Towards smart firefighting using the internet of things and machine learning

Gaby Bou Tayeh

UBFC
UNIVERSITÉ
BOURGOGNE FRANCHE-COMTÉ

UFC
UNIVERSITÉ
DE FRANCHE-COMTÉ

THÈSE DE DOCTORAT

DE L'ÉTABLISSEMENT UNIVERSITÉ BOURGOGNE FRANCHE-COMTÉ

PRÉPARÉE À L'UNIVERSITÉ DE FRANCHE-COMTÉ

École doctorale n°37

Sciences Pour l'Ingénieur et Microtechniques

Doctorat d'Informatique

par

GABY BOU TAYEH

Internet des Objets et Intelligence Artificielle pour les Pompiers de Demain

Thèse présentée et soutenue à Belfort, le 8 juillet 2020

Composition du Jury :

MITTON NATHALIE	Directrice de recherche à l'Inria Lille	Rapporteur
MOKDAD LYNDA	Professeur à l'Université Paris-Est Val de Marne	Rapporteur
CHRÉTIEN STÉPHANE	Professeur à l'Université Lyon 2	Examinateur
GUYEUX CHRISTOPHE	Professeur à l'Université de Franche-Comté	Examinateur
DEMERJIAN JACQUES	Professeur à l'Université Libanaise	Invité
BAHI JACQUES	Professeur à l'Université de Franche-Comté	Directeur de thèse
MAKHOUL ABDALLAH	Professeur à l'Université de Franche-Comté	Directeur de thèse

UBFC
UNIVERSITÉ
BOURGOGNE FRANCHE-COMTÉ

UFC
UNIVERSITÉ
DE FRANCHE-COMTÉ

THÈSE DE DOCTORAT

DE L'ÉTABLISSEMENT UNIVERSITÉ BOURGOGNE FRANCHE-COMTÉ

PRÉPARÉE À L'UNIVERSITÉ DE FRANCHE-COMTÉ

École doctorale n°37

Sciences Pour l'Ingénieur et Microtechniques

Doctorat d'Informatique

par

GABY BOU TAYEH

Towards Smart Firefighting Using The Internet of Things and Machine Learning

Thèse présentée et soutenue à Belfort, le 8 juillet 2020

Composition du Jury :

MITTON NATHALIE	Directrice de recherche à l'Inria Lille	Rapporteur
MOKDAD LYNDA	Professeur à l'Université Paris-Est Val de Marne	Rapporteur
CHRÉTIEN STÉPHANE	Professeur à l'Université Lyon 2	Examinateur
GUYEUX CHRISTOPHE	Professeur à l'Université de Franche-Comté	Examinateur
DEMERJIAN JACQUES	Professeur à l'Université Libanaise	Invité
BAHI JACQUES	Professeur à l'Université de Franche-Comté	Directeur de thèse
MAKHOUL ABDALLAH	Professeur à l'Université de Franche-Comté	Directeur de thèse

ABSTRACT

Towards Smart Firefighting Using The Internet of Things and Machine Learning

Gaby Bou Tayeh
University of Bourgogne Franche Comté, 2020

Supervisors: Abdallah Makhoul and Jacques Bahi

The Internet of Things (IoT) is the ability for "things" that contains embedded technologies to sense, communicate, interact, and collaborate with other things, thus creating a network of physical objects. These connected devices are meant to constantly log and communicate data to an end-user. Once, the data has been collected, it could be analyzed to extract extremely useful information. IoT is taking connectivity beyond laptops and smartphones towards connected cars, smart homes, smart industries, connected wearables, smart cities, and connected healthcare.

Firefighters are now operating in an ever-increasing sensor-rich environment that is creating vast amounts of potentially useful data. The comprehensive ability to collect, analyze and process this data is opening new possibilities for the fire service to perform work tasks in a highly effective and efficient manner either during pre-fire, trans-fire (i.e. during the event), and post-fire stages. This dissertation global objective is to propose solutions that effectively use the available sensor technology, the means to communicate that data, the knowledge base and algorithms to most process the data, convert it into significant knowledge/beneficial decision tools, and effectively communicate the information to those who need it, on the fire ground and elsewhere.

The first objective was to focus our efforts on the sensor devices which are the source of the valuable data. The most challenging problem is how to ensure a sufficient operational lifetime for these tiny devices that have very limited energy available for consumption. Based on the fact that the sensing, transmission, and processing are the governing energy-consuming activities of the sensor devices we worked on reducing these activities. We first proposed a novel data transmission reduction algorithm, our approach learns the moving trend of the collected data after a brief observation period. Then it uses this knowledge to build a prediction model on the sensor node that is able to forecast future mea-

surements. The latter, does not transmit any measurement to the destination (Sink) as long as its corresponding prediction is considered accurate. Having built the same prediction model simultaneously with the sensor node, the Sink produces the same predictions. Since packet loss is a common problem in Wireless Sensor Networks (WSNs) we coupled this transmission reduction algorithm with a technique that detects missing packets, ensures synchronization during occasional re-adjustment of the prediction model, and reconstructs all the missing data. Adding an additional adaptive sampling layer on the previously described transmission reduction algorithm is then proposed. If no significant variations in collected data over a certain period of time are recorded, the sensing sampling rate is reduced without compromising the quality of the reported information. Lastly, we developed a centralized data reduction algorithm for cluster-based sensor networks. This latter, periodically analyses the data reported by its member nodes and adjust their sampling/transmission rates according to their level of correlation with each other.

With the help of the fire brigade of the department of Doubs, we successfully attained our second objective which is the development of a firefighter 2.0 system prototype. This latter consisted of a smart-watch for heart rate and movement monitoring and two sensor devices, one for localization and the other for environmental data sensing. Lastly, a web application for real-time visualization of the transmitted data is integrated into the system. The latter was validated throughout small-scale experiments. In addition, our proposed energy management algorithms were implemented on the real devices composing the system and their energy consumption was analyzed.

The third objective was to use sensory and non-sensory data to help in decision making. We first studied and compare different recent clustering algorithms for massive IoT data. The aim was to use the obtained results in order to determine the best clustering strategy for our network and develop an indoor localisation system. Second, we investigated the possibility of predicting the number of future interventions by training Machine Learning methods on data-set aggregated from multiple sources. The latter contained information on all interventions that have taken place in the department of Doubs in France since the year 2012, meteorological data, traffic data, and other information. The aim was to help the fire brigade to better manage their human and mobile resources according to the anticipated number of interventions.

Keywords: Wireless Sensor Networks, Internet of Things, Machine Learning, Energy Management, Lora, LoraWan, Remote Monitoring, Fire Fighters.

RÉSUMÉ

L'Internet des objets (IoT) est l'extension de la connectivité Internet dans des appareils physiques et des objets quotidiens. Ces derniers sont équipés d'éléments actifs qui les rend capable de collecter, traiter, et transmettre des données. Ces dispositifs connectés sont destinés à enregistrer et à communiquer en permanence des données à un utilisateur final. Une fois les données collectées, elles peuvent être analysées pour en extraire des informations extrêmement utiles.

Les pompiers travaillent actuellement dans un environnement de plus en plus riche en capteurs qui génère de grandes quantités de données potentiellement utiles. La capacité globale de collecte, d'analyse et de traitement de ces données ouvre de possibilités d'amélioration du quotidien des sapeurs-pompiers. L'objectif global de cette thèse est de proposer des solutions qui exploitent cette nouvelle technologie de capteurs et de communication à faible coût, ainsi que des algorithmes de traitement de données, pour les convertir en connaissances significatives/outils de décision, et communiquer efficacement les informations à ceux qui en ont besoin, sur les lieux d'intervention ou ailleurs.

Le premier objectif était de développer des solutions algorithmiques pour la gestion d'énergie des dispositifs IoT qui ont une source énergétique limitée afin de maximiser la durée de vie du réseau. En premier lieu, nous avons proposé un nouvel algorithme de réduction de la transmission des données. Notre approche est basée sur un modèle de prédiction dual, qui est capable de prédire des mesures futures. Ce dernier, exécuté à la fois par les capteurs et la station de base (Sink), permet au dispositif IoT de ne plus transmettre les mesures collectées vers la station tant que leurs prédictions correspondantes sont correctes. Le Sink ayant construit le même model il reproduit alors toutes les données non transmises. Finalement, le Sink et le dispositif IoT mettront à jour systématiquement le model de prédiction en fonctions de nouvelles estimations réelles. Comme la perte de paquets pendant la transmission est un problème courant dans les réseaux de capteurs sans fil (WSN), nous avons couplé cet algorithme de réduction de la transmission à une technique qui détecte les paquets manquants. Notre proposition assure la synchronisation de la mise à jour occasionnel du modèle de prédiction et intègre un modèle de reconstruction de données dans le cas de perte de messages. L'ajout d'une couche d'échantillonnage adaptative supplémentaire sur l'algorithme de réduction de la transmission décrit précédemment est ensuite proposée. Il s'agit d'adapter la fréquence de la collecte de données en fonction de leur variation. Si aucune variation significa-

tive des données collectées sur une certaine période de temps n'est enregistrée, le taux d'échantillonnage sera réduit tout en garantissant l'intégrité de l'information. De plus, nous avons développé un algorithme centralisé de gestion d'activité des dispositifs IoT. Cette approche permet à un Cluster-Head (CH) d'analyser périodiquement les données communiquées par ses nœuds membres et ajuste leurs taux d'échantillonnage et de transmission en fonction de la corrélation spatio-temporelle intra et inter nœuds.

D'un autre côté, notre deuxième objectif de cette était le développement d'un prototype réel et un système IoT déployé et dédié aux sapeurs-pompiers. Ce système a été réalisé en partenariat avec les pompiers du département du Doubs. Il consistait en une montre intelligente pour la surveillance du rythme cardiaque et des mouvements des pompiers et en deux dispositifs de capteurs, l'un pour leur localisation et l'autre pour la collecte de données environnementales de la zone d'intervention. Le système a été validé par des expériences et de banc-d 'essais. De plus, ce système a été utilisé pour confirmer les résultats obtenus de nos algorithmes proposés sur une plateforme réelle.

Le troisième objectif était d'utiliser des données issues des capteurs et d'autres concernant les interventions dans un but d'aide à la décision. Nous avons étudié et comparé différents algorithmes de clustering récents pour les données IoT massives. Le but était d'utiliser les résultats obtenus pour sélection la méthode optimale pour la problématique de la localisation interne "indoor". Ensuite, en utilisant des méthodes de Machine Learning entrainées sur une base de données existante, nous avons étudié la possibilité de prédire le nombre d'interventions futures des sapeurs-pompiers. Cette base de données contenait des informations sur toutes les interventions qui se sont passées dans le département du Doubs depuis l'année 2012. Elle contient des données météorologiques, et d'autres valeurs indépendantes. Le but était d'aider les pompiers à gérer leurs ressources humaines et mobiles en fonction de nombre d'interventions anticipé.

Mots-clés: Réseaux de capteurs sans fils, Internet des Objets, Machine Learning, Gestion d'Energy, LoRa, LoRaWan, Surveillance à distance, Sapeurs-pompiers.

CONTENTS

DEDICATION

This dissertation is dedicated to my beloved parents, my mother Antoinette, and my father Tony. It is impossible to thank you adequately for everything you've done for me. I watched you go through very tough times, and I still cannot forget the day I saw my big, strong father, crying for not being able to provide for his family. That day, your tears turned me to a man, from that moment I know that life will not be easy and I have to fight for success. Your tears taught me to bare responsibilities at a very young age, to be tough, and hard-working. My dear father, I want you to know that because of the tears you shed that day I am writing this dissertation today. To my hard-working mother, you taught me perseverance, love, and kindness. You taught me to always have faith and things will eventually get better. I cannot remember a single time where you bought something for yourself, you always spent everything you had on our education and health. This dissertation is dedicated to you, I wouldn't have made it so far without your support.

ACKNOWLEDGEMENTS

I would like to express my deep and sincere gratitude to my supervisor, Professor Jacques M. Bahi, for providing me with invaluable guidance through this thesis. I wish also to show my gratitude to my other supervisor, Professor Abdallah Makhoul. He taught me the methodology to carry out research, he gave me full autonomy and had trust in me, he always intervened to guide me in the right direction when I felt lost. I would also like to thank him for his friendship, empathy, and his great sense of humor.

I would like to pay my special regards to Professor Christophe Guyeux, he was my technical reference, was always available whenever I needed him for inquiries, he had the kindest heart, and he always encouraged me and made me feel more confident. I wish to express my deepest gratitude to professor Jacques Demerjian, he was the one who encouraged me to travel to France and pursue a Ph.D. degree, he always believed in me and he knew that I had what it takes to do this. He guided me all the way through this thesis and was always there when I needed his advice. He was to me like a big brother and I am grateful that I have him in my life.

I would like to pay my special regards to my friends and colleagues, especially Joseph Azar, Maroun Koussaify, Theodore Al Saify, Elias Tannous, and Lama Sleem. I would also like to express my sincere gratitude to two more friends. First, Dr. Anthony Moussa who helped me find a scholarship to cover up some of my expenses during the first year in France and guided me throughout the procedure of applying for a Visa. Second, Dr. Christian Salim for helping me settle in Belfort when I first came to France.

Lastly, I am indebted to my family, my father Tony, my mother Antoinette, my brother Edmond and my sister Nancy. Although they were far away from me, they didn't let me feel alone, they were always there for me, they encouraged and supported me and I'm happy that I made them proud.

INTRODUCTION

GENERAL INTRODUCTION

The internet of things (IoT) represents a general concept about the ability of network devices to detect and collect data from the world around us and then share this data on the web where it can be processed and used for a variety of purposes. IoT is a technological revolution that is already underway and is becoming more and more part of our daily lives. However, it is not a new concept, it dates back to 2003 when the first connected lamp (DAL) was created. The main reason why IoT is considerably developed today is because of its association with currently trending technologies such as big data and artificial intelligence. Perhaps one of the examples that could give us a glance into the future of IoT is the newly registered patent by MICROSOFT under the name of "CRYPTOCURRENCY SYSTEM USING BODY ACTIVITY DATA", that shows how IoT is being pushed to even become a part of our body.

As the world progresses rapidly and new technologies and tools are invented every day, the firefighting industry will eventually catch up with the innovations. Would today's firefighter imagine fighting a fire in a woolen uniform with a rubber slicker? Most definitely not, but there was a time when that was considered the state of the art personal protective equipment (PPE). Tomorrow's firefighter may not be able to imagine fighting a fire without having his condition and the conditions around him continuously being monitored by wireless sensors. These sensors could have the ability to monitor in real-time, not just the intervention area (whether it is a fire intervention or another type) but also the firefighter's physical condition. Embedded sensors would do more than monitoring, they would provide alerts that will take safety to a new level.

Whether it is your fitness tracker or a smart-watch, wearable technology is becoming more and more a part of our daily life. Knowing how advantageous wearable technology can be, it is simple to understand why the industry is moving towards making wearable sensors an important part of a firefighter's PPE. The latter can monitor both body and environmental parameters and can emit an alert when these parameters are deemed dangerous or indicate physical distress. These are signs a firefighter may overlook in his adrenaline-fueled dedication to accomplishing his job. There are already numerous off the shelf sensors that can monitor location, movement, heart rate, sleep quality, etc. These sensors need to be adapted to provide more information and withstand harsh

environmental conditions. Wearable sensors are small by nature and can go wherever the firefighter goes. Still, it is a challenge to create sensors and networks that are small in size but efficient and have enough energy to withstand for hours. The sensor networks should also be able to have quantitative approaches to data, capture motion, and conditions accurately to meet the identified needs. Finally, they need to be integrated into PPE gear while keeping it comfortable and non-constrictive at the same time. Technologically-enabled PPE will be able to help improve the safety of firefighters and management more efficiently the interventions. They can increase productivity for both individuals and crews, but most importantly, preventing injury and protecting lives will be their most important role.

MOTIVATION/OBJECTIVES

In this dissertation and under the project “Firefighter 2.0”, we work on tackling the most prominent issues facing the development of a smart wearable system for firefighters. Some interventions can last for long hours and the energy powering the system is all that matters. Therefore, the first objective is to focus our efforts on energy management in wireless sensor networks in order to ensure a maximum operational lifetime for the sensors and the system in general.

The second objective is to move on to resolve the engineering problems of the remote monitoring system. Several vital questions need to be addressed such as what devices to use? How to reliably deliver the collected information to the distant command-station? How can we design a simple system that is non-obstructive and does not get rejected by the firefighters? What advantages should the system provide? How can we balance between cost and the quality of service? etc.

After developing the energy management algorithms, and proposing a remote monitoring system that meets all the requirements, the third objective is to take a step further and pass from theory to real implementation. The proposed energy management algorithms are verified on real sensor devices and the prototype of the proposed system is validated.

Finally, the last objective is to use Machine Learning (ML) to build decision making tools that could improve the efficiency of the daily operational life of firefighters. With our hands on a large data set containing information about all the interventions that happened during the past six years in the region of Doubs, France, we build a prediction model capable of predicting accurately the number of future interventions. This could help firefighters efficiently manage their human and mobile resources according to the frequency of events. Moreover, several recent clustering techniques for massive sensor data are reviewed and compared aiming to find the best clustering strategy for the proposed sensor network and

to pre-process the data for future usage in decision making tools.

Figure 1 illustrates how the three main parts of this thesis are interconnected and it also resumes our contributions.

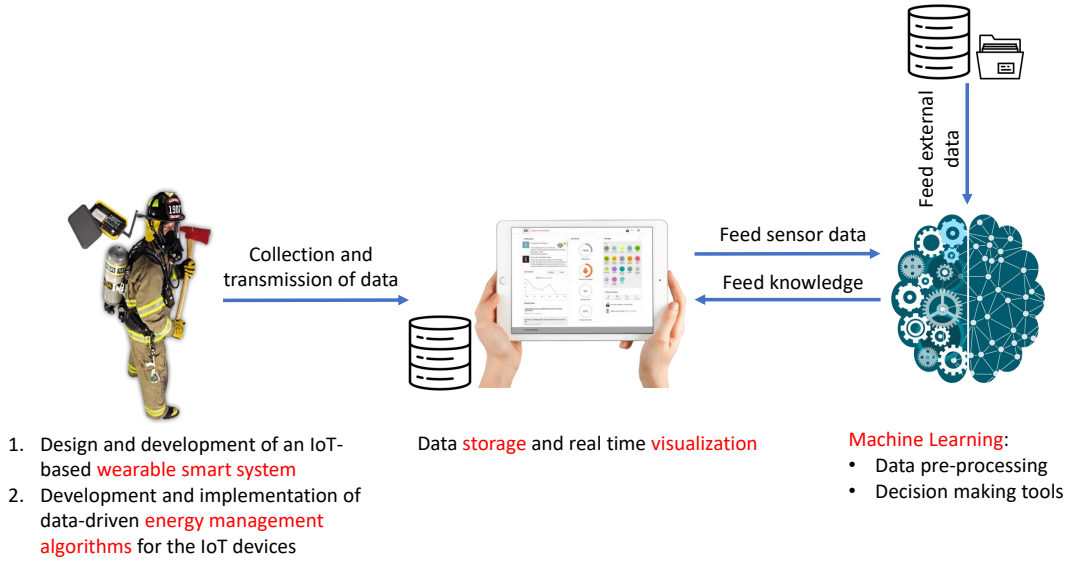


Figure 1: Graphical summary

Main Contributions of this Dissertation

Motivated by what has been discussed previously, the main contributions in this dissertation concentrate mainly on energy management in wireless sensor networks, design of the smart-wearable system for firefighters, validation of the proposed system and energy management algorithms through real implementation, and usage of ML models for help in decision making.

Energy Management: Energy management in wireless sensor networks is a broad and generalized term. There are a variety of approaches that aim to resolve this issue and they are discussed in detail in Chapter 1 of Part I. The common objective of these approaches is to reduce the energy consumption of the sensors in order to extend as much as possible the lifetime of the network. The root of the problem goes back to the fact that sensor nodes are, in general, powered by small batteries that have a short life span. Moreover, sensor nodes are generally designed for deployment either in large quantities or in a harsh environment, which renders the task of monitoring the battery levels of each individual node and periodically replacing depleted ones almost impossible.

Therefore, based on the fact that four main activities are responsible for consuming the majority of the provided energy namely sensing, processing, transmission, and powering the node's components, the research community developed various energy management approaches. In this dissertation, we focus mainly on two of them, namely data-driven and duty cycling. The former aims to reduce the volume of collected and transmitted data by a node using data-driven approaches such as compression, aggregation, prediction, adaptive sampling, etc. The latter aims to construct an efficient scheduling scheme that maximizes the sleep duration of a sensor. Our contributions regarding energy management in wireless sensor networks are listed below.

1. First, an algorithm that reduces the number of transmitted data between the sensor and the destination (Sink) is proposed. This algorithm exploits the temporal correlation of collected sensor measurements to build a simple yet robust model that is able to forecast future observations. This model is built simultaneously on the sensor and the Sink using a well-known approach called the Dual Prediction Model (DPM). Instead of transmitting each collected measurement, the sensor first compares it to the model's prediction, and in case there is no significant difference, the sensor node discards the measurement. The Sink, having produced the same prediction, considers it valid as long as it did not receive anything from the sensor. Finally, each time the wrong prediction is produced, the sensor transmits the real collected measurement (referred to as correction packet) to the Sink, and they will both use it to re-train the model.

2. Second, for a realistic implementation of the previously described algorithm, we took into consideration the packet loss during the transmission of correction packets. The Dual prediction mechanism works well as long as we consider a loss-free communication network, which of course is not possible in a real-world implementation. Packet loss is a major issue that could make the DPM approach unusable if not taken into consideration. In case a correction packet is lost during transmission and did not reach the sink, on the one hand, the sensor will proceed to re-train the model, on the other hand, the sink will consider that the model is still valid. Thus, the updated model on the sensor and the outdated one on the Sink will start producing different predictions. To prevent the model synchronization loss problem, we coupled the previously proposed data reduction algorithm with a mechanism that can identify lost packets, force synchronization, and reconstruct missing data.

3. Third, Knowing that the sensing activity does also require a significant amount of power, we extended the previous algorithm and added an additional adaptive sampling layer. The idea is to first store in separated data sets the collected/predicted data over multiple periods (duration pre-defined by the user). Then, these sets are analyzed using the Kruskal-Wallis statistical test, which determines whether there

is a significant difference between them or not. If not, the sensor is considered to be collecting redundant information, therefore, using the Behavior Curve function a new reduced sampling rate (sleep time) is assigned to it. By doing so, the sensor will wake up less often, thus the wake-up, sensing, and transmission energies will be reduced.

4. Finally, we also proposed a decentralized data reduction approach for cluster-based sensor networks. Instead of distributing the burden of data-reduction on the deployed sensors, it is now the responsibility of the Cluster-Head (CH). The latter is considered to have higher energy and memory storage, and more computational powers. Every time a CH receives a new measurement from a member node, it stores the value locally then forwards it to the destination. After having collected enough data from the sensors over a predefined period of time, the CH computes the correlation among member nodes using the stored data. Finally, following specific rules, it assigns a new reduced sampling rate (sleep time) for highly correlated ones.

Real Implementation - Application to Firefighters 2.0 After developing the previously described energy management algorithms, it was time to apply them in the context of our project “Firefighter 2.0”. First, we had to propose and develop a remote monitoring system prototype that is adapted according to the needs and requirements of the fire brigade of the region of Doubs in France. Several interviews were held with the personnel from the fire and emergency response department that included discussions regarding the reliability, cost, and security of the system and the collected data. The interviews led to developing and validating the first prototype of a plug and play system that is specifically tailored for the needs of the fire brigade and also designed to work in areas with no network coverage. The system is composed of two sensors, one for localization and the other for environmental data collection. In addition, a smart-watch is used to monitor the condition of the firefighter (mainly heart rate and movement monitoring). Finally, as a communication protocol, we used LoRaWan for its proven advantages of being low-cost, long-range, and infrastructure-less. The previously described energy management algorithms were implemented on the environment monitoring sensor. Real current consumption has been measured and the final results are presented in Chapter 5.

Machine Learning for Decision making The last part of this thesis is Machine Learning (ML) oriented. In connection to the duty cycling energy management algorithm that relies on a cluster type of networks. It is essential to have “good” clusters for maximum efficiency. Moreover, clustering enables cluster analysis which is very helpful for decision making. In light of this, we surveyed the different clustering algorithms that could be

used for wireless sensor networks (or IoT in general) and we determined throughout simulated experiments on real sensor readings which one performs better. While most works present the well-known K-means approach as the state-of-the-art clustering method, our results demonstrated a different reality.

Lastly, for an efficient allocation of mobile and human firefighting resources, we investigate the possibility of predicting future incidents using machine learning algorithms that are trained on a set of data containing information on almost 200,000 firemen interventions that happened during the last 6 years. After pre-processing the data we tested multiple machine learning algorithms and we compared their results, aiming to determine which algorithm performs better. The results look promising as we were successfully able to predict the number of future interventions with an acceptable error margin.

Outline of this Dissertation

The dissertation is divided into three parts. Part I is dedicated to energy management in WSN and it is composed of five chapters. Chapter 1 presents a brief introduction on wireless sensor networks (WSNs) and provides a high-level taxonomy of energy management in WSNs in addition to a detailed state of the art of the related literature. Chapters 2, 3, and 4 describes the transmission reduction algorithm, its extension through combination with adaptive sampling, and the duty-cycling approach respectively. Finally, Chapter 5 presents the proposed remote monitoring system and illustrates the energy consumption results of the implementation of the energy management algorithms on real sensor devices. Part II is Machine Learning oriented and it is composed of two chapters. In Chapter 6 the work related to comparing different recent clustering approaches for IoT is presented. The case study of the prediction of the number of interventions is provided in Chapter 7. Lastly, In Part III, Chapter 8 concludes the work that has been done in this thesis.

I

Energy Management in Wireless Sensor Networks

1

ENERGY MANAGEMENT FOR WSNS: STATE OF THE ART

Sustainable Wireless Sensor Networks (WSNs) are becoming more of a reality due to the key driving energy management techniques. Since sensor nodes are typically powered by attached batteries, some of these techniques propose battery-driven energy conservation schemes to ensure energy-efficient network operation. However, the constraints associated to the limited battery capacity shifted the focus towards either exploiting alternative energy sources by harvesting ambient energy or developing data-driven algorithms aiming to minimize the sensor's most energy-consuming activities. In this chapter, we briefly introduce wireless sensor networks and we present a high-level taxonomy of their energy management. Moreover, a detailed state of the art on energy management in WSN is also presented.

1.1/ INTRODUCTION ON WIRELESS SENSOR NETWORKS

Wireless Sensor Networks (WSNs) can be defined as a self-configured and infrastructure-free wireless network that is typically used to monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, movement or pollutants [Estrin et al., 1999], and transmit their data through the network to a central workstation (Sink) where the data can be visualized and analyzed [Davis et al., 2012]. A Sink acts like an intermediate interface that enables the user to either retrieve information from the network through query injections or to gather information transmitted to it. The number of sensor nodes consisting of the network can range from tens to thousands [Cerpa et al., 2001, Shih et al., 2001, Petriu et al., 2000], depending on the application and they are mainly equipped with sensing and computing devices, radio transceivers, and power components. These sensor nodes are inherently resource-constrained, they have limited processing speed, storage capacity, communication bandwidth, and energy

resource [Kahn et al., 1999]. After deployment, the sensor nodes self-organize an appropriate network infrastructure, often with multi-hop communication, in order to transmit the data to be collected to the Sink. The working mode of the sensor nodes may be either continuous, periodic, or event-driven. Wireless sensor networks (WSNs) paved the way for new applications and, due to several constraints, require non-conventional paradigms for protocol design [Borges et al., 2014]. Owing to the requirement for low device complexity and low energy consumption, proper and efficient management of the communications and data processing capabilities must be found. This has motivated a lot of researches on the design of energy and computationally efficient algorithms and protocols for WSNs. There are several methods and approaches to manage energy consumption in WSN. In the next section, we will introduce these techniques, explain them, and provide examples of their application from the literature.

1.2/ State of the art on Energy Management in WSN

Power management in WSNs is known as the set of rules that control the different mechanisms of energy supply to ensure efficient consumption of the energy provided to the sensor. The overall objective is to control resources in such a manner that slows down or prevents the depletion of the energy of a node for the network to operate continuously and to ensure connectivity and coverage.

In this section, we describe a high-level taxonomy of how to take energy management into consideration while designing or developing an algorithm for Wireless Sensor Networks (WSNs). At the same time, we present a detailed state of the art of all the different approaches proposed in the literature aiming to manage energy consumption in WSNs. We start by breaking down the taxonomy in order to clearly present for the reader a general idea on what are the possible options to manage energy in WSNs.

The first option is to search for external energy sources to provide as much surplus in energy as possible to the sensor nodes and extend the lifetime of the network. This option is referred to as energy provisioning. Energy provisioning can be further classified into three main schemes, namely, energy sourced by batteries, energy harvested from the environment, and transference energy. The second option is to develop efficient energy management schemes on the basis of energy consumption. The latter assumes that no external power source is available, thus energy optimization algorithms must be implemented on the sensor nodes in order to extend their lifetime. These energy management schemes can be generally divided into four schemes, namely, data-driven, duty cycling, routing protocols, and mobility. A detailed discussion and a state of the art on each sub-classification of this taxonomy is presented in the subsequent sections.

1.2.1/ Energy Provision

1.2.1.1/ Battery Driven

The major and most viable power supply for the overwhelming majority of commercially available sensor nodes is a fixed battery, especially for sensor networks deployed in remote areas. Most of the sensor nodes are either compatible with Lipo or compacted sized AA batteries. Unfortunately, no matter how large is the capacity of these batteries or how efficient are the protocols implemented on the nodes, the energy will eventually drain. Therefore, in order to avoid interruption in communication or a network outage, these batteries must be periodically replaced. However, it is not always possible to do so. In the next subsections, we will present replaceable and irreplaceable fixed batteries.

Irreplaceable Fixed Batteries: Irreplaceable fixed batteries with limited capacity, are usually found in scenarios where the sensor nodes are deployed in harsh inaccessible environments or hostile military locations. It is then practically impossible to physically access the deployed nodes and replace their batteries. An example of such a scenario can be found in [Mahamuni, 2016] where the authors highlight the problem of senor nodes accessibility in military applications for battery replacement and proposed an energy-efficient military surveillance system that aims to maximize the lifetime of the network.

Replaceable Fixed Size Batteries: Other WSN deployment scenarios are accessible for battery replacement. The replacement can be either performed by humans or robots. The latter could be a plausible solution for human hazardous locations.

[Tong et al., 2011] proposed a node reclamation and replacement strategy, where mobile robots or human labors periodically check up the sensor network in order to reclaim nodes with low or no power supply and replaces them with fully charged ones. Finally, the reclaimed nodes are brought back to an energy station for recharging.

[Sheu et al., 2005] presented a designed smart mobile robot that navigates towards low-energy nodes and automatically replaces them with recharged ones. The navigation algorithm relies on the received signal strength in order to localize depleted nodes.

1.2.1.2/ Energy Harvesting

The energy provisioning trend is now shifting more towards alternative energy sources rather than relying on limited supply provided by fixed capacity batteries. These alternative sources are renewable energy, specifically in the form of energy harvesting from the

surrounding environment. There are various sources of natural and man-made energy that can be harvested by sensor nodes and converted to electrical energy. The different types of energy that can be harvested and examples for their application in Energy-harvesting Wireless Sensor Networks (EH-WSNs) is provided below.

Radiant Energy: Radiant energy, by definition, is the energy harvested from solar radiations and radio frequency (RF) waves. Both of these energy sources have been extensively explored in an attempt to utilize them in powering sensor nodes.

[Varghese et al., 2016] proposed an RF power harvesting battery management scheme for aerospace applications that can be used even in harsh real-time applications. Using special synchronized MAC protocol, low power techniques, low leakage components, and systematic coding of the microcontroller firmware, they were capable of keeping sensor nodes, located 2 meters away from the RF power source, operating continuously.

[Nishimoto et al., 2010] proposed an RF energy harvesting prototype that exploits TV broadcasts airwaves as the source harvested energy in order to power up the sensor nodes. In addition, they propose an adaptive duty cycle scheduling method to turn on the radio module only when it is necessary in order to reduce furthermore energy consumption.

[Kaushik et al., 2016] proposed a low-cost wake-up receive hardware design for RF energy harvesting sensor nodes, capable of performing both range-based wake-up and directed wake-up using a single hardware.

[Li et al., 2015] proposed an intelligent solar energy-harvesting system that it is designed using Maximum Power Point Tracker (MPPT) circuit. The system is composed of a solar panel, a lithium battery, and a control circuit. The system can afford a stable power supply with 5-V output voltage through a standard USB interface. The Lithium battery-charging strategy is designed to avoid as much as possible the charge-discharge cycle, which greatly improves its lifetime. The system has been finally proven, through experiments, to have a stable and safe performance. Moreover, it has been demonstrated that the system has high reliability, high efficiency, and low power loss.

[Alippi et al., 2011] proposed a WSN framework for marine environment monitoring. The design of the system included all the essential aspects such as sensing, transmission, data storage, and visualization. But most importantly, each sensor node was equipped with a solar-energy-harvesting mechanism in order to harvest energy from the sun and prolong the lifetime of the network.

[Li, 2018] proposed an optimized solar energy harvesting system. The optimization process concerned the capacity of the supercapacitor and the size of the solar panel. The proposed system that is used to monitoring the environment in Sundsvall, Sweden, was proven through simulation to be low-power consuming and secure.

Mechanical Energy: Mechanical energy is another source of energy that can be harvested. Typically, it is harvested using piezoelectric devices. Numerous piezoelectric based-systems and other mechanical approaches for energy harvesting in WSNs have been developed throughout the years.

[Kahrobaee et al., 2013] proposed an EH-WSN for agriculture applications. It utilizes underground piezoelectric devices to harvest energy generated by the vibrations of an above-ground four-wheeler center pivot irrigation system. Moreover, they argue that the same technique can be used to harvest energy from road vibrations in road monitoring WSN applications.

[Zhu et al., 2012] proposed an energy harvesting system that generates energy from road vibrations caused by automobiles using piezoelectric devices. This technique is proposed as a viable solution for WSN applications dedicated to monitoring various automobile physical parameters such as ultrasonic sensors for distance detection, tire pressure monitoring, and other applications.

[Wijesundara et al., 2016] proposed an EH-WSN system to track and localize elephants in their habitats. The sensors are powered by the harvested Kinetic energy generated by the elephants' movements. The proposed system consists of a moving magnet, two fixed magnets, a poly-carbon tube, and two coils. Through experiments, they demonstrated that their system is capable of generating enough power for a mounted tracking sensor to transmit its location up to 24 times a day.

[Ye et al., 2011] proposed an EH-WSN system to harvest hydraulic energy using a microturbine installed in bypass tubes of water distribution systems. However, this method is found to either wastewater (releasing water from the water distribution system) or cannot generate enough power in the level of mW. Therefore, it remains partially impractical.

Thermal Energy: Thermal energy is produced when there is a temperature difference between two conducting materials. Many environmental studies on thermal powered sensor nodes have been conducted throughout the years.

[Woias et al., 2014] proposed a thermo-electrical generator (TEG) to track sheeps. The

generator is mounted on the sheep's collar and harvests thermal energy generated by the difference in temperature between the sheep's body and the ambient temperature. The system has been proven to generate enough power in order to power a low-power radio tracking system.

[Davidson et al., 2009] proposed a thermally powered sensor node that generates thermal energy by exploiting the difference between the ocean water surface and the ambient air to monitor ocean water quality. The average energy output is shown to be enough to potentially power aquatic sensors.

[Hou et al., 2018] proposed a thermal energy harvesting system for industrial IoT temperature monitoring applications. The obtained results demonstrate that the designed TEG-based system has an acceptable energy conversion rate, and is capable of indefinitely power a commercial sensor node when the sleep period of the device exceeds 16s.

Hybrid Energy-Harvesting Systems: We can also find in the literature some research related to hybrid energy-harvesting systems. The latter combines multiple types of energy harvesters into a single hybrid system. We list some of these existing approaches.

[Yu et al., 2008] proposed a solar/thermal hybrid system to power sensor nodes for WSN applications. The latter consisted of a solar panel and a TEG. In order to store energy, a Li-ion battery and a super-capacitor are used. Moreover, the system included a power management subsystem. The results demonstrated that the proposed solution could prolong the lifetime of a sensor node up to 5 years under normal sunny weather. However, it only lasts for 7 days when it is rainy and dark.

[Tan et al., 2010] proposed a hybrid indoor artificial lightning and thermal energy harvesting system. The latter consisted of a solar cell, a miniature TEG, and a power management circuit. The obtained results demonstrated that the proposed system can harvest three times more energy than a conventional single thermal energy harvesting one.

[Chottirapong et al., 2015] proposed a solar/thermal harvesting hybrid system for indoor ammonia gas concentration measurements in an organic fertilizer plant. The source of the energy harvested by the TEG is the heat difference between the hot side and the cold side of the deodorizer tank.

1.2.1.3/ TRANSFERENCE

With the recent breakthrough in the area of wireless energy transfer technology, energy management in WSNs took a new dimension. A technique called magnetic resonance enables wireless energy transfer from one storage device to another without any plugs or wires. This energy transference technique works regardless of the neighboring environment and does not require a direct line of sight between the energy transferring and receiving node. This technology has a positive impact on energy-constrained wireless networks. Instead of harvesting energy locally at a node, efficiently generated energy can be brought wirelessly to a sensor node periodically in order to charge its battery. Some of the efforts carried out in terms of wireless energy transference in the last few years are presented in this section.

[Mascarenas, 2008] developed a mobile charging host capable of transferring wireless electrical energy to remote nodes on a 2.4 MHz signal. This host can also retrieve the data collected by the deployed sensor nodes. The system has been tested and it was able to charge the deployed sensor nodes condensers with no less than 12 seconds for an assumed distance of two meters between the host and the charging node. However, the receiving and transmitting antennas used in the experiment are 18.7x3 and 15x15 inches in size respectively, which makes it impractical in terms of size and cost.

[Doost et al., 2010] have made some similar efforts, by transferring wireless energy through electromagnetic waves to sensor nodes equipped with rechargeable batteries. Experiments have been conducted to study the impact of distance and the nodes' location on the efficiency of the system. It was reviled through testing that a distance greater than 12m would render the system obsolete since it would take an infinite time to charge a battery. Moreover, the relative placement of the nodes plays a big role in reducing the charging time. In addition, a modified version of the Ad-hoc On-Demand Routing (AODV) is proposed as an energy charging cycle aware routing algorithm. Finally, in order to address the trade-offs of the charging and transmission duration that they occur on the same frequency band, an optimization framework is also proposed.

[Guo et al., 2013] proposed and approach to further enhance the network lifetime by optimizing the charging process for the deployed nodes. Instead of having a fixed data collection rate, a sensor node would rather tune its data rate based on the current energy replenishment status. Moreover, the Wireless Charging Vehicle (WCV) responsible for charging the sensor nodes dynamically adjusts its visiting time for each node.

[Watfa et al., 2011] proposed charging the sensor nodes using multi-hop energy transfer. In order to achieve that, the magnetic resonance-based technology, "Witricity" is used.

Three different energy transferring techniques are proposed, namely, store and forward, direct flow, and hybrid schemes. In-store and forward, the energy received by an intermediate node is first stored in a buffer and when the battery is fully charged it forwards the energy to the next hope. Indirect flow, the intermediate node directly forward the received energy to the next hope. Finally, the hybrid is the combination of the previous two approaches and it showed to be the most efficient.

[Summerer et al., 2009] demonstrated a LASER reflection-based energy transference, where energy is transferred from a host to a sensor node using laser beams.

1.2.2/ Energy Consumption

1.2.2.1/ Data Driven

The elements of energy dissipation in wireless sensor nodes that monitor a specific environment and report the collected information to a central workstation are numerous. However, in the monitoring stage, the micro-controller, sensor and radio components dominate the energy consumption and they are the main optimization targets. Therefore, the research community has proposed, developed and adopted several approaches to improve the battery-driven device's efficiency aiming to keep the network perpetual. In this section, we present the key approaches that include data aggregation, data compression, adaptive sampling, and data prediction.

Data Compression and Aggregation: Due to the nature of Wireless Sensor Networks, the reported measurements collected by the deployed sensor nodes are highly redundant. This redundancy offers no benefits for the end application, and it has a significant downside on the lifetime of the network. The redundant readings require energy to be transmitted to the Sink station. Why bother to lose scarce energy transmitting these readings while we can simply ignore them? In order to address this problem researchers have proposed compression and aggregation techniques that aim to eliminate redundant information from the source in order to reduce energy consumption and extend the lifetime of the network.

[Marcelloni et al., 2008] proposed a simple compression algorithm that assumes a high exploitable correlation between consecutive sampling. Following the principles of entropy compression, the proposed algorithm is able to compress sensed values on-the-fly. This algorithm is inspired by the baseline JPEG algorithm for compressing the DC coefficients of a digital image. It is lossless and compresses the data using a small size dictionary.

[Medeiros et al., 2014] proposed a lightweight compression mechanism for low resolution sensor nodes based on fixed Huffman encoding. Instead of using a specific dictionary for each new data-set, they argue that using a general dictionary created using a similar data-set would give a very close compression ratio. This makes the proposal modest in terms of computational power and memory requirements.

[Wu et al., 2017] proposed a holistic approach to compress and reconstruct data in ocean sensor network using Compression Sensing. It is based on an algorithm that re-orders the sensor nodes in such a way that improves the signal sparsity property in DCT or Fourier Transform Domain and improves the compression rate of the data.

[Basheer et al., 2017] proposed a cluster-based quality-aware adaptive data compression scheme, that takes into account the application's data quality and it also limits information loss by using adaptive clustering and novel coding algorithm.

[Uthayakumar et al., 2019] proposed a lossless compression scheme called Neighborhood Indexing Sequence (NIS) algorithm. This applies a procedure called "traversing data based on 0's and 1's" to produce a shorter length code-words for the characters in the input sequence. The algorithm adopts a method that decompresses each packet individually which makes it robust to packet losses.

[Du et al., 2015] proposed Dynamic Message List Technique Based Data Aggregation (DMLDA). This method is based on a data clustering algorithm and it provides real-time data aggregation of collected measurements. A special data structure called a dynamic list is the cornerstone of this mechanism. This list is used for storing history messages before transmission in every filtering node.

[Al-Karaki et al., 2004] proposed a hierarchical model that uses data aggregation and in-network processing in order to maximize the network lifetime. The sensor network is subdivided into two subsets. The first is called the Local Aggregators (LAs) which are a set of sensor nodes elected to form the routing path on which the first level of aggregation is performed. The second set, called Master Aggregators (MAs), is then selected to perform the second level of aggregation. In addition, in order to find the MAs, which is an NP-complete problem, they propose three near-optimal approximation algorithms.

[Çam et al., 2006] proposed an Energy-efficient and Secure Pattern-based Data Aggregation (ESPDA). This algorithm has two independent levels of operation. On a first level, this algorithm allows the Cluster Head (CH) to organize a sleep schedule for the deployed sensor nodes transmitting redundant information. On the second level, the CH uses pat-

tern codes to perform data aggregation. The usage pattern code instead of actual sensor data allows the content of the transmitted data to remain disclosed, which enables ESPDA to work in conjunction with the security protocol.

[Chao et al., 2014] proposed a structure-free and energy-balanced data aggregation protocol (SFEB). As an event occurs, data aggregators are dynamically selected using a lightweight aggregator election algorithm. Once selected, they collect sensory data from their neighbors to finally transmit them to the Sink station using an enhanced, structure-free version of Data-Aware Anycast and Randomized Waiting protocol (DAA+RW).

[Khorasani et al., 2017] study the capability Neural of Networks in aggregating sensory data. They implement and analyze four algorithms namely, Moments Estimation Data Aggregation (MEDA), Levenberg– Marquardt Training Back Propagation Network (LMTBPN), Radial Basis Data Aggregation (RBDA), and General Regression Data Aggregation (GRDA). According to the simulations, all of these algorithms show some promising results regarding energy efficiency in WSN.

[Humidi et al., 2019] proposed a lightweight, low latency data transmission reduction scheme based on data aggregation. Similarly to the approach proposed in [Al-Karaki et al., 2004], a centralized control algorithm is used to select the optimal Cluster Heads (CHs). Then, the selected CHs performed a first-level of data aggregation, followed by a second-level aggregation performed by a selected optimal subset of CHs.

Adaptive Sampling: The main objective of the aggregation and compression techniques is to eliminate the already collected but redundant sensor readings. What if we can avoid collecting these readings in the first place? This is where the adaptive sampling technique is presented as a solution. The latter allows the sensor node to dynamically increase or decrease its sampling rate according to the level of variance between collected measurements over a certain period of time. This approach prevents the sensor from collecting redundant information. The fewer are the collected readings, the less sensing and transmission is required, hence, reduced energy consumption. In this section, we list some of the proposed adaptive sampling approaches.

[Willett et al., 2004] proposed an adaptive sampling technique called “Backcasting”. The proposed sampling adaptation algorithm assumes the field being sensed is supported on one square meter, and that the sensors were deployed uniformly over the square. In the first phase a number of sensors are activated to produce a coarse estimate of the field at the Sink. Afterward, based on the coarse estimate, the Sink determines which regions of the field may contain boundaries or sharply varying behavior, and activates additional sensors in those regions.

[Alippi et al., 2009] proposed an adaptive sampling algorithm that estimates in real-time the optimal sampling frequency for each deployed sensor node. The algorithm adopts a modified version of the cumulative sum (CUSUM) to detect changes in values in non-stationary data and adapts the sampling rate accordingly.

[Nguyen et al., 2015] proposed an optimality criterion for designing a sampling strategy that finds the most informative locations in taking future observations for physical phenomena monitoring WSN applications. To achieve this, a Gaussian process (GP) is used to statistically model the physical spatial fields. This model is then used to produce observation predictions. This is where the optimal criterion is proposed to solve the sampling path optimization problem in a practically feasible time.

[Makhoul et al., 2015] proposed an adaptive sampling approach based on the dependence of conditional variance on measurements. Three different statistical tests (Fisher, Tukey, and Bartlett) for variance detection based on the One Way Anova Model are studied. If the tests indicate high variance, the new sampling rate is then determined using the Bezier curves while taking into account the residual energy level of the sensor node.

[Masoum et al., 2013] proposed an adaptive sampling approach for the applications that can tolerate variations in the collected measurements as long as these variations do not exceed a fixed error threshold. The proposed algorithm uses the Spatio-temporal correlation of the collected data in order to determine how often should a sensor sample and transmit.

[Monteiro et al., 2017] proposed a technique named Dual Prediction with Cubic adaptive sampling (DPCAS). They adopted an approach that merges an exponential time series predictive model with a TCP CUBIC congestion adaptive sampling technique. This approach enables the sensor to reduce the number of transmissions using the predictive model and adapt as well its sampling rate using the TCP CUBIC congestion technique.

[Yang et al., 2016] proposed an approach that takes into account both the system and the application context levels to adapt the sampling rate of the sensor nodes. For example, the energy available for harvesting represents the context of the system. One of the used criteria to set the node's maximum sampling rate is his availability. The application context is reflected by the user's request, where feedback from the system performing specific rules is used to optimally set sensor nodes sampling rates.

[Harb et al., 2018] proposed three distinct data collection and adaptive sample techniques in the context of industrial process monitoring. One uses ANOVA, while the second uses

similarity sets and the third uses distance functions in order to capture similarities in the collected data and adapt the sampling rate accordingly.

[Bhuiyan et al., 2017] proposed an event-sensitive adaptive sampling and low-cost monitoring (e-Sampling) scheme, where each sensor has short and recurrent bursts of high sampling rate in addition to a low sampling rate. Depending on the analysis of the frequency content of the signal, each sensor can autonomously switch between the two sampling speed.

Data Prediction: The principle of data prediction in Wireless Sensor Networks (WSNs) is to construct a predictive model that is capable of forecasting future measurements. This enables the sensor node to compare the sampled measurement with its matching prediction and transmit this measurement only when the prediction is deemed inaccurate. This model could be built either locally at the sensor level or at the destination server (Sink) using a collected training set. In both cases, the constructed model is re-shared between the two. Alternatively, the model could be built simultaneously on both entities to prevent communication overhead. This technique is generally referred to as the Dual Prediction Mechanism (DPM).

Several DPM model building approaches are presented in the literature. They all share the same assumption that the radio component of the sensor node used to transmit data consumes a substantial amount of energy. Therefore, they all aim to reduce the amount of transmitted data in order to preserve scarce energy and extend the lifetime of the network. In this section, we present a general overview of all the available techniques.

[Guestrin et al., 2004] proposed to build a prediction model on the sensor node using the kernel linear regression. Afterward, the node transmits the coefficients of the regression model to the Sink. The latter then can proceed to predict future measurements without having to receive anything from the sensor node. However, if the model is no longer producing accurate predictions, the sensor node re-trains it and re-transmits the new coefficients to the Sink. The obtained results showed that the proposed approach converges rapidly to the optimal solution and is robust to packet loss.

[Li et al., 2013] the authors adopted the Auto-Regressive Integrated Moving Average (ARIMA) as a forecasting model. In a similar fashion to the Previous approach, the sensor node will first build a time series forecasting model using recently collected data values, then it shares the parameters of the model with the Sink in order to proceed with predictions. Through experiments on a real data-set collected from TAO (Tropical Atmosphere Ocean) project, the model has shown to fit well the sensed data and is capable of reducing transmissions.

[Moghadam et al., 2011] proposed a hybrid model by combining both ARIMA and a Neural Network in order to cover the limitation of the former in adapting to non-linear changes. Experiments were conducted on real-world data-sets from the Intel Berkeley research lab and it shows that the presented approach outperforms other proposed hybrid models.

[Santini et al., 2006] proposed a DPM that is based on the Least Mean Squares (LMS) adaptive filter. The authors claim that their approach can reduce the number of transmissions up to 92% while maintaining a minimal accepted accuracy. These claims were backed by experiments conducted on a real data-set from the Intel Berkeley research lab.

[Tan et al., 2016] proposed the usage of Hierarchical Least Mean Squares (HLMS) adaptive filter as a dual prediction mechanism, which is a multi-level LMS filter that has been proven to achieve faster and more stable convergence than the regular single LMS filter. This is partially achieved due to a dynamic computation of the step size parameter (μ) that controls the convergence and the stability of the LMS filters.

[Wu et al., 2016] proposed an approach called the Optimal Step Size LMS (OSSLMS). The idea was to dynamically calculate the value of the step size parameter (μ) of the LMS filter by trying to minimize the mean-square derivative at each iteration. This has proven through simulation on a real data-set to achieve faster convergence and a more stable forecasting LMS filer.

[Raza et al., 2015] proposed a Derivative Based Prediction (DBP) approach. It is less complex than the adaptive-filter based approaches. The goal is to find the best line interpolating a window of data of size m, by connecting the mean value of the first and last l measures in this window. Once this "optimal" slop is found, the predicted values are considered to follow the same direction as the latter. Experiments on 4 different real data-sets have demonstrated that their approach can outperform other well-established ones.

[Fathy et al., 2019] proposed an Adaptive Method for Data Reduction (AM-DR), it is based on a DPM consisting of a convex combination of two adaptive filters with different learning window sizes. Similar to the other DPM approaches, AM-DR exploits fine-grained sensor readings in order to reconstruct the whole data-set. The proposed approach outperformed the baseline LMS method in experiments conducted on three different real-world data-sets.

[Aderohunmu et al., 2013] proposed a naive approach where the predicted value is considered to be exactly the same as the last received/transmitted one. Therefore, no model

needs to be built. The authors wanted to demonstrate that sometimes simple models can be as effective as more complex ones. Indeed, in their experimentation, they outperformed in most scenarios the LMS, ARIMA, and the fixed-Weighted Moving Average (WMA) approaches.

1.2.2.2/ DUTY CYCLING

Duty cycling, which is also referred to as scheduling, is considered to be the most effective way of improving the network's lifetime. The idea is to adjust the duty cycle of nodes by alternating between sleep and wake-up modes in order to preserve as much energy as possible. Instead of keeping a node awake when there is no processing, sensing, nor communication in progress, the latter is put into low power mode or deep-sleep. In the deep-sleep mode, the processor, co-processor, radio models, and peripherals are turned off, only the RTC memory is kept awake allowing the node to store data to be retrieved after wake-up. This process reduces drastically energy consumption since these components are the major energy consuming components of the node. Duty cycle based algorithms can be further categorized as topology control protocols, sleep/wake-up protocols, and MAC protocols. In this section, we will present some of the research works related to these approaches.

Topology Control Protocols: Topology control Protocols aim to reduce the network topology by turning off redundant nodes, for a certain period of time, while preserving network connectivity or ensuring coverage. Below we list a few approaches proposed for this matter.

[Xu et al., 2003] proposed Geographic Adaptive Fidelity (GAF) and Cluster-based Energy Conservation (CEC) as topology control protocols. GAF determines node redundancy based on the geographical position of the nodes. CEC organizes nodes into overlapping clusters where the members of the cluster are a subset of nodes that are mutually reachable in at most two hops. A detailed description and extensive analysis and comparison of these protocols have been conducted including energy conservation, network connectivity, sensitivity to network density, protocol overhead, and sensitivity to the propagation model.

[Mochaourab et al., 2008] proposed a localized algorithm that enables sensor nodes to self-organize in order to build an overall reduced network topology. Each node in the network chooses its appropriate neighbors according to a defined distance measure, which is dependent on the radio characteristics and the channel conditions while taking into

consideration the energy reserve of the neighboring node.

[Qureshi et al., 2011] proposed a topology construction protocol based on the idea of polygons (Poly). The proposed approach aims to form a Connected Dominating Set (CDS) by finding polygons formation of nodes. Poly was compared to three well established CDS-based topology construction protocols namely CDS-Rule K, Energy-efficient CDS (EECDS) and A3. The obtained results showed that the proposed approach has low message overhead and energy consumption, and can provide higher network reliability.

[Chiwewe et al., 2011] proposed a distributed topology control technique, where each node in the network makes a local decision about its transmission power using the Smart Boundary Yao Gabriel Graph (SBYaoGG) optimization algorithm. The culmination of these local decisions leads to the production of a network topology that preserves global connectivity while minimizing energy consumption.

[Deniz et al., 2016] proposed a distributed fault-tolerant topology control algorithm for heterogeneous WSNs, called Adaptive Disjoint Path Vector (ADPV). The role of this algorithm is to ensure the connectivity of resource-rich super-nodes. This is done by dynamically alternating through routing paths, that are computed during an initialization phase, each time the connectivity is broken. Moreover, the proposed algorithm takes into consideration the possibility of node failure while choosing the new routing paths.

[Zhang et al., 2019] proposed a 3D topology-control algorithm for underwater sensor networks, where sensor nodes are suspended at different depths in the water for monitoring purposes. The sensor nodes are first partitioned into units and clusters. Then, the algorithm arranges wake-sleep schedules within each cluster based on the residual energy of each sensor node.

Sleep/Wake-up Protocols: The sleep and wake-up protocols can be generally classified into on-demand protocols where nodes only wake up when they are needed for communication. Scheduled rendezvous protocols, where nodes wake-up on a synchronized, scheduled rendezvous with neighbors. Asynchronous protocols where the nodes wake-up independently of their neighbors without requiring global clock synchronization. Finally, data correlation-based scheduling protocols that aim to set asleep, periodically, sensor nodes showing a high-level correlation with neighboring ones. In this section, we present some of the research work dedicated to these approaches.

[Liang et al., 2007] proposed a passive RF wake-up (PRFW) scheme for on-demand wake up of sensor nodes. A low power radio model embedded into the node is always

on and listening for preambles. Once detected, the model gives an interrupt signal to the Micro Controler Unit (MCU) for wake-up.

[Bdiri et al., 2015] proposed to integrate into the sensor nodes an ultra-low-power radio Wake-Up Receiver (WuRex) for real-time applications. The latter has low wake-up packet decoding latency and approximately -60 dBm sensitivity. It remains active to listen to on-demand wake-up queries instead of the main energy-consuming RX transceiver that is only waking up when a transmission is needed.

[Lin et al., 2004] proposed two generic rendezvous schemes for dense wireless sensor networks, namely TICER (Transmitter Initiated CyclEd Receiver) and RICER (Receiver Initiated CyclEd Receiver).

[Kosanovic et al., 2013] proposed a rendezvous-based wake-up scheme that provides synchronous wake-up times for sensor nodes in a fully decentralized manner. the proposed scheme can rapidly detect new appended sensor nodes and has an implemented mechanism that prevents collision avoidance during the registration of a new node.

[Zheng et al., 2003] proposed a neighbor discovery and schedule bookkeeping protocol. First, the problem of generating wake-up schedules is formulated as a block design problem. Then, theoretical bounds are derived under different communications models. The optimal obtained results are then used to design the proposed asynchronous protocol. This can detect neighboring nodes infinite time without requiring slot alignment. Moreover, it is resilient to packet collision and network dynamics. Finally, two power management scheme are considered and stacked on top of the designed protocol.

[Dutta et al., 2008] proposed an asynchronous neighbor discovery and rendezvous protocol (DISCO). This latter allows sensor nodes to activate their radio modules at very low duty cycles, while still being able to discover and communicate with one another during infrequent, opportunistic encounters without requiring any prior synchronization information.

[Villas et al., 2013] proposed an Efficient Data Collection Aware of spatial-temporal Correlation (EAST). In this latter, the sink subdivides the event area into spatially correlated cells of the same size, then, in each cell, the node having the highest residual energy is elected as a representative node. This transmits data to the sink while also applying a temporal correlation suppression method on its collected data. Finally, the representative node is re-elected periodically according to the same previous rule.

[Karuppasamy et al., 2013] proposed a sleeping schedule algorithm that aims to minimize the total spatial-temporal coverage redundancy among neighboring nodes while maximizing coverage. Each sensor node compares itself with neighboring ones using weight criteria and it locally optimizes its scheduling according to its coverage redundancy.

[Dhimal et al., 2015] proposed a spatial-temporal correlation model that aims to extend the network lifetime by scheduling a sleeping period for sensors showing high similarities with other ones belonging to the same cluster. The similarity is measured by computing the Euclidean Distance, Cosine Similarity and Pearson Product-Moment Coefficient (PPMC). If the result of one of the three indicates a high similarity, the sensor node is set to sleep for a certain period of time.

MAC Protocols: MAC protocol duty cycling approaches can be classified as Time Division Multiple Access (TDMA) based, contention-based, and hybrid MAC protocols. In TDMA, the nodes' duty cycle is enabled only when channel access is required. In Contention based protocols duty cycling is achieved by integrating medium access functionality with sleep or wake-up process. Finally, Hybrid schemes refer to algorithms combining properties from both, TDMA based and contention-based schemes. In this section, we present some of the proposed methods related to these approaches.

[Song et al., 2009] proposed an Energy-efficient localized TDMA MAC protocol (TreeMAC) for sensor networks that aims to achieve high throughput and low congestion for high-data-rate sensor networks. The proposed algorithm divides a time cycle into frames and each frame into slots. A parent node determines the children's frame assignment based on their relative bandwidth demand, and each node calculates its own slot assignment based on its hop-count to the sink.

[Louail et al., 2016] proposed a cross-layer TDMA MAC protocol for sensor networks that uses the information of the routing protocol from the network layer, in order to enhance the performance in terms of latency of the scheduling TDMA MAC protocol situated at the data link layer.

[Liu et al., 2005] proposed an energy-efficient, QoS-aware MAC protocol for sensor networks. It consists of intra-node and inter-node scheduling. The former adopts a multi-queue based queuing architecture that uses the MAXMIN fairness algorithm and the packetized GPS algorithm to determine the next packet to be served. The latter employs the power conservation MACAW protocol and the loosely prioritized random access protocol for multiple access of the channel among neighboring sensor nodes.

[Rao et al., 2018] proposed a self-adaptive implicit contention window adjustment mechanism for the Sensor MAC (SMAC) protocol. This latter optimizes the model for the contention priority distribution of the nodes. Moreover, it integrates a delay-oriented optimizing contention window, contention priority optimization model with the estimation method of contenders' number in order to reduce the idle listening duration and collision probability.

[Salajegheh et al., 2007] proposed a hybrid MAC layer protocol for sensor networks (HYMAC) that combines the strengths of both TDMA and FDMA. The communication period in HYMAC is a fixed-length TDMA cycle composed of a number of frames that are further divided into several fixed slots. In addition, a fixed number of consecutive slots at the beginning of each cycle are scheduled slots while the remaining slots of that cycle are contention ones. The appropriate frequency, as well as a specific time slot(s) of each node, are assigned by the Sink station.

[Rhee et al., 2008] proposed Zebra MAC (Z-MAC) a hybrid MAC layer protocol for sensor networks that combines the strengths of TDMA and CSMA while offsetting their weaknesses. Under low contention, Z-MAC behaves like CSMA, and under high contention, like TDMA, Z-MAC uses CSMA as the baseline MAC scheme but uses a TDMA schedule as a "hint" to enhance contention resolution. Thus, in the worst case, its performance always falls back to that of CSMA.

[Rhee et al., 2008] proposed a hybrid time division multiple access/carrier sense multiple access (TDMA/CSMA) protocol for intra-satellite wireless network (ISWN). The network works in a periodic mode, where both centers controlled scheduling and contention access are adopted. Sensor nodes are categorized into 3 types, namely, A for high data rate, sensitive and insensitive nodes, B for low data rate sensitive, and C low data rate insensitive nodes. The main communication phase provides center scheduled slots that are assigned by the nodes' ID. Nodes in type A and type B will communicate in this period. By contrast, the extended communication phase provides contention-based slots, in which an improved slotted CSMA/CA strategy is performed and nodes in type C will transmit their packet in this period.

1.2.2.3/ ROUTING PROTOCOLS

In a large scale WSN, on the one hand, if we consider a star topology that enables a direct transmission from the source to the destination, the farther is a node from the destination Sink, the more energy it will consume. This is because the transmission energy is directly

related to the transmission distance. On the other hand, if we consider a cluster-based mesh topology, then each node will forward the packet it desires to transmit to its closest neighbor, which in turn moves it forward in the cluster hierarchy to finally reach the Sink. Thus, the higher is a node in the hierarchy, the more packets it needs to forward, the more energy it will consume. Therefore, nodes that are closer to the Sink will drain much faster than the ones that are farther away. In order to solve this problem, energy-efficient routing protocols are proposed. The purpose of these algorithms is to extend the lifetime of the network by equally distributing the transmission load on the deployed sensor nodes in such a way that it ensures balanced energy drainage regardless of the distance between the node and the Sink. In this section, we will present some of the research work related to energy-efficient routing in WSNs.

[Muruganathan et al., 2005] proposed a centralized routing protocol called Base-Station Controlled Dynamic Clustering Protocol (BCDCP). This assumes a base station with sufficient energy to set up clusters and routing paths, performs a randomized rotation of cluster heads, and carries out other energy-intensive tasks. BCDCP produces balanced clusters with an equal number of member nodes, uniformly selects clusters through the sensor field, and utilizes cluster-head to cluster-head communication to transfer packets to the Sink.

[Xiangning et al., 2007] proposed two improved versions of the Low-Energy Adaptive Clustering Hierarchy (LEACH) protocol. Namely, energy-LEACH and multihop-LEACH protocols. The former makes the residual energy of the node as the main criteria which decides whether these nodes are selected as a cluster head or not in the next round. The latter selects the optimal path and adopts a multi-hop between the cluster heads and the Sink.

[Wang et al., 2012] proposed a link-aware clustering mechanism (LCM), that determines an energy-efficient and reliable routing path. This mechanism relies on the transmit power consumption, residual energy, and link quality of the nodes to compute a clustering metric called “predicted transmission count (PTX)”. This latter is then, used to derive a priority that decides whether a node can be qualified as a cluster-head or not.

[Amiri et al., 2014] proposed an optimal routing protocol for WSN based on the Fuzzy Ant Colony Optimization Routing (FACOR). A source node sends an “ant”, which is modeled by an RREQ and an RREP message, to all of its neighbors. Then, each ant tries to find a route to the Sink by calculating the fuzzy amounts for their neighbors and choosing the next hope accordingly. The same step is repeated until the ants finally reach the Sink. While, the ones that could not reach it, they kill themselves. Finally, the BS chooses a

winner ant and returns it through the same path it came from to update the routing table of the intermediate nodes.

[Zeng et al., 2016] proposed an Improved Harmony Search-Based Energy Efficient Routing Algorithm (IHSBEER) for WSNs. The improvement included the encoding harmony memory and the improvisation of a new harmony. Moreover, the authors introduced dynamic adaptation for the Harmony Memory Considering Rate (HMCR) parameter and a local search strategy is proposed to enhance the local searchability. Finally, an objective function model that takes both the energy consumption and the length of the path in consideration is developed which has a great impact on the maximization of the network lifetime.

[Ayoub et al., 2017] proposed a Multi-Hop Advance Heterogeneity-aware Energy Efficient (MAHEE) clustering path planning algorithm for WSNs. MAHEE routing protocol operation is divided into time-based iterations, and each iteration is divided into rounds. At each round, the Sink selects the appropriate cluster-heads based on Hello packets broadcasted by the nodes and containing information about initial energy, remaining energy, location of node and status (active or dead).

[Wang et al., 2019a] proposed a special clustering method called Energy Centers searching using Particle Swarm Optimization (EC-PSO) that aims to optimize the heavy burden of forwarding on the cluster-head level. The protocol operates in rounds, and during the first rounds, cluster-heads are elected according to the location of nodes using geometric partitioning. After the energy of the network becomes heterogeneous, the Particle Swarm Optimization is used to search energy centers for the cluster-head election. Moreover, random re-initialization is used to avoid cluster-heads getting too close and a protection mechanism using threshold value is utilized to prevent the low energy nodes from forwarding.

1.2.2.4/ MOBILITY

In contrast to the previously described data routing approaches that aim to distribute the message forwarding burden on the sensor nodes of the network in such a way that ensures a maximum lifetime. Mobility based approaches aim to eliminate message forwarding by introducing mobile agents with high energy resources. This moves through the field where the sensor nodes are deployed in order to directly collect the data from the nodes through short-range communications that require no routing. These agents can either be a mobile Sink, or mobile relay nodes. In this section, we will present some of the research work adopting this technique.

[Kim et al., 2003] proposed a Scalable Energy-efficient Asynchronous Dissemination protocol (SEAD). SEAD builds and maintains a dissemination tree (d-tree for short) that a mobile Sink can traverse to collect data from the nodes. Stationary nodes are used as end-points on behalf of the mobile Sink and the adopted scheme caches sensed data in the d-tree in such a way that reduces the energy consumption.

[Liang et al., 2010] proposed a three-stage heuristic to optimize the mobility of the mobile Sink. First, the time for a mobile sink to collect data at each potential location is calculated. Then, a feasible tour is computed for the mobile sink in such a way that the sum of the times spent at the visited locations is maximized. Finally, a visit time schedule is created for the mobile Sink by determining its exact visit time at each location.

[Wang et al., 2018] proposed an improved ant colony optimization (ACO) based approach. The proposed approach considers the distance heuristic factor between the cluster heads and the distance to the mobile Sink. This considerably enhances the global search ability, prevents the algorithm from being trapped in a local optimal solution, and improves the convergence rate. On top of ACO, a clustering algorithm is added. The mobile Sink finds an optimal sink mobility trajectory by the improved ACO algorithm to communicate with the cluster heads only and collect data from them via short-range communications.

[Zhao et al., 2004] proposed a mobility-assisted approach called Message Ferrying (MF) which provides a communication service for the deployed sensor nodes using a set of special mobile nodes called message ferries. Two schemes are presented, the first is where the message ferries move according to a specific route that is known by nodes that take proactive movement periodically to meet up with the ferries. The second scheme is where the ferries take proactive movement to meet up with nodes after broadcasting their locations.

[Fayçal-Khelfi et al., 2016] proposed a solution to the problem of mobile data collectors alleviating high traffic load to the Sink and causing bottleneck and packet loss. First, the impact of the mobile data collector (mobile Sink or relay nodes) in terms of energy efficiency, latency, and traffic load, have been studied to determine the best approach. Then, three well-established mobility models have been compared, namely Gauss-Markov, Random Walk, and WayPoint. The conducted simulations demonstrated that using mobile relays node and models containing random movements give the best results.

[Wang et al., 2019c] proposed a clustering algorithm for 3D underwater sensor networks with mobile nodes. The network consists of mobile Sinks, mobile relay nodes and fixed

sensing nodes. First, the fixed nodes are organized into clusters and appropriate cluster-heads (CHs) are elected. Afterward, mobile relay nodes can pass to collected data from the cluster CHs and transmit them to the mobile Sink, which in turn forwards the packets to the base station.

1.2.3/ CONCLUSION

This chapter presented a brief introduction on WSNs and a survey of the relevant literature for their Energy management. The latter was classified into two main categories, energy provisioning, and energy consumption. Both categories share the same objective which is extending the lifetime of the network. Energy provisioning does so by systematically replenishing the energy source of the sensor nodes to prevent power outage. Such schemes can be further classified into battery-driven, energy harvesting, and energy transference based schemes. Energy consumption based schemes assume a limited energy resource that cannot be replenished. Therefore, algorithms and protocols that are implemented on the sensor nodes are proposed instead. Energy consumption-based approaches were categorize into data-driven, duty-cycling, routing protocols, and mobility-based energy management schemes. Assuming that activities such as sensing, processing, and transmission are the most energy-consuming activities performed by a sensor node. Some proposed algorithms aim to minimize these activities in order to preserve energy. Others aim to set the node asleep as long as possible. Lastly, other algorithms aim to efficiently route data packets through the network to reduce energy dissipation caused by overhead.

All the categories and sub-categories for data management in wireless sensor networks were briefly explained in this chapter and examples of related research works were presented for each one of them. Figure 1.1 represents a graphical summary of the detailed taxonomy. In this thesis, we focused mainly on two categories, namely, data-driven and duty cycling. More precisely for data-driven schemes we were interested in data prediction and adaptive sampling, and for duty cycling, we were interested in wake-up/sleep, spatio-temporal correlation approaches (marked in red in Figure 1.1). Next, in the upcoming chapters, we present in detail our proposed solutions in the scope of the previously mentioned categories.

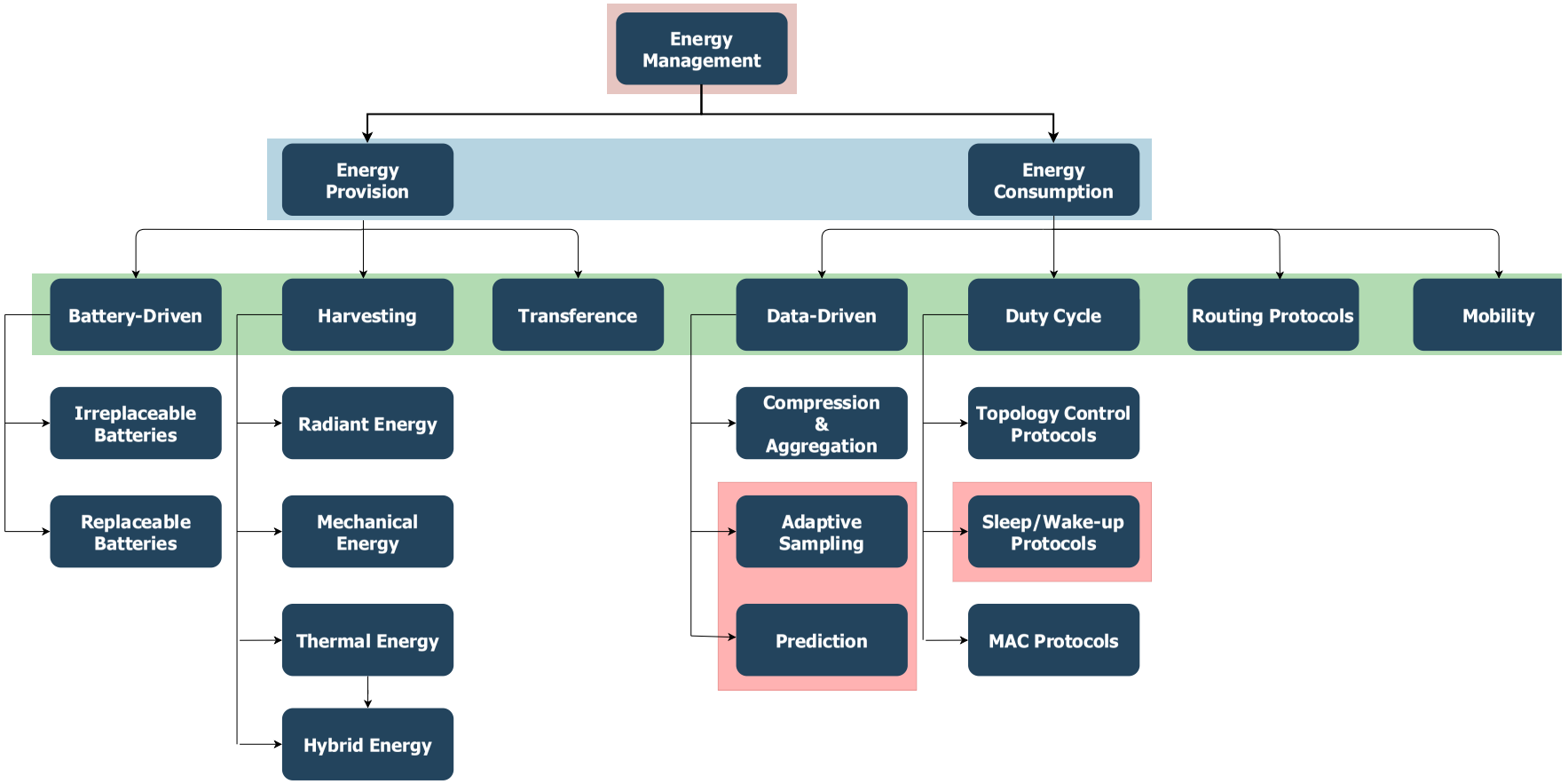


Figure 1.1: Energy management in WSN (in red, the targeted categories in this Thesis)

2

Fault-Tolerant Data Transmission Reduction For WSNs

The radio module of a sensor node is considered to be one of the most energy hungry components. Therefore, numerous approaches aiming to minimize the sensors-to-sink data transmissions have been proposed in the literature. In this chapter, we first present one of the most adopted data-driven transmission reduction techniques, we spot the light on one of its major vulnerabilities, to finally propose a novel transmission reduction algorithm that can overcome it while outperforming similar approaches.

2.1/ Introduction

Several studies have demonstrated that the Dual Prediction Mechanism (DPM) remains the most efficient data-driven technique for energy conservation in Wireless Sensor Networks (WSNs). In real world, the deployed sensor nodes suffer from packet loss and even failures which renders the DPM unreliable, since it requires flawless synchronization between the source (sensor node) and the destination (Sink). Most of the proposed DPM-based data reduction methods [Santini et al., 2006, Tan et al., 2016, Wu et al., 2016, Raza et al., 2015, Fathy et al., 2019, Aderohunmu et al., 2013, Guestrin et al., 2004, Li et al., 2013, Moghadam et al., 2011] consider loss-free communication links, which is not the case in a real-world distributed system. Data loss is a major issue that can reduce the performance of these techniques dramatically. Therefore, a mechanism that can identify missing data coupled with an algorithm that can reconstruct them is mandatory for a realistic application of the dual prediction mechanism.

In this chapter, we present a novel DPM-based technique that lends itself to be simple yet robust, and effective in terms of prediction accuracy and data reduction. In addition, we

coupled this technique with a data reconstruction algorithm that exploits both temporal smoothness and spatial correlation among different sensed features in order to estimate missing values. This proposed algorithm belongs to the data prediction sub-category presented in Chapter 1, Section 1.2.2.1.

The main contributions in this chapter include:

- Proposing a novel data reduction algorithm that integrates a mechanism capable of identifying missing sensor readings and maintaining a synchronized communication link between the source node and the Sink (which is mandatory for a successful implementation of the DPM technique).
- Adoption and adaptation of a data reconstruction algorithm in order to recover missing values that failed to reach the Sink.

The rest of the chapter is organized as follow. In Section 2.2 we briefly explain the Dual Prediction Mechanism (DPM). The adopted energy consumption model is presented in Section 2.3. The data-sets used for simulation are illustrated in Section 2.4. In Section 2.5 the data transmission reduction algorithm is explained. In Section 2.5.2 the method used to identify wrong predictions is presented. Section 2.5.3 illustrates and explains the data reconstruction algorithm. The experimental results are presented in Section 2.6, including comparison of performance in terms of data reduction, energy consumption, scalability, and prediction delay. Finally, the chapter is concluded in Section 2.7.

Let us first explain and describe the Dual Prediction Mechanism (DPM) which is the baseline of this contribution. In addition, we present and describe both the data-sets and the energy model considered for the simulations and experimentations.
NB: The presented DPM, energy model, and data-sets will also be considered in the next chapters of this part.

2.2/ THE DUAL PREDICTION-BASED MECHANISM

The Dual-Prediction Mechanism (DPM) is the most relevant technique adopted by most of the prediction-based data reduction approaches. This mechanism works by building a prediction model that analyzes the history of previously collected measurements in order to extract the moving trend of the data and estimate future readings. The same prediction model is shared between both the sensor node and the Sink. Using the shared model, they both proceed to periodically predict future observations. This approach allows the sensor node to avoid transmitting its collected measurements to the Sink, as long as their corresponding predictions are accurate. Meanwhile, the Sink always presumes that its prediction reflects the real observation unless it receives the corrections from the sensor

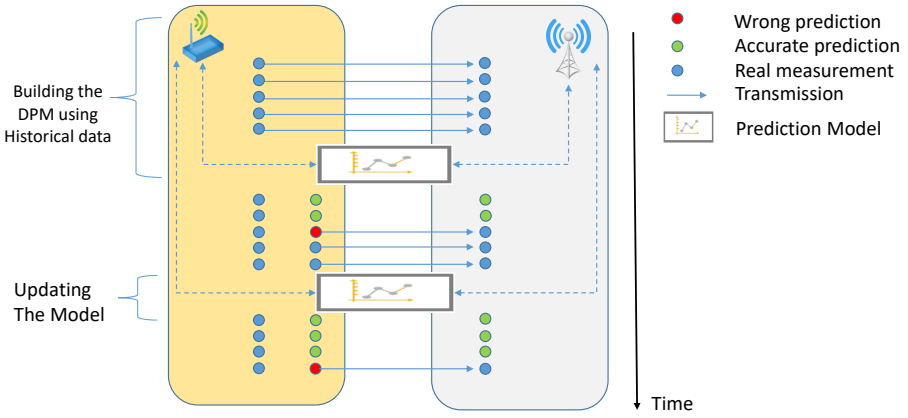


Figure 2.1: The concept of DPM

node. After receiving a number of corrections, the Sink assumes that the prediction model is no longer valid and it must be rebuilt. Therefore, they both proceed to build a new model using the newly transmitted data. The same cycle is repeated over and over again until the battery of the sensor node is depleted. Figure 2.1. illustrates in a simplified way how this mechanism works.

2.3/ The Energy Consumption Model

Most of the energy consumed by a sensor node is generally related to four main tasks, namely, memory logging, sampling, processing, and radio transmission. In order to estimate the energy consumed by a given algorithm we will be using the following model or at least a part of it depending on the experiment:

$$E_{node} = E_{sampling} + E_{processing} + E_{radio} + E_{logging} \tag{2.1}$$

$E_{sampling}$ is the energy required to transform a physical signal into a digital one. This value is calculated using Equation (2.2), where I_{sens} is the total current for the sensing activity, T_{sens} is the total duration of the latter, V is the supply voltage and b is the number of bits in the sensed packet when transformed to digital.

$$E_{sampling} = bVI_{sens}T_{sens} \tag{2.2}$$

$E_{proccessing}$ is the energy consumed by the CPU to perform a certain number of instructions. It is calculated using Equation (2.3), where N is the number of cycles required at each iteration of the algorithm, b is the number of bits to be processed, C is the average capacitance switched per cycle, I is the leakage current, n_p is a constant that depends on

the processor, V_t is the thermal voltage, and f is the sensor frequency.

$$E_{Processing} = bNCV^2 + bV(Ie^{\frac{V}{n_p V_t}})(\frac{N}{f}) \tag{2.3}$$

Finally E_{radio} is the energy required to transmit a b bits packet for a distance d. It could be calculated using Equation (2.4), where E_{elec} is the energy required to transmit electronics, E_{amp} is the energy consumed by the power amplifier, and n is the distance-based path loss exponent (n = 2 for free space fading, and n = 4 for multi-path fading)

$$E_{radio} = bE_{elec} + bd^n E_{amp} \tag{2.4}$$

$E_{logging}$ is the required energy to write/read a b bit packet into/from memory. Sensor logging energy consumption for a sensor node per round is evaluated using Equation (2.5), where I_{write} and T_{write} are the current and time duration for reading 1 byte of data respectively.

$$E_{logging} = \frac{b \times V}{8} \times (I_{write} \times T_{write}) \tag{2.5}$$

Symbol	Description	Value
N	Number of clock cycles per task	0.97×10^6
C	Avg. capacitance switch per cycle	$22pF$
V	Supply voltage to sensor	$2.7V$
V_t	Thermal voltage	$0.2V$
f	Sensor frequency	$191.42MHz$
I	Leakage current	$1.196mA$
b	Transmit packet size	$2kb$
n	Path loss exponent	2 or 4
n_p	Constant: depending on the processor	21.26
E_{elec}	Energy dissipation: electronics	$50nJ/bit$
E_{amp}	Energy dissipation: power amplifier	$100pJ/bit/m2$
T_{sens}	Time duration: sensor node sensing	$0.5ms$
I_{sens}	Current: sensing activity	$25mA$
T_{write}	Time duration: flash writing	$12.9ms$
I_{write}	Current: flash reading 1 byte data	$6.2mA$

Table 2.1: Base values of the energy model parameters

This energy model is inspired from the one discussed in [Halgamuge et al., 2009] and all

the base values used in this simulation for all the previously described parameters can be found in Table 2.1.

2.4/ THE DATA SETS

In this chapter and the upcoming ones, we consider two different data-sets for simulation and performance comparison. Both of them contain real data collected by real sensor nodes. We will briefly present and explain these data-sets in the next subsections.

2.4.1/ GRAND-ST-BERNARD PASS DATA-SET

This data-set consists of real sensor readings collected from a sensor network that was deployed at the Grand-St-Bernard pass between Switzerland and Italy [Sensorscope, 2007]. The network consisted of 23 sensors, each one of them collects 9 different environmental features with a fixed sampling rate of 1 sample every 2 minutes. We have chosen 4 out of these 9 features (ambient temperature $[C^{\circ}]$, Surface temperature $[C^{\circ}]$, relative humidity $[\%]$, and wind speed [m/s]) since the others are not complete. Environmental features are usually stationary, therefore, in addition to taking a sample every 2 minutes, and for a rigorous comparison of the algorithms, we set up two other scenarios. In the first one, a sample is taken every 10 minutes instead. In the second one, a sample is taken every 20 minutes. In this way, the data will become “non-stationary" which makes it more realistic and harder for data reduction algorithms to adapt to the stream.

The raw data set (sample every 2mins) consists of 10000 readings for each sensor, for the 1st scenario (1 sample per 10 minutes) we will end up with 2000 readings instead, and 1000 readings for the second one (1 sample per 20 minutes).

2.4.2/ DISC DATA-SET

Twenty data sets containing each, 300,000 readings (100,000 temperature, 100,000 humidity, and 100,000 infrared readings), have been collected using twenty Crossbow TelosB nodes with an integrated weatherboard that have been deployed in our laboratory. It is equivalent to approximately 35 days of non-stop data collection. Measurement was taken every 30 seconds and transmitted to a central Sink node called SG1000 connected to a laptop machine. The temperature value was measured by degrees Celsius, Humidity ranges between $0 - 100\%$, and Light is in Lux.

Figure 2.2 shows the geographical distribution of the deployed sensor nodes. The x and

y coordinates of sink and sensors (in meters relative to the upper left corner of the lab) are given in Table 2.2

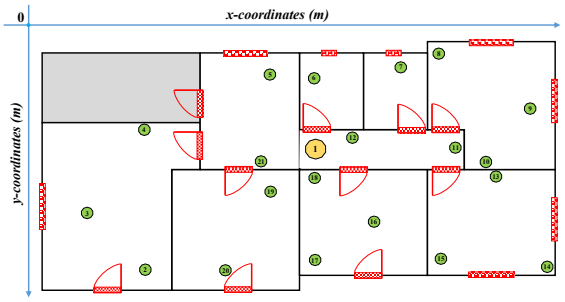


Figure 2.2: The geographical distribution of the sensor nodes

Sensor ID	x-coordinates (m)	y-coordinates (m)
1 (Sink)	9.6	3.82
2	3.63	7.97
3	1.57	6.06
4	3.61	3.1
5	8	1.2
6	9.54	1.3
7	12.63	0.94
8	13.97	0.47
9	17.16	2.36
10	15.58	4.26
11	14.57	3.72
12	10.93	3.37
13	15.95	4.73
14	17.79	7.91
15	14	7.64
16	11.65	6.31
17	9.57	7.71
18	9.57	4.77
19	8	5.3
20	6.42	8.11
21	7.7	4.24

Table 2.2: Sensor nodes coordinates

2.5/ THE PROPOSED APPROACH

Let us first describe our data transmission reduction method which exploits the temporal correlation among sensed measurements in order to limit the number of radio transmissions performed by each sensor node. As mentioned earlier, both the sensor and the sink need to build and maintain simultaneously a prediction model capable of forecasting future readings within a predefined error margin. This is achieved by following a few simple steps, described hereafter.

NB: we assume a periodic sensor network where every sensor collect and transmit a measurement every F seconds, where F is a value predefined by the user.

The moment when a sensor is activated, it transmits to the Sink the first two collected measurements, x_0 and x_1 at time t_0 and t_1. Afterward, they both calculate the value C_Temp that is supposed to represent the temporal correlation among the already collected and the upcoming measurements as shown in Eq. (2.6). Moreover, the Sink and the sensor need to store in their memories the last communicated value which is referred to as LC (in this case x_1), that will be used later to update the prediction model when it is needed.

$$C\_Temp_0 = x_1 - x_0 \tag{2.6}$$

Once C_Temp_0 is calculated, the sensor is no longer required to transmit to the sink any collected measurement at any time k as long as the predicted value $\hat{x}_k$ as shown in Eq. (2.7) is accurate.

$$\hat{x}_k = \hat{x}_{k-1} + C\_Temp_0 * \alpha \tag{2.7}$$

The parameter α is a rectification value that can range between 0 and 1. Its role is to harmonize the predictions with the real sensed readings. This is achieved by applying a dynamic penalty on C_Temp based on an error and a model accuracy feedback. By doing so, we aim to reduce the bias in the temporal correlation value C_Temp. A more detailed explanation of how α is automatically calculated is provided in the next subsection.

As mentioned earlier, as long as the prediction is accurate (the difference between the real sensed value and the predicted one does not exceed a user predefined error threshold $emax$), the sensor discards the real sensed value and does not transmit it to the Sink. In its turn, if the Sink does not receive any value at a given time where it is supposed to receive one, it acknowledges that the prediction outputted by the same shared model is accurate. However, if a prediction $\hat{x}_n$ at a given time n is not accurate, the sensor is required to transmit the real sensed value x_n to the Sink in order to simultaneously update the prediction model (C_Temp). This is done by subtracting LC from the current communicated reading x_n and dividing the results by prediction horizon N (Eq. (2.9)) as shown in Eq. (2.8). Finally, LC is set equal to x_n until the next update occurs.

$$C\_Temp_n = \frac{x_n - LC}{N}. \tag{2.8}$$

Where:

$$N = \frac{t_n - t_{LC}}{F}. \tag{2.9}$$

2.5.1/ UPDATING α

During the update phase, and in order to tune α for the next prediction phase, we need to know first how did the model performe in terms of prediction accuracy and durability. Therefore, both the sink and the node are required to calculate a value called the Accuracy Factor (AF). This is done by subtracting the value of the communicated measurement x_n from the erroneous prediction $\hat{x}_n$ that triggered the update, and dividing the result by the prediction horizon N as shown in Eq (2.10).

$$AF = \frac{\widehat{x_n} - x_n}{N}. \tag{2.10}$$

Then, a percentage value (P) of how much the value of AF is compared to $emax$ is calculated as shown in Eq (2.11). The smaller is the absolute value of AF the more accurate it was C_Temp_n, therefore, the latter must be updated relatively to the value of P.

$$P = \frac{AF \times 100}{emax} \tag{2.11}$$

If AF is negative, this means that C_Temp_n must be increased in order to fit better the upcoming data, therefore, α is increased by $P\%$. In contrast, if AF is positive this means that C_Temp_n must be decreased by decreasing α by $P\%$. In this way, we are automatically tuning alpha according to a model accuracy feedback extracted from the last produced error and the prediction horizon (Eq. (2.12)). This will lead to having a more accurate model, thus a longer prediction horizon (durability) and fewer transmissions.

$$\alpha^{new} = \alpha - \frac{P \times \alpha}{100} \tag{2.12}$$

A few rules are required to take into consideration in order to prevent any possible deadlock when updating α : AF is very small compared to emax (e.g. 10% of emax), this means that the error is very small, and α is almost optimal. Thus, it should remain unchanged. AF exceeds $emax$, in order to prevent α from having a negative value, it should be reset to 0.5. AF remains positive for multiple successive adjustments, α could start to deviate to 0. If this is the case, α should be reset to 0.5.

The Algorithm 1 below illustrates the proposed method that is implemented on the sensor.

2.5.2/ IDENTIFYING WRONG PREDICTIONS

Let us assume a scenario where a sensor fails to report a reading to the sink during the adaptation phase (where the prediction model update procedure is launched). The sen-

Algorithm 1 Transmission reduction

Require: $emax$ (maximum error threshold)

Read x_0 and x_1
Transmit x_0 and x_1 to Sink
$C\_Temp_0 \leftarrow x_1 - x_0$
$\alpha \leftarrow 0.5$
$LC \leftarrow x_1$
while $Energy \neq 0$ **do**
  Read x_k at time k
  $\hat{x}_k \leftarrow \hat{x}_{k-1} + C\_Temp_k \times \alpha$
  if $|x_k - \hat{x}_k| \geq emax$ **then**
    Send x_k to Sink
    $C\_Temp_k \leftarrow \frac{x_k - LC}{N}$
    $LC \leftarrow x_k$
    $AF \leftarrow \frac{\hat{x}_k - x_k}{N}$
    if $AF \geq emax$ **then**
      $\alpha \leftarrow 0.5$
    else if $AF \leq \frac{10 \times emax}{100}$ **then**
      Do not update α
    else
      $P \leftarrow \frac{AF \times 100}{emax}$
      $\alpha^{new} \leftarrow \alpha - \frac{P \times \alpha}{100}$
      if $alpha \approx 0$ **then**
        $\alpha \leftarrow 0.5$
      end if
    end if
    $\hat{x}_k \leftarrow x_k$
  end if
end while

sor will not know that the sink did not receive it, and it will use this reading to update its model. However, the sink considers that its prediction is within the error budget, therefore no update is needed. Hence, the prediction models on both sides will lose synchronization and start outputting different values. Therefore, we propose a solution that is based on an acknowledgment mechanism between the sensor and the sink. Consider that a sensor transmits a reading to the Sink, instead of switching immediately to the adaptation phase, the sensor must wait for an acknowledgment indicating that the reported value has been well received. As long as the sensor has not yet received an acknowledgment, it must keep reporting readings to the sink. This method ensures that both the sink and the node update their models simultaneously. Moreover, a sequence number is sent with each reading. If the sink detects a jump in the sequence number, it flags the corresponding measurements as missing, which allows the reconstruction algorithm to identify and reconstruct them. Yet, this is not sufficient to cover all potential failures, the batteries may deplete or the sensor could crash due to a software failure. Therefore, since the sensor operates in rounds, where each round is divided into several periods. At the beginning of

each period, a sensor must send the current reading even if it is within the error budget. Hence, if the sink does not receive a reading at the beginning of a given period, it will consider that the sensor has crashed, and all the future estimations will be replaced by a "NaN" value (flagged as missing).

The Algorithm 2 is implemented on the Sink, it illustrates how the latter can detect missing values. An instance of this algorithm is initialized for each sensor node.

Algorithm 2 Detecting missing data

Receive x_0 and x_1
$C\_Temp_0 \leftarrow x_1 - x_0$
$LC \leftarrow x_1$
while $Energy \neq 0$ **do**
  if x_k is received with sequence number SN **then**
    Send an acknowledgment to the sensor
    if SN $>$ 1 **then**
      for i=1:SN **do**
        $x_{k-i} \leftarrow$ "NaN"
      end for
    end if
    $C\_Temp_k \leftarrow \frac{x_k - LC}{N}$
    Update α
    $LC \leftarrow x_k$
    $\hat{x}_k \leftarrow x_k$
  else
    $\hat{x}_k \leftarrow \hat{x}_{k-1} + C\_Temp_k \times \alpha$
  end if
end while

2.5.3/ Reconstruction of Missing Data

At the end of the sensing period, all missing data (NaN values) must be reconstructed. Several methods have been proposed in the literature that exploit spatial correlation along with temporal smoothness to recover missing data in WSN [Wu et al., 2018, Pan et al., 2013, Gao et al., 2016, Gruenwald et al., 2010]. However, due to the sizes of the collected data-sets that are massively large and containing multiple dimensions, these methods are extremely expensive in terms of computation complexity. Therefore, we adopt and adapt the method proposed in [Li et al., 2009] that captures the correlations between multiple co-evolving time sequences, by identifying automatically a few hidden variables from the large data-set and mining their dynamics to impute missing values with low reconstruction errors.

Let us consider a time sequence X with a duration of T in m dimensions, where m is equal to the number of sensors in the monitoring area. This sequence X contains all the

data reproduced by the sink, and it also includes "NaN" values indicating that a reading is missing. The goal of this algorithm is to reconstruct these missing readings, by observing the values of the missing sensor and other ones at neighboring time ticks.

$$X = \begin{bmatrix} x_1^1 & x_1^2 & ... & NaN & ... & x_1^T \\ x_2^1 & x_2^2 & ... & x_2^T & ... & x_2^T \\ ... & NaN & ... & & & \\ ... & & & & & \\ x_m^1 & x_m^2 & ... & NaN & ... & x_m^T \end{bmatrix}$$

Let us denote the observed part as X_o, and the missing part as X_m. A probabilistic model (Figure 2.3) is built to estimate the expectation of missing values conditioned by the observed part $\mathbb{E}[X_m|X_o]$. A set of latent variables denoted Z_n are calculated using a belief propagation system. These latent variables model the dynamic and hidden patterns of the observed sequence. Moreover, the latent variables Z_n are assumed to be time-dependent with the value at time tick t is determined by the value at time tick $t-1$ using a linear mapping F. In addition, linear projection matrix G from the latent variables Z_n to the data sequence for each time tick, is assumed to represent the spatial correlation among different dimensions. Once the latent variables are calculated, they are used as input for an EM iterative algorithm [Shumway et al., 1982] in order to find the best-fit parameters (such as G and F) for the data reconstruction probabilistic model.

Figure 2.3 illustrates this probabilistic model used to estimate missing values at a given time tick. For instance, the figure shows a missing sensor value at the time tick 3. This value can be estimated by multiplying the linear projection matrix G with the estimated latent variables Z_3. A detailed explanation of how the parameters of the model and the latent variables are calculated can be found in [Li et al., 2009].

The size of the latent variables set can vary between 1 and m. There is an optimal value for the number of latent variables that can render the reconstruction algorithm more accurate. In [Li et al., 2009] this number was fixed during the experiments to 15. However, in order to adapt the reconstruction algorithm to our needs we have calculated the optimal value for the latent variables using the following method:

- We first divide the collected data set into two subsets, a training subset and a validation one. The values belonging to either one of the sets are chosen randomly. The values are equally divided between the two sets and one value can only belong to one of the two sets.
- In the training subset the values that belong to validation subset are set to “NaN”. For instance if x_i^j where $i \in [1, m]$ and $j \in [1, T]$ is selected to be in the validation set, the value of x_i^j in the training set will be set to “NaN”

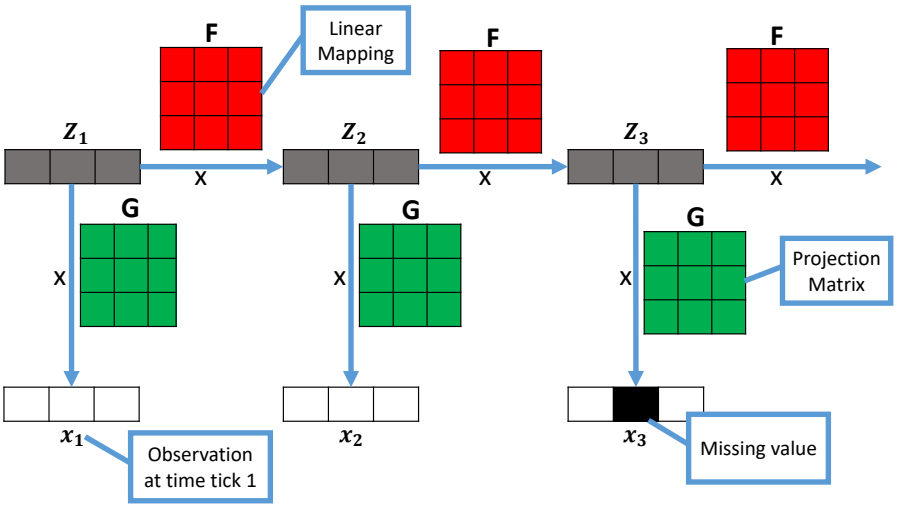


Figure 2.3: Probabilistic model

- For each value of n ranging from 1 to m we run the reconstruction algorithm and we give it as input to the training set and we set the number of latent variables equal to n. Then we calculate the Root Mean Square Error (RMSE) between the predicted values and their corresponding matches in the validation set.
- For each value of n, we select a random training and a validation set and we run the reconstruction algorithm 10 different times. Finally, we average the RMSE of the 10 runs and we chose the value of n that returned the lowest RMSE.

2.6/ EXPERIMENTAL RESULTS

In this section, we present the experimentation we have conducted so as to demonstrate the efficiency and the robustness of our proposed approach compared to three recent and well established linear data prediction algorithms (OSSLMS, HLMS, and DBP). Our goal is to maximize the data suppression ratio while maintaining a low computation cost, ensure a loss-free communication link between the source node and the Sink via synchronization, and preserve data quality via reconstruction of the missing observations.

For this experimentation, we consider the DISC data-set described in Section 2.4.2 of Chapter 2. The temperature readings are smooth, where neighboring measurements vary only slightly over time. However, humidity readings are not as smooth, a noticeable variation can be observed, but they are still easily predictable since they follow a clear trend. Last but not least, infrared data does not follow any specific trend and the data are

basically either irregular and highly varying or stable and barely changing. These three types of data can show weakness if existed in any of the 4 algorithms being compared in handling a specific type of upcoming readings.

2.6.1/ Performance Comparison With Other Approaches in Terms of Transmission Reduction

In this section, we evaluate the performance of our transmission reduction algorithm compared to the OSSLMS, HLMS, and DBP approaches. The simulation has been conducted using a custom WSN simulator built-in MATLAB and the following experimentation settings were used. The error threshold $emax$ was set to ± 0.1 for temperature and humidity, and ± 1 for Infrared. The number of sub-filters (m2) and the size of each one of them (m1) for the HLMS algorithm, were set to 2 and 3 respectively. The size of the OSSLMS filter was set to 5. For DBP, the number of edge points l, the learning window m, the relative error for all environmental features, and the time tolerance ϵ_T, were set to 3, 6, 5%, and 2 respectively. Finally, we would like to note that the simulation was repeated 10 different times and the results presented in this section are the average results of the 10 simulations. The final averaged results of the 20 sensor nodes are listed in Table 2.3.

All the proposed algorithms performed well in terms of data suppression. However, our data reduction method outperformed the other approaches in two out of three environmental features, and OSSLMS has outperformed our method in one feature. As shown in Table 2.3, all of the four algorithms have approximately the same Suppression Ratio (SR) for temperature data, except for DBP that has an SR that is around 2% lower than the others. The reason is that temperature data are very smooth and the variations in neighboring measured values are small. Thus, a linear prediction algorithm is very efficient in keeping up with these small changes. For humidity data, OSSLMS has adapted itself better and achieved the best suppression ratio of 94.1% among the three other approaches, including ours that achieved an SR of 93.6%. Finally, when we tested the algorithms on highly varying infrared data, our approach was significantly better in reducing the number of radio communication. For instance, our SR was 5.6%,13.1%, and 29.7% greater than OSSLMS, HLMS, and DBP respectively.

Table 2.3: Suppression ratio of transmitted data

	Supression ratio (%)			
	TR	OSSLMS	HLMS	DBP
Temperature	**99.8**	99.7	99.6	97.7
Humidity	93.6	**94.1**	90.3	82.7
Infrared	**93.2**	87.6	80.1	63.5

2.6.2/ Complexity Comparison With Other Approaches

The complexity of the data reduction algorithm is as important as its efficiency in reducing transmission. An optimal algorithm is the one who has the highest suppression ratio and the lowest complexity. In a periodic sensor network, a sensor wakes up performs a task (sensing, prediction or transmission) and then goes back to sleep for a predefined period of time before waking up again. As explained in Section 1.2.2.2 of Chapter 1, the sensor consumes the minimum amount of energy when he is in sleep mode. Therefore, the faster the required task is accomplished, the faster a sensor would go to sleep, the less energy it will consume. The more complex is an algorithm, the more time it will take a sensor to perform the given task, the more energy it will consume. For that matter, in this section, we will compare the complexity of the four algorithms by breaking down each one of them into a series of mathematical operations and counting their number at each iteration.

OSSLMS has achieved the best suppression ratio in one out of two features and was ranked second for the other two. However, despite its efficiency in suppressing transmissions, this algorithm has a very high complexity of order O(n^3). At each iteration, this algorithm requires $4N+4N^2$ multiplications, $2N+2N^2$ additions, N subtractions, 1 division, $2N+\frac{N^2-N}{2}$ swap operations, and $\frac{2N^3+N^2+2N^2+N}{6}-\frac{N^2-N}{2}$ operation to compute the pseudo-inverse of a matrix (N is the size of the adaptive LMS filter). HLMS ranks second in terms of complexity and it has achieved the third-best suppression ratio. For HLMS the following operations are required at each iteration: $4m_2m_1$ multiplications, $2m_2m_1+3m_2+m_1$ additions, m_2+1 subtractions, and m_2+m_1 divisions, where m2 represents the number of the LMS sub-filters, and m1 represents the length of each LMS filter. The complexity of this algorithm is then of order $O(n^2)$. DBP However is linear of order O(n) when an update is needed since it requires only 2(l-1) additions, 2 divisions, and 1 subtraction to readjust the model, and when no readjustment is needed, only 1 addition is required to calculate the prediction. Therefore during prediction DBP has a constant complexity of order O(1).

It seems that the higher the complexity of the algorithm, the better its efficiency. However, our method is the least complex one and achieves the best results. When no adjustment is needed for the model, 1 addition is required to calculate the prediction ($\alpha * C\_Temp$ is computed once during readjustment only). When an adjustment is needed, 2 subtractions, 4 divisions, and 3 multiplications are required to update the model. Thus our algorithm has a constant complexity of O(1). This means, our proposed algorithm provides the best trade-off between complexity and transmission reduction. Thus, in theory, it is the least energy consuming algorithm compared to the other approaches. This assumption will be further validated in the next section through simulations and in the next chapter of this dissertation through real implementation.

2.6.3/ Energy Consumption

In order to compute an estimate of the energy consumed by a specific sensor activity we have used the energy model described Section 2.3. Figure 2.4 shows the energy consumed by the transmission, and processing activities, in addition to the overall energy consumed by both activities combined. The sensing activity has been excluded since it is the same for all algorithms and it would not affect the end results. Moreover, we also did not include the logging activity. In this experiment we were only interested in transmission and computational costs. Therefore, the overall energy consumption is computed as following:

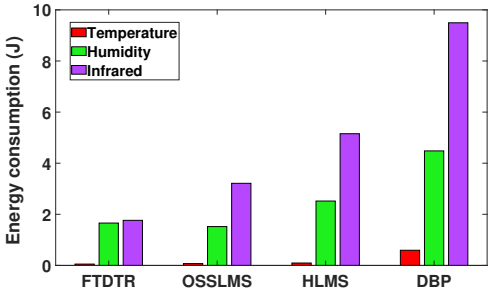
$$E_{total} = E_{radio} + E_{processing}. \tag{2.13}$$

Figure 2.4(a) shows that the energy consumed by the transmission activity is directly related to the number of data transmitted by the node. The obtained results are proportional to the suppression ratio results in Section 2.6.1. At every iteration, each algorithm performs a number of CPU cycles in order to execute the required instructions. The more complex is the algorithm the more CPU cycles are required to execute it, the longer it takes the sensor to go to sleep, which implies more energy consumption. Figure 2.4(b) shows the energy consumed by the processing activity. The results are aligned with the complexity of the algorithms discussed in the previous section. For instance, OSSLMS with a complexity of O(n^3) has consumed the most energy, followed by HLMS, DBP, and FTDTR.

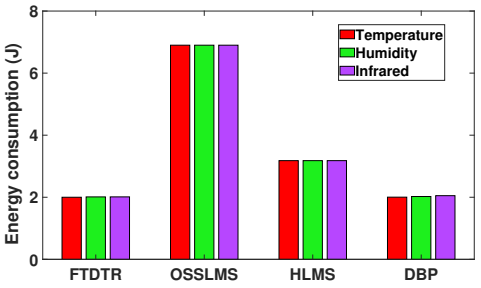
If we only take into consideration the energy consumed by data transmission, it seems that OSSLMS can outperform FTDTR with humidity data and holds second place with temperature and infrared data. However, when the computational energy consumption is considered, OSSLMS falls to the fourth place behind HLMS and even DBP with temperature and humidity data. Since our algorithm has a constant complexity and it can effectively reduce data transmission, as shown by the results (Figure 2.4(c)), FTDTR has consumed the least energy compared to OSSLMS, HLMS, and DBP.

2.6.4/ Scalability and Prediction Delay

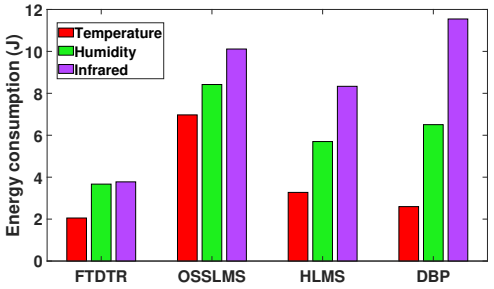
The scalability of the Dual-Prediction scheme, whether it is FTDTR, OSSLMS, HLMS, DBP or others greatly depends on two main factors. The computational power of the Sink and its memory capacity. Therefore, the more complex an algorithm is, and the more memory space it requires, the fewer nodes the Sink can handle simultaneously. The advantage that our proposal provides is that it requires a small memory footprint and it is not complex, yet, it is robust and efficient. For instance, Figure 2.5 shows the time each



(a) Transmission energy



(b) Processing energy



(c) Total energy

Figure 2.4: Energy consumption

algorithm needs to produce a prediction (prediction delay). The shown results are the aggregate sum of 10000 predictions. As we can see, OSSLMS takes significantly more time to produce a prediction than its counterparts, this is due to its high complexity. Since the CPU of the Sink will be blocked for a longer duration in order to produce a prediction, this would affect negatively the number of nodes a sink running OSSLMS can handle simultaneously. In the case of FTDTR the prediction delay is almost negligible. Thus the CPU and the memory will be liberated rapidly which gives more space to the Sink to handle a larger number of nodes.

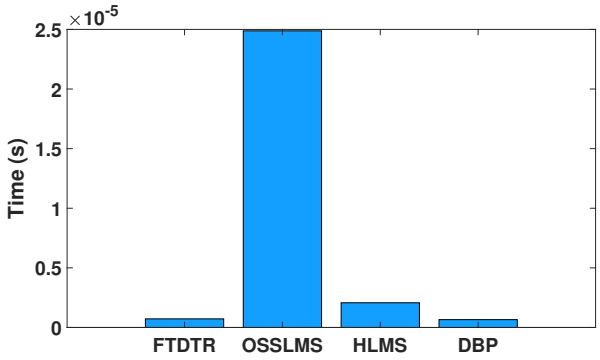


Figure 2.5: Prediction delay comparison

Eventually, sensor nodes can be grouped into clusters and each group can be assigned to a different Sink. The number of sensors belonging to the same group is influenced

by the complexity of the running algorithm and the computational capacity of the Sink. A large number of possible settings could be taken into consideration. Therefore in future work, a comprehensive study on the scalability of our proposal could be conducted.

2.6.5/ DATA LOSS/COMMUNICATION ERROR

In this section, we will compare the different approaches in a scenario where data loss occurs randomly when a sensor is trying to send a measurement to the Sink. The percentage of a sensor failing to transmit its readings can vary from 10% in an ideal environment and when there are low collision and overload in the network, up to 50% in harsh environments and a jammed network [Zhao et al., 2003, Zong et al., 2017]. We have evaluated the performance of each of the four methods with a transmission failure possibility ranging from 10% to 50% based on the Bernoulli distribution, where the probability of failed transmission p is varied within the range of $[0.1 - 0.5]$, and the probability of a successful transmission q $(1 - p)$ within $[0.5 - 0.9]$.

Figure 2.6 shows the number of temperature, humidity, and infrared data exceeding the error threshold $emax$ when different missing possibilities are considered. With a data loss detection mechanism, FTDTR was able to limit the number of wrong estimations by keeping the model at the sensor and the sink synchronized. Thus, only the readings that failed to reach the sink and were flagged as "NaN" values by the later are considered to exceed the error threshold. Oppositely, the number of estimations exceeding the error threshold for other approaches is far greater than the number of measurements that failed to be reported. The reason is that the readings that fail to reach the sink are used by the sensor to adjust the model, thus leaving the sink with an outdated one that produces wrong estimations, while the sensor is producing correct ones.

For each environmental feature (temperature, humidity, and infrared) the twenty data sets reproduced by the sink corresponding to the twenty deployed sensor nodes are passed to the reconstruction algorithm (DynnaMMO) in order to fill the blank values flagged as "NaN". Figure 2.7 shows the average percentage of the successfully reconstructed data. The reconstruction success rate can range between 45% and 72% according to the number of missing readings and on the temporal smoothness and spatial correlation of the data set.

The reconstruction of a missing reading at time t is considered to be unsuccessful if the difference between the values of the reconstructed measurement and the real one is greater than the maximum error tolerance ($|\hat{x}_t - x_t| > emax$). For a missing probability varying from 10% to 50%, Table 2.4 shows the Root Mean Square Error (RMSE) of the measurements that have been unsuccessfully reconstructed by FTDTR and the data exceeding $emax$ for OSSLMS, HLMS, and DBP.

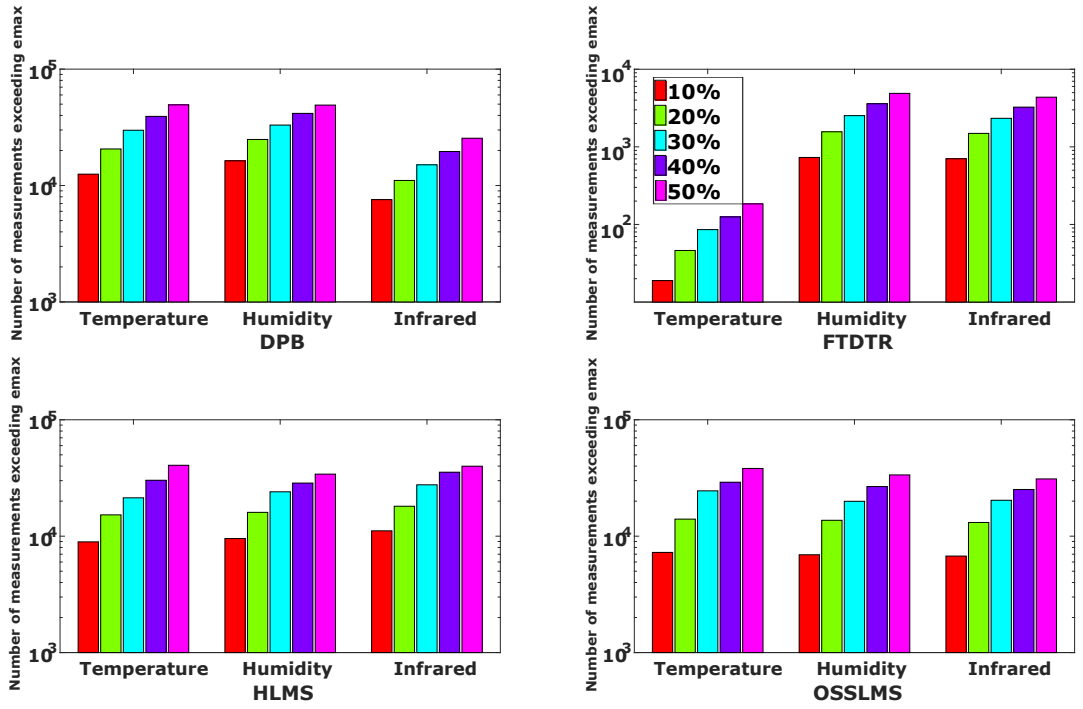


Figure 2.6: Number of measurements surpassing the error threshold

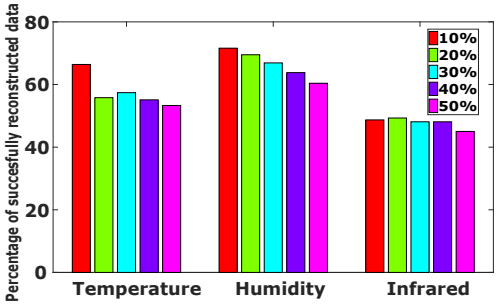


Figure 2.7: Percentage of successfully reconstructed data

Table 2.4: RMSE of data exceeding emax

	Temperature				Humidity				Infrared			
	FTDTR	OSSLMS	HLMS	DBP	FTDTR	OSSLMS	HLMS	DBP	FTDTR	OSSLMS	HLMS	DBP
10%	**0.103**	0.16	0.22	0.639	**0.13**	0.41	1.88	0.44	**2.3**	3.21	2.84	10.65
20%	**0.104**	0.188	0.238	0.849	**0.15**	0.29	2.15	0.47	**2.7**	3.39	4.39	15.52
30%	**0.104**	0.218	0.356	1.233	**0.14**	0.28	2.424	0.763	**2.8**	3.45	4.85	15.28
40%	**0.111**	0.242	0.782	1.025	**0.14**	0.312	2.937	0.834	**2.8**	4.06	5.42	18.46
50%	**0.110**	0.252	3.04	1.10	**0.16**	0.472	3.374	0.838	**3.3**	4.41	6.82	22.31

The results show that our method has the lowest RMSE for all environmental features and for all missing probabilities. Moreover, for temperature and humidity data, the RMSE's are very close to $emax$ (0.1). Therefore, when we increased $emax$ to 0.2, the reconstruction success rate for a 50% miss probability reached 99.6% and 94.3% for temperature and

humidity data respectively. For infrared data when we increased $emax$ to 4 instead of 1 (which is still an acceptable error for most applications) the reconstruction success rate increased to 91%.

The obtained results are in line with what has been previously emphasized, showing that the proposal outperforms existing works in maintaining high-quality estimations when a data loss scenario occurs.

2.7/ CONCLUSION

In this chapter, we have demonstrated that our proposed method is better at reducing the number of transmissions from the node to the sink compared with other approaches existing in the literature. Moreover, we take into consideration communication error, links failures, and battery depletion in order to prevent the loss of synchronization between the sink and the node, by applying an appropriate mechanism that identifies and reconstruct missing data. Simulations have been conducted using real data sets in a custom Matlab simulator. The performance of our proposal in terms of data reduction, complexity, energy consumption, scalability, and data quality, compared to other recent similar approaches has been proven through experimentation to outperform them.

3

A Distributed Prediction and Adaptive Sensing Approach

The sensor board of a sensor node is considered one of the most energy consuming components. Therefore, an important portion of the related literature proposed adaptive sampling algorithms that preserve the scarce sensor energy by minimizing the number of sampled data. In this chapter, we present an approach that combines the previously described transmission reduction technique with an adaptive sampling one into one efficient, data-driven, energy management algorithm.

3.1/ Introduction

In the previous chapter, we presented a novel fault-tolerant data transmission reduction algorithm. The proposed approach can extend the lifetime of the network by minimizing the number of transmitted packets. In this chapter, we will present an extension of this approach. To further reduce energy consumption we propose merging the fault-tolerant DPM with an adaptive sampling technique. This will enable the sensor node to adapt its sampling rate depending on the variations in collected data over a certain period of time. If no significant change is noticed, the sensor node could eventually reduce its sampling speed (the time between two consecutive samples) and sleep for a longer duration. This proposed algorithm belongs to the data prediction and adaptive sampling sub-categories presented in Chapter 1, Section 1.2.2.1. Adaptive sampling could, in theory, drastically reduce the energy consumption of the sensor. First, it reduces the energy consumed by the sensor board by minimizing the number of collected measurements. Second, it reduces the active duration of the sensor node by enabling it to sleep for a longer duration, which preserves an important amount of energy. The rest of this chapter is organized as follow. In Section 3.2 the Kruskal-Wallis based algorithm that allows the sensor to adapt its sampling rate is explained. Section 3.3 explains how the adaptive sampling and

transmission reduction techniques can be merged together. The obtained experimental results are shown in Section 3.4. Finally, the chapter is concluded in Section 3.5.

3.2/ THE KRUSKAL-WALLIS STATISTIC MODEL

Let us first begin by explaining the Kruskal-Wallis statistic model, which forms the core of the adaptive sampling algorithm. The Kruskal-Wallis test [McKight et al., 2010] takes as input a group of data sets to identify whether there is a difference between at least two of these sets. To understand how this test works and how it could help us to reduce the sampling rate of a sensor, we give the following illustrative example that explains its functionality and applicability in WSNs.

3.2.1/ ILLUSTRATIVE EXAMPLE

Let us consider that a sensor operates in rounds, where each round consists of p periods. To simplify the example, let us assume that p is equal to two. Table 3.1 shows a set of measurements collected by a sensor during two consecutive periods.

Table 3.1: Example of collected measures

Raw Measures		Measures Rankings	
Period 1	Period 2	Period 1	Period 2
3.4	4.6	1	2
6.2	5.8	4	3
7.0	7.0	~~5~~ 5.5	~~6~~ 5.5
7.3	7.5	7	8
7.6	8.0	9	10
10.3	10.2	~~12~~ 12.5	11
	10.3		~~13~~ 12.5
Number of Measures		Sum of Rankings	
6	7	39	52

The first step is to order the measurements in both periods by increasing order of their values and assign a rank denoted r to each one of them, representing its position in the ordered list. However, two or more measurements could have the same value. In this case, the mean value of their ranks is calculated and assigned to each one of them. For instance, in Table 3.1 the value 7.0 is repeated twice, both in period 1 and 2 with ranks 5 and 6 respectively. The mean value of both ranks is 5.5. Thus, the ranks of both measurements holding the value 7.0 are replaced by 5.5. The second step is to pass the ranked measurements as input to the Kruskal-Wallis test in order to find which one of the following assumptions is correct:

Assumption 1: the two groups of data (measurements in period 1 and 2) are significantly

different.

Assumption 2: the difference between the two groups of data is not significant.

The test is conducted using Equation (3.1), where N is the total number of measurements in all periods, n_i is the number of measurements inside the i^{th} period, and r_i is the sum of all ranks in the i^{th} period.

$$H = \frac{12}{N \times (N+1)} \sum_{i=1}^{p} \frac{r_i^2}{n_i} - 3 \times (N+1) \tag{3.1}$$

Using the data in Table 3.1 and based on Equation (3.1), H is calculated as follows:

$$H = \frac{12}{13 \times (13+1)} (\frac{39^2}{6} + \frac{52^2}{7}) - 3 \times (13+1) = 0.183$$

Finally, to check which assumption is the correct one, the result of this formula is compared with a "difference value" denoted H_t. H_t varies according to the false rejection probability predefined by the user, denoted α. The relation between α and H_t can be found in the $chi - square$ Table. The risk α is defined in a statistical test as the risk of rejecting the Null hypothesis when in fact it is true, it is also known as Type I error. This risk is stated in terms of probability (such as 0.05 or 5%). It corresponds to the confidence level of a statistical test, so a level of significance $\alpha = 0.05$ corresponds to a 95% confidence level. In our approach, it is the probability that a sensor node finds a high variance between its collected data while in reality there are no variances and it should adapt the sampling rate. Therefore, when α decreases the value of H_t increases and then the condition $H < H_t$ becomes more difficult to be satisfied. Consequently, the sampling rate increases when α decreases. Let us assume α = 0.05, for this value of α, H_t is equal to 5.991. Comparing the results of the previous equation we notice that $H < H_t$ $(0.183 < 5.991)$. Therefore, the first assumption is accepted. Hence, the sampling rate must be adapted.

3.2.2/ THE BEHAVIOR CURVE FUNCTION

Based on the Kruskal-Wallis test, when a node notices high variance differences, it increases its sampling rate in order to prevent missing important measurements and decreases its sampling rate when the variance is less than the threshold H_t. Following the example above low variance was detected, since $H < H_t$.

To compute the sampling rate of the sensor a behavior function (BV) is used, taking as input the risk of the application denoted R, or in other words, how important the quality of data is to the end-user. This BV function is expressed by a Bezier curve that passes through three points as shown in Figure 3.1: (0,0), (H_t, Maximum Sampling Rate), and R.

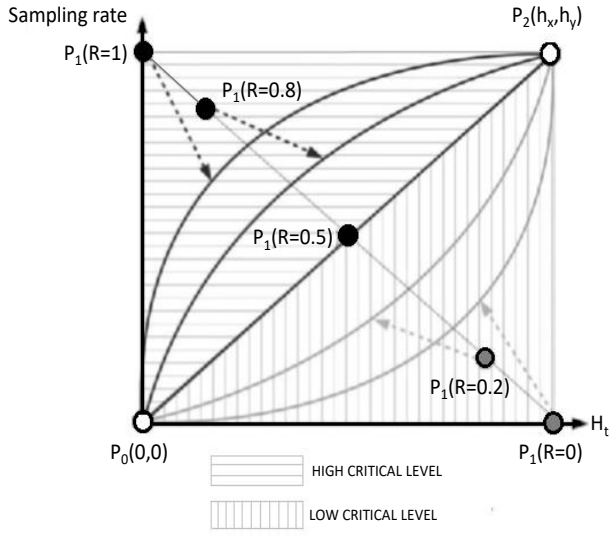


Figure 3.1: Sampling rate adaptation using the Behavior function

3.3/ Merging Adaptive Sampling and DPM based Transmission Reduction

One can notice that Adaptive Sampling (AS) and the Transmission Reduction (TR) algorithms are compatible. First, their work does not overlap and they do not affect each other's results since one is supposed to reduce sampling and the other is supposed to reduce transmission. Secondly, the prediction model is capable of filling the gap of "non collected data", since as mentioned before these measurements are mostly redundant or roughly similar to closely collected ones, and the prediction model efficiency is at a maximum when the change in values is smooth and slow. Therefore, on the one hand, the sampling rate is reduced and on the other hand, the end-user will still have access to the complete set of data. Finally, the complexity of the transmission reduction algorithm is constant. Thus, when combined with the adaptive sampling algorithm the overall complexity will remain unchanged. Therefore, we propose to combine these two techniques into a single algorithm (AS+TR), enabling us to achieve lower energy consumption compared to each one of them when implemented solely. Instead of waking up periodically after a fixed period of time, the sensor, at the end of each round, will automatically adapt its sleeping duration according to AS results. This is simply how the two algorithms could be merged. Figure 3.2 gives an illustrative example of the procedure, and Algorithm 3 shows how this method is implemented on the sensor.

Algorithm 3 Adaptive Sampling + Transmission Reduction

Require: $emax$ (maximum error threshold)

Read x_0 and x_1
Transmit x_0 and x_1 to Sink
$C\_Temp_0 \leftarrow x_1 - x_0$
$\alpha \leftarrow 0.5$
$LC \leftarrow x_1$
$j \leftarrow 0$
while $Energy \neq 0$ **do**
  Read x_k at time k
  $\hat{x}_k \leftarrow \hat{x}_{k-1} + C\_Temp_k \times \alpha$
  if period 1 **then**
    $x\_period1[j] = x_k$
  else if period 2 **then**
    Reset j to 0 once
    $x\_period2[j] = x_k$
  end if
  if End of round **then**
    Apply the KRUSKAL-WALLIS test on $x\_period1$ and $x\_period2$
    Update the sampling rate for the next round
    $j \leftarrow 0$
  end if
  if $|x_k - \hat{x}_k| \geq emax$ **then**
    Send x_k to Sink
    $C\_Temp_k \leftarrow \frac{x_k - LC}{N}$
    $LC \leftarrow x_k$
    Update α
    $\hat{x}_k \leftarrow x_k$
  end if
end while

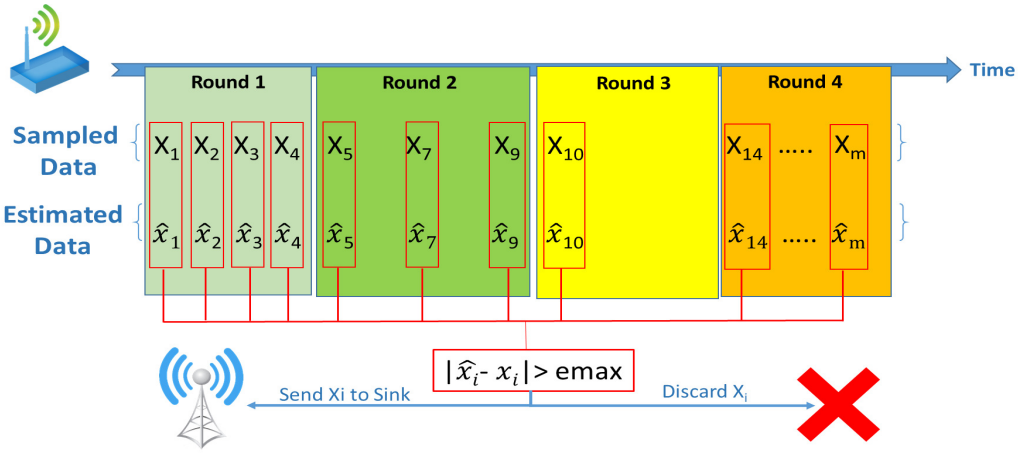


Figure 3.2: Illustrative example of the (AS+TR) method

3.4/ EXPERIMENTAL RESULTS

In this section, we present the experimentation we have conducted on the DISC wireless sensor data-set described in Section 2.4.2 of Chapter 2. We compared our approach with a recent similar one called Data Prediction with Cubic Adaptive Sampling (DP-CAS [Monteiro et al., 2017]), which merges an exponential time series predictive model with a TCP CUBIC congestion adaptive sampling technique. The setup parameters for this experimentation are shown in the Table 3.2.

Table 3.2: Setup parameters

DPCAS		**AS+TR**	
Smin	310 sec	Smin	310 sec
Smax	31 sec	Smax	31 sec
smoothing coefficient α & multiplicative reduction factor β	0.2	Risk R	0.6
cubic parameter C	0.4	Rejection Probability α	0.05

3.4.1/ SAMPLING RATE ADAPTATION

Let us first illustrate how the adaptive sampling algorithm works. Figure 3.3 shows the variation in the sampling rate for temperature data in two different nodes, namely Node1 and Node2 during 80 periods. In contrast with the naive approach where the sampling rate is fixed for all periods, the adaptive sampling method allowed the sensor to adapt its rate by either scaling it up or down according to the variation in measured data. However, we set a lower limit for the sampling rate S_{min} equivalent to 10% of S_{max} in order to reduce the risk of missing important information. This assumption will be demonstrated in the Subsection 3.4.3.

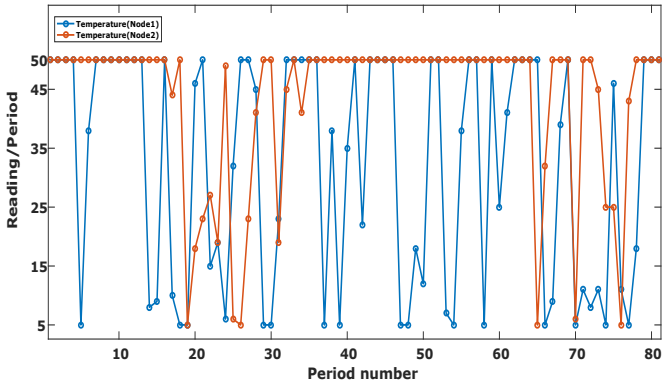


Figure 3.3: Number of samples during each period for temperature data

3.4.2/ Energy Consumption Compared to DPCAS

In this experiment, we focused only on the cost of radio transmission and data sensing, without taking into consideration the computational cost and the energy preserved while in sleep mode. Instead of simulations, a dedicate chapter for real implementation and energy measurement of all activities will be presented next chapter of this dissertation. In order to calculate the energy consumed by a sensor node, we used the energy consumption model described in Chapter 2, Section 2.3 (excluding $E_{logging}$). Figure 3.4, 3.5 and, 3.6 show a comparison between the amount of energy consumed by each one of the 20 sensor nodes when both ASTR and DPCAS are implemented. We can notice that DPCAS performs better with stationary temperature data. However, for non-stationary data, ASTR has the upper-hand.

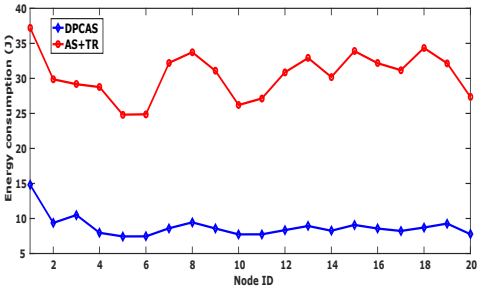


Figure 3.4: Energy consumption comparison for temperature data

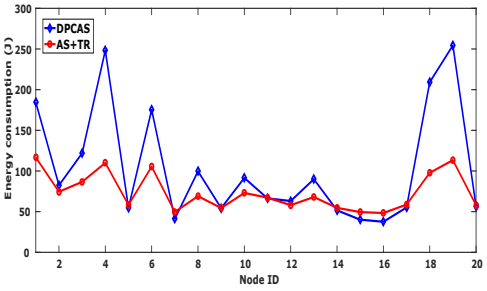


Figure 3.5: Energy consumption comparison for humidity data

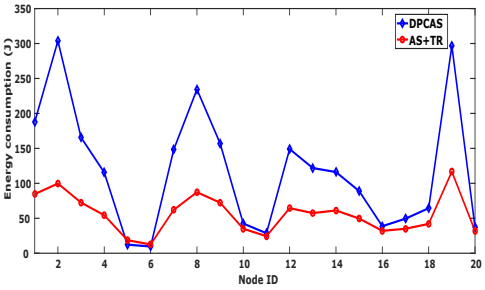


Figure 3.6: Energy consumption comparison for infrared data

3.4.3/ Quality of The Replicated Data

Data quality is a very important factor in WSNs since the end-user depends on it to make appropriate decisions. Accuracy, precision, completeness, and consistency are the attributes that measure the quality of data.

When we reduce the sampling rate within a certain period, we risk missing sudden variations in measurements. Thus, the estimation of these irregular non-sampled data may

exceed the desired error threshold. To study the impact of the adaptive sampling algorithms on the integrity of the replicated data we compare the estimated measurements with its corresponding raw data collected by the sensor nodes, and we calculate the values of 4 quality metrics: Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), Mean Square Error (MSE), Root Mean Square Error (RMSE). The lower the values of these metrics the better are the results.

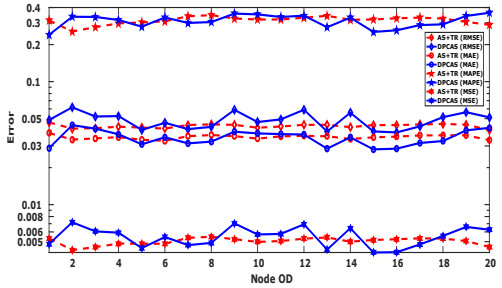


Figure 3.7: Quality metrics comparison for replicated temperature data

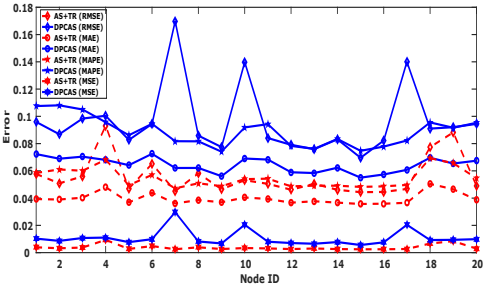


Figure 3.8: Quality metrics comparison for replicated humidity data

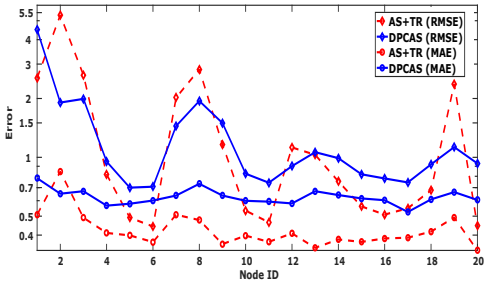


Figure 3.9: Quality metrics comparison for replicated infrared data

Figure 3.7, 3.8, and 3.9 show a comparison between the quality metrics for each set of data of the three environmental features. For temperature data, both algorithms are neck to neck. However, AS+TR has a clear superiority for humidity and Infrared. Since infrared contains 0 values, MPAE could not be computed. As for MSE in order to keep Figure 3.9 simple and comprehensible, instead of plotting the graph we provide the average values which are 2.49 and 2.72 for AS+TR and DPCAS respectively.

Adaptive sampling makes a trade-off between data quality and the amount of sampled measurements, to deliver a minimum amount of readings while satisfying the quality requirements of the application. Thus, the integrity of data depends on how tolerant is the end-user to the error in replications. The obtained results demonstrated that our method was able to reproduce the whole data set with less error and better quality compared with DPCAS.

3.5/ CONCLUSION

In this chapter, we proposed an energy-efficient data reduction method for Wireless Sensor Networks based on a combination of the adaptive sampling and dual prediction mechanism techniques. The former allows the sensor to adapt its sampling rate according to the variance in data. Thus, the sensor samples relevant data only and avoids the sampling of redundant and insignificant information. The latter enables the Sink to estimate the collected data through a prediction mechanism that is shared with the sensor node. Thus, instead of transmitting all the readings, the sensor reports to the Sink a measurement only when the estimation exceeds a predefined error threshold. By merging these two techniques together, we were able to reduce radio communication and data sensing at the same time. Since these two activities are considered to consume most of the energy resources, we were able to preserve a great amount of energy and extend the lifetime of the network significantly compared to another similar technique.

4

SLEEP SCHEDULING FOR CLUSTER-BASED SENSOR NETWORKS

It is evident that a sensor node consumes the most when it is activated. When the latter is put into sleep it consumes a negligible amount of energy. The wake-up sleep scheduling algorithms that gained a lot of traction in the research community aim to exploit this by organizing sleep schedules for the sensor nodes without sacrificing coverage, connectivity, nor data quality. In this chapter, we propose and present a periodical scheduling scheme for cluster-based sensor networks based on the spatial temporal correlation of the deployed sensor nodes.

4.1/ INTRODUCTION

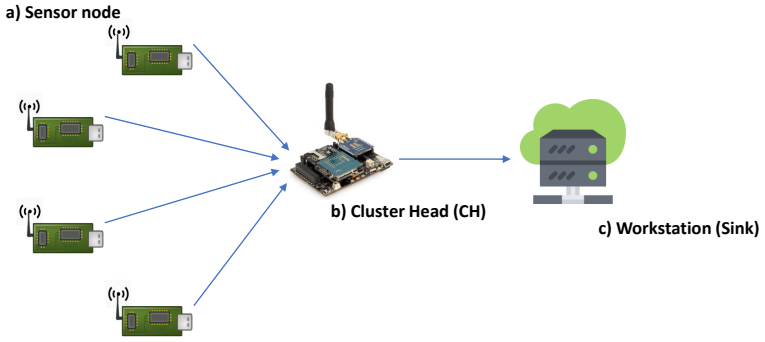
In this chapter, we present a Spatial-Temporal Correlation-based Approach for Sampling and Transmission rate Adaptation (STCSTA) in cluster-based sensor networks. Unlike the previous approaches, the sensor nodes do not need to run any distributed algorithm. It is rather a centralized approach where the cluster head is responsible for collecting data from its member sensor nodes and computing a correlation function in order to measure the correlation degree among them. The sensors that show high correlation will be asked to reduce their sampling rate and the ones showing low correlation will be asked to increase it. Moreover, in order to ensure the integrity of the data, the previously described reconstruction algorithm (Chapter 2, Section 2.5.3) is implemented on the Sink station in order to reconstruct the “non-sampled” measurements. This proposed algorithm belongs to the wake-up sleep sub-category of duty cycling presented in Chapter1, Section 1.2.2.2.

The rest of the chapter is organized as follows, in Section 4.2 the system model is briefly explained. A detailed explanation of the proposed approach is provided in Section 4.3, while experimental results are discussed in Section 4.4. This chapter ends with a conclu-

sion section, in which the contribution is summarized.

4.2/ System Model

We consider a set S of N sensor nodes and C cluster heads deployed over a specific monitoring area at locations LS=$\{ls_1, ls_2, ..., ls_N\}$ and LC=$\{lc_1, lc_2, ..., lc_C\}$ respectively, where a sensor S_i is located at the location ls_i and a cluster-head C_j is located at the location lc_j, and the Sink S is placed in a distant location at a position l_0. Sensor nodes are grouped into clusters, where each one of them belongs to one cluster only. The cluster heads are considered to be more powerful than sensor nodes in terms of processing capabilities and they have been allocated larger energy resources. Figure 4.1 illustrates an example of th



The network is periodic and operates in rounds, where each round R is exactly P seconds, and it is subdivided into m time slots, where at each time slot a sensor samples one measurement. Therefore, the maximum sampling rate (SR_{max}) is considered to be P/m samples per round. During the very first round, each sensor node collects data using the maximum sampling rate SR_{max} and transmits the readings to the CH after each acquisition. On the CH level, when the latter receives a measurement from any sensor S_i it stores the values in its memory and routes it directly to the Sink. At the end of the first round, the CH would have stored in his memory the following matrix M, where n is equal to the current sampling rate (SR_{max}) in this case, and N is the number of sensors in the cluster.

$$M = \begin{bmatrix} x_1^1 & x_1^2 & x_1^3 & \dots & x_1^n \\ x_2^1 & x_2^2 & x_2^3 & \dots & x_2^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_N^1 & x_N^2 & x_N^3 & \dots & x_N^n \end{bmatrix}$$

The CH than proceeds to computing the correlation between each pair of sensors (The number of possible pairs is $\frac{N(N-1)}{2}$). Using the obtained correlation results the CH calculates then transmit to each sensor node its new SR. A detailed explanation of how the correlation is calculated and how the new SR is determined is provided in Section 4.3. For the next round, each sensor samples data according to its new sampling rate provided by the CH. For Instance, if the latter demands a specific sensor to reduce its sampling rate by 40%, and supposing that SR_{max} is equal to 50 measures/round, the sensor is supposed to sample 30 measurements instead. If each round is 10 minutes long (600s), instead of sampling a measurement every 12 seconds (600/50), the sensor would sample a measurement every 20 sec (600/30). Moreover, knowing the duration of each round, the maximum sampling rate and the time stamp when each measurement was received, both the Sink and the CH are capable of identifying the non-sampled data, which will be replaced by "Nan" (see matrix M′) in order to reconstruct them later at the Sink station and in order to make the computation of the correlation among sensor nodes easier for the CH as explained in Section 4.3. Therefore, the stored matrix that is used to compute the correlation will actually be as shown below, where n is equal to the number of samples per round (SR):

$$M' = \begin{bmatrix} x_1^1 & x_1^2 & x_1^3 & \dots & x_1^{50} & Nan & x_1^n \\ x_2^1 & x_2^2 & x_2^3 & \dots & Nan & Nan & x_2^n \\ \vdots & \vdots & \vdots & Nan & \vdots & \vdots & \vdots \\ x_N^1 & x_N^2 & x_N^3 & \dots & x_1^{50} & Nan & x_N^n \end{bmatrix}$$

4.3/ THE PROPOSED APPROACH (STCSTA)

In this section, we will explain in detail, how the correlation between sensor nodes and the new sampling rates of each sensor are calculated.

4.3.1/ COMPUTING CORRELATION AND SAMPLING RATE ALLOCATION

Algorithm4 - line(1-13): After a round is completed, each sensor node would have transmitted to the cluster head a different number of measurements since the sampling rate of each one of them can be different. Nevertheless, as mentioned earlier the CH identifies the non-sampled data and fills their corresponding place in the vector by a “Nan” value, therefore all the vectors will have the same size n. However, the correlation between two vectors containing “Nan” values cannot be computed. Therefore, each one is replaced by the value of the first “non-Nan” value that comes before it in the same vector. For instance, in the “M′” matrix, x_1^{51} is Nan it will be set equal to the same value as x_1^{50}, and x_2^{50} and x_2^{51} are set equal to the same value as x_2^{49}, and so on.

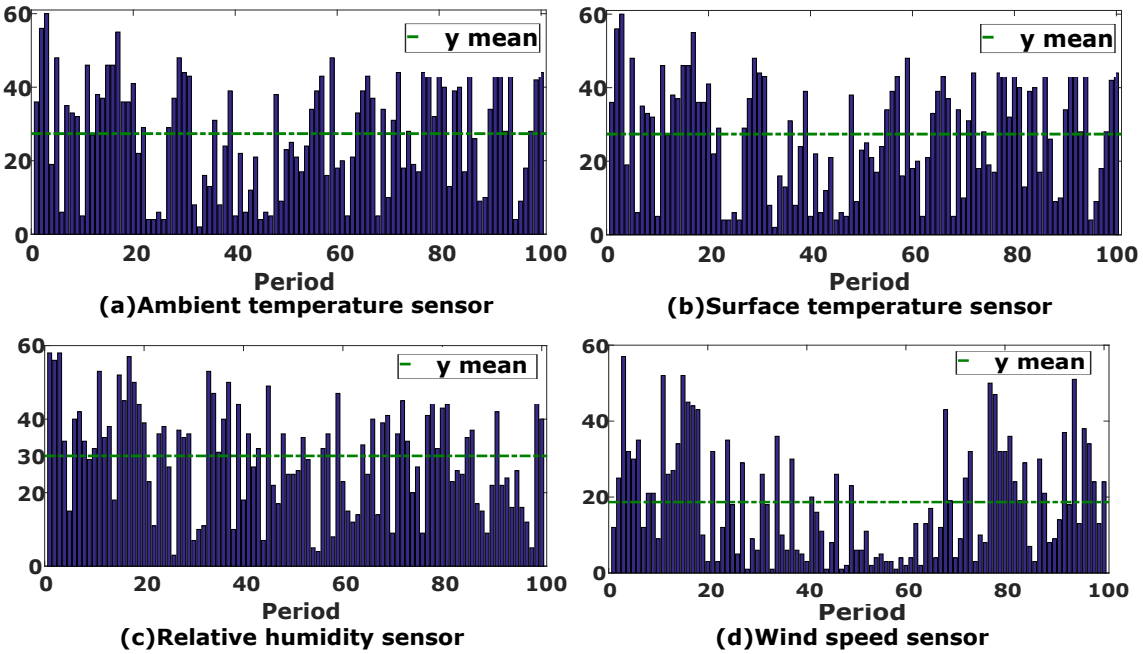


Figure 4.2: Figure showing the number of moderately & highly correlated sensors (pearson correlation coefficient ≥ 0.5) during each one of the first 100 Periods

Algorithm4 - line(16-21) : Afterward, the linear dependency of each pair of vectors (v_i,v_j) ∈ M′ is calculated using the Pearson correlation coefficient. The latter is known as the best method of measuring the association between variables of interest because it is based on the method of covariance. It gives information about the magnitude of the association, or correlation, as well as the direction of the relationship. The Pearson correlation coefficient is described in the Equation 4.1 below, where μ and σ are the mean and standard deviations.

$$\rho(v_i, v_j) = \frac{1}{n-1} \times \sum_{k=1}^{n} (\frac{\overline{v_{ik} - \mu_{v_i}}}{\sigma_{v_i}})(\frac{\overline{v_{ik} - \mu_{v_j}}}{\sigma_{v_j}}) \tag{4.1}$$

The justification behind using the Pearson correlation can is illustrated in Figure 4.2. We have used a data set of 92 sensors to generate 4 graphs that show the number of sensors that are moderately & highly correlated with 4 randomly chosen sensors during each period and for the first 100 periods. For instance, in Figure 4.2(a) we notice that this randomly chosen ambient temperature sensor correlates with a large number of sensors during each period. On average it correlates with 27 sensors as the mean values show. Same for Figure 4.2(b) and (c) on average these sensors correlate with approximately 30 other sensors that are in the same cluster. However, the mean value in Figure 4.2(d) is significantly lower (mean=19), in Section 4.4 we will see how this will reflect on the results.

Heterogeneous environmental data besides other types of data such as medical data

(vital signs), movement tracking data (speed, acceleration, location) and etc., are usually highly and/or moderately correlated. This correlation thus can be used in order to reduce the number of transmitted measurements by deriving values from other observed ones. This is indeed the motivation behind using correlation to adapt the sampling rate of the sensors.

Algorithm4 - line(22-27): After computing the correlation value of each sensor i with all the other sensors belonging to the same cluster, the CH looks for the sensor j that it correlates the most with as shown in Table 4.1.

Algorithm4 - line(28-37): Afterward, the CH counts the number of occurrences of each sensor j in the second column of the table and stores them in a list according to their ascending order.

Sensor i	Sensor j (has the max correlation with Sensor i)	Correlation degree
1	54	0.91
2	7	0.87
3	5	0.70
4	2	0.96
5	6	0.75
...	...	..
n	32	0.88

Table 4.1: The correlation table

Algorithm4 - line(38-48): Starting from the first sensor j in the ordered list, the CH looks in Table 4.1 for the sensor j in the first column and extracts the value of its max correlation from the third column. Then the CH notifies j that its sampling rate must be reduced proportionally to the correlation value. For instance, if sensor 5 was first in the ordered list, the CH would notify it that its sampling rate for the next round must be reduced by 75%, since its level of correlation with sensor 6 is 0.75. Then the sensor j (in this case 5) is flagged as already notified. Thus, for the next sensor j in the ordered list, if its matching sensor i is already flagged. Instead of reducing its sampling rate proportionally to the level of correlation, it is reduced by (100 - i's reduction $\%$). For instance, if the next sensor j in the list is 3, it matches with sensor 5 in Table 4.1, therefore, it's sampling rate will be reduced by $100 - 75 = 25\%$. And so on, until the last element in the ordered array.

Algorithm4 - line(49-55): However, some sensors may not appear in the second column of the Table 4.1, since they have not been matched with other sensors. Therefore, the CH looks for these sensors in the 1'st column of Table 4.1, and for each sensor i, it finds their matching sensor j in the second column, looks at how much the sampling rate was reduced for sensor j and notifies sensor i that its sampling rate must be reduced by (100

- sensor $j's$ reduction $\%$).

The same explained operation is repeated at the end of each round. Therefore, enabling each sensor node to adjust its sampling rate according to its level of correlation with other sensors in the network. The Algorithm 4 illustrates the proposed method that is implemented on the CH.

Algorithm 4 STCSTA

Require: $SRmax$ (1 sample/ X seconds)

$k \leftarrow 1$
for each sensor j in the cluster **do**
  receive the first value v_j^0 at the beginning of the round
  $data[j][0] \leftarrow v_j^0$
  $lastReceived[j] \leftarrow v_j^0$
end for
while ! end of round **do**
  if nothing is received from sensor j after X seconds **then**
    $data[j][k] \leftarrow lastReceived[j]$
  else if v_j^n is received during the X seconds count **then**
    $data[j][k] \leftarrow v_j^n$
    $lastReceived[j] \leftarrow v_j^n$
  end if
  $k \leftarrow k + 1$
end while
if end of round **then**
  for i=1 to N **do**
    for j=i+1 to N **do**
      $corrArray[i][j] \leftarrow PearsonCorr(data[i][:], data[j][:])$
    end for
  end for
  for i=1 to N **do**
    $maxCorr[i][0] \leftarrow i$
    $[index, value] \leftarrow max(corrArray[i][:])$
    $maxCorr[i][1] \leftarrow index$
    $maxCorr[i][2] \leftarrow value$;
  end for
  $k \leftarrow 1$
  for each element i $\in$ the second column of maxCorr **do**
    if i $\notin$ first column of countOcc **then**
      $count \leftarrow$ count how many times i occures in the second column of maxCorr
      $countOcc[k][0] \leftarrow i$
      $countOcc[k][1] \leftarrow count$
      $k \leftarrow k + 1$
    end if
  end for

```
    order countOcc in ascending order according to the second column
    k ← 1
    for each element j ∈ the first column of countOcc do
      match ← maxCorr[j][1]
      if reduce[match-1] is empty then
        Notify sensor j that its sampling rate must be reduced by (maxcorr[j][2]*100)%
        reduce[j − 1] ← (maxcorr[j][2] ∗ 100)
      else
        Notify sensor j that its sampling rate must be reduced by (100 - reduce[match-1])%
        reduce[j-1]=100 - reduce[match-1]%
      end if
    end for
    for j=1 to N do
      if reduce[j-1] is empty then
        match ← maxCorr[j][1]
        Notify sensor j that its sampling rate must be reduced by (100 - reduce[match-1])%
      end if
    end for
  end if
```

4.3.2/ Analysis Study

The objective of this algorithm is to create and manage a sampling rate balancing system based on the correlation degree between the nodes belonging to the same cluster. The idea is to match each sensor node with the one that correlates the most with, in such a way that, if one node of the paired couple reduces heavily its sampling rate, the other one keeps it high and vice versa, allowing them to compensate one another. This compensation mechanism is crucial for the success of the reconstruction algorithm in terms of minimizing the estimation error and increasing the quality of the replicated data. The latter relies on the correlation among sensor nodes in order to reconstruct the non-sampled measurements. Therefore, if highly correlated sensors are missing data simultaneously this would negatively affect the accuracy of the reconstructed measurement. When the balancing of non-sampled data is kept in check on the CH level, the reconstruction algorithm on the Sink will theoretically produce better estimations.

In this section, we will illustrate an example that explains our algorithm step by step. The latter provides a better analysis of what happens at the end of each round on the cluster head to better understand why and how this compensation system works. Let us start by assuming that at the end of a given period, the CH has already computed the correlation between each pair of sensors belonging to the same cluster. In addition, we assume that the CH already matched each sensor with the one that correlates the most with and stored the results in a table similar to Table 4.2. The next step is to count for the sensors

appearing in the second row of the table how many times it has been matched. For instance, sensor 7 has been matched 4 times, sensor 1 has been matched 2 times, and sensor 10, 9, 3, and 8 have been matched only one time. The matched sensors are then ordered in ascending order according to how many times they have been matched. the order will then be: {sensor 8, sensor 3, sensor 9, sensor 10, sensor 1, sensor 7}.

Sensor i	1	2	3	4	5	6	7	8	9	10
Sensor j (has max correlation with i)	8	1	7	3	9	1	10	7	7	7
Correlation degree $*10^{-2}$	78	69	54	92	85	72	79	83	89	90

Table 4.2: Table showing for each sensor its best match (maximum correlation) and the degree of correlation with this match

Starting from the first sensor in the list (sensor 8) the CH looks for the sensor that it matches with. Looking at Table 4.2 we see that sensor 8 matches with sensor 7. The CH then checks whether the sampling rate of sensor 7 for the next round has been decided yet. If it is not the case the CH notes that the sensor 8 must reduce its sampling rate for the next round by 83%, since the correlation degree for sensor 8 with its match is 0.83. The CH then follows the same procedure for the next sensor in the ordered list. Sensors 3, 9, and 10 they all match with sensor 7 too, and since the sampling rate of sensor 7 has not been decided yet, their sampling rate will be reduced by 54%, 89%, and 90% respectively for the next round. Now the CH searches for the sensor that matches with the next sensor in the ordered list (sensor 1). Looking at Table 4.2 we see that it is sensor 8. However, the sampling rate of sensor 8 has been already decided to be reduced by 83%, therefore instead of reducing the sampling rate of sensor 1 by 78% it will be reduced by 100-83%, therefore 17% only. Same for sensor sensors 7, it matches with sensor 10, therefore its sampling rate must be reduced by 100-90% (10% only).

The next step is to adapt the sampling rate of the sensors that do not appear in the second row of the table, or in other words they have not been matched with other sensors in the cluster. In this example, the non-matched sensors are sensors 2,4,5 and 6. Starting by sensor 2, its match is 1, therefore the sampling rate of sensor 2 for the next round must be reduced by 100-17% (83%), same for sensor 4,5, and 6 their sampling rate will be reduced respectively by 46%, 11%, and 83%.

Before computing the percentage of the reduction in sampling rate, the matched sensors are first ordered in ascending order according to how many times they have been matched. The reason for this crucial step can be explained as follows: let us suppose the list has not been ordered, and the CH started by sensor 7, which has been matched 4 times with 4 different sensors. The sampling rate of sensor 7 will be reduced by 79%.

Therefore, eventually, the sampling rates of sensors 3, 8, 9, and 10 will be reduced by 21% only compared to 54%, 83%, 89%, and 90% respectively if the list was ordered. In consequence of not ordering the list first, the overall reduction in the sampling rate of the sensors would be reduced, which would lead to an increase in data transmission and energy consumption. Since sensor 7 can compensate for 4 other sensors, it is wise to leave it until the end, allowing the sensors that it matches to reduce more their sampling rate.

A summary of the results is illustrated in Table 4.3. We notice that if a sampling rate of a particular sensor is highly reduced, the one of the sensor that it correlates the most with will be proportionally and slightly reduced (e.g. sensors 2 and 1). This balanced reduction is meant to compensate for the matched sensor since the non-sampled values will eventually be derived mostly from its best match. Similarly, if the sampling rate of a sensor is slightly reduced, this will give more freedom to its match thus allowing it to highly reduce its sampling rate (e.g. sensors 5 and 9).

Sensor i	1	2	3	4	5	6	7	8	9	10
SR reduction (%)	17	83	54	46	11	83	10	83	89	90
Sensor j (has max correlation with i)	8	1	7	3	9	1	10	7	7	7
SR reduction (%)	83	17	10	54	89	17	90	10	10	10

Table 4.3: Table showing the % of SR reduction for each sensor compared with its match

4.4/ EXPERIMENTAL RESULTS

We implemented our algorithm in addition to DPCAS [Monteiro et al., 2017] in a custom WSN simulator built in Matlab, and we conducted multiple experiments in order to evaluate and compare their performances. In the simulation, we used the real sensor data-set described in Section 2.4.1 of Chapter 2. In DPCAS the parameter ϵ defines the error tolerance of the application, the greater is ϵ, the less is the amount of data that will be sampled and transmitted. However, the error of the estimated data will increase. Therefore, the value of ϵ is the level of trade-off between the quality of the replicated data and the amount of sampled and transmitted measurements. In our experimentation, we set up five different values for ϵ ranging between 0.1 and 0.5 and we compare our approach to DPCAS for each value of ϵ.

4.4.1/ SAMPLING AND TRANSMISSION REDUCTION

In this section, we will explore and compare the effectiveness of each algorithm in reducing the number of both sampled and transmitted data in three different scenarios. In the used data-set, each one of the 23 sensor nodes collects 4 different environmental features (ambient temperature, surface temperature, relative humidity, and wind speed). For simplicity and better visualization of the results, all the figures will illustrate the percentage of the aggregated sum of the data sampled and transmitted by the 23 nodes combined and for all features.

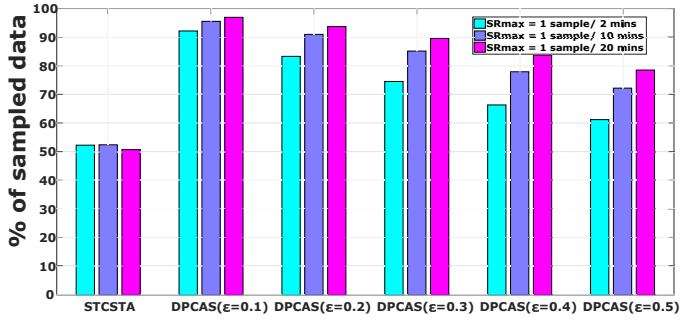


Figure 4.3: Average percentage of data sampled by each sensor node

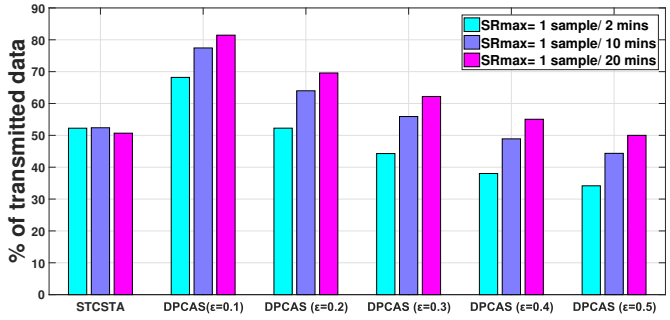


Figure 4.4: Average percentage of data transmitted by each sensor node

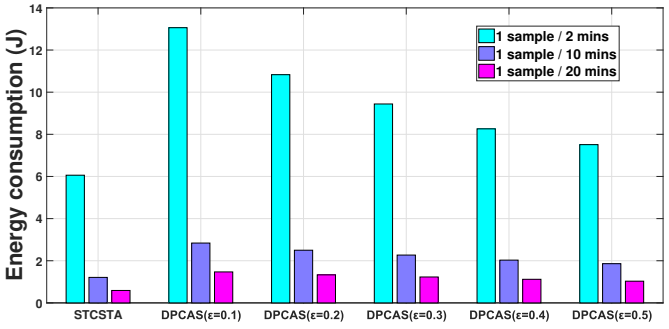
Figure 4.3 and Figure 4.4 show that on the one hand, the bigger is the sampling interval between two consecutive measurements (higher variations in data), the greater is the average percentage of both sampled and transmitted data will be when DPCAS is deployed. On the other hand, when our approach (STCSTA) is deployed, the average percentage remains stable despite the level of variations in collected measurements, which makes it more robust, dynamic and tolerable to high variations. This is not the case for DPCAS however, its effectiveness can be significantly affected (a double-digit increase in sampled and transmitted data) depending on the type of data being collected. For sampled data, Figure 4.3 shows that STCSTA outperforms DPCAS in all scenarios and for all the values of ϵ. Figure 4.4 shows the average percentage of data transmitted by each one of

the 23 nodes for both algorithms in 3 different scenarios and using different ϵ for DPCAS. The obtained results show the following: STCSTA outperforms DPCAS when $\epsilon \leq 0.2$ in all scenarios. However, for $\epsilon = 0.3$ DPCAS transmits less data in the first scenario (SR_{max} = 1 sample/ 2mins), but more data in the other two scenarios (SR_{max} = 1 sample/ 10 mins and 1 sample/ 20 mins). Finally, for $\epsilon = 0.4$ and 0.5, DPCAS is slightly better in the first two scenarios. To sum it all up, the results in Figure 4.4 show that STCSTA outperformed DPCAS 9 times, the latter outperformed STCSTA 5 times, and finally, there is 1 tie.

4.4.2/ ENERGY CONSUMPTION

In this paragraph, we present a comparison between the average energy consumed by the 23 sensor nodes when DPCAS and STCSTA are deployed.

Figure 4.5: Average energy consumption of each sensor node



Knowing that in DPCAS an algorithm must be deployed on the node that handles 4 different sensors at a time. The node needs to perform reading and writing in the memory, and it needs to compute instructions using the CPU. Therefore, the node will consume additional energy ($E_{logging}$ and $E_{Processing}$). However, for STCSTA, the node does not have to run an algorithm, nor to perform read and write in the memory, it simply collects a measurement using the integrated sensors, and directly transmits it to the CH. Therefore, no additional energy consumption is required. Figure 4.5 shows the average energy in Joule consumed by each one of the 23 deployed nodes. We can see that our approach consumes approximately from 20% up to 60% less energy than DPCAS depending on the scenario and the value of ϵ.

Comparison with a baseline method: The previously described results demonstrated that our approach STCSTA outperforms DPCAS in terms of energy preservation. The DPCAS algorithm in [Monteiro et al., 2017] was compared to two other approaches that use a similar technique, namely EDSAS [Gupta et al., 2011] and ASTCP [Al-Hoqani et al., 2015]. The ASTCP algorithm was inspired by the EDSAS. Moreover,

the DPCAS algorithm was inspired by both ASTCP and EDSAS. In this section, we will use the EDSAS as a baseline for comparison since it was the root algorithm that inspired both ASTCP and DPCAS. Table 4.4 below shows the average energy consumed by each node in all scenarios and for the same value of ϵ=0.1 used in [Monteiro et al., 2017]. The obtained results are fairly similar to the ones obtained in [Monteiro et al., 2017] and our approach remains better.

Algorithm	STCSTA			DPCAS			EDSAS		
Sampling Rate 1 sample/ x min	x=2	x=10	x=20	x=2	x=10	x=20	x=2	x=10	x=20
Energy (J)	6.06	1.21	0.59	13.06	2.84	1.47	13.38	2.92	1.52

Table 4.4: Table comparing STCSTA and DPCAS to the baseline EDSAS

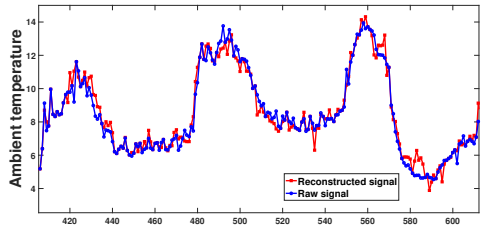
4.4.3/ The Quality of The Replicated Data

In order to measure the quality of the final set of data, we use the accuracy of the estimations as the validation criteria. Specifically, we use the Root Mean Square Error (RMSE) and the Mean Absolute Error (MAE) as an accuracy metric. Table 4.5 shows the RMSE and MAE of the estimated data for the three scenarios. For ambient temperature, surface temperature and relative humidity the errors are low. This is due to the fact that the spatial-temporal correlation of these features is strong, so the estimation algorithm can obtain an accurate and solid relationship based on mining correlation rules. Table 4.5 also shows that the error increases when the sampling interval widens. The bigger is the sampling interval, the weaker is the temporal correlation, therefore the harder is for the estimation algorithm to accurately estimate values. For Wind direction, the errors increase significantly but they are still proportionally low compared with the range of value for the wind speed (between 0 and 350 m/s). Wind speed has no spatial correlation with any other feature. Moreover, the wind speed value varies significantly between one sample and the other as shown in Figure 4.6d, therefore the temporal correlation is weak as well, that is why it has the highest error among other features.

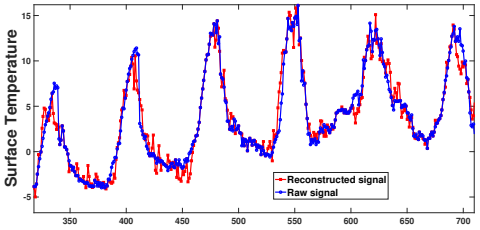
Figure 4.6 shows the reconstructed data for ambient temperature, surface temperature, relative humidity, and wind speed respectively. As shown in the figures, the data estimation (reconstruction) algorithm has been able to capture both the dynamics of the time series as well as the correlation across given inputs, therefore achieving a very satisfying reconstruction quality. To conclude on this, simulation results presented in this section, demonstrated that the Sink is capable of reproducing the “non-sampled” data with a tolerable error margin. Thus, using our approach a sensor node can significantly reduce its sampling rate without affecting the integrity of the data.

Table 4.5: Quality of the reconstructed data

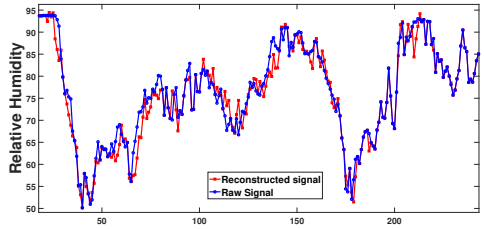
		Ambient Temp	Surface Temp	Relative Humidity	Wind Direction
1 sample/2 mins	**RMSE**	1.12	1.33	2.7	16.5
	MAE	0.71	0.91	1.89	8.78
1 sample/10 mins	**RMSE**	1.26	1.56	3.68	18.26
	MAE	0.74	1.09	2.55	9.13
1 sample/20 mins	**RMSE**	1.43	1.95	4.78	23.53
	MAE	0.87	1.43	3.21	11.93



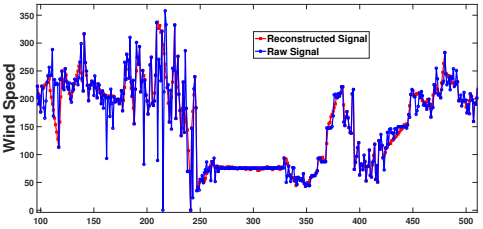
(a) Reconstructed ambient temperature data



(b) Reconstructed surface temperature data



(c) Reconstructed relative humidity data



(d) Reconstructed wind speed data

Figure 4.6: Quality of the reconstructed data sets

4.4.4/ The Effect of The Scheduling Strategy on Error Minimization

The previous results have evaluated the efficiency of our proposed approach (STCSTA) in terms of reducing data transmission and energy consumption as well as the quality of the data replicated on the Sink. However, as previously explained in Section 4.3.2, the objective of our algorithm is to guarantee that the highly correlated sensors are not skipping data sampling simultaneously in order to reduce the reconstruction error. That was in theory, therefore, in this section, we put the theory into practice in order to justify this claim.

Instead of building a list of matching sensors, ordering the list, and reducing the sampling rate of each sensor proportionally to its match, we eliminated the steps from line 29 and upward in Algorithm 4, only to allow a sensor to reduce its sampling rate according to its highest degree of correlation. For instance, let's assume that the sensor 1 has the highest correlation degree with sensor 5 (0.8). Without checking whether sensor 5 has

already reduced its sampling rate or not, it will automatically reduce it by 80%. There is a chance that sensor 5 has already reduced its sampling rate lets say by 70%. Thus, both sensors 5 and 1 will skip sampling simultaneously which would, in theory, affect negatively the reconstruction algorithm, which will lead to an increase in the reconstruction error. We will be calling this method "The exaggerated sampling reduction" method. Table 4.6 shows the % of the increase in the reconstruction error when this method is applied. We notice that the Reconstruction error increases significantly in all scenarios and for all environmental features, which justifies our controlled sampling strategy.

Table 4.6: Percentage of increase in reconstruction error (the exaggerated sampling reduction method)

		Ambient Temp	Surface Temp	Relative Humidity	Wind Direction
1 sample/2 mins	RMSE	16.9 %	34.5 %	20 %	45.3 %
	MAE	12.6 %	41.7 %	16.93 %	23.4 %
1 sample/10 mins	RMSE	26.9 %	44.8 %	35.8 %	52.7 %
	MAE	39.1 %	52.3 %	29.4 %	75.2 %
1 sample/20 mins	RMSE	25.8 %	48.2 %	50.2 %	36.0 %
	MAE	25.2 %	44.0 %	45.7 %	59.2 %

4.4.5/ Scalability and Limitations

Obviously, the scalability of such a network depends on the computational power of the CH and its memory capacity. A more powerful CPU and big memory size mean that the CH could handle a large number of sensors simultaneously. The weaker is the CPU and the smaller is the memory size, the fewer nodes a CH can handle. A great number of devices that can be used as a CH are currently available in the market, they all have different features and characteristics. One can find cheap less powerful CH device for personal use or an expensive and powerful device for commercial use. Therefore, the choice of the CH depends on the size of the network a user wants to deploy. A network consisting of thousands of nodes will certainly need a powerful CH. However, a network consisting of a few hundred or tens of nodes could work just fine with a less powerful CH.

Our proposed algorithm is not very complex though, it has a complexity that is linear in time (O(n)). This linear complexity allows the CH to handle a large number of nodes with minimal computational power. Regarding the memory size required by the STCSTA, assuming that the number of nodes in the cluster is N, and each transmitted value is encoded into 8 bytes.

- $8 \times (N(SR_{max} + \frac{1}{2}N + 4) + 1)$ bytes is the memory size required by the Algorithm 4 from line 1-27. Figure 4.7 shows the memory size needed by the CH in function of

Figure 4.7: Memory size needed for the first part of the Algorithm (line 1-28)

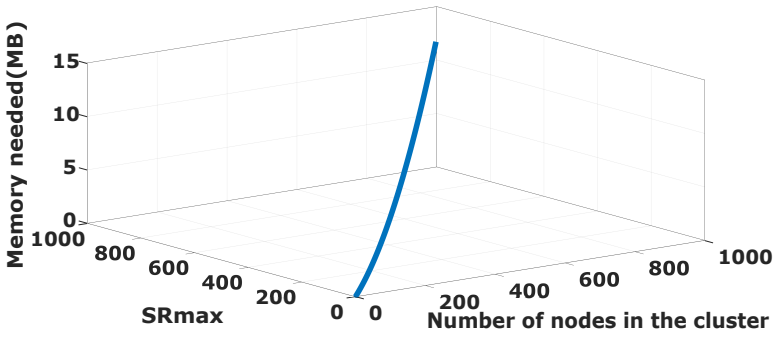
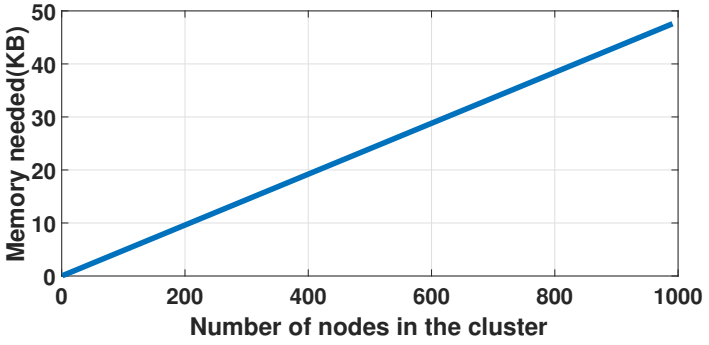


Figure 4.8: Memory size needed for the second part of the Algorithm (line 23-57)



SR_{max} and the number of nodes belonging to the cluster.

- $8(\times 6N + 1)$ bytes is the memory size required by the Algorithm 4 from line 22-55 if we assume that the matching sensors are at maximum equal to the number of sensors in the cluster. Figure 4.8 shows the maximum memory size needed by the CH in function of the number of nodes belonging to the cluster.

The maximum memory size required by the CH is $8 \times Max(N(SR_{max} + \frac{1}{2}N + 4) + 1, 6N + 1)$ bytes, since the values stored in the first part of the Algorithm (1-15), could be cleared once the sensors have been matched (Algorithm 4, line 16-27).

Nevertheless, the greater is the number of nodes belonging to the same cluster, the better is the correlation among these nodes, the fewer data a sensor will sample and transmit which eventually leads to less energy consumption. Therefore, the number of sensors belonging to the same cluster should be maximized in function of its computational and memory resources.

As for the limitation of our proposed algorithm, it is evident when there is no or little correlation among the collected measurements, the sampling rate of the sensors will be always kept high. Since the role of this algorithm is to minimize the sampling rate of the sensor node, it will not be as efficient as it should be.

4.5/ CONCLUSION

In this chapter, we proposed a sampling and transmission rate adaptation algorithm for cluster-based sensor networks. This algorithm is deployed on the Cluster-Head (CH) and it operates in rounds. The latter controls the sampling rate of each individual sensor node by increasing it or decreasing it according to its spatial correlation with other sensors in the network. Moreover, we adopted and adapted a data reconstruction algorithm that is implemented on the Sink station. The latter can identify the “non-sampled” data that are not collected due to a decrease in the sampling rate of a specific sensor and it estimates them using an EM iterative approach that is capable of capturing the temporal and spatial correlation among the reported measurements. We presented experimentation that we have conducted on real sensor data of a network that was deployed at the Grand-St-Bernard pass located between Switzerland and Italy. We have compared our approach with a recent data reduction technique that combines both adaptive sampling and transmission reduction. The obtained results demonstrate that our proposal is better at reducing the energy consumption of the sensor node, thus extending the operational lifetime of the network while preserving the integrity and the quality of the data.

5

FROM THEORY TO PRACTICE

In this chapter we present an LPWAN-based remote monitoring system for state assessment and localization of firefighters to improve their safety during intervention. A small scale prototype of this latter has been validated through real experiments. Moreover, we have implemented on some of the sensor devices composing the system the energy management algorithms described in the previous chapters and computed their energy consumption through real-time measurement of the consumed current.

5.1/ INTRODUCTION

Firefighters are equipped with an immobility detector device also called the Personal Alert Safety System (PASS) that is integrated into the user's Self-Contained Breathing Apparatus (SCBA). If a firefighter remains motionless for a certain period of time, a loud audible alert is triggered to notify the Firefighter Assist and Search Team (FAST) deployed in the area of intervention that the wearer of the PASS device is in trouble and in need of rescue. The first generations of the PASS devices needed to be manually armed by the firefighters, which leads to fatalities among crew members that entered the area of intervention without arming it. Therefore, new generation devices have been integrated into the user's Self-Contained Breathing Apparatus (SCBA), and they are automatically armed when the SCBA is turned on. When the SCBA is not being used the PASS device should be deactivated to prevent false alarm, but this does not happen all the time. For instance, a firefighter might need to remove the SCBA and put in on the ground to change the air cylinder, which triggers a false alarm thinking that he is not moving. Moreover, if a crew member did not move for a few seconds but he is okay and did not manually reset the PASS device, a false alarm will also be triggered. Therefore, this device is not reliable enough since it triggers frequently false positives which lead to developing a tolerance for sounding alarms among the crew. As a consequence, they do not seem to be concerned about it as they should and the alarms are just ignored sometimes.

In this chapter, we first present a **PER**sonal **L**PWAN s**Y**stem (PERLY) for state assessment and localization of firefighters during interventions. Our proposal is more reliable compared to the PASS, it also adds additional important functionalities and minimizes the false positive alarms. The designed prototype and the proposed solution were adapted according to the needs and requirements of the fire brigade of the region of Doubs in France. Several interviews were held with the personnel from the fire and emergency response department that included discussions regarding the reliability, cost, and security of the system and the collected data. The interviews led to developing and validating a prototype of a plug and play system that is specifically tailored for the needs of the fire brigade and also designed to work in areas with no network coverage. Secondly, in order to ensure a maximum operational lifetime for the network, our proposed data management algorithms were implemented on some of the sensor devices composing the system. We measured in real-time the energy consumption of sensing, processing, and transmission activities of the devices in addition to the overall energy consumed by each algorithm.

5.2/ Background and Motivation

A variety of Wireless Sensor Network (WSN) systems have been proposed in the literature to support firefighters during their interventions. such as wearable body-sensors systems for health monitoring [Chen et al., 2007, Shahriyar et al., 2009, Hao et al., 2008], navigation support systems [Yang et al., 2018, Klann et al., 2007, Ramirez et al., 2012], fire detection systems [Cantuña et al., 2017, Kanwal et al., 2017, Khamukhin et al., 2016], etc. However, while the usefulness of these systems is acknowledged [Sha et al., 2006], they are only partially usable either because they rely on the existence of a pre-deployed and fix communication and/or localization infrastructure or because they are simply too complex to be accepted and used by firefighters. In our interviews with the personnel of the fire brigade department of the Doubs region, we obtained very useful system design information that helped us propose, develop and validate a personal alert safety system that is specifically tailored for their needs. The main system design requirements are the following:

- A simple, lightweight, and convenient system that could be easily integrated with the current equipment used by the firefighters and that could be accepted and adopted by the personnel.
- A plug and play system that does not rely on a pre-existing infrastructure.
- A system that is low cost, fully automated and requires minimum user intervention.

- A system that ensures reliable delivery of the emergency alert to the concerned rescue team.
- A system that reduces false flags, monitor the heart condition of firefighters and is able to localize a firefighter in case of an emergency.
- A low power, low energy consuming system that can last until the intervention ends.

The rest of this chapter is organized as follows: in Section 5.3, a general overview description of the LoRaWan network architecture, components, and security is presented. In Section 5.4 the proposed personal LoRaWan-based system (PERLY) is described. A detailed explanation of the software implementation for emergency detection is presented in Section 5.5, while the system validation is demonstrated in Section 5.6 and the detailed energy consumption results are presented in Section 5.7. This chapter ends with a conclusion section, in which the contribution is summarized.

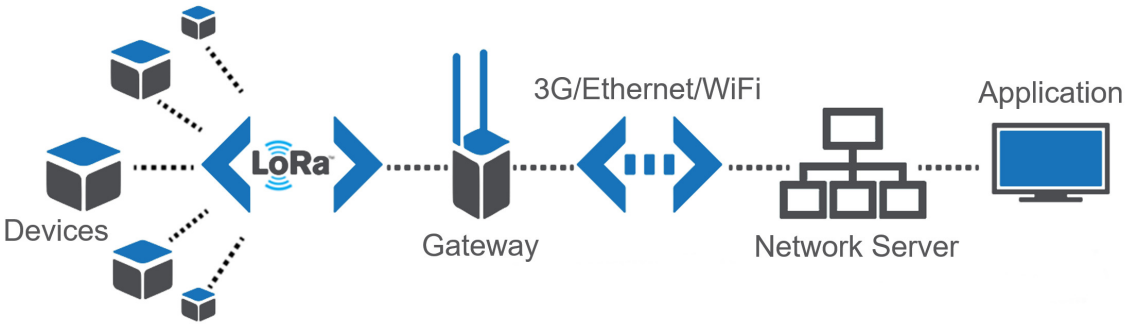
5.3/ A General Overview of The LoRaWan Network

In our proposed system, we use the LoRaWan network protocol to transmit the alert notifications to a centralized control room (CCR) and to the on-ground chief (OGC) supervising the intervention. The benefits that the LoRaWan protocol brings are long range, low cost, low energy, infrastructure-less, etc. which are essential for the requirements of the system. In this section, we will provide a brief introduction on the LoRaWan network architecture, the different components composing the network and finally its security.

LoRaWan is a wireless communication standard that stands for “Long Range Wide Area Network”. Its main characteristics are that it allows low powered IoT sensor devices to communicate with internet-connected applications over a long distance (multiple kilometers). LoRaWan operates in an unlicensed frequency band (867-869Mhz for Europe). Therefore, it is perfectly possible for anyone to set up a network for the cost of a few hundred Euros or Dollars, that has a coverage of a few kilometers. Figure 5.1 illustrates the different network components of the LoRaWan protocol, it mainly consists of four, namely:

- IoT sensor devices: these devices are categorized into three classes:
  1. Class A: it is the default class, it supports bi-directional communication with the gateway, and it requires low energy to operate. Uplink messages can be sent at any time, the device then opens two receive windows at specified times after an uplink transmission. If the server does not respond in either of these receive windows, the next opportunity will be after the next uplink transmission from the device.

Figure 5.1: The LoRaWan network architecture



2. Class B: it also supports bi-directional communication and it extends class A by adding additional scheduled receive windows. The gateway sends time synchronized beacon allowing to know when the device is listening.
3. Class C: are bidirectional with maximal receive slots. This means that they are continuously listening unless they need to transmit. Thus, Class C devices consume the most energy.

- The Gateway: it forms a bridge between the devices and the Network Server. Uplink messages are transmitted to the gateway using low power LoRa protocol, while the gateway uses the conventional high bandwidth networks such as WiFi, 4G, Ethernet, etc. A device is not assigned to a specific gateway, all the gateways in the range of this device will receive the messages and will forward them to the Network Server using a packet forwarder software. It is the responsibility of the Network Server to filter packets and remove redundant data. As for downlink, the Network Server chooses the best gateway to forward the message to the targeted device.

- Network Server: this component is the brain of the network, it is responsible for the following tasks:

  – Aggregate all the upcoming data forwarded by all the gateways and their associated devices in the network.
  – Assign each device to a specific application and route the messages transmitted by each device to its corresponding application.
  – Control the LoRa radio configuration of the gateway.
  – Select the best gateway for downlinks.
  – Store messages until a Class A or B device is ready to receive them.
  – Remove duplicates.
  – Monitor the devices and the gateway.

- Application: it is the final destination of the transmitted data, it is written and customized by the manufacturer or the developer to suit its needs. It could be a mobile application, a Web application or any other type of application.

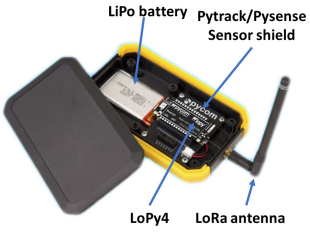
Security is also one of the main characteristics of the LoRaWan network. The communication between the sensor device and its corresponding application is protected via two security keys:

- The network session key: it is used by the sensor device to encrypt the whole frame (header + payload). The Network Server having the same key can verify the identity of the sender.
- The application session key: it is used to encrypt the payload in the frame. This key is only known by the application linked with this device, the Network Server cannot see what is in the payload, it only needs to know to which application should the message be forwarded. The latter decrypts the payload using the same key.

5.4/ THE PROPOSED SYSTEM'S COMPONENT DESCRIPTION

In this section, we will explain in depth the functionalities of each component, where it belongs, its role in the hierarchy of the network, and how these different components are interconnected in order to transmit information from inside the intervention area to a destination server.

Figure 5.2: The firefighter's equipement



(a) The IoT-device

(b) The smart-watch

In order to measure the heart rate of a firefighter, and detect mobility, a smart-watch is proposed as a sensing unit (Figure 5.2b). For this purpose, we use the Huawei Watch 2 that is equipped with the required PPG and accelerometer sensors. An algorithm is implemented on the watch that can detect abnormal heart rate measurements and the activity state of the firefighter. Then, it decides whether the system wearer is at risk. If this is the case the watch transmits an alert message to a GPS enabled IoT device and the

latter re-routes this message alongside the GPS coordinates of the concerned firefighter to the CCR and the OGC in order to facilitate extraction.

The GPS enabled IoT device used in our prototype is the Pytrack sensor shield from Pycom (Figure 5.2a), it includes an accurate Global Navigation Satellite System (GNSS) and Glonass GPS in addition to a 3 axis 12-bit accelerometer. A LoPy4 module is attached to the Pytrack, the latter includes a LoRa and WiFi radio modules. A WiFi or a BLE socket is opened between the smart-watch and the LoPy4, it is responsible for transmitting the alert messages from the watch to the IoT Device. Once the LoPy4 receives the message, it orders the Pytrack shield to acquire the GPS location, and finally, the LoPy4 transmits the alert alongside the GPS coordinates via its LoRa radio module to the CCR and OCG. **NB**: In our prototype, we have used WiFi instead of BLE to establish a communication link between the smart-watch and the IoT device, since BLE is not fully functional on Pycom devices.

Lastly, in addition to monitoring the firefighters, we were interested in monitoring the intervention area as well. For this purpose, we used the Pysense sensor shield from Pycom (Figure 5.2a) as an environmental sensor data collector. This device includes multiple sensors namely, ambient light, bio-metric pressure, temperature, humidity, dew point, and altitude. The role of the Pysense is to collect periodically (every 1 min) environmental data and transmit them automatically to the CCR and the OGC.

The alert message transmitted by the Pytrack and the data transmitted by the Pysense passes by an intermediate LoRaWan gateway before reaching the centralized control room and the on-ground chief. Numerous Indoor and outdoor gateways are available [Network,]. In our prototype, we used the 1Gate LoRaWan gateway [1Gate,] shown in Figure 5.3. Finally, for the network server, we have used the compact server for private LoRaWAN networks developed by Gotthardp [Gotthardp,].

Figure 5.3: The LoRaWan gateway

To summarize it all, Figure 5.4 illustrates the proposed system. Each firefighter is equipped with both a Pytrack and a Pysense sensor shield, in addition to a smart-watch. The Pytrack acquires the GPS location information, and immediately transmits them to the LoRaWan gateway when it receives an alert from the smart-watch indicating that the firefighter is at risk and needs extraction. The Pysense, regardless of the state of its

wearer, periodically collect and transmits environmental data to the gateway. This latter could be placed in a firetruck that is parked in the proximity of the intervention area in a location that provides the best line of sight with the firefighters. Moreover, in dense residual and industrial areas, pre-deployed gateways with optimally chosen locations that ensure maximum coverage could also be considered. The packets received by the gateway, are re-routed via the conventional WiFi/GSM Protocols to a local Network Server that in turn forwards the information to the concerned application. The end application is responsible for alerting the operators (in the centralized control room or the on-ground chief) when a firefighter is at risk. Finally, as explained in Section 5.3 the communication route is encrypted all the way up to the application via a network and an app session key, preventing any possible attack aiming to sabotage the ongoing operation.

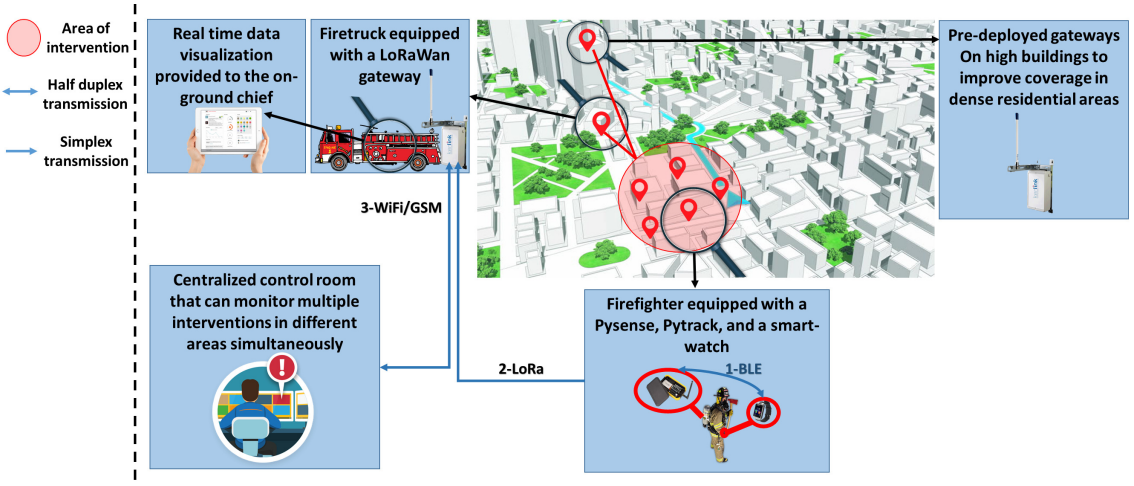


Figure 5.4: The proposed remote monitoring system

5.5/ Software Implementation for Emergency Detection

This section first introduces the Early Warning Score (EWS) system used to assess the smart-watch pulse samples, then describes the emergency detection algorithms that utilize the EWS system and are implemented for emergency detection on the smart-watch.

5.5.1/ Early Warning Score System

The use of an EWS system has long been enabled by acute care teams to allow a more auspicious response and assessment of patients who are highly sick or injured. An EWS system is based on a simple scoring scheme that assigns a score to each acute patient-monitored physiological measurement [Azar et al., 2018]. The EWS is used in this work to assess the heart rate samples gathered from the wearable device. For each vital

sign, a normal healthy range is defined, and the score assigned reflects how extreme the measured value varies from the standard. The higher the score assigned, the more a collected value is outside the normal healthy range.

Physiological parameter	Score 3	2	1	0	1	2	3
Respiration rate (per minute)	≤8		9–11	12–20		21–24	≥25
SpO_2 Scale 1 (%)	≤91	92–93	94–95	≥96			
SpO_2 Scale 2 (%)	≤83	84–85	86–87	88–92 ≥93 on air	93–94 on oxygen	95–96 on oxygen	≥97 on oxygen
Air or oxygen?		Oxygen		Air			
Systolic blood pressure (mmHg)	≤90	91–100	101–110	111–219			≥220
Pulse (per minute)	≤40		41–50	51–90	91–110	111–130	≥131
Consciousness				Alert			CVPU
Temperature (°C)	≤35.0		35.1–36.0	36.1–38.0	38.1–39.0	≥39.1	

Figure 5.5: National Early Warning Score (NEWS)

In this work, the scoring template for the National EWS (NEWS) used in U.K [NEW,], illustrated in Figure 5.5, is used to assess the severity level of the pulse in order to detect an emergency situation.

5.5.2/ Algorithms Implementation

This section presents two algorithms to detect heart issues and a man-down state. Three kinds of sensors can be used for such applications using contemporary wearable devices, namely the pulse sensor, motion sensor (accelerometer), and step counter sensor.

The first algorithm implemented on the wearable device is a periodic algorithm that runs after each period of time t (Algorithm 5). The pulse and step counter sensors collect data on an ongoing basis and store the values in the memory. After each period p, the total number of steps taken between p and $p-1$ in addition to the average heart rate values are used to detect an abnormal situation. Referring to [STE,], the average number of steps that can be taken in one minute for a low-intensity activity such as walking 3 miles per hour is 100. Having done less than 100 steps in a period with a pulse score of 3 (Figure 5.5) can reflect the possibility of an unusual situation such as injury or heat stroke. If the user has not done an extensive activity and yet has a pulse score of 3, a notification with vibration will be displayed on the watch for a small amount of time to inform the user about his pulse rate. If he believes his pulse rate is normal and this is a false alarm, the user can dismiss the notification by just shaking his hand. If the user has not responded to the notification, an emergency message will be transmitted to the Pytrack.

Algorithm 5 Periodic heart rate monitoring

Require: t (period time in seconds)
1: **for** each period p **do**
2: $steps \leftarrow$ number of steps
3: $hr \leftarrow$ average h eart rate
4: **if** $steps < 100$ and $score(hr) = 3$ **then**
5: notifyUser()
6: **if** notification dismissed **then**
7: no action taken
8: **else**
9: send emergency alert
10: **end if**
11: **end if**
12: **end for**

In order to detect a man-down state, Algorithm 6 has been implemented to run continuously in the background. This algorithm uses the accelerometer sensor to detect if the watch wearer is not moving for more than β amount of time. A notification with vibration will be displayed on the watch. If the user does not respond, an emergency message will be sent.

In addition to Algorithms 5 and 6, The user has the choice of sending an emergency message manually using the watch in case of he felt at risk.

Algorithm 6 Man-down state detection using accelerometer sensor

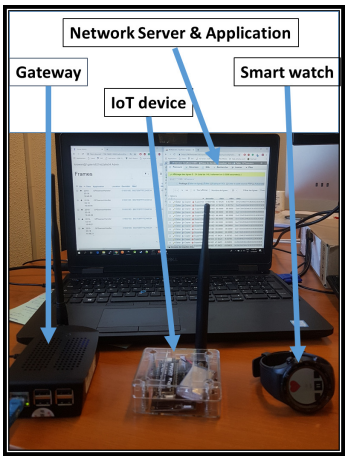
Require: TS (sampling period in ms)
α (motion threshold)
β (maximum inactive time in seconds)
$acc \leftarrow 0$ // Change rate
$last\_acc \leftarrow GRAVITY\_EARTH$
$current\_acc \leftarrow GRAVITY\_EARTH$
$inactive\_time \leftarrow 0$
for each period p of time TS **do**
$x, y, z \leftarrow get\_accelerometer\_data$
\\ calculating euclidean distance
$last\_acc \leftarrow current\_acc$
$current\_acc \leftarrow \sqrt{x^2 + y^2 + z^2}$
$delta \leftarrow current\_acc - last\_acc$
$acc \leftarrow acc \times 0.9 + delta$
if $|acc| < \alpha$ **then**
\\ not moving
$increment\_inactive\_time$
else
\\ movement detected
$inactive\_time \leftarrow 0$
end if

```
    if inactive_time > β then
      notifyUser()
      if notification dismissed then
        no action taken
      else
        send emergency alert
      end if
    end if
  end for
```

5.6/ Validation and Verification of the Proposed System

A system prototype consisting of a single smart-watch, a Pytrack, a Pysense and a gateway has been validated in an experiment that was carried out on the university campus where our laboratory is located. The used equipment and the setup are illustrated in Figure 5.6.

Figure 5.6: The used equipment



The wearer of the smart-watch (referred to as person A) has been asked to perform the following exercises: standing up, sitting down, running, walking, going upstairs, going downstairs, opening a box and taking items out of it while staying at his place, and finally, standing still and not moving for a certain duration. This test was carried out to check whether certain activities can trigger a false "Not Moving" alarm. None of the aforementioned activities has triggered an alarm as seen if Figure 5.7. The green diamond-shaped marker shows when the value of $Accel$ falls below the predefined threshold "trh" (alerted state), which means the watch wearer is not moving but is not considered as a "Man Down" since the duration in which $Accel$ is below trh is not greater than λ.

An alarm was triggered only when the smart-watch wearer was asked to stand still and did not move for a certain duration. In Figure 5.8 we can see that a "Man Down" alarm is triggered only when the "Alerted State" is maintained for a certain predefined duration. In

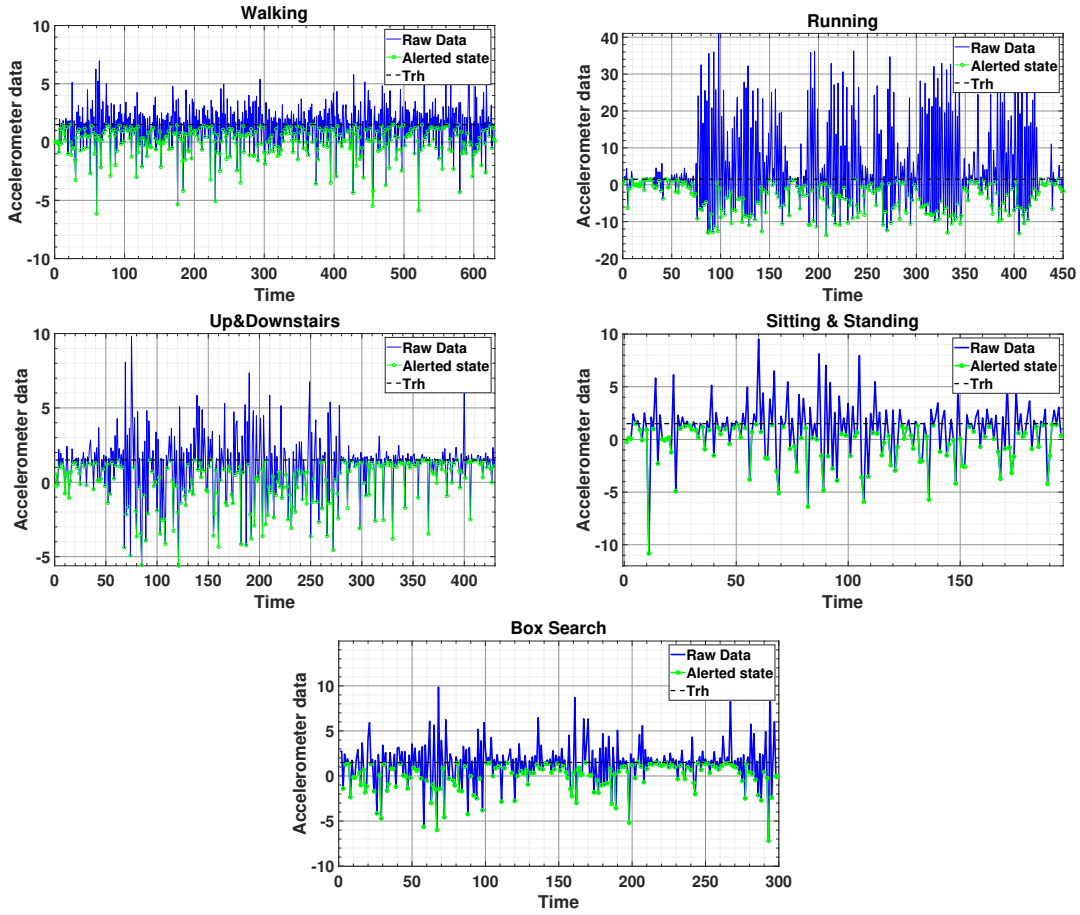
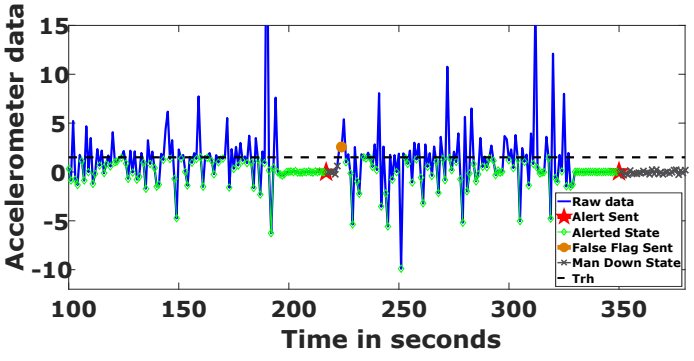


Figure 5.7: Accelerometer data for different movements

our application, for the sake of testing, we fixed this duration to 20 seconds. In Figure 5.8 we can see how an alarm is sent (star-shaped red marker) to the Pytrack when the watch-wearer did not move for 20 successive seconds, thus declaring a “Man Down” situation. Moreover, after the first alarm was sent, the person A was asked to move immediately. As a consequence, the smart-watch transmitted a false alarm to the Pytrack (represented by a circle-shaped orange marker). In contrast, when the second alarm was triggered, the person A was asked to remain immobile. Thus, no false alarm was transmitted afterward.

Figure 5.8: Accelerometer data



A second test is conducted to verify whether the smart-watch would send a “Heart Problem” alarm to the Pytrack in case of abnormal heart rate and how would the activity of the person A affects the decision of transmitting or not an alarm. Figure 5.9 illustrates abnormal heartbeat detection. It can be seen that the average heart rate rises and can bypass 100 bpm, which is quite normal when the amount of steps taken is big. However, if the score provided by Algorithm 5 is 3 and the amount of steps taken is small, this may be an unusual behavior such as the red dots shown in Figure 5.9 at the end of the graph. These dots show that a high heart rate is detected while an intense activity is not performed by the user. In addition, if the heart-rate falls below “40”, a minimum threshold that we agreed on with the firefighters, an alert is automatically transmitted.

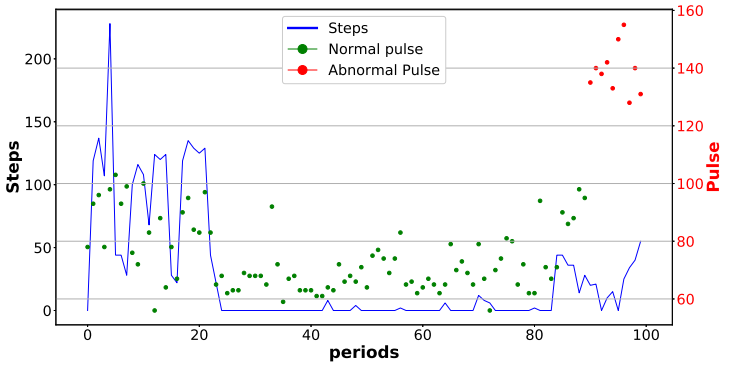
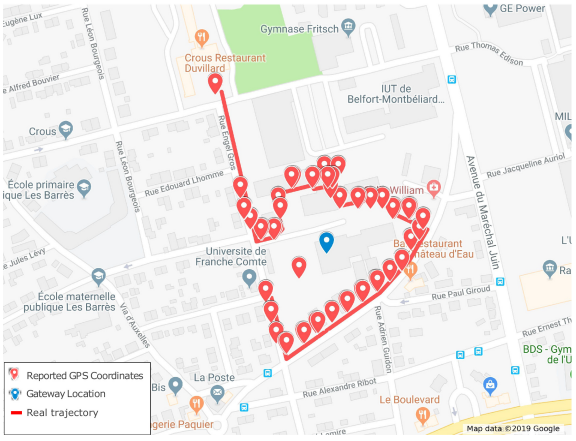


Figure 5.9: Abnormal heart beats detection

The two previously conducted tests demonstrated that our proposed system is able to detect heart problems or a “Man Down” while preventing false alarms. Moreover, we verified the interconnection between the smart-watch and the Pytrack, since we have successfully transmitted alert messages from the former to the latter. The final step is to test the feasibility of transmitting the alert messages from the Pytrack and the environmental data packets from the Pysense to the gateway. In order to do that we equipped person A with a Pysense and a Pytrack sensor shield and we asked him to go around the campus.
NB: for the sake of testing the Pytrack was programmed to transmit its location every 10 seconds. The data packets received by the gateway were visualized using a simple web application that we have developed for this purpose. Figure 5.10 shows the reported locations of person A, and Figure 5.11 shows a screen-shot of the web application.
NB: Figure 5.11 shows the reported locations on the map in addition to pre-filled locations used for API testing purposes.

The developed web application shows, the location of the firefighters on the map in case of emergency. It also shows underneath the map a list of the received alerts. The user could also chose a matriculation number belonging to a specific firefighter (Figure 5.11) to show historic environmental data (Figure 5.12).

Figure 5.10: Trajectory via reported coordinates



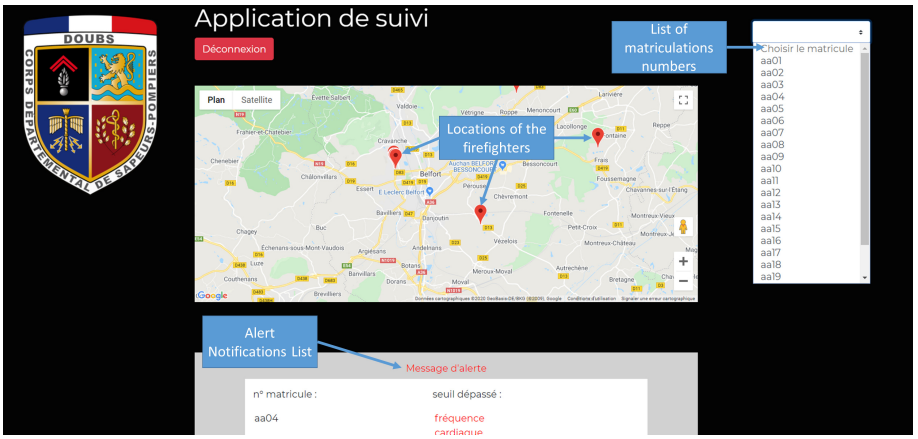


Figure 5.11: Web application screenshot 1

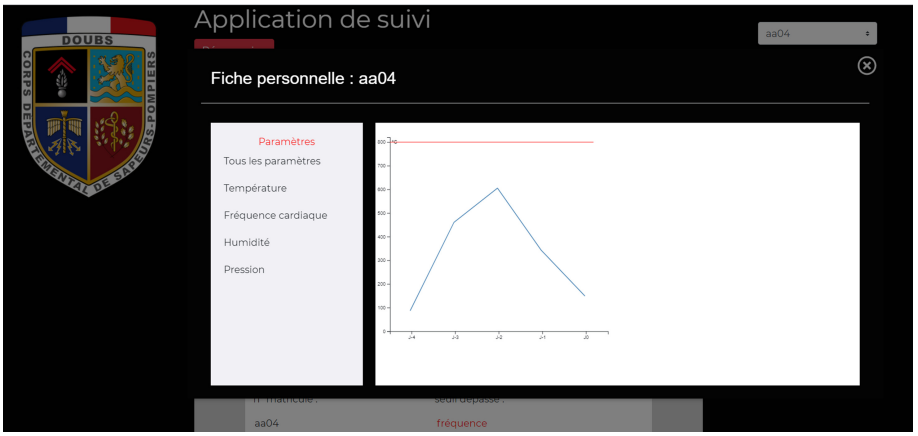


Figure 5.12: Web application screenshot 2

5.7/ Energy Consumption: From Simulation to Real Implementation

The energy management models described in Chapters chapters 2 to 4 could be implemented on the Pysense sensor shield of our previously proposed remote monitoring system. The simulations that have been conducted on real data-sets in the previous chapters have demonstrated that these proposed energy models could extend significantly the lifetime of the network. In this section, we put this into test, we implement the proposed algorithms on real Pysense sensor devices and we measure the energy consumption of each activity (transmission, sensing, processing, idle) independently, in addition to the overall energy consumption of each algorithm.

In order to do so, we have used the USB current tester “UM24C”. This latter could measure a current that ranges between 0 and 5000 Ampere. It also registers in real-time the current variation and automatically calculates the consumed current in mAh and the consumed energy in mWh. All this information is transmitted via Bluetooth to a laptop and a mobile device and are visualized with the help of an application. Figure 5.13 shows the UM24C and both the mobile and desktop applications.

Figure 5.13: Energy measurement device

At first, when we started developing our algorithms we were considering a remote monitoring system that is based on WiFi instead of LoRa, This is why we have used an energy model simulator (Section 2.3, Chapter 2) that considered a sensor equiped with a WiFi radio model. However, after digging upon better solutions we have discovered that LoRa would be a better choice for our project. Therefore, in addition to measuring the energy consumption of the devices, in this section, we also aim to verify if these algorithms are efficient on LoRa-based devices as they are on WiFi-based ones. Two of our proposed algorithms focused mainly on how to reduce transmitted packets and LoRa radio model

is known to consume a lot less than a WiFi radio model. Therefore, a question that we aim to answer in this section is, could the energy consumed by processing offset the energy preserved by reducing transmission which renders our proposal non-efficient for LoRa-based devices?

5.7.1/ WiFi-Based Devices

We first start by looking at how efficient are the FTDTR and ASTR algorithms on WiFi-based devices. We discuss separately the STCSTA algorithm since it falls rather in the sleep/wake up energy management categories rather than the data-driven one. In order to measure the energy consumption of each activity independently we adopted the following methodology:

We program the sensor to perform a cycle where it wakes up every one minute, performs a specific task (activity) before going back to sleep. The cycle is repeated for at least 1 to 5 hours. We then extract the collected real-time current data using the "UM24C" current tester, and we compute the energy consumption of this activity by averaging the peak in current consumption during each wake-up. Knowing that during deep sleep the sensor consumes between 2 and 8×10^{-6}A, we ignored these values and considered that each peak starts when the current surpasses 8×10^{-6}A and ends when the current goes back to less than 8×10^{-6}A. We then repeat this test 5 different times, and finally, we average the averaged values of the five tests and we consider it as the last result. In this experiment, we were interested in studying the current consumed by the sensing, processing, transmission, Idle activities and the ASTR, FTDTR, and Naive (wake, collect, transmit, sleep) methods.

As a first insight, let us look at Figure 5.14 that shows a comparison of one of the many peaks in current consumption between the different activities and algorithms. The shortest peaks are for Idle, sensing, and processing, and the longest peaks are for transmission and the Naive algorithm. Finally, FTDTR and ASTR have an intermediate, slightly long peak compared to the shortest and the longest. Judging by these results, we can assume for now that WiFi transmission has indeed a great effect on the overall energy consumption. Next, We will explain how the averaged current consumption for each different activity, algorithms is computed.

First, "E_{Idle}" is measured, which is how much current the sensor consumes by just waking up. The averages results using the previously described methodology shows that the sensor consumes on average at each wake-up (peak) while doing nothing else $0.6388A$. Then, the sensing activity "$E'_{Sensing}$" is measured, this time the task assigned to the sensor was to wake-up, collect a temperature measurement and then go back to sleep. To compute the exact current consumption of the sensing activity only, E_{Idle} is subtracted

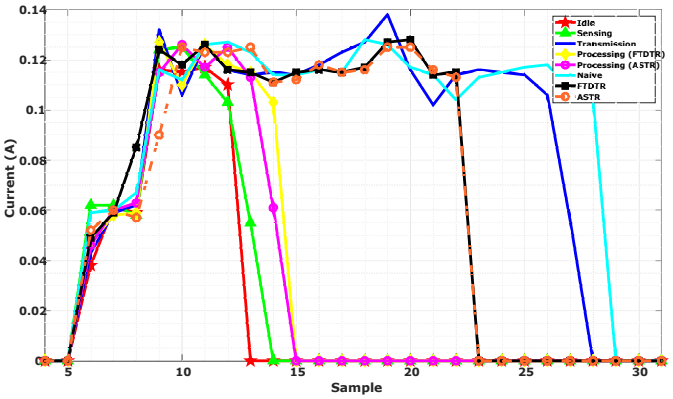


Figure 5.14: Peak current consumption comparison

from $E'_{Sensing}$ as shown in Equation 5.1. This is because the sensor needs to wake-up to perform sensing, thus we need to eliminate the current consumed by waking up in order to extract the one consumed by sensing only. The results show that $E_{Sensing}$ is on average $0.07425A$.

$$E_{Sensing} = E'_{Sensing} - E_{Idle} \tag{5.1}$$

To compute the current consumed by transmission ($E'_{Transmission}$) we did the same as we did for sensing, but instead of waking up and collecting a measurement, the sensor wakes up and transmits a random temperature measurement without collecting anything. To extract the energy consumed by transmission only, E_{Idle} is also subtracted from $E'_{Transmission}$ (Equation (5.2)). The results show that $E_{Transmission}$ is on average $1.751A$.

$$E_{Transmission} = E'_{Transmission} - E_{Idle} \tag{5.2}$$

In order to compute $E_{P_{FTDTR}}$, the current consumed while executing the FTDTR algorithm, we implemented the FTDTR algorithm on the sensor and we measured its current consumption at every peak E'_{FTDTR}. However, we eliminated transmissions but kept the sensing activity since the algorithm relies on collected measurements to function correctly. Therefore, $E_{P_{FTDTR}}$ is calculated according to the Equation (5.3). The results shows that $E_{P_{FTDTR}}$ is on average $0.1474A$.

$$E_{P_{FTDTR}} = E'_{FTDTR} - E'_{Sensing} \tag{5.3}$$

Last but not least, to compute the $E_{P_{ASTR}}$, the current consumed while executing the ASTR algorithm, we implemented the ASTR algorithm on the sensor and we measured its current consumption at every peak E'_{ASTR}. However, we eliminated transmissions, and de-

spite the calculation of a new sampling rate every round, we did not change it we kept it static. This would allow us to calculate $E_{P_{ASTR}}$ by simply using Equation (5.4). The results show that $E_{P_{ASTR}}$ is on average $0.115A$.

$$E_{P_{ASTR}} = E'_{ASTR} - E'_{Sensing} \tag{5.4}$$

As for the current consumed at every peak by ASTR, FTDTR, and the Naive method, We simply implemented these algorithms on the sensor let them operate 5 different times for at least 5h and averaged the peak current consumption for the 5 tries. The results shows that E_{Naive}, E_{FTDTR}, and E_{ASTR} are 2.489, 0.4832, 0.4888 respectively.

Figure 5.15 shows a comparison of the average current consumption of the sensing, transmission and processing activities, in addition to the current consumption of FTDTR, ASTR, and the Naive algorithm.

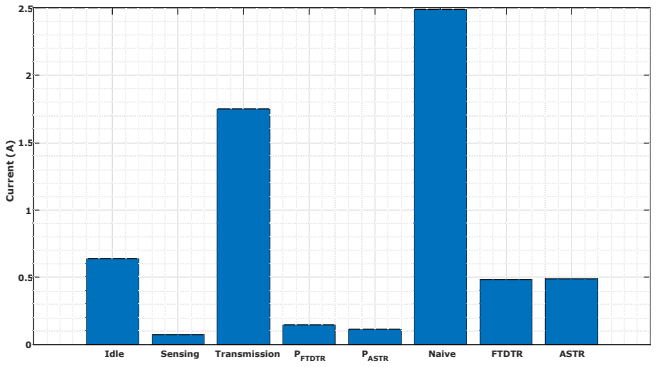


Figure 5.15: Current consumption comparison

Already we can notice that transmission consumes the most among other activities, and surprisingly in contrast to what has been assumed in the literature, sensing is the least consuming activity compared to idle that ranks second and the processing that ranks third. As for the algorithms, on average the naive method consumes the most, followed by ASTR and FTDTR. The effect in transmission reduction is already clear since both ASTR and FTDTR reduced drastically the average current consumption compared to the Naive method. However, we notice that ASTR despite integrating a mechanism that reduces sensing in addition to transmission by extending the deep-sleep duration, consumed more energy then FTDTR that only reduces transmission. This is because, we only computed the current consumption when the sensor is awake, and we did not consider the current consumed when the sensor is in deep sleep. This is what we are going to do next.

This time we did not compute the average current consumed at each peak (sensor is awake) as we did before, in contrast we computed the average of the overall consumed current, even when the sensor is in deep-sleep mode. Figure 5.16 shows the overall,

average current consumption of each algorithm/activity. Nothing much have changed for the activities except a slight proportional increase in the current consumption. However, ASTR now shows a much lower current consumption then FTDTR, this is mainly because the sensor implementing ASTR sleep-duration has increased on average. Thus, the sensor woke up less and consumed less E_{Idle}, $E_{Sensing}$, and $E_{Transmission}$.

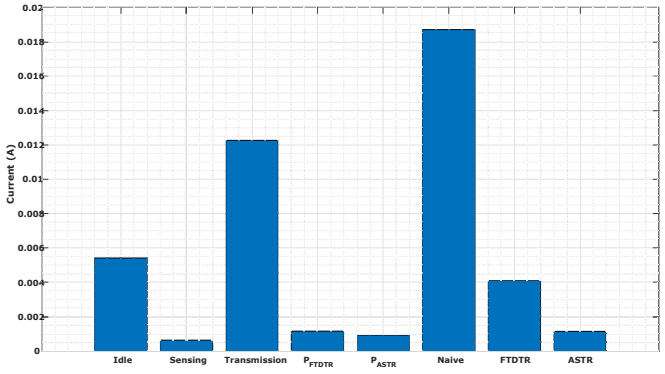


Figure 5.16: Overall average current consumption comparison

Knowing the average current consumption for each algorithm, and considering a Lipo battery with a capacity of 2000mAh, we can now compute an estimate of the operational lifetime of each algorithm in days. Figure 5.17 shows the estimated lifetime of a sensor implementing the Naive, FTDTR and ASTR methods. The Naive methods can last for only 4.456 days, FTDTR showed an increase of 357% (20.38 days) in operational lifetime, and ASTR showed an increase of 1535% (72.77 days). Therefore, the energy consumption results of the real implementation of our algorithms are aligned to what has been previously demonstrated in the previous chapters with the conducted simulations on real sensor data-sets.

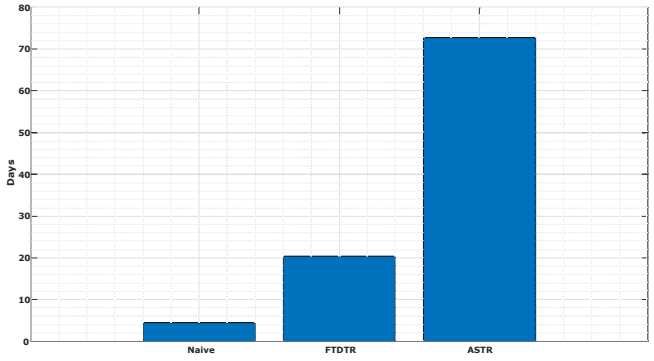


Figure 5.17: Lifetime estimate in days

5.7.2/ LoRa-Based Devices

In the previous subsection, we demonstrated that our proposed algorithm can greatly increase the operational lifetime of sensor nodes equipped with a WiFi radio module. In this section, we aim to study if this is still valid for sensor nodes equipped with a LoRa radio module since the transmission activity will be consuming much less with LoRa then with WiFi.

We adopt the same methodology to compute the peaks in current consumption, the average peak current consumption, the average overall consumption and the average operational lifetime in days. Figure 5.18 shows a comparison of one of the many peaks in current consumption between the different activities and algorithms. The first thing we notice is that the transmission peak compared to Figure 5.14 is shorter, which means it consumes less current. Next, we will see how this could affect the efficiency of FTDTR and ASTR.

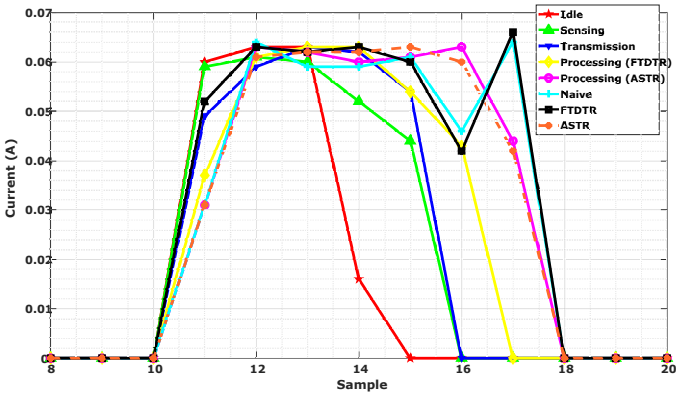


Figure 5.18: Peak current consumption comparison (LoRa)

Figure 5.19 shows a comparison of the average peak current consumption of the sensing, transmission and processing activities, in addition to the current consumption of FTDTR, ASTR, and the Naive algorithm. The first thing we notice that compared to Figure 5.15, the current consumption is reduced for all activities and algorithms, since the WiFi radio model was turned off, and only the LoRa radio model was kept on. Second, we notice that the current consumed by transmission has been drastically reduced compared to Figure 5.15 and to the other activities such as sensing and Idle. Finally, The difference between the current consumed by the Naive method and FTDTR has been drastically decreased, since transmission does not consume that much anymore. Moreover, ASTR is shown to consume more than the Naive method when the sensor is on.

Figure 5.20 shows the overall, average current consumption of each algorithm/activity. When we take the current consumption while in deep-sleep into consideration, ASTR is the least consuming. Still, the difference between the current consumed by FTDTR,

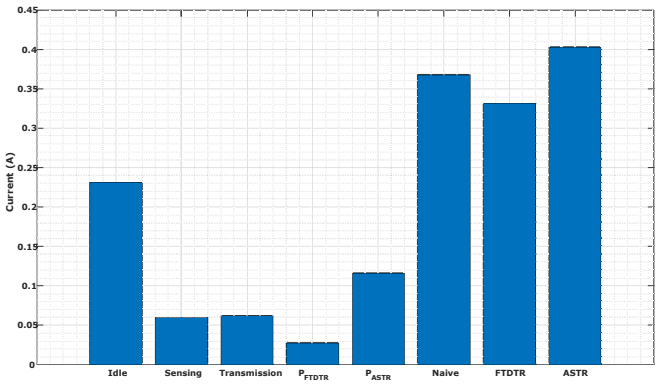


Figure 5.19: Peak current consumption comparison (LoRa)

ASTR, and Naive has been reduced compared to Figure 5.16.

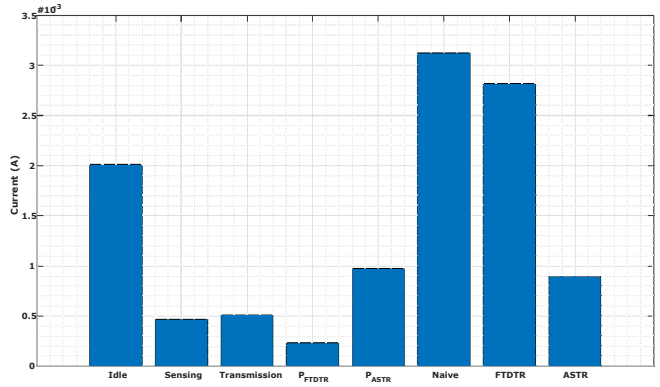


Figure 5.20: Overall, average current consumption comparison (LoRa)

Finally, Figure 5.21 shows the estimated lifetime of a sensor implementing the Naive, FTDTR and ASTR methods. The Naive methods can last for 26.68 days, FTDTR showed an increase of 10.79% (29.56 days) in operational lifetime instead of 357% previously, and ASTR showed an increase of 247% (92.74 days) instead of 1535% previously. We can conclude that our proposed data data-driven energy management algorithms are still efficient for LoRa-based sensors, but not as much as for WiFi-based sensors.

5.7.3/ STCSTA Energy Consumption

STCSTA is not a distributed algorithm like FTDTR and ASTR, thus, we cannot perform unitary tests on a single node as we did previously to study its energy efficiency. In order to study the efficiency of STCSTA, we deployed a sensor network in a $65m^2$ apartment as shown in Figure 5.22. The network consisted of 8 sensors distributed on 6 rooms as follows. Sensor 6 was placed in the first bedroom, sensor 5&4 in the living room, sensor

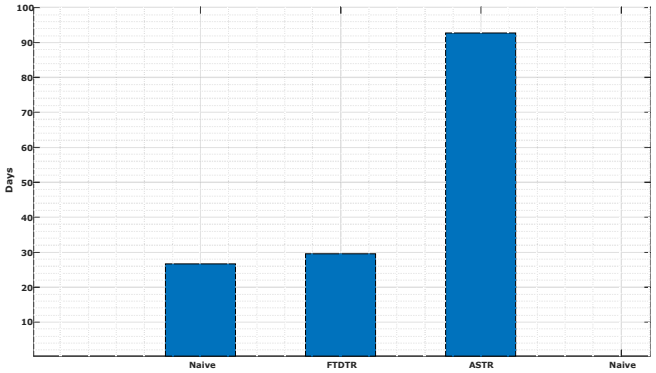


Figure 5.21: Lifetime estimation in days (LoRa)

7 in the entrance, sensor 3 in the bathroom, sensor 8 in the kitchen, and finally sensors 1&2 were placed in the second bedroom. All of the 8 sensors are considered to be in the same cluster, and the cluster-head (sensor 0) is a LoRaWan gateway. The gateway, through a network server, forwards all the received sensor data to a local database on a laptop PC. Since it would be difficult to monitor the energy consumption of each individual sensor, we adopted an approach that combines simulation with the real implementation. This approach will be explained step by step in the next subsections.

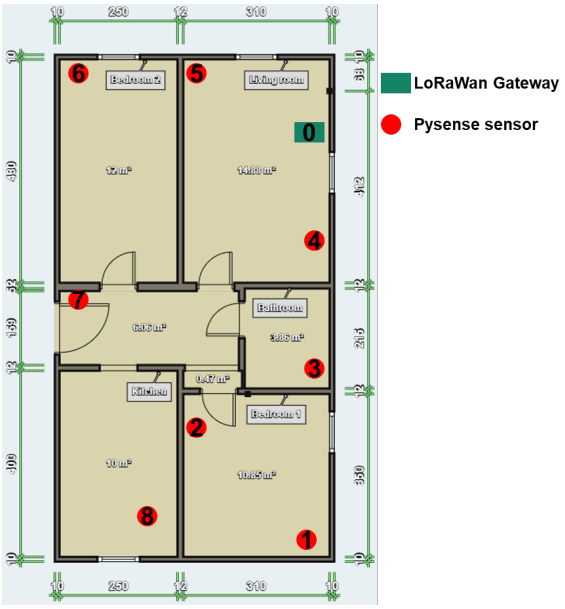


Figure 5.22: The sensor network geographical distribution

5.7.4/ COLLECTING AND PRE-PROCESSING THE DATA

The sensors were kept operating for approximately 42 hours, every one minute, each one of them wakes-up from deep-sleep to collect and transmit temperature, humidity, pres-

sure, dew point, and light data. Due to some communication failures caused by downed internet connection and since sensor 3 that is placed in the bathroom kept crashing for no reason, we had some missing data in the final collected data-set. Moreover, the collection times of the 8 senors were not consistent. For instance, the CH would receive a packet from sensor 1 at time 00:00:20 but from sensor 2 at 00:00:31 and from sensor 3 at 00:00:45, etc. Before moving to analyze the energy consumption, the missing data should be filled and the timing should be made consistent for all sensors.

Consistency of timing: in order to do that, using the python's pandas library we first separated the data set into 8 dataframes, each containing only the reported values of one unique sensor. Then we used the "resample" method provided by pandas for frequency conversion and resampling of time series. After doing that each one of the 8 time series stored in separate dataframes are now time consistent.

Filling missing data: on top of the already missing data, the resampling of the time series has created more gaps. In order to fill these gaps, we used the linear interpolation method.

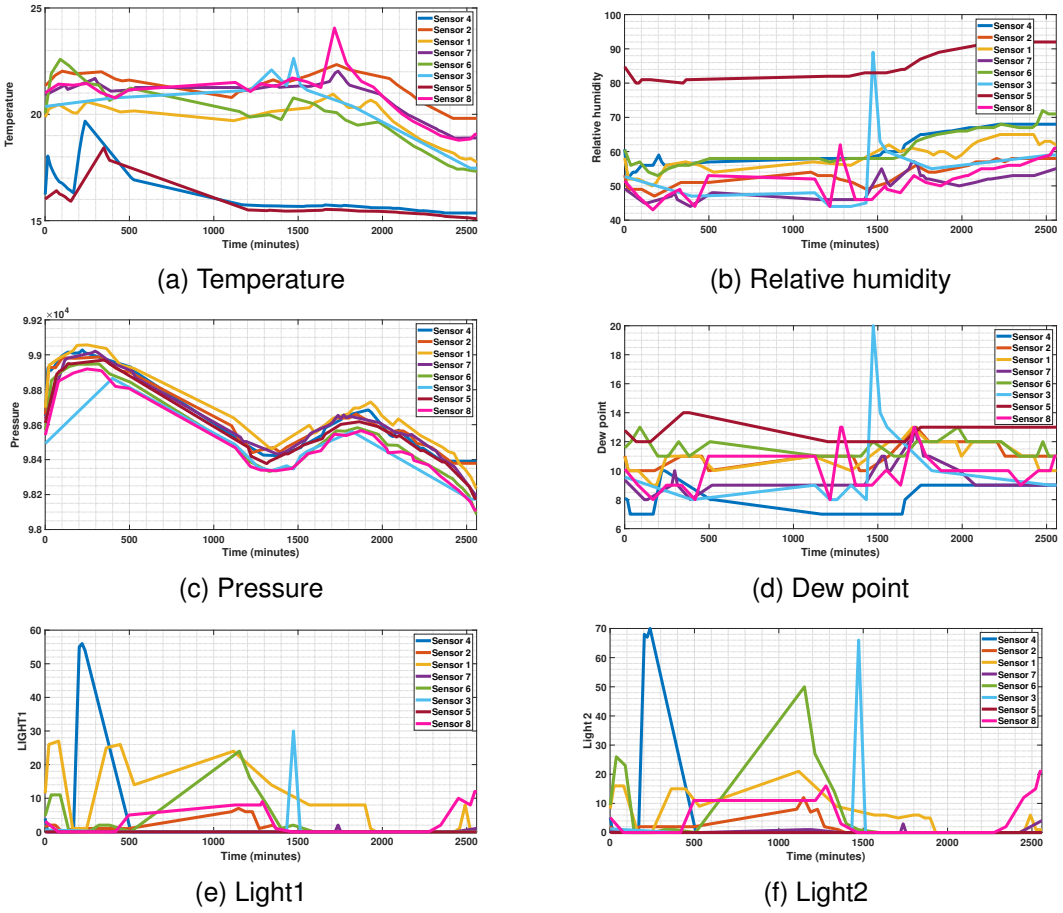
Now after the data is resampled and complete we can proceed to simulate the STCSTA algorithm using our custom-built Matlab simulator (mentioned in Section 4.4 of Chapter 4). Figure 5.23 shows the pre-processed data of all sensors for each environmental feature.

NB: Pysense has a dual light sensor that provides outputs for external light levels in lux. This is why we have two different light time series. Here are a couple of things we have noticed too. For **temperature**, sensors 4&5 have low temperature values, since when we are not using the living room, we turn off the heater. For **humidity and dew point** the peak that we see in figures 5.23b and 5.23d for sensor 3, is because of the steam produced when taking a shower.

5.7.4.1/ LIFETIME ESTIMATION

STCSTA works on homogeneous sensor data, however, the time-series data provided by the Pysense sensor nodes are heterogeneous and multi-feature. Therefore, a separate instance of the Matlab simulator is run for every environmental attribute separately, as if we're considering that each sensor is collecting and transmitting one feature only. The simulator will then give for every attribute and sensor, the percentage of transmission (wake-up) reduction.

For instance, considering the temperature attribute, let's assume the STCSTA Matlab simulator returned that sensor "x" (x∈1,8) has reduced its transmission rate by 50%. This means the sensor woke up 50% fewer times than the Naive approach. Since, we already have the time series of current measurement produced by the Naive approach for



(a) Temperature

(b) Relative humidity

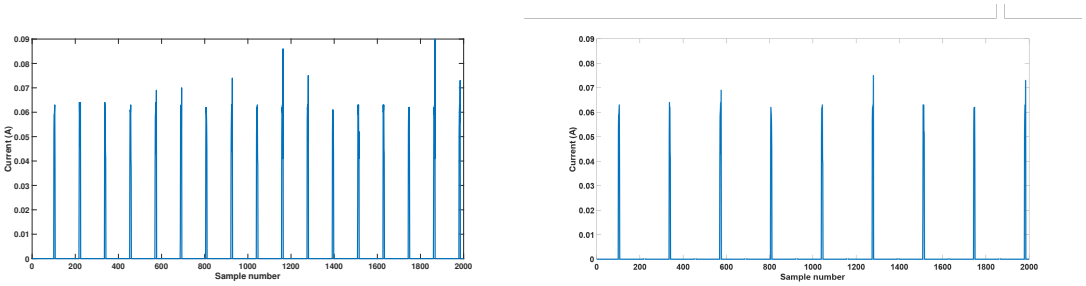
(c) Pressure

(d) Dew point

(e) Light1

(f) Light2

Figure 5.23: Visualisation of the collected data

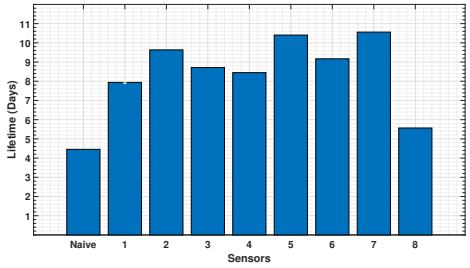


(a) 100% Transmission rate

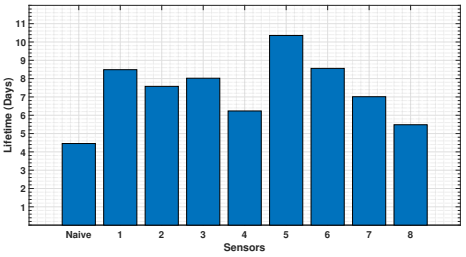
(b) 50% transmission rate

Figure 5.24: Current consumption times series comparison

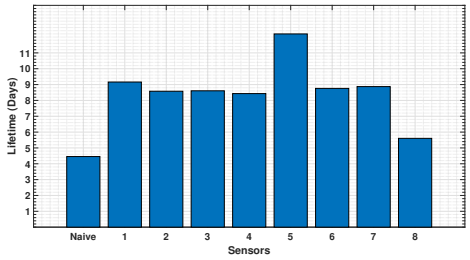
both LoRa and WiFi (sections 5.7.1 and 5.7.2), all we have to do is to remove 50% of the wake-up current peaks and replace their values by the average deep-sleep current consumption. Figure 5.24a shows a portion of the current measurement time series of the naive approach. Figure 5.24b shows how will this time series look like when 50% of the peaks are removed. We will do this to each sensor according to the reduction values returned by STCSTA, and then we will calculate the average current consumption value to estimate their lifetime.



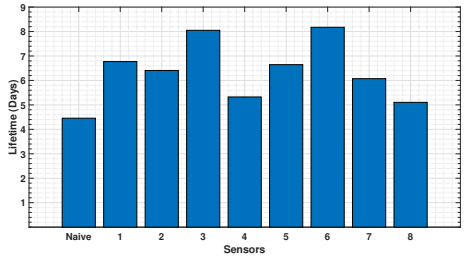
(a) Temperature



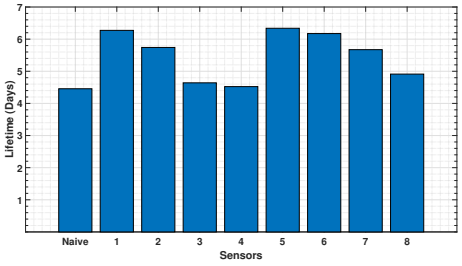
(b) Relative humidity



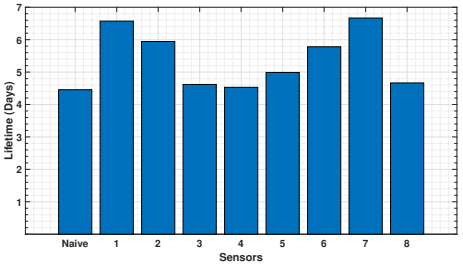
(c) Pressure



(d) Dew point



(e) Light1



(f) Light2

Figure 5.25: Increased lifetime in days of WiFi-based sensor devices compared to the naive approach

	Sensor 1	Sensor 2	Sensor 3	Sensor 4	Sensor 5	Sensor 6	Sensor 7	Sensor 8
Temperature	44	54	50	48	57	52	58	20
Relative humidity	48	42	45	29	58	48	34	19
Pressure	52	48	49	48	64	50	50	21
Dew Point	35	31	45	16	16	46	27	13
Light1	30	23	4	2	30	28	22	10
Light2	32	26	5	2	11	23	33	5

Table 5.1: Suppression ratio of all sensors for each environmental variable

Table 5.1 shows the suppression ration (SR) of each sensor node for all the environmental data returned by the simulator. The suppression ratio is the percentage of reduction in the number of wake-ups. The lighter is a table cell color the higher is SR. We notice that the best SRs are for temperature, humidity and pressure, since they showed a high degree of correlation (figures 5.23a to 5.23c). Sensor 8 has a low SR, although it shows a good correlation with other sensors. If we go back to Section 4.3.2, we explained that sensors

unit = %	Sensor 1	Sensor 2	Sensor 3	Sensor 4	Sensor 5	Sensor 6	Sensor 7	Sensor 8
Temperature	77	116	95	88	132	105	137	25
Relative Humidity	90	70	70	39	130	90	56	23
Pressure	105	90	90	88	174	90	90	20
Dew Point	50	44	80	19	48	80	36	14
Light1	40	30	4	1	41	8	28	10
Light2	40	30	3	1	11	29	49	4

Table 5.2: Percentage of Increase in Lifetime of all LoRa-based sensors for each environmental variable compared to the naive approach

that have a higher amount of matches they slightly reduce their sampling rate to allow the matching sensors to reduce it more. This could possibly explain why sensor 8 has a low SR while he correlates well with other sensors in the network.

As demonstrated previously in Section 5.7.1, with the naive approach a WiFi-based sensor node can last for 4.45 days approximately. Figure 5.25 shows the estimated lifetime of each WiFi-based sensor node and for every environmental feature. Judging by the obtained results, it is clear that STCSTA can significantly extend the lifetime of the network, even when there is a low correlation as in the case of Light data. Table 5.2 shows the percentage of increase in the operational lifetime of each LoRa-based sensor node for all environmental features compared to the naive approach that enables a sensor to operate for 26.68 days only. For instance, with pressure data, sensor 5 lifetime is increased by 174% compared to the naive approach, which means it is estimated to last for 46.42 days instead of 26.68. Looking at the results provided in Table 5.2, we can conclude that also for LoRa-based devices the network lifetime is significantly improved compared to the naive approach.

NB: The values are colored in shades of red and orange. Starting by the darkest shade of red for low percentages up to the highest percentages represented by the lightest shade of orange.

5.8/ CONCLUSION

In this chapter, we proposed a monitoring LoRaWan based system to improve the safety of firefighters. It uses a smart-watch to collect heart rate measurements and to detect mobility, and it reports any abnormal heart rate or immobility to a centralized control room and the on-ground chief. In addition to emergency detection, the system includes environmental sensors that enable real-time monitoring of the intervention area. A first prototype that aims to verify the intercommunication between the different components of the system was tested. The obtained results were promising and demonstrated its feasibility.

Moreover, we studied the possibility of implementing our proposed energy management algorithms on the sensor devices composing the LoRaWan system. Through real experimentation and real measurement of the current consumption, we demonstrated that our proposals could indeed significantly increase the lifetime of the LoRa-based sensor devices.

II

MACHINE LEARNING FOR DECISION MAKING

6

Comparing Recent Clustering Methods for IoT Data Management

The scheduling schemes for wireless sensor networks including our proposal in Chapter 4 relies on the presence of “good” clusters, where the sensor nodes that correlates the most are grouped into the same cluster instead of being partitioned on multiple ones. Otherwise, the cluster-head will will not be very efficient in detecting correlation and increasing the sleep duration of redundant sensors. Moreover, clustering is a great tool for processing and analysing the data received on the sink and is very helpful for decision making. Therefore, In this chapter, we survey and compare popular and advanced clustering schemes for management of massive IoT data and provide a detailed analysis of their performance.

6.1/ Introduction

The deluge of data manipulated and transiting in complex systems such as wireless sensor networks (WSN) for the Internet of Things (IoT) recently transformed, in a timely fashion, the field of data management and the associated technologies. The need for new scalable data analytics techniques that can guarantee quality of service, reliability, robustness. and a large battery-operated components’ lifespan has accordingly become paramount. It has been observed that for many different settings, clustering is a tool of choice for categorizing and interpreting the data in very large databases when no supervision is possible due to scale and time constraints issues. Moreover, and perhaps even more importantly, clustering is also relevant from a technological viewpoint: As sensors-to-sink information transmission is often very expensive in term of energy, sensors can be grouped into clusters where redundant ones belonging to the same cluster could be

scheduled to sleep for longer duration whilst enforcing data quality constraint, thereby leading to systematic reduction of energy consumption as seen in Chapter 4. Another example where clustering can be extremely useful is at the sink level, where massive and heterogeneous data arrive from myriads of various locations in the network. Clustering may then play a major role in real-time screening, as in the case of setting up alerts for the presence of dangerous observations. Finally, clustering also takes place when an extremely large number of sensors are deployed over large remote areas, suffering from poor signal quality. One specific aspect of clustering in WSN is that data quality is often poor due to signal transmission problems or even due to the use of certain energy saving scheduling strategies [Farhat et al., 2017]. Clustering should therefore be tailored with these peculiaritie in mind. In summary, in the face of massive and heterogeneous data, clustering is often an essential tool at the various stages of sensor network data analytics.

MOTIVATION AND CONTRIBUTION

Literature on WSNs or the Internet of Things (IoT), where clustering is frequently mentioned as a subroutine in scalable routing algorithms, is filled with very interesting specialized contributions. Nevertheless, general overviews and comparisons appear to be relatively scarce. Worse, most works present the well-known K-means approach as the state-of-the-art method without any more careful assessment of the pitfalls associated with this method, e.g., the non-convexity of the cost function which makes it hard to ascertain practical optimality (theoretical optimality being precluded by NP-completeness) or the possible presence of non-spherical clusters, to name a few.

The objective of the present chapter is to review recent clustering techniques and provide a more rigorous account of when and for which task they might be useful in the context of WSNs and the IoT. Moreover, this work aims to demonstrate that a few clustering techniques, such as K-means, are not necessarily the magical solution for every clustering problem. Instead, choosing the most appropriate clustering technique(s) according to the given scale (sensor, aggregator, sink), the type of data collected, their heterogeneity, and the possible presence of noise is a difficult but essential problem which deserves proper consideration.

The contributions provided in this chapter are the following, all the main important techniques which have been devised in the literature, including the necessary details for their implementation in the network, are described. A thorough comparison between the main important techniques which have been devised in the literature, as well as their respective advantages and disadvantages, is provided. Numerical experiments illustrating the performance of the methods are presented. An example with real data for a gas leak detection sensor network is analyzed to show how these various clustering techniques can

respond in different ways to data received at the sink level. We demonstrate that choosing the correct method makes a difference via performance comparisons on a massive dataset.

This chapter is divided into several sections, the first one giving a relatively complete overview of clustering methods and algorithms that are relevant for application to IoT. Section 6.3 presents the various established approaches to clustering assessment, divided into two subcategories: those that assume known ideal clustering and those that do not assume such a prior ideal model. These clustering and assessment techniques are then tested through simulation experiments, but also through a case study of a WSN consisting of gas sensors with massive data-sets. This article ends with a conclusion presenting a list of future research directions for the analysis of massive data-sets in the context of WSNs and IoT.

6.2/ An Overview of Modern Clustering Methods Applicable to the IoT

6.2.1/ Introducing Clustering for IoT

The number of devices and objects that are connected to the Internet is increasing rapidly. These connected objects generate a Iot data, which can be analyzed to identify trends and information for various purposes. This is where clustering for IoT [Xie et al., 2014, Zhou et al., 2009, Saeedi Emadi et al., 2018, Donghua Pan et al., 2011, Doreswamy et al., 2014, Hu et al., 2015, Kung et al., 2009, Muniraju et al., 2017, Sohn et al., 2016, ElGammal et al., 2009, Wang et al., 2019b, Sakthidasan et al., 2018, Zhang et al., 2018, Amaxilatis et al., 2018] becomes highly demanded. The advantages that clustering provides are numerous, such as it enables the scalability of the IoT network and it reduces the routing overhead by managing the routing decisions on the elected Cluster-Heads (CHs) [Yang et al., 2017, LI et al., 2018]. Moreover, it helps saving communication bandwidth and drastically reduces the overhead for topology maintenance. In addition, only the CHs and the gateway will form the backbone of the network, resulting in a simplified topology, reduced overhead, flooding, and collision. The end devices' only task is to connect to the CHs and forward the data without being affected by changes at the inter CH tier. The aggregation of collected data on the CH reduces the number of exchanged packets. Finally, various management strategies such as scheduling that could be implemented on the CH level can help preserve energy resources and extend the lifetime of the network [Rajasegarar et al., 2006, Bakaraniya et al., 2013, Pavithra et al., 2015].

As can be seen, clustering can be used in a variety of ways to improve the quality and

operational safety of wireless sensor networks, while extending their lifespan. But the situation changes from one network to another, depending on its scope, the number of sensors considered, their resources, the quality of the data produced, etc. A single way of clustering objects cannot be appropriate for all these situations, nor for the very diverse reasons (aggregation, hop-by-hop routing, data clustering at sink level) requiring their implementation. Therefore, in the following, various recent clustering techniques and their evaluations will be recalled or introduced, all of which have a potential interest in the Internet of Things.

6.2.2/ Ellipsoid-shaped Clusters

6.2.2.1/ K-means

Generalities The K-means algorithm clusters the data by trying to separate individuals into groups of equal variance, thus minimizing inertia, or the sum of intra-cluster squares. Its typical use case is presented in Table 6.1, while pros and cons are detailed in Table 6.2. Given k initial centres, we want to partition $\Omega = \{x_1, ..., x_m\}$ into k disjoint subsets, trying to minimize the Euclidean distance between each x_i and its assigned center. In other words, we want to minimize the criterion:

$$\min \left\{ \sum_{i=1}^{k} \sum_{x_j \in C_i} \|x_j - c_i\|, (C_1, ..., C_k) \in P(\Omega) \right\}$$

where:

- $P(\Omega)$ is the set of possible subsets of Ω.
- c_i is the center of the C_i cluster.

The initial centres can be chosen in various ways: random, with K-means++ (see below), etc. And the algorithm iterates as follows:

- assign each x_i to its nearest center c_j
- recalculate each center as an average of the x_i closest to it.

Note that there are various tricks, based on triangular inequality for example, to speed up the process. On the other hand, if K-means generally converges quite well with Euclidean distance, this convergence is less assured with other distances. Finally, the K-means has a K-Mediane variant, based as its name suggests on the median. As a result, it is less sensitive to outliers, but slower (including sorting to obtain the median).

Examples of the application of K-means in the context of IoT can be found in [Mittal et al., 2010, Rajasegarar et al., 2006, Tan et al., 2008]. For instance k-means clustering is used for data irregularities detection and pattern matching in [Mittal et al., 2010], to extend the lifetime of the network in [Rajasegarar et al., 2006], and in [Tan et al., 2008] for anomaly detection.

6.2.2.2/ K-MEANS++

The choice of initial centres is crucial for K-means. It has a significant impact on the results. K-means++ is an initialization algorithm. It avoids choosing two initial centres too closely. Its principle is as follows:

- The initial centres are chosen randomly, but with a non-uniform probability law.
- The probability of choosing a point as the initial center is proportional to the square of the distance from that point to the already selected centers.

This can be written in algorithmic form as listed in Algorithm 7.

Algorithm 7 K-means++ initialization

Require: K: Number of clusters
Require: m elements x_i to be clustered
  $c_1 \leftarrow x_1$
  for $k = 1, ..., K-1$ **do**
    $s_0 \leftarrow x_0$
    for $i = 1, ..., m$ **do**
      $s_i = s_{i-1} + min(\|x_j - c_1\|^2, ..., \|x_j - c_i\|^2)$
    end for
    pick r randomly in $[s_1, s_m]$
    find l such that r belongs in $[s_l, s_{l+1}]$
    $c_j \leftarrow x_l$
  end for

As we can see, as an input, K-means++ receives the m elements x_i to cluster (without the centers). At the output, it produces the initial K centres for K-means. Among the advantages of the method, K-means++ offers convergence guarantees, and has a parallelized version. However, this technique requires finding the nearest center to each element for each use, and this explodes with the number of centers.

Examples of the application of K-means++ in the context of IoT can be found in [Yang et al., 2017, LI et al., 2018]. For instance, K-means++ clustering is used for finding the optimal path for hop-by-hop routing in [Yang et al., 2017], and for an energy efficient routing algorithm in [LI et al., 2018].

6.2.2.3/ K-MEDOIDS

The K-Medoids [Park et al., 2009] or PAM (Partitioning Around Medoids) is an adaptation of the K-means, seeking to minimize the distance between the points of clusters and their center, the difference being that the medoids are necessarily points of the set to be clustered (which is not necessarily the case of the K-means centers). Thus a medoid can be seen as the individual of a cluster whose average dissimilarity to the individuals of said cluster is minimal: it is to some extent the most central individual of the cluster.

The PAM algorithm is the most common realization of the K-Medoids, but other approaches exist in the literature (e.g., a method based on Voronoi iterations). PAM follows a gluttonous approach, which does not guarantee to find the optimum but is much faster than an exhaustive search. Its operation is as follows:

- Initialization: choose k from n points for medoids.
- Associate each point with its closest medoid.
- As long as the cost of configuration decreases:
  - For each medoid m and each non-medoid o:
    - Exchange m and o, associate each point with its closest medoid, and recalculate the cost (sum of the distances of the points to their medoids).
    - If the total cost of the configuration has increased in the previous step, cancel the exchange.

The complexity of this algorithm is in $O(k(n-k)^2)$, and can be improved to $O(n^2)$.

Examples of the apllication of K-Medoids in the context of IoT can be found in [Bakaraniya et al., 2013, Pavithra et al., 2015]. For example, in [Bakaraniya et al., 2013], K-Medoids clustering is used to improve the Low Energy Adaptive Clustering Hierarchy (LEACH) and extend the lifetime of the network. Similarly, In [Pavithra et al., 2015] the K-Medoids is used to improve the data routing and reduce the energy consumption.

6.2.2.4/ GAUSSIAN MIXTURE MODEL

One of the main concerns about *K-means* is its naive use of the mean value for the cluster center.

- We can, for example, have two circular clusters of the same centre and different radii, which K-means will not be able to capture.

- Similarly, if the clusters are better modelled using a ellipsoid shaped subset, K-means might not converge to the correct solution.

The Gaussian Mixture Model (GMM) [McLachlan et al., 2019] gives more flexibility to K-means, by assuming that each points in the observed sample has been drawn from a Gaussian distribution with parameters depending on the cluster it belongs to. As a result, we no longer assume circular clusters, but allow for elliptical shapes in 2D, 3D, etc. Each cluster will then be defined by the associated probability of belonging to it, its mean vector and Covariance matrix.

More precisely, GMM assumes that the points follow a distribution:

$$\sum_{k=1}^{K} \pi_k \mathcal{N}(\mu_k, \Sigma_k)$$

with $\mu_1, ..., \mu_K$ the cluster centers, $\pi_1, ..., \pi_K$ the probability weights, and $\Sigma_1, ..., \Sigma_K$ the variance-covariance matrices of each cluster. It is an unsupervised and parametric model, as we are looking for a distribution in the form of a Gaussian distribution mean. The parameters are optimized according to a maximum likelihood criterion to get as close as possible to the desired distribution. This procedure is most often done iteratively via the expectation-maximization (EM) algorithm [McLachlan et al., 2007], and its Kullback proximal generalisations [Chrétien et al., 2000], [Chrétien et al., 2008], Gauss-Seidel version [Celeux et al., 2001].

One of the main advantages of the Gaussian Mixture Model is that it has statistical underpinnings that allow for various penalisation allowing principled model order selection based on e.g. AIC or BIC [McLachlan et al., 2019] of ICL [Biernacki et al., 2003], or even sparsity inducing penalised version of the maximum likelihood approach [Chrétien et al., 2012].

The EM algorithm works as follows:

1. We select the number of clusters, and randomly initialize the parameters of the Gaussian distributions of each cluster.

2. Given these Gaussian distributions for each cluster, we calculate the probability that each point belongs to each cluster: the closer a point is to a Gaussian centre, the more likely it is to belong to the associated cluster (Expectation part).

3. Based on these probabilities, we re-estimate the Gaussians: we calculate a new set of parameters of Gaussian distributions, in order to maximize the probability of the points to be in the clusters (Maximization part). These new parameters are calculated using a weighted sum of the point positions, the weights being the probabilities that the points belong to this cluster.

4. We repeat this until the distributions no longer change, or almost: we evaluate the log-likelihood of the data to test convergence (this is the objective function to be increased).

Finally, let us note the following remarks: (1) K-means is a special case of GMM, i.e. with a constant covariance per component (GMM is much more flexible in terms of covariance than K-means); (2) The EM depends in particular on the number of times it is launched (100 is a good choice), and strongly from the initialization: we can randomly extract k observations of X as means of the initial components (k : number of Gaussians), or choose them by K-means++, which is preferable.

As far as we know, the GMM clustering technique has not yet been implemented in any IoT related application

6.2.3/ Density Based Clustering

6.2.3.1/ Mean-shift

Mean-shift [Comaniciu et al., 2002] is a clustering algorithm based on a sliding window (ball), which searches for dense areas of points by moving the centroids of the balls. We try to locate the center of each class, updating the candidates as an average of the points in the window. The windows are finally post-processed to eliminate overlaps.

This method works as follows:

1. For nucleus, we consider a sliding ball centered at a random point C and of radius r. Through a hill climbing approach, this nucleus is iteratively moved to an area of higher density, until convergence.
2. This is done by moving the centre of the ball towards the centre of gravity of the ball's points, which causes the ball to move to denser areas.
    - As the ball has been moved, there are potentially new points, and therefore a new center of gravity.
    - Otherwise, we end up converging.
3. Points 1 and 2 are operated with various randomly placed balls, or according to a grid adapted to the data. And when several balls overlap, the less dense one is removed.

Once convergence is established, the remaining balls are the clusters. Refer to Tables 6.1 and 6.2 for the context of use, as well as the advantages and disadvantages of this method.

Examples of the application of mean-shift in the context of IoT can be found in [Xie et al., 2014, Zhou et al., 2009]: in [Zhou et al., 2009], for instance, the mean-shift clustering is used for localization and tracking in IoT applications.

6.2.3.2/ DBSCAN

The Density-Based Spatial Clustering of Applications with Noise (DBSCAN [Ester et al., 1996]) is another density-based clustering: it looks for high-density areas and then extends clusters from them. Its general operating principle is as follows.

1. We start from a point P of the data, not yet visited.
   - If there are enough neighbors (min_samples-1) at ε from this point, clustering starts with P as the first (core) point of the cluster.
   - Otherwise, the point is labeled as noise (which can later integrate a cluster) and visited.
2. The points at distance ε of P integrate the cluster of P, then all the points at ε of these points, etc., until it is no longer possible to expand the cluster.
3. We start again with a point not visited, and this until the points are exhausted. At the end, any point will either be in a cluster or labeled as noise.

DBSCAN is therefore similar in its operation to the mean-shift, although it has some advantages, see Table 6.2. It has an optimized version called HDBSCAN*, which extends DBSCAN "by converting it into a hierarchical clustering algorithm, and then using a technique to extract a flat clustering based in the stability of clusters" [McInnes et al., 2017].

Examples of the application of DBSCAN in the context of IoT can be found in [Saeedi Emadi et al., 2018, Donghua Pan et al., 2011, Doreswamy et al., 2014]. In [Saeedi Emadi et al., 2018], the DBSCAN clustering algorithm is used for anomaly detection, while in [Donghua Pan et al., 2011], an improved version of DBSCAN is used for uncertain data clustering problems. Finally, DBSCAN clustering is used to detect faulty sensor data in [Doreswamy et al., 2014].

6.2.4/ Tree-Based Clustering

6.2.4.1/ Hierarchical Clustering

Hierarchical clustering algorithms are either top-down or bottum-up: we start from points seen as simple clusters, which we iteratively aggregate in pairs, until we reach a single

final cluster. This bottum-up approach is precisely what is called agglomerative hierarchical clustering, and this hierarchy of clusters can be represented as a dendrogram: the root is the final single cluster, the leafs being the data points. Unlike agglomerative version of the hierarchical clustering, the divisive starts with a single cluster encompassing all points, then iterates divisions until only clusters at one point are obtained.

The agglomerative algorithm can be summarized as follows.

1. Each data point is treated as a single cluster, and a distance between clusters is fixed - for example, the average linkage: the average distance between the points of two clusters.
2. At each iteration, the two clusters with the shortest distance are merged, and the height of the new node in the dendrogram is the similarity between said clusters.
3. Repeat 2 until you only get one cluster left.
4. If a predefined number of clusters is wanted, step number 2 is stopped when the number is reached.

Its divisive version can be easily deduced from this. Various distances can be used:

- **single linkage:** the distance between two clusters is the distance corresponding to the two most similar points.
- **complete linkage:** as above, but with the least similar points.
- **average linkage:** the distance between two clusters is defined as the average of distances between all pairs of points.
- **Ward:** minimization of the sum of the squares of distances within each cluster. It is an approach of the variance minimization type, therefore a kind of K-means coupled with a hierarchical agglomerative approach.

6.2.4.2/ BIRCH

Birch [Zhang et al., 1996] is an online learning algorithm, memory efficient, which can be seen as an alternative to Mini Batch K-means. It builds a tree, in which the centroids of the clusters are at the level of the leaves. These can either be the centroids of the final clustering, or they can be passed as an input to another clustering algorithm, such as agglomerating clustering.

The data tree consists of nodes, each node consisting of a number of subclusters. The maximum number of subclusters in a node is determined by the connection factor. Each

subcluster stores various information necessary for the inline process, such as the number of samples in that subcluster or the average of its points providing the centroid. In addition, each sub-cluster can also have a node as a child, if the sub-cluster is not in a leaf.

Every new individual is introduced to the root. It is merged with the nearest subcluster and the various information of this subcluster is updated, which is recursively made until it reaches a leaf.

The parameters of this method are as follows:

- the threshold: the radius of the sub-cluster obtained by merging a new individual and the nearest sub-cluster must be less than this threshold. If this is not the case, a new subcluster is initialized. A low value of this threshold therefore multiplies the number of node breakdowns, and thus the clusters.
- the branching factor: maximum number of subclusters in each node. If a new individual entering a node causes the number of subclusters to exceed this branching factor, then the node is split into 2, which distribute the subclusters. The parent subgroup of this node is deleted, and two new subclusters are added as parents of these two cut nodes.
- number of clusters after the last clustering step.

Note that, as far as we know, the Birch clustering technique has not yet been considered in any IoT related application.

6.2.5/ Other Methods

6.2.5.1/ Spectral Clustering

The Spectral clustering [Von Luxburg, 2007] applies K-means to a low dimensional immersion of the affinity matrix between samples, i.e. to the normalized Laplacian matrix

$$D^{-1/2}(D-A)D^{-1/2},$$

where D is the degree matrix of the graph whose adjacency matrix is the affinity A. This affinity matrix is constructed using:

- either a kernel function, for example the RBF $e^{-\gamma d(X,X)^2}$ or the heat kernel $e^{-\beta\frac{d(X,X)}{std(d)}}$, where d is the distance between individuals, while β and γ are hyperparameters to set up;

- or a connectivity matrix in the k-nearest neighbors;
- or, via the precomputed parameter, an affinity matrix provided by the user.

As other clusterings detailed here, pros and cons of this technique are provided in Tables 6.1 and 6.2.

A study of Laplacian eigenvector based clustering in the context of overlaping clusters is avialable in [Chretien et al., tted].

Examples of the application of Spectral clustering in the context of IoT can be found in [Hu et al., 2015, Kung et al., 2009, Muniraju et al., 2017]. For instance, in [Hu et al., 2015], the Spectral clustering is used to detect disconnected segments of the sensor network caused by depletion of battery or physical tempering of nodes. Conversely, in [Kung et al., 2009], Spectral clustering is used to detect and eliminate sensor nodes that are not working properly.

6.2.5.2/ AFFINITY PROPAGATION

Affinity propagation (AP [Frey et al., 2007]) is a clustering algorithm based on the notion of passing messages between data points. As with the K-Medoids, the AP is looking for models ("exemplars"), i.e. points in the dataset that could be good cluster representatives. The AP starts from a similarity matrix, and exchanges messages (real numbers) between the points to be clustered until quality clusters emerge naturally. During message exchange, the algorithm identifies exemplary points, or models, that are able to properly describe a cluster.

Let s be a similarity matrix for the points to be clustered, in which $s(i,i)$ designates the "preference" of the i entry, which means how likely the i element is to be found as a model. These $s(i,i)$ are typically initialized at the median of the similarities, knowing that initializing them at a value close to the smallest of the similarities will lead to fewer clusters (and vice versa for a value close to the maximum).

The algorithm alternates two message passing steps, updating the following two matrices:

- The R matrix of competence (responsibility), in which the element (i,k) quantifies how much x_k is justified as a model for x_i, compared to the other candidates.
- The availability matrix A, in which the element (i,k) represents how appropriate it would be for x_i to take x_k as a model, once all the other points have been considered as a model.

These two matrices, which can be seen as log-probability tables, are initialized with 0's. The following steps are then iterated:

- First, we circulate the skill updates: $r(i,k) \leftarrow s(i,k) - \max_{k' \neq k} \{a(i,k') + s(i,k')\}$.
- Then the availability is updated by:
  - $a(i,k) \leftarrow \min \min \left(0, r(k,k,k) + \sum_{i' \notin \{i,k\}} \max(0, r(i',k))\right)$ for $i \neq k$, and
  - $a(k,k) \leftarrow \sum_{i' \neq k} \max(0, r(i',k))$.

Iterations are performed until either the cluster boundaries no longer move, or after a predefined number of iterations. The models are extracted from the final matrices, as being the self-competent and self-available elements (i.e. $r(i,i) + a(i,i)) > 0$).

Examples of the application of Affinity propagation clustering in the context of IoT can be found in [Sohn et al., 2016, ElGammal et al., 2009, Wang et al., 2019b, Sakthidasan et al., 2018]. For instance, in [Sohn et al., 2016], AP is used to improve the LEACH protocol, while in [ElGammal et al., 2009, Wang et al., 2019b], improved versions of the AP for sensor data clustering are proposed. Finally, in [Sakthidasan et al., 2018], AP is used for optimal routing path selection.

6.2.5.3/ CONSTRAINED CLUSTERING

There is a separate branch of clustering that integrates the notion of constraints. Thus, in COP (COnstrained Pairwise) K-means [Wagstaff et al., 2001], it is specified that this and that individual must be linked, when that and that other must not, while MinSizeK-means forces a minimum size for clusters. Other semi-supervised clustering can be found in the literature, like Seeded-KMeans, Constrainted-KMeans, Pairwise constrained K-means (PCK-means), Metric K-means (MK-means), or Metric pairwise constrained K-means (MPCK-means), to name a few. Implementation of such active semi-supervised clustering algorithms can be found, *e.g.*, in [con,], while [Pedregosa et al., 2011b] contains the Python implementation of most usual clustering methods.

6.3/ CLUSTERING EVALUATION: STATE-OF-THE-ART

There are two different approaches validation of clustering results. In external validation, we have access to the real clusters, and we want to measure to what extent the considered algorithm is able to recover the ground truth. In internal validation, we do not have access to the ground truth, and an alternative hint about the relevance of the number and shape of the clusters is needed. External validation provides more relevant and meaningful measures for testing on simulated data, but the hypothesis of having access to the desired solution is obviously not realistic in practical implementations. Many libraries already implement most of these metrics, such as [Pedregosa et al., 2011b].

Table 6.1: Use case of clustering techniques.

Method	Parameters	Cluster size	Nb of clusters	Geometry
Affinity propagation	damping sample preference	distinct	large nb of clusters; does not scale up with the nb of individuals	non flat
Birch	numerous	large data sets	large nb of clusters	non flat
DBSCAN	neighborhood size	distinct	Very large set of individuals average number of clusters	non flat
GMM	nb of clusters	not scalable	not scalable	flat
Hierarchical clustering	nb of clusters link type, distance	distinct	many clusters many samples	hierarchical
K-means	nb of clusters	regular	not too many	flat
K-Medoids	nb of clusters	regular	not too many	flat
Mean-shift	bandwidth	distinct	many	non flat
Spectral	nb of clusters	distinct	small	non flat

6.3.1/ External Validation

6.3.1.1/ Homogeneity, Completeness, and V-measure

If we know the expected labels, we can define some intuitive measures based on an analysis of conditional entropy. Thus, Rosenberg and Hirschberg [Rosenberg et al., 2007] have defined the following two objectives, which are obviously desirable for any clustering:

- **Homogeneity::** each cluster contains only members of a single class.
- **Completeness::** all members of a given class are in the same cluster.

These two scores are between 0 and 1 (the higher the value, the better the clustering), and the *V-measure* is the harmonic mean of the latter two.

With more details, homogeneity and completeness are defined mathematically as follows:

$$h = 1 - \frac{H(C \mid K)}{H(C)}, \quad c = 1 - \frac{H(K \mid C)}{H(K)}$$

where:

Table 6.2: Pros and cons of clustering techniques

Methods	Advantages	Disadvantages
Affinity propagation	- Nb of clusters not required - One main parameter: the damping factor[1]	- Quadratic complexity in the nb of points - There may not be convergence
Birch	- Outlier deletion, data reduction - If large number of subclusters is desired	- Does not adapt very well to large data[2]
DBSCAN	- No need to provide a priori the nb of clusters - Useful for identifying outliers, noise - Can find clusters of arbitrary size and shape - Works well for clusters of homogeneous density	- Works less well than others when the clusters to be found have different densities[3] - Does not work well in very high dimension
GMM	- Intuitive algorithm associated with maximum likelihood maximisation - Soft clustering: each point belongs to all clusters, but with different probabilities - Allows an estimation of density - model based approach that comes with easy to implement information theoretic penalties for model selection (number of clusters, etc.)	- Not scalable
Hierarchical clustering	- Nb of clusters not required - The most appropriate cluster nb can be chosen afterwards - Choice of distance not critical - Works well when data is naturally hierarchical allows you to recover this hierarchy	- Its specificity for hierarchical data - High impact of outliers - Slow: $O(n^3)$
K-means	- Fast (linear complexity) - Easy to implement - Proven in many applications - Scalable: for a large number of individuals and a reasonable number of clusters[4]	- We have to choose the nb of clusters - Not very good, outside of spherical clusters - Consistency problem: different runs produce different clustering, due to its random initialization
K-Medoids	- Robust against noise and presence of outliers	- We have to choose the nb of clusters
Mean-shift	- No need to provide a priori the nb of clusters - The centres of the balls converge towards the places of highest density	- Window size can be difficult to fix
Spectral	- Useful when cluster structure is highly non-convex - Very effective with sparse affinity matrix	- Nb of clusters must be provided - Not recommended for large nb of clusters

*N*otes: [1] increasing it would tend to reduce the number of clusters; [2] it is generally preferable to use MiniBatchKMeans; [3] HDBSCAN* solves this issue; [4] via Mini Batch K-means.

- $H(C \mid K)$ is the conditional entropy of classes knowing clusters, i. e. $-\sum_{c=1}^{|C|}\sum_{k=1}^{|K|}\frac{n_{c,k}}{n}log\frac{n_{c,k}}{n_k}$,
- $H(K)$ is the entropy of the class, equal to $-\sum_{c=1}^{|C|}\frac{n_c}{n}log\frac{n_c}{n}$,

with n the number of individuals, n_c and n_k that of the c class and the k cluster, and $n_{c,k}$ the number of individuals in the c class assigned to the k cluster. And, for the completeness, the adaptation of the latter. Finally, the V-measurement is the harmonic mean of h and c, i. e. :

$$v = 2\frac{hc}{h+c}.$$

The following points can be noted. First of all, a permutation of the class or cluster labels will not change the value of these scores. Also, homogeneity and completeness are not symmetrical: changing true labels with predicted labels will move from one score to another (in other words, homogeneity_score(a, b) = completeness_score(b, a)). V-measure, for its part, is symmetrical; it is equivalent to mutual information, with the arithmetic mean as an aggregation function. This V-measure can be used to evaluate the agreement of two independent clustering on the same data set.

Among the advantages of these measures, we can mention the fact that a clustering with a bad V-measure (i.e., close to 0) can be analyzed qualitatively, in terms of homogeneity and completeness, to better interpret the type of error made. On the other hand, there is no hypothesis on the structure of clusters, so we can compare various clustering approaches (K-means vs. spectral...)

Conversely, these methods have some disadvantages. First of all, the actual labelling must be known to calculate these scores. Moreover, these metrics are not adjusted to account for chance: depending on the number of individuals, clusters and real classes, completely random labelling will not necessarily produce the same values of homogeneity, completeness, and therefore V-measurement. In particular, random labelling is not necessarily zero, especially for a large number of clusters. However, this problem can be ignored if the number of individuals exceeds 1,000 for a number of clusters of less than 10. If this is not the case, prefer the ARI recalled below.

6.3.1.2/ Adjusted Rand Index

The Rand Index (RI) calculates a similarity between two clusterings, by looking at each peer of individuals, and counting those that are or are not in the same cluster, depending on whether you are in actual or predicted clustering:

$$RI = \frac{a+b}{\binom{n}{2}}$$

where a is the number of points that are in the same cluster for both clusterings, b is for those that are in a different cluster for both clusterings, and n is the total number of samples. As such, the RI does not guarantee that random assignment will produce a value close to 0. This is why this raw index is "adjusted to account for chance", which gives the ARI (Adjusted Rand Index) score:

$$ARI = \frac{RI - Expected\_RI}{max(RI) - Expected\_RI}.$$

The ARI, which is symmetrical, measures the similarity, the consensus of two assignments, ignoring permutations and normalizing against what would have happened by chance. ARI score has various advantages:

- Random labelling leads to an ARI close to 0, regardless of the number of points and clusters, which is not the case for RI or V-measure.
- The ARI varies between -1 and 1 included. Negative values are for independent labeling, when similar clusterings have a positive ARI (1 for an exact match).
- No hypotesis on the structure of the clusters: we can therefore compare K-means to spectral clustering, leading a priori to very different structures.

Its main disadvantage is that it assumes that the actual expected labelling is known. To have a full understanding of the values returned by the ARI, it can be noted that a label placing all individuals in the same clusters is complete, but not always pure, and is therefore penalized. In addition, the ARI is symmetrical, so labelling leading to pure clustering with members from the same classes, but with unnecessary class splits, is also penalized. Finally, if the class members are completely separated in different clusters, then the assignment is totally incomplete, and therefore the ARI is very low.

6.3.1.3/ FOWLKES-MALLOWS SCORE

When we know the classification of individuals whose clustering is done, we can calculate the Fowlkes-Mallows FMI score [Fowlkes et al., 1983] as the geometric mean of the precision and recall per pair:

$$FMI = \frac{TP}{\sqrt{(TP + FP)(TP + FP)(TP + FN)}}$$

where:

- TP is the number of true positives: the number of pairs of points in the same cluster in the real and predicted labelling,

- FP is the number of false positives: pairs with the same real labelling, but in different predicted clusters,
- FN is the number of false negatives: pairs in the same predicted clusters, but with different actual labels.

Clearly, labels can be swapped or renamed, leading to the same score. Also, a perfectly predicted clustering will have an FMI of 1, while a clustering independent of the real classes of 0. The FMI has various advantages:

- Unlike mutual information or V-measure, a random clustering will have an FMI score of 0, regardless of the number of clusters or individuals.
- A value close to 0 indicates largely independent labeling, while a value close to 1 indicates clustering agreement.
- Two equal labels (with permutation) have an FMI of 1.
- Finally, there is no hypothesis on the structure of clusters.

Its main disadvantage is that the actual labelling must be known in order to use this measure.

6.3.1.4/ Based on Mutual Information

Mutual information Entropy is the amount of uncertainty for a partitioning $U = (U_i)$, which can be defined by :

$$H(U) = -\sum_{i=1}^{|U|} P(i)\log(P(i))$$

where $P(i) = |U_i|/N$ is the probability that a randomly drawn object will end up in the U_i cluster. The mutual information between two partitionings U and V is defined by [Meilă, 2007]:

$$MI(U,V) = \sum_{i=1}^{|U|}\sum_{j=1}^{|V|} P(i,j)\log\left(\frac{P(i,j)}{P(i)P'(j)}\right)$$

where $P(i,j) = |U_i \cap V_j|/N$ and $P'(j) = |V_j|/N$ are probabilities in the obvious sense. This mutual information can also be written as follows:

$$MI(U,V) = \sum_{i=1}^{|U|}\sum_{j=1}^{|V|} \frac{|U_i \cap V_j|}{N}\log\left(\frac{N|U_i \cap V_j|}{|U_i|.|V_j|}\right)$$

This is a symmetrical measurement less than or equal to 1. Values close to 0 indicate that the labels are largely independent, while values close to 1 indicate a significant agreement

between the two clustering. However, perfect labelling does not necessarily have an MI of 1; on the other hand, random labelling has a negative MI. This score is independent of permutation or renaming. But it requires knowledge of true labelling, and unlike AMI defined below, MI is not adjusted to account for chance.

Normalized Mutual Information It is defined on the basis of mutual information, as follows [Vinh et al., 2010]:

$$NMI(U,V) = \frac{MI(U,V)}{mean(H(U),H(V))}$$

This is again a symmetrical measurement and increased by 1. Similarly, values close to 0 indicate that the labels are largely independent, when values close to 1 indicate a significant agreement between the two clustering. In addition, perfect labelling has an NMI of 1, and random labelling has a negative NMI. Finally, in the absence of true labelling, the NMI can be used as a model selection criterion: clusterings on various hyperparameters producing a score of 1 are agreed.

It should also be noted that this score is independent of permutation or renaming, but like all measurements in this subsection, it requires knowledge of true labelling. Finally, unlike the AMI defined below, the NMI is not adjusted to account for chance.

Adjusted Mutual Information The adjusted mutual information, which is a more recent technique than Normalized Mutual Information, is for its part adjusted to account for chance [Vinh et al., 2010]:

$$AMI(U,V) = \frac{MI(U,V) - E\{MI(U,V)\}}{\max\{H(U),H(V)\} - E\{MI(U,V)\}},$$

where $E\{MI(U,V)\}$ is the expected mutual information between two random clusterings.

This is again a symmetrical measurement and bounded by 1. Perfect labelling has an AMI of 1, and random labelling has an AMI close to 0: values close to 0 indicate that the labels are largely independent, when values close to 1 indicate a significant agreement between the two clustering. As before, the AMI can be used as a model selection criterion (clusterings on various hyperparameters producing a score of 1 are consensus). Finally, this score is independent of permutation or renaming.

6.3.2/ INTERNAL VALIDATION

6.3.2.1/ ELBOW METHOD

The “Elbow method” is the technique that considers the elbow that appears when the number of clusters is plotted on the x-axis and the percentage of variance explained on the y-axis [Thorndike, 1953]. This method is old and rather rudimentary, although popular.

In practice, it can be calculated in two ways, for each potential number of clusters k :

- we sum the intra cluster variances, which we divide by the overall variance.
- the sum of squared errors (SSE) is calculated: the sum, on all clusters, of the square distances between the points and their centroids.

For instance, in Figure 6.1, the elbow in the SSE appears for a number of clusters equal to 3.

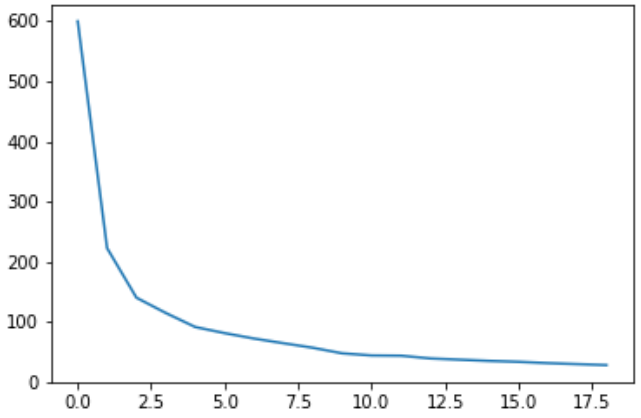


Figure 6.1: Illustration of the Elbow method: SSE versus nb of clusters

The following points can then be noted. For the K-means method, the average of each cluster is its centroid. For normalized data, the mean involved in the total variance is the origin. Finally, it is ultimately an F-test, in the variance approach.

6.3.2.2/ INDICES

Calinski-Harabaz In the absence of knowledge of actual labels, the *Calinski-Harabaz* index [Caliński et al., 1974], also known as the variance ratio criterion, measures how well clusters are defined. This score is defined as the ratio between the average of the dispersions between clusters and the intra-cluster dispersion; the larger the score, the better.

More precisely, for k clusters, this score is equal to:

$$s(k) = \frac{tr(B_k)}{tr(W_k)} \frac{N-k}{k-1},$$

where:

- $W_k = \sum_{q=1}^{k} \sum_{x \in C_q} (x - c_q)(x - c_q)^T$ is the intracluster dispersion,
- $B_k = \sum_{q=1}^{k} n_q (c_q - c)(c_q - c)^T$ is the dispersion matrix between groups,

with N the total number of points and c its center, when C_q is the set of points in the cluster q, n_q its number of points and c_q its center.

The advantages of this metric are that the score is calculated quickly, and it is higher when the clusters are dense and well separated, which is usually expected in good clustering. On the other hand, the Calinski-Harabaz index is generally broader for convex clusters, which does not fit well, for example, with clusters generally obtained by density-based techniques such as DBSCAN.

Davies-Bouldin This index is defined by the average similarity between each cluster $(C_i)_{i=1..k}$ and its most similar C_j. For the purposes of this index, this similarity is defined as the $R_{i,j}$ compromise between:

- the diameter s_i of the cluster C_i : average distance between each point of this cluster and its centroid,
- $d_{i,j}$: the distance between the centroids of the i and j clusters.

A simple way to construct such a measure, so that it is positive and symmetrical, is:

$$R_{i,j} = \frac{s_i + s_j}{d_{i,j}}.$$

The Davies-Bouldin index is then defined by [Davies et al., 1979]:

$$DB = \frac{1}{k} \sum_{i=1}^{k} \max_{i \neq j} R_{i,j}.$$

Clusters are better separated when the Davies-Bouldin index is low. 0 is the highest score, and partitioning is better when the index is low. The main advantages of this index is that:

- It can be calculated without the knowledge of true clustering.
- The calculation of the DB is simpler than the Silhouette scores, describe hereafter.

- The index is calculated only from distances.

Conversely, its disadvantages are:

- It does not evaluate very well in a situation of non-convexity.
- A good value produced by this index does not mean that the information collected is maximum.

Dunn The *Dunn index* [Dunn, 1974] is an internal clustering validation measure, which is calculated as follows:

- For each cluster, the distance between each point of the cluster and the points of the other clusters is calculated. The minimum of these distances corresponds to the inter-cluster separation (min.separation)
- The distance between the points of each cluster is calculated, its maximum diameter being the intra-cluster distance reflecting the compact nature of the clustering.

Dunn's index is then equal to:

$$D = \frac{min.separation}{max.diameter}$$

If clusters are compact and well separated, the diameter of the clusters should be small when the distance between clusters should be large. So a good clustering is associated with high values of this index.

6.3.2.3/ SILHOUETTE

The silhouette method [Rousseeuw, 1987] allows to estimate the consistency of the points belonging to clusters. The silhouette value measures how similar an object is to its own cluster (cohesion) compared to other clusters (separation). For each point, this value is between -1 and 1:

- Close to 1, the object fits very well with its own cluster and very badly with other clusters.
- Near 0, the data is at the border of two clusters.
- Close to -1, the object would fit better in the neighboring cluster: probably a bad assignment.

If most objects have a high silhouette value, then clustering is appropriate. Otherwise, it probably indicates too few or too many clusters. Let:

- $a(i) = \frac{1}{|C_i| - 1} \sum_{j \in C_i, i \neq j} d(i, j)$ the average distance between i point and the other points of its cluster, measuring how much i fits well with its own cluster (value is the smaller the better the assignment is), and
- $b(i) = \min_{i \neq j} \frac{1}{|C_j|} \sum_{j \in C_j} d(i, j)$ the smallest average distance between i and the points of all other clusters. The average dissimilarity of a point i to a cluster C being seen as the average of the distances between i and the points of C, the cluster with the smallest average dissimilarity is therefore the cluster close to i.

The silhouette value of object i is then $s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$ if the i cluster has more than one point, and 0 otherwise.

The average silhouette value on all points of a cluster assesses the extent to which the points of the cluster are closely grouped. The average value over the entire dataset indicates whether it has been appropriately clustered. If there are too many or too few clusters, some of them will typically have much weaker silhouettes than other clusters. Thus, the display and averages of the silhouettes can be used to determine the natural number of clusters in a dataset.

6.4/ Comparison of Modern Clustering Techniques for the IoT

6.4.1/ Experimental Protocol: A Network of Gas Sensors

The objective of this section is to show that the various clustering methods mentioned above may behave differently depending on the IoT application in question, and that it is therefore important to consider the framework in which the study is conducted. Typical scenarios for the use of each method have been described in a previous section, and we have indicated the particularities and application contexts associated with each method, the situations in which it can be expected to work well, as well as those in which these methods behave poorly. The metrics used to assess the quality of the results produced have been introduced to show that various measures provide different types of information on the performance of the various clustering techniques presented in this article. All this will be illustrated in this section, in a context of the Internet of Things already mentioned in this article: the difficult case of a network of sensors deployed for gas detection.

The data we will consider are based on a concrete experiment produced by Alexander Vergara of the University of California San Diego. It is entitled "Gas Sensor Array Drift Dataset at Different Concentrations Data Set", and can be found on the UCI Machine

Learning Repository [UCI,]. In details, this archive contains 13910 measurements from 16 chemical sensors exposed to 6 different gases (Ethanol, Ethylene, Ammonia, Acetaldehyde, Acetone, and Toluene) at various concentration levels. The scenario here is that the sink receives the values of these 13910 measurements, and it must cluster them in order to find 6 clusters corresponding to each of the 6 gases. In this clustering application at IoT, the operator knows which sensors he has deployed and the types of gases he wants to detect. In other words, he knows the number of clusters to recover, and since he himself has exposed his network to these 6 gases at various concentrations, he knows the true cluster. Its goal is to find the method to best separate all its measurements in order to make its sensor network operational once this cluster-based post-processing has been implemented. We would like to note that the gas sensors are not a generalized version of any IoT system, different data sets collected by different types of sensor networks can yield different results than the one obtained in this experiment.

In the following, we will therefore evaluate the clustering techniques for which it is possible to determine the expected number of clusters beforehand. In detail, these are K-means with random initialization or following K-means++, agglomerative clustering (we consider here, for comparison purposes, the linkages of type ward, complete, and average), the Gaussian Mixture Model, and Birch. First of all, it should be noted that spectral clustering was excluded from the study: the size of the dataset was too large and led to a memory error, showing that this technique should be excluded in the case of IoT networks producing a large amount of data (as expected following our previous presentation).

We also wish to show, in this study, that clustering work for the IoT often benefits greatly from being preprocessed, such as with an outlier removal step followed by a phase of dimensional reduction step. These two steps are generally neglected in the literature on clustering in an IoT context. But we will see, through the various clustering evaluations obtained, that adding such steps greatly improves the results. Concerning the detection and exclusion of outliers, we will consider the cases where such a step has not been carried out, then the implementation of a basic technique (i.e. isolation forest [Liu et al., 2008]), followed by an optimized version (extended isolation forest [Hariri et al., 2018]). In terms of dimension reduction, we will consider 4 scenarios: (1) the absence of such a step; (2) its realization by the most traditional method, namely the PCA; (3) a modern version of the latter optimized to the problem we are interested in, namely the Sparse PCA [Zou et al., 2006] (sensors for 6 different gases, exposed to a single gas, should ideally lead to a sparse matrix); (4) and finally the famous t-SNE [Maaten et al., 2008] technique. Regarding clustering quality measures, the following list was considered: Homogeneity, Completeness, Adjusted Rand Index, V-measure, Fowlkes-Mallows, and AMI.

The data were first standardized. Then we performed (or not, as the case may be) the outlier removal operations followed by dimension reduction, before performing the cluster-

ing and evaluating its quality. The calculations were carried out in Python. For extended forest isolation, we used the library available at the reference [Hariri et al., 2018]. The rest of the clustering methods, forest isolation, and various dimension reduction and evaluation techniques were performed with scikit-learn [Pedregosa et al., 2011b] library. At the parameter level, reasonable default values were considered: for instance, the branching factor and the threshold were set at 50 and 0.5 respectively for Birch, while the damping was set at 0.5 in the AP.

6.4.2/ Results of the Survey

First of all, it should be recalled that we have deliberately chosen to consider, in this experiment, the problem of detecting various gases at different concentrations, by means of a sensor network. A great diversity in the variability in the results after clustering has been observed, ranging from random-like results (frequently obtained) to results that carry richer structural information, given the complexity of the task considered in this experiment. In order to visualize the results better and assess the results in all the tables in this section, values are highlighted using a three color code:

- Blue: the values highlighted in blue are the highest recorded quality metrics with no outlier removal.
- Green: the values highlighted in green are the ones showing improvement as compared to the values obtained with no outlier removal.
- Red: the values highlighted in red are the ones showing degraded performance as compared to the values obtained with no outlier removal.

The clustering technique will be ranked according to who has the highest values for the majority of the quality metrics (the closer to 1.0, the better). In addition, as explained earlier in Section 6.3 all the metric values range from either 0 to 1 or from -1 to 1. However, using the python libraries mentioned in Section 6.4.1, we sometimes obtained values that exceeded 1 or below -1. Therefore, these values will be ignored and will be removed from the analysis.

Table 6.3 shows the obtained results when only the outlier removal operations (or not) have been performed without any reduction in dimensionality afterward. It is clear that on one hand, the K-means, AC Ward, and the Birch clustering techniques have been improved in term of quality when the Isolation Forest is used to remove outliers. On the other hand, only AC Ward and Birch have been improved when the Extended IF is used. However, Looking at GMM which had the best results (excluding Fowlkes-Mallows) when no outlier removal has been performed, a reduction in quality has been recorded when

Outliers	Clustering	Homogeneity	Completeness	ARI	V-measure	Fowlkes-Mallows	AMI
None	kmeans	0.1983	0.2471	0.0797	0.2201	0.2936	0.2196
—"—	kmeans++	0.1739	0.2409	0.0772	0.2020	0.3021	0.2015
—"—	AC ward	0.1757	0.2337	0.0869	0.2006	0.3006	0.2001
—"—	AC complete	0.0003	0.1554	-3.324	0.0006	0.4189	-5.143
—"—	AC average	0.0003	0.1554	-3.324	0.0006	0.4189	-5.143
—"—	GMM	0.3196	0.3613	0.2449	0.3392	0.4012	0.3388
—"—	Birch	0.1811	0.3273	0.0916	0.2332	0.3564	0.2327
Isolation Forest	kmeans	0.2059	0.2619	0.0850	0.2306	0.3020	0.2301
—"—	kmeans++	0.1608	0.2233	0.0772	0.1869	0.3031	0.1865
—"—	AC ward	0.1795	0.2341	0.0854	0.2032	0.2981	0.2028
—"—	AC complete	0.0003	0.1550	-3.581	0.0006	0.4184	-5.419
—"—	AC average	0.0003	0.1550	-3.581	0.0006	0.4184	-5.419
—"—	GMM	0.2783	0.3034	0.1669	0.2903	0.3291	0.2899
—"—	Birch	0.1856	0.3274	0.0922	0.2369	0.3557	0.2365
Extended IF	kmeans	0.1962	0.2447	0.0794	0.2177	0.2934	0.2173
—"—	kmeans++	0.1731	0.2401	0.0768	0.2012	0.3018	0.2007
—"—	AC ward	0.1792	0.2363	0.0889	0.2038	0.3017	0.2034
—"—	AC complete	0.0003	0.1551	-3.503	0.0006	0.4185	-5.334
—"—	AC average	0.0003	0.1551	-3.503	0.0006	0.4185	-5.334
—"—	GMM	0.2728	0.3460	0.1826	0.3051	0.3661	0.3047
—"—	Birch	0.1806	0.3293	0.0939	0.2333	0.3581	0.2328

Table 6.3: Clustering scores without dimensionality reduction

Isolation Forest and Extended IF are used. Moreover, despite the improvements in other clustering techniques, non of the values in green did match or exceed the quality metric values highlighted in blue. Therefore, we can conclude that none of the outlier removal techniques improved the clustering results. GMM without any outlier removal is by far the best.

Table 6.4 shows the measured quality metrics of the different clustering techniques when the PCA dimensionality reduction method is used after the removal (or not) of outliers. Indeed, a small improvement can be noticed for K-means, K-means++, AC Complete, and Birch, regardless of the outlier removal technique that has been used. However, GMM shows an improvement only with Extended IF. Similarly to the previous results, the best performing clustering technique, AC Ward, showed a reduction in quality instead of improvement with both Isolation Forest and Extended IF being used. Moreover, both AC Average and AC Complete had a sufficient improvement in their "Fowlkes-Mallows" and "Completeness" values respectively to surpass the matching blue metric values of AC Ward. However, still having the best Homogeneity, V-measure, ARI, and AMI, AC Ward without any outlier removal, could be considered the best choice.

Outliers	Clustering	Homogeneity	Completeness	ARI	V-measure	Fowlkes-Mallows	AMI
None	kmeans	0.1676	0.1893	0.0822	0.1778	0.2682	0.1773
—"—	kmeans++	0.1674	0.1891	0.0820	0.1776	0.2681	0.1771
—"—	AC ward	0.2302	0.2716	0.1131	0.2492	0.3072	0.2488
—"—	AC complete	0.0848	0.1794	0.0690	0.1152	0.3474	0.1146
—"—	AC average	0.0108	0.2496	-0.002	0.0208	0.4115	0.0199
—"—	GMM	0.2071	0.2580	0.0950	0.2298	0.3043	0.2294
—"—	Birch	0.1006	0.2026	0.0115	0.1345	0.3206	0.1339
Isolation forest	kmeans	0.1752	0.1920	0.0863	0.1833	0.2665	0.1828
—"—	kmeans++	0.1754	0.1923	0.0862	0.1835	0.2667	0.1830
—"—	AC ward	0.1889	0.2335	0.0763	0.2088	0.2915	0.2084
—"—	AC complete	0.0745	0.2994	-0.001	0.1194	0.3746	0.1186
—"—	AC average	0.0410	0.1675	-0.006	0.0659	0.3601	0.0651
—"—	GMM	0.2006	0.2346	0.0954	0.2162	0.2920	0.2158
—"—	Birch	0.1282	0.2751	0.0178	0.1749	0.3351	0.1743
Extended IF	kmeans	0.1746	0.1916	0.0855	0.1827	0.2664	0.1822
—"—	kmeans++	0.1750	0.1924	0.0857	0.1833	0.2668	0.1829
—"—	AC ward	0.1479	0.1761	0.0485	0.1608	0.2558	0.1603
—"—	AC complete	0.1315	0.3068	0.0219	0.1841	0.3423	0.1835
—"—	AC average	0.0080	0.2333	-0.001	0.0155	0.4131	0.0146
—"—	GMM	0.2181	0.2553	0.1022	0.2352	0.3011	0.2348
—"—	Birch	0.1522	0.2705	0.0696	0.1948	0.3361	0.1943

Table 6.4: Clustering scores on data reduced by PCA

In Table 6.5, where sPCA is used to reduce the dimension of the dataset, minor improvement can be found for K-means and K-means++, with both outlier removal techniques. However, AC Complete and GMM only showed partial improvement with Extended IF and Isolation Forest respectively. In this table, we notice that no clustering technique dominated the others in terms of quality metrics, thus, we have no clear “winner” yet. However, AC Ward has been improved enough with Extended IF in order to surpass the blue values in Homogeneity, ARI, V-measure, and AMI. Therefore, the combination of Extended IF, sPCA, and AC Ward could be considered better then the others.

The previous experiments clearly show that the performance can have large fluctuations, depending on how the methods are combined in the pipeline. This corroborated by the reported experiments in Table 6.6. In this table, we discover that when t-SNE is considered as the dimensionality reduction method, AC Average, GMM, and Birch have improved performance with both Isolation Forest and Extended IF. However, K-means, K-means++ only showed improvement with the former, and AC Wards only showed improvement with the latter. This time Birch is the one that stands out as the clustering technique with the

Outliers	Clustering	Homogeneity	Completeness	ARI	V-measure	Fowlkes-Mallows	AMI
None	kmeans	0.1975	0.2441	0.0796	0.2184	0.2937	0.2179
—"—	kmeans++	0.2039	0.2515	0.0885	0.2252	0.2989	0.2248
—"—	AC ward	0.2198	0.2851	0.0852	0.2482	0.3102	0.2478
—"—	AC complete	0.0706	0.3567	0.0062	0.1179	0.3887	0.1171
—"—	AC average	0.0129	0.2305	-0.002	0.0244	0.4098	0.0235
—"—	GMM	0.2249	0.2754	0.1017	0.2476	0.3095	0.2472
—"—	Birch	-6.291	1.0	0.0	-1.258	0.4190	-2.293
Isolation forest	kmeans	0.2297	0.2666	0.0979	0.2468	0.2981	0.2463
—"—	kmeans++	0.2297	0.2666	0.0979	0.2468	0.2981	0.2464
—"—	AC ward	0.2247	0.2793	0.0764	0.2490	0.3023	0.2486
—"—	AC complete	0.0732	0.2810	0.0168	0.1162	0.3781	0.1155
—"—	AC average	0.0074	0.2255	-0.001	0.0143	0.4133	0.0134
—"—	GMM	0.2388	0.2839	0.0990	0.2594	0.3074	0.2590
—"—	Birch	-3.143	1.0	0.0	-6.287	0.4186	-6.287
Extended IF	kmeans	0.2313	0.2685	0.0981	0.2485	0.2985	0.2481
—"—	kmeans++	0.2083	0.2522	0.0893	0.2281	0.2981	0.2277
—"—	AC ward	0.2484	0.3249	0.1045	0.2816	0.3294	0.2812
—"—	AC complete	0.1168	0.3113	0.0084	0.1699	0.3485	0.1693
—"—	AC average	0.0079	0.2364	-0.001	0.0154	0.4133	0.0145
—"—	GMM	0.2082	0.2560	0.0943	0.2296	0.3030	0.2292
—"—	Birch	3.1444	1.0	0.0	6.2889	0.4187	1.1054

Table 6.5: Clustering scores on data reduced by sPCA

majority of values coloured in blue. By looking at the results obtained using Isolation Forest and Extended IF, we also observe that AC Average outperforms all other approaches in terms of clustering quality. Therefore, we can safely conclude that the combination consisting of, Isolation Forest for outliers removal, t-SNE for Dimensionality reduction and AC average for clustering is by far the best combination for the given data set.

Outliers	Clustering	Homogeneity	Completeness	ARI	V-measure	Fowlkes-Mallows	AMI
None	kmeans	0.3519	0.3476	0.2418	0.3497	0.3723	0.3494
—"—	kmeans++	0.3537	0.3495	0.2427	0.3516	0.3730	0.3513
—"—	AC ward	0.3552	0.3593	0.2223	0.3572	0.3618	0.3569
—"—	AC complete	0.3123	0.3571	0.2209	0.3332	0.3793	0.3328
—"—	AC average	0.2922	0.3467	0.1994	0.3171	0.3684	0.3168
—"—	GMM	0.3443	0.3556	0.2138	0.3498	0.3587	0.3495
—"—	Birch	0.3547	0.3608	0.2370	0.3578	0.3749	0.3574
Isolation forest	kmeans	0.3883	0.3857	0.2904	0.3870	0.4134	0.3867
—"—	kmeans++	0.3888	0.3862	0.2908	0.3875	0.4137	0.3872
—"—	AC ward	0.3405	0.3527	0.2436	0.3465	0.3840	0.3462
—"—	AC complete	0.2992	0.3311	0.2338	0.3143	0.3801	0.3140
—"—	AC average	0.4221	0.5147	0.3666	0.4638	0.5066	0.4635
—"—	GMM	0.3608	0.3617	0.2666	0.3613	0.3957	0.3610
—"—	Birch	0.3648	0.3696	0.2820	0.3672	0.4104	0.3669
Extended IF	kmeans	0.3272	0.3233	0.2238	0.3253	0.3572	0.3249
—"—	kmeans++	0.3272	0.3233	0.2238	0.3253	0.3572	0.3249
—"—	AC ward	0.4051	0.4041	0.2972	0.4046	0.4201	0.4043
—"—	AC complete	0.2394	0.2432	0.1434	0.2413	0.2974	0.2409
—"—	AC average	0.3611	0.3723	0.2281	0.3666	0.3705	0.3663
—"—	GMM	0.3723	0.3802	0.2343	0.3762	0.3732	0.3759
—"—	Birch	0.4109	0.4201	0.2926	0.4154	0.4208	0.4151

Table 6.6: Clustering scores on data reduced by t-SNE

6.5/ CONCLUSIONS

In this chapter, we surveyed and classified modern clustering schemes applicable to the Internet of Things (IoT). We categorized the different approaches, highlighted their specificities and their domain of applicability, and we presented and explained the state-of-the-art clustering evaluation methods. Our findings were illustrated on the “Gas Sensor Array Drift at Different Concentration” data set, provided by the UCI Machine Learning Repository. For this dataset, we compared the most relevant clustering schemes and tested various combinations of outlier removal and dimensionality reduction approaches. Great diversity in the quality of clustering was observed, leading to the conclusion that there is no such thing as a simple rule for deciding the best clusters. In particular, the assumption that K-means should be used regardless of the application context did not appear to be true, and we were able to conclude that, instead, removing outliers using the Isolation Forest approach, reducing the dimensionality using t-SNE, and clustering using AC average yielded the best results.

7

On the Ability to Predict Firemen Interventions: A Case Study

In this chapter, we investigate the possibility of predicting future incidents using machine learning algorithms that are trained on a set of data containing information on almost 200,000 firemen interventions that happened during the last 6 years. After, pre-processing the data we tested multiple machine learning algorithms and we compared their results, aiming to determine which algorithm performs better. The results look promising as we were able to predict the number of interventions for each 3 hours block for a whole year, with an acceptable error margin.

7.1/ Introduction

Today's fire-fighters are operating in a technologically progressive environment. Tens of years ago, there was no such thing as a smoke alarm, water sprinkler system, jaws of life, and automatic shut-off valves, etc. While the tools are advancing rapidly, many fire departments are facing budget challenges, rising call volume, personnel and equipment shortages, and the overall expectation to do more with less. Therefore, there is a need for an intelligent system capable of making an assessment of the probability or likelihood that a particular event will occur. This can be achieved by building a model that can forecast future events using inputted information about incidents that happened in the past. This intelligent prediction system can help fire departments to manage more efficiently their allocated mobile and personnel resources, enabling them to have the required resources when an incident occur, reduce the response time, and save more lives with less effort.

7.1.1/ BACKGROUND

Various statistical, data-mining, and machine learning algorithms are available for use in predictive analysis model in several domains [Yaseen et al., 2018, Umair et al., 2018, Samanpour et al., 2018]. Each of these algorithms was developed to solve specific problems, which may make some of them more appropriate than others depending on the type, size, and other descriptions of the available data. Comparing different runs of different algorithms can bring surprising findings about the data. Doing so gives more detailed insight into the problem, and helps identify which variables within the data have the best predictive power.

One of the most known algorithms are the regression ones [Lewis-Beck et al., 2015], they can be used to forecast continuous data, such as predicting the trend for a stock movement given its past prices. The linear regression model [Montgomery et al., 2012] is one example, it attempts to model the relationship between two or more variables by fitting a linear equation to the observed data. One variable is considered to be an explanatory variable, and the other is considered to be a dependent variable. For example, a modeler might want to relate the weights of individuals to their heights using a linear regression model. Therefore, if we know a person's weight it is possible to estimate its height.

The decision tree [Breiman, 2017] is another approach to predictive analysis that is used for prediction and decision making. They are often chosen for predictive modeling because they are relatively easy to understand and effective. The goal of a decision tree is to split a set of data into smaller subsets that are related to each other. Starting at the root which includes the total set of data, and as we move down the tree, the goal is to split them into smaller and smaller subsets at each node of the tree. Each subset must be as distinct as possible from the other in terms of the target indicator. For instance, if we have a set containing information about people, an indicator could be sex, where we split the data into two subsets: one containing females only and the other males. Again, each subset can be split into multiple other subsets based on the age indicator, and so on. The optimal way to do that is by iterating through each indicator as it relates to the target indicator and then choosing the indicator that best splits the data into two smaller nodes. There are two stages to prediction. The first stage is to build the tree, test it, and optimize it using the available data set. In the second stage, the model is finally used to predict an unknown outcome.

An improved version of the decision tree method is the random forest [Liaw et al., 2002]. It is a supervised learning algorithm that builds a forest consisting of an ensemble of decision trees. The random-forest algorithm brings extra randomness into the model when it is growing the trees. Instead of searching for the best indicator while splitting a node, it searches for the best indicator among a random subset of indicators. This process creates a wide diversity, which generally results in a better model.

One cannot work on machine learning without considering the Support Vector Machines (SVMs) [Wang, 2005]. SVMs are based on the concept of decision planes that separates between a set of data having different class memberships. It was first used for classification purposes, then it showed great performances in Support Vector Regression as well (SVR) [Smola et al., 2004]. SVR aims to minimize a cost function using a kernel, which could be linear, Gaussian, or polynomial depending on data. The kernel determines the similarity between different features, and thus assign weights to their corresponding cost functions. Features that are close to each other and have the same output will be grouped together due to more weight, while outliers having less weight associated with them are discarded when the cost function is minimized. Thus, outliers will contribute very little to the final predictive model.

Last but not least, the least absolute shrinkage and selection operator (LASSO) [Tibshirani et al., 2015] is also widely used for prediction analyses. This algorithm works on the concept of penalized regression which helps to select the variables that minimize the prediction error. Ordinary Least Squares regression chooses the beta coefficients that minimize the residual sum of squares (RSS), which is the difference between the observed data and the estimated ones. LASSO adds a penalty to the RSS equal to the sum of the absolute values of the non-intercept beta coefficients multiplied by the parameter λ that slows or accelerates the penalty. E.g., if λ is less than 1, it slows the penalty and if it is above 1 it accelerates the penalty.

In this chapter, we compare several machine learning algorithms aiming to find the most efficient one in terms of prediction accuracy that estimates the number of interventions for each block of 3 hours. The rest of the chapter is organized as follow. Section 7.2 explains the procedure followed to collect, structure and clean the data. In Section 7.3 the data are visualized to understand better the hidden patterns and identify and extract the most important features. In Section 7.4 the previously mentioned machine learning algorithms are tested and their results are compared. In Section 7.5 a suggestion to improve the prediction results is presented, which opens the doors for a future work. Finally, Section 7.6 concludes our work and the intended future work is outlined.

7.2/ PRE-PROCESSING THE DATA

The fire department in the region of Doubs-France has provided us with a set of data containing information on a total number of ≈200,000 interventions that occurred during six years, from 2012 to 2017 (included). The data are separated into three different csv files: the list of departures by agents, the list of interventions, and the list of victims. These data files contain information about each incident, such as: The number of intervention (ID), The location, Y and X coordinates, date of intervention, used vehicle and its registration

number, departure motivation, alert reception time, departure time, end of intervention time, the total intervention time, age, sex and the state of the victim, etc. In this section, we will explain how we cleaned, prepared and enriched these data before passing to the prediction phase.

7.2.1/ DATA ACQUISITION

Weather conditions are considered to be a factor that affects significantly the number of road accidents, fires, and casualties. Therefore, including meteorological information to the analysis of incidents trend can improve the prediction results. Moreover, using the previously described csv files, we can extract for each individual intervention the hour of the day when it happened, as well as the day of the week, the month, and the year. This can help us detect tendencies correlated with these parameters (e.g., the number of car accidents increases on Saturday night because young people tend to drink during this period of time). Other parameters that could affect the number of road accidents, fires, and other events can be also taken into consideration, such as traffic hours, academic vacations, holidays, dawn and dusk time, moonrise, moonset, and the phase of the moon as well.

The idea is to predict the number of interventions per hour that will occur for a whole year. Therefore, we need to create a dataframe where we can aggregate all the available data provided by the fire department in addition to supplementary data that can be imported from other various sources. In order to build such a dataframe we performed the following:

- We initialized a dataframe containing keys ranging from '01/01/2012' until '31/12/17' of the form 'YYYYMMJJhhmmss'. The keys are generated by blocks of 3 hours.
- We imported the following weather related data from three weather stations located in Dijon-Longvic, Bâle-Mulhouse, and Nancy-Ochey [met,]: temperature, pressure, pressure variation each 3 hours, barometric trend type, total cloudiness, humidity, dew point, precipitation last hour, precipitation last three hours, average wind speed for every 10 minutes, bursts over a period, measurement of the burst period, horizontal visibility, and finally the current time.
- We added the imported meteorological information to the previously initialized dictionary. However, the data were not complete, some were missing and marked as "mq". Therefore, we applied a linear interpolation to fill the blanks. We then introduce various temporal features such as the day in the week (Monday, etc.), month, year, hour in the day, etc.
- We have extracted the number of interventions from the csv files sent by the fire brigade department. In the latter, there is one line per intervention, which includes

the time of the intervention to the second. We group these interventions by blocks of 1 hour.

- The features “holidays” and “startendVacation” are added to the dataframe, which is initialized to 0 (false). The first will increase to 1 for any 3 hours block within an academic holiday period, while the second will increase to 1 for the days corresponding to the beginning and end of holiday periods.
- We have added the public holidays (1 or 0, for true or false), as well as a second feature that is set to 1 the days before public holidays, for the hours ranging from 3:00 pm to 11:00 pm (otherwise 0).
- We have included information related to the “Bison Futé” which is a system put in place in France to communicate to motorists all the recommendations of public authorities regarding traffic, traffic jams, bad weather, accidents, advice, etc. It classifies the days at risk according to several colors: green = all is well, fluid traffic, orange = dense traffic, red = difficult traffic, traffic jams, black = to avoid because of traffic jams and slow traffic. We integrate these information through two additional features “bisonFuteDepart” and “bisonFuteRetour”. They are 0, 1, 2 or 3, depending on whether the traffic forecasts correspond to Green, Orange, Red or Black.
- Finally, we added to the dataframe the sunrises, moon phases, etc. A boolean feature ’night’ is added to the dataframe to know if it is a day (0) or night (1). Moreover, We add another boolean feature indicating if the moon has risen at h+30min, and what is its phase (an integer from 0 to 7, namely 0 for new moon, 2 for the first quarter, 4 for the full moon, and 6 for last quarter).

In the Table 7.1 we give an illustrated example showing how the final dataframe looks like. It contains all the information extracted from the provided csv data files (hour, day, etc.) and the ones imported from external sources (meteorological, ephemeris, traffic, vacations, etc.). Each column represents a block of 3 hours. Over the period of 6 years, for each day we have 8 columns representing the 24 hours of the day.‘

7.2.2/ Data Cleaning

In this subsection, we will explain how we detected and removed outliers that can affect negatively the end results. First, we noticed that the mean value of the number of interventions in the dataframe is 3.59 interventions/hour, the minimum number of intervention is 0, and the maximum is 85. Finally, in 75% of the cases the number of interventions is less than 5.

Looking at Figure 7.1, there seems to be some very particular situations, having generated a large number of interventions. As the latter can affect the learning phase, it is

Table 7.1: Illustrated example of the final dataframe

Intervention nbr	**0**	**1**	**..**	**..**	**..**	**52559**
year	2012	2012	..	2012	..	2017
..	..	..	..	..	..	..
StartEndVacation	0	0	..	0	..	1
..	..	..	..	..	..	..
WindDirDijon	190	175	..	120	..	100
..	..	..	..	..	..	..
HumBâle	89	85	..	79	..	78
Day	150	150	..	365	..	365
..	..	..	..	..	..	..
3 Hours block	0	3	..	24	..	24
Holidays	1	0	..	0	..	0
VisibiltyNancy	55000	56666	..	48333	..	60000
..	..	..	..	..	..	..
NbrIntervention	7	10		8		6

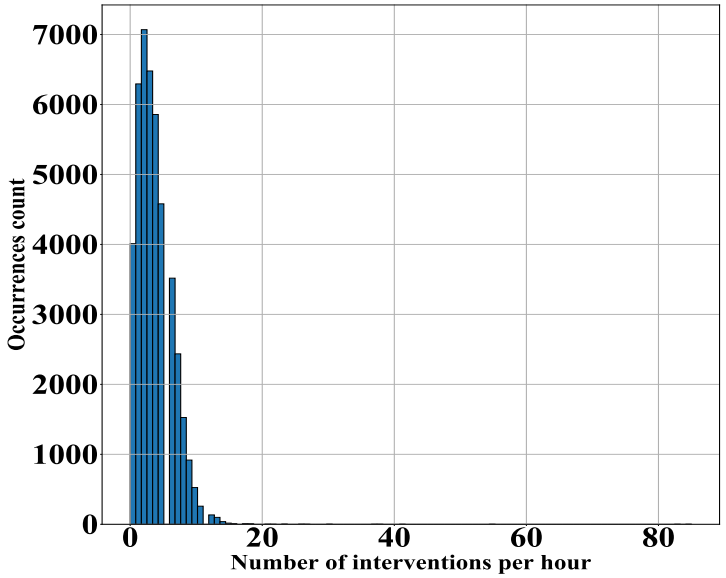
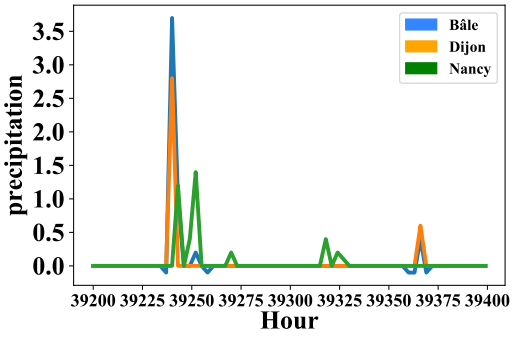
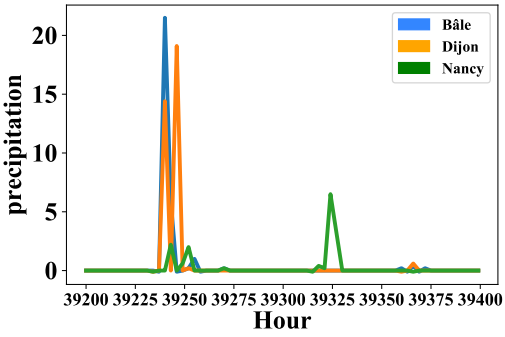


Figure 7.1: Histogram showing the frequency of each number of intervention

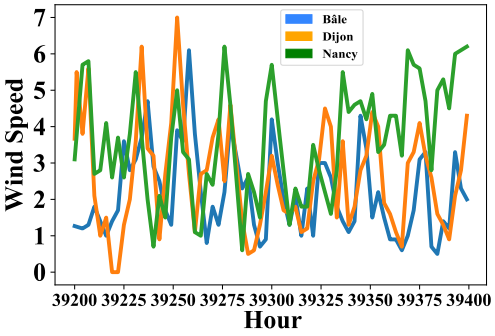
appropriate to look at them in more detail, asking whether or not they should be discarded (outliers). We sorted the IDs ranging from 0 to 52559 in descending order according to their corresponding number of intervention. We have noticed that among the top 8 IDs we have 7 neighboring ones (same day, neighboring hours). So we investigated more what happened during this period of time. The ID number 39243 has the maximum number of interventions of 85 interventions. We listed the year, month, day, and the hour of its neighboring ID's. We noticed that they all belong to the night of $24th$ to the $25th$ of June 2016. In the csv files, the following main causes were noted for these days: exhaustion, floods, protection of miscellaneous property, and accidents. That particular night there were very violent storms [est, 2016], leading to the recognition of the state of natural disaster in the region of Doubs. Therefore, we have two options, either we consider them as outliers and dismiss them in learning by smoothing out the missing data or consider that this is a consequence of exceptional weather, but that with the weather data we should be able to predict this. It remains to be seen whether it seems possible to predict this peak of intervention using meteorological data from Basel, Dijon, and Nancy. In what follows, we will look at an interval of 200 hours (a little less than 9 days) centered around this storm.



(a) Precipitation every 1h



(b) Precipitation every 3h



(c) 10-minute average wind speed

Figure 7.2: features visualisation

We start by looking at the precipitation during this period. Figure 7.2a shows that there is indeed a peak in rainfall, but it does not seem obvious: a little less than 4mm, while

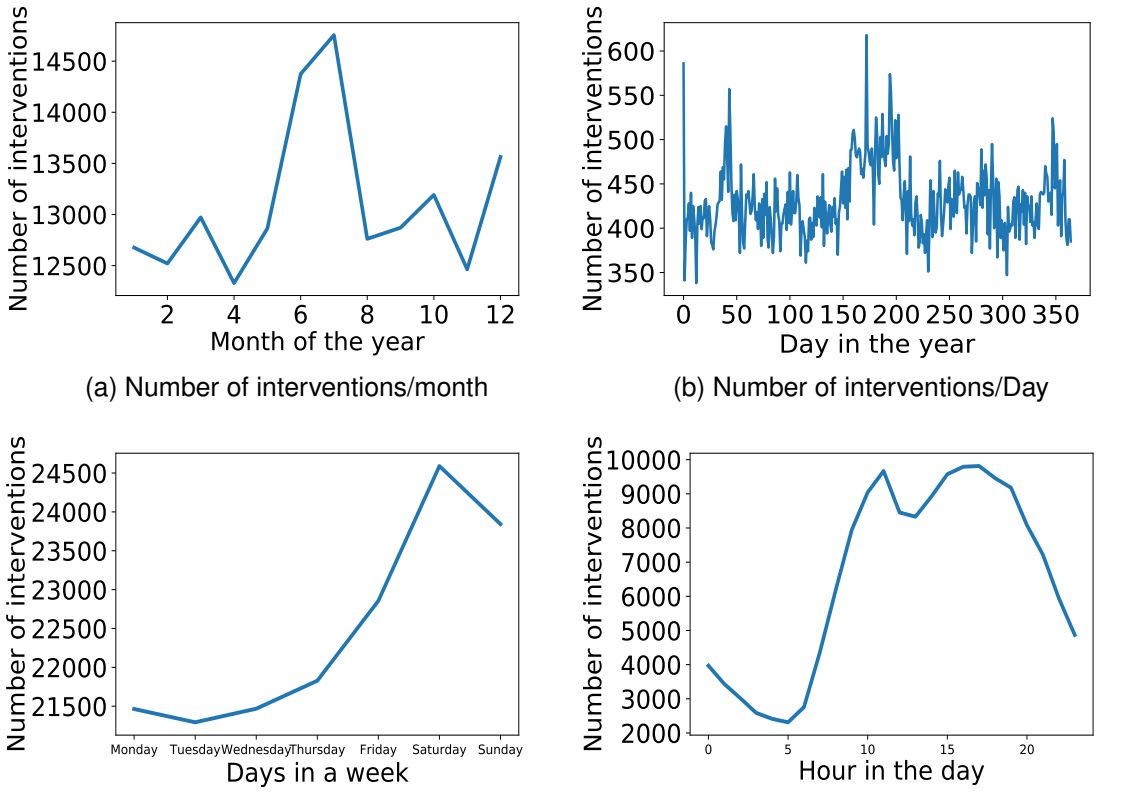
the article from the 'Est republicain' [est, 2016] speaks of 80mm in less than an hour. If we compare the IDs of the peaks with the IDs of the maximum precipitations in the data provided by the weather station in Basel, we notice that they are not among the most important values recorded in this station. This is probably due to Basel's distance from the storm, located between Sancey and L'Isle-sur-le-Doubs (although Basel is closer to this region than Dijon or Nancy). It may, however, be that, at this time, a lot of water has fallen for a relatively long time, so we can look at what it is over a longer period. Therefore, let us look at Figure 7.2b showing the rainfall over 3 hour blocks. A thunderstorm peak appears clearly, and the amount dropped is over 20mm (closer to 80mm) mentioned above. We checked if such a quantity is something frequent. It turned out to be the fifth highest rainfall recorded from 2012 to 2016 inclusive. We looked at how many interventions there were in the vicinity of the periods of heavier rainfall. On average, there are more interventions during maximum rainfall. But we remain very far from the number of interventions during the 6 hours of this stormy peak that is considered as a natural disaster.

To conclude on precipitation, one caused an extreme peak in interventions, but this is not the case for the other severe weather events. The most important of these leads to a number of interventions out of the ordinary, but far from the extreme studied situation. These precipitation data are therefore important for our prediction, but they do not allow us to predict the extreme situation of June 25, 2016 (due to weather measurements that are not sufficiently localized). We look to see if other weather information, measured in Basel, Dijon or Nancy, were remarkable at midnight on 25/06/16. We start with wind speed. As we can see in Figure 7.2c There is a small peak, but nothing exceptional. The wind speed (10-minute average) was less than 3.5 m/s. We are far from the maximum of 15.9m/s that we can find when sorting the wind speed in ascending order. We also looked at humidity, temperature, and pressure values; there was indeed a drop in pressure, but on the other variables, there was no particular situation at first sight.

It seems difficult, with the weather data currently considered, to predict such a peak of interventions. The situation is sufficiently exceptional and has a real impact on the considered data. For example, the number of interventions in June may be significantly overestimated. For these reasons, we chose to artificially smooth the intervention data on this date. For outliers, we put the same number of interventions as the next day at the same time.

7.3/ DATA VISUALIZATION

After cleaning our data and removing the outliers, it is time to analyze it in order to discover the tendencies correlated with the different feature of the created dataframe. At first, we calculated the degree of correlation of each parameter with the number of interventions.



(a) Number of interventions/month

(b) Number of interventions/Day

(c) Number of interventions/Day of the week

(d) Number of interventions/Hour of the day

Figure 7.3: features correlation with number of interventions

We discovered the following interesting facts:

1. The strongest positive correlation concerns the hour and the strongest negative concerns the night (there are more interventions when it is a day, whatever the season). Moreover, this is something cyclic, every day at 2 am there is less intervention than at 6 pm.

2. The weather data have an impact as well, although less pronounced. Temperature (positive) and humidity (negative: when it rains, people go out less) are first, wind speed and visibility comes second.

3. The eve of public holidays has some importance. The beginning or end of holidays have a greater impact than being on holiday or not. The fact that the moon is apparent plays a little, but less its phase.

4. The year is also weakly positively correlated: the number of interventions tends to increase from year to year

5. Temperature data are highly correlated with each other and are strongly but inversely correlated with humidity. In a slightly more surprising way, they are correlated with visibility.

The relevance of keeping several variables so highly correlated could be considered. If necessary, data could be reduced, for example by aggregating (average, etc.) those correlated, and see if this facilitates learning and predicting the number of interventions. The first thing we notice is that the number of interventions increases each year, which is normal due to population growth. If we accumulate the number of interventions by month (Figure 7.3a), we notice that the summer months are most of the time the busiest in terms of the number of interventions. The end of the years are also busy. One can imagine reasonable causes for this: summer vacations conducive to outings and physical activities, fires. And end-of-year parties. This assumption can be further emphasized by looking at Figure 7.3b, we can clearly see the trends related to summer, as well as a significant number of interventions around mid-February, and lows around April, August-September, and November. Let's move on to the day in the week (Figure 7.3c). Weekends are much busier, with a peak on Saturday (including Friday night after 24h). Let's look at the time in the day (Figure 7.3d). As expected, it is during the hours of the day that the number of interventions is highest, with a small drop during midday. And there is hardly any intervention at 5:00 in the morning.

7.4/ The Prediction of Interventions

7.4.1/ Learning and Testing Stages

In this section, we present different machine learning methods that we have used to predict future incidents. We also compare their results aiming to find the best approach that produces the smallest prediction error. We split our data-set into two sub-sets, a learning sub-set (2011-2016) and a verification one (2017). The former is used to build a model that will predict the number of interventions for each 3h blocks for the next year (2017), and the latter is used to verify the accuracy of these predictions.

The first step is to specify which data are numeric (year, humidity, day, month, etc.) and which are qualitative (night, holiday, day if the week, etc..), since the latter will be processed by full disjunctive coding when they are normalized (to avoid large data being mixed with small data). An encoder is introduced for qualitative data, for example, if we consider qualitative data at three levels (0, 1, 2), it will be encoded as a three-bit vector, as follows: [1, 0, 0], [0, 1, 0] and [0, 0, 1]. We also create two python pipelines, the first is on the numerical data, allowing them to be normalized (mean 0, standard deviation 1), and the second is on the qualitative data to perform a full disjunctive coding. Finally, the complete pre-processing pipelines for the explanatory variables are formed.

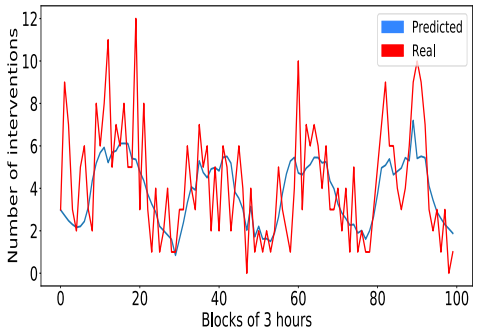
Before testing the different machine learning methods, we start by discovering some reference values concerning the mean square error. Having this information will allow us

to have a better estimate of the results produced thereafter. The mean square error, or MSE, reinforces the importance of large errors over small ones and is, therefore, a better score than the mean of absolute errors. We take the square root, leading to the RMSE, to have similar units. First, we only consider the average number of the interventions for each year, in order to predict the number of interventions for 2017. We obtain the following errors: MAE : 2.39, Root Mean Square Error (RMSE) : 2.94. If we try to do better by taking the average per hour, the following results are obtained: MAE: 1.75, RMSE: 2.28, which is better than the previous results. Let these numbers be our references numbers to test the efficiency of the different prediction models that we will use.

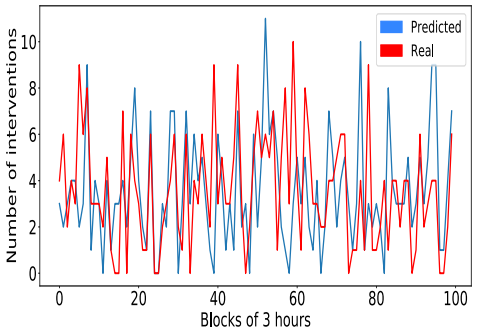
7.4.2/ PREDICTION

We train the prediction models after applying the previously described pre-processing pipelines. Afterward, we use the trained model to predict the number of interventions for the year of 2017. We used in this work the Scikit-learn Python module [Pedregosa et al., 2011a] that integrates a wide range of state-of-the-art machine learning algorithms (including the models that we are interested in) for medium-scale supervised and unsupervised problems.

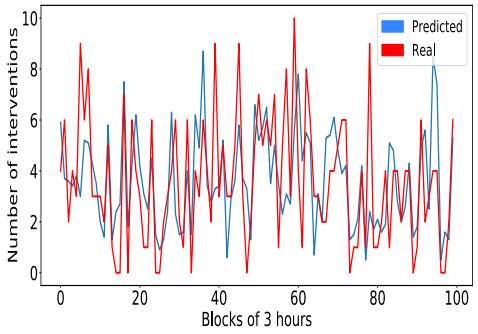
- Linear regression: The obtained RMSE for the Linear regression model is 2.59 which is below the standard deviation of the number of interventions (2.7). In addition, the value of the MAE is 1.763. According to these results, Linear regression turned out to be better than the “average of the year” method, but less accurate then the “average per hour”. Figure 7.4a shows a comparison between the number of actual interventions and that predicted. As we can see the prediction of the number of interventions follows the trend, but is clearly not extraordinary.
- Decision Tree: Since linear regression is not that great, we try the decision tree which is a more complex model to see what it gives. It is a powerful model, capable of finding complex non-linear relationships in data. To evaluate the decision tree model we use the cross-validation method. The training set is first partitioned into k=10 distinct blocks, then the training and then the evaluation of the decision tree model is performed in 10 successive passes, while reserving each time a different block for the evaluation and performing the training on the other nine blocks. The end results are 10 evaluation scores. The average of these results is then calculated to assess the model’s accuracy. Figure 7.4b shows a comparison between the predictions and the real number of interventions. The obtained RMSE and MAE are 2.86 and 2.15 respectively. It seems to work less well than the linear regression model and the naive models as well. Therefore, instead of using a single decision tree, we will move to random forests.



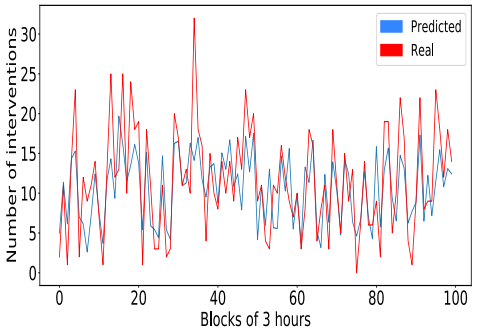
(a) Prediction using Linear Regression



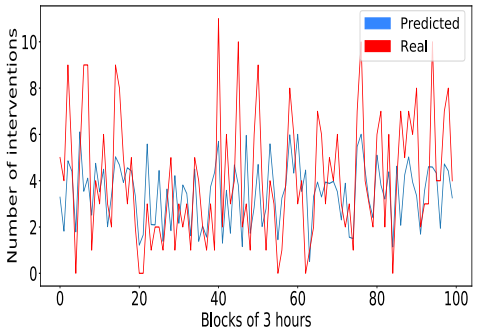
(b) Prediction using Decision Tree



(c) Prediction using Random Forest



(d) Prediction using LASSO



(e) Prediction using SVM

Figure 7.4: Visualization of prediction results

- Random Forest: The principle is to drive many decision trees on random subsets of variables and then average their predictions. The comparison between the real number of interventions and the predicted ones is shown in Figure 7.4c. The results look promising since after a random search of best hyper-parameters, the obtained RMSE and MAE are the lowest until now: 2.19 and 1.68 respectively.

- Support Vector Machine (SVM): In order to get the best results, we used the grid search technique to find the optimal hyper-parameter C. In the beginning, we select a grid of values for C of size N ranging between two numbers, say: a and b, then

we run the SVM for each value of C and we calculate the RMSE of each one of the N runs. We then select three values of C that returned the minimum RMSE, we reset the grid to range within these 3 selected values and we rerun the SVM again for each value in the new grid. Again, we calculate the RMSE and we repeat the same cycle until we find the optimal value for C that returns the smallest RMSE. The optimal value for the hyper-parameter C in our case is 1.36. The smallest RMSE for this value of C is 2.21 and the MAE is 1.68.

- Least Absolute Shrinkage and Selection Operator (LASSO): LASSO is a type of linear regression that uses shrinkage. It is where data values are shrunk towards a central point, like the mean. LASSO penalizing the absolute size of the regression coefficients, where some of the parameter estimates may be exactly zero. The larger the penalty applied, the further estimates are shrunk towards zero. λ is a parameter that defines the amount of shrinkage. In order to find the optimal value for λ, we used the same grid search technique we have used for SVM. The best value is found to be 0.0031 in our case. Finally, the smallest RMSE for a λ equal to 0.0031 is 4.53 and the MAE is 3.48.

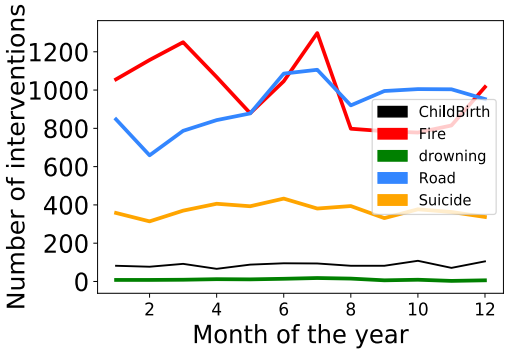
Table 7.2 summarizes the results. The random forest method achieved the lowest RMSE and MAE. SVM performed good as well, it has the same MAE as the random forest and a slightly bigger RMSE. Surprisingly LASSO achieved the worse results with very high errors.

Table 7.2: Error comparison

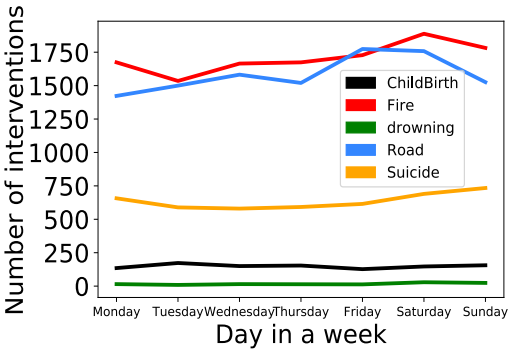
	RMSE	MAE
Average	2.94	2.39
Average per hour	2.28	1.75
Linear Regression	2.59	1.76
Decision Tree	2.86	2.15
Random Forest	**2.19**	**1.68**
SVM	2.21	1.68
LASSO	4.53	3.48

7.5/ DISCUSSION

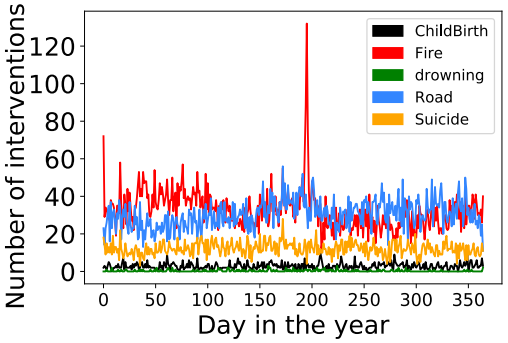
In this work, we are predicting the number of interventions regardless of what is the type of it (suicide, road accident, fire, etc.). The obtained results are fairly acceptable,



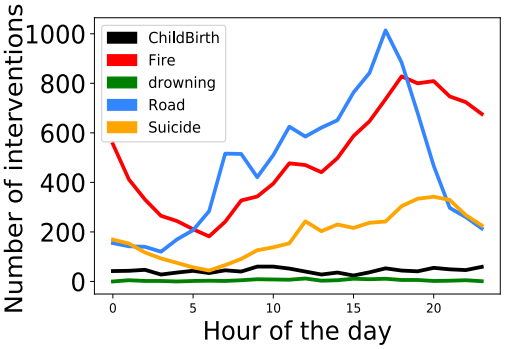
(a) Number of interventions/month



(b) Number of interventions/day



(c) Number of interventions/day of the week



(d) Number of intervention/hour of the day

Figure 7.5: Interesting visible correlations

and we were capable of predicting the number of interventions for each 3 hours block to some extent. However, if we look at Figures 7.5a, 7.5b, 7.5c, and 7.5d, we notice that the number of interventions differs greatly depending on the type. For instance, in Figure 7.5a we can see a peak in fire interventions starting approximately from May until the beginning of July, and a similar peak for road accidents starts from May until August. Drowning, childbirth, and Suicide remain stable over the whole period. This trend is also visible in Figure 7.5c where we notice a peak during the weekends in fire and road accident interventions. Moreover, Figure 7.5d shows a peak between 3 pm and 7 pm for road accidents, and from 4 pm and 8 pm for fire incidents. Therefore, if we can predict first the type of the incident, we will be able to predict more accurately the number of interventions. This indeed is a really interesting approach that we will investigate in more depth in future work.

7.6/ CONCLUSION

Given a large set of data containing information about 200,000 interventions that took place over a 6 years period in the region of Doubs (France), we aimed to determine

which technique performs best at predicting the number of interventions for the next 3 hours block with an accuracy that is sufficient for practical purposes. Our results show that the random forest is the best technique for predicting the number of future incidents with respect to the obtained RMSE and MAE values. The predictions are within an acceptable error margin and could help fire departments to anticipate future incidents and better manage their human and mobile resources. The improved management of resources leads to a reduction in the total intervention time and an increase in the response speed. This helps reduce the number of injuries significantly, save more lives, and limit the consequences of an event in the shortest delay possible.

For future work, we will check whether the number of interventions follows a specific probability distribution law and if the associated parameters evolve in time if so we could predict the evolution of these parameters instead of the actual number of interventions. Finally, we will be using Neural Networks for prediction and possibly add additional parameters to the models in order to increase the prediction accuracy.

III

CONCLUSION

8

GENERAL CONCLUSION

8.1/ SUMMARY OF THE PHD THESIS

Nowadays, the word “Smart” could be virtually placed before the name of any object or tool. Instances include Smart Houses, Smart Grid, Smart Cars, Smart Cities, etc. A key area, ripe for development, is Smart Fire Fighting. Today’s emergency responding including firefighting is data-poor and lacks an integrated data analysis and decision-making tools. The purpose of this thesis is to provide a stepping stone toward the development of a smart emergency responding environment. The goal is to make situational awareness significantly stronger by harnessing the power of emerging information, communication, sensors and building predictive models and decision-making tools. This thesis presents a series of both software and hardware solutions aimed to advance smart emergency responding. The full requirements are enormous for consideration in this thesis. Therefore, a hand-full of priority objectives were derived from guidelines drawn up through a variety of mechanisms, including the coordinated efforts between multiple teams with a background in sociology, sports, computer science, and emergency responding. Important feedback from multiple perspectives on this topic where considered during the course of the project. Three main objectives were set for this thesis namely development of energy-efficient algorithms for wireless sensor networks. Design, development, and validation of a smart emergency responding prototype system (proof of concept) that ensure a reliable delivery of data over a long distance and in harsh environments. Finally, development of data analysis and decision-making tools. Our main contributions and perspectives for this dissertation are listed in the following.

Energy Management: We presented a novel fault-tolerant DPM-based transmission reduction algorithm that is simple in terms of complexity yet robust, and effective in terms of prediction accuracy and data transmission reduction. This algorithm reduces data transmissions between the sensor node and the Sink by simultaneously building and maintaining on both of them a data forecasting model. This model enables the sensor node to

transmit a measurement only when the difference between it and its corresponding prediction exceeds a predefined error threshold. Meanwhile, the non-transmitted measurements are reproduced at the Sink using a similar instance of the jointly built prediction model. In addition, in order to ensure synchronization between the model on the Sink and the one on the sensor node, and to improve the quality of the replicated data, we coupled this technique with a missing data detecting mechanism and a data reconstruction algorithm. This letter, deployed on the sink, exploits both temporal smoothness and spatial correlation among different sensed features to recover missing values caused by communication link failures.

Then, we presented an extension for the previous approach. To further reduce energy consumption we proposed merging the fault-tolerant DPM-based algorithm with an adaptive sampling technique. This later, enable the sensor node to adapt its sampling rate depending on the level of variations in the collected data over a certain period of time. If no significant change is noticed, the sensor node is allowed to reduce its sampling speed (the time between two consecutive samples) and sleep for a longer duration. However, if important variations are observed, the sensor is forced to use the maximum sampling speed for not to miss important events. This gives the sensor more flexibility in managing its sampling rate instead of keeping it at a maximum level all the time even when it is not necessary which translates in a reduction in sampling energy consumption.

Finally, we presented a spatial-temporal correlation-based approach for sampling and transmission rate adaptation in cluster-based sensor networks. Unlike the previous approaches, the sensor nodes do not need to run any distributed algorithm. It is rather a centralized approach where the cluster head is responsible for collecting data from its member sensor nodes and computing a correlation function in order to measure the correlation degree among them. Finally, the sensors that show high correlation will be asked to reduce their sampling rate and the ones showing low correlation will be asked to increase it. Moreover, in order to ensure the integrity of the data, the same reconstruction algorithm described in (Chapter 2) is implemented on the Sink station in order to reproduce the whole data-stream including values that have not been sampled due to a prolonged sleep duration.

The previously described approaches have been validated trough both simulations on real sensor data-sets and implementation on real sensor devices. We measured the energy consumption of the sensing, processing, and transmission activities of the device in addition to the overall energy consumed by each algorithm. The final results have proven the efficiency of our proposed algorithms in reducing the amount of consumed energy and extending the lifetime of the network, all while preserving the quality of the collected data.

Proof of Concept: We presented a proof concept of a smart-system for state assessment and localization of firefighters during interventions. It consisted of a smart-watch for heart rate and movement monitoring and two sensor devices, one for localization and the other for environmental data sensing. A LoRaWan network architecture has been adopted for its ability to transmit the collected information over a long distance with the presence of obstacles and since it requires virtually no pre-existing infrastructure. Our proposal is shown through experiments to be more reliable compared to the traditional mechanical Personal Alert Safety System (PASS) used by firefighters.

Decision Making: We reviewed recent clustering techniques and provided a rigorous account of when and for which task they might be useful in the context of WSNs and the IoT. We Provided a thorough comparison between the main important techniques which have been devised in the literature, as well as their respective advantages and disadvantages. We Analyzed an example with real data for a gas leak detection sensor network to show how these various clustering techniques can respond in different ways to data received at the sink level. Finally, We demonstrated that choosing the correct method makes a difference via performance comparisons on a massive data set. Some methods showed excellent accuracy and scalability, whereas others appeared completely inappropriate for the task.

With our hands on a large data-set containing information on 200,000 intervention that happened during a period of 6 years in the region of Doubs, France. We studied the possibility of using Machine Learning to predict the number of future interventions. This could help the fire brigade to better manage their allocated mobile and human resources by increasing or decreasing them in a specific area according to the produced predictions which would prevent having excess resources in low-risk areas or shortage in high-risk ones. Using a hand-full of well established Machine Learning algorithms we managed to predict the number of interventions for every 3h block for a whole year with an acceptable error margin.

8.2/ PERSPECTIVES

Although a lot has been done in this thesis, more could be achieved. In this section, we list and discuss our perspectives for future work.

Energy Management: The proposed energy management algorithms could be implemented on the environmental data sensing device, but not on the localization device and the smart-watch since they operate differently. They are in constant active mode, never

set to sleep and they only transmit in case of emergency. We cannot apply the transmission reduction algorithm since every single emergency alert must be transmitted, and we cannot apply the adaptive sampling since constant monitoring of movements and heart-rate is indispensable for emergency detection. Therefore, In future work, we should consider the optimization of the embedded algorithms on these devices in order to ensure a prolonged operational lifetime.

The Monitoring System: The localization device is designed to work outdoors, however, if operating indoor, this information will be lost, the firefighter will no longer be localized. The signal strength of the LoRa packet could be used to estimate the location of an emitting device. In future work, we aim to conduct a comprehensive study on Indoor, GPS-free localization via LoRa.

We successfully validated a prototype of the proposed remote monitoring system, however, the experiments were conducted on a small-scale. In future work, we aim to conduct close to operational experiments with the fire brigade members in a controlled environment in order to observe how the system works on large-scale especially to test signal strength and delivery reliability of data packets in a harsh environment.

Decision Making: Although we managed to develop a system that transmits, stores, and visualizes the collected real-time sensor data using a web application, we did not yet exploit this data to develop any predictive tools for decision making. We did however investigated massive sensor data clustering which is a step-stone towards this goal. Eventually, the collected sensor data could be analysed and used to help forecast in real-time the fire location(s), size, propagation, and environmental conditions. More information could also be retrieved using more sensors and it could be possible to predict the time to significant failure in structural and situational tenability of buildings, compute the risk factors based on the incident (locations of fire, firefighters, occupants, and conditions), compute change in fire over time based on cleared area and draw an exit path from the intervention area. Moreover, by acquiring real-time weather data it is possible to analyze current and projected weather, especially wind-speed which could be a decisive factor in fire fighting. Information on hospital status (occupancy, resources, etc.) could also be used for efficient treatments of victims, etc. All this and other examples of predictive decision-making tools could be considered in future work.

PUBLICATIONS

JOURNAL PAPERS

[**1**] Gaby Bou Tayeh, Abdallah Makhoul, David Laymani, Jacques Demerjian. A Distributed Real-Time Data Prediction and Adaptive Sensing Approach for Wireless Sensor Networks. Perasive and Mobile Computing, Vol. 49, pages 62-75, 2018.

[**2**] Gaby Bou Tayeh, Abdallah Makhoul, Charith Perera ,Jacques Demerjian. A Spatial-Temporal Correlation Approach for Data Reduction in Cluster-Based Sensor Networks. IEEE Access 7, pages 50669-50680, 2019.

[**3**] Christophe Guyeux, Stépahne Chrétien, Gaby Bou Tayeh, Jacques Demerjian, Jacques Bahi. . Introducing and Comparing Recent Clustering Methods for Massive Data Management in the Internet of Things. Journal of Sensor And Actuator Networks, Vol. 8, page 56, 2019.

[**4**] Gaby Bou Tayeh, Abdallah Makhoul,Jacques Demerjian, Christophe Gueyeux, Jacques Bahi. Fault Tolerant Data Transmission Reduction Method for Wireless Sensor Networks. World Wide Web, page 1-20, 2020.

CONFERENCE PAPERS

[**5**] Gaby Bou Tayeh, Abdallah Makhoul, Jacques Demerjian, David Laymani. A New Autonomous Data Transmission Reduction Method for Wireless Sensor Networks. IEEE Middle East and North Africa Communications Conference (MENACOMM), pages 1-6, 2018.

[**6**] Gaby Bou Tayeh, Joseph Azar, Abdallah Makhoul, Christophe Guyeux, Jacques Demerjian. A Wearable LoRa-Based Emergency System for Remote Safety Monitoring. International conference on Wireless Communi-cations & Mobile Computing (IWCMC), in press, 2020.

SUBMITTED PAPERS

[**7**] Christophe Guyeux, Gaby Bou Tayeh, Abdallah Makhoul, Stépahne Chrétien, Jacques Demerjain, Jacques Bahi. On The Ability to Predict Firemen interventions: A Case Study.

[**8**] Joseph Azar, Gaby Bou Tayeh, Abdallah Makhoul, Raphael Couturier. Efficient IoT Data Compression and Reconstruction Using SZ and Autoencoder.

[**9**] Gaby Bou Tayeh, Christophe Guyeux, Abdallah Makhoul, Jacques Bahi, Sébastien Freidig. A Personal LPWAN System For State Assessment And Localization of Firefighters.

BIBLIOGRAPHY

[con,] **active-semi-supervised-clustering repository**. https://github.com/datamole-ai/active-semi-supervised-clustering. Accessed: 2019-05-29.

[STE,] **Average steps per minute for different exercises, july 2019.** https://www.verywellfit.com/pedometer-step-equivalents-for-exercises-and-activities-3435742. Accessed: 2019-08-28.

[met,] **Meteo France**. http://www.meteofrance.com/accueil.

[NEW,] **National early warning score (news) 2, royal college of physicians, london, u.k., december 2017.** https://www.rcplondon.ac.uk/projects/outputs/national-early-warning-score-news-2. Accessed: 2019-08-27.

[UCI,] **UC Irvine machine learning repository**. https://archive.ics.uci.edu/ml/index.php. Accessed: 2019-07-17.

[est, 2016] (2016). **Orages : des inondations aussi dans le Territoire de Belfort**. https://www.estrepublicain.fr/.

[1Gate,] 1Gate. **LoRaWan gateway**. http://www.1-gate.com/. Accessed: 2019-02-25.

[Aderohunmu et al., 2013] Aderohunmu, F. A., Paci, G., Brunelli, D., Deng, J. D., Benini, L., et Purvis, M. (2013). **An application-specific forecasting algorithm for extending wsn lifetime**. In *2013 IEEE International Conference on Distributed Computing in Sensor Systems*, pages 374–381.

[Al-Hoqani et al., 2015] Al-Hoqani, N., et Yang, S.-H. (2015). **Adaptive sampling for wireless household water consumption monitoring**. *Procedia Engineering*, 119:1356–1365.

[Al-Karaki et al., 2004] Al-Karaki, J. N., Ul-Mustafa, R., et Kamal, A. E. (2004). **Data aggregation in wireless sensor networks-exact and approximate algorithms**. In *2004 Workshop on High Performance Switching and Routing, 2004. HPSR.*, pages 241–245. IEEE.

[Alippi et al., 2009] Alippi, C., Anastasi, G., Di Francesco, M., et Roveri, M. (2009). **An adaptive sampling algorithm for effective energy management in wireless sensor networks with energy-hungry sensors**. *IEEE Transactions on Instrumentation and Measurement*, 59(2):335–344.

[Alippi et al., 2011] Alippi, C., Camplani, R., Galperti, C., et Roveri, M. (2011). **A robust, adaptive, solar-powered wsn framework for aquatic environmental monitoring**. *IEEE Sensors Journal*, 11(1):45–55.

[Amaxilatis et al., 2018] Amaxilatis, D., et Chatzigiannakis, I. (2018). **Design and analysis of adaptive hierarchical low-power long-range networks**. *J. Sens. Actuator Netw.*, 7(4):51.

[Amiri et al., 2014] Amiri, E., Keshavarz, H., Alizadeh, M., Zamani, M., et Khodadadi, T. (2014). **Energy efficient routing in wireless sensor networks based on fuzzy ant colony optimization**. *International Journal of Distributed Sensor Networks*, 10(7):768936.

[Ayoub et al., 2017] Ayoub, N., Asad, M., Aslam, M., Gao, Z., Munir, E. U., et Tobji, R. (2017). **Mahee: Multi-hop advance heterogeneity-aware energy efficient path planning algorithm for wireless sensor networks**. In *2017 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM)*, pages 1–6. IEEE.

[Azar et al., 2018] Azar, J., Habib, C., Darazi, R., Makhoul, A., et Demerjian, J. (2018). **Using adaptive sampling and dwt lifting scheme for efficient data reduction in wireless body sensor networks**. In *2018 14th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pages 1–8.

[Bakaraniya et al., 2013] Bakaraniya, P., et Mehta, S. (2013). **K-leach: An improved leach protocol for lifetime improvement in wsn**. *Int. J. Eng. Trends Technol.*, 4(5):1521–1526.

[Basheer et al., 2017] Basheer, A., et Sha, K. (2017). **Cluster-based quality-aware adaptive data compression for streaming data**. *J. Data and Information Quality*, 9(1):2:1–2:33.

[Bdiri et al., 2015] Bdiri, S., et Derbel, F. (2015). **An ultra-low power wake-up receiver for realtime constrained wireless sensor networks**. In *Proceedings of the AMA Conferences, Nurenberg, Germany*, pages 19–21.

[Bhuiyan et al., 2017] Bhuiyan, M. Z. A., Wu, J., Wang, G., Wang, T., et Hassan, M. M. (2017). **e-sampling: Event-sensitive autonomous adaptive sensing and low-cost monitoring in networked sensing systems**. *ACM Trans. Auton. Adapt. Syst.*, 12(1):1:1–1:29.

[Biernacki et al., 2003] Biernacki, C., et Chrétien, S. (2003). **Degeneracy in the maximum likelihood estimation of univariate gaussian mixtures with em**. *Stat. & Probab. Lett.*, 61(4):373–382.

[Borges et al., 2014] Borges, L. M., Velez, F. J., et Lebres, A. S. (2014). **Survey on the characterization and classification of wireless sensor network applications**. *IEEE Communications Surveys & Tutorials*, 16(4):1860–1890.

[Breiman, 2017] Breiman, L. (2017). **Classification and regression trees**. Routledge.

[Caliński et al., 1974] Caliński, T., et Harabasz, J. (1974). **A dendrite method for cluster analysis**. *Commun. Stat. -Theory Methods*, 3(1):1–27.

[Çam et al., 2006] Çam, H., Özdemir, S., Nair, P., Muthuavinashiappan, D., et Sanli, H. O. (2006). **Energy-efficient secure pattern based data aggregation for wireless sensor networks**. *Computer Communications*, 29(4):446–455.

[Cantuña et al., 2017] Cantuña, J. G., Bastidas, D., Solórzano, S., et Clairand, J.-M. (2017). **Design and implementation of a wireless sensor network to detect forest fires**. In *2017 Fourth international conference on eDemocracy & eGovernment (ICEDEG)*, pages 15–21. IEEE.

[Celeux et al., 2001] Celeux, G., Chrétien, S., Forbes, F., et Mkhadri, A. (2001). **A component-wise em algorithm for mixtures**. *J. Comput. Graph. Stat.*, 10(4):697–712.

[Cerpa et al., 2001] Cerpa, A., Elson, J., Estrin, D., Girod, L., Hamilton, M., et Zhao, J. (2001). **Habitat monitoring: Application driver for wireless communications technology**. *ACM SIGCOMM Computer Communication Review*, 31(2 supplement):20–41.

[Chao et al., 2014] Chao, C.-M., et Hsiao, T.-Y. (2014). **Design of structure-free and energy-balanced data aggregation in wireless sensor networks**. *Journal of Network and Computer Applications*, 37:229–239.

[Chen et al., 2007] Chen, S., Lee, H., Chen, C., Lin, C., et Luo, C. (2007). **A wireless body sensor network system for healthcare monitoring application**. In *2007 IEEE Biomedical Circuits and Systems Conference*, pages 243–246.

[Chiwewe et al., 2011] Chiwewe, T. M., et Hancke, G. P. (2011). **A distributed topology control technique for low interference and energy efficiency in wireless sensor networks**. *IEEE Transactions on Industrial Informatics*, 8(1):11–19.

[Chottirapong et al., 2015] Chottirapong, K., Manatrinon, S., Dangsakul, P., et Kwankeow, N. (2015). **Design of energy harvesting thermoelectric generator with wireless sensors in organic fertilizer plant**. In *2015 6th International Conference of Information and Communication Technology for Embedded Systems (IC-ICTES)*, pages 1–6. IEEE.

[Chrétien et al., 2012] Chrétien, S., Hero, A., et Perdry, H. (2012). **Space alternating penalized kullback proximal point algorithms for maximizing likelihood with nondifferentiable penalty**. *Ann. Inst. Stat. Math.*, 64(4):791–809.

[Chrétien et al., 2000] Chrétien, S., et Hero, A. O. (2000). **Kullback proximal algorithms for maximum-likelihood estimation**. *IEEE Trans. Inf. Theory*, 46(5):1800–1810.

[Chrétien et al., 2008] Chrétien, S., et Hero, A. O. (2008). **On em algorithms and their proximal generalizations**. *ESAIM: Probab. Stat.*, 12:308–326.

[Chretien et al., tted] Chretien, S., Jagan, K., et Barton, E. (submitted). **Clustering on low-dimensional latent manifolds via laplacianeigenmaps when clusters overlap**. *Meas. Sci. Technol.*

[Comaniciu et al., 2002] Comaniciu, D., et Meer, P. (2002). **Mean shift: A robust approach toward feature space analysis**. *IEEE Trans. Pattern Anal. & Mach. Intell.*, (5):603–619.

[Davidson et al., 2009] Davidson, J., Collins, M., et Behrens, S. (2009). **Thermal energy harvesting between the air/water interface for powering wireless sensor nodes**. In *Active and Passive Smart Structures and Integrated Systems 2009*, volume 7288, page 728814. International Society for Optics and Photonics.

[Davies et al., 1979] Davies, D. L., et Bouldin, D. W. (1979). **A cluster separation measure**. *IEEE Trans. Pattern Anal. Mach. Intell.*, (2):224–227.

[Davis et al., 2012] Davis, A., et Chang, H. (2012). **A survey of wireless sensor network architectures**. *International Journal of Computer Science and Engineering Survey*, 3(6):1.

[Deniz et al., 2016] Deniz, F., Bagci, H., Korpeoglu, I., et Yazıcı, A. (2016). **An adaptive, energy-aware and distributed fault-tolerant topology-control algorithm for heterogeneous wireless sensor networks**. *Ad Hoc Networks*, 44:104–117.

[Dhimal et al., 2015] Dhimal, S., et Sharma, K. (2015). **Energy conservation in wireless sensor networks by exploiting inter-node data similarity metrics**. *International Journal of Energy, Information and Communications*, 6(2):23–32.

[Donghua Pan et al., 2011] Donghua Pan, et Lilei Zhao (2011). **Uncertain data cluster based on dbscan**. In *2011 International Conference on Multimedia Technology*, pages 3781–3784.

[Doost et al., 2010] Doost, R., Chowdhury, K. R., et Di Felice, M. (2010). **Routing and link layer protocol design for sensor networks with wireless energy transfer**. In *2010 IEEE Global Telecommunications Conference GLOBECOM 2010*, pages 1–5. IEEE.

[Doreswamy et al., 2014] Doreswamy, et Narasegouda, S. (2014). **Fault detection in sensor network using dbscan and statistical models**. In Satapathy, S. C., Udgata, S. K., et Biswal, B. N., editors, *Proceedings of the International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA) 2013*, pages 443–450, Cham. Springer International Publishing.

[Du et al., 2015] Du, T., Qu, Z., Guo, Q., et Qu, S. (2015). **A high efficient and real time data aggregation scheme for wsns**. *International Journal of Distributed Sensor Networks*, 11(6):261381.

[Dunn, 1974] Dunn, J. C. (1974). **Well-separated clusters and optimal fuzzy partitions**. *J. Cybern.*, 4(1):95–104.

[Dutta et al., 2008] Dutta, P., et Culler, D. (2008). **Practical asynchronous neighbor discovery and rendezvous for mobile sensing applications**. In *Proceedings of the 6th ACM conference on Embedded network sensor systems*, pages 71–84. ACM.

[ElGammal et al., 2009] ElGammal, M., et Eltoweissy, M. (2009). **Location-aware affinity propagation clustering in wireless sensor networks**. In *2009 IEEE International Conference on Wireless and Mobile Computing, Networking and Communications*, pages 471–475.

[Ester et al., 1996] Ester, M., Kriegel, H.-P., Sander, J., Xu, X., et others (1996). **A density-based algorithm for discovering clusters in large spatial databases with noise.** In *Kdd*, volume 96, pages 226–231.

[Estrin et al., 1999] Estrin, D., Govindan, R., Heidemann, J., et Kumar, S. (1999). **Next century challenges: Scalable coordination in sensor networks**. In *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pages 263–270.

[Farhat et al., 2017] Farhat, A., Guyeux, C., Makhoul, A., Jaber, A., Tawil, R., et Hijazi, A. (2017). **Impacts of wireless sensor networks strategies and topologies on prognostics and health management**. *J. Intell. Manuf.*, *:1–27.

[Fathy et al., 2019] Fathy, Y., et Barnaghi, P. (2019). **Quality-based and energy-efficient data communication for the internet of things networks**. *IEEE Internet of Things Journal*.

[Fayçal-Khelfi et al., 2016] Fayçal-Khelfi, M., et others (2016). **Using mobile data collectors to enhance energy efficiency a nd reliability in delay tolerant wireless sensor networks.** *Journal of Information Processing Systems*, 12(2).

[Fowlkes et al., 1983] Fowlkes, E. B., et Mallows, C. L. (1983). **A method for comparing two hierarchical clusterings**. *J. Am. Stat. Assoc.*, 78(383):553–569.

[Frey et al., 2007] Frey, B. J., et Dueck, D. (2007). **Clustering by passing messages between data points**. *science*, 315(5814):972–976.

[Gao et al., 2016] Gao, Z., Cheng, W., Qiu, X., et Meng, L. (2016). **A missing sensor data estimation algorithm based on temporal and spatial correlation**. *Int. J. Distrib. Sen. Netw.*, pages 178:178–178:178.

[Gotthardp,] Gotthardp. **The network server from**. https://github.com/gotthardp/lorawan-server/tree/master/doc/. Accessed: 2019-02-25.

[Gruenwald et al., 2010] Gruenwald, L., Yang, H., Sadik, M. S., et Shukla, R. (2010). **Using data mining to handle missing data in multi-hop sensor network applications**. *Proceedings of the Ninth ACM International Workshop on Data Engineering for Wireless and Mobile Access*, pages 9–16.

[Guestrin et al., 2004] Guestrin, C., Bodik, P., Thibaux, R., Paskin, M., et Madden, S. (2004). **Distributed regression: an efficient framework for modeling sensor network data**. In *Third International Symposium on Information Processing in Sensor Networks.*, pages 1–10.

[Guo et al., 2013] Guo, S., Wang, C., et Yang, Y. (2013). **Mobile data gathering with wireless energy replenishment in rechargeable sensor networks**. In *2013 Proceedings IEEE INFOCOM*, pages 1932–1940. IEEE.

[Gupta et al., 2011] Gupta, M., Shum, L. V., Bodanese, E., et Hailes, S. (2011). **Design and evaluation of an adaptive sampling strategy for a wireless air pollution sensor network**. In *Local Computer Networks (LCN), 2011 IEEE 36th Conference on*, pages 1003–1010. IEEE.

[Halgamuge et al., 2009] Halgamuge, M. N., Zukerman, M., Ramamohanarao, K., et Vu, H. L. (2009). **An estimation of sensor energy consumption**. *Progress in Electromagnetics Research*, 12:259–295.

[Hao et al., 2008] Hao, Y., et Foster, R. (2008). **Wireless body sensor networks for health-monitoring applications**. *Physiological measurement*, 29(11):R27.

[Harb et al., 2018] Harb, H., et Makhoul, A. (2018). **Energy-efficient sensor data collection approach for industrial process monitoring**. *IEEE Trans. Industrial Informatics*, 14(2):661–672.

[Hariri et al., 2018] Hariri, S., Carrasco Kind, M., et Brunner, R. J. (2018). **Extended Isolation Forest**. *ArXiv e-prints*.

[Hou et al., 2018] Hou, L., Tan, S., Zhang, Z., et Bergmann, N. W. (2018). **Thermal energy harvesting wsns node for temperature monitoring in iiot**. *IEEE Access*, 6:35243–35249.

[Hu et al., 2015] Hu, H., Wang, X., Yang, Z., et Zheng, B. (2015). **A spectral clustering approach to identifying cuts in wireless sensor networks**. *Ieee Sens. J.*, 15(3):1838–1848.

[Humidi et al., 2019] Humidi, N. A., Chowdhary, V., et others (2019). **Lightweight data transmission scheme based on data aggregation technique in wireless sensor networks**. *Girish, Lightweight Data Transmission Scheme Based on Data Aggregation Technique in Wireless Sensor Networks (May 17, 2019)*.

[Kahn et al., 1999] Kahn, J. M., Katz, R. H., et Pister, K. S. (1999). **Next century challenges: mobile networking for "smart dust"**. In *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pages 271–278.

[Kahrobaee et al., 2013] Kahrobaee, S., et Vuran, M. C. (2013). **Vibration energy harvesting for wireless underground sensor networks**. In *2013 IEEE International Conference on Communications (ICC)*, pages 1543–1548. IEEE.

[Kanwal et al., 2017] Kanwal, K., Liaquat, A., Mughal, M., Abbasi, A. R., et Aamir, M. (2017). **Towards development of a low cost early fire detection system using wireless sensor network and machine vision**. *Wireless Personal Communications*, 95(2):475–489.

[Karuppasamy et al., 2013] Karuppasamy, K., et Gunaraj, V. (2013). **Optimizing sensing quality with coverage and lifetime in wireless sensor networks**. *International Journal of Engineering Research and Technology*, 2(2):1–7.

[Kaushik et al., 2016] Kaushik, K., Mishra, D., De, S., Chowdhury, K. R., et Heinzelman, W. (2016). **Low-cost wake-up receiver for rf energy harvesting wireless sensor networks**. *IEEE Sensors Journal*, 16(16):6270–6278.

[Khamukhin et al., 2016] Khamukhin, A. A., et Bertoldo, S. (2016). **Spectral analysis of forest fire noise for early detection using wireless sensor networks**. In *2016 International Siberian Conference on Control and Communications (SIBCON)*, pages 1–4. IEEE.

[Khorasani et al., 2017] Khorasani, F., et Naji, H. R. (2017). **Energy efficient data aggregation in wireless sensor networks using neural networks**. *International Journal of Sensor Networks*, 24(1):26–42.

[Kim et al., 2003] Kim, H. S., Abdelzaher, T. F., et Kwon, W. H. (2003). **Minimum-energy asynchronous dissemination to mobile sinks in wireless sensor networks**. In *Proceedings of the 1st international conference on Embedded networked sensor systems*, pages 193–204. ACM.

[Klann et al., 2007] Klann, M., Riedel, T., Gellersen, H., Fischer, C., Oppenheim, M., Lukowicz, P., Pirkl, G., Kunze, K., Beuster, M., Beigl, M., et others (2007). **Lifenet: an ad-hoc sensor network and wearable system to provide firefighters with navigation support**. *Adjunct Proc. Ubicomp 2007*, pages 124–127.

[Kosanovic et al., 2013] Kosanovic, M., et Stojcev, M. (2013). **Energy efficient rendezvous protocol for wireless sensor networks**. In *2013 2nd Mediterranean Conference on Embedded Computing (MECO)*, pages 215–218. IEEE.

[Kung et al., 2009] Kung, H. T., et Vlah, D. (2009). **A spectral clustering approach to validating sensors via their peers in distributed sensor networks**. In *2009 Proceedings of 18th International Conference on Computer Communications and Networks*, pages 1–7.

[Lewis-Beck et al., 2015] Lewis-Beck, C., et Lewis-Beck, M. (2015). **Applied regression: An introduction**, volume 22. Sage publications.

[Li et al., 2013] Li, G., et Wang, Y. (2013). **Automatic arima modeling-based data aggregation scheme in wireless sensor networks**. *EURASIP Journal on Wireless Communications and Networking*, (1):85.

[Li et al., 2009] Li, J., McCann, J., Pollard, N., et Faloutsos, C. (2009). **Dynammo: Mining and summarization of coevolving sequences with missing values**. *ACM SIGKDD, June/July 2009, pp 527–534*, (CMU-RI-TR-).

[LI et al., 2018] LI, L., LI, D., et LI, D. (2018). **An efficient routing algorithm based on k-means++ clustering and fuzzy for wireless sensor network***. In *2018 13th World Congress on Intelligent Control and Automation (WCICA)*, pages 716–720.

[Li et al., 2015] Li, Y., et Shi, R. (2015). **An intelligent solar energy-harvesting system for wireless sensor networks**. *EURASIP Journal on Wireless Communications and Networking*, 2015(1):179.

[Li, 2018] Li, Z. (2018). **The optimization of solar energy harvesting in wsn**. Master's thesis, Mid Sweden University, Department of Electronics Design.

[Liang et al., 2007] Liang, S., Tang, Y., et Zhu, Q. (2007). **Passive wake-up scheme for wireless sensor networks**. In *Second International Conference on Innovative Computing, Informatio and Control (ICICIC 2007)*, pages 507–507. IEEE.

[Liang et al., 2010] Liang, W., Luo, J., et Xu, X. (2010). **Prolonging network lifetime via a controlled mobile sink in wireless sensor networks**. In *2010 IEEE global telecommunications conference GLOBECOM 2010*, pages 1–6. IEEE.

[Liaw et al., 2002] Liaw, A., et Wiener, M. (2002). **Classification and regression by randomforest**. *R news*, 2(3):18–22.

[Lin et al., 2004] Lin, E.-Y., Rabaey, J. M., et Wolisz, A. (2004). **Power-efficient rendezvous schemes for dense wireless sensor networks**. In *2004 IEEE international conference on communications (IEEE Cat. No. 04CH37577)*, volume 7, pages 3769–3776. IEEE.

[Liu et al., 2008] Liu, F. T., Ting, K. M., et Zhou, Z.-H. (2008). **Isolation forest**. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, ICDM '08, pages 413–422, Washington, DC, USA. IEEE Computer Society.

[Liu et al., 2005] Liu, Y., Elhanany, I., et Qi, H. (2005). **An energy-efficient qos-aware media access control protocol for wireless sensor networks**. In *IEEE International Conference on Mobile Adhoc and Sensor Systems Conference, 2005.*, pages 3–pp. IEEE.

[Louail et al., 2016] Louail, L., et Felea, V. (2016). **Routing-aware tdma scheduling for wireless sensor networks**. In *2016 12th annual conference on Wireless On-demand Network Systems and Services (WONS)*, pages 1–8. IEEE.

[Maaten et al., 2008] Maaten, L. v. d., et Hinton, G. (2008). **Visualizing data using t-sne**. *J. Mach. Learn. Res.*, 9(Nov):2579–2605.

[Mahamuni, 2016] Mahamuni, C. V. (2016). **A military surveillance system based on wireless sensor networks with extended coverage life**. In *2016 International Conference on Global Trends in Signal Processing, Information Computing and Communication (ICGTSPICC)*, pages 375–381.

[Makhoul et al., 2015] Makhoul, A., Harb, H., et Laiymani, D. (2015). **Residual energy-based adaptive data collection approach for periodic sensor networks**. *Ad Hoc Networks*, 35:149–160.

[Marcelloni et al., 2008] Marcelloni, F., et Vecchio, M. (2008). **A simple algorithm for data compression in wireless sensor networks**. *IEEE communications letters*, 12(6):411–413.

[Mascarenas, 2008] Mascarenas, D. D. (2008). **" Mobile host" wireless sensor networks: a new sensor network paradigm for structural health monitoring applications**. PhD thesis, UC San Diego.

[Masoum et al., 2013] Masoum, A., Meratnia, N., et Havinga, P. J. (2013). **An energy-efficient adaptive sampling scheme for wireless sensor networks**. In *2013 IEEE Eighth International Conference on Intelligent Sensors, Sensor Networks and Information Processing*, pages 231–236. IEEE.

[McInnes et al., 2017] McInnes, L., Healy, J., et Astels, S. (2017). **hdbscan: Hierarchical density based clustering**. *J. Open Source Softw.*, 2(11).

[McKight et al., 2010] McKight, P. E., et Najab, J. (2010). **Kruskal-wallis test**. *Corsini Encyclopedia of Psychology*.

[McLachlan et al., 2007] McLachlan, G., et Krishnan, T. (2007). **The EM algorithm and extensions**, volume 382. John Wiley & Sons.

[McLachlan et al., 2019] McLachlan, G. J., Lee, S. X., et Rathnayake, S. I. (2019). **Finite mixture models**. *Annu. Rev. Stat. Its Appl.*, 6:355–378.

[Medeiros et al., 2014] Medeiros, H. P., Maciel, M. C., Demo Souza, R., et Pellenz, M. E. (2014). **Lightweight data compression in wireless sensor networks using huffman coding**. *International Journal of Distributed Sensor Networks*, 10(1):672921.

[Meilă, 2007] Meilă, M. (2007). **Comparing clusterings—an information based distance**. *J. Multivar. Anal.*, 98(5):873–895.

[Mittal et al., 2010] Mittal, R., et Bhatia, M. P. S. (2010). **Wireless sensor networks for monitoring the environmental activities**. pages 1–5.

[Mochaourab et al., 2008] Mochaourab, R., et Dargie, W. (2008). **A fair and energy-efficient topology control protocol for wireless sensor networks**. In *Proceedings of the 2nd ACM international conference on Context-awareness for self-managing systems*, pages 6–15. ACM.

[Moghadam et al., 2011] Moghadam, R. A., et Keshmirpour, M. (2011). **Hybrid arima and neural network model for measurement estimation in energy-efficient wireless sensor networks**. In *International Conference on Informatics Engineering and Information Science*, pages 35–48. Springer.

[Monteiro et al., 2017] Monteiro, L. C., Delicato, F. C., Pirmez, L., Pires, P. F., et Miceli, C. (2017). **Dpcas: Data prediction with cubic adaptive sampling for wireless sensor networks**. In Au, M. H. A., Castiglione, A., Choo, K.-K. R., Palmieri, F., et Li, K.-C., editors, *Green, Pervasive, and Cloud Computing*, pages 353–368, Cham. Springer International Publishing.

[Montgomery et al., 2012] Montgomery, D. C., Peck, E. A., et Vining, G. G. (2012). **Introduction to linear regression analysis**, volume 821. John Wiley & Sons.

[Muniraju et al., 2017] Muniraju, G., Zhang, S., Tepedelenlioglu, C., K. Banavar, M., Spanias, A., Vargas-Rosales, C., et Villalpando-Hernandez, R. (2017). **Location based distributed spectral clustering for wireless sensor networks**. In *2017 Sensor Signal Processing for Defence Conference (SSPD)*, pages 1–5.

[Muruganathan et al., 2005] Muruganathan, S. D., Ma, D. C., Bhasin, R. I., et Fapojuwo, A. O. (2005). **A centralized energy-efficient routing protocol for wireless sensor networks**. *IEEE Communications Magazine*, 43(3):S8–13.

[Network,] Network, T. T. **List of lorawan gateways**. https://www.thethingsnetwork.org/docs/gateways/start/list.html. Accessed: 2019-02-25.

[Nguyen et al., 2015] Nguyen, L. V., Kodagoda, S., Ranasinghe, R., et Dissanayake, G. (2015). **Information-driven adaptive sampling strategy for mobile robotic wireless sensor network**. *IEEE Transactions on Control Systems Technology*, 24(1):372–379.

[Nishimoto et al., 2010] Nishimoto, H., Kawahara, Y., et Asami, T. (2010). **Prototype implementation of ambient rf energy harvesting wireless sensor networks**. In *SENSORS, 2010 IEEE*, pages 1282–1287.

[Pan et al., 2013] Pan, L., Gao, H., Li, J., Gao, H., et Guo, X. (2013). **Ciam: An adaptive 2-in-1 missing data estimation algorithm in wireless sensor networks**. In *19th IEEE International Conference on Networks (ICON)*, pages 1–6.

[Park et al., 2009] Park, H.-S., et Jun, C.-H. (2009). **A simple and fast algorithm for k-medoids clustering**. *Expert Syst. Appl.*, 36(2):3336–3341.

[Pavithra et al., 2015] Pavithra, M., et Ghuli, P. (2015). **A novel approach for reducing energy consumption using k-medoids in clustering based wsn**. *Int. J. Sci. Res. (IJSR)*, 4(6).

[Pedregosa et al., 2011a] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et others (2011a). **Scikit-learn: Machine learning in python**. *Journal of machine learning research*, 12(Oct):2825–2830.

[Pedregosa et al., 2011b] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., et Duchesnay, E. (2011b). **Scikit-learn: Machine learning in Python**. *J. Mach. Learn. Res.*, 12:2825–2830.

[Petriu et al., 2000] Petriu, E. M., Georganas, N. D., Petriu, D. C., Makrakis, D., et Groza, V. Z. (2000). **Sensor-based information appliances**. *IEEE Instrumentation & Measurement Magazine*, 3(4):31–35.

[Qureshi et al., 2011] Qureshi, H. K., Rizvi, S., Saleem, M., Khayam, S. A., Rakocevic, V., et Rajarajan, M. (2011). **Poly: A reliable and energy efficient topology control protocol for wireless sensor networks**. *Computer Communications*, 34(10):1235–1242.

[Rajasegarar et al., 2006] Rajasegarar, S., Leckie, C., Palaniswami, M., et Bezdek, J. C. (2006). **Distributed anomaly detection in wireless sensor networks**. pages 1–5.

[Ramirez et al., 2012] Ramirez, L., Dyrks, T., Gerwinski, J., Betz, M., Scholz, M., et Wulf, V. (2012). **Landmarke: an ad hoc deployable ubicomp infrastructure to support indoor navigation of firefighters**. *Personal and Ubiquitous Computing*, 16(8):1025–1038.

[Rao et al., 2018] Rao, Y., Deng, C., Zhao, G., Qiao, Y., Fu, L.-y., Shao, X., et Wang, R.-c. (2018). **Self-adaptive implicit contention window adjustment mechanism for qos optimization in wireless sensor networks**. *Journal of Network and Computer Applications*, 109:36–52.

[Raza et al., 2015] Raza, U., Camerra, A., Murphy, A. L., Palpanas, T., et Picco, G. P. (2015). **Practical data prediction for real-world wireless sensor networks**. *IEEE Transactions on Knowledge and Data Engineering*, 27(8):2231–2244.

[Rhee et al., 2008] Rhee, I., Warrier, A., Aia, M., Min, J., et Sichitiu, M. L. (2008). **Z-mac: a hybrid mac for wireless sensor networks**. *IEEE/ACM Transactions on Networking (TON)*, 16(3):511–524.

[Rosenberg et al., 2007] Rosenberg, A., et Hirschberg, J. (2007). **V-measure: A conditional entropy-based external cluster evaluation measure**. In *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*.

[Rousseeuw, 1987] Rousseeuw, P. J. (1987). **Silhouettes: A graphical aid to the interpretation and validation of cluster analysis**. *J. Comput. Appl. Math.*, 20:53–65.

[Saeedi Emadi et al., 2018] Saeedi Emadi, H., et Mazinani, S. M. (2018). **A novel anomaly detection algorithm using dbscan and svm in wireless sensor networks**. *Wirel. Pers. Commun.*, (2):2025–2035.

[Sakthidasan et al., 2018] Sakthidasan, K., Vasudevan, N., Diderot, P. K. G., et Kadhiravan, C. (2018). **Woapr: An affinity propagation based clustering and optimal path selection for time-critical wireless sensor networks**. *IET Networks*, 8(2):100–106.

[Salajegheh et al., 2007] Salajegheh, M., Soroush, H., et Kalis, A. (2007). **Hymac: Hybrid tdma/fdma medium access control protocol for wireless sensor networks**. In *2007 IEEE 18th International Symposium on Personal, Indoor and Mobile Radio Communications*, pages 1–5. IEEE.

[Samanpour et al., 2018] Samanpour, A. R., Ruegenberg, A., et Ahlers, R. (2018). **The future of machine learning and predictive analytics**. In *Digital Marketplaces Unleashed*, pages 297–309. Springer.

[Santini et al., 2006] Santini, S., et Römer, K. (2006). **An adaptive strategy for quality-based data reduction in wireless sensor networks**. In *Proceedings of the 3rd International Conference on Networked Sensing Systems*, pages 29–36.

[Sensorscope, 2007] Sensorscope (2007). https://lcav.epfl.ch/page-145180-en.html. Online; accessed 13 Septembre 2018.

[Sha et al., 2006] Sha, K., Shi, W., et Watkins, O. (2006). **Using wireless sensor networks for fire rescue applications: Requirements and challenges**. In *2006 IEEE International Conference on Electro/Information Technology*, pages 239–244.

[Shahriyar et al., 2009] Shahriyar, R., Bari, M. F., Kundu, G., Ahamed, S. I., et Akbar, M. M. (2009). **Intelligent mobile health monitoring system (imhms)**. In *International Conference on Electronic Healthcare*, pages 5–12. Springer.

[Sheu et al., 2005] Sheu, J.-P., Cheng, P.-W., et Hsieh, K.-Y. (2005). **Design and implementation of a smart mobile robot**. In *WiMob'2005), IEEE International Conference on Wireless And Mobile Computing, Networking And Communications, 2005.*, volume 3, pages 422–429. IEEE.

[Shih et al., 2001] Shih, E., Cho, S.-H., Ickes, N., Min, R., Sinha, A., Wang, A., et Chandrakasan, A. (2001). **Physical layer driven protocol and algorithm design for energy-efficient wireless sensor networks**. In *Proceedings of the 7th annual international conference on Mobile computing and networking*, pages 272–287.

[Shumway et al., 1982] Shumway, R. H., et Stoffer, D. S. (1982). **An approach to time series smoothing and forecasting using the em algorithm**. *Journal of Time Series Analysis*, 3(4):253–264.

[Smola et al., 2004] Smola, A. J., et Schölkopf, B. (2004). **A tutorial on support vector regression**. *Statistics and computing*, 14(3):199–222.

[Sohn et al., 2016] Sohn, I., Lee, J., et Lee, S. H. (2016). **Low-energy adaptive clustering hierarchy using affinity propagation for wireless sensor networks**. *Ieee Commun. Lett.*, 20(3):558–561.

[Song et al., 2009] Song, W.-Z., Huang, R., Shirazi, B., et LaHusen, R. (2009). **Treemac: Localized tdma mac protocol for real-time high-data-rate sensor networks**. *Pervasive and Mobile Computing*, 5(6):750–765.

[Summerer et al., 2009] Summerer, L., et Purcell, O. (2009). **Concepts for wireless energy transmission via laser**. *Europeans Space Agency (ESA)-Advanced Concepts Team*.

[Tan et al., 2008] Tan, L., Gong, Y., et Chen, G. (2008). **A balanced parallel clustering protocol for wireless sensor networks using k-means techniques**. pages 300–305.

[Tan et al., 2016] Tan, L., et Wu, M. (2016). **Data reduction in wireless sensor networks: A hierarchical lms prediction approach**. *IEEE Sensors Journal*, 16(6):1708–1715.

[Tan et al., 2010] Tan, Y. K., et Panda, S. K. (2010). **Energy harvesting from hybrid indoor ambient light and thermal energy sources for enhanced performance of wireless sensor nodes**. *IEEE Transactions on Industrial Electronics*, 58(9):4424–4435.

[Thorndike, 1953] Thorndike, R. L. (1953). **Who belongs in the family?** *Psychometrika*, 18(4):267–276.

[Tibshirani et al., 2015] Tibshirani, R., Wainwright, M., et Hastie, T. (2015). **Statistical learning with sparsity: the lasso and generalizations**. Chapman and Hall/CRC.

[Tong et al., 2011] Tong, B., Wang, G., Zhang, W., et Wang, C. (2011). **Node reclamation and replacement for long-lived sensor networks**. *IEEE Transactions on Parallel and Distributed Systems*, 22(9):1550–1563.

[Umair et al., 2018] Umair, S., et Sharif, M. M. (2018). **Predicting students grades using artificial neural networks and support vector machine**. In *Encyclopedia of Information Science and Technology, Fourth Edition*, pages 5169–5182. IGI Global.

[Uthayakumar et al., 2019] Uthayakumar, J., Vengattaraman, T., et Dhavachelvan, P. (2019). **A new lossless neighborhood indexing sequence (nis) algorithm for data compression in wireless sensor networks**. *Ad Hoc Networks*, 83:149–157.

[Varghese et al., 2016] Varghese, B., John, N. E., Sreelal, S., et Gopal, K. (2016). **Design and development of an rf energy harvesting wireless sensor node (eh-wsn) for aerospace applications**. *Procedia Computer Science*, 93:230 – 237. Proceedings of the 6th International Conference on Advances in Computing and Communications.

[Villas et al., 2013] Villas, L. A., Boukerche, A., Guidoni, D. L., De Oliveira, H. A., De Araujo, R. B., et Loureiro, A. A. (2013). **An energy-aware spatio-temporal correlation mechanism to perform efficient data collection in wireless sensor networks**. *Computer Communications*, 36(9):1054–1066.

[Vinh et al., 2010] Vinh, N. X., Epps, J., et Bailey, J. (2010). **Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance**. *J. Mach. Learn. Res.*, 11(Oct):2837–2854.

[Von Luxburg, 2007] Von Luxburg, U. (2007). **A tutorial on spectral clustering**. *Stat. Comput.*, 17(4):395–416.

[Wagstaff et al., 2001] Wagstaff, K., Cardie, C., Rogers, S., Schrödl, S., et others (2001). **Constrained k-means clustering with background knowledge**. In *Icml*, volume 1, pages 577–584.

[Wang et al., 2018] Wang, J., Cao, J., Sherratt, R. S., et Park, J. H. (2018). **An improved ant colony optimization-based approach with mobile sink for wireless sensor networks**. *The Journal of Supercomputing*, 74(12):6633–6645.

[Wang et al., 2019a] Wang, J., Gao, Y., Liu, W., Sangaiah, A. K., et Kim, H.-J. (2019a). **An improved routing schema with special clustering using pso algorithm for heterogeneous wireless sensor network**. *Sensors*, 19(3):671.

[Wang et al., 2019b] Wang, J., Gao, Y., Wang, K., Sangaiah, A. K., et Lim, S.-J. (2019b). **An affinity propagation-based self-adaptive clustering method for wireless sensor networks**. *Sensors*, 19(11):2579.

[Wang, 2005] Wang, L. (2005). **Support vector machines: theory and applications**, volume 177. Springer Science & Business Media.

[Wang et al., 2019c] Wang, S., Nguyen, T. L., et Shin, Y. (2019c). **Energy-efficient clustering algorithm for magnetic induction-based underwater wireless sensor networks**. *IEEE Access*, 7:5975–5983.

[Wang et al., 2012] Wang, S.-S., et Chen, Z.-P. (2012). **Lcm: a link-aware clustering mechanism for energy-efficient routing in wireless sensor networks**. *IEEE sensors journal*, 13(2):728–736.

[Watfa et al., 2011] Watfa, M. K., AlHassanieh, H., et Selman, S. (2011). **Multi-hop wireless energy transfer in wsns**. *IEEE communications letters*, 15(12):1275–1277.

[Wijesundara et al., 2016] Wijesundara, M., Tapparello, C., Gamage, A., Gokulan, Y., Gittelson, L., Howard, T., et Heinzelman, W. (2016). **Design of a kinetic energy harvester for elephant mounted wireless sensor nodes of jumbonet**. In *2016 IEEE Global Communications Conference (GLOBECOM)*, pages 1–7. IEEE.

[Willett et al., 2004] Willett, R., Martin, A., et Nowak, R. (2004). **Backcasting: adaptive sampling for sensor networks**. In *Proceedings of the 3rd international symposium on Information processing in sensor networks*, pages 124–133. ACM.

[Woias et al., 2014] Woias, P., Schule, F., Bäumke, E., Mehne, P., et Kroener, M. (2014). **Thermal energy harvesting from wildlife**. In *Journal of Physics: Conference Series*, volume 557, page 012084. IOP Publishing.

[Wu et al., 2017] Wu, H., Wang, J., Suo, M., et Mohapatra, P. (2017). **A holistic approach to reconstruct data in ocean sensor network using compression sensing**. *IEEE Access*, PP(99):1–1.

[Wu et al., 2018] Wu, H., Xian, J., Wang, J., Khandge, S., et Mohapatra, P. (2018). **Missing data recovery using reconstruction in ocean wireless sensor networks**. *Computer Communications*, 132:1–9.

[Wu et al., 2016] Wu, M., Tan, L., et Xiong, N. (2016). **Data prediction, compression, and recovery in clustered wireless sensor networks for environmental monitoring applications**. *Information Sciences*, 329(Supplement C):800 – 818.

[Xiangning et al., 2007] Xiangning, F., et Yulin, S. (2007). **Improvement on leach protocol of wireless sensor network**. In *2007 International Conference on Sensor Technologies and Applications (SENSORCOMM 2007)*, pages 260–264. IEEE.

[Xie et al., 2014] Xie, Q. Y., et Cheng, Y. (2014). **K-centers mean-shift reverse mean-shift clustering algorithm over heterogeneous wireless sensor networks**. In *2014 Wireless Telecommunications Symposium*, pages 1–6.

[Xu et al., 2003] Xu, Y., Bien, S., Mori, Y., Heidemann, J., et Estrin, D. (2003). **Topology control protocols to conserve energy in wireless ad hoc networks**.

[Yang et al., 2016] Yang, J., Tilak, S., et Rosing, T. S. (2016). **An interactive context-aware power management technique for optimizing sensor network lifetime**. In *SENSORNETS*, pages 69–76.

[Yang et al., 2018] Yang, L., Liang, Y., Wu, D., et Gault, J. (2018). **Train and equip firefighters with cognitive virtual and augmented reality**. In *2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC)*, pages 453–459. IEEE.

[Yang et al., 2017] Yang, X., Yan, Y., et Deng, D. (2017). **Research on clustering routing algorithm based on k-means++ for wsn**. In *2017 6th International Conference on Computer Science and Network Technology (ICCSNT)*, pages 330–333.

[Yaseen et al., 2018] Yaseen, Z. M., Deo, R. C., Hilal, A., Abd, A. M., Bueno, L. C., Salcedo-Sanz, S., et Nehdi, M. L. (2018). **Predicting compressive strength of lightweight foamed concrete using extreme learning machine model**. *Advances in Engineering Software*, 115:112–125.

[Ye et al., 2011] Ye, G., et Soga, K. (2011). **Energy harvesting from water distribution systems**. *Journal of Energy Engineering*, 138(1):7–17.

[Yu et al., 2008] Yu, H., Li, Y., Shang, Y., et Su, B. (2008). **Design and investigation of photovoltaic and thermoelectric hybrid power source for wireless sensor networks**. In *2008 3rd IEEE International Conference on Nano/Micro Engineered and Molecular Systems*, pages 196–201. IEEE.

[Zeng et al., 2016] Zeng, B., et Dong, Y. (2016). **An improved harmony search based energy-efficient routing algorithm for wireless sensor networks**. *Applied Soft Computing*, 41:135–147.

[Zhang et al., 1996] Zhang, T., Ramakrishnan, R., et Livny, M. (1996). **Birch: An efficient data clustering method for very large databases**. In *ACM Sigmod Record*, volume 25, pages 103–114. ACM.

[Zhang et al., 2018] Zhang, T., Zhao, Q., Shin, K., et Nakamoto, Y. (2018). **Bayesian-optimization-based peak searching algorithm for clustering in wireless sensor networks**. *J. Sens. Actuator Netw.*, 7(1):2.

[Zhang et al., 2019] Zhang, W., Wang, J., Han, G., Zhang, X., et Feng, Y. (2019). **A cluster sleep-wake scheduling algorithm based on 3d topology control in underwater sensor networks**. *Sensors*, 19(1):156.

[Zhao et al., 2003] Zhao, J., et Govindan, R. (2003). **Understanding packet delivery performance in dense wireless sensor networks**. In *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*, SenSys '03, pages 1–13, New York, NY, USA. ACM.

[Zhao et al., 2004] Zhao, W., Ammar, M., et Zegura, E. (2004). **A message ferrying approach for data delivery in sparse mobile ad hoc networks**. In *Proceedings of the 5th ACM international symposium on Mobile ad hoc networking and computing*, pages 187–198. ACM.

[Zheng et al., 2003] Zheng, R., Hou, J. C., et Sha, L. (2003). **Asynchronous wakeup for ad hoc networks**. In *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*, pages 35–45. ACM.

[Zhou et al., 2009] Zhou, Q., Li, X., et Xu, Y. (2009). **Mean shift based collaborative localization with dynamically clustering for wireless sensor networks**. In *2009 WRI International Conference on Communications and Mobile Computing*, volume 2, pages 66–70.

[Zhu et al., 2012] Zhu, Q., Guan, M., et He, Y. (2012). **Vibration energy harvesting in automobiles to power wireless sensors**. In *2012 IEEE International Conference on Information and Automation*, pages 349–354. IEEE.

[Zong et al., 2017] Zong, C., Yang, X., Wang, B., et Liu, C. (2017). **Minimal explanations of missing values by chasing acquisitional data**. *World Wide Web*, 20(6):1333–1362.

[Zou et al., 2006] Zou, H., Hastie, T., et Tibshirani, R. (2006). **Sparse principal component analysis**. *J. Comput. Graph. Stat.*, 15(2):265–286.

List of Figures

List of Tables

List of Algorithms

Document generated with LaTeX and:
the LaTeX style for PhD Thesis created by S. Galland — http://www.multiagent.fr/ThesisStyle
the tex-upmethodology package suite — http://www.arakhne.org/tex-upmethodology/

école doctorale **sciences pour l'ingénieur et microtechniques**

Université Bourgogne Franche-Comté
32, avenue de l'Observatoire
25000 Besançon, France