. Every time the robot performs an action correctly receives a virtual reward, and every time it fails receives a penalty (falls, crash, etc.)

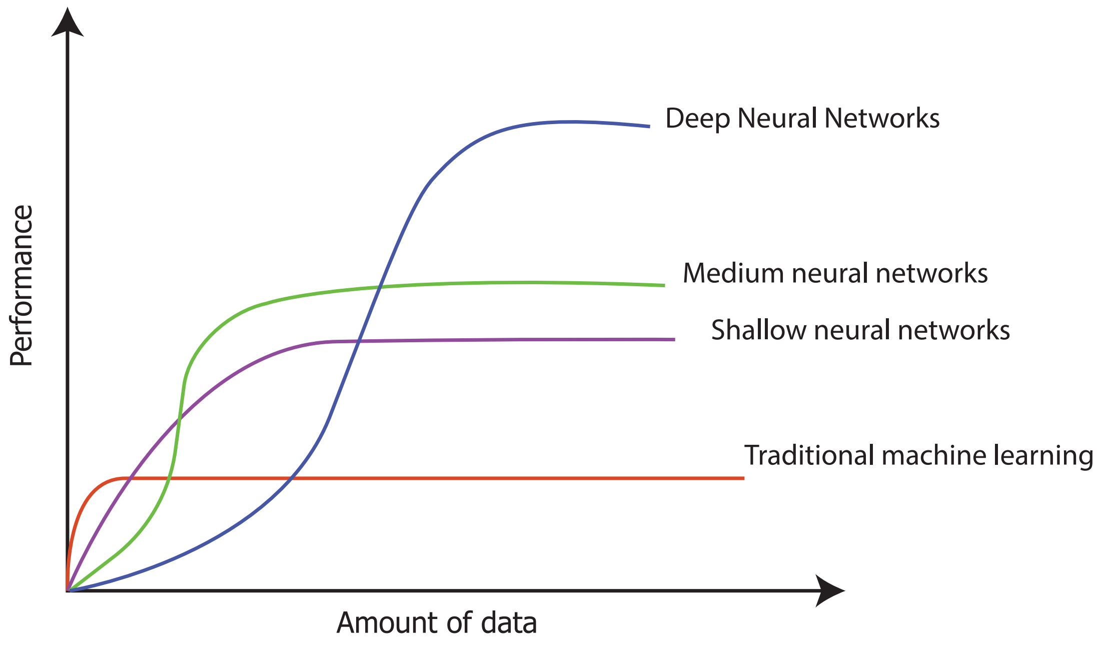
### 3.1.2 Deep Learning

Deep Learning is a subset of Machine Learning that uses algorithms based on Artificial Neural Networks. The difference with ML algorithms is that we can build long and deep neural networks adding more and more data and the performance will increase. With traditional ML algorithms the performance will reach a saturation point (see Figure 17), after that point it does not matter if the amount of data or the complexity of the algorithm increases, in contrast to deep neural networks, where performance usually increases.

The basic unit of a neural network is the neuron, which imitate biological neurons of the human brain. The neurons take an input and performs simple operations on it. The result of these operations is passed to other neurons. Whether the result will be passed, is determined by the activation function. ANN can be of many types, some of these are feed-forward neural network, radial basis function neural network, Kohonen self-organizing neural network, recurrent neural network, convolutional neural network and modular neural network (DAS, H. S.; ROY, 2019).

Neurons are grouped into layers, the depth of the network is defined by the num-

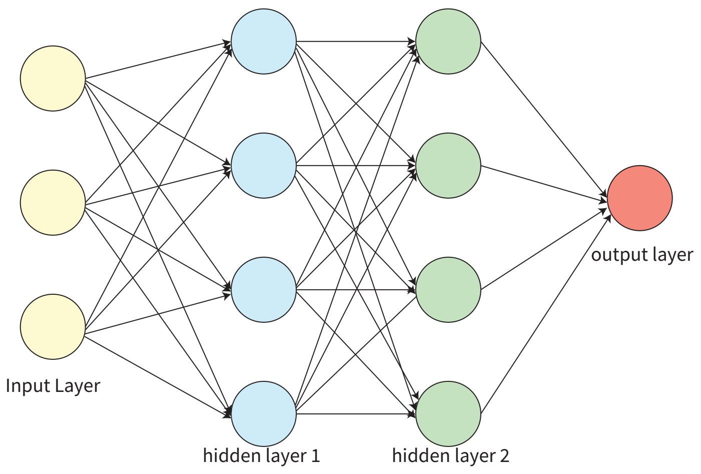
Figure 17 – Amount of data Vs performance graph.



Source – Author

ber of layers. A neural network is formed by three types of layers: An input layer, which is the initial data for the neural network; the hidden layers: which is the intermediate layer between the inputs and the output layer and where the transformation to the input data is made, and the output layer, which produces the output, as is represented in Figure 18.

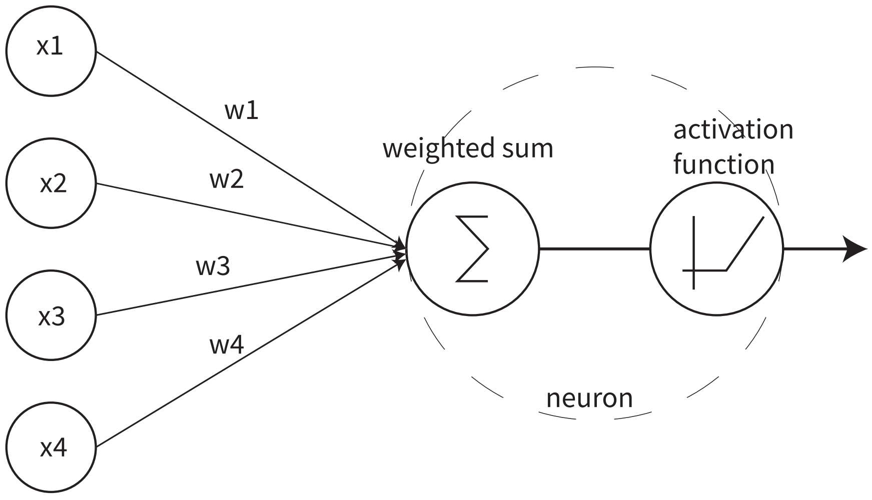
Figure 18 – Neural networks architecture.



Source – Author

The nodes from one layer are connected with the nodes of the subsequent layer through weights, these weights represent the impact that the previous node has in the next, we can see the neuron structure and the communication with the previous layer in Figure 19. The neuron performs a weighted sum of the inputs and pass it to an activation function which defines if the neuron will be activated, and if so, how active it will be in producing the output.

Figure 19 – Neural network node.



Source – Author

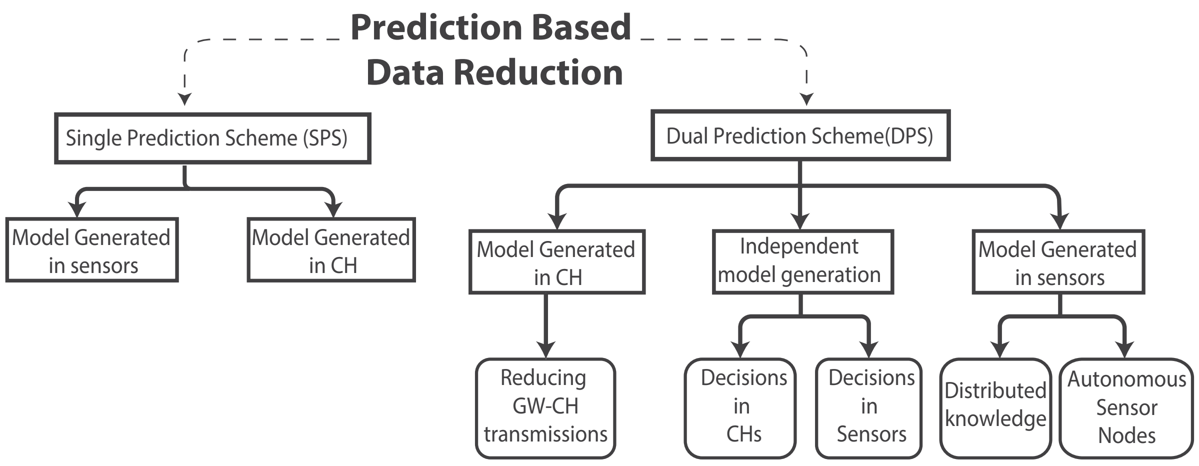
The goal of the network is to learn a function that can approximate as much as possible the behaviour of the training data (labeled data). The error to be minimized is given by a function called *Cost function* $J(\Theta)$, which represents the sum of the errors during training. This error is minimized during training by updating the weights of the neurons in a process called *backpropagation* (GOODFELLOW et al., 2016). The weights will be randomized in the beginning, before the learning process starts. When training starts, the weights are adjusted toward the desired values and the right output. The weights can be seen as the strength of the connection. Low weights will have no impact in the output, and larger weights will affect more significantly the output. There are many activation functions, one of the most important is the Sigmoid function, which is part of many models and some of the ones presented in section 3.3.4. The Sigmoid activation function is given by equation (1). This is a non linear function that allows the nodes to take any values between 0 and 1, therefore, nodes that have lower impact in the answer will output values closer to zero, and values with high impact will output values closer to 1, creating a probability distribution between layers.

$$S(x) = \frac{1}{1 + e^{-x}} = \frac{e^x}{e^x + 1} \tag{1}$$

## 3.2 DATA PREDICTION FOR ENERGY CONSERVATION IN WSN

WSNs are a good choice because of the low cost, low energy consumption, and large scale deployment capability. However, these sensor networks are normally battery-powered and sometimes their location makes the replacement of batteries a very difficult task, therefore energy conservation is one of the most important challenges of WSN. In a WSN approximately 80% of energy is consumed in transmission and the rest in sensing and processing (ANASTASI et al., 2009). Data prediction stands out as a possible solution to this problem by predicting part of the sensed data without triggering transmission, thereby saving energy and reducing congestion (DIAS et al., 2016; SHARMA et al., 2014; SONG, Y. et al., 2015; KRISHNA et al., 2018). Figure 20 presents a classification of architectures used for prediction based data reduction in WSNs, provided by (DIAS et al., 2016), where the author divides the possible architectures in Single Prediction Scheme (SPS) and Dual Prediction Scheme (DPS).

Figure 20 – Taxonomy of the architectures that use predictions for data reduction.


Prediction Based Data Reduction
Single Prediction Scheme (SPS)
Dual Prediction Scheme(DPS)
Model Generated in sensors
Model Generated in CH
Model Generated in CH
Independent model generation
Model Generated in sensors
Reducing GW-CH transmissions
Decisions in CHs
Decisions in Sensors
Distributed knowledge
Autonomous Sensor Nodes

Source – Adapted from (DIAS et al., 2016)

### 3.2.1 Single Prediction Scheme

Predictions are made in one location of the network, which can be either in the sensor, BS or CH. The CH or BS can forecast the data that was received from the sensor node and decide when to forecast more points based on the reliability of the current predictions (LIAZID et al., 2019). This architecture can be used in applications where sensors are close and present a well defined spatio-temporal correlation. The correlation can be used to generate a model that predicts the data and check if the results are inside a confidence interval (DIAS et al., 2016). An example of this can be found in (PACHARANEY; GUPTA, 2019), where the authors proposed a BS assisted cluster, spatially correlated with the rest of the nodes, that senses the data and uses compressive sensing to reduce the number of transmissions. Another example is in (TAYEH et al., 2019) where the authors proposed another correlation based approach for cluster based networks. In this case, the CH computes the correlation between the measured data and adjusts the sampling rate of the sensors accordingly; then a prediction algorithm is run in the BS to reconstruct the missing points.

Another alternative is to make predictions in the sensor node. This case can be useful in situations where it is more expensive to obtain a sample than to predict one. Adaptive sampling is one of the techniques used in this approach (LÓPEZ-ARDAO et al., 2021) to reduce the number of samples taken by the sensor. For example, in (ANASTASI et al., 2009) the authors proposed an adaptive sampling algorithm that estimates the best sampling frequency for the sensors dynamically. Another use of this approach is for object tracking WSNs. An example can be found in (SAMARAH et al., 2011), where the authors proposed a prediction-based tracking technique for object tracking WSNs. Sensors predict the future movements of the objects keeping the sensors outside their range in sleep mode.

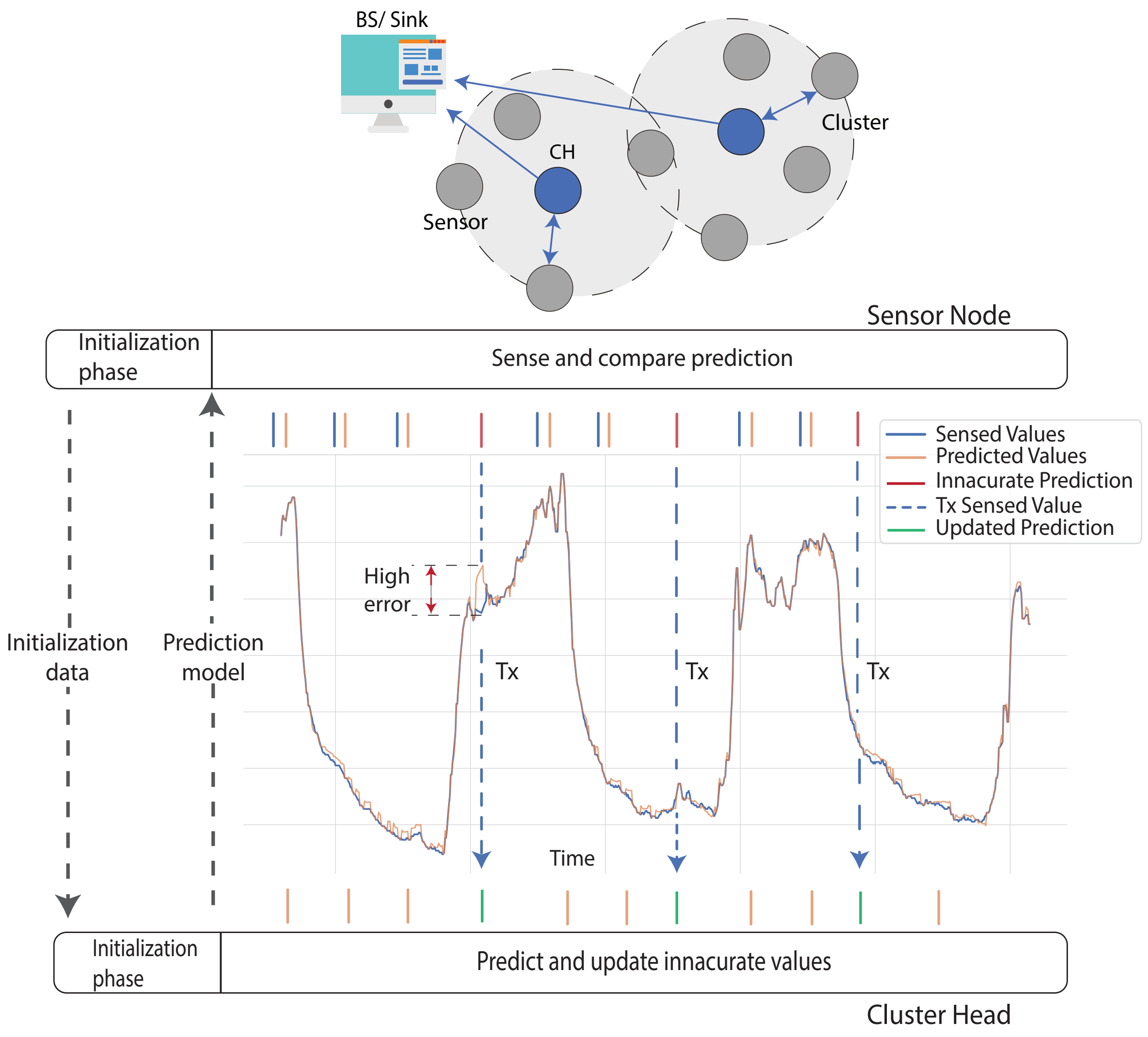
### 3.2.2 Dual Prediction Scheme

In a DPS predictions are made in both sensor nodes and CHs or gateway. Both devices are able to produce the same results, since the prediction model is shared between them, but sensors can check the accuracy of the predictions by comparing with the measured data and avoid an unnecessary transmission. The sensor node is constantly comparing the current observation with the predicted value and only when the difference falls outside a specified threshold, it transmits the measured value to the CH, which will substitute the prediction with the real value (DIAS et al., 2016; LÓPEZ-ARDAO et al., 2021; SHU et al., 2019). Then the substituted value will be used in the next prediction. Thus, sensors can reduce transmission and reduce energy consumption. Figure 21 shows the functioning of this scheme and the different strategies of model generation.

#### 3.2.2.1 Model Generated in CHs

Since CHs usually have more computational and energy resources, generating the model there makes more sense. Based on the data received from the sensors, CHs can create a model to generate new predictions for every sensor node and be in charge of updating and transmitting this model to the sensor. An example of this scheme is in (FATHY et al., 2018), where authors proposed an Adaptive Method for Data Reduction in a DPS using an Independent Model Generation.

Figure 21 – Dual prediction scheme functioning in a Wireless Sensor Network.



Source – Author.

### 3.2.2.2 Independent Model Generation

In an independent model generation, the prediction model is generated independently in both sensors and CHs, as is presented in Figure 21 for this scheme. There is an initialization phase where sensors transmit all the data to the CH to ensures that the CH has all the information possible from the environment before generating the model. After this, they both are capable of generating the same predictions, in this case, CHs do not need to do an extra transmission. Sensor nodes can check the accuracy of the prediction by comparing it with the measured value and transmit it to the CH if there is a need to overwrite the prediction. As a drawback, the variety of the models is restricted by the computing power limitations of sensor nodes (DIAS et al., 2017).

#### 3.2.2.3 Model Generated in Sensor Nodes

In this scenario, sensor nodes start transmitting the data to the CHs and they also have to produce the prediction model. This scheme requires much more computing power from the sensor node since is the one producing and transmitting the model. This method allows sensors a certain degree of autonomy because they can decide by themselves if they can stop making measurements for a period, based on the predictions. An example was presented for (SHEN; LI, X., 2008), where authors developed a model generated in sensor nodes, in which sensors change their internal status based on predictions and the amount of energy they will save.

### 3.2.3 Prediction Based Data Reduction Related Works

Many researches have addressed the problem of prediction in WSN. A single prediction scheme based on an Artificial Neural Network (ANN) and a request management algorithm was proposed (ABBOUD et al., 2016), obtaining a mean squared error of 0.2 after training 50 hidden layers for 1000 epochs, achieving a reduction of 88% of the transmitted data

In (BEN ARBI et al., 2017), the authors used a Self-Exciting Threshold Autoregressive (SETAR) to reduce energy consumption during electric engine vibration monitoring, obtaining an energy saving of approximately 73%. Also, an Autoregressive Model was proposed in (KRISHNA et al., 2018), saving up to 85% by applying a technique with a specific threshold.

Another prediction model based on Bidirectional Long Short-Term Memory Neural Network (LSTM)s with multiple features was proposed in (CHENG et al., 2019b). The model considers spatial correlation by combining the data from two sensor nodes in a final Fully Connected Layer (FCL). The architecture was compared with Elman Neural Network which is a typical local forward network, Nonlinear Autoregressive Exogenous Model (NARX), and General Regression Neural Network (GRNN) presenting much better results in terms of prediction accuracy and concluding that spatial-temporal correlation can improve the model performance.

The same author in (CHENG et al., 2019a) also considers spatial-temporal correlation by combining data from three sensor nodes, also in a final FCL. The model uses a combination of One Dimensional Convolutional Neural Networks (1D-CNN), to extract high level time independent features, and Bidirectional LSTMs for temporal correlation and time dependent features such as trends and seasonality. In this case, was used an End-to-End Deep Learning (DL) approach. The performance was compared with a Convolutional Neural Networks (CNN), a Bidirectional LSTM and a Gated Recurrent Unit Network (GRU), presenting a smaller error in prediction. The author concludes that the model can make a stable and accurate prediction on temperature and humidity data in the short and medium-term.

In (SINHA; LOBIYAL, 2015) authors proposed a model to reduce redundant transmission in a WSN, and an energy prediction-based cluster head rotation algorithm for load balancing within clusters. Another example is in (DAS, R. et al., 2018) where authors used a lightweight Bayesian Estimator in sensor nodes in a temporal manner. At the base station, they used Naive Bayes, Support Vector Machine (SVM) and Multi-layer Perceptron (MLP) based inference systems to generate decisions based on the estimated data send by the sensor nodes.

In (ALVES et al., 2017) the authors proposed a damage prediction system for wind turbines, named Delphos, based on WSN and actuator network. The system predicts the state of a turbine to know when it is going to be damaged and take action before it happens. It is based on the Auto Regressive Integrated Moving Average (ARIMA) method to forecast the state of the turbine and a fuzzy system to filter the effect of environmental temperature in prediction. The authors concluded that when removing the effect of temperature, predictions of damage are very accurate.

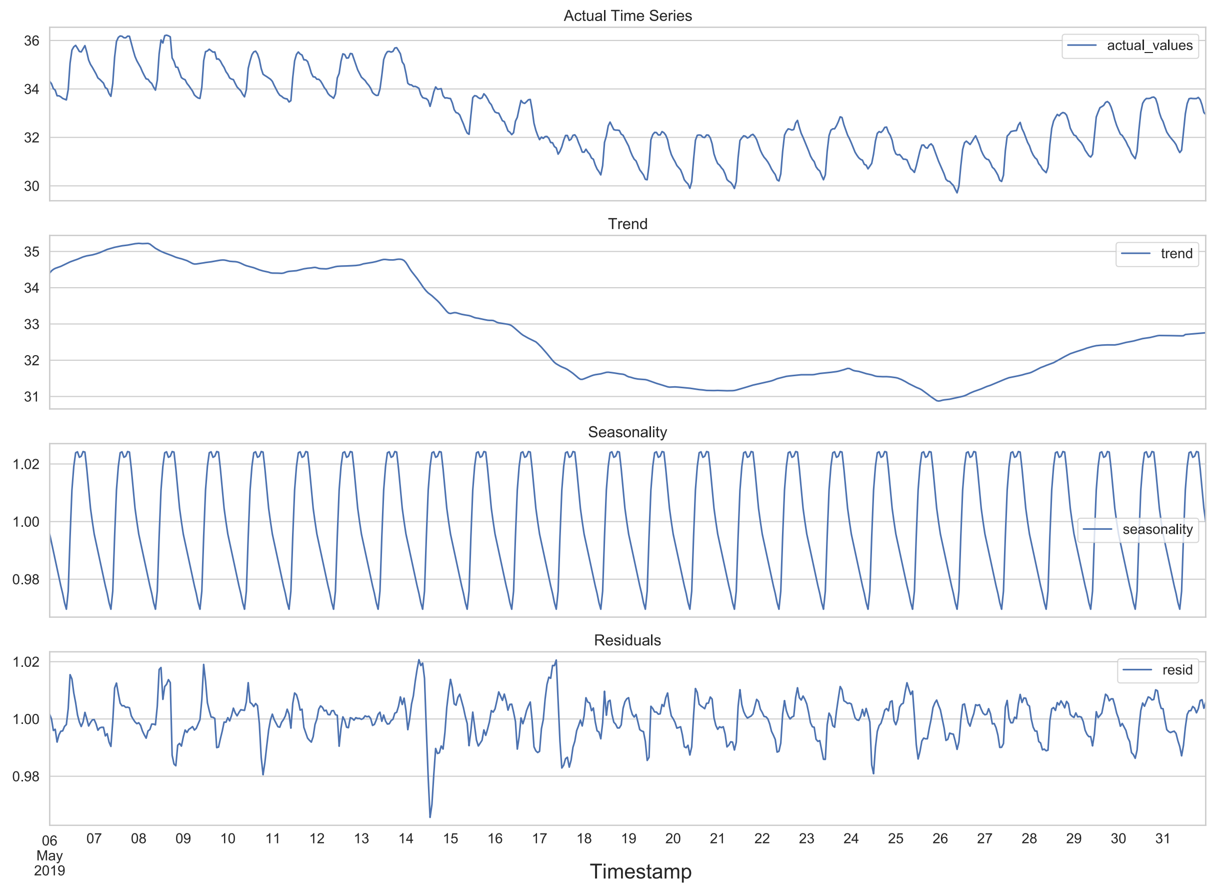
## 3.3 TIME SERIES FORECASTING

A time series is an ordered collection of observations recorded at a specific time period. Depending on the period of observations, a time series period may typically be hourly, daily, monthly, etc. In the case of data collected by sensors, the sampling period is generally much smaller, in the range of minutes and seconds. Time series can be univariate or multivariate. A univariate time series consists of a single observation for each point in time, and a multivariate time series is when we have a set of univariate time series describing the same range of time, therefore multiple measures for the same point in time. A sensor that measures temperature, humidity, and voltage produces a set of observations for the same point in time, these measures placed together are an example of a multivariate time series. Multivariate time series provides more information about the studied process and more accuracy in prediction tasks, but most state-of-the-art methods can only forecast/predict one variable while taking the other information into account (BAUER et al., 2020).

### 3.3.1 Components of a Time Series

Time series can be decomposed into a trend, seasonality, and irregular components or residuals. The trend is the long term development of the time series in one direction. It can be seen in the slope and it can be increasing, decreasing, or with no trend. A time series can present different trends in different periods of time. Whereas the seasonality is a distinctive pattern that is repeated in regular intervals as the result of seasonal patterns. The unpredictable part of a time series is the irregular component possibly following a certain statistical noise distribution. It is also considered as the residual time series after all other components have been removed (BAUER et al., 2020). Another aspect to consider in time series is the cyclic behavior, this is when there are rise and fall patterns without a fixed frequency and the amplitude and duration vary over time.

Figure 22 – Time Series Decomposition into Trend, Seasonality and Residuals.



Source – Author.

Figure 22 presents an example of a time series decomposition into its trend, seasonality, and residuals. The time series corresponds to temperature values read by one sensor node. We can see a downward trend in the time series, which corresponds to a smoother version of the actual one, we can see there is a seasonal component corresponding to the daily changes of the temperature, rises passed morning until night, and then falls, and finally, the residuals, which is the noisy factor of the time series.

### 3.3.2 Forecasting Time Series

A forecast is considered a prediction on the basis of time series data. Is an estimation of the future, based on historical data and considering the time dimension.

A forecast can be of one-step or multiple-steps-ahead. In the first case, only the next value is forecast, and in the second case, multiple values are forecast at the same time. The amount of values to be forecast is called *forecasting horizon*.

Most statistical forecasting methods are designed to function on stationary time series. A time series is stationary when statistical properties such as mean and variance are not a function of time. This reduces considerably the complexity of the models. Time series with trend and seasonal patterns are usually non-stationary by nature (ADHIKARI; AGRAWAL, 2013). In such cases, in order to apply statistical forecasting methods, first is needed to transform the non-stationary time series to stationary by removing the trend and seasonal components.

### 3.3.3 Forecasting Methods

There are many methods for time series forecasting. We can divide them into Statistical Methods and Machine Learning Methods.

#### 3.3.3.1 Statistical Methods

a) Naive and sNaive: this is the most basic forecasting method and a great baseline (BAUER et al., 2020). sNaive, defined in equation (2), is similar to the Naive method, but considering the observation from one period before. Both Naive and sNaive do not provide a model, only an instantaneous forecast point.

$$y_t = y_{t-period} \tag{2}$$

Where $y_t$ is the actual observation and $y_{t-period}$ is a point of a time period before.

b) Moving Average: this method, defined by the equation (3), takes the average of the time series over a fixed period $n$, called an *average window*. $X_t$ is the actual observation and $\hat{y}_t$ the prediction. The method reduces the noise, but does not anticipate trend or seasonality.

$$\hat{y}_t = \frac{X_t + X_{t-1} + ... + X_{t-n+1}}{n} = \frac{1}{n} \sum_{i=t-n+1}^{t} X_i \tag{3}$$

c) Exponential Smoothing (ETS): while in a Moving Average, all observations have the same weight, in this method, all observations are weighted in an exponentially decreasing form as we move back in time. Is given by the following formula:

$$\hat{y}(t) = \alpha \cdot y_t + (1 - \alpha) \cdot \hat{y}_{t-1} \tag{4}$$

The model is a weighted average between the most recent observation $y_t$ and the most recent forecast $\hat{y}_{t-1}$. $\alpha$ is a smoothing factor, which defines how fast the model forgets the last available observation. The less $\alpha$ the smoother the result is, and in the case of $\alpha = 1$ the result is the current observation. There is no formally correct procedure for choosing $\alpha$ but it can be optimized.

d) Double Exponential Smoothing (DETS): ETS does not perform well when the time series presents a trend. This method is a solution for such cases. The time series is divided into two functions, the first one describes the intercept which depends on the current observation, and the second function describes the trend, which depends on the intercept's latest change and the previous values of the trend.

$$l_x = \alpha y_x + (1 - \alpha)(l_{x-1} + b_{x-1}) \tag{5}$$

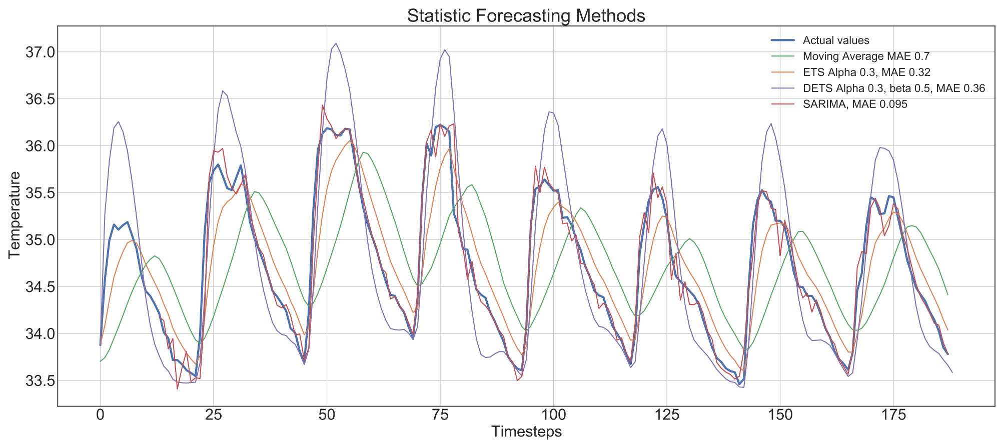
$$b_x = \beta(l_x - l_{x-1}) + (1 - \beta)(b_{x-1}) \tag{6}$$

$$\hat{y}_{x+1} = l_x + b_x \tag{7}$$

e) Seasonal Autoregressive Intergated Moving Average (SARIMA): the model is a combination of the Integrated ARMA model (ARIMA) and a seasonal component. The AR(p) part stands for Autoregressive, a regression of the time series onto itself, where $p$ is the maximum lag or number of past values. The MA(q) stands for Moving Average and models the time series assuming that this error depends on the previous error with some lag, defined as $q$. The $I(d)$ is the order of integration, which represents the number of nonseasonal differences needed to transform the sequence to stationary. Finally, the letter $S(s)$ completes the SARIMA model and it represents the seasonality present in the sequence, providing the model with the ability to model seasonal components. Although this method can provide an accurate forecast the selection and identification of the model parameters are very time-consuming. That is, the runtime may be quite high and unpredictable (BAUER et al., 2020).

Figure 23 presents a comparison of the statistical methods in the temperature time series. Predictions closer to the real values are achieved with the SARIMA model, but this is only achievable after many iterations searching for the best parameters. ETS and DETS are good at recognizing sinusoidal patterns, therefor they can approximate the real values with a reasonable accuracy, in the case of DETS a better tuning of the $\alpha$ and $\beta$ parameters is needed.

Figure 23 – Forecasts comparison of the statistical methods.



Source – Author.

### 3.3.3.2 Machine Learning Methods

a) eXtreme Gradient Boosting (XGBoost): This method is an efficient implementation of the Stochastic Gradient Boosting framework, which is a machine learning technique used for building predictive tree-based models. Boosting is an ensemble technique that consists of introducing new models to correct the errors or optimize the existing ones.

   The gradient boosting technique uses a gradient descend algorithm to reduce the loss of the model when boosting. For a data set of *n* examples and *m* features, a tree ensemble model with *k* additive functions is created where each function corresponds to an independent tree structure. For time series forecasting, the leaves of regression trees are summed up to predict the output (BAUER et al., 2020). The XGBoost considers the leaves of the current tree and evaluates if they can be transformed into a new separate prediction to improve the model. This is a fast and accurate model but requires many hyper-parameters settings to be adjusted and does not perform well with noisy data.

b) Artificial Neural Networks: An ANN is a computational network designed to emulate the behavior of the human brain. The main unit of these structures is the neuron, which is inspired in actual human neurons. The neuron receives weighted inputs that are multiplied by an activation function to produce an output (ALIYU et al., 2019). The networks consist of multiple layers of connected neurons that can learn patterns from the input data after repeatedly training them. The connections have learned weights, that represent the strength of the link between the neurons. There are different neural network architectures that perform particularly well with time series, these are more detailed subsection 3.3.4.

### 3.3.4 Deep Learning for Time Series

Deep Learning is a subset of Machine Learning. It refers to learning models based on deep ANNs architectures. One of the advantage of DL over traditional algorithms is that as we introduce more data, the network performance improves. In recent years the use of Artificial Neural Networks and Deep Learning algorithms have rocketed due to the rapid evolution in hardware capabilities, shifting the paradigm of avoiding complex algorithms. Today, we can train more complex algorithms and deploy them in a more compact form so they consume fewer resources, we can see that today in mobile phones and other embedded devices.

Data from sensors can contain noise, there can be missing values, correlations between different types of magnitudes and/or between data from different sensors, seasonality, and trends. Statistical and classical Machine Learning (ML) time series forecasting models (such as the presented in subsection 3.3.3) are limited in their ability to extract information from nonlinear patterns and data in such conditions, therefore they rely on clean, mostly stationary data and hand-crafted features, which is time consuming, in some cases requires high processing capacity and can introduce human biases (GAMBOA, 2017). However, ANNs can automatically extract information from complex relationships, supports nonlinear behaviors, are robust to noise, can learn with missing data, and support multiple time-dependent input variables and multiple outputs forecasting, in a single or multiple steps ahead forecast.

Some architectures are specifically good at learning from sequential data. 1D-CNNs, Recurrent Neural Networks (RNN)s, LSTMs, GRU and combinations of these architectures have been used in many classification and forecasting problems with sequential data (HANNUN et al., 2019; ISMAIL FAWAZ et al., 2019).

#### 3.3.4.1 Recurrent Neural Networks (RNN)

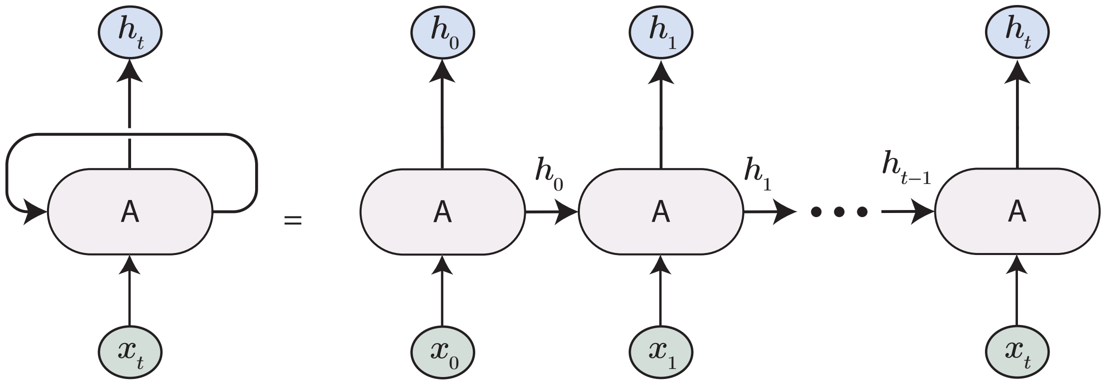
A RNN is a neural network that contains recurrent layers so it can learn from sequential inputs. They are RNNs are very flexible and capable to process all types of sequences. Figure 24 presents a folded and unfolded representation of a RNN. In a simple RNN, the state output *h* is just a copy of the output. Given an input sequence

$\{x_1, x_2, ..., x_t\}$ with $x_i \in R^n$, a RNN computes $h_t \in R^m$ for every time step $t$ and defines the recurrent function $F$ as (SHIH et al., 2019):

$$h_t = F(h_{t-1}, x_t), \tag{8}$$

Where $h_t$ is the current hidden state vector, $h_{t-1}$ is the previous hidden state and $x_t$ the input vector.

Figure 24 – Folded and Unfolded RNN.



Source – Author.

The problem with RNN is that during backpropagation, they suffer from the vanishing gradient problem, which means the gradient becomes smaller in some layers as the size of the sequence increases, this is usually in the firsts ones, therefore RNN fails in remembering long sequences.

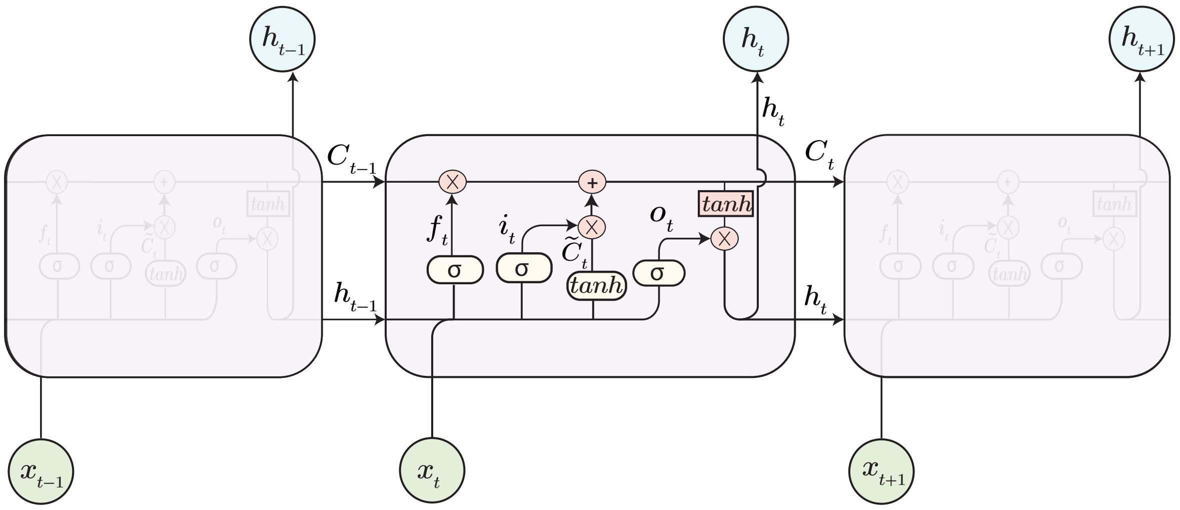
#### 3.3.4.2 Long Short Term Memory Neural Networks (LSTM)

LSTMs, introduced by (HOCHREITER; SCHMIDHUBER, 1997), are a type of recurrent networks designed to perform better with long sequences. Instead of simply having a single layer, there are four layers interconnected in what is called an LSTM cell. The cell is composed of different gates. Gates are a mechanism to control the passing of information through the cell, how much information or part of a sequence must be remembered or forgotten. This is achieved by combining a sigmoid layer and a pointwise multiplication.

Figure 25 represents the structure of an LSTM. The first part is the forget gate layer $f_t$. The forget gate decides which weights to remove from previous time steps. It takes the input $x_t$ and the previous hidden state $h_{t-1}$ and outputs a value between 0 and

1 for each one of the cell states $C_t$, using the Sigmoid function $\sigma$. A value of 0 represents the information that must be deleted. Next is the input gate layer $i_t$. The output from this layer is combined with an output state $\widetilde{C_t}$ generated by a *tanh* (hyperbolic tangent) layer, to update the old output state $C_{t-1}$ with the new information. The input gate determines what information continues according to its relevance. And finally the output gate layer $o_t$ decides which part of the information impacts the current time step.

Figure 25 – Structure of an LSTM cell.



Source – Author.

An LSTM can be defined by equations 9-14 (WU et al., 2020) where $W$ and $b$ are parameters learned by the network.

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \tag{9}$$

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \tag{10}$$

$$\widetilde{C_t} = tanh(W_c[h_{t-1}, x_t] + b_c) \tag{11}$$

$$C_t = f_t * C_{t-1} + i_t * \widetilde{C_t} \tag{12}$$

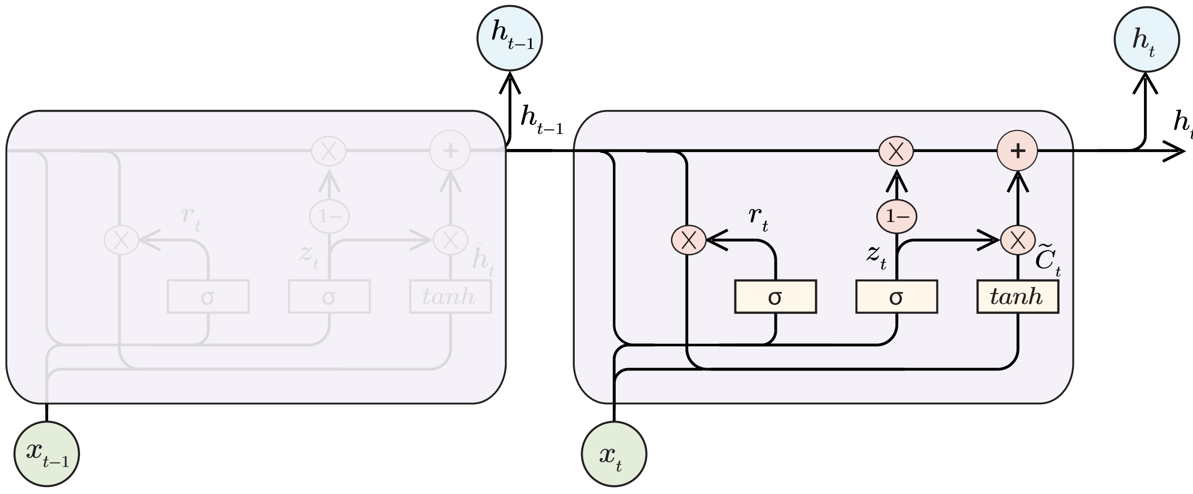
$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{13}$$

$$h_t = o_t * tanh(C_t), \tag{14}$$

### 3.3.4.3 Gated Recurrent Unit (GRU)

GRU networks also intend to solve the problem of the vanishing gradient. A GRU also has a forget gate like the LSTM, but in this case, it has fewer parameters and can generalize results with less data, Figure 26 presents a GRU cell structure. First is the update gate $z_t$, which works similarly to the input and forget gate in the LSTM network. The reset gate $r_t$ decides the amount of past information that must be forgotten.

Figure 26 – Gated Recurrent Unit Cell.



Source – Author.

### 3.3.4.4 Convolutional Neural Networks (CNN)

CNNs are a specialized kind of neural network for processing data that has a known, grid-like topology. Examples include time-series data, which can be thought of as a 1D grid taking samples at regular time intervals, and image data, which can be thought of as a 2D grid of pixels. As the name suggests, these networks use convolution in place of general matrix multiplication in at least one of their layers (GOODFELLOW et al., 2016).

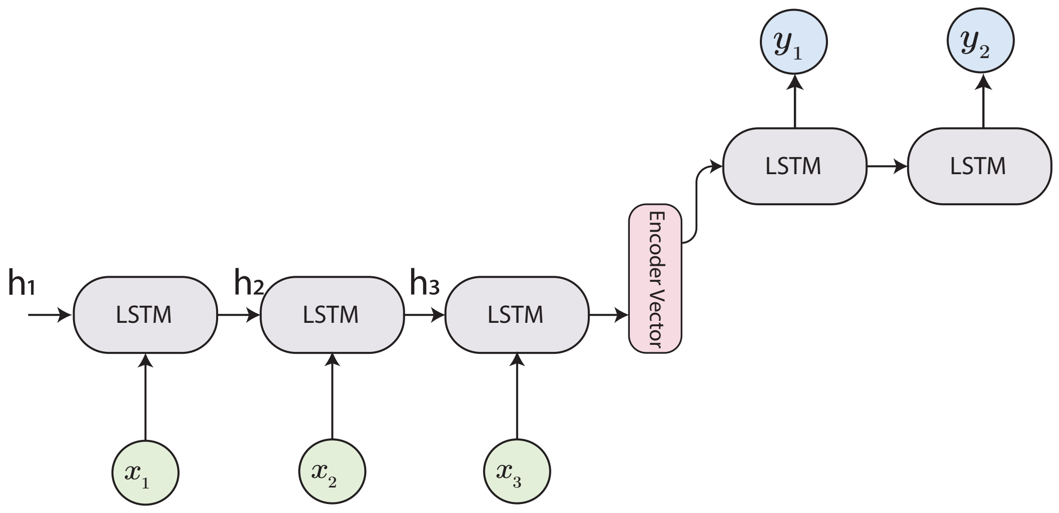
A 1D-CNN is a CNN to identify patterns in a sequence by performing convolution over a fixed length window (filter) that slides across the data. They can be used to extract features from sub-sequences or as a preprocessing stage in the complete Deep Neural Networks (DNN) model, extracting high level features and passing them to another layer. Due to their low computational requirements, compact 1D CNNs are well-suited for real-time and low-cost applications especially on mobile or hand-held devices (KIRANYAZ et al., 2021).

#### 3.3.4.5 Sequence to Sequence (Seq2Seq) Models

Seq2Seq models, introduced in (SUTSKEVER et al., 2014), has proven to be very effective in learning from sequential data. The model aims to map an input sequence to another sequence. Both sequences can be of arbitrary lengths. Seq2Seq models are behind applications like Google Translate and many online chatbots as they are very good at translation. These models are based on an encoder-decoder structure, where both are some composition of RNNs. The encoder creates an embedded representation of the inputs in a vector, which is passed to the decoder as the initial state, to process the information and produce the outputs.

The encoder is a stack of RNN units, in this case, LSTMs units. Every LSTM unit takes an input $x_i$, generates a hidden state, and passes the information to the subsequent cell. Finally, produces a vector with the codified information of every timestep, which is the encoded vector. Figure 27 illustrates the Seq2Seq model, where $h_i$ represents the hidden states generated by the LSTMs units.

Figure 27 – Sequence to Sequence model



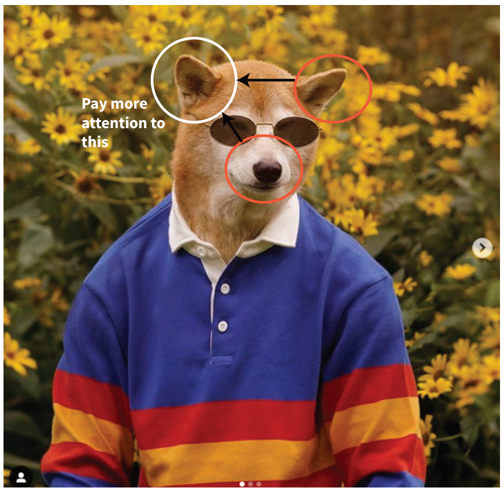
Source – Author.

The decoder is very similar to the encoder. It is also composed of LSTMs units and is initialized with the encoded vector. Each unit takes the previous hidden state with the information provided by the encoder and produces an output $y_t$ and its hidden state $s_t$. A problem with this model is that the neural network needs to compress all the information from an input sequence into a fixed-length vector, this becomes much harder with longer sequences, causing the network to forget the earlier parts of the sequence. To solve this problem the Attention mechanism was created (BAHDANAU et al., 2015).

### 3.3.5 Seq2Seq with Attention

The Attention mechanism was created to solve the problem of memorizing long source sequences in machine translation. Attention is about teaching the network to focus on the important parts of the input. For example, if we look at the dog in Figure 28, we do not need to inspect the entire image with detail to see that it is a dog in a human outfit. The attention mechanism allows the network to focus on the ears of the dog (white) and perceive the rest of the image in low resolution. It is capable of doing because first sees the dog's nose and another pointy ear on the right, then, the attention mechanism tells the network to focus on the other ear and identify if belongs to a dog.

Figure 28 – Example of Attention mechanism in image classification.

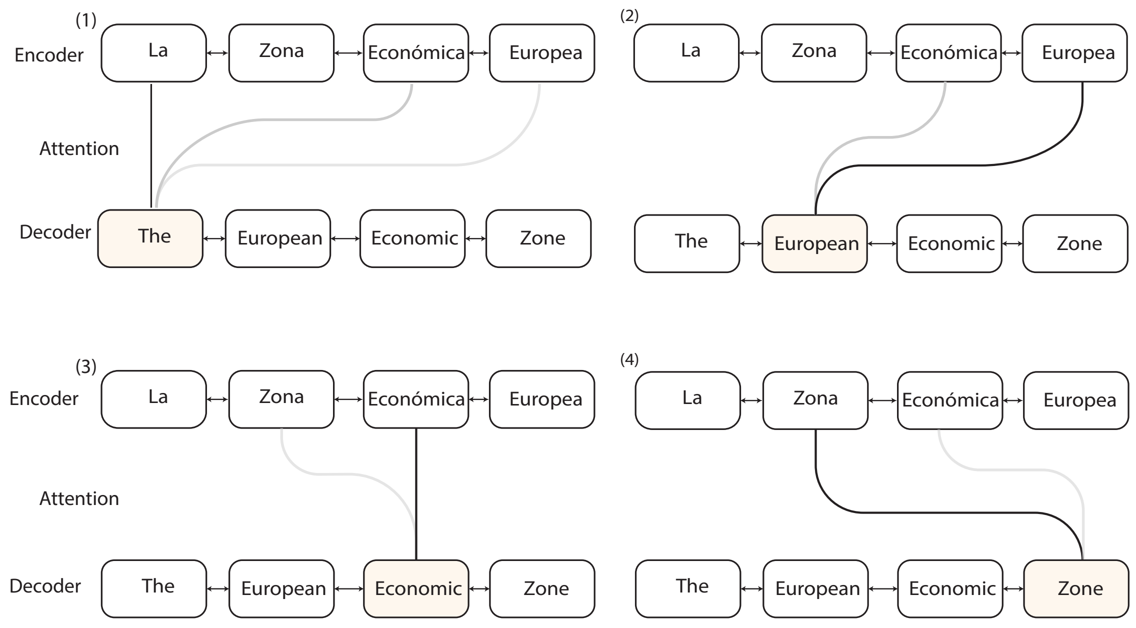


Source – Author.

The same happens with text translation and with any sequence. Instead of compressing the entire input into a single vector, for each input that the Encoder reads, the attention-mechanism takes into account the rest of the inputs at the same time and decides which ones are important by attributing different weights to those inputs. The Attention layer selects the relevant elements of the input sequence giving more *attention* to them. Then the Decoder will take the encoded sentence and the weights provided by the Attention mechanism producing the output. Figure 29 presents a representation of the algorithm when translating from Spanish to English.

First, the encoder creates a representation of the Spanish sentence as a list of vectors. Then the decoder starts translating the sentence one word at a time. To generate the translation, the attention mechanism defines what words are more important and provide more context to translate the current Spanish word (what words to pay more attention) by creating a weighted distribution of the input words. In the image,

Figure 29 – Example of text translation.



Source – Author.

when translating the second word "European", the attention mechanism decides that the word with more weight is going to be the immediate translation and assign some weight to "Económica", because it adds some context to the translation, the rest of the words are not important. This allows the network to process long sequences, because now the decoder does not need to remember an entire sequence and can pay attention only to the important parts.

### 3.3.5.1 Attention Mechanism

The encoder returns every hidden state $h_t$ to pass it to the attention layer, unlike simple Seq2Seq models. The decoder, in this case, takes the output from the Attention layer, the previous hidden state in the decoder $s_{t-1}$ and the previous output $y_{t-1}$ and outputs $y = \{y_1, y_2, ..., y_h\}$. Fig. 30 illustrate the model with attention.

Given an input sequence of length $n$ and the previous states from the decoder $s_t = \{s_1, s_2, ..., s_{t-1}\}$, a context vector $v_t$, defined in equation 16, is computed from the previous states in the attention mechanisms. The context vector $v_t$, is a weighted sum of the hidden states $h_i$ and the alignment scores. An alignment vector, defined in equation 15, with a size equal to the number of time steps on the source side, is derived by comparing the current source hidden state $h_i$ with each decoder hidden state $s_{t-1}$ and assigns a score $\alpha_{t,i}$. The attention layer is defined in the following equations (LUONG et al., 2015):

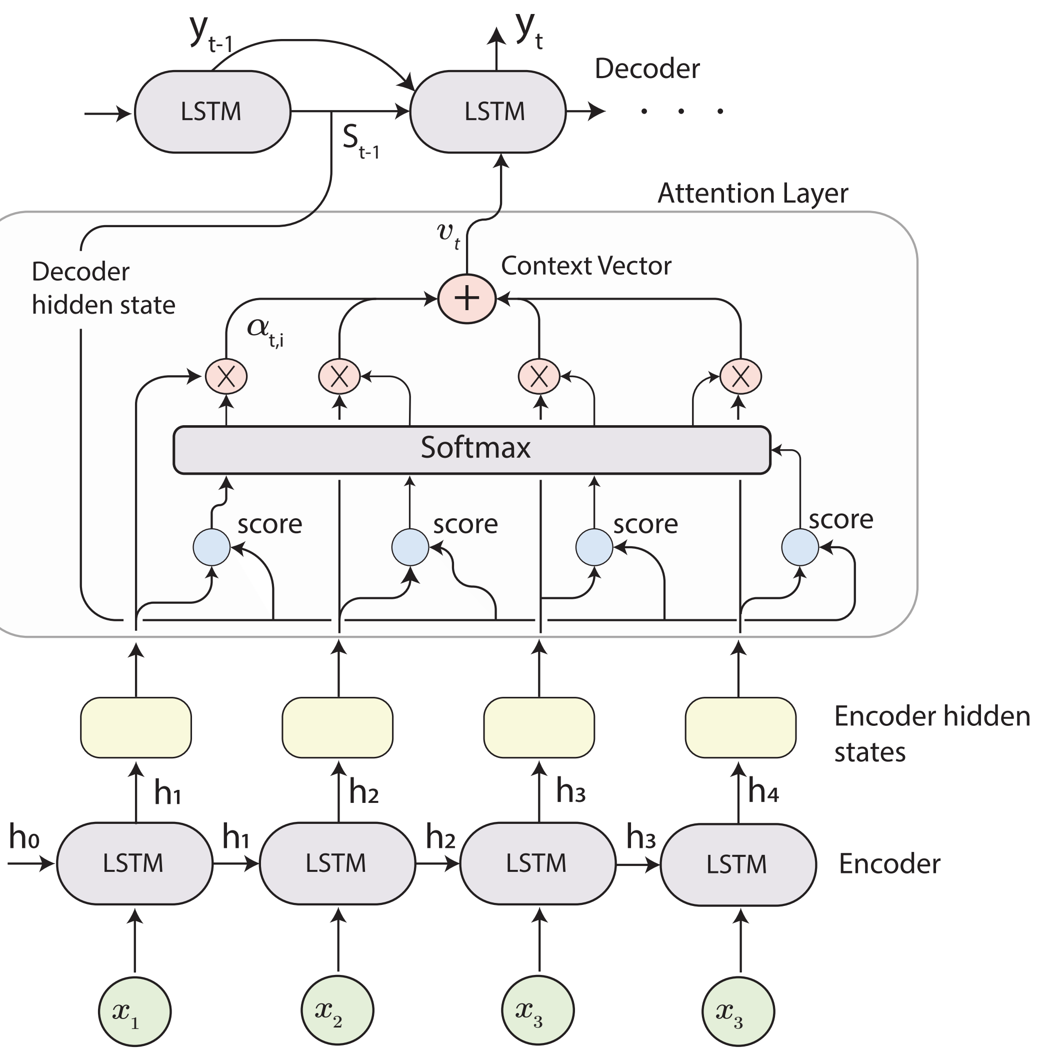
$$\alpha_{t,i} = align(y_t, x_i) = \frac{exp(score(s_{t-1}, h_i))}{\sum_{j=1}^{n} exp(score(s_{t-1}, h_j))} \tag{15}$$

$$v_t = \sum_{i=1}^{n} \alpha_{t,i} h_i \tag{16}$$

In this study, we are only considering Luong's Dot-Product attention score (LUONG et al., 2015) defined as:

$$score(s_t, h_i) = s_t^{\top} h_i \tag{17}$$

Figure 30 – Sequence to Sequence with Attention mechanism.



Source – Author.

Seq2Seq and Attention models for time series forecasting have been used in many researches showing promising results when dealing with long sequences (SHIH et al., 2019; LAI et al., 2018; LI, Y. et al., 2019; LI, S. et al., 2019).

## 3.4 CONCLUSIONS

The structure of the dataset and the network was presented. Different strategies of data prediction for saving energy in WSNs were presented together with a review of the most relevant works on this topic. A review of concepts related to time series and the most used and more effective forecasting methods including statistical, machine learning, and deep learning.

# 4 DATASET STRUCTURE AND EXPLORATORY DATA ANALYSIS OF THE WSN

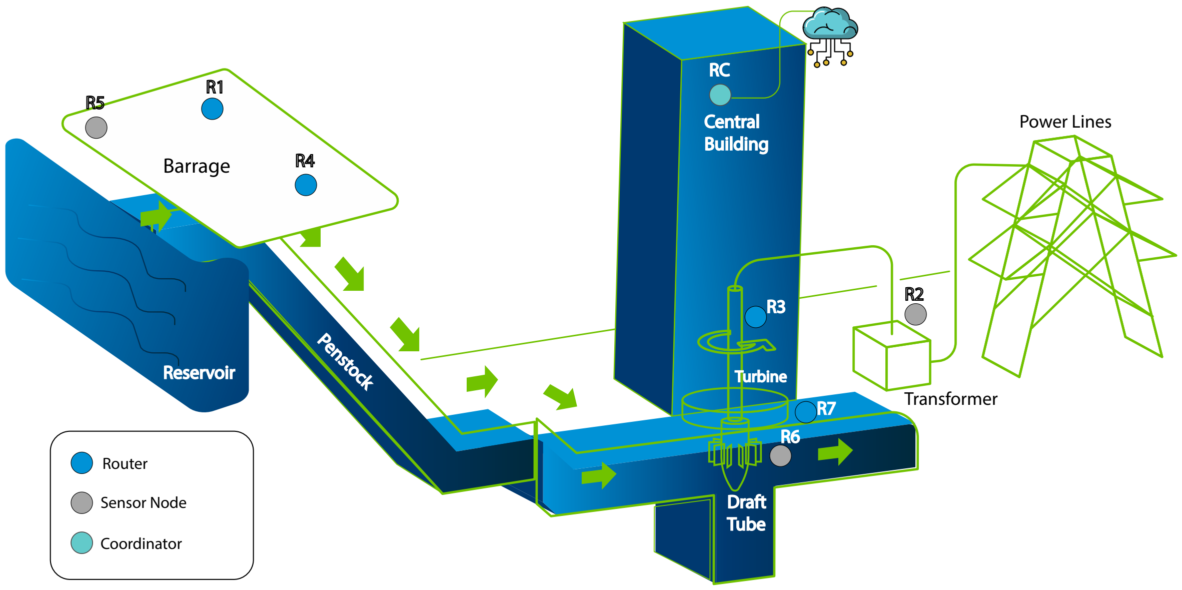
## 4.1 INTRODUCTION

This chapter describes the structure of the Dataset and the WSN, including the location, the function of each sensor node, and the topology of the network. The data science workflow is also performed in this chapter. A blind description of the dataset is performed in order to better understand the relationship between the different features and extract beneficial insights about the state of the WSN, possible problems, and solutions. The analysis was performed using Python as the programming language. Also, an anomaly detection stage that could be implemented in the CHs or the BS is also presented.

## 4.2 WIRELESS SENSOR NETWORK STRUCTURE

The data used in this work was gathered by a WSN deployed at the hydroelectric plant Cachoeira Dourada (MG, Brazil) with 8 routers nodes and 3 sensor nodes. All nodes report at least their internal temperature and power supply. Sensor nodes also send periodic readings from external industrial probes. The location of the nodes is presented in Figure 31, in a sketch of the plant. Only three nodes are connected to external probes, these ones will be identified as sensor nodes, the rest are used as routers.

Figure 31 – Scheme of the Wireless Sensor Network in the Hydroelectric Plant.



Source – Author.

The radios were deployed in a 92m × 78m area covering the barrage, the central building, and close to where the step-up transformers and high-voltage cables are located. Table 1 presents a description of the devices' role, the location where they are deployed, and their hexadecimal address. Radios R1, R4, and R5 are located at the top of the barrage. Radios R6, R3, and RC are located in the central building and radio R2 is located next to the transformer. Radios operate in the 2.4 GHz band under Zigbee 3.0 Protocol in a centralized mesh configuration. The theoretical receiver sensitivity of each node was -95 dBm. The nodes were configured to operate only in the Zigbee channel 15 (2425 MHz). All the radios were set as routers (relays), so each one was able to establish different routes to communicate up to the coordinator. The coordinator uploads the radios data to a local database (PEREIRA et al., 2018; ANTAYHUA et al., 2020).

Table 1 – Description of the nodes.

| **Radio** | **Function** | **Location** | **Hexadecimal Address** |
|---|---|---|---|
| RC | Coordinator | Central Building | 00.57.FE.04 |
| R1 | Router | Barrage | 00.57.FE.09 |
| R2 | Sensor | Transformer | 00.57.FE.03 |
| R3 | Router | Central Building | 00.57.FE.05 |
| R4 | Router | Barrage | 00.57.FE.01 |
| R5 | Sensor | Barrage | 00.57.FE.0E |
| R6 | Sensor | Central Building | 00.57.FE.0F |
| R7 | Router | Central Building | 00.57.FE.06 |

Source – (PEREIRA et al., 2018)

Sensor nodes communicate with different sensor modules that have their own hexadecimal address. Sensor Node 00.57.FE.0E, located at the barrage, is powered by solar energy and communicates with probes:

a) **29.E5.5A.24**: Temperature acquisition sensor that extract information from 2 PT100 sensors,and its own temperature and Bus Voltage (VBus).

b) **A7.CB.0A.C0**: Current / Voltage acquisition sensor that extract information about the level of the water wall at the dam using channel 1.

c) **34.B2.9F.A9**: Power acquisition sensor. Extract the MPPT voltage and the voltage from a solar panel.

Sensor Node 00.57.FE.0F, located at the drainage, is powered by AC from the Hydroelectric plant and communicates with modules:

a) **01.E9.39.32**: Current / Voltage acquisition sensor. Measure level of the water at the drainage.

b) **A4.0D.82.38**: Power AC/DC Input. Measures voltage from Battery and Supply Voltage.

The third sensor, 00.57.FE.03, is located at the Step Up Transformer, also powered by solar energy and communicates with modules:

a) **9F.8D.AC.91**: Temperature acquisition sensor. Which gives two temperatures measures, Oil and Cabinet, through channels 0 and 1. Reliable and accurate oil temperature measurement ensures a longer life of the transformer and is crucial to maintaining the overall health of the asset.

b) **50.39.E2.80**: Power Solar Panel sensor. Extract the MPPT voltage and the voltage from the solar panels.

## 4.3 EXPLORATORY DATA ANALYSIS

Exploratory data analysis (EDA) is used to summarize the characteristics of data sets using data visualization and statistics, to discover patterns, make or corroborate assumptions and find anomalies or outliers. EDA is usually used to reveal information about the data beyond the normal modeling or hypothesis testing process, providing a better understanding of the data set, or process described by it, and the relationships between the different elements and features of it. It also helps determine the statistical and ML techniques that can be considered for data analysis are appropriate.

### 4.3.1 Dataset Structure

The data was extracted from the source database as single modules. Each module corresponds to a radio or to a probe connected to the radio, and are identified with an hexadecimal code. Each module contain the type of the node (Router or Sensor), their measures, and a timestamp. After extracting each module, a single dataset composed of the measures from all nodes was created. A second dataset was created with the Received Signal Strength Indicator (RSSI) measures from all nodes. The structure of the dataset is described next and a shorter view of the datasets is presented in Figure 32, being the last one the one with the RSSI data:

a) The index of both datasets is the timestamp of the measures.

b) The shape of the first dataset is 7847000 rows and 16 features columns.

c) The date starts in 2017-10-01 00:00:05 and goes to 2020-02-21 19:29:04.

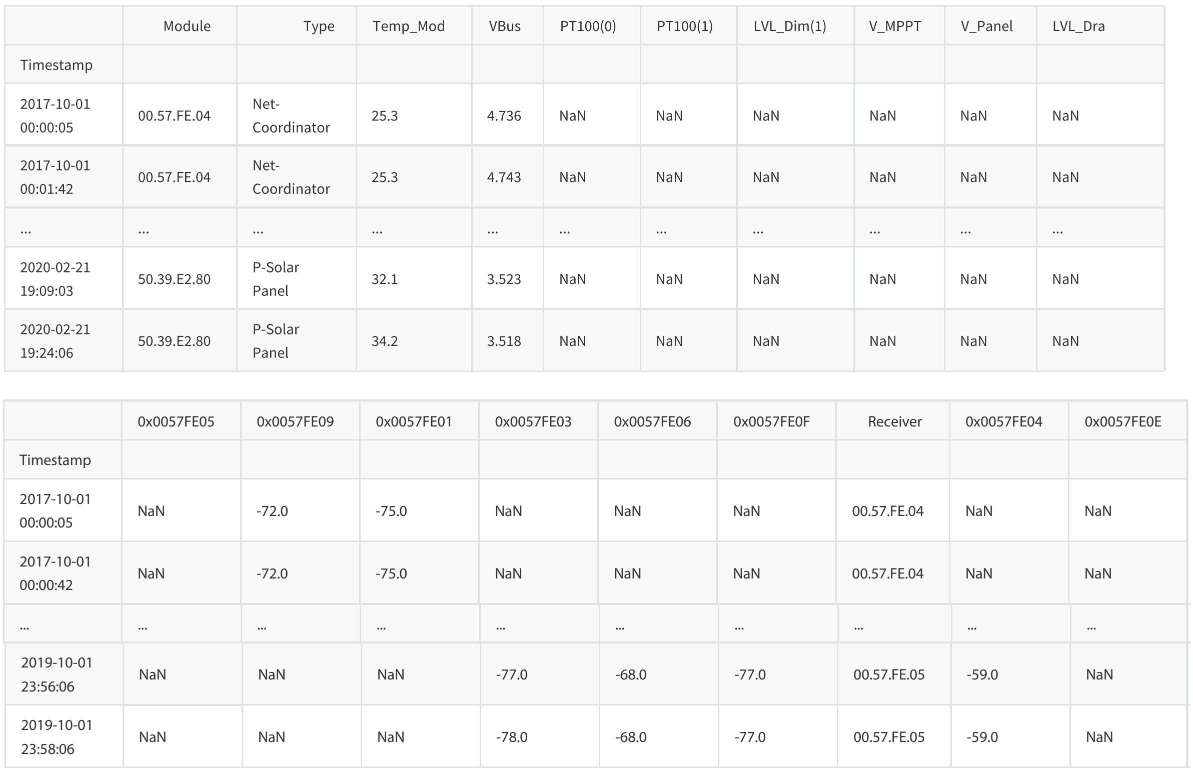
d) There are 15 modules

e) Every modules has different measures or features. Module 00.57.FE.04 only measures temperature and voltage because is the coordinator. Modules that belong to sensors present more features.

f) Different module have different sampling periods.

g) The dataset with the RSSI values has 5224684 rows and 9 columns. One of the columns (Receiver) indicates the module receiving the data at the moment. The rest of the columns are the transmitters.

Figure 32 – Resumed view of both datasets. Upper dataset corresponds to sensors and radio measures and lower dataset contains the RSSI measures.



| | Module | Type | Temp_Mod | VBus | PT100(0) | PT100(1) | LVL_Dim(1) | V_MPPT | V_Panel | LVL_Dra |
|---|---|---|---|---|---|---|---|---|---|---|
| Timestamp | | | | | | | | | | |
| 2017-10-01 00:00:05 | 00.57.FE.04 | Net-Coordinator | 25.3 | 4.736 | NaN | NaN | NaN | NaN | NaN | NaN |
| 2017-10-01 00:01:42 | 00.57.FE.04 | Net-Coordinator | 25.3 | 4.743 | NaN | NaN | NaN | NaN | NaN | NaN |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 2020-02-21 19:09:03 | 50.39.E2.80 | P-Solar Panel | 32.1 | 3.523 | NaN | NaN | NaN | NaN | NaN | NaN |
| 2020-02-21 19:24:06 | 50.39.E2.80 | P-Solar Panel | 34.2 | 3.518 | NaN | NaN | NaN | NaN | NaN | NaN |

| | 0x0057FE05 | 0x0057FE09 | 0x0057FE01 | 0x0057FE03 | 0x0057FE06 | 0x0057FE0F | Receiver | 0x0057FE04 | 0x0057FE0E |
|---|---|---|---|---|---|---|---|---|---|
| Timestamp | | | | | | | | | |
| 2017-10-01 00:00:05 | NaN | -72.0 | -75.0 | NaN | NaN | NaN | 00.57.FE.04 | NaN | NaN |
| 2017-10-01 00:00:42 | NaN | -72.0 | -75.0 | NaN | NaN | NaN | 00.57.FE.04 | NaN | NaN |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 2019-10-01 23:56:06 | NaN | NaN | NaN | -77.0 | -68.0 | -77.0 | 00.57.FE.05 | -59.0 | NaN |
| 2019-10-01 23:58:06 | NaN | NaN | NaN | -78.0 | -68.0 | -77.0 | 00.57.FE.05 | -59.0 | NaN |

Source – Author.

### 4.3.1.1 Exploring Missing Data

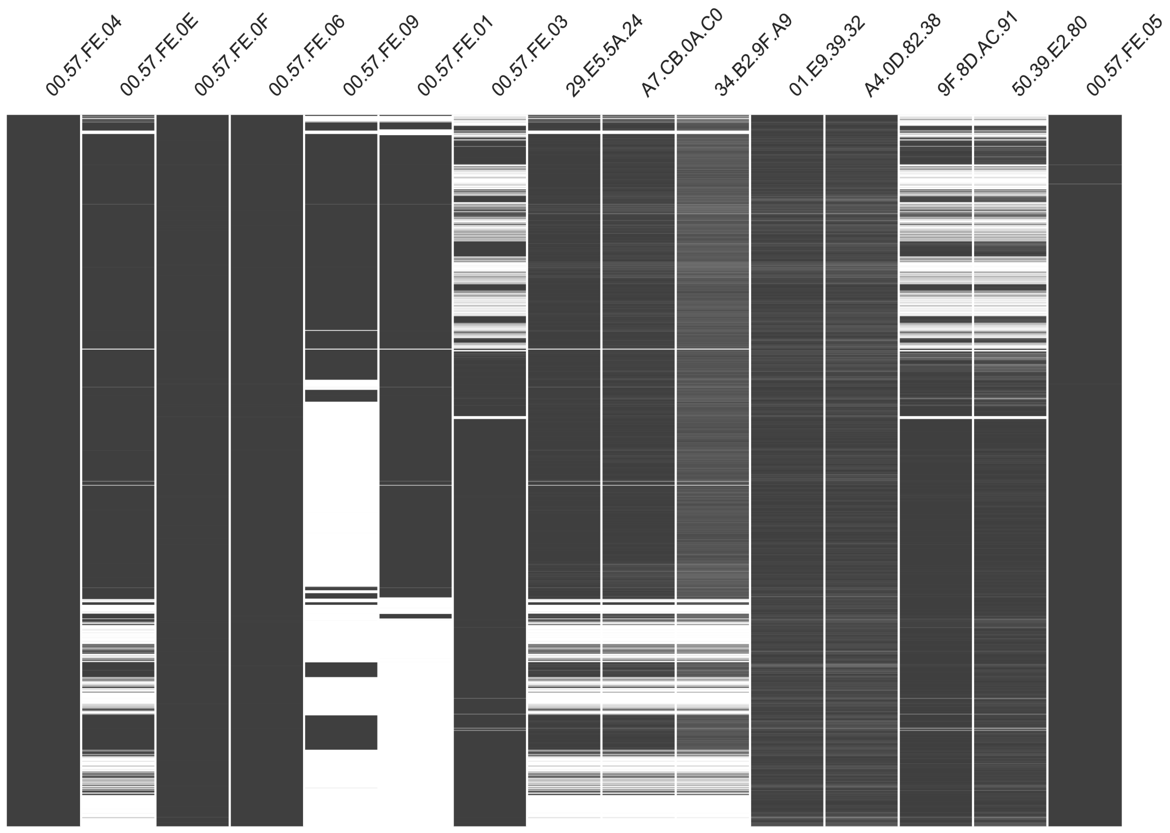
When exploring data sets, an important step is to analyze missing data. In almost every process that is represented with data, there will be some missing points due to errors that can occur during the collection. The analysis of missing data in sensors can help us understand about relationships between sensors and the stability of the measures in different periods of time.

Figure 33 represents the missing data in a nullity matrix of modules and time, to quickly visualize patterns in data completion. The white lines represent the missing values and the image is read from the top to the bottom, the top lines represent the firsts timestamps. From the matrix, we can see that module 09 is the one presenting more missing data. Although there is missing data in a random distribution in almost every module, most missing data are concentrated in a specific timestamps range.

Radios 00.57.FE.04 (Coordinator), 0F, 06, and 05 have almost no missing data. Sensor nodes A7.CB.0A.C0, 29.E5.5A.24, and 34.B2.9F.A9, located at the barrage and radio 00.57.FE.0E started to fail and present inconsistent data at the end of the period, however, sensor node 34.B2.9F.A9 presents much more missing data than the rest, this could mean the sensor is located in a less favorable place.

Module 01, located also at the barrage, stopped functioning around the same time that 0E started to present inconsistent data, which could explain the inconsistencies of module 0E. This can be checked by exploring if 01 was a constant link of 0E.

Figure 33 – Missing Data Representation in Null Matrix.

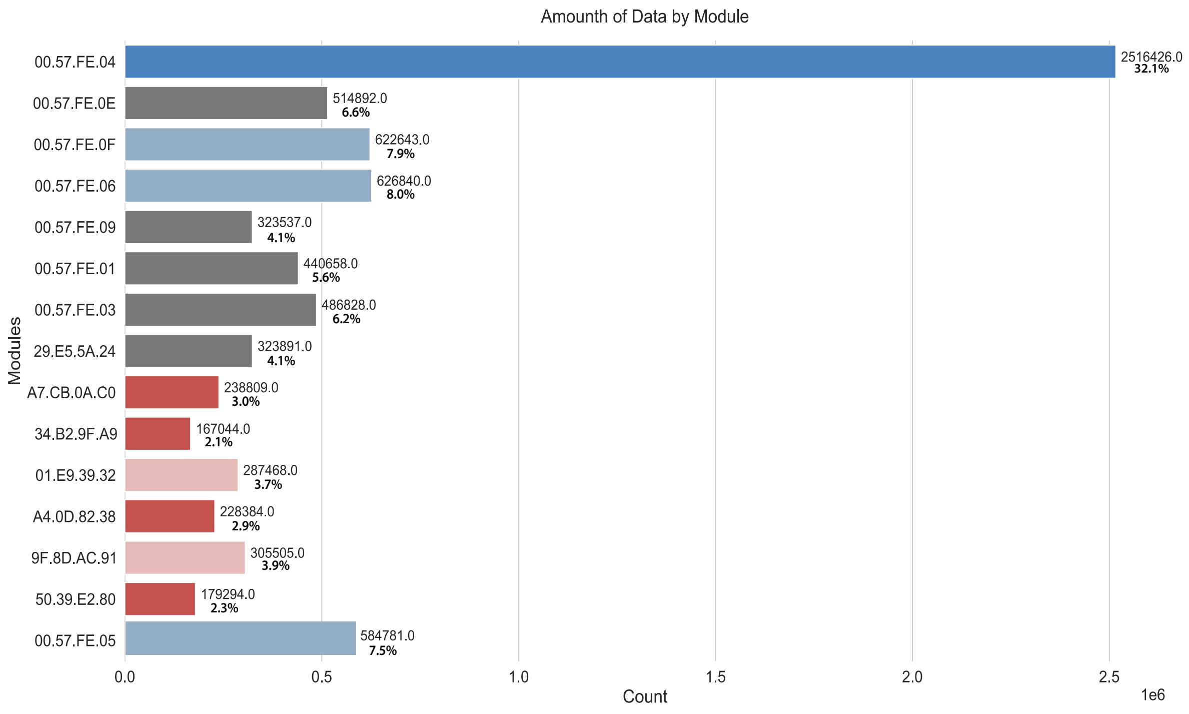


Source – Author.

#### 4.3.1.2 Exploring Temperature Data from Modules

Visualizing the amount of data by module helps us understand what modules transmit more data to the Base Station. Figure 34 shows a bar chart that represents the amount of data transmitted per module. It can be seen that the coordinator is the one receiving more data, which is the expected behavior, followed by modules 06, 0F, and 05 in that order. These are all located in the main building and serve as a path to the coordinator. Sensors transmit much fewer data than routers, as was also expected.

Figure 34 – Amount of Data by Module.

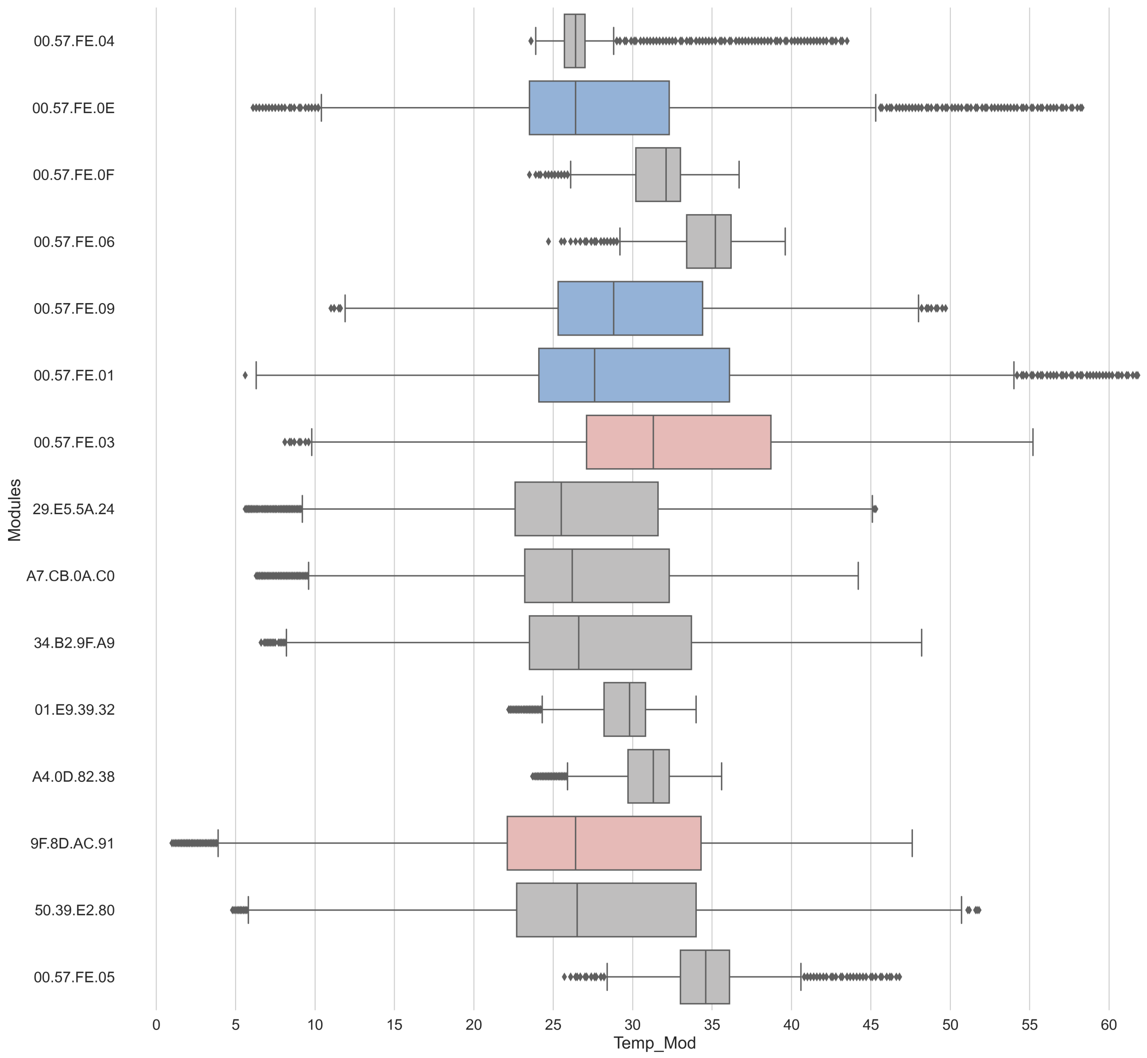


Source – Author.

An exploration of the temperature values of every module and how they vary in time is represented in Figure 35 using a box plot chart. Some modules present some explainable outliers and others that can be interpreted as corrupted data or miss readings, for example, module 00.57.FE.01 (Radio R4) has some points above 60 degrees Celsius, which can be considered a high temperature. This module is located in the barrage, close to modules 00.57.FE.09 (Radio R1) and 00.57.FE.0E (Radio R5), which has also some values close to 60 °C. For the location we could assume that these radios get more sunlight and are more exposed to the environment than the rest, therefore we could assume that on a hot sunny day these values are possible. To corroborate this, we can check the values close to those outliers and see if they follow the behavior of a normal day or if they are the result of corrupt data.

Module 00.57.FE.03 is Radio R2, located close to the Transformer, also presents high temperature values but due to the nature of the environment, these are not considered necessarily outliers. Values below 10 degrees can be studied to see if there was some problem those days or if there was a measurement error. We can see how the 3rd quartile for this module contains higher temperature values. Modules 00.57.FE.06, 00.57.FE.0F, 00.57.FE.04, and 00.57.FE.05 present a lower deviation, this means their temperatures vary less than the rest of the modules, which can be explained by their location (main building).

Figure 35 – Box plot detailing the variation of temperatures by Module.

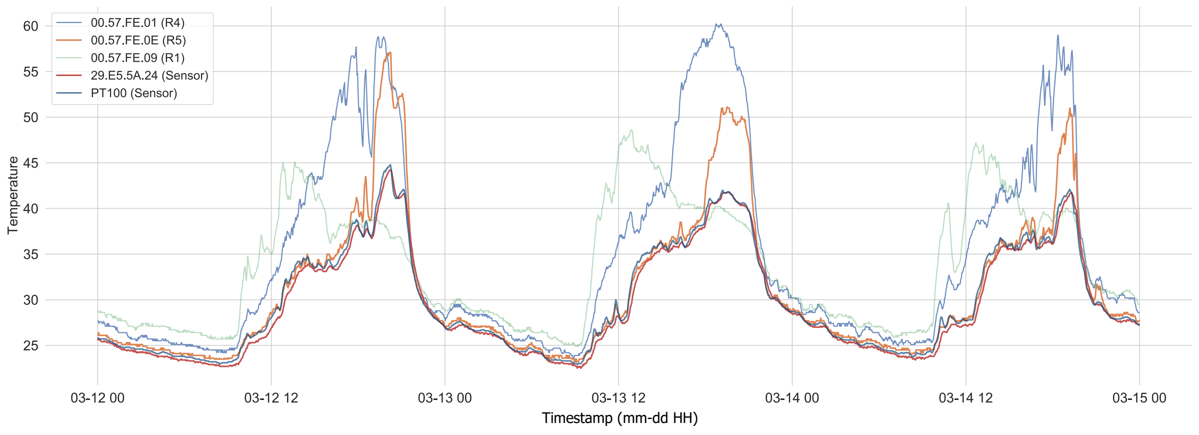


Source – Author.

From the Box Plot above, we can see how modules at the Barrage present some high temperature values. In order to check if there was some malfunctioning or corrupt data, we plotted the temperature behavior corresponding to that specific period of time, in Figure 36.

It can also be seen how the temperature follows the normal pattern of temperature, start rising during the day, and falls at night, and apparently, this was an unusually hot day. We can see how radios R4 and R5 are very correlated, which can be explained by their proximity (spatio-temporal correlation) with radio R4 with higher temperature

Figure 36 – Temporal Behaviour of Radios at the Barrage.



Source – Author.

values. The temperature in Radio R1 rises first than the others and is normally lower, probably because is in someplace with more shadow or ventilation during the day.

Temperature from the sensor module 29.E5.5A.24 and temperature from the PT100 follow almost the same distribution. Temperature from radio 00.57.FE.0E (R5) is highly correlated with the temperature from the sensor module, values from the radio are higher during the afternoon. Radio 00.57.FE.0E(R5) is also correlated with Radio 00.57.FE.01(R4), Radio R4 gets higher temperature values during the day. Radio R1 does not follow the same distribution as the rest despite being at the same location.

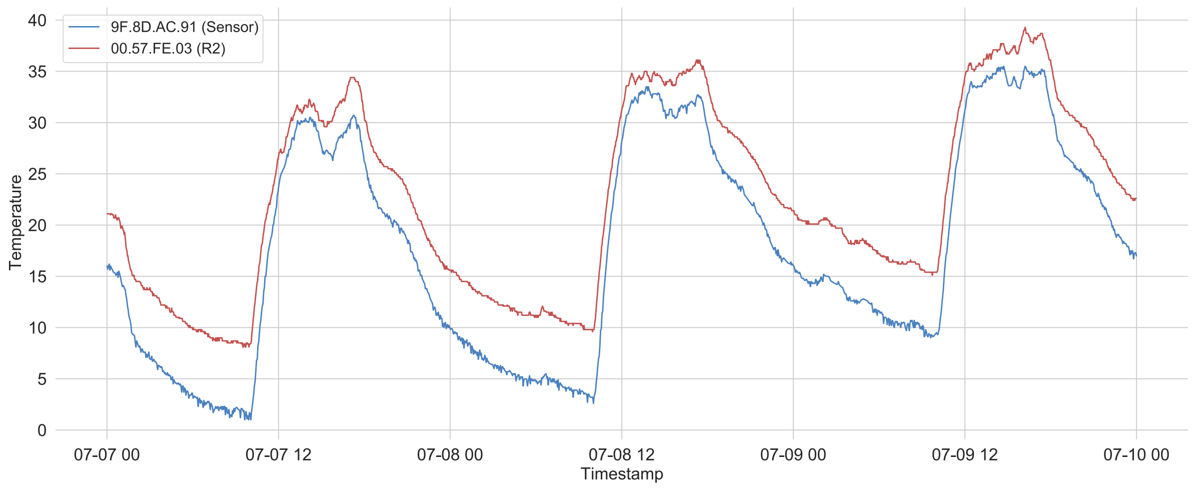
Since temperatures from PT100 and the sensor are almost the same, we could only transmit the temperature from the module, or the sensor, and not both values, reducing the amount of transmitted data and energy consumption. Also, the temperature from radio 00.57.FE.0E (R5) is almost the same that the temperature from the sensor with an offset, knowing this can be useful to recover some lost data.

Next, was examined the extremely low values recorded from the sensor at the transformer (see Figure 37) by exploring its temporal behavior in that specific range of time. We can see how values are lower these days but apparently, there was no malfunctioning of the sensor or radio, this was probably a particularly cold day. Both the values from the sensor and the radio are correlated, being values from the radio higher. This can be explained by the proximity between them and the monitoring of the same area. Looking at Figure 38 we can see the correlation between the temperatures.

#### 4.3.1.3 Exploring Relationships with Date

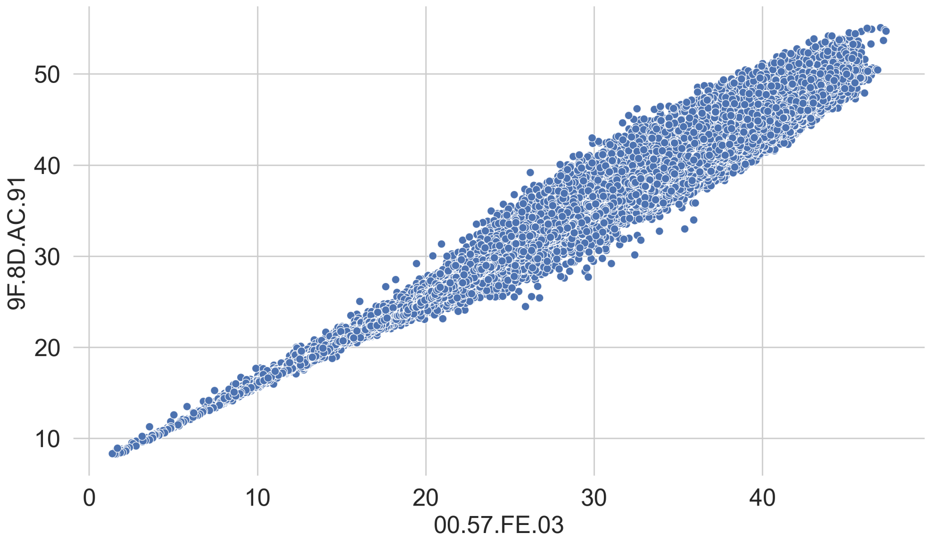
Exploring how the features vary in some specific time intervals, such as hourly, weekly and monthly variations can also add important knowledge about the dataset. In

Figure 37 – Temperature Behaviour of Radios in Step Up Transformer.



Source – Author.

Figure 38 – Correlation between Sensor Temperature and Radio.

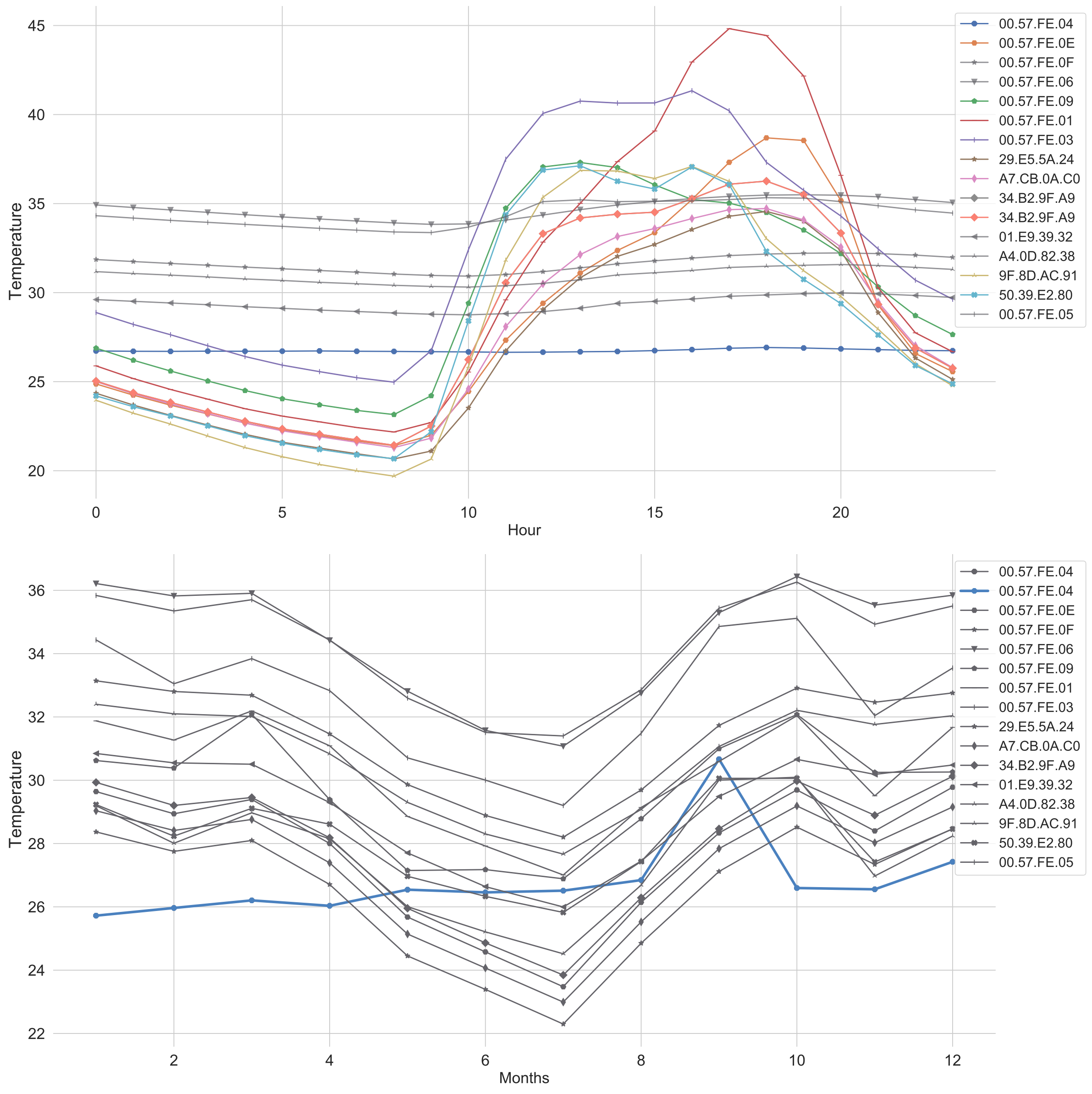


Source – Author.

order to do so, columns with the hour, month, year, day, and week were extracted from the data timestamp. We can see in Figure 39 how the temperatures start to increase after 8:00 until 17:00 when starts to decrease, this corresponds with the day cycle, also we can see how the temperatures are lower in months of winter and higher in summer.

Some modules do not have almost any variation in temperature during the day (marked in gray), these modules are probably in locations inside the building. When

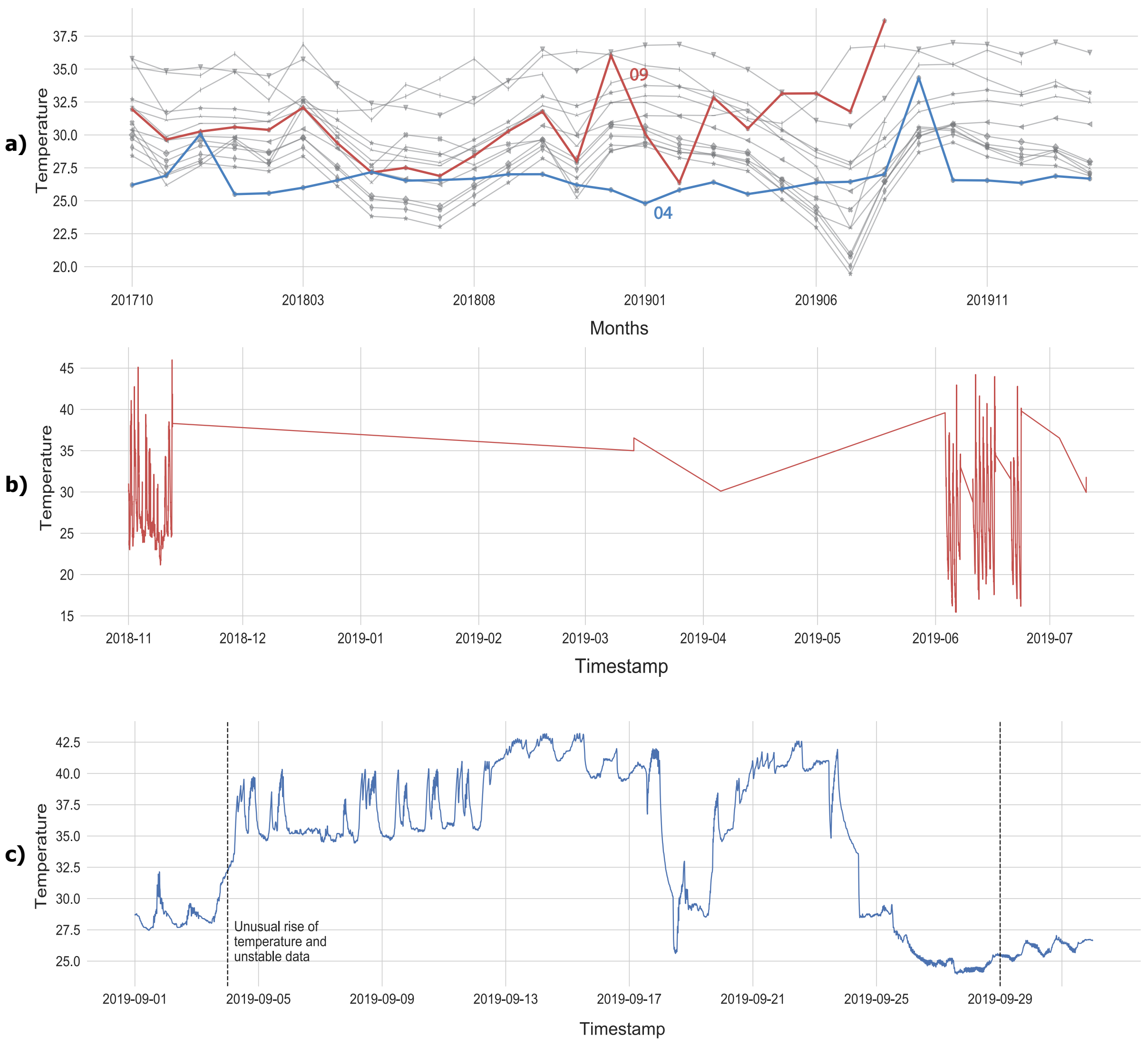
Figure 39 – Hourly and monthly mean behaviour of temperature by modules.



Source – Author.

seeing the behavior of the temperature over the year we can see how it starts dropping around April until around August or September when it starts rising again, this is because of changes in stations. Temperature values of the coordinator are usually stable and do not present a daily pattern like the rest of the modules, however, we can see in September 2019 (201909) there was a strange spike. Module 09 presents a very unstable behavior and many missing data. Apparently, the module stopped working in November 2018, worked in July 2019 and it went off again in 2019/10, we can see that in Figure 40 b). When inspecting the behavior of module 04 more closely during 2019 (see the Figure 40 c)), we can see there was an abnormal behavior in the entire month with missing points and sudden drops. This module is the main building, probably in a temperature controlled environment. Other modules do not present this abnormal behavior during this period of time, only the Coordinator, therefore we can assume it was something related to the environment where it is located.

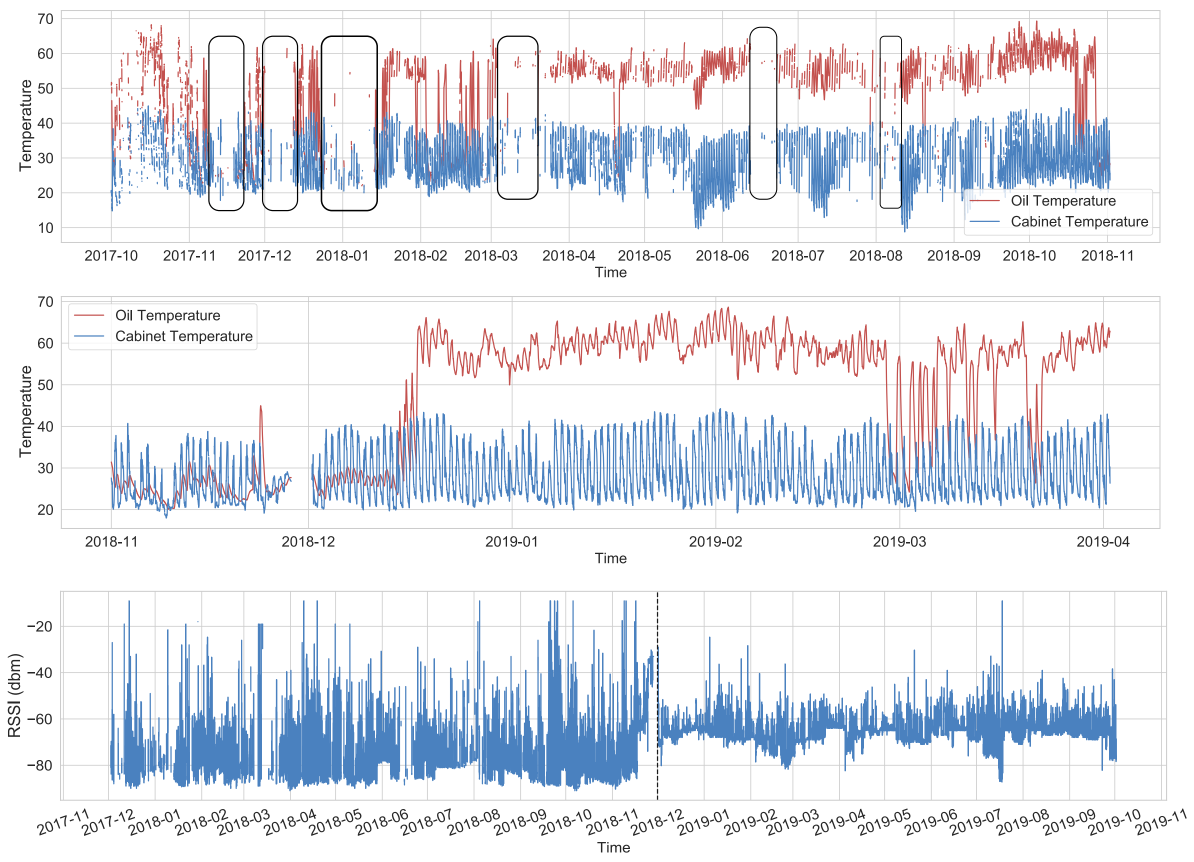
Figure 40 – Temperature analysis of Modules 04 and 09. a) Temperature of modules in a monthly average. b) Temperature of module 09 during abnormal period. c) Temperature behaviour of module 04 during 2019-09



Source – Author.

As was presented in Figure 33, there are many missing data from Module 00.57.FE.03 and sensor 9F.8D.AC.91, which senses Oil Temperature and Cabinet Temperature of the transformer, besides his own temperature. The missing data is mostly during 2018, until 2018/10, we can see it in Figure 41. We can see how the temperature data is much more stable and without so many missing points. It can also be seen that the oil temperature from the step-up transformer is lower at the end of 2018, probably because that part of the plant was not working, and then the temperature rises again at mid December.

Figure 41 – Cabinet and Oil temperatures together with the RSSI data during the unstable period of time.



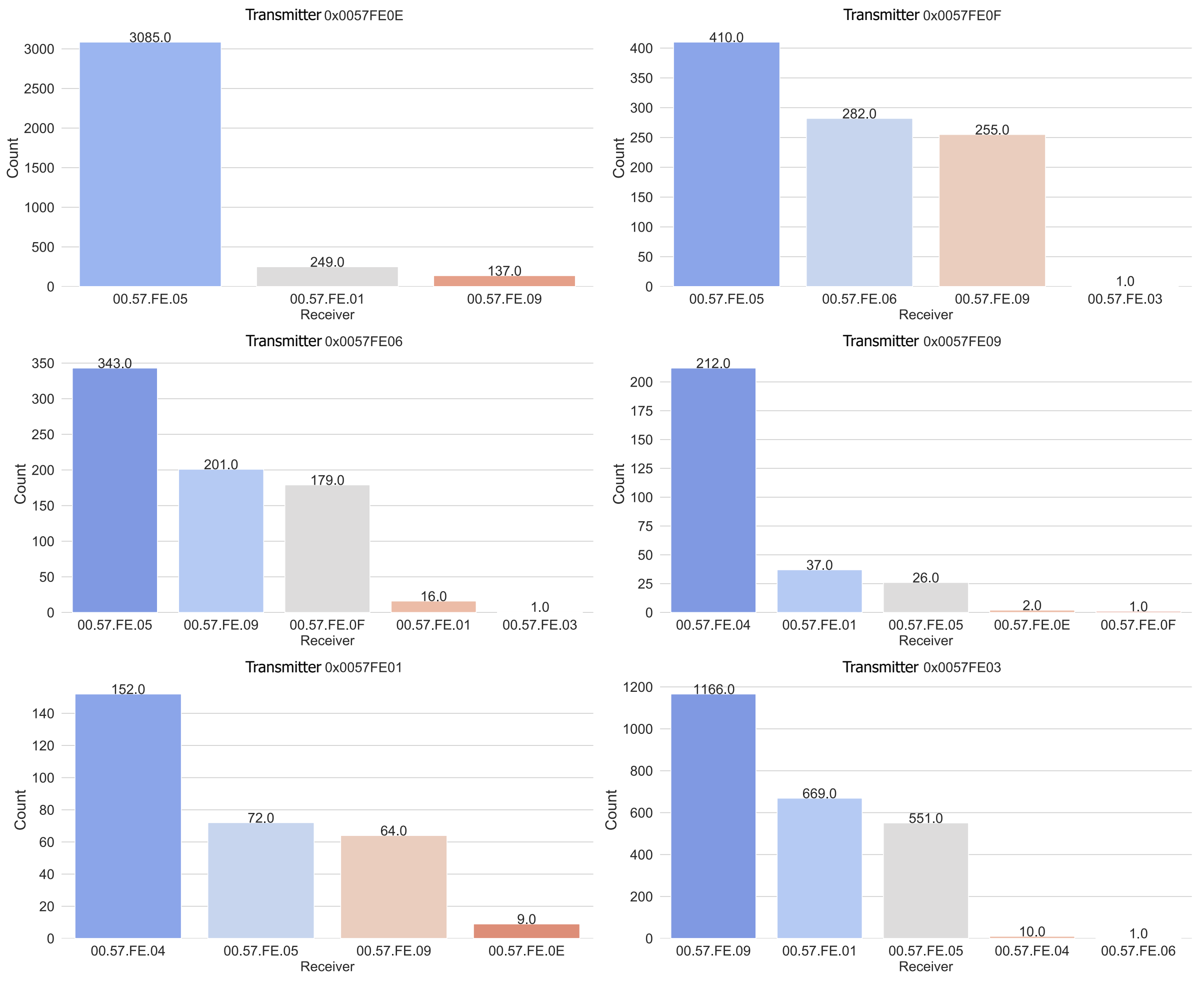
Source – Author.

In the image we can see how transmission from radio 03 (Transformer) starts very noisily. Radio 03 has a constant link with Radio 05 at the main building. The transmission presents many spikes until late November of 2018. There was also a power increase in transmission to Radio 05 of around 20db. After the increase in power, in December 2018, transmission presented fewer spikes and almost none missing data. This power rise could mean there was a change in the position of the Radio to a location closer to 05, or a location where is less affected by Electro Magnetic Interference (EMI) from the Transformer.

### 4.3.2 Exploring RSSI Data

The RSSI is a measure of the strength of a received wireless signal. The second dataset contains the measures of the RSSI data from all radios. Due to the presence of high-power equipment at the hydropower plant, EMI may be generated which can lead to misreadings in the RSSI values, generally registered as abrupt spikes. Also, some transmitters present maximum values of -9dBm. A RSSI of -9dBm corresponds to the maximum value of power that can be registered by node radios, presented as a spike in power. In this case, it also represents an abandoned communication between two neighbor radios. Therefore this data can be removed. The number of spikes in a communication link between two radios can give some insights about the stability between links, Figure 42 presents a counting of the abandoned links per communication.

Figure 42 – Abandoned links per transmitter.



Source – Author.

The Coordinator and module 05 do not present any spikes when transmitting to any other radio, meaning they do not have abandoned links in communication. These two modules are located in the main building and are the most stable ones. Module 0E at the barrage is the one with more missing links when communicating with module 05, probably because of the distance between them, we can also see there are much fewer missing links with radio 01, which is more closer. This means that communication of radio 0E can be improved to reduce the missing links. A possible solution can be routing through another radio and not to module 05. Transmitter 03 is also very unstable, presenting many spikes when transmitting to 09 and 01, which is understandable because of the distance between them. Figures 43 and 44 presents us a 5 minutes average of the RSSI data received by the radios over the entire measured period. We can see which radios communicate with each other and how stable is the communication.

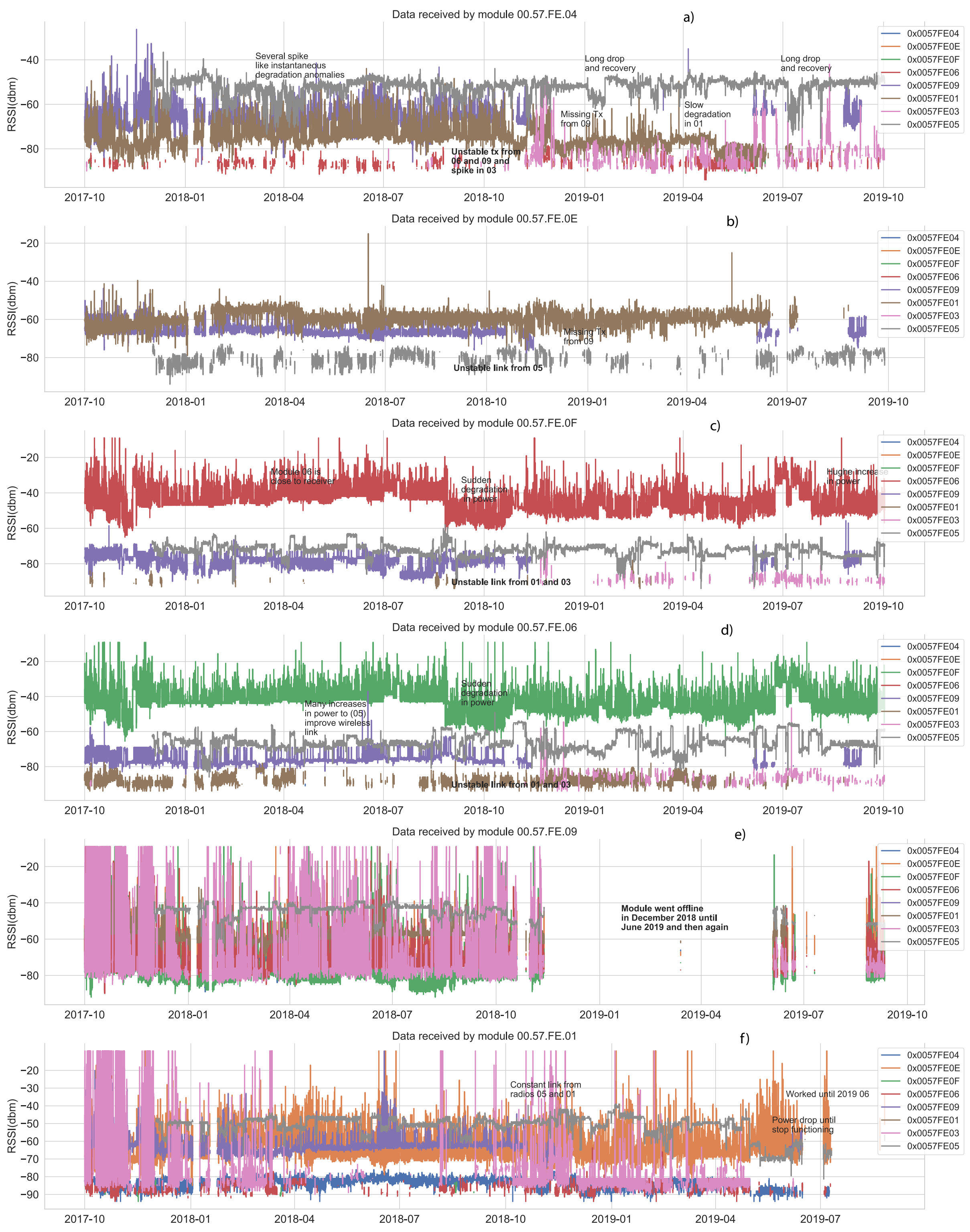
Since the radios can automatically increase their transmission power to improve their link quality, these increments are registered by other radios as transitory pulses in the RSSI. We can see the difference between data from radios in the main building (04 and 05) and radios that are more exposed to the industrial environment, such as radio 0E and 0F, in that they do not present as many spikes and power rises as the rest of the modules.

**Radio 04 (Coordinator in blue) Figure 43 a):** Receives constant data from 00.57.FE.01 (brown). This module is located at the Barrage with modules 00.57.FE.09 (purple) and close to 00.57.FE.0E (orange). Radio 01 (brown) shows a slow degradation in transmission power after 2018-04. Could be due to the aging of the electronics components as explained in (MANN et al., 2016). Its constant links are 05 (gray) and 01 (brown). The module started to receive constant data from module 00.57.FE.05 in December 2017, and there was a degradation in power around 2019-01 with a recovery and another in 2019-07 that lasted longer but recovered eventually. There are also several spikes like instantaneous degradation anomalies with fast recovery, these could be due to weather conditions or movement of big objects since the sensors are in an industrial zone. We can see data from 00.57.FE.06 (red), located at the Drainage is very inconsistent.

**Radio 00.57.FE.0E (orange) Figure 43 b):** Receives constant data from 00.57.FE.01 (brown). Communication is established mostly through this radio. Radio 00.57.FE.01 (brown) presents an increase in power that starts around 2018-01-10 until 2019-04. The increase in reception power could be attributed to an improvement in the Line of Sight between the sensors. These sensors are close but in an industrial environment, it can be seen how the data from 01 has more power but is noisier. Receives data from module 05 in an inconsistent way, this module is far from the main building where 05 is located, therefore, transmission occurs through module 01. Missing data from Module 00.57.FE.09 (purple) in until 2018-12 and then again in 2019-05.

**Radio 00.57.FE.0F (green) Figure 43 c):** It has a stable link with radio 00.57.FE.06 (red). RSSI from radio 06 is high because both radios are in the same location (Drainage). Started with power around -50 dBm and increased to -40 dBm in 2018-07, then went down to -50 dBm again and never returned to -40 dBm. A probable cause could be that a new object was located between them or the power was adjusted. Receives constant

Figure 43 – RSSI data received from radios 04, 0E, 0F, 06, 0E and 01.

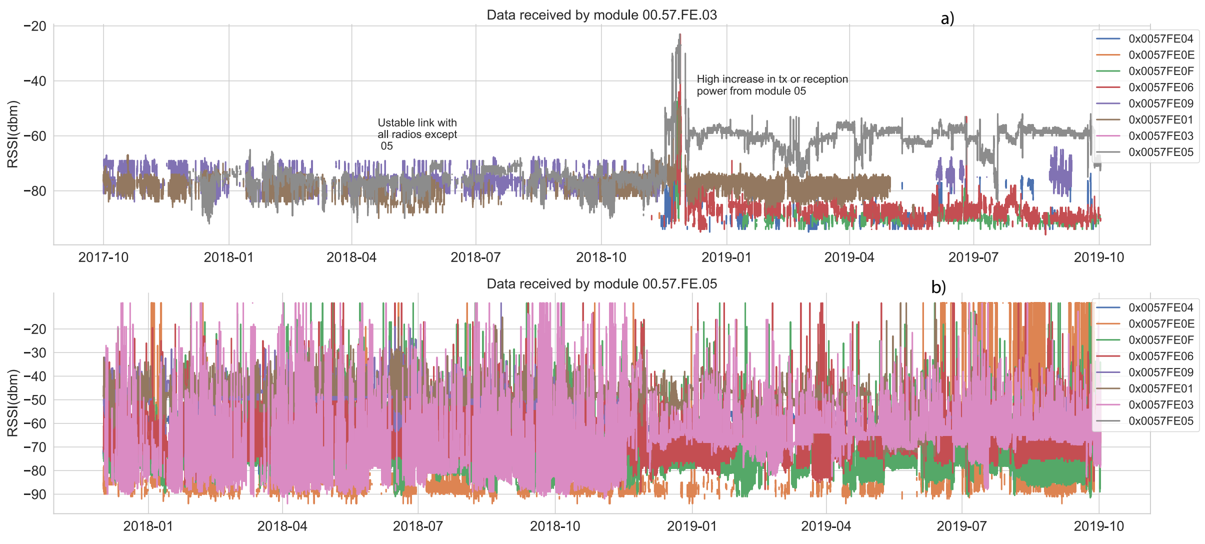


Source – Author.

data from radio 00.57.FE.05 (gray) since 2017-12 with some spike like power drops.

**Radio 00.57.FE.06 (red) Figure 43 d):** Stable link with radio 0F. Data from this

Figure 44 – RSSI data received from radios 03 and 05.



Source – Author.

module has a high power since they are in the same location, as was said before. There is good symmetry between what is transmitted and what is received from these two radios, however, communication between them presents many -9dbm spikes, which could be because of their location. Started to receive constant data from radio 00.57.FE.05 (gray) since 2017-12. Received data from 00.57.FE.01 (brown) until 2019-08 with the minimum possible power due to distance.

**Radio 00.57.FE.09 (purple) Figure 43 e):** The radio went offline in December 2018 until March 2019, then again in May. This was the most unstable radio, presenting a huge amount of spikes from all other radios except from 05 (gray), which presented high power and spikes free RSSI data.

**Radio 00.57.FE.01 (brown) Figure 43 f):** Receives constant data from 00.57.FE.05 (gray) with high power and stability, only some sudden degradations with recovery. It also receives data from the Coordinator with very low symmetry, around 8 dB of difference, and very low power. Transmission from radio 0E is also very noisy, presenting many -9dbm and EMI spikes.

**Radio 00.57.FE.03 (pink) Figure 44 a):** The radio has a constant link with radio 05 of around -75 dBm until 2018-11, where there was a power increase to -65 dBm, an increase of around 10 dB. After the power raise, data from 05 appears to be more consistent. Presents an inconsistent link with the rest of the modules.

**Radio 00.57.FE.05 (gray) Figure 44 b):** This module is the most consistent one. Receives data from all the other modules and transmits it to all. The difference with 03 (pink) is around 5 dB. The radio serves as a path to the Coordinator for the rest of the radios. The EMI and missing link spikes of radio 03 can be clearly seen in the RSSI received by this radio.

## 4.4 ANOMALY DETECTION

Since sensor nodes in WSNs are, in many cases, deployed in distant or unattended areas, sometimes it is very hard or not feasible to monitor all sensors. Therefore, sensors can be subjected to intruder attacks or unnoticed failures that can deteriorate the state of the network, these occurrences are identified as anomalies and are important to measure.

Anomalies or outliers are values that do not follow the same pattern as the majority of the observed data. An outlier is “an observation which deviates so much from other observations as to arouse suspicions that it was generated by a different mechanism” (HAWKINS, 1980). Outliers that are not the product of an error in measures or in the process or environment being measured, sometimes are classified as novelties. In the case of WSNs, an anomaly can indicate different events such as some equipment failure or soon to fail state, sudden changes in the environment, security attacks, etc (LUO; NAGARAJANY, 2018). Moreover, the main goal of some WSNs is to detect anomalies, such as ED networks.

Anomaly detection techniques can be implemented at the BS or at the cloud to not overload the sensors with more resource consuming tasks, however, this depends on the application. In ED networks, if there is not a necessity of transmitting the entire data and only recognize an event, then the detection can be made on edge. In cases where there is the need of detecting anomalies and also transmit the data to the BS, the detection of anomalies can be performed on edge, which mitigates network limitations and improves privacy and security, or at the cloud or BS to reduce the energy consumption of the sensors.

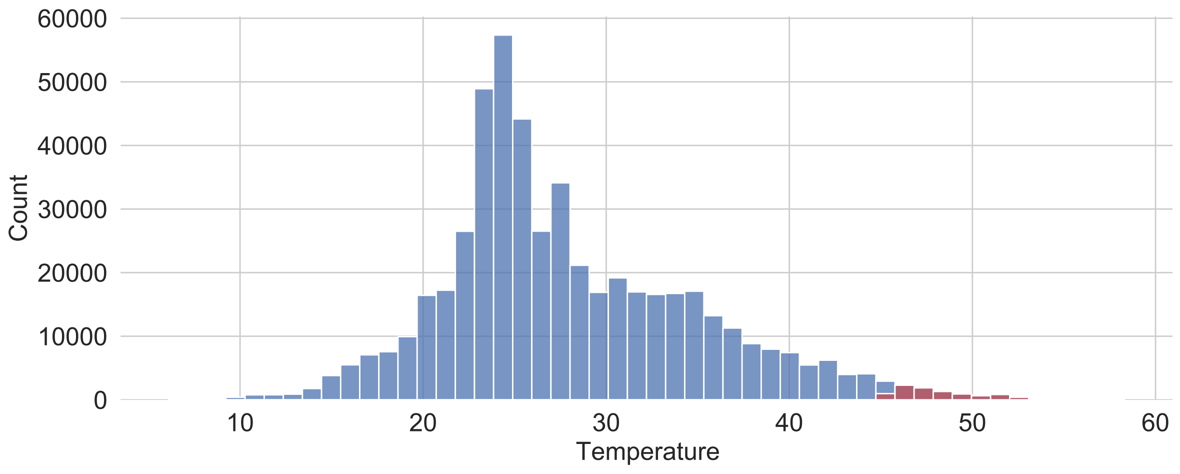
### 4.4.1 Statistical Methods Approach

Statistical rules can be applied very easily and fast to determine anomalies using a sample of the data. Some of these methods, such as Z-Score and Interquartile Range (IQR), are suitable for applying at the Gateway, where there is a larger concentration of data, and not on edge.

#### 4.4.1.1 Gaussian or Histogram Based

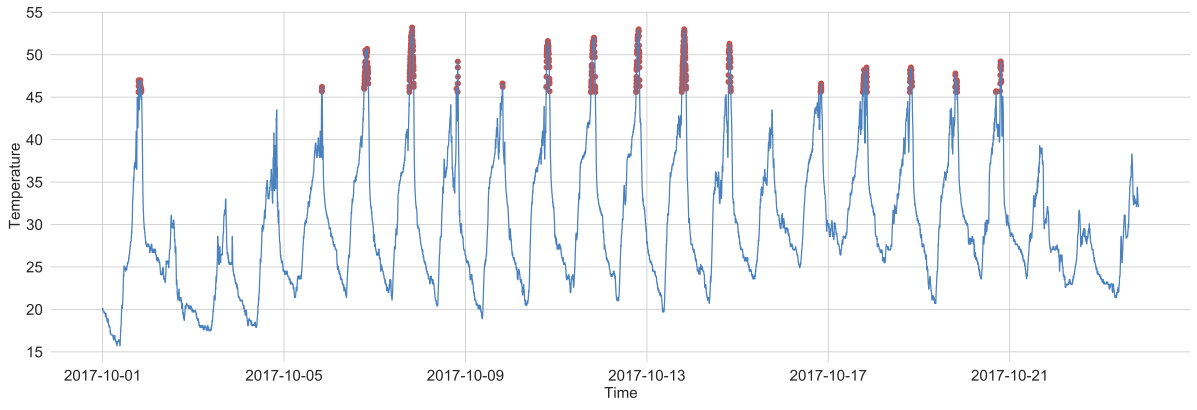
As the temperature is a variable measured by all sensors in the dataset it can be used as a baseline for an anomaly detection system at the WSN. If the data follows a Gaussian or Gaussian-like distribution, as the distribution in Figure 45, a possible solution to detect anomalies is to use the standard deviations as a cut-off for identifying outliers. A value that falls outside of three standard deviations is very rare to occur and can be considered an outlier, Figure 45 presents the distribution of the anomalies. Three or four standard deviations, for smaller datasets, is the common norm for identifying outliers in a Gaussian or Gaussian-like distribution. Figure 46 shows the points detected as anomalies in the time series, we can see it only detects high temperatures values.

Figure 45 – Distribution of Data and Outliers.



Source – Author.

Figure 46 – Anomalies in temperature detected by Histogram Based Method.



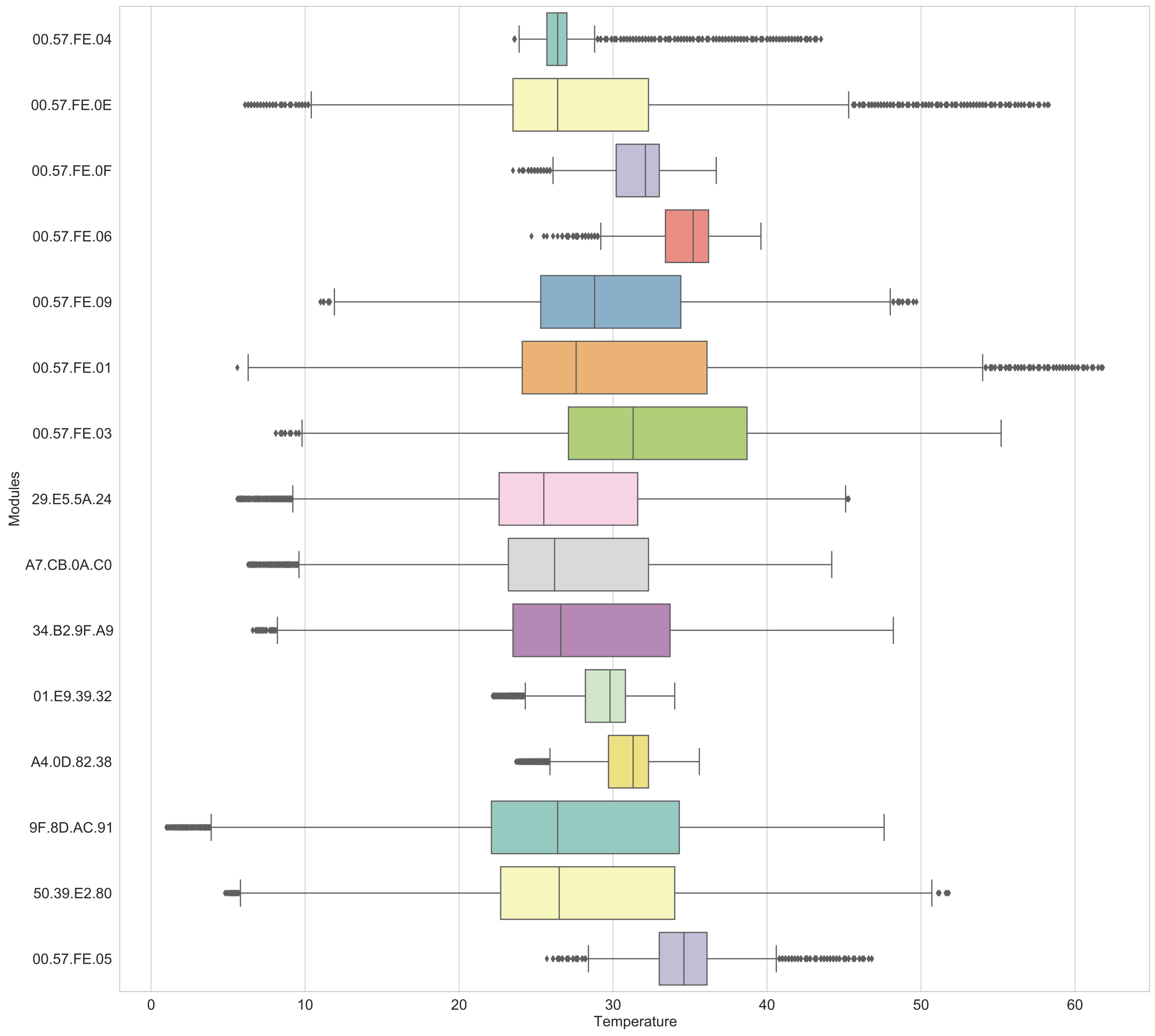
Source – Author.

#### 4.4.1.2 Interquartile Range Method

Not all data follows a Gaussian distribution. A method for summarizing non Gaussian distributions is the Interquartile Range. This method can also be used for outliers detection by defining a threshold that is a factor of the IQR above the 75th percentile or below the 25th percentile. The factor is commonly between 1.5 and 3. Three is used for detection of the extreme outliers, which are represented in the whiskers in a box plot.

Figure 47 shows a box chart of the temperature by module. It can be seen the number of extreme values in the whiskers, most of these values are considered anomalies. Figure 48, shows the temperature once the IQR method with a k factor of three was applied to the data. It can be seen how a considerable amount of outliers were removed.

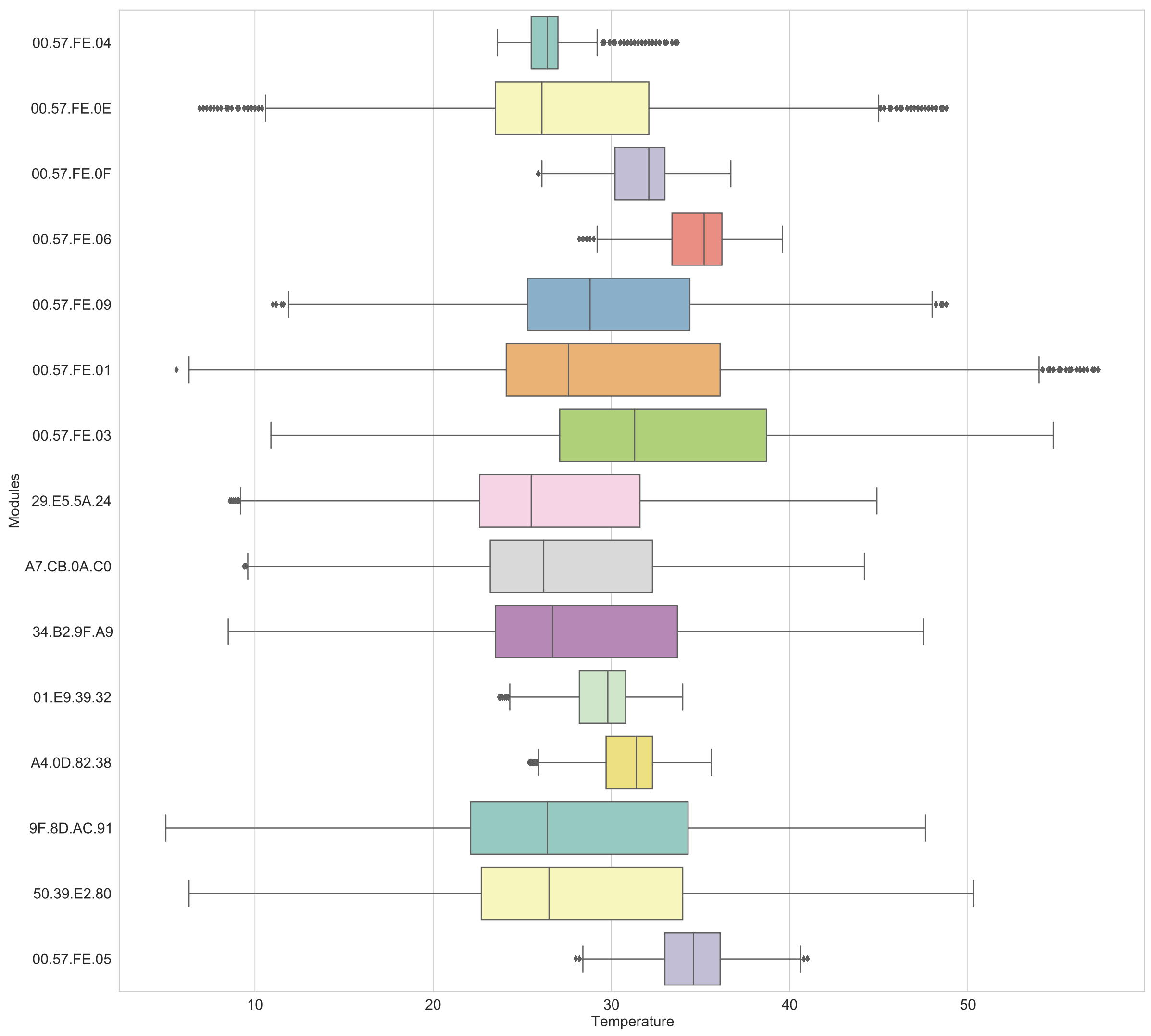
Figure 47 – Box Plot of Temperature by Modules with outliers.



Source – Author.

However, there is no insurance that all this data were in fact outliers. Therefore, a more accurate method of detection of outliers must be implemented. Table 2 shows some of the outliers detected in the original dataset.

Figure 48 – Box Plot of Temperature by Modules with outliers Removed.



Source – Author.

Table 2 – Sample of anomalies detected by the IQR method.

| **Timestamp** | **Module** | **Temp** | **VBus** | **VPanel** |
|---|---|---|---|---|
| 2019-07-07 05:58:44 | 34.B2.9F.A9 | 7.9 | 3.266 | 0.000 |
| 2019-07-07 06:03:57 | 34.B2.9F.A9 | 7.9 | 3.254 | 0.000 |
| 2019-07-07 11:08:56 | 34.B2.9F.A9 | 7.9 | 3.236 | 15.240 |

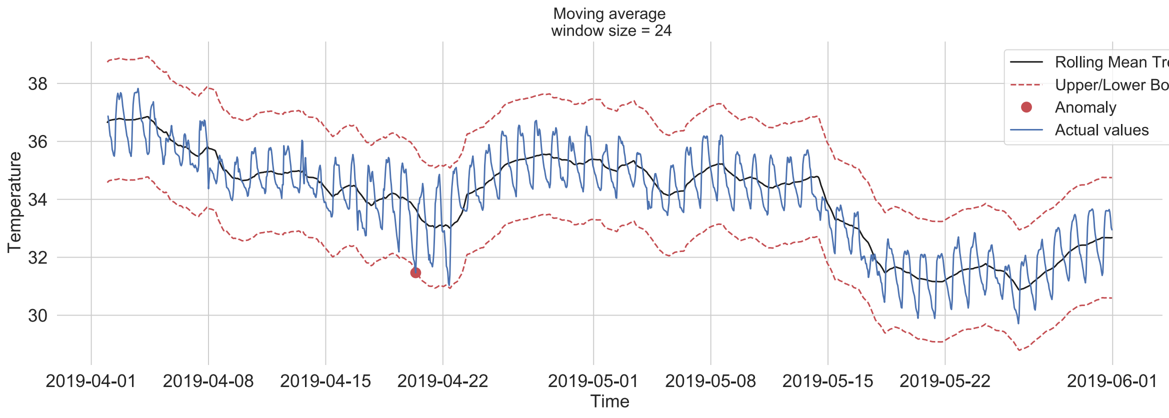
Source – Author

The temperature of 7.9 degrees is considered an outlier in module 34.B2.9F.A9, however, this temperature was recorded early in the morning and all the points in the vicinity are close to the value, giving the possibility of just a very cold day and not some random phenomenon.

#### 4.4.1.3 Moving Average Method

This method can be implemented on edge, in sensor nodes, because it only needs a small batch of data. The moving average method takes a window of points and computes the mean of the window. For a wider window, the smoother will be the series. In this case, the same analogy as before can be applied by taking the standard deviation by a factor of k as a threshold or confidence boundaries, then, any point outside these boundaries is considered an outlier. Figure 49 shows the rolling mean method on the temperature data with the borders defined. The point that falls outside the boundaries is then marked as an anomaly.

Figure 49 – Anomaly detection using the rolling mean.



Source – Author.

The first three methods are not very accurate because they do not discriminate between outliers and novelties and extreme values will always be considered as outliers. They are better for situations where there is a threshold or the interest is to find extreme points.

### 4.4.2 Machine Learning Methods

Conventional statistical methods are usually easier to interpret. Machine learning methods are sometimes more useful and perform better than statistical ones because they can model more complex data and detect more complex anomalies. Machine Learning methods can be in the form of Supervised Learning (e.g. Decision Tree, SVM, LSTM Forecasting), Unsupervised Learning (e.g. K-Means, Hierarchical Clustering, DBSCAN, Isolation Forest, LSTM Autoencoder).

Supervised anomaly detection depends on the previous labeling of the data as "normal" or "abnormal". Algorithms trained this way are usually more accurate because of the previous knowledge of how an anomaly looks like, or how normal data looks. One can label the data as "normal" for what is normal behavior and label the rest as "abnormal", and vice versa. This approach requires a profound knowledge of the data and an extensive labeling process.

Unsupervised anomaly detection is a more automated task where anomalies are identified from unlabeled data by assuming that only a small fraction of the data is anomalous. In this case, the standard deviation criteria is also used but applied to algorithms that can model better the data. These methods are usually less accurate than supervised ones, however, they require less knowledge of the data and can detect unknown abnormal behaviors. A common solution to improve the trust of this approach is to use Wisdom of the Crowds thinking, which is using two or three unsupervised anomaly detection algorithms and compare or combine the results.

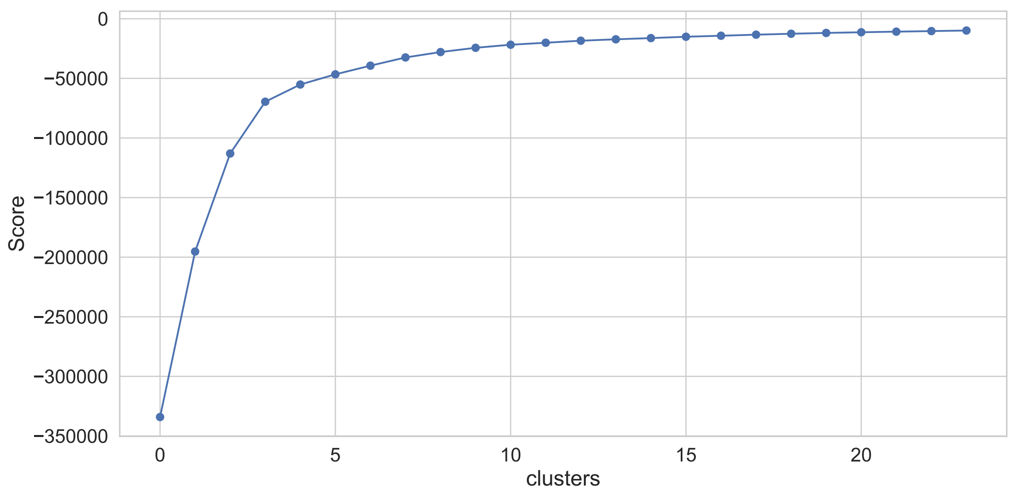
In order to test the Machine Learning methods, a sample of the data from November 2018 to January 2019 was taken, with a total of 23040 training examples and 8 features (Temperature, VBus, hour, day, month week, weekday, daylight).

#### 4.4.2.1 K-Means and PCA for collective anomalies

Principal component analysis (PCA) is commonly used for dimensionality reduction by projecting each data point onto a few components to obtain lower-dimensional data while preserving as much information of the data as possible. In this case, PCA was used to reduce the number of features in the dataset to two main components. Then, enters K-Means Clustering, which is an algorithm that successively divides the data into clusters minimizing a criterion known as the inertia or within-cluster sum-of-squares. A cluster is a group of points aggregated together because of the similarities they present. The points that are far from the cluster are points with an unusual combination of features. Those points can be considered anomalies.

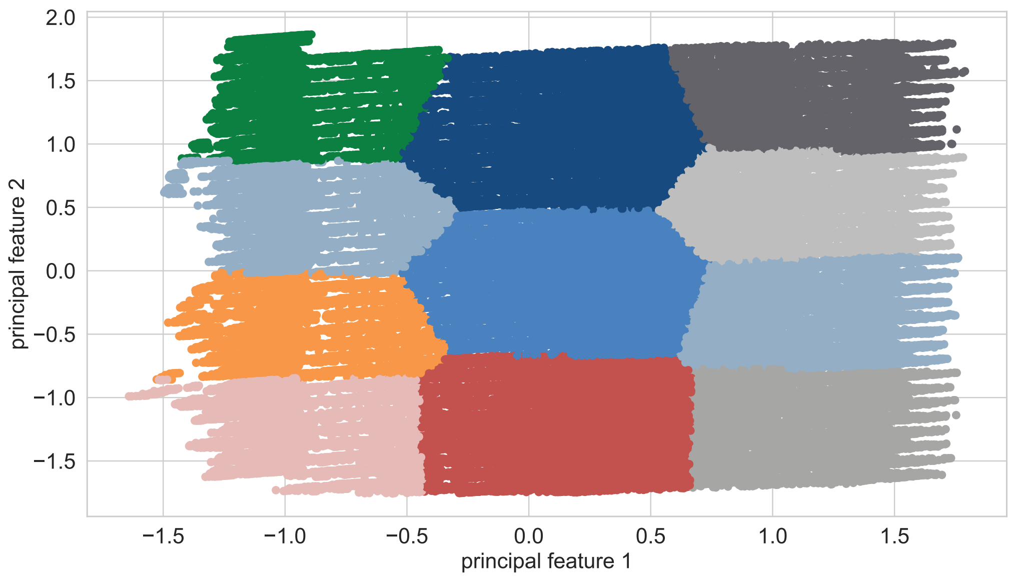
To implement K-Means, an optimal number of clusters must be chosen. The Elbow Method is used to select the number of clusters for the algorithm. While increasing the number of clusters, the difference between points inside clusters increases, and the differences between clusters decrease. The Elbow Method consists of finding the number of clusters that reduce the sum-of-squares close to zero. Figure 50 shows the results of the method. It can be seen that the curve stabilizes at around 10 clusters, which means that the addition of more clusters does not add a significant improvement in performance. Figure 51 shows the two features resulting from the PCA analysis divided into 10 clusters, after applying K-Means. Every color represents a cluster, and every cluster has a centroid. The assumption is that normal data will be closer to the centroid of the cluster than abnormal data.

Figure 50 – Elbow Method for K-Means clustering.



Source – Author.

Figure 51 – Principal components divided into 10 clusters.

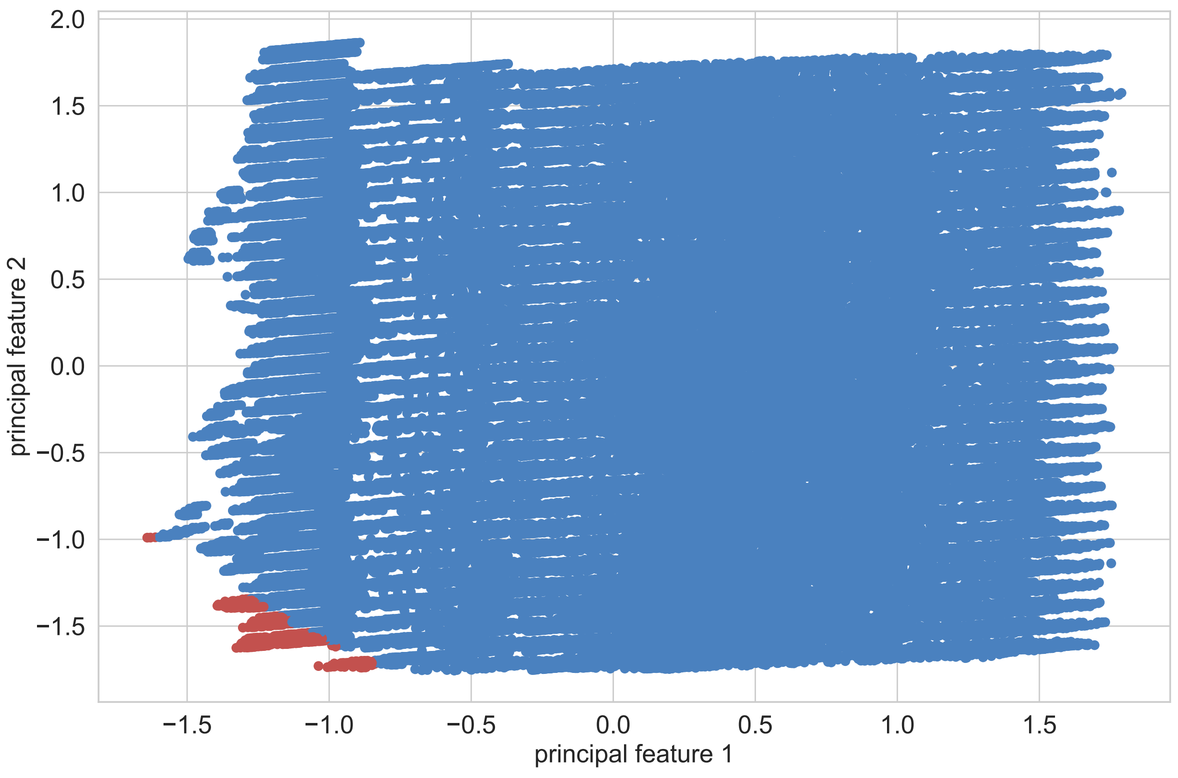


Source – Author.

In order to determine the anomalies, the distance from each point to its nearest centroid is computed, considering the longer distances as not normal points. An outlier fraction was assumed as a hyper-parameter to provide information about the proportion of anomalies expected (normal values are between 0.01 and 0.1). The threshold was considered as the minimum distance of the outliers. After calculating the distance from the points to the closest centroid, clusters were further divided into two clusters, representing the normal data and the abnormal (points with higher distances from the centroid). Figure 52 shows the anomalies detected by the method in a cluster form, red points are those detected as anomalies.

The advantage of using this method is similar to other machine learning techniques that allow introducing many features to improve the model performance, also, setting an outlier fraction allows to make the algorithm more sensitive to anomalies. However, the hyper-parameters must be selected carefully because it can lead to many false positives. A large number of clusters makes the algorithm very slow and consumes high computational resources, for example, running the same algorithm with 25 clusters took around two hours and using 10 clusters around 6 minutes. The algorithm recognizes anomalies mostly at lower temperature values, as is shown in Figure 53, and does not take high temperature values as outliers.

Figure 52 – Visualization of anomalies in clusters.

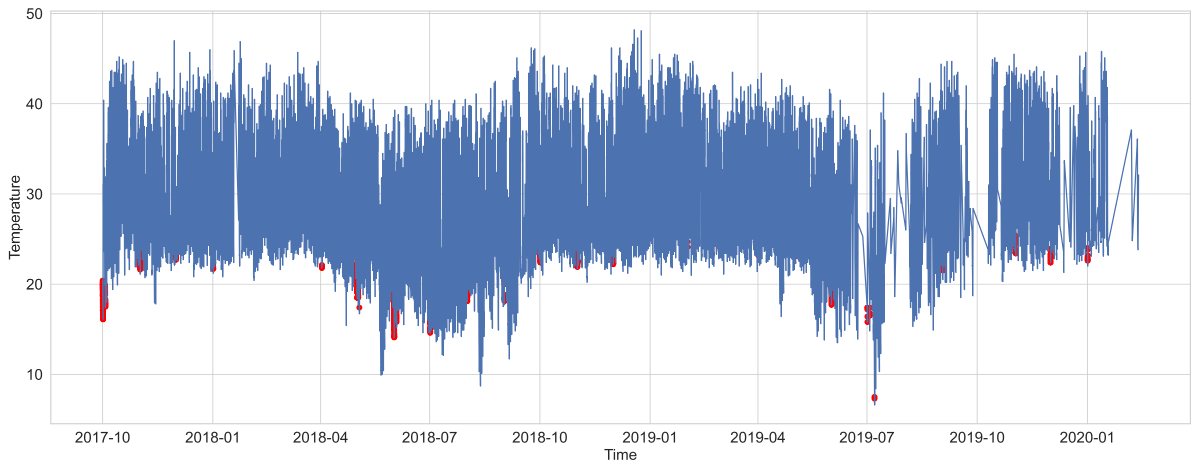


Source – Author.

#### 4.4.2.2 Isolation Forest for collective anomalies

Isolation forest is another machine learning algorithm, based on the Decision Tree algorithm, used for detecting anomalies in an unsupervised way. The idea of anomaly detection algorithms is to define what is "normal" data and select the rest

Figure 53 – Anomalies detected using K-Means represented as time series.



Source – Author.

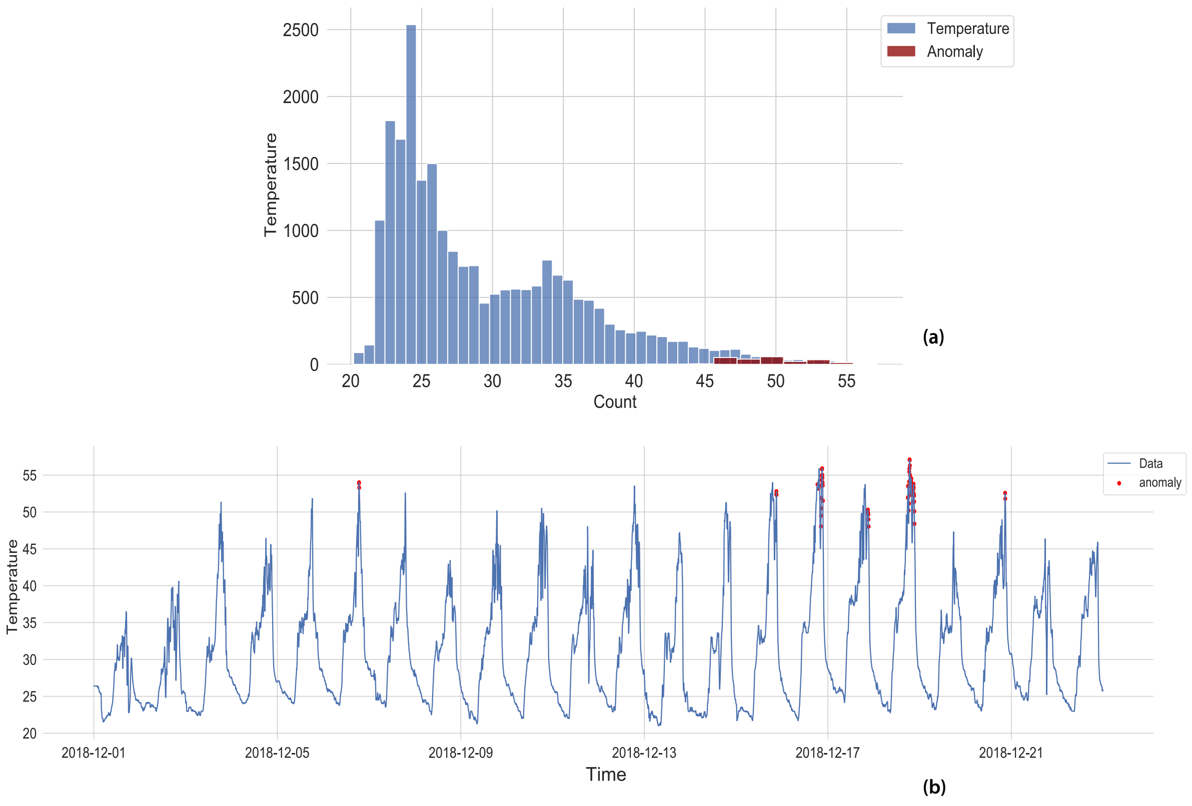
of the data as anomalous. However, isolation forest is based on the assumption that anomalies are very few when compared to the rest of the data and in different forms. Based on this, Isolation Forest recursively partitions the data by randomly selecting a feature and randomly splitting it, assuming that isolating the anomaly will take fewer random partitions compared to normal points. Figure 54(a) shows a stacked histogram that represents the distribution of the data and the anomalies that the isolation forest find. Figure 54(b) shows the anomalies as points in the time series. From the images, we can see how the Isolation forest algorithm detected as anomalies high value points only.

#### 4.4.2.3 LSTM with multivariate features for sequential anomalies

The idea of using LSTM networks or any other neural network is to make a prediction of the next timestamp or a set of timestamps and compare the prediction with the actual value, measuring an error that will be compared with a threshold. The threshold can be a factor of the standard deviation plus the mean of the prediction errors in the training set. Then, if the error between the prediction and the actual value passes a certain value, the point can be considered as an anomaly.

These ANN methods are used to inspect anomalies in a certain window of time and not in a bigger set of values and have the advantage of finding complex relationships in multivariate time series data. The system looks to the trends of the Temperature, Voltage, day, hour, daylight, month, and two more features in the case of sensors. The data is divided into training and testing sets. The training set is divided into batches of 64 time steps. An LSTM network was created with two staked LSTM layers,

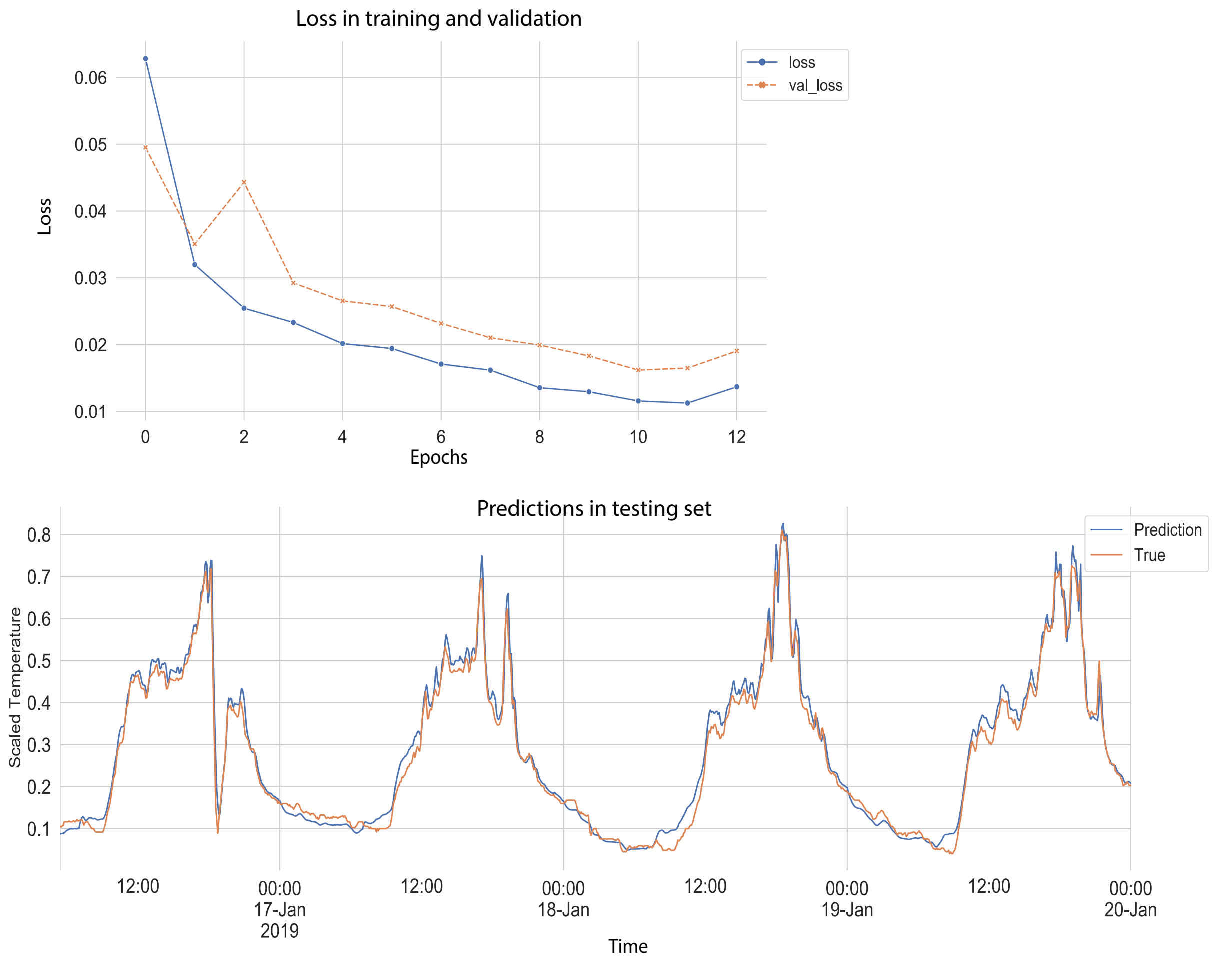
Figure 54 – Anomalies detected in the temperature data with Isolation Forest. (a) Distribution of Anomalies. (b) Anomalies in the data.



Source – Author.

with 64 units each and a final Dense layer, using the Mean Absolute Error (MAE) as loss. The network was trained using callbacks for monitoring the validation loss and with a batch size of 256. The training set has 21824 examples and the validation set 1088. Figure 55 shows the training loss and the validation loss in training and a comparison of the predictions and the testing set actual values below.

Figure 55 – Training and validation loss and predictions on testing set.

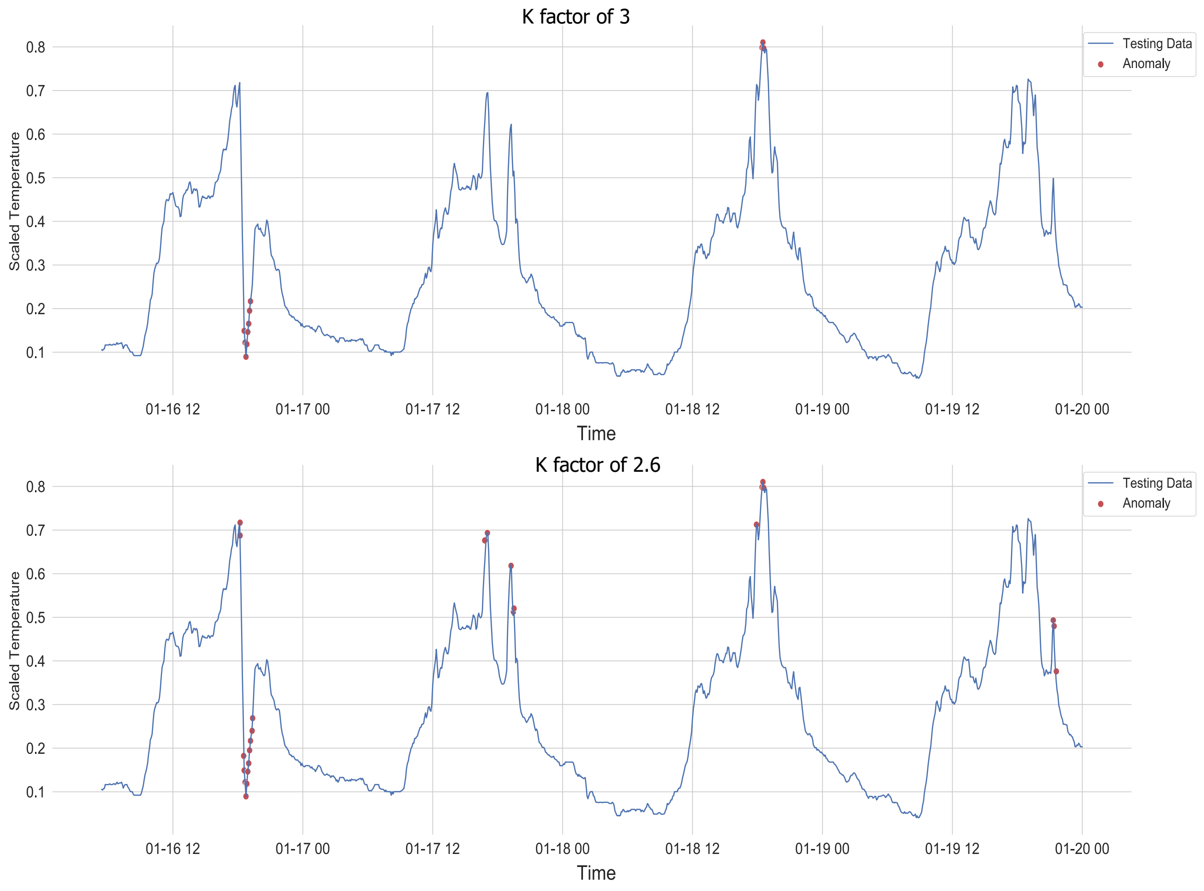


Source – Author.

Figure 56 shows the anomalies detected by the LSTM algorithm. In order to set a threshold, the error during training was computed. It can be seen that the algorithm when selecting a factor of 3 standard deviations from the mean, recognizes the hard fall between 01-16 and 01-17, which can be seen as an anomalous behavior and the highest point temperature value.

This method is more robust to false positives, however, it all depends on what should be considered abnormal. Another behavior that could be considered abnormal is the rise of temperature at the end of the day, such as the double spikes between 01-17 and 01-18 and the final spike close to 01-20. These points can be detected by the algorithm by reducing the value of the threshold, for example to 2.6 as presented in Figure 56, making the algorithm more sensitive to outliers.

Figure 56 – Anomalies detected using LSTM networks with k factor of 3 and 2.6.



Source – Author.

## 4.5 CONCLUSIONS

An exploration of the dataset was made, to help better understand the dataset and the relationship between features. Using visualizations it was possible to provide an understanding of which radios and sensors are more stable regarding the amount of missing data and data received/transmitted. The RSSI dataset was also analyzed, allowing us to see what radios share a constant link and how good is the link. Strange behaviors and degradation in the RSSI measures from some radios were found, which could be correlated to physical movements in the area. An anomaly detection analysis was performed using basic and machine learning techniques. The analysis allowed the detection of strange behaviors in the temperature data, such as unexpectedly high and low values, and sudden temperature drops during the day or spikes at night, as Figure 56 presents. The reason for these spikes and drops could be correlated to some events taking place at the location of the sensor.

# 5 MULTIVARIATE DATA PREDICTION IN THE WSN

## 5.1 INTRODUCTION

A solution for the problem of energy conservation using data prediction is presented in this chapter using a Deep Learning strategy in a DPS. The structures of the models are presented together with their parameters. All models are compared using different error metrics when forecasting. Finally, a measure of the reduction in data transmission is given using the model with the best performance and considering different error thresholds. Results show that the best model can save a considerable amount of data in transmission and still maintain a good representation of the measured data.

## 5.2 PROBLEM FORMULATION

Our goal in is to forecast multiple time series data coming from a sensor. Formally, given a time series representing a sensor signal $X = \{x_1, x_2, ..., x_T\}$ where $x_t \in R^n$, the task is to predict a series of future sensor values $x_{t+h}$ where $h$ is the number time steps ahead (horizon) of the current time. In order to predict the future values in a time series, we assume the previous are available.

After the training is performed, a process of iterative prediction is made over $\{x_1, x_2, ..., x_T\}$ to obtain the future values $\{x_{T+1}, ..., x_{T+h}\}$. Then a new prediction is made over the historical data and the previous predicted steps. The process is then continuously repeated. The set $\{x_{T+1}, x_{T+2}, ..., x_{T+h}\}$ is the output of every prediction step and is defined as $y = \{y_1, y_2, ..., y_h\}$.

In order to evaluate the quality of the predictions and compare the models, different evaluation metrics are used, including Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and R-square. Details of these metrics are specified in Section 5.4.1.

## 5.3 DATA PREPARATION AND MODELS ARCHITECTURE

Three time series from one sensor node were considered for the task of forecasting. Each time series corresponds to a physical parameter (temperature, voltage, and MPPT voltage) over 80 days with a 5 minutes period.

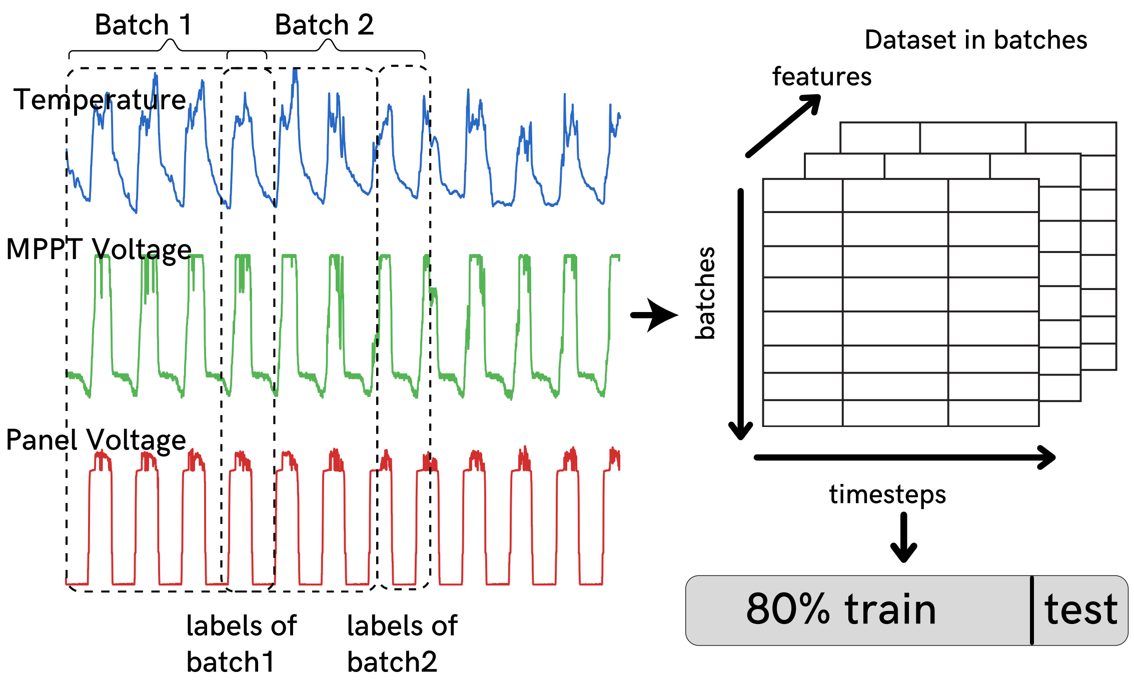
In order to apply DL forecasting methods, the data was divided into batches of (time steps, features) using a rolling window method to create a supervised training dataset. In the case of multiple-step-outputs, the model predicts multiple future time steps, therefore the rolling window is of the size of the forecast horizon, see Figure 57.

The dataset was split in 80% for training and 20% for validation, resulting in 16358 samples for training and 4090 for testing, and was first scaled using the maximum and minimum values of each feature, as presented in the following transformation:

$$X_{scaled} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{18}$$

Where $X_{max}$ is the maximum value encountered and $X_{min}$ the minimum of each time series. This transformation is needed to avoid explosion of the gradient.

Figure 57 – Data Preparation steps.



Source – Author.

An LSTM, GRU, DNN, 1D-CNN, Seq2Seq and a Seq2Seq with Attention were trained with the same data to compare the results. All neural networks were created and trained using TensorFlow 2.0 (ABADI et al., 2016).

The length of the input sequence and the batch size were set to 290 and 245 respectively. The first contains the observations of one day and the second one is a divisor of the training set. For all models was chosen the architecture considering a trade-off between simplicity and performance. Moreover, the training process was programmed for 200 epochs and monitored to stop when the performance on the validation set started to diminish, considering a certain tolerance. To achieve this was used the early stopping function implemented in Keras API. The efficient Adam version of the Stochastic Gradient Descent (SGD) was used as optimizer for all the models. The Mean Squared Error (MSE) was used as loss function and MAE as a metric for validation. The final layer of all models consists of a Dense layer with three output neurons for the multivariate predictions. All the hyperparameters were selected after experimenting with a range of different values, keeping the ones that made the models reached a better performance.

a) **LSTM Model:** The LSTM model consists of two stacked LSTMs layers respectively with 128 units and a fully connected layer (Dense) with 120 units, dropout of 0.2, and learning rate of 0.01, defined after different trials.

b) **GRU Model:** The GRU model consist of two stacked GRUs layers respectively with 128 units with also a fully connected layer of 120 units, dropout of 0.2, and learning rate of 0.01.

c) **DNN Model:** The DNN model is composed of two Dense layers with 128 units, a dropout layer of 0.2, a Max Pooling layer with pool size of 2, a Flatten layer to convert the pooled feature map into a vector and pass it to a final Dense layer with the output neurons.

d) **CNN Model:** The CNN model consists of two 1D CNN layers, followed by a dropout layer serving as regularization, a Max Pooling layer, and two FCLs. It is very common to use CNN in groups of two so the model can learn more features from the input data. CNNs are very fast at learning, which is why its good to use a dropout layer to slow down this process and avoid overfitting.

   The learned features are transformed into a vector using a flatten layer to serve as inputs to a fully connected layer. The use of this intermediate FCL is to serve as an interpreter of the learned features before passing them to the output layer. A standard configuration of 128 feature maps with a kernel size of 3 and dropout of 0.2 was used.

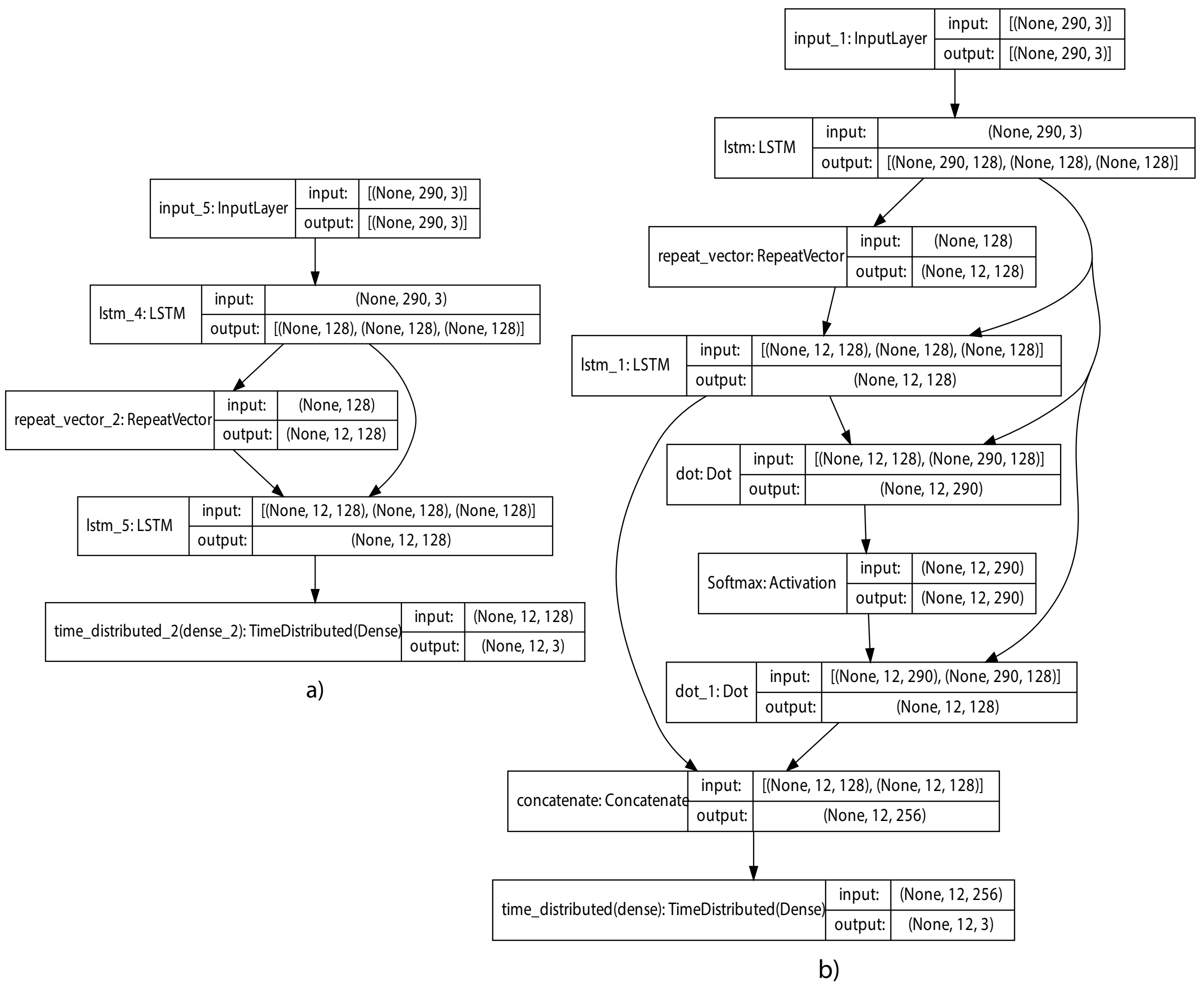
e) **Seq2Seq Model:** A Seq2Seq model without attention was also trained for comparison. Both Seq2Seq models consist of LSTM networks for the encoder and decoder with 128 units and a dropout of 0.2. The final layer is a fully connected (Dense) layer with three output neurons for the task of multivariate prediction. A learning rate of 0.01 was chosen because it showed better results.

   The decoder in the Seq2Seq model receives the last cell state and hidden state from the encoder as the initial state. The last hidden state of the encoder is also copied as many times as the number of time steps (Repeat Vector layer), and each copy is fed to the decoder LSTM cell together with the previous cell state and hidden state. The decoder outputs a hidden state for every time step which is connected to a Dense layer.

f) **Seq2Seq+Attention Model:** For the attention model, we need all the hidden states from the encoder to the Alignment Score calculation. A Repeat Vector layer is also used to repeat the last hidden state as many times as the number of time steps, then we created the Alignment Score using a Dot layer and applying a Softmax on it, as stated in Luong Attention, then we concatenate the decoder hidden states with the context vector in a layer which is used as input to a fully connected output layer. The diagram of both models, (e) and (f), can be seen in Figure 58.

An important consideration that must be made, is that in the cases of the LSTM, GRU, CNN and DNN models, predictions are made in the form of single-step prediction, since our goal is to predict multiple time series at the same time from a sensor node. For the Sequence to Sequence (Seq2Seq) and the Attention model, predictions can be made in the form of multiple-steps-ahead fashion, forecasting, in this case, the next 12 points of the time series, that correspond to the next hour of measures.

Figure 58 – Flow Diagram of Seq2Seq Models. a) Seq2Seq model flow diagram. b) Attention model flow diagram.


input_5: InputLayer | input: [(None, 290, 3)] | output: [(None, 290, 3)]
lstm_4: LSTM | input: (None, 290, 3) | output: [(None, 128), (None, 128), (None, 128)]
repeat_vector_2: RepeatVector | input: (None, 128) | output: (None, 12, 128)
lstm_5: LSTM | input: [(None, 12, 128), (None, 128), (None, 128)] | output: (None, 12, 128)
time_distributed_2(dense_2): TimeDistributed(Dense) | input: (None, 12, 128) | output: (None, 12, 3)
a)
input_1: InputLayer | input: [(None, 290, 3)] | output: [(None, 290, 3)]
lstm: LSTM | input: (None, 290, 3) | output: [(None, 290, 128), (None, 128), (None, 128)]
repeat_vector: RepeatVector | input: (None, 128) | output: (None, 12, 128)
lstm_1: LSTM | input: [(None, 12, 128), (None, 128), (None, 128)] | output: (None, 12, 128)
dot: Dot | input: [(None, 12, 128), (None, 290, 128)] | output: (None, 12, 290)
Softmax: Activation | input: (None, 12, 290) | output: (None, 12, 290)
dot_1: Dot | input: [(None, 12, 290), (None, 290, 128)] | output: (None, 12, 128)
concatenate: Concatenate | input: [(None, 12, 128), (None, 12, 128)] | output: (None, 12, 256)
time_distributed(dense): TimeDistributed(Dense) | input: (None, 12, 256) | output: (None, 12, 3)
b)

Source – Author.

## 5.4 RESULTS AND DISCUSSION

### 5.4.1 Comparison

A model evaluation was provided to better understand and compare the performance of the forecasting models. The comparison of the performances was made by using three different error metrics in a multi-step scenario, Root Mean Squared Error (RMSE), MAE and R-Square ($R^2$), defined equations 19,20 and 21 respectively. RMSE is useful to measure the stability of the forecast and is sensitive to outliers. MAE measures the average of the forecast error values, where all the values are forced to be positive, it is also very robust to outliers. It is very useful for training datasets corrupted by many outliers. The R-square metric represents how much of the variance for a dependent variable is explained by an independent variable or variables. An $R^2$ score of 0.50 means that approximately half of the observed variation can be explained by the model. As higher the $R^2$ coefficient, the higher the correlation between real values and forecast is.

$$\mathrm{MAE}(y,\hat{y}) = \frac{1}{n}\sum_{i=0}^{n-1}\left|y_i - \hat{y}_i\right| \tag{19}$$

$$\mathrm{RMSE}(y,\hat{y}) = \sqrt{\frac{1}{n}\sum_{i=0}^{n-1}(y_i - \hat{y}_i)^2} \tag{20}$$

$$R^2(y,\hat{y}) = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y}_i)^2} \tag{21}$$

Every model was trained 20 times and evaluated in the testing set using the defined metrics. Table 3 presents the averaged prediction scores of the models when performing iterative predictions over the normalized testing data. Figure 59 shows a graphic representation in a bar chart of the MSE, RMSE, MAE, and R-Square of the models.

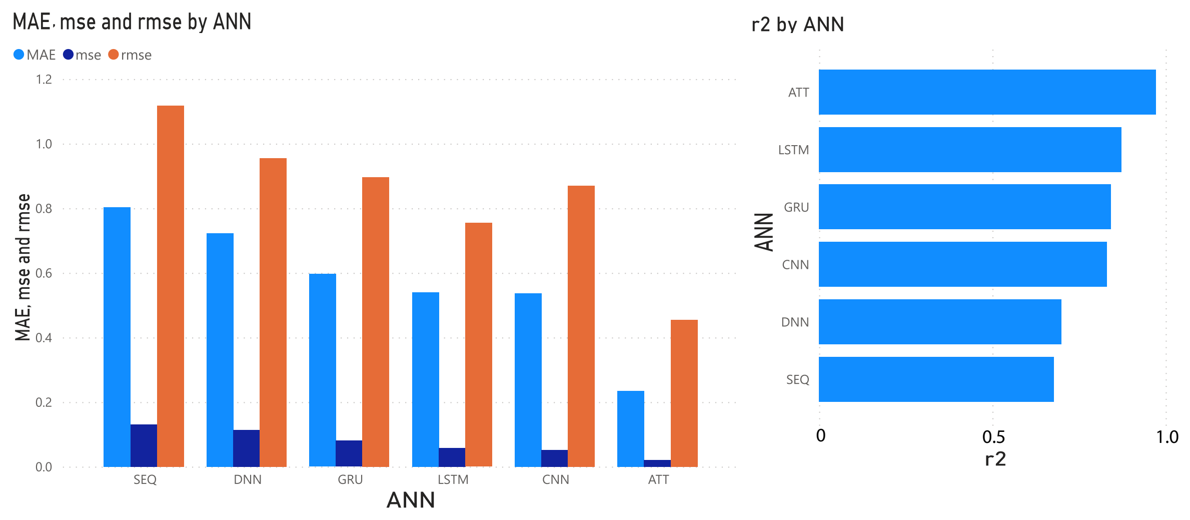
Table 3 – Predictions evaluations on temperature data.

| **Model** | **MAE** | **RMSE** | **R-Square** |
|---|---|---|---|
| LSTM | 0.054004 | 0.075457 | 0.913425 |
| GRU | 0.058296 | 0.083487 | 0.851738 |
| CNN | 0.053656 | 0.08737 | 0.831054 |
| DNN | 0.072263 | 0.095527 | 0.727253 |
| Seq2Seq | 0.115742 | 0.171256 | 0.750568 |
| Seq2Seq+Attention | 0.023398 | 0.045421 | 0.958567 |

Source – Author

It can be seen in Figure 59 that the model with better performance by every metric was the Attention model. The Seq2Seq without attention falls behind the others,

Figure 59 – Error comparison by ANN.

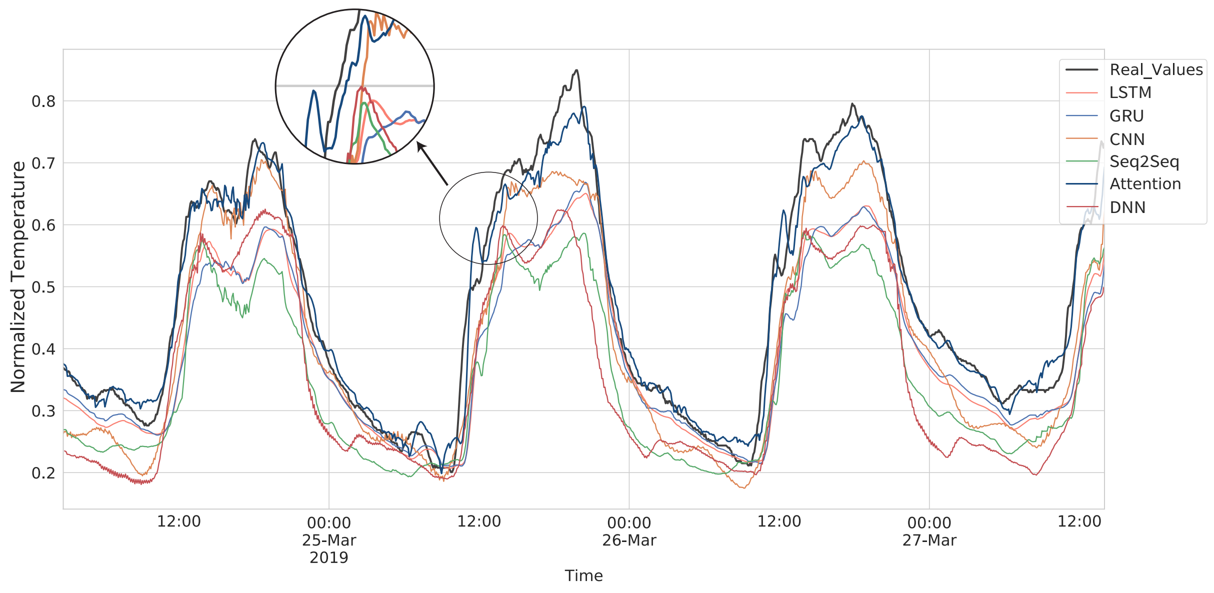


Source – Author.

being the less accurate one. This is because it has to compress all the information into a single vector, which makes it weaker to long sequences.

Figure 60 shows a comparison between the predictions made by the models and the real temperature values. The comparison shows how the values predicted by the Attention model are closer to the real ones.

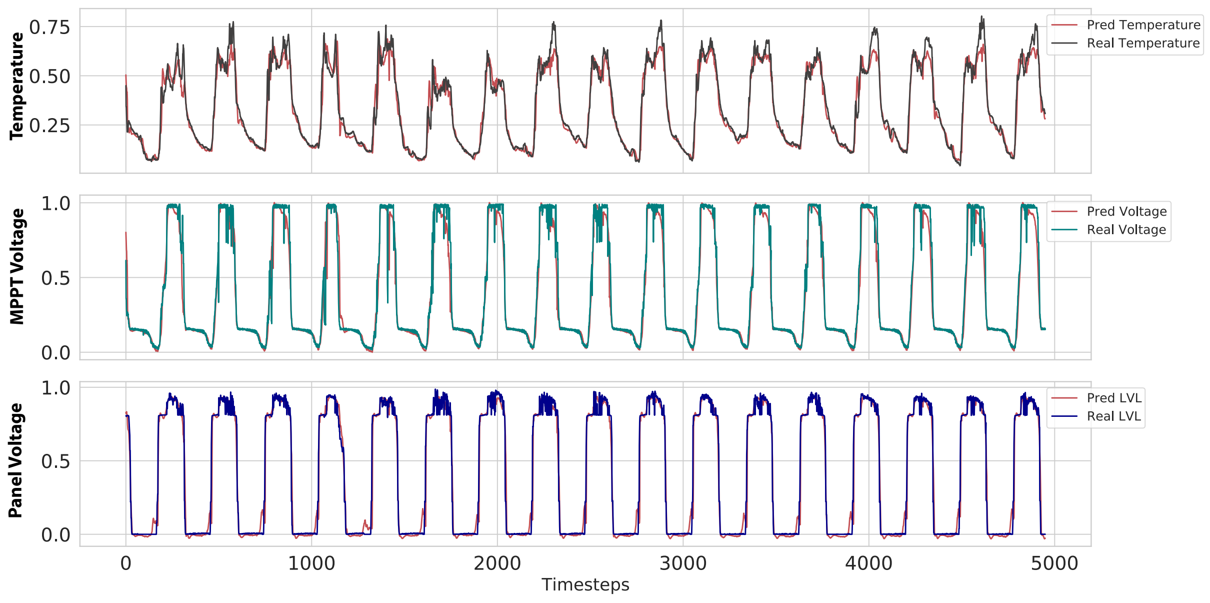
Figure 60 – Predictions and measured values.



Source – Author.

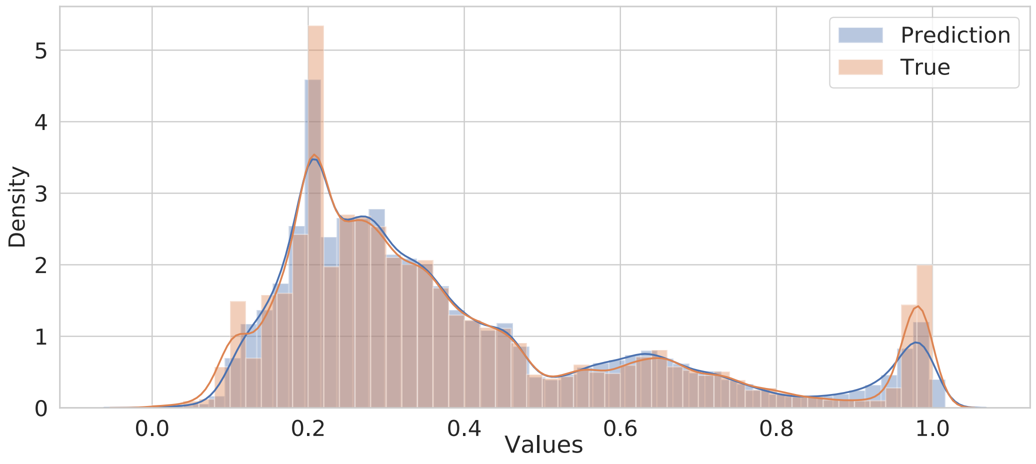
Figure 61 shows the prediction made with the attention model in the testing set of the three time series, Figure 62 shows the error distribution of the real temperature values and the predicted by the model, we can see there how close the predictions are from the truth.

Figure 61 – Predictions over Testing Data with Attention.



Source – Author.

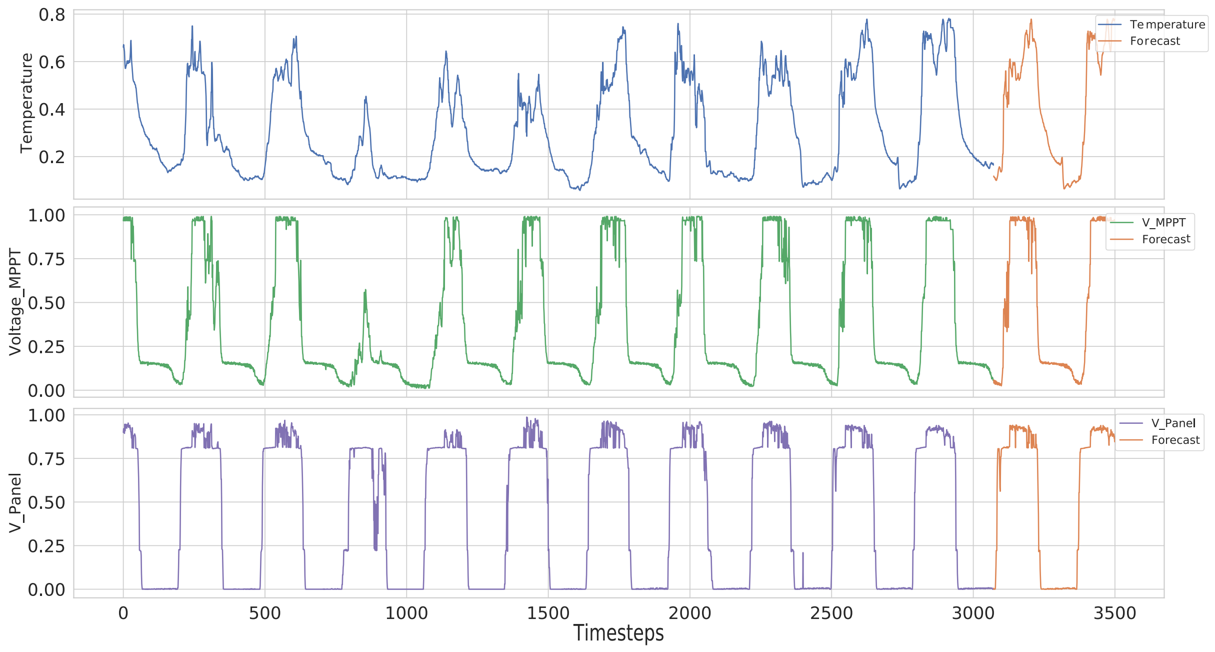
Figure 62 – Distribution of Predictions and Real Values.



Source – Author.

Figure 63 is a forecast of one and a half days ahead, using only the points predicted by the model to see how well it understands the time series.

Figure 63 – Multivariate Forecast of Scaled Time Series.



Source – Author.

### 5.4.2 Forecasting and Transmitting

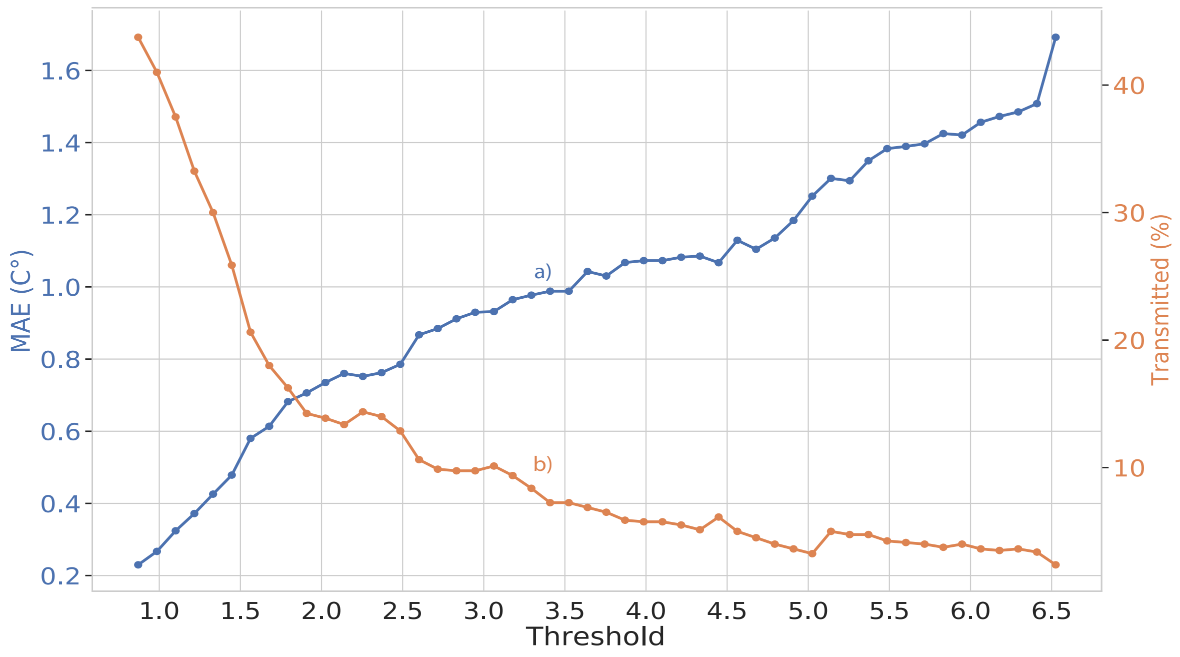
In a dual prediction scenario, the sensor and the BS share the same model and start forecasting from the same point. The sensor performs a prediction every time a measure is made and will transmit only if the error between the predicted value and the sensed one is bigger than some considered threshold. The point transmitted by the sensor is used in the next prediction. The advantage of performing a multiple-step-ahead forecast is that nodes need to make fewer predictions in time (one prediction every one hour), possibly saving more energy than with single-step-ahead forecasts.

To see the effect in transmission of different transmission thresholds, the model was used to forecast three days while measuring the error between the prediction and the real value using different thresholds. In the left axis of Fig. 64 is the MAE between the real temperature values read by the sensor and the entire forecast period when using different thresholds, whereas in the right axis the percent of points transmitted from the sensor to the BS is plotted.

We can see how for smaller thresholds the error between the real data and the predicted is smaller because the sensor has to transmit more points to help predictions be more accurate, however, we can see that for a small error of 0.25 degrees, and a threshold of 1 C$^\circ$, we can save approximately 60% of transmission.

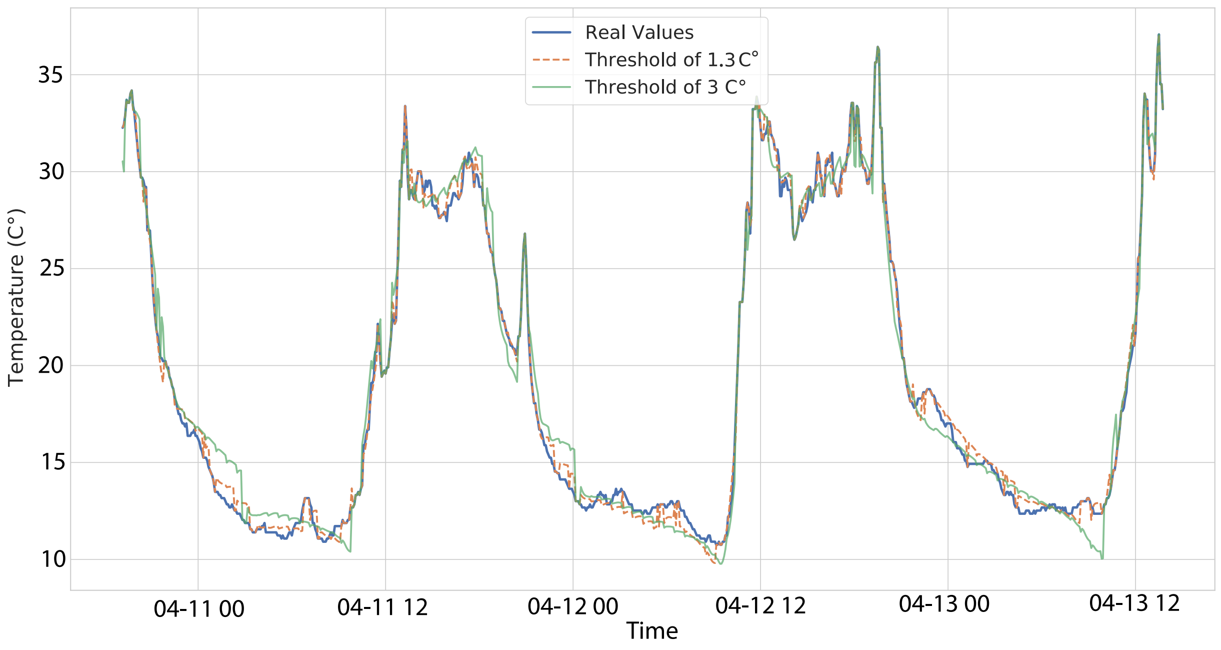
Fig. 65 shows a comparison between the real temperature data and two final predictions from close, one with a small threshold of 1.3 C$^\circ$ and 30.62% of transmitted

Figure 64 – Prediction performance by threshold. a)MAE per thresholds b)Percent of points transmitted per threshold.



Source – Author.

Figure 65 – Forecast of temperature with different thresholds.



Source – Author.

points and another with a threshold of 3 C° and 10% of transmitted points. We can see that, with a maximum error of 3 C° the predicted data still captures the patterns and the abrupt changes of the real data, saving up to 90% of transmission. Figure 66 shows the predictions and the real values of all the predicted time series with even more accuracy, transmitting around 40% of data.

### 5.4.3 Multi Channel Model for Single Prediction Schemes and BS Predictions

Besides traffic and energy reduction, predictions in WSNs can also be useful to recover abnormal or lost data, in case the sensor fails. This recovery can be performed either in the CHs or leader nodes or at the BS. In both cases, both devices are responsible for gathering information from a set of sensor nodes. In WSNs, sensor nodes monitoring the same area can present spatial-temporal correlation, which can be used to improve the prediction error.

Sensor nodes can often gather redundant data from the environment when they are deployed in the same area. In the studied WSN, there is a sensor node monitoring the water temperature of the dam and the environment temperature, and another sensor, also located at the dam that monitors the temperature of the solar panels and also the environmental temperature, then it is highly possible that there is some data redundancy and correlation there.

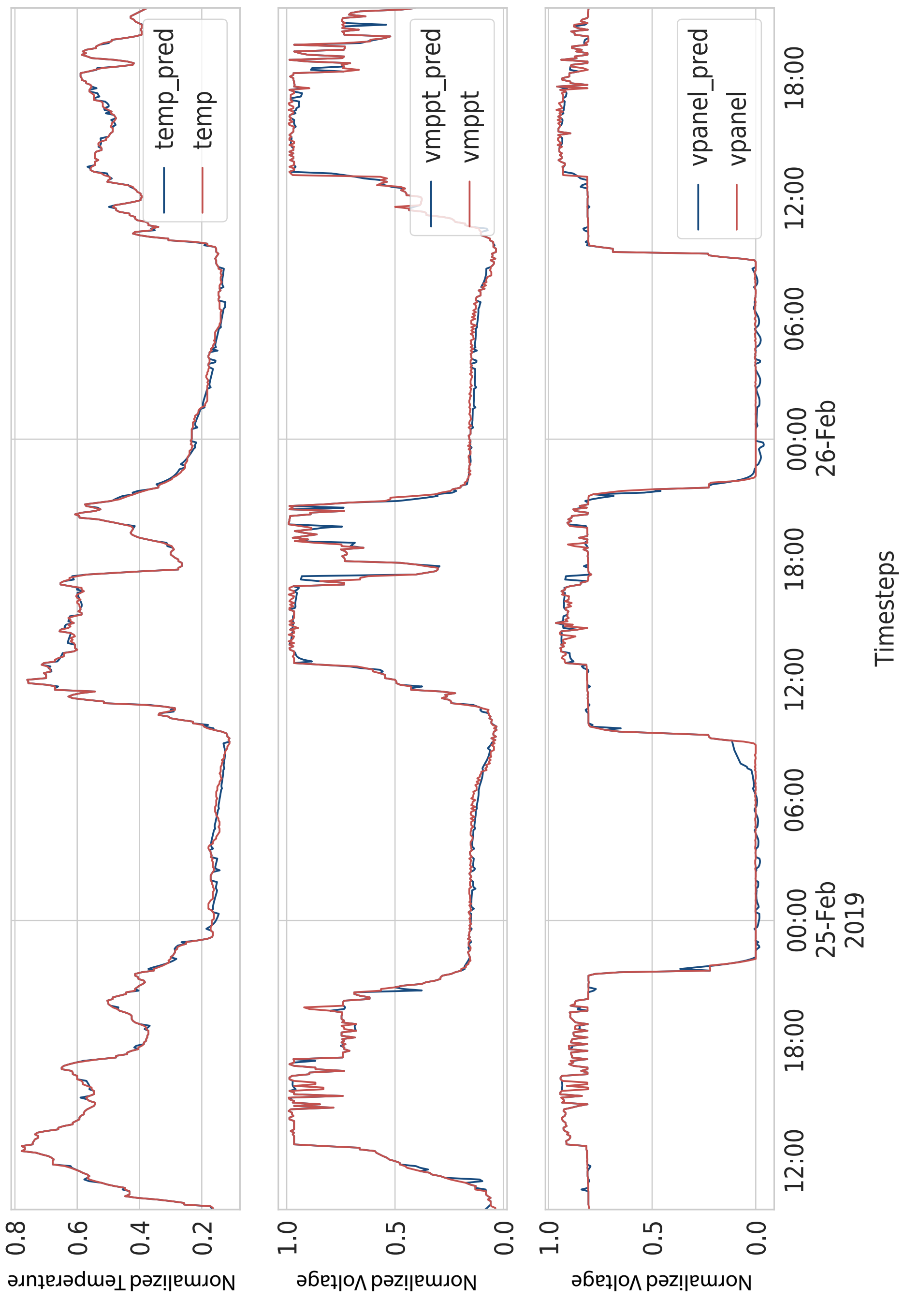
As explained before, classical data prediction methods are not capable of fully extract such relationships and abstract features. However, with Deep Learning methods, it is possible to fully combine the features from multiple inputs into a single model by merging all the independently learned relationships into a single layer and exploiting the spatial-temporal correlations, improving predictions.

Different types of sensory data, from two different sensors, are used for multivariate data prediction. Sensors are located at Montante UG5 location. One of the sensors (*34.B2.9F.A9*) measures Temperature, MPPT Voltage, and Voltage from a solar panel, and the other (*29.E5.5A.24*) also temperature, voltage, and temperature from a PT100 sensor in the dam, Figure 67 shows the time series from these sensors, which are the inputs for the multi channel model.

The model receives the same amount of input timesteps and outputs three time series. In this case, we are using the data from the second sensor to help improve the prediction from the first one. Figure 68 presents the architecture of the model. The firsts layer represents the inputs and the next ones are the same as the LSTM model used before, until the *concatenate* layer. This layer is used to merge the parameters learned by the two LSTMs networks into a single layer. Then, a FCL was used to give the network a chance to better understand the learned parameters and finally the output layer, with three output neurons for multivariate prediction.

As expected, the performance of the LSTM model improved considerably using

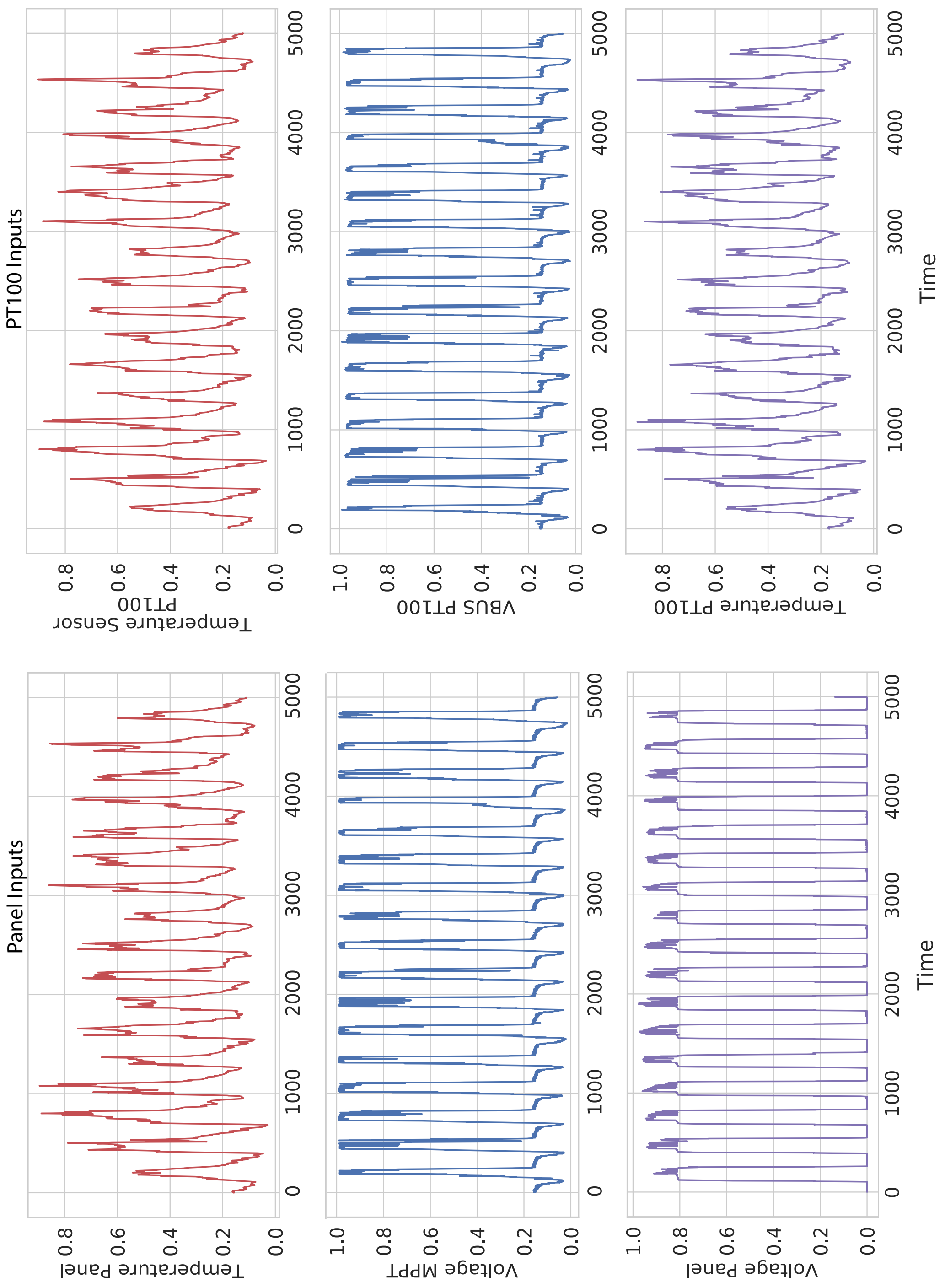
Figure 66 – Comparison of real and all predicted time series saving 60% of transmission.



Source – Author.

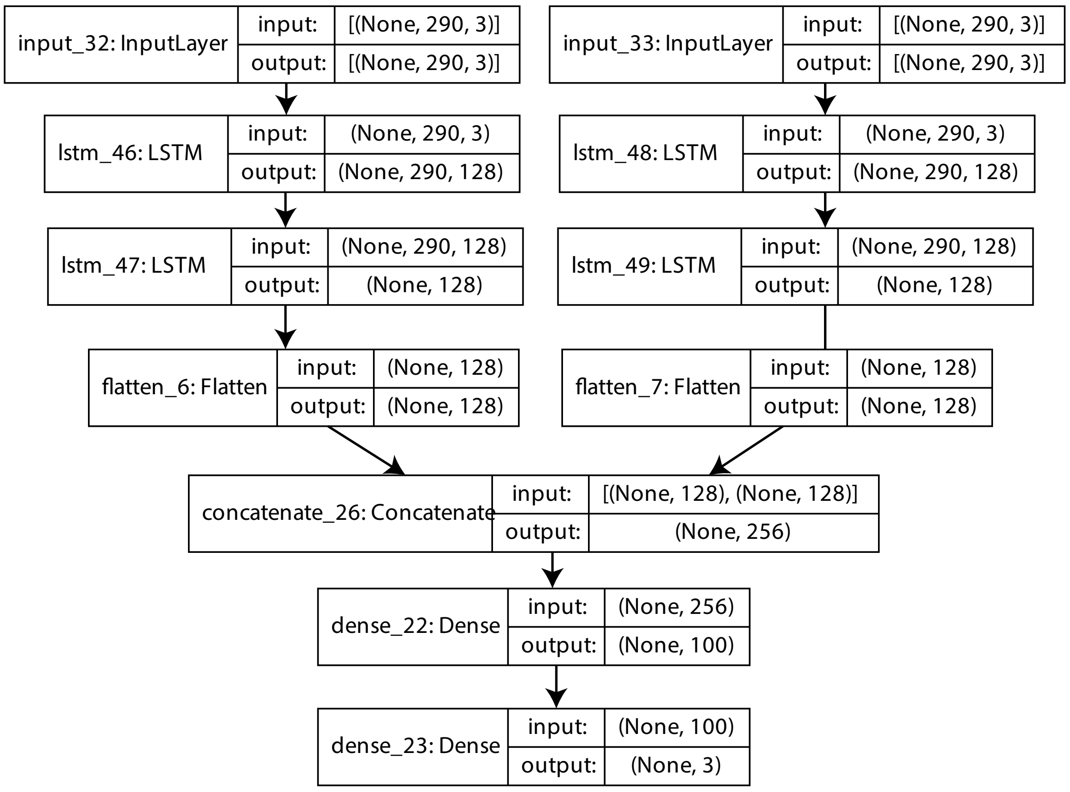
the second sensor values. The model reached a $R^2$ = 0.974 and a $MSE$ = 0.0026, which is much better than the previous one of $R^2$ = 0.958. The prediction over the testing set is shown in Figure 69. Therefore we can conclude that the combination of sensory data with high correlation, data from sensors in the same location, can help improving prediction and it is a viable solution for CHs and BSs when there is a need for data prediction. The same approach is not relevant in the case of the Attention model, since the results did not present a considerable improvement, and the model gets more complex.

Figure 67 – Time Series generated by sensors 34.B2.9F.A9 and 29.E5.5A.24.



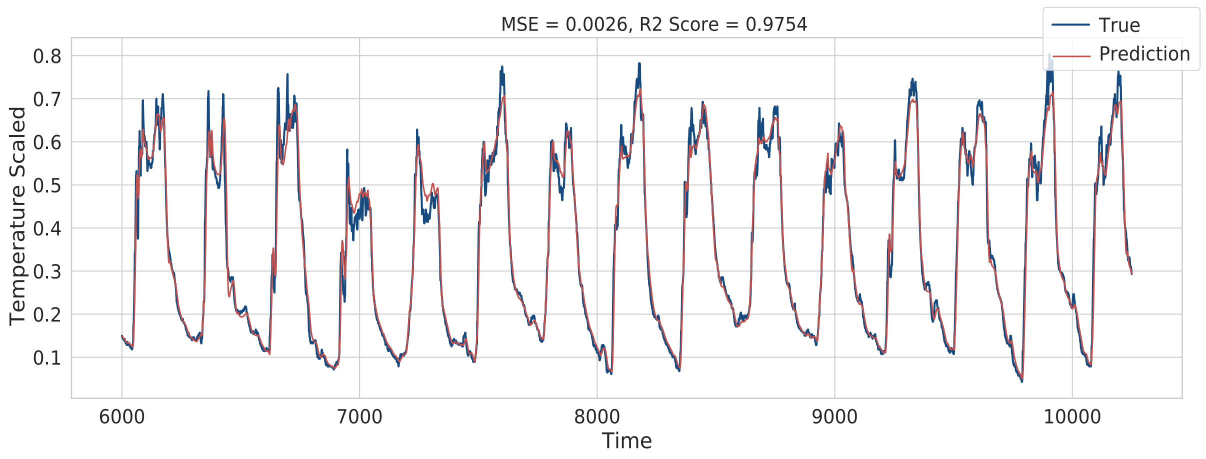
Source – Author.

Figure 68 – Multi Channel LSTM model architecture.


input_32: InputLayer | input: [(None, 290, 3)] | output: [(None, 290, 3)]
input_33: InputLayer | input: [(None, 290, 3)] | output: [(None, 290, 3)]
lstm_46: LSTM | input: (None, 290, 3) | output: (None, 290, 128)
lstm_48: LSTM | input: (None, 290, 3) | output: (None, 290, 128)
lstm_47: LSTM | input: (None, 290, 128) | output: (None, 128)
lstm_49: LSTM | input: (None, 290, 128) | output: (None, 128)
flatten_6: Flatten | input: (None, 128) | output: (None, 128)
flatten_7: Flatten | input: (None, 128) | output: (None, 128)
concatenate_26: Concatenate | input: [(None, 128), (None, 128)] | output: (None, 256)
dense_22: Dense | input: (None, 256) | output: (None, 100)
dense_23: Dense | input: (None, 100) | output: (None, 3)

Source – Author.

Figure 69 – Comparison between predictions and real values in testing set using Multi Channel LSTM.



Source – Author.

## 5.5 CONCLUSIONS

Data prediction can help reduce energy consumption in Wireless Sensor Networks by reducing transmissions. This chapter presented a comparison of some of the more effective data prediction methods for this application. The comparison was performed between deep learning models because of the advantages they provide compared to traditional ones. The results show that the Attention model can make accurate predictions and stable long-term forecasts. Setting different error limits when forecasting we can adjust the percent of transmitted points by the sensor and the accuracy of the final data, therefore, depending on the accuracy required by the application, we can set a small value, such as 1.3 $C^\circ$ and save approximately 70% in transmission, or a bigger one like 3 $C^\circ$ and save up to 90% in transmission with relatively good accuracy.

# 6 CONCLUSIONS

Wireless Sensor Networks is an exciting field of research. These networks are used in many fields and numerous applications because of the many advantages they present regarding to conventional networks. This work presented the exploration of a WSN deployed at the Hydropower plant of Cachoeira Dourada using statistical and machine learning methods to retrieve insights about its current state.

By analyzing the data from the sensors it was possible to understand which radios and sensors are more stable regarding to the number of missing transmissions and the analysis of the RSSI measures provided information about the state of the sensors and the stability of the links between the radios.

An anomaly detection analysis was performed using basic and machine learning techniques. The analysis allowed the detection of strange behaviors in the temperature data, such as unexpectedly high and low values, and sudden temperature drops during the day or spikes at night. The reason for these spikes and drops could be correlated to some events taking place at the location of the sensor. The tested methods could later be applied in real-time to detect strange events or in an upper layer, such as the cloud.

As one of the key aspects to consider in these networks is energy and its conservation, an algorithm based on neural networks was developed and tested in the dataset to reduce the amount of transmitted data from sensor nodes to cluster heads, in a dual prediction scheme to optimize energy. The method was able to provide accurate predictions and stable long-term forecasts. The percentage of transmitted points and the accuracy of the final data can be adjusted by tunning a threshold value. Results showed that it was possible to save up to 90% in transmission maintaining a transmission with relatively good accuracy of the transmitted data, concluding that the algorithm, in theory, can reduce considerably the power consumed by transmission.
The solution resulted in a paper presented in the 2021 IEEE International Instrumentation and Measurement Technology Conference (I2MTC) and published by the IEEE under the name of Multivariate Data Prediction in a Wireless Sensor Network based on Sequence to Sequence Models.

Possible future works can focus on:

a) Implementing the algorithm in the actual sensors to measure the real impact in transmission.

b) Test and compare other networks and algorithms, such as Transformer networks.

c) Further explore the correlation between sensors and how this can be used to reduce transmissions or improve missing data.

d) Further explore the relationship between the anomalies and events at the locations of the sensors to discover root causes.

## REFERENCES

ABADI, Martín et al. TensorFlow: A system for large-scale machine learning. **Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2016**, p. 265–283, 2016. arXiv: 1605.08695.

ABBASI, Ameer Ahmed; YOUNIS, Mohamed. A survey on clustering algorithms for wireless sensor networks. **Computer Communications**, v. 30, n. 14-15, p. 2826–2841, 2007. ISSN 01403664. DOI: 10.1016/j.comcom.2007.05.024.

ABBOUD, Ahmad; YAZBEK, Abdel-Karim; CANCES, Jean-Pierre; MEGHDADI, Vahid. Forecasting and skipping to Reduce Transmission Energy in WSN, p. 1–8, 2016. arXiv: 1606.01937. Available from: http://arxiv.org/abs/1606.01937.

ABDULKAREM, Mohammed; SAMSUDIN, Khairulmizam; ROKHANI, Fakhrul Zaman; A RASID, Mohd Fadlee. Wireless sensor network for structural health monitoring: A contemporary review of technologies, challenges, and future direction. **Structural Health Monitoring**, v. 19, n. 3, p. 693–735, 2020. ISSN 17413168. DOI: 10.1177/1475921719854528.

ADHIKARI, Ratnadip; AGRAWAL, Ramesh K. An introductory study on time series modeling and forecasting. **arXiv preprint arXiv:1302.6613**, 2013.

ALIYU, Farouq; AL-SHABOTI, Mohammed; GARBA, Yau; SHELTAMI, Tarek; BARNAWI, Abdulaziz; MORSY, Mohammed A. Hydrogen sulfide (H2S) gas safety system for oil drilling sites using Wireless Sensor Network. **Procedia Computer Science**, Elsevier Masson SAS, v. 63, p. 499–504, 2015. ISSN 18770509. DOI: 10.1016/j.procs.2015.08.375. Available from: http://dx.doi.org/10.1016/j.procs.2015.08.375.

ALIYU, Farouq; UMAR, Sani; AL-DUWAISH, Hussain. A survey of applications of artificial neural networks in wireless sensor networks. **2019 8th International Conference on Modeling Simulation and Applied Optimization, ICMSAO 2019**, IEEE, 2019. DOI: 10.1109/ICMSAO.2019.8880364.

ALVES, Maicon Melo; PIRMEZ, Luci; ROSSETTO, Silvana; DELICATO, Flavia C.; FARIAS, Claudio M. de; PIRES, Paulo F.; SANTOS, Igor L. dos; ZOMAYA, Albert Y. Damage prediction for wind turbines using wireless sensor and actuator networks. **Journal of Network and Computer Applications**, Elsevier, v. 80, November 2016,

p. 123–140, 2017. ISSN 10958592. DOI: 10.1016/j.jnca.2016.12.027. Available from: http://dx.doi.org/10.1016/j.jnca.2016.12.027.

ANASTASI, Giuseppe; CONTI, Marco; DI FRANCESCO, Mario; PASSARELLA, Andrea. Energy conservation in wireless sensor networks: A survey. **Ad Hoc Networks**, Elsevier B.V., v. 7, n. 3, p. 537–568, 2009. ISSN 15708705. DOI: 10.1016/j.adhoc.2008.06.003. Available from: http://dx.doi.org/10.1016/j.adhoc.2008.06.003.

ANTAYHUA, Roddy A.R.; PEREIRA, Maicon D.; FERNANDES, Nestor C.; SOUSA, Fernando Rangel de. Exploiting the rssi long-term data of a wsn for the rf channel modeling in eps environments. **Sensors (Switzerland)**, v. 20, n. 11, p. 1–15, 2020. ISSN 14248220. DOI: 10.3390/s20113076.

BAHDANAU, Dzmitry; CHO, Kyung Hyun; BENGIO, Yoshua. Neural machine translation by jointly learning to align and translate. In: 3RD International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings. [S.l.: s.n.], 2015. P. 1–15. arXiv: 1409.0473.

BAUER, Andre; ZUFLE, Marwin; HERBST, Nikolas; ZEHE, Albin; HOTHO, Andreas; KOUNEV, Samuel. Time Series Forecasting for Self-Aware Systems. **Proceedings of the IEEE**, v. 108, n. 7, p. 1068–1093, 2020. ISSN 15582256. DOI: 10.1109/JPROC.2020.2983857.

BEN ARBI, Imen; DERBEL, Faouzi; STRAKOSCH, Florian. Forecasting methods to reduce energy consumption in WSN. **I2MTC 2017 - 2017 IEEE International Instrumentation and Measurement Technology Conference, Proceedings**, IEEE, p. 0–5, 2017. DOI: 10.1109/I2MTC.2017.7969960.

BURATTI, Chiara; CONTI, Andrea; DARDARI, Davide; VERDONE, Roberto. An overview on wireless sensor networks technology and evolution. **Sensors**, v. 9, n. 9, p. 6869–6896, 2009. ISSN 14248220. DOI: 10.3390/s90906869.

CAMTEPE, Seyit; YENER, Bulent. Key Distribution Mechanisms for Wireless Sensor Networks: a Survey, June 2005.

CARLOS-MANCILLA, Miriam; LÓPEZ-MELLADO, Ernesto; SILLER, Mario. Wireless sensor networks formation: Approaches and techniques. **Journal of Sensors**, v. 2016, 2016. ISSN 16877268. DOI: 10.1155/2016/2081902.

CHENG, Hongju; XIE, Zhe; SHI, Yushi; XIONG, Naixue. Multi-Step Data Prediction in Wireless Sensor Networks Based on One-Dimensional CNN and Bidirectional LSTM. **IEEE Access**, IEEE, v. 7, p. 117883–117896, 2019a. ISSN 21693536. DOI: 10.1109/ACCESS.2019.2937098.

CHENG, Hongju; XIE, Zhe; WU, Leihuo; YU, Zhiyong; LI, Ruixing. Data prediction model in wireless sensor networks based on bidirectional LSTM. **Eurasip Journal on Wireless Communications and Networking**, EURASIP Journal on Wireless Communications and Networking, v. 2019, n. 1, 2019b. ISSN 16871499. DOI: 10.1186/s13638-019-1511-4.

DAS, Himanish Shekhar; ROY, Pinki. A deep dive into deep learning techniques for solving spoken language identification problems. In: INTELLIGENT Speech Signal Processing. [S.l.]: Elsevier, 2019. P. 81–100.

DAS, Rik; GHOSH, Saurav; MUKHERJEE, Dhritiman. Bayesian Estimator Based Weather Forecasting using WSN. **3rd International Conference and Workshops on Recent Advances and Innovations in Engineering, ICRAIE 2018**, IEEE, v. 2018, November, p. 22–25, 2018. DOI: 10.1109/ICRAIE.2018.8710410.

DENG, Dawei; YUAN, Haiwen; LV, Jianxun; JU, Yong. Distributed wireless sensor network system for electric field measurement. **Journal of Computational and Theoretical Nanoscience**, IEEE, v. 14, n. 4, p. 1844–1851, 2017. ISSN 15461963. DOI: 10.1166/jctn.2017.6515.

DIAS, Gabriel Martins; BELLALTA, Boris; OECHSNER, Simon. A survey about prediction-based data reduction in wireless sensor networks. **ACM Computing Surveys**, v. 49, n. 3, 2016. ISSN 15577341. DOI: 10.1145/2996356. arXiv: 1607.03443.

DIAS, Gabriel Martins; BELLALTA, Boris; OECHSNER, Simon. Using data prediction techniques to reduce data transmissions in the IoT. **2016 IEEE 3rd World Forum on Internet of Things, WF-IoT 2016**, IEEE, p. 331–335, 2017. DOI: 10.1109/WF-IoT.2016.7845518.

DILHAC, Jean-Marie; BOITIER, Vincent. **Wireless Sensor Networks**. Ed. by Jean-Marie Dilhac and Vincent Boitier. [S.l.]: Elsevier, 2016. P. 1–11. ISBN 978-1-78548-123-9. DOI: https://doi.org/10.1016/B978-1-78548-123-9.50001-2.

Available from:
`https://www.sciencedirect.com/science/article/pii/B9781785481239500012`.

FARAHANI, Shahin. Zigbee basics. In: ZIGBEE Wireless Networks and Transceivers. [S.l.]: Newnes, 2008. P. 1–24.

FATHY, Yasmin; BARNAGHI, Payam; TAFAZOLLI, Rahim. An adaptive method for data reduction in the Internet of Things. **IEEE World Forum on Internet of Things, WF-IoT 2018 - Proceedings**, 2018-January, p. 729–735, 2018. DOI: `10.1109/WF-IoT.2018.8355187`.

FU, Xiuwen; YANG, Yongsheng. Modeling and analysis of cascading node-link failures in multi-sink wireless sensor networks. **Reliability Engineering and System Safety**, Elsevier Ltd, v. 197, February, p. 106815, 2020. ISSN 09518320. DOI: `10.1016/j.ress.2020.106815`. Available from: `https://doi.org/10.1016/j.ress.2020.106815`.

GADEPALLY, Vijay; GOODWIN, Justin; KEPNER, Jeremy; REUTHER, Albert; REYNOLDS, Hayley; SAMSI, Siddharth; SU, Jonathan; MARTINEZ, David. Ai enabling technologies: A survey. **arXiv preprint arXiv:1905.03592**, 2019.

GAMBOA, John. Deep Learning for Time-Series Analysis. **arXiv**, 2017. ISSN 23318422. arXiv: `1701.01887`.

GEETHA, V.; KALLAPUR, P.V.; TELLAJEERA, Sushma. Clustering in Wireless Sensor Networks: Performance Comparison of LEACH and LEACH-C Protocols Using NS2. **Procedia Technology**, v. 4, p. 163–170, 2012. ISSN 22120173. DOI: `10.1016/j.protcy.2012.05.024`.

GÉRON, Aurélien. **Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems**. [S.l.]: O'Reilly Media, 2019.

GETACHEW, Yidnekachew; CHUNG, Tae Sun. Energy efficient routing for wireless sensor networks. **Proceedings - 2012 8th International Conference on Computing and Networking Technology (INC, ICCIS and ICMIC), ICCNT 2012**, p. 358–361, 2012. ISSN 2288-2847. DOI: `10.7236/ijasc2014.3.2.1`.

GHAYVAT, Hemant; MUKHOPADHYAY, Subhas; GUI, Xiang; SURYADEVARA, Nagender. WSN- and IOT-based smart homes and their extension to smart buildings. **Sensors (Switzerland)**, v. 15, n. 5, p. 10350–10379, 2015. ISSN 14248220. DOI: `10.3390/s150510350`.

GOODFELLOW, Ian; BENGIO, Yoshua; COURVILLE, Aaron. **Deep Learning**. [S.l.]: MIT Press, 2016. `http://www.deeplearningbook.org`.

HADJILA, Mourad; GUYENNET, Hervé; FEHAM, Mohammed. A Chain-Based Routing Protocol to Maximize the Lifetime of Wireless Sensor Networks. **Wireless Sensor Network**, v. 05, n. 05, p. 116–120, 2013. ISSN 1945-3078. DOI: `10.4236/wsn.2013.55014`.

HANNUN, Awni Y.; RAJPURKAR, Pranav; HAGHPANAHI, Masoumeh; TISON, Geoffrey H.; BOURN, Codie; TURAKHIA, Mintu P.; NG, Andrew Y. Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network. **Nature Medicine**, Springer US, v. 25, n. 1, p. 65–69, 2019. ISSN 1546170X. DOI: `10.1038/s41591-018-0268-3`. Available from: `http://dx.doi.org/10.1038/s41591-018-0268-3`.

HAWKINS, Douglas M. **Identification of outliers**. [S.l.]: Springer, 1980. v. 11.

HEINZELMAN, Wendi Rabiner; CHANDRAKASAN, Anantha; BALAKRISHNAN, Hari. Energy-efficient communication protocol for wireless microsensor networks. **Proceedings of the Annual Hawaii International Conference on System Sciences**, 2000-January, n. 100, p. 1–10, 2000. ISSN 15301605.

HOCHREITER, Sepp; SCHMIDHUBER, Jürgen. Long Short-Term Memory. **Neural Computation**, v. 9, n. 8, p. 1735–1780, 1997. ISSN 08997667. DOI: `10.1162/neco.1997.9.8.1735`.

INC., Crossbow Technology. **MCS-KIT410CA CRICKET-KIT Datasheet**. San Jose, CA, USA: Crossbow Technology, INC. DOI: `10.1002/ss.37119936209`. Available from: `https://www.willow.co.uk/Cricket_Kit_Datasheet.pdf`.

ISMAIL FAWAZ, Hassan; FORESTIER, Germain; WEBER, Jonathan; IDOUMGHAR, Lhassane; MULLER, Pierre Alain. Deep learning for time series classification: a review. **Data Mining and Knowledge Discovery**, v. 33, n. 4,

p. 917–963, 2019. ISSN 1573756X. DOI: 10.1007/s10618-019-00619-1. arXiv: 1809.04356.

JINDAL, Vandana. History and architecture of wireless sensor networks for ubiquitous computing. **International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)**, v. 7, n. 2, p. 214–217, 2018.

KAKRIA, Priyanka; TRIPATHI, N. K.; KITIPAWANG, Peerapong. A real-time health monitoring system for remote cardiac patients using smartphone and wearable sensors. **International Journal of Telemedicine and Applications**, v. 2015, December, 2015. ISSN 16876423. DOI: 10.1155/2015/373474.

KANDRIS, Dionisis; NAKAS, Christos; VOMVAS, Dimitrios; KOULOURAS, Grigorios. Applications of wireless sensor networks: An up-to-date survey. **Applied System Innovation**, v. 3, n. 1, p. 1–24, 2020. ISSN 25715577. DOI: 10.3390/asi3010014.

KHEDO, Kavi K.; BISSESSUR, Yasdeo; GOOLAUB, Datta S. An inland Wireless Sensor Network system for monitoring seismic activity. **Future Generation Computer Systems**, Elsevier B.V., v. 105, p. 520–532, 2020. ISSN 0167739X. DOI: 10.1016/j.future.2019.12.025. Available from: https://doi.org/10.1016/j.future.2019.12.025.

KIM, Young Hwan; JANG, Kuk Jin; LEE, Seung Chul; PARK, Chang Won; YOUN, Hee Yong. A robust wearable health monitoring system based on WSN. **2013 IEEE 10th Consumer Communications and Networking Conference, CCNC 2013**, IEEE, p. 288–293, 2013. DOI: 10.1109/CCNC.2013.6488460.

KIRANYAZ, Serkan; AVCI, Onur; ABDELJABER, Osama; INCE, Turker; GABBOUJ, Moncef; INMAN, Daniel J. 1D convolutional neural networks and applications: A survey. **Mechanical Systems and Signal Processing**, The Author(s), v. 151, p. 107398, 2021. ISSN 10961216. DOI: 10.1016/j.ymssp.2020.107398. arXiv: 1905.03554. Available from: https://doi.org/10.1016/j.ymssp.2020.107398.

KRISHNA, Gopal; SINGH, Sunil Kumar; SINGH, Jyoti Prakash; KUMAR, Prabhat. Energy Conservation Through Data Prediction in Wireless Sensor Networks. **SSRN Electronic Journal**, p. 986–992, 2018. ISSN 1556-5068. DOI: 10.2139/ssrn.3172770.

KRISHNAMURTHY, Sudha et al. VigilNet: An Integrated Sensor Network System for Energy-Efficient Surveillance. **ACM Transactions on Sensor Networks**, v. 2, n. 1, p. 1–38, 2006.

KUMAR, Pawan; REDDY, S. R. N. Wireless sensor networks: a review of motes, wireless technologies, routing algorithms and static deployment strategies for agriculture applications. **CSI Transactions on ICT**, Springer India, v. 8, n. 3, p. 331–345, 2020. ISSN 2277-9078. DOI: `10.1007/s40012-020-00289-1`. Available from: `https://doi.org/10.1007/s40012-020-00289-1`.

LAI, Guokun; CHANG, Wei Cheng; YANG, Yiming; LIU, Hanxiao. Modeling long- and short-term temporal patterns with deep neural networks. **41st International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2018**, July, p. 95–104, 2018. DOI: `10.1145/3209978.3210006`. arXiv: `1703.07015`.

LI, Hui; JIANG, Hongtao; LI, Hua. The improvement algorithm of cluster head election based on LEACH. **Proceedings - 2013 6th International Conference on Intelligent Networks and Intelligent Systems, ICINIS 2013**, IEEE, p. 95–98, 2013. DOI: `10.1109/ICINIS.2013.31`.

LI, Shiyang; JIN, Xiaoyong; XUAN, Yao; ZHOU, Xiyou; CHEN, Wenhu; WANG, Yu Xiang; YAN, Xifeng. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. **arXiv**, NeurIPS, 2019. ISSN 23318422. arXiv: `1907.00235`.

LI, Yuanjun; SUN, Ruixiao; HORNE, Roland. **Deep learning for well data history analysis**. 2019-Septe. [S.l.: s.n.], 2019. ISBN 9781613996638. DOI: `10.2118/196011-ms`.

LIAZID, Hidaya; LEHSAINI, Mohamed; LIAZID, Abdelkrim. An improved adaptive dual prediction scheme for reducing data transmission in wireless sensor networks. **Wireless Networks**, Springer US, v. 25, n. 6, p. 3545–3555, 2019. ISSN 15728196. DOI: `10.1007/s11276-019-01950-7`. Available from: `https://doi.org/10.1007/s11276-019-01950-7`.

LIM, Hock Beng; MA, Di; WANG, Bang; KALBARCZYK, Zbigniew; IYER, Ravishankar K.; WATKIN, Kenneth L. A soldier health monitoring system for military applications. **2010 International Conference on Body Sensor Networks, BSN 2010**, p. 246–249, 2010. DOI: `10.1109/BSN.2010.58`.

LINDSEY, Stephanie; RAGHAVENDRA, Cauligi S. PEGASIS: Power-efficient gathering in sensor information systems. **IEEE Aerospace Conference Proceedings**, v. 3, p. 1125–1130, 2002. ISSN 1095323X. DOI: `10.1109/AERO.2002.1035242`.

LIU, Wei Dong; WANG, Zheng Dong; ZHANG, Shen; WANG, Qing Qing. A low power grid-based cluster routing algorithm of wireless sensor networks. **Proceedings - 2010 International Forum on Information Technology and Applications, IFITA 2010**, IEEE, v. 1, p. 227–229, 2010. DOI: `10.1109/IFITA.2010.254`.

LOPES PEREIRA, R.; TRINDADE, J.; GONALVES, F.; SURESH, L.; BARBOSA, D.; VAZÃO, T. A wireless sensor network for monitoring volcano-seismic signals. **Natural Hazards and Earth System Sciences**, v. 14, n. 12, p. 3123–3142, 2014. ISSN 16849981. DOI: `10.5194/nhess-14-3123-2014`.

LÓPEZ-ARDAO, J. Carlos; RODRÍGUEZ-RUBIO, Raúl F.; SUÁREZ-GONZÁLEZ, Andrés; RODRÍGUEZ-PÉREZ, Miguel; SOUSA-VIEIRA, M. Estrella. Current trends on green wireless sensor networks. **Sensors**, v. 21, n. 13, 2021. ISSN 14248220. DOI: `10.3390/s21134281`.

LUO, Tie; NAGARAJANY, Sai G. Distributed anomaly detection using autoencoder neural networks in WSN for IoT. **IEEE International Conference on Communications**, 2018-May, 2018. ISSN 15503607. DOI: `10.1109/ICC.2018.8422402`. arXiv: `1812.04872`.

LUONG, Minh Thang; PHAM, Hieu; MANNING, Christopher D. Effective approaches to attention-based neural machine translation. **Conference Proceedings - EMNLP 2015: Conference on Empirical Methods in Natural Language Processing**, p. 1412–1421, 2015. DOI: `10.18653/v1/d15-1166`. arXiv: `1508.04025`.

MALHOTRA, Geetika; VIRMANI, Deepali. Intelligent information retrieval in a Tsunami detection system using wireless sensor networks. **Proceeding - IEEE International Conference on Computing, Communication and Automation, ICCCA 2016**, IEEE, Iccca, p. 939–944, 2017. DOI: `10.1109/CCAA.2016.7813873`.

MAMUN, Quazi. A qualitative comparison of different logical topologies for Wireless Sensor Networks. **Sensors (Switzerland)**, v. 12, n. 11, p. 14887–14913, 2012. ISSN 14248220. DOI: `10.3390/s121114887`.

MANN, Jaspreet Kaur; PERINPANAYAGAM, Suresh; JENNIONS, Ian. Aging Detection Capability for Switch-Mode Power Converters. **IEEE Transactions on Industrial Electronics**, v. 63, n. 5, p. 3216–3227, May 2016. ISSN 1557-9948. DOI: 10.1109/TIE.2016.2535104.

MITCHELL, Tom. **Machine learning**. [S.l.]: McGraw hill Burr Ridge, 1997.

MOSCHITTA, Antonio; NERI, Igor. Power consumption Assessment in Wireless Sensor Networks. In: ICT - Energy - Concepts Towards Zero - Power Information and Communication Technology. [S.l.]: InTech, Feb. 2014. DOI: 10.5772/57201.

PACHARANEY, Utkarsha S.; GUPTA, Rajiv Kumar. Clustering and Compressive Data Gathering in Wireless Sensor Network. **Wireless Personal Communications**, Springer US, v. 109, n. 2, p. 1311–1331, 2019. ISSN 1572834X. DOI: 10.1007/s11277-019-06614-5. Available from: https://doi.org/10.1007/s11277-019-06614-5.

PADMANABHAN, K; KAMALAKKANNAN, P. Clustering Protocol in Wireless S ensor Networks. IEEE, n. 978, p. 45–49, 2012.

PENTARIS, F. P.; STONHAM, J.; MAKRIS, J. P. A cost effective wireless structural health monitoring network for buildings in earthquake zones. **Smart Materials and Structures**, v. 23, n. 10, 2014. ISSN 1361665X. DOI: 10.1088/0964-1726/23/10/105010.

PEREIRA, M. D.; ROMERO, R. A.; FERNANDES, N.; SOUSA, F. R. de. Path loss and shadowing measurements at 2.4 GHz in a power plant using a mesh network. In: 2018 IEEE International Instrumentation and Measurement Technology Conference (I2MTC). [S.l.: s.n.], May 2018. P. 1–6. DOI: 10.1109/I2MTC.2018.8409563.

SAMARAH, Samer; AL-HAJRI, Muhannad; BOUKERCHE, Azzedine. A Predictive Energy-Efficient Technique to Support Object-Tracking Sensor Networks. **IEEE Transactions on Vehicular Technology**, v. 60, n. 2, p. 656–663, 2011. DOI: 10.1109/TVT.2010.2102375.

SANDEEP VERMA. Network Topologies in Wireless Sensor Networks: A Review 1. **International Journal of Electronics and Communication Technology**, v. 4, n. 3, p. 1–5, 2013. DOI: 10.1.1.308.796.

SENOUCI, Mustapha Reda; MELLOUK, Abdelhamid. **Deploying Wireless Sensor Networks Theory and Practice Preface**. [S.l.: s.n.], 2016. P. ix+. ISBN 978-0-08-101188-1; 978-1-78548-099-7.

SHARMA, Sukhwinder; BANSAL, Rakesh Kumar; BANSAL, Savina. Issues and challenges in wireless sensor networks. **Proceedings - 2013 International Conference on Machine Intelligence Research and Advancement, ICMIRA 2013**, IEEE, p. 58–62, 2014. DOI: 10.1109/ICMIRA.2013.18.

SHEN, Yan; LI, Xunbo. Wavelet Neural Network Approach for Dynamic Power Management in Wireless Sensor Networks, p. 376–381, July 2008. DOI: 10.1109/ICESS.2008.36.

SHIH, Shun Yao; SUN, Fan Keng; LEE, Hung yi. Temporal pattern attention for multivariate time series forecasting. **Machine Learning**, v. 108, n. 8-9, p. 1421–1441, 2019. ISSN 15730565. DOI: 10.1007/s10994-019-05815-0. arXiv: 1809.04206.

SHU, Tongxin; CHEN, Jiahong; BHARGAVA, Vijay K.; DE SILVA, Clarence W. An Energy-Efficient Dual Prediction Scheme Using LMS Filter and LSTM in Wireless Sensor Networks for Environment Monitoring. **IEEE Internet of Things Journal**, v. 6, n. 4, p. 6736–6747, 2019. ISSN 23274662. DOI: 10.1109/JIOT.2019.2911295.

SINHA, Adwitiya; LOBIYAL, D. K. Prediction Models for Energy Efficient Data Aggregation in Wireless Sensor Network. **Wireless Personal Communications**, Springer US, v. 84, n. 2, p. 1325–1343, 2015. ISSN 1572834X. DOI: 10.1007/s11277-015-2690-x.

SONG, Jianping; HAN, Song; MOK, Aloysius K.; CHEN, Deji; NIXON, Mark. **Centralized control of wireless sensor networks for real-time applications**. [S.l.]: IFAC, 2007. v. 7, p. 25–32. ISBN 9783902661340. DOI: 10.3182/20071107-3-fr-3907.00003. Available from: http://dx.doi.org/10.3182/20071107-3-FR-3907.00003.

SONG, Yanchao; LUO, Juan; LIU, Chang; HE, Wenfeng. Periodicity-and-linear-based data suppression mechanism for WSN. **Proceedings - 14th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, TrustCom 2015**, IEEE, v. 1, p. 1267–1271, 2015. DOI: 10.1109/Trustcom.2015.516.

SUTSKEVER, Ilya; VINYALS, Oriol; LE, Quoc V. Sequence to sequence learning with neural networks. **Advances in Neural Information Processing Systems**, v. 4, January, p. 3104–3112, 2014. ISSN 10495258. arXiv: `1409.3215`.

TAYEH, Gaby Bou; MAKHOUL, Abdallah; PERERA, Charith; DEMERJIAN, Jacques. A spatial-temporal correlation approach for data reduction in cluster-based sensor networks. **IEEE Access**, IEEE, v. 7, p. 50669–50680, 2019.

VAN DER BENT, Henk. Wireless technology in industrial automation. **EngineerIT**, v. 93, MARCH, p. 30–31, 2010. ISSN 19915047.

WU, Neo; GREEN, Bradley; BEN, Xue; O'BANION, Shawn. Deep transformer models for time series forecasting: The influenza prevalence case. **arXiv**, 2020. ISSN 23318422. arXiv: `2001.08317`.

XU, Y.; HEIDEMANN, J.; ESTRIN, D. Geography-informed energy conservation for ad hoc routing. **Proceedings of the Annual International Conference on Mobile Computing and Networking, MOBICOM**, p. 70–84, 2001. DOI: `10.1145/381677.381685`.